UNIVERSIDAD COMPLUTENSE DE MADRID
FACULTAD DE CIENCIAS ECONÓMICAS Y EMPRESARIALES

TESIS DOCTORAL

Machine Learning Applications in Economics

Aplicaciones en Economía del Aprendizaje Automático

MEMORIA PARA OPTAR AL GRADO DE DOCTOR

PRESENTADA POR

Matthew James Smith

Director

Francisco Álvarez González

Madrid

© Matthew James Smith, 2022

UNIVERSIDAD COMPLUTENSE DE MADRID

FACULTAD DE CIENCIAS ECONÓMICAS Y EMPRESARIALES

TESIS DOCTORAL

Machine Learning Applications in Economics
Aplicaciones en Economía del Aprendizaje Automático

MEMORIA PARA OPTAR AL GRADO DE DOCTOR

PRESENTADA POR

Matthew James Smith

DIRECTOR

Francisco Álvarez González

Madrid

UNIVERSIDAD COMPLUTENSE DE MADRID

FACULTAD DE CIENCIAS ECONÓMICAS Y EMPRESARIALES

TESIS DOCTORAL

Machine Learning Applications in Economics
Aplicaciones en Economía del Aprendizaje Automático

MEMORIA PARA OPTAR AL GRADO DE DOCTOR

PRESENTADA POR

Matthew James Smith

DIRECTOR

Francisco Álvarez González

Madrid

Table of Contents

List of Figures

List of Tables

Resumen en Castellano

Este Tesis examina problemas en economía desde la perspectiva de Aprendizaje Mecánico. Se hace hincapié en la interpretabilidad de los algoritmos de Aprendizaje Mecánico en lugar de modelos de predicción de "black-box".

Capítulo 1 Proporciona el resumen de la terminología y los métodos de Aprendizaje Mecánico utilizados a lo largo de esta tesis. El objetivo de este capítulo es construir la trayectoria desde un simple árbol de decisión hasta algoritmos impulsados por conjuntos más avanzados. También se explican otros modelos de Machine Learning. Asimismo, se proporciona una discusión de los avances en el Aprendizaje Mecánico en economía junto con algunos de los escollos que enfrenta el aprendizaje automático. Además, un ejemplo sobre cómo se utilizan los valores de Shapley de coalición de teoría de juegos y muestran cómo se puede tomar inferencia de los modelos de predicción.

Capítulo 2 Analiza el problema de la predicción de quiebra en la economía española y cómo Aprendizaje Mecánico, no sólo proporciona una mayor precisión predictiva, sino que también puede proporcionar una interpretación diferente de los resultados en la que los modelos econométricos tradicionales no pueden. Se construyen una serie de ratios financieros y se pasan a una serie de algoritmos de Aprendizaje Mecánico. Se proporcionan estudios de casos que pueden ayudar a mejorar la toma de decisiones por parte de las instituciones financieras. También se proporciona una sección que contiene material complementario basado en un análisis más detallado.

Capítulo 3 analiza una política pública de contaminación en Madrid. Se propone un nuevo enfoque que utiliza Machine Learning para evaluar críticamente el impacto de la política. Una serie de características estacionales y de temperatura se pasan al modelo de

Aprendizaje Mecánico para predecir el nivel de contaminación de los próximos días. El capítulo se centra en la interpretación del modelo de Aprendizaje Mecánico, especialmente en torno al período en que se promulgó la política y, para ello, se aplican los valores de Shapley.

Capítulo 4.1 analice el problema de predicción de mortalidad entre pacientes ingresados en el hospital con COVID19. Los datos consisten en características de salud a nivel de pacientes a partir de datos de muestras de sangre. Estas variables son esenciales para realizar predicciones precisas a partir de modelos de Machine Learning. Este subcapítulo constituye la base del siguiente subcapítulo. Capítulo 4.2 extiende el capítulo anterior como analizando el problema de la prueba de estrés en los admitidos hospitales. Utilizando las predicciones de mortalidad de los capítulos anteriores se aplica un modelo de simulación de Monte Carlo donde se analizan diferentes escenarios en función del número de ingresos hospitalarios y camas disponibles.

UNESCO Codes: 1203.04 (Inteligencia artificial), 1203.07 (Modelos causales), 5302.02 (Modelos econométricos), 5311.07 (Investigación operativa).

Abstract in English

This Thesis examines problems in economics from a Machine Learning perspective. Emphasis is given on the interpretability of Machine Learning algorithms as opposed to black-box predictions models.

Chapter 1 provides an overview of the terminology and Machine Learning methods used throughout this Thesis. This chapter aims to build a roadmap from simple decision tree models to more advanced ensemble boosted algorithms. Other Machine Learning models are also explained. A discussion of the advances in Machine Learning in economics is also provided along with some of the pitfalls that Machine Learning faces. Moreover, an example of how Shapley values from coalition game theory are used to help infer inference from the Machine Learning models' predictions.

Chapter 2 analyses the problem of bankruptcy prediction in the Spanish economy and how Machine Learning, not only provides more predictive accuracy, but can also provide a different interpretation of the results that traditional econometric models cannot. Several financial ratios are constructed and passed to a series of Machine Learning algorithms. Case studies are provided which may aid in better decision-making from financial institutions. A section containing supplementary material based on further analysis is also provided.

Chapter 3 analyses a pollution public policy in Madrid. A new approach using Machine Learning is proposed to critically evaluate the impact of the policy. Several seasonal and temperature characteristics are passed to the Machine Learning model to predict the next day's pollution level. The chapter focuses on the interpretation of the Machine Learning model especially around the period the policy was enacted and for this purpose, Shapley values are applied.

Chapter 4.1 analyses the problem of predicting mortality among patients admitted to hospital with COVID19. The data consists of patient-level health characteristics from blood sample data. These variables are essential to make accurate predictions from Machine Learning models. This sub-chapter forms the basis of the next sub-chapter. Chapter 4.2 analyses and extends the problem discussed previously in Chapter 4.1 by extending the analysis to the problem of stress testing hospital admissions. Using the previous chapter's predictions of mortality a Monte Carlo simulation model is applied where different scenarios based on the number of hospital admissions and available beds are analysed.

UNESCO Codes: 1203.04 (Artificial Intelligence), 1203.07 (Causal modelling), 5302.02 (Econometric models), 5311.07 (Operations research).

Agradecimientos

I am grateful for the help and support of my supervisor Francisco Álvarez González. It has been a pleasure working with him on this Thesis, he has taught me how to construct papers of publishable quality and how to analyse research results in a clear and definitive way. Moreover, he has also allowed me the freedom to analyse 3 very different topics in economics. This, in turn, has allowed me to explore an exciting area which lies at the intersection between economics, programming, statistics and computer science, without his support this would not have been possible. I would like to extend my sincere thanks to the academic committee and the facultad de ciencias económicas y empresariales at the Universidad Complutense de Madrid for giving me the opportunity to write my Thesis at this university, it has opened many opportunities for me for which I am forever grateful.

I also would like to thank Stefano Sacchetto, José Azar of IESE Business School along with Carolina Villegas Sanchez, Vicente Jose Bermejo, Giulia Redigolo and Petya Platikanova of ESADE Business School who during the PhD provided me with employment which helped to fund my way through the doctorate.

Finally I would like to thank my family for their continued support in getting me to this point, without them, making it to this stage would not have been possible.

Thesis overview

This thesis applies Machine Learning to three different problems in economics. The problems are, (i) to predict corporate bankruptcy from financial statement data, (ii) assessing the impact of temporary traffic restrictions on NO_2 emissions, and (iii) how days spent in hospitals marginally reduce COVID19 probability of mortality. These problems are clearly very different in nature from one another but all of them have some common roots which make Machine Learning *a priori* more appropriate than classical econometric models. Machine Learning contains a broad body of knowledge with many of its applications being cross-disciplinary, with that said, this Thesis aims to apply Machine Learning to the field of economics. More specifically, it makes use of a relatively narrow family of Machine Learning techniques. The first chapter presents a guided tour from very general concepts down to the specific techniques that constitute the central part of the Thesis. The chapters that follow are devoted to each of the above mentioned problems. This forward aims to summarise the contents of each chapter. A special focus is put on the economic relevance of the main findings, omitting technical details for later chapters.

Machine Learning in economics, guided tour: Chapter 1

This chapter introduces some of the fundamental concepts and methods in Machine Learning with a focus on how it can be applied to economics. Moreover, whilst this chapter covers a wide range of concepts it is not an exhaustive overview of Machine Learning. More emphasis is placed on the methods and models that are used later in the Thesis. Generally, Machine Learning consists of a collection of computational methods in which the computer learns from some dataset. For example, a model is able to learn a mapping of previous patient characteristics and determine the likelihood of suffering from a given illness in the

future. Additionally, a model is able to make more accurate predictions on consumer sales preferences from past user activity. More generally, the range of potential applications of Machine Learning is vast, and so is the number of available models, with different fields all contributing to the advancement of Machine Learning.

Among the extensive collection of Machine Learning models, the fundamental concept of a *tree* is central for this Thesis. As an illustration, consider the example of students passing an applied macroeconomics course, a binary classification task. Students previous grades in other courses are used as the predictor variables. A tree may make a first decision rule by splitting students into two groups, those who had grades less than 80% and those who had grades greater than or equal to 80% in mathematics. Sticking with the side of those students who obtained greater than 80% a second decision rule might split these students into two further sub-groups. Those students who obtained less than 70% and those who obtained greater than or equal to 70% in microeconomics. Finally, sticking with the students who obtained greater than 70% the tree may make a further split on this group. Those who speak English and those who do not. Therefore, the students who are most likely to pass the macroeconomics course are those students who fit into the sub-group - obtained greater than 80% in mathematics, those who had greater than 70% in microeconomics and those who speak English. There are other non-linear paths a student could follow and thus we have partitioned the students into sub-groups by using two type of data, quantitative, the continuous variables (course grades) and qualitative, a binary variable (language). Naturally, the tree can become more complex with the inclusion of more variables and the central question becomes: what tree should we use in order to make valuable and accurate predictions?

If we have past data on students grades and language capabilities, along with the observed grade in macroeconomics we can compare the predictions to the actual value and

thus select a *better* tree which is called *supervised learning*. The chapters throughout the Thesis focus on *supervised learning* in which the model improves by correcting the residual error of its predictions. The main algorithm in the Thesis is Extreme Gradient Boosting (XGBoost) which sequentially selects trees which minimises a loss function by minimising the prediction errors and penalising the complexity of the trees.

In order to predict a given response variable, Machine Learning automatises the search for the functional form which best fits the data. Conversely, econometrics requires the analyst to take a more active role in this search. Moreover, econometrics assumes that there is a *true* population that generates the data and depending on the characteristics of the population, certain sample statistics are *reasonable*[1] estimators for certain parameters that define the population from which the data has been sampled. In contrast, Machine Learning does not presume the existence of an underlying -unobserved- population. Roughly speaking, if we know the grades and language capabilities for *each* and *every* one of the students then we *observe* the population itself. The key question then is how to extract from this the relevant information for the prediction exercise under consideration. The aim in Machine Learning is generalisability onto new unseen data. Despite these essential differences between Econometrics and Machine Learning, both fields share some common problems in the handling and treatment of the data, such as endogeneity issues and variable selection. The latter can be somewhat *solved* in Machine Learning and tree based models since if a variable does not contribute to the models prediction, the model will simply not use it. The comparison between econometrics and Machine Learning along with the problems in both disciplines are discussed in the first chapter of the Thesis.

[1] More formally, several concepts apply, such as unbiased, efficiency and consistency.

Bankruptcy prediction: Chapters 2.1 and 2.2

The first research problem addressed in this Thesis involves bankruptcy prediction. Annual data on company financials was downloaded from ''Sistema de Análisis de Balances Ibéricos" from 1992 to 2016 and from this data a number of financial ratios relevant in the literature was constructed. The data contains 58,000 companies, among which some 6,000 went bankrupt during the sample period. The response variable is the bankruptcy status of the firms, a binary variable which indicates if a firm has gone bankrupt or not. Due to the relative scarcity of bankruptcy events, Machine Learning is a natural fit for this type problem.

This chapter deals with prediction exercises and compares the predictions of different Machine Learning models along with a classical logistic regression model. Moreover, Machine Learning models do not just outperform traditional models but the focus and emphasis is put on what Machine Learning can teach us about the data that can not be learned using traditional econometric models. In this regard, two findings are worth mentioning.

Firstly, the impact of a given variable when determining the probability of bankruptcy, averaging across all firms, is not linear. For example, the XGBoost algorithm shows that one of the most influential ratios is Total Liability to Total Assets (TL.TA) and generally the higher the TL.TA, the higher the probability of bankruptcy. Moreover, the influence of this variable is $\mathcal{S}$-shaped rather than linear, that is, an increase in the TL.TA has a lower impact for the likelihood of bankruptcy when the departure point is a low level of TL.TA than when it is high, which makes sense economically.

Secondly, XGBoost allows us to explore how any two unique firms may have roughly the same probability of bankruptcy but for different reasons. There are a number of scenarios under which the reasons that lead to bankruptcy predictions are important as -if not more

than- the probability itself. In this regard, if the variables are pooled additively in a logistic regression model, the marginal contribution of a given variable is bound to be independent of the values of all other variables. Admittedly, non-linear terms can be also tested, but not systematically tested. In contrast, the function space that XGBoost explores is much larger than any *by-hand* testing procedure. Consequently, the final model delivers virtually firm-specific impacts of each variable, that is, firm-specific explanations for each of the given firms probability of bankruptcy. Case studies to illustrate this fact are presented in this chapter.

The final comment is in regards to the prediction exercises. By standard statistics presented on prediction errors from a confusion matrix, XGBoost exhibits a clear advantage over a more simpler logistic regression. In fact, most of the Machine Learning models outperform the logistic regression model. The underlying reason is presumably quite simple: for large datasets a *one-equation-fits-all* breaks down and the models which work best are the ones which are able to choose flexible functional forms relating the regressors to the dependent variable. One of the demands of a referee as part of the submission process was a deeper and wider discussion of the Machine Learning models. As for this Thesis, I present in chapter 2.1 the accepted version of the paper published in Computational Economics Smith and Alvarez (2021c), while chapter 2.2 includes the supplementary material from the suggestions of the external reviewers.

Assessing the impact of traffic restrictions: Chapter 3.1

This chapter contains the second research problem studied in the Thesis. Madrid's local government established in 2017 a protocol in which traffic restrictions are enforced when the measurement of NO (Nitrogen Oxide) levels are above a pre-defined threshold. These restrictions are set on a daily basis and take effect gradually: in the first day the speed limit in the inner ring (M-30) reduces from 90 to 70 km/h, and from here further restrictions

are enforced if NO levels continue to be above the pre-defined threshold. There are 24 pollution measuring stations located throughout Madrid and depending on how close they are to congested traffic routes, they are defined as *Traffic*, *Background* or *Suburban*. The central question of this research chapter is to assess, at a station level, the impact of the protocol.

The protocol has been controversial in the media since the beginning and so an assessment of its impact has interest beyond academia. For this task, there are two major challenges. First, unlike permanent protocols, such as zero-emissions zones, the protocol switches between active and inactive on a daily basis. Since 2017, it has been active only 59 days, never more than 11 consecutive days. Second, there is an endogeneity issue. When we plot the daily time series of NO pollution and days in which the protocol was activated, the peaks of the former coincide with the days of the active protocol and the question remains, who is influencing who?

In order to try and analyse this problem I used XGBoost and Shapley values. The dependent variable is daily NO pollution levels and the protocol is a binary regressor. Other regressors were used such as weather and seasonal variables. In the category of the former, wind speed was one of the most significant variables in the model. Other weather variables such as humidity, temperature and rain were also included in the model. It is worth mentioning that NO emissions are strongly connected to road traffic, so the perception, *it seems it is going to rain today*, might be more influential than the fact, *it is raining*, as it is the perception what might determine the choice between public and private transportation. Among the seasonal variables, time of the year, whether it is a working day or not and variable of a similar nature were included. All weather and seasonal variables are at the daily frequency and independent models are constructed for each measuring station.

Whilst in the previous problem, the dependent variable had relatively few instances of bankruptcy observations, here the regressor of interest has very few observations - few days the protocol was activated. Additionally, as previously discussed, there is an endogeneity issue. To overcome both difficulties I proceeded in the following two steps. Use *Shapley values* to determine the impact of each regressor in the models predictions. Secondly, a closer look at the indirect effects is analysed.

The underlying idea behind Shapley values comes from cooperative game theory. Suppose we want to measure the impact of the feature value $X = \{Wind\ speed\ is\ high\}$. We consider all possible coalitions, or groups of feature values, in which X is a member. For each of those coalitions, we compute the difference between the prediction using the whole coalition and the prediction using the whole coalition without X. The Shapley value of X is the weighted average difference across all coalitions. Once Shapley values are computed, the endogeneity is overcome by looking at the indirect effects. The most clear indirect effect is for the variable wind speed. The Shapley values for the different wind speed values show that the higher the wind speed, the lower the pollution level, which is in line with previous literature. The analysis shows the indirect effect of the protocol as follows: the effect of an increase in wind speed is higher when the protocol is active than when it is not. In this Thesis, I present in chapter 3.1 the accepted version of the paper published in the International Journal of Environmental Science and Technology Alvarez and Smith (2021).

COVID19 mortality characteristics: Chapter 4.1 and 4.2

This chapter is split into two parts, the first part uses patient characteristics in order to predict which patients are at most risk of mortality from COVID19. This part contains data which is quite far removed from economics but does contain an important feature which is crucial to the second part. The variable of interest is the number of days a patient remains in hospital. However, using just this variable to predict patient mortality is not going to

produce any meaningful predictions and for this reason further patient characteristics on blood sample data are also used such that the model has sufficient data to make predictions. Shapley values are again applied but with a focus on the days in hospital and age variables.

The second part of this chapter concerns the variable days in hospital and its link with reduced mortality rates of COVID19 in patients. The motivation for this paper comes from a simple reflection: how should we measure pressure put on hospitals in times of crisis? Consider a hospital with, say, a hundred beds available, under two different scenarios, $\mathcal{A}$ and $\mathcal{B}$. In scenario $\mathcal{A}$, a hundred new patients come to the hospital every day, each patient needs exactly one day in hospital for full recovery. In scenario $\mathcal{B}$ twenty new patients come every day, each patient requires ten days for recovery. Under either scenario, the hospital will be operating at full capacity in just a few days, in just a single day under scenario $\mathcal{A}$. However, scenario $\mathcal{A}$ seems preferable. In short, in addition to the percentage of occupied beds, from an economic perspective, the *productivity* in hospitals should be taken into account, the question remains, how to combine both features.

Overall, this chapter combines two elements. The first being the estimation procedure. Using data on patient characteristics an estimation of the probability of mortality as a function of these characteristics is made. The second utilises the previous Machine Learning probabilities and Monte Carlo simulations are run under different stress testing scenarios in which combinations of the number of available beds and the number of new patients arriving at the hospital are analysed. In this Thesis, I present in chapter 4.1 the accepted version of the paper published in Expert Systems with Applications Smith and Alvarez (2021a) along with the code being verified as reproducible code and published on CodeOcean Smith and Alvarez (2021b).

Chapter 1

Representative models of Machine Learning and applications in economics

Abstract: This paper is a discussion of Machine Learning applied to the subject of economics and finance. The paper first introduces the background of Artificial Intelligence (in which Machine Learning is a subset of) and then aims to relate the problems in Machine Learning to the same problems faced in Econometrics and discusses how Machine Learning tries to overcome these problems. A comprehensive review of the main models and methods is also made. Finally, a discussion is made on the transition from decision tree models to more advanced models such as Gradient Boosting, the fundamental building blocks of the main models used in this Thesis.

JEL Codes: C01 (Econometrics), C14 (Semiparametric and Nonparametric Methods), C44 (Operations Research), C58 (Financial Econometrics).

1.1 Introduction

The past few decades has seen substantial growth in the field of economics. There has also been a transition away from more traditional theoretical work and a move more towards applied empirical research Hamermesh (2013). Previously, empirical research was often difficult to do in part due to lack of computational resources but also in part due to inadequate access to data. This has changed over the last few decades and the growth of

the internet has allowed researchers to obtain better access to data. It is now more easier than ever to access various data sets for empirical analysis. However, with the increase in data brings about its own challenges related to data retrieval, management and storage. Typically, economists often analyse data which can fit into a spreadsheet or data which is in a *rectangular* format with N observations and K variables, and $K < N$. However, more granular data is increasingly being collected which naturally increases the size of a dataset. Additionally, there are econometric challenges, conventional econometric models may now not be the most suitable method to apply to the data collected such as the case when analysing text or imagery datasets, but also when analysing *big data*. When dealing with *big data* it might be the case that there are more predictors than appropriate for estimation and thus variable selection becomes important. Moreover, the size of the data may require more powerful data manipulation tools. The term *big data* not only comes from an increase in the scale of the data but also in terms of the type of characteristics that the data has. Machine Learning can help sift through data that contains many predictors and seek non-linear and highly interactive combinations which reliably predict outcomes. Einav and Levin (2014) provides an early survey addressing the question of big data's influence in economics and how big data may impact economic policy and economic research and point at the need for new methods to be employed, but do not elaborate further. (Varian, 2014) extends the discussion of big data in economics and suggests the field of Machine Learning to address the problems arising from big data. Athey (2017) discusses how *big data* can be used for policy problems.

Econometrics is based on sound mathematical statistics and probability theory and the models are robust with attractive properties. Given a dataset and a specified model, the parametric regression parameters are obtained through an algebraic formula. Ordinary Least Squares (OLS) gives the Best Linear Unbiased Estimator (BLUE) of the regression's coefficients. Moreover, Machine Learning algorithms are usually complex and cannot be

mathematically described in a single formula. The solution to a Machine Learning model is determined algorithmically and the best solution is chosen depending on the structure of the data and is therefore more of an approximation than an exact solution. Machine Learning in its most simplest form is allowing a model to learn -or recognise patterns- from the data in order to make some decision without being explicitly programmed, it is a sub-field of computer science that largely evolved from computational learning theory and pattern recognition studies.

Machine Learning utilises the use of data-driven model selection and uses the learning of a function f which maps input variables $x = (x_1, \ldots, x_p)$ to output variables y, $y = f(x)$. The form of the function is unknown and the form is in part determined as a function of the data, as opposed to conducting a single estimation. Unlike in traditional econometrics, the field of Machine Learning has an overwhelming number of models or functional forms that relate x to y and it is difficult to know when and where a given algorithm fits a specific problem. Usually, different Machine Learning algorithms are applied in order to see which algorithm is best at approximating the underlying function. The algorithm learns the mapping function from a *training* dataset and the objective is to achieve goodness of fit in the independent held-out *test* dataset, through the minimisation of errors between the predicted and actual outcome. Moreover, econometrics attempts to understand the estimator $\hat{\mu}(x)$ and evaluate the marginal impact from the change of one covariate when all others are held constant which, is not the explicit aim of Machine Learning.

Data analysis in statistics and econometrics can be broken down into four categories: 1) prediction, 2) summarisation, 3) estimation, and 4) hypothesis testing. Machine Learning is concerned primarily with prediction but it also offers a set of tools that can usefully summarise various sorts of non-linear relationships in the data. Much of applied econometrics is concerned with detecting and summarising relationships in the data Varian (2014). In

smaller data sets, these traditional econometric models often perform better than complex Machine Learning models. However, when the data becomes larger, Machine Learning models tend to outperform traditional models.

Machine Learning has been growing in both academia and in the private sector in recent years, it not only provides economists a new set of tools but also solves different problems. Figure 1.1 plots some keyword searches from the academic database `Web of Science` based on a given keyword appearing in the *title* to demonstrate the growth in Machine Learning related research relative to more traditional topics.

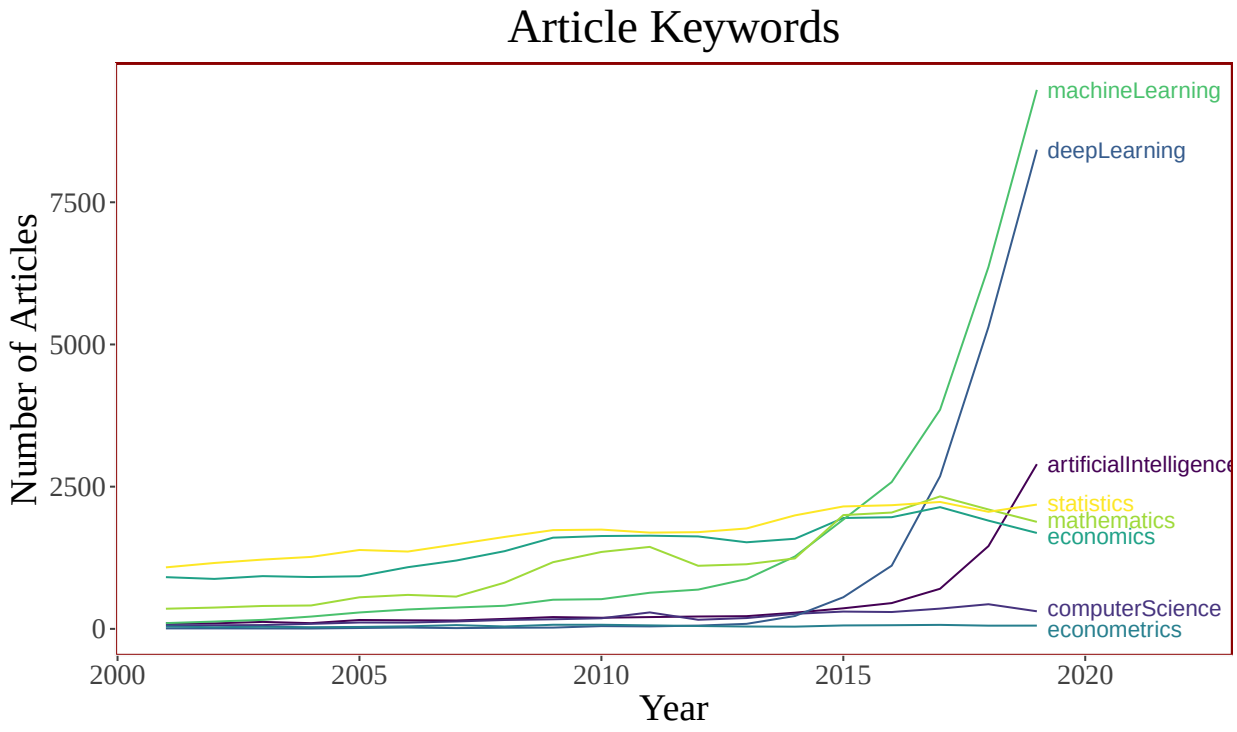


Figure 1.1: Number of times a given keyword appears in articles published on *Web of Science*

This paper aims to be a guided tour from econometrics to a subset of advanced Machine Learning algorithms that are well suited for a number of applied economic applications. The following paper is broken-down as follows, first an introduction to artificial intelligence and

its impacts on the economy and applications to economic problems. Then, how Machine Learning fits in with traditional econometrics. Followed by, an overview of different types of Machine Learning problems and how to handle them. Finally, an introduction to decision trees and their evolution to Extreme Gradient Boosting.

1.2 Artificial Intelligence

The term Artificial Intelligence usually refers to the broad definition of a computer system that is able to perform tasks that normally require human intelligence. Due to the broadness of the definition the goalposts of what defines Artificial Intelligence keeps changing with its continued advancement and thus Artificial Intelligence defines what machines can not currently do. That is, a computer that can beat an expert chess player would have been considered Artificially Intelligent a few decades ago. However, after IBM's Deep Blue beat Garry Kasparov in the 1990's the definition evolved and playing chess was considered a computer science problem and other challenges became Artificial Intelligence problems. The development of Artificial Intelligence can be categorised into periods of growth and low interest. The periods of low interest are often defined as *AI winters* and is a period of quiet time in the research and development where funding for Artificial Intelligence initiatives dry up. AI winters historically came about when the promise of Artificial Intelligence failed, the return on investment disappointed or when computational power was not advanced enough for more complex systems. These periods of quietness were followed by periods of growth and renewed interest, often when Artificial Intelligence systems became more cost effective and computational power improved. Figure 1.2 gives a brief overview of Artificial Intelligence and its corresponding sub-fields.

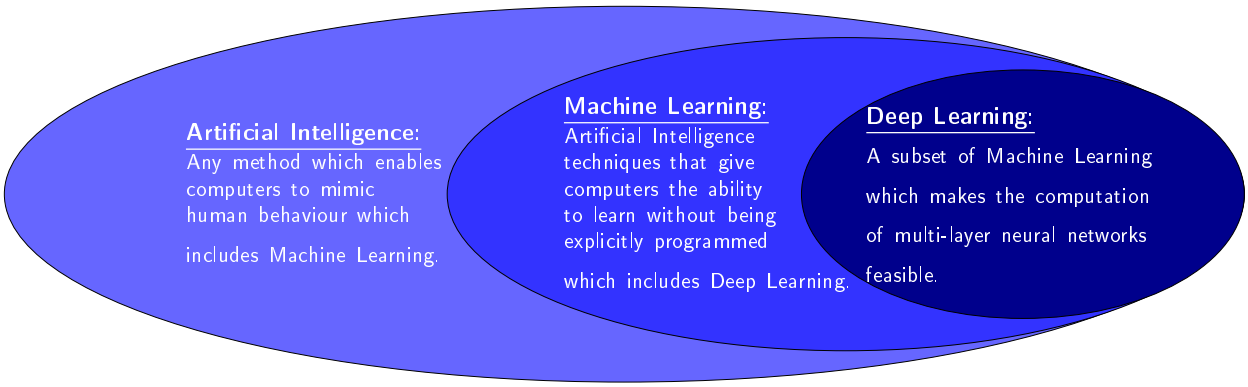


Figure 1.2: Overview of Artificial Intelligence and sub-fields

1.2.1 Impact of Artificial Intelligence in the Economy

The prospect that computer intelligence could be able to complete tasks that a human can do would be extraordinary and would impact economies in a big way. However, this does not come without potential societal problems. This section deals with a brief review of how Artificial Intelligence is impacting or could potentially impact economies.

Electricity, internal combustion engines and semiconductors facilitated the automation in the last century, however, Artificial Intelligence looks to automate human tasks which were previously thought to be too difficult to automate. Korinek and Stiglitz (2017) discuss the general conditions under which advances in Artificial Intelligence may lead to a Pareto improvement. Moreover, Artificial Intelligence necessitates large adjustments, and whereas society may be able to adjust to slow changes, it may be more difficult to adjust when the pace is more rapid, due to imperfections in capital markets. These outcomes can also be Pareto inferior and the more willing a society is to support this transition and to provide support to those who are "*left behind*", the faster the pace of innovation a society can accommodate, ensuring outcomes that are Pareto improvements. If Pareto improvements can no longer be ensured, it may lead to resistance in innovation from those in society who are losing, leading to uncertain political and economic consequences.

The displacement effect brought about by the introduction of Artificial Intelligence tends to lower the demand for labour and wages. However, this effect could be counteracted by a productivity effect, resulting in cost savings generated by automation and therefore increases the demand for labour in non-automated tasks. Acemoglu and Restrepo (2018) summarise the implications of automation and Artificial Intelligence on the demand for labour, wages and employment. They discuss the displacement effect of machines taking over human tasks and the expansion effect as growth leads to the creation of new tasks where human labour has a comparative advantage over machines. Furman (2018) discusses how Artificial Intelligence has both potential benefits for productivity but also consequences, such as higher inequality and a drop in labour force participation. In order to overcome these consequences, institutions should help workers adapt to labour market changes. This adaptation to Artificial Intelligence would be more successful than imposing large-scale changes to social policies, such as universal basic income in the future, thus maximising the benefits of Artificial Intelligence and mitigating the disruptive side effects. Moreover, inequality brought on by the advancement in Artificial Intelligence may also affect skills-based inequality and disproportionately increase the wages of highly educated people and may suppress the wages of the lower educated due to a shock in job automation. Job losses for low-wage jobs are significantly more likely than job losses from higher educated jobs since the higher educated jobs will have people with an innate ability to grasp new and more complicated tools and thus higher educated people are better at learning the necessary new skills that Artificial Intelligence will bring about and therefore will likely benefit disproportionally to the uneducated with this advancement. On the contrary, the opposite may occur, high paid jobs such as a doctor and financial analyst may be at greater risk, since prediction is at the core of what they do and thus if Artificial Intelligence is better at prediction, then these highly skilled jobs are at risk, therefore the diffusion of Artificial Intelligence could lead to a de-skilled labour force and reduced overall inequality. Moreover, it has been more challenging for lower-skilled workers to learn the skills required as technology automates

certain aspects of their jobs when compared to higher-skilled workers Agrawal et al. (2019). Nilsson (1984) studied one of the early implications of Artificial Intelligence on employment and the distribution of income stating that unemployment has historically been thought of as a problem that needs correcting and that unemployment caused by the introduction of Artificial Intelligence will be more liberating than unemployment caused by historic automation technologies. That is, people will be able to spend time on activities that will be more gratifying and humane than time spent on working. Bessen (2018) applies a model of demand to predict employment levels over the next 10 or 20 years and show how Artificial Intelligence is likely to affect jobs. Another important reason why inequality might increase relates to the capital share in the economy. Labour share of GDP is falling Autor et al. (2020) and therefore if Artificial Intelligence is a more efficient form of capital, then capital share will rise at the expense of labour share Acemoglu and Restrepo (2018) and Sachs (2019).

Aghion et al. (2017) analyse the potential impact that Artificial Intelligence might have on economic growth and the division between labour and capital. They demonstrate that if Artificial Intelligence was an input to the production of ideas, then exponential growth could be generated, even without an increase in the number of humans generating ideas. Moreover, Bloom et al. (2020) show that across a number of industries, research effort is rising substantially while research productivity is declining sharply, therefore scientific research ideas are becoming more difficult to find. Moreover, if Artificial Intelligence is an input to the production of ideas, then we would have a way out of this potential driver of slowing productivity growth and instead, productivity growth could accelerate Agrawal et al. (2019).

Varian (2018) discusses how Artificial Intelligence and Machine Learning can impact many industries and how it might affect the industrial organisation of firms which provide Artificial Intelligence services and industries which adopt the technology. Cockburn et al.

(2018) discuss the impact of deep learning on innovation and how deep learning represents a new general-purpose invention that can reshape the nature of the innovation process and organisation of R&D.

Kleinberg et al. (2015) highlight a case in health policy resource allocation considering the problem of which patients should not be given a hip replacement. They use Machine Learning to identify high-risk patients by predicting the probability that a person who undergoes a hip replacement will die within a year from the operation from other causes. Therefore, surgery should only be given if the patient is predicted to live long enough to enjoy the benefits of the surgery, where giving hip replacements for a patient who dies soon after surgery is futile, a waste of money and an unnecessary painful imposition on the last few months of the patient's life. Moreover, these kinds of prediction problems fail to answer the question of which patients - among those who are most likely to survive past a year - should be given the highest priority when receiving surgery. Machine Learning can partially, but not fully address the resource allocation problem since it requires estimates of heterogeneity in the effect of surgery. Therefore, optimally allocating medical resources to patients for whom the causal effect of the surgery on patient welfare is highest is a more difficult problem.

1.2.2 Applications of Artificial Intelligence in Economics

Athey and Imbens (2019) provides an introduction to a number of the most popular and useful Machine Learning methods along with core concepts from the perspective of econometricians, highlighting that more sophisticated work can be achieved by incorporating Machine Learning into empirical economic research. Athey (2018) also provides a comprehensive overview of the early contributions of Machine Learning in economics as well as predictions about its future contributions in economics. This section aims to extend this overview by providing a further review of the literature of Machine Learning applications in

economics.

Early research efforts in economics applying *Neural Networks*[1] to macroeconomic time series data have mostly analysed the problem of univariate time series, Nakamura (2005) applied Neural Networks in order to forecast U.S. inflation levels and found that Neural Networks outperformed univariate autoregression models on average for short horizons of one and two quarters. Additionally, Chen et al. (2001) applied Neural Networks to forecasting U.S. inflation rates and found that they outperform a benchmark linear model in terms of forecast mean squared error and forecast mean absolute deviation error. Stock and Watson (1998) compared a number of linear and non-linear econometric models including a Neural Network to forecast a series of univariate U.S. macroeconomic time series. Gonzalez (2000) applied Neural Networks to forecast Canada's real GDP growth and found that the forecasting accuracy of the Neural Network was superior to linear regression but found that there was little evidence that the improvement in forecasting accuracy was statistically significant.

Other more recent works evaluating macroeconomic models include; Athey et al. (2019) analysed ensemble methods - stacked regressions Breiman (1996b), for causal effects in a panel data setting in which the goal is to estimate causal effects of an intervention by predicting the counterfactual values of outcomes for treated units, had they not received the treatment. Other approaches have been previously proposed for this problem, such as regression, synthetic control and matrix completion methods, moreover, they show that the ensemble approach performs better than any of the individual methods when predicting

[1]See section 1.19.2 in the Appendix for a more detailed definition of a Neural Network. Briefly, a Neural Network takes input variables that are fed as input to neurons in the first layer, these are connected to the next layer through channels and are assigned weights. The inputs are then multiplied by the weights and passed to the next hidden layer as inputs. A bias is computed and added to the input sum, this is then passed to an activation function. Activated neurons transmit data to the next hidden layer in the network. The data is propagated through the network (forward propagation) and probabilities are calculated and given as an output. The output is compared with the actual output and the error is transmitted back through the neural network (backward propagation) and the weights are adjusted accordingly. This forward propagation and backwards propagation is performed iteratively until the network is able to correctly predict the outcomes.

GDP, logGDP and the Growth rate of GDP. Smalter Hall and Cook (2017) used Deep Neural Networks to forecast U.S. unemployment and found that they outperformed benchmark models, such as Directed Autoregressive Models - (DARM) and analysts' consensus forecasts - such as the Survey of Professional Forecasters (SPF) on short time horizons. Diebold and Shin (2019) applied partially egalitarian LASSO (peLASSO) and compared it with RIDGE,[2] LASSO, eRIDGE and eLASSO to forecast quarterly 1-year ahead Euro-area real GDP growth (year-on-year percentage change) from Q1 1990 to Q4 2016. Sermpinis et al. (2014) applied Support Vector Machines[3] to forecast U.S. inflation and unemployment comparing it with a benchmarked random walk model, an autoregressive moving average model, a moving average convergence/divergence model, a multi-layer perceptron, a recurrent Neural Network and a genetic programming algorithm. They found that the Support Vector Machine outperformed the benchmark models and identified which macroeconomic variables can be relevant predictors of U.S. inflation and unemployment. Ng (2014) and Döpke et al. (2017) used Random Forests and Boosting models to try and predict recessions. Coulombe et al. (2020) study some important underlying features which drive Machine Learning models in the context of macroeconomic forecasts - on Industrial Production, Unemployment rates, Consumer Price Index, SPREAD[4] and housing starts HOUST - discussing non-linearities, regularisation, cross-validation and alternative loss-functions in data-rich and data-poor environments. They find that non-linearities are important in data-rich environments when predicting real activity series and long time horizons. Jeong et al. (2016) compared Random

[2]See section 1.19.3 in the Appendix for the definition of RIDGE, LASSO and Elastic Net regression models. More briefly, these models attempt to correct for the bias-variance trade-off from an OLS model. In RIDGE regression, a penalty term is added to the OLS model which is the square of the coefficient multiplied by a penalty λ. In LASSO regression, a penalty term is added to the OLS model which is the absolute sum of the coefficients multiplied by a penalty term λ. In both models when $\lambda = 0$ we have a normal OLS model. The difference between the two models is that RIDGE never sets the value of the coefficient to absolute zero whereas LASSO can. Elastic Net combines the regularisation of both RIDGE and LASSO.

[3]See section 1.19.4 in the Appendix for the definition of Support Vector Machines. More briefly, a SVM attempts to find a hyperplane that best divides the dataset into classes. The support vectors correspond to the data points which are nearest to the hyperplane and the objective is to maximise the margin from the hyperplane to the data points.

[4]Difference between the 10-year Treasury Constant Maturity Rate and Federal Funds Rate.

Forest models to multiple linear regressions when predicting crop yields of wheat, maize and potato using climate and biophysical variables at global and regional scales. They found that Random Forests outperformed traditional linear regression in all performance metrics. Chalfin et al. (2016) used stochastic gradient boosting and regression with LASSO regularisation in order to improve on teacher tenure decisions and police hiring decisions.

Additionally, some of the problems in which regression problems cannot be applied include the analysis of text, imagery, sound and video data in which the number of variables is often larger than the number of observations. For example, when determining when and which restaurant needs to undergo a hygiene inspection is a difficult task for public health policy-makers and hygiene inspectors. Machine Learning can learn a mapping between customer reviews about a restaurant and signals in the overall hygiene quality based on the written text customers are leaving on social media, and it can discriminate between severe offenders and good quality restaurants through the comments left by previous customers. That is, poor-quality restaurants will have comments corresponding to the hygiene standards and quality of food. The learned underlying functional form from the model can help inspectors in determining which restaurant requires immediate inspections and which restaurant does not, freeing up many resources at health departments Kang et al. (2013). Gentzkow et al. (2019) provide a comprehensive literature review on text analysis applied to economics and finance.

Big data and Machine Learning is also being extensively applied to imagery datasets. The classical approach to machine vision consisted of a rules-based approach which had limited success. Researchers would identify pixels in the images based on colour brightness and edges and then use these features in order to predict what an object in an image was. More modern approaches to the task have had much more success, it removes the rule-based system and instead uses layered Neural Networks to identify these features from the raw

pixels. The application to image recognition in economics is huge, it is possible to collect high-resolution satellite imagery and apply similar techniques to economic problems. Some applications are identifying shipping containers and analyse shipping traffic along a number of different shipping routes in order to understand supply and demand between countries and how this changes over time. Moreover, satellites can collect images from a number of superstore car parks in order to gauge whether a company is expected to increase its revenues in the next quarter based on the identification and detection of an increase in the number of cars in the parking lot over time. Additionally, satellite imagery has been used to analyse economic output and GDP estimates based on how much light is emitted from a number of cities. Henderson et al. (2012) show how nighttime luminosity can correlate with economic output. Donaldson and Storeygard (2016) discuss an overview of how remote sensory satellite data can be applied in economics.

1.3 Prediction and Causation

Econometrics bases itself on cause and effect and establishing the link between the two is fundamental in social sciences and assumptions on human behaviour imply restrictions on the plausible values of causal effect. Additionally, restrictions on these causal effects may depend on additional parameters of human behaviours and decision-making. Machine Learning may be a useful tool in estimating these additional parameters, providing high-quality estimates of causal effects and counter-factual outcomes. Econometrics involves *parameter estimation*, that is, producing unbiased and consistent estimates of the parameters β which underlines the relationship between a dependent variable y and independent variables x. The estimates produced from Machine Learning algorithms that output regression coefficients are rarely consistent. That is, econometrics assumes that there is an unobserved population from which we have a sample and the goal is to estimate the unobserved parameters which characterise the population. Depending on the properties and sampling of the population the estimator has desirable properties, Best Linear Unbiased Estimator (BLUE)

and consistency. Machine Learning does not attempt to be consistent and emphasis is given on the within-sample evaluation, therefore, concepts such as the marginal contribution of a given variable are measured in a much less rigid way than linear regression. Therefore, if we construct a model for $\hat{y}$ then it is unlikely in Machine Learning that its corresponding $\hat{\beta}$ will have the properties that we would expect from traditional econometrics. Therefore, Machine Learning is concerned with the component $\hat{y}$ rather than $\hat{\beta}$ and its power comes in its flexibility in functional form and accuracy in predictive modeling, however, as previously discussed it does not produce stable estimates in its underlying parameters and its application is mostly suited for $\hat{y}$ problems. This gives Machine Learning an advantage over classical econometrics in that they do not make any pre-specified assumptions about the functional form of the model equation, the interaction between variables and the statistical distribution of parameters.

Unlike in OLS regression ensembles *Random Forest*, *XGBoost* and *LightGBM* can handle irrelevant features. Linear regression requires us to hand select a number of variables and manually generate interactions between variables and this could produce more predictors than observations $p > n$, then there is no unique solution to the standard linear regression problem. There are unnecessary linear dependencies among the predictors in the matrix and therefore once we have the coefficients for n predictors, the coefficients for $(p-n)$ predictors can be expressed as arbitrary linear combinations of those first n predictors. One case is when faced with the facial recognition problem and the combinations of pixels to identify faces are non-linear, the predictors here are the pixels that could contain many predictor interactions. This would break down in OLS and in order to proceed Machine Learning searches for the interactions automatically.

One of the most important areas that Econometrics and Machine Learning can collaborate involves causal inference. If we understand why something happened, we can change

our behaviour to improve future outcomes. Econometricians have developed a number of causal inference methods such as *instrumental variables*, *difference-in-differences*, *regression discontinuity* and natural and designed experiments Angrist and Krueger (2001). The primary interest in economics is in the estimate of a causal effect, for example, the effect of an increase in the minimum wage or the effect of price reductions. Whereas the primary interest in Machine Learning is the goal of prediction.

As previously discussed, it is often easy for Machine Learning algorithms to over-fit the training data as opposed to uncovering the real and causal relationships which should hold on new data drawn from the same distribution as the one used to train the model. Deep learning algorithms are very good at finding patterns in the data but can not explain how they might be connected. For many use cases, Machine Learning's ability to find correlations is sufficient but these correlations can not help us understand why something happened. Causality specifies a one-way direction $X \rightarrow y$ or $X \leftarrow y$ whereas, correlations are weaker since they characterise a two-way relationship $X \leftrightarrow y$. If the Machine Learning model can not tell us why something happened, it can only tell us the probability of what might happen next. Moreover, Machine Learning algorithms which are able to capture the causal relationships will be more generalisable onto unseen data. Causal inference would allow us to analyse what would happen when we change some of the underlying assumptions of a model. Therefore, understanding the cause and effect of Machine Learning models would allow them to be more efficient and smarter. An AI robot that is capable of understanding that when it drops things it causes them to break, would not need to go through the phase of throwing many things onto the floor in order to learn that things break when thrown onto the floor. Moreover, if an AI robot is given a hose and learns that the hose puts out a fire, then in the next task it is given a bucket of water it would be unable to know what to do with the bucket of water without having to explicitly learn from scratch what to do with it since it has not learned the causal relationship between water and fire. Causal inference can

help with the problem that in reality much of the real-world data is not generated in the same way as the data the Machine Learning algorithm trained on. Causal inference can help overcome this problem by considering what may have happened when the data was slightly different. *Transfer learning* focuses on utilising what a model has learned while solving one problem and reusing it as a starting point for a model on a related second problem, e.g. use a trained model which recognises cars and apply it when trying to recognise trucks Pan and Yang (2009).

There are a number of methods used to distinguish actual causality from spurious correlations. The direction of the causal relation is often based on sound economic theory, that is, economic theory tells us that rainfall affects the future price of commodities however, the future price of commodities does not affect rainfall. Moreover, instrumental variables (IV) are often applied to remove any reverse causation through introducing other instruments (variables) which are unaffected by the dependent variable. It is also assumed that causes must be prior in time than their effects and econometric tests such as Granger-causality tests are applicable to time series models and rather than testing whether Y causes X, it tests whether Y forecasts X. The proposition by Hume (1748) and formalisation by Lewis (1973) in which Lewis defines a notion of causal dependence between events, paraphrased as: Where C and E are two distinct possible events, E causally depends on C if and only if, E always follows after C and E does not occur when C has not occurred (unless something else caused E)[5] Prosperi et al. (2020).

Beery et al. (2018) argue that Machine Learning, and more specifically image classification research aims to beat the most popular image classification benchmark datasets[6] where the models are evaluated on data from test distributions which come from the same train

[5] Originally, Lewis (1973) *Where c and e are two distinct possible events, e causally depends on c if and only if, if c were to occur e would occur; and if c were not to occur e would not occur.*

[6] ImageNet (Deng et al. (2009), Microsoft-COCO Lin et al. (2014), PascalVOC Everingham et al. (2010) and Open Images Krasin et al. (2017)).

distributions. Whereas, it is evident that the model should perform well on the data coming from the same distributions it is also important to characterise the generalisation behaviour of the models, especially when the test distribution deviates from the train distribution. In their study, they analysed how drastic landscape and vegetation changes affect the capabilities of the Machine Learning models generalisation. They found that state-of-the-art image classification algorithms show excellent performance when tested at the same location where they were trained but found that this performance - and generalisation - dropped significantly when new locations were used. For example, most camels are photographed on sand and most cows are photographed on grass, however, the underlying feature characteristics between camels and cows will not change and the model can only truly learn these feature characteristics when the animals are photographed in different backgrounds. Therefore the algorithm has found causality, that is, the true features that make a camel a camel.

1.3.1 Machine Learning and Instrumental Variables

Instrumental Variables (IV) is a well-known and widely used technique for causal inference in econometrics. However, IV's can provide imprecise or biased estimates of causal effects and the approach becomes less effective for policy decisions. A key challenge is in the endogeneity of variables when analysing a causal relationship which comes from one or more of a number of sources, such as, *omitted variables bias*, *simultaneity bias*, *sample selection bias* or *measurement errors*. Rutz and Watson (2019) find that mentions of endogeneity and related procedures to correct for it has risen 5 times in the past 20 years. Therefore, there are concerns about endogeneity and applying Instrumental Variables is a natural solution to try to solve these issues.

In order to partition the variance of the endogenous explanatory variable into endogenous and exogenous components, we can use instruments z_i which are correlated with the

endogenous explanatory variable x_i but not correlated with the dependent variable y_i. That is, z_i affects the behaviour of interest x_i and z_i only affects the outcome of interest y_i through its effect on x_i. Instrumental Variables uses the variation in the exogenous component of the explanatory variable caused by the variation in the IV to allow us to make causal inference on the outcome variable.

The first stage of two-stage instrumental variables (IV) is in effect a prediction task. First, we regress x on the instrument z, such as, $x = \gamma' z + \delta$, second we regress y on the fitted values of $\hat{x}$, such as, $y = \beta' \hat{x} + \epsilon$. We say that the first stage is an estimation, however, it is effectively a prediction task since the predictions $\hat{x}$ are applied to the second stage and the coefficients in the first stage are a means to these fitted values. The finite-sample biases in IV are a result of over-fitting, that is, the fitted values $\hat{x}$ pick up the signal $\gamma' z$ but also the noise δ and therefore $\hat{x}$ is biased towards x. The second-stage IV estimate $\hat{\beta}$ is also biased towards the OLS estimate of y on x. Over-fitting is greater when sample sizes are low or the number of instruments are high, or they are weak instruments Bekker (1994) and Staiger and Stock (1994). It is difficult to find strong instruments and often instruments are weakly correlated with the endogenous variables leading to imprecise estimates of causal effects. This problem becomes worse when more weak instruments are added to the IV estimation Hausman et al. (2005). Therefore, the classical approach has a few problems, one when there is a lack of instruments, the other when the instruments are weak and also when there are many potential instruments. In the latter case, the choice of instruments in an IV estimation can be passed to a Machine Learning problem which can use variable selection on the first stage of a 2-stage-least-squares (2SLS) estimation. Therefore finding optimal instruments can be a variable selection problem Bound et al. (1995). Machine Learning can offer a systematic way to try out not only all potential instruments but also non-linear combinations of them.

In Machine Learning held-out validation sets are used to evaluate a model's predictive accuracy and cross-validation is applied to tune hyper-parameters in models. These techniques allow us to be confident that the model is unbiased. Moreover, similar methods are used in finite sample instrumental variables Angrist and Krueger (1995) and "*jackknife*" IV Angrist et al. (1999). Belloni et al. (2012) applied LASSO regression to form first-stage predictions and estimate optimal instruments in linear instrumental variables, even when the number of instruments is larger than the sample size, they show that their results are root-n consistent and asymptotically normal when the first-stage is approximately sparse. Carrasco (2012) applied regularisation of three modified IV estimators based on three different methods of inverting the covariance matrix of the instruments, Tikhonov-RIDGE, Landweber-Fridman and Principal Component Analysis which all involve a regularisation parameter. They find that the three estimators are asymptotically normal and attain the semi-parametric efficiency bound. Hansen and Kozbur (2014) proposed a jackknife instrumental variables estimator (JIVE) with regularisation (RIDGE) at each jackknife iteration which helps to alleviate the bias in the first-stage associated with many weak instruments. They proposed a ridge-regularised JIV estimator (RJIVE) and found that it is consistent and asymptotically normal under the conditions required when there are more instruments than observations. Hartford et al. (2016) shows that the causal effects in the presence of instrumental variables and that flexible IV specification can be solved using Deep Neural Networks. A first-stage network for treatment prediction and a second-stage network whose loss function involves integration over the conditional treatment distribution.

1.4 Parametric vs Non-parametric models

Parametric Algorithms

Machine Learning algorithms that simplify a function to a known form are called parametric algorithms, this simplification can also limit what the algorithm can learn. Regardless of the

amount of data thrown at a parametric model, it will not change its mind about the number of parameters it needs. Moreover, in a parametric model, a functional form for the function is chosen and then the coefficients for that functional form are learned from the training data. If we assume that the functional form is a linear combination of input variables we can estimate the model's coefficients and obtain a predictive model. However, the actual underlying functional form may not be a linear function, it may require transformations of the input data to get the correct functional form. Moreover, the underlying functional form may be nothing like a linear line and therefore, the model's assumptions are wrong and this approach would yield poor results. Examples of parametric Machine Learning algorithms might be, *logistic regression*, *Naïve Bayes*, *Linear Discriminant Analysis*, *perception* and *Simple Neural Networks*.

Parametric tests assume underlying statistical distributions in the data and that the information about the population is known and is used to make inferences about population parameters. The p-value associated with a parametric test will be lower than the p-value associated with a non-parametric equivalent and therefore parametric tests are more able to lead to a rejection of the $H0$. The Null hypothesis is made on parameters of the population distribution.

Non-parametric Algorithms

Machine Learning algorithms that do not make strong assumptions about the underlying functional form are called non-parametric algorithms. Since non-parametric algorithms do not make assumptions, they are therefore free to learn any functional form from the training data, whether that be a linear or non-linear functional form. Moreover, non-parametric algorithms try to find the best mapping function of input x to an output y on the training data whilst also maintaining the ability to generalise on unseen data. Examples of non-parametric Machine Learning algorithms are *k-Nearest Neighbour*, *Supper Vector Machines*

and *Decision Trees*.

Non-parametric tests do not reply on any underlying distribution structure and since fewer conditions of validity apply to non-parametric tests they are more robust than parametric tests and can be used in a broader range of situations. The Null hypothesis is free from parameters.

Advantages and Disadvantages

Parametric models are simpler and easier to interpret. They are also faster and require less training data. Non-parametric models use a flexible number of parameters and are more flexible since they can fit a larger number of functional forms to the data, they are also more powerful since no assumptions about the underlying function are made, they also yield higher performance when it comes to prediction over parametric models.

Parametric models are constrained to the specified functional form chosen and allow for less complexity and thus do not work well on complex problems, they have a fixed number of parameters and are also unlikely to match the underlying mapping function since they make stronger assumptions about the data. Non-parametric models require more training data when learning the mapping function and are thus slower due to having more parameters to train, they are also more prone to over-fitting the training data and it is more difficult to take inference from why specific predictions were made. This thesis focuses on non-parametric models.

1.5 Black-box models, explainability and Interpretability

Organisations are deploying Machine Learning algorithms in many mission-critical situations and therefore in order to trust the model's behaviour, Machine Learning should become in-

telligible, either through inherently interpretable models or through the development of new methods for their explanation. This will allow researchers to detect spurious results and help humans learn from Machine Learning models. The increased need for Machine Learning interpretability is a natural consequence of the increased use of Machine Learning in general. Ensemble models and Deep Learning models were not present in the last hype of Artificial Intelligence (which primarily consisted of expert systems and rule-based approaches). If Artificial Intelligence models are going to continue to affect humans' lives (e.g. in medicine, economics and day-to-day) then, a critical feature for the successful deployment of Artificial Intelligence models this time around involves solving the underlying problem of model explainability. There are notable differences between explainability and interpretability in Machine Learning. Explainability may refer to the procedure taken by a model in order to detail its internal functions. In contrast, interpretability may refer to the level at which humans can make sense of a model. Some Machine Learning models are inherently interpretable such as linear models or decision trees, however, many Machine Learning models are more complex and difficult to understand and model-agnostic interpretations are needed. Model agnostic interpretation simply means that the methods of interpretability can be applied to any Machine Learning model and are applied post-hoc after the model has been trained. These post-host methods try to explain the predictions of the model by treating the models as a black-box and generating explanations without inspecting the internal model parameters. The independence of model agnostic interpretability allows the model to be complex, powerful and flexible. Some model-agnostic explanations such as LIME (Local Interpretable Model-agnostic Explanations) and Shapley values explain how individual predictions are made. That is, model-agnostic models alter the input of the Machine Learning model and measure the changes in predictions. Whereas, other models describe the average behaviours of a model such as permutation feature importance. Counter-factual explanations produce data points as explanations.

Arrieta et al. (2020) discussed different concepts underlying model explainability and interpretability. Weld and Bansal (2019) offers a survey of work relating to the intelligibility of Artificial Intelligence models. Belle and Papantonis (2020) surveys the field of explainable Machine Learning focusing on data-driven methods and pattern recognition models. Dietterich (2017) suggest necessary steps for robust Artificial Intelligence and states that AI is not yet sufficiently robust to support applications in high-stakes applications - such as driving cars autonomously, managing power-grids, trading financial portfolios and controlling autonomous weapons systems. Typically, since Machine Learning models attempt to build a structural form on the underlying data given to them, it has often been difficult to gain inference from what the model has found. Explainable Machine Learning often comes from a second (post-hoc) model which is created in order to explain the first black-box model. This can lead to unreliable and misleading explanations. A more reliable model would be inherently interpretable providing their own explanations without the need for a second model to do the explaining, however, as discussed previously the fundamental objective of Machine Learning is predictive power which comes at a cost of explanation.

1.6 Learning Types

Data can come in two distinct forms, labelled data and unlabelled data with some grey area in between such as semi-labelled data. The input data into any Machine Learning model is called the *training* data. The main difference between *supervised* and *unsupervised* learning lies in the training data, where the dependent variable is observed in *supervised* learning and unobserved in *unsupervised* learning.

1.6.1 Supervised Machine Learning

Supervised Machine Learning primarily focuses on the problem of prediction, the data follows the same structure as typical econometrics data. That is, there is a dependent variable and

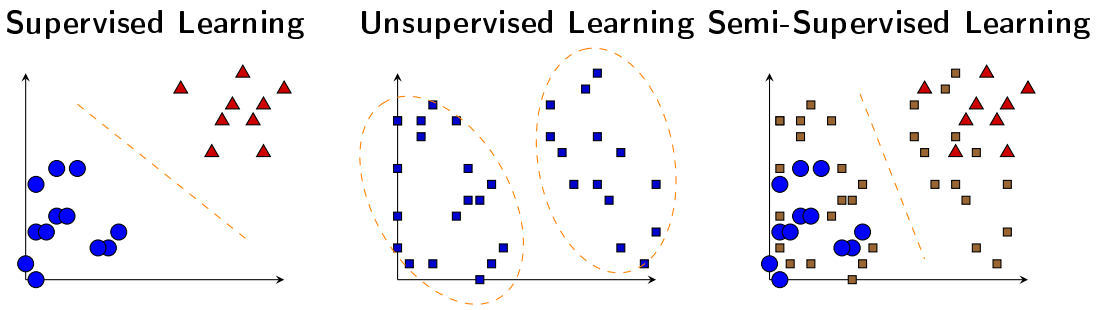


Figure 1.3: Supervised, Unsupervised and Semi-supervised Learning

explanatory variables. The data set is split into a training and testing data set and the objective is to learn a function that maps an input to an output. The model gets to see the output variable in the training data and therefore use the input observation in order to learn this mapping where the joint values of the variables are known. The joint distribution in the training set is the same as the joint distribution in the test set and the observations are assumed to be independent.

1.6.2 Unsupervised Machine Learning

Unsupervised Machine Learning is concerned with learning patterns from unlabelled data, it is therefore given a dataset without any explicit information on what the outcome may look like. Unlike in supervised learning, the objective is to infer the properties of the probability density without the help of a supervisor providing the correct result for each observation. Thus, the model attempts to find structure through the relationships between covariates. Unsupervised learning allows researchers to ask the algorithm questions to which they may not already know the answer to. A popular example of unsupervised learning refers to the classification of images. If an algorithm is given images of cats and dogs without explicitly being labelled as a cat and dog it can learn features of each animal in order to distinguish between the two, the model is not told that a given picture is a cat or a dog, it learns the structural differences and outputs two classifications.

1.6.3 Semi-supervised Machine Learning

Semi-Supervised Machine Learning combines the use of supervised and unsupervised learning in which the data may contain a small amount of labelled data and a large amount of unlabelled data.

Semi-supervised learning may involve either *transductive* or *inductive* learning. is the reasoning from observed, specific training data to specific test data, that is, the goal is to infer the correct labels for the given unlabelled data. learning observes all of the training and testing data, it learns from the already observed training data and then predicts the labels of the test dataset, despite not knowing the labels in the test data, we make use of the structure and patterns present in the test data during the learning process. This approach predicts the labels of unlabelled data using the knowledge of the labelled data. On the other hand, induction is the reasoning from observed training data to general rules, which are then applied to the test data. The difference between the two is that during training, transduction encounters both the training and testing datasets, whereas induction encounters only the training data. Moreover, transduction does not build a predictive model and if a new observation is added to the test data, the model will need to be re-trained in order to predict the labels. In contrast, induction builds a predictive model and therefore when new observations are added to the test dataset, we do not need to re-build the model. Consider the following examples. Suppose we have 100,000 images and we want to classify whether an image has a car in it or not. We know that 1,000 images contain a car and 1,000 images do not contain a car, which is our labelled data, the remaining 98,000 we do not know whether they contain a car or not and are therefore unlabelled. Inductive learning will take the 2,000 labelled data and build a classifier based on this data and then try to predict the remaining 98,000 observations. Transductive learning tries to use the information in the remaining 98,000 observations to tell it something about the problem space.

Consider the Semi-Supervised Learning example in Figure 1.3, the classifier has made a decision classifier that perfectly separates the two classes. Moreover, it has also used the unlabelled data points (as given by the *squares*) when making its decision rule based on their proximity to the labelled data points (given by their *circles* and *triangles*).

1.6.4 Reinforcement Machine Learning

Reinforcement Machine Learning is learning through the interaction with an environment. The agent learns from the consequences of its actions, as opposed to being explicitly taught as in previous types of learning. The agent seeks to choose its actions based on its past experiences which maximise the cumulative reward over time. It is fundamentally different to supervised learning in not needing labelled input and output data. The focus of reinforcement learning is on finding a balance between exploitation (agents past experiences) and exploration (agents new choices). Many reinforcement learning models use *dynamic programming* techniques since it aims to learn the optimal behaviours through trial and error interactions with a dynamic environment. In economics and game theory reinforcement learning can be used to explain how equilibrium may arise under bounded rationality.[7] Reinforcement learning has been a traditional way in economics to model dynamics of learning under limited or bounded rationality and topics such as operational research has a vast amount of literature on the trade-off between exploration and exploitation. Kaelbling et al. (1996) provides a survey on reinforcement learning. Reinforcement learning is a type of sequential experimentation and is fundamentally about causality. That is, moving a piece on a chessboard to a new position *causes* the probability to win or lose to change and thus the state of the environment has changed.[8]

[7] *Bounded rationality* is the idea that rationality is bounded because there are limits to our thinking capacity, available information and time and that decision-makers seek a satisfactory solution rather than an optimal one.

[8] A more formal description: Consider an agent - or decision-maker - which interacts with an environment over a sequence of observations, it seeks to maximise the reward over the period. Formally, the model has a finite set of environment states S, a finite set of agent actions A and a set of scalar reinforcement signals (i.e. rewards) R. At each step or iteration i, the agent observes the environment's state $s_i \in S$. The agent chooses an action $a_i \in A(s_i)$, where $A(s_i) \subseteq A$ denotes the set of actions available in state s_i. The

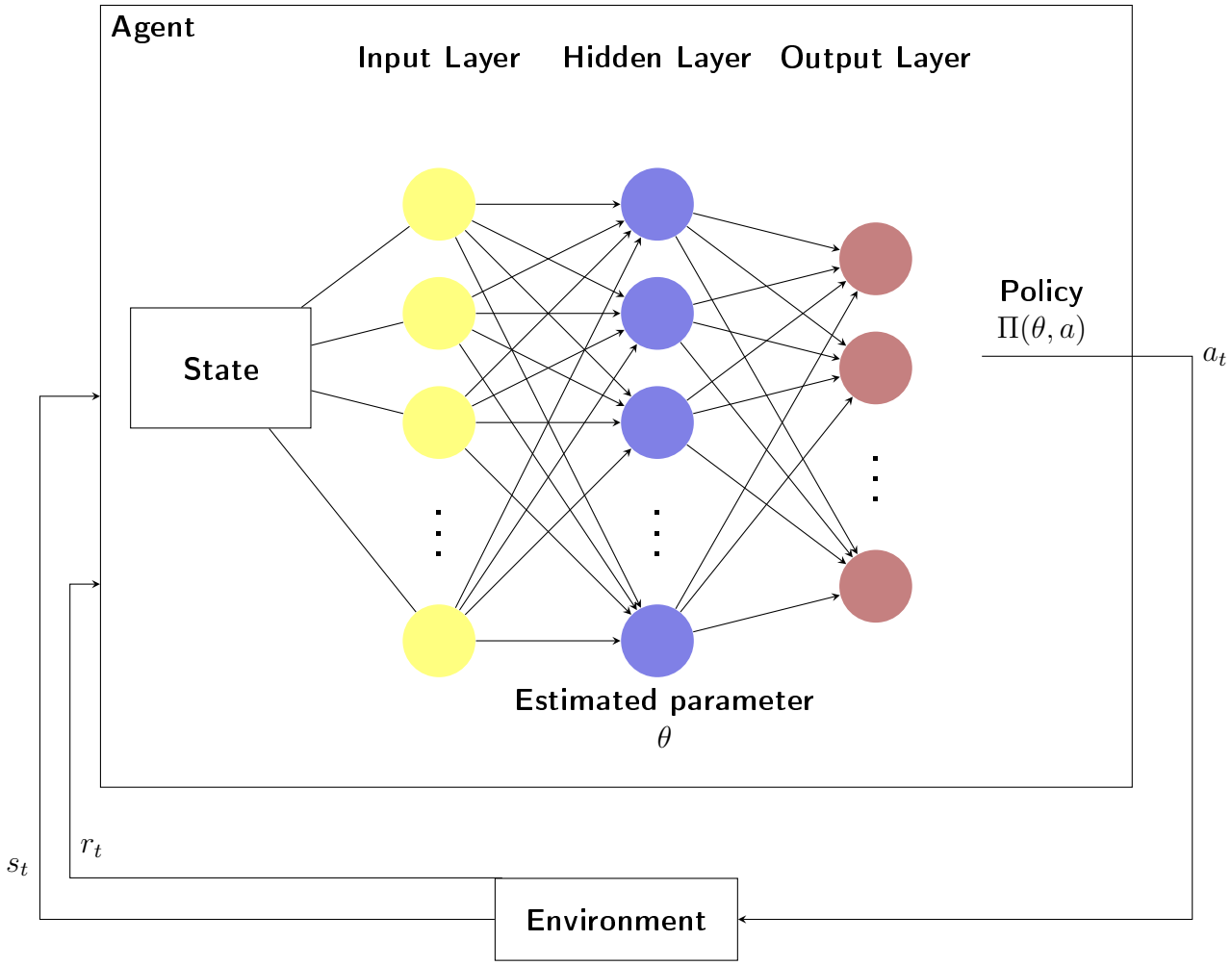


Figure 1.4: Reinforcement learning flow diagram

1.7 Bias and Variance

The *bias-variance* trade-off concerns model complexity, under-fitting and over-fitting. Under-fitting occurs when a model was unable to capture the underlying pattern in the data. Such

agent obtains a reward $r_{i+1} \in R$ at each iteration i and observes a new state s_{i+1}. A state-action function is determined $Q(s_i, a_i)$ which defines an expected value of each possible action a_i in each state s_i. If the state-action $Q(s_i, a_i)$ is known, then the optimal policy is $\pi^*(s_i, a_i)$ given by the action a_i which maximises the state-action $Q(s_i, a_i)$ given the state s_i. Moreover, the agent wants to maximise the expected reward through learning the optimal policy function $\pi^*(s_i, a_i)$ where π is the probability of taking action a if the state of the environment is s.

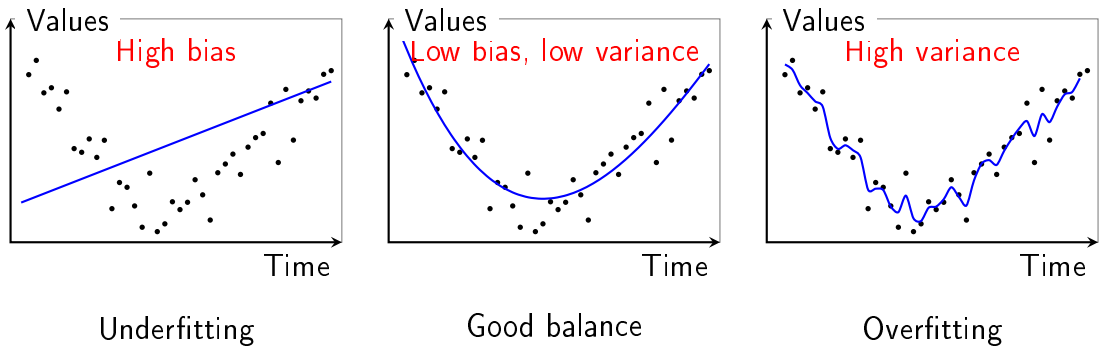


Figure 1.5: Bias-Variance Trade Off

models usually have a high bias and a low variance. Under-fitting can occur when trying to fit a linear model to a non-linear dataset. Over-fitting occurs when a model captures the noise along with the underlying pattern in the data. Such models usually have a low bias and a high variance. Over-fitting can occur when using decision trees since trees can grow very large and become very complex.

Consider the *underfitting* panel in Figure 1.5. Simply fitting a linear regression line through the data is not going to capture the true relationship and no matter how well we try to fit the line it will never curve. The bias occurs due to the inability of the model to capture the true relationship. Next, consider the *overfitting* panel in Figure 1.5 which fits the true relationship well and thus it has very little bias. Taking the sum of squares for each model we will see that the *overfitting* panel will have a value close to zero. However, once we introduce new unseen data and recalculate the sum of squares, the linear regression model will have the lower sum of squares. The difference in fits between the old data and the new data is the *variance*. Therefore, the *overfitting* panel has a low bias since it is flexible in adapting to the data but it will also have a high variance since it results in vastly different sums of squares for different datasets and therefore it has over-fit the dataset it was modelled on. The linear model has a high bias since it cannot capture the curve in the data but it will have a relatively low variance since the sums of squares are expected to be

similar for different datasets. The ideal model will have a low bias and a low variance such that it will continue to produce consistent predictions across different datasets. In order to find the optimal model *regularisation*, *bagging* and *boosting* can be used, which is discussed in more detail later. The error can be decomposed as:

$$Err(x) = \text{Bias}^2 + \text{Variance} + \text{Irreducible Error}$$

Where, the irreducible error is the amount of noise in the data and no matter how good the model is, there will be a certain amount of noise that can not be reduced.

Consider Figure 1.6 where the red bullseye represents a model which perfectly predicts correct values. The further out from the centre the worse the model's predictions. Models which under-fit the data have *high bias* and *low variance* and usually fail to capture the underlying structure of the data as in the case when fitting a linear model on non-linear data as depicted in Figure 1.5. Models which over-fit the data have *low bias* and *high variance* and the model captures noise along with the underlying structure in the data as in the case when fitting complex decision trees which are prone to over-fitting.

In the context of econometrics, the objective is to obtain unbiased and consistent estimates of coefficients whereas in the context of Machine Learning the objective is to obtain precise and unbiased predictions through controlling the trade-off between bias and variance. Moreover, a model which is unbiased in its prediction may not necessarily be unbiased in its coefficients. If a model has too few parameters and is too simple then it will have *high bias* and *low variance*. Moreover, if the model has a large number of parameters and is too complex then it is likely to have *high variance* and *low bias*. Therefore, it is important to find the right balance between bias and variance in order to control the model complexity. Figure 1.7 illustrates the bias-variance trade-off and their relationship with model complexity on the error rate.

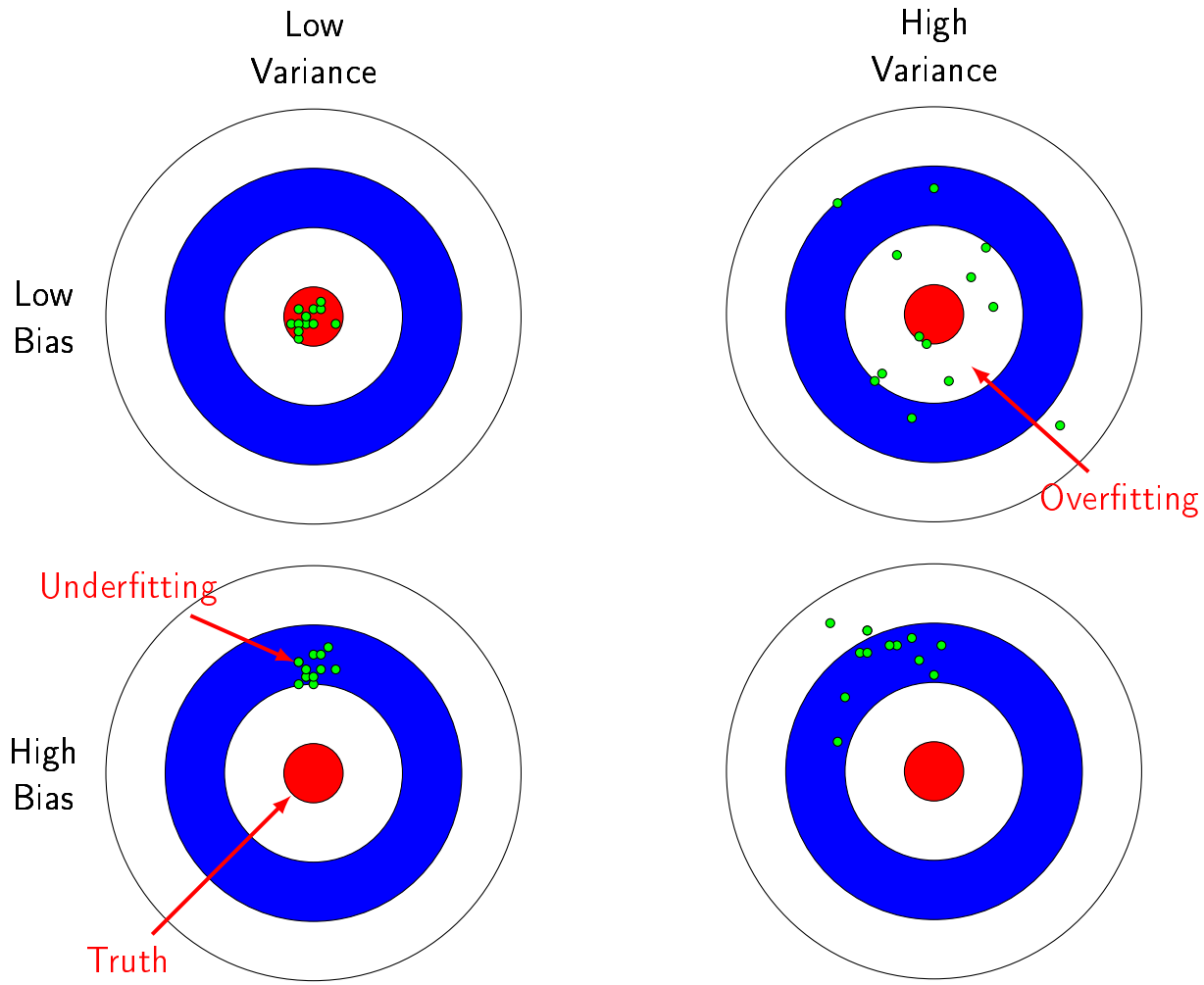


Figure 1.6: Graphical illustration of bias and variance

1.8 Ensemble Methods

Typical Machine Learning modes might take a single learner such as a Logistic Regression, Decision Tree, Support Vector Machine, or Artificial Neural Network, provide it with some data and then train the model. Ensemble models expand on single learner models by combining the learners in order to try and enhance the performance over any single individual learner. That is, instead of having a single learner the ensemble model consists of many learners combined together in order to obtain a stronger overall model, Schapire (1990). The idea behind ensemble models is that each ensemble or learner must be weak and may

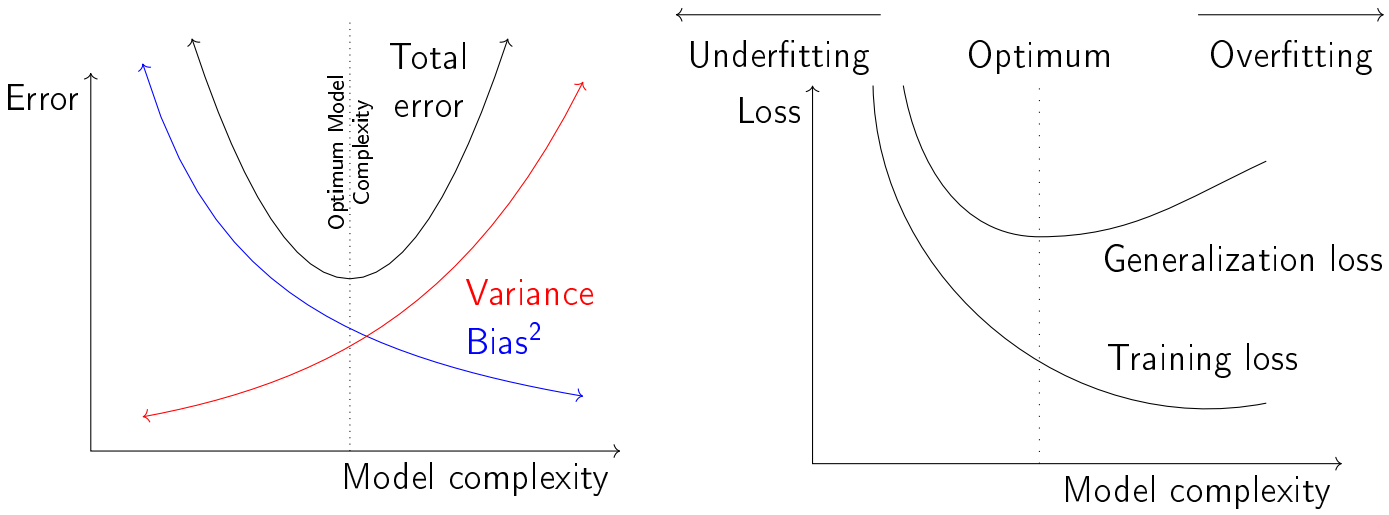


Figure 1.7: Model Complexity and Bias/Variance Trade-off

only slightly achieve better than a random guess. If the learners were strong then this would be a clear indication that the learner has over-fit the data and there will be little or no residual errors to be passed onto subsequent learners in order to build or learn upon. Originally Bayesian averaging was the main ensemble method but other meta-learning algorithms such as *bagging* and *boosting* have proved to be strong ensemble methods Dietterich (2000a). Generally, ensemble methods reduce bias and variance which in turn help increase the stability and performance by eliminating the dependency of a single estimator.

Take 3 individual models - weak learners - given as h_1, h_2, h_3 and a new observation x. If the three models are identical then they will have the same errors on the new data point such that when $h_1(x)$ is wrong, $h_2(x)$ and $h_3(x)$ is also wrong. However, the errors of each of the individual classifiers may be uncorrelated such that when $h_1(x)$ is wrong, $h_2(x)$ and $h_3(x)$ may be correct and we may accurately predict x. There are three fundamental reasons why it is possible to build strong ensemble models.

The first being *statistical*. Consider a learning algorithm that searches over a space $\mathcal{H}$ of hypotheses in order to identify the best hypothesis in the space. If the training data is

small compared to the hypotheses space then the learning algorithm can find many different hypotheses in $\mathcal{H}$ which would give the same accuracy on the training data. Consider the panel Statistical in Figure 1.8, the hypothesis space $\mathcal{H}$ is depicted as the outer curve whereas the inner curve depicts a set of hypotheses which all give good accuracy on the training data. The point f represents the true hypothesis and selecting any one of the individual hypotheses h_i leads to an incorrect hypothesis, however, averaging the hypotheses we can come up with a good approximation to f which is better than any individual h_i alone.

The second being *computational*. Learning algorithms often perform a localised search and therefore the model may become stuck in a local optima. Neural Networks use gradient descent to minimise an error function whereas decision tree models use greedy splitting rules to grow the decision trees. An ensemble that is constructed by running the local search from a number of different random starting points may be able to provide a better approximation to the unknown function f as opposed to a single algorithm having only 1 starting point.

The third being *representational*. The true function f can not always be represented by any of the hypotheses in $\mathcal{H}$. Forming a weighted sum of the hypotheses drawn from $\mathcal{H}$, it may be possible to expand the space of representable functions Dietterich (2000a). Ensemble methods help to reduce these three shortcomings of standard statistical learning algorithms.

1.8.1 Bagging

Bagging (Bootstrap Aggregating) Breiman (1996a) considers a set of homogeneous *weak learners* in which each weak learner is learned independently from each other. In the end, each weak learner is aggregated or a majority vote is taken of all of the ensembles in order to obtain a final prediction. For example, a single decision tree can be unreliable, since when the data is changed slightly, the new decision tree can be quite different from other

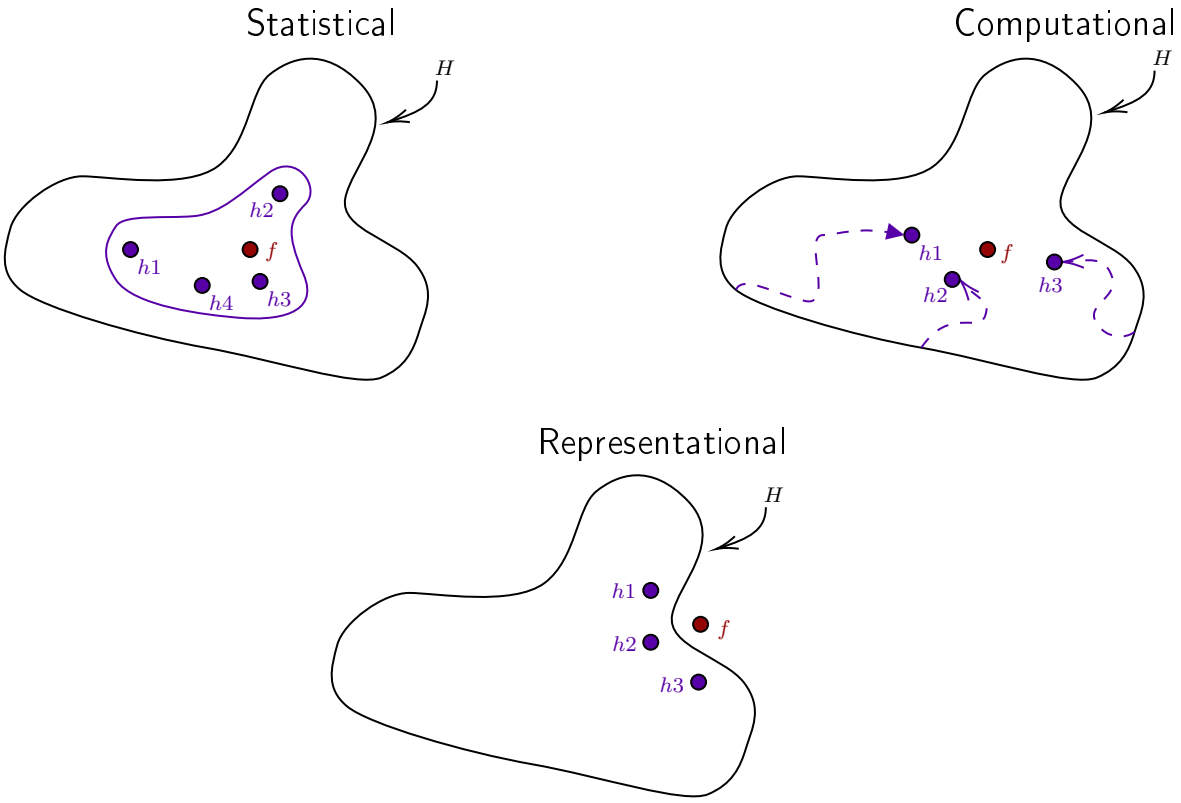


Figure 1.8: Fundamental reasons why it is necessary to build strong ensemble models, *statistical*, *computational* and *representational*

decision trees. These differences introduce high variance between individual trees, however, averaging across the trees will yield a more stable prediction Dietterich (2000b). Contrast that to the situation when weak learners are stable (high bias) and have low variation, the combination of these weak learners will form a weaker predictive model since each of the weak learners will be similar to each other. The idea is to take at random (and with replacement) a subset of the training data and build a weak learning model and repeat this n times, combining all the weak learning models in order to obtain a final predictive model. These models are learned independently and in parallel to each other and each weak learner will over-fit its sample of the data therefore introducing high variance across sampled datasets. For regression, the model fits the same regression tree to the bootstrap sampled versions of the training data, the result is then averaged to form a prediction. For classification, a

committee of trees each cast a vote and the majority vote forms the predicted class. An ensemble of randomised decision trees is also known as a Random Forest model which is discussed in more detail later.

1.8.2 Boosting

Boosting considers a set of homogeneous *weak learners* and learns them sequentially in an adaptive way with each subsequent weak learner added to the ensemble trying to correct the errors of its predecessor, therefore unlike *bagging*, the weak learners evolve as the ensemble grows. A remarkably rich theory has evolved around boosting, with connections to a wide range of topics including statistics, game theory, convex optimisation and information geometry Schapire and Freund (2013). *Boosting* Schapire (1996), Schapire et al. (1998), Friedman (2001), Friedman (2002) helps to reduce variance through using multiple models and it also reduces the bias by passing the errors of previous models to subsequent models. Rather than growing each tree independently, the algorithm attempts to improve or *boost* at each step in the model from the previous step. The main difference between *bagging* and *boosting* is that the weak learners in *bagging* are trained in parallel using randomness, whereas in *boosting* the weak learners are trained sequentially which allows for sequential weighting. A boosting model (see AdaBoost, LightGBM, XGBoost etc.) is trained with all of the observations starting with the same weights which are used to train an individual weak learner - e.g. a decision tree. Once the weak learner has been built, a prediction error is calculated, which increases the weights of the observations which have a bigger error and therefore makes them more important in training the subsequent weak learner. Each individual weak learner is also assigned an importance / weighted score and receives a higher weight if it did a good job at predicting the sample of observations. Thus a model which provides very good predictions will have a high amount of say in the final decision. The weighted observations are passed to the posterior weak learner and the process is repeated until n learners are reached or a criterion is met for fitting the data well enough.

1.8.3 Stacked Models

Stacking considers a set of homogeneous *base learners*, previously *bagging* and *boosting* considered the set of *weak learners* and these weak learners formed a model which is referred to as a base learner. Stacking Wolpert (1992) and Breiman (1996b) involves combining the predictions of the base learners in order to build a *super learner*. An example of a base learner might be a single Support Vector Machine, Random Forest or Gradient Boosted model. Therefore stacking combines Machine Learning models in order to obtain a final generalised model. Stacking can outperform the individual base learners and the super learners can learn an optimal combination of the base learners predictions Van der Laan et al. (2007). Stacking machine learning models usually performs best when stacking base learners that have high variability, and uncorrelated predicted values, similar to that in bagging. Unlike bagging, the models are typically different (e.g. not all decision trees) and are fit on the same dataset (e.g. instead of samples of the training dataset). Unlike boosting, the model combines the predictions from each of the models (e.g. as opposed to a sequence of models which correct the predictions of prior models).

1.9 Econometrics, Machine Learning and Regularisation

Much of econometrics is not optimised for prediction tasks since the focus is on unbiased and consistent estimates, OLS is only the best linear unbiased estimator and minimises the in-sample error which can lead to poor predictions out-of-sample. Suppose we want to select a function $\hat{f} \in \mathcal{F}$ to predict y of a set of new observations. The objective is to minimise the loss function $(y - \hat{f}(x))^2$ and OLS will select the line which minimises this function on the in-sample data. Moreover, prediction tasks are more concerned with doing well out-of-sample and since f varies between each sample of the data, it produces variance. Machine Learning tries to handle the bias-variance trade-off such that predictions are maximised and are then

generalisable to out-of-sample data. Therefore it not only tries to minimise in-sample error but also adds a regularisation parameter that penalises functions that creates variance and in doing so aims to build a model which can be generalised onto unseen data.

More specifically, Machine Learning aims to minimise;

$$\hat{f}_{ML} = arg\,min_{f \in \mathcal{F}} \sum_{i=1}^{n} (y^i - f(x^i))^2 + \lambda \mathcal{R}(f) \tag{1.1}$$

where the OLS properties are just concerned with the first term. The addition is the regularisation term $\mathcal{R}(f)$. The λ parameter allows us to control the trade-off between bias and variance. The value of λ is determined on the in-sample training data.

The addition of regularisation and the choice of the regularisation method permits a model to take in more variables than observations. LASSO and RIDGE[9] regression automatically penalise variables that contribute very little to the regression model, setting the coefficients of these variables downwards towards zero. The regularisation terms for LASSO and RIDGE are $\mathcal{R}(f_\beta) = |\beta|$ and $\mathcal{R}(f_\beta) = \beta^2$, respectively. Moreover, the addition of regularisation allows a model to take on more flexible functional forms, including higher-order interaction terms, i.e. by construction, decision trees allow for a high degree of interactivity. However, optimising for $\hat{y}$ does not produce very useful β's.

1.10 Cross-validation

Re-sampling is the process of sampling more than once from a dataset and then re-fitting a model using this newly sampled data. The most widely used are bootstrap and cross-validation. Since Machine Learning models require that the trade-off between bias and variance be satisfied such that the model does not over-fit the data, a common practice

[9] See section 1.19.3 in the Appendix for the definition of RIDGE and LASSO.

in supervised learning problems is to hold out part of the data as a *testing* dataset. The moment we apply our model evaluation on the test data is the moment we introduce look-ahead bias into our model since we might be tempted to adjust the parameters of the model to give more satisfactory test results.

Ideally, the approach would be to split the data into three randomly divided parts *training*, *validation* and *test* sets. The *training* set is used when fitting the model, the *validation* set is used when estimating the prediction error from the model selection. From these two splits of the data, we estimate the performance for a number of different models in order to see which model performs the best. Finally, the *test* set is then used at the end to assess the generalisation error of the chosen model - which was selected at the *training* and *validation* stage.

1.10.1 k-fold Cross-Validation

In practice there is usually not enough data in order to simply split between *training*, *validation* and *testing* datasets. In order to select the best model and search for the most optimal parameters we can randomly shuffle the data and then split the *training* data up into K equal sized validation parts or *k-folds*. Suppose we split the *training* and *validation* data into *10-folds* as depicted in Figure 1.9. The model is trained on the section indicated by the white space and validated on the grey space in each partition.

More formally, the kth part is the shaded region and the model is fit to $K-1$ segments of the data, the prediction error of the fitted model is calculated on the kth segment. This is done for $k = 1, 2, \ldots, K$ segments and the prediction error is calculated for each segment. An indexing function $k : \{1, \ldots, N\} \mapsto \{1, \ldots, K\}$ indicates a partition in which observation i is allocated through randomisation. The cross-validation estimate is given as;

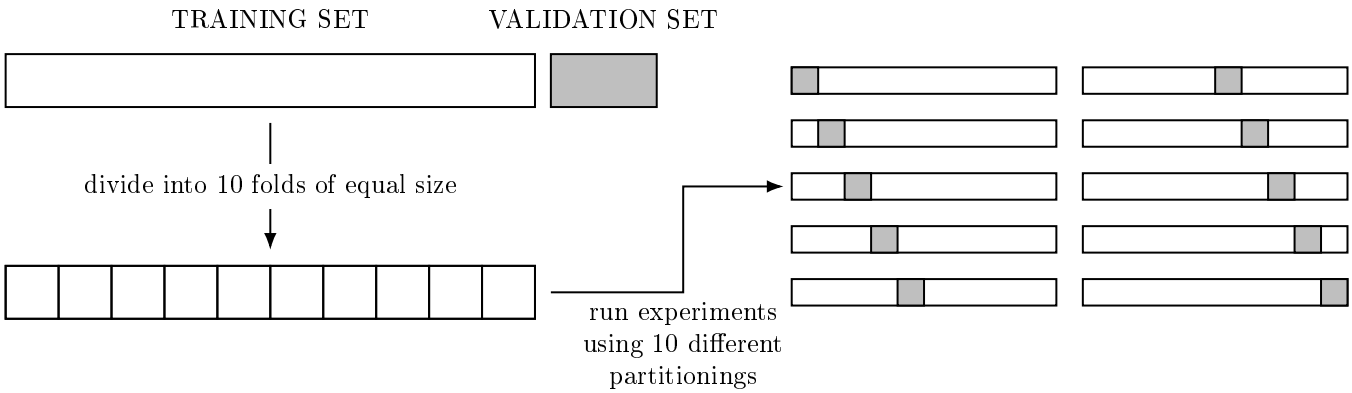


Figure 1.9: K-fold cross-validation where K = 10

$$CV(\hat{f}) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, \hat{f}^{-k(i)}(x_i)) \tag{1.2}$$

where $\hat{f}^{-k}(x)$ is the fitted function. When $K = N$ the cross-validation methodology becomes *leave-one-out* cross-validation and is useful when the dataset is small. That is, using the observation $k(i) = i$, the fit is computed using all observations except the ith. *Leave-one-out* cross-validation has a low bias but it can also have a high variance. It is usual to use *5-fold* or *10-fold* cross-validation as recommended Breiman and Spector (1992) and Kohavi et al. (1995).

1.10.2 Stratified cross-validation

Bootstrapping simulates the effect of drawing a new sample from a population, however, it does not ensure distinct test samples. K-Fold cross-validation ensures that there are K distinct test folds, however, it is repeated R times for different random partitioning this allows independence assumptions to hold for K-fold cross-validation, but this is lost with repetition.

Stratified cross-validation splits the data into K folds ensuring that each fold is an appropriate representation of the original data through class distribution, mean, variance

etc. Stratification address the defects in the classification algorithms since they can be easily biased by over -or- under-representation of classes. Suppose we have a sample of 100 observations, 90 in class $\mathcal{A}$ and 10 in class $\mathcal{B}$. The dataset is said to be unbalanced and sub-sampling using randomised groups may lead to constructing models on very few - or even zero - observations from the $\mathcal{B}$ class. Therefore the model would be unable to learn or predict the $\mathcal{B}$ class. Stratified cross-validation allows the randomisation whilst taking into account these unbalanced classes ensuring that each fold contains a representative distribution of the classes.

1.10.3 Bootstrap

The *bootstrap* Efron (1992) method is a general tool for assessing statistical accuracy and it is a re-sampling technique that involves iteratively re-sampling a dataset with replacement. Samples are taken from a larger data sample and some statistics are applied on each sample, the data is then returned to the larger sample after it has been used. This allows observations to be included in the smaller samples more than once and is therefore *sampling with replacement*. Bootstrapping can be used to quantify the uncertainty associated with a given estimator or statistical learning method. In Machine Learning, this is done through training a model on the sampled data and then evaluating the model's performance on those samples not included in the sample the model trained on. From here, additional statistics can be summarised, measures of variance can be taken, such as standard deviation and standard error. Moreover, a confidence interval can be calculated to show the bounded estimates. These statistics are useful when determining the model's performance across different samples of the data which should be generalised onto a held-out test set. Since bootstrapped data are drawn with replacement, a bootstrapped dataset may contain multiple instances of the same original data and may omit completely other data points. cross-validation re-samples without replacement and therefore ensures all of the data points are used across the splits.

1.11 Decision tree models

A decision tree is a supervised learning model used for both classification and regression problems and was popularised by Breiman et al. (1984). Tree-based models partition the feature space into a set of homogeneous rectangles and then fit a simple model in each one. Decision trees are non-linear models made up of piecewise linear components in each neighbourhood. It is useful in classifying non-linearly separable data. That is, they are non-parametric and therefore are not subject to normality assumptions of the data, they are also able to handle different data types such as categorical and continuous variables. Transformations of the variables are also not a requirement and it can therefore be useful in identifying outliers, variable interactions and important variables. Morgan and Sonquist (1963) came up with one of the earliest concepts of a regression tree, the Automatic Interaction Detection (AID) which recursively splits data depending on impurity and stops splitting when a certain level of impurity is reached. Messenger and Mandell (1972) extended this idea to classification problems and developed THeta Automatic Interaction Detection (THAID) which recursively splits the data in order to maximise the number of observations in each modal category. Kass (1980) developed the Chi-square Automatic Interaction Detection (CHAID) originally developed for classification and then extended to incorporate regression problems. Breiman et al. (1984) improved on AID and THAID and developed the Classification and Regression Tree (CART) which improved accuracy through instead of using stopping rules, it grows the tree and then prunes it to a size that has the lowest cross-validation estimate of error. The Iterative Dichotomiser 3 (ID3) Quinlan (1986) was developed which uses information entropy in order to quantify impurity and is used to compute the gain ratio for binomial decision classifiers. Quinlan (2014) developed the successor to the ID3 algorithm which can handle multi-class problems and laid the groundwork for further developments. Loh (2014) provides a comprehensive literature review of the main developments in decision tree models over the past 50 years.

Decision trees are simple and interpretable which involves the linear partitioning of the predictor space into subsets. Decision trees consists of *nodes* which can be defined as a *root nodes*, *inner nodes* and *end nodes/leaves*. *Edges* represent the decision taken from the previous node. The decision tree begins at the *root node* and therefore has no incoming edges. The tree continues down and the *inner nodes* contain exactly one incoming edge and has at least two outgoing edges. *Leaf nodes* contain the solution to the decision tree problem and contain exactly one incoming edge and no outgoing edges. Each node of a decision tree represents an attribute, each branch represents a decision rule and each leaf represents an outcome that can either be categorical or continuous. The objective is to minimise the error in each leaf and the model needs to learn the mapping between the input vector variables and the predictor variables. Moreover, the decision tree requires a *split criterion* which computes a value for all variables, this value corresponds to the amount of information that is gained from the split using this variable. The optimal value from all of the variables is selected and then the *node* is split into new outcomes from this respective variable. This process is carried out recursively at each generated sub-tree until a *stop criterion* is reached. The *stop criteria* could be setting the maximum number of branches for the tree and when the tree reaches this limit, it terminates. Additionally, it could be when the maximum number of observations in a given node is less than some threshold value or if the optimal split information gain does not surpass a given threshold value. Algorithm 1 shows the pseudo-code for training a decision tree model.

Algorithm 1: Decision Tree Training Algorithm

```
Initialization;
Training set = S;
Attribute set = A;
Target Attribute = C;
Split Criterion = sC;
Stop Criterion = Stop;
Objective = Grow(S, A, C, sC, Stop);
if Stop(S) = false then
    forall a_i ∈ A do
        find a_i with the best sC(S);
    end
    label current Node with a;
    forall values v_i ∈ a do
        label outgoing edge with v_i
        Ssub = S where a = v_i;
        create subNode =
          Grow(Ssub, A, C, sC, stop);
    end
else
    currentNode = leaf;
    label currentNode with c_i where c_i is the
      most common value of C ∈ S;
end
```

1.11.1 Regression Trees

Regression trees can fit almost any type of statistical model, such as least-squares, logistic, quantile, Poisson along with models for longitudinal and multi-response data. Given a sample of (x_i, y_i) of size I, for $i = 1, 2, \ldots, N$. The algorithm needs to automatically decide on the splitting variables and split points which minimise the total variation of y_i inside the two child clusters which need not have the same size. That is, in the first step, it searches, for each variable x_i the optimal splitting point, which corresponds to the first **forall** in the pseudo-code for algorithm 1. In the second step, it selects the variable which achieves the highest level of homogeneity in y_i which corresponds to the second **forall** in algorithm 1.

Firstly, the model finds the best split for each variable by solving $\underset{c^{(k)}}{\operatorname{argmin}}\, V_I^{(k)}(c^{(k)})$ with

$$V_I^{(k)}(c^{(k)}) = \underbrace{\sum_{x_i^{(k)} < c^{(k)}} \left(y_i - m_I^{k,-}(c^{(k)})\right)^2}_{\text{Total dispersion of first cluster}} + \underbrace{\sum_{x_i^{(k)} > c^{(k)}} \left(y_i - m_I^{k,+}(c^{(k)})\right)^2}_{\text{Total dispersion of second cluster}}, \tag{1.3}$$

where

$$m_I^{k,-}(c^{(k)}) = \sum_{\{x_i^{(k)} < c^{(k)}\}} y_i \frac{1}{\#\{i, x_i^{(k)} < c^{(k)}\}} \quad \text{and}$$
$$m_I^{k,+}(c^{(k)}) = \sum_{\{x_i^{(k)} > c^{(k)}\}} y_i \frac{1}{\#\{i, x_i^{(k)} > c^{(k)}\}}$$

are the average values of Y, which is conditional on X^k being smaller or larger than c. For variable k, the optimal split satisfies $c^{k,*} = \underset{c^{(k)}}{\operatorname{argmin}}\, V_I^{(k)}(c^{(k)})$ and thus $c^{k,*}$ is therefore the split with the smallest total dispersion not only over all splits, but also over all variables $k^* = \underset{k}{\operatorname{argmin}}\, V_I^{(k)}(c^{k,*})$. The cardinal function $\#\{\cdot\}$ simply counts the instances of its argument, therefore the best c is just the average y_i in the partition space. Having found the best split, we partition the data into two resulting regions and repeat the splitting process on each of the two regions.

The question remains, how large should we grow the tree? A large tree will over-fit the data whereas a small tree might miss important structure. One solution might be to split the tree only if the model's decrease in sum-of-squares due to the split exceeds a given threshold. However, a seemingly worthless split may have a very good split below it.

1.11.2 Classification Trees

When the target Y variable is a classification outcome, the only difference is the measure of dispersion or heterogeneity which pertain to the criteria for splitting nodes and pruning the tree. In *regression trees* the loss function was the squared-error impurity measure, where the same cannot be used for classification. The target $\hat{y}_i$ contains as many elements as categories in the label where the elements correspond to the probability that a given observation belongs in that category. The classification algorithm seeks *purity* through searching for a split criterion which will result in clusters that are as pure as possible, that is, find clusters with a single - *or as few as possible* - dominant classes. The loss penalises the outputs which do not concentrate on a single class, i.e. a classification of the following (0.20, 0.30,

0.30, 0.20) is more difficult than a classification of (0.80, 0.05, 0.05, 0.10) since many of the classifications will fall into the 0.80 category.

For J classes, if p_j is the proportion assigned to class j then for each leaf the loss functions are;

$$\text{Misclassification error} = 1 - \max_j p_j \tag{1.4}$$

$$\text{Gini impurity index} = 1 - \sum_{j=1}^{J} p_j^2 \tag{1.5}$$

$$\text{Cross-Entropy or deviance} = H(T) = -\sum_{j=1}^{J} \log(p_j) p_j \tag{1.6}$$

where $p_1, p_2, \ldots, p_J$ sum up to 1 and represent the percentage of each class present in the child node, whereas J represents the number of classes for this feature.

All three are similar, however *cross-entropy* and the *gini index* are differentiable and are used more in numerical optimisation. *Information gain* is defined as;

$$\overbrace{IG(T,a)}^{\text{Information Gain}} = \overbrace{H(T)}^{\text{Entropy / Gini (parent)}} - \overbrace{H(T|a)}^{\text{Weighted Sum of Entropy / Gini (Children)}} \tag{1.7}$$

$$IG(T,a) = Entropy(T) - \sum_a p(a) Entropy(T|a) \tag{1.8}$$

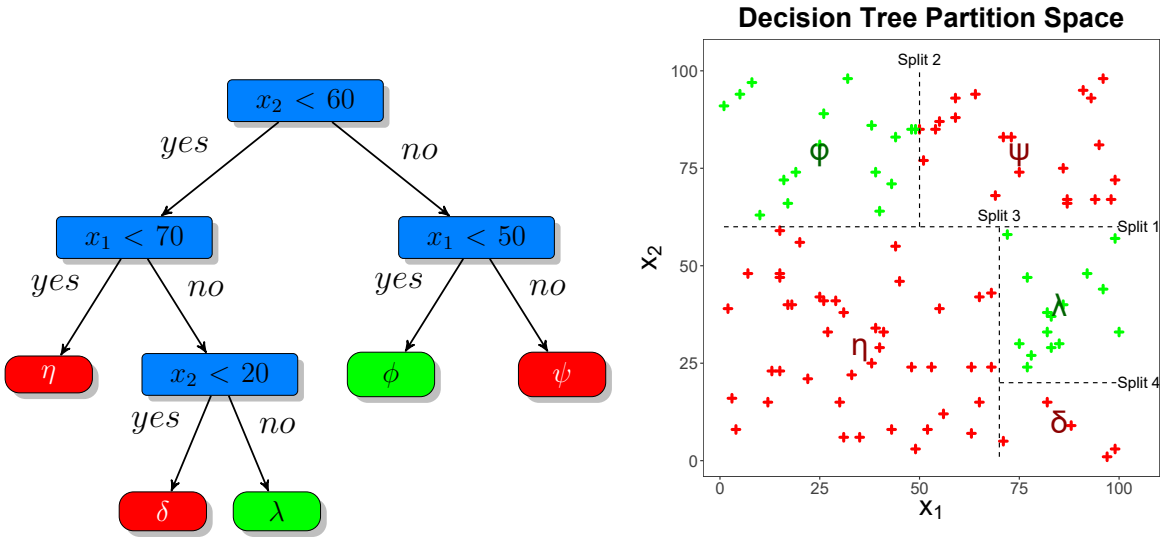
1.11.3 Tree Pruning

It is possible to grow trees to form very deep trees where all observations belong to separate leaves which perfectly fits the training data. Moreover, when it comes to unseen testing data the performance will be significantly different and therefore the tree has over-fit the data. The most reliable sections of trees are those branches closest to the *root* node since they used a large number of observations to make the splits, as the branches get deeper

and deeper the newer branches are using fewer and fewer observations to make their split. Therefore the first few splits are the splits that matter the most since they determine the most general patterns and structure in the data.

In order to correct for over-fitting, a method of *pruning* is applied with the objective of removing non-productive branches of the tree. It is possible to impose a minimum number of observations that each *leaf* contains ensuring that each terminal node contains a sufficient number of observations. Additionally, a threshold can be set such that if a split does not reduce the loss by a set threshold, no further splits will be made. Alternatively, the depth of the tree can have a hard limit in which the maximum number of splits from the *root* to *leaf* are pre-determined. Through *pruning* the tree, we remove unnecessary structure which helps to reduce the tree complexity and makes the tree easier to interpret.

Figure 1.10 shows two variables X_1 and X_2. The decision tree is plotted on the left and the corresponding partition splits are shown on the right. Beginning at the top of the decision tree on the left, if $X_2 < 60$ or *Yes* then a simple straight line is drawn on the corresponding scatter plot at the point X_2, denoted as *Split 1*. Moving down the decision tree, if $X_1 < 70$ we reach a terminal node denoted as *red*. This is denoted as *Split 3* on the partition plot and thus all points which lie between $X_2 < 60$ and $X_1 < 70$ are shaded in *red*. Alternatively, if the path to the terminal node was the following $X_2 < 60$ and $X_1 \not< 70$ then there is an additional decision split at $X_2 < 20$. If $X_2 < 20$ is true then the points would lie in the region of $X_1 > 70$ and $X_2 < 20$ on the corresponding partition plot, coloured in *red*. The *green* points would lie in the region between *Split 3*, *Split 1* and *Split 4* which follows the path $X_2 < 60$, $X_1 \not< 70$, $X_2 \not< 20$. The path on the right side of the decision tree follows the same splitting structure.



(a) Decision Tree (b) Partition Space

Figure 1.10: Decision tree and corresponding scatter plot partition space

1.12 Random Forest models

As discussed previously, decision trees often over-fit the training data and therefore it fits to the particular details of the data as opposed to the overall properties of the distributions they are drawn from. If we trained two completely different trees on two random samples of the data set we are going to obtain different decision boundaries in the region where the decision trees are least certain but combining the information from both of the trees might yield a more robust result and subsequent decision boundaries. Extending this to multiple trees is what the *Random Forest* model does.

A *Random Forest* model, Breiman (2001), is a collection of *ensemble* of many decision trees and is a modification of *bagging* which builds a collection of *de-correlated trees* and then averages them, it uses the same fundamental principals as *decision trees* and *bagging*. Recall that, the essential idea in *bagging* was to average many noisy and unbiased models with the aim of reducing variance and since *trees* are notoriously noisy, averaging them helps to mitigate much of this noise. That is, *bagging* introduces a random component

into the tree-building process by randomly sampling with replacement samples of the data, *bagging* then averages the predictions across all of the bootstrapped trees which reduces the variance of the final prediction. *Random Forest* models try to reduce the variance of *bagging* by reducing the correlation between trees and does so through randomly selecting the input variables to each of the *trees*. If the Random Forest model uses very similar trees or correlated sub-samples then the result will not be much different to using a single decision tree and thus the success of Random Forest models comes from having uncorrelated decision trees, this is achieved via bootstrapping and variable randomness.[10] That is, given p input variables, before each split, the model selects $m \leq p$ variables at random where, default input variables are usually, $m = \frac{p}{3}$ for regression and $m = \sqrt{p}$ for classification. Reducing m will reduce the correlation between tree pairs in the ensemble model and will reduce the variance of the average. Moreover when $m = p$ the model is equivalent to *bagging trees*.

In regression problems, the *Random Forest* predictions from each bootstrapped tree are averaged, whereas in classification, a class vote from each tree is taken and the majority vote forms the prediction. For example, in the case of regression, suppose house characteristics are the input variables and house prices are the output variable we wish to predict. A *Random Forest* model will randomly select m variables and n observations and construct many parallel decision trees. Each *terminal node* in each tree produces a prediction and these predictions are averaged across the model and a final house price prediction is given based on the characteristics of the house. Moreover, in a classification task, each decision tree holds one *equally weighted vote* and suppose our model contained 100 independent trees, 60 trees say "*it will rain*" and the remaining 40 trees say "*it will be sunny*" then the score for "*it will rain*" is $(60/40)$ or $\frac{2}{3}$ and the score for "*it will be sunny*" is $\frac{1}{3}$ therefore the model predicts that it will rain with a democratic majority vote in which all decision trees

[10] Bootstrapping is the selection of random samples from the training data (with replacement). Variable randomness selects a random number of variables for each decision tree in the Random Forest model. A combination of these two methods allows the Random Forest model to have uncorrelated trees.

in the *Random Forest* model decide independently. This independence allows each tree in the model to be run in parallel to one another and can therefore speed up computational time significantly. Just as in *decision tree* models, *Random Forest* models have a number of parameters. The number of trees in the model should be defined, along with the number of variables to consider at each split, the complexity of each tree, the splitting rule used in the tree construction and the sampling method. Figure 1.11 depicts the structure of a Random Forest model. It first takes the training data, then takes a random sample $n < N$ with replacement from this training data, select $m < M$ input variables with replacement (m held constant across all trees in the forest). Once each tree has been constructed take a majority vote (classification) or average (regression) across the trees to obtain the final prediction. Algorithm 2 shows the pseudo-code for training a *Random Forest* model.

Algorithm 2: Random Forest Training Algorithm

```
Initialization;
Training set = S;
Attribute set = A;
Number of trees to build = N;
Stop Criterion = Stop;
;
if Stop(S) = false then
    forall i = 1 to c do
        K =
          Bootstrap sample from S with replacement S_i;

        Grow a regression/classification tree
          to the bootstrapped data;
        foreach split do
            Select m variables at random
              from al variables;
            Choose best variable/split among
              m;
            Split the node into two child
              nodes;
        end
        Use stopping criteria Stop to
          determine when a tree is complete
    end
else
    Aggregate or class vote across all trees
end
return Learned ensemble model
```

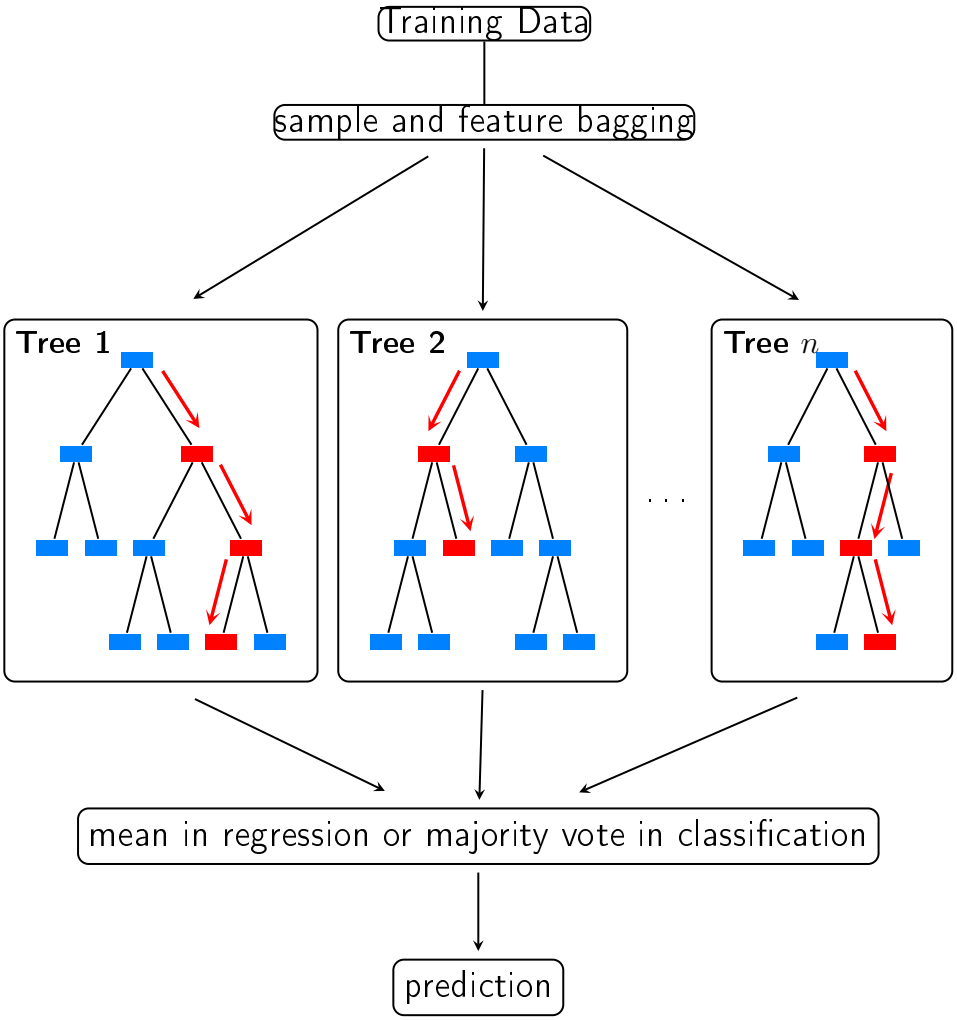


Figure 1.11: Random Forest model

1.13 AdaBoost

Boosting is a powerful Machine Learning idea and combines the outputs of many weak learners to produce a powerful model. It appears similar to *bagging* as previously discussed, however, it is fundamentally different. The purpose of boosting is to sequentially apply weak learners to repeatedly modified versions of the data and thus producing a sequence of weak learners. The objective is to give higher influence to the more accurate learners in the sequence. At each boosted step modifications are made to the data by applying weights $w_1, w_2, \ldots, w_N$ to each of the training observations. At the start, all weights are

set to $w_i = 1/N$ and for each successive iteration $m = 2, 3, \ldots, M$ observation weights are modified and the model is re-applied to the weighted observations. At each step m the observations which were miss-classified or had a higher error have their weights increased, therefore, at each successive iteration, the observations which are difficult to classify or predict are given ever-increasing influence in the model.

AdaBoost (Adaptive Boosting), Freund and Schapire (1997), uses the concept of *boosting* which relies on *weak learners* in which each *weak learner's* error rates are only slightly better than random guessing. The most common way AdaBoost is used is with a form of decision-tree, where each weak classifier is just a *stump*, a two terminal node classification tree, i.e. a decision tree with just a root node and two leaf nodes, where only one variable of the data is evaluated. Once AdaBoost has created its first decision stump, all observations are weighted equally. In order to correct the previous weak learner's error, the miss-classified observations now carry more weight than observations that were correctly classified, such that the next classifier in the sequence will give extra attention to incorrect classifications in order to try and classify them correctly. The order of the stumps is important and the errors that the first stump has made influence how the second stump is made and so on.

Consider the following example in 1.1. A simple classifier has been fitted on the data, also called a decision stump. In panel (A) all of the weights are the same (indicated by the size of the $+$ and $-$) and the model makes a first decision rule, given by the vertical line at $D1$. Whatever the model correctly classifies is given less weighting in the next iteration. The model incorrectly partitioned 3 of the $+$ observations and correctly classified all other points. Since the model incorrectly classified these 3 observations the weights are increased and more emphasis is given to these observations, given by an increase in the size of the points in panel (B). The correctly classified points obtain less weight, given by a reduction in the size of these points, in panel (B). The model makes a new classification depicted at

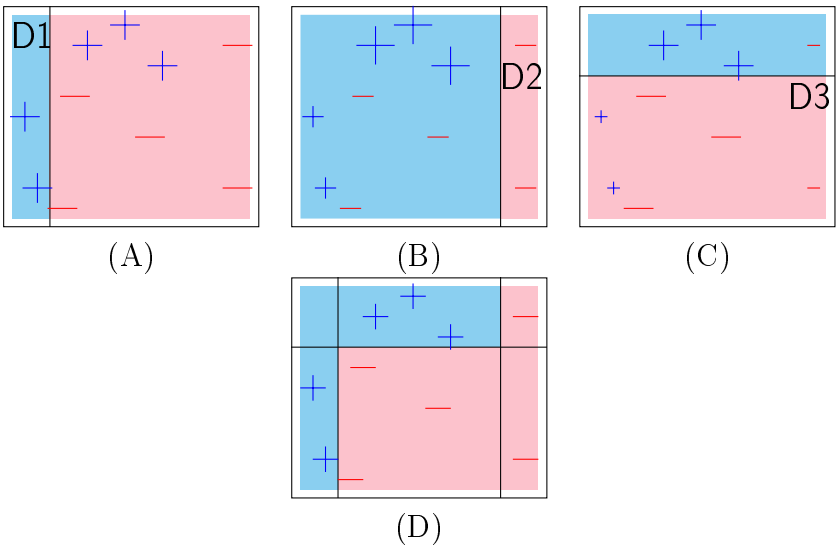


Table 1.1: AdaBoost classification weights illustration

decision rule $D2$. The model correctly classified the previous incorrectly classified + observations and their weights are subsequently decreased (depicted in panel (C)). The two left-most +'s and right-most −'s are again correctly classified and therefore their weights are further reduced in panel (C), given by a smaller + and − signs. Finally, in panel (C) the final classification is given by the horizontal line $D3$ and the weights are recalibrated and updated. The final classification model is given in panel (D) in which all points have been partitioned correctly and this model is a stronger classifier than any individual model previous to it. That is, the model used a combination of linear classifiers in order to build a stronger non-linear classifier based on weighted voting.

1.14 Gradient Boosting Machines (GBM)

Gradient Boosted Machines Friedman (2001) is built of the hypothesis of *weak learners*. Similar to AdaBoost, Gradient Boosted Machines sequentially add predictors to an ensemble with each predictor correcting its predecessor. However, instead of adjusting the weights for each incorrect classified observation at each subsequent iteration as in AdaBoost, Gradient

Boosted Machines tries to fit the new predictor to the residual error made by the previous predictor and in doing so uses Gradient Descent. Due to this we no longer have sample weights as in typical *boosting* ensembles and now all of the weak models have the same amount of say or importance. Unlike AdaBoost, the decision trees are no longer decision stumps, they are larger, fixed-size decision trees. Gradient Boosting uses a learning rate and takes small incremental steps towards better results, conceptually to what is done in Gradient Descent. That is, Gradient Boosting fits a model to the data $F_1(x) = y$, generates a new model $F_2(x) = F_1(x) + h_1(x)$ and so on, more generally we have, $F_n(x) = F_{n-1}(x) + h_{n-1}(x)$. Therefore after combining weak learners with error-corrected predictions, the final model is able to account for much of the error from the first weak learner. The additivity of *Gradient Boosted* models ensures that trees are added sequentially and previous existing trees are not modified when new trees are added to the model. Moreover, new trees are added which minimise the loss function - or follows the gradient and reduce the residual loss. The loss function is minimised through iteratively modifying parameters and the quickest way to do so is to take the path with the steepest slope. *Gradient descent* measures the local gradient of the loss function for a given set of parameters Θ, it then follows the direction of the descending gradient, as depicted in Figure 1.12.

Consider the loss function;

$$L(f) = \sum_{i=1}^{N} L(y_i, f(x_i)). \tag{1.9}$$

The objective is to minimise $L(f)$ with respect to f. The *steepest descent* is one which chooses $h_m = -\rho_m g_m$ where ρ_m is a scalar and $g_m \in \mathbb{R}^N$ is the gradient of $L(f)$ evaluated at $f_{m-1}(x_i)$. Moreover, the components of the gradient g_m are

$$g_{im} = \left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)} \right]_{f(x_i)=f_{m-1}(x_i)} \tag{1.10}$$

and the *step length* ρ_m is the solution to

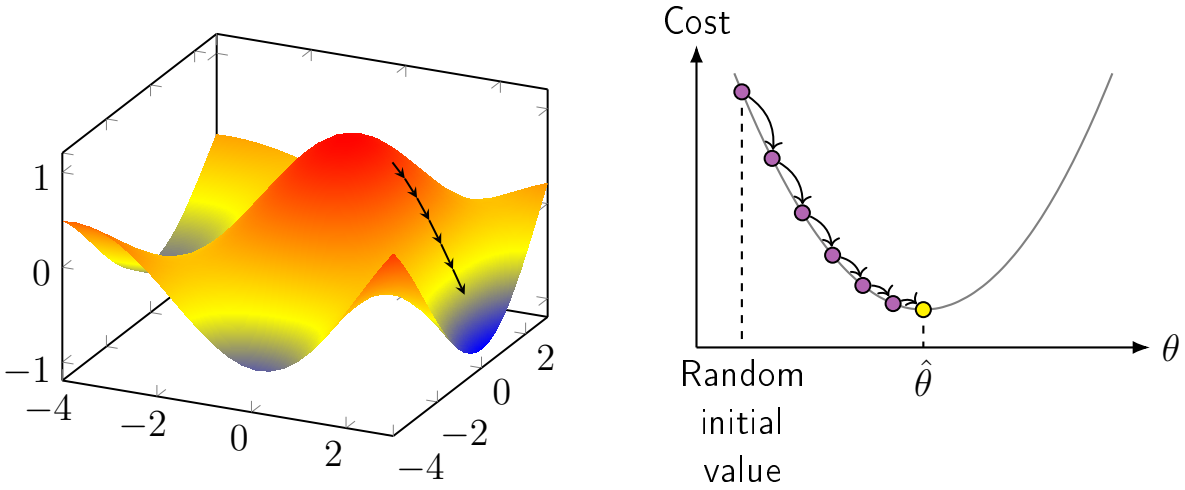


Figure 1.12: Gradient decent for gradient boosted models

$$\rho_m = \arg\min_{\rho} L(f_{m-1} - \rho g_m) \tag{1.11}$$

The current solution then updates

$$f_m = f_{m-1} - \rho_m g_m \tag{1.12}$$

and the process is repeated at the next iteration. The steepest descent is said to be a greedy strategy, since $-g_m$ is the local direction in $\mathbb{R}^N$ for which $L(f)$ is most rapidly decreasing at $f = f_{m-1}$ Hastie et al. (2009).

One of the significant parameters in *gradient descent* is the *learning rate* or *shrinkage*, or the magnitude in step when following the descending gradient. For small *learning rates* the model will converge to a minimum in too many iterations and thus take longer to train the model. Conversely, since not all loss functions are convex, for *learning rates* which are higher, the model may "*jump*" over the global minimum and end up in a local minimum which is not optimal. The *number of trees* and *depth of trees* are also *hyperparameters* in *Gradient Boosted Machines*. Finally *sub-sampling* controls the fraction of observations used in training. Sampling (without replacement) fewer than 100% of the observations uses

stochastic gradient descent which can help the model from being stuck in a local minima and get close to the global minimum. Algorithm 3 and 4 shows the pseudo-code for training a gradient boosted model for regression and classification respectively.

Algorithm 3: Gradient Boosting (Regression) Training Algorithm

Initialization;
$f_0(x) = \arg\min_\gamma \sum_{i=1}^{N} L(y_i, \gamma)$;
forall $m = 1$ *to* M **do**
  forall $i = 1, 2, \ldots, N$ **do**
    (a) $r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}$;
    (b) Fit a regression tree to the targets r_{im} giving terminal regions R_{jm}, $j = 1, 2, \ldots, J_m$;
    (c) **forall** $j = 1, 2, \ldots, J_m$ **do**
      $\gamma_{jm} = \arg\min_\gamma \sum_{x_i \in R_{jm}} L(y_i, f_{m-1}(x_i) + \gamma)$
    end
    (d) Update $f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$
  end
end
return $\hat{f}(x) = f_M(x)$

Algorithm 4: Gradient Boosting (Classification) Training Algorithm

Initialization;
$f_{k0}(x) = 0,\ k = 1, 2, \ldots, K$;
forall $m = 1$ *to* M **do**
  (a) Set $p_k(x) = \frac{e^{f_k(x)}}{\sum_{\ell=1}^{K} e^{f_\ell(x)}}$, $k = 1, 2, \ldots, K$;
  (b) **forall** $k = 1$ *to* K **do**
    i. Compute $r_{ikm} = y_{ik} - p_k(x_i)$, $i = 1, 2, \ldots, N$;
    ii. Fit a regression tree to the targets r_{ikm}, $i = 1, 2, \ldots, N$, giving terminal regions R_{jkm}, $j = 1, 2, \ldots, J_m$;
    iii. Compute;
    $\gamma_{jkm} = \frac{K-1}{K} \frac{\sum_{x_i \in R_{jkm}} r_{ikm}}{\sum_{x_i \in R_{jkm}} |r_{ikm}| \left(1 - |r_{ikm}|\right)}$, $j = 1, 2, \ldots, J_m$;
    iv. Update $f_{km}(x) = f_{k,m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jkm} I(x \in R_{jkm})$;
  end
end
return $\hat{f}_k(x) = f_{kM}(x),\ k = 1, 2, \ldots, K$

1.15 Extreme Gradient Boosting (XGBoost)

Gradient Boosting Machines and *Extreme Gradient Boosting* Extreme Gradient Boosting (XGBoost) both follow the same principle of *Gradient Boosting*. However, XGBoost adds regularisation (L1 and L2 regularisation) to the objective function in order to control the problem of over-fitting, this means that XGBoost has a greater set of hyper-parameters that are tunable, this in turn allows it to achieve a significant increase in performance. Regularisation has an impact on variable weights in the cost function and is a form of variable selection, therefore regularisation reduces noise in the explanatory variables by taking the sum of the variable weights multiplied by some regularisation term and added to the cost function. Additionally, XGBoost is able to handle missing values automatically and is able

to scale well onto sufficiently large datasets. It also allows the user to plug in their own optimisation objective function. Moreover, the *extreme* part of the name is related to the engineering goal to push the computational resource limit for boosted tree models. For example, each tree in a Random Forest model is created independently and therefore it is possible to parallelise the ensembles across different processors in a computer which makes Random Forest models very fast. Moreover, for Gradient Boosted models each of the trees are dependent on the errors of the previous trees and thus it is simply not possible to parallelise the ensembles across different processors as with Random Forest models, however, XGBoost can parallelise the nodes within each depth of each tree. That is, the algorithm is not growing the decision trees in parallel but it is using pre-sorting and block storage methods in parallel processing to determine what the optimal split point will be. A linear scan can then be used across the blocks to determine where the optimal split point is for each variable being modelled. Additionally, XGBoost uses sparse matrices, better data structures and cache utilisation allowing for a better balance between in-memory and out-of-memory processes. This type of attention to the engineering of XGBoost allows it to be so fast, flexible and accurate.

More formally, the *objective function* in XGBoost consists of two parts, a *training loss* and a *regularisation term*:

$$\text{obj}(\theta) = L(\theta) + \Omega(\theta) \tag{1.13}$$

Where L is the *training loss*[11] which measures how *predictive* the model is and Ω is the *regularisation term*. The *regularisation term* $\Omega(\theta)$ helps to control for the complexity of the model and therefore avoids over-fitting.

[11] For regression problems one common choice for $L(\theta)$ is the *mean square error* $L(\theta) = \sum_i (y_i - \hat{y}_i)^2$ and for logistic another choice is the *logistic loss function* $L(\theta) = \sum_i [y_i \ln(1 + e^{-\hat{y}_i}) + (1 - y_i) \ln(1 + e^{\hat{y}_i})]$

Mathematically, the model is as follows;

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i), \qquad f_k \in \mathcal{F} \tag{1.14}$$

Where K is the number of *trees*, f is a function selected from some functional space $\mathcal{F}$ where $\mathcal{F}$ is the set of all possible Classification and Regression Trees (CARTs). The generalised objective function which is to be optimised is:

$$\text{obj}(\theta) = \sum_{i}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k) \tag{1.15}$$

A more formal description of XGBoost has been left to the appendix.

Consider Figure 1.13 and how changes in the regularisation term control the complexity of the model and its response to over-fitting. The objective is to fit a step function given in the upper left quadrant. We want a simple and predictive model. The figure in the upper right quadrant is too complex and over-fits the data, the figure in the lower left quadrant is simple but not very predictive. Therefore, the most simple and most predictive model is in the lower right quadrant. The trade-off between a simple and predictive model is also referred to as the bias-variance trade-off previously discussed.

1.16 Confusion Matrix

Suppose we have a dataset that consists of an independent variable and a dependent variable where the dependent variable contains either a 0 or a 1. After splitting the data between a training and testing dataset and training the model we want to summarise how the model performed on the testing data. For classification tasks, a confusion matrix can be constructed in which the rows correspond to what the model predicted and the columns correspond to the actual known true observed value.

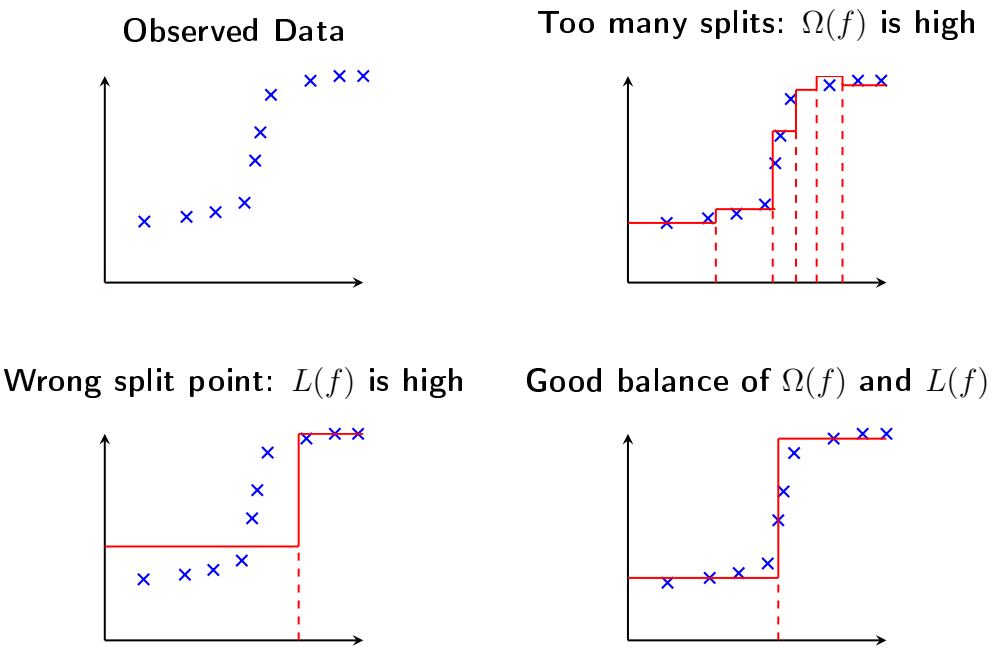


Figure 1.13: XGBoost Regularisation

Using 0 and 1 as our outcome variables where 1 corresponds to the class of interest and 0 is included such that the model can learn the distinct differences between the two classes. The *true positives* refer to observations that had a 1 that were correctly identified by the model. The *true negatives* refer to observation that had a 0 that were correctly identified by the model. The *false negatives* refer to the observations which had a 1 but where the model incorrectly identified them as 0. Finally, the *false positives* refer to the observations which had a 0 but the model incorrectly identified them as 1. A series of summary statistics can be derived from the confusion matrix. The confusion matrix can be expanded to multi-class classification problems which naturally increases the dimensions of the matrix nxn where n is the number of classes to predict.

The *sensitivity* tells us what percentage of observations of the 1 class were correctly identified and is given as True Positives/(True Positives + False Negatives). The *specificity* tells us what percentage of observations of the 0 class were correctly identified and is given

as True Negatives/(True Negatives + False Positives).

		Predicted class		
		Positive	Negative	
Actual class	Positive	True positive T_p *(Correct)*	False positive F_p *(Incorrect)*	Precision $\frac{T_p}{T_p+F_p}$
	Negative	False negative F_n *(Incorrect)*	True negative T_n *(Correct)*	Negative $\frac{T_n}{T_n+F_n}$
		Sensitivity/Recall $\frac{T_p}{T_p+F_n}$	Specificity Rate $\frac{T_n}{T_n+F_p}$	Accuracy $\frac{T_p+T_n}{(T_p+T_n+F_p+F_n)}$

Table 1.2: Confusion Matrix

1.17 Shapley Values

Shapley values Shapley (1953) adapted from coalition game theory allows us to interpret the predictions of Machine Learning models by treating each variable as a *player* in a game with the prediction being the payout. A cooperative game can be considered as the following. A characteristic function game G is given by a pair (N, v) where N is the number of players and $v : v^N \to \mathbb{R}$ is a characteristic function which maps every coalition of players to a payoff.

Consider the following example which has three variables and the output is a prediction from a Machine Learning model. Each variable contributes differently to the prediction (some variables contribute more others less). The characteristics function $v(c)$ tells us that if the variable X_1 was the only variable in the model then it would make a contribution of 80, the variables X_2 and X_3 on their own make a contribution of 56 and 70 respectively. The value of the grand coalition is 90 with all three variables X_1, X_2 and X_3 contributing

to the prediction, there are also different contribution scores for each possible coalition of variables. The Shapley value is defined as $\phi_i(G) = \frac{1}{n!}\sum_{\pi\in\prod_n}\Delta_\pi^G(i)$. The table on the right considers every permutation of players, that is the first line considers the permutation of X_1, X_2 and X_3 with a score of (80, 0, 10). Looking $v(c)$ on the left we can see that X_1 on its own contributes the value of 80, additionally, we can see that when X_1 and X_2 are in a coalition the payout is also 80, therefore X_1 contributes to the prediction and X_2 contributes nothing, finally, when X_1, X_2 and X_3 are in a coalition, the score is 90, therefore X_3 must contribute 10. Consider now when we swap the order of the variables, X_1 still contributes 80 but the coalition of X_1 and X_3 is 85, therefore X_3 contributes 5. Still, the coalition of X_1, X_2 and X_3 is 90, therefore X_2 must contribute 5 also. Finally, if the order is as in line three, then X_2 contributes 56 on its own with the coalition of X_1 and X_2 being 80, X_1 must contribute 24 and the coalition of all variables is 90, therefore X_3 must contribute 10. This is done for all permutations of the variables and then the average of these values are taken, given as ϕ which gives us the average marginal contribution for each variable over all of the possible subsets.

$$v(c) = \begin{cases} 80, \text{if c} = \{X_1\} \\ 56, \text{if c} = \{X_2\} \\ 70, \text{if c} = \{X_3\} \\ 80, \text{if c} = \{X_1, X_2\} \\ 85, \text{if c} = \{X_1, X_3\} \\ 72, \text{if c} = \{X_2, X_3\} \\ 90, \text{if c} = \{X_1, X_2, X_3\} \end{cases}$$

π	δ_π^G
(X_1, X_2, X_3)	$(80, 0, 10)$
(X_1, X_3, X_2)	$(80, 5, 5)$
(X_2, X_1, X_3)	$(24, 56, 10)$
(X_2, X_3, X_1)	$(18, 56, 16)$
(X_3, X_1, X_2)	$(15, 5, 70)$
(X_3, X_2, X_1)	$(18, 2, 70)$
ϕ	$(39.2, 20.7, 30.2)$

1.18 Conclusion

This chapter gives an overview of Machine Learning and has aimed to cover the fundamental concepts of Machine Learning from an econometric standpoint. The focus and emphasis have been placed on the models used throughout the Thesis. Many of the problems which persist in econometrics also persist in Machine Learning, the difference being in the way

both statistical fields handle the problem, the focus of econometric models is causal inference whereas the core focus of Machine Learning lies in predictive accuracy.

A discussion on the different use cases of where and how Machine Learning is being applied in economics has also been covered. The future of Machine Learning applications in economics will continue to expand allowing economists to analyse problems in new ways and with new tools. This will bring about changes in how empirical work is conducted, moreover, econometricians can contribute to the field of Machine Learning by solving causal inference problems in which current Machine Learning models are not adequately designed to capture. Athey (2018) states that there will be a *development of new econometric methods based on machine learning designed to solve traditional social science estimation tasks* and that there will be *no fundamental changes to theory of identification of causal effects.* The combination of econometrics and Machine Learning will allow synergy of both disciplines, taking advantage of one another and further advance the understanding of complex economic problems.

The structure of this chapter was to try and ease the reader into what Machine Learning is, more specifically, the focus has been on easing the reading into the XGBoost model by starting with a description of a simple decision tree, then moving onto the notion of ensemble modelling by introducing *Random Forest* models which use *bagging*. Then moving into the concept of *boosting* through the introduction of the *AdaBoost* model and ultimately introducing gradient boosting models *Gradient Boosted Machines* and finally *XGBoost* and regularisation.[12]

[12] Other models have been briefly discussed such as *Neural Networks*, *RIDGE*, *LASSO*, *Elastic Net* and *Support Vector Machines* (SVM).

1.19 Appendix

1.19.1 XGBoost details

A more formal description of XGBoost is the following, moreover, when considering XGBoost for regression and classification, the only difference between the two models is in the loss function as given in equations 1.16 and 1.17 for regression and classification respectively. Throughout this section, equations given on the left-hand side are related to regression trees whereas equations on the right-hand side are related to classification trees, when there is a single equation, the math is the same for both regression and classification trees.

$$L(\theta) = \sum_i (y_i - \hat{y}_i)^2 \quad (1.16) \qquad L(\theta) = \sum_i [y_i \ln(1 + e^{-\hat{y}_i}) + (1 - y_i) \ln(1 + e^{\hat{y}_i})] \quad (1.17)$$

XGBoost makes an initial prediction of $p_i^0 = 0.5$ regardless of whether it is solving a *regression* or *classification* problem. The residuals tell us how good the initial prediction is. This is quantified using a loss function $L(y_i, \hat{y}_i)$, in which the loss function for *regression* is the squared residual and the loss function for *classification* is the negative log likelihood shown previously in equations 1.16 and 1.17. The objective function is given as;

$$[\sum_{i=1}^{n} L(y_i, \hat{y}_i)] + \gamma T + \frac{1}{2}\lambda O_{value}^2 \quad (1.18)$$

Where $\sum_{i=1}^{n} L(y_i, \hat{y}_i)$ is the loss function we wish to minimise, the γT part of the equations contains T which is the number of terminal nodes (or leaves) in a given tree and γ is a penalty parameter we assign in order to encourage pruning the trees. The $\frac{1}{2}\lambda O_{value}^2$ contains a *regularisation* λ term which scales the *output value*. When constructing the tree the objective is to find an *output value* O_{value} for each leaf which minimises the objective function given in Equation 1.19. As we increase the value of $\lambda > 0$ then the O_{value} shrinks towards zero since more regularisation weight has been emphasised on it and when $\lambda = 0$ the regularisation term is removed.

We want to find the *output value* O_{value} which minimises the following:

$$[\sum_{i=1}^{n} L(y_i, y_i + O_{value})] + \gamma T + \frac{1}{2}\lambda O_{value}^2 \tag{1.19}$$

XGBoost uses the *second order Tayler approximation* in order to simplify the problem when solving for the optimal *output value* O_{value} for both regression and classification.

$$L(y, \hat{y}_i + O_{value}) \approx L(y, \hat{y}_i) + [\frac{d}{d\hat{y}_i} L(y, \hat{y}_i)] O_{value} + \frac{1}{2}[\frac{d^2}{d\hat{y}_i^2} L(y, \hat{y}_i)] O_{value}^2 \tag{1.20}$$

Where $L(y, p_i)$ is the *loss function* from the previous prediction, $g = [\frac{d}{dp_i} L(y, p_i)]$ and $h = [\frac{d^2}{dp_i^2} L(y, p_i)]$ are the first and second order derivatives of that *loss function* which correspond to the *gradient* and the *Hessian*, now denoted as g and h respectively. Equation 1.20 can be reduced to the following:

$$L(y, \hat{y}_i + O_{value}) \approx L(y, \hat{y}_i) + g O_{value} + \frac{1}{2} h O_{value}^2 \tag{1.21}$$

$$\begin{aligned} L(y, \hat{y}_i + O_{value}) \approx L(y_1, \hat{y}_1^0) + g_1 O_{value} + \frac{1}{2} h_1 O_{value}^2) \\ + L(y_2, \hat{y}_2^0) + g_2 O_{value} + \frac{1}{2} h_2 O_{value}^2 \\ + ... + \\ L(y_n, \hat{y}_n^0) + g_n O_{value} + \frac{1}{2} h_n O_{value}^2 \\ + \frac{1}{2}\lambda O_{value}^2 \end{aligned}$$

Since $L(y, \hat{y}_i)$ does not depend on the output value O_{value}, we can omit it since it has no effect on the optimal output value. Therefore, it reduces to;

$$\frac{d}{dO_{value}} \big[(g_1 + g_2 + ... + g_n) O_{value} + \frac{1}{2}(h_1 + h_2 + ... + h_n + \lambda) O_{value}^2\big] = 0 \tag{1.22}$$

Taking the derivative and solving for O_{value} gives us the optimal output value for the leaf;

$$O_{value} = \frac{-(g_1 + g_2 + ... + g_n)}{(h_1 + h_2 + ... + h_n + \lambda)} \tag{1.23}$$

Since the loss function is different for regression and classification then the values for the gradients are given as;

$$\begin{aligned} g_i &= \frac{\partial}{\partial \hat{y}_i} \frac{1}{2}(y_i - \hat{y}_i)^2 \\ &= -(y_i - \hat{y}_i) \end{aligned} \tag{1.24}$$

$$\begin{aligned} g_i &= \frac{\partial}{\partial log(odds)} L(y_i, log(odds)_i) \\ &= -(y_i - \hat{y}_i) \end{aligned} \tag{1.25}$$

For regression and classification respectively which are simply the negative residual.

The Hessians are;

$$\begin{aligned} h_i &= \frac{\partial^2}{\partial \hat{y}_i^2} \frac{1}{2}(y_i - \hat{y}_i)^2 \\ &= \frac{d}{d\hat{y}_i} - (y_i - \hat{y}_i) \\ &= 1 \end{aligned} \tag{1.26}$$

$$\begin{aligned} h_i &= \frac{\partial^2}{\partial log(odds)^2} L(y_i, log(odds)_i) \\ &= \hat{y}_i \times (1 - \hat{y}_i) \end{aligned} \tag{1.27}$$

Plugging in the g_i and h_i we obtain:

$$O_{value} = \frac{-(-(y_i - \hat{y}_i) + -(y_i - \hat{y}_i) + \cdots + -(y_n - \hat{y}_n))}{1 + 1 + \cdots + h_n + \lambda} \tag{1.28}$$

$$O_{value} = \frac{-(-(y_i - \hat{y}_i) + -(y_i - \hat{y}_i) + \cdots + -(y_n - \hat{y}_n))}{\hat{y}_i \times (1 - \hat{y}_i) + \hat{y}_i \times (1 - \hat{y}_i) + \cdots + h_n + \lambda} \tag{1.29}$$

Or, more generally we have an optimal leaf (where ω is the same as O_{value});

$$\omega^* = -\frac{\sum g_i}{\sum h_i + \lambda} \tag{1.30}$$

Therefore, we have the following for regression and classification;

$$O_{value} = \frac{\sum \text{Residuals}_i}{\text{Number of Residuals} + \lambda} \tag{1.31}$$

$$O_{value} = \frac{\sum \text{Residuals}_i}{\sum \left[\text{Previous Probability}_i \times (1 - \text{Previous Probability}_i)\right] + \lambda} \tag{1.32}$$

In order to grow the tree, the similarity scores are calculated from;

$$(g_1 + g_2 + ... + g_n)O_{value} + \frac{1}{2}(h_1 + h_2 + ... + h_n + \lambda)O_{value}^2 \tag{1.33}$$

Multiplying Equation 1.33 by -1 and plugging in the optimal values from Equation 1.23 we obtain;

$$-(g_1+g_2+...+g_n)\left[\frac{-(g_1+g_2+...+g_n)}{(h_1+h_2+...+h_n+\lambda)}\right]-\frac{1}{2}(h_1+h_2+...+h_n+\lambda)\left[\frac{-(g_1+g_2+...+g_n)}{(h_1+h_2+...+h_n+\lambda)}\right]^2 \tag{1.34}$$

Which simplifies to give the following similarity score;[13]

$$\left[\frac{(g_1+g_2+...+g_n)^2}{(h_1+h_2+...+h_n+\lambda)}\right]-\frac{1}{2}\left[\frac{(g_1+g_2+...+g_n)^2}{(h_1+h_2+...+h_n+\lambda)}\right]=\frac{1}{2}\frac{(g_1+g_2+...+g_n)^2}{(h_1+h_2+...+h_n+\lambda)} \tag{1.35}$$

Or, more generally;

$$\text{Similarity}=\frac{(\sum_i g_i)^2}{\sum_i h_i+\lambda} \tag{1.36}$$

Since g_i is the negative residual $g_i=-(y_i-\hat{y}_i)$ for both regression and classification (from Equations 1.24 and 1.25, respectively) then the numerator is just the sum of the residuals squared. For regression, the denominator contains, $h_i=\frac{d}{d\hat{y}_i}-(y_i-\hat{y}_i)=1$, from Equation 1.26. For classification, the denominator contains $h_i=\hat{y}_i\times(1-\hat{y}_i)$ from Equation 1.27. Therefore, for both regression and classification the similarity scores are;

$$\text{Similarity}=\frac{\sum\text{Residuals}_i^2}{\text{Number of Residuals}+\lambda} \tag{1.37}$$

$$\text{Similarity}=\frac{\sum\text{Residuals}_i^2}{\sum\left[\text{Previous Probability}_i\times(1-\text{Previous Probability}_i)\right]+\lambda} \tag{1.38}$$

The similarity scores - or tree structure $\mathcal{Q}$ is not given in advance and the model *learns* the tree structure, the tree is built from the root node and then the *most important* variables are subsequently sought and the best split is chosen. That is, in each round the algorithm greedily separates one after another (sequentially) the features and decides to split on the feature which gives the maximum reduction calculated by $(\tilde{L}(q))$.

Moreover, define $G=\sum g_i$ and $H=\sum h_i$ then $\tilde{L}(q)^*$ becomes (where the previous analysis omitted γT for simplicity);

[13]Note: XGBoost's implementation omits the $\frac{1}{2}$ constant which is one example of how XGBoost tries to reduce the amount of computations it makes.

$$(\tilde{L}(Q))^* = -\frac{1}{2}\sum \frac{G^2}{H+\lambda} + \gamma T \tag{1.39}$$

The *Gain* G is defined as follows;

$$G = \frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{(G_L+G_R)^2}{H_L+H_R+\lambda} - \gamma \tag{1.40}$$

With $\frac{G_L^2}{H_L+\lambda}$ being the score for the left child of the split and $\frac{G_R^2}{H_R+\lambda}$ the score for the right child of the split. Let $\frac{(G_L+G_R)^2}{H_L+H_R+\lambda}$ be the score when the split does not occur and γ penalises the addition of leaves in the tree structure.

Cover

The cover relates to the minimum number of residuals in a leaf and is simply the denominator of Equation 1.37 and 1.38 minus λ for regression and classification, respectively, i.e. it is the sum of the Hessians.

$$\text{Cover} = h_1 + h_2 + \cdots + h_n \tag{1.41}$$

For regression, the Hessian $= 1$ and since there is one Hessian per residual in each leaf, then the Cover $= h_1+h_2+\cdots+h_n =$ Number of Residuals. For classification, the Hessian $= \hat{y}_i \times (1-\hat{y}_i)$ so the Cover is equal to the sum of the previously predicted probability multiplied by one minus the previous predicted probability, Cover $= h_1+h_2+\cdots+h_n = \sum[\hat{y}_i \times (1-\hat{y}_i)]$.

1.19.2 Neural Networks

A **Neuron** is a single node in a Neural Network which consist of a set of inputs, weights and an activation function. The neuron translates these into an output which is passed to the next layer in a Neural Network. Consider the neuron in Figure 1.14, it consists of two parts, it takes x_i as its input and assigns a weight w_{ji}, the first part computes s_j using these inputs and weights, the second part performs the activation on s_j giving the final output

d_j, then we compute an error $d_j - x_j$.

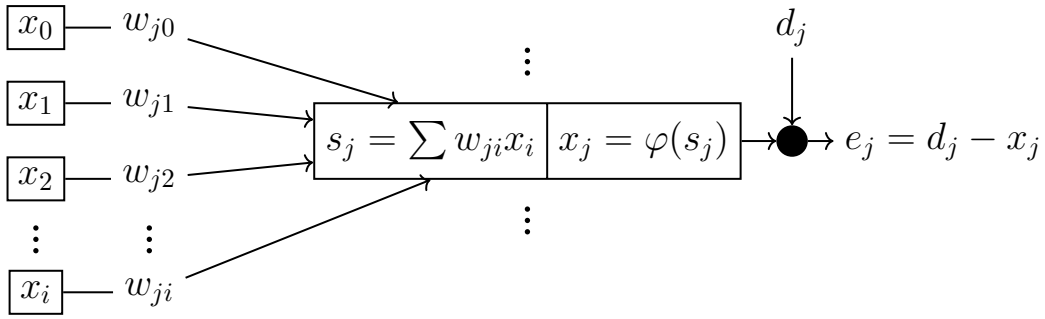


Figure 1.14: Neural Network Neuron example

Shallow Neural Networks consist of a single or few hidden layers. Consider the left network in the Figure 1.15, which contains an *input layer*, a *hidden layer* and an *output layer*. The input layer contains the input variables and in the hidden layer each h_i corresponds to a neuron defined previously, which takes in inputs, assigns weights and an activation function.[14]

Deep Neural Networks consists of adding more hidden layers to a Neural Network structure. It allows for very complex relationships between the inputs and weights and often performs very well on large complex datasets. An example is given in the right-side of Figure 1.15.

1.19.3 RIDGE, LASSO and Elastic Net Regression

Just as in linear regression RIDGE minimises the sum of the squared residuals but with the addition of a penalty added to the least squared regression with a parameter λ determining how severe this penalty is (L2 regularisation). RIDGE is defined as, $\hat{\beta}^{ridge} =$

[14] Activation functions decide whether a neuron should be activated or not and this introduces non-linearity into the output of a neuron. Popular activation functions are the *sigmoid*, *hyperbolic tangent*, *softmax*, *rectified Linear Unit (ReLU)* and *Exponential Linear Units (ELUs)* functions

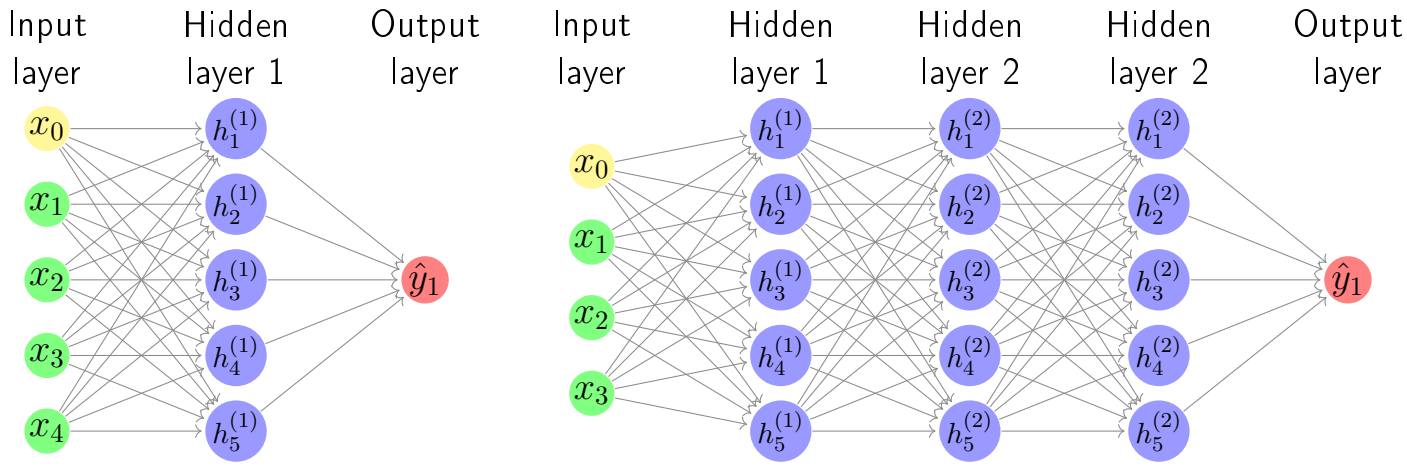


Figure 1.15: Shallow (left) and Deep (right) Neural Network example

$\sum_{i=1}^{n}(y_i - x_i\beta)^2 + \lambda\sum_{j=1}^{p}\beta_j^2$, therefore it is trying to minimise the sum squared residuals plus λ times the slope squared, where λ can be any value from 0 to +infinity. When $\lambda = 0$ then RIDGE only minimises the sum of squared residuals and the RIDGE regression line is the same as the least-squares line. As the value of λ increases, the slope of the regression line gets smaller and the larger the value of λ the slope moves asymptotically close to zero and the prediction for Y gets less and less sensitive to the value of X_i. Through cross-validation, the value of λ is determined by choosing the value which gives the lowest variance. Therefore RIDGE regression helps to reduce variance by shrinking parameters and making the predictions less sensitive to them.

LASSO is similar to RIDGE but instead of taking the squared value of the slope we take the absolute value of the slope with λ determining how severe this penalty is (L1 regularisation). LASSO is defined as, $\hat{\beta}^{lasso} = \sum_{i=1}^{n}(y_i - x_i\beta)^2 + \lambda\sum_{j=1}^{p}|\beta_j|$ and it is trying to minimise the sum squared residuals plus λ times the absolute value of the slope, where λ can be any value from 0 to +infinity. As with RIDGE the LASSSO regression line has less variance than the least-squares regression line. When RIDGE and LASSO shrink parameters, it does not necessarily shrink all parameters equally. As λ increases in value, the slope gets smaller until the slope equals zero whereas RIDGE regression can only shrink

the slop asymptotically close to zero. Therefore, LASSO can shrink irrelevant variables to zero whereas RIDGE can only shrink them asymptotically close to zero, this allows LASSO regression to be a little better than RIDGE at reducing variance in models which contain irrelevant variables, it also allows LASSO equations to be simpler and easier to interpret. RIDGE regression may do better when all of the variables are relevant to the model.

Elastic Net combines the LASSO and RIDGE regression penalty. Elastic Net is defined as, $\hat{\beta}^{elasticnet} = \sum_{i=1}^{n}(y_i - x_i\beta)^2 + \lambda_1 \sum_{j=1}^{p}|\beta_j| + \lambda_2 \sum_{j=1}^{p}\beta_j^2$ and thus it is minimising the sum squared residuals plus the LASSO and RIDGE penalties. Cross-validation is used to determine the optimal values for each of the λ's and when λ_1 and λ_2 are equal to zero we get the original least square estimates, when $\lambda_1 > 0$ and $\lambda_2 = 0$ we get LASSO regression and when $\lambda_1 = 0$ and $\lambda_2 > 0$ we get RIDGE regression, finally when $\lambda_1 > 0$ and $\lambda_2 > 0$ we have Elastic Net. Through the combination of LASSO and RIDGE regression, Elastic Net shrinks the parameters associated with the correlated variables, leaving them in the equation or removing them all at once. Figure 1.16 shows different regularisation function isosurfaces for different values of p where the L_p regulariser is defined as $(\sum|\theta_i|^p)^{1/p}$. For the Euclidean distance $p = 2$ we have the circle with radius 1, when $p = 1$ we obtain the sum of the absolute values and the isosurface corresponds to the star shape. Increases in one value of β is exactly offset by decreases in another value of β. As p gets larger the isosurface shape approaches a square shape and p small we are able to have large values of one parameter β only if the other parameter β is close to zero.

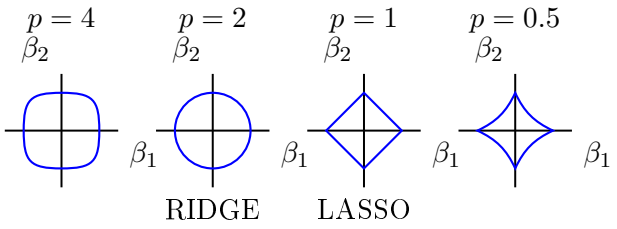


Figure 1.16: Isosurfaces of different L_p regularisers

The most common choices of regularisation are the L1 and L2 norm. This can be visualised in Figure 1.17 as the balance of two different losses, the first loss wants to minimise the $\widehat{\beta}$ point (as we move away from this point it increases quadratically) and the second is a regularisation term which is minimised at the origin (all parameters equal to zero) and in the case of the L2 norm it increases quadratically. The combination of the data plus the regularisation is minimised at the point in which both surfaces are touching. That is, in order to reduce the regularisation we would have to leave the data isosurface which will increase the data loss. Therefore, we can not reduce either loss without increasing the other. L1 penalties can incorporate sparse parameter vectors (a vector which lies exactly on an axis as in the *Opt* point in the left plot in Figure 1.17 - i.e. $\beta_1 = 0$). L2 penalties will only be sparse if the minimum mean square error point ($\widehat{\beta}$) is also exactly on the axis. Therefore the L1 optimum can be on the axis even if the $\widehat{\beta}$ is not and L1 regularised solutions encourages some level of sparsity, making the model more efficient which helps determine which variables are most important in prediction. L_p norms form a balance between sparsity and convexity, when $p \geq 1$ the norm is convex and for $p \leq 1$ it induces some sparsity. Therefore, the L1 norm is the only norm that both induces sparsity and remains convex for easier optimisation.

1.19.4 Support Vector Machine (SVM)

Support Vector Machines can be used for both regression and classification problems. Suppose that we have two classes of data as depicted in Figure 1.18 that we want to separate, in order to do so there are many possible hyperplanes that can be chosen. The objective is to find the hyperplane which has the maximum margin (maximum distance between the nearest data points of the two classes). The linear classifier it defines is known as the maximum margin classifier. The support vectors are the data points that are closest to the hyperplane and therefore influence the orientation of the hyperplane (given by the red points), these points help to maximise the margin of the classifier. The SVM classifier takes on values of

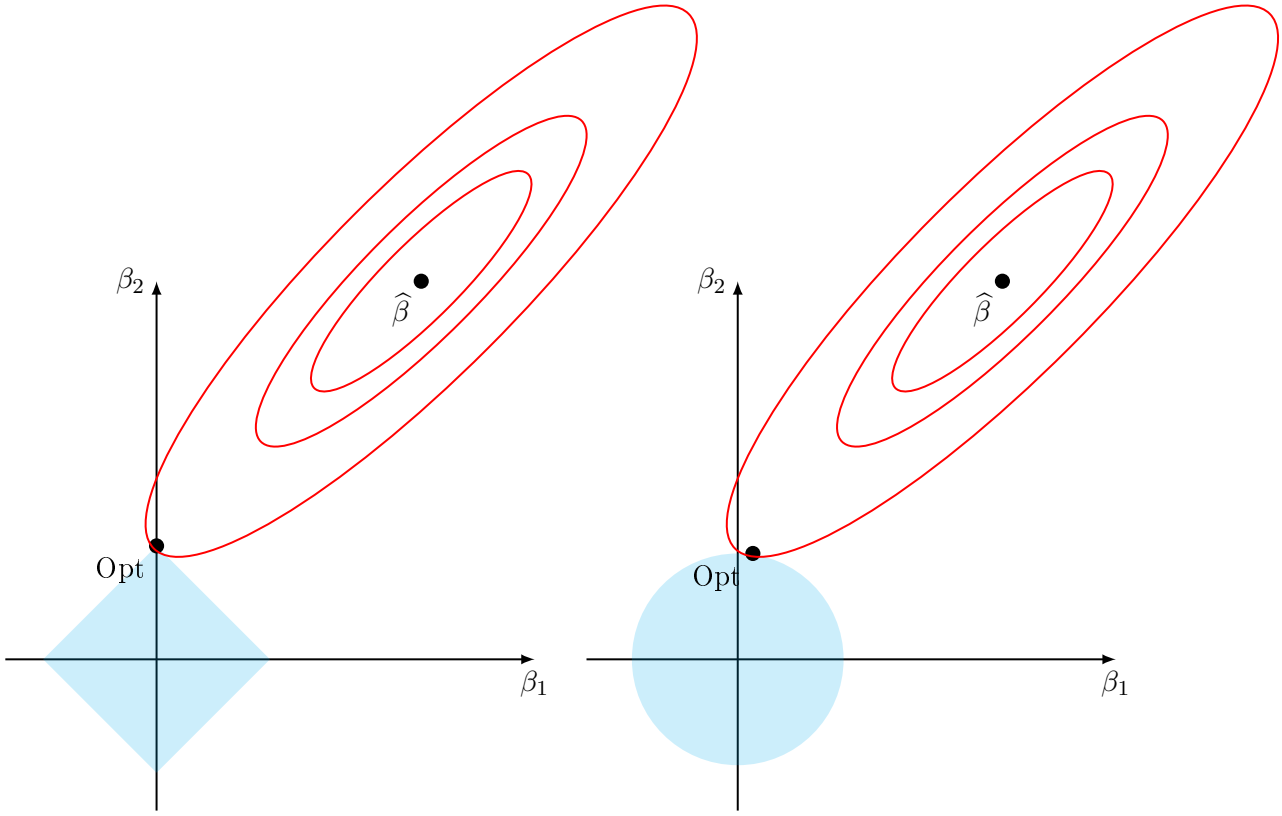


Figure 1.17: Lasso Ridge caption here

[-1, 1], a value of -1 for one class and a value of 1 for the other class for which x_i belongs to. Each x_i is a p-dimensional real vector and the objective is to find the *maximum-margin hyperplane* which divides the group of points x_i where $y_i = 1$ and $y_i = -1$. The hyperplane is represented as $w^T x - b = 0$, where w is a normal vector to the hyperplane. The parameter $\frac{b}{\|\mathbf{w}\|}$ determines the offset of the hyperplane from the origin along the normal vector w. Given a linearly separable dataset, two parallel hyperplanes separate the two classes such that the distance between the two hyperplanes is as large as possible. The bounded region by the two hyperplanes is defined as the *margin* with the *maximum-margin hyperplane* corresponding to the line which sits halfway between the two hyperplanes. Geometrically, the distance between the two hyperplanes is given as, $\frac{2}{\|\mathbf{w}\|}$, therefore maximising the distance between the two planes, requires the minimisation of $\|\mathbf{w}\|$. Therefore, the optimisation problem becomes, minimise $\|\mathbf{w}\|$ subject to $y_i(w^T x_i - b) \geq 1$ for $i = 1, \ldots, n$. Such that w

and b solve the problem. Figure 1.18 shows an illustration of Support Vector Machines on a sample linearly separable dataset.

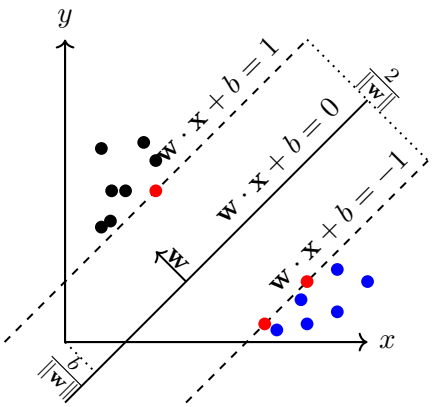


Figure 1.18: Support Vector Machine example

Chapter 2.1

Corporate bankruptcy prediction in Spain from 1992 to 2016

Abstract: We apply a Machine Learning (ML) algorithm in order to predict bankruptcy rates among companies within the Spanish economy from 1992 - 2016. The model identifies some relevant variables when predicting bankruptcy: such as the ratio Total Liabilities to Total Assets or Current Liability to Financial Expenses along with size factors such as the log of sales. Additionally, the model allows us to analyse firms individually: the marginal contribution of a given variable to the firm's prediction depends on all its other observed characteristics. This can be particularly useful in analysing case by case lending decisions within financial institutions. An exercise on the cost of extending the forecasting horizon up to four years ahead is also provided, as financial institutions are naturally interested in the early detection of bankruptcy. We also compare XGBoost to a number of Machine Learning models, such as a Logistic model, Support Vector Machine, Neural Network, Random Forest and LightGBM.

JEL Codes: G17 (Financial forecasting and simulation), G33 (Bankruptcy), C53(Forecasting and prediction methods).

2.1.1 Introduction

Recent developments in regulatory requirements from central banks and governments have caused credit risk management to take a leading role amongst practitioners. The Basel agreement requires financial institutions to limit the amount of risk-weighted assets that a bank can hold, this affects the type and number of loans that a bank can issue in relation to its capital. Therefore, it has become more apparent to the banking industry that new and improved, more innovative ways should be adopted in order to identify counter-party default risk amongst its corporate clients early on. On a more economic scale the ability to adequately shift financial resources from an ailing firm to more positive recipients will help clear up inefficiencies within the financial lending sectors. Forecasting the failure of an organisation is an important economic and financial challenge, failure to adequately manage financially distressed firms within a timely manner in an economy can have profound negative economic consequences. Additionally the insolvency procedure can stretch across many years. Hernandez-Tinoco and Wilson (2013) found that UK firms have an average time gap of 1.17 years from the events which caused the firm to go bankrupt and the date in which bankruptcy was filed. Theodossiou (1993) found that firms in the US fail to provide financial accounts two years prior to bankruptcy. This suggests that firms feel the struggle of financial distress before filing for bankruptcy and it is therefore critical for financial institutions to identify these firms early on.

This paper addresses the issue of new and innovative bankruptcy prediction models by applying a Machine Learning algorithm in order to better classify and distinguish bankrupt firms from non-bankrupt firms. We apply our model over 4 years of financial accounts, aiming to identify financially distressed firms early on. We aim to capture the rarity of financially troubled firms in an economy by using an imbalanced dataset and finally, we aim to capture the imperfectness of financial data through the inclusion of extreme and missing values as information. Moreover we show that Machine Learning models need not

give just a simple black-box prediction but can be interpretable through a number of ways as in traditional regression models. We finally compare our results with a series of other Machine Learning models, notably a Support Vector Machine (SVM), Neural Network, Logistic Regression, Random Forest and Light Gradient Boosted Machine (LightGBM). The higher predictive accuracy and interpretability are just two ways why the model proposed in this paper is considered one of the most prominent Machine Learning models currently in use and therefore would be well posed for the problem of bankruptcy prediction.

The problem of bankruptcy prediction is well suited for classical binary Machine Learning classification problems. A company can either be in a state of bankruptcy or not. The companies with the status of bankruptcy are described as the positive class since we wish to positively identify these firms. The negative class are the non-bankrupt firms, which are included such that the model can learn the differences between the two classes. When a model correctly predicts that a company is in a state of bankruptcy, then it is called a True Positive (TP). Conversely when a model correctly predicts that a company is in a state of non-bankruptcy, then it is called a True Negative (TN). The case when a model predicts that a company is bankrupt but the actual status is non-bankrupt is called a False Positive (FP or Type I error). Finally, the case when the model predicts that a company is non-bankrupt but the actual status is bankrupt is called a False Negative (FN or Type II error). The values TP, FP, FN, TN conform, by rows, a 2×2 matrix denoted as the confusion matrix.

The first challenge corresponds to analysing the cost of extending the forecasting horizon. Predicting bankruptcy early is an important factor in any lending policy. We make predictions for firms that went bankrupt using data 1, 2, 3 and 4 years prior to bankruptcy, in all cases using the whole dataset (including firms that remained active for the whole period). A number of statistics based on a confusion matrix have been computed, all of them with a common -and expected- message: extending the forecasting horizon impairs future pre-

dictions. The relevant question, of course, is to quantify this impairment through relevant statistics. A representative statistic of the performance of any classification algorithm is the Area Under the Curve (AUC) or the Area Under the Precision-Recall Curve (AUPRC). The AUC takes a value of 1 for perfect classification, that is, 0 Type I and II errors, while it takes a value of 0.5 for pure random guessing.[1] The AUPRC takes a value of 1 for a perfect model and 0 for a poor model and is more informative than the AUC for imbalanced data sets. Our model achieves AUC values ranging from 0.84 to 0.74 and AUPRC values ranging from 0.66 to 0.42 for one year to four year prior prediction respectively. It is worth noting that these values can be understood as *optimal* in some sense. Roughly, the goal of Extreme Gradient Boosting (XGBoost) is to push the limit of computational resources for boosted tree algorithms. XGBoost minimizes numerically a loss function which contains a number of parameters which need to be optimised. For this we carried out a grid search on a parameter space in order to minimise prediction errors on an in-sample test set using 10 fold cross validation, which is briefly discussed in more detail later. This implies that we located the parameter values which maximised the in-sample test AUC, AUPRC and minimise some other loss functions. We report our final analysis and statistics on a held-out test set.

Next, we analyse which variables have a higher impact on the likelihood of bankruptcy. As mentioned, the XGBoost model minimizes a loss function. More precisely, the algorithm selects a loss minimizing collection of trees, denoted by ϕ. Each tree in ϕ is a step function that assigns a score to each firm depending on the firm's characteristics. The overall score of the firm is just the sum of these scores across all trees in ϕ. There is a standard monotonic function that maps overall scores into a probability of bankruptcy and, finally, the firm is predicted to be bankrupt whenever that probability overtakes a certain threshold, which we define. This scheme provides a natural environment to identify the most important variables

[1] It takes value 0 for a *perfect misclassification*, which can trivially be turned into a perfect classification.

in predicting bankruptcy. On average, across firms, the variables with the highest marginal contribution to the overall score are the ratio Total Liabilities to Total Assets (TL.TA), the logarithm of Total Assets (logTA), the logarithm of Sales (logSALES), Current Liabilities to Financial Expense (CL.FinExp) and Earnings Before Interest and Tax (EBIT.FinExp). Our analysis indicates that there are slight variations depending on the forecasting horizon: when we take one year prior predictions the most relevant characteristic is TL.TA, while for longer horizons it turns out to be logTA or logSALES. Perhaps more importantly, the across-firm variability is essential for any lending policy. The algorithm allows us to compute marginal impacts of each variable within each firm. A case study is provided for each quadrant of the confusion matrix. Generally, we illustrate the contribution of each variables changes from one firm to another in a way that is highly non-linear but roughly well captured by ϕ.

Our final step is to compare XGBoost to other Machine Learning models. The key difference between the models is not non-linear vs linear - although this is an important characteristic of the models - but on how to deal with complexity. The loss function to be minimized under the XGBoost algorithm has two terms: the prediction errors and the overall complexity of ϕ, respectively.[2] The parameters which help control the complexity are decided by the practitioner, through domain knowledge and cross validation, otherwise default parameter values are given. Once the parameters are chosen, the algorithm starts from a very simple tree structure and recursively adds trees as to minimise an objective function up to a maximum number of trees, again usually determined at the cross validation stage.[3] To summarize, for a parametrized loss function, the XGBoost model automatically evaluates whether each increment of complexity pays off in terms of error prediction improvement.

[2]The measure of complexity is detailed later in the paper.

[3]In our case we ran our model on a maximum number of 1500 tress and told the algorithm to stop learning once it had failed to learn after 500 trees. That is, the AUC failed to improve for 500 consecutive rounds or trees. The model stopped learning after 88 tress and therefore our model recursively adds trees up to tree number 88.

This automatism is absent in logistic models and other Machine Learning models.

The rest of the paper is organized as follows. Section 2.1.2 positions this paper within the existing literature, in section 2.1.3 we present the data, section 2.1.4 discusses the methodology, section 2.1.5 presents the main results, interpretation and case studies, section 2.1.6 compares the performance to other Machine Learning models. Finally, section 2.1.7 concludes the paper.

2.1.2 Previous literature

The discussion of financial ratios as indicators of bankruptcy stretches as far back as the 1930s, see Fitzpatrick (1932) and Winakor and Smith (1935). Statistical modelling was first introduced through Beaver (1966), who analysed 30 univariate financial ratios one by one in order to classify firms as bankrupt or not. Altman (1968) expanded upon this approach by using multivariate discriminate analysis, constructing a Z-score, a measure of the likelihood of bankruptcy in the US manufacturing sector ($Z > 2.675$: healthy firm, $Z < 2.675$: unhealthy firm). Multivariate conditional probability models were then applied. West (1985) used factor scores and applied a multivariate logistic model as an early warning bankruptcy prediction system. The factors they used closely resemble the components of the CAMEL (*Capital Adequacy, Asset Quality, Earnings & Liquidity*) rating system used by bank examiners at the time. Martin (1977), Ohlson (1980), applied financial ratios to a multivariate logistic model, Zmijewski (1984) to a Probit model. Logistic models are still amongst the most popularly used models since no assumptions are made about the distribution of predictor variables however, Begley et al. (1996) show that traditional models used in Altman (1968) and Ohlson (1980) - which were estimated using data from the 1940s through the 1970s - obtain higher measurement errors when estimated on data from the 1980s, thus traditional models do not generalise well on new, unseen data. Frydman et al. (1985) found that recursive partitioning outperformed discriminant analysis and that additional informa-

tion can be derived from the assessment of both models.

The next step in the advancement of predicting financial default comes in the form of a subset of artificial intelligence. Throughout the early 1990s artificial neural networks became a popular method, Odom and Sharda (1990), Bell et al. (1990), Hansen and Messier Jr (1991), Tam and Kiang (1992) and through to the mid 1990s, Altman et al. (1994), Wilson and Sharda (1994), Etheridge and Sriram (1996), Pompe and Feelders (1997). The objective in all studies was to capture firm-level counter-party insolvency using balance sheet and income statement data. Recursive partitioning models such as decision trees, Random Forests and gradient boosting are becoming increasingly common in classification and regression problems. Decision trees are simple and interpretable, Random Forests, see Breiman (2001) are similar but with the added advantage of using multiple decision trees in order to make a classification. Creamer and Freund (2004) were one of the first to apply Random Forests to the problem of bankruptcy prediction. Barboza et al. (2017) show a series of bagging, boosting and Random Forest classifiers outperform traditional statistical models. Zhao et al. (2017) uses a Kernel Extreme Learning Machine (KELM) to discriminate bankrupt companies with non-bankrupt companies. They find that KELM performs better than Support Vector Machines, Extreme Learning Machines, Random Forest, Particle Swarm Optimisation Enhanced Fuzzy K-Nearest Neighbour and a Logistic model in terms of overall accuracy, Type I error, Type II error and AUC.

Zhou and Lai (2017) applied AdaBoost to corporate bankruptcy prediction with missing values and found that it performs better than other benchmark models. Adaptive Boosting (AdaBoost) Freund et al. (1996) is similar to XGBoost in that both models build weak learners, However, XGBoost uses a regularised model formalisation to control over-fitting and improves on the recursive tree-based partitioning method of gradient boosting Friedman (2001), Friedman (2002). The model builds sequentially a series of shallow trees, in which

each additional tree corrects the residual error from all previous trees. XGBoost is more efficient since it applies a sparsity-aware split finding method when training on sparse data. Zięba et al. (2016) applied an Extreme Gradient Boosting approach to predict bankruptcy within the Polish market. They used 700 bankrupt companies and 10,000 non-bankrupt companies to train and test their model. However, they only report on the mean and standard deviation of an AUC evaluation metric. Carmona et al. (2018) applied the same method to predict bankruptcy amongst U.S. national commercial banks. They used a balanced model of 78 failed banks and 78 non-failed banks, reporting on the Sensitivity, Specificity, Accuracy and AUC of their model. This study differentiates itself from the previous two studies by using a larger sample of firms with extreme and missing values, we document a number of new evaluation metrics better suited for imbalanced data classes such as AUPRC and MCC. We go deeper into the interpretability of the model and analyse four case studies. Finally, we compare the model to a series of other Machine Learning models. The model proposed in this paper has been more widely applied to credit scoring models as opposed to bankruptcy default, therefore, we contribute through using new models applied to a different kind of dataset. Xia et al. (2017) applied XGBoost to credit risk default and found that it performs as well if not better than the many other Machine Learning models they applied.

While there is extensive literature surrounding the prediction of bankruptcy, much of it has focused on an equally balanced sample of bankrupt and non-bankrupt firms, unrepresentative of bankruptcy filings in the real economy. There are far more non-failed firms operating within an economy than there are failed, therefore, we feel it is important to capture this class imbalance within our study. Other scientific fields are utilising Machine Learning models on such imbalanced data, it is frequently used in medicine when classifying patients with a rare disease from a large population. Computer sciences use imbalanced data when classifying images of a specific type from a population of many images, i.e. classifying correctly stop signs from all other images taken in automated driving. The model

proposed in this paper has been recognised by CERN as the leading method in identifying the extremely rare Higgs Boson particle from the Large Hadron Collider.[4]

Olmeda and Fernández (1997) analysed Spanish banking data from 1977 to 1985 the Spanish banking sector suffered its worst - at the time - crisis in its history, affecting 52% of the 110 banks operating at the beginning of the period. They used a data set consisting of 9 balance sheet ratios, split into a training set of 15 failed and 19 non-failed banks and a testing set of 14 failed and 18 non-failed banks. They found that Neural Networks provided the best results. De Andrés et al. (2005) applied two parametric models, Linear Discriminant Analysis (LDA) and a Logistic model along with two non-parametric models, Neural Network and Additive Fuzzy Rule-based Systems to the case of classifying Spanish commercial and industrial company profitability groups, based on a set of financial ratios. They show that non-parametric models performed better than parametric models when classifying companies into two distinct profitability groups. Callejón et al. (2013) used a Neural Network to predict the failure of European industrial companies using information for two years prior to bankruptcy. Moreover, Spain accounted for approximately 20% of their data set. Fernández-Gámez et al. (2016) applied a Neural Network using financial and non-financial variables on a sample of a 108 Spanish hotels which went bankrupt between 2005 and 2012, they studied the periods one, two and three year prior to insolvency.

2.1.3 Data

Firm level data was collected from "Sistema de Análisis de Balances Ibéricos" (SABI), which contains balance sheet, income statement and cash flow information that firms are due to report annually. The sample period stretches from 1992 to 2016, firms with total assets of less than 25,000 EUR were removed along with some additional criteria to define a relatively

[4] See Chen and He (2015).

homogeneous population of firms.[5] The data consists of 6057 bankrupt firms along with 58,000 non-bankrupt firms,[6] for each firm the last four years of available financial accounts were collected. Bankruptcy responds to a situation of definitive insolvency as the asset is less than the liability although it could also be the case of being a provisional insolvency. The status of bankruptcy is assigned when the process is declared officially. We define the bankruptcy state as 1 and the non-bankruptcy state as 0 throughout the paper.

A number of ratios which are standard in the literature on financial default were constructed. The analysis will reveal that the most relevant variables in predicting bankruptcy are the ratios Total Liabilities to Total Assets, Current Liabilities to Financial Expense, Earnings Before Interest and Tax to Financial Expense, denoted as TL.TA, CL.FinExp and EBIT.FinExp respectively, along with the variables the logarithm of Total Assets and the logarithm of Sales, denoted logTA and logSALES, respectively. A description of the ratios and variables under consideration is presented in 2.1.8.1.

Before we describe the data, a few remarks on the Spanish economy throughout the sample period are in order. Clearly, the most salient feature of that period was the crisis that affected most of western economies by 2008, which also hit the Spanish economy. The impact in Spain was probably more severe than in other European countries in terms of GDP, unemployment and government debt. Yearly GDP growth was steadily around 5% during 1992 to 2007, whereas it fell to 1.6% between 2008 and 2016, with some recovery starting in 2015. The unemployment rate, which had been around 11% during the pre-crisis period, went above 20% in 2011, the highest in the Euro zone, up until 2016 it remained

[5] The data was filtered by firms with Spanish national legal form in order to eliminate multinational firms with subsidiaries in Spain. Only firms with the entity type as Corporate was selected, eliminating financial companies and banks which have different balance sheets and initial capital requirements than ordinary firms. Data was removed for consolidated accounts where unconsolidated accounts exist, in order to remove the issue of double accounting. Firms were removed with no recent financial data along with Non-profit/Public authorities/State and Government-owned companies, since the decision for filing for bankruptcy is different to that of private firms.

[6] A ratio of 0.104 of bankrupt firms to non-bankrupt.

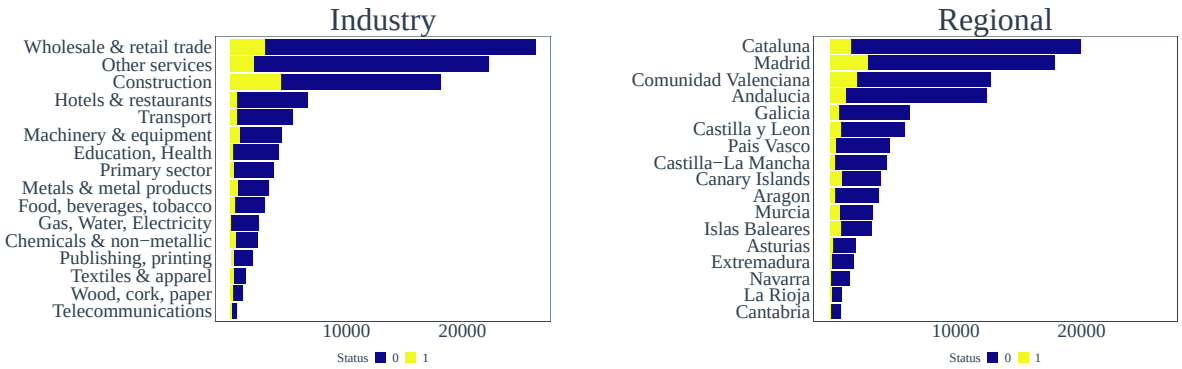


Figure 2.1.1: Sample by sectors and regions.

roughly at that level. Moreover, government debt, as percentage of GDP, increased dramatically, from a yearly average of 60% in the pre-crisis period to 90% in 2012 and it further increased in the following years.[7]

Figure 2.1.1 plots the breakdown of bankrupt and non-bankrupt companies by industry and by regions, in the left and right plot, respectively. By industry, the highest percentage of bankruptcy rates occurs in the construction sector, which in turn, was the third most important sector by number of firms. The construction sector in Spain was one of the more severely affected sectors during the financial crisis. By regions, the top four, most populous regions, Madrid, Catalonia, Valencia and Andalusia all report a large number of bankruptcies over the study period, however Castilla y Leon, the Canary Islands, Murcia and the Balearic Islands all report relatively high numbers of bankruptcy rates relative to their size, which are significantly smaller compared to the top four regions in Spain. Table 2.1.6, in the Appendix, presents some basic statistics for the variables under consideration, separately for bankruptcy and non-bankruptcy firms.

[7]Source: OECD data.

2.1.4 Methodology

We propose a recently developed Machine Learning algorithm, *Extreme Gradient Boosting*, or XGBoost, developed by Chen and Guestrin (2016). Extreme Gradient Boosting can be thought of as a regularised gradient boosting model. Gradient boosting uses an ensemble learning method, which essentially combines the predictive power of several weaker models -also called trees or classifiers- in order to obtain a superior predictive model. These individual models are called base learners or weak learners and may only be slightly better than random guessing. The combination of these weak learners will yield better predictive performance than any individual base learner on its own. The effect is that the combination of weak learners can quickly fit and then potentially over-fit the training data, to correct for this, regularisation is added to the learning objective.

In order to describe XGBoost more precisely, some notation is required. We consider a set of firms, $\mathcal{I} := \{1, \ldots, I\}$, with $i \in \mathcal{I}$ being a generic firm. Each firm i has some characteristics $\mathbf{x}_i$ which imperfectly determine its bankruptcy state, $y_i \in \{0, 1\}$. We denote $y_i = 1$ if firm i is bankrupt and $y_i = 0$ otherwise. The aim is to predict y_i from $\mathbf{x}_i$ for every $i \in \mathcal{I}$. A tree[8] is as depicted in Figure 2.1.2. It is nothing but a step function that maps a vector of characteristics into an score. Let f_k denote a specific tree, such that $f_k(\mathbf{x}_i)$ is the score, or weight, that the tree f_k assigns to any firm with characteristics $\mathbf{x}_i$. If we have a set of K trees, $\phi = \{f_1, \ldots, f_K\}$, the overall score for any such firm is $\hat{y}_i := \sum_{k=1}^{K} f_k(\mathbf{x}_i)$, which constitutes the prediction for y_i based on $\mathbf{x}_i$ using ϕ.[9]

The question, of course, is how to select each element in ϕ from some function space $\mathcal{F}$ of step functions. The XGBoost algorithm selects trees from $\mathcal{F}$ as to minimize a loss

[8] Mathematically, it is a *directed tree* in graph theory.

[9] There is a slight abuse of notation here. The scores are scalars. For our problem, it is necessary that some transformation is needed to turn the overall score into a probability of bankruptcy. Specifically, we compute that probability as $\hat{y}_i = (1 + exp(-\sum_k f_k(\mathbf{x}_i)))^{-1}$.

function. The optimisation procedure is done recursively, such that at every iteration a new tree enters ϕ. The loss function is defined as:

$$L(\phi) = \sum_{i \in \mathcal{I}} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k), \tag{2.1.1}$$

where; l is a logistic loss function, and Ω is a regularisation term which penalizes the complexity of each tree in ϕ. XGBoost measures the complexity of each $f_k \in \phi$ as follows:

$$\Omega(f_k) = \gamma T_k + \frac{1}{2}\lambda ||\omega_k||^2, \tag{2.1.2}$$

where $\gamma \in (0, \infty)$ and $\lambda \in (0, 1]$ are parameters, T_k is the number of terminal nodes, or leaves, while ω_k is a vector of scores, one at each leaf, in f_k. For instance, if f_k is the tree in Figure 2.1.2, we have $T_k = 3$ and $\omega_k = (4, 2, -1)$, thus $||\omega_k||^2 = 21$. Both γ and λ are *regularisation parameters*, while the first and second terms in (2.1.2) are L_1 and L_2 regularisation terms, respectively. Two additional comments on including complexity in the loss function are in order. First, a tree with enough number of leaves can reduce to zero the prediction errors, i.e: $y_i = \hat{y}_i$ for all $i \in \mathcal{I}$. Trivially, it suffices to use one leaf for each firm. However, by doing so we lose the capacity to identify the essential features within the characteristic space that determine bankruptcy. Thus, we do not want *too many* leaves, and the first term in (2.1.2) accounts for that. Second, the L_2 normalisation, which is the second term in (2.1.2), prevents the tree from having too high a score on any particular leaf. Equivalently, it prevents any leaf in any tree to be *too important* within ϕ. Moreover, if such a large score is allowed, it will be ingrained over multiple trees and the significance of this score will be incorporated across the whole set of trees. Finally, a common feature to all boosted methods is that trees are recursively added to ϕ analogously to gradient descent algorithms in classical optimization. The novelty of XGBoost is that it includes the second term in (2.1.2). In what follows, the set of trees ϕ is called *a model*.

We firstly split the data into a training set and a hold-out test set, 75% and 25% respectively, then we apply 10-fold cross-validation on the training set in order to find optimal

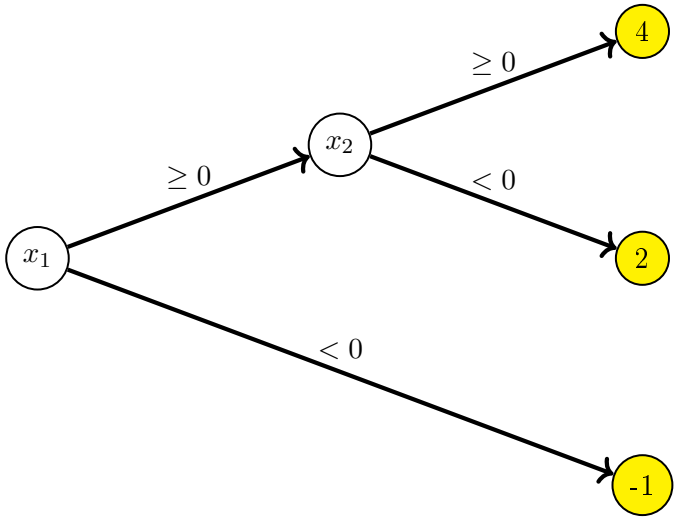


Figure 2.1.2: Example of a tree. This tree assigns scores depending on characteristics x_1 and x_2. The scores are located in the *leaves*, or terminal nodes, in yellow. For instance, a firm with $x_1 \geq 0$ and $x_2 \geq 0$ is classified into the upper leaf, and thus it is given a score of 4. Using the notation in the text and taking $\mathbf{x} = (x_1, x_2)$, we have, for instance, $f(2, 1) = 4$.

parameters. In k-fold cross-validation, the sample is partitioned into k equally sized subsamples, each of which is called a *fold*. The model (ϕ) is trained on $k - 1$ folds. Then the resulting ϕ is tested using data in the remaining fold. There are k rounds, such that each fold becomes part of the training sample $k - 1$ times and the testing sample just once. The purpose of k-fold cross validation is to analyse the performance of different models on multiple in-sample test data sets, this helps avoid over-fitting. The model which performs well within k-fold cross validation should be able to be generalised onto the held-out test data in which the model is finally evaluated.

We have used the implementation of XGBoost in R[10]. The algorithm that minimizes the loss function uses some hyper-parameters which we document below. We carried out a grid search during the cross validation phase in order to locate parameter values which produce the best average performances on the in-sample test data, in terms of the AUC, AUPRC and

[10]See R Core Team (2013) for the programming language and Chen et al. (2018) for the package.

some loss functions. The optimal parameter values are presented in Table 2.1.1, in which the notation is as in the documentation of the XGBoost package. In the table there are a number of hyper-parameters, other hyper-parameters can be optimised but were omitted. `Max Tree Depth` is the maximum depth a tree is allowed before it is forced to make a decision, which can be understood as the maximum number of nodes from the initial node to any leaf node. In theory, an uncontrolled tree can grow to the size of the number of instances in the data such that each terminal node represents a given firm, however this would most likely not generalise well onto new unseen data. To control for this we can set the maximum tree depth or terminate a tree once a node has a minimum number of observations. `Eta` controls the marginal contribution of each tree within ϕ, higher values of `Eta` would very quickly reduce the loss but also very quickly plateau after relatively few trees, smaller values allow us to obtain a lower loss, but at the cost of doing so over many more trees. `Gamma` appears in (2.1.2) and basically specifies the minimum loss reduction required to make an additional split. `Lambda` also appears in (2.1.2) and is an L2 regularisation parameter on the scores. `Sub Sample` and `Col Sample` are the percentage of randomly selected observations and columns when growing each tree.

Hyper-parameter	Optimal Value	Other Values
`Max Tree Depth`	5	(3, 5, 8)
`Eta`	0.1	(0.05, 0.1, 1)
`Gamma`	0.5	(0, 0.5, 1, 1.5)
`Lambda`	1	(1)
`Sub Sample`	1	(0.75, 1)
`Col Sample`	0.75	(0.75, 1)

Table 2.1.1: Main Hyper-parameters Space.

2.1.5 Results

This section contains the results. They are organized around three questions, analysed in separate subsections: what is the cost of enlarging the bankruptcy forecasting horizon?

How do different variables contribute to the prediction of bankruptcy? How can XGBoost be used to analyse individual case studies?

2.1.5.1 Forecasting horizon

Let us consider a firm that switches from an active state to a bankrupt state in, say, year 2013. We can try to predict that switch with 2012 data or, being more ambitious, we can try to predict it with 2011, 2010 or even 2009 data. We label these forecasting horizons as *one* to *four year prior* predictions, respectively. For firms that remain active for the whole sample period, we predict the firm's state at the last year available in the dataset using the same forecasting horizons. Of course, increasing the horizon should have a cost in terms of the quality of the prediction, measured by the number of prediction errors. The aim of this subsection is to report these costs.

A brief description on how prediction errors are usually reported in Machine Learning models is in order. A model is a classifier that assigns -or predicts- bankruptcy probabilities to each firm depending on the firm's characteristics. We have denoted $\hat{y}_i$ to be the predicted probability of bankruptcy for firm i, whereas the firm's actual state is either bankrupt or active, denoted as $y_i = 1$ and $y_i = 0$, respectively.[11] In order to assign these predictions to actual states, we define a probability threshold y^*, such that we say the model predicts bankruptcy for firm i or, equivalently, the model classifies firm i as bankrupt, whenever $\hat{y}_i \geq y^*$ holds. For a given y^*, the possible outcomes are usually structured in a *confusion matrix*, which as we define has predicted states by rows and actual states by columns.

In Figure 2.1.3 we present the confusion matrix for all four years of data. We set $y^* = 0.5$ as the cut-off probability threshold.[12] The model made a significant number of

[11] The standard notation is to denote 1 to the state that we positively want to identify.

[12] We set $y^* = 0.5$ since we apply a positive weighting scale to control the balance of positive and negative weights which is useful for imbalanced class distributions. Positive weight scale = sum(negative instances)/sum(positive instances)

correct classifications across all four years, correctly classifying firms as bankrupt and non-bankrupt, however it did make a number of misclassifications. The model made a relatively large number of Type I errors (1917 for one year prior) and Type II errors (309 for one year prior) over the sample period. Table 2.1.2 reports a number of standard summary statistics for classification problems for imbalanced data Bekkar et al. (2013). Roughly, Accuracy tells us overall how often the model is correct, but it is biased when -as in our case- a state is much more frequent than the other.[13] This motivates the use of state specific statistics: Sensitivity (also called Recall), Specificity and Precision. Sensitivity tells us when a firm is bankrupt, how often did the model correctly predict bankrupt. Specificity is similar, but for the active state. Precision tells us the proportion of correctly predicted bankrupt firms over the total number of bankrupt predictions or how many of the predicted bankrupt firms are actually bankrupt. F1 combines Precision and Recall using the Harmonic mean, thus its highest occurs whenever Precision is equal to Recall. The Matthew's Correlation Coefficient (MCC), is usually presented with a more complex formula.[14] To ease its interpretation, Table 2.1.3 shows its connection with the usual chi-squared statistic in a 2×2 contingency table. Its range is $[-1, 1]$, with -1 indicating a perfectly opposite classification and $+1$ indicating a perfectly correct classification while the center, 0, indicates perfect randomness. The statistics presented thus far are constructed under the threshold $y^* = 0.5$. The choice of this threshold depends on the relative weights between Type I and Type II errors: by increasing the threshold we become more strict when predicting bankruptcy, thus we reduce Type I errors, but at the cost of increasing Type II errors. The last two rows in Table 2.1.2 are linked to Figures 2.1.4 and 2.1.5 presented next.

[13] Consider a sample with 99 active firms and just one bankrupt firm. If we predict 100 active firms, we have a high accuracy, though we completely fail to predict our target state.

[14] That formula being:

$$MCC = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

Where; *True Positive*, *True Negative*, *False Positive* (Type I error) and *False Negative* (Type II error) are denoted as TP, TN, FP, FN.

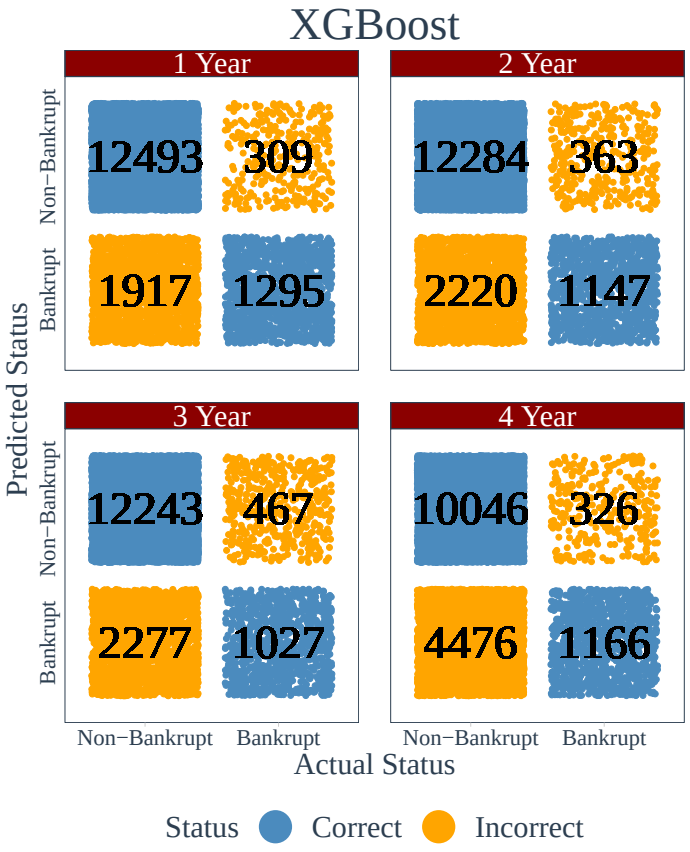


Figure 2.1.3: XGBoost: Confusion Matrix Graphic

Table 2.1.2: XGBoost: Confusion Matrix Analysis

Statistics: XGBoost

Metric	1 Year	2 Year	3 Year	4 Year
Accuracy	0.86	0.84	0.83	0.70
Sensitivity	0.81	0.76	0.69	0.78
Specificity	0.87	0.85	0.84	0.69
Precision	0.40	0.34	0.31	0.21
F1	0.54	0.47	0.43	0.33
MCC	0.51	0.43	0.38	0.29
AUC	0.84	0.80	0.77	0.74
AUPRC	0.66	0.55	0.50	0.42

Accuracy	$\frac{TN+TP}{Total}$	Sensitivity, Recall	$\frac{TP}{TP+FN}$	Specificity	$\frac{TN}{TN+FP}$
Precision	$\frac{TP}{TP+FP}$	F1	$H_{Precision,Recall}$	MCC	$\left(\frac{\chi^2}{Total}\right)^{1/2}$

Table 2.1.3: Notation: Total is a summation of all four quadrants, $H_{x,y}$ is the Harmonic mean of x, y, χ^2 is the chi-square statistic in a 2×2 contingency table.

The ROC curves (Receiver Operating Characteristic), in Figure 2.1.4, emphasizes the trade off between Type I and Type II errors by plotting Sensitivity (vertical) vs. the complementary Specificity (horizontal), also called false positive rate, defined $FPR = 1 - Specificity$. Each curve represents a different forecasting horizon.[15] In order to evaluate

[15]Each point within a curve corresponds to a value of a threshold $y^* \in [0, 1]$, the extreme points (0,0) and (1,1) are for $y^* = 1$ and $y^* = 0$, respectively. The performance of the model is better the closer the ROC curve is to the left and upper contour of the figure, while a purely random classifier would sit on the diagonal line.

the overall performance of the model in a single number, taking into account each threshold decision, the Area Under the Curve or AUC is calculated which is reported in the last but one row in Table 2.1.2, with values of 1 indicating perfect classification.

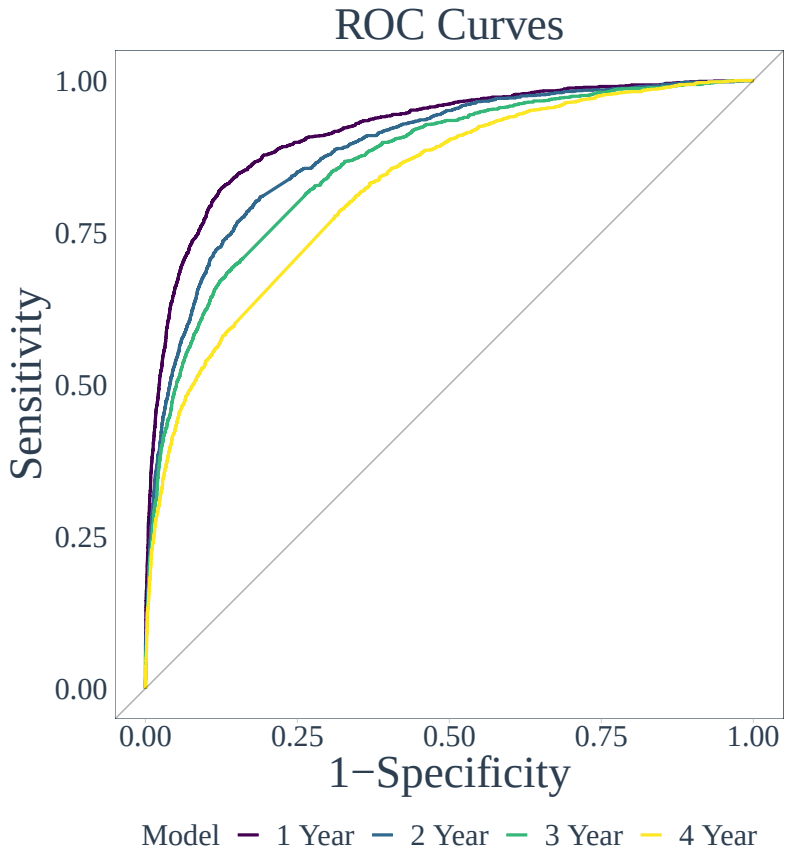


Figure 2.1.4: Receiver Operating Characteristic Curves

The ROC curve is a popular measure when evaluating binary classification models, however caution must be made when the data structure is of an imbalanced nature. CROC (Concentrated ROC) and CC (Cost Curves) have been suggested as alternatives to ROC curves, however ROC curves are much more widely used. An alternative is the Precision-Recall Curve (PRC) in Figure 2.1.5 and is considered a more informative measure than the ROC, CROC and CC plots for binary classification with an uneven class distribution

since the PRC is the only plot which changes in relation with the ratio of positives and negatives, see Saito and Rehmsmeier (2015). The Precision-Recall curves plots for every threshold $y^* \in [0, 1]$ the ratio between the Precision and Recall and in order to evaluate the model across all thresholds in one number the AUPRC (Area Under the Precision Recall Curve) is calculated which corresponds to the last row in Table 2.1.2. The AUPRC takes on values between 0 and 1 with values closer to 1 indicating perfect accuracy, which would be indicated by a line running horizontally along the top of Figure 2.1.5.

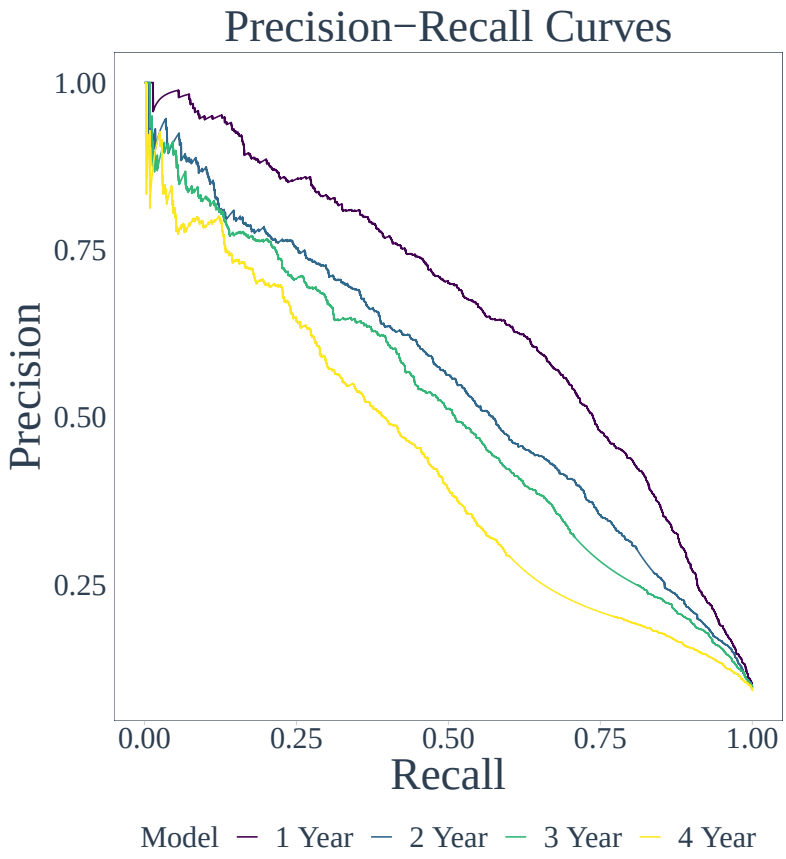


Figure 2.1.5: Precision-Recall Curves

Figure 2.1.6 plots the distribution of predicted probabilities for each of the years. The threshold was set to $y^* = 0.5$ so all firms above this threshold were assigned a 1 for bankrupt

and all firms below were assigned a 0 for non-bankrupt. The colours indicate the true status of the firm in the test data, blue being non-bankrupt firms and red being bankrupt firms. The blue shading, above the $y^* = 0.5$ threshold corresponds to misclassified non-bankrupt firms (false positives - Type I error) and the red shading, below the $y^* = 0.5$ threshold corresponds to the misclassified bankrupt firms (false negatives - Type II error). It is evident that the density of the XGBoost model predicted probabilities is more compact at the upper and lower tails of the distribution the closer we are to the bankruptcy event, the models then become wider with spikes appearing around the $y^* = 0.5$.

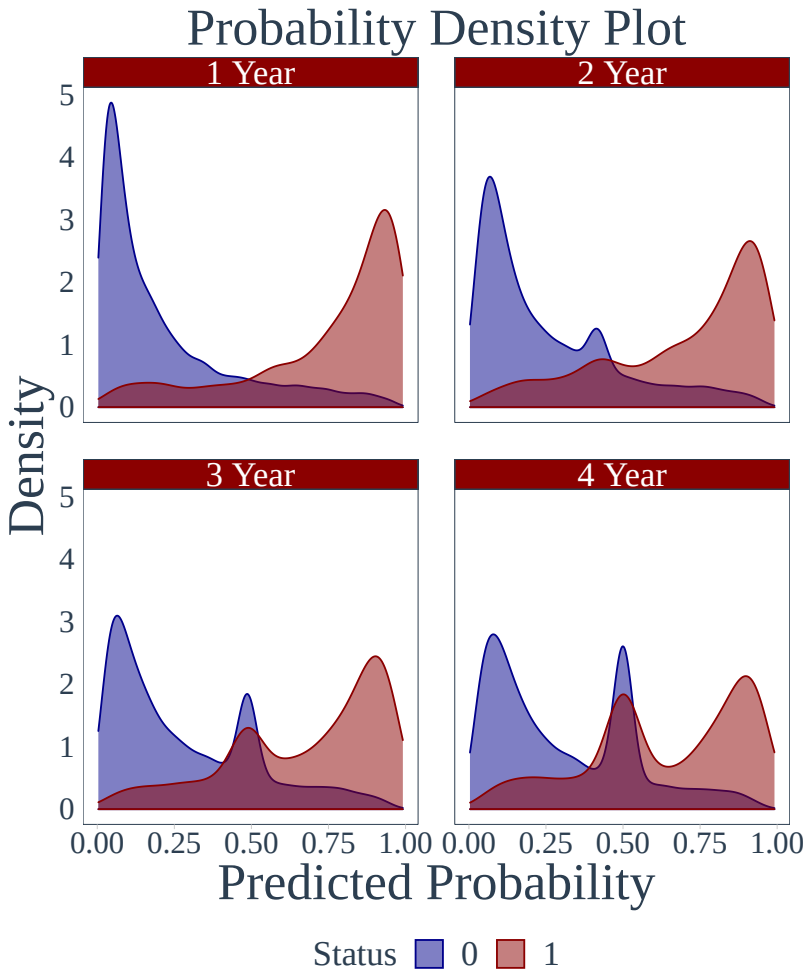


Figure 2.1.6: Probability Density plots

2.1.5.2 Variable importance

After decision tree based models are constructed it is possible to obtain variable importance scores. These scores indicate how useful each variable was when constructing each of the boosted decision trees in the model. It takes the contribution each variable made at each tree in the model and measures the relative contribution over all trees. Higher values indicate that the variable contributed more to the overall *Gain* over all trees. The variable importance plot takes an average of all the *Gains* each variable contributed at each tree. Figure 2.1.7 plots the variable importance plot for each of the models.

The most important variables found in the model for all four years of data were TL.TA, logTA, logSALES, CL.FinExp and EBIT.FinExp. That is, these variables contributed the most to the overall model prediction relative to other variables. The model found that a leverage ratio (TL.TA), size factors (logTA) and (logSALES), a debt to interest expense ratio (CL.FinExp) and an interest coverage ratio (EBIT.FinExp) were among the determining factors when making a decision on bankruptcy.

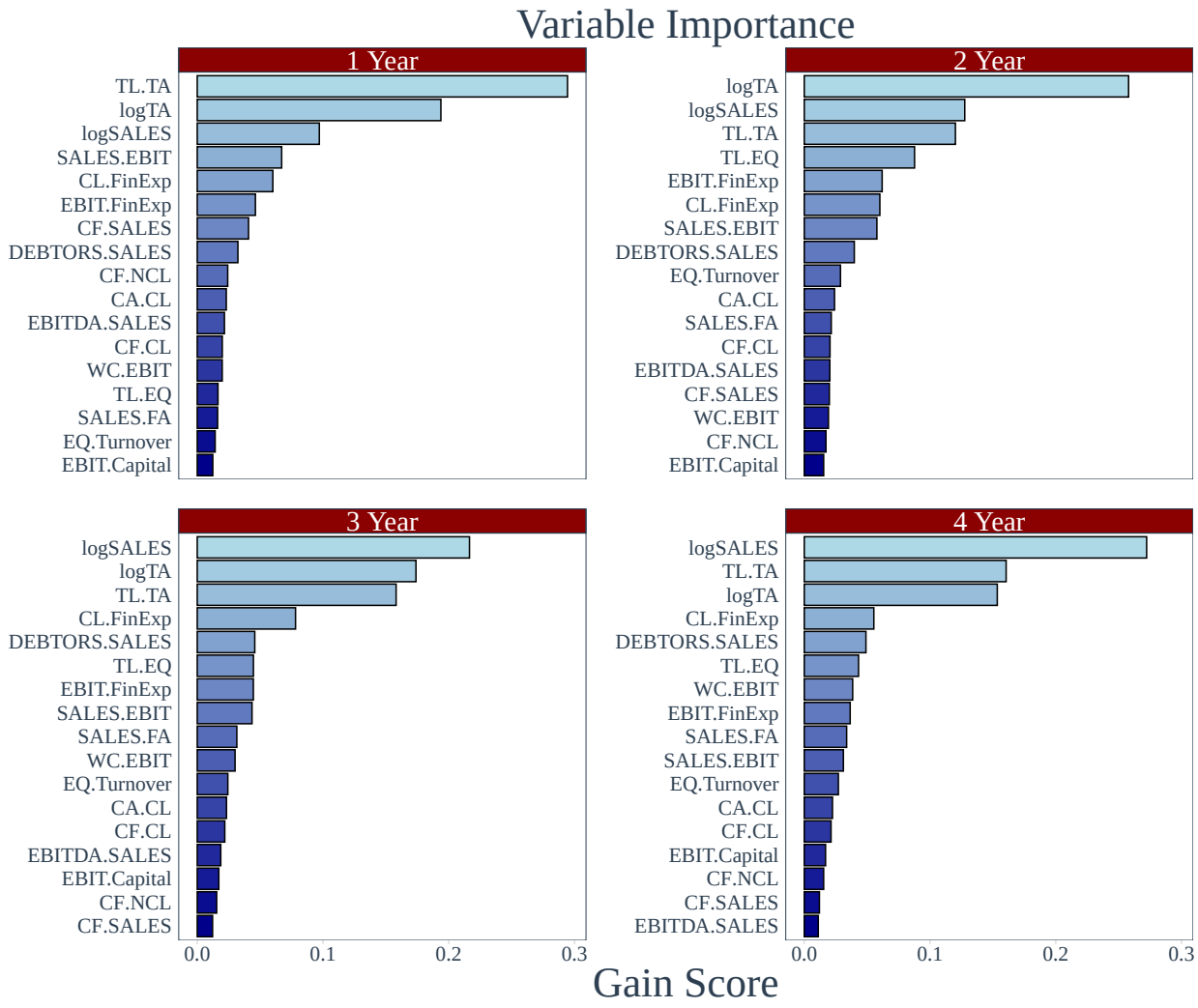


Figure 2.1.7: Variable importance plots

It is important to remark that Figure 2.1.7 reports the *aggregated and unconditional* impact of each variable on the prediction of bankruptcy. There are two basic differences in measuring this impact with Machine Learning models with respect to measures provided by estimates in linear models. First, in our model, the effect of each variable might be non-

monotonic. Second, in our model the marginal contribution to the score of a given variable will generally differ across firms, as this marginal contribution depends on other characteristics of the firms under consideration. Figure 2.1.8 illustrates how the log odds scores changes with respect to Total Liability to Total Assets (TL.TA). The non-monotonicity of the ratio Total Liability to Total Assets (TL.TA) shows that as the ratio approaches 0.75 and above, the log-odds score goes from being negative to positive. That is, low values of TL.TA had a negative impact on the prediction of bankruptcy, i.e. most of these companies assets are financed through equity. Higher values of the same ratio suggest most of the companies assets are financed through debt and are therefore given positive log-odds scores, thus positively contributing to the prediction of bankruptcy. The *blue* points indicate a bankrupt firm and *yellow* points a non-bankrupt firm. Many of the *blue* dots appear at the upper end of the TL.TA ratio.[16]

Figure 2.1.12, in the Appendix, shows the analogous plots for all variables and ratios under consideration. Some additional comments on other ratios are in order. The current ratio (CA.CL) shows a lot of dispersion when companies have a CA.CL ratio around 0 with the variable contributing negative log-odds scores (~ -0.8) and positive log-odds scores ($\sim +0.5$). Moreover, when the ratio begins to grow greater than 1.0 the log-odds scores begin to fall and thus firms with these ratios have a negative contribution to the prediction of bankruptcy. This is intuitive since ratios greater than 1.0 is a desirable situation to be in since it means the Current Assets $>$ Current Liabilities and therefore firms can more readably pay off its short-term debt obligations with its short-term assets. The EBITDA.SALES is an operating profitability and cash flow ratio indicating the percentage of earnings remaining once operating expenses have been accounted for. Values equal to 1 highlight that a company has no interest, no taxes, no depreciation and no amortisa-

[16]Note: The figure is dominated by *yellow* dots due to the greater number of non-bankrupt firms in the data. The data for this plot were filtered in order to remove the top and bottom 10% of extreme outliers which distorts the graphics axes and renders the plots unreadable, thus the reader should be mindful that there are points which lie outside of the plot region.

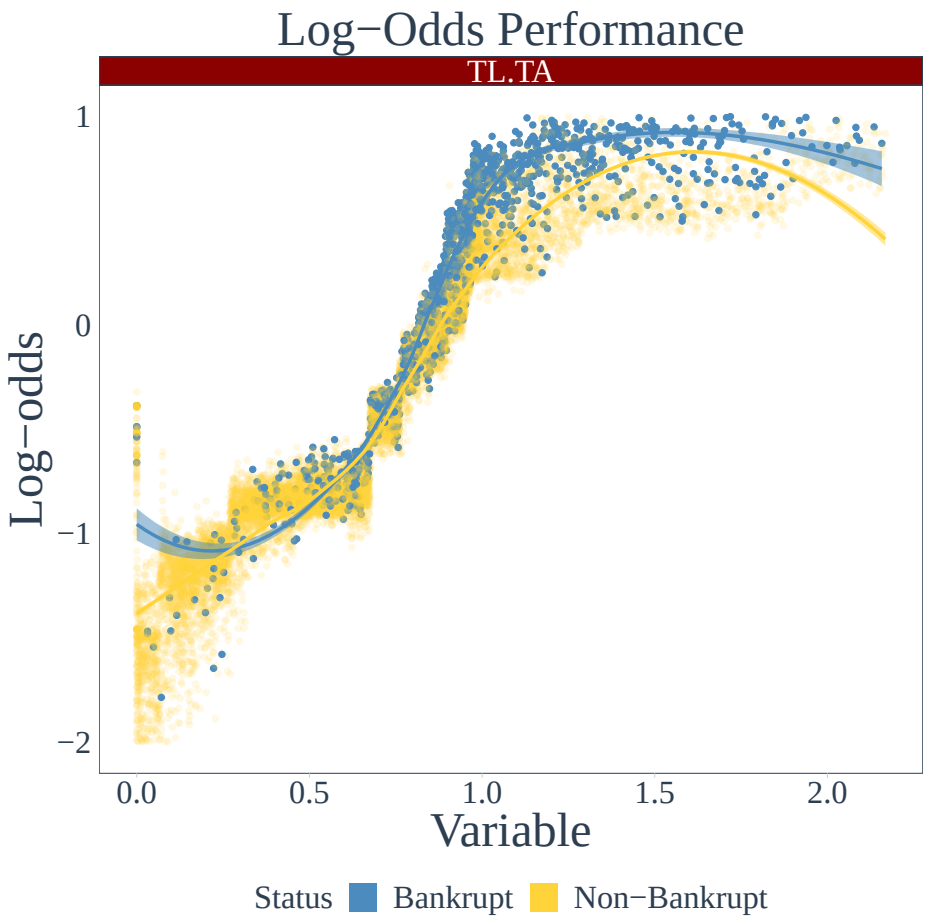


Figure 2.1.8: How log-odds scores are affected by changing variable values.

tion. Thus, the ratio shows the percentage of revenue left once the company has paid its operating expenses and therefore values closer to 1 suggest a healthier firm. The model assigns negative log-odds scores as the ratio tends to 1 and positive log-odds scores when the ratio tends to 0. Moreover the model finds that ratios $>\sim +0.25$ have a negative impact when predicting bankruptcy and ratios $<\sim +0.20$ have a positive impact when predicting bankruptcy. The model is able to learn a number of interesting characteristics which are used by analysts and have been known in the literature for a number of decades.

2.1.5.3 Case Studies

A decision tree is interpretable but not very good at prediction, Extreme Gradient Boosting (an ensemble of decision trees) is very good at prediction, however trying to interpret all of the individual decision trees in a model is simply not feasible. The following subsection allows us to interpret the XGBoost model. What sets this model (along with other tree models) apart, from other traditional black-box Machine Learning models is that it is possible to see how each variable contributes to the overall prediction for each observation or firm in the model. There are four possible cases, each representing a different position in the confusion matrix. In the main text we present a True Positive (TP) and a False Positive (FP) case, while the TN and FN cases are left to Figure 2.1.13 in the Appendix.

True positive (TP). Figure 2.1.9 shows the breakdown of how a positive case (bankruptcy) was correctly predicted. Given a particular variable, shown in the x-axis, a log-odds score is calculated (displayed inside each box), the sum of the log-odds scores are added up in order to obtain a final log-odds result (displayed in the final black box) and then a logistic function is applied to the result in order to obtain a predicted probability (shown on the y-axis). The horizontal line demonstrates a $y^* = 0.5$ probability cut-off threshold previously defined. Firms above this line are classified as bankrupt and firms below this line are classified as non-bankrupt. Notice, that the final log-odds prediction score is 4.58, which is assigned a predicted probability of bankruptcy $(1 + exp(-4.58))^{-1} = 0.990$.

False positive (FP). Figure 2.1.10 shows a firm that was incorrectly predicted as bankrupt. The model incorrectly predicted that the firm would be bankrupt with a final log-odds score of 0.02 and a subsequent bankruptcy probability of $(1 + exp(-0.02))^{-1} = 0.505$, sitting just above the cut-off threshold $y^* = 0.5$. It is possible that a more informed decision can be made by financial institutions through the use of these plots as opposed to other black-box Machine Learning models which would essentially make a decision simply based

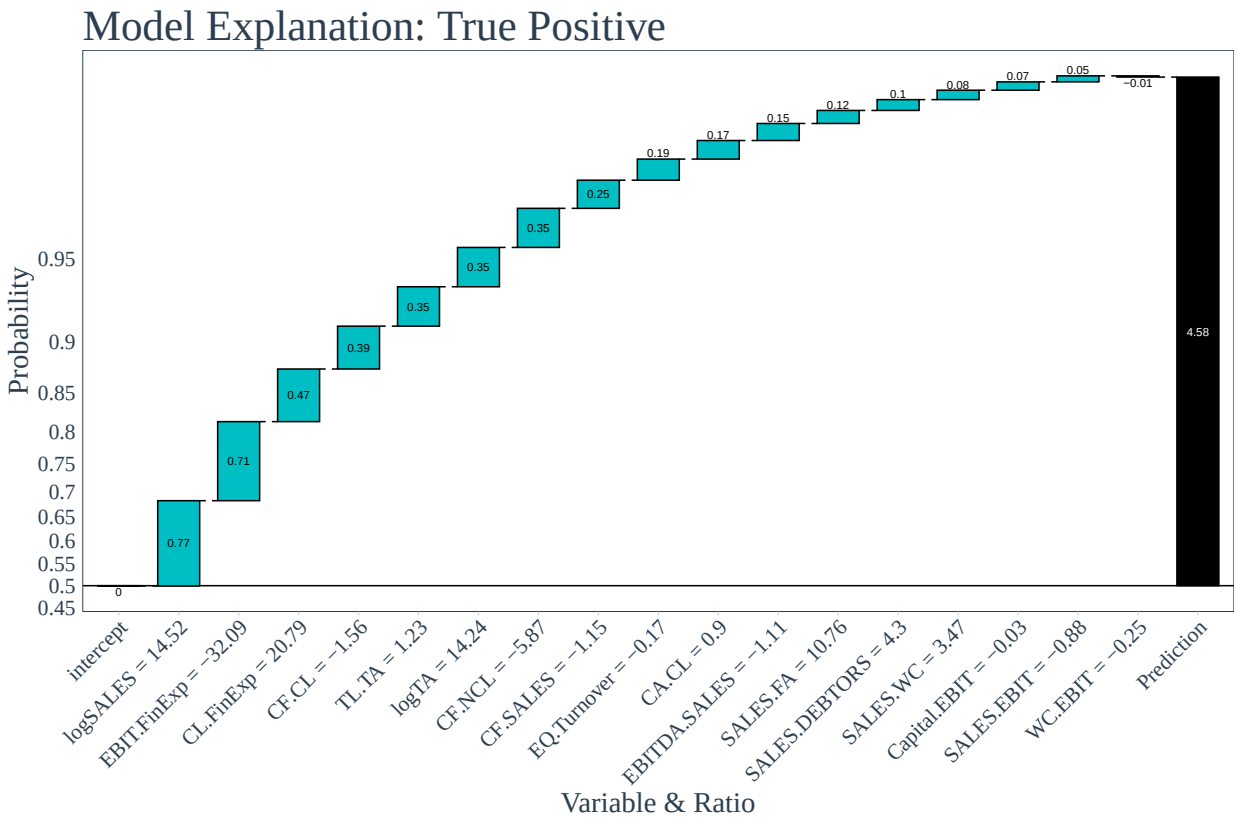


Figure 2.1.9: True Positive: Firm correctly predicted to be bankrupt case

on this firm lying on the wrong side of a cut-off threshold.

Table 2.1.7 in the Appendix, contains the contributions of the five main variables to the score of the firms for the four case studies. The table summarises some points that have been raised in the previous discussion. First, the relative weight of different variables changes across firms. For instance, the TL.TA makes the 5th highest contribution to the TP case but it makes the highest in the FP case. Second, the TP and TN cases are very clear. Recall that a contribution to the score increases the bankruptcy probability whenever it is positive. In the TP case, the table shows that all the contributions are positive. Analogously, the TN case is very clear with its main contributions being negative. In contrast, the FP and FN case have both positive and negative signs in their main contributions, and in

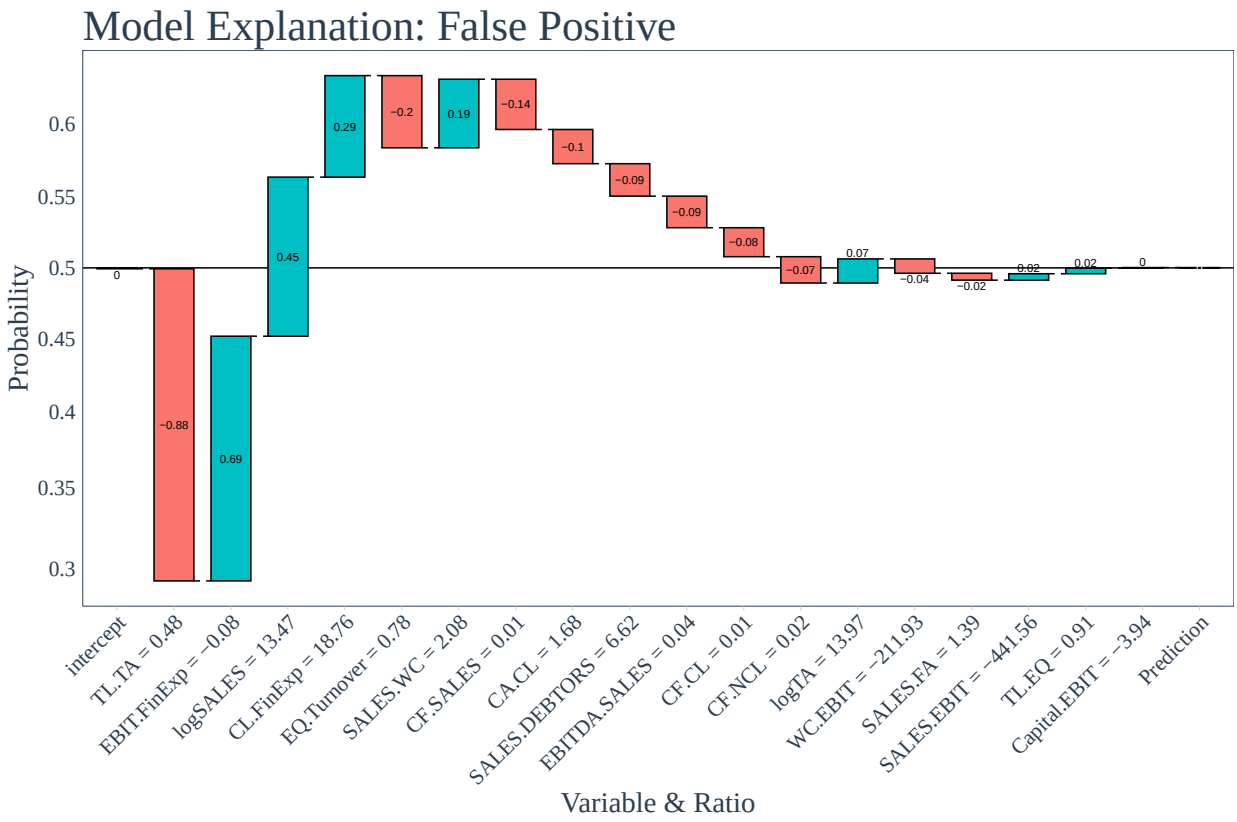


Figure 2.1.10: False Positive: Firm incorrectly predicted to be bankrupt case

both cases, the overall probability sits very close to the cut-off threshold. Moreover, unlike in linear models, the marginal contribution of each variable is not independent of the other variables. In other words, within a row, there is no linear relation between the values of the variable and its contribution to the score. The relation is not monotonic, as for instance the EBIT.FinExp row shows.

Overall these plots allow for a deeper understanding of how and why the model made a given prediction. The four cases can be linked to the log odds performance figures. Consider the TN case (in Table 2.1.7 in the Appendix) and the TL.TA variable, with a value of 0.02. It gives the second-highest marginal contribution to the prediction of bankruptcy for this firm (negatively contributing to the prediction of bankruptcy by assigning a log-odds score

of -1.08). This variable would sit as a yellow point in the lower left corner of the TL.TA log-odds plot in Figure 2.1.8 where points in this space negatively contribute to bankruptcy prediction. Contrast that with the TP case with a TL.TA ratio of 1.23 in which the model assigns a log-odds score of 0.35. This would sit as a blue point in the upper right segment of the TL.TA plot in Figure 2.1.8. Moreover, the other ratios can be interpreted and analysed in a similar manner where we are able to analyse the relationship between the log-odds scores that the model assigns and the values for each variable, Figure 2.1.12 in the Appendix shows the log-odds performance for all variables in the model.

2.1.6 Comparison to other Machine Learning models

This section compares XGBoost to other Machine Learning models. We removed all missing values, extreme values and centred and scaled the data.[17]

We compare XGBoost with: two different Neural Networks using the Keras API with TensorFlow back-end, see Chollet et al. (2015) and Abadi et al. (2015), a Support Vector Machine with a linear kernel and a radial kernel, see Cortes and Vapnik (1995) and Boser et al. (1992), a Logistic Regression, a Random Forest, see Breiman (2001), and a Light Gradient Boosting Model (LightGBM), see Ke et al. (2017).

Table 2.1.4 presents the metrics based on the confusion matrix for each of the models for one year prior predictions. The overall accuracy for the linear models Logistic Regression and Support Vector Machine are markedly higher than the non-linear models XGBoost, LightGBM, Random Forest and Neural Networks since the linear models make significantly less bankrupt predictions than the non-linear model's and thus predicting that all compa-

[17] Scaling by $\frac{x - mean(x)}{sd(x)}$ since Neural Networks perform better when the data is centred and scaled, K-Means forms better clusters when the data is centred and scaled. Logistic models perform better with the exclusion of extreme values and all models with the exception of LightGBM and XGBoost cannot handle missing data points, thus we remove the observations with missing values.

nies are non-bankrupt will yield higher accuracy results when the data class is imbalanced, this can be seen in comparing the linear model's Specificity scores to that of the non-linear models Specificity. The linear models score significantly lower for the Sensitivity results than the non-linear models.

The XGBoost model makes significantly more bankrupt predictions than any of the other models, the LightGBM model also makes a significant amount of bankrupt predictions. A cost to this is that both models make more Type 1 errors or commit more False Positives. However, all of the other models - excluding XGBoost and LightGBM - make significantly more Type 2 errors or commit more False Negatives, that is, these models predict that a company is non-bankrupt when the companies actual status was bankrupt. These models can be seen as far more costly to lending institutions than boosted models.[18]

After removing missing values and correcting for outliers in order to compare the different Machine Learning models, the XGBoost model here compares relatively similar or only marginally better to the original model which included extreme values, missing values and outliers.

Metric	XGBoost	Logistic	Light GBM	Shallow NN[a]	Deep NN	R. Forest	SVM (R)[b]	SVM (L)
Accuracy	0.87	0.91	0.89	0.91	0.91	0.92	0.90	0.91
Sensitivity	0.73	0.25	0.61	0.32	0.30	0.37	0.16	0.18
Specificity	0.88	0.99	0.92	0.98	0.98	0.99	0.99	0.99
Precision	0.42	0.66	0.47	0.60	0.62	0.79	0.66	0.71
F1	0.54	0.36	0.53	0.42	0.41	0.51	0.26	0.29
MCC	0.49	0.37	0.47	0.39	0.39	0.51	0.30	0.33
AUC	0.81	0.81	0.76	0.65	0.64	0.68	0.58	0.59

[a] NN: Neural Network. [b] SVM: Support Vector Machine with (R) Radial, (L) Linear kernel.

Table 2.1.4: Comparison to other Machine Learning models for one year prior predictions.

[18] We also note that a well trained Neural Network model on sufficient enough data would also be able to make more bankruptcy predictions at least on par with the two gradient boosted models presented in this paper.

We take our analysis a step further in order to understand why XGBoost and LightGBM models made significantly more bankrupt predictions than all other models. We filtered the data down to a random sample of 100 observations and plot the decisions boundary for each model. Figure 2.1.11 plots the boundaries for different models regarding two variables TL.TA (vertical axis) and logSALES (horizontal axis). The hollow circles, common to all plots, represent the 100 observations the different models are trained on. The blue and red area (in fact it is a densely populated set of dots) represent synthetically created non-bankruptcy and bankruptcy states, respectively.[19]

Figure 2.1.11 complements the previous analysis by showing how different models accommodate the relationship among variables. The linear models make a crude linear classification decision boundary, ultimately misclassifying many observations. The Shallow Neural Network model is able to adapt slightly more.[20] The Random Forest (a bagging) model overfits the data and makes erratic decision boundaries. The LightGBM and XGBoost models use boosting and add a regularisation parameter when fitting each tree. These two models are able to make more robust and generalisable decision boundaries over linear models.

Finally, table 2.1.8 in the Appendix, shows the estimates of the logistic regression model. This table illustrates how Machine Learning methods can build upon classical econometrics. Consider, for instance, the estimated coefficient of the TL.TA ratio in the table, which is 0.976. According to Figure 2.1.7, that is unconditionally the most relevant variable for one year prior predictions. Its marginal effect, that is, summing along all other variables, is in fact non-monotonic, as Figure 2.1.8 shows. Furthermore, *augmenting the zoom* down to a firm-level view, the case studies presented in the previous section show that its impact

[19] We synthetically created new data points by creating a sequence - with increments of 0.01 - from the min(TL.TA) to the max(TL.TA) and also from the min(logSALES) to the max(logSALES) of the 100 randomly sampled observations. The model was trained on the 100 observations and tested on 40,000 synthetically created observations constrained by the bounds of the min/max of the TL.TA and logSALES variables. The decision boundary line was drawn at the frontier of the predictions.

[20] Trained with a Rectified Linear Unit (ReLU) and a Sigmoid activation function.

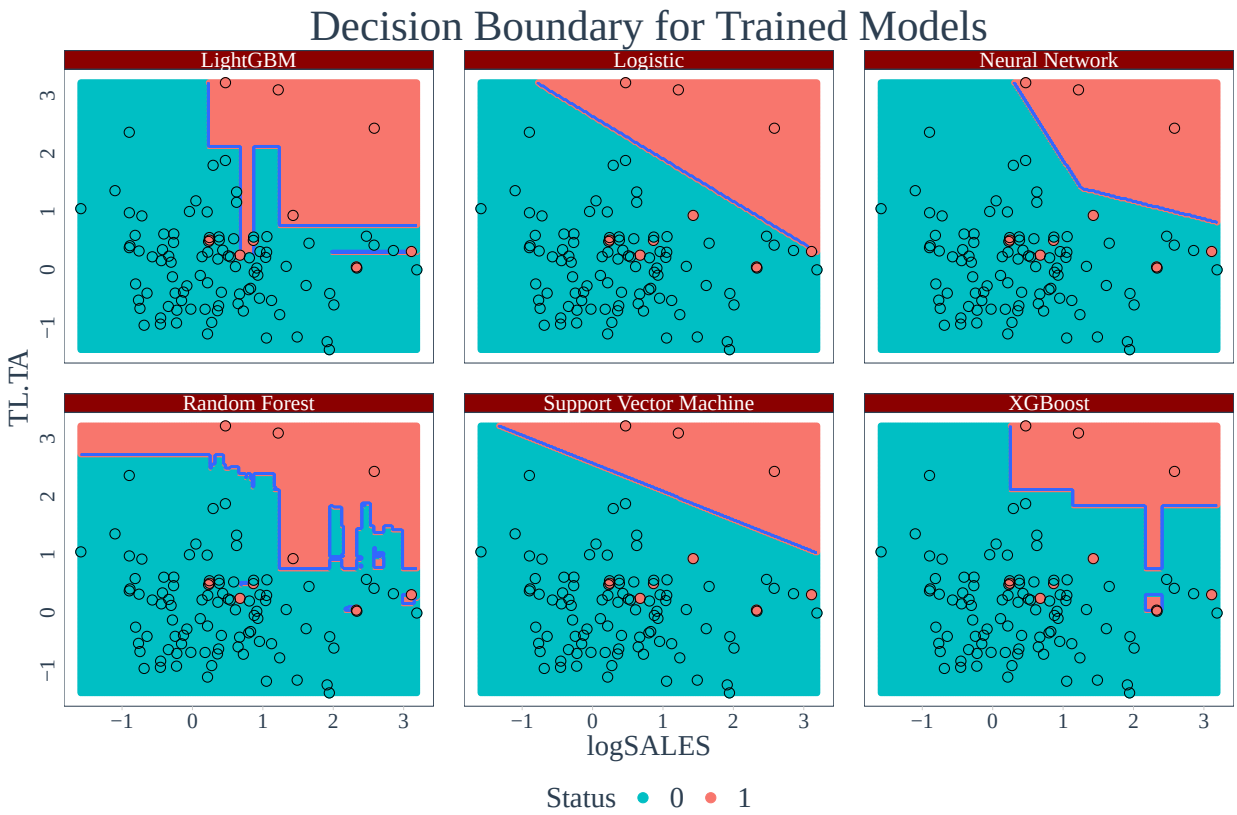


Figure 2.1.11: Decision Boundary

changes depending on other firms characteristics, in our examples it goes from 1.23 (TP case) to 0.48 (FP), see Figures 2.1.9 and 2.1.10, respectively. Finally, its relationship with other variables, as logSALES, shown in Figure 2.1.11, follows highly non-linear boundaries.

Two additional comments on the comparison of XGBoost to other ML models are worth mentioning. First, XGBoost allows for individual firm-level interpretations. The United States Equal Credit Opportunity Act (ECOA)[21] prohibits creditors from discriminating against any credit applicants and mandates that credit lenders give reasoning for any credit rejection. Moreover, models such as the one proposed in this paper are useful for identifying such reasons for loan application rejections, whether that be for personal credit or corporate

[21] https://www.justice.gov/crt/equal-credit-opportunity-act-3

credit denial, banks are able to identify reasons why an application should or should not be given credit in a way other models are unable to do so. Second, XGBoost allows for a systematic or optimized treatment of complexity. It allows the practitioner to set the price of complexity in terms of accuracy within a loss function, then the algorithm chooses sequentially the loss-minimizing elements in a function space given these prices. Thus, putting things together, it is about how rather than what to predict.

We have performed some additional robustness analysis on the comparison across methods. We have computed the McNemar's and Cochran's Q tests. In addition, since some models might be sensitive to variable selection, we have applied different regression models for variable selection (Stepwise Logistic, LASSO and RIDGE), then we have run our analysis again after removing the variables that the regression models had flagged as irrelevant. Finally, we have also done some optimization on hyper-parameter values for each model. Our results show quantitatively similar conclusions to the presented results here. In a nutshell, in terms of predictive capacity, XGBoost does at least as good as any other method, while it also delivers very interpretable outputs for economic applications. Additionally, it also requires less treatment of the data.[22]

2.1.7 Conclusion

This paper applies a state of the art Machine Learning algorithm, specifically XGBoost, in order to classify firms as either being bankrupt or non-bankrupt. We use annual financial statements of 58,000 Spanish companies from 1992 to 2016 collected from "Sistema de Análisis de Balances Ibéricos", around 6000 of which went bankrupt at some point during the study period.

[22]We especially thank an anonymous referee for specific and very helpful suggestions on making the comparison across Machine Learning methods more robust. Some supplementary material with details on this analysis will be made available by the corresponding author upon request.

We find that the ratio Total Liabilities to Total Assets (TL.TA), Current Liabilities to Financial Expense (CL.FinExp), Earnings Before Interest and Tax to Financial Expense (EBIT.FinExp), the logarithm of Total Assets (logTA) and the logarithm of Sales (logSALES) were consistently ranked amongst the most important variables for all four years of financial accounts when determining the state of bankruptcy. That is, a leverage ratio, a debt to interest expense ratio and an interest coverage ratio along with two size factors were seen to contribute more than other variables when classifying firms as bankrupt or not. An interesting component of tree-based models is that we can track the marginal contribution of each variable for each individual firm. We note that these marginal contributions can be different across firms and thus we present case studies for illustration.

We also quantify the cost of extending the forecasting horizon, that is, we aim to predict bankruptcy up to four years before the event itself. Our analysis yields a slight drop in performance the further out from the event we go, which is to be expected, however, the model was still able to correctly classify bankrupt and non-bankrupt firms and remain consistent throughout the time horizon. In this regard, it should be noted that the sample period under consideration includes some years of a deep economic recession within the Spanish economy, XGBoost may be able to capture these non-linearities better than traditional models.

Finally, we remove all missing values and compare XGBoost to other Machine Learning models, such as Support Vector Machines, a Logistic model, Neural Networks, Random Forest and LightGBM. Generally, XGBoost significantly outperforms the Logistic model over a wide range of performance criteria. XGBoost and other Machine Learning models deliver roughly a similar capacity to systematically capture dependencies that vary across firms, particularly in populations -as in our dataset- with an imbalanced response variable.

2.1.8 Appendix

2.1.8.1 Variable description and Summary Statistics

Variable	Definition and Ratio
CA.CL	The current ratio to determine a companies ability to pay its short-term debt obligations (i.e. how a company's cash or soon to be cash items can be used to pay its short-term obligations within a year). $$\text{CA.CL} = \frac{\text{Current Assets}}{\text{Current Liabilities}}$$
CF.CL	An operating cash flow liquidity ratio measuring how current liabilities are being covered by the cash flows generated from operations. Ratios above 1 show that a company has generated more cash than what is required to pay it's current liabilities in the same period. Low ratios might indicate that a company needs more capital. $$\text{CF.CL} = \frac{\text{Cash Flow}}{\text{Current Liabilities}}$$
CF.NCL	A cash flow liquidity ratio with a longer term horizon, measuring the companies ability to pay long-term debts with the cash generated from operations in the current period. $$\text{CF.NCL} = \frac{\text{Cash Flow}}{\text{Non-Current Liabilities}}$$
CF.SALES	A operating ratio showing a company's ability to turn its sales into cash. Low ratios may suggest a change in the terms of sales or inefficient management of accounts receivables. $$\text{CF.SALES} = \frac{\text{Cash Flow}}{\text{Sales}}$$
CL.FinExp	A debt to interest payments ratio measuring the rate of interest a company is paying on its short term debt obligations. $$\text{CL.FinExp} = \frac{\text{Current Liabilities}}{\text{Financial Expenses}}$$

Variable	Definition and Ratio
DEBTORS.SALES	A liquidity ratio measuring how much a company's sales occur on credit. A high ratio can be a negative indicator to debt providers, since it suggests that the company operates with high credit sales and therefore compromise the company's ability to pay back its interest payment obligations, since the money is tied up in credit and not cash. $$\text{DEBTORS.SALES} = \frac{\text{Debtors}}{\text{Sales}}$$
EBIT.Capital	Return on Capital Employed (ROCE) states the amount of capital & equity the company has used to generate its profits. $$\text{EBIT.Capital} = \frac{\text{EBIT}}{\text{Capital Employed}}$$
EBIT.FinExp	An interest coverage ratio which measures a company's ability to pay back interest on its outstanding debt. High ratios indicate a company can more easily pay back its interest. $$\text{EBIT.FinExp} = \frac{\text{EBIT}}{\text{Financial Expenses}}$$
EBITDA.SALES	EBITDA margin ratio. A profitability ratio showing the amount in which a company expects to receive after operating costs have been paid. Higher values indicate that efficient processes have kept expenses low which in turn keeps earnings high. $$\text{EBITDA.SALES} = \frac{\text{EBITDA}}{\text{Sales}}$$
EQ.Turnover	A ratio to determine whether a company is creating enough turnover to justify continued operations for its shareholders. $$\text{EQ.Turnover} = \frac{\text{Shareholders Equity}}{\text{Turnover}}$$

Variable	Definition and Ratio
logSALES	A proxy variable to measure firm size. Adjusted for 2016 inflation levels using the Spanish CPI index $$\text{logSALES} = log(\text{Sales})$$
logTA	A proxy variable to measure firm size. Adjusted for 2016 inflation levels using the Spanish CPI index $$\text{logTA} = log(\text{Total Assets})$$
SALES.EBIT	A profitability ratio indicating the percentage of a company's earnings remaining after operating expenses and before interest and tax expenses have been considered. $$\text{SALES.EBIT} = \frac{\text{Sales}}{\text{EBIT}}$$
SALES.FA	Fixed Asset Turnover ratio which measures a firms operating performance and efficiency. It is a measure of a company's ability to generate sales from its fixed asset investments such as, property, plant and equipment. Higher ratios indicate that a company has used its fixed asset investments to generate sales more effectively. $$\text{SALES.FA} = \frac{\text{Sales}}{\text{Fixed Assets}}$$
TL.EQ	A gearing ratio measuring how a company finances its operations through debt or through the shareholders own funds and reflects the ability of a businesses needing to cover outstanding debt obligations through its shareholders. $$\text{TL.EQ} = \frac{\text{Total Liabilities}}{\text{Shareholder's Equity}}$$

Variable	Definition and Ratio
TL.TA	A leverage ratio measuring a companies ability to use its assets to pay off its liabilities. The higher the ratio the higher the degree of leverage. $$\text{TL.TA} = \frac{\text{Total Liabilities}}{\text{Total Assets}}$$
WC.EBIT	Working Capital (Current Assets - Current Liabilities) A short-term liquidity to earnings ratio. $$\text{WC.EBIT} = \frac{\text{Working Capital}}{\text{EBIT}}$$

Table 2.1.5: Definitions of financial ratios

Table 2.1.6: One year Summary Statistics[a]

	Mean		SD		Median		Kurtosis		Skewness		Missing	
Variable	0	1	0	1	0	1	0	1	0	1	0	1
CA.CL	92	27	4,141	1,395	1.40	0.97	21,388	5,813	133	76	1,034	53
CF.CL	11	3	564	78	0.17	0.12	12,553	3,886	103	59	1,981	170
CF.NCL	34	22	3,089	1,079	0.30	0.26	34,066	4,791	183	69	22,380	1,170
CF.SALES	1,402	22	318,015	1,017	0.07	0.09	51,606	3,841	228	61	6,392	736
CL.FinExp	3,464	3,050	200,765	82,437	52	23	33,931	3,103	179	53	14,937	522
DEBTORS.SALES	1.90	5	66	100	0.17	0.27	7,917	2,232	83	46	9,347	787
EBIT.Capital	9	18	92	138	1.10	1.80	11,109	2,177	91	44	1,393	96
EBIT.FinExp	612	253	13,370	4,469	8	3	4,645	2,367	62	44	14,681	511
EBITDA.SALES	3	20	239	917	0.08	0.11	48,307	3,902	217	61	6,382	730
EQ.Turnover	559	269	110,990	15,484	0.36	0.23	53,100	5,058	231	71	4,601	524
logSALES	13	14	1.80	1.80	13	14	5	5	-0.03	-0.68	6,364	683
logTA	13	15	1.60	1.70	13	15	4	5	0.72	-0.28	12	0
SALES.EBIT	155	76	8,179	1,068	16	10	31,708	3,429	166	55	6,389	686
SALES.FA	79	96	2,710	2,089	4	4	19,967	2,313	132	47	8,450	798
TL.EQ	25	48	1,074	799	1.80	5	36,892	2,762	178	50	18	0
TL.TA	0.77	8	1.20	250	0.67	0.96	2,575	2,413	38	48	18	0
WC.EBIT	92	127	4,612	1,963	4	5	38,495	1,603	185	36	2,292	112

[a] One Year means the last year of available data for a firm, either before it turned into bankruptcy state, 1, or it did not, 0.

2.1.8.2 Variable importance

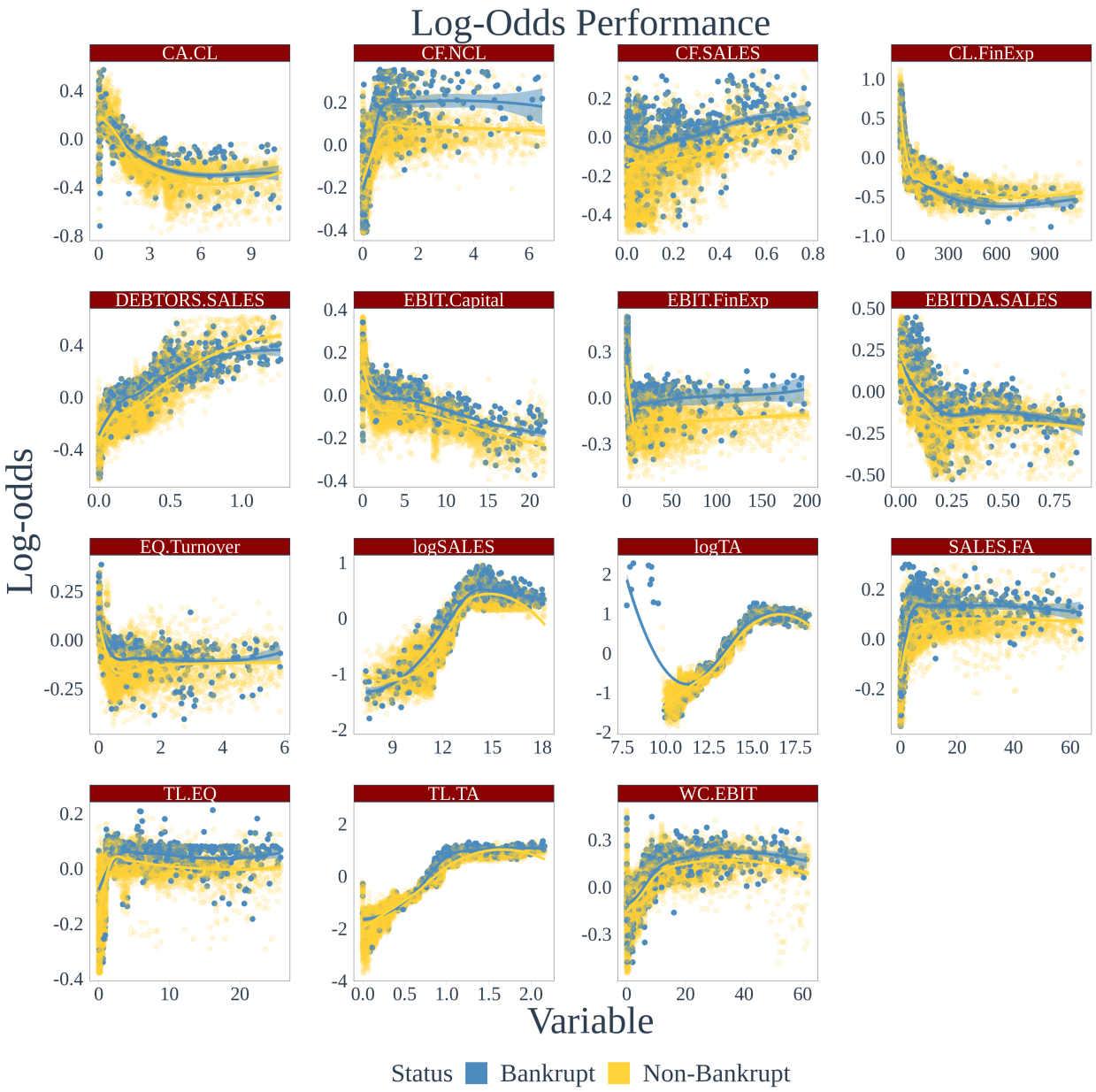


Figure 2.1.12: How log-odds scores are affected by changing variable values.

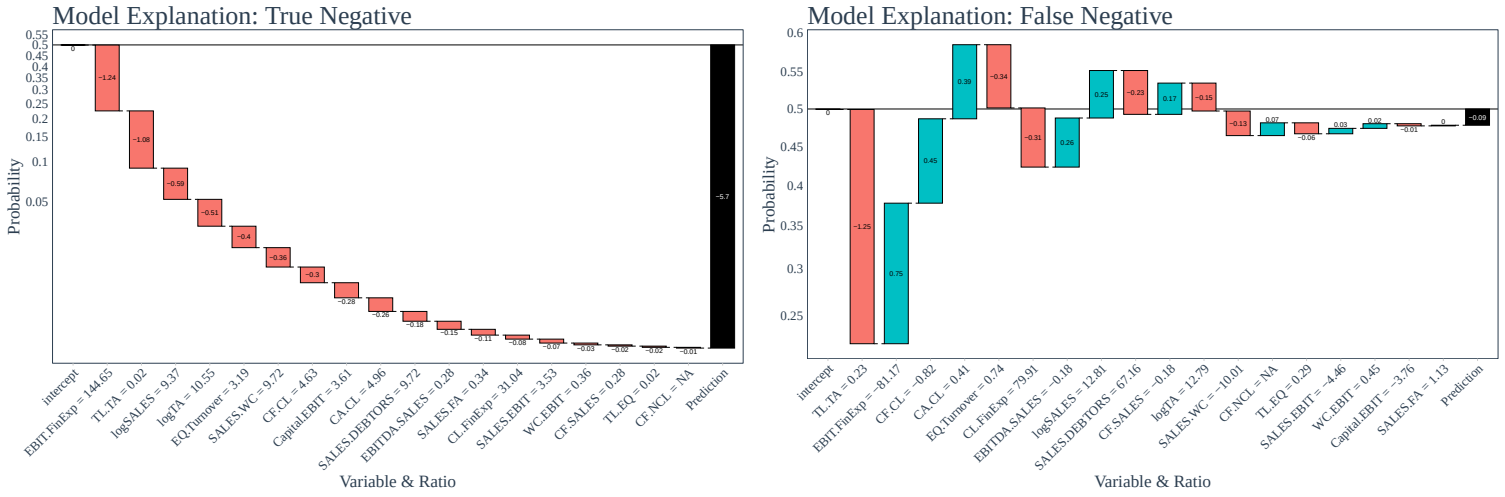


Figure 2.1.13: Two case studies: a True Negative (left) and a False Negative (right).

The left plot of Figure 2.1.13 shows a True Negative. The model assigns negative log-odds scores to each of its variables, indicating that these ratios negatively contribute to the prediction of bankruptcy. The probability of bankruptcy for this case is; $(1+exp(-(-5.70)))^{-1} = 0.003$. Perhaps a more costly scenario would be the False Negative case, which corresponds to the right plot in Figure 2.1.13 where the model predicts that a firm is a non-bankrupt firm and it turns out to be a bankrupt firm. This firm had a negative log-odds score of -0.09 and a subsequent probability of bankruptcy of $(1+exp(-(-0.09)))^{-1} = 0.480$. Given the fact that it sits a little below the threshold of y^* this firm was incorrectly classified as non-bankrupt.

Figure	2.1.9	2.1.13 (left)	2.1.10	2.1.13 (right)
Predicted Actual	TP Bankrupt Bankrupt	TN Active Active	FP (Err Type I) Bankrupt Active	FN (Err Type II) Active Bankrupt
TL.TA	5th (0.35) 1.23	2nd (-1.08) 0.02	1st (-0.88) 0.48	1st (-1.25) 0.23
logTA	6th (0.35) 14.24	4th (-0.51) 10.55	13th (0.07) 13.97	11th (-0.15) 12.79
logSALES	1st (0.77) 14.52	3rd (-0.59) 9.37	3rd (0.45) 13.47	8th (0.25) 12.81
CL.FinExp	3rd (0.47) 20.79	13th (-0.08) 31.04	4th (0.29) 18.76	6th (-0.31) 79.91
EBIT.FinExp	2nd (0.71) -32.09	1st (-1.24) 144.65	2nd (0.69) -0.08	2nd (0.75) -81.17
Prob	0.990	0.003	0.505	0.480

Table 2.1.7: Summary of contributions of TL.TA, logTA, logSALES, CL.FinExp, EBIT.FinExp for the different case studies. Cases are in columns while variables are in rows. For instance, the cell corresponding to logSALES and the TP case shows, for that firm, that variable makes the 1st (highest in absolute value) contribution to the score with 0.77, and the actual value of that variable for that firm is 14.52. A contribution to the score increases the bankruptcy probability whenever it is positive. Predicted probabilities are in the bottom line. The cut-off threshold value is 0.5.

2.1.8.3 Additional information on other ML methods

	Dependent variable:			
	Dependent Variable: Binary Status of Bankruptcy			
	1 Year	2 Year	3 Year	4 Year
TL.TA	0.976***	0.606***	0.463***	0.428***
	(0.059)	(0.060)	(0.060)	(0.064)
CA.CL	−0.253***	−0.053	−0.073	−0.016
	(0.087)	(0.070)	(0.065)	(0.065)
TL.EQ	0.257***	0.208***	0.209***	0.188***
	(0.037)	(0.035)	(0.035)	(0.037)
EBIT.Capital	−0.096***	−0.046	−0.053*	−0.014
	(0.035)	(0.032)	(0.032)	(0.032)
WC.EBIT	0.385***	0.331***	0.401***	0.387***
	(0.055)	(0.049)	(0.046)	(0.049)
EBIT.FinExp	0.512***	0.445***	0.285**	−0.304**
	(0.123)	(0.117)	(0.121)	(0.145)
SALES.EBIT	−0.631***	−0.434***	−0.435***	−0.512***
	(0.072)	(0.059)	(0.057)	(0.059)
CL.FinExp	−1.870***	−1.791***	−1.237***	−0.413***
	(0.201)	(0.191)	(0.158)	(0.123)
EQ.Turnover	−0.250**	−0.244***	−0.140*	−0.198**
	(0.098)	(0.084)	(0.080)	(0.090)
CF.NCL	0.106**	−0.094*	−0.059	0.014
	(0.047)	(0.051)	(0.043)	(0.042)
logTA	0.269*	0.111	−0.101	−0.460***
	(0.143)	(0.137)	(0.140)	(0.153)
logSALES	0.992***	1.079***	1.287***	1.680***
	(0.157)	(0.146)	(0.148)	(0.158)
CF.CL	−0.668***	−0.915***	−0.649***	−0.473***
	(0.097)	(0.102)	(0.091)	(0.087)
SALES.FA	−0.016	−0.029	0.051	−0.012
	(0.056)	(0.047)	(0.043)	(0.045)
CF.SALES	0.817***	0.595***	0.537***	−0.175
	(0.101)	(0.097)	(0.099)	(0.114)
EBITDA.SALES	−0.386***	−0.219**	−0.188*	0.459***
	(0.103)	(0.094)	(0.098)	(0.106)
DEBTORS.SALES	0.248***	0.346***	0.312***	0.233***
	(0.040)	(0.037)	(0.036)	(0.038)
Constant	−4.044***	−3.610***	−3.235***	−3.148***
	(0.092)	(0.083)	(0.071)	(0.066)
Observations	11,794	11,355	10,920	10,483
Log Likelihood	-2,743.099	-3,250.657	-3,353.721	-3,300.723
Akaike Inf. Crit.	5,522.198	6,537.314	6,743.442	6,637.446
Note:				*p<0.1; **p<0.05; ***p<0.01

Table 2.1.8: Logistic Regression Results for all years prior.

Chapter 2.2

Corporate bankruptcy prediction in Spain from 1992 to 2016: Supplementary material

Abstract: This sub-chapter provides supplementary material on the methodological comparison across Machine Learning models and a rolling windows extension. The purpose of this section was to answer external reviewers comments and questions as part of the submission process to the journal, Computational Economics.

JEL Codes: G17 (Financial forecasting and simulation), G33 (Bankruptcy), C53(Forecasting and prediction methods).

2.2.1 Introduction

This sub-chapter is an extension to the previous in that additional statistical and analytical comparisons are made to the bankruptcy dataset. These suggestions mainly came from trying to adequately answer external reviewers comments on the paper and consists of two rounds of external reviewer comments.

We firstly apply McNemar's and Cochran's Q tests to the confusion matrix results in order to see if there is any statistically significant difference between the Machine Learning models analysed. Additionally, we report the confusion matrix summary statistics presented

in the previous chapter. We do this for each of the 5 cross-validation folds and also the held-out test dataset. The results are consistent across the folds and on the held-out test set. As previously discussed, Machine Learning requires the optimisation of parameters, therefore, we next report the hyper-parameter values for each of the Machine Learning models used in the previous chapter along with their corresponding optimal values. We next present a discussion on the variable selection procedure, using Stepwise Logistic, LASSO, RIDGE and Elastic Net regression models to remove variables which contribute little to the overall prediction. We do this for each year prior to bankruptcy (1 year), (2 years) etc. and see if the results are consistent the further we go from the event of bankruptcy. We analyse the regression coefficients from the models along with the confusion matrix statistics and finally compare the results (after removing variables) to the original confusion matrix table in the previous chapter. We find that after removing the variables which obtain zero or close to zero coefficients from the regressions that the marginal impact on the models improvement in performance is minimal when compared with the original confusion matrix. We then further scrutinise the variable selection procedure by analysing which variables were previously used in the literature that we cited in the original paper, making sure that we used consistent, relevant and correct variables from the literature.

We analyse the data further by applying the XGBoost model to a rolling time series window using two methods (1) a fixed size training period and (2) a cumulative training period in order to see if there is a change in the predictions over time. That is, in the former case the training data has a constant size when moving through the years and in the latter case the training data increases in size as the model shifts through the years. We also graphically represent the rolling bankrupt and non-bankrupt companies to show the distribution over this time period, additionally, we plot the time series of bankrupt firms in the dataset. Finally, we present with an interpretation an XGBoost decision tree which is one of the trees which come from the XGBoost model output using the bankruptcy data.

2.2.2 Methodological comparison across models

This section presents some additional analysis on the comparison between the different models used in the paper. Two different tests are performed (McNemar's and Cochran's Q test) in order to see if there is a statistically significant difference between the Machine Learning models performances. We then report the hyper-parameter selection process along with each models optimal hyper-parameters. Variable selection models are then also applied and the performance metrics of each model is compared. Finally, an overview of the financial variables used in previous literature is conducted.

2.2.2.1 McNemar and Cochran tests

McNemar test. The McNemar's test checks the null hypothesis of equal proportion of correct classification against the alternative of different proportions. The McNemar's test analyses the homogeneity or - marginal homogeneity - of the contingency table by looking at the disagreements between two models. The test is widely used in medicine in order to compare the effect of treatment and control groups. When used in binary classification tasks, the test compares whether two models disagree in the same way (or not). It is not commenting on whether one model is more or less accurate or error prone than another, see Westfall et al. (2010). Consider Table 2.2.1 the cells $\mathcal{B}$ and $\mathcal{C}$ (the off-diagonal) tells us how the two models differ from each other, the McNemar's test statistic with ("chi-squared") continuity correction is then given as $\chi^2 = \frac{(|\mathcal{B}-\mathcal{C}|-1)^2}{\mathcal{B}+\mathcal{C}}$. The Null Hypothesis is that the classifications of $p(\mathcal{B})$ and $p(\mathcal{C})$ are the same whereas the Alternative Hypothesis is that they are not.

Tables 2.2.2 to 2.2.7 report the p-values from the paired comparison of each Machine Learning model for each of the folds and finally for the held-out test set. We can see that across each of the folds the XGBoost, Random Forest and LightGBM models are all statistically different from one another. Comparing the Logistic, SVM Radial and SVM Linear

		Model 2	Model 2
		Correct	Incorrect
Model 1	Correct	$\mathcal{A}$	$\mathcal{B}$
Model 1	Incorrect	$\mathcal{C}$	$\mathcal{D}$

Table 2.2.1: Comparing two Machine Learning classifiers

models, for the most part they are not statistically different at the 5% level across each of the folds. Therefore these three models have a similar distribution in their errors and there is not enough evidence to suggest that each of these models are different to each other in their prediction errors - i.e. each of the models tends to agree with each other on their predictions. It is natural to think that the SVM Radial and SVM Linear make similar predictions to each other since they come from the same model. Additionally, these three models are the most simple and they are predicting very few instances of *Bankruptcy* and are predicting that most of the firms are *Non-Bankrupt*, which misses the point of predicting the *rare-event* of bankruptcy. Looking at these models confusion matrix statistics later on for Specificity they are somewhat low and are roughly the same as each other. Comparing the Neural Network to the Random Forest model, across each fold the models are not statistically different from one another.

	XGBoost	Logistic	NN	RandomForest	SVMRadial	SVMLinear	LightGBM
XGBoost							
Logistic	0.82197						
NN	0.01675	0.02833					
RandomForest	0.00003	0.00013	0.22780				
SVMRadial	0.66804	0.92726	0.04530	0.00050			
SVMLinear	0.21152	0.13442	0.00163	0.000002	0.19360		
LightGBM	0.00871	0.13781	0.35523	0.00598	0.22780	0.00149	

Table 2.2.2: McNemar P-values for model combinations (fold 1), 2359 Obs.

Cochran's test. McNemar tests might be misleading when comparing an imbalanced

	XGBoost	Logistic	NN	RandomForest	SVMRadial	SVMLinear	LightGBM
XGBoost							
Logistic	0.64850						
NN	0.00008	0.00052					
RandomForest	0.00009	0.00075	0.81736				
SVMRadial	0.10130	0.06751	0.000000	0.000001			
SVMLinear	0.31049	0.16913	0.000004	0.00001	0.43951		
LightGBM	0.00035	0.03365	0.17698	0.15473	0.00003	0.00006	

Table 2.2.3: McNemar P-values for model combinations (fold 2), 2359 Obs.

	XGBoost	Logistic	NN	RandomForest	SVMRadial	SVMLinear	LightGBM
XGBoost							
Logistic	0.00924						
NN	0.000000	0.00109					
RandomForest	0.000001	0.01187	0.32839				
SVMRadial	0.05281	0.44352	0.00007	0.00086			
SVMLinear	0.00013	0.00001	0	0	0.000001		
LightGBM	0.00016	1	0.00239	0.00921	0.28884	0	

Table 2.2.4: McNemar P-values for model combinations (fold 3), 2358 Obs.

	XGBoost	Logistic	NN	RandomForest	SVMRadial	SVMLinear	LightGBM
XGBoost							
Logistic	0.12551						
NN	0.00200	0.03401					
RandomForest	0.00010	0.00638	0.60254				
SVMRadial	0.42801	0.63043	0.01623	0.00174			
SVMLinear	0.49964	0.43472	0.00360	0.00162	1		
LightGBM	0.03983	1	0.04876	0.00413	0.56628	0.52005	

Table 2.2.5: McNemar P-values for model combinations (fold 4), 2359 Obs.

dataset such as our bankruptcy prediction data. We can only state that the models have a statistically significant difference in their error distributions and not that a given model is better than another at detecting positives. Table 2.2.8 shows the Cochran's test for each of the folds and held-out test set. Notice that Cochran test does not perform pairwise comparison but all models in one summary statistic. The table shows the corresponding p-value for each fold.

	XGBoost	Logistic	NN	RandomForest	SVMRadial	SVMLinear	LightGBM
XGBoost							
Logistic	1						
NN	0.00937	0.00300					
RandomForest	0.00260	0.00179	0.70766				
SVMRadial	0.75946	0.85205	0.00400	0.00167			
SVMLinear	1	0.90052	0.01359	0.00515	0.69627		
LightGBM	0.00041	0.00764	0.92034	0.54429	0.00446	0.00337	

Table 2.2.6: McNemar P-values for model combinations (fold 5), 2359 Obs.

	XGBoost	Logistic	NN	RandomForest	SVMRadial	SVMLinear	LightGBM
XGBoost							
Logistic	0.23533						
NN	0.00001	0.00041					
RandomForest	0	0.000000	0.05092				
SVMRadial	0.92262	0.25273	0.000005	0			
SVMLinear	0.40679	0.67499	0.00006	0.000000	0.38228		
LightGBM	0.00212	0.37489	0.00588	0.000001	0.01906	0.17090	

Table 2.2.7: McNemar P-values for model combinations (Held-Out Test)

	Q	DegFreedom	P-Value
Fold 1	40.45	6	0.00000037
Fold 2	69.63	6	0.00000000000048
Fold 3	105.29	6	0
Fold 4	30.94	6	0.000025
Fold 5	33.93	6	0.0000069
Test	89.06	6	0

Table 2.2.8: Cochran's Q tests across folds

Finally in this subsection, in order to show the performances between models, the table 2.2.9 shows summary statistics from the confusion matrix across each of the folds along with the held-out test set. The results for each of the Machine Learning models are consistent over each of the cross validation folds. Additionally, the held-out test set is also consistent with the results found from the 5-folds.

	XGBoost	Logistic	NN	RandomForest	SVMRadial	SVMLinear	LightGBM
Fold1	Total Obs	2359					
Accuracy	0.90	0.90	0.91	0.92	0.90	0.89	0.91
Sensitivity	0.90	0.91	0.93	0.92	0.91	0.90	0.91
Specificity	0.93	0.64	0.66	0.76	0.68	0.66	0.82
Precision	1.00	0.98	0.97	0.99	0.99	0.99	0.99
Recall	0.90	0.91	0.93	0.92	0.91	0.90	0.91
F1	0.95	0.95	0.95	0.95	0.95	0.94	0.95
Fold2	Total Obs	2359					
Accuracy	0.91	0.91	0.93	0.93	0.90	0.91	0.92
Sensitivity	0.91	0.92	0.93	0.93	0.92	0.91	0.92
Specificity	0.93	0.65	0.77	0.78	0.55	0.75	0.86
Precision	1.00	0.98	0.99	0.99	0.98	1.00	1.00
Recall	0.91	0.92	0.93	0.93	0.92	0.91	0.92
F1	0.95	0.95	0.96	0.96	0.95	0.95	0.96
Fold3	Total Obs	2358					
Accuracy	0.91	0.92	0.93	0.93	0.91	0.90	0.92
Sensitivity	0.91	0.93	0.94	0.94	0.91	0.90	0.92
Specificity	0.91	0.73	0.78	0.79	0.89	0.75	0.90
Precision	1.00	0.99	0.98	0.99	1.00	1.00	1.00
Recall	0.91	0.93	0.94	0.94	0.91	0.90	0.92
F1	0.95	0.96	0.96	0.96	0.95	0.95	0.96
Fold4	Total Obs	2359					
Accuracy	0.91	0.91	0.92	0.92	0.91	0.91	0.91
Sensitivity	0.91	0.92	0.94	0.93	0.92	0.92	0.92
Specificity	1.00	0.73	0.71	0.76	0.71	0.68	0.82
Precision	1.00	0.99	0.98	0.99	0.99	0.99	0.99
Recall	0.91	0.92	0.94	0.93	0.92	0.92	0.92
F1	0.95	0.95	0.96	0.96	0.95	0.95	0.95
Fold5	Total Obs	2359					
Accuracy	0.91	0.91	0.93	0.93	0.91	0.92	0.93
Sensitivity	0.92	0.93	0.94	0.94	0.92	0.92	0.93
Specificity	0.80	0.60	0.67	0.69	0.60	0.78	0.81
Precision	1.00	0.98	0.98	0.98	0.98	1.00	0.99
Recall	0.92	0.93	0.94	0.94	0.92	0.92	0.93
F1	0.95	0.95	0.96	0.96	0.95	0.96	0.96
HeldOutTest	Total Obs	3955					
Accuracy	0.90	0.91	0.92	0.93	0.90	0.91	0.91
Sensitivity	0.90	0.92	0.93	0.93	0.91	0.91	0.91
Specificity	0.91	0.66	0.74	0.78	0.66	0.71	0.82
Precision	1.00	0.99	0.99	0.99	0.99	0.99	1.00
Recall	0.90	0.92	0.93	0.93	0.91	0.91	0.91
F1	0.95	0.95	0.96	0.96	0.95	0.95	0.95

Table 2.2.9: Confusion Matrix Performance Metrics based on 5-Fold Cross Validation.

2.2.2.2 Hyper-parameter selection

This subsection discuses how hyper-parameters were selected for the different models analyzed in the paper. For all of the models, we have used a grid search approach in which we searched over all combinations of the following parameters, whilst leaving other parameters to the default values. Next, we provide specific information for the different models.

XGBoost. In other studies using XGBoost, the hyper-parameters `Eta`, `Max Tree Depth` and `Sub Sample` parameters are the more important hyper-parameters and thus we optimized on these parameters leaving some other parameters to default values. The grid of hyperparameter values explored is shown in table 2.2.10. Optimal values are in bold. This table extends the information shown in the Table 2.1.1 in the main paper.

`Eta`	`Gamma`	`Max Depth`	`Col Sample by Tree`	`Sub Sample`	`Min Child`
0.05, **0.1**, 0.5	0, **0.5**, 1, 1.5	3, 4, **5**, 8, 10	**0.75**, 1	0.75, **1**	1, **5**, 10

Non-optimized parameters: `Lambda` = 1, `Alpha` = 0, `Min Child Weight` = 5, `Max Delta Step` = 0, `Col Sample by Level` = 1, `Scale pos Weight` = number of active companies / number of bankrupt companies.

Table 2.2.10: XGBoost Hyper-parameters grid search. Optimal values in bold type.

LightGBM. Since LightGBM is somewhat similar to XGBoost, a similar grid search was applied. Since it uses a *leaf-wise* tree growth approach a grid search was applied using a series of different values for the `Number of Leaves` where values were kept below $2^{\texttt{Max Depth}}$ as suggested by the LightGBM documentation.[1] Additionally, the `Minimum Amount of Data in a Leaf`, `Feature Fraction`, `Max Depth` of the tree and the `Learning Rate` were optimised. The grid is shown in table 2.2.11.

Random Forest. We optimised the `Number of Trees`, `Max Depth` and the `Number of Variables` in the model. The grid is shown in table 2.2.12.

[1] https://lightgbm.readthedocs.io/en/latest/Parameters-Tuning.html

Eta	Max Depth	Number of Leaves	Feature Fraction
0.01, 0.3, 0.5	2, 3, ..., **7**, 8	4, 6, 8, **18**	0.75, **1**

Table 2.2.11: LightGBM Hyper-parameters grid search. Optimal values in bold type.

Number of Trees	Max Depth	Number of Variables
50, **100**, 200, 500, 1000	4, 6, **8**,10, 12	2, 5, **10**, 15

Table 2.2.12: Random Forest Hyper-parameters grid search. Optimal values in bold type.

Support Vector Machine. The hyper-parameters of the SVM model (both for the Radial and Linear Kernels) were tuned using the `tune` function in the `e1071` package. The parameter `Gamma` took on values $2^{(-1:1)}$ and the `Cost` parameter took on values $2^{(2:4)}$, therefore, the optimal model was found by searching over the following grid for each of the years prior to bankruptcy. The grid is presented in table 2.2.13.

Gamma	Cost
0.5, 1, 2	**4**, 8, 16

Table 2.2.13: SVM Grid Search. Optimal values in bold type.

Neural Network. An implementation of `keras` and `tensorflow` was used to build the Neural Network. The search for optimal parameters for the Neural Network was slightly different to the other methods. During training time a number of different combinations of parameters such as the `Learning Rate` and `Momentum` were manually looked at. We found that lower values of the `Learning Rate` and no `Momentum` parameters were optimal from the various parameters observed. Different `Batch Sizes` and `Dropout Rates` were also applied along with increasing / decreasing the `Depth` of the network. Moreover, we used the R package `tfruns` to tune different parameters in the Neural Network, such as a `Dropout =`

`c(0.2, 0.3, 0.4, 0.5)`. We have also used a Deep Neural Network, for which we used two dense layers, two dropouts with a rate 0.3 and 0.2, respectively, all parameters are presented in table 2.2.14

Learning Rate	Momentum	Decay	Epochs	Batch Size
0.01	**0.9**	**0.01**	**50**	**20**

Table 2.2.14: Neural Network. Optimal values.

2.2.2.3 Variable selection

This subsection discusses the variable selection procedure. The results are presented in tables 2.2.15 to 2.2.18 for 1 year to 4 years prior models. By columns, the tables show the different regression models used: the original Logistic regression model for comparison, a Stepwise logistic, a LASSO, a RIDGE and an Elastic Net regression. Both LASSSO and RIGDE models have a hyper-parameter `Lambda`. The *min* column corresponds to the value of `Lambda` that minimizes the mean cross-validated error, while *1se* stands for a regularised model with `Lambda` such that the error is within one standard error of the minimum. For all columns, the blank cells are for coefficients with zero point estimates.

Basically, the tables suggest to remove the variables EBIT.Capital, EBIT.FinExp, SALES.FA and EBITDA.SALES since their corresponding point estimates are either zero or close to zero across the number of variable selection models Stepwise, LASSO, RIDGE and Elastic Net. The coefficients are zero or close to zero across the number of years prior to bankruptcy also. Additionally, according to the feature importance plots from the XGBoost model in the original paper, these variables (with the exception of EBIT.FinExp) also featured consistently low down in the plot for each of the years analysed. After the removal of the variables, we then compare the performance of each of the models to the original paper.

Variable	Logistic	Stepwise	LASSOmin	LASSO1se	RIDGEmin	RIDGE1se	ELASTICNET1se
(Intercept)	-4.044	-4.041	-4.012	-3.587	-3.559	-3.352	-3.507
TL.TA	0.976	0.976	0.976	0.942	0.881	0.775	0.897
CA.CL	-0.253	-0.258	-0.245	-0.114	-0.22	-0.205	-0.158
TL.EQ	0.257	0.255	0.255	0.24	0.229	0.222	0.224
EBIT.Capital	-0.096	-0.096	-0.093	-0.035	-0.065	-0.047	-0.04
WC.EBIT	0.385	0.383	0.374	0.215	0.236	0.174	0.199
EBIT.FinExp	0.512	0.514	0.468		0.008	-0.077	
SALES.EBIT	-0.631	-0.63	-0.621	-0.447	-0.45	-0.361	-0.402
CL.FinExp	-1.87	-1.872	-1.793	-0.865	-0.833	-0.597	-0.769
EQ.Turnover	-0.25	-0.252	-0.246	-0.098	-0.254	-0.215	-0.197
CF.NCL	0.106	0.104	0.103	0.027	0.09	0.071	0.057
logTA	0.269	0.283	0.28	0.284	0.482	0.474	0.446
logSALES	0.992	0.977	0.976	0.895	0.643	0.559	0.679
CF.CL	-0.668	-0.664	-0.641	-0.311	-0.315	-0.234	-0.279
SALES.FA	-0.016		-0.01		0.02	0.02	
CF.SALES	0.817	0.817	0.783	0.324	0.428	0.32	0.301
EBITDA.SALES	-0.386	-0.387	-0.358		-0.128	-0.054	-0.003
DEBTORS.SALES	0.248	0.248	0.247	0.237	0.239	0.23	0.23

Table 2.2.15: Regression Coefficients for each model 1 year prior to bankruptcy

Variable	Logistic	Stepwise	LASSOmin	LASSO1se	RIDGEmin	RIDGE1se	ELASTICNET1se
(Intercept)	-3.61	-3.599	-3.582	-3.055	-3.178	-2.908	-3.17
TL.TA	0.606	0.602	0.607	0.582	0.535	0.413	0.579
CA.CL	-0.053		-0.049		-0.098	-0.111	-0.022
TL.EQ	0.208	0.217	0.206	0.189	0.189	0.185	0.189
EBIT.Capital	-0.046	-0.053	-0.043		-0.022	-0.009	
WC.EBIT	0.331	0.33	0.322	0.09	0.219	0.154	0.174
EBIT.FinExp	0.445	0.419	0.4		-0.037	-0.131	
SALES.EBIT	-0.434	-0.458	-0.425	-0.166	-0.316	-0.224	-0.261
CL.FinExp	-1.791	-1.756	-1.717	-0.724	-0.793	-0.469	-0.846
EQ.Turnover	-0.244	-0.213	-0.24	-0.065	-0.251	-0.187	-0.169
CF.NCL	-0.094	-0.112	-0.089		-0.087	-0.086	-0.028
logTA	0.111		0.13	0.473	0.45	0.449	0.431
logSALES	1.079	1.195	1.054	0.557	0.619	0.487	0.638
CF.CL	-0.915	-0.954	-0.885	-0.348	-0.454	-0.291	-0.456
SALES.FA	-0.029		-0.024		0.022	0.021	
CF.SALES	0.595	0.624	0.558		0.229	0.1	0.105
EBITDA.SALES	-0.219	-0.201	-0.194		-0.078	-0.029	
DEBTORS.SALES	0.346	0.349	0.343	0.312	0.317	0.287	0.315

Table 2.2.16: Regression Coefficients for each model 2 year prior to bankruptcy

Variable	Logistic	Stepwise	LASSOmin	LASSO1se	RIDGEmin	RIDGE1se	ELASTICNET1se
(Intercept)	-3.235	-3.253	-3.208	-2.777	-2.971	-2.667	-2.928
TL.TA	0.463	0.5	0.473	0.434	0.426	0.288	0.455
CA.CL	-0.073		-0.063		-0.091	-0.086	-0.027
TL.EQ	0.209	0.202	0.199	0.166	0.168	0.154	0.166
EBIT.Capital	-0.053	-0.051	-0.045		-0.026	-0.006	
WC.EBIT	0.401	0.377	0.378	0.135	0.268	0.169	0.229
EBIT.FinExp	0.285	0.285	0.227		-0.014	-0.121	
SALES.EBIT	-0.435	-0.401	-0.406	-0.104	-0.281	-0.161	-0.222
CL.FinExp	-1.237	-1.259	-1.161	-0.444	-0.685	-0.351	-0.636
EQ.Turnover	-0.14	-0.186	-0.158		-0.232	-0.169	-0.146
CF.NCL	-0.059		-0.045		-0.034	-0.039	
logTA	-0.101		0	0.249	0.403	0.415	0.345
logSALES	1.287	1.185	1.178	0.76	0.672	0.469	0.715
CF.CL	-0.649	-0.675	-0.6	-0.189	-0.332	-0.193	-0.294
SALES.FA	0.051		0.055		0.1	0.081	0.043
CF.SALES	0.537	0.527	0.468		0.217	0.064	0.061
EBITDA.SALES	-0.188	-0.2	-0.157		-0.099	-0.034	
DEBTORS.SALES	0.312	0.306	0.306	0.28	0.284	0.242	0.283

Table 2.2.17: Regression Coefficients for each model 3 year prior to bankruptcy

Variable	Logistic	Stepwise	LASSOmin	LASSO1se	RIDGEmin	RIDGE1se	ELASTICNET1se
(Intercept)	-3.148	-3.142	-3.125	-2.734	-2.925	-2.604	-2.849
TL.TA	0.428	0.426	0.43	0.356	0.378	0.236	0.391
CA.CL	-0.016		-0.014		-0.064	-0.076	
TL.EQ	0.188	0.186	0.18	0.171	0.148	0.14	0.166
EBIT.Capital	-0.014		-0.008		0.012	0.022	
WC.EBIT	0.387	0.383	0.365	0.133	0.221	0.126	0.193
EBIT.FinExp	-0.304	-0.308	-0.271	-0.028	-0.194	-0.155	-0.078
SALES.EBIT	-0.512	-0.51	-0.482	-0.174	-0.284	-0.146	-0.262
CL.FinExp	-0.413	-0.404	-0.423	-0.265	-0.393	-0.241	-0.336
EQ.Turnover	-0.198	-0.202	-0.226	-0.031	-0.334	-0.215	-0.149
CF.NCL	0.014		0.016		0.039	0.008	
logTA	-0.46	-0.452	-0.332		0.361	0.398	0.065
logSALES	1.68	1.673	1.547	1.058	0.742	0.485	1.04
CF.CL	-0.473	-0.478	-0.455	-0.223	-0.273	-0.172	-0.271
SALES.FA	-0.012		0		0.08	0.078	
CF.SALES	-0.175	-0.161	-0.146		-0.125	-0.078	
EBITDA.SALES	0.459	0.453	0.398		0.138	0.022	
DEBTORS.SALES	0.233	0.232	0.226	0.19	0.2	0.169	0.199

Table 2.2.18: Regression Coefficients for each model 4 year prior to bankruptcy

In order to check the validity of each of the models we compute some relevant statistics from the confusion Matrix. The results are shown in tables 2.2.19 to 2.2.22. The models do not appear to differ significantly when compared with the original logistic regression confusion matrix results - under the column *Logistic*.

Metric	Logistic	Stepwise	LASSOmin	LASSO1se	RIDGEmin	RIDGE1se	ElasticNet
Accuracy	0.908	0.908	0.909	0.906	0.906	0.904	0.904
Sensitivity	0.985	0.985	0.986	0.988	0.988	0.991	0.988
Specificity	0.249	0.247	0.249	0.201	0.203	0.165	0.182
Precision	0.918	0.918	0.918	0.914	0.914	0.910	0.912
F1	0.951	0.950	0.951	0.949	0.950	0.949	0.949

Table 2.2.19: Variable Selection Confusion Matrix Results (1 year prior)

Metric	Logistic	Stepwise	LASSOmin	LASSO1se	RIDGEmin	RIDGE1se	ElasticNet
Accuracy	0.888	0.888	0.888	0.882	0.886	0.881	0.886
Sensitivity	0.981	0.981	0.981	0.987	0.984	0.988	0.985
Specificity	0.231	0.229	0.226	0.132	0.191	0.115	0.178
Precision	0.901	0.901	0.900	0.890	0.897	0.889	0.895
F1	0.939	0.939	0.939	0.936	0.938	0.936	0.938

Table 2.2.20: Variable Selection Confusion Matrix Results (2 year prior)

Metric	Logistic	Stepwise	LASSOmin	LASSO1se	RIDGEmin	RIDGE1se	ElasticNet
Accuracy	0.882	0.882	0.882	0.875	0.879	0.873	0.877
Sensitivity	0.983	0.983	0.984	0.991	0.987	0.994	0.988
Specificity	0.235	0.233	0.229	0.127	0.180	0.096	0.163
Precision	0.892	0.892	0.892	0.880	0.886	0.876	0.884
F1	0.935	0.935	0.935	0.932	0.934	0.931	0.933

Table 2.2.21: Variable Selection Confusion Matrix Results (3 year prior)

Metric	Logistic	Stepwise	LASSOmin	LASSO1se	RIDGEmin	RIDGE1se	ElasticNet
Accuracy	0.869	0.869	0.867	0.865	0.868	0.867	0.867
Sensitivity	0.979	0.979	0.979	0.987	0.984	0.993	0.985
Specificity	0.163	0.163	0.148	0.077	0.119	0.058	0.102
Precision	0.883	0.883	0.881	0.873	0.878	0.872	0.876
F1	0.928	0.928	0.927	0.927	0.928	0.928	0.928

Table 2.2.22: Variable Selection Confusion Matrix Results (4 year prior)

Finally in this subsection, we compare models using the original data (without the removal of variables) and the same comparison after removing variables. Both comparisons are done in tables 2.2.23 and 2.2.24, respectively. Table 2.2.23 is identical the the table reported in the main paper, Table 2.1.4 and included here for ease of comparison.

Metric	XGBoost	Logistic	Light GBM	Shallow NN[a]	Deep NN	R. Forest	SVM (R)[b]	SVM (L)
Accuracy	0.87	0.91	0.89	0.91	0.91	0.92	0.90	0.91
Sensitivity	0.73	0.25	0.61	0.32	0.30	0.37	0.16	0.18
Specificity	0.88	0.99	0.92	0.98	0.98	0.99	0.99	0.99
Precision	0.42	0.66	0.47	0.60	0.62	0.79	0.66	0.71
F1	0.54	0.36	0.53	0.42	0.41	0.51	0.26	0.29
MCC	0.49	0.37	0.47	0.39	0.39	0.51	0.30	0.33
AUC	0.81	0.81	0.76	0.65	0.64	0.68	0.58	0.59

[a] NN: Neural Network. [b] SVM: Support Vector Machine with (R) Radial, (L) Linear kernel.

Table 2.2.23: Comparison to other Machine Learning methods for one year prior predictions *without* the removal of variables.

Metric	XGBoost	Logistic	Light GBM	Shallow NN[a]	Deep NN	R. Forest	SVM (R)[b]	SVM (L)
Accuracy	0.85	0.90	0.88	0.90	0.91	0.92	0.90	0.90
Sensitivity	0.74	0.25	0.62	0.36	0.28	0.36	0.23	0.15
Specificity	0.87	0.98	0.91	0.97	0.98	0.99	0.99	0.99
Precision	0.42	0.65	0.46	0.64	0.70	0.77	0.71	0.78
F1	0.53	0.37	0.53	0.46	0.40	0.49	0.35	0.26
MCC	0.48	0.37	0.46	0.43	0.40	0.49	0.37	0.32
AUC	0.80	0.80	0.76	0.67	0.63	0.67	0.61	0.57

[a] NN: Neural Network. [b] SVM: Support Vector Machine with (R) Radial, (L) Linear kernel.

Table 2.2.24: Comparison to other Machine Learning methods for one year prior predictions *after* the removal of variables.

The tables show that, as for the performance metrics, to remove variables has a rather marginal impact. After removing the variables that obtained zero or close to zero coefficients from the Stepwise, LASSO, RIDGE and Elastic Net we find little change in the performance metrics when comparing between the original models and the more simpler models. A simpler model with few variables is preferential over more complex models with more variables if the performance between the two models are somewhat similar. In the paper we take a step further to analyse the XGBoost models interpretation as opposed to focusing on just accuracy and performance metrics. The variable EBIT.FinExp was found to be a significant and important variable in the original paper whereas in the variable selection models proposed here it was suggested to be removed. Moreover, a balance should be made between removing variables which may (or may not) marginally improve accuracy and at the cost of the interpretation of relevant variables.

2.2.2.4 Financial variables in previous literature

Table 2.2.25 summarizes the use of the different variables used in our analysis which are found in the previous literature.

2.2.3 Rolling Extension

In this section we construct rolling window models in order to test the models performance over time. We provide two methods of rolling windows, method 1 uses a fixed size rolling training and test window whereas method 2 uses a cumulative rolling training window with a fixed rolling test window.

We split our data into a rolling window as per Figure 2.2.1. The black dots indicate past data not used in the current model. The blue dots indicate the training data set and the red dots indicate the testing data set, whereas the grey dots indicate future data not

Variable	No. Citations	Citations
TL.TA	iiiiiiiiii	Beaver (1966) Begley et al. (1996) Callejón et al. (2013) Creamer and Freund (2004) Fernández-Gámez et al. (2016) Frydman et al. (1985) Hansen and Messier Jr (1991) Hernandez-Tinoco and Wilson (2013) Zhao et al. (2017) Zhou and Lai (2017) Zięba et al. (2016) Zmijewski (1984)
CA.CL	iiiiiiiiii	Beaver (1966) Begley et al. (1996) Callejón et al. (2013) De Andrés et al. (2005) Fernández-Gámez et al. (2016) Frydman et al. (1985) Hansen and Messier Jr (1991) Zhao et al. (2017) Zhou and Lai (2017) Zięba et al. (2016) Zmijewski (1984)
TL.EQ	iiii	De Andrés et al. (2005) Hansen and Messier Jr (1991) Zhao et al. (2017) Zięba et al. (2016)
EBIT.Capital	i	West (1985)
WC.EBIT		
EBIT.FinExp	iii	Fernández-Gámez et al. (2016) Hernandez-Tinoco and Wilson (2013) Tam and Kiang (1992)
SALES.EBIT	iiiii	Barboza et al. (2017) Creamer and Freund (2004) Hansen and Messier Jr (1991) Zhao et al. (2017) Zhou and Lai (2017)
CL.FinExp		
EQ.Turnover		
CF.NCL	i	Fernández-Gámez et al. (2016)
logTA	iiiii	Bell et al. (1990) Callejón et al. (2013) Frydman et al. (1985) Zhou and Lai (2017) Zięba et al. (2016)
logSALES	i	Hansen and Messier Jr (1991)
CF.CL	iii	Beaver (1966) Frydman et al. (1985) Zhou and Lai (2017)
SALES.FA	ii	Fernández-Gámez et al. (2016) Zięba et al. (2016)
CF.SALES	ii	Beaver (1966) Frydman et al. (1985)
EBITDA.SALES	i	Zięba et al. (2016)
DEBTORS.SALES	iii	Fernández-Gámez et al. (2016) Zhao et al. (2017) Zięba et al. (2016)

Table 2.2.25: Frequency of variables used

used in the current model. We apply two different approaches, first a rolling training and testing data set in which the sizes are held constant but shift forward each year, secondly a cumulative training data set with a constant testing data set, as set out in Figure 2.2.1.

The data starts in 2010 - which is our first year of training data, which aims to predict the 2011 observations. The first model has a constant number of non-bankrupt firms in both the training and testing data set with small changes in the number of reported bankrupt firms, whereas the second model cumulatively adds the previous years non-bankrupt and bankrupt observations into the next periods training data.

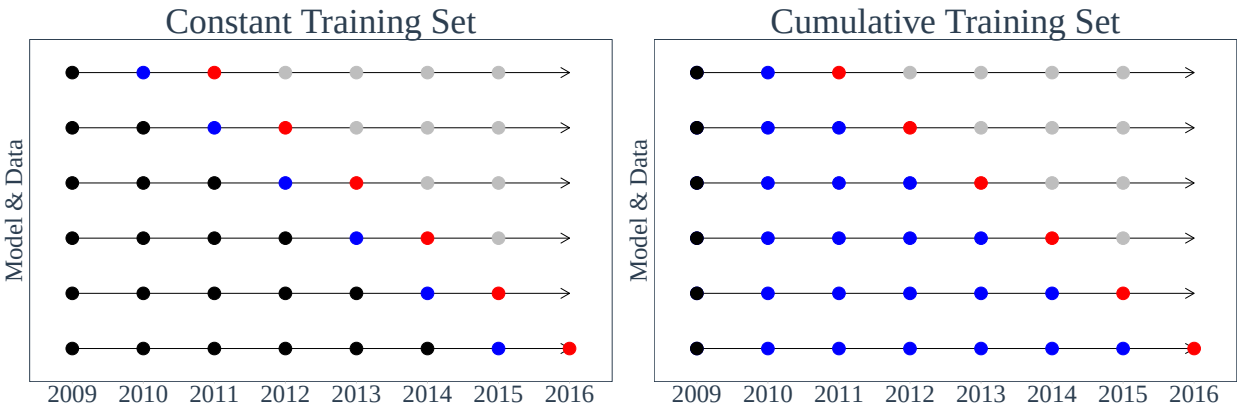


Figure 2.2.1: Rolling Testing Data with different Training Periods

Figure 2.2.2 reports the constant and cumulative rolling training and test results corresponding to Figure 2.2.1. The first confusion matrices (2011) were trained on the 2010 data and were then tested using the 2011 data (for both models). The second confusion matrix trained on the 2011 data and tested on the 2012 data for the constant rolling window model. Whereas the rolling window model trained on the 2010 and 2011 data in order to test on the 2012 data.

The both models report noticeably higher correct bankruptcy predictions from 2011 to 2013 which could be a spill over effect from the financial crisis, since Spain during this period

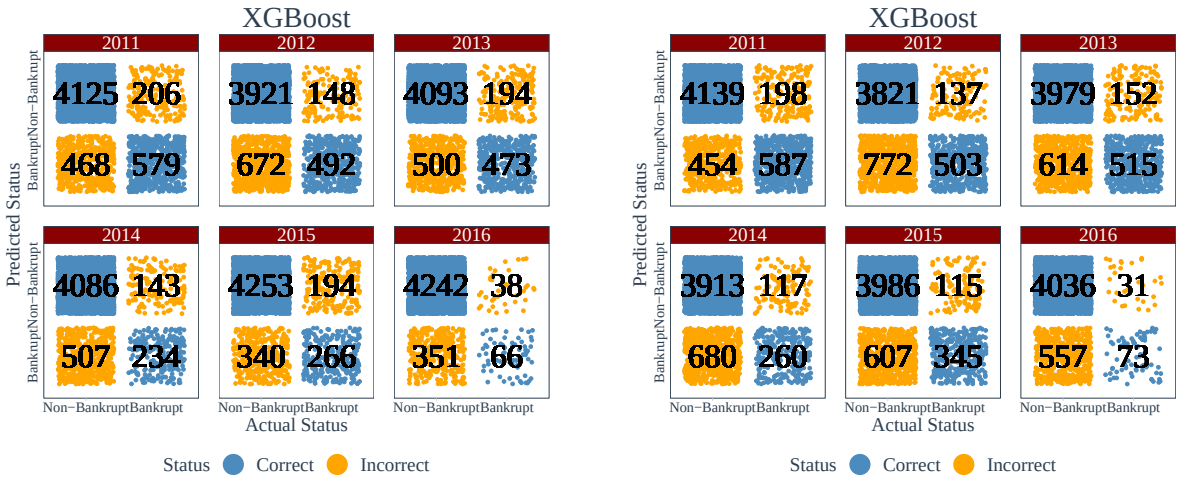


Figure 2.2.2: Left: Constant training set | Right: Cumulative training set

was still undergoing economic strain where companies balance sheets may reflect this. In comparison the both models makes significantly less correct bankrupt predictions in 2014 and 2015.[2] We comment in the main paper that the Spanish economy started recovering in 2015 with GDP growth at 1.6% throughout 2008 to 2016. That is, in 2011 the model made 579 correct bankrupt predictions yet in 2014 it made less than half this number of correct bankrupt predictions (234).

Tables 2.2.26 and 2.2.27 report the corresponding confusion matrix statistics. As expected the first year of results for both models report relatively the same results. The sensitivity, AUC and AUPRC results for the 2015 cumulative model is somewhat higher than the constant rolling window model - which we attribute to the fact that the cumulative model is able to *see* or train on more bankrupt data and is thus better at detecting bankrupt firms.

[2]We refrain from commenting on results for 2016 since Figure 2.2.3 shows that there were significantly less bankrupt observations in our data set for this year.

Statistics: XGBoost (Rolling)

Metric	2011	2012	2013	2014	2015	2016
Accuracy	0.87	0.84	0.87	0.87	0.89	0.92
Sensitivity	0.74	0.77	0.71	0.62	0.58	0.63
Specificity	0.90	0.85	0.89	0.89	0.93	0.92
Precision	0.55	0.42	0.49	0.32	0.44	0.16
F1	0.63	0.55	0.58	0.42	0.50	0.25
MCC	0.57	0.49	0.51	0.38	0.45	0.29
AUC	0.82	0.81	0.80	0.76	0.75	0.78
AUPRC	0.69	0.56	0.65	0.45	0.51	0.37
Bankrupt (trn)	584	785	640	667	377	460
Non-Bankrupt (trn)	4593	4593	4593	4593	4593	4593
Bankrupt (tst)	785	640	667	377	460	104
Non-Bankrupt (tst)	4593	4593	4593	4593	4593	4593

Table 2.2.26: XGBoost rolling analysis confusion matrix statistics

Statistics: XGBoost (Cumulative)

Metric	2011	2012	2013	2014	2015	2016
Accuracy	0.88	0.83	0.85	0.84	0.86	0.87
Sensitivity	0.75	0.79	0.77	0.69	0.75	0.70
Specificity	0.90	0.83	0.87	0.85	0.87	0.88
Precision	0.56	0.39	0.46	0.28	0.36	0.12
F1	0.64	0.53	0.57	0.39	0.49	0.20
MCC	0.58	0.47	0.52	0.37	0.45	0.25
AUC	0.82	0.81	0.82	0.77	0.81	0.79
AUPRC	0.69	0.59	0.63	0.42	0.56	0.35
Bankrupt (trn)	584	1369	2009	2676	3053	3513
Non-Bankrupt (trn)	4593	9186	13779	18372	22965	27558
Bankrupt (tst)	785	640	667	377	460	104
Non-Bankrupt (tst)	4593	4593	4593	4593	4593	4593

Table 2.2.27: XGBoost cumulative analysis confusion matrix statistics

2.2.3.1 Banckruptcy firms

Figure 2.2.3 plots the number of bankrupt and non-bankrupt observations for each year since 2010, which clearly shows we have an unbalanced sample. We also plot the number of bankrupt firms from 1995. The figure shows the number of bankrupt firms increased significantly for the years after the financial crash of 2008.

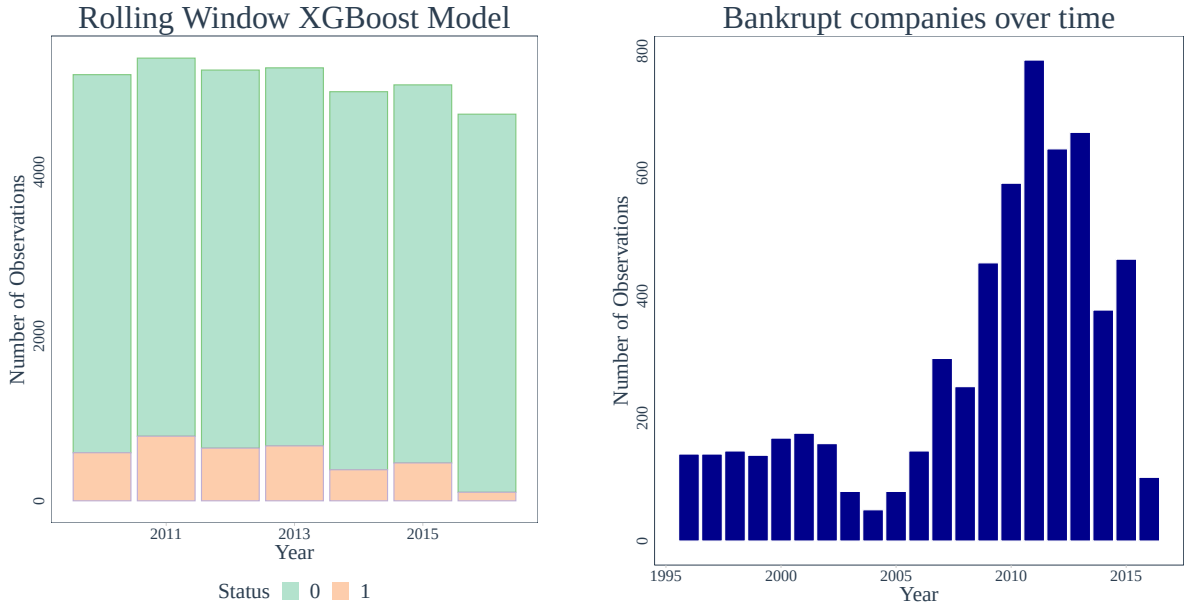


Figure 2.2.3: Bankrupt and non-bankrupt firms in the sample period

2.2.3.2 XGBoost Decision Tree

A typical tree in the model can be presented as Figure 2.2.4. A typical path can be as follows. We first start out at the root node 87-0, that is tree 87, root node 0. The model made a split on the variable CL.FinExp, that is all companies with a CL.FinExp ratio less than 8.13267899 went to the right side of the decision tree and all firms greater or equal went to the left side. Assuming that our observation fell on the left side of the tree we would then move down to the decision node 87-2 where the model makes a split on TL.TA. The same reasoning applies and we would fall to the left side of the decision tree again if our ratio is greater than 4.09176731, moving to decision node 87-6, again splitting on TL.TA, this time

with a ratio 4.19334126, only marginally higher than the previous split ratio. We continue left to decision node 87-14 splitting on CA.TA and finally to decision node 87-28. The tree is forced to terminate here since one of the parameters we set was the maximum tree depth of 5, in theory the tree can grow as large as it wants until it classifies every observation into a decision node or until it runs out of variables to split upon.

Each terminal leaf is associated with a weight or score, w, such that terminal leaf 87-52 assigns a weight of 0.056077268 to all firms who fell into this category in this tree. Notice that this weight is positive whereas other terminal nodes have negative weights, that is firms who followed this path have been given a positive score to determine bankruptcy. There are a total of 88 trees similar to this one in our model with each tree having a different structure, therefore each firm will end up in different terminal nodes in each tree and be assigned different weights or scores. Finally each of the firms weights are summed up to give an overall weight which corresponds to the log-odds score and can be converted to a probability of bankruptcy by taking the inverse-logit transformation of the sum.[3] Additionally Figure 2.2.4 reports two other values *Cover* and *Gain*. The *Cover* value is the number of instances that the variable was used to split the data across all trees, weighted by the number of training observations which passed through those splits. The *Gain* is the average training loss reduction gained when using that variable for making a split.

[3] $Prob(y = 1 \mid X) = \frac{1}{1+\exp(-z)}$ where, $z = Xw$

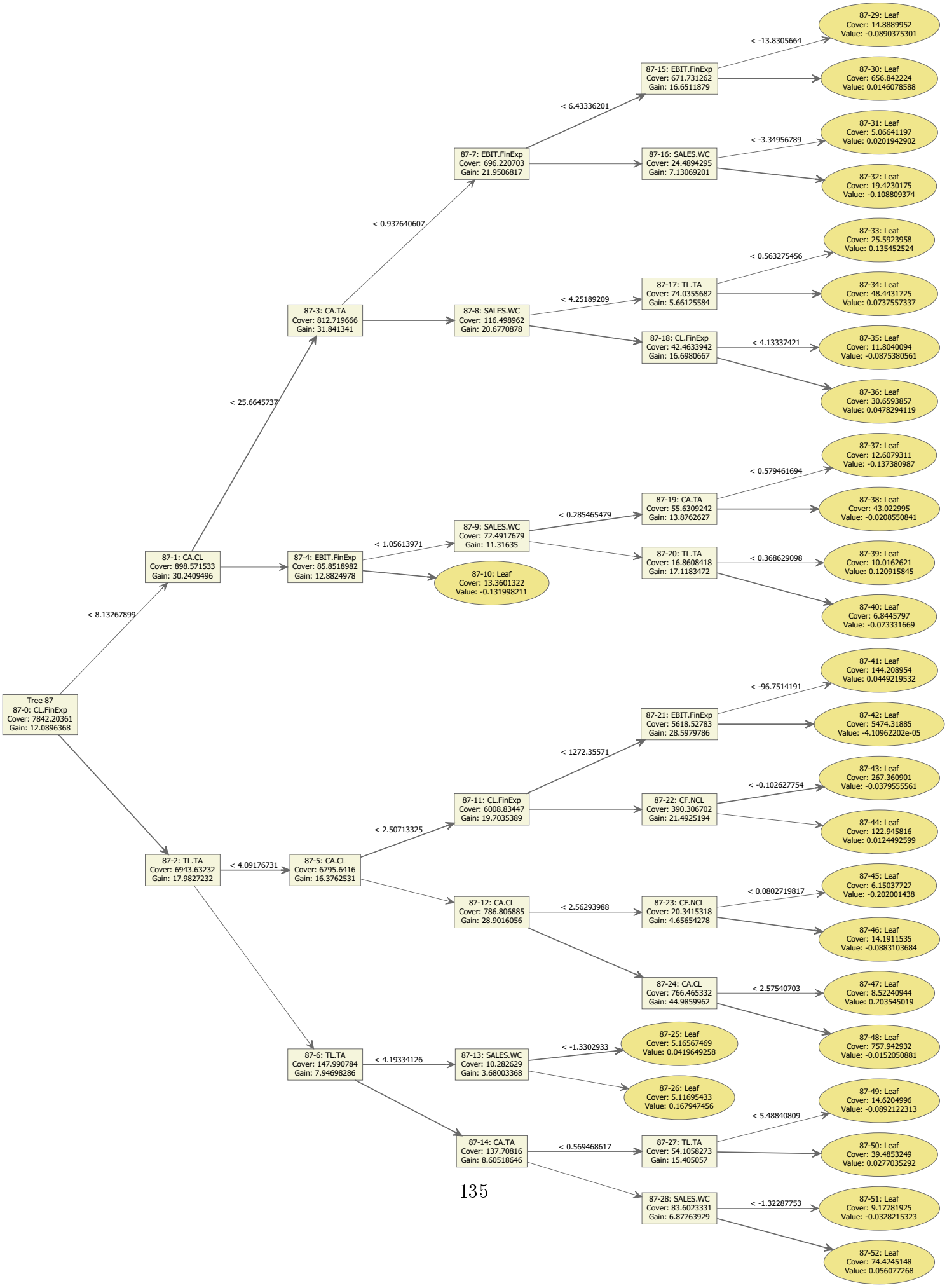


Figure 2.2.4: Decision Tree 87

Chapter 2.3

Are Predictions Time Consistent

The analysis carried out in this chapter makes a limited use of the underlying dynamics. It uses the dynamics in the sense that past variables (financial ratios) determine the future bankruptcy state. However, there is more to it. There should be some dynamic consistency when we make simultaneous predictions using a given dataset for different time horizons. For instance, say with information up to time t we make both a prediction for the bankruptcy state at $t+1$, $t+2$ and so on. These predictions should be, in some sense, *mutually consistent* with one another.

The way that consistency is imposed -at least in economics- is not at the point of predictions, but in the probabilistic structure that is generated from the observed data. In a more general form, each observed value, either for the response or explanatory variables, is assumed to be a realisation of a random variable. The referred consistency condition among predictions for different time horizons is usually stated in terms of the underlying probability distributions that generate the observations.

Stating this consistency condition within a *minimal setting* we can emphasise the elements that cannot be easily *exported* to a (supervised) Machine Learning approach. Additionally, it allows for Monte Carlo Simulations. Essentially, we build upon an index function

model[1] with very simple probability distributions that allow for closed-form expressions. Let us consider a discrete time model, indexed by t, with $t \in \{0, 1, \dots\}$. There is a binary response variable, denoted by Y, such that $Y_t = 1$ represents the event "*the firm (assuming a single firm in isolation) is in a state of bankruptcy at time t*, while $Y_t = 0$ stands for the complementary event "*the firm is healthy at time t*". To make things as simple as possible, assume there is a single explanatory variable, X, that fully determines the response variable. More specifically, there is some threshold value, say k, such that the firm turns to bankruptcy the first time X is greater than k. Mathematically, for any time t, we assume:

$$Y_t = 1 \iff \exists\, s \in \{0, 1, \dots, t\} \text{ such that } X_s > k$$

Clearly, the above statement implies the bankruptcy state is irreversible, as in our sample. We finally assume the sequence $\{X\}_t$ evolves over time according to a random walk.[2] We can write the dynamics for X as:

$$X_{t+1} = X_t + \varepsilon_t$$

where the sequence $\{\varepsilon_t\}$ is a white noise process that conforms the shock. Now, let $E\{Y_{t+k} \mid Y_t = 0, X_t = x\}$ denote the prediction of Y_{t+k} conditional on the firm being healthy at time t and the observed value of X at t being x. We restrict ourselves to one and two period ahead predictions, that is, $k = 1$ and $k = 2$, respectively. Straightforward computations lead to:

$$E\{Y_{t+1} \mid Y_t = 0, X_t = x\} = Pr\left(X_{t+1} > K \mid X_t = x\right) \tag{2.3.1}$$

where implicitly we have assumed the Markovian property of a random walk, namely, $X_t = x$ contains all relevant information up to time t to predict X_{t+1}. In addition, $K - x > 0$

[1] See Greene (1993) page 642 for the reference list.

[2] The choice of the stochastic process for the regressor is irrelevant for our illustrative purposes. While the random walk is non-stationary, all we need is its conditional probability distribution, which is well defined. This allows us to introduce some dynamics without need for presumably unknown parameters in the first order moments.

must hold by construction.

In order to make the prediction two periods ahead, we must notice that $Y_{t+2} = 1$ holds conditional on $Y_t = 0$ if and only if any of the two following disjoint events holds: (i) $X_{t+1} > k$, (ii) both $X_{t+1} \leq k$ and $X_{t+2} > k$. If we additionally condition on $X_t = x$, the probability of the first event is precisely in equation (2.3.1), whereas the probability of the second event is:

$$Pr\left(X_{t+2} > k \wedge X_{t+1} < k \mid X_t = x\right)$$

Thus,

$$E\{Y_{t+2} \mid Y_t = 0, X_t = x\} = E\{Y_{t+1} \mid Y_t = 0, X_t = x\} + Pr\left(X_{t+2} > k \wedge X_{t+1} < k \mid X_t = x\right) \tag{2.3.2}$$

Equation (2.3.2) is the consistency condition among the one and two period ahead predictions. It simply indicates that the difference in the prediction from one to two period ahead is precisely the probability that there is a change -the only possible one- in the state from $t+1$ to $t+2$. Furthermore, that probability is written in terms of the stochastic process of X's.

Now we can address the basic question of this subsection. The Machine Learning model, such as, XGBoost, generates one and two period ahead predictions. Could we expect that those predictions satisfy (2.3.2)? Two different arguments come into to play. First, (2.3.2) does depend on the probabilistic structure, so the predictions from the Machine Learning model might not satisfy that consistency requirement because the underlying probabilistic model -if it actually exists- is not as specified above. Second, perhaps more fundamentally, the Machine Learning model in this chapter is essentially a *classifier*. It classifies the observed firms into two mutually exclusive groups: bankrupt and healthy. This is done through a series of classification trees that balances complexity and prediction errors. Both of these two concepts are *exclusively measured* in terms of the observed data: they are algebraic

rather than statistical. In other words, Machine Learning assumes we do not observe a sample drawn from a population, but we observe the whole population itself, whose description is merely exhaustive, and the task is to summarise it conveniently.

The above discussion drives us to the following question: does Machine Learning produce consistent predictions when, *as a matter of fact*, there is an underlying population? Let us assume we have artificially generated a sample from a population as described above. Specifically, let us assume our sample consists of information for a set of N firms, for each firm containing $\{(X_t, Y_t)\}$ for $t \in \{0, 1, 2\}$. Thus, we only have information on three periods, the minimum length that allows for a two-period ahead prediction, while the cross sectional dimension is reasonably large as to apply a Machine Learning algorithm. We can use such an algorithm to generate one and two period ahead predictions, then compute the difference between both predictions and see if that difference qualitatively agrees with the last term in equation (2.3.2), which can be computed on the basis of the underlying population.

Let us specify a very simple artificial setting which allows for the analytical computation of that last term. Let us assume that X_0 is uniformly distributed in the $[0, 1]$ interval, which we succinctly write as $X_0 \sim \mathcal{U}[0, 1]$. Using an analogous notation, we set $\varepsilon_t \sim \mathcal{U}[-1, 1]$, with the usual i.i.d. qualifier applying over time. This structure meets the basic features of a white noise process.[3] Denoting by F_ε as the cumulative probability distribution of ε and f_ε as its corresponding density. Furthermore, let $k \in [1, 2]$, so that none of the firms in our sample will be in a state of bankruptcy at time 0, since $X_0 \leq k$ holds with probability one by construction, while there is a positive probability for firms to be bankrupt for the first time at period 1 or 2 (or none).

[3]Using Normal distribution for the shocks involves numerical integration for the term to be computed, and we try to remain analytical as much as possible in order to avoid additional sources of discrepancies between model and Machine Learning based computations of the probability in (2.3.2).

Now, we focus on the computation of the probability in the last term in (2.3.2). Let $X_0 = x$, that is, the realisation of X_0 is x. We want to compute the probability that a given firm enters a state of bankruptcy at $t = 2$, that is, $X_2 > k$ and $X_1 < k$. Necessarily, it has to be $X_1 \in (k-1, k)$. Clearly, a value for X_1 lower than $k-1$ would make it impossible to enter bankruptcy at $t = 2$ while a value larger than k would make it impossible to enter bankruptcy at $t = 1$. Substituting, $X_1 \in (k-1, k)$ is equivalent to:

$$k - 1 - x < \varepsilon_0 < k - x \tag{2.3.3}$$

Using Bayes rule, we can write:

$$Pr\left(X_2 > k \wedge X_1 < k \mid X_0 = x\right) = \int_{k-1-x}^{k-x} Pr\left(X_2 > k \mid X_1 = x + \varepsilon_0\right) f_\varepsilon(\varepsilon_0) d\varepsilon_0 \tag{2.3.4}$$

where the limits of integration follow from (2.3.3). Now, for a given value of ε_0 within the considered range, we have $X_2 > k$ if and only if $\varepsilon_1 > k - x - \varepsilon_0$, whose probability is $1 - F_\varepsilon\left(k - x - \varepsilon_0\right)$. Substituting in (2.3.4), we have:

$$Pr\left(X_2 > k \wedge X_1 < k \mid X_0 = x\right) = \int_{k-1-x}^{k-x} \left(1 - F_\varepsilon\left(k - x - \varepsilon_0\right)\right) f_\varepsilon(\varepsilon_0) d\varepsilon_0 \tag{2.3.5}$$

The computation of the right hand side in (2.3.5) is straightforward just using the cumulative distribution and density functions for the shock. In doing so, we must notice that since $k - x > 0$, the integration lower limit lies in $(-1, 1)$, whereas the upper limit might be greater than 1.

Since the shock is distributed $\mathcal{U}[-1, 1]$, it is $F_\varepsilon(z) = \frac{1}{2}(1 + z)$ and $f_\varepsilon(z) = \frac{1}{2}$ for any $z \in [-1, 1]$. Thus, $\left(1 - F_\varepsilon(z)\right) f_\varepsilon(z') = \frac{1}{4}(1 - z)$. Substituting in the integral in (2.3.5), for $k - x \leq 1$ we have:

$$\int_{k-1-x}^{k-x} \left(1 - F_\varepsilon\left(k - x - \varepsilon_0\right)\right) f_\varepsilon(\varepsilon_0) d\varepsilon_0 = \frac{1}{4} \int_{k-1-x}^{k-x} (1 - (k - x) + \varepsilon_0) d\varepsilon_0 = \frac{1}{8}$$

For $k - x > 1$, it is

$$\int_{k-1-x}^{k-x} \left(1 - F_\varepsilon\left(k - x - \varepsilon_0\right)\right) f_\varepsilon(\varepsilon_0) d\varepsilon_0 = \frac{1}{4} \int_{k-1-x}^{1} (1-(k-x)+\varepsilon_0) d\varepsilon_0 = \frac{1}{4}\left(2 - 2(k-x) + \frac{1}{2}(k-x)^2\right)$$

To summarise, denoting by $g(k-x)$ to the right hand side of (2.3.5), it is:

$$g(k-x) = \begin{cases} \frac{1}{8} & \text{if } k - x \leq 1 \\ \frac{1}{4}\left(2 - 2(k-x) + \frac{1}{2}(k-x)^2\right) & \text{if } k - x > 1 \end{cases}$$

By construction, $k-x$ lies in $(0, 2)$, which constitutes the support of g. It is straightforward to prove that g is continuous and positive in its support and it is strictly decreasing in $(1, 2)$.

As a merely illustrative example, we have done some numerical exploration. We take $k = 1.5$ and $N = 1000$ firms, among which 750 are used to train the model and the rest is used for testing. Figure 2.3.1 shows the scatter-plot that compares the last term in (2.3.2) computed numerically by subtracting the corresponding predictions from XGBoost, in the vertical axis, with the closed-form computation above presented. The blue line in the figure is not a regression line but a 45° degree line. Thus, the distance between that lines and the point represent deviations from consistence of the Machine Learning based predictions.

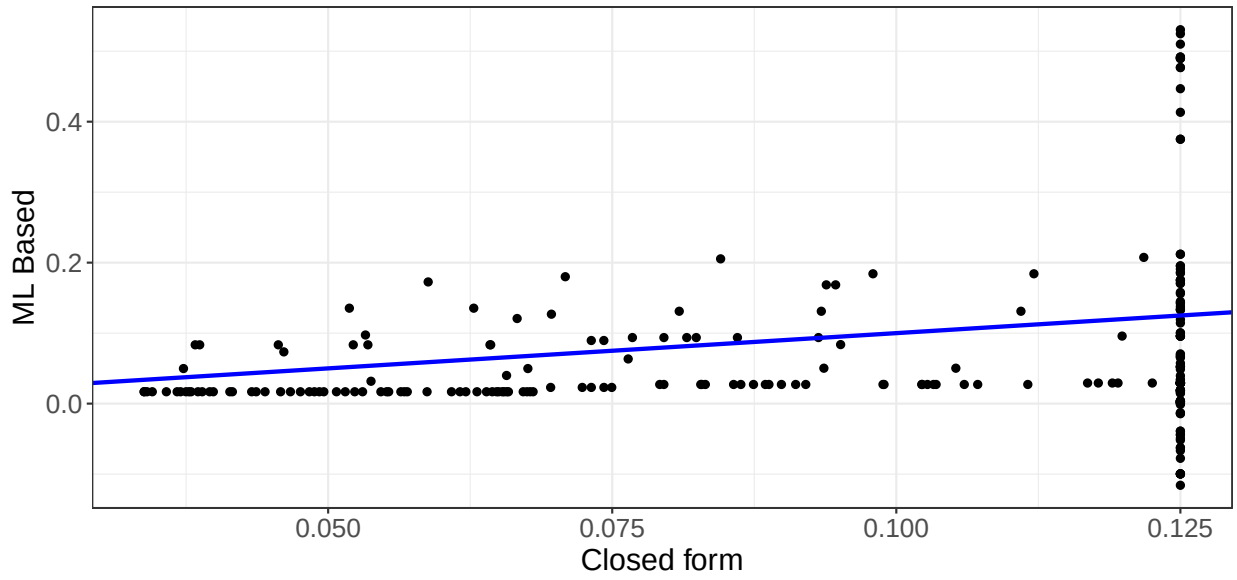


Figure 2.3.1: Machine Learning predictions and closed form: The distance between the blue line and the points represent deviations from the Machine Learning predictions.

Chapter 3.1

Impact of temporary traffic restrictions on NO pollution levels in Madrid

Abstract: This paper illustrates how Machine Learning techniques can be used to assess the impact of environmental protocols that are sparsely activated over time. A case study is analysed: the impact of a protocol that sets traffic restrictions on NO_2 levels in Madrid's urban area. The protocol specifies that these restrictions are active only when NO_2 level reaches above a certain threshold. Since the protocol was first enacted, in 2017, restrictions have only been active for 59 days, never longer than ten consecutive days. Cross effects are identified: the protocol magnifies the effect of other relevant features, especially wind speed. Pollution decreases with wind speed. Consequently, episodes of high pollution levels generally occur when wind speed is low. The analysis shows that precisely at these low values of wind speed, an increase in the wind speed has a higher effect on pollution when the protocol is activated than when it is not. The analysis is carried out at a measuring station level, considering eight representative stations. The referred cross effect is clearer at centrally located stations across Madrid. Cross effects of the protocol with other weather features are weaker than that with wind speed. The assessment is based on the computation of Shapley values for regression trees that are built using XGBoost and rolling time series windows.

JEL Codes: C01 (Econometrics), C14 (Semiparametric and Nonparametric Methods),

R40 (Transportation Economics), P25 (Urban, Rural, and Regional Economics).

3.1.1 Introduction

The air quality in big cities has become a central and key topic within environmental themes in modern societies. Citizens increasingly demand from policy-makers improved local environmental policies and these policy-makers, accordingly, have given an increasing weight to these issues in their agendas over recent years. After a reasonable time-span that any given policy has been enforced, an assessment of its performance comes in order. This paper addresses the question of the impact of such a policy and how much it has actually contributed to cleaner air in Madrid.

Specifically, Madrid's local government started to run in 2017 a protocol in which different traffic restrictions are triggered as a result of observed NO_2 levels. Roughly, those restrictions are gradually enforced on a daily basis, starting with a reduction in the speed limit from 90 to 70 km/h in Madrid's inner ring road (M-30). The protocol on traffic restrictions is only active as long as the NO_2 levels continue to lie above a certain threshold and usually last for a few days at a time. The protocol was first enforced in November 2017 and it has been active for only 59 total days since, never longer than ten consecutive days. This contrasts with other policies, such as low emission zones, which constitute a permanent shift, and thus its effects are more easily observable. Some further details on the protocol are given later in the paper.

The interest in assessing the contribution of this protocol is twofold. On the one hand, since the protocol only lasts for a few days at a time, it technically demands more than traditional linear regression analysis. On the other hand, the protocol has been long criticised by different social actors with the critiques persisting over time. Precisely because of this controversy, a politically neutral analysis of the available information is needed.

This paper proposes the use of a Machine Learning model in order to provide an assessment of the protocol's impact. This method has been proven useful for many other statistical problems across a wide variety of disciplines and fields. The model is able to identify highly non-linear dependencies from explanatory variables to a response variable from large amounts of data. Specifically, it is able to predict relatively few positives of the response, i.e: identify relatively few infectious diseased members within a given population sample. In line with that, the sample under consideration in this paper has relatively few positives of an explanatory variable, i.e: relatively few days in which the protocol in active.

The response variable is the daily measure of NO_2 levels from a number of measuring stations spread across Madrid's urban area. Specifically, eight measuring stations are included in the analysis, considered as representative due to their locations throughout Madrid and their distance to congested roads. Thus, there are eight response variables, one for each measuring station. The explanatory variables, aside from whether the protocol was active or not, consist of calendar and time-specific variables such as the month of the year and whether a given day was a working day or not, which are all binary variables. Additionally, a number of continuous weather variables such as wind speed and temperature are also included as explanatory variables. All of the explanatory variables are common to all measuring stations, although the response variable NO_2 levels, naturally vary across stations.

A brief description of the method is in order, with further details being left to a later section. Essentially, two elements are combined. The first is an Extreme Gradient Boosting (XGBoost) model, which represents the dependence from a set of explanatory variables to a response variable with a decision tree structure. Within the tree, there are several levels of branching, each branch being defined by an endogenously chosen value of an explanatory variable. The final node of the tree contains predictions of the response variable. A tree, or

a collection of trees, is called a model. The algorithm used (XGBoost), selects the model that minimises some loss function which penalises both prediction errors and the complexity of the model. Once a model is built, the second element is to use Shapley values to measure the importance of any given feature value, such as *the protocol being active*. Shapley values were first introduced in cooperative game theory and have been recently used in the Machine Learning literature. Essentially, the Shapley value measures the contribution of any given feature value by computing the difference in predictions between including and not including the feature value in a *coalition* of explanatory variables, averaging that difference across all possible coalitions.

In order to introduce the basic findings, we must start pointing out an endogeneity issue. The aim is to identify how much the protocol reduces pollution, but the protocol is activated only when pollution peaks. On the basis of observed data, it is hard to distinguish this statement from: pollution peaks because the protocol is activated. However, this issue is overcome by measuring the real impact of the protocol through cross effects. As it is well known, and it is also reported in this paper, wind speed reduces pollution. Accordingly, days with a high pollution level are also days with relatively low wind speed. The analysis presented in this paper shows that in those days, a small increase in the wind speed has a higher impact on pollution when the protocol is active than when it is not. This overall result exhibits some heterogeneity across measuring stations, such that, the effect is clearer in centrally located stations. Cross effects of the protocol with other weather features are also explored, particularly temperature, humidity and barometer. Roughly, it is possible to identify a cross effect with wind speed because in the sample there is enough variation in the wind speed values when the protocol is activated, which does not occur for the other variables. For instance, temperature varies greatly over the year in Madrid, but mostly between summer and winter, and the protocol has been never activated in the summer. Essentially, the analysis of these other features helps to understand necessary conditions to

identify cross effects.

The rest of the paper is organised as follows. Section 3.1.2 presents the background on air quality policies and previous literature. Section 3.1.3 describes the data. Section 3.1.4 discusses the methodology. Sections 3.1.5 contains the results, focusing on wind speed. Finally, section 3.1.6 concludes.

3.1.2 Background and Literature Review

This section is divided into two subsections. The first subsection summarises some basic facts on the overall performance of Spain's air quality and how road traffic affects pollution, particularly in urban areas. The second subsection focuses on the empirical literature that accounts for the impact on air quality policies, with a special emphasis on works that study urban air pollution.

3.1.2.1 Assessing air quality

In July 2019 the European Commission referred Spain to the European Court of Justice for its failure to protect its citizens from air pollution. Spain has infringed the Ambient Air Quality (AAQ) directive, see European Parliament (2008) by breaching 2 out of the 3 limits, PM_{10} and NO_2, with only Bulgaria breaching all 3 limits with the addition of breaching the SO_2.[1] In total there are 14 infringement cases pending against Member States for exceeding NO_2 limits (Austria, Belgium, the Czech Republic, Germany, Greece, Denmark, France, Spain, Hungary, Italy, Luxembourg, Poland, Portugal, and the United Kingdom).[2] According to the WHO global guidelines for NO_2 levels, the 1-hour mean should not exceed 200 ug/m^3 and the annual mean should not exceed 40 ug/m^3 (Directive 2008/50/EC).[3] EU countries are required to put in place procedures when levels exceed the Ambient Air

[1] https://op.europa.eu/webpub/eca/special-reports/air-quality-23-2018/en/
[2] https://ec.europa.eu/commission/presscorner/detail/en/IP_19_1475
[3] https://www.who.int/en/news-room/fact-sheets/detail/ambient-(outdoor)-air-quality-and-health

Quality Directives annual limits with the aim of limiting the exceeded levels to as short a time frame as possible.

Within the overall scenario describing the air quality, road traffic is often the driving causal effect of polluted air within urban areas despite different transport methods being identified as key sources of poor air quality, see Colvile et al. (2001). Borge et al. (2016) show that road traffic accounts for up to 90% of NO_2 emissions. Borge et al. (2012) also find that road traffic significantly contributed to NO_x emissions and that passenger cars, heavy-duty vehicles and buses are the main emitting sources of NO_2 in Madrid, with passenger cars emitting more emissions in the city centre and heavy-duty vehicles and buses emitting more on the outer areas of the city.

On-road transport is the main anthropogenic emitter of O_3 in Madrid and Barcelona metropolitan areas, accounting for 65% & 59% of NO_x and 67% & 85% of CO_x, see Valverde et al. (2016). Pérez et al. (2019) report that passenger cars are responsible for 80.7% of the total mileage made in Madrid. In fact, they are the main contributors of emissions with 65%, 73% and 72% for NO_x, CO_2 and $PM_{2.5}$, respectively. Salvador et al. (2004) identify sources of PM_{10}, they found that road traffic has explained almost (48%) of PM_{10} along with crustal (26%) and secondary particles (18%). O_3 records higher levels on the outer perimeter of the city in part due to reduced levels of NO.

The discussion of the quality of air and its effects on the health of inhabitants in a given region is not new. Odriozola et al. (1998) quantify the impact of air pollution on mortality and morbidity by measuring daily hospital admissions in Madrid. Sunyer et al. (1996) study the effects in Barcelona. Núñez-Alonso et al. (2019) show that there is a correlation between the pollutants NO, NO_2, PM_{10} and O_3 in the city of Madrid. They use Principal Component Analysis (PCA) and hierarchical clustering to classify stations based on pollution

levels. They show that the maximum annual average pollution levels are registered in the very centre of Madrid, and exceed the annual average threshold set by regulators.

There are multiple factors that may be the driving force behind certain pollutants being at high or low levels. Demuzere et al. (2009) investigate the relationship between climatology, O_3 and PM_{10} levels in the Netherlands. They find that rain and low humidity levels are significantly negatively correlated with PM_{10} in winter, which they state could point out atmospheric removal due to wet deposition. They find that wind speed is strongly negatively correlated with PM_{10} throughout the whole year, it is negatively correlated with O_3 in the wintertime and positively correlated with O_3 in the summer.

3.1.2.2 Empirical literature policies on air quality

The paper, Vedrenne et al. (2015) studies the effects of air pollution abatement and policy-making in Spain between 2000 and 2020. They find that the number of zones in infringement of the European air quality Directive 2008/50/EC for NO_2 and PM_{10} sustained reductions in their levels over the study period. They state that air quality policy-making has delivered improvements in air quality levels throughout Spain, mitigating the severity of the impacts on ecosystems and health. The decreases in NO_2 is found to be primarily down to measures concentrating on road-traffic along with industrial and domestic combustion processes. Whereas, the reduction in PM_{10} concentrations is associated with power generation and off-road sources. They state that in 2020 a number of air quality management zones will still infringe limits for both types of pollutants.

Considering specific policies at the city level, cities throughout Europe have adopted different methods to reduce pollution levels with varying degrees of access restrictions and rules for protected areas. Swedish cities, Stockholm, Malmo and Goteborg in 1996 were the first to apply such restrictions for heavy-duty vehicles, followed by Germany, the Netherlands,

Italy and London in 2007 - 2008. The remainder of this subsection lists some illustrative works grouped by the policy instrument under consideration.

Congestions charges. Green et al. (2018) analyse the case of London congestion charges which were introduced in 2003. They find significant reductions for a number of pollutants when compared to other cities for the same period. They focus on NO_2 levels, which is closely linked to diesel-powered vehicles. They state that, as a result of the congestion charge, commuters switched from personal cars to public transport, a mode of transport which more heavily relies on diesel fuel, thus they find that the reduction in other pollutants needs to be weighed against the adverse health effects associated with increases in NO_2 emissions which came from an increased use of other modes of transport, such as buses and taxis.

Low emission zones. The number of low emission zones has been on the increase with over 250 zones in place in Europe.[4] There is some literature for the effectiveness in EU cities, see Lutz (2009) for Berlin, Panteliadis et al. (2014) for Amsterdam, Carslaw et al. (2016) for London and Jiang et al. (2017) for German cities. For instance, Amsterdam introduced in 2006 restriction zones for heavy-duty vehicles. Regarding that policy, Panteliadis et al. (2014) find that NO_2 levels were reduced by 2.65 ugm^{-3} translating to a 4.9% reduction for areas under low emissions zones. They also found that NO_x and PM_{10} levels reduced by 5.6% and 12.9% respectively. In general, the empirical literature on low emission zones generally finds that the introduction of such policies has a net positive effect on reducing pollutants PM_{10}, see Holman et al. (2015) and Jiang et al. (2017). However, some empirical research fails to find significant results when analysing the effect on NO_2 levels. This could be down to shifts in the mode of transport of commuters when they commute to work within these low emission zones, shifting from petrol cars to diesel-operated public transport. Salas et al. (2019) use a *Diff-in-Diff* approach to evaluate the effectiveness of the Madrid Central

[4] https://www.transportenvironment.org/sites/te/files/publications/2019_09_Briefing_LEZ-ZEZ_final.pdf

low emission zones. They find a significant and robust reduction in NO_2 measurements once time effects and meteorological conditions are controlled for. They find that most of the monitoring stations outside the restriction zone exhibit statistically significant reductions in NO_2 levels, albeit by a lower amount. They state that these stations experienced a *spillover* effect and that pollution simply did not just transfer to other areas of the city and transportation habits changed within Madrid.

Transport models. Romero et al. (2019) study whether the use of more environmentally friendly transport models should be enforced when Madrid's NO_2 protocol is active. They analyse whether a shift from private transport towards transit occurs for the main transport link connecting Madrid to other municipalities in the Metropolitan Area. They find that travellers' behaviour in their modal choice of suburban trips is influenced by the mobility restrictions in the event of high pollution levels. They suggest that more severe measures should be implemented since lowering highway speed limits and parking restrictions did not evidence to be particularly effective for modal shifts towards public transport.

Technological improvements. Lumbreras et al. (2008) find that technological improvements linked to traffic-related European Legislation (EURO III-V) decreased emissions for most of the pollutants analysed, with CO_2 emissions expected to increase over their sample period, stating that the technological improvements would not be able to counteract the effect of a large mobility increase.

Speed limit. Perez-Prada and Monzon (2017) analyse the impact of speed limit reduction on traffic volumes, times and average speeds as well as NO_x and CO_2 emissions. They show that lowering speed limits from 90 km/h to 70 km/h on a section of Madrid's inner ring-road (M-30) has significant benefits in reducing these emissions and consequently benefiting the environment without substantially impacting traffic performance and adding

to travel times. However, they do not take into account weather features.

3.1.3 Data

This section presents a description of the data set used for the models. Essentially, the pollution level is the dependent -or response- variable and weather and seasonal features, together with the protocol being active or not, are the explanatory variables. The pollution level is measured at a station level for NO_2 on a daily basis for eight stations in the city of Madrid. This section is split into two subsections. The first subsection presents a description of the variables in the dataset. The second subsection introduces some stylised facts on NO_2 time series in Madrid.

3.1.3.1 Description

Daily pollution data was downloaded from the *Open Data* Madrid website, which is provided by the local government and contains data from 24 pollution stations located throughout the city of Madrid. The sample size goes from January, 1st, 2010 to October, 1st, 2019, both included, which consists of 3560 days. The pollution stations measure a number of different pollutants. The location of the stations are shown in Figure 3.1.8 in Appendix 3.1.7. The local government classifies stations by geography and by their proximity to pollution sources. There are stations located to the north, centre and south of Madrid. Regarding their proximity to the pollution source, there are three kinds of stations, labelled as Traffic, Background and Suburban. The Traffic stations are close to streets or roads with heavy traffic. Background stations measure the background exposition of the whole population. Suburban stations are located far from the city centre, which better captures the ozone levels. Eight stations have been selected for the analysis, whose typology is presented in Table 3.1.1. The shadowed cells in the table correspond to typologies for which Madrid has no stations. In the cells for the centre stations, two stations were selected, if available, in order to study intra-type variability. For the sake of space, most of the exposition and

graphical material in the main text focuses just on Escuelas Aguirre and Plaza Elíptica, though analysis was carried out on all eight stations.

	Traffic	Background	Suburban
North		Arturo Soria	Juan Carlos I
Centre	Escuelas Aguirre Plaza de España	Plaza del Carmen Mendez Alvaro	
South	Plaza Elíptica	Vallecas	

Table 3.1.1: Typology of selected stations

The protocol under analysis is regulated by Madrid's local government.[5] It establishes different pollution level thresholds and restrictions -or scenarios- that come into force as those thresholds are successively reached. The reference pollutant for the thresholds has been NO_2 for most of the sample period.[6] Under the least restrictive scenario, labelled as *scenario 1*, the speed limit is reduced in the city centre (defined by the inner-ring road, M-30), from 90 km/h to 70 km/h, together with other measures that imply no effective restriction, such as the recommendation to use public transportation. From this initial scenario, restrictions are successively added to define subsequent scenarios. Those additional restrictions include, from *scenario 2* onward, restrictions to private vehicles that have no *green label* from circulating within the city centre.[7] During the sample period the protocol has been active a total of 59 days as shown in Table 3.1.3, in Appendix 3.1.7.

[5] See:

1. https://www.madrid.es/UnidadesDescentralizadas/Sostenibilidad/CalidadAire/Ficheros/ProtocoloNO2AprobFinal_201809.pdf
2. http://www.mambiente.munimadrid.es/opencms/opencms/calaire/Episodios/listados_informes_episodios/index.html

1 shows the current legal text, 2 contains the days in which the protocol has been active. Both links are in Spanish.

[6] In addition, ozone levels have started to be considered since 2019.

[7] There are different categories of green labels, depending on the vehicle registration date and fuel type.

The dataset also contains data on seasonal features, including public holidays, and data on weather conditions in Madrid.[8] The source of data provides both, real-valued and categorical features on the weather. The real-valued features are wind speed, temperature, humidity and barometer, which are provided on an hourly basis and aggregated to a daily frequency. Furthermore, agents' decision on taking the car vs using public transportation might depend as well on features that have a more categorical nature, like a *Sunny* vs *Rainy* day. For this reason, some categorical features were also accounted for. Obviously, these features raise some complexity: it could be *Sunny* in the morning, then turn to *Passing Clouds* in the mid-afternoon and turn to *Scattered Clouds* later in the day, this day will have multiple dummy variables describing its weather conditions. Table 3.1.4, in Appendix 3.1.7, shows the description of the seasonal and weather features.

A final comment on missing values is in order. The Machine Learning algorithm considered in this paper - introduced in the next section - allows for missing values in the explanatory but not in the response variables, since the model cannot learn to predict nothing. For this reason, a Kalman filter with a structural model fitted by maximum likelihood was applied in order to impute the missing values in the pollution levels. This imputation method has been proved useful for imputing missing univariate time-series data in other studies, see Moritz et al. (2015). The number of missing values in the response variable differs across the stations, with 31.5 missing observations on average, the maximum is 60 for one station. Since there are so few missing values in the response variable, the choice of the imputation method is expected to have a negligible impact on the analysis.

3.1.3.2 Stylised facts on NO_2

The NO_2 pollution level time series has a strong seasonal component, without a clear trend over the sample period under consideration and with differences among stations that are

[8]Public holidays in Madrid and Spain: https://www.timeanddate.com/holidays/spain/. Weather data for Madrid: https://www.timeanddate.com/weather/spain/madrid/historic.

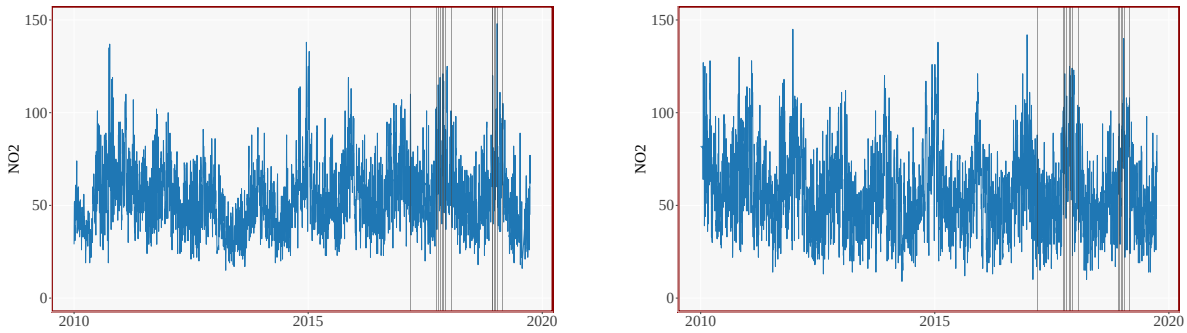


Figure 3.1.1: **Pollution levels.** Daily data for two traffic stations. Escuelas Aguirre (left panel) and Plaza Elíptica (right) are located to the Centre and South of Madrid, respectively. Grey vertical lines correspond to days in which the protocol was active.

persistent over time. Figure 3.1.1 shows daily levels for two representative traffic stations over the sample period. Throughout the paper, all figures plotting daily time series show grey vertical lines for days in which protocol was active.

Figure 3.1.2 emphasizes the seasonal component of the daily series plotted in Figure 3.1.1 for Escuelas Aguirre. The upper-left panel shows the month on month average NO_2 level for each of the years in the data. Nitrogen dioxide emissions are originated from fuel burning, thus it is closely related to road traffic, which in Madrid is higher in winter than in the summer. The seasonal patterns are common to all measuring stations. The upper right panel shows a polar plot and the lower panel shows the monthly average over the whole period.

Figure 3.1.3 removes the seasonal component by showing a 365 day rolling average for the eight stations under consideration. The figure illustrates that differences across stations are apparent when the seasonal component is removed. Escuelas Aguirre lies among the two highest levels along the whole sample period. In addition, while there is not a clear trend over time, there are yearly variations. Recall that the vertical grey lines represent dates at which the protocol was active. In this regard, Figures 3.1.1 and 3.1.3 illustrate the complexity of identifying the effect of the protocol due to the relative scarcity of such days.

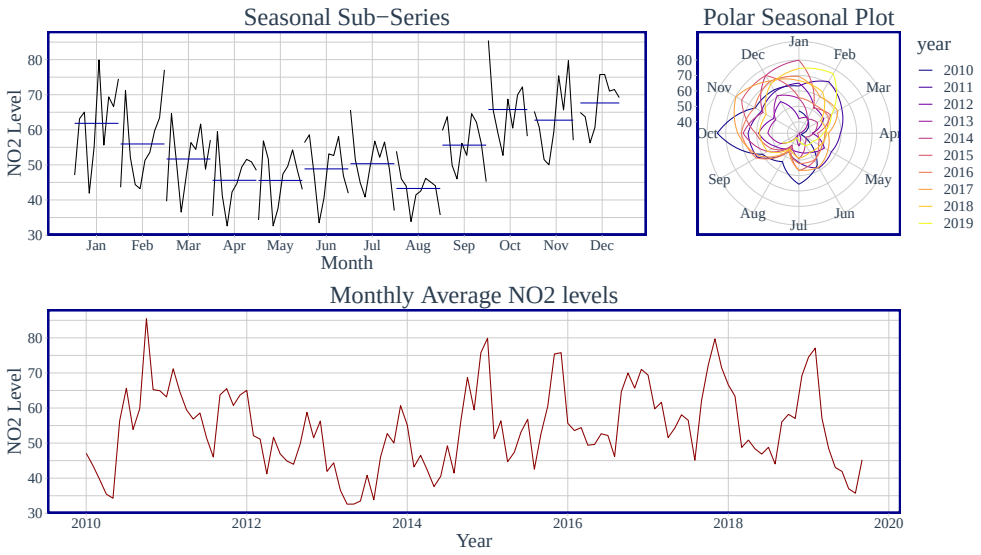


Figure 3.1.2: **Seasonal component for Escuelas Aguirre**. The upper-left panel gathers data by month for all years and the horizontal line is the corresponding monthly average. The upper right panel shows similar information using a polar plot. The lower panel shows the monthly average.

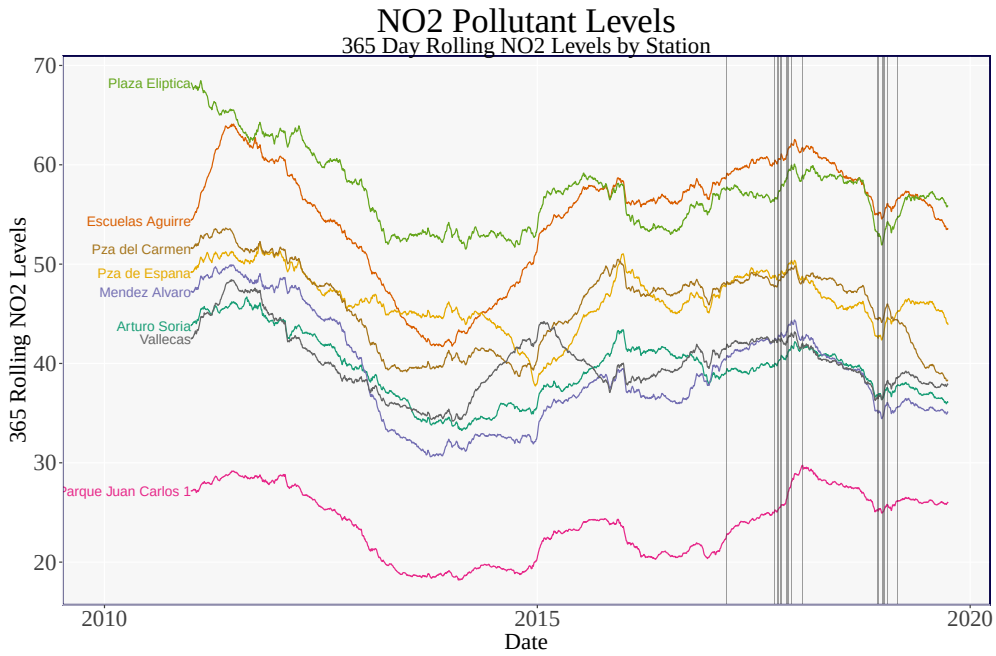


Figure 3.1.3: **365 day rolling mean levels**. Each station is represented by a different colour. The grey vertical lines correspond to days of active protocol.

According to the WHO global guidelines for NO_2 levels, the 1-hour mean should not exceed 200 $\mu g/m^3$ and the annual mean should not exceed 40 $\mu g/m^3$ (Directive 2008/50/EC).[9] In this regard, Figure 3.1.3 shows that many locations throughout Madrid are consistently exceeding the threshold guideline set by the WHO over an annual period and failing to fall below *safe* levels at all. According to Borge et al. (2016), 80% of the monitoring stations in Madrid exceeded the ambient air quality standards in 2007. Madrid set a target of reducing NO_2 levels in its local air quality plan for the period 2011 to 2015, see de Madrid (2012). Borge et al. (2016) find that over that time-span there was a positive trend and that 2015 presented a generalised non-compliance situation with 13 (out of 24) monitoring stations recording NO_2 annual average concentration levels above the annual limit value (40 ug/m^3) imposed by the European air quality Directive.[10] Figure 3.1.3 shows a downward sloping trend of the 365 day rolling average prior to 2015 for all stations, though it does not continue thereafter. Figure 3.1.3 also shows an interesting response of the rolling average when the protocol is active, indicated by the horizontal grey lines. All stations record a drop in their averages when the protocol is active, with stations located in the city centre showing significantly more reaction than stations located on the outer perimeter of Madrid.

3.1.4 Methodology

The data is split into a rolling window time-series dataset. Let t index time, with $t \in \{1, 2, \ldots, T\}$. The dataset has data for $T = 3560$ days. Windows have a fixed length, denoted by τ. The first window consists of data for days $\{1, 2, ..., \tau\}$. The model is trained with this data in order to predict the *response variable*, NO_2 pollution level, at day $\tau + 1$. Next, the second window follows: the model is trained with data for days $\{2, 3, ..., \tau + 1\}$ in order to predict the response variable at day $\tau + 2$, and so forth until a prediction for day

[9] https://www.who.int/en/news-room/fact-sheets/detail/ambient-(outdoor)-air-quality-and-health

[10] Borge et al. (2016) also report that 8 (out of 24) stations exceed the hourly limit (more than 18 hours with concentrations over 200 ug/m^3).

T is reached. Moreover, $\tau = 2190$, which, for daily frequencies, corresponds to six years of data. For instance, the first training window goes from January, 1st, 2010, which is the first day in our sample, to December, 31st, 2015. That window is used to train a model in order to make the first prediction, for January, 1st, 2016. The last day for which a prediction is made is October, 1st, 2019, which is the last day in the sample. This procedure is carried-out separately for each measuring station. While the response variable, the pollution level, is station-specific, the features are common to all stations: weather conditions, seasonality variables (day of week, month,...) and whether the protocol was active or not. One of the contributions of this paper is precisely to show that the impact of the protocol changes across stations.

Each window is used to generate a model, which combines *features*, or *explanatory variables*, measured on a daily basis, such as weather conditions or whether the protocol was active or not, to predict the response variable - the NO_2 level at a given measuring station. There are 1369 days between January 1st 2016 and October 1st 2019 and for each of those days a prediction based on a model is made, a total of 1369 models from the rolling time series data are generated.

Each model is built using *Extreme Gradient Boosting*, or XGBoost, developed by Chen and Guestrin (2016). Essentially, each model is the combination of a set of *weak learners*, each learner being a *directed tree*, using the terminology of graph theory. A tree is a step function that maps a vector of characteristics, or features, into a score that correlates positively with the response variable. An example of a tree is depicted in Figure 3.1.4. From a function space of step functions, or trees, the algorithm selects sequentially trees in order to minimise a loss function which penalises both prediction errors and the complexity of the tree.[11]

[11] More specifically, each dataset conforming to a window is partitioned in training and testing sets, which are used first to train and then to test the algorithm, respectively.

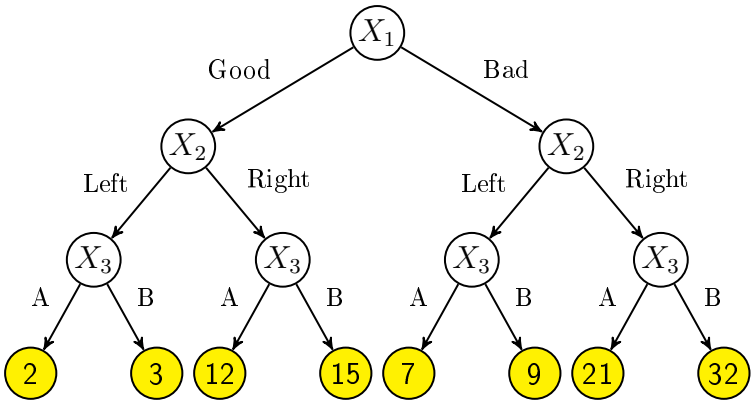


Figure 3.1.4: **Example of a tree.** There are three features, X_1, X_2 and X_3, in white nodes. The set of arrows from each white node constitutes a partition of the possible values of the corresponding feature at that node. The yellow nodes are terminal nodes, which contain the *score*, or predicted value for the response variable. The score for an *instance* with feature values $\{X_1 = Good, X_2 = Left, X_3 = A\}$ is 2, shown in the leftmost path. For simplicity, this tree assumes binary features and that any given feature has the same branching from all of its nodes, which is not essential.

The main target of this paper is not on predicting pollution levels but on measuring the impact of the different feature values, with a particular focus on the protocol being active. For that, Shapley values are used, which is a classical concept in cooperative game theory, see Shapley (1953), and has been recently applied to understand Machine Learning model predictions, see Lundberg and Lee (2017). In order to introduce the problem, consider again the tree in Figure 3.1.4. Assume that the response variable is the pollution level, whose scores (predictions) are in the terminal -yellow- nodes. A simple inspection of the tree shows that the feature value $\{X_1 = Good\}$ reduces pollution with respect to $\{X_1 = Bad\}$, analogously for $\{X_2 = Left\}$ and $\{X_3 = A\}$ with respect to their corresponding alternative values, but it is not obvious how to assign a quantitative measure to the marginal impact of each feature value.

In order to illustrate the basic idea on how Shapley values delivers such assessment, a simplified example for the feature value $\{X_3 = A\}$ follows. Consider first a *coalition*, denoted by S, formed by a single feature value, say $S = \{X_1 = Bad\}$. The contribution of $\{X_3 = A\}$

to the coalition S is measured by the difference in predictions between conditioning by S and $S \cup \{X_3 = A\}$. In order to simplify non-essential details, assume that all of the branches from any given node are equally populated in the sample, thus half of the instances -a fixed day and station- have $\{X_1 = Good\}$ and the other half have $\{X_1 = Bad\}$, and so forth down to the terminal nodes. Thus, conditional expectations can be easily computed to make predictions. Let Y denote the score, then:

$$E\{Y \mid X_1 = Bad, X_3 = A\} = \frac{1}{2}(7 + 21) = 14,$$
$$E\{Y \mid X_1 = Bad\} = \frac{1}{4}(7 + 9 + 21 + 32) = 17.25.$$

The contribution of $\{X_3 = A\}$ to the coalition S is $14 - 17.25 = -3.25$, where the negative sign indicates that $\{X_3 = A\}$ *reduces pollution with respect to* S. Next, consider all other coalitions in which $\{X_3 = A\}$ might contribute, which are all combinations of feature values of X_1, X_2 and both, like $S' = \{X_1 = Good, X_2 = Right\}$, and compute the contribution of $\{X_3 = A\}$ to each of those coalitions. The Shapley value for $\{X_3 = A\}$ is its weighted average contribution across all coalitions.[12]

The Shapley values satisfy the conditions *Efficiency*, *Symmetry*, *Dummy* and *Additivity* which define a fair payout, see Lundberg and Lee (2017). It might be considered as an alternative to permutation feature importance. The latter uses the decrease in the model's performance, whereas Shapley values use the magnitude of feature attributions. An alternative and traditional concept for variable selection is the Gain, see Breiman et al. (1984). However, the output from tree-based models relies on the order in which a model sees a given variable and, in turn, affects the model's predictions and importance scores, which

[12]The contribution to a coalition that occurs frequently has a higher weight. In general, the algorithm relies on Monte Carlo sample-based simulations to numerically compute the expectations above.

makes it inconsistent, see Lundberg et al. (2018) and Lundberg et al. (2019).[13] In contrast, Shapley values are *model-agnostic*, meaning that they do not depend on the structure of the model. In the remainder of the paper, we use the term *Shap values* or *scores*, indistinctly, instead of Shapley.

The application of XGBoost to the sample requires two additional details. First, in order to compare the Machine Learning model's results across stations over time, each of the model's response variable (NO_2) was standardised using the mean and standard deviation from each of the in-sample-training sets across all stations. Alternatively, scaling using the whole dataset from 2010 to 2019 for each station would introduce look-ahead bias into the model. Only NO_2 data were scaled, while all explanatory variable observations remained unscaled since those variables are non-station specific. Second, the default XGBoost parameters of the algorithm implemented in `R` were used, see Chen et al. (2020b), since searching for the optimal parameters on each window, as mentioned, there are 1369 models for each measuring station, would be computationally too expensive for such a time series task. The default parameters have been found to perform reasonably well in a number of other prediction-based models in the Machine Learning literature. Yet, a small experimentation was carried out using different parameter values for some randomly selected periods over the time series and for a few stations. The results were similar to the ones presented in the paper.

3.1.5 Results

The main objective is to capture the marginal contribution of the protocol to reduce NO_2 pollution levels. The focus is on the relationships among the features that have the largest variation in Shap scores. This section shows the impact of the protocol by itself is negligible,

[13] The *Gain* is the total reduction of loss or impurity contributed by all splits for a given feature. As for the *Gain*, the relative contribution of the corresponding variable to the model is calculated by taking each variable contribution for each tree in the model. Higher values when compared with other variables implies that this variable is more important for generating a prediction.

which is in part down to the fact that there are too few observations of the protocol being activated. However, there is evidence that it might strengthen the effect of other factors, particularly wind speed. In short, departing from a low wind speed, an increment in the wind speed reduces pollution, and for some stations that reduction is larger when the protocol is active. This is termed *cross* effect.

While an analysis for all-weather features in the dataset is carried out, this section focuses on wind speed, for which the cross effect is more clear. Figure 3.1.5 plots the average Shap values for each of the trained models. That is, the first average Shap value occurs on January, 1st, 2016 for which it shows the average Shap value for the period January, 1st, 2010 until December 31st, 2015. The figure illustrates some differences across the two stations under consideration. Moreover, the most important message is that the daily average Shap value is positive, which -following the discussion in the previous section- might lead to the surprising conclusion that wind increases pollution. This example helps to understand the basic ideas in computing Shap values. In order to simplify the reasoning, consider only two possible values for wind speed, say *low* and *high wind speed*. As it is well known, a higher wind speed reduces pollution. In line with that fact, the model learns that low wind speed *goes with* high pollution and high wind speed goes with low pollution. If, as is the case for Madrid, low wind speed is relatively much more frequent than high wind speed, the average across feature values returns a single Shap value for just *wind*, as the figure does, and that value might lead to the conclusion that wind increases pollution. Thus, the essential message of the figure is that low wind speed days are much more frequent than high wind speed ones, which will be important for the analysis.

Figures 3.1.6 shows daily Shap scores of feature values for four continuous magnitudes related to weather conditions for stations Escuelas Aguirre where each point is a day. In contrast to the previous figure, this figure shows Shap values for different feature values.

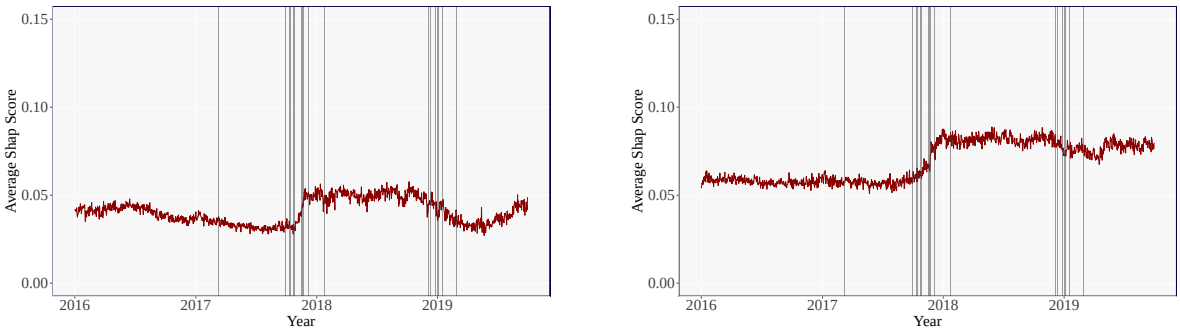


Figure 3.1.5: **Shap values for wind speed.** Daily average of Shap scores for wind speed for Escuelas Aguirre (left panel) and Plaza Elíptica (right).

For instance, there is a positive Shap value when wind speed is low (yellow) and a negative Shap value when the wind speed is high (blue). As mentioned, a negative Shap value means that the corresponding feature value reduces pollution. Blue points are scarce compared to yellow since low wind speed is relatively more frequent in Madrid. There are three panels, each one considering a different training time-period, with the more recent at the bottom, which essentially shows stability over time of the Shap values. The figure also shows Shap values for other features: temperature, humidity and air pressure. In general, extreme values of the wind speed have a higher Shap score, either positive or negative, than extreme values of any other feature. Temperature has a seasonal component: higher temperature occurs in the summertime, with lower road traffic. The figure has similar patterns for all stations under analysis.

In order to simplify the overall exposition, the weather features of a categorical nature, such as the month of the year or weekday, are omitted from Figure 3.1.6. Regarding months of the year, the summer months have lesser road traffic in Madrid, and consequently have negative Shap scores or, equivalently, reduce pollution. In other words, the seasonal component in the pollution time series presented in the previous section is translated into the Shap scores assigned to each month.

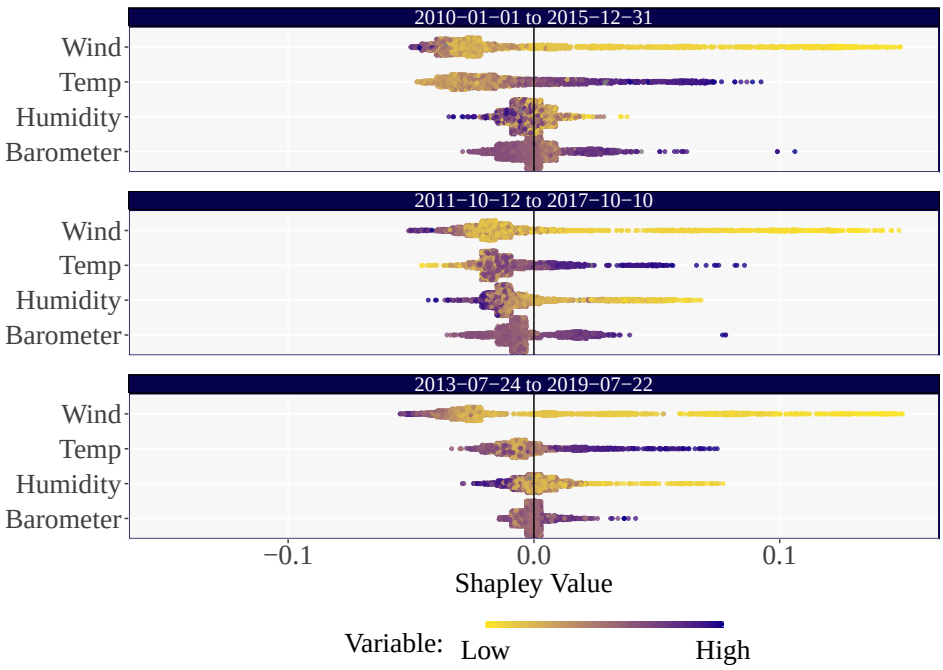


Figure 3.1.6: **Shap scores for a range of feature values: Escuelas Aguirre**. Each point represents Shap scores of a given feature on a given day. The colour denotes the corresponding feature value. Features are wind speed (Wind), temperature (Temp), Humidity and air pressure (Barometer). The differences between the three panels are the training time window under consideration. From top to bottom panel, periods are 2010-01-01 to 2015-12-31, 2011-10-12 to 2017-10-10 and 2013-7-24 to 2019-7-22, respectively.

Figure 3.1.9, presented in Appendix 3.1.7, is analogous to Figure 3.1.5 but for the protocol variable. As mentioned previously, vertical grey lines correspond to days in which the protocol was active. The figure shows a logical behaviour of the model: it assigns zero Shap scores to the protocol until it is activated for the first time. Thus, the figure shows a zero until the first grey line, which of course, is common to all stations. Furthermore, after the first date on which the protocol was activated, the Shap scores are positive, which might lead us to think that protocol increases pollution. In addition, there is a large variance across stations, both in the variability and in the level of the series.

Figure 3.1.7 is central to the analysis. It shows dependencies that explain why the models assign a positive Shap score to the protocol. Essentially, when it tries to assign a Shap

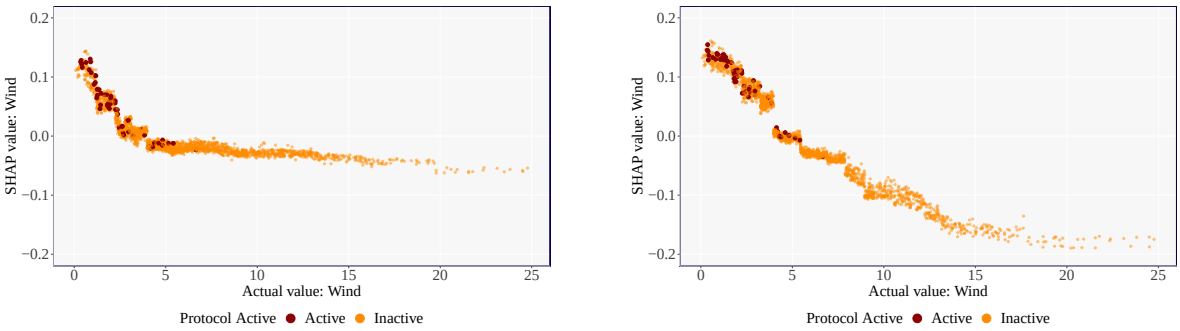


Figure 3.1.7: **Shap dependence for wind speed values and the Protocol.** Each point is a day. Red (Orange) points are for days in which the protocol was (not) active. Left and right panels are for Escuelas Aguirre and Plaza Elíptica, respectively.

value to the protocol, it faces an *endogeneity* problem. The aim is to observe how much the protocol helps to reduce pollution, but in fact, the protocol is only activated when pollution levels are at their highest over the whole time series, so what the model learns is the opposite direction of causality: the highest pollution levels are *caused by* the protocol. Wind speed has the highest impact on pollution among all of our set of weather features: high wind speed reduces pollution (negative Shap values) while low wind speed increases pollution (positive Shap values). This is illustrated both in Figures 3.1.6 and 3.1.7. But, in addition, this latter figure shows that the protocol is only activated precisely when wind speed is low.

Figure 3.1.7 suggests an alternative measure to overcome the endogeneity issue. Instead of looking at the direct effect of the protocol on the pollution level, the key is an *indirect* or *cross* effect: does a change in the wind speed have a higher impact when the protocol is active? The basic idea is the following. Consider variations in the Shap scores between the minimum and maximum wind speed within the range of wind speed values for which the protocol is activated, that is, consider the *red* and *orange* points in Figure 3.1.7 for approximately the lowest 10% range of the wind speed values - or for wind speed values between 0 and 5, thus capturing all of the observations the protocol was activated. Some of these variations in the Shap scores of the wind speed have occurred when the protocol

was active and some others, a majority in fact, while it was not. If the protocol is helping to reduce emissions, then the variations in Shap scores (in absolute values) would be larger when the protocol is active as opposed to when it is not active as the wind speed increases. Operationally, for each station two regressions were run, in both of them, Shap scores were regressed against wind speed, restricting the sample to approximately the lowest 10% of the wind speed values. One regression considers only points in which the protocol was active while the other was run with points in which the protocol was not active. If the protocol helps within the low wind speed range, an increase in the wind speed has a higher impact, and thus, a further reduction in the Shap score, for the *active protocol* data, that is, for the red points in Figure 3.1.7, thus the corresponding regression slope is steeper.[14]

The regression results are shown in Table 3.1.2 for the models in Figure 3.1.7 and the corresponding Figures in 3.1.7.2 in Appendix 3.1.7, which shows the slope of the regression lines for active and non-active protocol samples, respectively, for each of the eight stations under consideration. When the regression line for the active case is stepper (larger in absolute value) than the non-active case, the protocol is helping and then a check-mark is shown in the last column of the table. This indirect measure of the impact of the protocol brings positive news: for six out of eight stations, the protocol is helping. Furthermore, looking at the typology of the stations for which protocol does not help, they are located in the north of the city. Probably more importantly than its geographical location, using Figure 3.1.3, the two stations for which the protocol fails to help are those with lower emission levels once seasonality is discounted.

[14] Analogous figure to Figure 3.1.7 for all eight stations are included in Appendix 3.1.7

Station	Type	Active	Non-active	Protocol helps?
Arturo Soria	North, Background	-0.015	-0.015	
Juan Carlos I	North, Suburban	0.001	-0.002	
Escuelas Aguirre	Centre, Traffic	-0.060	-0.058	✓
Plaza España	Centre, Traffic	-0.044	-0.034	✓
Plaza del Carmen	Centre, Background	-0.032	-0.032	✓
Méndez Alvaro	Centre, Background	-0.032	-0.027	✓
Plaza Elíptica	South, Traffic	-0.013	-0.012	✓
Vallecas	South, Background	-0.073	-0.065	✓

Table 3.1.2: Slope of regression lines for Shap dependence plots

Table 3.1.2 shows a snapshot of one of the time series models for the most important variable in the model: wind. The steadiness over-time of the results of the table as well as a similar analysis for other weather features were also explored. The cross effect of the protocol with these other features is not so clear. The underlying reason is simple. In order to identify cross effects, it is necessary to have enough variability of the corresponding feature both when the protocol is activated and when it is not, and the sample under analysis does not exhibit such variability for other weather features.

In order to see if the protocol variable had any meaningful contribution to the model's prediction, some additional noise was introduced into the protocol variable. Additional *fake* days were added with the protocol suggesting that on these days the protocol was active - when it was in fact not active in reality. Figure 3.1.13, in Appendix 3.1.7, shows a similar plot to Figure 3.1.9 in Appendix 3.1.7 but with the addition of some light grey lines which represent the dates selected as the fake dates. Only dates similar to the dates in which the protocol was actually activated were selected - i.e. no fake dates in the summer months. The figure shows that the model reacts minimally to the introduction of these fake dates across all stations and only reacts substantially after the first introduction of the actual protocol being activated, and reacts even more when it sees more protocol active dates. It seems plausible that when the model observed the fake dates there was no *shock* to the

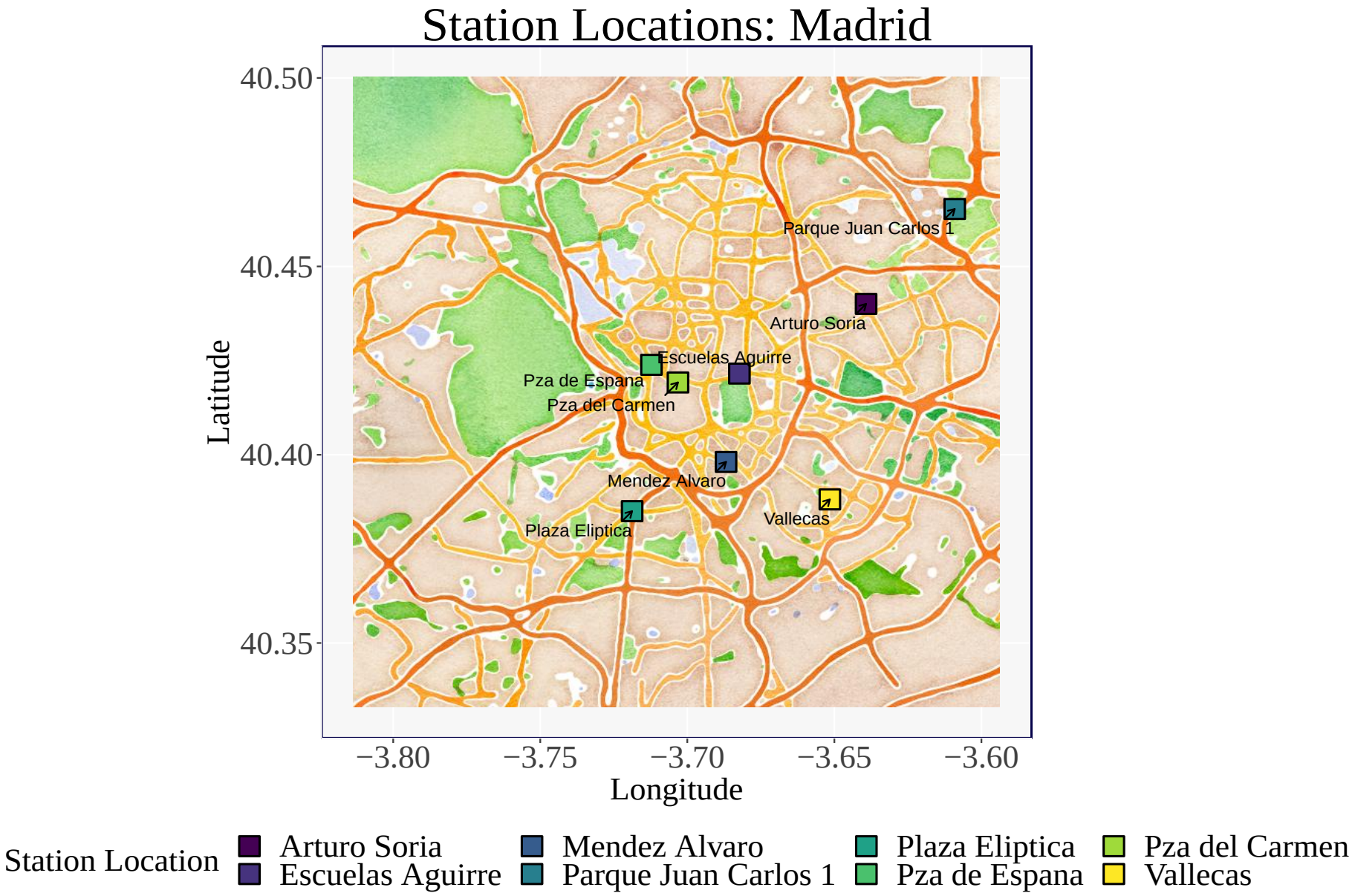
NO_2 levels during this time, however, when the model first encountered the date in which the protocol was first introduced, there would have seen a shock to the NO_2 levels on that date - i.e. reduced speed limits would have reduced the NO_2 levels on that day, the model sees this and attributes this shock to reductions in NO_2 levels. Appendix 3.1.7 delivers also some measure of the goodness of fit of the models.

3.1.6 Conclusions

The evaluation of environmental protocols which are active just a few days per year demand statistical techniques that go beyond standard linear models. At the same time, some of those protocols are controversial. This paper considers a case study of one such protocol: gradually increasing traffic restrictions when the NO_2 levels go beyond some threshold value. The protocol has been active for just 59 days since it started to operate in 2017. The cost of such a protocol for citizens, when commuting in a big city like Madrid, are far more obvious than its positive impact on pollution reduction.

This paper shows that Machine Learning methods can shed light on the evaluation of policy decisions which may be difficult to quantitatively measure using traditional regression models. The dataset contains public daily data on pollution and a number of weather and seasonal features. The basic finding is that the protocol magnifies the effect on pollution reduction of an increase in wind speed. Other weather features have also been analysed, but their impact is not as clear, probably because there is not enough variation in these variables. Interestingly, the cross effect between protocol active and wind speed is heterogeneous across measuring stations in Madrid. It is particularly stronger in more centrally located stations. This paper suggests that a rigorous treatment of publicly available information might help citizens in big cities to know the reward of some pollution reduction protocols.

3.1.7 Appendix



(a)

Figure 3.1.8: Map of Station Locations in Madrid

3.1.7.1 Shap scores

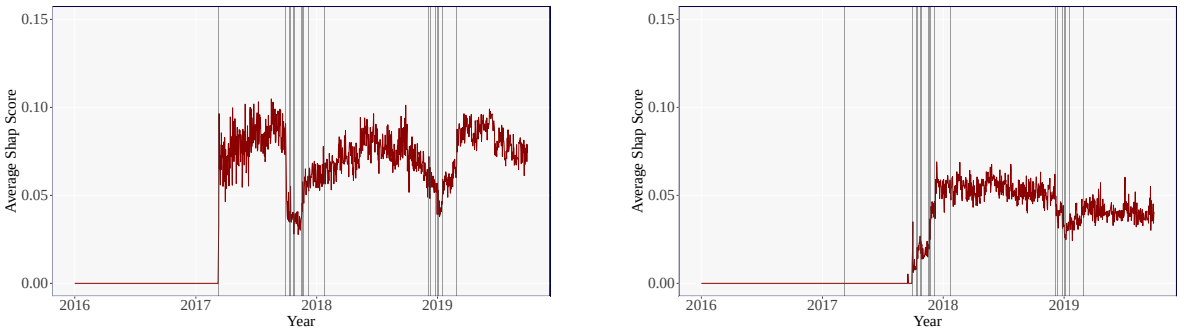


Figure 3.1.9: **Shap values for the Protocol.** Daily average of Shap scores for the Protocol. Left and right panels are for Escuelas Aguirre and Plaza Elíptica, respectively.

3.1.7.2 Shap dependence

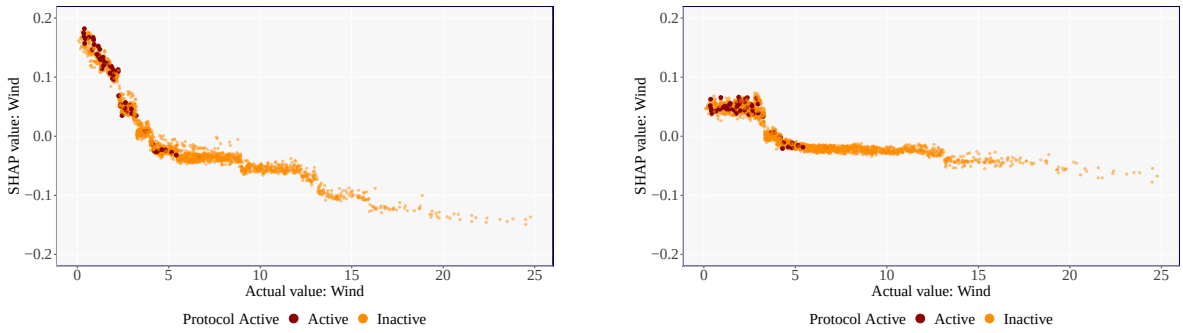


Figure 3.1.10: **Shap dependence for wind speed values and the Protocol.** Plaza de España (left panel) and Juan Carlos I (right) are traffic and suburban station, respectively.

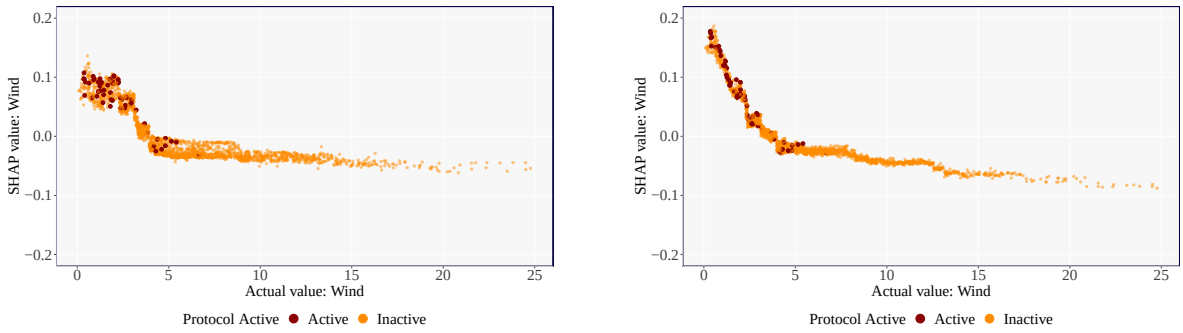


Figure 3.1.11: **Shap dependence for wind speed values and the Protocol.** Arturo Soria (left panel) and Vallecas (right) are both background stations.

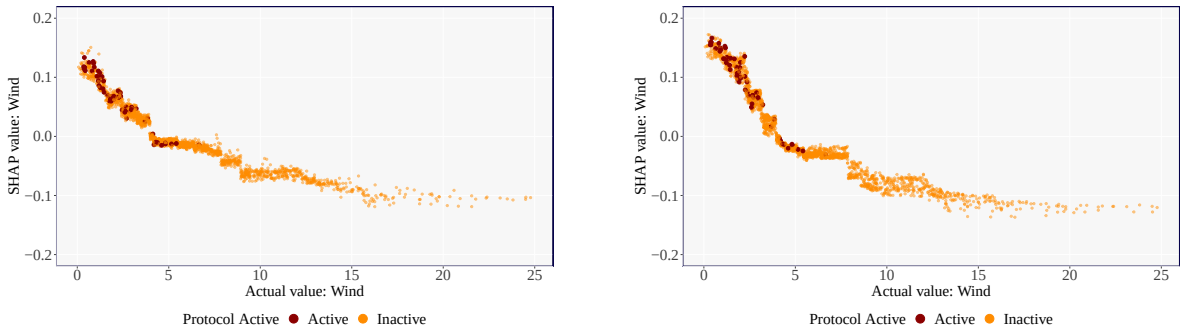


Figure 3.1.12: **Shap dependence for wind speed values and the Protocol.** Plaza del Carmen (left panel) and Méndez Alvaro (right) are both background stations.

3.1.7.3 Data characteristics

Start	End	Days Active
2017-03-09	2017-03-11	3
2017-09-28	2017-10-01	5
2017-10-10	2017-10-15	6
2017-10-23	2017-10-28	6
2017-11-15	2017-11-24	10
2017-12-05	2017-12-08	4
2018-01-23	2018-01-24	2
2018-12-05	2018-12-07	3
2018-12-11	2018-12-12	2
2018-12-25	2018-12-28	4
2019-01-01	2019-01-06	6
2019-01-15	2019-01-18	4
2019-02-26	2019-03-01	4
	Total	59 days

Table 3.1.3: NO_2 Protocol Dates

Feature	Values
Temperature	Fahrenheit
Wind speed	MPH
Humidity	Percentage
Barometer	HG
Weather categorical	Mostly Cloudy, Passing Clouds, Scattered Clouds, Fog, Sunny, Light Rain and Rain
Holiday	Binary
Month	Integer (1 to 12)
Weekday	Binary
Weekend	Binary
Weekend on Holiday	Binary

Table 3.1.4: Seasonal and Weather features

3.1.7.4 Robustness and prediction

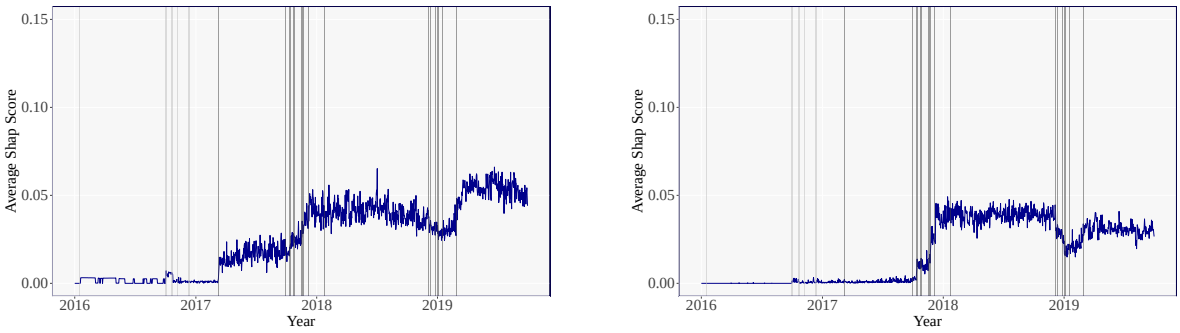


Figure 3.1.13: **Robustness for protocol with fake dates.** The light grey lines from 2016 to 2017 indicate fake *protocol active* dates. The darker grey lines are the actual dates the protocol became active. Left and right panel are for Escuelas Aguirre and Plaza Elíptica, respectively.

Machine Learning is largely focused on prediction. In each of the 1369 models it was held out a sample of 1 observation for each model, thus there are 1369 *out-of-sample* testing points. These *out-of-sample* predictions were compared to the actual observed NO_2 levels for each station. Table 3.1.5 shows a series of performance metrics.

The analysis here is restricted to the Symmetric Mean Absolute Percentage Error (SMAPE), see Makridakis (1993). The lowest prediction errors are for Plaza Elíptica, located in the South West, with errors of around 26%, followed by three stations located closest to each other in the centre, Plaza España, Plaza del Carmen and Escuelas Aguirre, with errors ranging between 29.0% and 31%. The prediction errors for stations Arturo Soria, Mendez Alvaro and Vallecas, located in the East, South and South East, respectively, range between 38% and 43%. Finally, the highest error is for Juan Carlos I, located in the North East, with an error of 57%.

It appears that the station with the lowest SMAPE score, Plaza Elíptica, has the highest (365 day) rolling average pollution levels, whereas the highest SMAPE station, Juan Carlos I, has the lowest pollution levels, according to Figure 3.1.3 - recall that all stations NO_2 levels were scaled between the minimum and maximum values across all stations. Plaza España and Plaza del Carmen follow a similar 365 day rolling average time series and they have a comparably similar SMAPE score also.

Station	*Performance Metrics*				
	MSE	**RMSE**	**MAE**	**MAPE**	**SMAPE**
	$\frac{1}{N}\sum_{i=1}^{N}(y_i-\hat{y}_i)^2$	$\sqrt{\text{MSE}}$	$\frac{1}{N}\sum_{i=1}^{N}\lvert y_i-\hat{y}_i\rvert$	$\frac{1}{N}\sum_{i=1}^{N}\lvert\frac{y_i-\hat{y}_i}{y_i}\rvert$	$\sqrt{\frac{1}{N}\sum_{i=1}^{N}\frac{2\cdot\lvert y_i-\hat{y}_i\rvert}{\lvert y_i\rvert+\lvert\hat{y}_i\rvert}}$
Arturo Soria	0.0149	0.1221	0.1002	0.3613	0.4369
Escuelas Aguirre	0.0161	0.1269	0.1022	0.3459	0.3184
Mendez Alvaro	0.0151	0.1227	0.0991	0.3640	0.4349
Juan Carlos I	0.0126	0.1122	0.0922	0.4615	0.5730
Plaza Elíptica	0.0126	0.1123	0.0866	0.2611	0.2666
Plaza España	0.0095	0.09767	0.0763	0.2992	0.2932
Pza del Carmen	0.0089	0.0943	0.0773	0.2694	0.2962
Vallecas	0.0120	0.1092	0.0887	0.3506	0.3883

Table 3.1.5: ***Performance errors*: For all 8 stations:** The data was previously scaled based on the minimum and maximum values across all stations for each time-series rolling model.

Chapter 3.2

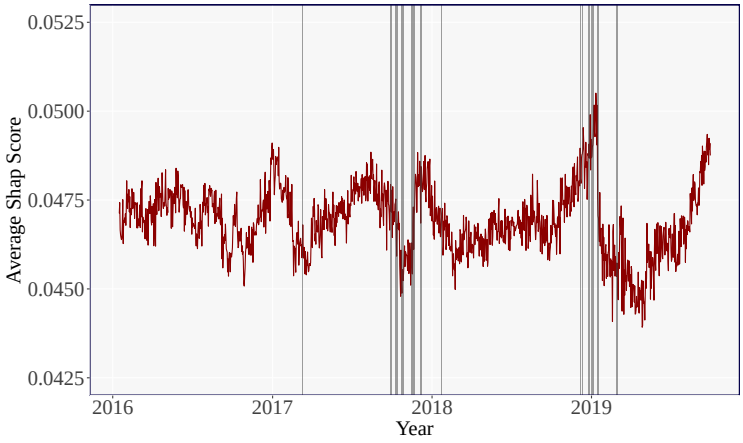
Lag and Lead Protocol: Search for Instrumental Variables

This section aims to address the problem of endogeneity of the explanatory variables. The reverse causation occurs with the activation of the protocol. The protocol is activated when the pollution levels in Madrid pass a given threshold and the protocol is deactivated when the levels fall back down.

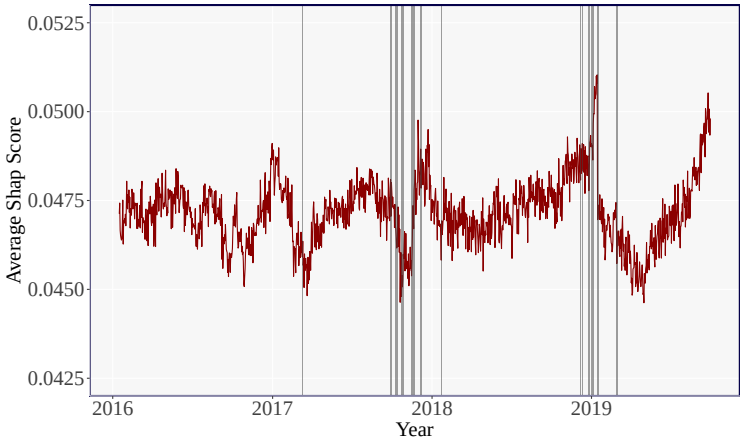
In order to try and address this problem, the focus has been placed on a single measuring station, Escuelas Aguirre which was one of the stations presented in the main text. The idea is to adjust the protocol variable, leaving all other variables fixed. That is, the first analysis does not alter the protocol variable, the XGBoost model sees the true protocol activation date. The second type of analysis adds a lead of seven days to the protocol variable, keeping all other variables fixed. The third type of analysis adds a lag to the protocol variable, keeping all other variables fixed. For example, by adding a seven day lag to the protocol variable we have removed the endogeneity problem since today's pollution is not causing the protocol's state seven days in the past, the same goes for adding a lead.

Essentially, we have a problem of simultaneous mutual dependence between a regressor and the dependent variable. The classical solution in Econometrics for that is the use of an instrumental variable, which must be correlated with past values of the regressor and in addition it must be simultaneously uncorrelated with the response variable. Very frequently, the lagged regressor itself meets trivially those two requirements. Regardless the choice of instrument, as with any other regressor, we need enough variability of the instrumental variable within our sample in order to identify its effect on the dependent variable. We have tried with different lags, within a reasonable range, of the protocol, and the lack of variability seems crucial. As mentioned in the paper, the protocol has been activated a very small number of days along the sample period (59 days in total). Accordingly, the switch between active and inactive occurs very few times in our sample, which means that taking a few days lag (or lead) is roughly taking the same observations. This is basically what the analysis that follows illustrates. Below each of the figures we present the analysis of

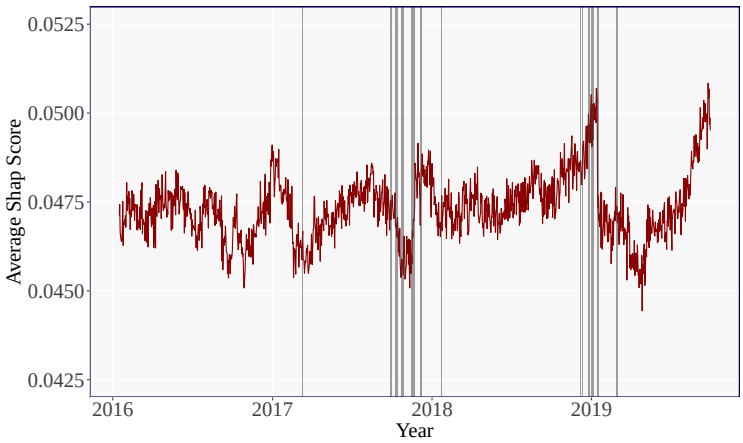
the different plots. All plots which report a lead / lag follow the same structure, such as, panel (a) will show the results when the analysis has no lead or lag. Panel (b) will show the results when the analysis has a lead of seven days and finally panel (c) will show the results when the analysis has a lag of seven days. All other variables remaining constant in time.



(a) No lag or lead

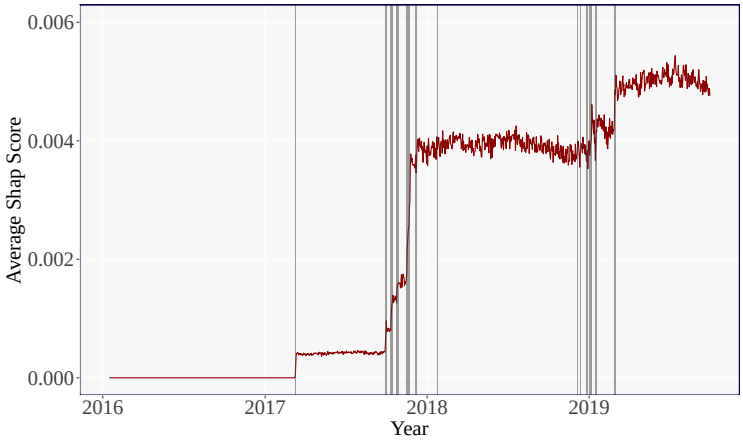


(b) Lead of seven days

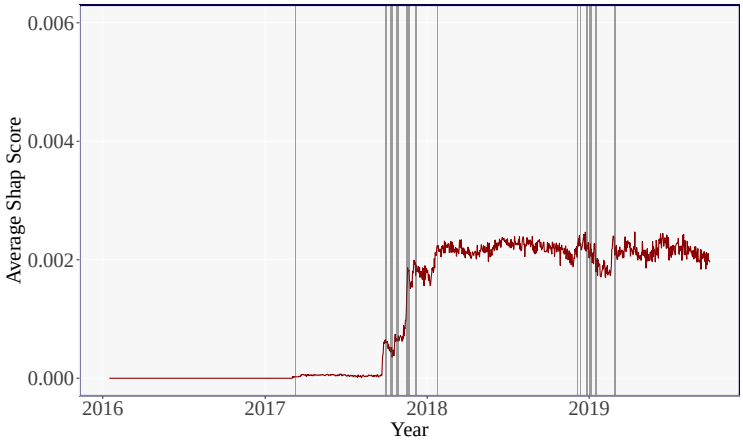


(c) Lag of seven days

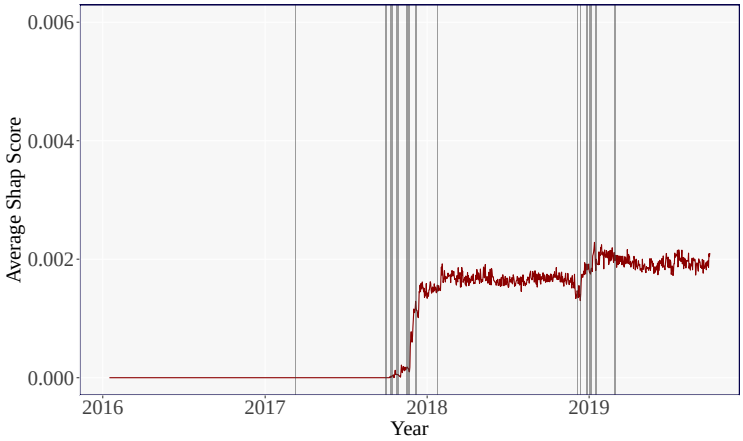
Figure 3.2.1: **Wind Shap scores.** The results do not change all that significantly upon visual inspection since the Shap scores for the *wind* variable do not shift seven days into the future or past. That is, we held the *wind* variable fixed (only adjusting the *protocol* variable) so, it is to be expected that the results will not change significantly for this variable.



(a) No lag or lead

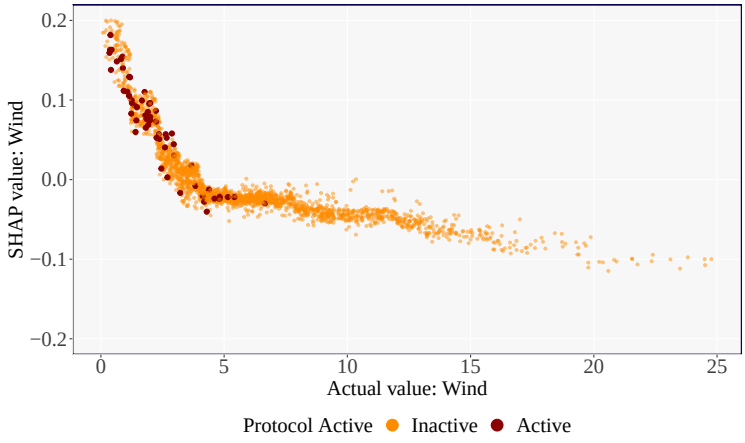


(b) Lead of seven days

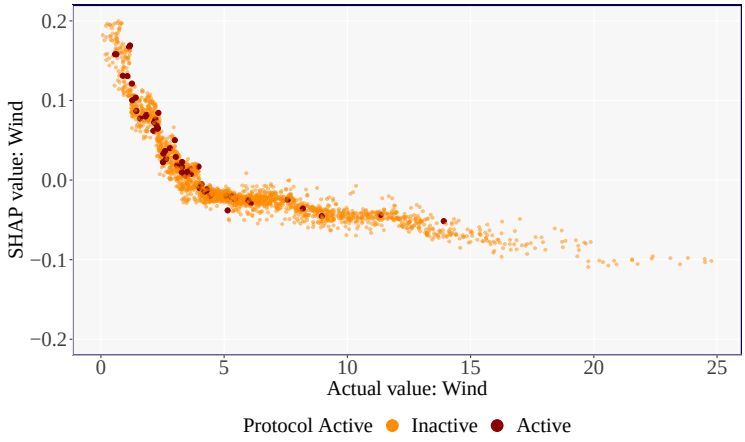


(c) Lag of seven days

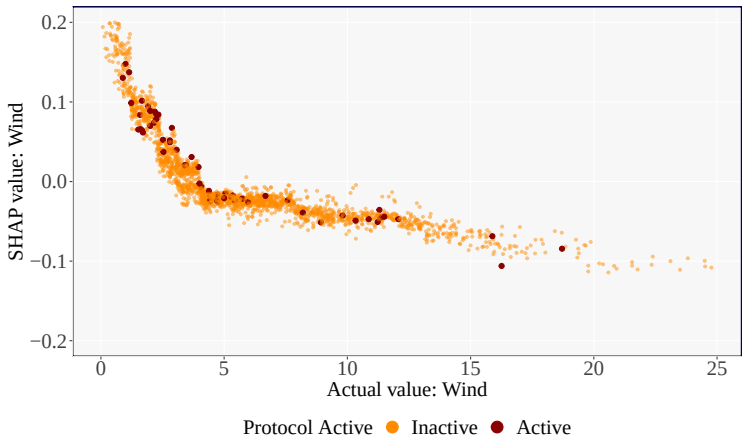
Figure 3.2.2: **Protocol Shap scores.** When there is no modification to the protocol variable (no lead or lag) the Shap scores are higher at the point of activation of the protocol, as opposed to when the protocol variable is shifted. Panel (b) and (c) show the Shap scores when the protocol is activated seven days before or seven days after the true event and the magnitude of the Shap impact is halved.



(a) No lag or lead

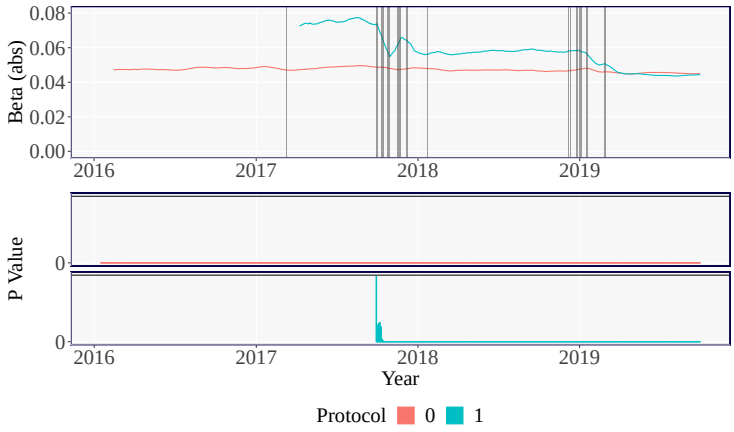


(b) Lead of seven days

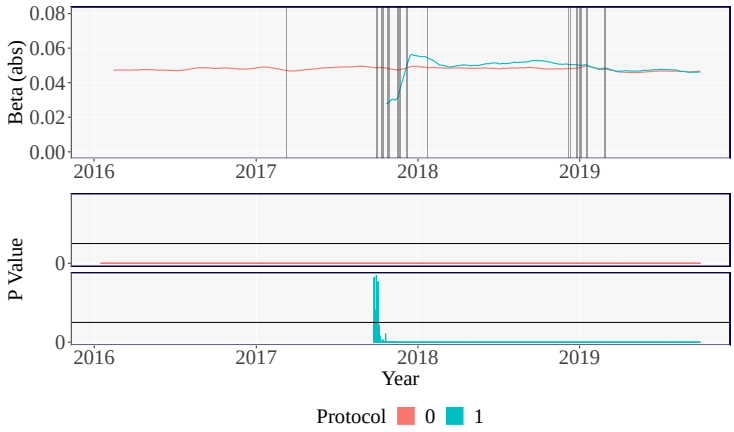


(c) Lag of seven days

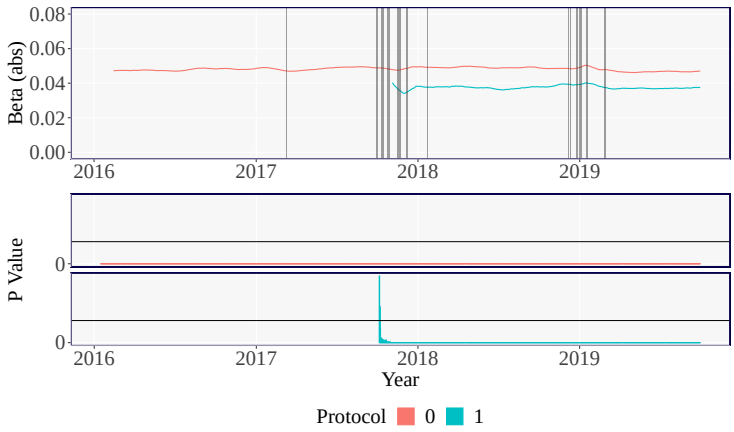
Figure 3.2.3: **Shap scores and wind speed.** Each point is a day in the data. Red points correspond to days that the protocol was activated. Panel (a) - no lead or lag shows us that the protocol is activated only on days in which wind speed is low. The second plot shows the protocol pushed forward seven days. There are now some days in which the model and corresponding Shapley computations are seeing the protocol being activated when the wind speed is high. When the wind speed is high enough, the pollution gets pushed out of the city and there is less pollution inside Madrid. However, the model should associate the protocol with higher levels of pollution when first activated. Panel (c) shows similar results, the model now sees the protocol being activated on days when the wind speed is sufficiently high also. These days/points are given negative Shap scores.



(a) No lag or lead

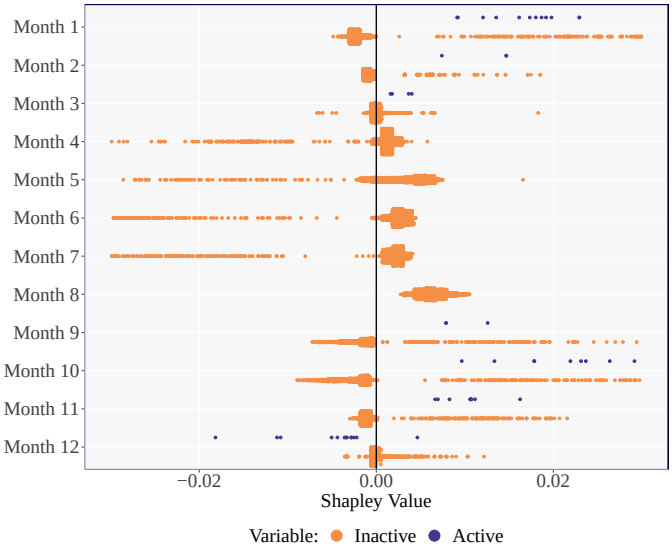


(b) Lead of seven days

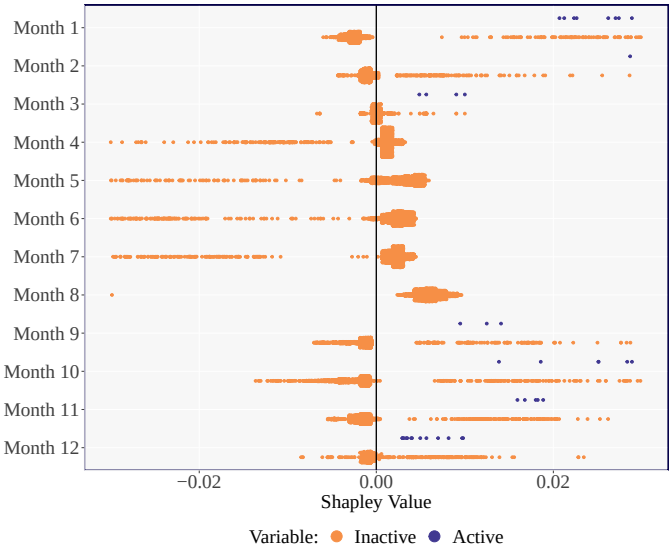


(c) Lag of seven days

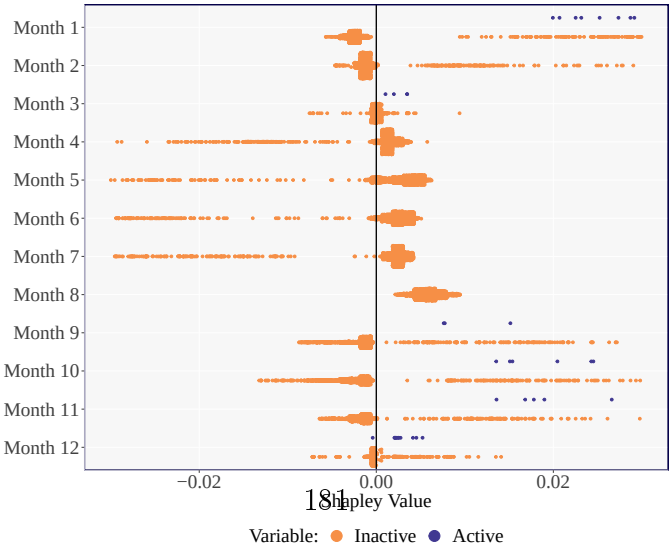
Figure 3.2.4: **Regression results from Figures 3.2.3a to 3.2.3c.** In order to compute the regressions, the data is filtered to just the points below the actual wind value of 5 in the previous figures, in order to be consistent with the original paper. Two regressions are computed for the active and inactive data points and the slopes are compared. Panel (a) shows the regression when there is no shift in the protocol variable. The slope is steeper when the protocol is activated (blue line) than when it is not (red line). Panel (b) doesn't show the same relationship and shows that the slopes are almost the same. Panel(c) shows the opposite relationship. The lower panel of each of the figures shows the rolling p-values for the active and inactive regressions. The p-values are statistically significant except in the first few days in which the protocol was first activated (the black line corresponds to the 5% significance level).



(a) No lag or lead



(b) Lead of seven days



(c) Lag of seven days

Figure 3.2.5: **Monthly Shap values.** For each month the Shap values are computed and the data is represented by the active protocol (blue) and the inactive protocol (orange). The warmer months take on negative Shap values with the exception of August (less people in Madrid to cause pollution) whereas the colder months take on positive Shap values. The plot also shows that the protocol is only ever activated in the colder months.

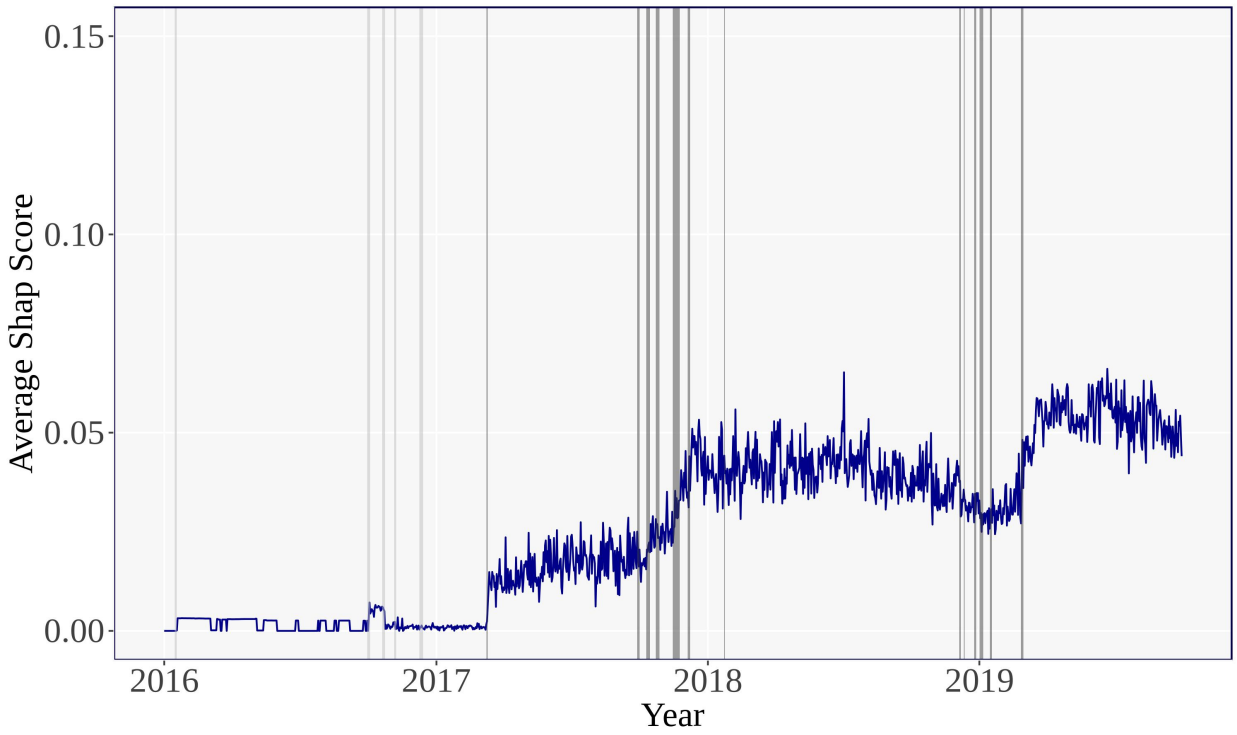


Figure 3.2.6: **Shap scores for the protocol variable with fake dates.** The introduction of fake protocol dates is fed into the model, lighter grey horizontal lines denotes the false dates and darker grey horizontal lines denotes the real dates. If the protocol was not contributing anything to the model then we would not expect to see any distinction between the fake protocol active dates and the real protocol active dates. The figure shows that Shap values reacts more strongly on the first day the protocol is introduced (date the protocol was first introduced: 2017-03-09 - 2017-03-11). It was introduced for just 3 days and at this point the model began to assign Shapley scores, assigning more as it sees more protocol active dates.

Chapter 4.1

Identifying mortality characteristics in COVID19 patients

Abstract: In this paper, we apply a series of Machine Learning models to a recently published unique dataset on the mortality of COVID19 patients. We use a dataset consisting of blood samples of 375 patients admitted to a hospital in the region of Wuhan, China. There are 201 patients who survived hospitalisation and 174 patients who died whilst in hospital. The focus of the paper is not only on seeing which Machine Learning model is able to obtain the absolute highest accuracy but more on the interpretation of what the Machine Learning models provide. We find that *age*, *days in hospital*, *Lymphocyte* and *Neutrophils* are important and robust predictors when predicting a patients mortality. Furthermore, the algorithms we use allow us to observe the marginal impact of each variable on a case-by-case patient level, which might help practitioners to easily detect anomalous patterns. This paper analyses the *global* and *local* interpretation of the Machine Learning models on patients with COVID19.

JEL Codes: H51 (Government Expenditures and Health), I18 (Public Health), C01 (Econometrics).

4.1.1 Introduction

The interest in COVID19 in the academic and data science community has been growing at an unprecedented rate since its outbreak, with new datasets being released on a continuous basis.[1] In this paper we use a unique dataset recently published in the supplementary material of Yan et al. (2020a). They applied a Machine Learning algorithm, Extreme Gradient Boosting (XGBoost) on blood samples from 485 infected COVID19 patients. From their sample, we downloaded patient blood sample features for 375 patients, 201 patients who survived and 174 who perished from COVID19 between January and February 2020. As far as we are aware this dataset is the only dataset publicly available that contains patient characteristics on who survived and who died from COVID19 and due to the sensitivity of

[1] https://www.kaggle.com/allen-institute-for-ai/CORD-19-research-challenge/

such patient-level information, such datasets are hard to come by.

In contrast to Yan et al. (2020a), we take a more *data science* approach. We compare other Machine Learning models to XGBoost. We also present a way to analyse individual patient-by-patient predictions quickly, which may be useful in high-stress environments in the case another pandemic outbreak occurs in the future. Additionally, this patient-by-patient analysis is potentially very relevant, as the marginal effect of a given feature might change from one patient to another depending on other feature values. Additionally, we aggregate the patient-by-patient analysis to deliver *feature importance* scores for the whole sample. For that, we use Shapley values, which is a concept recently taken from cooperative game theory and applied to Machine Learning. It measures the contribution of each feature value, abstracting away from the model specification. Finally, we apply *what-if* analysis from the Machine Learning model, which answers the question, how does the predicted probability of mortality change with a marginal increase (decrease) in the patient's characteristics, such as, age or number of days spent in the hospital when all other variables are held constant.

4.1.2 Literature review

There is an ever-increasing literature in relation to COVID19 not just from medical sciences but from all angles of the scientific community. We keep this literature review specific to Machine Learning applications to the COVID19 pandemic however some other sciences have also analysed the COVID19 situation. Fernandes (2020), Atkeson (2020) and Makridis and Hartley (2020) analysed the economic impact of COVID19, whereas Wang et al. (2020) analysed the psychological impact on children during the COVID19 lock-down.

To date clinical studies have found that the majority of COVID19 patients have suffered from lung infection and therefore many academics have sought X-ray imagery for early automatic detection systems. Apostolopoulos and Mpesiana (2020), Narin et al. (2020), Zhang et al. (2020), apply different Neural Networks on lung X-ray images in order to classify patients with and without COVID19. Wang and Wong (2020) apply deep convolutional networks on chest X-Ray images to detect patients with COVID19. They released their dataset as an open-source benchmark dataset which contains 13,975 chest X-Ray images. Majeed et al. (2020) apply 12 convolutional neural networks on X-Ray images. They use two COVID19 X-Ray image datasets along with a large image dataset of non-COVID19 viral infections, bacterial infections and normal X-Rays. Shi et al. (2020) offers a comprehensive literature review of Artificial Intelligence methods applied to imagery data in relation to COVID19.

Randhawa et al. (2020) applied a decision tree approach to analyse over 5000 unique viral genomic sequences including 29 COVID19 virus sequences. Arentz et al. (2020) discuss a number of patient characteristics of 21 critically ill patients with COVID19 in Washington State. The patients they analysed has a mean age of 70 years (min 43, max 92) with 52%

being male. The characteristics of these critically ill patients related to this study were a mean *absolute lymphocyte count* of $889/\mu L$, mean *platelet count* $10^3/\mu L$ of 215 and a mean *white blood cell count* of $515/\mu L$.

Wynants et al. (2020) apply a review and critical appraisal of 27 studies and 31 prediction models from the academic community. They found that the most important reported predictors for patients with COVID19 were *age*, *sex*, *tomography scan features*, *C reactive proteins*, *lactic dehydrogenase* and *lymphocyte count*. They state that all studies were at risk of high bias due to non-representative selection of control patients and high risk of model over-fitting. Salman et al. (2020) achieved a 100% sensitivity, 100% specificity, 100% accuracy, 100% Positive Prediction and 100% Negative Prediction when applying deep learning models on the detection of COVID19 from 260 X-Rays images.

Yan et al. (2020b) analysed patients with COVID19 and found that *fever* was the most common initial symptom, followed by a *cough*, *fatigue* and *shortness of breath*. They used over 300 variables and found that *lactic dehydrogenase*, *lymphocyte* and *high-sensitivity C-reactive protein* were key clinical features. Chen et al. (2020a) analysed the clinical characteristics of COVID19 in pregnancy, they found that out of 9 patients, 7 presented a *fever*, 4 a *cough*, 3 *muscle pain* and 2 a *sore throat*.

There is a fast-growing literature proposing Machine Learning models to predict COVID19 mortality. An illustrative -though ever-expanding- list of works are the following: An et al. (2020), Assaf et al. (2020), Bertsimas et al. (2020), Chowdhury et al. (2020), Di Castelnuovo et al. (2020), Ikemura et al. (2020), Laguna-Goya et al. (2020), Lalmuanawma et al. (2020), Malki et al. (2020), Metsky et al. (2020), Osi et al. (2020), Peng and Nagata (2020), Randhawa et al. (2020) and Singh et al. (2020). In our analysis and like many of the papers listed previously, we will compare different Machine Learning models in terms of their predictive capacity. In contrast to most of these papers, we go a step further in trying to understand the models predictions by observing figures for patient-level case studies. The use of Shapley values, which is absent in all of the previous papers, will be essential for that. Our motivation is purely practical: a practitioner, a non-expert in Machine Learning, who aims to understand the prediction that the *application* (Machine Learning) is generating for a given incoming patient at the triage room in a hospital.

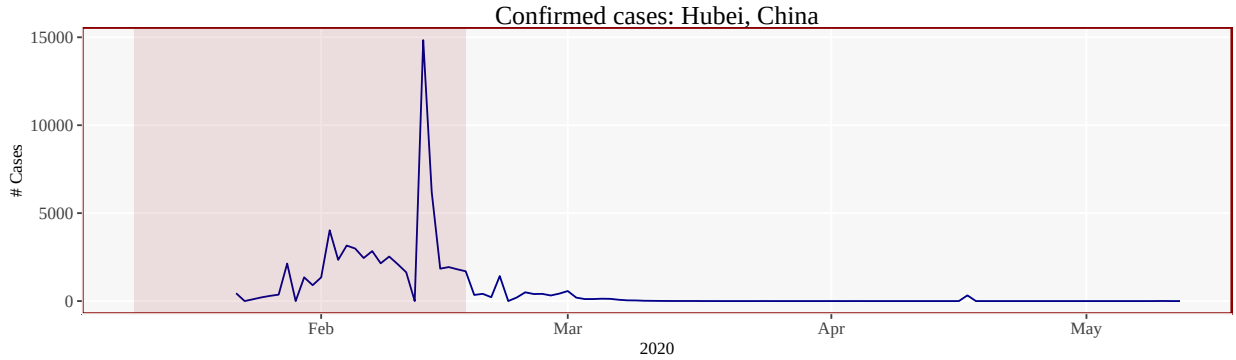


Figure 4.1.1: Confirmed cases for the region Hubei, China which contains the hospital in which the data was collected from. The darker region contains the region under analysis.

4.1.3 Data

The data used in this study can be found in the supplementary material from Yan et al. (2020a).[2] The original dataset was collected between the 10th January to the 18th February 2020, pregnant, breastfeeding women and patients under 18 years of age, along with patients with more than 80% incomplete data were omitted from their dataset. In total there were 375 patients in the dataset, 201 patients who survived and 174 patients who died from COVID19. Figure 4.1.1 reports the number of confirmed cases for the region Hubei, China. The shaded region indicates the time period for which we have the data which contains the most confirmed cases.

4.1.3.1 Summary statistics

The summary statistics, reported in Table 4.1.1, show that there are distinct differences between patients who survived and passed away from COVID19. On average older patients were most likely to pass away as a result of COVID19, additionally the longer you stayed in the hospital the higher the chances of survival. The blood sample data also show significant differences between the two classes. Whereas there seems to be a heavy skew of males who passed away from COVID19 in the dataset.

The original dataset contained a significant number of missing values. Panel (A) in Figure 4.1.11 in the Appendix reports the percentage of missing values for each patient, by patient outcome, whereas Panel (B) in Figure 4.1.11 reports the number of missing values for each variable, by patient outcome. For a number of patient cases, the number of missing values are high - above 60% whereas the number of cases by variable is also high $\approx 100\%$ for many variables. We therefore filter out these variables and use a cut-down version of the data. We set a cut-off percentage threshold of 50% - that is, all variables with more than

[2] https://www.nature.com/articles/s42256-020-0180-7

	Survived	Perished
	0	1
Age		
Mean	50.23	68.75
SD	15.02	11.83
Days in Hospital		
Mean	13.42	7.91
SD	6.72	7.36
Lymphocyte Count		
Mean	1.46	0.62
SD	3.99	0.35
Lymphocyte		
Mean	24.47	7.25
SD	11.15	5.43
Neutrophils Count		
Mean	3.61	10.10
SD	2.21	5.92
Neutrophils		
Mean	66.03	87.64
SD	13.64	8.05
Gender (Percent)		
Females	51.00	28.00
Males	49.00	72.00

Table 4.1.1: Summary statistics of patient characteristics

50% of *NA* values were removed, given by the vertical line in Figure 4.1.11.

Figure 4.1.12 plots an *alluvial* plot showing the distribution of patients by gender, mapped into the number of weeks that patient spent in hospital, then mapped into an age category, finally, mapped into that patients outcome. It is clear that a larger proportion of the gender 0 category who spent less than a week in the hospital and was over 60 years of age died of COVID19 related illnesses. The gender category 1 fared significantly better when following a similar path.

Figure 4.1.13 in the Appendix plots the characteristics of *age* and *age bins* on the *outcome* variable. Panel B shows the outcome by *age bins*. The *triangles* on the left side show the outcome of mortality whereas the right side shows the outcome of survival. The size of the *triangle* dictates the number of patients in that outcome. For instance, we can see that for *age bins* $(30, 40]$ there is a larger triangle on the *right* side than its corresponding colour on the *left* side (*which is 180 degrees opposite*). Therefore the patients in the *age bin* $(30, 40]$ had a high success rate of survival. Moreover, contrast that with the $(80, 90]$ *age*

bin and we see an opposite trend - a higher triangle on the *left* side of the plot than the *right* side of the plot, indicating more people in this *age bin* perished. Panel (A) shows the *violin* plots for the *age* variable by *gender* and *outcome*. We can see that there is a distinct bump in the kernel density plot for *males* around the ages of 30 for the patients who died which is not seen in the sample of the patients who survived.

4.1.4 Results

We next report the comparisons between different Machine Learning models and show the interpretability from the *classification tree* model. Moreover, we show four patient-level case studies along with variable importance plots demonstrating which variables the models found *most important*. Additionally, we report model interpretation from a subset of co-operative game theory, SHapley Additive exPlanations (SHAP) scores from one of the models. Finally we report *ceteris paribus* & *what-if* analysis of a patients survival probability. We discuss each of the above in more detail in each of the corresponding subsections.

4.1.4.1 Machine Learning Comparisons

Metric	Naive.Bayes	Logistic.Regression	Random.Forest	adaBoost	Classification.Tree	Light.GBM	XGBoost
Accuracy	0.84	0.91	0.90	0.83	0.81	0.88	0.93
Sensitivity	0.86	0.97	0.94	0.88	0.86	0.93	0.95
Specificity	0.83	0.84	0.84	0.79	0.77	0.85	0.90
Precision	0.80	0.88	0.87	0.77	0.75	0.83	0.89
F1	0.83	0.92	0.91	0.82	0.80	0.88	0.92
MCC	0.68	0.82	0.79	0.67	0.62	0.77	0.85
AUC	0.84	0.91	0.89	0.83	0.81	0.89	0.93
AUPRC	0.84	0.91	0.88	0.85	0.82	0.90	0.94
TP	36.00	35.00	34.00	37.00	36.00	39.00	40.00
FP	6.00	1.00	2.00	5.00	6.00	3.00	2.00
FN	9.00	5.00	5.00	11.00	12.00	8.00	5.00
TN	43.00	26.00	26.00	41.00	40.00	44.00	47.00

* Note: The Logistic Regression and Random Forest model removes missing values from its final results and cannot be adequately compared with the other results.
MCC: Matthew's Correlation Coefficient
† AUC: Area Under the Curve
AUPRC: Area Under the Precision Recall Curve
TP: True Positive | FP: False Positive | FN: False Negative | TN: True Negative

Table 4.1.2: Comparison of difference Machine Learning models

We split the sample of 375 observations up into a *training* and *testing* dataset, in which 75% corresponds to the training data and 25% corresponds to the testing data. The above table reports the confusion matrix statistics for a number of Machine Learning models such as *Naive Bayes*, *Logistic Regression*, *Random Forest*, *adaBoost*, *Classification Tree*, *Light-*

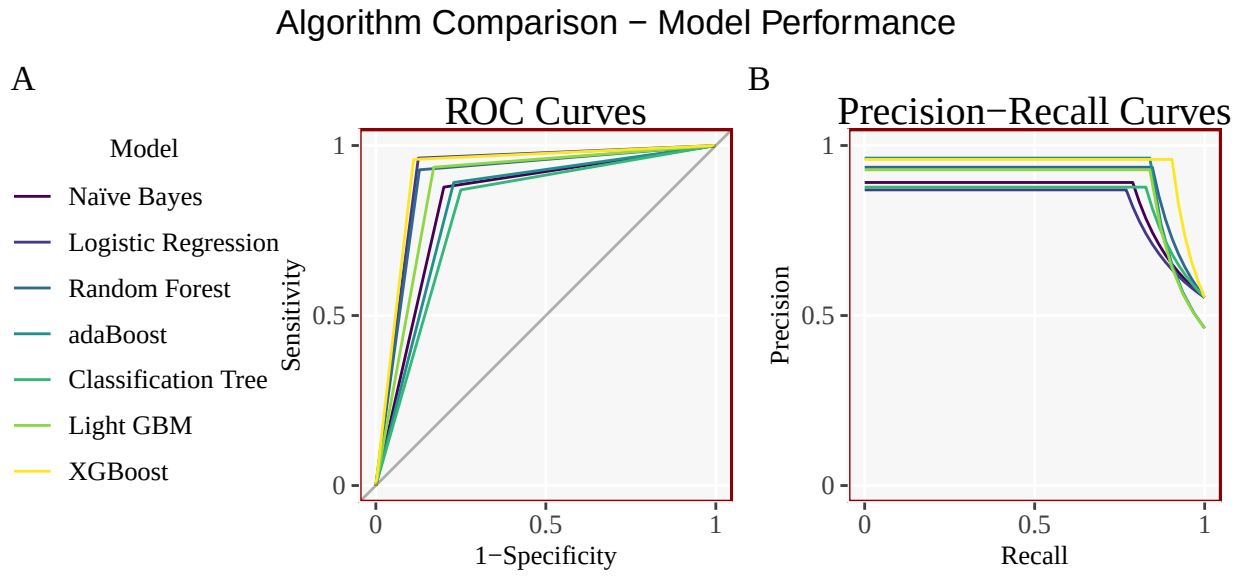


Figure 4.1.2: Characteristic (ROC) and Precision-Recall curves

GBM and *XGBoost*.[3] Each of the models show very similar performance metrics, with the ensemble learning models performing slightly better over the more simpler models.

4.1.4.2 Classification Tree

Figure 4.1.3 plots an example of a decision tree from the *Classification Tree* model. Roughly, a decision tree, or simply a tree, represents a piece-wise mapping from a set of features, such as *Neutrophils* or *age*, into a response variable, which in our application is *probability of mortality*. Machine Learning algorithms, such as XGBoost, select the tree (or collection of trees) that minimizes some loss function.[4] Naturally, to select a tree conveys to select both the order of the features as we move down the tree and the threshold values at each split.

In the figure 4.1.3, as we go downward, the first split at the first node, is made on *Neutrophils* which shows the predicted probabilities of being in each class along with the percentage of the observations in this split. We can see that patients who have *Neutrophils* levels $x < 79$ and *age* $x < 63$ fall into *node*4 which contains 44% of the total observations and has predicted probabilities of 0.93 of survival and 0.07 of mortality. Therefore, patients

[3]Note: We omit *Neural Network*, *SVM* and *K-nn* models since there is a substantial amount of missing values in the data and an insufficient number of data points to adequately impute the missing values.

[4]For instance, XGBoost uses a loss function that weights prediction errors and *complexity* of the tree. For more details, see Chen and Guestrin (2016).

who fall into this terminal node are predicted to survive. Contrast that with a more complex non-linear node at *node*21 where patients have the following characteristics *Neutrophils* of $x < 79$, *age* of ≥ 63, *Eosinophils* of $x < 0.1$ and *Days in hospital* of $x < 7$ fall into *node*21 which has a predicted probability of 0.17 of survival and 0.83 of mortality, 9% of the sample fell into this node. To finalise, people who followed a similar path down the decision tree but stayed in the hospital for more than 7 days fell into *node*20 where they had a predicted probability of survival of 0.67 and 0.33 probability of mortality, 10% of the sample fell into this terminal node and thus the model found that the length of time spent in hospital has a significant impact on the probability of survival.

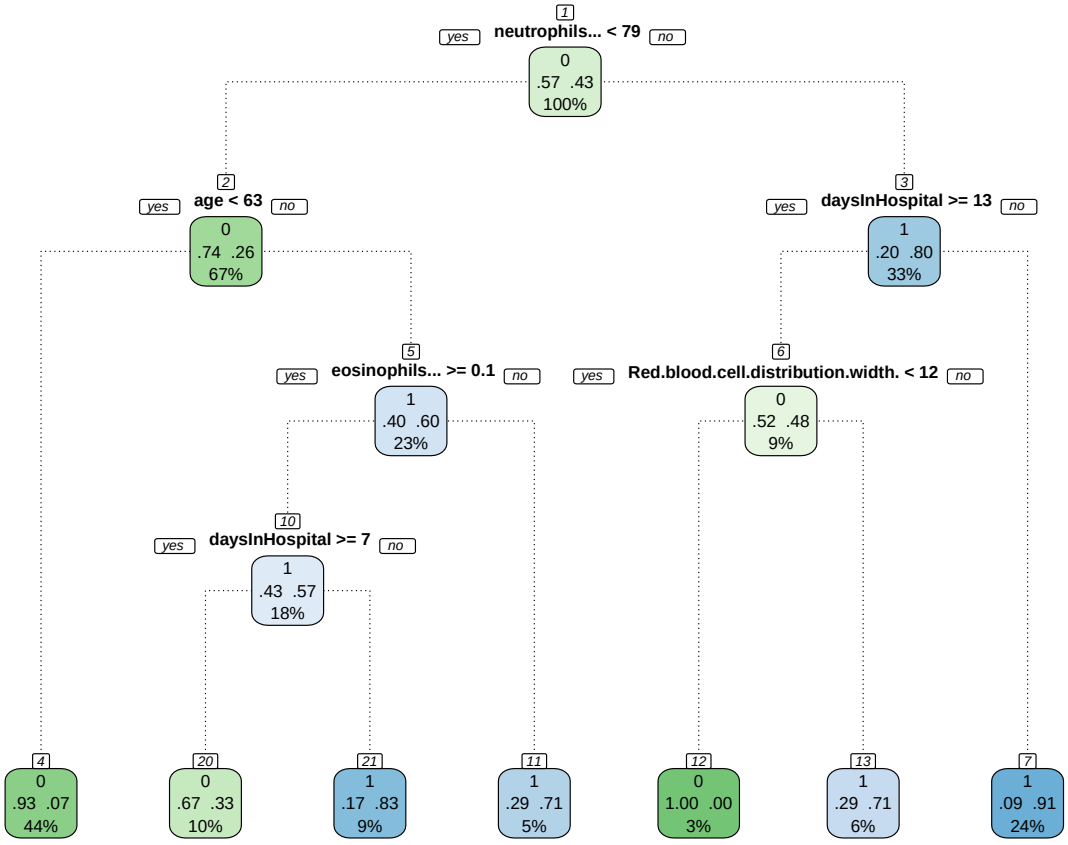


Figure 4.1.3: Decision Tree from the Classification Tree Model

4.1.4.3 Case Studies (Local Level)

A single decision tree as depicted in Figure 4.1.3 is highly interpretable but not very good at prediction as is evidenced by the worst performing model in the column *Classification Tree*. In order to overcome this issue of performance, an *ensemble* of decision trees can be used to make a prediction. The combination of decision trees improves greatly the prediction, though interpretability becomes *a priori* more complex. In this section, we show how more advanced decision tree models can be interpreted through case studies.

What sets the XGBoost model (along with other tree models) apart, from traditional black-box Machine Learning models is that it is possible to see how each variable contributes to the overall prediction for each observation or patient in the model. There are four possible cases, each representing a different position in the confusion matrix - or each representing one of the statistics of a **True Positive (TP)**, **False Positive (FP)**, **True Negative (TN)** and **False Negative (FP)**. We briefly discuss the results for two of the cases, leaving the other two in Figure 4.1.14 in the Appendix.

True Positive (TP). Panel (A) in Figure 4.1.4 shows the breakdown of how a positive case (deceased) was correctly predicted. Given a particular variable, shown in the x-axis, a log-odds score is calculated *(displayed inside each box)*, the sum of the log-odds scores are summed up in a cumulative manner and a final log-odds score is given *(displayed in the final black box)* and then a logistic function is applied to the final log-odds result in order to obtain a predicted probability *(shown on the y-axis)*. The horizontal line demonstrates a $y* = 0.5$ probability cut-off threshold. Patients above this line are classified as deceased and patients below this line are classified as survived. Notice, that the final log-odds prediction score is 1.19, which is assigned a predicted probability of mortality $(1 + exp(-1.19))^{-1} = 0.77$. **False Negative (FN).** Panel (B) in Figure 4.1.4 shows a patient who was incorrectly predicted to have survived. The model incorrectly predicted that the patient would have survived with a final log-odds score of -1.26 and a subsequent survival probability of $(1+exp(--1.26))^{-1} = 0.22$, sitting below the cut-off threshold $y* = 0.5$.

4.1.4.4 Feature Importance (Global Level)

From the case studies presented previously in Figure 4.1.4 we can see that certain patient characteristics are often given the largest (in absolute) values log-odds scores regardless of whether the patient survived or died. Such features include, *age*, *daysInHospital*, *Lymphocyte* and *Neutrophils*. That is, the variables presented in the summary statistics table previously.

Panel (A) and Panel (B) in Figure 4.1.5 reports the variable importance scores from both the *XGBoost* and *LightGBM* model. We can see that the most important variables are consistent across both models, with *age*, *daysInHospital*, *Lymphocyte* and *Neutrophils*

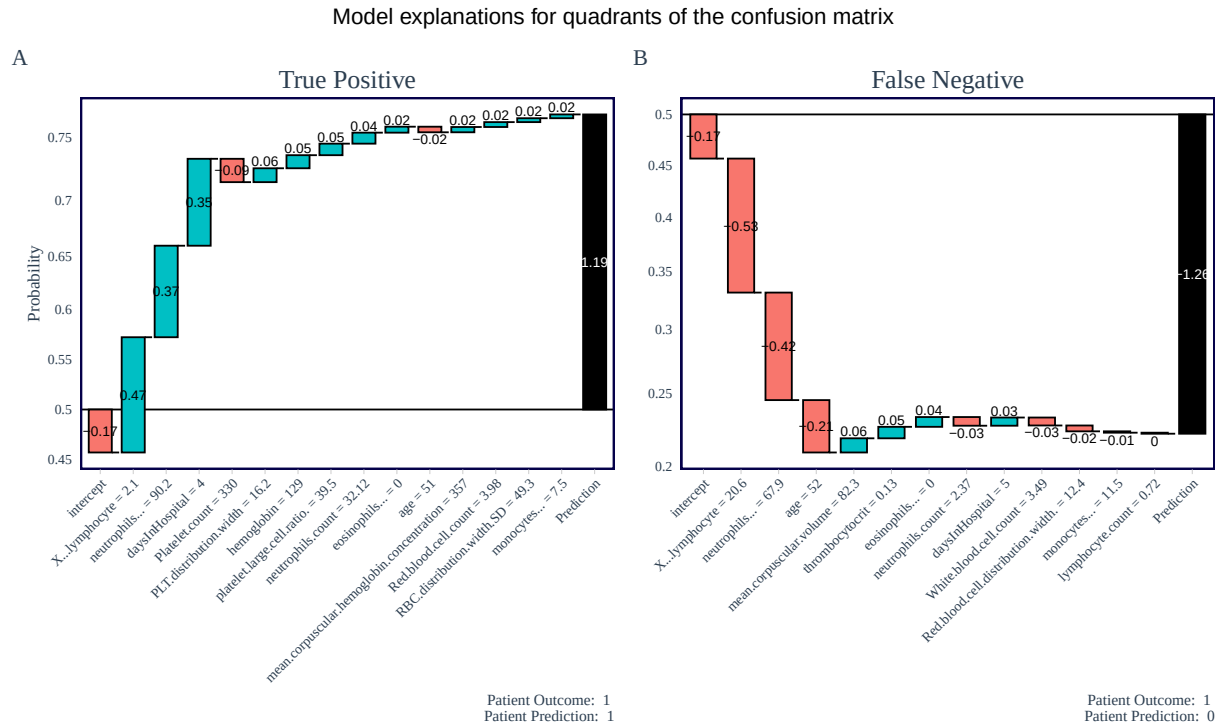


Figure 4.1.4: Two Case Studies, a True Positive (TP) and False Negative (FN). Figures inside each bar represent log-odds scores with the final black bar being a summation of all preceding bars scores. A logistic function is applied to the final log-odds result and a prediction probability is obtained (shown on the y-axis). The horizontal line at point 0.5 represents the y* cut-off threshold. The figures on the x-axis correspond to the values of the variables.

being ranked in the top four in both.

From Figure 4.1.4 we can see that different individual patient characteristics are associated with different (positive & negative) prediction scores. From Figure 4.1.5 we can also see that certain variables contribute more to the model than other variables. Moreover, Figure 4.1.5 does not tell us whether, for example, different *ages* contribute *more* or *less* to the probability of mortality, just that *age* is *important* at a global level. In order to overcome this issue we turn to a subset of *coalition game theory* and analyse *Shapley values*.

4.1.4.5 Cooperative Game Theory (SHapley Additive exPlanations)

Shapley values, which is a classical concept in cooperative game theory, see Shapley (1953) has been recently applied to understanding a Machine Learning models predictions, see Lundberg and Lee (2017) and Lundberg et al. (2018). Shapley values offer a *global interpre-*

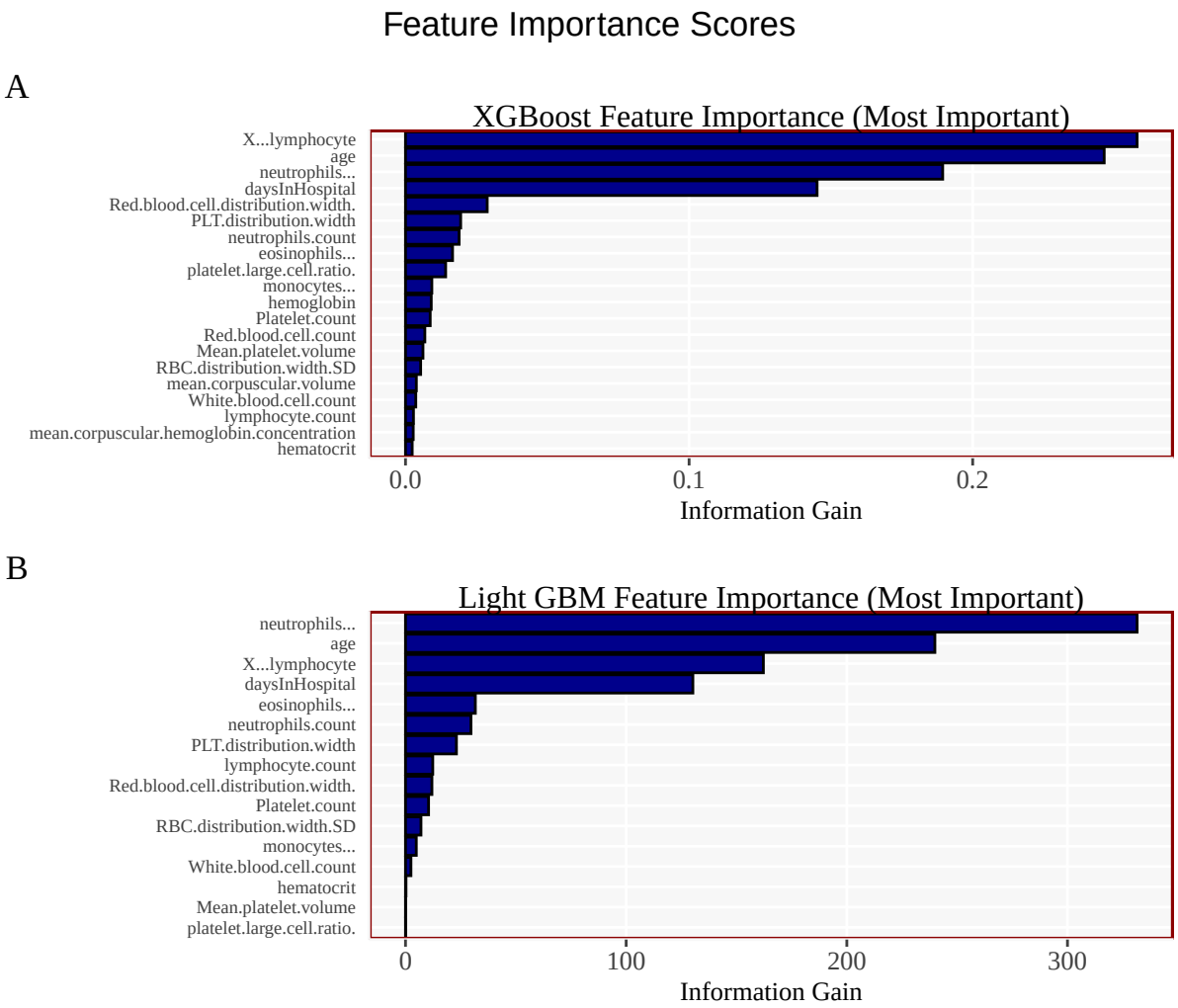


Figure 4.1.5: Feature Importance Scores for XGBoost and Light GBM

tation where we can measure how patient characteristics contribute - *positively* or *negatively* to the prediction of mortality. A similar measure is shown previously in Figure 4.1.5, however, unlike the feature importance plot shown there we are now able to see the *positive* or *negative* relationship between each variable and patient mortality prediction.

That is, given Figure 4.1.6 we can see that *age* has the greatest variability in *Shapley* values. Low values of *age* correspond to younger patients and more importantly, are assigned negative *Shapley* values and thus it tends to reduce the prediction of mortality. Contrast that with high values of *age* which corresponds to older patients and more importantly are assigned positive *Shapley* values and thus it has a higher marginal impact on the prediction probability of mortality. Conversely, the variable *daysInHospital* has the opposite impact. The higher the number of days the patient remains in hospital is associated with a negative

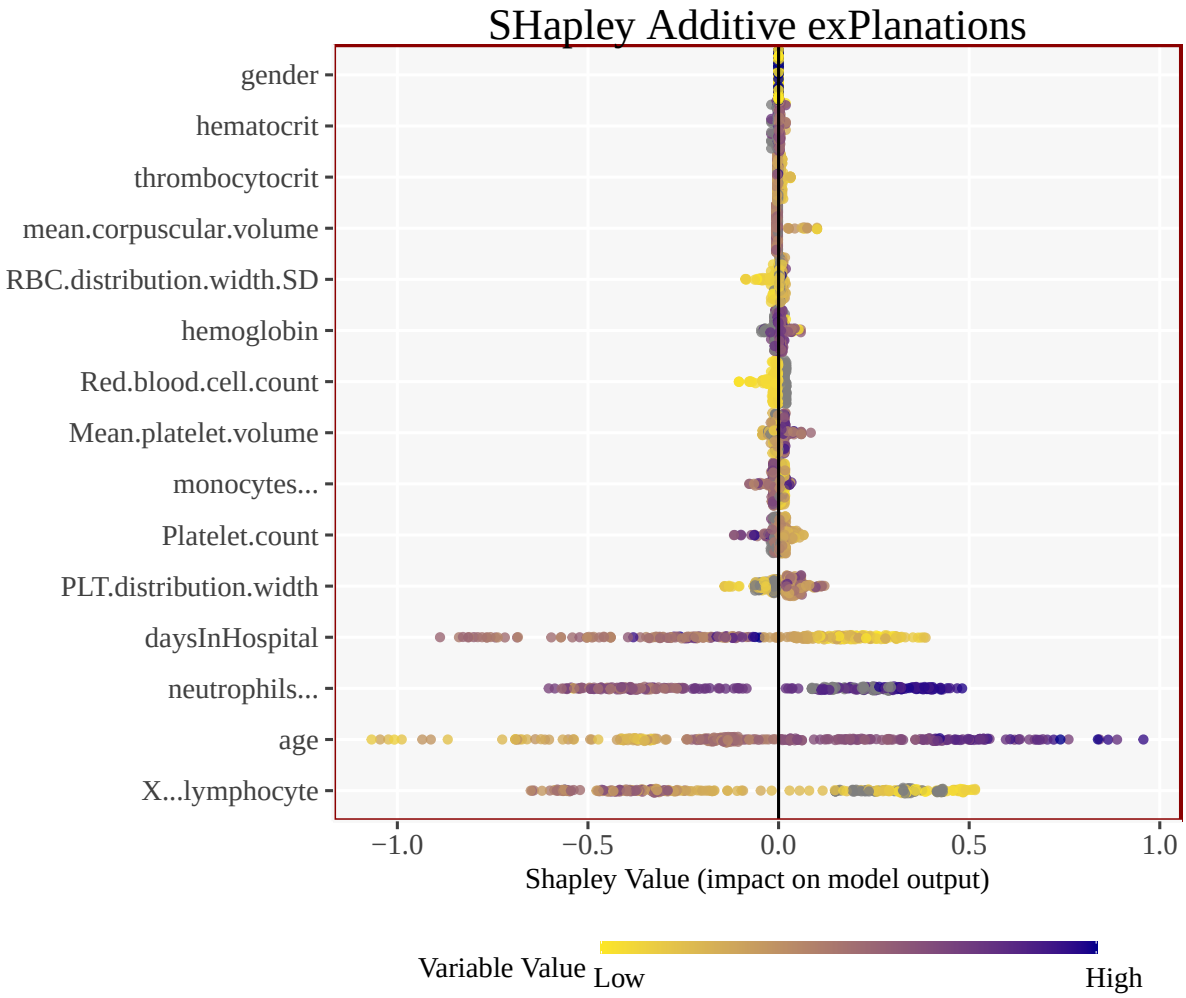


Figure 4.1.6: SHapley Additive exPlanations

marginal impact on the prediction of mortality whereas, the lower the number of days the patient remained in hospital is associated with a positive marginal impact on the prediction of mortality. Other variables follow similar and very distinct patterns. Figure 4.1.17 in the Appendix plots the mean Shapley values for each variable for the highest average Shap scores, which is somewhat similar to Figure 4.1.5. We note that the top four variables are consistent across models and across evaluation criteria.

Shapley Values also give a *local interpretation* and each patient obtains a total Shapley value (a summation of each of the variables Shapley value). This allows us to explain why a patient receives its prediction and the corresponding contribution of each feature. Figure 4.1.15 in the Appendix shows the breakdown of the four most important variables for all patients in the dataset, ranked by each patient's total Shapley value (lowest to highest by each outcome). Figure 4.1.7 shows four randomly sampled case studies, two from the *deceased*

side and two from the *survived* side of Figure 4.1.15 (where the background is coloured by red = deceased & blue = survived) along with that patients feature characteristic for the four most important variables in the model *age*, *daysInHospital*, *Lymphocyte* and *Neutrophils*.[5] That is, we get to see the patients characteristics along with the corresponding Shapley value assigned to that feature. Note, that these plots differ significantly from those presented in Figure 4.1.4 since the Shapley value plots are derived from the *training* data whereas the XGBoost case studies are obtained from the *test* data. Moreover, the Shapley value case studies can be thought of as *why the model learned a mapping of features to a prediction* whereas the XGBoost case studies can be thought of as *why the model made a mapping of features to a prediction*. Figure 4.1.15 is essentially the patient observations presented in Figure 4.1.7 but stacked more compactly side-by-side (and without the patients feature attribution characteristic.

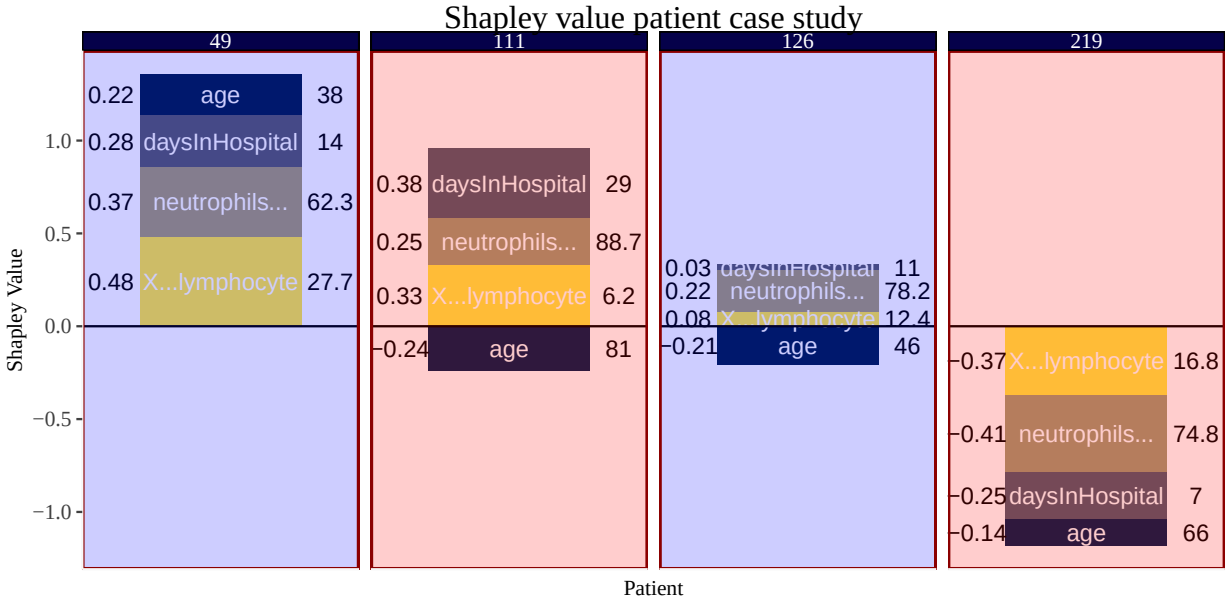


Figure 4.1.7: Shapley values for a sample of four observations of patient characteristics for the top four variables in the model. The background colour indicates the mortality rate - red = deceased and blue = survived. The numbers in the title of the plot correspond to the patient ID in the dataset. The text on the left-hand-side of the bars contain the Shaply value whereas the text on the right-hand-side of the bars contains the feature characteristic for that patient. The y-axis contains the summation of the features Shapley values for that patient.

We next study the non-linear interaction effects of different variables on the positive and

[5]Note that Figure 4.1.15 in the Appendix is a compressed and stacked version of Figure 4.1.7 and therefore we are able to obtain similar figures to that of Figure 4.1.7 for all patients along with their corresponding patient characteristics and Shapley values.

negative Shapley values. Panel (A) in Figure 4.1.8 shows the interaction effects of patient *age* and its corresponding feature Shapley value. Each point represents a patient, colour-coded by that patients *Lymphocyte* value, older patients have lower *Lymphocyte* values and are mostly placed in the upper right-hand-side of the plot in which they were given positive Shapley values. Recall, positive values to the prediction of mortality. Younger patients tended to have higher *Lymphocyte* values and subsequently obtain negative Shapley values. Panel (B) in Figure 4.1.8 shows the interaction between the number of days a patient spent in hospital and that patient's corresponding Shapley value, colour-coded by each patient's outcome. A far higher proportion of deceased patients occur on the left-hand-side of the horizontal line (< 10.5 days in hospital) when compared with the survived patients. These patients are given positive Shapley values.

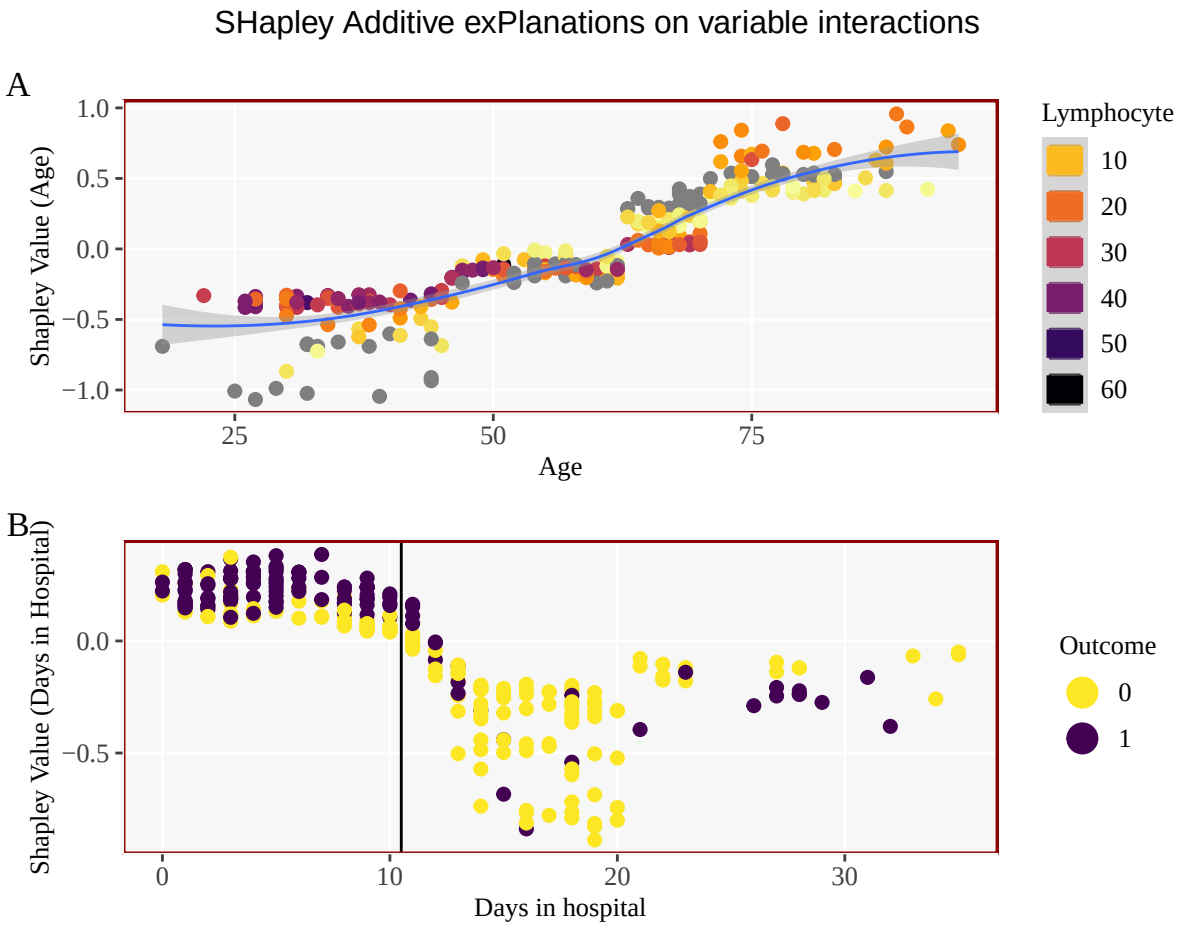


Figure 4.1.8: Non-linear variable interaction with Shapley values

4.1.4.6 Ceteris paribus

Finally, Figure 4.1.9 plots the models *what-if* analysis for a single patient. We can see that when holding all other variables fixed how the model's prediction probability changes with changes in the *x-axis* or changes in the patient feature characteristic. That is, given that this patient had an age of 66, when holding all other variables fixed an increase in that person's age increases the predicted probability of mortality. Moreover, the patient also spent 7 days in the hospital and thus if the patient spent more than 10 days in hospital the *what-if* analysis suggests that the patient would have a marginally lower predicted probability of mortality - holding all other variables constant. Similar analysis can be carried out for all patients and for all variables.

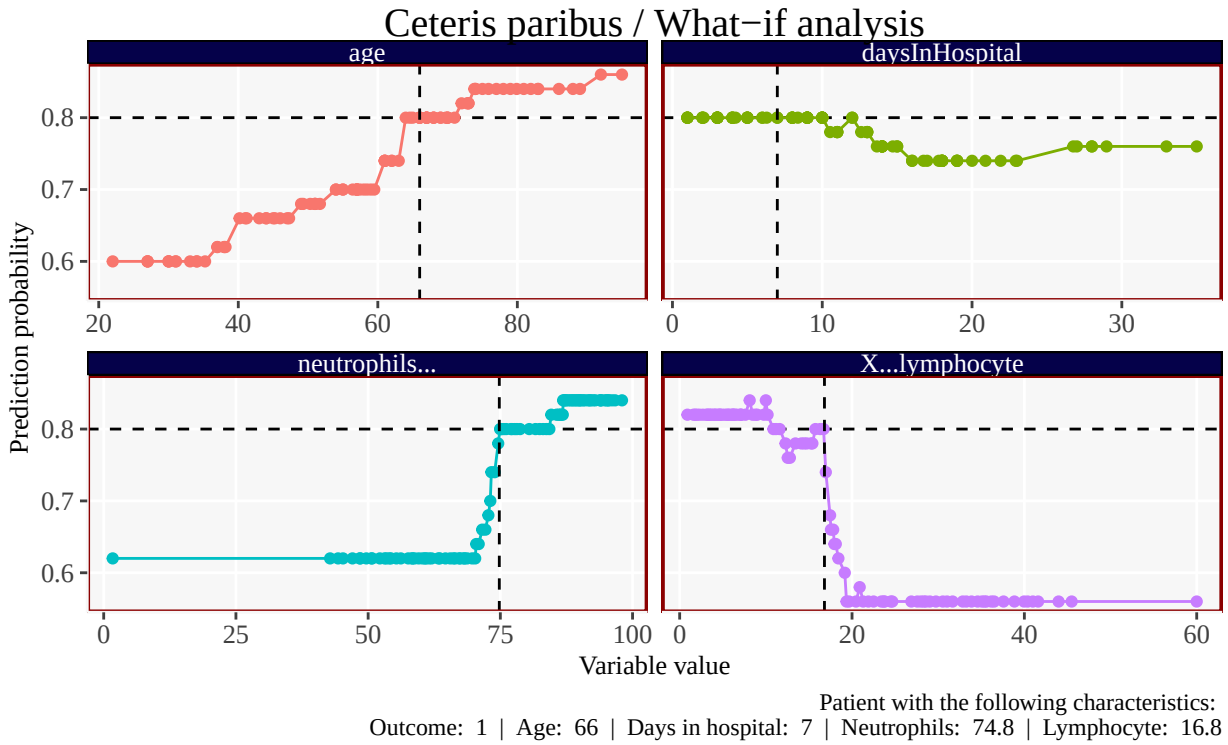


Figure 4.1.9: What-if analysis for the four most important variables in the model. The vertical dotted black line corresponds to the patients true characteristic (shown on the x-axis) and the horizontal dotted line corresponds to the models predicted probability of mortality. The intersection of the two lines shows where the patients true value / predicted probability lies. The points show how the predicted probability changes with changes in the x-axis (patient characteristic) holding all other variables fixed.

4.1.5 The Issue of Endogeneity

The dataset under analysis in this chapter contains cross-sectional information. For each patient, we have a *snapshot* of their health condition at the time of admittance to the hospital, along with how many days the patient spent in hospital and the outcome of the patients condition, survived or not. It is quite natural to think that the patients initial condition determines both, the number of days they stayed in hospital and their survival status. Thus, among these two later features, to establish which one is the dependent and which one is the explanatory variable is rather arbitrary. In other words, it is arbitrary to say which one is endogenous and which is exogenous.

In this section, we take a different approach, by restricting ourselves to the relationship in which the causality direction is obvious. Clearly, the initial health condition determines how many days the patient is in hospital, but not reversely. Thus we use a Machine Learning algorithm to estimate that relationship, without using the survival status at all. The outcome is a model that predicts the number of days in hospital from the conditions at the patients arrival time. We proceed analogously to study the causal relationship from initial health conditions to the survival status, disregarding information on how many days the patient spent in hospital. The outcome from this second model predicts a probability of mortality from the conditions at the point of arrival. To summarise, two different models predict respectively two different response variables for any given patient from a set of common explanatory variables.

This analysis allows us to emphasize the relationship between both responses. Figure 4.1.10 shows a scatter-plot, each point is a patient with their corresponding prediction for mortality in the horizontal axis and for days of hospital in the vertical axis. The analysis was restricted only to survivors, so we can interpret the predictions as conditional on surviving. The reason to take only survivors is that most of the deceased people were in hospital for a very short period. In a sense, we interpret them as people who left going to hospital *too late* and when they eventually arrived at hospital they only had a few days at most left of their life. Recall, that this dataset was collected during the first wave in China where hospitals were less equipped for an influx of patients.

Essentially, the figure shows a slightly negative slope of the linear fit to the points: those who are predicted to stay longer in hospital are also predicted to have a lower probability of mortality.

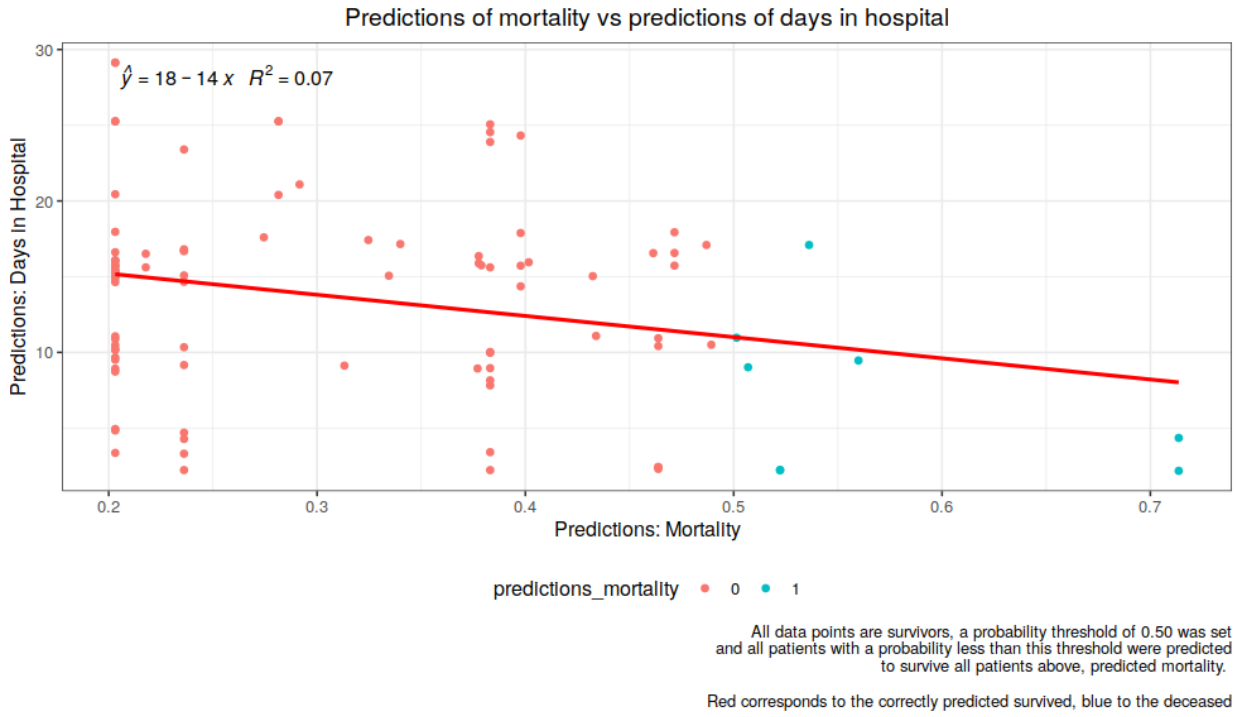


Figure 4.1.10

4.1.6 Conclusion

This paper analyses a number of patient characteristics by applying a series of Machine Learning models in order to predict the mortality of patients admitted to the hospital with COVID19. There were 375 patients in the dataset with 201 patients who survived and 174 patients who died from COVID19. Ensemble tree-based models obtained the highest prediction scores over more simplistic - yet easier to understand - classical models.

We focus our analysis on the interpretability of Machine Learning models. Firstly, by introducing patient case studies for each quadrant in the confusion matrix which helps understand why a model made a correct prediction or not. We also show that there is consistency in both across models and across evaluation criteria on what the four most *important* variables are. Moreover, we find that the variables *age*, *daysInHospital*, *Lymphocyte* and *Neutrophils* are the most important variables when making a prediction. We discuss how variations in patient characteristics have a positive and negative effect on the model's prediction through the use of SHapley Additive exPlanations (Shapley values) from cooperative game theory. Moreover, we use patient-level Shapley values to understand how the model assigns Shapley scores to each patient based on each patient's characteristics for four case studies. We also study the interaction between patient characteristics and their corresponding Shapley values. Finally, we briefly discuss *ceteris paribus* analysis in order to

understand how the models predictions change with *what-if* scenarios.

Tree-based models could be useful in analysing patients during peak epidemic outbreaks when hospitals may be overloaded and quick analysis is in order, especially given the non-linear nature of patient characteristics when admitted to hospitals.

The robustness of our findings are bound by the *diversity* of our dataset. We take data from Yan et al. (2020a), which leverage's a database of blood samples. It would be interesting to apply the Machine Learning algorithms used in this paper to a wider population of patients. Another relevant dimension worth exploring is to enlarge the range of potentially relevant features, this study primarily focused on blood cell data but, including other features such as aspartate aminotransferase (AST) and alanine aminotransferase (ALT) could potentially raise more interesting analysis of patient characterises and morbidity from COVID19. To summarize, our paper shows a promising direction on how relatively standard classification trees in Machine Learning combined with Shapley values help to identify mortality factors for COVID19, however, more robust conclusions require richer datasets.

From a more operational angle, a growing branch of the literature proposes the use of a number of Machine Learning models, say, at a triage phase in hospitals. In this regard, our differential factor, as mentioned, is to propose patient case studies and patient-level Shapley values, that can be easily interpreted -learnt- by practitioners in the field, even those who are not so familiar with the terminology used in Machine Learning, which facilitates the real implementation.

4.1.7 Appendix

4.1.7.1 Data Characteristics

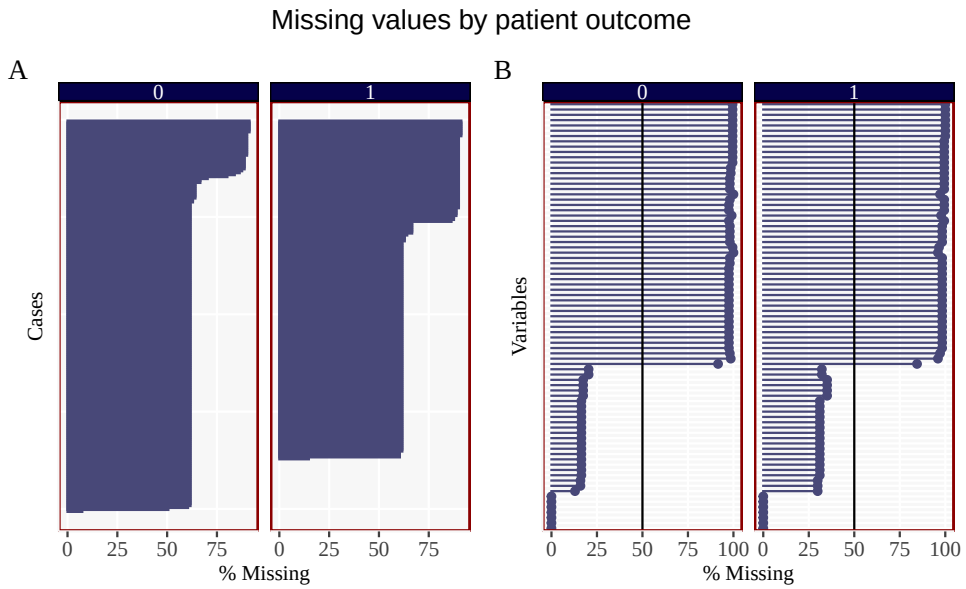


Figure 4.1.11: Panel (A) reports the number of missing values for each patient, by patient outcome. Ordered by highest number of missing values. The number of missing values by case seems to be slightly higher for the patients who perished as opposed to the patients who survived. Panel (B) reports the number of missing values for each variable, by patient outcome. Variable names have been removed to save on space. We remove all variables in the model whose percentage of missing values exceed 50% (as shown by the vertical line).

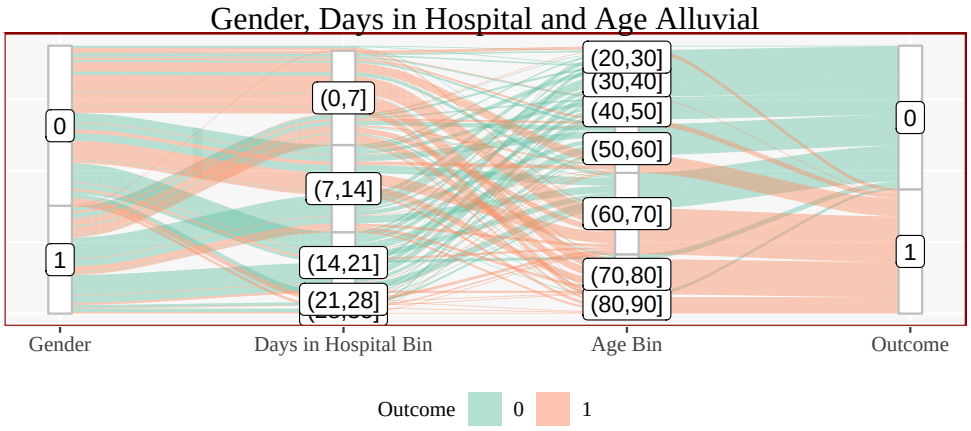


Figure 4.1.12: Alluvial plot for Gender, Days in Hospital Bins and Age Bins, coloured by mortality. A patient has gender 0 (male) may pass through to the days in hospital, bin (0, 7] (less than a week in hospital) and also be in the age category (60, 70]. These patients would be at high risk of mortality indicated by the red colour flowing through the plot. Additionally the age bind (70, 80] and (80, 90] also have a high risk of mortality for these patients. The size of the bars indicate the number of observations in each section, i.e. there appears to be slightly more males than females in the dataset.

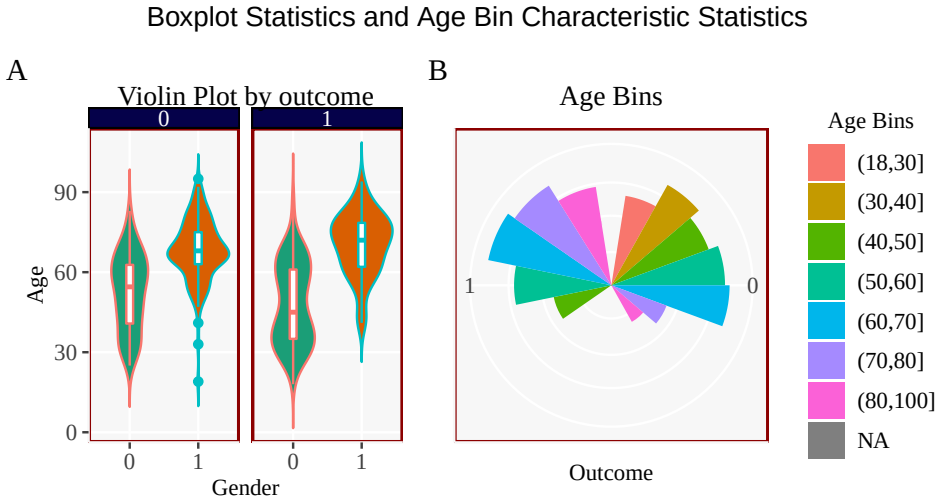


Figure 4.1.13: Panel (A) plots the Violin plot showing the distribution of patients ages and gender by survival. Pane (B) plots the distribution of patients age bins by the patients outcome.

4.1.7.2 XGBoost Case Studies

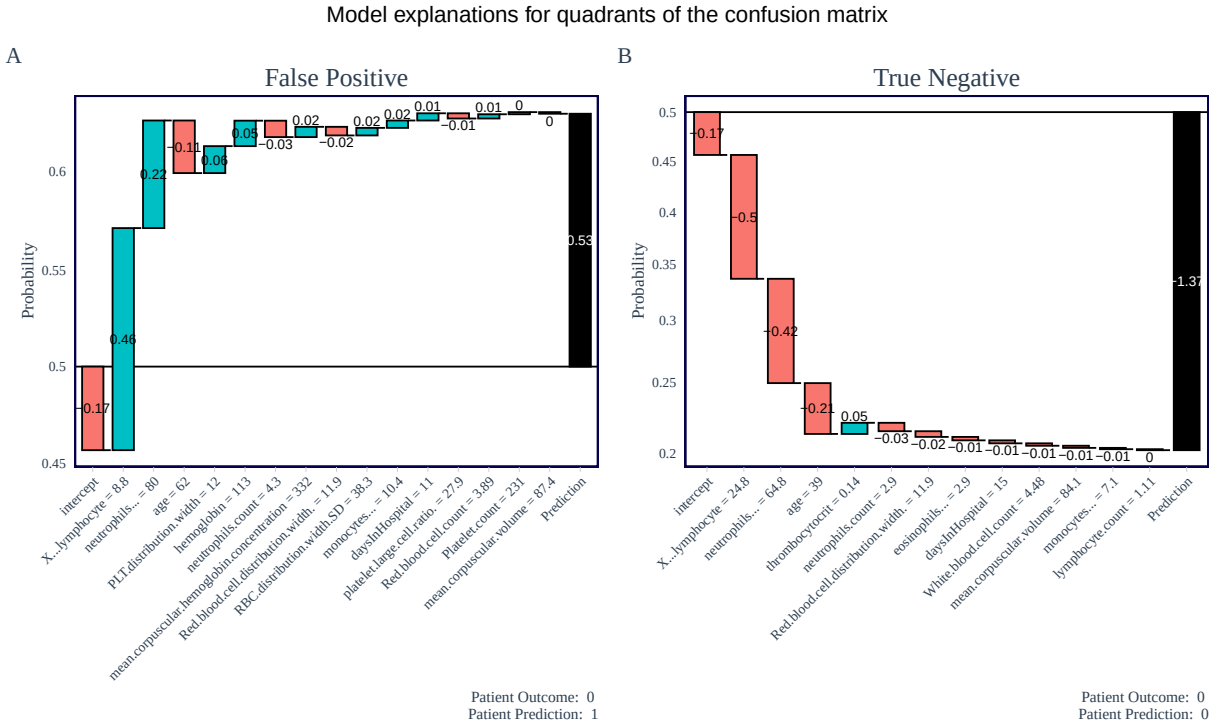


Figure 4.1.14: **False Positive (FP).** Panel (A) shows a patient that was incorrectly predicted to be deceased. The model incorrectly predicted that the patient would be deceased with a final log-odds score of 0.53 and a subsequent deceased probability of $(1 + exp(-0.53))^{-1} = 0.63$, sitting just above the cut-off threshold $y* = 0.5$. **True Negative (TN).** Panel (B) shows a patient who was correctly predicted to have survived with a final log-odds score of -1.37 and a subsequent probability of$(1 + exp(- - 1.37))^{-1} = 0.2$.

4.1.7.3 Shapley Values

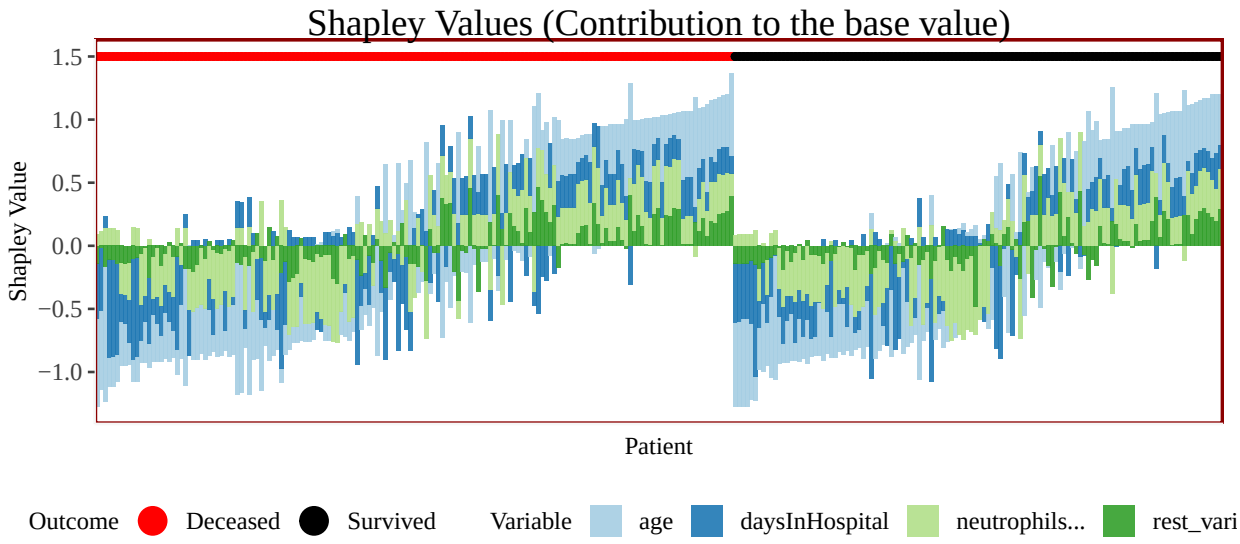


Figure 4.1.15: Patient level Shapley values: Each stacked bar represents a patient and that patients total Shapley score. The accompanying colours represent the individual variable Shapley scores for the four most important variables in the model (with the result of the other variables being summed up into the category rest variables). The patients are split according to whether the patient was deceased or survived and each group is ordered by that patients total Shapley value.

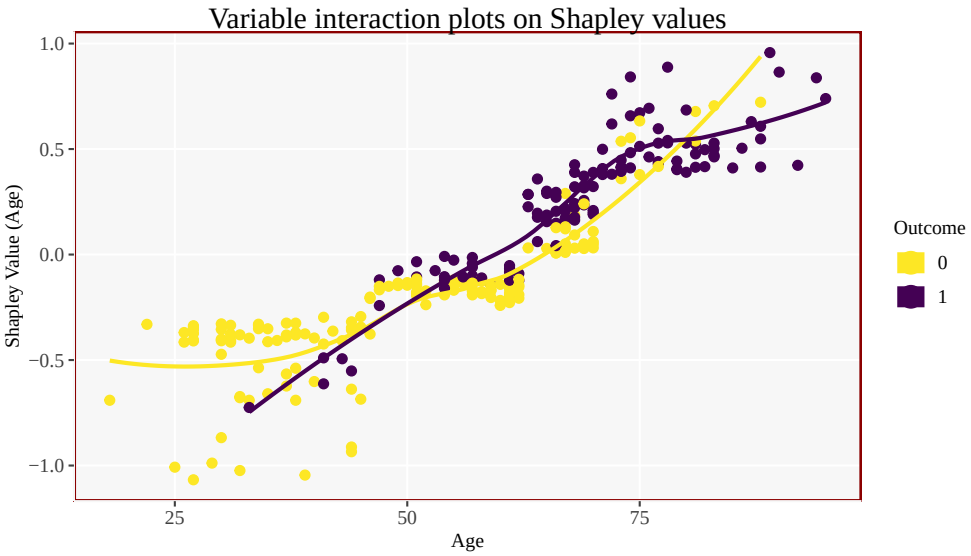


Figure 4.1.16: Variable interaction plot showing how the Shapley value changes with different ages, coloured by that patients outcome. Older patients are given positive Shapley values whereas younger patients are given negative Shapley values.

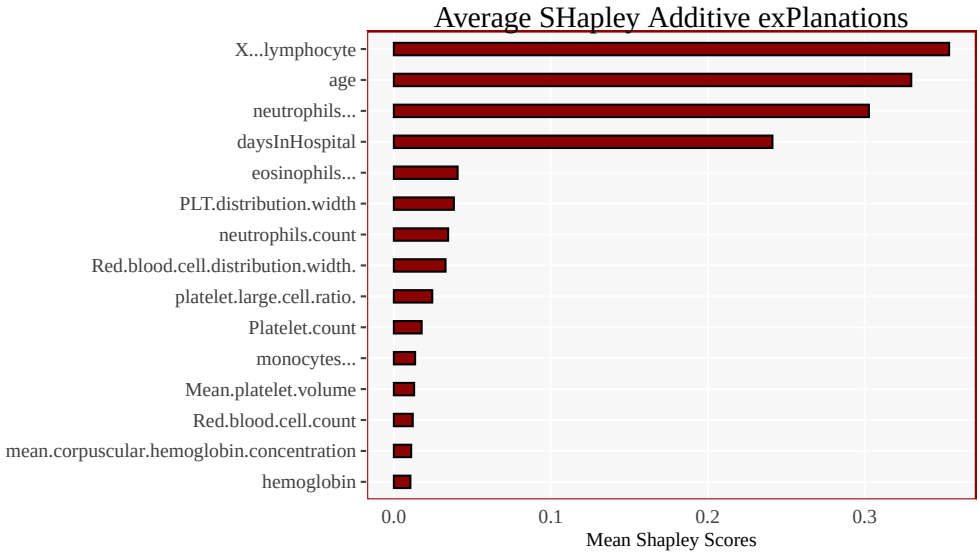


Figure 4.1.17: Average Shapley scores for the top 15 variables in the model. The results are consistent with the variable importance plots from the XGBoost and LightGBM model reported previously.

Chapter 4.2

Optimal Attendance Policy in a Replacement Problem

4.2.1 Introduction

The ability of hospitals, or ICU's, to attend to new patients, has played a central role in the fight against COVID19, a fight whose magnitude and duration has largely surpassed the prior belief of most of us. Sadly, around the peak days of contagion, hospital staff have repeatedly gone through dealing with an insufficient number of mechanic ventilators. At these peak days, the big question is: are we using our resources optimally?

Let us pose that question within a more specific -oversimplified- scenario. Two patients need simultaneously medical attention, while there are resources to attend just one patient at a time. Which one should be attended first? Continuing with the example, let us suppose that the different features to be taken into account from each patient can be summarised into a single real value, say the probability of mortality. In other words, each patient is fully characterised by his or her probability of mortality, perhaps by applying a Machine Learning model on the patient's characteristics, as in the first part of this chapter. Still, the question remains: how to prioritise between them based on the probability of mortality?

A number of conditioning factors must be stated. First, let us suppose the aim is to minimise the expected death toll, which implicitly assumes each patient's life is equally important. Second, we must specify how the treatment impacts the probability of mortality, that is, what is, in terms of that probability, the difference between receiving treatment or not. Third, we claim uncertainty and dynamics matter. The probabilistic nature of the problem is clear. In addition, it is also clear that not all patients come at the same time, which does not mean there is no congestion, but that congestion is between current and future ill people.

We claim that a dynamic optimisation problem, more specifically the so-called replace-

ment problem constitutes a natural mathematical construction that allows us to combine all of these factors and to study the optimal policy. This is, essentially our contribution with this paper. We provide a complete characterization of the optimal policy in a two period problem, while a general treatment of a T-period problem is left for ongoing research. Our two-period analysis illustrates that prioritising the patient with the higher probability of survival is rationalised when the probability of mortality is relatively low and, in addition, the marginal impact of being attended is small, which might have been the modal case for the COVID19 outbreak in western societies.

The presentation of the rest of the paper is as follows. Section 4.2.2 reviews some previous literature. Sections 4.2.3 and 4.2.4 present the theoretical analysis, containing the model and the main results using dynamic programming, respectively. Finally, Section 4.2.5 analyses the two period model.

4.2.2 Literature Review

Weissman et al. (2020) used Monte Carlo simulations of susceptible, infected and removed (SIR) models to estimate the timing of surges in clinical capacity demand across three hospitals, analysing the best-case and worst-case scenarios of COVID19 hospital capacity. Their worst-case scenario, across all three hospitals, was 12,650 occupied hospital beds, 1,608 ICU beds and 599 ventilators which has a 100% probability of exceeding the hospital's current capacity for beds, ICU and ventilators just from COVID19 patients. Their best-case scenario, across all three hospitals, was 3,131 occupied hospital beds, 338 ICU beds and 118 ventilators which has a 99.6%, 40% and 10% probability of exceeding the hospitals current capacity for beds, ICU and ventilators respectively just from COVID19 patients.

Previous studies have sought to analyse hospital admission outside of a pandemic environment. Hulshof et al. (2016) applies Approximate Dynamic Programming to the problem of tactical resource allocation and patient admission in hospitals with uncertain treatment paths and an uncertain number of arrivals in each time period. Hulshof et al. (2013) applies a tactical resource allocation and elective patient admission planning to allocate available resources to a number of care processes and determine the selection of patients to be served which are at a given stage of their care process. They use a Mixed Integer Linear Programming (MILP) framework with multi-resource, time and patient groups with uncertain treatment paths through the hospital. Ayvaz and Huh (2010) applies a dynamic programming approach to solve the problem of resource allocation in hospitals by considering two types of patients (one who waits in the hospital system until they receive a service and the other patients leave the system, or are *lost* if they cannot be accommodated immediately upon arrival). Asplin et al. (2003) developed a model of emergency department crowding which partitions the patients into three interdependent components, input, throughput and output. Their goal was to provide a practical framework on which research and policy decisions can be based in order to alleviate emergency department crowding. Zhao and Lie

(2010) applies Queuing Theory with Markov Chains (QTMC) and Discrete Event Simulation (DES) to analyse patient flow through a hospital system.

Monte Carlo simulations have also been applied in order to better understand hospital admissions under different scenarios. Kahn et al. (2007) studied the effect of transferring critically ill patients to other acute care hospitals. They apply a Monte Carlo simulation from 85 ICUs from 2002 to 2003 and compute a standardised mortality ratio (SMR) computed as the observed divided by the expected mortality for each ICU. A set number of patients were randomly assigned to be transferred out alive rather than experience their original outcome. They find that the baseline SMR was 1.06 and in the simulations, when increasing the number of transfers by 2% and 6% over the baseline, it decreased the SMR by 0.1 and 0.14, respectively. They find that increasing the number of critically ill patients out of the hospital can significantly improve the SMR of an ICU. Antognini et al. (2015) used Monte Carlo simulations to determine the number of operating rooms needed to minimise patient waiting times while optimising resources. They used patient arrival data along with the length of surgical procedures. The number of operating rooms needed to achieve acceptable waiting times depends on the arrival rate of patients and the length of surgical procedures. They found that as the number of operating rooms increased the waiting times decreased. Wu et al. (2019) analysed hospital bed resource planning using Monte Carlo simulations and queuing theory for hospital bed allocations. They consider different rates of patient arrival and length of hospital stay. They find that the average waiting time for patients in the system increases with the number of patients, indicating that the system cannot reach a steady-state and additional beds are required.

4.2.3 The Model

We consider a discrete time finite horizon model. While the natural unit of time for the problem under analysis is a day, in this section we refer to it as *period*, indexed by t, with $t \in \{0, 1, \ldots, T-1\}$, giving a total of T periods, the terminal period is $T-1$. To simplify the analysis, we assume T is known. Two fundamental features of the problem under analysis are, first, there are more patients than resources, and second, patients come sequentially, thus the whole set of patients cannot be evaluated at once nor at the start. This leads us essentially to a replacement problem. For the sake of tractability, we assume the resource is indivisible and unique, say one bed. At every period, a new ill person comes to the hospital, referred to as a *newcomer* in the period. The hospital has to make a decision of whether to release the patient who is currently occupying the bed, denoted as the *current patient*, in order to make room for the newcomer or, contrarily, the current patient stays for at least one more period, while the newcomer is left out.

Within any period t, events unfold as follows. First, the newcomer comes to the hospital. The severity of his illness is evaluated by a probability of mortality, denoted by ω_t. If there is a current patient, let x_t denote his probability of mortality. We assume that, in both cases,

that probability is a sufficient statistic of the condition of the patient. In addition, θ_t denotes the death toll at the start of the period. The pair (x_t, θ_t) conforms the state on the system. Second, based on ω_t and the state, a replacement decision is taken. We denote by $u_t = 1$ when there is a replacement: which means the current patient is released from hospital and the newcomer takes his place. Contrarily, we denote $u_t = 0$ if there is no replacement, so that the current patient stays in hospital for at least one more period, while the newcomer is left out of the hospital. Third, a shock, denoted by ε_t, is realised, by which the patient in hospital, either the newcomer or the current patient depending on the period's replacement decision, dies or not in that period, which is denoted as $\varepsilon_t = 1$ or $\varepsilon_t = 0$, respectively. Once the shock is realised, a new period starts. If there is no survivor at the end of period t, then period $t+1$ has no current patient, so that the policy at $t+1$ will be trivially $u_{t+1} = 1$. The probability distribution of ε_t depends naturally on the replacement decision and implicitly assumes that the probabilities of mortality are correctly computed, that is:

$$Pr(\varepsilon_t = 1 \mid u_t) = \omega_t u_t + x_t(1 - u_t). \tag{4.2.1}$$

Equation (4.2.1) gives a precise meaning to the probabilities ω_t and x_t. Both are *overnight*, or per-period, probabilities. We assume that, if the patient survives, the probability of mortality decreases monotonically over time. Mathematically, there is some function h with support in $[0, 1)$ satisfying $h(0) = 0$, $h(z) < z$ for all $z \in [0, 1)$ and strictly increasing such that, if there is no replacement at period t and additionally the patient does not die at that period, $x_{t+1} = h(x_t)$. In addition, if there is replacement and the patient does not die at that period, $x_{t+1} = \lambda h(\omega_t)$, where $\lambda \in (0, 1)$ is a parameter that measures the overnight improvement for being in hospital. Since it only applies when there is a replacement, we implicitly assume that the *direct added value* of staying in hospital takes place at the very first period of the stay. There is an indirect added value, which we explain below. The dynamics for the x summarises the above ideas:

$$x_{t+1} = \big(h(x_t)(1 - u_t) + \lambda h(\omega_t)u_t\big)\,(1 - \varepsilon_t) \tag{4.2.2}$$

Suppose a patient is released from hospital when his overnight probability of mortality is z. Rather than just looking at this overnight probability, when released we should look at his *overall*, that is, along some number of periods, counterpart. In the sequel, the words *overnight* or *overall* refer to the corresponding probability of mortality unless otherwise specified. Let Φ denote a function that maps overnight into overall, so that $\Phi(z)$ is the overall when z is the overnight. The construction of such a mapping involves the iterative application of h which we demonstrate below. Now, the indirect effect of being in hospital is that because the overnight decreases in the first period, the overall also decreases, with respect to not having been in hospital. Now, at each period t the death toll increases by one if the patient in hospital dies, and it additionally increases by the overall of the patient who is left out of hospital. Formally:

$$\theta_{t+1} = \theta_t + \Phi(\omega_t)(1 - u_t) + \Phi(x_t)u_t + \varepsilon_t. \tag{4.2.3}$$

The objective is to minimise the death toll at the end of the time horizon. More specifically, at the start of period T the state vector is (x_T, θ_T). We aim to select the control sequence $\{u_0, \ldots, u_{T-1}\}$ such as to minimise $\theta_T + x_T$. The control is contingent on the current state.

4.2.3.1 Exponential decay

We will assume the discrete version exponential decay to model the time evolution of the overnight, say:

$$x_{t+1} - x_t = -\delta x_t \tag{4.2.4}$$

where δ is the decay ratio, which we assume to lie in $(0, 1)$. Obviously, (4.2.4) is equivalent to take $h(x) = (1 - \delta)x$, so h satisfies the above mentioned properties. To compute the corresponding overall, we define exogenously a number of periods, say τ, and then we compute the probability of dying at any period in $\{0, 1, \ldots, \tau\}$, when the initial overnight is x_0. Clearly, there is survival along that whole time horizon if there is no death at any of those periods. Since the shocks are i.i.d. over time, the overall probability of survival is the product of all of the overnight probabilities of survival, that is, $\Pi_{t=0}^{\tau-1}(1 - x_t)$. Thus, the overall probability of mortality is:

$$\Phi = 1 - \Pi_{t=0}^{\tau-1}(1 - x_t)$$

From (4.2.4), we have $x_t = (1 - \delta)^t x_0$ for any t. Substituting in the previous expression we obtain:

$$\Phi(x_0) = 1 - \Pi_{t=0}^{\tau-1}(1 - (1 - \delta)^t x_0) \tag{4.2.5}$$

Some properties of Φ are straightforward. Increasing x_0 decreases $1 - (1 - \delta)^t x_0$ for all t, so Φ is strictly increasing. In addition, $\Phi(x_0) > x_0$ holds if and only if

$$1 - x_0 > (1 - x_0)\Pi_{t=1}^{\tau-1}(1 - (1 - \delta)^t x_0).$$

This latter inequality holds true since each factor $1 - (1 - \delta)^t x_0$ lies in $(0, 1)$ as x_0 does. Finally, direct substitution shows that $\Phi(0) = 0$ holds.

4.2.4 The basic Dynamic Programming algorithm

The essential concept in dynamic programming is the value function. At period t the value function is denoted by J_t and it maps any feasible state at the start of that period into $\mathbb{R}_+$ such that: $J_t(x, \theta)$ is the expected death toll if at the start of the period the state is (x, θ) and we undertake the optimal replacement policy thereafter. Since there is a newcomer at every period and there is just one bed, by construction it has to be $J_0(0, 0) \leq T$. The Bellman's principle of optimality establishes an equation that characterises the value function

by recursive backward induction.

We start that recursion at period T. At the start of that period (x_T, θ_T) is given, there is no replacement decision to take, so that, clearly, the value function is:

$$V_T(x_T, \theta_T) = \theta_T + \Phi(x_T) \tag{4.2.6}$$

The second term in (4.2.6) simply reflects the idea that, since there is no more replacement to do, the residual is the overall probability of mortality of the patient currently in hospital, no matter if he is released from hospital or not.

Now let us consider a given period t, with the state being $(x_t, \theta_t) = (x, \theta)$. Let us further denote the observed overnight of the newcomer by $\omega_t = \omega$. In addition, let us assume, as induction hypothesis, that next period's value function is $V_{t+1}(a, b) = b + k_{t+1}(a)$ on all of the admissible range, where k_{t+1} is some real-valued function. Notice that the induction hypothesis holds true for $t = T - 1$ just defining $k_T(a) := \Phi(a)$.

For the period t we define two auxiliary functions, V_0 and V_1, such that, for $i \in \{0, 1\}$, V_i is the expected death toll if at the current period we undertake $u_t = i$ and we behave optimally thereafter. Mathematically,

$$V_i(x, \theta, \omega) := E\{J_{t+1}(x_{t+1}, \theta_{t+1}) \mid \theta, x, \omega, u_t = i\} \qquad i \in \{0, 1\}$$

where the arguments of the function indicate what is known at the time of making the replacement decision. Clearly, the optimal replacement decision is:

$$u_t = 1 \iff V_1(x, \theta, \omega) \leq V_0(x, \theta, \omega) \tag{4.2.7}$$

We obtain:

$$V_0(x, \theta, \omega) = \theta + \Phi(\omega) + x + k_{t+1}(h(x))(1 - x) + k_{t+1}(0)x$$

where the first three terms in the right hand side correspond to $E\{\theta_{t+1} \mid x, \theta, \omega, u_t = 1\}$, whereas the latter two terms are the analogous expectation on $k_{t+1}(x_{t+1})$. Similarly,

$$V_1(x, \theta, \omega) = \theta + \Phi(x) + \omega + k_{t+1}(\lambda h(\omega))(1 - \omega) + k_{t+1}(0)\omega$$

Substituting in (4.2.7) and, re-arranging terms, we have:

$$u_t = 1 \iff k_{t+1}(\lambda h(\omega))(1-\omega)+k_{t+1}(0)\omega-\big(\Phi(\omega) - \omega\big) \leq k_{t+1}(h(x))(1-x)+k_{t+1}(0)x-\big(\Phi(x) - x\big) \tag{4.2.8}$$

In addition, it is clear that $J_t(x, \theta) = E_\omega\{min\{V_0(x, \theta, \omega), V_1(x, \theta, \omega)\}\}$. Clearly, J_t has the form $J_t(x, \theta) = \theta + k_t(x)$, just defining:

$$k_t(x) := E_\omega\{min\{\Phi(\omega)+x+k_{t+1}(h(x))(1-x)+k_{t+1}(0)x, \Phi(x)+\omega+k_{t+1}(\lambda h(\omega))(1-\omega)+k_{t+1}(0)\omega\}\} \tag{4.2.9}$$

Notice that the right hand side of (4.2.9) only depends on x. To summarise, (4.2.8) defines the optimal policy for any given next period's function k_{t+1}, whereas (4.2.9) defines the recursion to obtain k_t from k_{t+1} starting with the terminal function $k_T = \Phi$. This completes the dynamic programming algorithm to solve the replacement problem.

4.2.5 Two period problem

The above characterisation of the optimal policy is analytically complex. For this reason, this subsection deals with its simplest version, the two-period version, in which there is just one replacement decision to be made. Accordingly, we take $\tau = 2$ in (4.2.5) to define the overall, which implies $\Phi(z) = (2-\delta)z - (1-\delta)z^2$.

In the two period problem the last period is $t = 1$. At the start of that period (x_1, θ_1) is given and there is no decision to make. The value function is $J_1(x, \theta) = \theta + k_1(x)$ for any feasible state (x, θ), where $k_1 = \Phi$. Now consider period 0, with some $(x_0, \theta_0) = (x, \theta)$. While it is natural to assume $(x_0, \theta_0) = (0, 0)$, we prefer to consider general values, particularly for x_0, which determines the replacement decision. Let the observed overnight of the newcomer be $\omega_0 = \omega$. By using (4.2.8) with $t = 0$ and $k_1 = \Phi$, we have:

$$u_0 = 1 \iff \Phi(\lambda h(\omega))(1-\omega) - \big(\Phi(\omega) - \omega\big) \leq \Phi(h(x))(1-x) - (\Phi(x) - x)$$

where we have used that $k_1(0) = \Phi(0) = 0$. Noticing that $\Phi(z) - z = (1-\delta)z\,(1-z)$ and using that $h(z) = (1-\delta)z$, we can rewrite:

$$\begin{aligned}\Phi(h(x))(1-x) - (\Phi(x) - x) &= \big((2-\delta)(1-\delta)x - (1-\delta)(1-\delta)^2x^2\big)\,(1-x) - (1-\delta)x(1-x) \\ &= (1-\delta)^2(1-(1-\delta)x)(1-x)x\end{aligned}$$

Similarly,

$$\Phi(\lambda h(\omega))(1-\omega) - \big(\Phi(\omega) - \omega\big) = \big((2-\delta)\lambda - 1 - (1-\delta)\lambda^2\omega\big)\,(1-\delta)(1-\omega)\omega$$

Substituting back in the previous implication and after simplifying the common factor $1-\delta$, we have:

$$u_0 = 1 \iff \big((2-\delta)\lambda - 1 - (1-\delta)\lambda^2\omega\big)\,(1-\omega)\omega \leq (1-\delta)(1-(1-\delta)x)(1-x)x$$

Notice that the right hand side of the latter inequality is positive, so there is replacement for any pair (x, ω) if the left hand side is negative. In turn, the left hand side is negative if $(2-\delta)\lambda - 1 < 0$ holds, that is, if the improvement for being in hospital, which we can measure as λ^{-1}, is large enough in terms of the natural decay rate of the overnight, δ.

Now let us consider the case when the comparison is non-trivial, that is, when $(2-\delta)\lambda - 1 > 0$ holds. It is straightforward to prove that $(2-\delta)\lambda - 1 < 1-\delta$ holds, so that there exists a single $\mu \in (0,1)$ defined by $(2-\delta)\lambda - 1 = (1-\delta)\mu$. Using this latter equality, we can write:

$$u_0 = 1 \iff \left(\mu - \lambda^2\omega\right)(1-\omega)\omega \leq (1-(1-\delta)x)(1-x)x$$

Recall that this case is when $(2-\delta)\lambda - 1 > 0$, so we are considering a situation in which the improvement for being in hospital is small, or equivalently, $\lambda \to 1$. For the limiting case, $\lambda = 1$, we have:

$$u_0 = 1 \iff (1-\omega)^2\omega \leq (1-(1-\delta)x)(1-x)x \tag{4.2.10}$$

Let us denote by g_ω and g_x to the left and right hand side, respectively, of the inequality in (4.2.10). Straightforward analysis shows that $g_\omega(z) < g_x(z)$ for any $z \in (0,1)$, both functions are strictly positive on that interval and $g_\omega(z) = g_x(z) = 0$ for $z \in \{0,1\}$. In addition, both functions are strictly quasi-concave in $(0,1)$.[1]

Figure 4.2.1 illustrates the geometry of the inequality in (4.2.10). First, there is an interval (x_a, x_b), strictly included in $(0,1)$, such that if x lies in that interval, there will be replacement for any possible ω. If x lies outside that interval, there is replacement even if ω is very extreme, either low or high enough.

For deceased patients with low probability of mortality, it is frequently argued that the doctors should take the first the patient with the highest probability of survival. Interestingly enough, if we interpret that high probability, equivlently, low probability of mortality as both x and ω smaller than x_a, our two-period model rationalises that statement. We must bear in mind that Figure 4.2.1 assumes a large λ, that is, the treatment in hospital does not make a big difference, this is neccessary for the rationalisation. Still within a large value of λ, as the probability of mortality gets higher, a new element comes to scene, a newcomer that is very likely to die is also very likely not to use the resource for a long time.

We conclude with a final comment on the solution of the T-period problem. To the best of our knowledge, numerical methods are necessary in order to get further than what

[1] It suffices to prove the statement holds for g_x for any $\delta \in [0,1)$. Straightforward algebra shows that $g_x'(z) = 0 \iff z = 2 - \delta \pm \sqrt{1-\delta+\delta^2}$. Since $\delta \in (0,1)$, the only root of g_x in $(0,1)$ can possibly be $z^* := 2 - \delta - \sqrt{1-\delta+\delta^2}$. For $\delta = 0$, it is $z^* = 1$. For $\delta \in (0,1)$, z^* lies in $(0,1)$. In effect, $\delta < 1$ implies $\sqrt{1-\delta+\delta^2} < 1$, thus $z^* > 2 - \delta - 1 > 0$. On the other hand, $z^* < 2 - \delta - \sqrt{1-2\delta+\delta^2} = 1$. In addition, $g_x'(0) > 0$ holds. Thus, strict quasi-concavity follows.

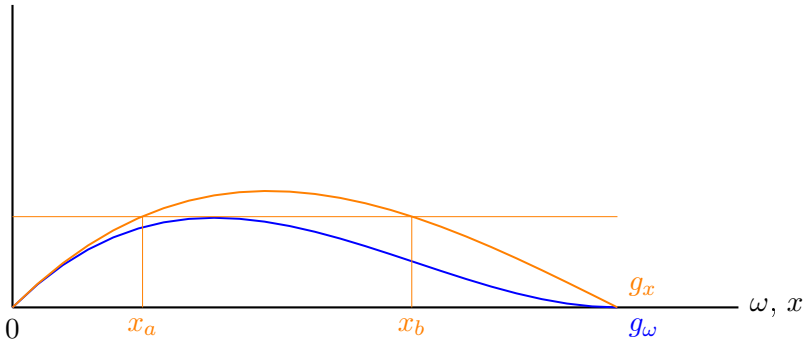


Figure 4.2.1: For shake of clarity, g_ω and g_x are represented in blue and orange, respectively. If $x \in (x_a, x_b)$, the inequality in (4.2.10) holds for any $\omega \in [0, 1]$. If both x and ω are smaller than x_a, to give priority to the patient with lowest probability of mortality is roughly optimal.

we present in this Thesis. The recursion defined by (4.2.9) allows for an exact numerical solution. In that recursion, we must notice that the support of the relevant state concerning the replacement policy, which is x, is the interval $(0, 1)$ at any period, which greatly reduces the *curse of dimensionality*, which usually limits the applicability of dynamic programming techniques.

Independently of the computational feasibility, the choice of parameter values as a previous step to apply numerical methods remains an issue. Essentially, we need a database that allows to distinguish overall from overnight probabilities of mortality, and we need to know how the latter evolve over time and how they are affected by being in hospital. The theoretical model has no other requirements. Thus, the question is, what data do we need to calibrate those probabilities? Unfortunately, the database we have used in the first part of this chapter only contains health conditions of patients when they first came to hospital, but we do not observe how those conditions evolved while they were in hospital. Alternatively, we might start with some population-wide statistics. With some exceptions, the mortality rate from Covid in western countries seems to be stabilised around 1.86%.[2] We can understand this rate as the overall probability of mortality. In addition, fixing an average duration of the illness,[3] we can fit an exponential decay law to obtain the overnights. Finally, the marginal decrease in the overnight of being in hospital, λ in our model, requires some further elaboration. Essentially, we need to compare, say overalls, for infected people who were in hospital to others who were not in hospital. This calibration exercise and the subsequent computation of the optimal policy is left as further research in this Thesis.

[2] Computed as the average for sixteen EU countries reported by Ministerio de Sanidad. Reports are done on a daily basis, the rates show great stability over time.

[3] It is not about how long the symptoms last, but for how long the probability of mortality remains relatively high as compared to its initial value.

Chapter 5

Conclusions and overall perspective

This section discusses the final remarks on the Thesis. Section 5.1 highlights the common link between the different chapters in the Thesis. Section 5.2 discusses why the XGBoost algorithm was chosen for this Thesis. Section 5.3 provides an overview of the future of Machine Learning and where I see the field heading as its own field but also in conjunction with economics. Finally, section 5.4 discusses some closing remarks.

5.1 Relation between the chapters

This Thesis has aimed to use, at the time of writing, the most up-to-date algorithms in Machine Learning to different problems in economics. The chapter topics are all very different from one another, however, the underlying and fundamental connection between the chapters is in the use of the algorithm Extreme Gradient Boosting (XGBoost) and in general, Machine Learning. This Thesis has shown how decision tree based models can be applied in economics and how their non-linear nature can be suitable for complex problems that economists face. Moreover, Shapley values from coalition game theory has also been shown to be a natural method to understand the complex predictions that Machine Learning models make.

Overall, this Thesis has tried to show that applications of Machine Learning tends to be more successful when applied to micro-level data, since, at sufficiently large scales of aggregation, the relational patterns among economic variables become more simple. Chapter 2.1, the case of bankruptcy prediction is an example of this. What followed after the financial crisis of 2008 was severe credit restrictions in which economists correctly anticipated moves in the aggregate GDP fall and increased unemployment, however, predicting how the big picture affects entities at a firm level is a more complex problem. Chapter 2.1 aimed to demonstrate how economists might analyse, not just at the aggregated level but also at the micro-level, in which, Machine Learning is instrumental. Moreover, chapter 3.1 and chapter 4.1 show that aggregation either does not make sense (chapter 3.1) or does not help (chapter 4.1).

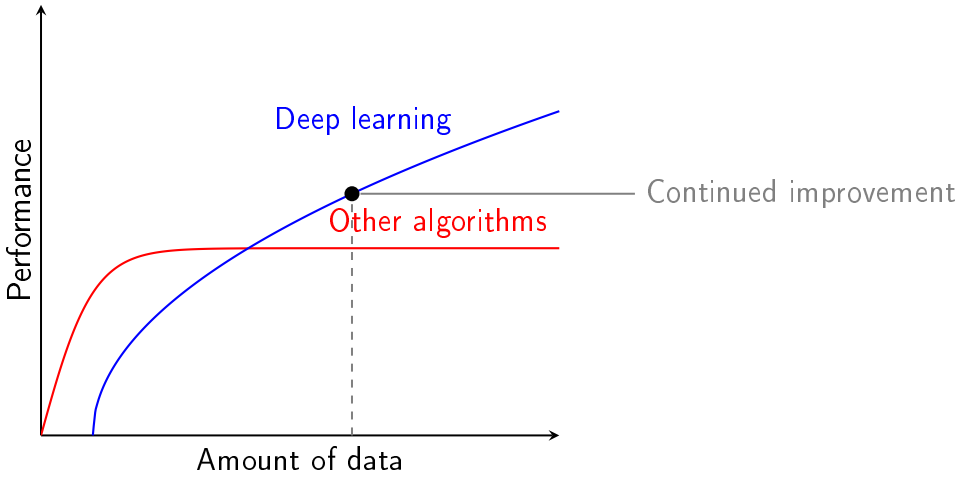


Figure 5.1: Deep Learning performance compared with other algorithms

5.2 Why XGBoost for this Thesis

This paper has used the algorithm XGBoost which is currently among the most popular Machine Learning algorithms being applied to data science tasks. Its popularity comes from its ease of use and its ability to produce powerful out-of-the-box predictions without much need for fine-tuning hyperparameters from the user. Being an ensemble learning algorithm, it combines the predictions of multiple base learners, it does so in a sequential fashion, passing to the next tree in the sequence information on how right or wrong the previous trees have been. This is turn, allows it to quickly build very powerful models, a downside to this is that it is relatively easy to overfit the training data when a tree building stopping criteria is not implemented. However, with correct procedures in place these type of problems can be accounted for.

Understandably, Deep Learning gets all of the attention these days given its advancements in Natural Language Processing, computer vision and transfer learning. However, Deep Learning models require significantly more effort in terms of training and deployment which is why it is often not used when we are required to build a quick, high-performance classifier/regressor on a structured dataset. Having said that, when the amount of data becomes sufficiently large, Deep Learning is often the best approach for the problem. Consider Figure 5.1 which illustrates the point that as the amount of data increases, Deep Learning will continue to perform whereas other algorithms plateau. So, when we do not have large amounts of training data, as is the case in many economics problems and in this Thesis, alternatives to Deep Learning are often superior, especially on supervised learning problems where the data is structured.

5.3 Future of Machine Learning and economics: My perspective

The growth in Machine Learning over recent years has been increasing at an unprecedented rate. The size of the Machine Learning market in 2019 stood at $8.43 billion and is estimated to grow to $117.19 billion by 2027 with a compound annual growth rate (CAGR) of 39.2%. Private companies and governments are investing heavily into the technology. This section discusses from my perspective what I believe the future of Machine Learning will look like and how it will impact the field of economics.

In general the future of Machine Learning lies in its ability to improve on transfer learning, which is an area in which economists can offer value. Currently, much of Machine Learning takes a task and then trains a model to perform well on that particular task. A new Machine Learning model is then trained to perform well on a different task and in doing so begins from zero. To illustrate this, suppose a human becomes very specialised at a given task over his/her life, if we want to learn a new task, then we would have to forget all previous knowledge and education and begin from being an infant again, from here we would learn everything about this new specific task. This analogy is the same as imagining that we take a brain out and replace it with a new one every time we want to learn a new task. Not only is this time consuming and inefficient from a human example, from a computational standpoint it is extremely inefficient to begin from zero each time a new task is to be learned.

One of the advances in Machine Learning would be to connect previously trained models together in order to harness what other models have already learned, that is, construct a larger network of models which can leverage previously trained models expertise. Consider a Machine Learning model which can correctly classify different species of dogs, correctly recognise your face and be used in self-driving cars. Each task is sufficiently different from each other such that, separate Machine Learning models are required at training time, however, the combination of the models will be very powerful. Moreover, if a *super* model contains thousands of trained models, each specialised in a given task, not all of the models will be required for each task, it would not be optimal to activate all of the models at the same time therefore, the *super* model will operate sparsely, activating different pieces of the *super* model when that pieces expertise is required, i.e. it will most likely be 99% idle for much of the time. The big question remains, how to route the different pieces of the model at the right time.

Transfer learning, the method of reusing previous models for different tasks is expanding, it is especially powerful when the dataset for a new task is very small and we do not have sufficient training data. Consider a model that Google trained, BERT[1], which was trained on text data from Wikipedia and BookCorpus, it took 4 days to train on Google's Tensor

[1]Bidirectional Encoder Representations from Transformers (BERT) for transformer based Machine Learning in Natural Language Processing.

Processing Units (TPU), which is quite expensive for training a Machine Learning model. However, this is a one-time procedure and now researchers can use the pre-trained model on their own Natural Language Processing (NLP) problems, even if the dataset is sufficiently small. The same model can be used on documents focusing on financial jargon, then additionally, be used on a completely new set of documents in current world news reports or sports articles. Utilising the output from the pre-trained model speeds up research time and additionally provides powerful and computationally expensive models to people who do not have the processing capabilities that Google has.

The improvement in the accuracy of Machine Learning models over the past years has been unprecedented. For example, in 2011 the most accurate model for classifying images on the ImageNet dataset[2] was 26%, humans typically have a 5% error rate on the same dataset. In 2012 the highest accuracy was 15.3% a significant drop from the previous year, this was in part due to the model being trained on a Graphical Processing Unit (GPU) which speeds up training and allowed it to process more data. In 2016, the most accurate model achieved a 3% error rate. Thus, in just a few years the error of the models fell substantially, due to improved computing power and advancements in Machine Learning research. The trend in both technological improvement and research will continue to accelerate and more accurate models will become available. This achievement has real world applications which can improve the lives of people throughout the world. It can be used in third world countries and rural villages to make diagnostics on a patients medical images where doctors are scarce, it can also be used to provide a second opinion for doctors. In relation to economics, using the advancements from the Machine Learning and computer science literature will allow economists to analyse data in different ways from traditional research, allowing them to ask and answer new research questions not previously thought of.

The improvements in technology since the 1960's has improved exponentially which has allowed the size of computers to get smaller and smaller and more powerful at the same time, however, now they are reaching their physical limits, computer parts are approaching the size of an atom and Moore's law[3] is beginning to slow. Therefore, using current technology, the exponential growth in Machine Learning capabilities will at some point begin to slow also. In order to overcome this obstacle advances in other areas will be required. One such area is in quantum computing, advances in this area will allow advances in Machine Learning to grow at an even faster rate than what we see today. These advances will see rise to a new subset of technology called *Quantum-Enhanced Machine Learning* which deals with how quantum computers can learn patterns in data which cannot be learned by classical Machine Learning algorithms. Quantum computers will allow us to move from a deterministic world to a probabilistic world which in turn increases the number of computations a computer can make. If a quantum computer had 4 Qubits (the quantum version

[2]ImageNet is a large computer vision competition in which researchers apply Machine Learning models to classify images to more than 20,000 categories. Categories such as *balloon*, *leopard*, *car* etc.

[3]Moore's law: The number of transistors on microchips doubles every two years.

of the classical bit (0 or 1) in traditional computers) it will have 16 states at the same time, as opposed to a traditional bit which can only be in one state at any given time. With 8 QuBits, we can have 256 calculations simultaneously, increasing in 2^N, where N is the number of Qubits. With a very small number of quantum bits, we can have states that represent vectors in a very high-dimensional vector space. Thus, the advancement of quantum computing will directly benefit the advancement in Machine Learning capabilities through exponential increases in computational power.

In econometrics it is up to the researcher to choose the correct model based on sound statistical theory, however in Machine Learning the model is chosen by the structure of the data. The search process for the correct model in econometrics is more difficult to automate since it requires assumptions being made about the data. In Machine Learning it is possible to systematically and programmatically search for the best model and each models corresponding optimal parameters. This automisation will allow Machine Learning to be more accessible to researchers who are not yet familiar with the programming languages utilising Machine Learning models. It will be possible to upload the data to a dashboard application, select a Machine Learning model and the output will provide relevant graphics and performance statistics for the model, without requiring knowledge of the underlying programming language used to generate the results. Therefore the accessibility of these models will be more widely available increasing its usage in disciplines which are not reliant on programming. Moreover, Machine Learning can help econometricians when the datasets have a large number of covariates and variable selection becomes an issue, Machine Learning models can help select the most relevant variables from the data and discard irrelevant ones. This in turn, would allow economists to apply a subset of precise variables chosen by Machine Learning to standard econometrics models, thus, intertwining the contribution of both fields in which one helps the other and better economic decisions can be made.

Conversely to the benefits of Machine Learning and the technological advancement, there are some caveats, not everybody uses the technology for good. For instance, advertising is about discrimination, targeting a specific group of the population in order to send promotional offers to them. This is largely accepted as good practice in the business world however, it may start to become unethical in some instances such as, a company trying to make predictions on whether somebody is pregnant or not based on surveillance data. In doing so, the company would have ascertained medically sensitive, un-volunteered data about an individual. When Machine Learning models predict incorrectly, we know the costs of such predictions, however, the aforementioned example provides a situation in which the Machine Learning model predicts correctly, but it is unethical. These same ethics apply when trying to predict sexual orientation, race, intentions to leave ones job, health status etc. Facial recognition can also be used to detect and track a persons location, security cameras can identify people and track their movements at specific times and on given days, limiting a persons freedom to move without disclosure. If a technology exists which can discriminate a subgroup of the population, somebody will use it to repress that subgroup. Machine Learning algorithms may adversely affect minorities within a society since once

identified, they may be treated differently based on which subgroup of society the model classified them into. Moreover, there has been a significant increase in the use of Deep Fakes, which changes the content of a video. This technology is very accessible and publicly available as mobile phone applications and no knowledge of Machine Learning is required. That is, a Neural Network model is given a video of, for example, Barak Obama, the voice of the president is changed to say something perhaps controversial, however the video looks very realistic and viewers are lead to believe that such a person would say such controversial things. In the future, the validity of what people say and do may come under question especially when the technology becomes so powerful and distinguishing between what a real and a fake video looks like becomes more difficult.

5.4 Closing remarks

To conclude, the future of Machine Learning is bright, the benefits far outweigh the drawbacks. However, the drawbacks still need to be taken into consideration when designing and deploying Machine Learning models in the real world. In the future, as part of Machine Learning courses, a section of the course should be devoted to ethical behaviour surrounding the use of Machine Learning and researchers and practitioners should conform to an industry standard in which they adhere to. Moreover, researchers and practitioners developing and implementing Machine Learning algorithms may not realise the full effect their models have on the lives of everyday citizens. Using chapter 2.1 as an example, if banks implemented the models proposed into their lending decision pipeline it would have a direct impact not only on the ability for the company to gain access to credit but, failure to acquire credit due to the models decision will also affect the employees of the company and/or other stakeholders, such as suppliers, customers, government and the wider-society in general. Therefore, understanding the consequences of how the end product will affect the users on the receiving end of the models prediction is an important consideration to keep in mind. Moreover, Machine Learning can and does go further than just a black-box prediction, they can provide a richer set of analytical tools for the analyst to make more informed decisions. If a bank defines a lending policy, no policy will be error free, thus the Machine Learning predictions can be combined with the case study examples such that better lending decisions can be made.

Bibliography

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X., 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. URL: http://tensorflow.org/. software available from tensorflow.org.

Acemoglu, D., Restrepo, P., 2018. Artificial intelligence, automation and work. Technical Report. National Bureau of Economic Research.

Aghion, P., Jones, B.F., Jones, C.I., 2017. Artificial intelligence and economic growth. Technical Report. National Bureau of Economic Research.

Agrawal, A., Gans, J., Goldfarb, A., 2019. Economic policy for artificial intelligence. Innovation Policy and the Economy 19, 139–159.

Altman, E.I., 1968. Financial ratios, discriminant analysis and the prediction of corporate bankruptcy. The journal of finance 23, 589–609.

Altman, E.I., Marco, G., Varetto, F., 1994. Corporate distress diagnosis: Comparisons using linear discriminant analysis and neural networks (the italian experience). Journal of banking & finance 18, 505–529.

Alvarez, F., Smith, M., 2021. Using shapley values to assess the impact of temporary traffic restrictions on no 2 levels in madrid urban area. International Journal of Environmental Science and Technology , 1–14.

An, C., Lim, H., Kim, D.W., Chang, J.H., Choi, Y.J., Kim, S.W., 2020. Machine learning prediction for mortality of patients diagnosed with covid-19: a nationwide korean cohort study. Scientific reports 10, 1–11.

Angrist, J.D., Imbens, G.W., Krueger, A.B., 1999. Jackknife instrumental variables estimation. Journal of Applied Econometrics 14, 57–67.

Angrist, J.D., Krueger, A.B., 1995. Split-sample instrumental variables estimates of the return to schooling. Journal of Business & Economic Statistics 13, 225–235.

Angrist, J.D., Krueger, A.B., 2001. Instrumental variables and the search for identification: From supply and demand to natural experiments. Journal of Economic perspectives 15, 69–85.

Antognini, J.M., Antognini, J.F., Khatri, V., 2015. How many operating rooms are needed to manage non-elective surgical cases? a monte carlo simulation study. BMC health services research 15, 1–9.

Apostolopoulos, I.D., Mpesiana, T.A., 2020. Covid-19: automatic detection from x-ray images utilizing transfer learning with convolutional neural networks. Physical and Engineering Sciences in Medicine , 1.

Arentz, M., Yim, E., Klaff, L., Lokhandwala, S., Riedo, F.X., Chong, M., Lee, M., 2020. Characteristics and outcomes of 21 critically ill patients with covid-19 in washington state. Jama 323, 1612–1614.

Arrieta, A.B., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., García, S., Gil-López, S., Molina, D., Benjamins, R., et al., 2020. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. Information Fusion 58, 82–115.

Asplin, B.R., Magid, D.J., Rhodes, K.V., Solberg, L.I., Lurie, N., Camargo Jr, C.A., 2003. A conceptual model of emergency department crowding. Annals of emergency medicine 42, 173–180.

Assaf, D., Gutman, Y., Neuman, Y., Segal, G., Amit, S., Gefen-Halevi, S., Shilo, N., Epstein, A., Mor-Cohen, R., Biber, A., et al., 2020. Utilization of machine-learning models to accurately predict the risk for critical covid-19. Internal and emergency medicine 15, 1435–1443.

Athey, S., 2017. Beyond prediction: Using big data for policy problems. Science 355, 483–485.

Athey, S., 2018. The impact of machine learning on economics, in: The economics of artificial intelligence: An agenda. University of Chicago Press, pp. 507–547.

Athey, S., Bayati, M., Imbens, G., Qu, Z., 2019. Ensemble methods for causal effects in panel data settings, in: AEA Papers and Proceedings, pp. 65–70.

Athey, S., Imbens, G.W., 2019. Machine learning methods that economists should know about. Annual Review of Economics 11, 685–725.

Atkeson, A., 2020. What will be the economic impact of COVID-19 in the US? Rough estimates of disease scenarios. Technical Report. National Bureau of Economic Research.

Autor, D., Dorn, D., Katz, L.F., Patterson, C., Van Reenen, J., 2020. The fall of the labor share and the rise of superstar firms. The Quarterly Journal of Economics 135, 645–709.

Ayvaz, N., Huh, W.T., 2010. Allocation of hospital capacity to multiple types of patients. Journal of Revenue and Pricing Management 9, 386–398.

Barboza, F., Kimura, H., Altman, E., 2017. Machine learning models and bankruptcy prediction. Expert Systems with Applications 83, 405–417.

Beaver, W.H., 1966. Financial ratios as predictors of failure. Journal of accounting research , 71–111.

Beery, S., Van Horn, G., Perona, P., 2018. Recognition in terra incognita, in: Proceedings of the European Conference on Computer Vision (ECCV), pp. 456–473.

Begley, J., Ming, J., Watts, S., 1996. Bankruptcy classification errors in the 1980s: An empirical analysis of altman's and ohlson's models. Review of accounting Studies 1, 267–284.

Bekkar, M., Djemaa, H.K., Alitouche, T.A., 2013. Evaluation measures for models assessment over imbalanced data sets. J Inf Eng Appl 3.

Bekker, P.A., 1994. Alternative approximations to the distributions of instrumental variable estimators. Econometrica: Journal of the Econometric Society , 657–681.

Bell, T.B., Ribar, G.S., Verchio, J., 1990. Neural nets versus logistic regression: a comparison of each model's ability to predict commercial bank failures, in: Auditing Symposium on Auditing Problems, pp. 29–53.

Belle, V., Papantonis, I., 2020. Principles and practice of explainable machine learning. arXiv preprint arXiv:2009.11698 .

Belloni, A., Chen, D., Chernozhukov, V., Hansen, C., 2012. Sparse models and methods for optimal instruments with an application to eminent domain. Econometrica 80, 2369–2429.

Bertsimas, D., Lukin, G., Mingardi, L., Nohadani, O., Orfanoudaki, A., Stellato, B., Wiberg, H., Gonzalez-Garcia, S., Parra-Calderón, C.L., Robinson, K., Schneider, M., Stein, B., Estirado, A., a Beccara, L., Canino, R., Dal Bello, M., Pezzetti, F., Pan, A., Group, T.H.C..S., 2020. Covid-19 mortality risk assessment: An international multi-center study. PLOS ONE 15, 1–13. URL: https://doi.org/10.1371/journal.pone.0243262.

Bessen, J., 2018. Artificial intelligence and jobs: The role of demand, in: The Economics of Artificial Intelligence: An Agenda. University of Chicago Press, pp. 291–307.

Bloom, N., Jones, C.I., Van Reenen, J., Webb, M., 2020. Are ideas getting harder to find? American Economic Review 110, 1104–44.

Borge, R., de Miguel, I., de la Paz, D., Lumbreras, J., Pérez, J., Rodríguez, E., 2012. Comparison of road traffic emission models in madrid (spain). Atmospheric Environment 62, 461–471.

Borge, R., Narros, A., Artíñano, B., Yagüe, C., Gómez-Moreno, F.J., de la Paz, D., Román-Cascón, C., Díaz, E., Maqueda, G., Sastre, M., et al., 2016. Assessment of microscale spatio-temporal variation of air pollution at an urban hotspot in madrid (spain) through an extensive field campaign. Atmospheric environment 140, 432–445.

Boser, B.E., Guyon, I.M., Vapnik, V.N., 1992. A training algorithm for optimal margin classifiers, in: Proceedings of the fifth annual workshop on Computational learning theory, pp. 144–152.

Bound, J., Jaeger, D.A., Baker, R.M., 1995. Problems with instrumental variables estimation when the correlation between the instruments and the endogenous explanatory variable is weak. Journal of the American statistical association 90, 443–450.

Breiman, L., 1996a. Bagging predictors. Machine learning 24, 123–140.

Breiman, L., 1996b. Stacked regressions. Machine learning 24, 49–64.

Breiman, L., 2001. Random forests. Machine learning 45, 5–32.

Breiman, L., Friedman, J., Stone, C.J., Olshen, R.A., 1984. Classification and regression trees. CRC press.

Breiman, L., Spector, P., 1992. Submodel selection and evaluation in regression. the x-random case. International statistical review/revue internationale de Statistique , 291–319.

Callejón, A., Casado, A.M., Fernández, M.A., Peláez, J.I., 2013. A system of insolvency prediction for industrial companies using a financial alternative model with neural networks. International Journal of Computational Intelligence Systems 6, 29–37.

Carmona, P., Climent, F., Momparler, A., 2018. Predicting failure in the us banking sector: An extreme gradient boosting approach. International Review of Economics & Finance .

Carrasco, M., 2012. A regularization approach to the many instruments problem. Journal of Econometrics 170, 383–398.

Carslaw, D.C., Murrells, T.P., Andersson, J., Keenan, M., 2016. Have vehicle emissions of primary no2 peaked? Faraday discussions 189, 439–454.

Chalfin, A., Danieli, O., Hillis, A., Jelveh, Z., Luca, M., Ludwig, J., Mullainathan, S., 2016. Productivity and selection of human capital with machine learning. American Economic Review 106, 124–27.

Chen, H., Guo, J., Wang, C., Luo, F., Yu, X., Zhang, W., Li, J., Zhao, D., Xu, D., Gong, Q., et al., 2020a. Clinical characteristics and intrauterine vertical transmission potential of covid-19 infection in nine pregnant women: a retrospective review of medical records. The Lancet 395, 809–815.

Chen, T., Guestrin, C., 2016. Xgboost: A scalable tree boosting system, in: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, ACM. pp. 785–794.

Chen, T., He, T., 2015. Higgs boson discovery with boosted trees, in: NIPS 2014 Workshop on High-energy Physics and Machine Learning, pp. 69–80.

Chen, T., He, T., Benesty, M., Khotilovich, V., Tang, Y., Cho, H., Chen, K., Mitchell, R., Cano, I., Zhou, T., Li, M., Xie, J., Lin, M., Geng, Y., Li, Y., 2018. xgboost: Extreme Gradient Boosting. URL: https://CRAN.R-project.org/package=xgboost. r package version 0.71.2.

Chen, T., He, T., Benesty, M., Khotilovich, V., Tang, Y., Cho, H., Chen, K., Mitchell, R., Cano, I., Zhou, T., Li, M., Xie, J., Lin, M., Geng, Y., Li, Y., 2020b. xgboost: Extreme Gradient Boosting. URL: https://CRAN.R-project.org/package=xgboost. r package version 1.0.0.2.

Chen, X., Racine, J., Swanson, N.R., 2001. Semiparametric arx neural-network models with an application to forecasting inflation. IEEE Transactions on neural networks 12, 674–683.

Chollet, F., et al., 2015. Keras. https://keras.io.

Chowdhury, M.E., Rahman, T., Khandakar, A., Al-Madeed, S., Zughaier, S.M., Hassen, H., Islam, M.T., et al., 2020. An early warning tool for predicting mortality risk of covid-19 patients using machine learning. arXiv preprint arXiv:2007.15559 .

Cockburn, I.M., Henderson, R., Stern, S., 2018. The impact of artificial intelligence on innovation. Technical Report. National bureau of economic research.

Colvile, R., Hutchinson, E., Mindell, J., Warren, R., 2001. The transport sector as a source of air pollution. Atmospheric environment 35, 1537–1565.

Cortes, C., Vapnik, V., 1995. Support-vector networks. Machine learning 20, 273–297.

Coulombe, P.G., Leroux, M., Stevanovic, D., Surprenant, S., 2020. How is machine learning useful for macroeconomic forecasting? arXiv preprint arXiv:2008.12477 .

Creamer, G., Freund, Y., 2004. Predicting performance and quantifying corporate governance risk for latin american adrs and banks .

De Andrés, J., Landajo, M., Lorca, P., 2005. Forecasting business profitability by using classification techniques: A comparative analysis based on a spanish case. European Journal of Operational Research 167, 518–542.

Demuzere, M., Trigo, R., Vila-Guerau de Arellano, J., Van Lipzig, N., 2009. The impact of weather and atmospheric circulation on o 3 and pm 10 levels at a rural mid-latitude site. Atmospheric Chemistry and Physics 9, 2695–2714.

Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L., 2009. Imagenet: A large-scale hierarchical image database, in: 2009 IEEE conference on computer vision and pattern recognition, Ieee. pp. 248–255.

Di Castelnuovo, A., Bonaccio, M., Costanzo, S., Gialluisi, A., Antinori, A., Berselli, N., Blandi, L., Bruno, R., Cauda, R., Guaraldi, G., et al., 2020. Common cardiovascular risk factors and in-hospital mortality in 3,894 patients with covid-19: survival analysis and machine learning-based findings from the multicentre italian corist study. Nutrition, Metabolism and Cardiovascular Diseases 30, 1899–1913.

Diebold, F.X., Shin, M., 2019. Machine learning for regularized survey forecast combination: Partially-egalitarian lasso and its derivatives. International Journal of Forecasting 35, 1679–1691.

Dietterich, T.G., 2000a. Ensemble methods in machine learning, in: International workshop on multiple classifier systems, Springer. pp. 1–15.

Dietterich, T.G., 2000b. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. Machine learning 40, 139–157.

Dietterich, T.G., 2017. Steps toward robust artificial intelligence. AI Magazine 38, 3–24.

Donaldson, D., Storeygard, A., 2016. The view from above: Applications of satellite data in economics. Journal of Economic Perspectives 30, 171–98.

Döpke, J., Fritsche, U., Pierdzioch, C., 2017. Predicting recessions with boosted regression trees. International Journal of Forecasting 33, 745–759.

Efron, B., 1992. Bootstrap methods: another look at the jackknife, in: Breakthroughs in statistics. Springer, pp. 569–593.

Einav, L., Levin, J., 2014. The data revolution and economic analysis. Innovation Policy and the Economy 14, 1–24.

Etheridge, H., Sriram, R., 1996. A neural network approach to financial distress analysis. Advances in Accounting Information Systems 4, 201–222.

European Parliament, C.o.t.E.U., 2008. Eu directive 2008/50/ec of the european parliament and of the council of 21 may 2008 on ambient air quality and cleaner air for europe, 2008.

Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A., 2010. The pascal visual object classes (voc) challenge. International journal of computer vision 88, 303–338.

Fernandes, N., 2020. Economic effects of coronavirus outbreak (covid-19) on the world economy. Available at SSRN 3557504 .

Fernández-Gámez, M.Á., Cisneros-Ruiz, A.J., Callejón-Gil, Á., 2016. Applying a probabilistic neural network to hotel bankruptcy prediction. Tourism & Management Studies 12, 40–52.

Fitzpatrick, P., 1932. A comparison of ratios of successful industrial enterprises with those of failed companies, certified public accountants. Certified Public Accountant 6, 727–731.

Freund, Y., Schapire, R.E., 1997. A decision-theoretic generalization of on-line learning and an application to boosting. Journal of computer and system sciences 55, 119–139.

Freund, Y., Schapire, R.E., et al., 1996. Experiments with a new boosting algorithm, in: icml, Citeseer. pp. 148–156.

Friedman, J.H., 2001. Greedy function approximation: a gradient boosting machine. Annals of statistics , 1189–1232.

Friedman, J.H., 2002. Stochastic gradient boosting. Computational statistics & data analysis 38, 367–378.

Frydman, H., Altman, E.I., Kao, D.L., 1985. Introducing recursive partitioning for financial classification: the case of financial distress. The Journal of Finance 40, 269–291.

Furman, J., 2018. Should we be reassured if automation in the future looks like automation in the past? The Economics of Artificial Intelligence: An Agenda , 317–328.

Gentzkow, M., Kelly, B., Taddy, M., 2019. Text as data. Journal of Economic Literature 57, 535–74.

Gonzalez, S., 2000. Neural networks for macroeconomic forecasting: a complementary approach to linear regression models .

Green, C., Heywood, J.S., Navarro Paniagua, M., 2018. Did the london congestion charge reduce pollution? mimeo .

Greene, W.H., 1993. Econometric analysis, macmillan. New York .

Hamermesh, D.S., 2013. Six decades of top economics publishing: Who and how? Journal of Economic Literature 51, 162–72.

Hansen, C., Kozbur, D., 2014. Instrumental variables estimation with many weak instruments using regularized jive. Journal of Econometrics 182, 290–308.

Hansen, J.V., Messier Jr, W.F., 1991. Artificial neural networks: foundations and application to a decision problem. Expert Systems with Applications 3, 135–141.

Hartford, J., Lewis, G., Leyton-Brown, K., Taddy, M., 2016. Counterfactual prediction with deep instrumental variables networks. arXiv preprint arXiv:1612.09596 .

Hastie, T., Tibshirani, R., Friedman, J., 2009. The elements of statistical learning: data mining, inference, and prediction. Springer Science & Business Media.

Hausman, J., Stock, J.H., Yogo, M., 2005. Asymptotic properties of the hahn–hausman test for weak-instruments. Economics Letters 89, 333–342.

Henderson, J.V., Storeygard, A., Weil, D.N., 2012. Measuring economic growth from outer space. American economic review 102, 994–1028.

Hernandez-Tinoco, M., Wilson, N., 2013. Financial distress and bankruptcy prediction among listed companies using accounting, market and macroeconomic variables. International Review of Financial Analysis 30, 394–419.

Holman, C., Harrison, R., Querol, X., 2015. Review of the efficacy of low emission zones to improve urban air quality in european cities. Atmospheric Environment 111, 161–169.

Hulshof, P.J., Boucherie, R.J., Hans, E.W., Hurink, J.L., 2013. Tactical resource allocation and elective patient admission planning in care processes. Health care management science 16, 152–166.

Hulshof, P.J., Mes, M.R., Boucherie, R.J., Hans, E.W., 2016. Patient admission planning using approximate dynamic programming. Flexible services and manufacturing journal 28, 30–61.

Ikemura, K., Goldstein, D.Y., Szymanski, J., Bellin, E., Stahl, L., Yagi, Y., Saada, M., Simone, K., Gil, M.R., 2020. Using automated-machine learning to predict covid-19 patient survival: Identify influential biomarkers. medRxiv .

Jeong, J.H., Resop, J.P., Mueller, N.D., Fleisher, D.H., Yun, K., Butler, E.E., Timlin, D.J., Shim, K.M., Gerber, J.S., Reddy, V.R., et al., 2016. Random forests for global and regional crop yield predictions. PLoS One 11, e0156571.

Jiang, W., Boltze, M., Groer, S., Scheuvens, D., 2017. Impacts of low emission zones in germany on air pollution levels. Transportation research procedia 25, 3370–3382.

Kaelbling, L.P., Littman, M.L., Moore, A.W., 1996. Reinforcement learning: A survey. Journal of artificial intelligence research 4, 237–285.

Kahn, J.M., Kramer, A.A., Rubenfeld, G.D., 2007. Transferring critically ill patients out of hospital improves the standardized mortality ratio: a simulation study. Chest 131, 68–75.

Kang, J.S., Kuznetsova, P., Luca, M., Choi, Y., 2013. Where not to eat? improving public policy by predicting hygiene inspections using online reviews, in: Proceedings of the 2013 conference on empirical methods in natural language processing, pp. 1443–1448.

Kass, G.V., 1980. An exploratory technique for investigating large quantities of categorical data. Journal of the Royal Statistical Society: Series C (Applied Statistics) 29, 119–127.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T.Y., 2017. Lightgbm: A highly efficient gradient boosting decision tree, in: Advances in neural information processing systems, pp. 3146–3154.

Kleinberg, J., Ludwig, J., Mullainathan, S., Obermeyer, Z., 2015. Prediction policy problems. American Economic Review 105, 491–95.

Kohavi, R., et al., 1995. A study of cross-validation and bootstrap for accuracy estimation and model selection, in: Ijcai, Montreal, Canada. pp. 1137–1145.

Korinek, A., Stiglitz, J.E., 2017. Artificial intelligence and its implications for income distribution and unemployment. Technical Report. National Bureau of Economic Research.

Krasin, I., Duerig, T., Alldrin, N., Ferrari, V., Abu-El-Haija, S., Kuznetsova, A., Rom, H., Uijlings, J., Popov, S., Veit, A., et al., 2017. Openimages: A public dataset for large-scale multi-label and multi-class image classification. Dataset available from https://github.com/openimages 2, 2–3.

Van der Laan, M.J., Polley, E.C., Hubbard, A.E., 2007. Super learner. Statistical applications in genetics and molecular biology 6.

Laguna-Goya, R., Utrero-Rico, A., Talayero, P., Lasa-Lazaro, M., Ramirez-Fernandez, A., Naranjo, L., Segura-Tudela, A., Cabrera-Marante, O., de Frias, E.R., Garcia-Garcia, R., et al., 2020. Il-6–based mortality risk model for hospitalized patients with covid-19. Journal of Allergy and Clinical Immunology 146, 799–807.

Lalmuanawma, S., Hussain, J., Chhakchhuak, L., 2020. Applications of machine learning and artificial intelligence for covid-19 (sars-cov-2) pandemic: A review. Chaos, Solitons & Fractals , 110059.

Lewis, D.K., 1973. Causation. The Journal of Philosophy 70, 556–567.

Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L., 2014. Microsoft coco: Common objects in context, in: European conference on computer vision, Springer. pp. 740–755.

Loh, W.Y., 2014. Fifty years of classification and regression trees. International Statistical Review 82, 329–348.

Lumbreras, J., Valdés, M., Borge, R., Rodríguez, M., 2008. Assessment of vehicle emissions projections in madrid (spain) from 2004 to 2012 considering several control strategies. Transportation Research Part A: Policy and Practice 42, 646–658.

Lundberg, S.M., Erion, G., Chen, H., DeGrave, A., Prutkin, J.M., Nair, B., Katz, R., Himmelfarb, J., Bansal, N., Lee, S.I., 2019. Explainable ai for trees: From local explanations to global understanding. arXiv preprint arXiv:1905.04610 .

Lundberg, S.M., Erion, G.G., Lee, S.I., 2018. Consistent individualized feature attribution for tree ensembles. arXiv preprint arXiv:1802.03888 .

Lundberg, S.M., Lee, S.I., 2017. A unified approach to interpreting model predictions, in: Advances in Neural Information Processing Systems, pp. 4765–4774.

Lutz, M., 2009. The low emission zone in berlin–results of a first impact assessment, in: Workshop on "NOx: Time for Compliance", Birmingham Available at: http://www.berlin.de/sen/umwelt/luftqualitaet/de/luftreinhalteplan/download/paper_lez_berlin_en.pdf November.

de Madrid, A., 2012. Plan de calidad del aire de la ciudad de madrid 2011–2015. Área de Gobierno de Medio Ambiente y Movilidad del Ayuntamiento de Madrid URL: http://www.mambiente.munimadrid.es/opencms/export/sites/default/calaire/Anexos/Plan_2011_15.pdf.

Majeed, T., Rashid, R., Ali, D., Asaad, A., 2020. Covid-19 detection using cnn transfer learning from x-ray images. medRxiv .

Makridakis, S., 1993. Accuracy measures: theoretical and practical concerns. International Journal of Forecasting 9, 527–529.

Makridis, C., Hartley, J., 2020. The cost of covid-19: A rough estimate of the 2020 us gdp impact. Special Edition Policy Brief .

Malki, Z., Atlam, E.S., Hassanien, A.E., Dagnew, G., Elhosseini, M.A., Gad, I., 2020. Association between weather data and covid-19 pandemic predicting mortality rate: Machine learning approaches. Chaos, Solitons & Fractals 138, 110137.

Martin, D., 1977. Early warning of bank failure: A logit regression approach. Journal of banking & finance 1, 249–276.

Messenger, R., Mandell, L., 1972. A modal search technique for predictive nominal scale multivariate analysis. Journal of the American statistical association 67, 768–772.

Metsky, H.C., Freije, C.A., Kosoko-Thoroddsen, T.S.F., Sabeti, P.C., Myhrvold, C., 2020. Crispr-based covid-19 surveillance using a genomically-comprehensive machine learning approach. bioRxiv .

Morgan, J.N., Sonquist, J.A., 1963. Problems in the analysis of survey data, and a proposal. Journal of the American statistical association 58, 415–434.

Moritz, S., Sardá, A., Bartz-Beielstein, T., Zaefferer, M., Stork, J., 2015. Comparison of different methods for univariate time series imputation in r. arXiv preprint arXiv:1510.03924 .

Nakamura, E., 2005. Inflation forecasting using a neural network. Economics Letters 86, 373–378.

Narin, A., Kaya, C., Pamuk, Z., 2020. Automatic detection of coronavirus disease (covid-19) using x-ray images and deep convolutional neural networks. arXiv preprint arXiv:2003.10849 .

Ng, S., 2014. Boosting recessions. Canadian Journal of Economics/Revue canadienne d'économique 47, 1–34.

Nilsson, N.J., 1984. Artificial intelligence, employment, and income. AI magazine 5, 5–5.

Núñez-Alonso, D., Pérez-Arribas, L.V., Manzoor, S., Cáceres, J.O., 2019. Statistical tools for air pollution assessment: Multivariate and spatial analysis studies in the madrid region. Journal of analytical methods in chemistry 2019.

Odom, M.D., Sharda, R., 1990. A neural network model for bankruptcy prediction, in: Neural Networks, 1990., 1990 IJCNN International Joint Conference on, IEEE. pp. 163–168.

Odriozola, J.A., Jimenez, J.D., Rubio, J.M., Pérez, I.M., Ortiz, M.P., Rodrigues, P.R., 1998. Air pollution and mortality in madrid, spain: a time-series analysis. International archives of occupational and environmental health 71, 543–549.

Ohlson, J.A., 1980. Financial ratios and the probabilistic prediction of bankruptcy. Journal of accounting research , 109–131.

Olmeda, I., Fernández, E., 1997. Hybrid classifiers for financial multicriteria decision making: The case of bankruptcy prediction. Computational Economics 10, 317–335.

Osi, A.A., Dikko, H.G., Abdu, M., Ibrahim, A., Isma'il, L.A., Sarki, H., Muhammad, U., Suleiman, A.A., Sani, S.S., Ringim, M.Z., 2020. A classification approach for predicting covid-19 patient survival outcome with machine learning techniques. medRxiv .

Pan, S.J., Yang, Q., 2009. A survey on transfer learning. IEEE Transactions on knowledge and data engineering 22, 1345–1359.

Panteliadis, P., Strak, M., Hoek, G., Weijers, E., van der Zee, S., Dijkema, M., 2014. Implementation of a low emission zone and evaluation of effects on air quality by long-term monitoring. Atmospheric environment 86, 113–119.

Peng, Y., Nagata, M.H., 2020. An empirical overview of nonlinearity and overfitting in machine learning using covid-19 data. Chaos, Solitons & Fractals 139, 110055.

Pérez, J., de Andrés, J.M., Borge, R., de la Paz, D., Lumbreras, J., Rodríguez, E., 2019. Vehicle fleet characterization study in the city of madrid and its application as a support tool in urban transport and air quality policy development. Transport Policy 74, 114–126.

Perez-Prada, F., Monzon, A., 2017. Ex-post environmental and traffic assessment of a speed reduction strategy in madrid's inner ring-road. Journal of Transport Geography 58, 256–268.

Pompe, P., Feelders, A., 1997. Using machine learning, neural networks, and statistics to predict corporate bankruptcy. Computer-Aided Civil and Infrastructure Engineering 12, 267–276.

Prosperi, M., Guo, Y., Sperrin, M., Koopman, J.S., Min, J.S., He, X., Rich, S., Wang, M., Buchan, I.E., Bian, J., 2020. Causal inference and counterfactual prediction in machine learning for actionable healthcare. Nature Machine Intelligence 2, 369–375.

Quinlan, J.R., 1986. Induction of decision trees. Machine learning 1, 81–106.

Quinlan, J.R., 2014. C4. 5: programs for machine learning. Elsevier.

R Core Team, 2013. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing. Vienna, Austria. URL: http://www.R-project.org/.

Randhawa, G.S., Soltysiak, M.P., El Roz, H., de Souza, C.P., Hill, K.A., Kari, L., 2020. Machine learning using intrinsic genomic signatures for rapid classification of novel pathogens: Covid-19 case study. PLoS One 15, e0232391.

Romero, F., Gomez, J., Rangel, T., Vassallo, J.M., 2019. Impact of restrictions to tackle high pollution episodes in madrid: Modal share change in commuting corridors. Transportation Research Part D: Transport and Environment 77, 77–91.

Rutz, O.J., Watson, G.F., 2019. Endogeneity and marketing strategy research: an overview. Journal of the Academy of Marketing Science 47, 479–498.

Sachs, J.D., 2019. R&d, Structural Transformation, and the Distribution of Income. University of Chicago Press, Chicago.

Saito, T., Rehmsmeier, M., 2015. The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. PloS one 10, e0118432.

Salas, R., Pérez Villadóniga, M.J., Prieto-Rodriguez, J., Russo, A., 2019. Restricting traffic into the city centre: Has madrid central been effective to reduce no2 levels? Available at SSRN .

Salman, F.M., Abu-Naser, S.S., Alajrami, E., Abu-Nasser, B.S., Alashqar, B.A., 2020. Covid-19 detection using artificial intelligence .

Salvador, P., Artíñano, B., Alonso, D.G., Querol, X., Alastuey, A., 2004. Identification and characterisation of sources of pm10 in madrid (spain) by statistical methods. Atmospheric Environment 38, 435–447.

Schapire, R.E., 1990. The strength of weak learnability. Machine learning 5, 197–227.

Schapire, R.E., Freund, Y., 2013. Boosting: Foundations and algorithms. Kybernetes .

Schapire, R.E., Freund, Y., Bartlett, P., Lee, W.S., et al., 1998. Boosting the margin: A new explanation for the effectiveness of voting methods. The annals of statistics 26, 1651–1686.

Schapire, Y.F.R.E., 1996. Experiments with a new boosting algorithmmachine learning: Proceedings of the thirteenth international conference.

Sermpinis, G., Stasinakis, C., Theofilatos, K., Karathanasopoulos, A., 2014. Inflation and unemployment forecasting with genetic support vector regression. Journal of Forecasting 33, 471–487.

Shapley, L.S., 1953. A value for n-person games. Contributions to the Theory of Games 2, 307–317.

Shi, F., Wang, J., Shi, J., Wu, Z., Wang, Q., Tang, Z., He, K., Shi, Y., Shen, D., 2020. Review of artificial intelligence techniques in imaging data acquisition, segmentation and diagnosis for covid-19. IEEE Reviews in Biomedical Engineering .

Singh, S., Parmar, K.S., Makkhan, S.J.S., Kaur, J., Peshoria, S., Kumar, J., 2020. Study of arima and least square support vector machine (ls-svm) models for the prediction of sars-cov-2 confirmed cases in the most affected countries. Chaos, Solitons & Fractals 139, 110086.

Smalter Hall, A., Cook, T.R., 2017. Macroeconomic indicator forecasting with deep neural networks. Federal Reserve Bank of Kansas City Working Paper .

Smith, M., Alvarez, F., 2021a. Identifying mortality factors from machine learning using shapley values - a case of covid19. Expert Systems with Applications , 114832URL: https://www.sciencedirect.com/science/article/pii/S0957417421002736, doi:https://doi.org/10.1016/j.eswa.2021.114832.

Smith, M., Alvarez, F., 2021b. Identifying mortality factors from machine learning using shapley values - a case of covid19. https://www.codeocean.com/. doi:10.24433/CO.1107431.v1.

Smith, M., Alvarez, F., 2021c. Predicting firm-level bankruptcy in the spanish economy using extreme gradient boosting. Computational Economics , 1–33.

Staiger, D., Stock, J.H., 1994. Instrumental variables regression with weak instruments. Technical Report. National Bureau of Economic Research.

Stock, J.H., Watson, M.W., 1998. A comparison of linear and nonlinear univariate models for forecasting macroeconomic time series. Technical Report. National Bureau of Economic Research.

Sunyer, J., Castellsagué, J., Sáez, M., Tobias, A., Antó, J.M., 1996. Air pollution and mortality in barcelona. Journal of Epidemiology & Community Health 50, s76–s80.

Tam, K.Y., Kiang, M.Y., 1992. Managerial applications of neural networks: the case of bank failure predictions. Management science 38, 926–947.

Theodossiou, P.T., 1993. Predicting shifts in the mean of a multivariate time series process: an application in predicting business failures. Journal of the American Statistical Association 88, 441–449.

Valverde, V., Pay, M.T., Baldasano, J.M., 2016. Ozone attributed to madrid and barcelona on-road transport emissions: Characterization of plume dynamics over the iberian peninsula. Science of the total environment 543, 670–682.

Varian, H., 2018. Artificial intelligence, economics, and industrial organization. Technical Report. National Bureau of Economic Research.

Varian, H.R., 2014. Big data: New tricks for econometrics. Journal of Economic Perspectives 28, 3–28.

Vedrenne, M., Borge, R., Lumbreras, J., Conlan, B., Rodríguez, M.E., de Andrés, J.M., de la Paz, D., Pérez, J., Narros, A., 2015. An integrated assessment of two decades of air pollution policy making in spain: Impacts, costs and improvements. Science of the Total Environment 527, 351–361.

Wang, G., Zhang, Y., Zhao, J., Zhang, J., Jiang, F., 2020. Mitigate the effects of home confinement on children during the covid-19 outbreak. The Lancet 395, 945–947.

Wang, L., Wong, A., 2020. Covid-net: A tailored deep convolutional neural network design for detection of covid-19 cases from chest radiography images. arXiv , arXiv–2003.

Weissman, G.E., Crane-Droesch, A., Chivers, C., Luong, T., Hanish, A., Levy, M.Z., Lubken, J., Becker, M., Draugelis, M.E., Anesi, G.L., et al., 2020. Locally informed simulation to predict hospital capacity needs during the covid-19 pandemic. Annals of internal medicine 173, 21–28.

Weld, D.S., Bansal, G., 2019. The challenge of crafting intelligible intelligence. Communications of the ACM 62, 70–79.

West, R.C., 1985. A factor-analytic approach to bank condition. Journal of Banking & Finance 9, 253–266.

Westfall, P.H., Troendle, J.F., Pennello, G., 2010. Multiple mcnemar tests. Biometrics 66, 1185–1191.

Wilson, R.L., Sharda, R., 1994. Bankruptcy prediction using neural networks. Decision support systems 11, 545–557.

Winakor, A., Smith, R., 1935. Changes in the financial structure of unsuccessful industrial corporations. Bulletin 51, 44.

Wolpert, D.H., 1992. Stacked generalization. Neural networks 5, 241–259.

Wu, K., Zhu, X., Zhang, R., Liu, S., 2019. Hospital bed planning in a single department based on monte carlo simulation and queuing theory, in: 2019 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), IEEE. pp. 644–648.

Wynants, L., Van Calster, B., Bonten, M.M., Collins, G.S., Debray, T.P., De Vos, M., Haller, M.C., Heinze, G., Moons, K.G., Riley, R.D., et al., 2020. Prediction models for diagnosis and prognosis of covid-19 infection: systematic review and critical appraisal. bmj 369.

Xia, Y., Liu, C., Li, Y., Liu, N., 2017. A boosted decision tree approach using bayesian hyper-parameter optimization for credit scoring. Expert Systems with Applications 78, 225–241.

Yan, L., Zhang, H.T., Goncalves, J., Xiao, Y., Wang, M., Guo, Y., Sun, C., Tang, X., Jing, L., Zhang, M., et al., 2020a. An interpretable mortality prediction model for covid-19 patients. Nature Machine Intelligence doi:10.1038/s42256-020-0180-7.

Yan, L., Zhang, H.T., Xiao, Y., Wang, M., Sun, C., Liang, J., Li, S., Zhang, M., Guo, Y., Xiao, Y., et al., 2020b. Prediction of criticality in patients with severe covid-19 infection using three clinical features: a machine learning-based prognostic model with clinical data in wuhan. MedRxiv .

Zhang, J., Xie, Y., Li, Y., Shen, C., Xia, Y., 2020. Covid-19 screening on chest x-ray images using deep learning based anomaly detection. arXiv preprint arXiv:2003.12338 .

Zhao, D., Huang, C., Wei, Y., Yu, F., Wang, M., Chen, H., 2017. An effective computational model for bankruptcy prediction using kernel extreme learning machine approach. Computational Economics 49, 325–341.

Zhao, L., Lie, B., 2010. Modeling and simulation of patient flow in hospitals for resource utilization. Simul. Notes Eur. 20, 41–50.

Zhou, L., Lai, K.K., 2017. Adaboost models for corporate bankruptcy prediction with missing data. Computational Economics 50, 69–94.

Zięba, M., Tomczak, S.K., Tomczak, J.M., 2016. Ensemble boosted trees with synthetic features generation in application to bankruptcy prediction. Expert Systems with Applications 58, 93–101.

Zmijewski, M.E., 1984. Methodological issues related to the estimation of financial distress prediction models. Journal of Accounting research , 59–82.

Index