Examining Hospital Readmissions

Evaluation of Machine Learning-Based Predictive Models

Names	Numbers	Group 28
Catarina Alexandre dos Reis Coelho	20230374	
Catarina Ferreira Rodrigues	20230435	
David Psiuk	20230818	
Dzmitry Nisht	20230776	
Pedro Miguel Silva Carvalho	20230487	

22nd of December of 2023

Abstract

Hospital readmissions are a significant burden in healthcare, indicating both poor care quality and high costs. The process entails using Machine Learning (ML) techniques to examine patient data and find trends that point to impending readmissions. To address this issue, this study takes a twofold approach.

Firstly, a binary classification model is created to predict whether a patient will be readmitted within 30 days. This will help healthcare practitioners implement preventive measures to reduce the likelihood of readmission within this timeframe.

Secondly, a multiclass classification model is developed to categorize readmissions into "No", "<30 days", and ">30 days", providing a deeper understanding of patient risk to be readmitted and allowing hospitals to adjust follow-up procedures.

Ten ML models were trained on a dataset consisting of 71,236 unique encounters using a subset of selected features, varying according to the procedures used to select them, the model in which they were trained on, and the classification problem in hand.

The findings indicate that different pre-processing techniques affect the performance of ML models, suggesting the need for a customized strategy. Age, Primary Diagnosis, and Discharged Disposition were found to be among the most predictive variables. The most influential feature for the best-performing model was the patient´s frequency of occurrences.

The performance metrics of the best model, Random Forest, align with findings reported in the literature. After tuning, the model achieved an accuracy of 0.7453 and 0.7096 for training set, 0.7225 and 0.6740 for validation set, precision of 0.2322 and 0.6443, recall of 0.6444 and 0.6740, F1-score of 0.3414 and 0.6993 for training set, and 0.4054 and 0.6501 for validation set, for binary and multiclass classification problems, respectively.

Despite the potential for overfitting, this study provides a viable approach to leverage ML models for predicting readmissions in clinical practice following this two-step approach.

Index

1. Introduction

Our two main goals for this project are to predict hospital readmissions within a 30-day time frame, and the period of hospital readmission, which includes no readmission, readmitted within the next 30 days, or more. We compare the performance between 10 Machine Learning (ML) algorithms: Linear Regression, Logistic Regression, K-Nearest Neighbours, Naïve Bayes, Neural Networks, Support Vector Machine, Decision Tree, Random Forest, Gradient Boosting and AdaBoost, for both problems.

Hospital readmissions have been used for several decades to improve the quality of care provision (Fischer et al., 2014), because when a patient is readmitted to the hospital, it often means that their initial treatment was ineffective, which can exacerbate their condition (Dixit, 2021); but also, for reasons related to the monitoring and control of costs in the health sector (Fischer et al., 2014), because when patients are readmitted to the hospital, they require additional medical attention and resources (Dixit, 2021).

A 2013 paper by Francisco Manuel dos Santos Foundation studies the impact of readmissions on the number of resources allocated to the hospital sector. €98,435.557 is the excess cost attributed to readmissions, which implies a cost of €2,919 per readmitted episode value, which is about 9% higher than the average cost per episode of internment. This group made an extensive literature review to find research with similar objectives than ours. Findings on selected research, namely the ML algorithms used, their comparative performance, and the performance metrics used to evaluate them, are summarized in **Table 1,** in the annex.

According to the knowledge acquired in ML classes, as well as the articles presented, the algorithm with the best results is Random Forest. Moreover, by analysing the results obtained, we can predict that, for the binary classification problem, the AUC should be between 0.66 and 0.79, the accuracy will also be close to 0.70 (in the articles it varies between 0.64 and 0.73), the precision will possibly have lower values between 0.2 and 0.3, the AUPRC will be around 0.23 and F1-score will be close to 0.4.

2. Data Exploration and Pre-Processing

To perform data exploration and preprocessing we start by preparing the dataset. We therefore replaced the target variable for the binary classification problem by 0 (“No”) and 1 (“Yes”), and for the multiclass problem by 0 (“No”), 1 (“<30 days”) and 2 (“>30 days”). For organizational purposes and according to the metadata, we set the column ‘encounter id’ as the index since it represents a unique identifier of the encounters we are analyzing.

By exploring the dataset, we concluded that the column ‘country’ is the only one which is invariable with all encounters being in the USA. Thus, we deleted that column as it will be no help to distinguish any case in the target variable. Further investigation in the dataset revealed the occurrence of the value ‘?’ in several columns. We replaced these with ‘Not a Number’ (NaN) values to assure the quantity of missing values in the dataset. **Table 2** illustrates the number of missing values for each column. We can observe that the column ‘weight’ and ‘medical spciality’ contain a high proportion of missing values. According to the metadata, there is no meaning in missing values for these features, which is why we will not consider these variables further in the project to reduce the dimensionality and noise in the dataset. Finally, we gained insights into the ranges, behaviors, and characteristics of our dataset to consider them in the next stages.

Subsequently, we explicitly defined our metric and non-metric features. We used Histograms and box plots to visualize the distribution and occurrences of outliers for the metric features (**Figures 1 and 2**, respectively). We will deal with these outliers after performing the Feature transformation. **Figure 3** visualizes the distribution within non-metric features and the cardinality of different categories. Here we can observe that the columns, which cover diagnosis, medication, payer code, admission source, admission type and discharge disposition have a high number of categories. This increases the complexity of the problem and will be dealt with ahead.

2.1. Feature Transformation/Engineering

In this sub-section, our aim was to simplify mainly non-metric variables to be of better use in our predictive models. We also focused on solving many misinterpretations of missing values to an actual category. With this in mind, we want to reduce the complexity of the dataset and avoid losing relevant information by integrating the data of missing values.

The first column we transformed was ‘payer code’ where missing values were reflecting a person with no insurance. As so, we build the binary column ‘Insurance’ with values of 0 for “No insurance” and 1 for “Insurance” to reduce complexity and summarize the prior information of the column ‘payer code’ which we then deleted.

Subsequently, we conducted a **coherence check** for the 'number diagnoses' attribute. In cases where a patient possessed only a single diagnosis, it was evident that secondary and additional diagnoses were not applicable. Similarly, for patients with exactly two diagnoses, the concept of additional diagnoses does not apply. To address this, we introduced a category labelled "Not Applicable" to replace missing values in these cases. Moreover, we created broader categories for the different diagnoses. According to our research, the ICD-9 codes fit under 19 broader categories that can help facilitate analysis (Ostling et al., 2017; Garcia-Arce et al., 2017). We have then created them, as seen in **Table 3**, sending each code to its correspondent.

Following that, for the variables ‘admission type’, ’discharge disposition’, and ‘admission source’, we assumed that ‘Not Mapped’, ‘Not Available’, and missing values all represented similar conditions, and thus, we standardized these values as ‘Not Available’. To reduce the dimensionality of ‘discharge disposition’ variable, we formed broader categories for the different destinations given to the patient after being discharged. The values “Discharged to home”, “Discharged/transferred to SNF” and “Discharged/transferred to home with home health service” are kept in their own category, since these dispositions occurred relatively often in the dataset. In addition, we consolidated discharges to other facilities into the category ‘Discharged to Other’ and grouped other disposition types under the category ‘Other’. We employed a similar approach for the ‘admission source’ column, retaining ‘Emergency Room’ and ‘Physician Referral’ due to their relatively high occurrence frequencies, while integrating all forms of transfers into the ‘Transfer from Hospital/Facility’ category and categorizing other admission types as ‘Other’. Furthermore, within the column ‘admission type’, we grouped infrequently occurring types such as “Trauma Center” and “Newborn” under the category “Not Available”.

The reason for this was to not lose information and prevent having values with little occurrences in the dataset that create noise.

Regarding the column 'gender', we have taken steps to address the presence of "Unknown/Invalid" entries by appropriately marking them as missing values. Furthermore, we transformed this attribute into a binary variable, where "Female" is represented by the value 1, and "Male" by the value 0. Next, we turned “change in meds during hospitalization” and "prescribed diabetes meds" binary, being “No” as 0 and 1 equal to "Ch"/” Yes”, respectively.

Since the goal for this project is to predict hospital readmissions, especially ones of diabetic patients, we have decided to split the column ‘medication’ according to its type: “Takes insulin”, if the patient took insulin, “Doesn’t take insulin”, if the patient didn’t take it and missing values were treated as no medication were taken. The last changes were conducted for the columns ‘glucose test result’ and ‘a1c test result’ where according to the interpretation of the metadata, missing values are replaced with a category “Not Taken”.

Additionally, we created the features ‘number lab tests per day’ and ‘patient id occurrences’. The former is a ratio between the ‘number of lab tests’ and the ‘length of stay in hospital’ to get the average number of lab tests performed per day during the hospital stay. The latter is the number of times a patient occurred in the encounters we were provided with. As we can see in **Figure 4**, this feature gives especially good insights when referring to the target variable. The proportion of being readmitted is visibly higher the more encounters the patient has.

2.2. Outliers

When looking at the initial visualizations we can observe that some features contain a significant presence of outliers. Although the standard practice is to conduct the train-test split prior to outlier treatment, we addressed outliers on the full dataset upfront due to an additional, separate test set provided for the final validation. While this deviates from the usual methodology to prevent data leakage, this approach allowed for a comprehensive understanding and consistent handling of outliers across the entire training dataset, ensuring that our data's preprocessing reflects that of the separate test set. Our strategy to treat these outliers was to **apply ceilings** to the specific features. In this manner, we avoid having to remove possibly valuable observations by keeping the information of these outliers. Especially the different types of visits in the previous year have a high number of outliers. This is due to the fact that most patients did not visit the hospital in the year preceding the encounter. The columns we applied value ceilings for outliers’ removal were: ‘outpatient visits in previous year’, 'number of medications', 'number diagnoses', ’emergency visits in previous year’, ’inpatient visits in previous year’, ’number lab tests’ a ‘number lab tests per day’ and ‘patient id occurrences’, being the top values {1,40,10,1,3,90,50,10}, respectively. In certain features like the number of encounters of a patient, we decided to take a relatively high threshold for the ceiling as these values might provide a better understanding of patients being readmitted. For non-metric features, we assumed that low cardinality categories could aid the models in predicting readmissions. **Figures 5 and 6** show all the variables after transformations and outlier treatments and visualize the significant improvement in the understandability of the data.

2.3. Missing Values

At this stage, only non-metric features contained missing values. Initially, we considered employing an imputation technique based on nearest neighbours after encoding categorical data to numerical form, but given that it uses distance metrics, this could result in improper imputations for categorical data. This recognition led us to conclude that alternative imputations that don't rely on such metrics would be more appropriate to keep the integrity of the dataset. Another method would be to use **target encoding** with the missing values. This would give these values their own category and could be useful if they are not at random. However, this approach is often criticized for the tendency to overfit due to target leakage (Groen et al., 2022).

Having said this, we replaced the few missing values in the column 'gender' with the **most frequent value** and added the missing values in the column 'race' to the category 'Other'. Furthermore, we **grouped the missing values** of the different diagnoses into the already existing category 'Not Applicable'. Thus, we don´t have to delete any rows in the dataset and by manipulating the missing values accordingly, the information in the dataset is not lost. For that reason, we have allocated the missing entries within 'discharge_disposition' and 'admission_type' to the 'Not available' category. Additionally, for the variable 'age', we addressed missing values by applying the same method as previously described, rather than imputing with a numerical value. This decision was made by the anticipation of target encoding in subsequent stages of the analysis, recognizing the potential correlation between the observability of a patient's age and their likelihood of readmission seen in literature (Welvaars et al., 2023; Du et al., 2020; Li et al., 2020; Gao et al., 2023).

2.4. Train Validation Split

Before further preprocessing, it is important to split the dataset into subsets of training and validation to avoid data leakage. In that way we will perform further encoding and scaling techniques without considering the information of the validation set to keep the integrity of the training process. From now on, we will consider the binary and multiclass cases independently. We applied a split of 25% for validation and 75% for training, ensuring that the split was stratified to maintain the same proportion of each class in validation and training (Li et al., 2020).

2.5. Target Encoding for Non-metric features

As part of our research, we were looking for a method of converting categorical features to numeric without needing to apply one hot encoding, due to the curse of multidimensionality, while also not requiring a specific order like in label encoding. We identified target encoding as an effective solution. This method encodes categories based on the average of their likelihood for each target category, thereby integrating the influence of the categorical feature on the target variable directly (Groen et al., 2022). As explained before we did this after the split to avoid data leakage of the validation mean being influenced by the mean from the training set. This allows encoding categories without increasing the data dimensionality preserving the original information of the features (Groen et al., 2022). On top of this, target encoding already leaves values between 0 and 1, which will be good for comparison purposes further ahead.

2.6. Normalization

After performing target encoding on the non-metric features, we applied normalization for the metric features to bring all features to a common scale since most of the models we use are sensitive to the scale of the input data. It is also relevant to prevent any kind of feature from being dominant in terms of impact in the model. We utilized the method of **Min-Max scaling,** as it is an effective approach by keeping all feature values between 0 and 1 as in the target encoding. For both Target Encoding and normalization, we calculated the parameters using the training set and then applied these parameters to the validation set to prevent data leakage. After completing this step, all values in the dataset ranged between 0 and 1.

2.7. Balancing Data

To complete the preprocessing phase, we focused on balancing the dataset due to its **imbalance**, since only 11% of patients were readmitted. This leads to standard classifiers such as logistic regression and decision tree being not so well trained in those categories, resulting in a decrease of precision and accuracy (Haixiang et al., 2016). Therefore, we decided to balance our data using Synthetic Minority Oversampling Technique (**SMOTE**), which resulted in a dataset of equally distributed target values. Further info on this is presented in the Annex. After this, we considered our dataset prepared for the feature selection part.

3. Binary Classification

3.1. Feature Selection

The feature selection methods we considered covered the **3 types of methods: filter, wrapper and embedded**. Starting with the **filter method**, we ensured that the used features hold variance and therefore contain possible useful information. Additionally, the **spearman's correlation** amongst the explanatory features is calculated to understand the relationship between them. The threshold of the correlations is set to |0.6| to identify features, which highly correlate with each other and therefore might be redundant in the information they provide. These features are further considered to be discarded.

The **wrapper method** applied in this project is the **recursive feature elimination** technique. It is used to identify the optimal number of features considering the F1-Score for the validation set with the pre-processed data. For timesaving and simplicity purposes this process is only performed on a Logistic Regression. The best performance of the model is achieved when using 13 Features as shown in **Graph 1**. Another wrapper method is suggested by Groen et al. (2022): **Mean Decrease Impurity.** This method ranks the features based on their importance, putting it side by side with the decrease in impurity they bring (Groen et al., 2022). The **Random Forest** algorithm is used to calculate the feature importance. To consider a feature not relevant, we used a minimum threshold of 0.01. This led us to identify 7 possible irrelevant variables, as shown in **Graph 2**. We additionally used **Lasso regression** to calculate feature importance's and determined 10 possible irrelevant features (**Graph 3**). Considering different ML models for the feature selection gives us a more robust outcome than focusing only on one specific model.

The choosing criteria is present in **Table 4.** As a way of combining all perspectives, **Table 5** summarizes all selection methods we applied. Therefore, the final set of features for the binary classification are the following: 'non lab procedures', 'number diagnoses', 'emergency visits in previous year', 'inpatient visits in previous year', 'length of stay in hospital', 'number lab tests', 'number lab tests per day', 'primary diagnosis', 'additional diagnosis', 'discharge disposition', 'admission source', 'race', 'age', 'ac1 test result', 'patient id occurrences'.

3.2. Model Selection

There are several preprocessing steps that were considered in this work. To assess whether these modifications bring additional value to the performance, the models will be evaluated on several datasets that were pre-processed in different manners. The **first dataset** that will be used to train the model contains all preprocessing steps supra explained. It is **balanced** with the oversampling technique SMOTE and **outliers are capped**. The **second dataset** contains the same information as the first, except for **outliers not being manipulated**. The reason for this is to find out if the outliers provide valuable information to the model. In the **third dataset** outliers were manipulated in the same manner as before, but it is **not balanced**. With this additional dataset we want to test the utility of the oversampling technique SMOTE, since it can potentially lead to overfitting as it creates synthetic samples which might cause the model to be too specific to these samples and not generalize well to unseen data (Chawla et al., 2002). The **fourth dataset** to evaluate is also **not balanced** with **outliers not being manipulated.**

Under imbalanced scenarios, minority class samples can easily be discarded as noise; **however**, if the irrelevant features in the feature space are removed, this risk is reduced (Haixiang et al., 2017). Especially in imbalanced datasets such as the one provided, it is important to discuss the performance metric used for the model evaluation, which is the F1 score. It is **more informative than accuracy** because it considers both precision and recall, providing a better measure of the incorrectly classified cases, particularly for the minority class. It is therefore providing a more balanced view of the performance on both classes (Jeni et al., 2013). Also, accuracy doesn't consider false negatives, and knowing the goal of this project, it can be somewhat harmful to patients because not predicting a readmission might lead to less medical resources spent on a patient trying to search for the root of the problem, which in patients that already have morbidities (mostly diabetes in this case), might cause the patient's life.

Graph 7 shows the performance of selected ML models for the four different datasets on the training and validation set. This is only the initial performance of the algorithms with **default settings**. The plot shows that there are a lot of variances between the performance on the training set and the validation set such as on the different models and datasets. We can see that non-balanced datasets perform worse than balanced ones. The models are **generally highly overfitting**, especially the Decision tree-based models don't generalize well with default settings

To get a more balanced view of the performance from the models, we will run the same iteration again with parameters used to decrease its overfitting (**Graph 8**). Thus, for the Decision Tree and the Random Forest the maximum depth of the tree is now set, so that the nodes don´t expand until all leaves are pure. Furthermore, the weight of the class is set to "balanced" which automatically adjusts the weights proportional to class frequencies in the input data. This is an important aspect considering the imbalanced dataset that is used.

For the ensemble models – Adaptive Boosting and Gradient Boosting – we reduced the maximum number of estimators to be used in the boosting process. Since the estimators in these models are trained to correct the errors from the previous week learners, a high number of estimators will make the model more complex and so we lowered the number for less overfitting. **Graph X** shows the performance of the models with the adapted arguments. As stated, the **F1 Score depends on both the model and the preprocessing steps of the dataset.** Overall, the balanced datasets still overfit a lot and outperform non-balanced datasets for most of the models. Nevertheless, the non-balanced datasets seem to perform better on the Decision Tree and the Random Forest with the adjusted settings. Moreover, the gap between the F1 Score for the training and validation set is clearly smaller, resulting in a model that generalizes better.

3.3. Tuning selected Machine Learning Models

This part focuses on tuning selected models from the model evaluation. To include a linear model in this process, the **Logistic Regression** will be used on the best performing dataset which is **balanced and containing the original outliers**. Since the **Random Forest** performs better than the Decision Tree, this Classifier will be considered on a **non-balanced dataset with outliers** being treated. The graph from the model evaluation shows that **Gradient Boosting** performs slightly better than AdaBoost. Gradient Boosting shows a higher F1 Score for **a balanced dataset with outlier treated** which is why this dataset will be considered for further investigation.

To get the best conditions for each individual model, we will perform an **additional recursive feature elimination** (RFE) procedure. The best number of features will be calculated according to the F1 Score on the validation set on the corresponding dataset for each selected model. The resulting features for each model will be used when performing **additional tuning**. This ensures that the model will be tuned on a dataset with features that give the most value to it.

Starting with the **Logistic Regression** we obtain the best performance when using **11 features** with a F1 Score of 0.3195 on the validation set. The model is relatively highly **overfitting** which is probably caused by the oversampling technique used. The same problem occurs for **Gradient Boosting** which shows no improvement for more than two features and results in a F1 Score of 0.3297 for the validation set. As we perform the RFE on a **Random Forest** with a dataset which was not balanced by SMOTE, the model **overfits less**, best performing when using **16 features** which results in a score of 0.3413 for the validation set and 0.4054 for the training set.

In the following, we will try to improve the results by performing hyperparameter tuning on each of the specific models. Therefore, we will make use of a **grid search method with cross validation**. Grid Search explores a range of hyperparameter values for a ML model and identifies the best performing combination. We will additionally use cross validation when performing hyperparameter tuning because it reduces the risk of overfitting to specific characteristics of a single data partition and therefore it provides a more reliable assessment of the performance by using multiple data subsets for training and validation (Kohavi, 1995). Furthermore, cross validation is **stratified** by default which ensures the same proportions of the target value for each subset of the data which is important especially for an imbalanced dataset.

After performing grid search and RFE on the **Logistic Regression**, the model performed better and improved from an initial score of 0.2585 to 0.3216 on the validation set. The same technique for the **Gradient Boosting** model resulted in an actual decrease of performance for the validation set. The F1 score dropped from 0.3297 to 0.2420 with the new hyperparameters. This decline is probably due to the difference in the training and validation set provoked by the oversampling technique. As we used a balanced dataset for this model, the Grid Search supposably selected hyperparameters, which are too specific to the training set and don't generalize well. Therefore, the new hyperparameters only showed progress in the training set which underlines the problem of different distributions in training and validation set created by synthetic oversampling. After applying hyperparameter tuning on the **Random Forest** algorithm with the non-balanced dataset, the validation score improved a bit from 0.3414 to 0.3444.

After evaluating differently pre-processed datasets on different ML models with corresponding feature selection and hyperparameter tuning, the **best result** was obtained on a **Random Forest** model with a **non-balanced and outlier-handled dataset.** We additionally tested different scaling techniques during preprocessing to find out if they would result in different performances. Therefore, we compared the Min-Max Scaling method with Standardization and Robust Scaling. **Graph 9** shows the performance of the best model compared with the different scaling techniques. We can observe that there are only slight differences in performance, which is why we will keep utilizing the Min-Max scaling method.

4. Multiclass Classification

4.1. Feature Selection

The feature selection for the multiclass problem tracked the same approach as the binary classification, with the only difference being the update to |0.5| on the correlation matrix. In here, we expected a larger number of variables to be selected since it has 3 distinct values on the target, so a larger blend of features might be needed. **Graphs 4**, **5, 6, and Table 6** summarize feature selection for multiclass, following the same selection criteria as the binary one.

After trying different combinations, the final set of features were: 'non lab procedures', 'number diagnoses', 'number lab tests', 'Insurance', 'primary diagnosis', 'secondary diagnosis', 'additional diagnosis', 'admission type', 'discharge disposition', 'admission source', 'race', 'gender', 'age', 'medication', 'a1c test result', 'patient id occurrences' and 'glucose test result'.

4.2. Model Selection

We applied the same methodology as for the binary classification problem. So, we generated visualizations using both default and adjusted parameters for each model across all datasets (**Graphs 10 and 11**), with the results showing no significant variances between the performance on the training and validation sets such as on the different models and datasets. A key consideration for the multiclass problem lies in analyzing the performance of the weighted F1 score because of the imbalance in the dataset. The classes may not have the same proportion, so it is important to give them different weights/importance.

Of all the models tested, which are the same as in the binary, we chose the three with the best values and different theoretical perspectives: **Logistic Regression, Adaptative Boosting** and **Random Forest**. We develop a model with balanced class weights to handle imbalanced classes, eliminate and select the optimal number of features that track of the highest F1 Score on the validation set using RFE.

Starting with the **Logistic Regression** we obtain the best performance when using **7 features** with a F1 Score of 0.5860 on the validation set. The model has a **relatively low overfitting**. For **Adaptative Boosting**, we obtain the best performance when using **16 features** which results in a score of 0.6212 for the validation set and 0.6237 for the training set. The **Random Forest** is **relatively overfitted,** even after selecting the **16 best variables**. Before applying the grid search in this model, the F1 Score shows the value of 0.7013 for the training set and the value of 0.6488 for the validation set.

4.3. Tuning selected Machine Learning Models

After performing grid search on the **Logistic Regression**, the model performed better and went from 0.5860 to 0.6076 for the F1 score on the validation set. The same technique for the **Adaptative Boosting** model resulted in an actual little increase of performance for the validation set. The initial F1 score slightly improved from 0.6212 to a score of 0.6224 with the new hyperparameters. After hyperparameter tuning on the **Random Forest** algorithm with the non-balanced dataset, the validation score improved a bit from 0.6488 to 0.6500.

Each of these models brings unique theoretical foundations and distinct algorithms, providing a more comprehensive analysis of the problem at hand. **Logistic Regression**, being linear, captures linear relationships between variables, while **Adaptative Boosting** and **Random Forest**, as ensemble-based models, explore the synergy of various models. The diversity of algorithms was a key criterion in our choice to enhance our understanding of patterns in the data.

Furthermore, when considering sensitivity to different features, each model offers a unique perspective. This diversity allows us to explore specific nuances of the problem that may be better captured by specific algorithms, enriching our analysis. The resilience to overfitting was enhanced by including ensemble-based models like **Adaptative Boosting** and **Random Forest**. These models are designed to mitigate the risk of excessive fitting, contributing to better generalization of the model to new datasets.

Exploring different hyperparameters in each model during cross-validation provided a deeper understanding of how performance is influenced by different settings. This not only improves the model fit but also reveals valuable insights into the sensitivity to these parameters.

5. Conclusion

The final section of this research aims to draw conclusions, derive final reflections, address how certain limitations influenced our work and suggest directions for future research.

The study´s alignment with the existing literature led us to anticipate that Random Forest would excel as the top-performing model. In fact, our work proved to be in line with our expectations and this was our best performing model as well, with its performance closely mirroring literature-reported results. The final model achieved an F1-score of approximately 0.343 and of 0.650 for the binary and multiclass problems, respectively.

A key consideration we could perceive based on our work is the importance of considering multiple pre-processing lines. By evaluating four differently pre-processed datasets, we obtained a timesaving and efficiency advantage that allowed us to identify the most suitable strategy to adopt for each specific model. We have learned that maintaining a clean dataset, particularly by addressing outliers, is a commendable practice. However, in an imbalanced scenario the theory changes and the effectiveness of each model largely depends on its response to the applied technique, SMOTE in our case. In this study, the F1 score served as the primary performance metric for model evaluation. This is particularly important when dealing with imbalanced datasets, as it provides a more accurate assessment of model effectiveness. Distinct from the F1-score provided by the binary models, multiclass cases compute an F1-score for each class, being why we used the overall weighted F1-score to deal with the imbalance problem. We reinforce again that as expected these scores fell within the thresholds set up by literature.

In this work, we encountered certain limitations. The first limitation is the lack of knowledge in the medicine field. By having more knowledge about the variables used in the problem or the typical behavior in these situations, the decision-making process would have been facilitated, especially both in feature-engineering and feature selection, especially on a multiclass classification problem. The second limitation was related to time. Given the complexity of the problem, a deeper and more detailed analysis of the dataset, models to implement and different techniques could have been undertaken with more time. However, we tried to opt always for the most informed and correct choices based on our knowledge.

Taking all the above into consideration, we can offer some insights and provide suggestions for future research. Unfortunately, due to time constraints, we were not able to do further exploration of the SVM model despite its potential as a suitable choice. Therefore, we suggest a more extensive analysis of this model. We also recommend further exploration of feature engineering techniques to assess whether they can potentially enhance model performance.

In summary, this research underscores the significance of thoughtful pre-processing strategies, model evaluation metrics, and the need for domain knowledge in tackling complex problems. The insights gained from this research lay the foundation for future investigations, offering opportunities to enhance predictive modelling for readmissions in clinical practice.

6. Annex

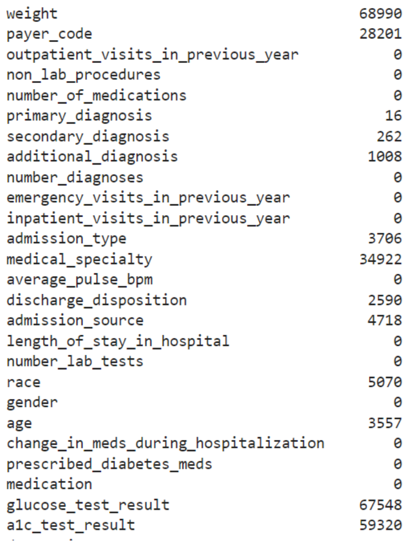
In this section are contained all figures, graphs and tables referenced along our report content.

Table 1: *Summary of similar research and associated results*

Reference	Title	Objectives	ML algorithms used	Best algorithm	Performance metrics
Garcia-Arce, A., et al. (2017	Comparison of Machine Learning Algorithms for the Prediction of Preventable Hospital Readmissions	Compared predictive models based on ML algorithms for 30-day preventable hospital readmissions of Medicare patients for 5 diseases	Random Forest (RF), neural networks (NN), Gradient Boosted Trees, Support Vector Machine (SVM)	Neural Network	Area under the curve (AUC) – between 0.65 and 0.75 among 5 diseases studied
Michailidis, P., et al. (2022)	Forecasting Hospital Readmissions with Machine Learning	Model to predict the risk of patients' readmission within 30 days of their index discharge	SVM (Linear and RBF Kernel), Random Forest (Balanced and Weighted)	Balanced Random Forest	AUC – 0.78 Accuracy – 0.73 Specificity – 0.74 Precision – 0.32 F1-Score – 0.44
Gao, X., et al. (2023)	Interpretable machine learning models for hospital readmission prediction: a twostep extracted regression tree approach	Develop a machine-learning algorithm that can predict 30- and 90- day hospital readmissions.	Logistic Regression, Decision Tree, SVM, Extremely Randomized Trees, Light Gradient Boosting Machine (LGBM), Extreme Gradient Boosting (XGB), RF, and NN	30 days – RF 90 days – LGBM[1]	For RF only: AUC – 0.69 Area under the precision-recall curve (AUPRC) – 0.23 Accuracy – 0.70
Sutter, T., et al. (2020)	A comparison of general and disease-specific machine learning models for the prediction of unplanned hospital readmissions	Compare the predictive performance of specialized and general ML models in predicting unplanned readmissions based on routine healthcare data in general and in	Random forest, Logistic Regression, LASSO Regression and Neural Network	Random Forest and LASSO Regression	AUC – 0.79 for both

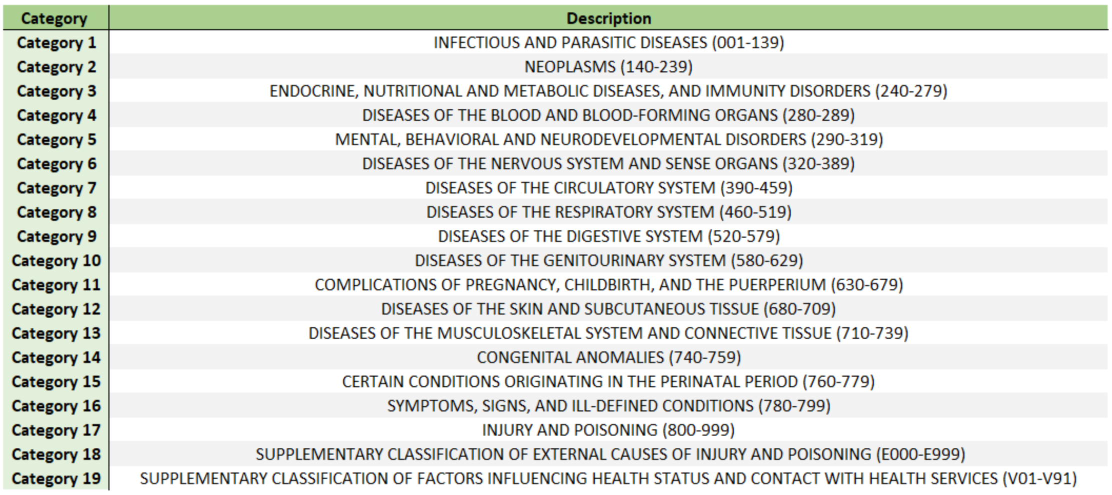
		patients discharged from medical, surgical, and gynecological departments.			
Eckert, C., et al. (2019)	Development and Prospective Validation of a Machine Learning-Based Risk of Readmission Model in a Large Military Hospital	Demonstrate the development and prospective validation of a ML risk of readmission to be utilized by clinical staff for all-cause inpatient readmissions.	Decision Tree, AdaBoost, Random Forest and Logistic Regression	AdaBoost	AUC – 0.68 Accuracy – 0.64 Precision – 0.18 Recall – 0.73
Dixit, R. R., (2021)	Risk Assessment for Hospital Readmissions: Insights from Machine Learning Algorithms	Investigate the consequences for the patients and for the healthcare system after the readmissions, using ML models.	Decision Tree, Random Forest and XGBoost	Random Forest	R-Squared- 0.98 RMSE- 0.07

Table 2: *Initial number of Missing Values present in the dataset*



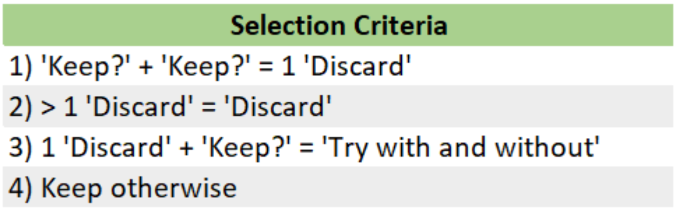
```
weight                                  68990
payer_code                              28201
outpatient_visits_in_previous_year          0
non_lab_procedures                          0
number_of_medications                       0
primary_diagnosis                          16
secondary_diagnosis                       262
additional_diagnosis                     1008
number_diagnoses                            0
emergency_visits_in_previous_year           0
inpatient_visits_in_previous_year           0
admission_type                           3706
medical_specialty                       34922
average_pulse_bpm                           0
discharge_disposition                    2590
admission_source                         4718
length_of_stay_in_hospital                  0
number_lab_tests                            0
race                                     5070
gender                                      0
age                                      3557
change_in_meds_during_hospitalization       0
prescribed_diabetes_meds                    0
medication                                  0
glucose_test_result                     67548
a1c_test_result                         59320
```

Table 3: *ICD-9 Broader Categories Description*



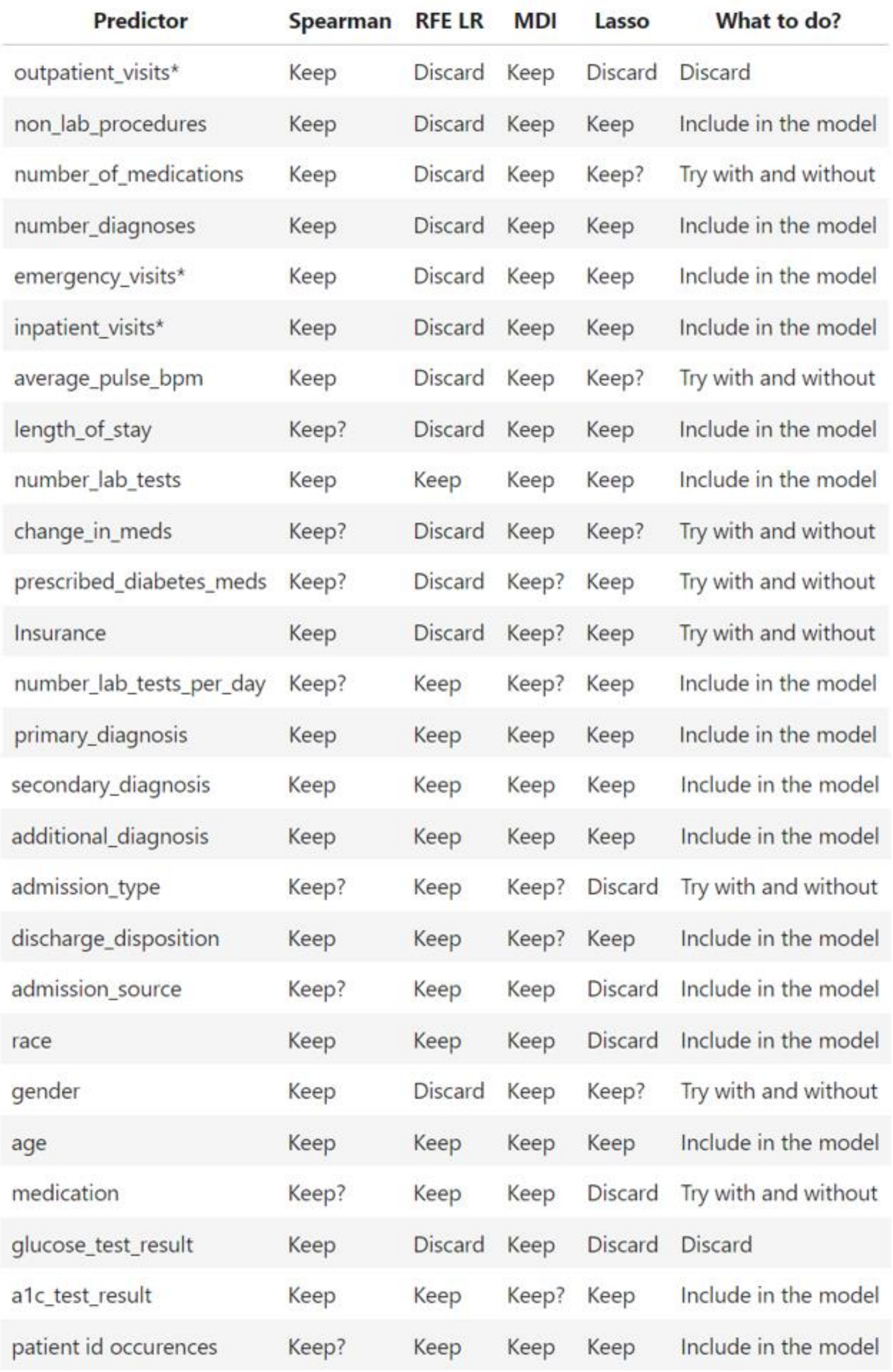
Category	Description
Category 1	INFECTIOUS AND PARASITIC DISEASES (001-139)
Category 2	NEOPLASMS (140-239)
Category 3	ENDOCRINE, NUTRITIONAL AND METABOLIC DISEASES, AND IMMUNITY DISORDERS (240-279)
Category 4	DISEASES OF THE BLOOD AND BLOOD-FORMING ORGANS (280-289)
Category 5	MENTAL, BEHAVIORAL AND NEURODEVELOPMENTAL DISORDERS (290-319)
Category 6	DISEASES OF THE NERVOUS SYSTEM AND SENSE ORGANS (320-389)
Category 7	DISEASES OF THE CIRCULATORY SYSTEM (390-459)
Category 8	DISEASES OF THE RESPIRATORY SYSTEM (460-519)
Category 9	DISEASES OF THE DIGESTIVE SYSTEM (520-579)
Category 10	DISEASES OF THE GENITOURINARY SYSTEM (580-629)
Category 11	COMPLICATIONS OF PREGNANCY, CHILDBIRTH, AND THE PUERPERIUM (630-679)
Category 12	DISEASES OF THE SKIN AND SUBCUTANEOUS TISSUE (680-709)
Category 13	DISEASES OF THE MUSCULOSKELETAL SYSTEM AND CONNECTIVE TISSUE (710-739)
Category 14	CONGENITAL ANOMALIES (740-759)
Category 15	CERTAIN CONDITIONS ORIGINATING IN THE PERINATAL PERIOD (760-779)
Category 16	SYMPTOMS, SIGNS, AND ILL-DEFINED CONDITIONS (780-799)
Category 17	INJURY AND POISONING (800-999)
Category 18	SUPPLEMENTARY CLASSIFICATION OF EXTERNAL CAUSES OF INJURY AND POISONING (E000-E999)
Category 19	SUPPLEMENTARY CLASSIFICATION OF FACTORS INFLUENCING HEALTH STATUS AND CONTACT WITH HEALTH SERVICES (V01-V91)

Table 4: *Selection Criteria for Feature Selection*



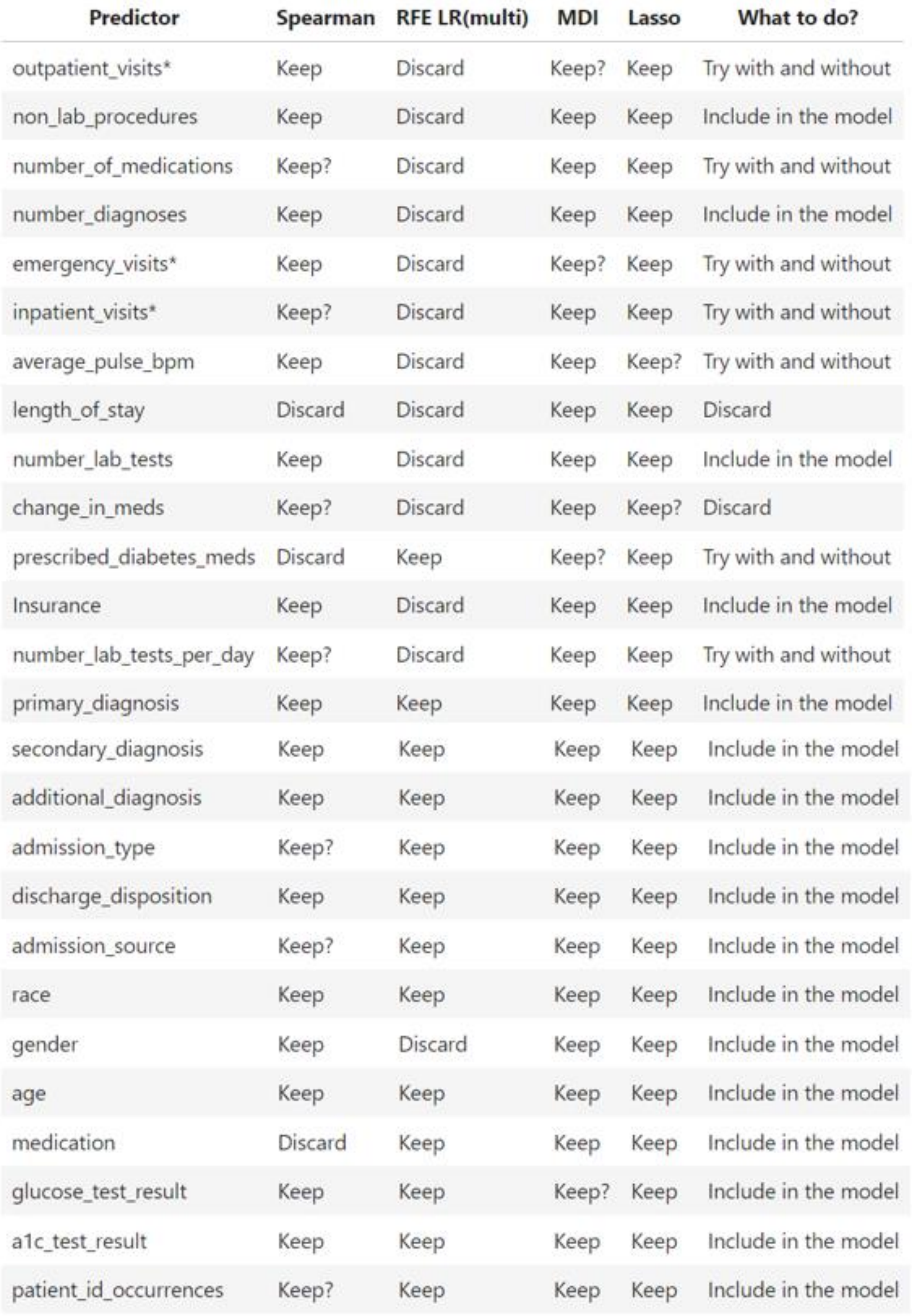
Selection Criteria
1) 'Keep?' + 'Keep?' = 1 'Discard'
2) > 1 'Discard' = 'Discard'
3) 1 'Discard' + 'Keep?' = 'Try with and without'
4) Keep otherwise

Table 5: *Binary Feature Selection Methods Summary*



Predictor	Spearman	RFE LR	MDI	Lasso	What to do?
outpatient_visits*	Keep	Discard	Keep	Discard	Discard
non_lab_procedures	Keep	Discard	Keep	Keep	Include in the model
number_of_medications	Keep	Discard	Keep	Keep?	Try with and without
number_diagnoses	Keep	Discard	Keep	Keep	Include in the model
emergency_visits*	Keep	Discard	Keep	Keep	Include in the model
inpatient_visits*	Keep	Discard	Keep	Keep	Include in the model
average_pulse_bpm	Keep	Discard	Keep	Keep?	Try with and without
length_of_stay	Keep?	Discard	Keep	Keep	Include in the model
number_lab_tests	Keep	Keep	Keep	Keep	Include in the model
change_in_meds	Keep?	Discard	Keep	Keep?	Try with and without
prescribed_diabetes_meds	Keep?	Discard	Keep?	Keep	Try with and without
Insurance	Keep	Discard	Keep?	Keep	Try with and without
number_lab_tests_per_day	Keep?	Keep	Keep?	Keep	Include in the model
primary_diagnosis	Keep	Keep	Keep	Keep	Include in the model
secondary_diagnosis	Keep	Keep	Keep	Keep	Include in the model
additional_diagnosis	Keep	Keep	Keep	Keep	Include in the model
admission_type	Keep?	Keep	Keep?	Discard	Try with and without
discharge_disposition	Keep	Keep	Keep?	Keep	Include in the model
admission_source	Keep?	Keep	Keep	Discard	Include in the model
race	Keep	Keep	Keep	Discard	Include in the model
gender	Keep	Discard	Keep	Keep?	Try with and without
age	Keep	Keep	Keep	Keep	Include in the model
medication	Keep?	Keep	Keep	Discard	Try with and without
glucose_test_result	Keep	Discard	Keep	Discard	Discard
a1c_test_result	Keep	Keep	Keep?	Keep	Include in the model
patient id occurences	Keep?	Keep	Keep	Keep	Include in the model

Table 6: *Multiclass Feature Selection Methods Summary*



Predictor	Spearman	RFE LR(multi)	MDI	Lasso	What to do?
outpatient_visits*	Keep	Discard	Keep?	Keep	Try with and without
non_lab_procedures	Keep	Discard	Keep	Keep	Include in the model
number_of_medications	Keep?	Discard	Keep	Keep	Try with and without
number_diagnoses	Keep	Discard	Keep	Keep	Include in the model
emergency_visits*	Keep	Discard	Keep?	Keep	Try with and without
inpatient_visits*	Keep?	Discard	Keep	Keep	Try with and without
average_pulse_bpm	Keep	Discard	Keep	Keep?	Try with and without
length_of_stay	Discard	Discard	Keep	Keep	Discard
number_lab_tests	Keep	Discard	Keep	Keep	Include in the model
change_in_meds	Keep?	Discard	Keep	Keep?	Discard
prescribed_diabetes_meds	Discard	Keep	Keep?	Keep	Try with and without
Insurance	Keep	Discard	Keep	Keep	Include in the model
number_lab_tests_per_day	Keep?	Discard	Keep	Keep	Try with and without
primary_diagnosis	Keep	Keep	Keep	Keep	Include in the model
secondary_diagnosis	Keep	Keep	Keep	Keep	Include in the model
additional_diagnosis	Keep	Keep	Keep	Keep	Include in the model
admission_type	Keep?	Keep	Keep	Keep	Include in the model
discharge_disposition	Keep	Keep	Keep	Keep	Include in the model
admission_source	Keep?	Keep	Keep	Keep	Include in the model
race	Keep	Keep	Keep	Keep	Include in the model
gender	Keep	Discard	Keep	Keep	Include in the model
age	Keep	Keep	Keep	Keep	Include in the model
medication	Discard	Keep	Keep	Keep	Include in the model
glucose_test_result	Keep	Keep	Keep?	Keep	Include in the model
a1c_test_result	Keep	Keep	Keep	Keep	Include in the model
patient_id_occurrences	Keep?	Keep	Keep	Keep	Include in the model

Figure 1: *Metric Features Initial Histograms*

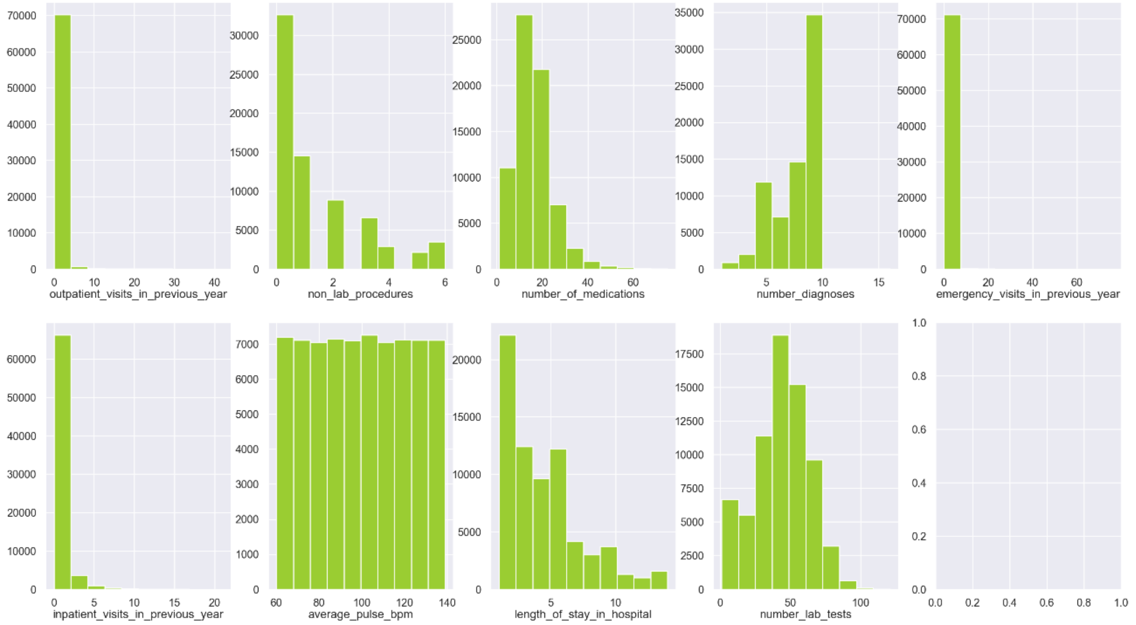


Figure 2: *Metric Features Initial Box Plots*

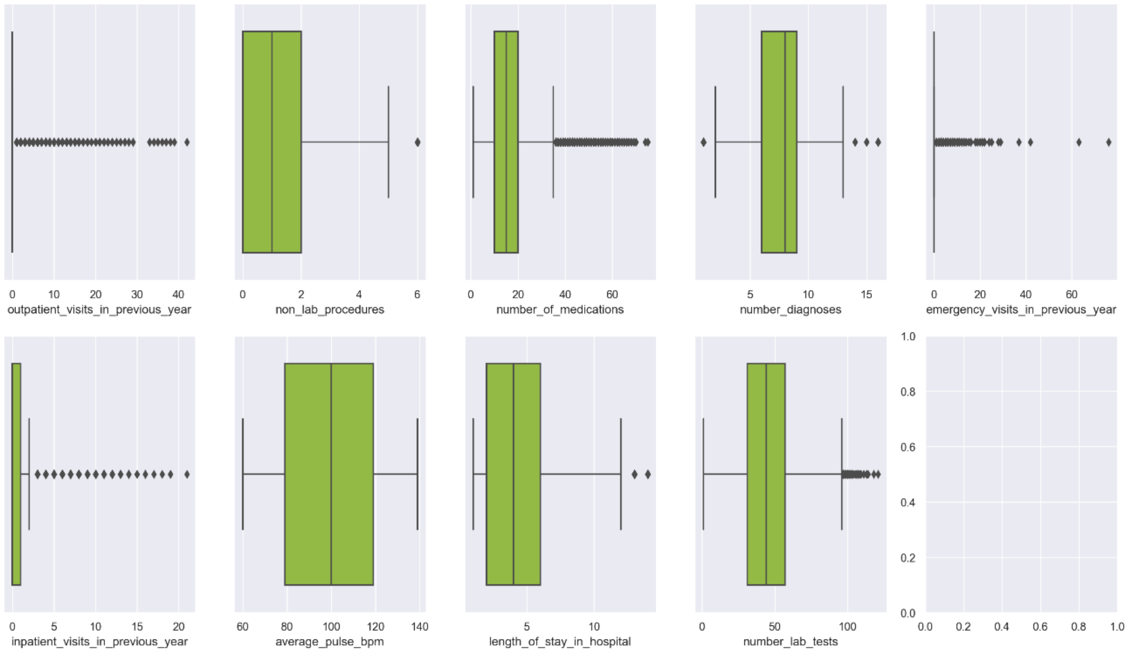


Figure 3: *Non-Metric Features Initial Count Plots*

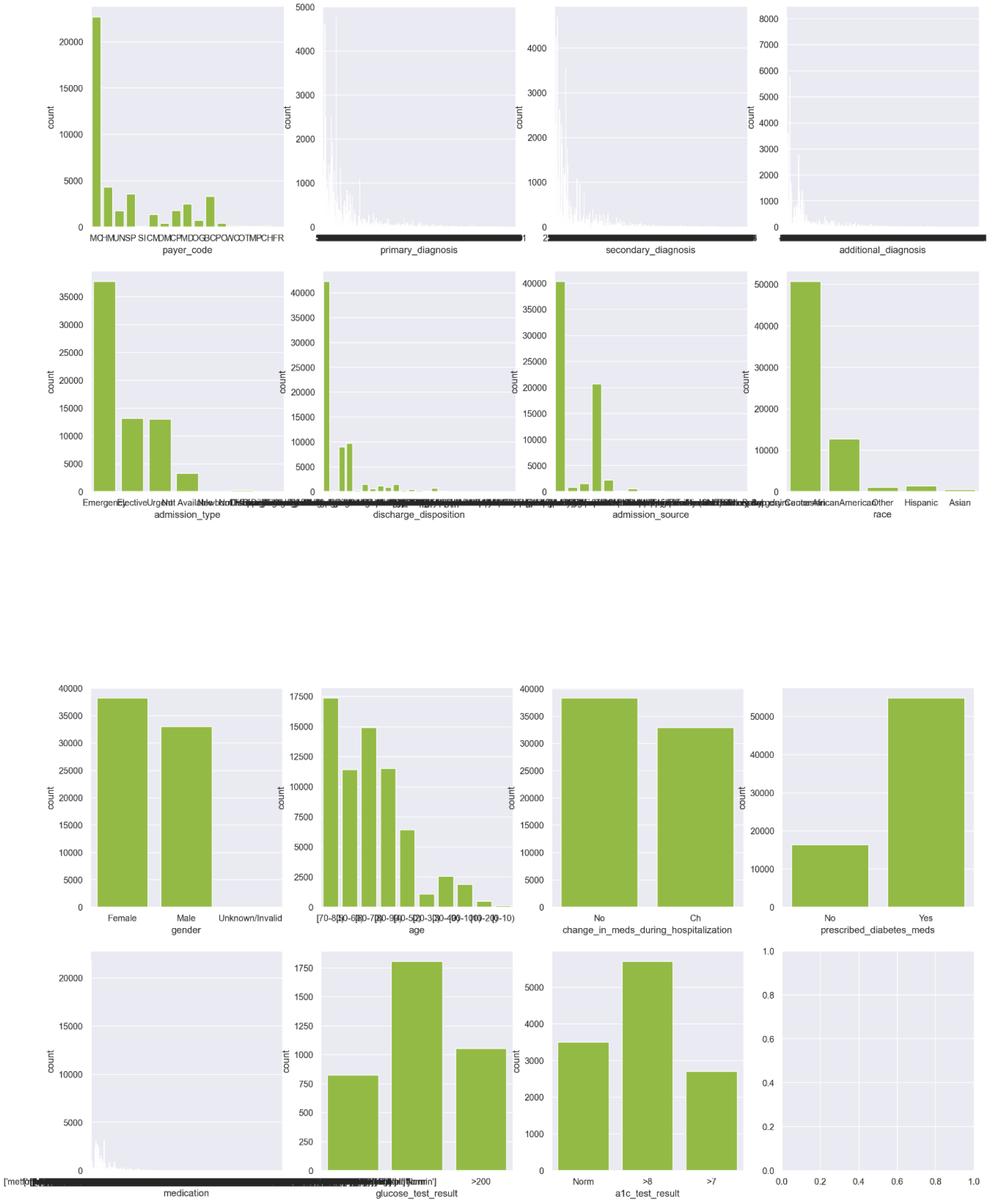


Figure 4: *Patient_id_ocurences target distribution*

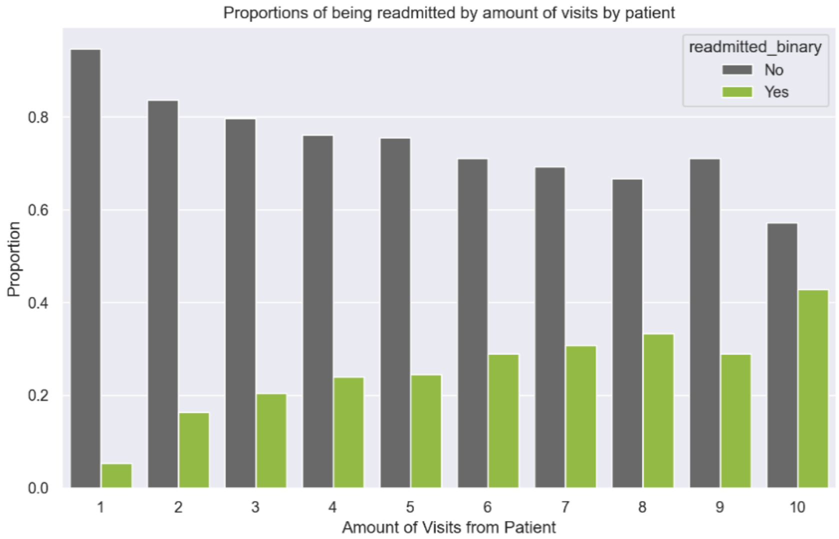


Figure 5: *Non-Metric Features Count Plots after Transformations*

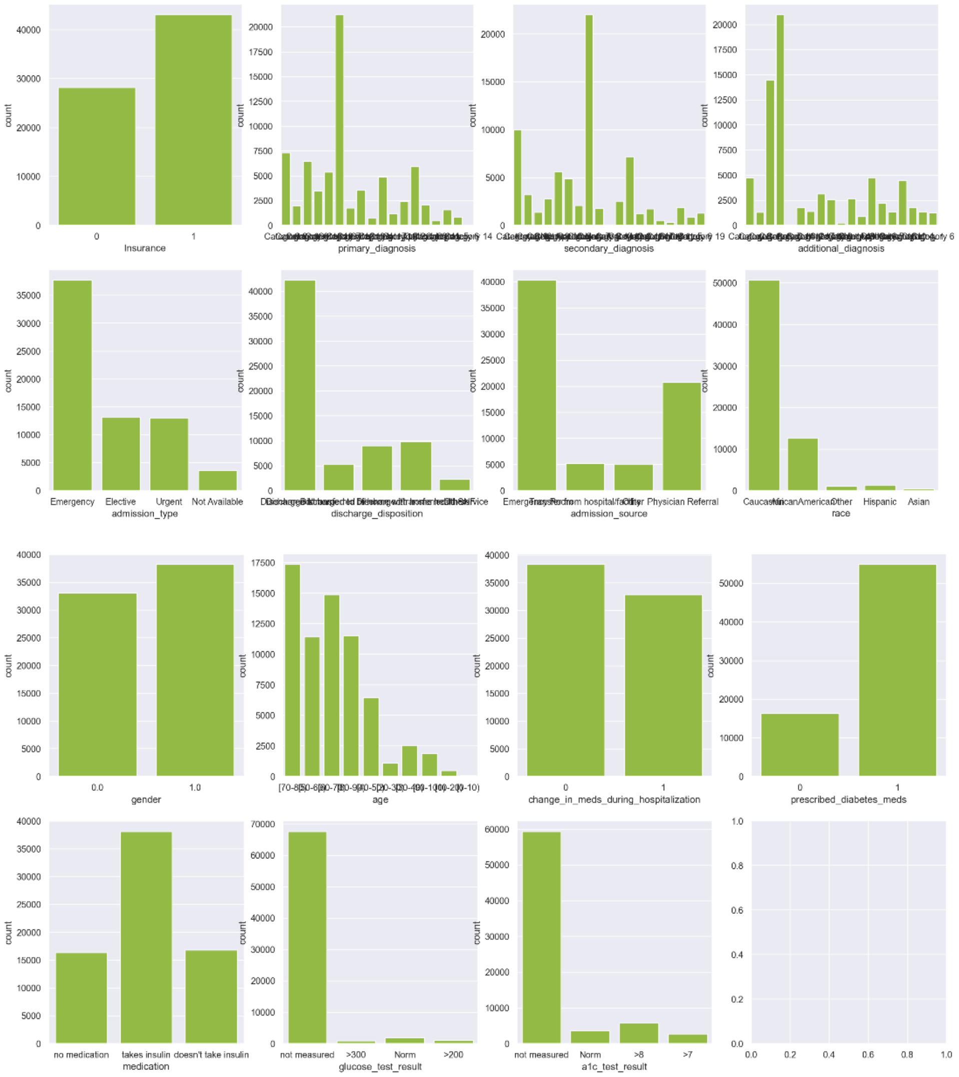
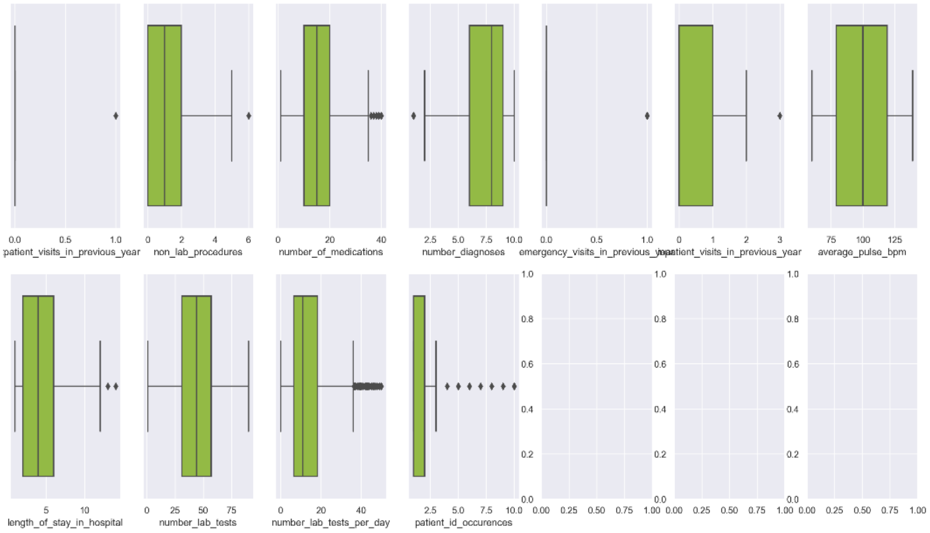
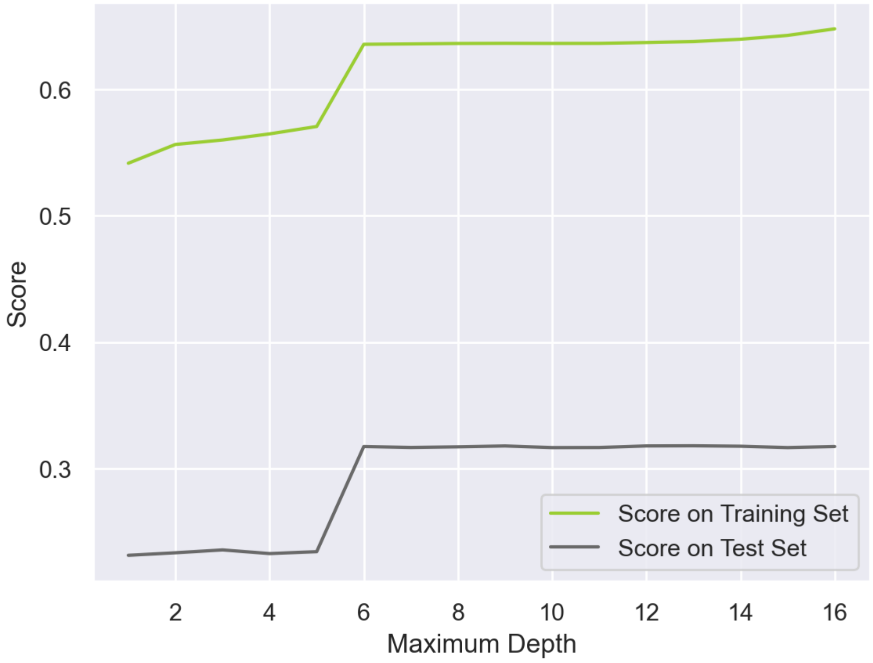


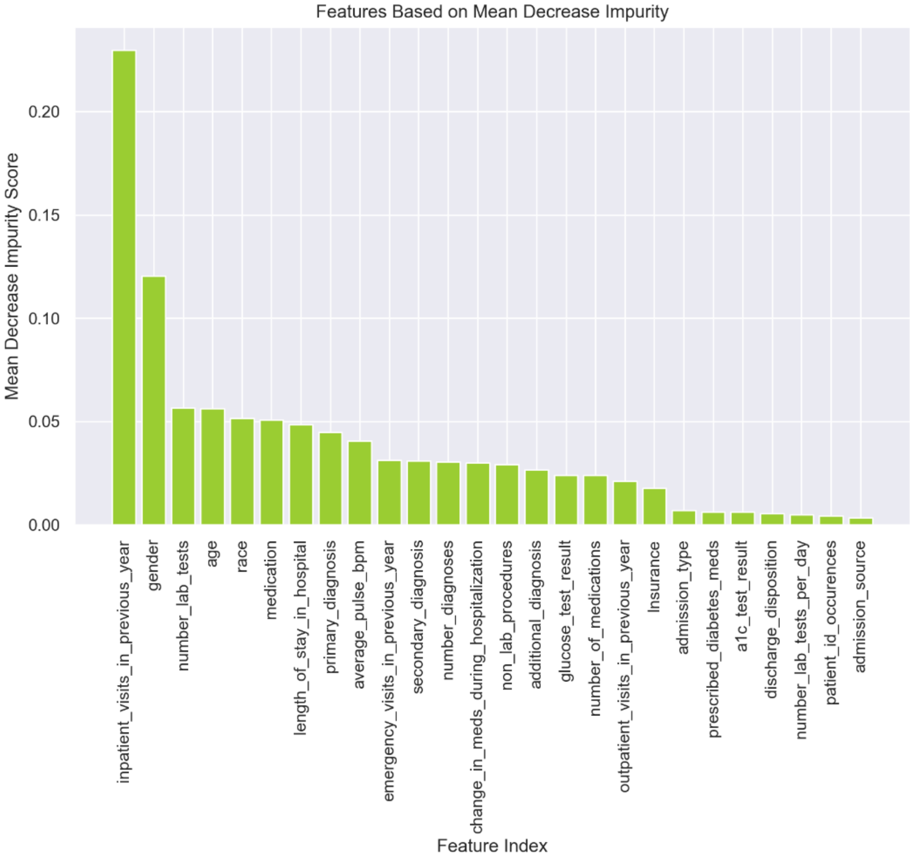
Figure 6: *Metric Features Distributions after Outliers and Transformations*



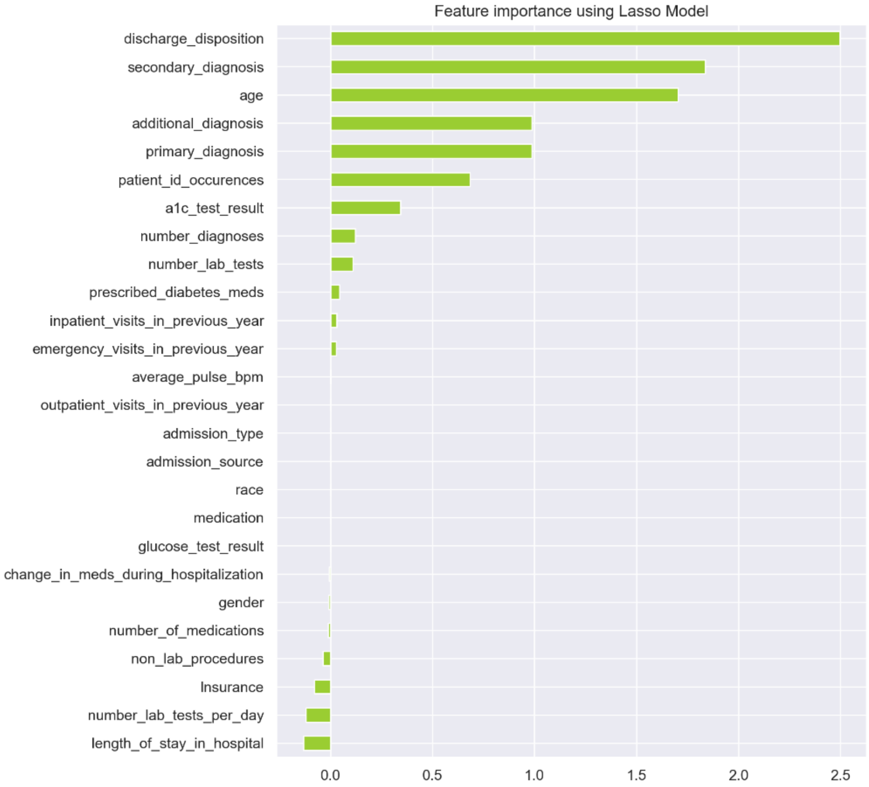
Graph 1: *RFE optimal score based on number of features- Binary*



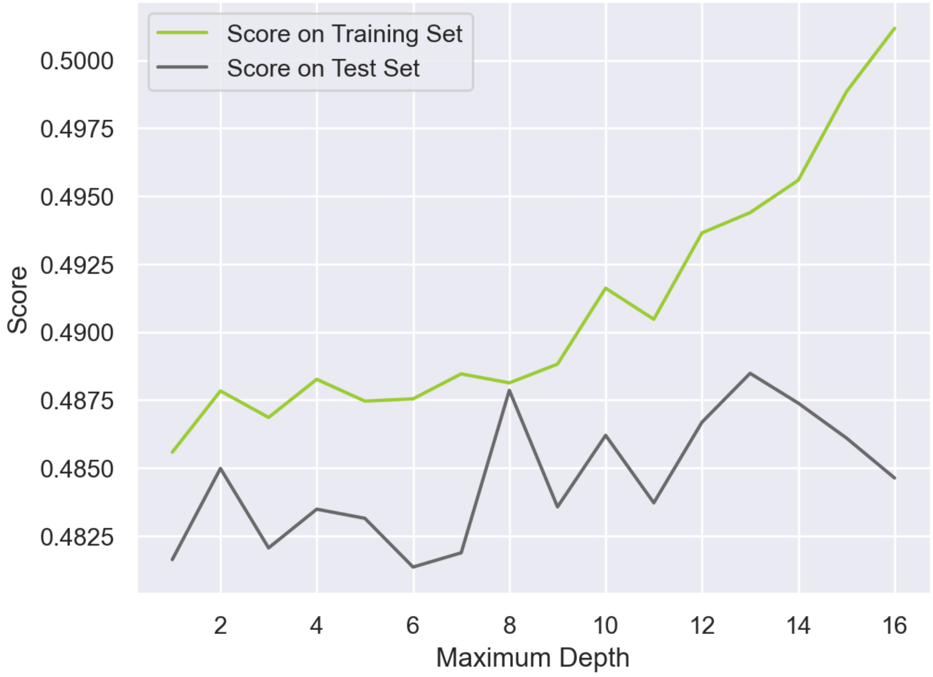
***Graph 2:** MDI feature importance- Binary*



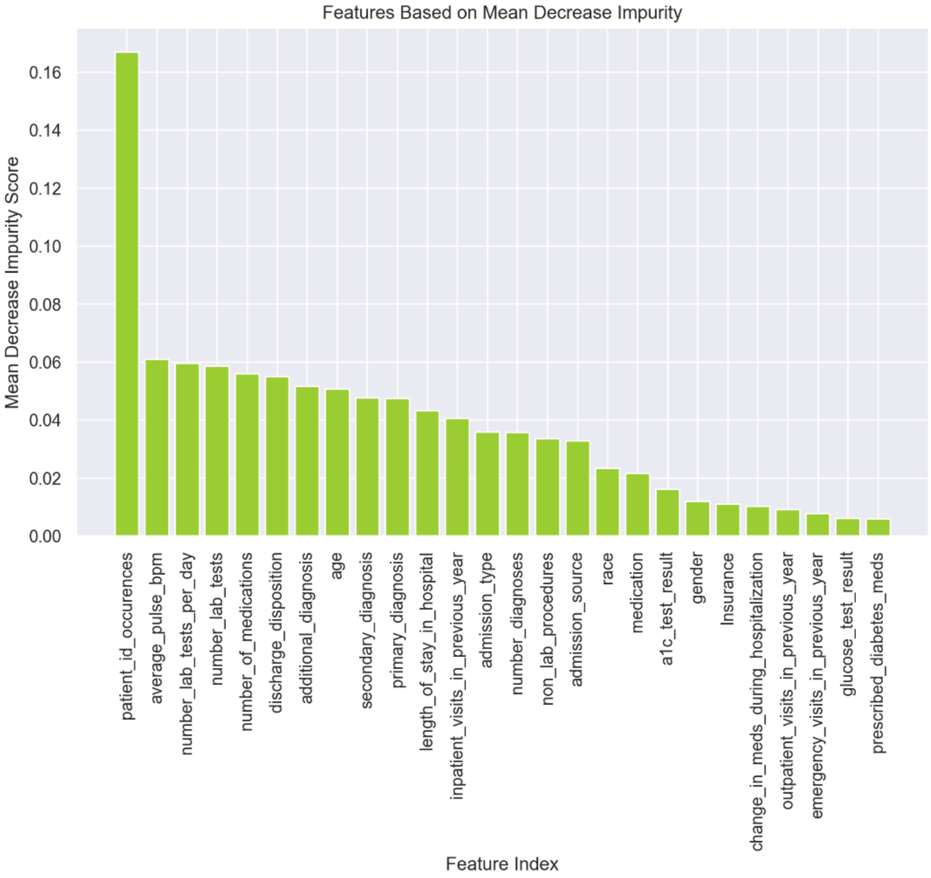
***Graph 3:** Lasso Regression Results- Binary*



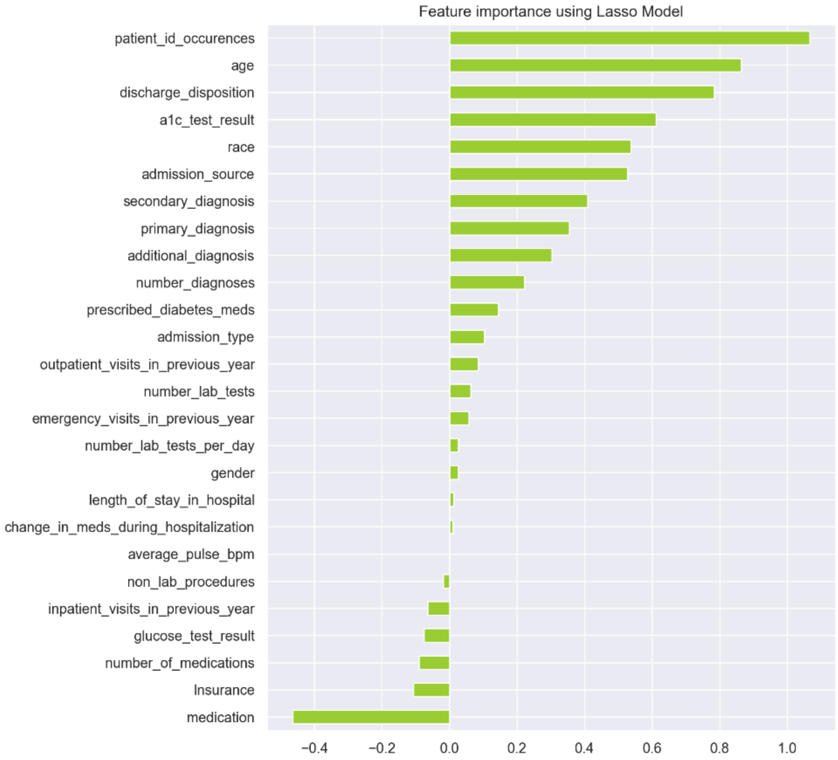
Graph 4: *RFE optimal score based on number of features- Multiclass*



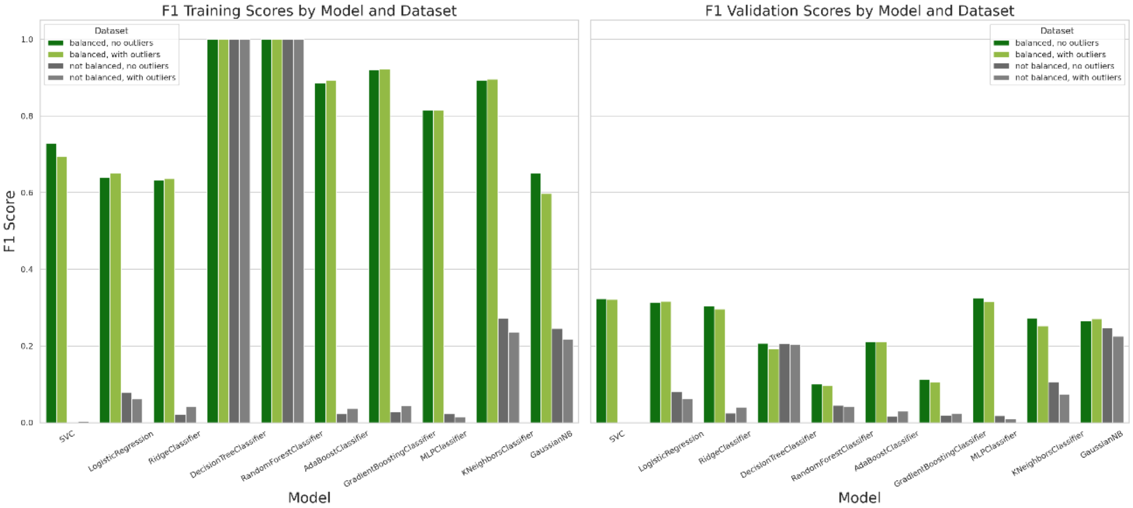
Graph 5: *MDI feature importance- Multiclass*



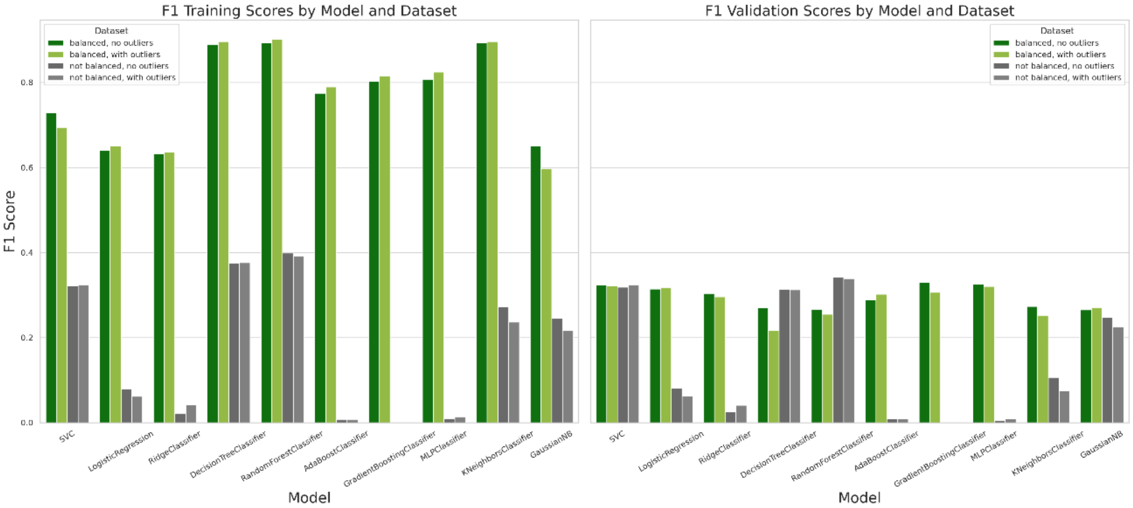
Graph 6: *Lasso Regression Results- Multiclass*



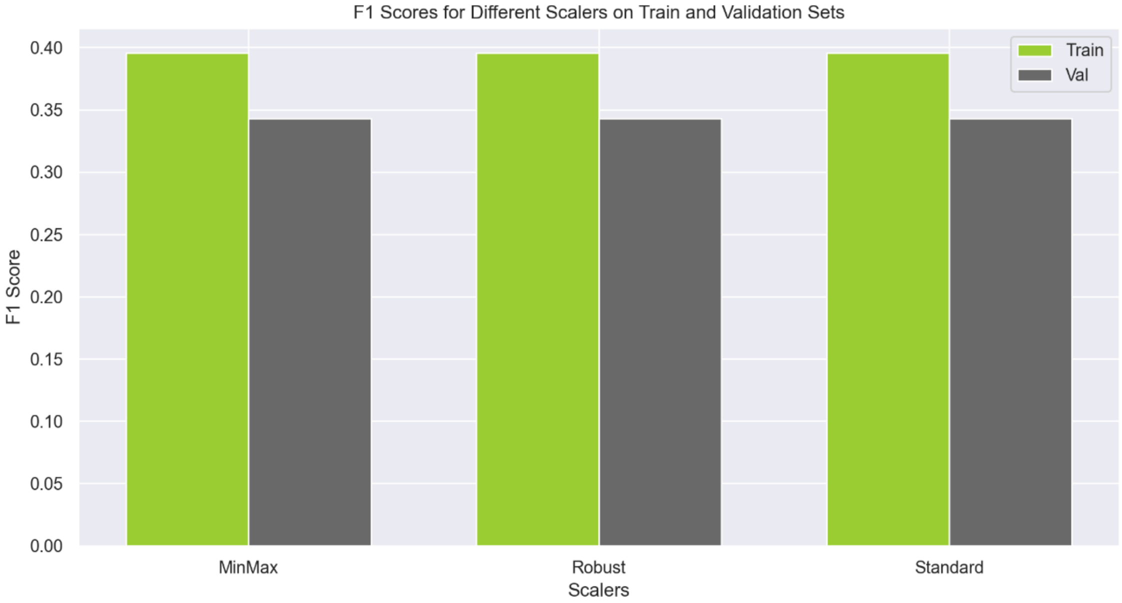
Graph 7: *ML Models F1-Score binary comparison on different datasets* ***before*** *initial parameter tuning*



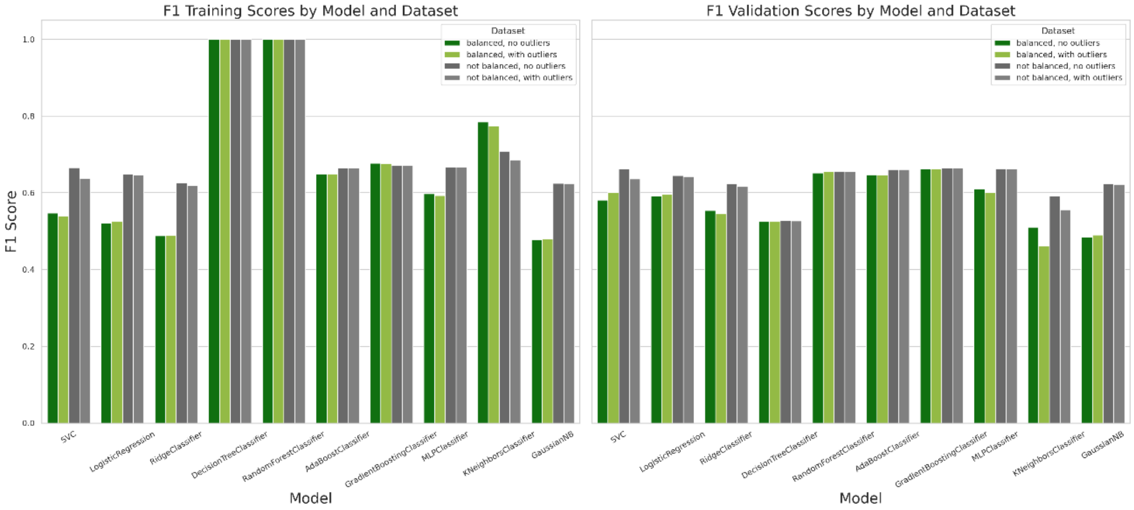
Graph 8: *ML Models F1-Score binary comparison on different datasets* ***after*** *initial parameter tuning*



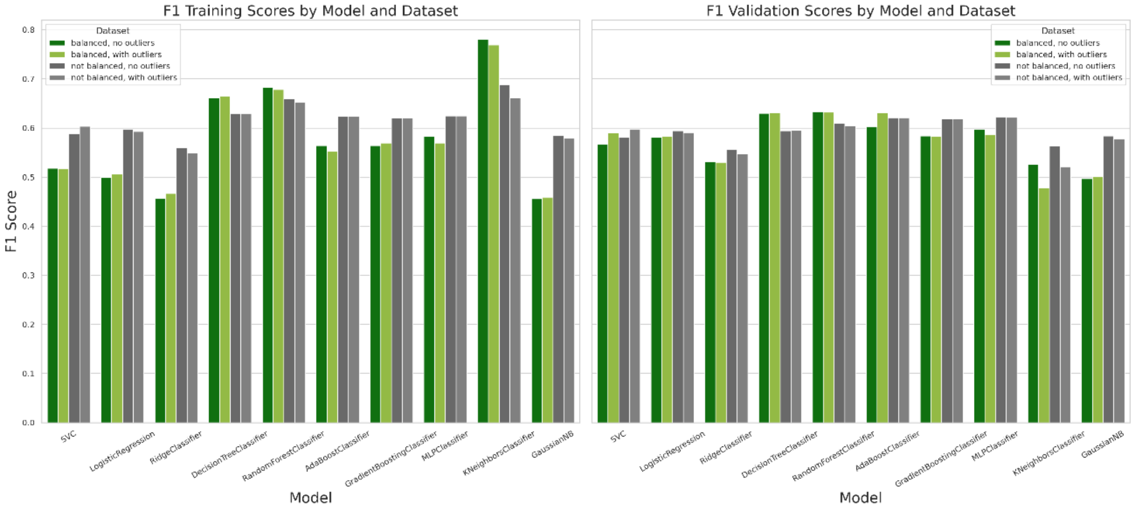
Graph 9: *Scalers comparison*



***Graph 10:** ML Models F1-Score multiclass comparison on different datasets **before** initial parameter tuning*



***Graph 11:** ML Models F1-Score multiclass comparison on different datasets **after** initial parameter tuning*



SMOTE EXPLANATION AND CONSIDERATIONS:

SMOTE- Synthetic Minority Oversampling Technique. SMOTE goes to a minority class instance, finds the k closer minority neighbors, chooses one and the new instance is the mean between the initial instance and chosen neighbor (Welvaars et al., 2023 & Garcia-Arce et al., 2017 & Groen et al., 2022). With it, we added artificial samples of our desired class making their presence in the train data set 1/2 for the binary and 1/3 for the multiclass. To prevent data leakage, data was split before resampling, that was only performed on the training set (Welvaars et al., 2023).

7. References

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. Journal of artificial intelligence research, 16, 321-357. https://www.jair.org/index.php/jair/article/view/10302

Du, G., Zhang, J., Luo, Z., Ma, F., Ma, L., & Li, S. (2020). Joint Imbalanced Classification and Feature Selection for Hospital Readmissions. Knowledge-Based Systems 200. https://doi.org/10.1016/j.knosys.2020.106020

Eckert, C., Nieves-Robbins, N., Spieker, E., Louwers, T., Hazel, D., Marquardt, J., Solveson, K., Zahid, A., Ahmad, M., Barnhill, R., McKelvey, T., Marshall, R., Shry, E., & Teredesai, A. (2019). Development and Prospective Validation of a Machine Learning-Based Risk of Readmission Model in a Large Military Hospital. Applied Clinical Informatics, 10(02), 316–325. https://doi.org/10.1055/s-0039-1688553

Gao, X., Alam, S., Shi, P., Dexter, F., & Kong, N. (2023). Interpretable machine learning models for hospital readmission prediction: A two-step extracted regression tree approach. BMC Medical Informatics and Decision Making, 23(1), 104. https://doi.org/10.1186/s12911-023-02193-5

Garcia-Arce, A., Rico, F., & Zayas-Castro, J. L. (2018). Comparison of Machine Learning Algorithms for the Prediction of Preventable Hospital Readmissions. Journal for Healthcare Quality, 40(3), 129–138. https://doi.org/10.1097/JHQ.0000000000000080

Groen, D., De Mulatier, C., Paszynski, M., Krzhizhanovskaya, V. V., Dongarra, J. J., & Sloot, P. M. A. (Eds.). (2022). Computational Science – ICCS 2022: 22nd International Conference, London, UK, June 21–23, 2022, Proceedings, Part III (Vol. 13352), pp. 122–136, 2022. Springer International Publishing. https://doi.org/10.1007/978-3-031-08757-8

Haixiang, G., Yijing, L., Shang, J., Mingyun, G., Yuanyue, H., & Bing, G. (2017). Learning from class-imbalanced data: Review of methods and applications. Expert Systems with Applications, 73, 220–239. https://doi.org/10.1016/j.eswa.2016.12.035

Jeni, L. A., Cohn, J. F., & De La Torre, F. (2013). Facing imbalanced data--recommendations for the use of performance metrics. In 2013 Humaine Association Conference on Affective Computing and Intelligent Interaction (pp. 245-251). IEEE. https://ieeexplore.ieee.org/document/6681438

Kohavi, R. (1995). "A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection." In Proceedings of the 14th International Joint Conference on Artificial. https://www.ijcai.org/Proceedings/95-2/Papers/016.pdf

Li, Q., Yao, X., & Échevin, D. (2020). How Good Is Machine Learning in Predicting All-Cause 30-Day Hospital Readmission? Evidence From Administrative Data. Value in Health, 23(10), 1307–1315. https://doi.org/10.1016/j.jval.2020.06.009

Michailidis, P., Dimitriadou, A., Papadimitriou, T., & Gogas, P. (2022). Forecasting Hospital Readmissions with Machine Learning. Healthcare, 10(6), 981. https://doi.org/10.3390/healthcare10060981

Ostling, S., Wyckoff, J., Ciarkowski, S. L., Pai, C.-W., Choe, H. M., Bahl, V., & Gianchandani, R. (2017). The relationship between diabetes mellitus and 30-day readmission rates. Clinical Diabetes and Endocrinology, 3(1), 3. https://doi.org/10.1186/s40842-016-0040-x

Sutter, T., Roth, J. A., Chin-Cheong, K., Hug, B. L., & Vogt, J. E. (2021). A comparison of general and disease-specific machine learning models for the prediction of unplanned hospital readmissions. Journal of the American Medical Informatics Association, 28(4), 868–873. https://doi.org/10.1093/jamia/ocaa299

Welvaars, K., Van Den Bekerom, M. P. J., Doornberg, J. N., Van Haarst, E. P., OLVG Urology Consortium, Van Der Zee, J. A., Van Andel, G. A., Lagerveld, B. W., Hovius, M. C., Kauer, P. C., & Boevé, L. M. S. (2023). Evaluating machine-learning algorithms to Predict 30-day Unplanned REadmission (PURE) in Urology patients. BMC Medical Informatics and Decision Making, 23(1), 108. https://doi.org/10.1186/s12911-023-02200-9