NOVA IMS Information Management School

MAA

Mestrado em Métodos Analíticos Avançados
Master Program in Advanced Analytics

The importance of Quality Assurance as a Data Scientist

Common pitfalls, examples, and solutions found while validating and developing supervised binary classification models

Vitor Manuel Cruz Manita

Internship Report presented as partial requirement for obtaining the Master's degree in Data Science & Advanced Analytics

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

THE IMPORTANCE OF QUALITY ASSURANCE AS A DATA SCIENTIST

COMMON PITFALLS, EXAMPLES, AND SOLUTIONS FOUND WHILE VALIDATING AND DEVELOPING SUPERVISED BINARY CLASSIFICATION MODELS

Author

Vitor Manita

Internship Report presented as a partial requirement for obtaining the Master's degree in Advanced Analytics

Advisor: Flávio L. Pinheiro

Co-Advisor*:* Catarina R. Calhau

November 2020

DEDICATION

This report is dedicated to my parents Maria and Vitor, who gave me invaluable education opportunities and supported me with endless love.

Everything I have or will ever achieve is because, and to, them.

ACKNOWLEDGEMENTS

I would like to acknowledge my advisor, Flávio L. Pinheiro, for his teachings, understanding, and guidance during this time. From my professor to a colleague and friend, he always pushed me to achieve more.

I also want to express my gratitude to my co-advisor at EDP D, Catarina R. Calhau, for her kind support and attention since the first day. This appreciation is also extended to my amazing team. I look forward to the upcoming and disruptive projects that we, together, will accomplish.

Lastly, my gratefulness goes to my girlfriend, Andreia, for the unconditional love and motivation, and to the ones most close to me, Mafalda, Xavier, Rita, Helena, Rodrigo, João, and Rafael, who supported me during this entire journey.

ABSTRACT

In today's information era, where Data galvanizes change, companies are aiming towards competitive advantage by mining this important resource to achieve actionable insights, knowledge, and wisdom. However, to minimize bias and obtain robust long-term solutions, the methodologies that are devised from Data Science and Machine Learning approaches benefit from being carefully validated by a Quality Assurance Data Scientist, who understands not only both business rules and analytics tasks, but also understands and recommends Quality Assurance guidelines and validations.

Through my experience as a Data Scientist at EDP Distribuição, I identify and systematically report on seven key Quality Assurance guidelines that helped achieve more reliable products and provided three practical examples where validation was key in discerning improvements.

KEYWORDS

EDP; Data Science; Machine Learning; Supervised Learning; Quality Assurance

INDEX

1. Introduction 1
   1.1. Enterprise Context 1
      1.1.1. EDP Distribuição History and Timeline 1
      1.1.2. EDP Distribuição in Numbers 2
      1.1.3. Distribuição Team and Hierarchy 3
   1.2. The Project Overview 3
2. Theoretical Framework 5
   2.1. Quality Assurance (QA) 5
   2.2. Supervised Classification Machine Learning 6
      2.2.1. Classification Algorithms: Black Box vs. White Box 8
      2.2.2. Performance Metrics for Supervised Classification Problems 12
   2.3. OFGEM 14
      2.3.1. Initial Health Score (Initial End of Life). 15
      2.3.2. Current Health Score (Intermediate End of Life) 16
      2.3.3. Long-term Health Score (Forecasting End of Life) 17
      2.3.4. Forecasting Proof of Failure 17
3. Framework and Data Architecture 19
4. Quality Assurance Guidelines 21
   4.1. Documentation: Handover and Code 21
   4.2. Data Visualization can be Misleading 23
   4.3. Data Leakage by ignoring time in time-series data 24
   4.4. Blindly encoding categorical data 25
   4.5. Democratizing knowledge and managing expectations 26
   4.6. Model interpretability vs predictive performance 26
   4.7. One metric (does not) fit all 27
5. Quality Assurance in Practice: Projects 29
   5.1. Predictive Failure Classification 30
      5.1.1. Data Understanding 30
      5.1.2. Data Preparation 30
      5.1.3. Data Modelling 31
      5.1.4. Quality Assurance 32
   5.2. Health Index 34
      5.2.1. OFGEM Methodology 34
      5.2.2. Health Index Work and Results 35

5.2.3. Quality Assurance 36
5.3. Investment and Maintenance Planning Dashboards 37
5.3.1. Quality Assurance 37
6. Conclusions 38
7. Bibliography 39

LIST OF FIGURES

Figure 1. EDP & EDP Distribuição logos 1
Figure 2. EDP D organizational chart 3
Figure 3. Six Sigma DMAIC methodology 5
Figure 4. Supervised classification machine learning process....7
Figure 5. K-Cross Validation with average classifier performance 8
Figure 6. Logistic Regression curved fitting example....9
Figure 7. Decision Tree classifier example of how predictive features are split into branches that either lead to other splitting nodes or the classification 10
Figure 8. Random Forest classifier example on how independent trees work together to vote on a prediction....11
Figure 9. OFGEM Exponential Curve: Expected Asset Health Index considering the aging rate factor 15
Figure 10. Data flow for EDP D projects 19
Figure 11. Documentation of code - Function example 22
Figure 12. Lie Factor (illustrative example) 23
Figure 13. Sliding window cross-validation 24
Figure 14. Nested cross-validation....24
Figure 15. An alternative to One-Hot-Encoding to high dimensional classification datasets ..25
Figure 16. Classifier metrics comparison - How accuracy can be misleading 27
Figure 17. Train (80%) Test (20%) Split....30
Figure 18. Asset Failure - How visualizing failure rate can be misleading (illustrative example) 32
Figure 19. OFGEM framework architecture - How factors influence Health score and Probability of Failure....34
Figure 20. Health Index understandability adaptation 35
Figure 21. Health Index results by asset....36
Figure 22. Investment Planning Dashboard snapshot 37

LIST OF TABLES

Table 1. Confusion matrix for binary classification problems 12
Table 2. Classification metrics to evaluate the performance 13
Table 3. One-Hot-Encoding (illustrative example).....25
Table 4. Final model iterations comparison using AUC as the decisive metric 31
Table 5. One-Hot-Encoding alternative (illustrative example).....33
Table 6. Health Index Bands – HI1 (New condition) to HI5 (Poor condition) 35

LIST OF ABBREVIATIONS AND ACRONYMS

EDP	Electricidade de Portugal, S.A.
EDP D	EDP Distribuição
ML	Machine Learning
QA	Quality Assurance
MVP	Minimum Viable Product
OFGEM	Office of Gas and Electricity Markets
HI	Health Index

1. INTRODUCTION

In this report, I provided a discussion of the best Data Science practices that can be incorporated in the industry, through my experience as a Data Scientist for EDP Distribuição (EDP D). Having participated in the development of advanced analytics solutions, I took responsibilities as a Quality Assurance Data Scientist, having validated key analytical tasks and decisions made during projects.

Quality Assurance (QA) systematic activities offer significant benefits, ensuring the quality standards of a product are being met within an organization. Such tasks, when done right, increase the efficiency and effectiveness of operations, minimizing costs, time, and resources. Within a Data Science project, the role of a Quality Assurance Data Scientist, who understands both business rules and analytical tasks, helps to assure minimum bias during development, and to guarantee that a set of short-term validations are satisfied, which, consequently, have a long-term impact.

I identify and systematically report on seven key QA guidelines and three projects that express the importance of these validations in any Machine Learning approach. It is also important to express that, during the development of the discussed practical cases, despite having participated as a developer, my main role was the one of Quality Assurance Data Scientist.

1.1. ENTERPRISE CONTEXT

Electricidade de Portugal, S.A. (EDP) is a publicly-traded holding company among the major European operators in the energy sector (Gas and Electricity). It is also one of the largest energy operators of the Iberian Peninsula, the largest Portuguese industrial group, and the world's fourth-largest producer of wind energy.

Figure 1. EDP & EDP Distribuição logos

1.1.1. EDP Distribuição History and Timeline

The history of electricity in Portugal began in 1878 when the first step towards the future was given on public illumination when republican members of the parliament were firstly elected in the Portuguese Parliament. On the 15th birthday of Prince D. Carlos, the royal family imported six voltaic arc lamps from Paris to be lit on the Cascais Citadel esplanade, an offer from the city of Lisbon. They were reused in the following month in the Chiado area and the Portuguese people were thrilled with this disruptive technology. Around ninety-eight years into the future, after the Portuguese dictatorship was overthrown in the 1975 revolution, the country's power generation assets and distribution chains were nationalized in a Marxist approach towards the country's governance. In the following year, the Portuguese Minister of Industry and Technology published decree-law Nr. 502/76, published on June 30, 1976, that merged the major Portuguese electricity companies and founded EDP as a state-controlled organization.

Twenty-one years later, in 1996, the first community directive was published, leading to significant changes in the organization and jurisdiction of the European electricity sector. Antonio de Almeida, as the new chairman of EDP, began the first phase of privatization in 1997, allowing for the sale of 30% of EDP's capital, accounting for roughly 368 Billion Euros. With it, EDP's stocks rose 38% on Lisbon's exchange and became the largest Portuguese company. Four privatization phases later, in 2000, EDP was now a majority private company, where private stakeholders accounted for 70% of the capital while the State share was reduced to 30%. Furthermore, one of the market liberalization consequences was the obligation to juridically separate the transportation and distribution activities from production and commercialization. In this turn of the century, EDP Distribuição was also born.

Starting in 2006, all electricity consumers were given the capacity to choose their supplier. In addition to the responsibility of operating the distribution network, EDP D is also responsible for managing all supplier's changes processes. In 2010, Évora was chosen to hold a disruptive idea – *InovCity* – the first smart city in the Iberic Peninsula. Around 2013, EDP becomes the *"Utilities: Electricity, Water & Gas"* leader in the Dow Jones Sustainability Index World and Europe, achieving the highest score ever recorded of 90 points. In the same year, between January 19 and January 23, Portugal was hit hard by the Gong Storm, causing significant losses, and many EDP customers to lose their electricity service. As an acknowledgment of the capability of recovering the energy network and customer service, EDP D was awarded the "*Most Effective Recovery of the Year"* prize by the Business Continuity Institute in 2014. In 2015, EDP D joins Alliander, ENEL, EVN, and other industry partners and universities as a founding member of the European Energy Centre- Information Sharing and Analysis (EE-ISAC). EDP D also becomes a European reference through the certification of the Business Continuity Management System by the Business Continuity International Standard ISSO 22301:2012. In 2016, EDP Distribuição embraces an APP for consumers to submit meter readings or report power cuts, increasing the efficiency and effectiveness of these procedures. It was also in this year that EDP Distribuição was certified by Societé Générale de Surveillance (SGS), making the Management System for Research, Development, and Innovation (SGIDI) the first national utility to be certified for innovation.

In the following year, 2017, EDP D achieves the *"Zero Fatal Injuries"* landmark for the first time, as a result of the operational teamwork, security training, and awareness campaigns. Lastly, in 2018, EDP approves the EDP D 2020 sustainability strategy, aiming to contribute to the United Nations Sustainable Development Goals (ODS) agenda. EDP D was also awarded by the Renewables Grid Initiative (RGI) in the *"Environmental Protection"* category for *"Good Practice of the Year Awards"*, showcasing the initiatives and outstanding practices by EDP in grid development. EDP D is also 100% certified in the Environmental Management System (EMS) according to ISO 13001:2015.

1.1.2. EDP Distribuição in Numbers

EDP D is responsible for supplying electricity from its source to the final customer. Its mission is to ensure the distribution of electricity with the highest quality, safety, and efficiency. It also provides support to the electrical market, the energy transition, and the decarbonization of energy consumption.

In 2019, EDP Distribuição network served 6 277 358 clients, accounting for the regulated and free market, where 74 (0.001%) were VHV (Very High Voltage), 316 (0.005%) HV (High Voltage), 25 022 (0.399%) MV (Medium Voltage), 37 144 (0.592%) SLV (Special Low Voltage), 6 152 431 (98.010%) NLV (Normal Low Voltage), and lastly, 62 371 (0.994%) PL (Public Lighting). As such, the Normal Low Voltage clients account for almost the whole share of the EDPD network, whereas Very High Voltage and High Voltage clients are almost inexistent. Furthermore, in this network, 27 000 new MV

(Medium Voltage) and LV (Low Voltage) connections were ordered, as well as 52 214 000 Readings, and 671 000 power cuts.

The total network size holds 228 046 Km, whereas the High Voltage Network accounts for 9 532 Km (4.2%), the Medium Voltage Network for 73 850 Km (32.4%), and the Low Voltage Network for 144 664 Km (63.4%). Additionally, there are also 69 190 secondary substations and 431 primary substations. In terms of operational and financial indicators, EDPD was accountable for 3 085 employees, having a Net Profit of 78 Million Euros, an Operational Investment (CAPEX) of 305 Million Euros, and a 474 Million Euros EBITDA**.**

1.1.3. Distribuição Team and Hierarchy

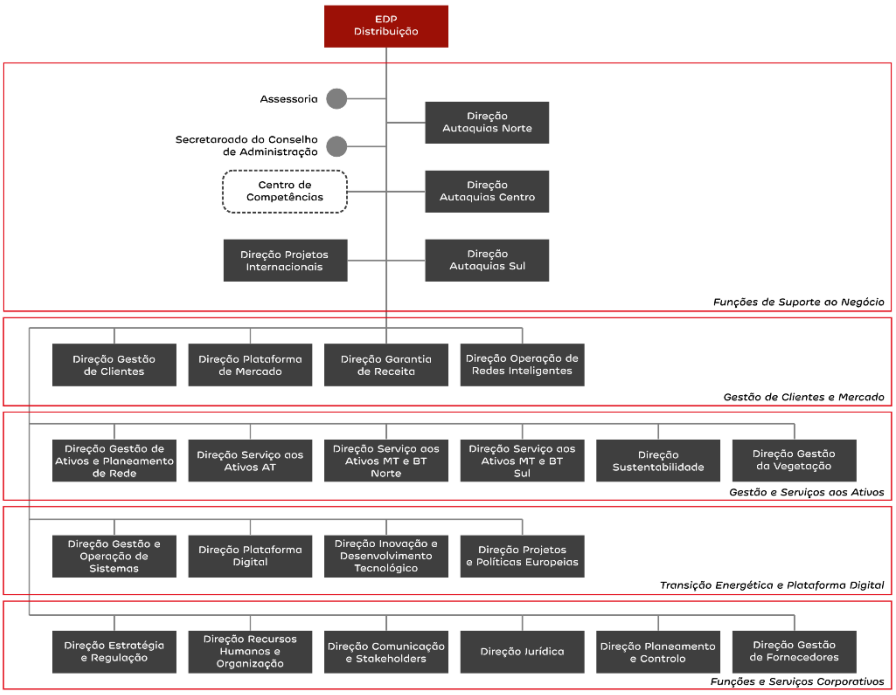


Figure 2. EDP D organizational chart

At EDP D, I occupy the role of a Data Scientist in the Data Analytics and Management team (Analítica e Gestão de Dados), a group inside the Digital Boost Area (Área de Acelaração Digital) that, in turn, is a sub-group of the Digital Platform Management (Direção Plataforma Digital), in the Energetic Transition and Digital Platform (Transição Energética e Plataforma Digital) (see Figure 2). In other words, the full hierarchy chain to my position is EDPD-DPD-AAD-AGD. This team's vision is to implement a data-driven and data-informed culture at EDP D, by developing an audit analytics-related projects, validating methodology and documentation, monitoring analytical models, promoting and improving their evolution, empowering other business units by organizing analytics sessions that democratize this knowledge with business-oriented experts, and it makes an effort to be aware of new technologies, tools, and methods in this field.

1.2. THE PROJECT OVERVIEW

EDP D is aiming to achieve its digital strategy and vision by unlocking business value through advanced analytics, thus building a data-driven and data-informed culture. In this context, my team and I have a key role in validating projects where data is the main resource. Hence, Quality Assurance validations are fundamental in our participation. Having some guidelines identified, I will provide practical

examples of how they can be used as improvements for more robust solutions. The considered projects that aimed to support decision-making by data means were Predictive Asset Failure Classification, Asset's Health Index estimation, and the construction of Investment and Maintenance Planning Dashboards to hold this information

2. THEORETICAL FRAMEWORK

In today's information era, where Data galvanizes change, companies are aiming towards competitive advantage by mining this important resource to achieve actionable insights, knowledge, and wisdom. However, in a time wherever larger and more variated data is being generated, some challenges oppose the simple techniques and tools that once were optimal for these kinds of analysis. Therefore, new solutions were devised, and Data Science and Machine Learning are now indispensable terms when dealing with data. In this chapter, we will describe the fundamentals of supervised classification machine learning and techniques to assess asset's health which will be useful to better understand the quality assurance guidelines provided in this report, and the discussed practical cases.

2.1. QUALITY ASSURANCE (QA)

Quality Assurance (QA) is a systematic planed model of activities that ensure the quality standards of a product are being met within an organization. Such tasks, when done right, are key to increase operations efficiency and effectiveness, minimizing costs, time, and resources.

QA, however, is not a recent concept. During the First World War, also known as the War of Production, workers were paid for each unit they produced (Piece Work). Given the high demand for supplies, the World witnessed an era of mass production. Nevertheless, as manpower increased and manufacturing processes became more complex, units also became more susceptible to defects and poor quality amongst assembly lines. Visionaries, such as Henry Ford, introduced disruptive methods that tackled the lack of quality by standardizing the design and component standards, while also assigning accountability to the "Machine Inspectors", which were quality assurance professionals placed in each department, guaranteeing faulty operations were limited to a short period (Wilson, 2014).

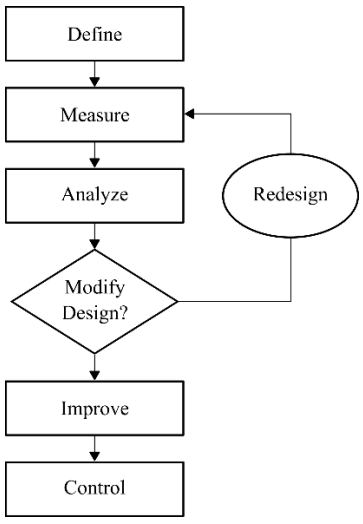


Figure 3. Six Sigma DMAIC methodology

Many years and several developments later, QA evolved into a well-known concept. One of the most respected methodologies is the Six Sigma DMAIC (Define, Measure, Analyze, Improve Control) methodology. It consists of a data-driven quality strategy, where each letter in the acronym represents five different phases (see Figure 9). First, the process requires a definition of the problem, the opportunity for improvement, the project goals, and the customer requirements. Then, it measures the performance of the process by collecting all the necessary data to determine the types of defects and metrics. Afterward, the previously collected data is analyzed, identifying root-causes of defects and gaps

between current performance and goal performance. Then, it improves the process performance by addressing and eliminating the root causes, designing creative solutions. Finally, it controls the improvements to keep the project on the new course (Tsung, 2006).

In data science, QA tasks can be commonly misinterpreted as the activities done during the final phases of a Machine Learning project pipeline, such as accessing the error on the testing set. However, there is a lot to learn from methodologies such as the Six Sigma DMAIC. During the exploratory data analysis and feature engineering phases, the data quality must be ensured, requiring input data sources to be validated by business experts and automated tests, including data transformations, comparisons, data types checking, and accessing missing values and outliers. When modeling, overfitting should also be controlled thoroughly, ensuring the model training simulates the reality as much as possible, which means splitting the data into different partitions, choosing and comparing models, choosing evaluation metrics, and accessing the error. Furthermore, there are also transversal guidelines that should be tackled, such as the correct documentation of all steps, ensuring systems integration, and deployment success. In this report, I intend to explore possible applications of QA in a Machine Learning project as best-practices to follow, which you can find in chapter *4. Quality Assurance Guidelines*, and practical examples on real projects in chapter *5. Quality Assurance in Practice: Projects*.

2.2. SUPERVISED CLASSIFICATION MACHINE LEARNING

Machine Learning (ML) can be described as the automation of data exploration and analytics tasks by using advanced techniques, usually beyond the traditional Business Intelligence solutions, aiming to acquire more knowledge about a problem. The overall idea is that the system should be able to learn from data, identify patterns, and make decisions requiring the minimum Human interaction.

Every dataset in Machine Learning is represented by a set of features, which can be continuous, categorical, or binary. When the corresponding label for each data point is known (see Table 1), we refer to Supervised Learning, as it's the case of Classification and Regression models. In contrast, when previous-labeled data does not exist, it is called Unsupervised Learning, as it happens with clustering algorithms, unsupervised classification of patterns into groups (Jain et al., 1999). Lastly, Machine Learning use cases can also be described as Reinforced Learning examples. This is the development of intelligent models that, according to the received data, must find the patterns and actions that yield better results, awarding or penalizing them (Botvinick et al., 2011).

Observation	Feature 1	Feature 2	…	Feature k	Class
Obs. 1	X	X	…	X	1
Obs. 2	X	X	…	X	0
…	…	…	…	…	…
Obs. n	X	X	…	X	1

Table 1. Dataset with known labels: Predictive features and corresponding target class

In this report, and practical cases discussed, I will be focusing on Supervised Learning, more precisely, on Classification problems and techniques. Classification is one of the fundamental problems of Supervised Learning, and one of the most common in Machine Learning applications. It requires previously labeled data, so-called examples, and builds models to classify data points based on predictor features, learning and adjusting the algorithm's weights according to the error, or difference, between the observed labels and the predicted ones. The result is a classifier that classifies new data points whose label is unknown (Kotsiantis, 2007).

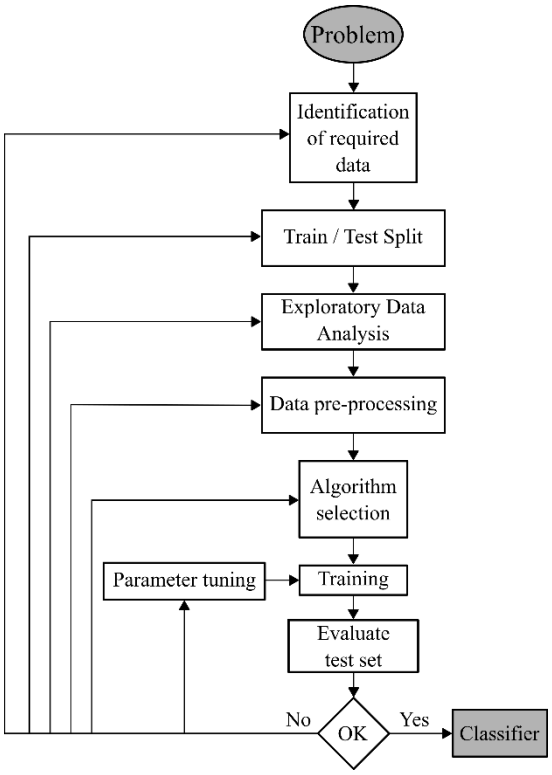


Figure 4. Supervised classification machine learning process

The process of building supervised classification models can follow a similar structure to the ones found in the CRISP-DM methodology (Shearer et al., 2000) and OSEMN (Mason, 2010) framework (see Figure 3). Having identified and understood a business problem that Machine Learning can tackle, the process begins by identifying which data is needed, thus requiring multidisciplinary teams, that benefit from domain-knowledge inputs that will facilitate the following steps, exploratory data analysis (EDA) and data pre-processing.

However, before proceeding, it is important to define the training set. This will allow to assess the final classifier performance on unseen data and reduce the bias of developing an overfitted model (Leinweber, 2007) where results are too adjusted to a particular set of data, and the classifier is likely to fail when predicting unseen observations. Thus, depending on the existing amount of data, it is possible to define a training set in which all discoveries, decisions, and classifier training are made, applied, and tested on the testing set. This can be done by simply splitting the data into these two subsets and, to have a prior estimation of the classifier's performance, one can leave a small proportion of the training set as validation, even though the more decisions are made according to the validation results, the more biased the classifier will be to that data subset. A better approach is to use cross-validation techniques (see Figure 4), in which the training set is divided into mutually exclusive and equal-sized subsets for *n* iterations and, in each iteration, the greater number of subsets is used to train the classifier while the remaining subset is used for validating the classifier estimations performance. In each iteration the fold that is selected as validation is independent of each other, this is, it will not be part of the validation subset on the other iterations, thus ensuring all data was, at some point, selected to train the classifier and validate its performance, but never in the same iteration. After all *n* iterations, the classifier's performance estimation will be an average of all performance metrics recorded in each independent iteration.

The most common cross-validation techniques are the K-Fold cross-validation, in which data is divided into *K* equal-sized subsets, and Stratified K-Fold cross-validation, which assures the proportion of

samples from a given class label is preserved in all folds, thus, a favorable option when dealing with imbalanced datasets.

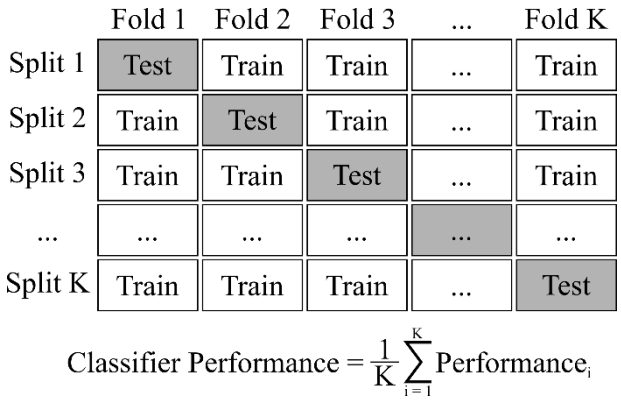


Figure 5. K-Cross Validation with average classifier performance

During the EDA phase, data exploration and summarization can lead to initial discoveries that facilitate data pre-processing tasks, such as the identification of patterns and associations in data, data types, missing values, and outliers. Then, data is carefully processed by cleansing noisy data, and applying feature engineering techniques, such as feature encoding, e.g. One-Hot Encoding; feature selection, the process of selecting a subset of features based on a criterion, e.g. Entropy or linear correlation between variables; and feature extraction, the process of reducing the dataset dimensionality by creating new features that summarize most of the information contained in the original set of features, e.g. Principal Component Analysis (PCA).

2.2.1. Classification Algorithms: Black Box vs. White Box

After the data pre-processing tasks, specific classification algorithms must be selected. This step is key and often based on intuition, a prior hypothesis on the data, successful past use-cases for similar problems, or even time and/or computing limitations to implement in a reasonable time. Having a set of algorithms, it is necessary to train them on the available training data, in which the classifier will find the patterns within a set of independent features, or predictor features, that lead to a certain label.

From a practical point of view, we can frame classification models, and ML models in general, into White Box or Black Box models (Loyola-Gonzalez, 2019). White Box models are explainable and interpretable AI models that can easily be compared and understood by human experts, e.g. Regression and simple Tree-based methods where the importance and weight of each predictive feature are traceable, and it is clear how features affect the predicted labels. On the other hand, Black Box models are high-performance models that generally provide more accurate results than the ones obtained with White Box models. However, it is quite challenging to explain how predictive variables affect the predicted target labels. Black Box algorithms include ensemble methods, such as Support Vector Machines and Neural Networks, e.g. Long Short-Term Memory (LSTM) Neural Networks and Convolutional Neural Networks (CNN). In this report, we will dive deeper into four specific classifiers, two White Box models, Logistic Regression and Logistic Regression, and two Black Box models, Random Forests and XGBoost Classifiers.

2.2.1.1. Logistic Regression

The logistic regression is one of the simplest and easily understandable classification algorithms available which, despite its simplicity, proves to be very powerful. The traditional algorithm derives from linear regression, as follows:

$$E[y] = \beta_0 + \beta_1 X_1 + \cdots + \beta_i X_i \quad (1)$$

Where, *E[y]* is the expected value of the dependent variable *Y* we intend to predict, $\beta_0, \ldots, \beta_p$ are linear coefficients, and $X_1, \ldots, X_i$ is the values of the *p* predictor variables for each data point. The problem, however, is that linear regression is far from ideal for classification problems, due to strict ordinary least squares (OLS) assumptions (James et al., 2000): linearity, normality, and continuity for OLS regression and multivariate normality with equal variances and covariances for discriminant analysis. Simultaneously, this equation will regress on continuous values, whereas in classification problems the target class is binary or categorical. The logistic model can be denoted as:

$$logit(Y) = log\left(\frac{p(X)}{1 - p(X)}\right) = \beta_0 + \beta_1 X_1 + \cdots + \beta_i X_i \quad (2)$$

Where *P(X)* is the probability of *X* belonging to a determined target class:

$$p(X) = \frac{e^{\beta_0 + \beta_1 X_1 + \cdots + \beta_i X_i}}{1 + e^{\beta_0 + \beta_1 X_1 + \cdots + \beta_i X_i}} \quad (3)$$

Logistic models use Sigmoid functions, thus, when visualizing binary problems, where the target labels for all data points are either from Class 1 or Class 2, it is possible to use a scatter plot to observe how the *Y*-axis ranges from zero to one (see Figure 5).

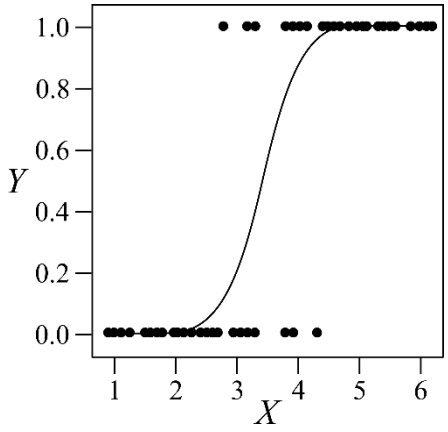


Figure 6. Logistic Regression curved fitting example

In machine learning pipelines, Logistic Regression models benefit by not requiring a high computational effort and, as a White Box model, its results are highly interpretable. Furthermore, it is not required to scale and/or standardize the predictive independent variables to use this model. However, it will not produce good predictions if the predictive variables are correlated amongst each other or if they are not correlated with the dependent variable we wish to predict. Also, it is still a simple derivation of a linear model and a more complex algorithm will likely surpass the predictions of a Logistic Regression, thus, I would advise using this model as a baseline for benchmarking other algorithms.

2.2.1.2. Decision Tree Classifier

Another White Box model discussed in this report: Decision Tree Classifiers. These tree-based classify instances by sorting them based on feature values. A node in a decision tree denotes a predictive feature and each branch represents a value that this feature can assume (see Figure 6). These models are non-parametric, that is, they do not make strong assumptions on the distribution of the data, which presents some benefits, especially when a lot of data is available and we do not have any prior knowledge of it, allowing more leverage when choosing the right features (Ligeza, 1995). Non-parametric algorithms are also flexible, powerful and can achieve good results, however, they require high amounts of data, the training can be slow and there is also the risk of overfitting.

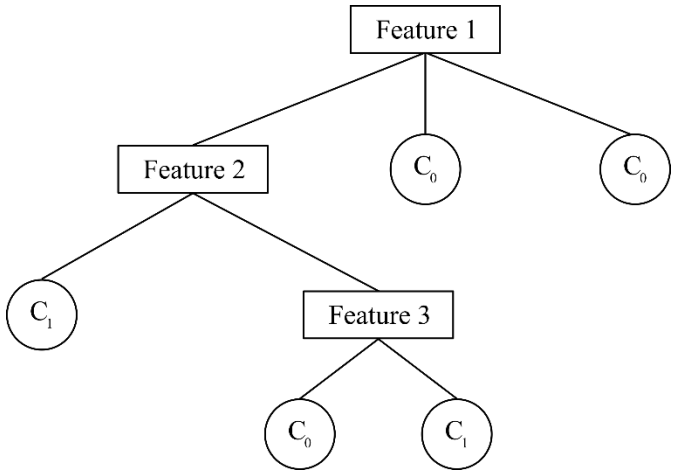


Figure 7. Decision Tree classifier example of how predictive features are split into branches that either lead to other splitting nodes or the classification

There are several decision tree algorithms, such as ID3 (Quinlan, 1986), C4.5 (Quinlan, 1993), and CART (Gordon et al., 1984). Trees are greedy and recursive algorithms, meaning that, for each iteration, the algorithm will try to find an appropriate splitting partition of the dataset into smaller subsets based on a predictive feature. This process will then repeat itself until all branches lead to only an exclusive target class, or a stopping criterion is satisfied. Since the referred tree classifiers are top-down algorithms, the root node will represent the feature that better discriminates the target class, the second node the subsequent feature that better discriminates the target class, and so on. This discrimination ability is measured by using the Entropy and Information Gain concepts. Entropy measures the homogeneity, or randomness, of a sample, and its equation can be written as:

$$Entropy(S) = -\sum_{i=1}^{C} -p_i log_2(p_i) \quad (4)$$

Where P_i is the probability of an element of class I in the subset, and C the number of possible classes. For binary problems, the entropy can assume a value from 0 to 1, where values close to one mean it is harder to discriminate between target classes, and on the other hand, a completely homogeneous subset will have an entropy of zero. If the distribution of classes is equal in a subset, then the entropy will be equal to one. Another metric on how to split data and finding the first nodes is by using Information Gain, which measures the ability for a predictive feature to discriminate between classes, using entropy as the impurity measure:

$$\text{Information } Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} . Entropy(S_v) \quad (5)$$

Where *S* is the entropy existing in the dataset if the *A* predictive feature was the only one existing, and S_v the entropy in the subset of each distinct value in *A*. Ultimately, Information Gain measures the decrease in entropy, computing the entropy before and after splitting the dataset based on a given attribute.

Decision Trees, as a classification tool, have many advantages, they are easily understandable and self-explanatory since they belong to the White Box models category and can be translated to a set of if-then-else rules. Moreover, Decision Trees are capable of handling numerical and categorical inputs and are robust to datasets containing errors or missing data. Furthermore, they are also nonparametric, as stated earlier, meaning they have no assumptions about the data distribution (Rokach & Maimon, 2006). On the other hand, they have some disadvantages as well. Tree classifiers require an exclusively discrete target and are likely to provide weak results if no relevant attributes exist. Also, it is important to remember they are greedy algorithms which, depending on the dataset size, can lead to a high computational effort.

2.2.1.3. Random Forest Classifier

Random Forests, as the name suggests, elevate the concept of Decision Trees, as a combination of tree predictors that operate as an ensemble (Breiman, 2001). Each tree receives a random subset of features to classify, with equal target distribution, which in the end, the predictions are voted on and the class with most votes wins (see Figure 7).

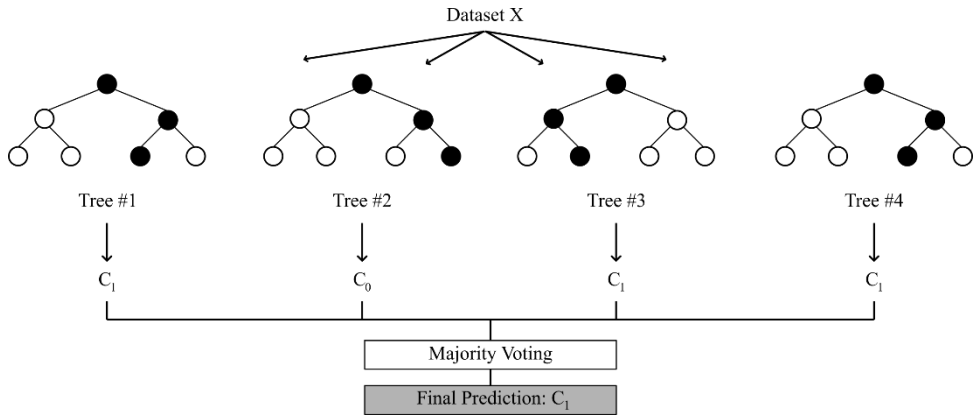


Figure 8. Random Forest classifier example on how independent trees work together to vote on a prediction

As an ensemble method, the overall idea is that many relatively uncorrelated trees working together will outperform any of the predictions provided by a single tree. The reason why this algorithm is so good at overall problems is precise because the trees are protected from the individual errors each one makes, otherwise, they would tend to provide the same output. Since each Individual Tree is sensitive to the data it is trained on, Random Forests allow each one to randomly sample data and features, resulting in different trees, also known as bagging.

As a Black Box model, Random Forests suffer from a lack of interpretability and little control on the model's operability, since they can be very complex. Also, they require more training time and computational effort when compared to Decision Trees and Logistic Regression, for example. On the other hand, Random Forests can solve both classification and regression problems, they don't require predictive features to be previously scaled, and they are robust to noisy data, e.g. missing values and outliers. Furthermore, they are very stable and powerful algorithms that tend to provide good results in a wide range of problems, since they can handle high dimensional datasets well and benefit from ensemble and bagging techniques that reduce overfitting and improve accuracy.

2.2.1.4. XGBoost Classifier

The final discussed classifier in this report, also a member of the Black Box models family, is called extreme gradient boosting classifier, or as it is commonly denoted, XGBoost. Similar to the Random Forest, XGBoost is also a decision-tree-based ensemble algorithm, a highly effective and widely used machine learning method (Chen & Guestrin, 2016). The reason why XGBoost performs so well in overall problems is that it derives from the boosting concept, which aims to create a strong classifier based on weaker classifiers. By adding models on top of each other, the errors of the previous model are corrected by the following predictor (Schapire, 2003). Then, as the name suggests, the gradient aspect of the algorithm looks to minimize the residual loss of adding the last prediction by fitting new residuals to the appended model, instead of just assigning different weights in each iteration.

Ultimately, XGBoost classifiers are considered one of the best algorithms that yield good results in a wide range of problems, especially in small to medium-sized datasets, when compared to Neural Networks, for example. Furthermore, they are fast and allow for the parallelization of trees, they are efficient when handling noisy data, and they use regularization that reduces the risk of overfitting. However, since XGBoost classifiers are Black Box models, the prediction process interpretability is highly difficult. For example, when accessing the feature importance of a trained model, it is possible to use the Weight, which represents the number of times a feature is used to split the data across all trees, the Cover, which represents the number of times a feature is used to split the data across all trees weighted by the number of training data points that go through those splits, or the Gain, which represents the average training loss reduction gained when using a feature for splitting. Any of these is likely to provide different importance ranks to each predictor feature, which leads to a lack of interpretability on the importance and effect of each feature. Nevertheless, a novelty solution called SHAP[1] (Lundberg & Lee, 2017) was proposed to tackle this problem. It is based on a game-theoretic approach that assigns an importance value for each feature value for a determined prediction and allows for a visualization-based method for interpreting results from Black Box models.

2.2.2. Performance Metrics for Supervised Classification Problems

Having a set of models to use, it is critical to identify which metrics are more suited towards a classification problem, to evaluate the optimal classifier's performance. Commonly, Accuracy is used to discriminate classifiers, however, it can lead to major bias, since it only evaluates how many data points the classifier got right, from all existing data points, not taking into account class distribution (M & M.N, 2015). During the training phase, evaluation metrics are used to learn and compare different classifiers, to evaluate which can discriminate the target labels the best. Then, in the testing phase, they evaluate the classifier's performance, or effectiveness, on unseen data. For binary classification problems, as is the case of the cases tackled in this report, most metrics derive and can be summarized by using a confusion matrix (see Table 1).

	Predicted Negative Class (N')	**Predicted Positive Class (P')**
Actual Negative Class (N)	True Negatives (TN)	False Positives (FP)
Actual Positive Class (P)	False Negatives (FN)	True Positives (TP)

Table 1. Confusion matrix for binary classification problems

[1] https://github.com/slundberg/shap

Since the target is binary, its actual label is commonly translated to *Negatives* or *Positives*, whereas the corrected classified instances are denoted as *True Negatives (TN)* and *True Positives (TP)*. Moreover, data points whose label is incorrectly classified can be either *False Negatives (FP)* or *False Positives (FP)*. As stated before, in cases where the target class is imbalanced, the accuracy can lead to false conclusions on the model's performance: In the extreme case of a dummy classifier, where all predicted labels are of class C_0, predicting no labels of class C_1, the classifier's accuracy will be equal to the number of data points of class C_0, which in the presence of an imbalanced dataset leads to high accuracy, yet a useless model.

When choosing a classifier or evaluating performance we should, however, look at other existing metrics for classification problems (see Table 2). Metrics such as Recall, or Precision provide a better overview of the classifier's prediction performance. Furthermore, when the implications of failing to predict a certain class are critical, domain-knowledge experts can identify if having False Positives is, or not, more critical than having False Negatives and thus, greater weight can be given to either Precision or Recall. If not, F1-Score is a very robust metric that calculates the harmonic mean between these previous two metrics and facilitates the quest of choosing algorithms or evaluate the final classification predictions on unseen data.

Metric	Formula	Description
Accuracy (acc)	$\frac{TP + TN}{TP + FP + TN + FN}$	The ratio of correct predictions by all predictions made.
Error Rate (err)	$\frac{FP + FN}{TP + FP + TN + FN}$	The ratio of incorrect predictions by all predictions made.
Precision (p)	$\frac{TP}{TP + FP}$	The ratio of correct Positive predictions by all Positively predicted classes.
Recall (r)	$\frac{TP}{TP + FN}$	The ratio of correct Positive predictions by all True positive classes.
F1-Score (F1)	$\frac{2 \times p \times r}{p + r}$	The harmonic mean between Precision and Recall.
Sensitivity (sn)	$\frac{TP}{TP + FN}$	The ratio of correct Positive predictions by all True Positive classes.
Specificity (sp)	$\frac{TN}{TN + FP}$	The ratio of correct Negative predictions by all True Negative classes.
Averaged Accuracy	$\frac{\sum_{i=1}^{c} \frac{TP_i + TN_i}{TP_i + FP_i + TN_i + FN_i}}{c}$	The average effectiveness of all classes.
Averaged Error Rate	$\frac{\sum_{i=1}^{c} \frac{FP_i + FN_i}{TP_i + FP_i + TN_i + FN_i}}{c}$	The average error rate of all classes.
Averaged Precision	$\frac{\sum_{i=1}^{c} \frac{TP_i}{TP_i + FP_i}}{c}$	Average class-specific precision.
Averaged Recall	$\frac{\sum_{i=1}^{c} \frac{TP_i}{TP_i + FN_i}}{c}$	Average class-specific recall.
Averaged F1-Score	$\frac{2 \times p_m \times r_m}{p_m + r_m}$	Average class-specific F1-Score.

Note: Variable c denotes the possible binary classes in the dataset; TP_i the True Positives for C_i; TN_i the True Negatives for C_i; FP_i the False Positives for C_i; FN_i the False Negatives for C_i; and m the macro-averaging.

Table 2. Classification metrics to evaluate the performance

It is important to notice that the overviewed metrics are only theoretical, thus the classifier will benefit and yield better results if business or domain metrics are developed. For example, in a predictive maintenance problem, the total cost of failure and/or replacing a component should be quantified and compared to the cost of a maintenance check that would avoid the subsequent implications. This way, it would be possible to build a metric that can be used in the classifier training and evaluation to account for real costs.

After accessing the performance of the model, likely, the results are not desirable in a first iteration, so it is necessary to go back to one or more of the classification machine learning process and fine-tune several parameters or rules. The final product should be a pipeline that is ready to receive unseen data, apply all the transformation steps with the weights and parameters used in the training set and use the classifier to predict the labels of these new data points.

2.3. OFGEM

Risk is the probability of an expected event outcome to differ from the actual event. It is present in everyday activities and, despite us, as individuals, do not often evaluate risk consciously, companies and organizations have the highest interest in evaluating risk impact and how to minimize impacts. By identifying risk, it is possible to analyze it, evaluate it, and ultimately, mitigate it. To help organizations with such tasks, the International Standards Organization has produced ISO 31000:2009 *Risk Management – Principles and guidelines*, which includes definitions, principles, and guidelines associated with risk management. Furthermore, BS EN 60912:2006 *Analysis Techniques for System Reliability* provides guidance on analysis techniques for system reliability. In this report, we will be focusing on the OFGEM (Office of Gas and Electricity Markets) methodology for the Electricity sector, which is based upon the content from ISO 55001, ISO 31000, and BS EN 60812.

Expressing risk as a combination of the likelihood of an event and its consequences, both factors can be measured qualitatively or quantitatively and must be, for analytics purposes, described mathematically as a probability. This combination is often represented in a risk matrix that crosses the likelihood and consequence of a certain risk in a 2-dimension plan, that can be expressed as:

$$Risk = Likelihood \times Consequence \tag{6}$$

In this case, the risk requires the Likelihood and Consequence factors to be quantitative. This equation can also be expressed in the Energy and Utilities sector, to estimate the risk associated with each asset. This can be translated as the sum of the expected values of the risk consequences associated with a certain asset and the probability of Failure:

$$AR = \sum_{j=1}^{n} PoF_j \times CoF_j \tag{7}$$

Where *AR* represents the Asset Risk, PoF_j the Proof of Failure *j* occurring during a given time, CoF_j the monetized Consequence of Failure *j*, and *n* the number of failures associated with the asset. In the scope of the OFGEM formula, the described theoretical framework applies in the following assets: Circuit Breakers, Transformers, Reactors, Underground Cables, and Overhead Lines (Conductor, Fittings, and SPT and SHE-T Towers). Adding the risk associated with each asset, it is possible to estimate the Network Risk, a collective measure of all assets in question:

$$NR = \sum_{k=1}^{n} AR_k \tag{8}$$

Where *NR* is the Network Risk, AR_k the asset risk associated with asset *k*, and *n* the number of assets on the network.

The OFGEM methodology starts from the premise that the health of a certain asset evolves along an increasing exponential curve: The higher the health value, the worse that asset's health. In Figure 8, we can observe how this curve is influenced by three segments: The initial Health Score, the Current Health Score, and the Long-Term Health Score.

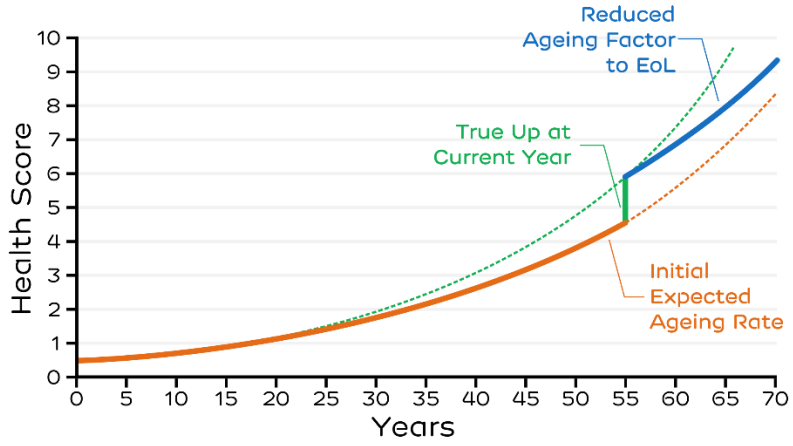


Figure 9. OFGEM Exponential Curve: Expected Asset Health Index considering the aging rate factor

2.3.1. Initial Health Score (Initial End of Life).

Initially, the OFGEM curve is determined by an initially expected aging rate, which depends on two other factors, the Expected Life (L_E) and the Initial Ageing Rate ($\beta 1$). The asset expected life can be decomposed on three main factors: Normal Expected Life, Location Factor, and Duty Factor. The Normal Expected Life (L_E) represents the expected time, in years, a certain asset is expected to last until such deterioration leads to functional failures. This factor considers the original specification and manufacturer data (see Equation 9) where L_A represents the Expected Average Life, the F_{LSE} the LSE Factor, and F_{DY} the Duty Factor.

$$L_E = \frac{L_A}{F_{LSE} \times F_{DY}} \tag{9}$$

Then, the Location Factor (*LSE)* represents the wear factor that the asset will suffer as a result of its location. This is a combination of 3 segments, which are the Distance to coast (km), meaning the closer to the coast, the higher the wear, the Altitude (m), where the higher the asset is located, the higher the wear, and lastly the Corrosion (rating 1 to 5), where the higher the corrosion rate, the higher the wear.

$$Location\ Factor = Max(Distance\ to\ Coast\ Factor, Altitude\ Factor, Corrosion\ Zone\ Factor) \tag{10}$$

Having determined this, it is possible to calculate the *LSE* factor, which stands for the Location, Situation and Environment factors, considering, besides the location, if an asset is situated indoors or outdoors and the severity of the local environment. The minimum location factor is constant.

$$LSE\ Factor = \left(\begin{matrix}((Location\ Factor - Minimum\ Location\ Factor) \times Situation\ Factor) + \\ Minimum\ Location\ Factor\end{matrix}\right) \times Environment\ Factor \quad (11)$$

Lastly, the Duty factor *(F_{DY})* is specific to each different asset, representing the usage rate of an asset: The higher the usage, the higher the asset's wear. For Circuit Breakers, this factor considers the presence of feeder protection (*Prot)*; the presence of Auto-Reclose *(R_A)*; Operational experience in the form of a *high duty* exception report *(D_H);* the fault level compared to fault rating *(D_{FAULT})*; and the latest record of the total number of Fault clearances undertaken by the circuit breaker *(D_{CLEAR}).* The Duty Factor changes depending on the circuit breaker: SHE-T (see Equation 12) or SPT (see Equation 13).

$$F_{DY} = Max(Prot, R_A, D_H) \quad (12)$$

$$F_{DY} = D_{Fault} \times D_{Clear} \quad (13)$$

In the case of Transformers, this factor considers the maximum operating temperature recorded against each transformer *(T_{max})*, the maximum demand placed upon the transformer as a percentage of its stated rating *(D_{max})***,** the average demand placed upon the transformer as a percentage of its stated rating *(D_{ave})*, and the severity of frequency of through faults *(T_F)*. Again, the Duty Factor will depend on the Transformer: SHE-T (see Equation 14) or SPT (see Equation 15).

$$F_{DY} = Max(T_{max}, D_{max}) \times T_F \quad (14)$$

$$F_{DY} = Max(D_{max}, D_{ave}) \quad (15)$$

To complete the Expected Initial Ageing Rate, it is required to estimate the Initial Ageing Rate, which is based upon the End of Life modifier *(EoL)*.

$$Initial\ Ageing\ Rate\ (\beta_i) = \frac{\ln\left(\frac{EoL_{MAL}}{EoL_{NEW}}\right)}{L_E} \quad (16)$$

Where EoL_{MAL} represents the *EoL* modifier of the asset when it reaches its expected life (set to 5.5), the EoL_{New} the *EoL* modifier of a new asset (typically set to 0.5), and the L_E stands for the Expected Asset Life. Thus, we have reunited the conditions to assess the Initial Health Score, or the Initial End of Life modifier *(EoL)*, which is modeled through the Initial Ageing Rate and a projection on the asset Expected Life. This will represent a theoretical projection where it places the asset's health in the aging curve. This measure can range from 0.5, a new asset, to 5.5, when the first signs of significant deterioration are expected to appear.

$$Initial\ End\ of\ Life\ indicator\ (EoL_{1,i}) = EoL_{NEW} \times e^{(\beta_{1,i} \times Age_i)} \quad (17)$$

Where $EoL_{1,i}$ represents the initial indicator of asset *i*, the EoL_{NEW} the *EoL* modifier of a new asset (commonly set to 0.5), and $\beta_{1,i}$ the initial aging rate of asset *i*.

2.3.2. Current Health Score (Intermediate End of Life)

Having established how to estimate the initial health index of an asset or initial end of life indicator *(EoL_2)*, it is possible to assess the current health index of each asset or intermediate end of life modifier. This estimation now considers the asset's observed conditions, inspection surveys, maintenance test results, operator's experience of each asset, and reliability inputs.

$$EoL_2 = EoL_1 \times FV_1 \tag{18}$$

Combining said information and holding individual weights for each one of the previously denoted factors, the intermediate end of life modifier *(EoL_2)* is achievable by multiplying the initial end of life modifier *(EoL_1)* by a conditional factor value *(FV_1).*

2.3.3. Long-term Health Score (Forecasting End of Life)

To estimate a long-term health index score, we first determine the End of Life modifier in future years as:

$$EoL_{Y(n)} = EoL_{Y(0)} e^{b\Delta T} \tag{19}$$

Where ΔT represents the time between years 0 and *n.* Using the expected life of the asset as ΔT, and the max and min EoL_s as $EoL_{(Yn)}$ and $EoL_{(Y0)}$ respectively, it is also possible to calculate all other variables, therefore allowing to estimate variable *b*. Similar to the initial and current health score, it is necessary to capture the effect of the aging rate in the *EoL*, which we will now express as:

$$\beta_{Final,i} = Max \left[\frac{\ln\left(\frac{EoL_{Y0}}{EoL_{NEW}}\right)}{Age_i}, \beta_{1,i} \times \beta_{Ratio} \right] \tag{20}$$

Where β_{Ratio} is the max ratio between the final aging rate and the initial aging rate. Therefore, we have all the necessary factors and conditions to estimate the forecasting the End of Life of an asset:

$$EoL_{YN,1} = Max \left(EoL_{Y0,i} \times e^{\left(\frac{\beta_{Final,i} \times (t_{YN} - t_{Y0})}{F_{age,i}}\right)}, EoL_{YN,max} \right) \tag{21}$$

Where $EoL_{YN,i}$ represents the *EoL* modifier of asset *i* in future year *Yn*, $\beta_{Final,i}$ the final aging rate of asset *i*, $F_{Age,i}$ the aging reduction factor for asset *i*,$(t_{YN} - t_{Y0})$ the number of years over which the asset moves from $EoL_{Y0,i}$ *to* $EoL_{YN,i}$, and $EoL_{YN,max}$ the maximum allowable value for the future indicator, typically set to 15 years in the future.

2.3.4. Forecasting Proof of Failure

Besides an evaluation of the health status of an asset, OFGEM also allows for an estimation of the likelihood of an asset to fail, in the short, and long-term. To do so, it is necessary to translate the *EoL* values into a probability, which depends on the specific asset type. This approach is not only influenced by the aging process but from external events as well, such as environmental conditions and poor installation. The distribution curve that represents the probability of failure can be expressed using two functions: An exponential function *(x^2)* which provides a rapid rise in the probability of failure as the *EoL* modifier value increases, e.g. as deterioration approaches the point of failure, and a cubic expression *(x^3),* e.g. the first three terms of a Taylor series for an exponential function.

The use of an exponential function provides a predicted failure rate that generally falls in the range of simulated predictions around 15 years ahead (Black et al., 2005). After this period, the predicted failure rates start to overestimate. Therefore, a better solution is a hybrid form of the cubic function (see Equation 22). This allows for the probability of failure to be constant for low *EoL* modifier values, e.g. an asset in excellent condition, before increasing rapidly as the *EoL* Modifier increases, e.g. as the asset begins to degrade significantly. The cubic function is responsible for modeling the asset's behavior more

closely than the exponential. To determine at which point the probability of failure is derived using the cubic expression, a threshold, or calibration value, called EoL_{Lim} is used. Below the EoL_{Lim} threshold, the probability of failure is set at a constant value (see Equation 22); above the EoL_{Lim} the cubic function applies (see Equation 23).

$$PoF = k \times \left(1 + (EoL \times C) + \frac{(EoL \times C)^2}{2!} + \frac{(EoL \times C)^3}{3!}\right), EoL > EoL_{Lim} \quad (22)$$

And

$$PoF = k \times \left(1 + (EoL_{Lim} \times C) + \frac{(EoL_{Lim} \times C)^2}{2!} + \frac{(EoL_{Lim} \times C)^3}{3!}\right), EoL \leq EoL_{Lim} \quad (23)$$

Where *PoF* is the probability of failure, *EoL* represents the end of life modifiers, *K* and *C* are pre-determined constants, and EoL_{Lim} is the *EoL* modifier limit below which the probability of failure is constant. Regarding the constants, *C* is responsible for fixing the relative values of the *PoF* for different modifiers, acting as the slope of the curve, while *K* determines the absolute value. These constants must be calibrated for each different asset. Along with this report, we will discuss a practical implementation of the OFGEM formula and its adaptation to the EDP D assets.

3. FRAMEWORK AND DATA ARCHITECTURE

In the practical examples discussed further in this report, due to the processing and ingestion of high amounts of data required, Big Data-oriented solutions were also a must. Having said this, all tools and technologies used to fall under the Microsoft Azure Cloud Ecosystem[2], to increase the synergies between services and deploying at-scale solutions. To better understand what was used, I divided the technologies into 4 major components: Big Data Storing & Analytics, Relational Database Storing, Systems Integration and Deployment, and Reporting and Dashboards. In all referred projects, the same data architecture and sources were used (see Figure 10).

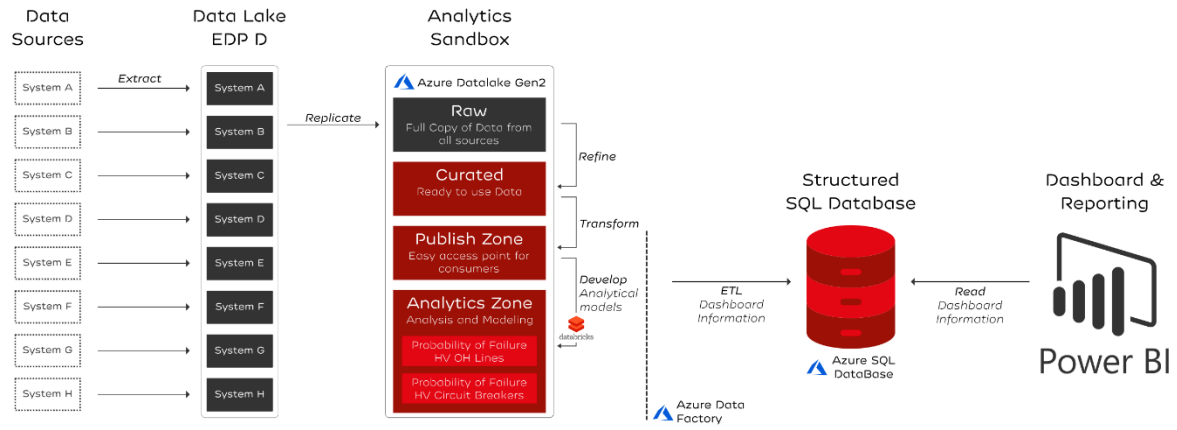


Figure 10. Data flow for EDP D projects

Big Data Storing & Analytics:

- **Microsoft Azure Data Lake** - Azure Data Lake is a centralized cloud repository built on YARN that holds structured and unstructured data, highly accessible, scalable, and quick to update that can be used for analytical tasks.
- **Microsoft Azure Blob Storage** - Azure Blob Storage is Microsoft's cloud-focused storage solution. It is optimized for storing huge amounts of unstructured data, such as text or images, and allows users to access information via HTTP/HTTPS, from anywhere, at any desired moment. Objects stored in Blob storage are also available through different languages, including .NET, Java, Node.Js, Python, Go, PHP, and Ruby.
- **Microsoft Azure Storage Explorer** - Azure Storage Explorer is a useful tool to manage Azure cloud Blob storage resources, anywhere, anytime. It includes features such as create and/or delete containers, upload files, create and/or delete folders within a container, amongst others.
- **Microsoft Azure Databricks** - Azure Databricks is an Apache Spark-based unified platform for data and AI that allows for large-scale data processing inside the Microsoft Azure Cloud, useful for analytics tasks, allowing an interactive workspace with code in several programming languages and collaborative work between different users and developers, as well as the deployment of Machine Learning algorithms into production. In a Big Data environment, the data is ingested into Databricks using Azure Data Factory in batches. The analytics platform inside the Apache Spark ecosystem contains five key components:

[2] https://azure.microsoft.com/pt-pt/

- **Spark SQL and DataFrames** – Spark module for structured data. DataFrames in Spark is the distributed equivalent of relational database tables or Python DataFrames.
- **Streaming** – Real-time data processing and analysis for analytics purposes. It integrates with HDFS, Flume, and Kafka.
- **MLlib** – Machine Learning oriented library that offers advanced analytic solutions, such as supervised and unsupervised algorithms, utilities, and optimization primitives.
- **GraphX** – Graphs computation that yields cognitive analytics to data exploration.
- **Spark Core API** – Includes development in R, SQL, Python, Scala, and Java.

Database Storing:

- **Microsoft Azure SQL Database -** Azure SQL Database is a component inside Azure SQL Family, offering scalable relational database services for the cloud. It is always based on the latest version of Microsoft SQL Server. With it, it is possible to create highly available and high-performance data storage layers for applications and solutions within Azure.
- **Microsoft SQL Server Management Studio (SSMS) -** SQL Server Management Studio is an integrated Azure environment for managing any SQL infrastructure. It allows for data governance capabilities such as access, configure, manage, administer, and develop all components of Azure SQL family services, providing a single comprehensive utility that combines several graphical tools with script editors, serving developers and users of all skill levels.

Systems Integration and Deployment:

- **Microsoft Azure Data Factory -** Azure Data Factory is a service that allows integrating hybrid data at-scale. It grants access to on-premises data in SQL Server and Cloud data both in Azure Data Lake and Azure Blob Storage and in Azure SQL Databases. With Data Factory, it is possible to build ETL and ELT pipelines easily and code-free inside an interactive visual environment.

Reporting and Dashboards:

- **Microsoft Azure Power BI -** In Power BI it is possible to create powerful, visually immersive, and interactive visualizations by connecting different services and/or data sources from local Excel files to cloud-based stored data. The built dashboards allow for actionable insights and support decision-making, easily accessed, and used by a broad range of final users.

4. QUALITY ASSURANCE GUIDELINES

As a Data Scientist at EDP D, one of my main tasks is to audit projects that aim to solve business problems employing analytics, validating methodology; documentation; models; and promoting the highest possible quality. This is done by assuming and establishing Quality Assurance (QA) activities, which are fundamental. Furthermore, given the high number of products at EDP D, most projects are only possible by outsourcing specialized consulting teams that focus only on a specific problem. Therefore, a significant part of my role is to be accountable, to some extent, investing time in oversight analytics techniques and steps implemented, and recommend solutions that aim towards a long-term and reliable final product.

During my experience as a Data Scientist and projects in which I assumed QA roles, I intend to summarize the lessons learned; main pitfalls and problems encountered; and the systemized advised solutions that are being used as a baseline for similar future projects.

4.1. DOCUMENTATION: HANDOVER AND CODE

During the making of any project, the active developers and contributors can identify where every step and/or decision is being made and why it is being made. However, this may not be so transparent for someone who did not follow the working developments of this work, or for future contributors. As it is often done in Academia, this problem can be solved with a document that states the entire problem and defines every step of the delivered product. Ultimately, it represents the best knowledge source for anyone who seeks more information on the subject. Thus, a handover document structure and coding documentation rules should become the best practices to be adopted.

For example, after the conclusion of any project, improvements along the time are expected to be done. However, if the original developing team is not present, which happens mostly with outsourced projects, and there is a lack of information and code documentation, changes and understanding of what was done will be remarkably hard, delaying project schedules. Despite such documentation being a short-term effort by the developers, it may result in significant time-consuming tasks of understanding the work done.

In this situation, my recommendation was inspired by the CRISP-DM methodology (Shearer et al., 2000) and OSEMN (Mason, 2010) framework steps, by creating two main knowledge transfer sources that can support anyone who needs them: and Handover document and Code documentation guidelines. Regarding the Handover document, this should contain the main and irreplaceable topics that fit the knowledge transfer needs. I recommended that it should start by clearly stating the **Problem and Objectives** of the problem at hand, from a business and/or business perspective, as well as the proposed product. Then, to set who are the key-experts of the project, the document should define the developing **Team and the Project Plan**, that will support future similar tasks planning. The identification of all data sources and where they are stored and/or processed (Hardware and Software) is also a must, framing the **Data Architecture** of the project and defining how the sources and different tools are connected. This information should also be supported by a visual representation of the data flow. To conclude the introductory part of the document, the **Business Problem Relevant Definitions** should be presented, in other words, a glossary of all domain-knowledge terms and/or explanation and relevant definitions of domain-knowledge concepts. Having said this, a **Methodology** segment should follow, where it defines the overview of the project's methodology, specifying all steps and techniques used to achieve the final product. It should allow the reader to assess the validity and reliability of the decisions made. Ultimately, it must describe how the data was collected and how it was processed/analyzed. Then, the **Exploratory**

Data Analysis phase should summarize the main aspects of the data, allowing to make early conclusions, describe trends, associations, and/or patterns in the data, as well as the identification of outliers, missing values, and other noisy data. It is of high importance that this chapter's conclusions are supported by visualizations. After the exploratory phase, a detailed description of all transformations applied to data follows, stating the **Data Preparation** steps, including and justifying the reasoning behind every significant decision, such as imputing missing values, dealing with outliers, feature engineering, feature selection, and extraction, amongst others. After this phase, I suggested the **Modelling** chapter, in which the models and algorithms used are described, summarizing how they work, and a benchmark of the tested models, clearly allowing a comparison between parameters and results, both in training, validation, and testing data. Closing the developments, a presentation of the obtained **results** is fundamental, showcasing the final model, the parameters, and the corresponding metrics that support such conclusions. In the case of a forecasting or classification problem, the final metrics should reflect the model's performance on testing data. It is also important that those theoretical metrics are translated into business gains and costs. I also suggested the inclusion of **Improvements and Follow-up Work** to identify the main obstacles encountered during the project, clearly summarizing what should be done differently to improve the final product. **Additional Support** is also relevant to support the reader's understanding of the problem, such as detailed model benchmarks or detailed feature mappings that allow to trace back the entire source path. Lastly, a **User-Manual** is a good knowledge source on how to use the delivered product, containing information on, for example, what each notebook does or how to change certain parameters.

```python
def sample_function(a,b):
    '''
    This function multiplies the received arguments

    Inputs
    ----------
    - a (float): Argument1 to multiply
    - b (float): Argument2 to multiply

    Outputs
    ----------
    - multiply (float): The product result of a x b
    '''

    multiply = a * b

    return multiply
```

Figure 11. Documentation of code - Function example

Furthermore, regarding the code documentation, all relevant steps of code should be followed by a small description of their nature. This is a harder aspect to assure, given that each developer has a coding behavior and relative documentation can be subjective to some. However, I recommended the use of Markdown titles to separate blocks and the use of functions to optimize the code and avoid repeating tasks. Functions are not only reusable, meaning that they can be used repeatedly, but they are abstract as well since we do not need to know how a function works, only its name, location, inputs, and outputs. However, it is expected that functions may encapsulate complicated logic tasks, thus, achieving a high complexity level. So, to understand its objective, the solution advised was to structure the documentation in three parts: **Description**, a summary of what the function does; the **Inputs**, the arguments of the function; and the **Outputs**, what is returned or transformed by the function.

4.2. DATA VISUALIZATION CAN BE MISLEADING

When presenting the major findings in Data conclusions must be supported by visual methods. By using visual representation, it is easier for the speaker to communicate his conclusion and, for the reader and/or audience, easily understand patterns in data. Furthermore, I am an advocate of the idea that results are only as good as the way they are communicated. The problem is that data visualization can be misleading as well, having a snowball effect, especially if it is done incorrectly during the Exploratory Data Analysis (EDA) phase, where a summarization of data and early conclusions are made to proceed with certain transformations in the data preprocessing phase.

A common example of how visualization can be misleading is the over-beautification of data in presentation support. Usually, to *"sell an idea",* data conclusions are presented using 3D charts, visual templates; illustrations of infographics that, even though more attractive to the eye and easier to communicate, can exaggerate the reality. Common charts can also suffer from incorrect plotting inaccuracies, such as truncated axis, data aggregation (e.g. cumulative graphs), or ignoring conventions where standard visualization practices are violated (e.g. Pie charts that do not account for 100%). To tackle possible misleading conclusions when data is presented using 3D charts, visual templates; illustrations of infographics, the Lie Factor (Tufte, 1988) is a good metric on the relationship between the size of the effect shown in a graphic and the size of effect shown in data:

$$Lie\ Factor = \frac{Size\ of\ the\ effect\ shown\ in\ graphic}{Size\ of\ the\ effect\ in\ Data} \tag{22}$$

Where the size of the effect can be described as the relative proportional change of the second observed value with the first observed value. If data is well represented, the Lie factor accounts for a value between 0.95 and 1.05. This range allows for some minor inaccuracies, caused by plotting, for example, however, a Lie Factor value higher than this range indicates intended distortions.

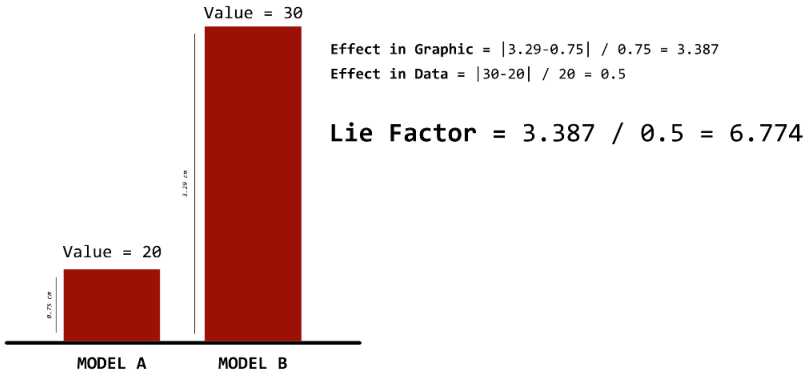


Figure 12. Lie Factor (illustrative example)

In the given example (see Figure 12), a bar chart is presented with 2 models, and their corresponding recorded values. The effect shown in the data is 50%, while the effect shown in the graphic is 338.7%, leading to a Lie Factor of 6.774, clearly showing this data was intentionally distorted by a visual representation. For other misleading conclusions from charts, it is necessary to have an idea of data visualization best practices and it is a part of the QA process to assess if these rules are being complied with. Another recommendation is to change the charts used according to the data they are representing. Listing the most common charts, Line Plots are good when showing time evolutions, trends and how data is connected; Bar Plots to represent categories, whereas the length of each bar is proportional to the value it represents; Pie Charts to express how a whole is divided into parts, however, it is a very specific visualization that should only be used when plotting 5 to 7 categories maximum; Scatter Plots for

representing how numerical data associate with each other; and lastly Box Plots that allow to summarize high quantities of data and to visualize data distributions.

4.3. DATA LEAKAGE BY IGNORING TIME IN TIME-SERIES DATA

As discussed previously on the theoretical framework, cross-validation is an excellent way of reducing the risk of overfitting, by dividing the training set into mutually exclusive and equal-sized subsets of data during *n* iterations. In each iteration, most of the subsets are used to train the classifier, while the remaining subset has a validation role. In the end, the classifier's performance on the data will be the average of the classifier's performance on each iteration. However, this can prove to be problematic when dealing with time-series by risking data leakage problems in the analysis.

Traditional cross-validation for time series data should not be used mainly because of temporal dependencies. When developing predictive analytical models, we want to simulate the "real world" the best way possible, which leads to algorithms being more robust to unseen data. Thus, events that occur chronologically after the events within the data subset that is being used for train, should not be considered (Bergmeir & Benítez, 2012). Using K-Fold Cross Validation, for example, the partition of data into folds is done randomly, not considering this chronological order, e.g. having data from 2010 to 2020, we are introducing data leakage in our analysis by, in a certain iteration of K-Fold Cross Validation, we use data points from 2019 and 2020 to predict data points labels from 2015. In other words, we would introduce a look-ahead bias by predicting the "past" using "future" data, when, we are achieving the exact opposite.

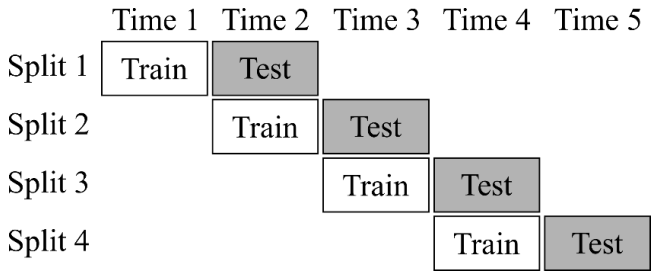


Figure 13. Sliding window cross-validation

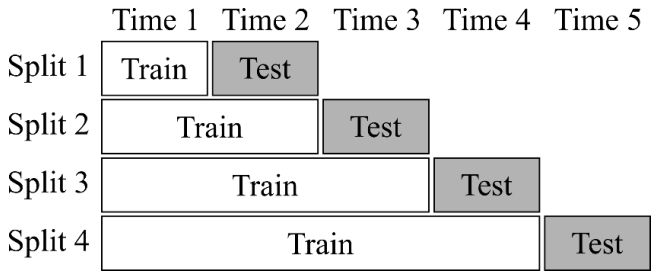


Figure 14. Nested cross-validation

To solve this, it is first important to understand if the problem we are aiming to solve with analytical methods has time dependencies. If it has, variations of the K-Fold cross-validations can be used. Two simple examples are by using sliding windows (see Figure 13), which uses data points from time *t-1* to train and data points from time *t* to validate results, or by using expanding windows (see Figure 14), or nested cross-validation, which use data points from time *t* to validate results as well but also use the cumulative data points from previous time to *t* to train the classifier. (Bergmeir et al., 2018). This is an often-good solution that despite reducing the overfitting risk and solving the time-series problem, provides good insights on how much data from *t* time is needed to correctly predict labels.

4.4. BLINDLY ENCODING CATEGORICAL DATA

Many existing algorithms do not support a categorical Input, for example, text fields in a table. They are, however, ready to process numerical data. Even though some algorithms allow these types of variables, it is a good practice to apply transformations that fit the data in question, which will, most likely, lead to better results. From the many ways to encode categorical data, the most common relies on creating a column for each possible category, or *k-1* columns to avoid multicollinearity issues, a strong dependency between independent predictor features, where *k* is the number of distinct values, and filling that new feature with a 1, if that category applies to a certain row, and 0 otherwise. This technique of creating binary features is called One-Hot-Encoding (see Table 3).

Id	Category
Id_2	A
Id_2	B
Id_3	C

➡

Id	A	B	C
Id_2	1	0	0
Id_2	0	1	0
Id_3	0	0	1

Table 3. One-Hot-Encoding (illustrative example)

Despite all the benefits, encoding categorical variables without any ponderation on how many distinct values exist can lead to numerous new features, therefore, facing the curse of dimensionality (Shultz et al., 2011). For example, if we consider a dataset containing one categorical feature with one thousand possible distinct values, a simple application of the One-Hot-Encoding technique would lead to one thousand new binary features, or a nice hundred and ninety-nine if it was decided to drop one value for multicollinearity issues. Furthermore, contrary to Ordinal Encoding, where categorical values are replaced by rank, the new binary features created by a One-Hot-Encoding transformation will be independent amongst themselves. Thus, if a prior ordinal relationship of greatness between the values of a categorical variable exists, this relationship will be lost after applying this encoding technique.

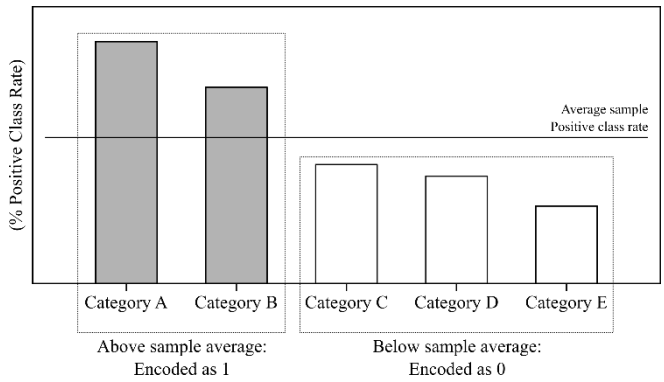


Figure 15. An alternative to One-Hot-Encoding to high dimensional classification datasets

Even though I see One-Hot-Encoding as a technique that generally provides good results, and it is an easy and fast method for dealing with categorical variables, when facing many unique values within a category, this blind application of the technique must be thought carefully. As a solution, in these cases and the scope of a supervised classification problem, I advise creating a single binary feature, instead of *k* new variables. This feature will represent the relationship between the variable categories and the target labels and aims to group categories that lead to a similar class (see Figure 15). From several existing options, this is a simple approach that estimates the average positive target class distribution of the sample, for binary problems, and then estimate the same but for each unique value of the categorical features, as in a frequency table. Categories with an average target class higher than the sample average, are encoded to one, otherwise zero.

4.5. DEMOCRATIZING KNOWLEDGE AND MANAGING EXPECTATIONS

As time and technology progress, companies generate ever larger and more variated data sets, which also comes with the challenge of how to analyze that data, since traditional techniques and tools are falling behind. Data Scientists aim to tackle this problem by acquiring data and transform it into actionable insights and wisdom that support decision making. However, for any project whose objective is to solve a business problem using analytics, a multidisciplinary team is required. However, a common mistake is to delegate all analytical tasks exclusively to the Data Scientists or Data Analysts of a team: In the long run, this is unsustainable. In every problem, the analytics experts should have, at least, some basic knowledge about the business domain, and vice-versa. This leads to a better model of the reality of the problem and, on the other hand, a better interpretation of the results. It is also important to democratize this knowledge with business stakeholders, particularly on how to read the metrics returned by predictive models, e.g. avoid making decisions based on Accuracy. This will help to build a data-driven and data-informed culture within a company.

Furthermore, it is necessary to manage some expectations. Developing classification and prediction models is not an exact science, instead, it resembles more trial-and-error experiments. The time it will take to achieve desirable results depends on the availability and familiarity with data, the familiarity with the problem domain, the availability of all intervening contributors, the available time, and ultimately, the budget. In fact, after the first results, it is expected and recommended to fine-tune parameters or implement new techniques into the analysis. It is also fundamental not to rely too much on the modeling phase and overlook what is behind, since in most cases, problems and improvements start at data identification, extraction, and pre-processing.

4.6. MODEL INTERPRETABILITY VS PREDICTIVE PERFORMANCE

When talking about the universe of existing analytical models, it is easier to understand if we separate them into two classes: White Box models & Black Box models. As described earlier, White Box models are explainable AI and interpretable models that can easily be understood by business-oriented professionals. Such models include algorithms as Linear Regressions, Logistic Regression, and Decision-Trees. In all the mentioned models, it is possible to identify the importance of each variable towards the output, the weight of the variable, and how it affects the result. On the other specter, Black Box models are high-performance algorithms that provide generally better results than the ones obtained with White Box models. However, it is very hard to explain what is happening inside these algorithms, or how a specific predictive feature impacts the outcome. Black Box algorithms include ensemble methods, such as Random Forest and XGBoost, Support Vector Machines, and Neural Networks such as LSTM or CNN.

To tackle this pitfall it is necessary to, in the beginning, and modeling phase of a project, clearly explain the difference between these two types of models and evaluate the trade-off for each one. If the business interpretability of the outcome and understanding of the algorithm's calculations is of high importance towards the project, then a White Box model is recommended, even though it might compromise the performance. On the other hand, if the business is willing to compromise interpretability and trust in the algorithm predictions and evaluation of the data received, then a Black Box method should yield better results. In summary, if the project's stakeholders require to have something "palpable" that they can interpret easily, then you can skip on exploring more advanced Black Box models, it is just necessary that they understand the trade-off between interpretability and performance.

4.7. ONE METRIC (DOES NOT) FIT ALL

When dealing with supervised problems, it is possible to measure the performance of a model by comparing its Output with the observed labels or values. Since this report focuses on Supervised Classification problems, the commonly used metrics are the Confusion Matrix, Accuracy, Precision, Recall, F1-Score, ROC-AUC, and Lift Curve. When evaluating performance, one metric is not above the other, instead, it is necessary to carefully balance these metrics with the problem at hand. In a real problem, it is very difficult to encounter a perfectly balanced classification problem, and referring to some use-cases such as Fraud Detection or Failure Detection, the bias between classes can be as significant as over 90% since normal behaviors are far more common than fraudulent or failures ones. Ultimately, it is very common to have a majority class that can overshadow the minority class. Like data visualization, making conclusions from only one metric can be misleading, as it is commonly done with Accuracy. Furthermore, it is important to highlight these are only theoretical metrics, that can and should be adapted to the problem in question. When developing a model, one should ask *"What has more impact? False Positives or False Negatives?"* and how to translate the results into business metrics, passible of estimating the return over investment (ROI) of the delivered product.

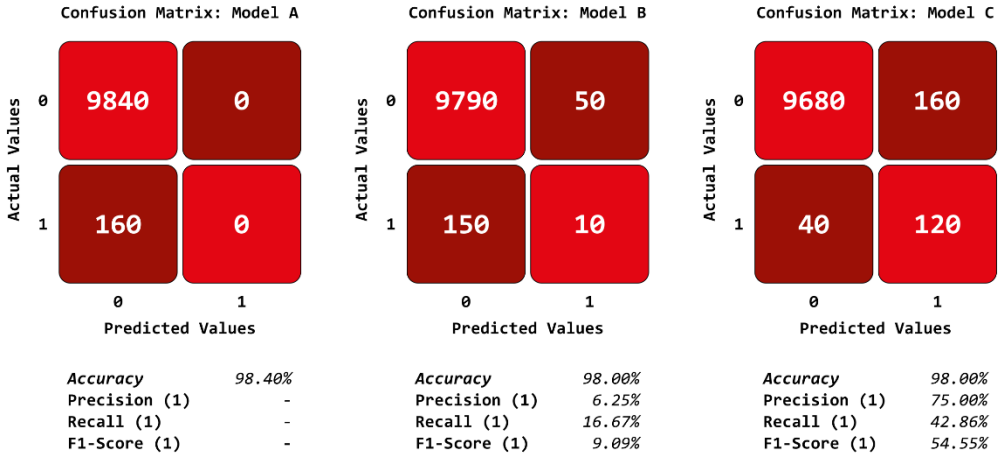


Figure 16. Classifier metrics comparison - How accuracy can be misleading

To illustrate this pitfall (see Figure 16), let's assume we have a supervised binary classification problem, where the target variable can be either 0 or 1. After an analysis of the class distribution, we conclude we are dealing with an Imbalanced problem, whereas from the 10 000 data points, class 0 accounts for 9 849 observations (98.4%) of the data points, while class 1 for the 160 observations (1.6%). To accurately predict this label for unseen data, after the exploration and transformation of data, three models are built and compared using the Confusion Matrix and consequent metrics, Accuracy, Precision, Recall, and F1-Score. As illustrated, it is possible to observe the best Accuracy score belongs to Model A, however, it appears for any input this model receives, it will always output the target 0. These kinds of models are commonly called Dummy Models, and since the proportion of data points of the majority class is 98.4%, the accuracy will naturally have equal value. Without any positive prediction of class 1, it is not possible to estimate Precision, Recall, and F1-Score. Focusing on Models B and C, the Accuracy is the same, however, by balancing other metrics with the imbalanced problem at hand, it is easy to conclude that Model C has the best performance, since the other metrics are more favorable.

Thus, regarding this pitfall, all main stakeholders must understand what is being presented to them, which is done by sharing knowledge on this subject. Furthermore, I also recommended that conclusions based on simply Accuracy were not enough to assume anything about the model, instead, with imbalanced problems, I highlighted the importance of understanding the impact of False Positives and False Negatives: If, for some problem, the impact of having False Positives was much greater, then the Recall should have more weight in the analysis. On the other hand, if False Negatives are more relevant, then Precision should have more weight. Ultimately, when this does not apply, the F1-Score is a good measure that averages the results obtained in the previous two metrics.

5. QUALITY ASSURANCE IN PRACTICE: PROJECTS

Having systemized the major pitfalls found while performing Quality Assurance as a Data Scientist at EDP D, we will now discuss three practical cases where such problems are more likely to be found. It is also important to notice that, despite having some participation as a developer in these projects, my main responsibilities here were the ones of quality assurance, by validating methodology, documentation, models, promoting the highest possible quality, and recommending solutions. The objective was to ensure the final product, developed mainly by the consulting outsourced team, had the reliability and quality that responded to the EDP D standards. Furthermore, to protect the integrity and confidentiality of the project and EDP data, most of the findings and/ or concrete examples are given in each project are anonymously encoded or just an illustrative example.

EDP D is aiming to achieve its digital strategy and vision by unlocking business value through advanced analytics and its use on asset management. In this context, three projects with the prime objective of applying Data-Driven approaches to solve a business problem were born. The first project, which is also called Minimum Viable Products (MVP), was regarding Failure Prediction, to predict possible failures 15 days into the future. Then, the estimation of the Health Index scores of each asset according to the OFGEM formula and its factors adaptation to EDP D reality. Lastly, all previous findings would be made accessible using Investment and Maintenance Planning Dashboards that allow the monitorization of the asset's health score, probability of failure, and other relevant insights that support decision-making. The scope of these projects regarded Overhead Lines, Transformers, and Circuit-Breakers. These projects started in September 2019 and ended in May 2020

All projects required enormous data ingestion and transformations. In other words, we had Big-Data problems that required Big-Data solutions. Therefore, the data that was ingested from several different sources and systems, such as Geographical systems, Weather systems, Asset Sensors, amongst others, was stored in the EDP Distribuição Microsoft Azure Data Lake, a centralized repository that holds structured and unstructured data, highly accessible and quick to update that can be used for analytical tasks. Afterward, this data was replicated into a project-specific Data lake, where four databases were created, each one with a specific purpose. The first one *(Raw)* was a full copy of the data from all sources, then a second database *(Curated)* was created with processed and ready to use data tables, later a third database *(Publish Zone)* was generated, representing an easy access point for consumers, and lastly, a final database *(Analytics Zone)* where data was used and modeled to solve the specific business problems, including the failure prediction and health index calculations.

All development in the *Analytics Zone* was made possible using Azure Databricks, an Apache-Spark unified platform for data and AI that allows for large-scale data processing inside the Microsoft Azure Cloud, useful for analytics tasks, allowing the development of code in several programming languages and collaborative work, as well as the deployment of Machine Learning algorithms into production. The developed steps in this phase are organized in different Databricks notebooks and then compiled and structured into a pipeline that runs inside Microsoft Azure Data Factory, a service that allows integrating of hybrid data at-scale, simplifying ETL tasks. The result of this pipeline returns information into a structured Microsoft Azure SQL Database that holds structured information to be read by the Investment and Maintenance Planning Dashboards, implemented in Microsoft Power BI, accessible to be analyzed by final users, supporting decision-making.

5.1. PREDICTIVE FAILURE CLASSIFICATION

The asset failure prediction project, which aimed at predicting possible asset failures within 15 days, followed a similar methodology to the ones found in the CRISP-DM and OSEMN frameworks. It began by understanding the data, including ingestion, exploratory data analysis, and data quality verification. Then, the data was preprocessed in a Data Preparation phase, where it was consolidated and cleaned, as well as the modeling hypothesis, based on predefined scope key objectives and data. Lastly, the Data Modelling phase where models were developed in an iterative supervised process, and results were evaluated concerning evaluation criteria and metrics.

5.1.1. Data Understanding

The data understanding phase was one of the most relevant steps in the entire project. It was where the knowledge of the data sources, in this case, asset sensor data, maintenance data, geographical data, weather data, and others, helped to understand how all this was connected. After the data was ingested and replicated, the data was split into two sets: Training and Testing (see Figure 17). Most of the data points, around 80%, were placed in the training set, representing the part of the data where all decisions and training of algorithms were made. The remaining data points, around 20%, were held in the testing set, where the performance of the final models was going to be evaluated. It is also important to highlight that, due to the high-class imbalance in this problem, where the minority class – failures – was around 0.7% for circuit breakers and 10% for overhead lines (after aggregating incipient and catastrophic failures), a stratified split was requiring, to ensure both training and testing sets had the same class proportion.

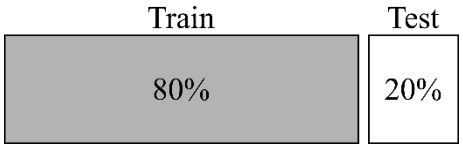


Figure 17. Train (80%) Test (20%) Split

After the split, an Exploratory Data Analysis (EDA) phase was required to summarize the data and make some early findings. Using data visualization techniques, the team began to analyze the distribution of failures by asset types, models, geographical location (districts), year, and month, amongst other relevant visualizations. The business and domain-knowledge in this phase were highly important because it validated some early findings and point out anomalous conclusions from data points that require additional attention. As a main task in the EDA phase, a verification of noisy data, such as Missing Values and Outliers was also performed.

5.1.2. Data Preparation

Having searched for anomalous values, missing values, or outliers, we proceeded to correct them. When dealing with outliers, we used a univariate simple technique that established the percentile 1 and percentile 99 of the identified predictive features to clean as a threshold limiter – All values below or above the thresholds would be considered as outliers, therefore, they would be corrected and their value would be set to the correspondent percentile, 1 or 99. This way, we ended up not removing any extreme observations but correcting them instead. Most of the missing information was filled by extracting additional data with the help of business-experts, leading to an overall higher quality dataset. When the information could not be extracted from other sources, an imputation step was applied by using the median value, which is a more suitable solution when compared to the mean value since it reduces the bias of more extreme observations.

Furthermore, in the data preparation phase, new features and transformations were also created, to extract the most information out of our data. One of the first transformations was regarding categorical variables. Since most algorithms cannot deal with categorical data, e.g. text, as Input, a One-Hot-Encoding technique was used to transform these variables into numerical data. This approach converts categorical columns by creating a binary feature for each unique category. The disadvantage of this, however, is the risk of increasing the dataset dimension in a significant way – the curse of dimensionality. At the end of this process, and creating several new variables, the dimension of the dataset was higher than what was required. Furthermore, there was also multicollinearity and redundancy between predictive variables, followed by a significant linear correlation. To avoid this problem, two variable selection techniques were applied: Feature selection using correlation, where a pairwise linear correlation between predictive features was performed, and all pairs with a value higher than 70% were filtered by keeping only the predictor that presented higher correlation with the target variable. This helped to reduce the dimensionality by half. Then, a feature selection using the purity measure was performed, which uses tree-based methods to evaluate the Information Gain and Gini index of the features. The importance of each feature was measured by the number of times each attribute split improved the performance measure, weighted by the number of observations the node was responsible for. After features were ranked and compared, the most important ones were selected based on a minimum defined criterion.

5.1.3. Data Modelling

In terms of modeling, there were two possible approaches to this supervised classification problem. We could either use White Box models, that are simpler and have a weaker predictive capability, however, they are more interpretable, or Black Box models, that despite losing significant interpretation ability, are more powerful. Regarding White Box algorithms, we tested a Logistic Regression and a Decision Tree classifier, which also worked as performance baselines for stronger Black Box models, which were a Random Forest classifier and an XGBoost classifier.

Asset	Model	AUC
Overhead Lines	Logistic Regression	0.78
	Random Forest Classifier	0.79
	XGBoost Classifier	0.80
Circuit Breakers	Decision Tree Classifier	0.61
	Random Forest Classifier	0.67
	XGBoost Classifier	0.73

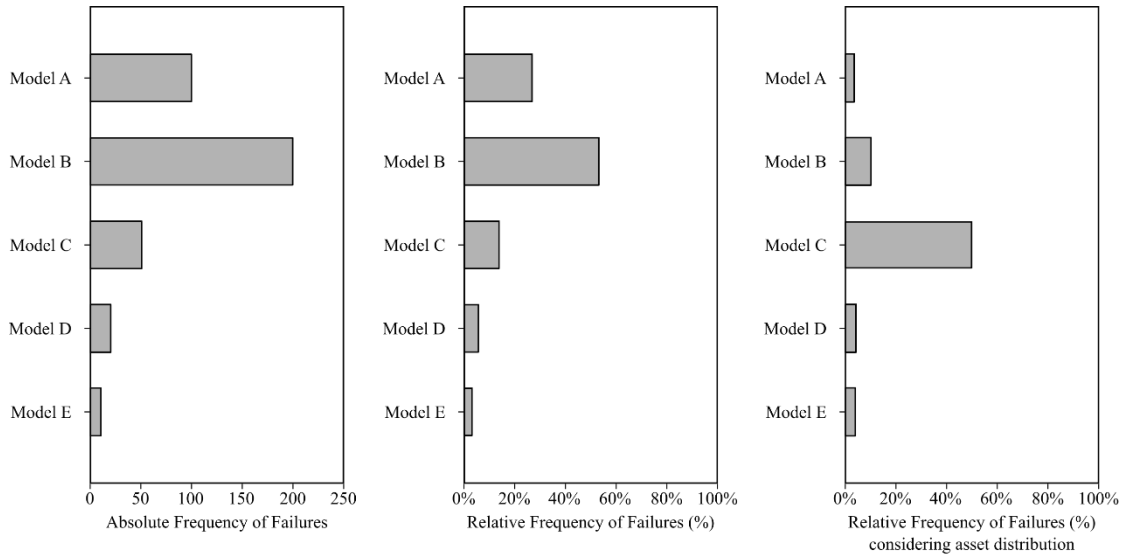
Table 4. Final model iterations comparison using AUC as the decisive metric

After trying and tuning several hyperparameters in each algorithm, the final assessment in the testing data was made using several metrics, however, the final decisions were made using the Area Under the Curve (AUC) that measures the area underneath a Receiver Operating Characteristic Curve (ROC), establishing a relationship between the true positive rates and false-positive rates at different classification threshold. Despite the differences being marginal (see table 4), they revealed the XGBoost classifier as the winning model. Therefore, this algorithm was selected and a pipeline to extract, process, and predict the target class from future unseen data was developed.

5.1.4. Quality Assurance

During this project, several improvements were identified, and solutions were advised, with the due corrections taking place. However, due to time and budget limitations, some of the discussed recommendations were left as future improvements that are now being developed. Furthermore, they served as lessons learned that will now be applied to future analytics projects.

Since this work followed a similar methodology to the one found in traditional supervised classification machine learning approaches, some of the identified and most significant Quality Assurance guidelines are more likely to appear in some specific phases. In the Data Understanding phase, it is required special attention to how visualizations are built, analyzed, and the importance of democratizing knowledge in this phase to better explore data and interpret early results. In this project, some conclusions were initially incorrectly interpreted due to visualization reasons.



Model	Freq.	Nr. Failures	Freq. of Failure (%)
A	3000	100	3%
B	2000	200	10%
C	100	50	50%
D	500	20	4%
E	300	10	3%

Figure 18. Asset Failure - How visualizing failure rate can be misleading (illustrative example)

When talking about asset failures, we had the objective of plotting which specific assets were most likely to fail, when compared to the training sample. Therefore, a bar plot was built that distributed the absolute number of failures for each specific asset (see Figure 18). The problem was that this visualization did not account for the distribution of the assets. The recommended solution was that every visualization whose objective was to relate a categorical feature with the failure rate should be done considering the category values distribution. Observing the example above, it is possible to see that if we only count the number of failures, in absolute or percentage, of the total number of failures per asset, we get a very similar distribution, that can lead to conclusions such as *"Model B tends to fail more"*, however, this is incorrect since we are not taking into consideration the distribution of the asset. In fact, despite having fewer registered failures than models A and B, model C failed around 50% of the time, making it the faultiest asset model.

Then, while preparing the data, it is likely to face some pitfalls such as data leakage and ignoring time in time-series, data and blindly encoding categorical data. Data leakage will happen if the proper separation between train and testing data does not occur. During this project, it was early decided that the definition of an asset's failure did not change during the years, thus, time was not considered as a key component of the analysis. While doing so, partitioning the data randomly between train and test and applying kcross-validation techniques in the training set will lead to data points from different years to be scattered in different partitions. In the end, the algorithm was likely to be trained using data from "future" years, e.g. 2018, 2019, and 2020, and tested with data points from "past" years, e.g. 2013, 2014, 2015. Despite the definition of failures not changing during the time, some predictive features had some time dependency that biased the results, thus, predicting on past data was likely overfitted. The solution was to consider and respect the time factor in this analysis, and to use cross-validation techniques that solved this problem, such as nested cross-validation (see Figure 13). However, due to the big structural changes, this would bring towards the project, this was identified as an improvement that is now being worked on.

County	**Relative Freq. Failures (%)**		**County**	**Risk County**
Ponte de Lima	15%		Ponte de Lima	1
Loures	12%		Loures	1
Abrantes	1%	➡	Abrantes	0
Lagos	2%		Lagos	0
Gouveia	5%		Gouveia	0
Matosinhos	20%		Matosinhos	1

Note Illustrative values if the average county fails rate was 10%. Relative failure rates above this threshold would lead to a *Risk County* of 1, otherwise 0.

Table 5. One-Hot-Encoding alternative (illustrative example)

Furthermore, when encoding categorical variables, a One-Hot-Encoding technique was applied, not considering the number of unique possible values. In the dataset used, there was a variable that indicated the Portuguese county in which a certain asset was located. Therefore, when applied to this encoding technique, 278 dummy variables were created, one for each county in Mainland Portugal. A better solution was later advised and discussed in chapter *4.5 Blindly encoding categorical data*, which aims to solve this problem by accessing the categories relationship with the target variable and encode it binarily, which we would then call *Risk Features*. These would be the comparison of each category failure rate with the average sample failure rate, creating a binary column that would indicate if a category presented had a higher or lower failure rate when compared to the average. In the county example (see table 5), the relative frequency of failures regarding the number of existing assets in each county would be calculated. If that value was above the sample average of all counties, then the *Risk Feature* would be 1, otherwise 0. This solution was scaled for the remaining categorical variables that presented a high number of unique values.

Then, during the data modeling phase, it necessary to be aware of the model interpretability vs. performance trade-off, as well as the evaluation and comparison using the right metrics. On one occasion, one of the main challenges in this project was to explain and extract the importance of each variable towards the outcome to business stakeholders. Since the selected model was an XGBoost ensemble, a black-box model that, despite the power, lacks interpretability, the developing team decided to use a decision tree to explain the importance of each predictive feature and how it affected the output. This approach was identified as incorrect since the conclusions drawn from a Decision Tree classifier are most likely different from what the XGBoost classifier is doing. So, the difference between these

types of models and the trade-off was clearly explained, and some other techniques to highlight feature importance, such as SHAP (Lundberg & Lee, 2017), were explored.

Lastly, the documentation was one of the most relevant guidelines that are cross-sectional to all framework phases. The description and identification of all sources, data, tools, and decisions made during the project should be stored in a document that is easily accessed and understood by anyone who wishes more knowledge about the subject. Due to the ambition of this project, the available time was short, so documentation was flagged as a minor priority when compared to the analysis itself. Having recommended some guidelines towards this documentation, denoted previously in this report, there was not enough time for the developing team to apply them all, especially regarding the documentation of code. Now, during improvements, changes are harder to perform, since you must trace back decisions to the core. This led to an understanding of the importance of documentation, where, in future projects, this will be a guideline to highlight from the very beginning.

5.2. HEALTH INDEX

To assess the asset's state of deterioration, EDP D decided to elevate its current manual formulas, which we call the Health Index, and innovate, automate and scale them to all assets nationally. To do so, a benchmark of three distinct methodologies was conducted. The objective was to assess which one strengthens the current EDP D Health Index methodology and obtain better results. The tested methodologies were OFGEM (Office of Gas and Electricity Markets), WAPA (Western Area Power Administration), and UNITEN (University Tenaga Nasional). After a close analysis from all involved teams, the OFGEM methodology was selected and we began adapting its formula, factors, and weights to the EDP assets reality.

5.2.1. OFGEM Methodology

The OFGEM approach starts from the premise that the health of a certain asset evolves along an increasing exponential curve: The higher the health value, the worse the asset's health. This curve is affected by several factors, such as the asset's expected life, aging factor, and many others. It is also allowed to forecast the health score of each asset 10 years into the future and estimate current and future failure probabilities, which complements the previous failure prediction project. A simplification of relevant factors and estimations can be found in Figure 19.

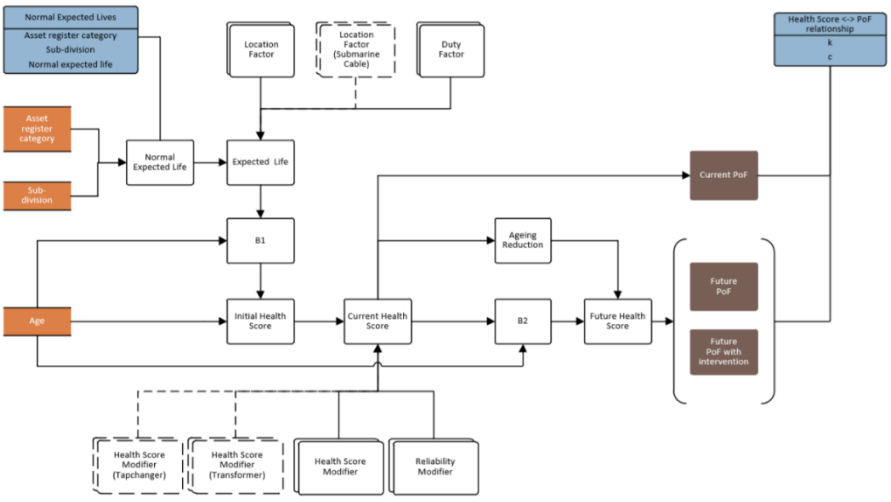


Figure 19. OFGEM framework architecture - How factors influence Health score and Probability of Failure

5.2.2. Health Index Work and Results

After several sessions adapting each factor and corresponding weights to the EDP D assets reality, a pipeline of Databricks notebooks containing the necessary Python code to estimate the health index of each asset was developed and connected using Microsoft Azure Data Factory. The result was a Health Index score on a continuous scale between 0.5 and 10 and extended up to 15 when forecasting future health, and assets were grouped into 5 Health Index bands according to the OFGEM's methodology, to make the results easier to understand (see Table 6).

Health Index Band	Health Index Banding Criteria	
	Lower Limit of Health Score	Upper Limit of Health Score
HI1	≥ 0.5	< 4.0
HI2	≥ 4.0	< 5.5
HI3	≥ 5.5	< 6.5
HI4	≥ 6.5	< 8.0
HI5	≥ 8.0	≤ 15.0

Table 6. Health Index Bands – HI1 (New condition) to HI5 (Poor condition)

As denoted, a HI5 will represent an asset in a poor deterioration state, while HI1 an asset in new conditions. However, users are more used to perceive high values as good, and low values as otherwise, so, to feed the planning and investment dashboards, transformations were made to make the health index more understandable (see Figure 20). The first step was to invert the OFGEM scale, where an excellent condition of 0.5 now represents a bad score. Then, the 0-10 scale was transformed into a 0-100 scale, for values to be read similarly to a percentage: 0% represents a significantly deteriorated asset while 100% a factory-new asset.

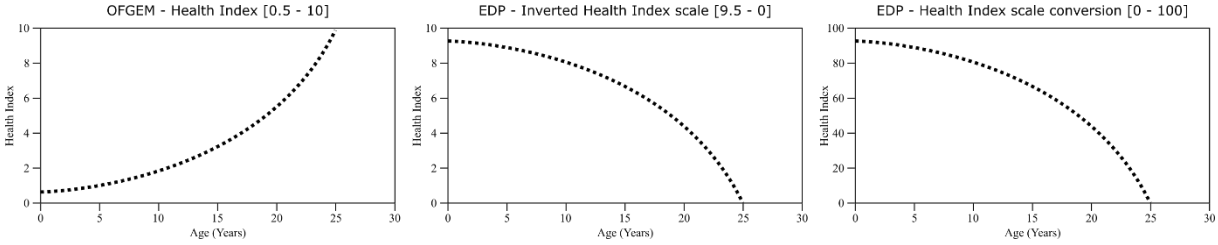


Figure 20. Health Index understandability adaptation

Calculating the scores massively, to every Transformer, Circuit breaker, and Overhead Line, we concluded Circuit Breakers were the assets in better conditions (see Figure 21), whereas 94.9% of them are in the first and second Health Index Band, with a HI score between 0.5 and 4.0 for the first HI band and 4.0 to 5.5 in the second HI band. On the other hand. 87.7% of Transformers' assets are in good conditions, despite having the highest proportion of deteriorated assets, around 5%. Lastly, Overhead Lines reveal 75% to be in good condition while 19.2% are classified as being in Health Index Band 3, an intermediate health score.

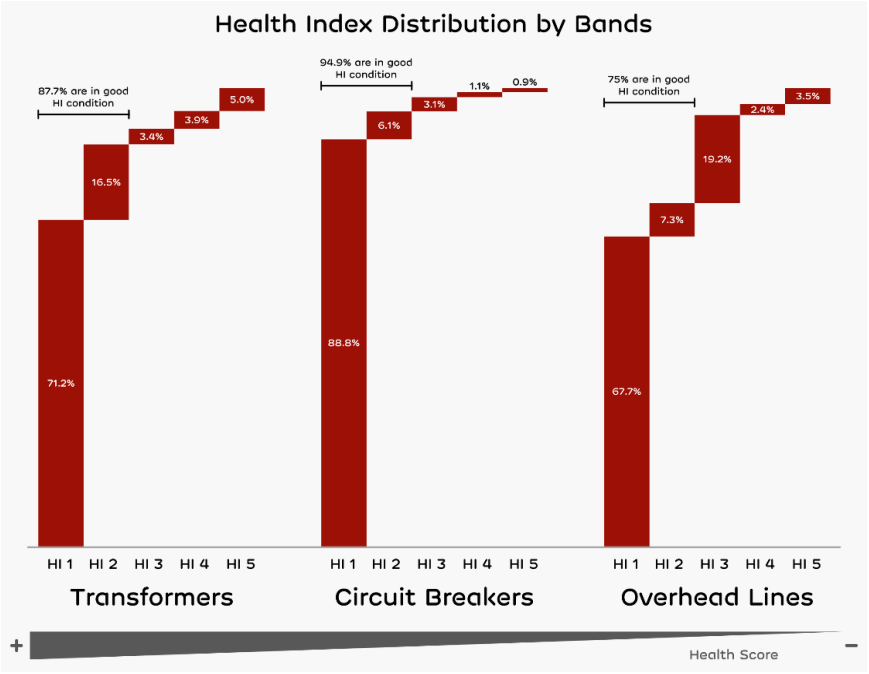


Figure 21. Health Index results by asset

5.2.3. Quality Assurance

Despite the health index calculations being quite different from the predictive failure classification project, it still left some room for quality assurance guidelines. However, the validation effort here requires less analytics knowledge and more business-domain expertise, since the major tasks were adapting EDP D assets factors to the OFGEM methodology. The main identified topic I identified in this project was code documentation. There was unused code in production pipelines and little code documentation. My recommendation here was to cleanse all unnecessary code and to utilize functions to avoid code redundancy. Furthermore, I also highlighted the importance of documenting these newly created functions with a description of the function, the inputs, and the outputs, for future developers to understand.

5.3. INVESTMENT AND MAINTENANCE PLANNING DASHBOARDS

After all, data was ingested, prepared and ready to use from the previous two projects, a pipeline in Data Factory was built that processed the necessary code to export the data, refreshed twice a day, to feed investment and maintenance planning dashboards, built-in PowerBI, a powerful, visually immersive, and interactive visualization platform that allows for actionable insights and support decision-making.

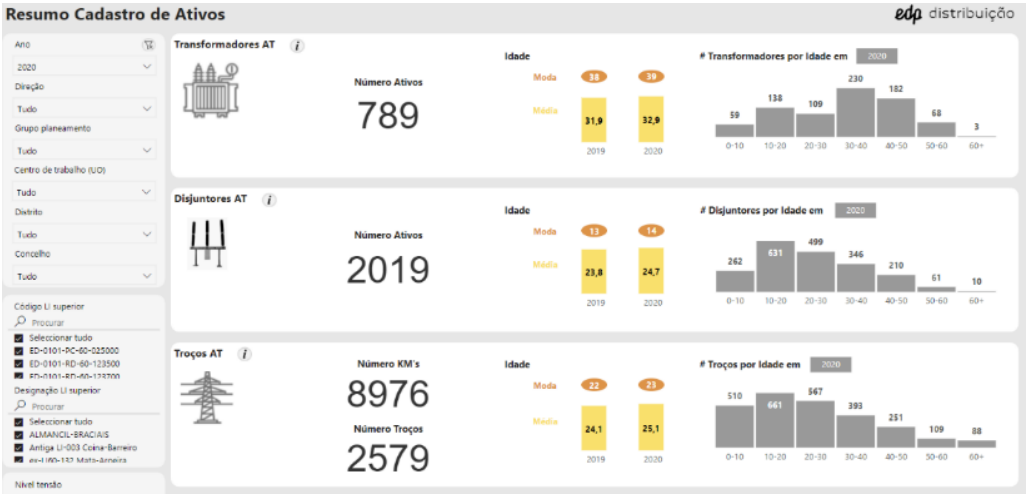


Figure 22. Investment Planning Dashboard snapshot

For the Investment Planning Dashboard (see Figure 22), the objective is to allow an overview of all assets in the project's scope, to observe future projections of asset investment needs, to analyze and filter asset characteristics for prioritization and investment allocation, and to monitor the distribution network asset condition. Furthermore, in the Maintenance Planning Dashboard, the purpose is that this becomes a centralized and unified solution for parameter analysis by an asset, including data from multiple sources. It also provides actionable insights such as current and future asset condition and deterioration status and failure prediction. Lastly, it allows for higher visibility over-scheduled interventions on simple and complex assets, in an easy-to-use platform for the end-user. The dashboards account for multiple asset visibility and focus on four main views: Global view, Condition and Risk view, Complex Asset view, and Asset Detailed view.

5.3.1. Quality Assurance

During this project, I had no role as a developer, working only on accessing the validity of the results. However, since this is only a platform that summarizes the major findings in the previous two projects, my focus was mainly on the validation of the predictive failure classification and health index. Nevertheless, I was able to identify some gaps in the documentation of this project and advise more information on the sources, the DAX code necessary to build tables in PowerBI, and how to read all indicators available in the dashboard.

6. CONCLUSIONS

This report explores the importance of Quality Assurance in the development of a data science project. In particular, it focuses on a set of recommendations delivered during the development of two projects by the EDP Distribuição Data Science team. The report highlights several pitfalls, solutions, and guidelines that sum up the lessons learned by the team and that can be applied in future projects. Furthermore, despite the project time limitations, the quality assurance validations were key to ensure minimum bias during analysis and that business rules were respected, as well as ensuring there was a solid knowledge base to support future developments that are now being made. In the end, these short-term validations can be the key to a long-term reliable final product.

Commonly, data science professionals will only focus on minimizing bias when developing analytical models, while forgetting some best practices when it comes to documenting the project or code, for example. Besides all cutting-edge tools or algorithms, it is necessary to learn from QA methodologies such as the Six Sigma DMAIC and apply them in these kinds of projects to ensure viability and continuity. As processes become more complex to model, the existence and importance of the role of a *"Quality Assurance Data Scientist"* professional in Data Science projects becomes increasingly relevant. His focus would be on bridging the gap between business and analytical processes understanding, as well as ensuring a transversal quality to the project, something only an expert on both fields, business, and analytics, would be able to do.

7. BIBLIOGRAPHY

Bergmeir, C., & Benítez, J. M. (2012). On the use of cross-validation for time series predictor evaluation. *Information Sciences*. https://doi.org/10.1016/j.ins.2011.12.028

Bergmeir, C., Hyndman, R. J., & Koo, B. (2018). A note on the validity of cross-validation for evaluating autoregressive time series prediction. *Computational Statistics and Data Analysis*. https://doi.org/10.1016/j.csda.2017.11.003

Black, M., Brint, A. T., & Brailsford, J. R. (2005). A semi-Markov approach for modelling asset deterioration. *Journal of the Operational Research Society*. https://doi.org/10.1057/palgrave.jors.2601967

Botvinick, M. M., Niv, Y., & Barto, A. G. (2011). Hierarchically organised behaviour and its neural foundations: A reinforcement-learning perspective. In *Modelling Natural Action Selection*. https://doi.org/10.1017/CBO9780511731525.017

Breiman, L. (2001). Random forests. *Machine Learning*. https://doi.org/10.1023/A:1010933404324

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. https://doi.org/10.1145/2939672.2939785

Gordon, A. D., Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). Classification and Regression Trees. *Biometrics*. https://doi.org/10.2307/2530946

Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: A review. *ACM Computing Surveys*. https://doi.org/10.1145/331499.331504

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2000). An introduction to Statistical Learning. In *Current medicinal chemistry*. https://doi.org/10.1007/978-1-4614-7138-7

Kotsiantis, S. B. (2007). Supervised machine learning: A review of classification techniques. In *Informatica (Ljubljana)*. https://doi.org/10.31449/inf.v31i3.148

Leinweber, D. J. (2007). Stupid Data Miner Tricks. *The Journal of Investing*. https://doi.org/10.3905/joi.2007.681820

Ligeza, A. (1995). Artificial Intelligence: A Modern Approach. *Neurocomputing*. https://doi.org/10.1016/0925-2312(95)90020-9

Loyola-Gonzalez, O. (2019). Black-box vs. White-Box: Understanding their advantages and weaknesses from a practical point of view. In *IEEE Access*. https://doi.org/10.1109/ACCESS.2019.2949286

Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems*.

M, H., & M.N, S. (2015). A Review on Evaluation Metrics for Data Classification Evaluations. *International Journal of Data Mining & Knowledge Management Process*. https://doi.org/10.5121/ijdkp.2015.5201

Mason, H. (2010). A Taxonomy of Data Science. *25.09.2010 Dataists. Fresher than Seeing Your Model Doesn't Have Heteroscedastic Errors*.

Quinlan, J. R. (1986). Induction of Decision Trees. *Machine Learning*. https://doi.org/10.1023/A:1022643204877

Quinlan, J. R. (1993). Combining Instance-Based and Model-Based Learning. In *Machine Learning Proceedings 1993*. https://doi.org/10.1016/b978-1-55860-307-3.50037-x

Rokach, L., & Maimon, O. (2006). Decision Trees. In *Data Mining and Knowledge Discovery Handbook*. https://doi.org/10.1007/0-387-25465-x_9

Schapire, R. E. (2003). *The Boosting Approach to Machine Learning: An Overview*. https://doi.org/10.1007/978-0-387-21579-2_9

Shearer, C., Watson, H. J., Grecich, D. G., Moss, L., Adelman, S., Hammer, K., & Herdlein, S. a. (2000). The CRISP-DM model: The New Blueprint for Data Mining. *Journal of Data Warehousing*.

Shultz, T. R., Fahlman, S. E., Craw, S., Andritsos, P., Tsaparas, P., Silva, R., Drummond, C., Ling, C. X., Sheng, V. S., Drummond, C., Lanzi, P. L., Gama, J., Wiegand, R. P., Sen, P., Namata, G., Bilgic, M., Getoor, L., He, J., Jain, S., … Mueen, A. (2011). Curse of Dimensionality. In *Encyclopedia of Machine Learning*. https://doi.org/10.1007/978-0-387-30164-8_192

Tsung, F. (2006). Six Sigma. In *Springer Handbooks*. https://doi.org/10.1007/978-1-84628-288-1_50

Tufte, E. R. (1988). The visual display of quantitative information. In *IEEE Power Engineering Review*. https://doi.org/10.1109/MPER.1988.587534

Wilson, J. M. (2014). Henry Ford vs. assembly line balancing. *International Journal of Production Research*. https://doi.org/10.1080/00207543.2013.836616