NOVA IMS Information Management School

MDSAA

Master Degree Program in

Data Science and Advanced Analytics

Reduction of emergency department returns after discharge from hospital

Machine learning model to predict emergency department returns 30 days post hospital discharge for medical patients

Ikram Bouziri

Dissertation

presented as partial requirement for obtaining the Master Degree Program in Data Science and Advanced Analytics

NOVA Information Management School

Instituto Superior de Estatística e Gestão de Informação

Universidade Nova de Lisboa

REDUCTION OF EMERGENCY DEPARTMENT RETURNS AFTER DISCHARGE FROM HOSPITAL

by

Ikram Bouziri

Dissertation report presented as partial requirement for obtaining the Master's degree in Advanced Analytics, with a Specialisation in Business Analytics

Supervisors: Professor Roberto Henriques and Professor Sílvia Lopes

November 2022

STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration. I further declare that I have fully acknowledge the Rules of Conduct and Code of Honor from the NOVA Information Management School.

Ikram Bouziri

Lisbon, 25/11/2022

ACKNOWLEDGEMENTS

I would like to extend my deepest recognition to my academic advisors Prof. Roberto Henriques and Prof. Sílvia Lopes for giving me the opportunity to work on this project, and I would like to thank them for the guidance, encouragement and commitment they have been showing throughout this thesis period.

Additionally ,This journey could not be possible without the efforts of my parents, I can never thank them enough for what they have been offering to me, for always believing in me , for their unconditional love and sacrifices. I am also very grateful to my beloved brother for his continuous encouragement and positive energy.

Moreover, I would like to express my deep gratitude to my whole family and my friends in Tunisia, Portugal and all over the world. I would like to thank them for the moral support and constant source of inspiration, their belief in me has kept my spirit and motivation high during this process.Thank you for all the beautiful moments shared with you, they counted a lot.

I am also thankful to my office team for their support and understanding during these months.

I dedicate this work to everyone that contributed in making this incredible experience happen and helped me become the person I am today.
I am very grateful to have you all by my side, thank you for all the support, love and care.

ABSTRACT

Post-hospital discharge returns to emergency departments are associated with reducing the efficiency of the emergency department (ED) utilisation and the quality of healthcare. These returns are often related to the nature of the disease and/or inadequate care.

This thesis aims to develop a machine-learning model that predicts ED returns within 30 days of inpatient discharge from Portuguese public hospitals. Different binary classification models were trained and evaluated with a particular focus on sensitivity (predictive power of the critical class of returning patients). The selected model was the Extreme gradient boost Classifier, which showed the best performance on recall and the other considered performance metrics. A cohort of 93 449 medical hospitalisations of adult patients discharged between January 1st, 2018, and December 31st, 2019, was assembled with diagnoses details to be used in this study. According to the problem's requirement, the recall was the performance metric to be maximised. Therefore, Performance optimisation methods were considered, and the final model resulted in a recall of 84.38%, precision of 84.35%, F1 score of 84.36% and accuracy of 84.10%.

Future deployment and integration of this ED return predictive analytics into the inpatient care workflow may allow identifying patients that require targeted care interventions that reduce overall healthcare expense and improve health outcomes.

KEYWORDS

Machine Learning; Binary Classification; Emergency Department Returns; Performance Metrics; Performance Optimisation

INDEX

TABLE OF FIGURES

LIST OF TABLES

LIST OF EQUATIONS

1. INTRODUCTION

1.1. THESIS CONTEXT AND RESEARCH QUESTIONS:

Post-discharge returns to emergency departments are undesirable, not only from the point of view of patients and their families but also providers and the health system since they increase healthcare costs and utilisation. In Portugal, previous studies showed that about 23.26% of patients discharged from the hospital return to the emergency department (ED) at least once within 30 days(Salgado et al., 2022). ED revisits can be classified as treat-and-discharge visits, observation stays, and hospital readmissions and are often associated with inadequate post-discharge care.

This thesis is conducted as part of a research project entitled READY[1], which aims to reduce avoidable ED utilisation by identifying patients at increased risk of returning to the ED in the 30 days post inpatient discharge from Portuguese hospitals, enabling targeted post-discharge interventions such as phone calls, home visits or online monitoring of patients with increased risk.

The project READY has three specific objectives. The first is to identify predictors of return to the ED within 30 days after discharge and develop a prediction model to identify high-risk patients at discharge. The second is to characterise the factors reported by patients that contributed to the ED return 30 days post-hospital discharge and the measures they feel could have been avoided. The final aim is to identify innovations in transitional care with a high potential to reduce avoidable returns to ED.

This thesis tackles the project's first objective and aims to develop a classification machine-learning model to predict the risk of returns to ED within 30 days of discharge from hospital inpatient departments.

For the READY project, files containing inpatient hospitalisation episodes and emergency episodes for patients discharged between 2018 and 2021 were provided from 3 university hospitals in Portugal - CHU[2] Algarve, Central Lisbon, and São João.

The integration of the files from the three hospitals, followed by the primary transformations and feature engineering, were performed in the first place. Then, a study sample of medical adult patients (aged >= 18 years) discharged between January 1st 2018 and December 31th, 2019, from the participating hospitals was selected and preprocessed to train and evaluate the model for this study.

The completion of this project will allow a deeper understanding of the patterns of care utilisation and the development of recommendations that will contribute to the improvement of transitional care by promoting the reduction of avoidable returns to ED.

[1] Reduction of Emergency department returns After Discharge from hospital

[2] Centro Hospitalar Universitário

1.2. THESIS STRUCTURE:

The present document is structured into five chapters, including the Introduction:

- Chapter 2 includes a review of previous studies related to the topic.
- Chapter 3 describes the methodology used, from the exploration and understanding of the data to the different pre-processing steps performed and the modelling techniques experimented with.
- In Chapter 4, the results of the experiments were reported and discussed, and the model selection process was explained.
- Chapter 5 concludes this thesis with the limitations, achieved objectives and future work

2. LITERATURE REVIEW

Risk prediction of Emergency Department (ED) returns is a research topic of great interest, and it has been extensively studied in recent years since it helps identify patients requiring further post-discharge attention and reduces healthcare costs and utilisation. Previously built predictive models generally try to evaluate the risk of a specific subgroup within ED returns which is the risk of unscheduled hospital readmissions. This prediction is performed within a given period after the patient's discharge from the hospital.

Several previous risk prediction studies of early hospital readmission have been published. Most of them targeted a particular subpopulation, such as elderly patients (>65 years old) or postoperative patients (Marcantonio et al., 1999; Mišić et al., 2020). Others focused on specific conditions or chronic diseases, for example, AIDS, Pneumonia, Congestive Heart Failure and/or chronic obstructive pulmonary disease (COPD) (Artetxe et al., 2017; Grant et al., 1999; Krumholz et al., 1997; Smith et al., 2000).

Artetxe et al. used the Kaiser Permanente Risk Stratification Model to predict 30-day ED readmission risk for high-risk patients, including patients with a specific organ disease and high multi-morbidity. Different classifiers were tested, with a particular focus on sensitivity when evaluating the performance, and the best sensitivity was achieved by the Support Vector Machine SVM algorithm using over-sampling methods to deal with the class imbalance problem (Artetxe et al., 2017).

Most studies that predicted hospital readmission risk focused primarily on medical admissions, whereas surgical admissions received less attention (Mišić et al., 2020). In recent research, a sample of surgical patients was used to show that tree-based machine learning methods can accurately predict readmissions in postoperative patients via the emergency department 30 days after surgery with excellent discrimination using surgical and demographic features along with lab features(Mišić et al., 2020). The same study further demonstrated that the risk of readmission could be confidently calculated the 36 hours post-surgery which increases the efficacity of the model allowing medical care to take action earlier when needed (Mišić et al., 2020). This would avoid waiting for discharge-level data, which leads to ineffective transitional care coordination or eventual prolongation of a hospital stay.

Moreover, some studies on hospital readmission risk considered patients discharged from a single hospital or unit such as an acute-care teaching hospital (Phillips et al., 1987) or Veterans Affairs medical centres (Smith et al., 2000). However, studies conducted using data from more than one hospital also exist, such as Billings et al.2006, and Bottle et al. 2006 that both used admissions from hospitals within

the National Health Service hospital trust in England (Billings et al., 2006; Bottle et al., 2006) which is also the case for this work since data was provided from 3 public hospitals in Portugal.

The downside of Bottle et al.'s model for identifying patients at high risk of emergency hospital admissions is that it requires access to data such as community-level and socio-economic-level (community-level admission rates, ethnicity and education) that are difficult to collect. Another impractical model, in terms of variables, used administrative data and detailed sociodemographic and health information collected from the patients during interviews conducted within 48 hours of admission and 30 days after discharge (Hasan et al., 2009).

A trendy and straightforward model that embraced a general population of medical and surgical patients is the LACE index (van Walraven et al., 2010). It was developed to predict the risk of death or unscheduled readmissions within 30 days after discharge from hospitals using four main admission-level variables: length of stay ("L"); acuity of the admission ("A"); comorbidity of the patient ("C"); and emergency department use ("E") (van Walraven et al., 2010). LACE index showed satisfactory performance and accuracy at predicting outcome risk. The same authors further improved this index and developed an extension called LACE+ that utilises administrative data to predict better the risk of post-discharge returns (van Walraven et al., 2012).

The period fixed for predictions in the previously mentioned studies is 30 days post-discharge. However, other models were trained to predict the risk of readmissions within only 72 h of discharge, and they were called "Short-term reattendances" or "early bounce-backs" (Chmiel et al., 2021; Davazdahemami et al., 2022).

From this review of literature, it can be noticed that the previous studies on this research topic focused particularly on predicting hospital readmissions rather than ED returns, which helps healthcare units to manage better the utilisation of the departments by identifying patients with a high risk of being readmitted to the hospital after discharge. However, the information about the overall ED returns is missing; this is where this thesis comes to use. In this study, all the patient's emergency department admissions were considered when creating the target variable. Thus, the model was trained to predict if the patient will return to ED or not independently of whether it is a treat-and-discharge visit, an observation stay, or hospital readmission. This would give the vision to understand better the ED returns patterns and increase the efficiency of the ED utilisation. Moreover, for adult patients, the model developed in this thesis could be applied to any medical inpatient admission case, independently of the category or severity of diagnoses.

3. METHODOLOGY

This chapter presents all steps followed from data understanding until the modelling phase as shown in Figure 1. Section 3.1. describes the files collected from the hospitals and their integration. Further, in Section 3.2., the data preparation steps of data transformation, features engineering, target creation, sample selectiopn and pre-processing are explained along with the techniques used for feature selection. At last, Section 3.3. details the steps done for modelling explaining the class imbalance methods performed, the classification models trained and the performance metrics used to evaluate them.

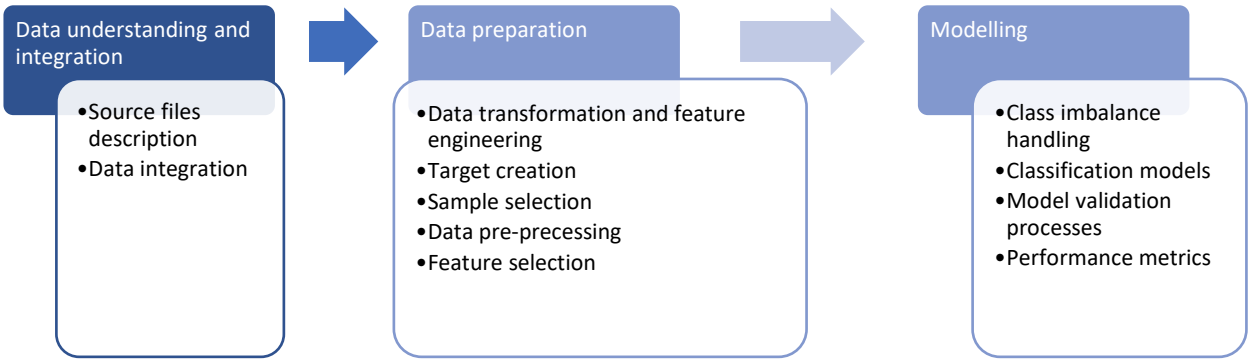

Figure 1: Methodology diagram

3.1. DATA UNDERSTANDING AND INTEGRATION

3.1.1. Sources files description

Data was collected from three university hospitals in Portugal, specifically from Algarve, Central Lisbon, and São João. After approval from the ethics committee, each hospital provided four source files containg the following types of information:

- Inpatient hospitalisations episodes files: record all episodes discharged from inpatient care between January 2018 and December 2021.
- Emergency episodes files: record all ED admissions from January 2018 until December 2021.
- Diagnoses files: detail the principal and additional diagnoses recorded within each inpatient hospitalsation.
- Death files: record the death dates of all the patients that died

3.1.2. Data integration

The source files were preprocessed and transformed to deal with inconsistent variable names, values and data types. Then, files from the three hospitals containing the same information type were merged

to obtain four primary datasets as illustrated in Figure 2. Table 1 details the features of each resulting dataset and their descriptions.

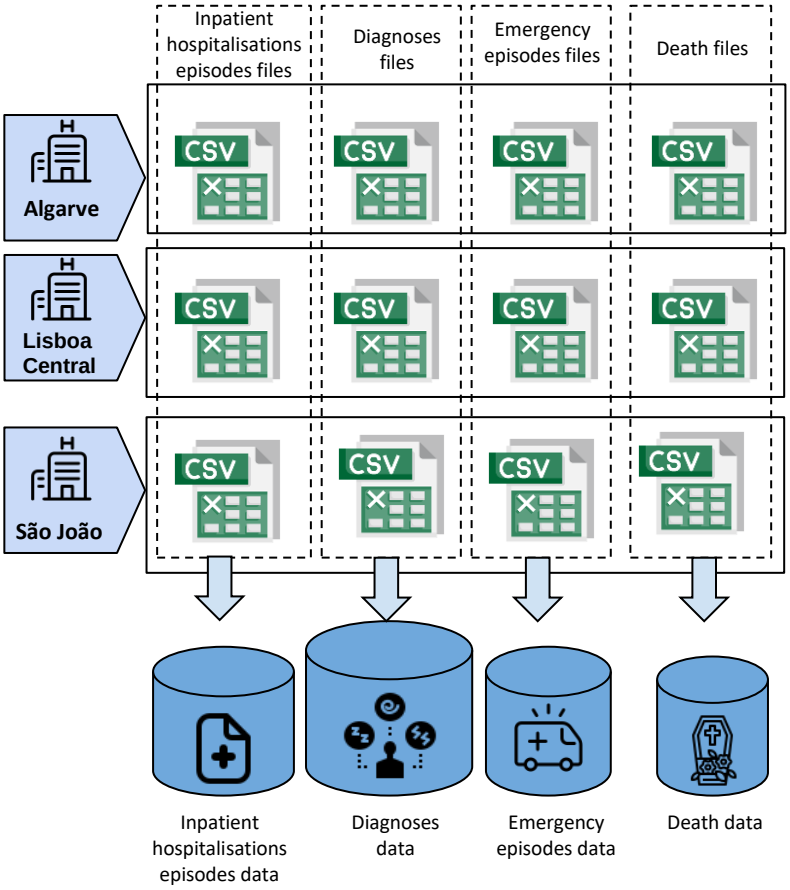

Figure 2: Data integration

Dataset	level	feature	Description
Inpatient hospitalisations episodes	Admission-level variables	Inpatient admission ID	Admission identifier
		Admission date	Date of admission to the hospital
		Major Diagnostic Category (MDC)	Major Diagnostic Category code
		Diagnosis Related Group (DRG)	Diagnosis Related Group code
		DRG severity level	Level of severity
		DRG mortality level	Level of mortality
		Length of stay	Number of days of the stay
		Type of stay	Type of the stay (long, short…)
		Discharge date	Date of discharge from hospital (between January 2018 and December 2021)
		Discharge status	Reason of discharge or destination after discharge

	Patient-level variables	Patient ID (fictitious)	Patient fictitious identifier (created for this project purposes, not possible to link with other databases)
		Age	Age of the patient
		Gender	Gender of the patient
		Residence location	3 variables : district, municipality and parish of the patient's residence
Emergency episodes	Emergency-level variables	Emergency ID	ED admission identifier
		Patient ID (fictitious)	Patient fictitious identifier (created for this project purposes, not possible to link with other databases)
		Emergency admission date	Date of admission to ED (between January 2018 and December 2021)
		Emergency area	Specific emergency area
		Triage priority	Priority of the ED admission (very urgent, urgent, not urgent…)
		Cause	Primary cause of the ED admission
		Origin	From where the patient came (home, another hospital..)
		Emergency discharge date	Date of discharge from ED
		Emergency discharge status	Reason of discharge or destination after discharge from ED
Diagnoses	Diagnosis-level variables	Inpatient admission ID	Admission identifier
		Patient ID (fictitious)	Patient fictitious identifier (created for this project purposes, not possible to link with other databases)
		ICD version	International Classification of Diseases used
		Diagnosis order	Ranking of the diagnoses (rank 0 is the main diagnosis …)
		Diagnosis code	ICD code of the diagnosis
		Diagnosis description	Description of the diagnosis
		Present on admission	Whether the diagnosis is present on the admission or not
Death		Patient ID (fictitious)	Patient fictitious identifier (created for this project

	Patient-level variables		purposes, not possible to link with other databases)
		Death date	Date of death

Table 1: Description of the datasets

3.2. DATA PREPARATION

3.2.1. Data transformation and feature engineering

As mentioned in the last section, diagnoses data details all the principal and additional diagnoses recorded within each inpatient hospitalsation. The resulted diagnosis dataset was used to create more useful features. In fact, a new variable of the total number of diagnoses per inpatient admission was created. Moreover, the shape of this dataset was transformed to aggregated the diagnoses information by creating two comorbidity indices; Charlson and Elixhauser. The Charlson and the Elixhauser comorbidity indices are methods used to classify patients' comorbidities based on the International Classification of Diseases (ICD) diagnoses codes found in administrative data (Anne Elixhauser et al., 1998; Charlson et al., 1987). These indices are further explained in the Appendix of this document.

As shown in Figure 3, the new diagnosis-level features were added to the inpatient episodes dataset containing a total of 435 883 records. Then, to remove the records that should not be considered, three exclusions rules were performed:

- 68 records of duplicated inpatient IDs that were removed because they were assumed to be mistaken.
- 5021 records of the patients discharged after December 2nd, 2021 were excluded. In fact, the 30 days interval could not be captured for those records since the latest provided emergency admission data was until December 2021.
- 51937 records having one of the following descriptions in the discharge status variable were excluded:
    - Death during the hospital stay
    - Home hospitalisation
    - Transfer to another hospital
    - Left against medical advice

After performing these exclusion rules, the final dataset contained 378 875 inpatient episode records as shown in Figure 3.

3.2.2. Target creation

This thesis aims to develop a classification machine-learning model to predict whether the patient will return to ED within 30 days of discharge from hospital inpatient department or not. To build this classification model, a binary target variable should be created. First, the number of ED admissions within 30 days of hospital discharge was calculated for each patient having an inpatient episode, using the emergency admission date from the emergency episodes dataset. Then, this newly created numerical variable was transformed to binary, creating the target variable in the final inpatient episodes dataset as shown in Figure3 :

- 1: the patient concerned with the admission returned to ED within 30 days of hospital discharge
- 0: the patient concerned with the admission returned to ED within 30 days of hospital discharge

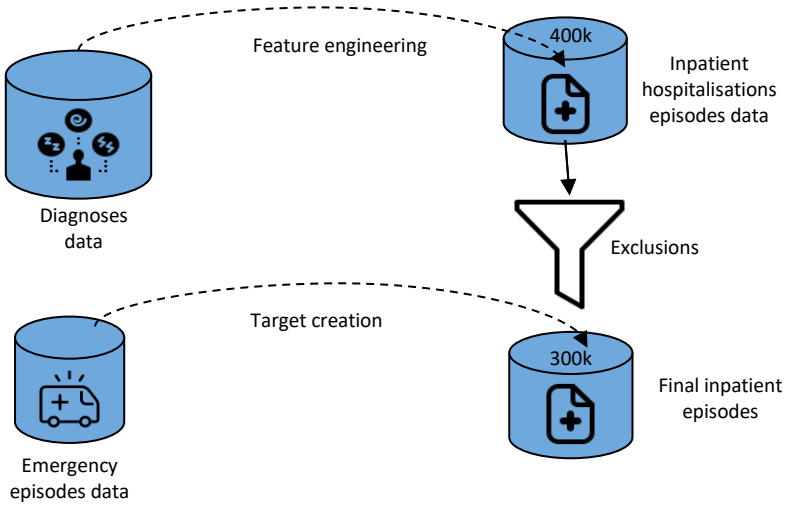

Figure 3: Feature engineering and target creation

3.2.3. Sample selection

To build the model, a sample including adult patients (age greater or equal to 18 years old) hospitalised for medical admissions and discharged between January 1st 2018, and December 31st 2019, was selected from the final dataset. This sample represented 93 449 records, 20 independent variables, and a target variable described in Table 2.

Variable name	Variable type	description
ID_inpat	String	Identifier of the inpatient episode (index of the dataset)
ID_patient	String	Identifier of the patient
hospital	String	Hospital name
age_rounded	Integer	Patient's age rounded

age_group	String	Patient's age group
gender	String	Patient's gender
disch_status_inpat	String	Description of the discharge status of the patient after the episode
drg	String	Diagnosis Related Group code
mdc	String	Major Diagnostic Category code
severity_level	String	Severity level
mortality_level	String	Mortality level
district	String	Patient's residence district
municipality	String	Patient's residence municipality
parish	String	Patient's residence parish
los	Integer	Length of stay
LOS_type	String	Length of stay type
number_diagn	Integer	Number of diagnoses of the inpatient episode (principal and additional)
charlson	Integer	Charlson comorbidity index
elixhauser	Integer	Elixhauser comorbidity index
diagn_cat	String	Main diagnosis category
target_return	Binary	Target variable, indicates if the patient returned with 30 days of discharge or not

Table 2: Independent and target variables

3.2.4. Data pre-processing

After selecting the sample, further data cleaning and transformation were done by treating missing data and outliers, standardising the numerical variables, and encoding the categorical variables.

3.2.4.1. Data cleaning:

- Outliers handling

Outliers are extreme values that appear in the dataset and can be extremely small or large. Outliers are abnormal values, and their presence can often skew the results of statistical analyses on the dataset. Since machine learning models learn from data to understand the trends and relationship between data points, outliers can impact the overall effectiveness and usefulness of the model (Brownlee Jason, 2020).

Outlier detection and removal are critical in safeguarding data quality and ensuring that the trained model generalises well to the valid range of test inputs. It also must be noted that outlier detection is also performed after deployment to maintain the effectiveness of models.

Outliers were visualised using boxplots as shown in Figure 4.

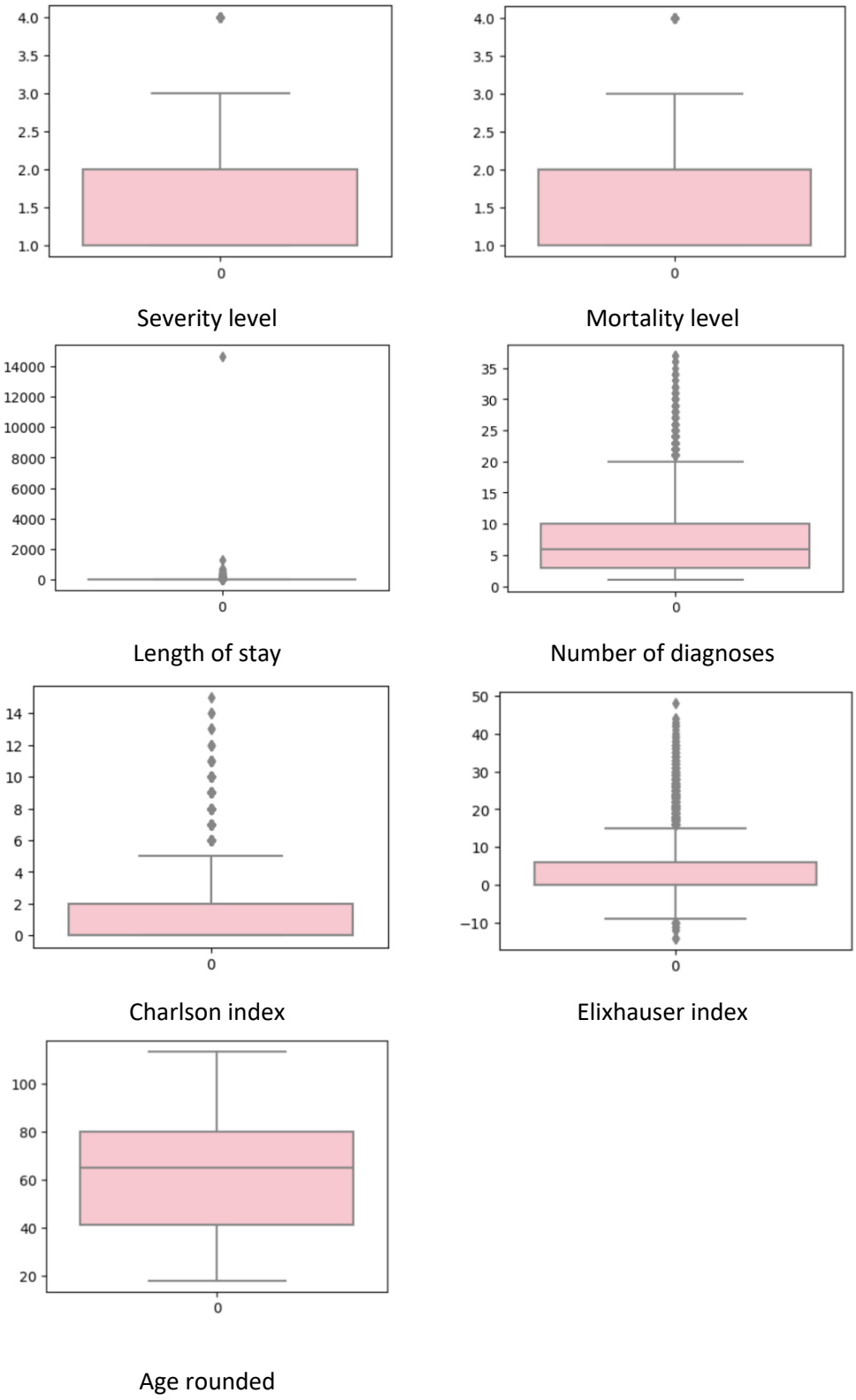

Figure 4: Outliers boxplots

Except for the variable age, outliers were removed from all numerical variables. For this purpose, the clipping method was used where values outside a given interval are clipped to the interval edges. The intervals, in this case, were defined using percentiles, the 1st percentile defines the lower limit, and the 99th percentile defines the upper limit. Therefore, all values below the 1st percentile become the

value of the 1st percentile for each variable. All values greater than the 99th percentile become equal to the value of the 99th percentile.

Table 3 presents the defined intervals and number of outliers:

Variable	Defined interval	Number of lower outliers	Number of upper outliers
Length of stay	[1 ; 63]	0	923
Number of diagnoses	[1 ; 21]	0	809
Charlson index	[0 ; 7]	0	789
Elixhauser index	[-5 ; 24]	842	819

Table 3: Number of outliers

- Missing values handling

One of the most common problems in data pre-processing when working with real datasets is handling missing values. Missing data is the values or data that is not stored (or not present) for some variable/s in the given dataset. This could occur because of many reasons. It could be because some observations were not recorded due to human errors, or data is corrupted, or it could be intentional when data is not provided for some reason. Missing values can bias the results of the machine learning models and reduce their accuracy and performance. Therefore, it is essential to treat them (Brownlee Jason, 2020). Many strategies exist to handle this problem, and the most basic one is to ignore all the records having a missing value in one of the columns. This approach is simple, but it leads to losing valuable data. That is why it is used in those cases where the number of missing values is minimal.

Other strategies to impute or substitute this incomplete data are:

- Drop the records with missing values: performed only for data errors and small data
- Drop the column with missing values: usually performed only when the percentage records having missing values os greater than 20%.
- Substitute or impute the missing value: for numerical features missing values could be substituted by zero or any user-defined value, the mean, the median, the maximum value or the minimum value. For categorical features, missing values could be substituted by a user-defined value, a new category inidicating missing value or the mode. The missing values could also be imputed using an imputation algorithm like K-Nearest Neigbors.

After analysing the missing values in all the variables, Table 4 presents the decisions made to treat them:

Variable	Percentage of missing values	Strategy
District	0.01%	Drop the records
Municipality	0.01%	Drop the records
Parish	0.01%	Drop the records
LOS_type	54%	Drop the column

Table 4: Percentage of missing values and strategies applied

3.2.4.2. Data transformation:

The data transformation step is one of the fundamental steps in data pre-processing and helps increase accuracy in the models. In this case, data transformation was performed when scaling the feature and encoding the categorical variables.

- Feature scaling

Generally, datasets include different kinds of variables, and the range of values varies widely in the same. A significant issue is that when using the original scale, the model may put more weight on the variables with an extensive range; thus, results will be biased (Brownlee Jason, 2020). In order to deal with this problem, we need to apply the technique of feature scaling to ensure features are on almost the same scale so that each feature is equally important and makes it easier to process by most ML algorithms (Brownlee Jason, 2020). Some machine learning models, such as K-Nearest-Neighbours and SVM, are fundamentally based on distance matrix; they are also known as distance-based classifiers. Feature scaling is exceptionally essential to those models, especially when the range of the features is very different (Brownlee Jason, 2020). Otherwise, the model will assign larger weights to features with an extensive range, thus, influencing the computation of the distance.

Also, machine learning algorithms that use gradient descent as an optimisation technique, like linear regression, logistic regression, neural network, etc., require data to be scaled to ensure that the gradient descent moves smoothly towards the minima and that the steps for gradient descent are updated at the same rate for all the features. Moreover, gradient descent converges much faster towards the minima with feature scaling than without it.

However, tree-based algorithms are relatively invariant to the scale of the features because a decision tree splits a node on a single feature, which increases the homogeneity of the node. Therefore, the remaining features have virtually no effect on the split.

There are several techniques to scale the variables, but, in this case, the two most known methods, standardisation and normalisation, were tested:

- Standardisation (standard scaler):

Standardisation, also called Z-score normalisation, is a scaling technique that gives each attribute zeromean and a standard deviation of one.

$$X' = \frac{X - \mu}{\sigma}$$

Equation 1: Standardisation equation

μ : mean of the feature values

σ : standard deviation of the feature values

It must be noted that, in this case, the values are not restricted to a particular range. Standardisation is more effective when the attribute has a Gaussian distribution. It is useful when the data has varying scales and the algorithm used makes assumptions that the data has a Gaussian distribution, such as linear regression, logistic regression, and linear discriminant analysis.

- Normalisation (Min Max scaler):

The normalisation, also known as Min-Max scaling, is another scaling technique where, for every feature, the minimum value of that feature gets transformed into 0, and the maximum value gets transformed into one resulting in all features having a distribution value between 0 and 1.

$$X' = \frac{X - X\max}{Xmax - Xmin}$$

Equation 2: Normalisation equation

Normalisation is a simple technique, and it is good to use when the distribution of data is not known or not Gaussian. It is useful when the data has varying scales and the algorithm used does not make assumptions about the distribution of the data, such as k-nearest neighbours and artificial neural networks.

However, Normalisation does not treat outliners very well. On the contrary, standardisation is more robust to outliers since it does not have a bounding range and facilitates convergence for some computational algorithms like gradient descent. Therefore, in many cases, it is preferable to use standardisation over Max-Min normalisation, but it highly depends on the use case and the machine learning algorithm being used.

In this project, Feature scaling using standardisation method showed better results that normalisation.

- Categorical data handling

A machine learning model's performance depends on the model and the hyperparameters and on how we process and feed different variables to the model. Most machine learning algorithms cannot directly operate on categorical data and require independent variables to be numeric; treating the categorical variables becomes necessary. Encoding categorical data is turning categorical data into integer format so that data with converted categorical values can be fed into models and improve the accuracy of predictions (Brownlee Jason, 2020).

Usually, there are two kinds of categorical data:

- Ordinal Data: The categories have an inherent order. Thus, the information regarding the order in which the category is provided should be retained while encoding.
- Nominal Data: The categories do not have an inherent order. While encoding, only the presence or absence of a feature must be considered.

Compared to tree-based models, linear models are more sensitive to the order of ordinal data, which is why the appropriate encoding methods should be selected depending on the algorithm used.

The following is a brief presentation of four popular approaches for categorical data encoding:

- Label encoding

This type of encoding is used when the categorical feature in the data is ordinal, and order retention is essential. Ordinal encoding is an easy and informative process that assigns an integer value to each label (Brownlee Jason, 2020). Therefore, the encoded data reflects the sequence of labels.
Label Encoding causes a prioritisation issue because machine learning models assign a high value with a higher priority than the lower value. Consequently, it is mostly only applicable to ordinal data and would mislead the model when used with data that does not have any meaningful order.

- One-hot encoding

This encoding technique is used when the data is nominal. It transforms each category of any categorical variable into a new binary column represented by 0 or 1 to signify the presence of the category value. Newly created binary features can be considered dummy variables. After one hot encoding, the number of dummy variables depends on the number of categories presented in the data. One-Hot Encoding is helpful for categorical columns without any order and less cardinality because it treats all the values within the categorical column as equal. Moreover, a high cardinal categorical column would produce many columns, leading to the curse of dimensionality, which is the main drawback of this technique (Brownlee Jason, 2020).
It must also be noted that One-Hot Encoding might not be suitable for Tree-based Machine Learning because it causes inefficiency during splitting.

- Target encoding

Target encoding is a Bayesian technique that uses information from that target variable to encode the categorical data. It converts a categorical value into the mean of the target variable. It is performed for train data only, and the test data is coded using results obtained from the training dataset. The main issue with this technique is that it can lead to target leakage or overfitting. However, some modified versions of this technique helps to reduce this problem, such as the Leave-one-out encoding (LOO encoder). As its name mentions, this encoding type leaves out the value's target value to be encoded when calculating the mean. Another technique is the Generalised Linear Mixed Model (GLMM encoder), which basically applies a linear regression on target encoding. This method gives robust results but is time-consuming compared to the other methods.
In the context of this thesis, a high cardinality issue was identified, which eliminated the possibility of using one-hot encoding. As the type of all the categorical features in the dataset is nominal, it was

impossible to apply label encoding. Therefore, trials were made using the target encoder and its more advanced versions. Table 5 shows the details of the cardinality of all the categorical variables:

Variable	Cardinality
disch_status_inpat	11 unique values
drg	161 unique values
mdc	25 unique values
district	28 unique values
municipality	292 unique values
parish	2251 unique values
Diagn_cat	1166 unique values

Table 5: Cardinality of categorical variables

3.2.5. Feature selection techniques

Adding redundant variables when developing a predictive model reduces the generalisation capability of the model, increases its overall complexity, and could also reduce the overall accuracy of a classifier. Therefore, it is preferable to do the feature selection process, which is performed using some techniques to reduce the number of input variables to reduce the computational cost of modelling and increase the performance (Brownlee Jason, 2020).

There are two main types of feature selection techniques:

- Supervised feature selection techniques: they consider the target variable and can be used for the labelled dataset
- Unsupervised feature selection techniques discard the target variable and can be used for the unlabelled dataset.

Supervised feature Selection may be divided into three other techniques:

3.2.5.1. Filter methods

These methods evaluate the importance of features based on their inherent characteristics, not including any machine learning algorithm. Instead, they use statistical measures to score the correlation between the input variables as well as the relationship between each input variable with the target, then select the subset of features that have the most substantial relationship with the target variable and choose between the independent variables that are highly correlated with each other. The advantages of filter methods are that they do not overfit the data, and due to their simplicity, they tend to be fast and have a low computational cost. Thus, these approaches are extensively used on high-dimensional data.

Although these methods are fast and effective in most cases, it can be challenging to choose the appropriate statistical measure for feature selection because it highly depends on the data type of both the input and response variables. This tree graph in Figure 5 presents a way to choose the relevant

statistical measure for filter-based feature selection based on whether the input and output variables are numerical or categorical.

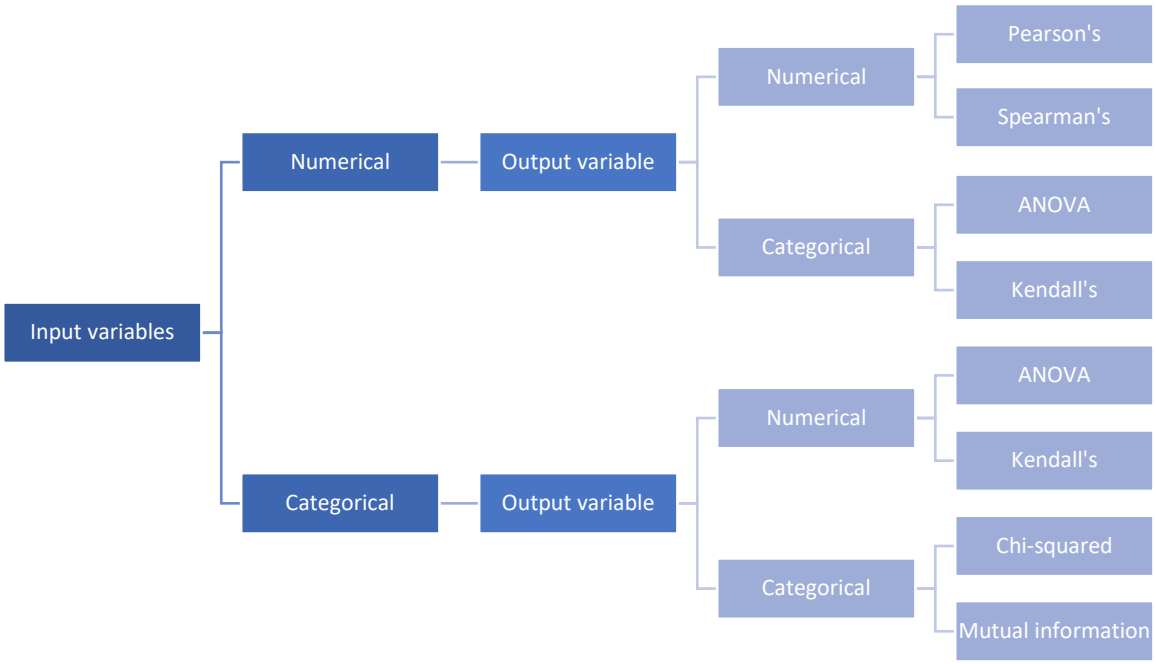

Figure 5: Statistical measure for filter-based feature selection

In this case, the target variable is numerical, and the dataset contains different input features. For numerical features, Spearman's correlation was used, and the results of the highly correlated features are:

- Severity level and mortality level with a correlation value of 0.6
- Severity level and number of diagnoses with a correlation value of 0.6
- Mortality level and number of diagnoses with a correlation value of 0.6
- Charlson index and Elixhauser index with a correction value if 0.8

3.2.5.2. Wrapper methods

These methods are based on algorithms and consider selecting a set of features as a search problem. A subset of features is used to train a model, and based on the resulting performance of the applied learning algorithm, features and added or removed from the subset that will be used to train the algorithm again. These methods are usually computationally costly and are subject to overfitting. Some common examples of wrapper methods are forward feature selection, backward feature elimination and recursive feature elimination (RFE), that was used in this model.

With the number of features to select equal to 4, RFE selected these variables:

- Severity_level
- Mortality_level
- Number_diagn

- Age_rounded

3.2.5.3. Embedded methods

These methods encompass the advantages of both filter and wrapper methods. These methods are iterative; they evaluate each iteration of the model training process and optimally find essential features that contribute the most to training in a particular iteration. Generally, they use built-in penalisation functions to reduce overfitting, and this is usually implemented by using a sparsity regulariser or constraint, which decreases the weight of some features to (near) zero.

The most famous examples of these methods are LASSO and RIDGE regression.

The example that was used to implement embedded methods, in this case, was LASSO which picked six important features as shown in Figure 6:

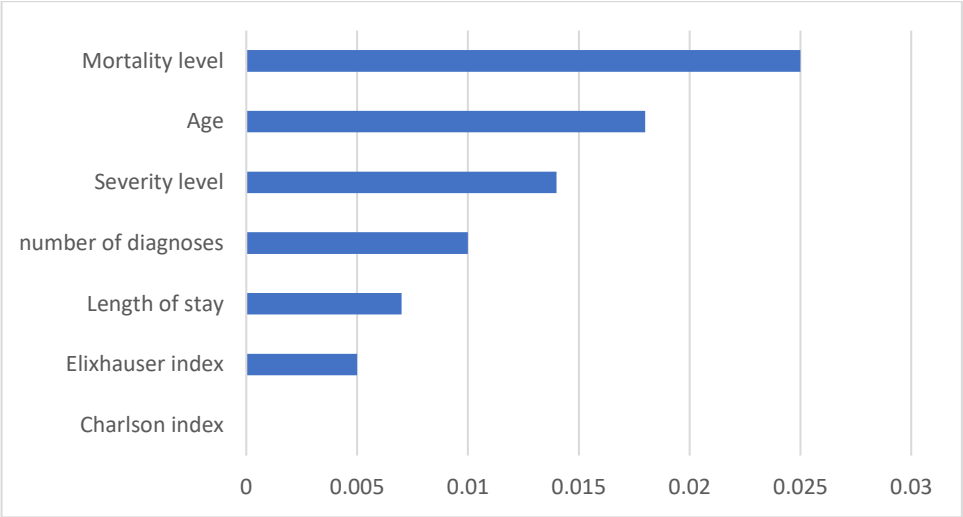

Figure 6: Feature importance using the LASSO model

After training the model with different subsets of variables based on the mentioned techniques, the subset that gave the best results contained these ten features: mortality_level, los, number_diagn, elixhauser, age_rounded, district, mdc, disch_status_inpat, gender, diagn_cat

3.3. MODELLING

The dataset prepared in the previous step is used to create the classification model. Several experiments will be taken to select the best model based on the performance metrics. These experiments tested various input variables subsets and different class imbalance handling methods. Moreover, for the modelling part, two model validation processes were performed with many classification algorithms. This section describes all the techniques, processes and algorithms used in the modelling phase and the metrics used to evaluate the performance of the models.

3.3.1. Class imbalance handling

Most real-world classification problems have an imbalanced class distribution, such as fraud detection, spam detection, and churn prediction. This problem occurs when there is an unequal distribution between the known classes. As most of the machine learning algorithms used for classification assume an equal number of examples for each class, classifiers tend to be overwhelmed by the majority of classes and ignore the small ones when faced with an imbalanced dataset. Therefore, the performance decreases significantly, specifically for the minority positive class. This is a severe problem because the classification errors will affect the minority class, which is more important in most cases.

The imbalance may be slight or severe depending on the ratio of the positive class over the negative one. A slight imbalance can often be negligible and treated like a normal case; however, a severe one should be carefully treated by implementing specialised techniques. In this case, the ratio is

Various strategies exist, but the main ones are:

3.3.1.1. Over-sampling

This technique synthesises new examples from the existing examples in the minority class. The most popular approach is called Synthetic Minority Over-sampling Technique (SMOTE). SMOTE works by selecting close examples in the feature space, drawing a line between them, and then creating a new sample at a point along that line.

Some extensions to SMOTE are more efficient in selecting the examples from the minority class that represent the basis for generating new synthetic examples. Borderline-SMOTE and Borderline-SMOTE SVM are popular extensions to SMOTE that involve selecting the instances of the minority class that are misclassified using algorithms such as k-nearest neighbour or SVM. Another approach called Adaptive Synthetic Sampling (ADASYN) generates synthetic samples inversely proportional to the density of the examples in the minority class in the feature space.

3.3.1.2. Under-sampling

In this technique, examples from the majority class are deleted. The disadvantage of this technique is that it reduces the number of records in the dataset and may delete helpful examples from the majority class. Random Undersampling is the most straightforward technique that randomly removes samples from the majority class. On the other hand, other techniques focus on which events to keep rather than which ones should be deleted. For example, the Condensed nearest neighbours (CNN) undersampling method keeps, in a "store", all the examples in the minority class and, using the KNN algorithm, only keep examples from the majority set that cannot be classified correctly by the current contents of the store.

More efficient approaches focus on which events to delete are Tomek Links method, where cross-class nearest neighbours pairs called Tomek links are created based on Euclidean distance. Then, all the examples in the majority class closest to the minority class are removed from the dataset, which will not be balanced along the classes, only less ambiguous. Another approach for finding and removing noisy examples is called Edited Nearest Neighbors (ENN), which uses the three-nearest neighbour rule to identify the misclassified examples. Then, the majority class instances among the neighbours are removed. Moreover, combinations of Keep and Delete under-sampling techniques were developed, such as One-sided selection (OSS) that combines Tomeks Links and CNN and Neighborhood cleaning rule (NCR) that combines CNN and ENN.

3.3.1.3. Combining Data Undersampling and Oversampling

Another possible strategy is to manually combine both techniques by defining specific oversampling and undersampling in a pipeline or using pre-defined combinations such as the combination of SMOTE with Tomek Links undersampling and SMOTE with Edited Nearest Neighbors undersampling.
In this specific project, the class imbalance had a ratio of 17:73. Both pre-defined combinations were performed as well as manual approaches where ADASYN and SVMSMOTE were used for over-sampling and Random undersampling or Tomek Links technique for under-sampling.

3.3.2. Classification algorithms

Several classification algorithms were experimented to select the model with the best performance, namely RF, Adabosst, XGBoost..

Random Forest is a popular machine learning algorithm based on ensemble learning. Ensemble methods combine multiple weak learners, such as decision trees, to produce a stronger learner, showing a better predictive performance than could be obtained using any of the constituent learning algorithms alone. Random forest is a typical application of one of the ensemble methods called bagging. Bagging or Bootstrap Aggregation creates a different training subset from the original training data with replacement. Each subset is independently used to train a model and generate a result. Then the final output is based on majority voting. Random forest follows this process using multiple decision trees as a learning algorithm—generally, the greater the tree number, the higher the accuracy and the lower the risk of overfitting.

Ada-boost or Adaptive Boosting is another algorithm based on ensemble learning. Unlike RF, AdaBoost follows boosting approach where models are trained sequentially and, in each iteration, the weights are improved and updated to train the next model in the sequence, aiming to correct the prediction

errors made by the prior model. AdaBoost is one of the first successful boosting approaches, and it also uses decision trees as a constituent learning algorithm.

Extreme Gradient Boosting or XGBoost is a popular and efficient open-source library that implements gradient-boosted trees algorithm. Gradient boosting is an extension of boosting that uses a gradient descent algorithm in additively generating weak models to minimise the loss. XGBoost is a scalable and highly accurate implementation of gradient boosting that pushes the limits of computing power for boosted tree algorithms. It is known to be highly efficient and effective in terms of computational speed and performance. Some studies showed that XGBoost is almost always faster than the other benchmarked implementations.

Multi-layer Perceptron Classifier (MLP) is a neural network method. It is a fully connected multi-layer feedforward artificial neural network composed of neurons called perceptions, an input layer, an output layer and hidden layers. Each perception, apart from the input nodes, has a nonlinear activation function. An MLP uses backpropagation for training the network. MLP is widely used for solving problems that require supervised learning and research into computational neuroscience and parallel distributed processing. Applications include speech recognition, image recognition and machine translation.

3.3.3. Model validation processes

After Model Training, it is required to carry out Model validation using a testing data set. Model validation determines whether the trained model is trustworthy and performs as expected.
The testing data may or may not be a chunk from the same dataset that is used to build the model
There are various techniques of model validation, among which the two most known methods are Cross Validation and train test split, both tested to validate the model in this thesis.

3.3.3.1. Train/Test Split

The most basic technique of Model Validation is to perform a train/validate/test split on the data. The model is first trained with the training set. Then, the results are validated, and the hyperparameters are tuned using the validation. This process is repeated until a good performance metric is reached. Once this stage is completed, the model is tested with the remaining test set to predict and evaluate the performance.

3.3.3.2. K-fold cross-validation

This technique is used to avoid losing valuable data on the validation set and preserve as much as possible for the training step. Instead of giving up the validation set to be used only for testing, in this

approach, the dataset is randomly divided into *k* number of folds of approximately equal size, where one fold will be used as the test set, and the remaining k-1 folds will be used as the training dataset. This will be repeated K number of times specified by the user. The final result of k-fold cross-validation in a classification problem is the average score of the results of each performance metric from all repetitions.

A repeated k-fold cross validation exists where after completing the above-explained process, it will be fully repeated *k* number of times.

In our case, a repeated k-fold Cross Validation method with the following parameters was finally selected as a model validation process:

- Repeated stratified k-Fold with 10 splits and 3 repeats
- Scoring = accuracy, F1 score, recall and precision

3.3.4. Performance metrics

Test datasets are used to determine the model's effectiveness using some evaluation metrics. The metric choice depends on the problem type because classification models are evaluated using different metrics from regression models. Further, in classification, the nature of the problem, whether it is a recommender, spam detection, client churn or the presence of a disease, will determine which metric to focus more on.

The most popular metrics used to evaluate the performance of classification models are Confusion matrix, Accuracy, Presison, Recall and F1-score (Sunasra Mohammed, 2017).

3.3.4.1. Confusion matrix

The Confusion matrix is an intuitive and easy tool for finding the correctness and accuracy of a classification model where the output can be of two or more classes. It is not an actual performance measure, but the numbers inside it are the basis of almost all performance metrics (Sunasra Mohammed, 2017). The confusion matrix is a table with two dimensions ("Actual" and "Predicted") and sets of "classes" in both dimensions. The Actual classifications are columns, and the Predicted ones are rows (Sunasra Mohammed, 2017) (Figure 7).

		Actual values	
		Positive	Negative
Predicted values	Positive	TP	FP
	Negative	FN	TN

Figure 7: Confusion matrix

True Positives (TP): the actual class of the data point was 1(True), and the predicted is also 1(True)

True Negatives (TN): the actual class of the data point was 0(False), and the predicted is also 0(False

False Positives (FP): the actual class of the data point was 0(False), and the predicted is 1(True).

False Negatives (FN): the actual class of the data point was 1(True), and the predicted is 0(False).

Model errors result in False Positives and False Negatives. The decision of which error should be minimised depends on the business needs and problem context. Based on that, it might be better to minimise either False Positives or False negatives.

For example, in a spam detection problem, it is more important to minimise false positives than false negatives because classifying an email as spam while it is not worse than classifying a spam email as essential or not spam where it is spam.

However, in our case, it is more important to minimise false negatives than false positives because classifying a patient who will return to ED as a negative case is worse than classifying a patient who will not return as a positive case.

3.3.4.2. Accuracy

Accuracy in classification problems is the number of correct predictions made by the model over all kinds of predictions made (Joshi Renuka, 2016).

Accuracy is a relevant measure when the target variable classes in the data are nearly balanced, and it should never be used in case of a class imbalance.

$$Accuracy = TP + TN/TP + FP + FN + TN$$

Equation 3: Accuracy equation

3.3.4.3. Precision

Precision is the ratio of correctly predicted positive observations over the total predicted positive observations (Joshi Renuka, 2016).

$$Precision = TP/TP + FP$$

Equation 4: Precision equation

3.3.4.4. Recall (Sensitivity)

The recall is the ratio of correctly predicted positive observations and overall positive observations in the actual class. Recall answers the question, "Of all the truly positive observations, how many did we label correctly?" (Joshi Renuka, 2016).

$$Recall = TP/TP + FN$$

Equation 5: Recall equation

3.3.4.5. F1 score

The F1 score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. It is not as intuitive as accuracy, but it is usually more helpful, especially in the cases of an uneven class distribution (Joshi Renuka, 2016).

$$F1\ Score = 2 * (Recall * Precision) / (Recall + Precision)$$

Equation 6: F1 score equation

4. RESULTS AND DISCUSSION

Different undersampling and oversampling techniques were used to handle the class imbalance problem. For oversampling, the most performing approach was SVM SMOTE. Then, both random undersampling and Tomek Links were tested for undersampling.

	Accuracy		F1 score		Recall		Precision	
	Train	Test	Train	Test	Train	Test	Train	Test
SVM SMOTE and Random Undersampler	0. 838	0. 799	0. 815	0. 769	0. 713	0. 670	0. 949	0. 903
SVM SMOTE and Tomek Links	0.887	0. 876	0. 877	0. 862	0. 786	0. 770	0. 991	0. 979

Table 6: Performance measure using two under-sampling techniques

Table 6 shows that using SVM SMOTE as the oversampling technique, Tomek Links improved approximately 10% in all the performance measures compared to the random Undersampler. Thus, Tomek Links was selected as the undersampling method to remove the ambiguous instances from the majority class.

The combination of SVM SMOTE with Tomek links resulted in a nearly balanced dataset and an augmentation of the data with synthesised instances done by SVM SMOTE. Table 7 summarises the class distribution of the records before and after handling the class imbalance.

	Total number of records	Class 1 records	Class 0 records
Before	93 449	17 971	75 477
After	146 134	73 794	72 340

Table 7: Class distribution before and after class imbalance handling

The classification models explained in the previous chapter were tested using the ten selected variables and the target as input features to select the best model. Considering the default threshold of 0.5, Table 8 summarises the cross-validated average performance measures across all folds.

	Accuracy		F1 score		Recall		Precision	
	Train	Test	Train	Test	Train	Test	Train	Test
MLP	0.6764	0.675	0.689	0.6877	0.7102	0.7088	0.6693	0.668
Adaboost	0.8648	0.8644	0.8486	0.8481	0.7501	0.7495	0.9768	0.9764
XGBoost	0.8873	0.8755	0.8754	0.862	0.7838	0.77	0.991	0.979
Random Forest	0.7257	0.7248	0.734	0.7332	0.7495	0.7486	0.7192	0.7184

Table 8: Performance measures using different classification models

As verified in Table 8, the XGBoost Classifier yielded the best scores in terms of all metrics and Table 9 shows the hyperparameters used for this model.

Hyperparameter	Explanation	Value
n_estimators	Number of gradient-boosted trees	1000
max_depth	Max tree depth for base learners	4
reg_lamba	Regularization term on weights	2
learning_rate	Boosting learning rate	0.2
random_state	Random number seed	123

Table 9: XGBoost hyperparameters

		Actual values	
		Positive	Negative
Predicted values	Positive	71864	476
	Negative	16094	57 700

Figure 8: Confusion matrix using XGBoost

According to the confusion matrix results (Figure 8), false positives are more significant than false negatives. Thus, the model makes more mistakes predicting actual positive values than negative ones. Furthermore, this indicates that accuracy values do not give an accurate indication of the model's usefulness. The accuracy of 87.55% may indicate that the model is performing well, but considering the recall of 77% and the number of false negatives in the confusion matrix, it can be noticed that the model has complications in correctly predicting the positive class.

Since this specific medical case study requires maximising the ability of the model to correctly identify patients that will return to ED (class 1), false negatives should be reduced as much as possible; thus, a relatively higher recall score is preferred.

The performance of a model significantly depends on the value of hyperparameters passed to it. A first attempt to improve the results was made by tuning the model's hyperparameters using a 10-fold cross-validated grid search technique (GridSearchCV). Table 10 presents the hyperparameters an the corresponding set of values that were defined and passed to the GridsearchCV function. GridsearchCV evaluates the model with all the possible combinations using the cross-validation method.

Hyperparameter	Values
n_estimators	{1000, 1500, 2000}
max_depth	{3, 4, 5}
reg_lamba	{2, 5, 8}
learning_rate	{0.1, 0.2, 0.3}

Table 10: Hyperparameters passed to GridSearchCV

Since the objective is to reduce the false negatives and increase recall, GridsearchCV was performed with a refit equal to recall. After running the Gridsearch, the function "best_params_" displays the best combination of hyperparameters that optimise the recall (Table 11).

Hyperparameter	Value
n_estimators	1500
max_depth	5
reg_lamba	5
learning_rate	0.1

Table 11: Optimal hyperparameters

The model with optimised hyperparameters was evaluated using the repeated 10-fold cross-validation, and the average values across all folds for test and training are reported in Table 12. Table 12 shows that tuning the hyperparameters slightly improved the model's performance in terms of recall, and this is confirmed by the confusing matrix (Figure 9) that indicates a slight decrease in the value of false negatives.

	Accuracy		F1 score		Recall		Precision	
	Train	Test	Train	Test	Train	Test	Train	Test
XGBoost	0.8895	0.8759	0.8784	0.8630	0.7886	0.7721	0.9913	0.9781

Table 12: Performance measures using tuned XGBoost

		Actual values	
		Positive	Negative
Predicted values	Positive	71438	517
	Negative	15743	58051

Figure 9: Confusion matrix using XGboost after hyperparameters tuning

Since tuning the hyperparameters did not improve the model recall as expected, an additional optimisation method was considered by tuning the model's decision threshold. A discrimination threshold visualisation (Figure 10) was plotted with the help of the Yellowbrick library that provides DiscriminationThreshold class to which the XGBoost model with the optimised hyperparameters was passed.

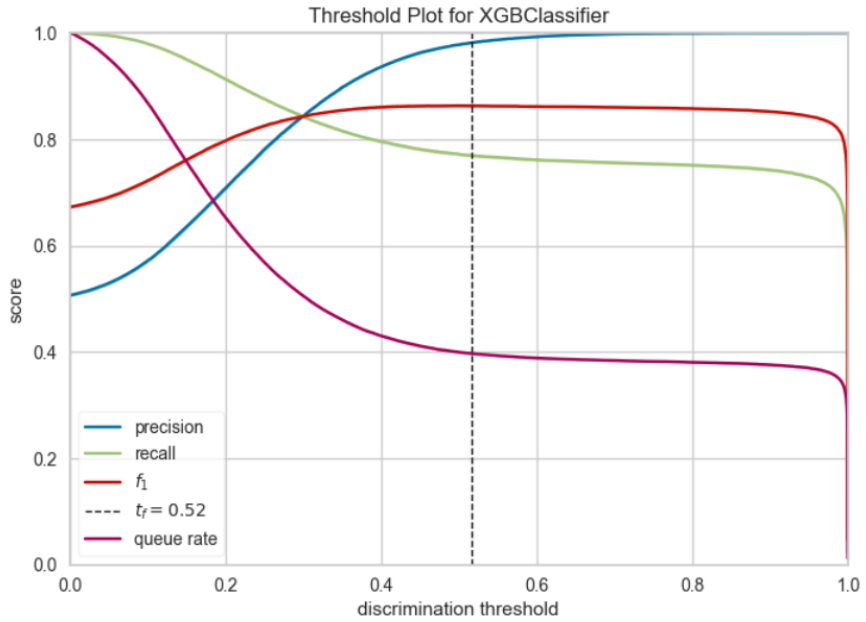

Figure 10: Discrimination threshold chart

As shown in the legend of the discrimination threshold chart, the blue, green and red lines correspond to the performance metrics. The dotted line is plotted where F1 is maximised, meaning that the balance between precision and recall is ideal (threshold = 0.52).

Figure 10 also shows that when the threshold value increases, the precision increases and recall decreases. Consequently, depending on the requirement of the problem, a trade-off between precision and recall should be made to select the best threshold. Since the objective of this particular case is to reduce the false negatives, the optimal threshold value is the one that increases the recall score.

The Threshold function from the library scikit-lego's meta models was used to embed the threshold into the model and evaluate its performance. Meta models in scikit-lego can "decorate" an estimator, and the Thresholder function helps move the prediction's threshold value.

	Accuracy		F1 score		Recall		Precision	
	Train	Test	Train	Test	Train	Test	Train	Test
0.20	0.8019	0.7646	0.8295	0.7978	0.9482	0.9137	0.7372	0.7080
0.30	0.8791	0.8413	0.8806	0.8432	0.8804	0.8425	0.8807	0.8439
0.52	0.8873	0.8760	0.8761	0.8633	0.7842	0.7706	0.9926	0.9814

Table 13: Threshold moving impact on performance measures

Table 13 shows the performance evaluation of the XGBoost model using different threshold values and that the values of measures depend on the threshold value embedded in the model. A lower threshold results in a high recall and a low precision, and vice versa. The ideal threshold equal to 0.52 given by figure 10 yielded a good F1 score and precision but not the best recall because this is the point that only optimises the harmonic mean between the recall and precision. Although the highest recall was

given when the threshold value was equal to 0.2, to increase recall and maintain a good enough precision score, it was decided to fix the threshold at 0.3.

		Actual values	
		Positive	Negative
Predicted values	Positive	62357	9049
	Negative	9076	64718

Figure 11: Confusion matrix using XGboost after threshold tuning

The performance of the final model after tuning the threshold value is shown in the final confusion matrix (Figure 11). The considerable decrease in the number of false negatives and the increase in the number of false positives reflects the trade-off between precision and recall scores.

5. CONCLUSION, LIMITATION AND FUTURE WORK

5.1. CONCLUSION

After hospital discharge, patients face a risk of complications that can lead to a return to the ED shortly. Since these returns are costly and decrease the healthcare quality, it is becoming vital to evaluate the risk of returning to ED after discharge, as it would reduce the number of missed critical illnesses and help clinicians identify patients who might need more post-discharge care to prevent their return.
The objective of this project was to reduce emergency returns costs and improve the efficiency of emergency department utilisation by identifying patients' return to the emergency department 30-days post inpatient hospital discharge. A classification machine learning model was built using historical real-word data provided by three different hospitals in Portugal.

This study was conducted on a cohort of adult patients with medical admissions, and the time scope was considered before the COVID-19 pandemic. Admission-level data, including patient demographic details and inpatient information, along with diagnosis-level data, were used to build the model.

The methodology followed to develop this project was divided into three sections. The first section was the data understanding, where the different data sources were explored. Then, pre-processing data tasks were implemented to prepare the dataset for posterior modelling and evaluation phases. This was achieved by integrating all data sources, cleaning the resulting dataset from outliers, and missing values, scaling numerical features, encoding categorical variables, and performing feature selection techniques. Lastly, different approaches to handle class imbalance were conducted, and four classification algorithms were trained and evaluated using Repeated ten-folds Cross-Validation. Accuracy, F1 score, recall and precision were passed to the cross-validation process as model evaluation metrics. According to the surveyed literature, the models selected to be tested were mainly tree-based ensemble learning techniques, more specifically, Random Forest, Adaboost and Xgboost classifiers. Multi-layer perceptron classifier was also used but underperformed compared to the other models. According to obtained experimental results, it was shown that Xgboost outperforms most of the remaining tested classification algorithms for all evaluation metrics.

Although the model gave good results, especially in accuracy and precision, the confusion matrix showed difficulties in correctly predicting the positive class. Since for this problem sensitivity measure should be maximised, hyperparameters and threshold tuning techniques were performed to optimise the recall score. The resulting optimised hyperparameters from GridsearchCV slightly improved the averaged cross-validated recall in the test set from 77% to 77.21% and after testing different threshold values, it was decided to fix the value to 0.3, where recall was increased, and precision still had a good

score. In conclusion, the chosen model was an XGBoost classifier with tuned hyperparameters and the defined threshold value of 0.3, and it resulted in a recall of 84.38%, precision of 84.35%, F1 score of 84.36% and an accuracy of 84.10%.

5.2. LIMITATIONS

The data used in this study has a before-pandemic scope since discharge dates considered are between January 2018 and December 2019. A model using after-pandemic data could be more relevant in predicting the return of new patients.

Regarding hyperparameter tuning, GridSearchCV was performed with a limited set of hyperparameters and values because it is time-consuming and computationally expensive. However, better model performance could be achieved if GridSearchCV was fed with more hyperparameters and more possible values that could yield better hyperparameter combinations. Also, other classification models could be passed to GridSearchCV to more efficiently select the optimal hyperparameters and the best model, leading to a better model.

In the modelling phase of the project, only machine learning algorithms were tested. However, considering the large amount of data collected, a deep learning model using configured layer-by-layer Artificial Neural Network could be approached.

5.3. FUTURE WORK

During the target creation process, the number of times that the patient returned to the emergency department the following month after discharge was calculated. In future work, this numerical variable could be used as a target variable to transform the problem from a binary to a multiclass classification model. in this case, the added value would be that the model would pass from prediting whether the patient will return to the ED or not to predicting, how many times he/she will come back.

Moreover, this study could be extended to include surgical admissions and pediatric patients. As mentioned in the literature review, most studies focus on a specific kind of admission, and surgical admissions receive less attention. Therefore, a predictive study of emergency returns that includes medical and surgical admissions could be very useful in the health sector.

REFERENCES

Artetxe, A., Beristain, A., Graña, M., & Besga, A. (2017). Predicting 30-Day Emergency Readmission Risk. *International Joint Conference SOCO'16-CISIS'16-ICEUTE'16, Advances in Intelligent Systems and Computing 527*. https://doi.org/10.1007/978-3-319-47364-2_1

Grant, R. W., Charlebois, E. D., & Wachter, R. M. (1999). Risk Factors for Early Hospital Readmission in Patients with AIDS and Pneumonia. *Journal of General Internal Medicine*, *14*(9), 531. https://doi.org/10.1046/J.1525-1497.1999.08157.X

Krumholz, H. M., Parent, E. M., Tu, N., Vaccarino, V., Wang, Y., Radford, M. J., & Hennen, J. (1997). Readmission After Hospitalization for Congestive Heart Failure Among Medicare Beneficiaries. *Archives of Internal Medicine*, *157*(1), 99–104. https://doi.org/10.1001/ARCHINTE.1997.00440220103013

Mišić, V. v., Gabel, E., Hofer, I., Rajaram, K., & Mahajan, A. (2020). Machine Learning Prediction of Postoperative Emergency Department Hospital Readmission. *Anesthesiology*, *132*(5), 968–980. https://doi.org/10.1097/ALN.0000000000003140

Marcantonio, E. R., McKean, S., Goldfinger, M., Kleefield, S., Yurkofsky, M., & Brennan, T. A. (1999). Factors associated with unplanned hospital readmission among patients 65 years of age and older in a medicare managed care plan. *American Journal of Medicine*, *107*(1), 13–17. https://doi.org/10.1016/S0002-9343(99)00159-X

Phillips, R. S., Safran, C., Cleary, P. D., & Delbanco, T. L. (1987). Predicting emergency readmissions for patients discharged from the medical service of a teaching hospital. *Journal of General Internal Medicine*, *2*(6), 400–405. https://doi.org/10.1007/BF02596366

Bottle, A., Aylin, P., & Majeed, A. (2006). Identifying patients at high risk of emergency hospital admissions: a logistic regression analysis. *Journal of the Royal Society of Medicine*, *99*(8), 414. https://doi.org/10.1258/JRSM.99.8.406

Billings, J., Dixon, J., Mijanovich, T., & Wennberg, D. (2006). Case finding for patients at risk of readmission to hospital: development of algorithm to identify high risk patients. *BMJ (Clinical Research Ed.)*, *333*(7563), 327–330. https://doi.org/10.1136/BMJ.38870.657917.AE

Smith, D. M., Giobbie-Hurder, A., Weinberger, M., Oddone, E. Z., Henderson, W. G., Asch, D. A., Ashton, C. M., Feussner, J. R., Ginier, P., Huey, J. M., Hynes, D. M., Loo, L., & Mengel, C. E. (2000). Predicting non-elective hospital readmissions: A multi-site study. *Journal of Clinical Epidemiology*, *53*(11), 1113–1118. https://doi.org/10.1016/S0895-4356(00)00236-5

Hasan, O., Meltzer, D. O., Shaykevich, S. A., Bell, C. M., Kaboli, P. J., Auerbach, A. D., Wetterneck, T. B., Arora, V. M., Zhang, J., & Schnipper, J. L. (2009). Hospital Readmission in General Medicine Patients: A Prediction Model. *J Gen Intern Med*, *25*(3), 211–220. https://doi.org/10.1007/s11606-009-1196-1

van Walraven, C., Dhalla, I. A., Bell, C., Etchells, E., Stiell, I. G., Zarnke, K., Austin, P. C., & Forster, A. J. (2010). Derivation and validation of an index to predict early death or unplanned readmission after discharge from hospital to the community. *CMAJ : Canadian Medical Association Journal = Journal de l'Association Medicale Canadienne*, *182*(6), 551–557. https://doi.org/10.1503/CMAJ.091117

van Walraven, C., Wong, J., & Forster, A. J. (2012). LACE+ index: extension of a validated index to predict early death or urgent readmission after hospital discharge using administrative data. *Open Medicine*, *6*(3), 90. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3659212/pdf/OpenMed-06-e80.pdf

Chmiel, F. P., Burns, D. K., Azor, M., Borca, F., Boniface, M. J., Zlatev, Z. D., White, N. M., Daniels, T. W. V., & Kiuber, M. (2021). Using explainable machine learning to identify patients at risk of reattendance at discharge from emergency departments. *Scientific Reports 2021 11:1*, *11*(1), 1–11. https://doi.org/10.1038/s41598-021-00937-9

Davazdahemami, B., Peng, P., & Delen, D. (2022). A deep learning approach for predicting early bounce-backs to the emergency departments. *Healthcare Analytics*, *2*. https://doi.org/10.1016/J.HEALTH.2022.100018

Anne Elixhauser, Claudia Steiner, D. Robert Harris, & Rosanna M. Coffey. (1998). Comorbidity Measures for Use with Administrative Data on JSTOR. *Medical Care*, *36*, 8–27. https://www.jstor.org/stable/3766985

Charlson, M. E., Pompei, P., Ales, K. L., & Mackenzie, C. R. (1987). A NEW METHOD OF CLASSIFYING PROGNOSTIC COMORBIDITY IN LONGITUDINAL STUDIES: DEVELOPMENT AND VALIDATION. *J Chron Dis*, *40*(5), 373–383.

Chang, H. J., Chen, P. C., Yang, C. C., Su, Y. C., & Lee, C. C. (2016). Comparison of elixhauser and charlson methods for predicting oral cancer survival. *Medicine (United States)*, *95*(7), e2861. https://doi.org/10.1097/MD.0000000000002861

Carl van Walraven, Peter C. Austin, Alison Jennings, Hude Quan, & Alan J. Forster. (2009). A Modification of the Elixhauser Comorbidity Measures into a Point System for Hospital Death Using Administrative Data. *Medical Care*, *47*, 626–633. https://www.jstor.org/stable/pdf/40221931.pdf

Brownlee Jason. (2020). *Data Preparation for Machine Learning: Data Cleaning, Feature Selection, and Data Transforms in Python.* Machine Learning Mastery. https://machinelearningmastery.com/data-preparation-for-machine-learning/

Joshi Renuka. (2016, September 9th). *Accuracy, Precision, Recall & F1 Score: Interpretation of Performance Measures - Exsilio Blog*. https://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/

Sunasra Mohammed. (2017, November 11th). *Performance Metrics for Classification problems in Machine Learning | by Mohammed Sunasra | Medium*. https://medium.com/@MohammedS/performance-metrics-for-classification-problems-in-machine-learning-part-i-b085d432082b

Salgado, R., Moita, B., & Lopes Id, S. (2022). Frequency and patient attributes associated with emergency department visits after discharge: Retrospective cohort study. *PLOS ONE*, *17*(10). https://doi.org/10.1371/JOURNAL.PONE.0275215

APPENDIX

Charlson index: The first version of this index was developed in 1987 with 19 categories (Charlson et al., 1987). Each comorbidity category was assigned a weight (from 1 to 6) based on the adjusted risk of mortality or resource use. The sum of all the weights results in a single comorbidity score for a patient, and a score of zero indicates that no comorbidities were found. The higher the score, the higher risk of mortality or resource use (Chang et al., 2016; Charlson et al., 1987). Over time, adaptations have been performed, such as the modifications to 17 categories, the translation from ICD-9-CM codes to ICD-10-CM codes, and the modification of the original weights(Chang et al., 2016).

Elixhauser index: The original Elixhauser comorbidity measure was developed in 1998 with 30 categories (Anne Elixhauser et al., 1998) where each comorbidity category is dichotomous - it is either present or it is not- and the system required 30 binary variables (Carl van Walraven et al., 2009). Initially, comorbidities were not simplified as an index because each comorbidity affected outcomes differently. Later, a set of weights was developed, based on the association between comorbidity and death, to summarise disease burden and produce an overall numeric score called Elixhauser Index (Carl van Walraven et al., 2009).

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação

Universidade Nova de Lisboa