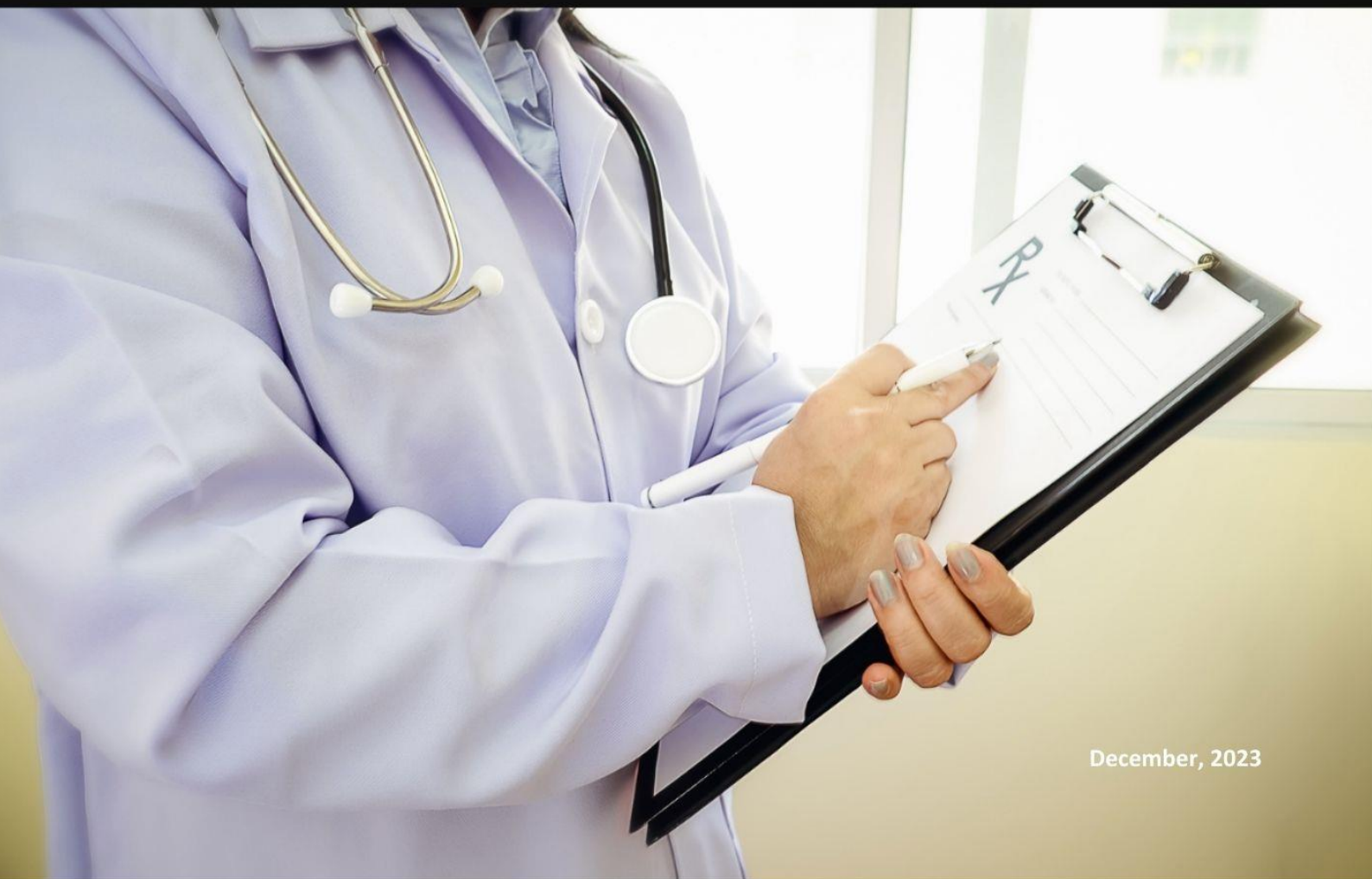
Master Degree Program in Data Science and Advanced Analytics
Specialization: Business Analytics

READY TO BE DISCHARGED: EXAMINING HOSPITAL READMISSIONS

Professors:
Roberto Henriques
Ricardo Santos
Rafael Pereira

Group 14:
Alina Metzger | 20230998
Andreia Guerreiro | 20201563
Hugo Alves | 20230438
João Lopes | 20230748
Wesley de Oliveira | 20230475

GROUP PROJECT
MACHINE LEARNING 2023/2024



December, 2023

Abstract

High rates of readmissions are burdensome to hospitals since they often mean failure to provide effective healthcare. Chronic diseases, like diabetes, pose an additional challenge since they increase the likelihood of short-term readmissions. Therefore, the medical staff must identify whether a patient would be readmitted to understand the factors increasing readmission risks, develop tailored solutions for patients, and improve hospitals' effectiveness. Given these circumstances, we approached the readmissions problem from two perspectives: a binary model to predict if a patient would be readmitted 30 days post-discharge and a multiclass model, broadening the options to predict readmissions after 30 days. Using a diabetes dataset, we tested several models for both cases after a rigorous data cleaning and feature selection comprising eight filter, wrapper, and embedded methods, including ANOVA, decision tree, and LASSO. In both cases, we have single models, including logistic regression and neural networks, and ensemble models, such as random forests and stacking. We offer four key findings: first, for the binary, a bagging classifier with a tuned neural network as a base estimator yielded the best results: the highest f1 score (0.2993) and the second highest recall (0.4356). Second, we recommended a Bernoulli Bayesian model, an explainable model with lower scores (0.2968 in f1 score and 0.4218 in recall) due to interpretability. Third, for the multiclass, a voting classifier with decision tree, random forest, bagging, gradient boosting, and stacking as base estimators yielded the best results: the highest f1 score (0.5842) and recall (0.6142). Fourth, we recommended a decision tree, an explainable model with lower scores (0.5718 in f1 score and 0.5975 in recall) due to interpretability. Lastly, although the scores were lower than expected, we offered consistent models that went through a rigorous and transparent decision-making process.

Contents

List of Tables

List of Figures

I. Introduction

The ever-evolving healthcare landscape demands innovative approaches to address critical challenges, such as the high rate of readmissions. In short, readmissions imply readmitting patients for the same or correlated disease after discharge, often within a 30-day timespan (1,2). Readmissions are problematic since patients and regulators perceive them as improper care stemming from inaccurate diagnoses and inadequate follow-up appointments, to name a few (3–5). Consequently, hospital readmissions within 30 days became a core for measuring healthcare quality and associated costs (1,4). For instance, hospitals in the United States (US) must comply with a readmission reduction program established by the Centers for Medicare and Medicaid Services (CMS) to decrease unnecessary short-term readmissions and avoid penalties (4,6). Therefore, practitioners and researchers must better understand the conditions leading to readmissions to reduce the burden on patients and healthcare systems (4,7).

The literature proposes several solutions to the readmission problem through machine learning (ML) models, with approaches like logistic regression as a hallmark in the field (7). Furthermore, the uprising of models (e.g., neural networks) able to handle non-linear and more complex relationships, as well as unstructured and huge amounts of data, poses promising options for predicting patients' readmissions (7,8). Therefore, researchers empirically applied several ML models in contexts such as heart failure (9) and diabetes (1,10) to predict whether a patient would be readmitted within 30 days post-discharge.

Frizzell and colleagues (9) tested some models, such as random forest and logistic regression, on a US heart failure dataset. They followed a holdout method by splitting 70% of the data in a training dataset and the rest in a validation. The authors did not provide details regarding data preprocessing and dropped over 75% of the original data due to insufficient information. Furthermore, the researchers presented performance metrics on the validation dataset based on the receiver operating characteristic (ROC) curve. Based on that, they expected other models, including random forest, to outperform traditional approaches like logistic regression, which was not the case, as the scores were almost similar (0.61 and 0.62, respectively).

Shang et al. (1) tested three models on a diabetes dataset: random forest, naive Bayes, and a model labeled as a tree ensemble classifier. They also followed a holdout method by splitting 80% of the data in a training dataset and the rest in a test. For predicting purposes, the authors dropped irrelevant variables with high rates of missing values (e.g., weight) and attributes from several variables (e.g., attributes from medical specialty). Furthermore, they used sampling techniques on the training dataset to handle imbalance issues and ROC scores to compare the models. They concluded by recommending using a random forest since it has the best ROC score. Moreover, age, inpatient admissions, and gender are influential factors in identifying patients at high risk of readmission.

Lastly, Ramírez and Herrera (10) tested logistic regression, neural network, and random forest on the same diabetes dataset. They followed a 10-fold cross-validation to split the dataset, dropped variables with high rates of missing values (e.g., medical specialty), used oversampling in the training dataset, and principal component analysis to reduce the data's dimensionality. Furthermore, they kept only the first observation for each patient ID to ensure independence on the dataset. Based on several scores, including ROC and recall, they concluded that random forest is the best model to predict readmissions.

Given the circumstances above, it is imperative for hospitals to have solutions to identify patients at high risk of readmission. Thus, we aim to address such a problem through binary and multiclass classification models based on a US diabetes dataset. First, a binary model enables predicting 30 days post-discharge readmissions. Successful predictions in this context hold the potential to facilitate preventive measures and cost savings. Second, a multiclass model extends the predictive capabilities to a nuanced scenario, classifying readmissions into three categories: no readmissions (i.e., "No"), readmissions within the 30-day timespan (i.e., "≤30 days"), and readmissions after 30 days of discharge (i.e., ">30 days."). This model provides unique insights for tailored post-discharge care, enabling hospitals to address patients' needs better and improve their short- and long-term

quality of life. Third, diabetes is a relevant context to approach due to its prevalence in the American population, including hospitalized patients, increasing the risks of short-term readmissions (4).

Finally, given the results outlined above, we expect to deliver ML models capable of predicting readmissions fairly in both scenarios. Second, we aspire to propose models outside the logistic regression hallmark approach since, as stated before, they would enable capturing non-linear relationships (7). Concurrently, these models would be harder to interpret due to the black box issue (8,11). Third, directly comparing our project with the papers above is challenging because Frizzell and colleagues (9) used a dataset unrelated to diabetes. Furthermore, although the other two studies (1,10) used a similar diabetes dataset, they had more variables initially available, increasing their chances of grasping more nuanced diabetes-related aspects. All in all, the absence of a benchmark dataset hampers comparing studies (7).

In the following sections, we will delve into dataset exploration and preprocessing, outline our strategies for binary and multiclass classification, discuss our findings and limitations, and conclude by offering suggestions for future projects.

II. Data Exploration and Preprocessing

Data Characteristics and Initial Insights

The project's dataset consists of detailed information regarding US diabetic patient records from 1999 to 2008 based on the dataset developed by Strack and colleagues (12). This multivariate dataset includes patient demographics (e.g., *gender*), medical history (e.g., *medication*), and encounter details (e.g., *discharge_disposition*). Check section A in the annexes for more details regarding the initial variables. However, our dataset has important modifications: first, it consists of training and test sets with 71,236 entries and 31 variables and 30,530 entries and 29 variables, respectively. Second, it has fewer variables than Strack et al. (12), which has 55.

At the beginning of our analysis, we uploaded both diabetes sets to the notebook to perform the subsequent data preprocessing on the training and test sets simultaneously. Afterward, similar to previous studies (1,9), we applied a holdout method. Therefore, we randomly split our training set in a stratified way, which retained 70% of the data while the rest became our validation set. Such stratification enabled the project dataset class distribution to be maintained during the split. Furthermore, the reason for using holdout lies in computational constraints that hindered the use of more robust methods, such as k-fold cross-validation (CV) (check the conclusion chapter for more details regarding the project's limitations).

We proceeded to data exploration in the training set by counting the initial variables' values frequencies and plotting histograms and boxplots. These techniques allowed us to capture and understand the dataset's inherent characteristics, including unusual and missing values, two aspects that demand closer attention (check section B in the annexes for more details). For the former, a common unusual value identified throughout the set was a question mark as entries for several variables, including *weight*, with roughly 97% of occurrences, and *payer_code*, with nearly 40%. For the latter, we initially grasped 5% of occurrences for the variables *gender* and *age* and over 94% and 83% for *glucose_test_result* and *a1c_test_result*, respectively. Furthermore, data exploration also showed us a point to acknowledge in later steps regarding both target variables: a highly imbalanced class distribution. In the binary, almost 89% of the observations belonged to the class of no readmissions within the 30-day timespan, while roughly 54% of the observations fell into the class of no readmissions for the multiclass. Although readmission problems are inherently imbalanced, studies often neglect such an issue and disregard the fact that some ML models require balanced datasets (7,8).

Data Preprocessing: Unusual and Missing Values

Given the observations above, we moved to data preprocessing by handling the unusual and missing values based on the aspects of each variable. First, we replaced the question marks on *race* with "Other", since we assumed that the medical staff collected such information from patients, but it was a different race than the existing ones in the dataset. Furthermore, we changed the missing values to "Unknown", assuming that the patients refused to provide such information or the medical staff did not collect it. Second, we assumed that the remaining question marks in the other variables (e.g., *weight* and *payer_code*) represent missing values. Thus, we transformed these marks into missing values to facilitate the preprocessing. Third, we dropped the variables *medical_specialty* and *weight* due to their high number of missing values (49% and 97%, respectively), avoiding biasing our results. Fourth, we assigned the missing values from *discharge_disposition* and *admission_source* into an existing category named "Not Mapped". Consequently, we maintained the essence of unknown values for these observations and reduced redundancy by avoiding having two values referring to the same phenomenon. Fifth, we replaced the missing values from *primary_diagnosis*, *secondary_diagnosis*, and *additional_diagnosis* with "Other", considering that either the physician could not find a diagnosis for a patient or there was no need for a secondary or additional diagnosis.

Lastly, after handling such issues in a specific variable of the training set, we performed the same steps on the validation and test to ensure consistency among the sets. Check section B in the annexes to see the identified unusual and missing value problems and the measures taken for each variable.

Data Preprocessing: Feature Engineering and Scaling

The steps above enabled us to move to feature engineering to transform, remove, and create new variables. First, we dropped the *country* variable, as it only states that each patient is American. Second, we transformed the variable *payer_code* into a binary labeled *health_insurance*, stating whether a patient has insurance (i.e., "1") or not (i.e., "0"). Although it had a substantial number of missing values (around 39%), we kept the variable since previous studies found that insurance is a significant risk factor for diabetes readmissions (4). Third, we also transformed the glucose and A1C test-related variables in binaries (*took_glucose_test* and *took_a1c_test*, respectively), stating whether the medical staff measured the patient's sugar and A1C levels (i.e., "1" in both cases) or not (i.e., "0" in both cases). Researchers encourage the use of these modifiable variables in ML models to check whether they learn the nuances of patients' characteristics and improve their predictive capabilities (4,8).

Fourth, we assessed the content of the categorical variables (e.g., *discharge_disposition* and *admission_source*) and grouped the values of each according to their similarities, resulting in new variables named *merged_discharge_disposition* and *merged_admission_source*. Consequently, we reduced the noise in the data by decreasing the range of possible values and eliminating redundant information. Fifth, we created specific diabetes-related variables based on the International Classification of Diseases, Ninth Revision (ICD-9) codes used for the *primary_diagnosis*, *secondary_diagnosis*, and *additional_diagnosis*. Considering all the diagnoses in these three variables, those directly related to diabetes were the only ones with additional information due to the decimal numbers in the code. Check section C in the annexes for a detailed explanation of the logic followed in categorizing the values for each variable, including the diabetes-related ones.

Sixth, we created binary variables to consider the repeated hospitalizations of patients (i.e., *multiple_encounters*) and whether a patient is still active after discharge (i.e., *patient_status*). For *multiple_encounters*, it returns "1" when a specific patient ID has a chain of repeated hospitalizations in the dataset, regardless of a specific encounter being the patient's first or last record. Otherwise, it returns "0". Acknowledging the repeated hospitalizations yields more realistic predictions. Studies show that disregarding repeated readmissions increases the chances of having overly optimistic results on models, yielding inaccuracy when predicting in the wild (4).

However, we must also consider that keeping all the observations may introduce noise in the data, especially when the discharge outcome is expired or hospice (12). The reason is that expired patients cannot be readmitted, and hospice patients require special policies and approaches for end-of-life care (13). Consequently, they do not represent a standard outcome if their readmissions happen. Therefore, *patient_status* returns "0" for encounters where the discharge destination is expired or hospice. Otherwise, it returns "1".

Seventh, we transformed the categorical variables into numbers to facilitate the feature selection step, as some techniques we employed require a prior numerical transformation of categories (e.g., analysis of variance (ANOVA) and logistic regression tailored to feature elimination). For instance, we used the midpoint for the variable *age* since the median was not a whole number, and *gender* became a binary variable with the value "0" assigned to "Female" and "1" to "Male". Furthermore, we transformed variables such as *admission_type* and *merged_admission_source* based on their frequencies, considering that our goal was to avoid increasing the data's dimensionality whenever possible (*admission_type_freq* and *admission_source_freq*, respectively). Check section D to see lists of variables we kept, dropped, and created. Furthermore, we normalized the variables except the binary ones using a range from zero to one to bring them to the same scale. Such a scaling yielded the best results among the ones we tried (e.g., a normalization ranging from minus one to one and a standardization). Finally, after adjusting the training set, we modified the validation and test sets to maintain consistency.

III. Binary Classification

Over-sampling and Feature Selection

Given the circumstances of the addressed problem, we must adopt an over-sampling technique before proceeding to feature selection. Thus, we chose the Synthetic Minority Over-sampling Technique (SMOTE) for two reasons: first, it addresses the issue stemming from a highly imbalanced dataset by focusing on the minority class sampling – the readmitted observations in our case – rather than replacing existing samples. Consequently, it avoids overfitting (14,15). Second, we needed a technique that would enable us to keep all observations while addressing imbalance. Therefore, we could not consider under-sampling techniques, as their core is to remove observations from the majority class (15). Third, we employed a support vector machine-based SMOTE (SVMSMOTE) to mitigate the misclassification issue of the minority class often introduced by SMOTE (16) (more details in section E in the annexes).

After over-sampling, defining the optimal subset of variables for the modeling stage is a vital step, as the resulting subset serves as a basis for tuning and training the models. To that extent, we employed filter, wrapper, and embedded methods to define our optimal subset. For the filter methods, we considered Pearson and Spearman's correlations, as well as ANOVA (section F in the annexes). First, using correlations allowed us to identify highly correlated independent variables in which the linear (Pearson's) or non-linear (Spearman's) correlations were over the set threshold of 0.65. Furthermore, we detected the independent variables' correlation with the target (i.e., *readmitted_binary*), which highlighted the lack of significant relationships between the predictors and the dependent variable. Second, the correlations between the independent variables enabled us to define the following dropping criteria: When two independent variables had a correlation above 0.65, we assessed their individual correlations with the target and dropped the one with the lowest value. Consequently, we removed six and seven variables through Pearson and Spearman's correlation, respectively (check Table 15 in section G in the annexes for details). Third, the desired number of 25 variables in ANOVA resulted in the rejection of the other seventeen (Table 15 in the Annex). Setting the ideal number of variables as 25 results from an attempt to be more permissive in terms of selecting variables for the optimal subset. The reason lies in having less than nine variables selected at the end of the feature selection, which led us to poor results when training the models.

For the wrapper methods, we applied recursive feature elimination (RFE) through logistic regression and linear SVC, both with L1 regularization, a standard choice for feature selection since it allows for the less relevant variables to equal zero (17), something which would not verify with a L2 penalization. Furthermore, we used decision trees and random forests (both with a 0.03 threshold, considering the lack of overall feature importance of the predictors in relation to the target), making use of these models' ability to retrieve variables' importance (18,19) (Table 15). Lastly, For the embedded method, we used the least absolute shrinkage and selection operator (LASSO) due to, as stated before, its suitability for selecting variables by ignoring those equal to zero.

The criteria we set for defining the subset of variables was to keep those selected by at least half of the eight methods. Our original approach was to keep those picked by at least five methods, but this would result in having only eight variables (as presented in Table 15 in the annexes). Therefore, we kept the following eleven variables by modifying the mentioned criteria: *took_a1c_test*, *race_hispanic*, *discharge_disp_freq*, *diabetes_severity_moderate*, *multiple_encounters*, *patient_status*, *age*, *inpatient_visits_in_previous_year*, *length_of_stay_in_hospital*, *non_lab_procedures*, and *number_diagnoses*.

Modeling

After feature selection, we trained and tuned the models in order to compare them and choose the most suitable one. We used precision, recall, f1 score, and the score resulting from the submission in Kaggle as comparison metrics. Additionally, we retrieved the accuracy and ROC-AUC scores, so as to get additional insights regarding each model's performance, helping us improve the tuning of the hyperparameters. Table 16 in section H in the annexes shows the score for all these metrics. In the modeling stage, we first assessed several single models, namely logistic regression, Gaussian and Bernoulli naive Bayes, neural networks, decision trees, stochastic gradient descent (SGD) (check section I in the annexes for a brief conceptual explanation), support vector machines (SVM) with a polynomial kernel and linear SVC. The k-nearest neighbors (KNN) algorithm was not tested due to its dependence on the distance function and the existence of binary and non-binary variables in our selected subset. Afterwards, we assessed ensemble models resulting from combining top-performing single models, along with other ensemble methods, including random forest, gradient boosting through the histogram-based variant due to its suitability to large datasets, and voting (check section J in the annexes for conceptual explanations). In regard to the single models, we performed grid searches for logistic regression, decision trees, SVM, and SGDs to find the optimal parameters, whilst, for the remaining, manual tuning was preferred due to lack of necessity (e.g. only one parameter was tuned in Gaussian and Bernoulli naive Bayes). Furthermore, we unsuccessfully tried to increase the decision tree's generalization ability by post pruning it (20). In the end, neural networks, using the "adam" solver, a constant learning rate, a logistic activation function, a value for alpha (the L2 regularization term) of 0.001, one hidden layer with one hundred neurons, and a batch size of five hundred, yielded the highest f1 score among single models, followed by the Bernoulli naive Bayes classifier (with the binarize parameter set to "0.05"). Table 17 in the section K of the annexes comprises the metrics of each single model.

For the ensembles, we performed a single grid search for random forests, while the remaining algorithms were optimized manually for the same reason presented above. The top-performing bagging model (based on the f1 score on the validation dataset) used the previous neural network as the base estimator and the best adaboost, a Bernoulli naive Bayes classifier. For stacking, a combination of the bagging, neural networks and Bernoulli naive Bayes provided the best results, while we obtained the best score for voting with f1-weighted hard votes for the bagging, neural networks, Bernoulli naive Bayes, linear SVC, and random forest models (more details on voting weights in section J in the annexes). In the end, bagging was the model with the highest f1 score, followed by voting by a very narrow margin. Check Table 1 for the validation dataset scores that were relevant for posterior decision of the best model.

Table 1: Binary Classification Metrics for Decision-Making (validation set results sorted by f1 score)

Model	Precision	Recall	F1 Score	Kaggle Score
Bagging	0.228051	0.435639	0.299380	0.3171
Voting	0.235055	0.412159	0.299376	0.3205
Stacking	0.227534	0.431027	0.297842	0.3189
Neural networks	0.226010	0.436478	0.297811	0.3191
Bernoulli naive Bayes	0.229053	0.421803	0.296887	0.3135
Linear SVC	0.227185	0.396646	0.288899	0.3131
SVM (polynomial kernel)	0.236252	0.371069	0.288697	0.3153
Stochastic gradient descent	0.229965	0.387421	0.288615	0.3172
Logistic regression	0.229228	0.388679	0.288381	0.3152
Adaboost	0.229484	0.385744	0.287770	0.3130
Gaussian naive Bayes	0.224228	0.395807	0.286277	0.3161
Random forest	0.227735	0.384906	0.286160	0.3075
Gradient boosting	0.230077	0.375262	0.285259	0.3129
Decision tree	0.215797	0.367715	0.271980	0.2834

Source: Authors.

Final Model Selection and Main Findings

After training all models, we unsuccessfully attempted to increase the scores of the two top-performing single (neural networks and Bayesian naive Bayes) and ensemble models (bagging and voting), by performing a new RFE tailored to each. Thus, we kept the same optimal subset of variables and chose the bagging classifier, using neural networks as the base estimator, repeated ten times using 20% of randomly selected (bootstrapped) variables and 10% of the samples, drawn without replacement.

Despite not presenting an f1 score that makes the decision more unequivocal (considering its similarity to the one achieved by the voting classifier), this model poses some advantages that make it the one that best fits our purpose. Given the nature of this project, having a higher recall is more important than a greater value for precision, since failing to predict patients that end up being readmitted to the hospital is more serious than the opposite situation. Considering that bagging, when compared to voting, achieved a higher score for recall, this enhances our confidence in its choice as the best model. Nonetheless, we used the voting classifier as our final submission on Kaggle since it provided the best f1 score.

Finally, it is worth noting that interpretability is often a concern for medical staff using ML models in their tasks (11,21). Thus, using the bagging model with a neural network as a base estimator compromises the interpretability due to the black box issue embedded in deep learning models (22). Consequently, if interpretability is to be prioritized, we recommend using the Bernoulli naive Bayes instead, our second best-performing single model, as it offers a more straightforward understanding of the outcomes (23,24).

As a final note for this chapter, we would like to highlight that class 1 seemed to be misrepresented in all the models, displaying the lowest f1 scores. A display of this phenomenon can be seen in the Annex's section Q.

IV. Multiclass Classification

Over-sampling and Feature Selection

When it came to the multiclass segment, whenever possible, we followed a similar approach to what had been done in the binary classification for consistency purposes. Evidence shows that multiclass classification in imbalanced datasets is usually more challenging than binary classification due to class overlapping, thus deteriorating the classifiers' performance. Consequently, oversampling becomes imperative (25). Therefore, we employed SMOTE this time to balance our data, as SVMSMOTE was transforming observations to have negative or even missing values, thus influencing our models' performance and not allowing us to use multinomial naive Bayes that can only take positive inputs.

We applied the same filter, wrapper, and embedded methods employed in the binary classification to select the most appropriate variables. From the first group, Pearson and Spearman's Rank showed similar results to those in the binary chapter – something later contradicted by the remaining methods, resulting in the vote to remove seven variables. Given the immediate higher feature importance obtained in the following two groups, we adjusted ANOVA to become stricter and identify the twenty-one most important variables, which is half of the available variables. Regarding the wrapper methods, the RFE, linear SVC, decision trees, and random forests were once again applied, following the same principles as previously mentioned, but with adjustments: The decision trees and random forest had their threshold pushed to 0.04, given the higher feature importance; the RFE with logistic regression maintained the same parameters as before, only adjusting the multi_class parameter to “auto” to fit the multiclass context; and, in the linear SVC, only the C parameter was adjusted, as the ones previously used (i.e., “0.5” and “0.2”) considered all variables as relevant, thus having its final value lowered to “0.05”. Lastly, LASSO was the embedded method employed, with the same threshold value of 0.1, this time selecting only 39. After applying these eight models, the results were combined in Table 18 displayed in section L of the Annex, where we decided to keep the variables voted by five or more models to remain in the dataset, maintaining, in the end, 24 variables. The higher feature relevance in this stage came as no surprise, as multiclass classification problems tend to generate decision boundaries that are more complex than in binary classifications, resulting in higher feature importance (26).

Modeling

The next step was to implement and develop multiclass models. As our goal was to obtain the model that best predicted the different classes of our problem, we computed several metrics to allow a fair comparison between them. We decided to use a weighted f1 score, precision, and recall, as the macro version of these metrics revealed the worst results each time, we employed them. Studies reveal that weighted metrics for multiclass classifications tend to be more reliable than macro metrics, namely when working with imbalanced datasets (27). Nevertheless, we computed the same metrics for all models using the macro averages, which are displayed in Annex's section P, with a detailed explanation of the difference between these methods. Accuracy was also considered a reliable metric to compare the models, but this time, we did not compute ROC scores as they are mostly applied in binary contexts and, when used on multiclass classifications, they commonly overlook the performance of minority classes (28).

We began by implementing different single models and proceeded to test ensemble methods with the models that returned the highest f1 scores, while simultaneously considering the other mentioned metrics. To guide us through this process, we followed the same models that were applied in previous studies on multiclass classification problems under medical contexts, namely logistic regression (29), decision trees (30,31), neural networks (32), and linear SVM (33), as it was less computational heavy than the standard SVM. We decided not to apply any KNN algorithm for the same reason mentioned in the binary chapter. Given its recognition and potential when it comes to making predictions and classifications in different fields, namely clinical medicine

(34), we decided to give naive Bayes models an opportunity by implementing Gaussian naive Bayes (35), Bernoulli naive Bayes, multinomial naive Bayes (36), and complement naive Bayes (37) (check section M in the annexes for details regarding the latter two models). Due to computational constraints, for this task, randomized searches were applied to all models instead of grid searches, where, to reduce the implication of not making a full exploration of the optimal parameters, we adjusted the n_iter and CV parameters to give us the best possible solutions in a shorter time. For more details on this approach, please refer to section N in the Annex.

This time, the single models seemed to better predict the classes than in the previous binary chapter. After using the best parameters for each model and applying Post Pruning to the decision tree to reduce the previous overfitting and increase its f1 score, decision tree became our model with the highest f1 score, counting for approximately 0.5718, followed by neural networks with an f1 score of around 0.5665. For a better understanding, these results were summarized in Table 19 in the Annex's section O.

When it comes to the ensemble methods employed, we decided to follow the ones used for binary classification purposes, which also revealed promising performances in past multiclass classification problems, namely random forest (38), adaboost (39), bagging (40), gradient boosting (41), preferring once again histogram-based gradient boosting, stacking (42), and voting classifiers (43). Regarding these models, the bagging classifier used multiclass neural networks as its base estimator, which, from all models tested to be the base estimators, was the one that got a higher f1 score on the validation set. To pursue the same objective, the stacking and voting models were built using all models that returned the highest f1 scores while also including Gaussian naive Bayes as an estimator, due to their superior performance compared to the remaining Bayesian models tested, to increase their predictive power. In the stacking method, a combination of random forest, bagging with a multiclass neural network base estimator (with the remaining parameters set as default, allowing the stacking model to run faster), neural network, Gaussian naive Bayes, and decision tree made this model the second best. However, using this same model combined with a decision tree, random forest, bagging, and gradient boosting allowed voting to become our top-performance model, with an f1 score in the validation set of approximately 0.5842. All the models and their metrics were summarized in Table 2, allowing for a quick analysis of the achieved results from simple models to ensemble methods. It is worth mentioning that because we used a weighted approach, accuracy and recall have the same value (44).

Table 2: Multiclass Classification Metrics for Decision-Making (validation set results sorted by f1 score)

Model	Accuracy	Precision	Recall	F1 Score
Voting	0.614244	0.582200	0.614244	0.584186
Stacking	0.590239	0.581618	0.590239	0.578845
Bagging	0.600955	0.573788	0.600955	0.576780
Random forest	0.605961	0.574438	0.605961	0.573433
Decision tree	0.597492	0.565042	0.597492	0.571757
Gradient boosting	0.613916	0.575112	0.613916	0.570841
Neural networks	0.588648	0.569965	0.588648	0.566490
Adaboost	0.611670	0.566602	0.611670	0.560738
Logistic regression	0.569791	0.575822	0.569791	0.536025
Gaussian naive Bayes	0.515699	0.553799	0.515699	0.528996
Linear SVC	0.572130	0.583385	0.572130	0.523747
Bernoulli naive Bayes	0.508586	0.546284	0.508586	0.511442
Multinomial naive Bayes	0.546207	0.582282	0.546207	0.472359
Complement naive Bayes	0.536054	0.581169	0.536054	0.444275

Source: Authors.

Final Model Selection and Main Findings

When analyzing Table 2, it becomes clear that our best-performing model is voting, achieving the highest scores for the selected metrics compared to the other models, except for precision, which is slightly inferior to the linear SVC and multinomial naive Bayes, with a difference of 0.001484 and 0.000261, respectively. As such, if the hospital staff's goal is to make accurate and reliable predictions on which patients will be or not be readmitted in less or more than 30 days, we suggest using the voting model's result.

However, as mentioned before, interpretability is a considerable concern when applying ML models in a medical context (11,21). Therefore, if the staff wants to have a clear understanding of these results and how different factors are influencing them, the best option would be to use the decision tree, as it was our single model with the highest f1 score and still considerable results on the remaining metrics, and is a method known for its simple and easy results' interpretability (45).

Given our already satisfactory results in the different metrics and the past unsuccessfulness in increasing the f1 score when applying a new feature selection for our top-performing models on the binary classification, we decided not to perform feature selection a second time on the best models in the multiclass classification. Instead, we proceeded to make the predictions for the test dataset using our voting classifier, similar to what we did in the binary classification.

To conclude this section, it is worth highlighting that in multiclass classification, we noticed that class "1" (i.e., people readmitted within 30 days) was slightly misrepresented, displaying lower f1 scores for all the tested models, displayed in Annex's section R, just like it happened in binary classification. Nevertheless, all models employed revealed greater efficacy in making class predictions within a multiclass classification context as opposed to a binary classification one.

V. Conclusion

We can summarize our findings as follows: First, regarding the binary classification problem, voting provided better scores for the test set, justifying the choice of this model as our final submission to the Kaggle competition. Second, voting and bagging provided very similar f1 scores in the validation set, but bagging's recall offered crucial information showing that it is a more reliable model for obtaining correct predictions (i.e., patients that will indeed be readmitted). Third, Bernoulli naive Bayes emerged as a suitable option in the binary classification if interpretability is a must for the clinical staff. Fourth, voting stood out as our top-performing and most reliable multiclass model in the validation set. Fifth, the decision tree appeared as an alternative option in the multiclass if the medical staff aimed to prioritize interpretability over reliability.

Addressing the readmission problem through different lenses provides peculiar findings: First, we expected models in both binary and multiclass to provide decent predictions, which was not the case for the former. Although a direct comparison with previous studies is misleading, the ROC scores obtained for our models (Table 16 in section H) are in the same range as most of those reported in previous studies (1,9,10). However, we did not expect the f1 scores to be very low, especially compared to Ramírez and Herrera (10). For instance, our logistic regression had a f1 score of 0.2883, while the authors had 0.5550. A possible explanation for such a difference lies in the characteristics of our dataset, which differs from the original diabetes dataset in terms of available variables, as well as in the decision to keep all the records instead of dropping some. Second, we succeeded in offering models in both cases different from logistic regression. As expected, in both perspectives, we had more complex models capable of capturing non-linear relationships, outperforming the mentioned model, which is simpler. However, we also expected to find those complex models harder to explain in terms of the decision-making behind the outcomes. Therefore, we found it important to recommend explainable models (i.e., Bernoulli naive Bayes for the binary and decision tree for the multiclass) since interpretability is often a concern among clinical staff in several medical conjunctures (11,21).

Regarding limitations, the project has the following: First, the lack of domain knowledge in the healthcare sector and the specificities of diabetes hinder the decision-making process. For instance, it is hard to assess if having binary variables for the glucose and A1C tests stating the presence or absence of test results (i.e., *took_glucose_test* and *took_a1c_test*) is preferable to the initial values stating the results for each (i.e., *glucose_test_result* and *a1c_test_result*). Furthermore, there is uncertainty regarding how precise the grouping logic followed for categorizing the severity of diabetes is (Table 9 in section E in the annexes). Second, computational resources pose a substantial constraint that may have negatively influenced our results. For example, even though it is well-known that k-fold CV is an ideal choice when splitting the dataset, the need for computational power also increases (46). Unfortunately, we faced severe problems even in solutions like Google Colab and Microsoft Azure by running out of available compute units before finishing executing the notebook. Such constraints also reflect on choices related to performing randomized rather than grid searches in multiclass classification. Third, our strategy concerning avoiding creating variables as much as possible could lead to undesired outcomes. For instance, considering the variable *primary_diagnosis_freq*, having the values related to the primary diagnosis encoded through frequency encoding might induce the models to prioritize certain diagnoses over others. Such an observation is also valid for variables like *secondary_diagnosis_freq* and *additional_diagnosis_freq*.

Lastly, besides considering the mentioned limitations, future projects should also test the models in the wild with the medical staff to better assess and validate the results (a step out of the scope of our project). The business literature has studies showing how vital such a step is, especially in the medical context (11,21). However, evidence suggests that such a point is still scarce (8). Furthermore, as one class seemed misrepresented (i.e., readmitted patients in the binary context and readmitted within 30 days in the multiclass classification) by all models in both classification problems, we suggest that, to improve future model results' reliability and accuracy, the projects should include more patients belonging to those classes, for instance, by employing a stratified random sampling method.

Despite the drawbacks, we are confident we succeeded in proposing the most reliable models possible with consistency and transparency in terms of choices and reasoning for keeping or dropping a variable, as well as parameters used in feature selection and model testing for the available dataset.

References

1. Shang Y, Jiang K, Wang L, Zhang Z, Zhou S, Liu Y, et al. The 30-days hospital readmission risk in diabetic patients: predictive modeling with machine learning classifiers. BMC Med Inform Decis Mak [Internet]. 2021;21(57):1–11. Available from: https://doi.org/10.1186/s12911-021-01423-y

2. Sheetrit E, Brief M, Elisha O. Predicting unplanned readmissions in the intensive care unit: a multimodality evaluation. Sci Rep [Internet]. 2023;13(1):1–9. Available from: https://doi.org/10.1038/s41598-023-42372-y

3. Eby E, Hardwick C, Yu M, Gelwicks S, Deschamps K, Xie J, et al. Predictors of 30 day hospital readmission in patients with type 2 diabetes: A retrospective, case-control, database study. Curr Med Res Opin [Internet]. 2015;31(1):107–14. Available from: https://doi.org/10.1185/03007995.2014.981632

4. Rubin DJ, Shah AA. Predicting and Preventing Acute Care Re-Utilization by Patients with Diabetes. Curr Diab Rep [Internet]. 2021;21–34. Available from: https://doi.org/10.1007/s11892-021-01402-7

5. Rubin DJ, Donnell-Jackson K, Jhingan R, Golden SH, Paranjape A. Early readmission among patients with diabetes: A qualitative assessment of contributing factors. J Diabetes Complications [Internet]. 2014;28(6):869–73. Available from: http://dx.doi.org/10.1016/j.jdiacomp.2014.06.013

6. Centers for Medicare and Medicaid Services (CMS). Hospital Readmissions Reduction Program (HRRP) [Internet]. Available from: https://www.cms.gov/medicare/payment/prospective-payment-systems/acute-inpatient-pps/hospital-readmissions-reduction-program-hrrp

7. Artetxe A, Beristain A, Graña M. Predictive models for hospital readmission risk: A systematic review of methods. Comput Methods Programs Biomed [Internet]. 2018;164:49–64. Available from: https://doi.org/10.1016/j.cmpb.2018.06.006

8. Long J, Wang M, Li W, Cheng J, Yuan M, Zhong M, et al. The risk assessment tool for intensive care unit readmission: A systematic review and meta-analysis. Intensive Crit Care Nurs [Internet]. 2023;76(103378). Available from: https://doi.org/10.1016/j.iccn.2022.103378

9. Frizzell JD, Liang L, Schulte PJ, Yancy CW, Heidenreich PA, Hernandez AF, et al. Prediction of 30-day all-cause readmissions in patients hospitalized for heart failure: Comparison of machine learning and other statistical approaches. JAMA Cardiol. 2017;2(2):204–9.

10. Ramírez JC, Herrera D. Prediction of Patient Readmission Using Machine Learning Techniques. Commun Comput Inf Sci. 2019;41–4.

11. Lebovitz S, Levina N, Lifshitz-Assaf H. Is AI ground truth really true? The dangers of training and evaluating AI tools based on experts' know-what. MIS Q Manag Inf Syst. 2021;45(3):1501–25.

12. Strack B, DeShazo JP, Gennings C, Olmo JL, Ventura S, Cios KJ, et al. Impact of HbA1c measurement on hospital readmission rates: Analysis of 70,000 clinical database patient records. Biomed Res Int [Internet]. 2014;2014:1–10. Available from: http://dx.doi.org/10.1155/2014/781670

13. Treece J, Ghouse M, Rashid S, Arikapudi S, Sankhyan P, Kohli V, et al. The Effect of Hospice on Hospital Admission and Readmission Rates: A Review. Home Heal Care Manag Pract [Internet]. 2018;30(3):140–6. Available from: https://doi.org/10.1177/1084822318761105

14. Gosain A, Sardana S. Handling Class Imbalance Problem Using Feature Selection Techniques: A Review. In: 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI). 2017. p. 79–85.

15. Wang L, Han M, Li X, Zhang N, Cheng H. Review of Classification Methods on Unbalanced Data Sets. IEEE Access [Internet]. 2021;9:64606–28. Available from: https://doi.org/10.1109/ACCESS.2021.3074243

16. Nguyen HM, Cooper EW, Kamei K. Borderline Over-sampling for Imbalanced Data Classification. In: Fifth International Workshop on Computational Intelligence & Applications [Internet]. 2009. p. 24–9. Available from: https://doi.org/10.1504/IJKESDP.2011.039875

17. Ng AY. Feature selection, L1 vs. L2 regularization, and rotational invariance. In: Proceedings, Twenty-First International Conference on Machine Learning, ICML 2004 [Internet]. 2004. p. 615–22. Available from:

https://doi.org/10.1145/1015330.1015435

18. Grąbczewski K, Jankowski N. Feature selection with decision tree criterion. In: Proceedings - HIS 2005: Fifth International Conference on Hybrid Intelligent Systems [Internet]. 2005. p. 212–7. Available from: https://doi.org/10.1109/ICHIS.2005.43

19. Chen R-C, Dewi C, Huang S-W, Caraka RE. Selecting critical features for data classification based on machine learning methods. J Big Data [Internet]. 2020;7(1). Available from: https://doi.org/10.1186/s40537-020-00327-4

20. Shamrat FMJM, Chakraborty S, Billah M. M, Das P, Muna JN, Ranjan R. A comprehensive study on pre-pruning and post-pruning methods of decision tree classification algorithm. In: Proceedings of the 5th International Conference on Trends in Electronics and Informatics, ICOEI 2021 [Internet]. IEEE; 2021. p. 1339–45. Available from: https://doi.org/10.1109/ICOEI51242.2021.9452898

21. Lebovitz S, Lifshitz-Assaf H, Levina N. To Engage or Not to Engage with AI for Critical Judgments: How Professionals Deal with Opacity When Using AI for Medical Diagnosis. Organ Sci. 2022;33(1):126–48.

22. Lee I, Shin YJ. Machine learning for enterprises: Applications, algorithm selection, and challenges. Bus Horiz [Internet]. 2020;63(2):157–70. Available from: https://doi.org/10.1016/j.bushor.2019.10.005

23. Gelman A, Carlin JB, Stern HS, Dunson DB, Vehtari A, Rubin DB. Probability and inference. In: Bayesian Data Analysis: Third edition. 3rd Editio. 2021. p. 3–28.

24. Wang S, Jiang L, Li C. Adapting naive Bayes tree for text classification. Knowl Inf Syst [Internet]. 2015;44(1):77–89. Available from: https://doi.org/10.1109/ICOEI51242.2021.9452898

25. Sáez JA, Krawczyk B, Woźniak M. Analyzing the oversampling of different classes and types of examples in multi-class imbalanced datasets. Pattern Recognit [Internet]. 2016;57:164–78. Available from: https://doi.org/10.1016/j.patcog.2016.03.012

26. Ma G, Lu J, Liu F, Fang Z, Zhang G. Multiclass Classification With Fuzzy-Feature Observations: Theory and Algorithms. IEEE Trans Cybern. 2022;PP:1–14.

27. De Diego IM, Redondo AR, Fernández RR, Navarro J, Moguerza JM. General Performance Score for classification problems. Appl Intell [Internet]. 2022;52(10):12049–63. Available from: https://doi.org/10.1007/s10489-021-03041-7

28. Yang Z, Xu Q, Bao S, Cao X, Huang Q. Learning with Multiclass AUC: Theory and Algorithms. IEEE Trans PATTERN Anal Mach Intell Learn [Internet]. 2022;44(11):1–53. Available from: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.recall_score.html

29. Sultana J, Khader Jilani A. Predicting Breast Cancer Using Logistic Regression and Multi-Class Classifiers. Int J Eng Technol [Internet]. 2018;7(4.20):22–6. Available from: https://doi.org/10.14419/ijet.v7i4.20.22115

30. Nuti G, Jiménez Rugama LA, Cross AI. An Explainable Bayesian Decision Tree Algorithm. Front Appl Math Stat. 2021;7(March):1–9.

31. Azar AT, El-Metwally SM. Decision tree classifiers for automated medical diagnosis. Neural Comput Appl. 2013;23(7–8):2387–403.

32. Mall PK, Singh PK, Srivastav S, Narayan V, Paprzycki M, Jaworska T, et al. A comprehensive review of deep neural networks for medical image processing: Recent developments and future opportunities. Healthc Anal [Internet]. 2023;4(April):100216. Available from: https://doi.org/10.1016/j.health.2023.100216

33. Uddin S, Khan A, Hossain ME, Moni MA. Comparing different supervised machine learning algorithms for disease prediction. BMC Med Inform Decis Mak. 2019;19(1):1–16.

34. Chen H, Hu S, Hua R, Zhao X. Improved naive Bayes classification algorithm for traffic risk management. EURASIP J Adv Signal Process. 2021;2021(1).

35. Iqbal Z, Yadav M. Multiclass Classification with Iris Dataset using Gaussian Naive Bayes. Int J Comput Sci Mob Comput [Internet]. 2020;9(4):27–35. Available from: https://www.ijcsmc.com/past_issues/volume_9_issue_4

36. Kalcheva N, Marinova G, Todorova M. Comparative Analysis of the Bernoulli and Multinomial Naive Bayes Classifiers for Text Classification in Machine Learning. In: (ICAI), 2023 International Conference Automatics and Informatics. 2023. p. 28–31.

37. Anagaw A, Chang Y-L. A new complement naïve Bayesian approach for biomedical data classification. J Ambient Intell Humaniz Comput [Internet]. 2019;10(10):3889–97. Available from: http://dx.doi.org/10.1007/s12652-018-1160-1

38. Kuzu SY. Random Forest Based Multiclass Classification Approach for Highly Skewed Particle Data. J Sci Comput [Internet]. 2023;95(1):1–17. Available from: https://doi.org/10.1007/s10915-023-02144-2

39. Hao W, Luo J. Generalized multiclass AdaBoost and its applications to multimedia classification. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. 2006. p. 113.

40. Injadat M, Moubayed A, Nassif AB, Shami A. Multi-split optimized bagging ensemble model selection for multi-class educational data mining. Appl Intell. 2020;50(12):4506–28.

41. Tanha J, Abdi Y, Samadi N, Razzaghi N, Asadpour M. Boosting methods for multi-class imbalanced data classification: an experimental review. J Big Data [Internet]. 2020;7(70):1–47. Available from: https://doi.org/10.1186/s40537-020-00349-y

42. Lorbieski R, Nassar SM. Impact of an extra layer on the stacking algorithm for classification problems. J Comput Sci. 2018;14(5):613–22.

43. Rojarath A, Songpan W. Cost-sensitive probability for weighted voting in an ensemble model for multi-class classification problems. Appl Intell. 2021;51(7):4908–32.

44. scikit-learn. sklearn.metrics.recall_score [Internet]. Available from: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.recall_score.html

45. Blockeel H, Devos L, Frénay B, Nanfack G, Nijssen S. Decision trees: from efficient prediction to responsible AI. Front Artif Intell. 2023;6.

46. Yadav S, Shukla S. Analysis of k-fold cross-validation over hold-out validation on colossal datasets for quality classification. In: Proceedings - 6th International Advanced Computing Conference, IACC 2016. IEEE; 2016. p. 78–83.

Annexes

Section A: Variables' Metadata

Table 3 below comprises the type of information, the variable's name, description, and datatype for each variable we initially had in our dataset.

Table 3: List of Variables

Information	Variable	Description	Type
Encounter details	*encounter_id*	Encounter's unique identifier	int64
	patient_id	Patient's identifier	int64
	admission_type	Type of patient's admission (e.g., emergency, urgent, etc...)	object
	admission_source	The department the patient was in before being admitted to the current encounter	object
	medical_specialty	Medical specialty in which the patient was admitted	object
	discharge_disposition	Patient's destination after discharge	object
	length_of_stay_in _hospital	Number of days in hospital between admission and discharge	int64
Patient demographic	*country*	Patient's country	object
	race	Patient's race	object
	gender	Patient's gender	object
	age	Patient's age bracket	object
	weight	Patient's weight bracket	object
	payer_code	Health insurance provider code, if any	object
Medical record	*outpatient_visits_in _previous_year*	Number of outpatient visits (i.e., with the intention of leaving on the same day) to the hospital a year before the encounter	int64
	emergency_visits_in _previous_year	Number of emergency visits to the hospital a year before the encounter	int64
	inpatient_visits_in _previous_year	Number of inpatient visits (i.e., with the intention to stay overnight) to the hospital a year before the encounter	int64
	average_pulse_bpm	Patient's average pulse during their stay (in beats per minute)	int64
	number_lab_tests	Number of lab tests taken in the encounter	int64
	non_lab_procedures	Number of non-lab procedures in the encounter	int64
	number_of _medications	Number of distinct medications taken by the patient during the encounter	int64

	primary_diagnosis	Patient's primary diagnosis (based on the ICD9 first three digits)	object
	secondary_diagnosis	Patient's secondary diagnosis (based on the ICD9 first three digits)	object
	additional_diagnosis	Patient's additional secondary diagnosis (based on the ICD9 first three digits)	object
	number_diagnoses	Number of diagnoses entered into the system during the encounter	int64
	glucose_test_result	Glucose test results. Possible values: ">200," ">300," "normal," and "none" if not measured.	object
	a1c_test_result	A1C test results. Possible values: ">7", ">8", "normal", and "none" if not measured.	object
	change_in_meds _during _hospitalization	Whether the diabetic medications (dosage or generic name) changed. Possible values: "change" and "no change"	object
	prescribed_diabetes _meds	Yes, if the patient has diabetes medication prescribed. Otherwise, no.	object
	medication	A list with all medications' generic names prescribed to the patient in the encounter. The list is empty in the absence of prescribed medications.	object
Target variables	*readmitted_binary*	Binary target: "Yes", if the patient was readmitted within 30 days post-discharge; "No" otherwise.	object
	readmitted_multiclass	Multiclass target: "≤30 days" if the patient was readmitted within the 30-day timespan; ">30 days" if the patient was readmitted more than 30 days after discharge; "no" if the patient was not readmitted.	object

Source: Adapted from the project's guidelines.

Section B: Data Exploration

Before applying any preprocessing to the variables, it is imperative to begin by exploring the characteristics of our data, as this allows us to deeply understand the topic and how each variable should be treated, what inconsistencies to address, how missing values should be managed, among many other factors (1).

It is worth emphasizing that all concerns identified in this section have been detailed with explanations of how we solved them in the following annex sections. We began by making a superficial exploration to gain insights about the datatypes, missing values, and possible inconsistencies in our data to advance to a visual analysis of our variables to identify patterns, distributions, and frequencies.

We immediately noticed that many of the variables were of type "object", something we knew we had to solve before starting modeling as some algorithms cannot operate with nominal data. Additionally, when analyzing the number of observations in each variable, some appeared to have, for instance, missing and unusual values (Table 4), a concern that should also be addressed before the modeling phase. At this step, we decided that *encounter_id* would make a better index variable than *patient_id*, as it had all its observations as unique values.

Table 4: Unusual and Missing Values

Variable	Problem	Solution
admission_type	5.17% missing values	Replacing them with “Unknown/Other”
admission_source	6.50% missing values	Replacing them with “Not Mapped”
medical_specialty	49.32% unusual (“?”) / missing values	Transforming the “?” into missing values. After, dropping the variable since it had more than 49% missing values
discharge_disposition	3.58% missing values	Replacing them with “Not Mapped”
country	Only one value	Dropping the variable
race	2.14% unusual (“?”) / 5.05% missing values	Assigning the “?” to “Other” and missing values to “Unknown”
gender	Two observation with unusual value (“Invalid/Unknown”)	Assigning it to the variable’s mode (“Female”)
age	4.96% missing values	Replacing them with the variable’s mode (“[70-80)”)
weight	96.88% unusual (“?”) / missing values	Transforming the “?” into missing values. After, dropping the variable since it had almost 97% missing values
payer_code	39.43% unusual (“?”) / missing values	Transforming the variable into a binary one, where “0” represents the absence of health insurance, including the missing values
primary_diagnosis	0.02% unusual (“?”) / missing values	Replacing them with “Other”
secondary_diagnosis	0.36% unusual (“?”) / missing values	Replacing them with “Other”
additional_diagnosis	1.39% unusual (“?”) / missing values	Replacing them with “Other”
glucose_test_result	94.84% missing values	Transforming the variable into a binary one, where “0” represents the absence of a test result, including missing values
a1c_test_result	83.20% missing values	Transforming the variable into a binary one, where “0” represents the absence of a test result, including missing values

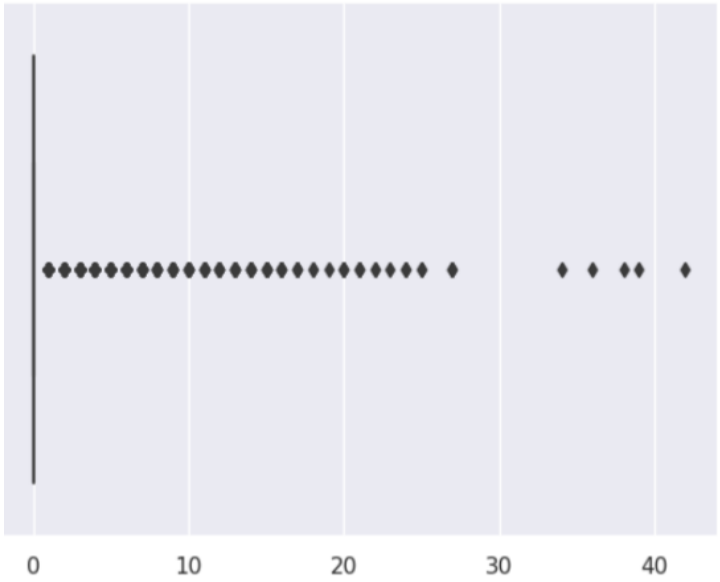
Source: Authors.

As we started exploring the number of unique categories for each variable, we noticed particular aspects: First, some variables, such as *race* and *weight*, included a category labeled with "?"; second, *country* only had one category ("USA"); third, *gender included an "Unkown/ Invalid" option; forth, age* and *weight* are represented as bins; *fifth, discharge_disposition* and *admission_source* had excessive labels; and sixth, *medical_specialty* had

too many categories (67 options). When making a final analysis of our data, it also caught our attention that the dataset did not contain any duplicated information and that, when it came to the binary and multiclass targets, "0" was the predominant value.

Outpatient_visits_in_previous_year's boxplot in Figure 1 revealed some consistency until 30 outpatient visits, and after that value, it becomes slightly more dispersed. When we analyzed what observations had more than 40 outpatient visits, the result was only one patient with 42 visits.

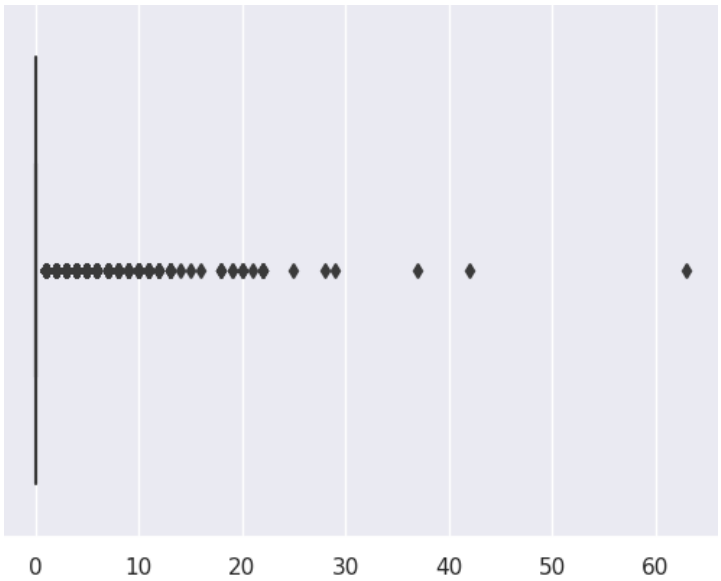
Figure 1: *outpatient_visits_in_previous_year* Box Plot



Source: Authors.

Emergency_visits_in_previous_year, however, seemed to have values even more dispersed and far from the rest, with a patient with precisely 63 emergency visits standing out from the rest, as highlighted in Figure 2.

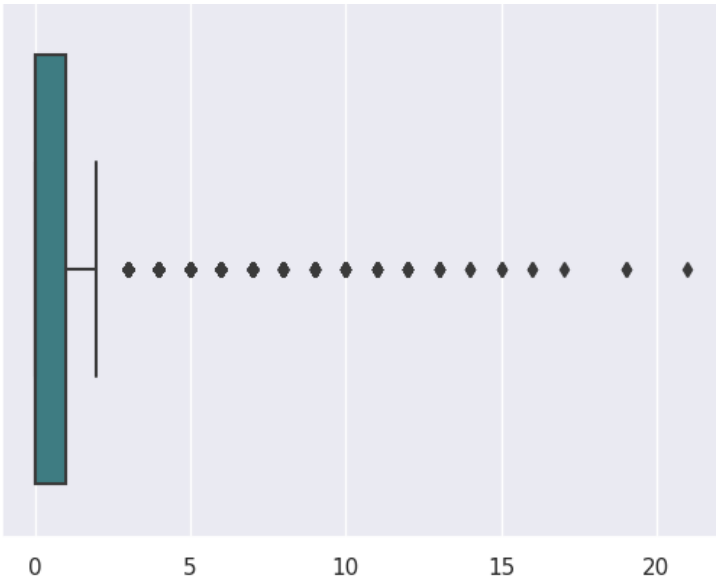
Figure 2: *emergency_visits_in_previous_year* Box Plot



Source: Authors.

Figure 3 showcases that *inpatient_visits_in_previous_year* followed a more consistent distribution, with the maximum number of inpatient visits slightly after 20.

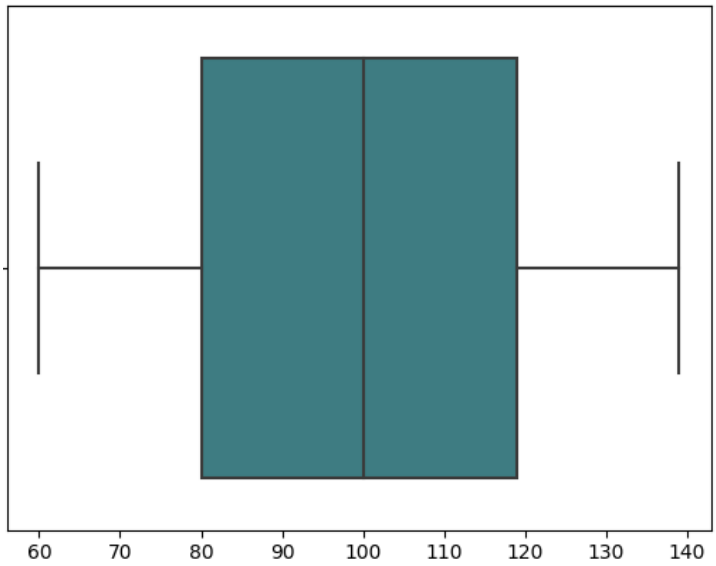
Figure 3: *inpatient_visits_in_previous_year* Box Plot



Source: Authors.

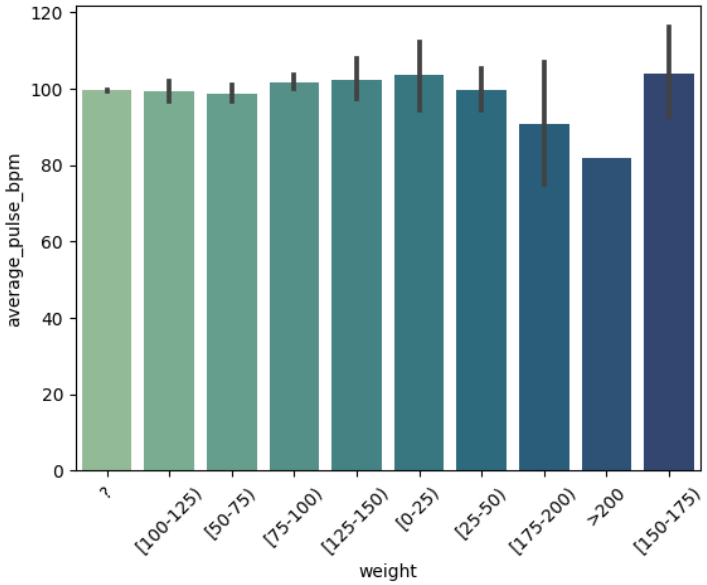
Average_pulse_bpm was perhaps the only variable that did not present values beyond the boxplot's upper and lower limits, as suggested by the boxplot in Figure 4. To extract more information regarding this variable, we plotted a bar chart to check how the average heart rate would change according to the patient's weight. From Figure 5, we can see that patient with weight class "[150 - 175)" have the highest heart rate, with values going beyond 100, which could indicate that we have individuals with more severe health conditions in our dataset.

Figure 4: *average_pulse_bpm* Box Plot



Source: Authors.

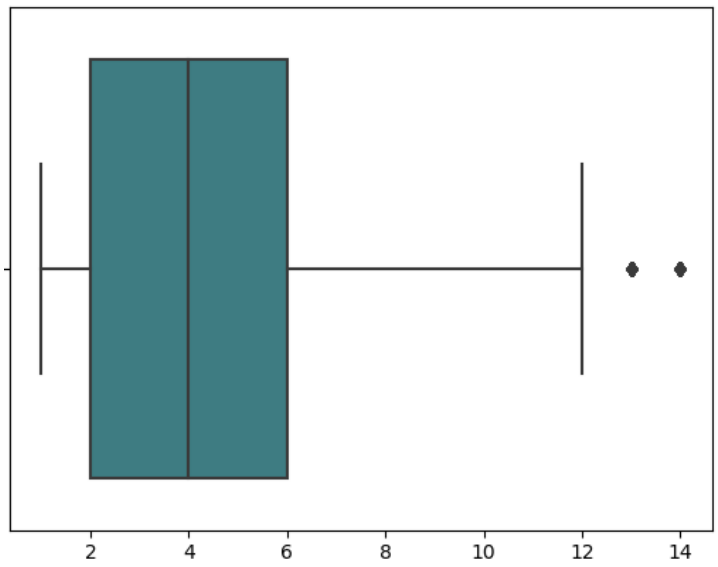
Figure 5: *average_pulse_bpm* change with *weight*



Source: Authors.

Regarding the *length_of_stay_in_hospital*, Figure 6 did not reveal any unusually extreme value, so we decided to further investigate this variable's distribution according to the patient's weight, displayed in Figure 7, and their age, plotted in Figure 8.

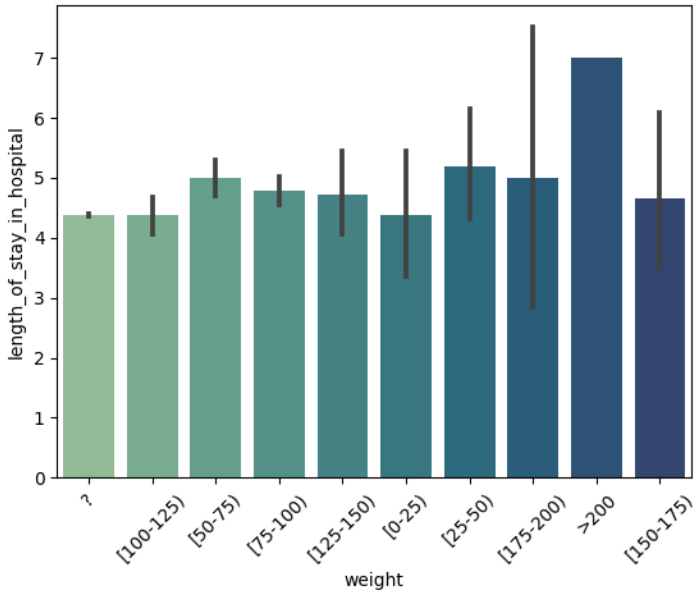
Figure 6: *lenght_of_stay_in_hospital* Box Plot



Source: Authors.

From the graph below, the patients belonging to the weight category "[175-200]" and ">200" stood out with the highest number of days spent between admission and discharge in the hospital, with 1,137 patients from the first weight category spending ten days, and only one individual with the second class weight reaching seven days in the hospital, which could be an indicator that in our dataset, people with higher weights demonstrate the most severe health conditions, as their heart rate tends to be far from the average and they spend more days in the hospital between their admission and discharge.

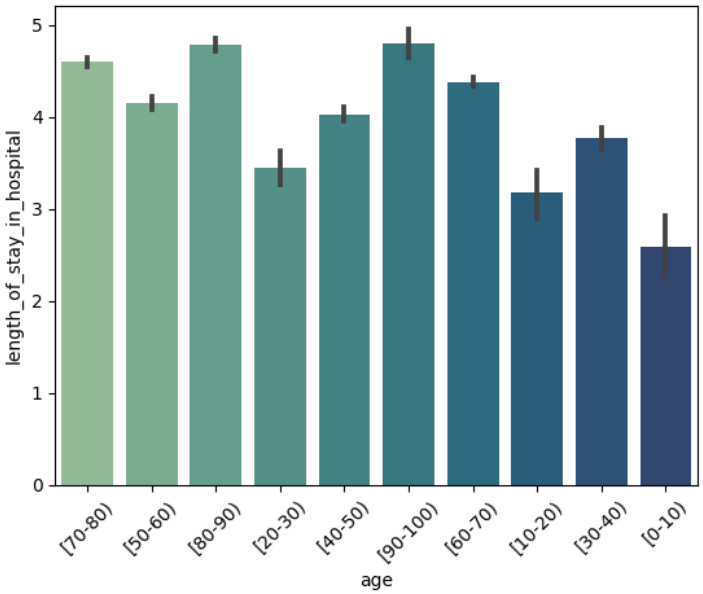
Figure 7: *length_of_stay_in_hospital* Change with *weight*



Source: Authors.

In addition, the following graph revealed that older patients in our dataset tend to spend more days in the hospital.

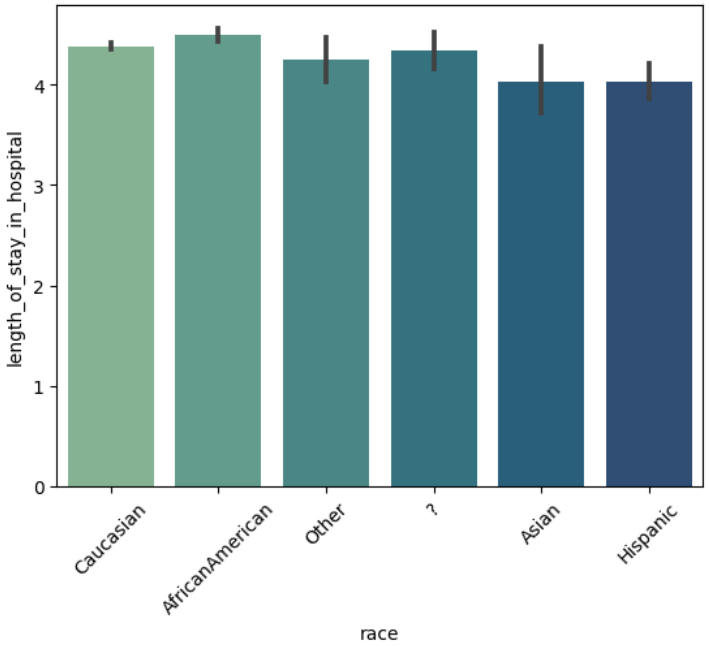
Figure 8: *length_of_stay_in_hospital* Change with *age*



Source: Authors.

Besides, we plotted in Figure 9 the *length_of_stay_in_hospital* according to the patient's race, but no pattern was found that enabled us to form any conclusion regarding these variables' relation.

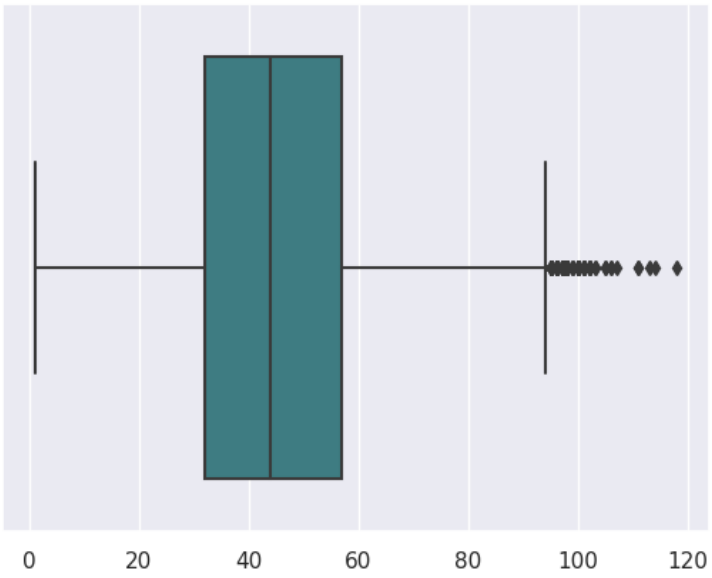
Figure 9: *length_of_stay_in_hospital* Change with *race*



Source: Authors.

The *number_lab_tests* plot in Figure 10 did not reveal any extreme value in this variable, although we noticed a tendency to have a very high number of lab tests that can go up to 120.

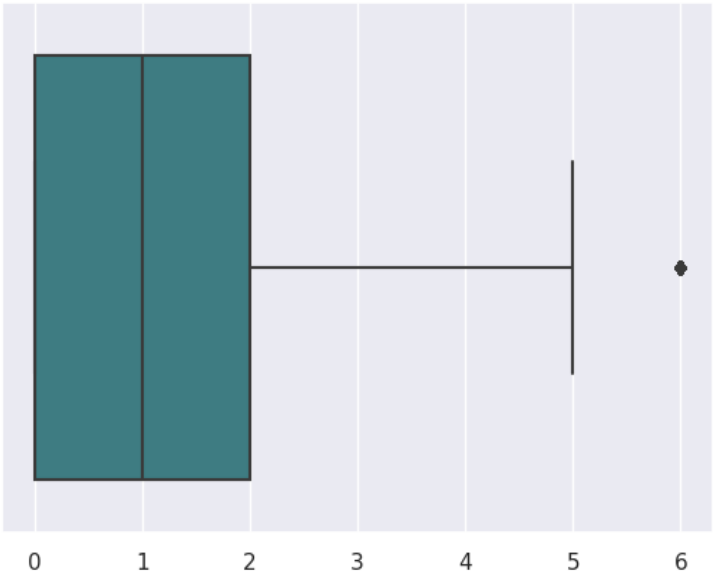
Figure 10: *number_lab_tests* Box Plot



Source: Authors.

On the other hand, *non_lab_procedures* demonstrated a very right-skewed distribution in Figure 11, but with much lower values than the previous variable, where the maximum in this one is only six.

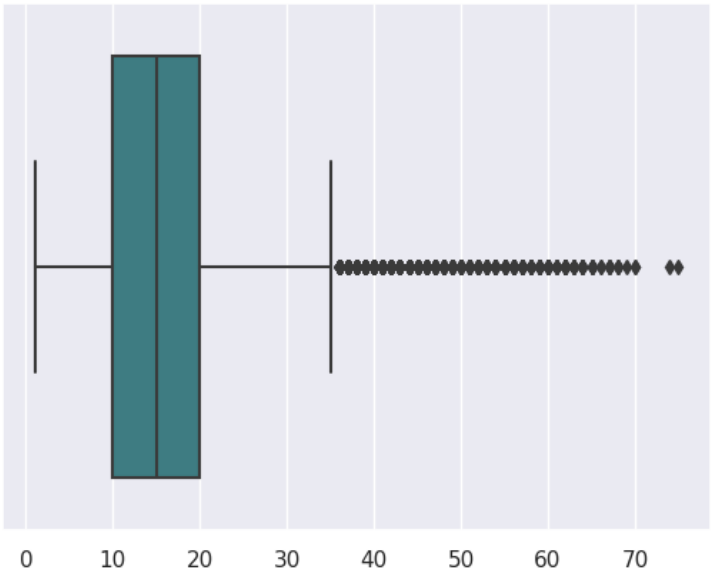
Figure 11: *non_lab_procedures* Box Plot



Source: Authors.

When we analyzed *the number_of_medications* in Figure 12, this variable displayed an extensive right-tail, with values over 70. We decided to explore the patients with this high number of medications and concluded that they belonged to observation in the "[60-70)" and "[70-80)" age groups but with unknown weight (class label equal to "?").

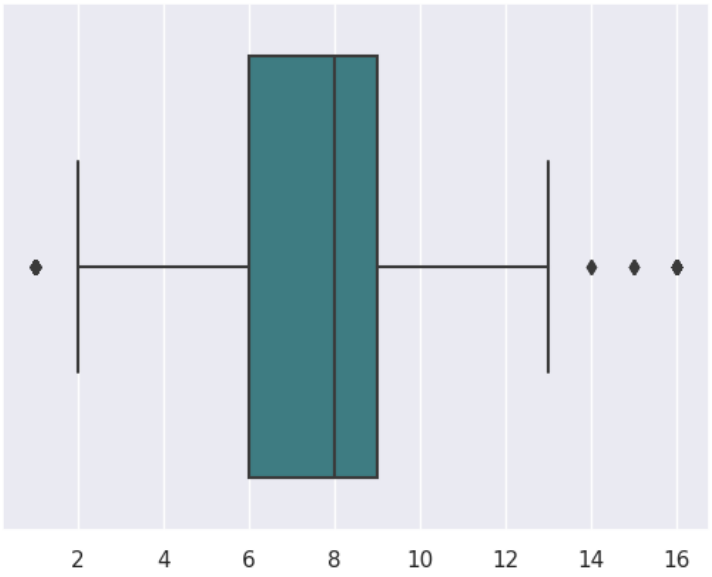
Figure 12: *number_of_medications* Box Plot



Source: Authors.

Contrary to the previous one, *number_diagnosis* exhibited a left-tail without extreme values, as displayed in Figure 13.

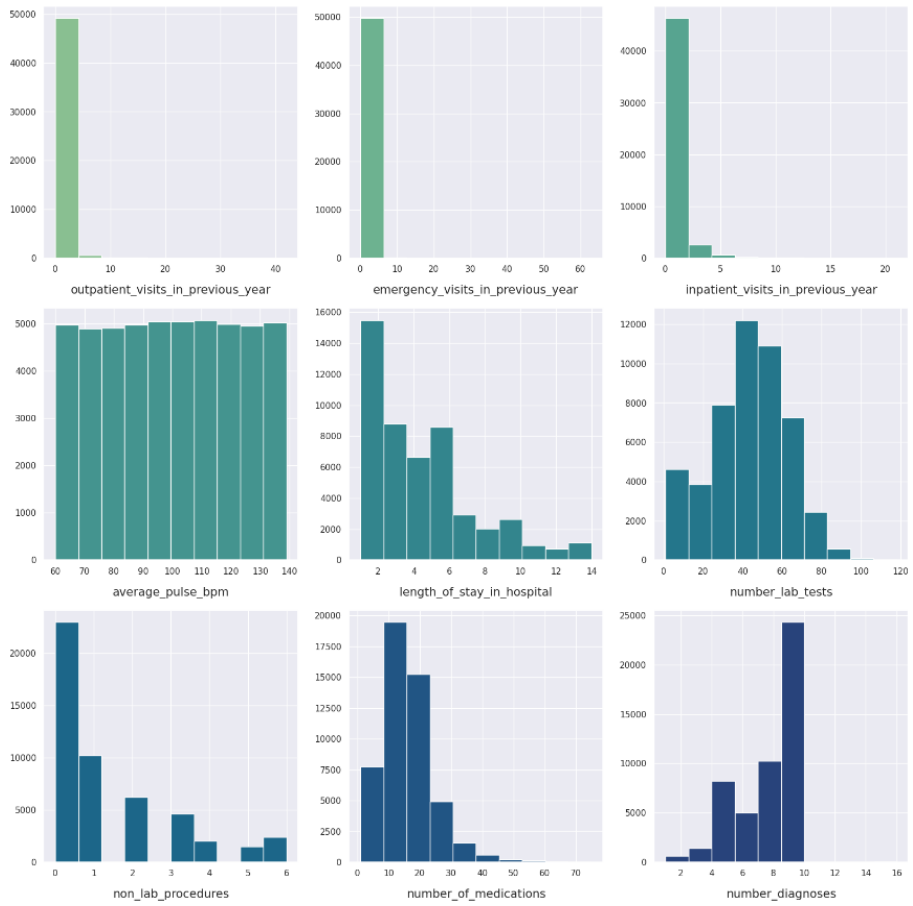
Figure 13: *number_diagnoses* Box Plot



Source: Authors.

To form our conclusions, we not only used the box plots for the numerical variables, which are extremely useful to extract quick summaries regarding the distributions and values characteristics (2), but we also plotted their histograms, as this type of graphic allows us to identify the data's shape and spread and analyze its skewness in more detail (3). All histograms used in our analysis were combined in Figure 14.

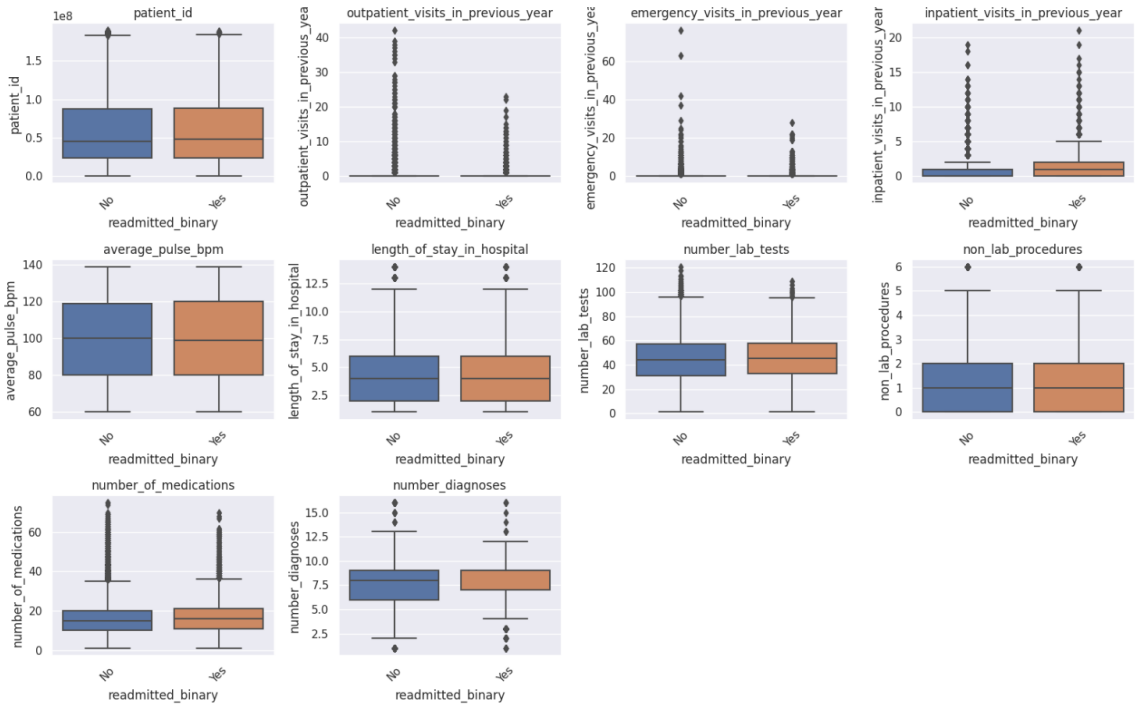
Figure 14: Variables' Histograms



Source: Authors.

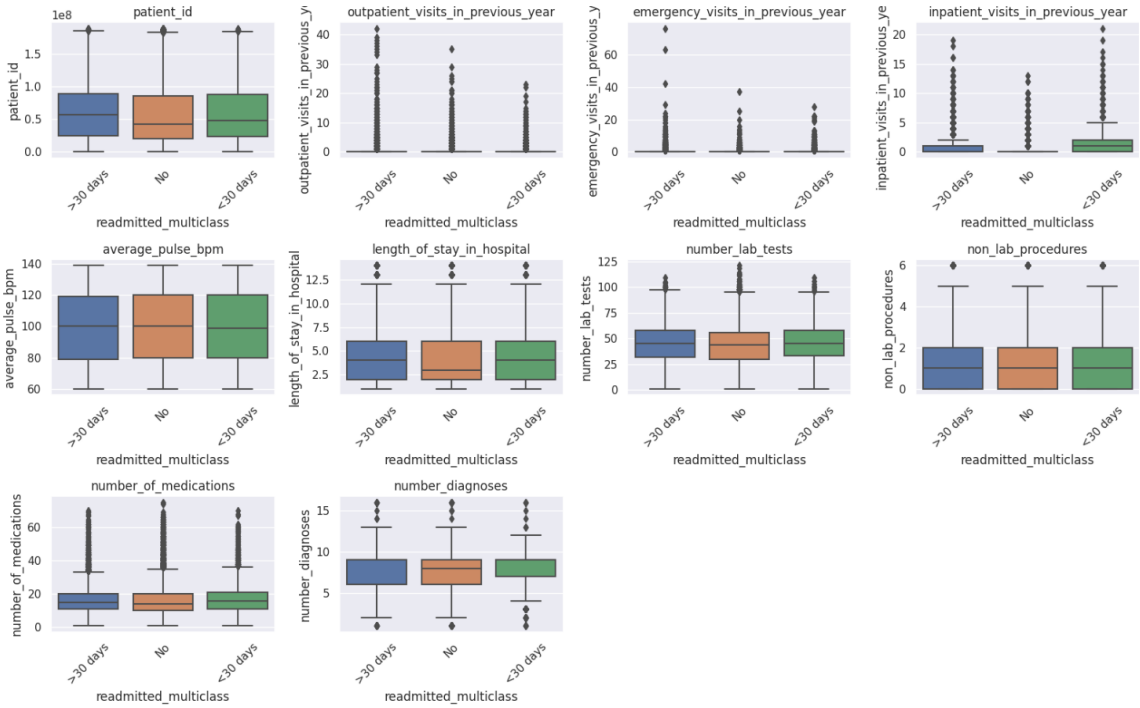
Lastly, we plotted the same variables in box plots to analyze how they changed according to the *readmitted_binary* target, displayed in Figure 15, and also according to *the readmitted_multiclass*, plotted in Figure 16.

Figure 15: Variables' Box Plot Change with *readmitted_binary*



Source: Authors.

Figure 16: Variables' Box Plot Change with *readmitted_multiclass*



Source: Authors.

After all this analysis, we decided not to remove any observation from our train dataset, as no observation was extreme enough to be considered an outlier. Furthermore, we could be missing crucial variance in our data if we removed the most extreme observations given that, in the end, our goal was to develop a model capable of predicting the patient's readmission, no matter how extreme the data we input.

References:

1. Navandar S, Bhutkar G. A Review on Data Exploration and Data Mining Evolution. Int J Creat Res Thoughts. 2022;10(10):24–32.
2. Jaiswal AS. Importance of Data Exploration in Data Analysis A Review Paper. Int J Adv Res Sci Commun Technol. 2022;2(2):1046–53.
3. Nuzzo RL. Histograms: A Useful Data Analysis Visualization. PM R. 2019;11:309–12.

Section C: Grouping the Categorical Variables' Values

The logic followed for the variable *admission_type* was to group "Emergency" and "Urgent" since they represent the same phenomenon. Furthermore, we grouped the missing values ("NaN"), "Not Available", "Not Mapped", "Trauma Center", and "Newborn" into a single value labeled "Unknown/Other". The reason lies in the low frequencies of the last two values, which have only nine and two occurrences, respectively. Consequently, having a group with 11 occurrences would add unnecessary noise to the data. It is worth mentioning that after adjusting the training set, we modified the validation and test set following the same rationale for consistency purposes. Table 5 below comprises each value and respective frequency the variable initially had in the training set, while Table 6 shows the grouped values and their frequencies.

Table 5: Initial values and Frequencies of *admission_type*

Value	Frequency
Emergency	26471
Elective	9267
Urgent	9078
NaN	2580
Not Available	2320
Not Mapped	138
Trauma Center	9
Newborn	2

Source: Authors.

Table 6: Values, Keywords Grasped, and Groups for *discharge_disposition*

Value	Frequency
Emergency	35549
Elective	9267
Unknown/Other	5049

Source: Authors.

Regarding *discharge_disposition*, the logic employed was similar to the previous variable. However, the long length of the strings and the wide range of possible values made it more challenging to group the values meaningfully. Therefore, our approach was to uncover similarities between the strings through keywords and by assessing the content of each. For instance, the hospital-related keywords (e.g., nursing and outpatient) enabled us to categorize the values into the "Hospital care" group. Table 7 shows each value from the training set, as well as their respective keywords and groups. As a result, we created a new variable labeled *merged_discharge_disposition* with only four possible values related to destination post-discharge (Table 8). Furthermore, we adjusted the validation and test sets based on the modifications made to the training.

Table 7: Values, Keywords Grasped, and Groups for *discharge_disposition*

Values (frequency)	Keywords	Group
Discharged to home (42256)	Home	Home discharge
Discharged/transferred to SNF (9780)	SNF	Hospital care
Discharged/transferred to home with home health service (9005)	Home	Home discharge
Discharged/transferred to another short-term hospital (1488)	Hospital	Hospital care
Discharged/transferred to another rehab fac including rehab units of a hospital (1393)	Rehab Hospital	Hospital care
Expired (1135)	Expired	Expired/hospice
Discharged/transferred to another type of inpatient care institution (822)	Inpatient Care Institution	Hospital care
Not Mapped (679)	-	Other
Discharged/transferred to ICF (571)	ICF	Hospital care
Left AMA (421)	-	Other
Discharged/transferred to a long-term care hospital (280)	Care Hospital	Hospital care
Hospice / medical facility (261)	Hospice	Expired/hospice
Hospice / home (258)	Hospice	Expired/hospice
Discharged/transferred/referred to a psychiatric hospital of psychiatric distinct part unit of a hospital(98)	Hospital	Hospital care
Discharged/transferred to home under care of Home IV provider (81)	Home	Expired/hospice

Discharged/transferred within this institution to Medicare approved swing bed (44)	Institution	Hospital care
Discharged/transferred to a nursing facility certified under Medicaid but not certified under Medicare (32)	Nursing Facility	Hospital care
Admitted as an inpatient to this hospital (13)	Inpatient Hospital	Hospital care
Discharged/transferred/referred to this institution for outpatient services (8)	Institution Outpatient Services	Hospital care
Discharged/transferred/referred another institution for outpatient services (7)	Institution Outpatient Services	Hospital care
Expired at home. Medicaid only, hospice (6)	Expired Hospice	Expired/hospice
Discharged/transferred to a federal health care facility (3)	Health Facility Care	Hospital care
Neonate discharged to another hospital for neonatal aftercare (2)	Hospital Care	Hospital care
Still patient or expected to return for outpatient services (2)	Outpatient Services	Hospital care
Expired in a medical facility. Medicaid only, hospice (1)	Expired Hospice	Expired/hospice

Source: Authors.

Table 8: Values and Frequencies of *merged_discharge_disposition*

Value	**Frequency**
Home discharge	35996
Hospital care	10133
Other	2542
Expired/hospice	1194

Source: Authors.

The rationale followed for *admission_source* was similar to *discharge_disposition*. Thus, we created a regrouped variable named merged_admission_source with four possible values stating how a patient ended up in the current encounter. However, categorizing the values was easier than in *merged_discharge_disposition* due to each string's lower range of possibilities and smaller size. Again, we performed the same adjustments to the validation and test sets after modifying the training. Like the previous variable, Table 9 shows the values, groups, and keywords from *admission_source*, while Table 10 comprises the values and frequencies for *merged_admission_source*.

It is worth noting that the low frequency of specific cases (e.g., "Court/Law Enforcement") made it hard to categorize them into a separate group. Consequently, we grouped them with the unknown sources (e.g., "Not Mapped"). Second, we distinguished the transfer and referral cases by considering the former as situations when patients refuse to follow the medical staff recommendations of who should better address their needs. In such situations, patients are fully responsible for defining the next steps. We have the latter (i.e., referral) when patients instead follow the medical advice (1).

Table 9: Values, Keywords Grasped, and Groups for *admission_source*

Values (frequency)	**Keywords**	**Group**
Emergency Room (40319)	Emergency	Emergency
Physician Referral (20678)	Referral	Referral
NaN (4718)	Unknown	Unknown/Other
Transfer from a hospital (2230)	Transfer	Transfer
Transfer from another health care facility (1562)	Transfer	Transfer
Clinic Referral (779)	Referral	Referral
Transfer from a Skilled Nursing Facility (595)	Transfer	Transfer
HMO Referral (129)	Referral	Referral
Not Mapped (107)	Unknown	Unknown/Other
Not Available (88)	Unknown	Unknown/Other
Court/Law Enforcement (11)	Other	Unknown/Other
Transfer from hospital inpt/same fac reslt in a sep claim (8)	Transfer	Transfer
Transfer from critical access hospital (7)	Transfer	Transfer
Transfer from Ambulatory Surgery Center (2)	Transfer	Transfer
Extramural Birth (1)	Other	Unknown/Other

Source: Authors.

Table 10: Values and Frequencies of *merged_admission_source*

Value	**Frequency**
Emergency	28324
Referral	15088
Unknown/Other	3390
Transfer	3063

Source: Authors.

The diabetes-related variables (*diabetes_severity* and *diabetes_control_type*) stem from a detailed evaluation of the ICD-9 codes (2) in the diagnosis-related variables (*primary_diagnosis*, *secondary_diagnosis*, and *additional_diagnosis*). The ICD-9 codification is a standard structure from the World Health Organization for comparing and presenting causes of disease internationally (3). The codes offered crucial insights about the

disease type, whether it was controlled, the presence or absence of additional complications, and the disease severity (Table 11).

The table below shows that, first, the column diabetes states whether the decimals in the diabetes codes indicate an additional complication due to the disease. Second, the even last number of the decimals denotes type II diabetes or unspecified, and the odd, type I. It is worth mentioning that some codes did not report the decimals, making us label them unspecified. Third, the last decimal number confirms that the disease is uncontrolled when ending with two or three. For the cases lacking the second decimal number, we labeled them unspecified. Although we separated the type and the absence of control for the disease in the table, we created variables combining both outcomes to avoid unnecessarily increasing the dataset dimensionality (we presented those variables in section E). Fourth, based on the ICD-9 website descriptions for the disease (2), we set the disease's severity based on whether the conditions reported represented an immediate life-threatening condition. For instance, those labeled as severe in the severity column relate to comatose outcomes requiring quick healthcare assistance. The ones labeled as moderate have serious complications that can be threatening in the medium term. Furthermore, we assigned the label unspecified for cases lacking the decimal numbers (i.e., only reporting the general code 250). The severity also enabled us to create a specific variable with this information (also available in section E). Lastly, we adjusted the validation and test sets after creating the two variables in the training set.

Table 11: Diabetes-related Information

Diabetes	**Type**	**Severity**	**Controlled**
General - 250	Unspecified	Unspecified	Unspecified
No mention to complication - 250.01	I	Mild	Not stated as uncontrolled
No mention to complication - 250.02	II or unspecified	Mild	Uncontrolled
No mention to complication - 250.03	I	Mild	Uncontrolled
With ketoacidosis - 250.1	Unspecified	Severe	Unspecified
With ketoacidosis - 250.11	I	Severe	Not stated as uncontrolled
With ketoacidosis - 250.12	II or unspecified	Severe	Uncontrolled
With ketoacidosis - 250.13	I	Severe	Uncontrolled
With hyperosmolarity - 250.2	Unspecified	Severe	Unspecified
With hyperosmolarity - 250.21	I	Severe	Not stated as uncontrolled
With hyperosmolarity - 250.22	II or unspecified	Severe	Uncontrolled
With hyperosmolarity - 250.23	I	Severe	Uncontrolled
With other coma - 250.3	Unspecified	Severe	Unspecified
With other coma - 250.31	I	Severe	Not stated as uncontrolled
With other coma - 250.32	II or unspecified	Severe	Uncontrolled
With other coma - 250.33	I	Severe	Uncontrolled
With renal manifestations - 250.4	Unspecified	Moderate	Unspecified
With renal manifestations - 250.41	I	Moderate	Not stated as uncontrolled
With renal manifestations - 250.42	II or unspecified	Moderate	Uncontrolled
With renal manifestations - 250.43	I	Moderate	Uncontrolled
With ophthalmic manifestations - 250.5	Unspecified	Moderate	Unspecified
With ophthalmic manifestations - 250.51	I	Moderate	Not stated as uncontrolled

With ophthalmic manifestations - 250.52	II or unspecified	Moderate	Uncontrolled
With ophthalmic manifestations - 250.53	I	Moderate	Uncontrolled
With neurological manifestations - 250.6	Unspecified	Moderate	Unspecified
With peripheral circulatory disorders - 250.7	Unspecified	Moderate	Unspecified
With other specified manifestations - 250.8	Unspecified	Moderate	Unspecified
With other specified manifestations - 250.81	I	Moderate	Not stated as uncontrolled
With other specified manifestations - 250.82	II or unspecified	Moderate	Uncontrolled
With other specified manifestations - 250.83	I	Moderate	Uncontrolled
With unspecified complication - 250.9	Unspecified	Moderate	Unspecified
With unspecified complication - 250.91	I	Moderate	Not stated as uncontrolled
With unspecified complication - 250.92	II or unspecified	Moderate	Uncontrolled
With unspecified complication - 250.93	I	Moderate	Uncontrolled

Source: Authors.

Finally, due to the wide range of possible values, we also used the ICD-9 codes to relabel the diagnosis-related variables for the diagnosis-related variables (*primary_diagnosis*, *secondary_diagnosis*, and *additional_diagnosis*). For instance, the large number of values for these variables is a challenge even in the original dataset (Ref.GCV4). Thus, we regrouped the values based on their broad category in the ICD-9 coding schema and stored the results in the variables *prim_diagnosis_cat*, *sec_diagnosis_cat*, and *add_diagnosis_cat* (these variables are also in section E). For example, the results we have for diabetes (ICD-9 code as 250) and thyroid gland disorders (ICD-9 code as 240) are within the same large ICD-9 group of diseases, which we labeled as "Metabolic disorders" (Table 12). We employed the same regrouping logic to the three variables in the training set and in the validation and test sets later.

Table 12: Regrouping Schema for the Diagnosis-related Variables

ICD-9 code range	**Group**
Letter “V” or “E” in the code	Supplementary conditions
0 to 139	Infectious disorders
140 to 239	Neoplasms
240 to 279	Metabolic disorders
280 to 289	Hematologic disorders
290 to 319	Mental disorders
320 to 389	Neurological disorders
390 to 459	Cardiovascular disorders
460 to 519	Respiratory disorders
520 to 579	Digestive disorders
580 to 629	Genitourinary disorders
630 to 679	Maternal complications

680 to 709	Dermatologic complications
710 to 739	Musculoskeletal disorders
740 to 759	Congenital anomalies
760 to 779	Perinatal conditions
780 to 799	Ill-defined conditions
800 to 899	Trauma conditions
Anything else	Other

Source: Authors.

References:

1. Jones-Nosacek C. Referral vs Transfer of Care: Ethical Options When Values Differ. Linacre Q [Internet]. 2022;89(1):36–46. Available from: https://doi.org/10.1177/00243639211055970
2. Centers for Disease Control and Prevention (CDC). International Classification of Diseases, Ninth Revision (ICD-9) [Internet]. 2022. Available from: https://www.cdc.gov/nchs/icd/icd9.htm
3. 2015 ICD-9-CM Diagnosis Codes [Internet]. 2015. Available from: http://www.icd9data.com/2015/Volume1/default.htm

Section D: Variables for Feature Selection

Table 13 below shows the initial variables we changed, kept, and dropped for feature selection, while Table 14 comprises the variables created. The decisions made result from the rigorous data exploration and preprocessing we employed in the dataset.

Table 13: Initial Variables for Feature Selection

Variable	**Context**	**Decision**
encounter_id	It only consists of unique values.	Set as index
patient_id	It does not contain valuable information as is.	Dropped
admission_type	For regrouping, see Tables 3 and 4. We then transformed the variable using frequency encoding.	Regrouped and normalized values in the newly created numerical variable
admission_source	For regrouping, see Tables 7 and 8. We then transformed the variable using frequency encoding.	Regrouped and normalized values in the newly created numerical variable
medical_specialty	With almost half of its values missing, it creates unnecessary noise in the dataset.	Dropped
discharge_disposition	For regrouping, see Tables 5 and 6. We then transformed the variable using frequency encoding.	Regrouped and normalized values in the newly created numerical variable
length_of_stay_in _hospital	Numerical variable	Kept as is

country	It is a constant variable	Dropped
race	The values have no ordinal relationship. To keep the nominal essence, we used one-hot encoding.	Transformed into new binary variables
gender	After handling unusual values, the only two possible values are “Female” and “Male”.	Kept but transformed into a binary variable
age	The observations are represented in bins. To keep the numerical essence, we assigned each interval its respective rounded-up median to guarantee a whole number.	Kept but transformed into a numerical variable
weight	With nearly all its values missing, imputation would bias the results.	Dropped
payer_code	The essential information is not the health insurance provider but whether health insurance was stated or not.	Transformed into a new binary variable
outpatient_visits_in _previous_year	Numerical variable	Kept as is
emergency_visits_in _previous_year	Numerical variable	Kept as is
inpatient_visits_in _previous_year	Numerical variable	Kept as is
average_pulse_bpm	Numerical variable	Kept as is
number_lab_tests	Numerical variable	Kept as is
non_lab_procedures	Numerical variable	Kept as is
number_of _medications	Numerical variable	Kept as is
primary_diagnosis	For regrouping, see Tables 10 and y. We then transformed the variable using frequency encoding.	Regrouped and normalized values in the newly created numerical variable
secondary_diagnosis	For regrouping, see Tables 10 and y. We then transformed the variable using frequency encoding.	Regrouped and normalized values in newly created numerical variable
additional_diagnosis	For regrouping, see Tables 10 and y. We then transformed the variable using frequency encoding.	Regrouped and normalized values in newly created numerical variable
number_diagnoses	Numerical variable	Kept as is
glucose_test_result	With the high amount of missing values, the crucial information is not the result but whether the test was taken.	Transformed into a new binary variable
a1c_test_result	With the high amount of missing values, the crucial information is not the result but whether the test was taken.	Transformed into a new binary variable

change_in_meds _during _hospitalization	The only two possible values are “Yes” and “No”.	Kept but transformed into a binary variable
prescribed_diabetes _meds	The only two possible values are “Yes” and “No”.	Kept but transformed into a binary variable
medication	The essential information is the number of medications prescriped for each patient during their encounter.	Transformed into a new numerical variable

Source: Authors.

Table 14: Newly Created Variables for Feature Selection

Variable	**Description**	**Logic**
multiple_encounters	Indicates whether a patient had multiple encounters within the current year. Values: “1” if *patient_id* appeared more than once and “0” if it only appeared once	Hypothesis that an increasing number of encounters within a given year may correlate with a higher likelihood of readmission
patient_status	Indicates whether a patient is alive. Values: “1” if *discharge_disposition* is not “Expired/hospice” and “0” if it is “Expired/hospice”	Serving as an indicator of patient’s current status, adding a penalty to cases where readmission might not be applicable
admission_type_freq	Indicates the frequency of each type of admission (Emergency, Elective, Unknown/Other). Values: frequency-encoded *admission_type*	Keeping the level of prevalence between the categories and their uniqueness as well as avoiding the creation of new variables as much as possible
admission_source _freq	Indicates the frequency of each source of admission (Emergency, Referral, Unknown/Other, Transfer). Values: frequency-encoded *admission_source*	Keeping the level of prevalence between the categories and their uniqueness as well as avoiding the creation of new variables as much as possible
discharge_disp_freq	Indicates the frequency of each kind of disposition at discharge (Home discharge, Hospital care, Other, Expired/hospice). Values: frequency-encoded *discharge_disposition*	Keeping the level of prevalence between the categories and their uniqueness as well as avoiding the creation of new variables as much as possible
race_asian	Indicates whether a patient stated their race as asian. Values: “1” if the value of *race* is “Asian” and “0” if it has another value	Transformation through one-hot encoding to maintain the variables nominal essence whereas approaches like label, ordinal or frequency encoding may introduce a non-existent ordinal relationship between the categories

race_caucasian	Indicates whether a patient stated their race as caucasian. Values: "1" if the value of *race* is "Caucasian" and "0" if it has another value	Transformation through one-hot encoding to maintain the variables nominal essence whereas approaches like label, ordinal or frequency encoding may introduce a non-existent ordinal relationship between the categories
race_hispanic	Indicates whether a patient stated their race as hispanic. Values: "1" if the value of *race* is "Hispanic" and "0" if it has another value	Transformation through one-hot encoding to maintain the variables nominal essence whereas approaches like label, ordinal or frequency encoding may introduce a non-existent ordinal relationship between the categories
race_other	Indicates whether a patient stated their race as something else than african american, asian, caucasian or hispanic. Values: "1" if the value of *race* is "Other" and "0" if it has another value	Transformation through one-hot encoding to maintain the variables nominal essence whereas approaches like label, ordinal or frequency encoding may introduce a non-existent ordinal relationship between the categories
race_unknown	Indicates whether a patient did not state their race. Values: "1" if the value of *race* is "Unknown" and "0" if it has another value	Transformation through one-hot encoding to maintain the variables nominal essence whereas approaches like label, ordinal or frequency encoding may introduce a non-existent ordinal relationship between the categories
health_insurance	Indicates whether a patient has health insurance. Values: "1" if *payer_code* has a value and "0" if it has a missing value	Hypothesis that the fact that a patient has health insurance may correlate with the likelihood of readmission
total_visits_in _previous_year	Total number of visits the patient made to the hospital in the year preceding the encounter. Values: counts of *outpatient_visits_in_previous_year* and *emergency_visits_in_previous_year*	Hypothesis that the combined number of outpatient and emergency visits may correlate with a higher likelihood of readmission
total_number_of _procedures	Total number of tests and procedures performed during the patient's encounter. Values: counts of *number_lab_tests* and *non_lab_procedures*	Hypothesis that the combined number of lab tests and non-lab procedures may correlate with a higher likelihood of readmission
primary_diagnosis _freq	Indicates the frequency of each primary diagnosis (Cardiovascular disorders, Trauma conditions, Respiratory disorders, Digestive disorders, Ill-defined disorders, Genitourinary disorders, Neoplasms, Metabolic disorders,	Keeping the level of prevalence between the categories and their uniqueness as well as avoiding the creation of new variables as much as possible

	Dermatologic disorders, Mental disorders, Supplementary conditions, Neurological disorders, Hematological disorders, Maternal complications, Congenital anomalies, Other). Values: frequency-encoded *primary_diagnosis*	
seconday_diagnosis_freq	Indicates the frequency of each secondary diagnosis (Cardiovascular disorders, Trauma conditions, Respiratory disorders, Digestive disorders, Ill-defined disorders, Genitourinary disorders, Neoplasms, Metabolic disorders, Dermatologic disorders, Mental disorders, Supplementary conditions, Neurological disorders, Hematological disorders, Maternal complications, Congenital anomalies, Other). Values: frequency-encoded *secondary_diagnosis*	Keeping the level of prevalence between the categories and their uniqueness as well as avoiding the creation of new variables as much as possible
additional_diagnosis_freq	Indicates the frequency of each additional diagnosis (Cardiovascular disorders, Trauma conditions, Respiratory disorders, Digestive disorders, Ill-defined disorders, Genitourinary disorders, Neoplasms, Metabolic disorders, Dermatologic disorders, Mental disorders, Supplementary conditions, Neurological disorders, Hematological disorders, Maternal complications, Congenital anomalies, Other). Values: frequency-encoded *additional_diagnosis*	Keeping the level of prevalence between the categories and their uniqueness as well as avoiding the creation of new variables as much as possible
daily_diagnosis	Average number of diagnoses per day during the patient's encounter. Values: rates of *number_diagnoses* per *length_of_stay_in_hospital*	Hypothesis that the number of daily diagnoses may correlate with the likelihood of readmission
took_glucose_test	Indicates whether a patient took a glucose test. Values: "1" if *glucose_test* has a value and "0" if it has a missing value	Hypothesis that the fact that a patient took a glucose test may correlate with the likelihood of readmission
took_a1c_test	Indicates whether a patient took a A1C test.	Hypothesis that the fact that a patient took a A1C test may correlate with the likelihood of readmission

	Values: "1" if *glucose_test* has a value and "0" if it has a missing value	
medication_total	Total number of medications prescribed to the patient during their encounter. Values: counts of the occurrences of each medication in *medication*	Hypothesis that the number of prescribed medications may correlate with the likelihood of readmission
daily_medications	Average number of medications per day administered during the patient's encounter. Values: rate of *number_of_medication* per *length_of_stay_in_hospital*	Hypothesis that the number of daily administered medications may correlate with the likelihood of readmission
diabetes_uncontrolled_typei	Indicates whether a patient's type of diabetes is uncontrolled type 1. Values: "1" if if any of the *primary_*, *secondary_*, or *additional_diagnosIs* values match the predefined control type codes (250.03, 250.13, 250.23, 250.33, 250.43, 250.53, 250.63, 250.73, 250.83, 250.93) and "0" if it has another value	Transformation through one-hot encoding to maintain the variables nominal essence whereas approaches like label, ordinal or frequency encoding may introduce a non-existent ordinal relationship between the categories
diabetes_uncontrolled_typeii	Indicates whether a patient's type of diabetes is uncontrolled type 2. Values: "1" if if any of the *primary_*, *secondary_*, or *additional_diagnosIs* values match the predefined control type codes (250.02, 250.12, 250.22, 250.32, 250.42, 250.52, 250.62, 250.72, 250.82, 250.92) and "0" if it has another value	Transformation through one-hot encoding to maintain the variables nominal essence whereas approaches like label, ordinal or frequency encoding may introduce a non-existent ordinal relationship between the categories
diabetes_unspecified	Indicates whether a patient's type of diabetes is unspecified. Values: "1" if if any of the *primary_*, *secondary_*, or *additional_diagnosIs* values match the predefined control type codes (250, 250.1, 250.2, 250.3, 250.4, 250.5, 250.6, 250.7, 250.8, 250.9) and "0" if it has another value	Transformation through one-hot encoding to maintain the variables nominal essence whereas approaches like label, ordinal or frequency encoding may introduce a non-existent ordinal relationship between the categories
diabetes_unstated_typei	Indicates whether a patient's type of diabetes is unstated type 1. Values: "1" if if any of the *primary_*, *secondary_*, or *additional_diagnosIs* values match the predefined control type codes (250.01, 250.11, 250.21, 250.31, 250.41, 250.51, 250.61, 250.71,	Transformation through one-hot encoding to maintain the variables nominal essence whereas approaches like label, ordinal or frequency encoding may introduce a non-existent ordinal relationship between the categories

	250.81, 250.91) and "0" if it has another value	
diabetes_severity _mild	Indicates whether a patient's level of severity in diabetes is mild. Values: "1" if if any of the *primary_*, *secondary_*, or *additional_diagnosIs* values match the predefined severity codes (250.00, 250.01, 250.02, 250.03) and "0" if it has another value	Transformation through one-hot encoding to maintain the variables nominal essence whereas approaches like label, ordinal or frequency encoding may introduce a non-existent ordinal relationship between the categories
diabetes_severity _moderate	Indicates whether a patient's level of severity in diabetes is moderate. Values: "1" if if any of the *primary_*, *secondary_*, or *additional_diagnosIs* values match the predefined severity codes (250.4, 250.40, 250.41, 250.42, 250.43, 250.5, 250.50, 250.51, 250.52, 250.53, 250.6, 250.60, 250.61, 250.62, 250.63, 250.7, 250.70, 250.71, 250.72, 250.73, 250.8, 250.80, 250.81, 250.82, 250.83, 250.4, 250.90, 250.91, 250.92, 250.93) and "0" if it has another value	Transformation through one-hot encoding to maintain the variables nominal essence whereas approaches like label, ordinal or frequency encoding may introduce a non-existent ordinal relationship between the categories
diabetes_severity _severe	Indicates whether a patient's level of severity in diabetes is severe. Values: "1" if if any of the *primary_*, *secondary_*, or *additional_diagnosIs* values match the predefined severity codes (250.1, 250.10, 250.11, 250.12, 250.13, 250.2, 250.20, 250.21, 250.22, 250.23, 250.3, 250.30, 250.31, 250.32, 250.33) and "0" if it has another value	Transformation through one-hot encoding to maintain the variables nominal essence whereas approaches like label, ordinal or frequency encoding may introduce a non-existent ordinal relationship between the categories
diabetes_severity _unspecified	Indicates whether a patient's level of severity in diabetes is unspecified. Values: "1" if if any of the *primary_*, *secondary_*, or *additional_diagnosIs* values match the predefined severity code (250) and "0" if it has another value	Transformation through one-hot encoding to maintain the variables nominal essence whereas approaches like label, ordinal or frequency encoding may introduce a non-existent ordinal relationship between the categories

Source: Authors.

Section E: Synthetic Minority Over-sampling Technique (SMOTE)

Rather than the traditional under- or over-sampling with replacement, SMOTE proposes the creation of synthetic data based on the observed samples from the minority class. The execution of this technique requires the (usually random) selection of a point from the minority class, as well as the definition of the number of desired neighbors for that point. Once these steps are performed, one of these neighbors is randomly chosen, and the synthetic point is generated by linear interpolation of the two selected points (1).

On top of its technique, a combination between SMOTE and SVMs was also developed. By applying SVM and computing the support vectors that define the boundary for both classes, this technique aims to prevent the generation of samples that fall in the majority class region - the main drawback of the SMOTE method -, potentially leading to misclassification problems (2).

References:

1. Chawla NV., Bowyer KW, Hall LO, Kegelmeyer WP. SMOTE: Synthetic Minority Over-sampling Technique. J Artif Intell Res [Internet]. 2002;30(2):321–57. Available from: https://doi.org/10.1613/jair.953

2. Nguyen HM, Cooper EW, Kamei K. Borderline Over-sampling for Imbalanced Data Classification. In: Fifth International Workshop on Computational Intelligence & Applications [Internet]. 2009. p. 24–9. Available from: https://doi.org/10.1504/IJKESDP.2011.039875

Section F: Analysis of Variance (ANOVA)

The Analysis of Variance, most commonly referred to as ANOVA, finds its purpose in the testing of whether a certain independent factor, when applied differently to several populations, has a significant effect over the outcome of the dependent variable. In this test, the null hypothesis is that all means are equal ($\mu_1 = \ldots = \mu_k$), against the alternative hypothesis where at least one pair of means differs statistically (1).

In the context of feature selection, the process is as follows: for each predictor, the observations are first grouped by their value for the explained variable; then, the variability within and between groups is calculated, allowing performing the F-test and determining the subsequent p-value. In the end, the aim is to retain the variables that allow for the rejection of the null hypothesis, as this implies that there are statistically significant differences in the values of that variable when grouped by the target. This can either be done with respect to the desired level of significance (usually equal to 0.05), where the selected variables are the ones whose p-value falls below that threshold - as done by Arowolo et al. (2)- or by pre-defining a k-number of desired variables, and keeping them by finding the k-lowest p-values. In summary, ANOVA picks the most crucial variables for the addressed problem based on their impact on the target variable.

References:

1. Reis E, Melo P, Andrade R, Calapez T. Ensaios de hipóteses. In: Estatística Aplicada: Volume 2. 6th Editio. Sílabo; 2018. p. 139–225.

2. Arowolo MO, Abdulsalam SO, Saheed YK, Salawu MD. A Feature Selection Based on One Way Anova For Microarray Data Classification. Al-Hikmah J Pure Appl Sci. 2016;30–5.

Section G: Feature Selection For Binary Classification

As mentioned in the report's body, we have displayed in Table 15 the final feature selection for the binary problem, noting every decision to keep or discard each variable in the corresponding method.

Table 15: Binary's Feature Selection

Feature	Pearson	Spearman	ANOVA	DT	RandomForest	Log.Reg.	SVCLin.	LASSO	Result
gender	Keep	Keep	Discard	Discard	Discard	Discard	Discard	Discard	Discard
change_in_meds_during_hospitalization	Keep	Keep	Discard	Discard	Discard	Discard	Discard	Discard	Discard
prescribed_diabetes_meds	Keep	Keep	Keep	Discard	Discard	Discard	Discard	Discard	Discard
health_insurance	Keep	Keep	Keep	Discard	Discard	Discard	Discard	Discard	Discard
took_glucose_test	Keep	Keep	Discard	Discard	Discard	Discard	Discard	Discard	Discard
took_a1c_test	Keep	Keep	Keep	Discard	Discard	Keep	Discard	Discard	Keep
race_asian	Keep	Keep	Discard	Discard	Discard	Keep	Discard	Discard	Discard
race_caucasian	Keep	Keep	Discard	Discard	Discard	Discard	Discard	Discard	Discard
race_hispanic	Keep	Keep	Keep	Discard	Discard	Keep	Discard	Discard	Keep
race_other	Keep	Keep	Keep	Discard	Discard	Discard	Discard	Discard	Discard
race_unknown	Keep	Keep	Keep	Discard	Discard	Discard	Discard	Discard	Discard
admission_type_freq	Keep	Keep	Keep	Discard	Discard	Discard	Discard	Discard	Discard
discharge_disp_freq	Keep	Keep	Keep	Keep	Keep	Keep	Discard	Discard	Keep
admission_source_freq	Keep	Keep	Keep	Discard	Discard	Discard	Discard	Discard	Discard
primary_diagnosis_freq	Keep	Keep	Discard	Discard	Discard	Discard	Discard	Discard	Discard
secondary_diagnosis_freq	Keep	Keep	Discard	Discard	Discard	Discard	Discard	Discard	Discard
additional_diagnosis_freq	Keep	Keep	Keep	Discard	Discard	Discard	Discard	Discard	Discard
diabetes_uncontrolled_typei	Keep	Keep	Discard	Discard	Discard	Discard	Discard	Discard	Discard
diabetes_uncontrolled_typeii	Keep	Keep	Discard	Discard	Discard	Discard	Discard	Discard	Discard
diabetes_unspecified	Keep	Keep	Discard	Discard	Discard	Discard	Discard	Discard	Discard
diabetes_unstated_typei	Keep	Keep	Discard	Discard	Discard	Discard	Discard	Discard	Discard
diabetes_severity_mild	Keep	Keep	Keep	Discard	Discard	Discard	Discard	Discard	Discard
diabetes_severity_moderate	Keep	Keep	Keep	Discard	Discard	Keep	Discard	Keep	Keep
diabetes_severity_severe	Keep	Keep	Discard	Discard	Discard	Discard	Discard	Discard	Discard
diabetes_severity_unspecified	Discard	Discard	Keep	Discard	Discard	Discard	Discard	Discard	Discard
multiple_encounters	Keep	Keep	Keep	Keep	Keep	Keep	Keep	Keep	Keep
patient_status	Keep	Keep	Keep	Discard	Discard	Keep	Keep	Keep	Keep
age	Keep	Keep	Keep	Keep	Keep	Keep	Discard	Keep	Keep
outpatient_visits_in_previous_year	Keep	Keep	Discard	Discard	Discard	Discard	Discard	Discard	Discard
emergency_visits_in_previous_year	Keep	Keep	Keep	Discard	Discard	Discard	Discard	Discard	Discard
inpatient_visits_in_previous_year	Keep	Keep	Keep	Keep	Keep	Keep	Keep	Keep	Keep
average_pulse_bpm	Keep	Keep	Discard	Discard	Keep	Discard	Discard	Discard	Discard
length_of_stay_in_hospital	Keep	Keep	Keep	Discard	Keep	Discard	Discard	Discard	Keep
number_lab_tests	Keep	Keep	Discard	Discard	Discard	Keep	Discard	Discard	Discard
non_lab_procedures	Keep	Keep	Keep	Keep	Keep	Discard	Discard	Discard	Keep
number_of_medications	Keep	Keep	Discard	Discard	Discard	Keep	Discard	Discard	Discard
number_diagnoses	Keep	Keep	Keep	Keep	Keep	Discard	Discard	Discard	Keep
medication_total	Discard	Discard	Discard	Discard	Discard	Discard	Discard	Discard	Discard
total_visits_in_previous_year	Discard	Discard	Keep	Keep	Keep	Discard	Discard	Discard	Discard
daily_medications	Discard	Keep	Keep	Discard	Discard	Discard	Discard	Discard	Discard
daily_diagnosis	Discard	Discard	Keep	Discard	Keep	Keep	Discard	Discard	Discard
total_number_of_procedures	Discard	Discard	Discard	Discard	Discard	Discard	Discard	Discard	Discard

Source: Authors.

Section H: Metrics Calculation for Binary Classification (all models)

We display below the table that summarizes our results before introducing the ensemble methods' outcomes on the same metrics. It is worth reinforcing that all decimal cases were included in the table to highlight the minor differences in some models' scores.

Table 16: Metrics Calculation for Binary Classification (validation set results sorted by f1 score)

Model	Accuracy	ROC-AUC	Precision	Recall	F1 Score	Kaggle Score
Bagging	0.772449	0.625199	0.228051	0.435639	0.299380	0.3171
Voting	0.784708	0.621833	0.235055	0.412159	0.299376	0.3205
Stacking	0.773197	0.623604	0.227534	0.431027	0.297842	0.3189
Neural networks	0.770296	0.624354	0.226010	0.436478	0.297811	0.3191
Bernoulli naive Bayes	0.777034	0.621730	0.229053	0.421803	0.296887	0.3135
Linear SVC	0.782088	0.613576	0.227185	0.396646	0.288899	0.3131
SVM (polynomial kernel)	0.795938	0.610190	0.236252	0.371069	0.288697	0.3153
Stochastic gradient descent	0.786861	0.612230	0.229965	0.387421	0.288615	0.3172
Logistic regression	0.785925	0.612253	0.229228	0.388679	0.288381	0.3152
Adaboost	0.786907	0.611523	0.229484	0.385744	0.287770	0.3130
Gaussian naive Bayes	0.779748	0.611893	0.224228	0.395807	0.286277	0.3161
Random forest	0.785691	0.610471	0.227735	0.384906	0.286160	0.3075
Gradient boosting	0.790136	0.608757	0.230077	0.375262	0.285259	0.3129
Decision tree	0.780310	0.599927	0.215797	0.367715	0.271980	0.2834

Source: Authors.

Section I: Stochastic Gradient Descent Classifier

Gradient descent is an optimization algorithm employed in the minimization of a predetermined loss function, which iteratively updates its parameters at a speed defined by the learning rate (1). The stochastic gradient descent, in its turn, computes the gradient based on a single subset of the data for each iteration - these are unbiased estimates of the real gradient (2).

The stochastic gradient descent classifier applies the reasoning behind SGD to make predictions in a classification task, often having another equivalent classifier depending on the loss function used. For instance, if a logistic loss function is deemed as a parameter, a logistic regression will be executed, whilst, if the hinge function is utilized, a linear support vector machine will be computed. The "modified_huber" function, employed in this project, allows for more robustness against outliers and heavy-tailed sampling distributions (3).

References:

1. Wang X, Yan L, Zhang Q. Research on the Application of Gradient Descent Algorithm in Machine Learning. In: 2021 International Conference on Computer Network, Electronic and Automation (ICCNEA). 2021. p. 11–5.

2. Tian Y, Zhang Y, Zhang H. Recent Advances in Stochastic Gradient Descent in Deep Learning. Mathematics. 2023;11(3):1–23.

3. Tong H. Nonasymptotic analysis of robust regression with modified Huber's loss. J Complex [Internet]. 2023;76:101744. Available from: https://doi.org/10.1016/j.jco.2023.101744

Section J: Voting Classifier

This classification algorithm belongs in the "stacking" subcategory of ensemble methods, meaning that it enables the combination of various different models before making a final prediction - this is in contrast to Bagging and Boosting algorithms, whose main characteristics reside in the use of a single algorithm (either combined in parallel or built sequentially, respectively) to predict a certain outcome (1). The main difference of voting, when compared to stacking's Sci-Kit learn implementation, lies in the utilization, by stacking, of an estimator (by default, a logistic regression) to generate a prediction, whilst voting decides the output based on majority voting.

Moreover, the voting classifier allows for the assignment of weights to each estimator. One way of defining such weights is by the use of classification metrics, such as the f1 score or accuracy - as employed by Osamor and Okezie (2) - which allocate a larger weight (and, therefore, more power in the decision of the final prediction) to the models that achieved a higher score in these metrics. This reasoning was also applied in this project, utilizing the f1 score to define the influence of each estimator. First, the average f1 score was calculated. Then, a reward/penalization was attributed to each one of the models, by subtracting its specific f1 score from the average (if it is above average, then this value will be positive; otherwise, it will be negative). Second, the final weight was computed by adding this reward/penalization to 1, the default vote weight for each model.

References:

1. Swamynathan M. Step 4 – Model Diagnosis and Tuning. In: Mastering Machine Learning with Python in Six Steps. 2nd Edition. Apress Media; 2017. p. 209–50.

2. Osamor VC, Okezie AF. Enhancing the weighted voting ensemble algorithm for tuberculosis predictive diagnosis. Sci Rep [Internet]. 2021;11(1):1–11. Available from: https://doi.org/10.1038/s41598-021-94347-6

Section K: Metrics Calculation for Binary Classification (single models)

Table 17: Metrics Calculation for Binary Classification (validation set results sorted by f1 score)

Model	Precision	Recall	F1 Score	Kaggle Score
Neural networks	0.226010	0.436478	0.297811	0.3191
Bernoulli naive Bayes	0.229053	0.421803	0.296887	0.3135
Linear SVC	0.227185	0.396646	0.288899	0.3131
SVM (polynomial kernel)	0.236252	0.371069	0.288697	0.3153
Stochastic gradient descent	0.229965	0.387421	0.288615	0.3172
Logistic regression	0.229228	0.388679	0.288381	0.3152
Gaussian naive Bayes	0.224228	0.395807	0.286277	0.3161
Decision tree	0.215797	0.367715	0.271980	0.2834

Source: Authors.

Section L: Feature Selection for Multiclass Classification

As mentioned in the report's body, we have displayed in Table 18 the final feature selection for the multiclass problem, noting every decision to keep or discard each variable in the corresponding method.

Table 18: Multiclass' Feature Selection

Features	Pearson	Spearman	ANOVA	DT	RandomForest	Log.Reg.	SVCLin.	LASSO	Result
gender	Keep	Keep	Discard	Discard	Discard	Keep	Keep	Discard	Discard
change_in_meds_during_hospitalization	Discard	Discard	Discard	Discard	Discard	Keep	Keep	Discard	Discard
prescribed_diabetes_meds	Keep	Keep	Keep	Discard	Discard	Keep	Keep	Discard	Keep
health_insurance	Keep	Keep	Discard	Discard	Discard	Keep	Keep	Keep	Keep
took_glucose_test	Keep	Keep	Discard	Discard	Discard	Keep	Keep	Discard	Discard
took_a1c_test	Keep	Keep	Keep	Discard	Discard	Keep	Keep	Discard	Keep
race_asian	Keep	Keep	Discard	Discard	Discard	Keep	Keep	Keep	Keep
race_caucasian	Keep	Keep	Discard	Discard	Discard	Keep	Keep	Discard	Discard
race_hispanic	Keep	Keep	Discard	Discard	Discard	Keep	Keep	Discard	Discard
race_other	Keep	Keep	Keep	Discard	Discard	Keep	Keep	Discard	Keep
race_unknown	Keep	Keep	Discard	Discard	Discard	Keep	Discard	Discard	Discard
diabetes_uncontrolled_typei	Keep	Keep	Discard	Discard	Discard	Keep	Keep	Discard	Discard
diabetes_uncontrolled_typeii	Keep	Keep	Discard	Discard	Discard	Keep	Keep	Discard	Discard
diabetes_unspecified	Keep	Keep	Discard	Discard	Discard	Keep	Keep	Discard	Discard
diabetes_unstated_typei	Keep	Keep	Discard	Discard	Discard	Keep	Keep	Discard	Discard
diabetes_severity_mild	Keep	Keep	Discard	Discard	Discard	Keep	Keep	Discard	Discard
diabetes_severity_moderate	Keep	Keep	Keep	Discard	Discard	Keep	Keep	Discard	Keep
diabetes_severity_severe	Keep	Keep	Discard	Discard	Discard	Keep	Keep	Discard	Discard
diabetes_severity_unspecified	Discard	Discard	Keep	Discard	Discard	Keep	Keep	Discard	Discard
multiple_encounters	Keep	Keep	Keep	Keep	Keep	Keep	Keep	Keep	Keep
patient_status	Keep	Keep	Keep	Discard	Discard	Keep	Keep	Keep	Keep
age	Keep	Keep	Keep	Keep	Keep	Keep	Keep	Keep	Keep
outpatient_visits_in_previous_year	Keep	Keep	Keep	Discard	Discard	Keep	Keep	Keep	Keep
emergency_visits_in_previous_year	Keep	Keep	Keep	Discard	Discard	Keep	Keep	Discard	Keep
inpatient_visits_in_previous_year	Discard	Discard	Keep	Keep	Keep	Keep	Keep	Keep	Keep
average_pulse_bpm	Keep	Keep	Discard	Keep	Keep	Keep	Keep	Discard	Keep
length_of_stay_in_hospital	Keep	Keep	Keep	Discard	Keep	Discard	Keep	Discard	Keep
number_lab_tests	Keep	Keep	Keep	Keep	Keep	Keep	Keep	Keep	Keep
non_lab_procedures	Keep	Keep	Keep	Discard	Discard	Keep	Keep	Discard	Keep
number_of_medications	Keep	Keep	Keep	Keep	Keep	Keep	Keep	Keep	Keep
number_diagnoses	Keep	Keep	Keep	Discard	Discard	Keep	Keep	Keep	Keep
admission_type_freq	Keep	Keep	Discard	Discard	Discard	Keep	Keep	Discard	Discard
discharge_disp_freq	Keep	Keep	Keep	Discard	Discard	Keep	Keep	Keep	Keep
admission_source_freq	Keep	Keep	Keep	Discard	Discard	Keep	Keep	Keep	Keep
medication_total	Discard	Discard	Discard	Discard	Discard	Keep	Keep	Keep	Discard
primary_diagnosis_freq	Keep	Keep	Discard	Discard	Discard	Keep	Keep	Discard	Discard
secondary_diagnosis_freq	Keep	Keep	Discard	Keep	Discard	Keep	Keep	Discard	Keep
additional_diagnosis_freq	Keep	Keep	Keep	Keep	Keep	Keep	Keep	Discard	Keep
total_visits_in_previous_year	Keep	Keep	Keep	Keep	Keep	Keep	Keep	Keep	Keep
daily_medications	Discard	Discard	Discard	Keep	Keep	Keep	Keep	Discard	Discard
daily_diagnosis	Discard	Discard	Keep	Discard	Keep	Keep	Keep	Keep	Keep
total_number_of_procedures	Discard	Discard	Discard	Keep	Keep	Discard	Discard	Discard	Discard

Source: Authors.

Section M: Multinomial and complement naive Bayes

As previously mentioned, in the multiclass stage, we decided to expand our possibilities and try different naive Bayes models, given their notorious performance and simplicity when it comes to making predictions (1).

One of the implemented models was multinomial naive Bayes, a method used in different multiclass classification problems, such as the one described by Resti and colleagues on corn plant diseases and pests (2). Despite its use in different multiclass contexts, this model is particularly suitable for classifications with discrete features, namely on text classifications (3), revealing increased performance compared to traditional methods and

improvement over Bernoulli naive Bayes models (4). To make predictions, the model assumes the variables' independence given the class label and estimates the probabilities from a training set using a multinomial distribution with additive smoothing, avoiding zero probabilities for unseen words. Having the prior probabilities for the class labels, the model applies Bayes' theorem to compute posterior probabilities for each class label of new data, where the final prediction will be the class label with the higher posterior probability.

Having this model tested, we also employed complement naive Bayes, specifically designed to improve multinomial naive Bayes' performance for imbalanced datasets. This method provides more stable parameter estimates, which is helpful for datasets with skewed distributions. Furthermore, it uses statistics from each class's complement to compute the model weights, thus avoiding favoring the majority class and includes a normalization step to reduce the dominance of longer documents in the parameter estimates (3). With all this, complement naive Bayes was revealed to return even higher performance than multinomial naive Bayes in text classification (5).

Considering that both models are commonly used for text classification purposes, it was no surprise that they revealed lower f1 scores in our multiclass classification compared to the other models.

References:

1. Chen H, Hu S, Hua R, Zhao X. Improved naive Bayes classification algorithm for traffic risk management. EURASIP J Adv Signal Process. 2021;2021(1).
2. Resti Y, Irsan C, Neardiaty A, Annabila C, Yani I. Fuzzy Discretization on the Multinomial Naïve Bayes Method for Modeling Multiclass Classification of Corn Plant Diseases and Pests. Mathematics. 2023;11(8).
3. scikit-learn. 1.9. Naive Bayes [Internet]. Available from: https://scikit-learn.org/stable/modules/naive_bayes.html#multinomial-naive-bayes
4. Abbas M, Ali Memon K, Aleem Jamali A, Memon S, Ahmed A. Multinomial Naive Bayes Classification Model for Sentiment Analysis. IJCSNS Int J Comput Sci Netw Secur. 2019;19(3):62–7.
5. Seref B, Bostanci E. Performance Comparison of Naïve Bayes and Complement Naïve Bayes Algorithms. In: 6th International Conference on Electrical and Electronics Engineering (ICEEE). 2019. p. 131–8.

Section N: Randomized search CV

Much like grid search CV, randomized search CV is a hyperparameter optimization technique implemented to find the best hyperparameter combination for an ML model that maximizes a selected metric, like accuracy and f1 score (1). This technique comes as a variant of the first one, in the sense that instead of performing an exhaustive search over the parameter space, it samples a fixed number of parameter settings to test (1). Therefore, randomized search is more computational and time efficient than grid search, namely when it comes to high-dimensional data or model parameters to be tested (2). Despite that, this approach can fall short compared to grid searches since, if not given sufficient trials to explore the dataset and all parameters, obtained results might not be very accurate (2).

In our case, this technique seemed to be the best one to implement, given the computational constraints already mentioned. To compensate for not making an exhaustive search on the models' hyperparameters, we decided to increase the randomized search's parameters: n_iter, responsible for defining the number of parameter settings to be sampled (1), from the default "10" to "30"; and the parameter CV, which determined the cross-validation splitting criteria (1), from "5" to "10", thus increasing this technique' search space and improving our probabilities of getting closer to the best solution. Furthermore, we used Scipy's randint with the rvs method. The first is commonly used to generate discrete random variables with uniform distribution (3), and the second generates a random sample with the same probability distribution as the object it is used on (4). Their

combination becomes advantageous when using randomized search CV because most parameters, like the decision tree's maximum depth or the neural networks' hidden layer size, can only take integer values. Consequently, Scipy's randint becomes a natural choice, while the rvs generates more samples for the specified object, allowing the randomized search to explore a more diverse set of hyperparameter settings.

Unfortunately, because some models had more hyperparameters or were more computationally heavy, namely for the SVM, adaboost, bagging, and gradient boosting, the values for these parameters were left as the default ones.

It is worth mentioning that the randomized search CV was applied to all our models, except for the multinomial and complement naive Bayes, given their significantly lower performance compared to the other models. As such, we decided to focus on improving those that immediately showed potential, that is, the ones with an f1 score above 0.50 when run with only the default parameter values.

References:

1. scikit-learn. sklearn.model_selection.RandomizedSearchCV [Internet]. Available from: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html
2. Bergstra J, Bengio Y. Random search for hyper-parameter optimization. J Mach Learn Res. 2012;13:281–305.
3. Scipy. scipy.stats.randint [Internet]. Available from: https://docs.scipy.org/doc/scipy-0.11.0/reference/generated/scipy.stats.randint.html
4. Scipy. scipy.stats.rv_continuous [Internet]. Available from: https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.rv_continuous.html

Section O: Metrics Calculation for Multiclass Classification (single models)

We display below the table that summarizes our results before introducing the ensemble methods' outcomes on the same metrics. It is worth reinforcing that all decimal cases were included in the table to highlight the minor differences in some models' scores.

Table 19: Metrics Calculation for Multiclass Classification (validation set results sorted by f1 score)

Model	Accuracy	Precision	Recall	F1 Score
Decision tree	0.597492	0.565042	0.597492	0.571757
Neural networks	0.588648	0.569965	0.588648	0.566490
Logistic regression	0.569791	0.575822	0.569791	0.536025
Gaussian naive Bayes	0.515699	0.553799	0.515699	0.528996
Linear SVC	0.572130	0.583385	0.572130	0.523747
Bernoulli naive Bayes	0.508586	0.546284	0.508586	0.511442
Multinomial naive Bayes	0.546207	0.582282	0.546207	0.472359
Complement naive Bayes	0.536054	0.581169	0.536054	0.444275

Source: Authors.

Section P: Difference Between Macro and Weighted Metrics

In multiclass classification problems, some adaptations must be made to compute evaluation metrics such as the f1 score, precision, and recall. In particular, the average parameter has to be adjusted to determine how the scores for each class are calculated (1).

The weighted average method derives scores by calculating a weighted average for individual classes, addressing the class imbalance concern that the macro average fails to manage (1,2).

The macro average method computes scores by averaging independently for each class without considering their sizes (1), which makes it less reliable than the weighted average in the presence of imbalanced data (2).

Below, in Table 20, we displayed the metrics for all models but calculated using a macro average, emphasizing the difference between the results, in particular, the fact that they tend to be lower using a macro approach.

Table 20: Macro Metrics for Multiclass Classification (validation set results sorted by f1 score)

Model	Accuracy	Precision	Recall	F1 Score
Bagging	0.590239	0.486196	0.479224	0.476616
Stacking	0.600955	0.489413	0.465591	0.467343
Voting	0.614244	0.499706	0.462755	0.464459
Neural networks	0.588648	0.475741	0.458193	0.456321
Random forest	0.605961	0.484377	0.453643	0.452522
Decision tree	0.597492	0.488134	0.449800	0.450921
Gaussian naive Bayes	0.515699	0.449781	0.469454	0.446959
Complement naive Bayes	0.536054	0.490137	0.441001	0.444275
Logistic regression	0.569791	0.485623	0.469644	0.442425
Gradient boosting	0.613916	0.502028	0.442216	0.435031
Linear SVC	0.572130	0.495905	0.463160	0.429661
Bernoulli naive Bayes	0.508586	0.437955	0.447403	0.417803
Adaboost	0.611670	0.487120	0.430649	0.415578
Multinomial naive Bayes	0.546207	0.491129	0.444708	0.379584

Source: Authors.

References:

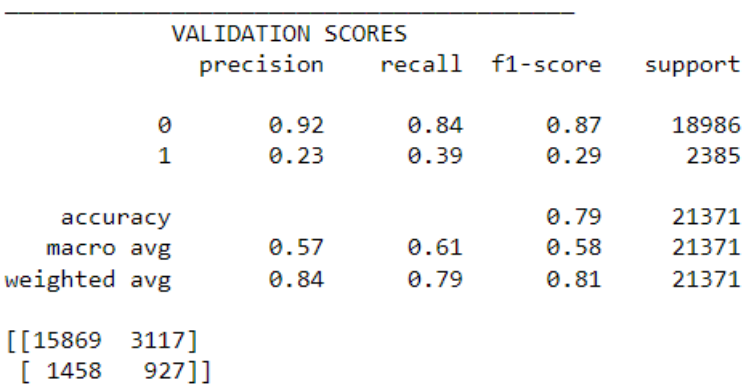
1. scikit-learn. sklearn.metrics.precision_score [Internet]. Available from: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.precision_score.html

2. De Diego IM, Redondo AR, Fernández RR, Navarro J, Moguerza JM. General Performance Score for classification problems. Appl Intell [Internet]. 2022;52(10):12049–63. Available from: https://doi.org/10.1007/s10489-021-03041-7

Section Q: Binary Validation Set Metrics and Confusion Matrices

As mentioned in the report's body, class 1 in the binary classification revealed lower f1 scores for all employed models. Figures 17 through 30 display this phenomenon, allowing for a direct comparison between the two classes' scores.

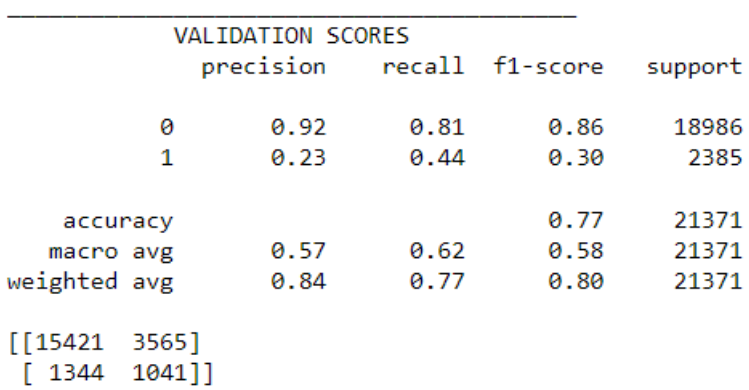
Figure 17: Binary Validation Set Metrics and Confusion Matrix for Logistic Regression



```
_________________________________________
            VALIDATION SCORES
              precision    recall  f1-score   support

           0       0.92      0.84      0.87     18986
           1       0.23      0.39      0.29      2385

    accuracy                           0.79     21371
   macro avg       0.57      0.61      0.58     21371
weighted avg       0.84      0.79      0.81     21371

[[15869  3117]
 [ 1458   927]]
```

Source: Authors.

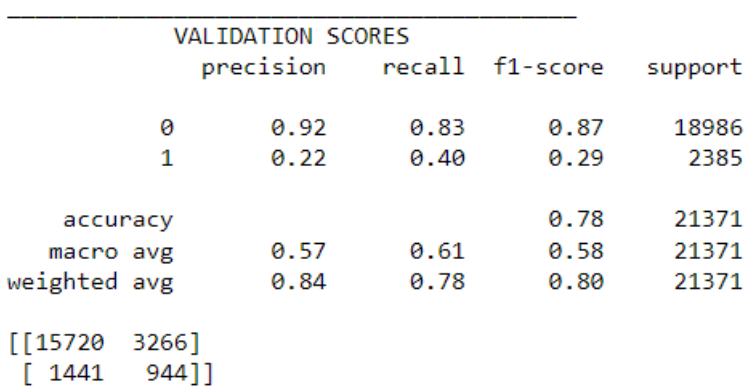
Figure 18: Binary Validation Set Metrics and Confusion Matrix for Neural Networks



```
_________________________________________
            VALIDATION SCORES
              precision    recall  f1-score   support

           0       0.92      0.81      0.86     18986
           1       0.23      0.44      0.30      2385

    accuracy                           0.77     21371
   macro avg       0.57      0.62      0.58     21371
weighted avg       0.84      0.77      0.80     21371

[[15421  3565]
 [ 1344  1041]]
```

Source: Authors.

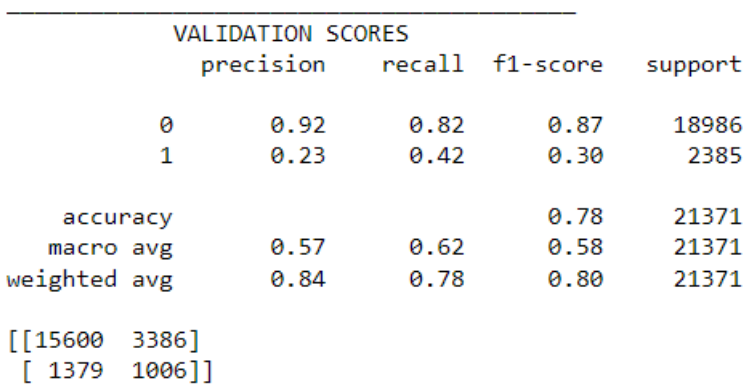
Figure 19: Binary Validation Set Metrics and Confusion Matrix for Gaussian Naive Bayes



```
_________________________________________
            VALIDATION SCORES
              precision    recall  f1-score   support

           0       0.92      0.83      0.87     18986
           1       0.22      0.40      0.29      2385

    accuracy                           0.78     21371
   macro avg       0.57      0.61      0.58     21371
weighted avg       0.84      0.78      0.80     21371

[[15720  3266]
 [ 1441   944]]
```

Source: Authors.

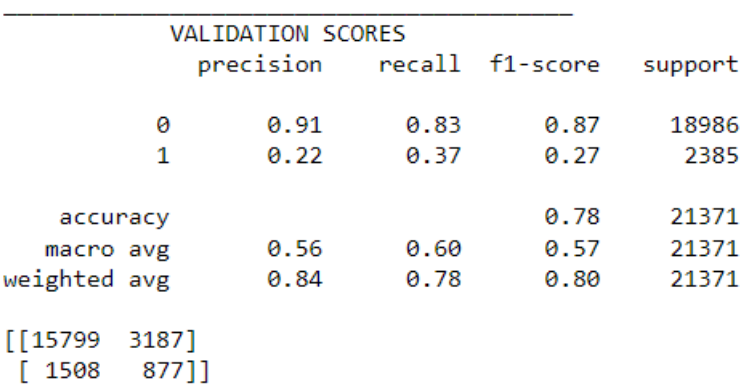
Figure 20: Binary Validation Set Metrics and Confusion Matrix for Bernoulli Naive Bayes



```
VALIDATION SCORES
              precision    recall  f1-score   support

           0       0.92      0.82      0.87     18986
           1       0.23      0.42      0.30      2385

    accuracy                           0.78     21371
   macro avg       0.57      0.62      0.58     21371
weighted avg       0.84      0.78      0.80     21371

[[15600  3386]
 [ 1379  1006]]
```

Source: Authors.

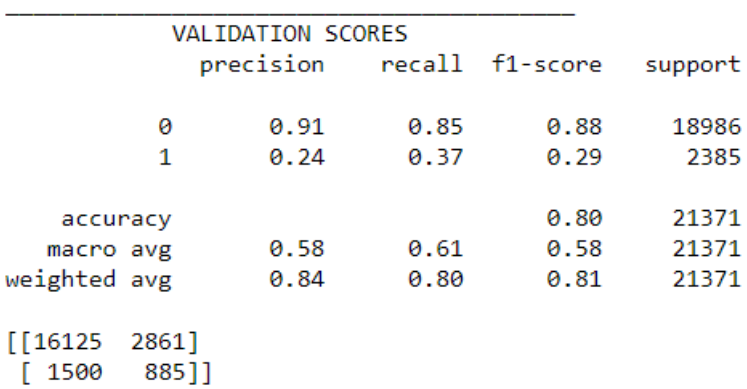
Figure 21: Binary Validation Set Metrics and Confusion Matrix for Decision Tree



```
VALIDATION SCORES
              precision    recall  f1-score   support

           0       0.91      0.83      0.87     18986
           1       0.22      0.37      0.27      2385

    accuracy                           0.78     21371
   macro avg       0.56      0.60      0.57     21371
weighted avg       0.84      0.78      0.80     21371

[[15799  3187]
 [ 1508   877]]
```

Source: Authors.

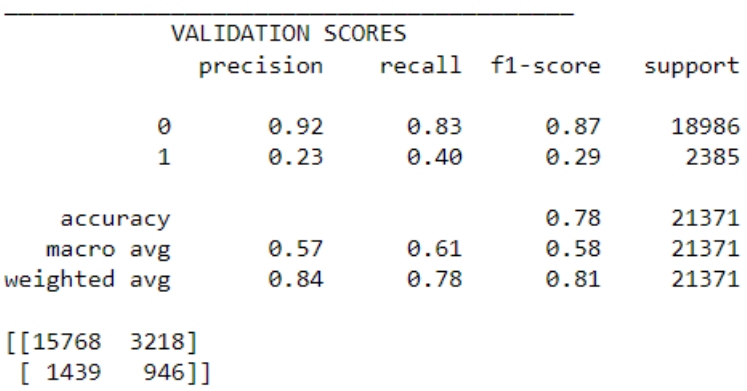
Figure 22: Binary Validation Set Metrics and Confusion Matrix for SVM (polynomial kernel)



```
VALIDATION SCORES
              precision    recall  f1-score   support

           0       0.91      0.85      0.88     18986
           1       0.24      0.37      0.29      2385

    accuracy                           0.80     21371
   macro avg       0.58      0.61      0.58     21371
weighted avg       0.84      0.80      0.81     21371

[[16125  2861]
 [ 1500   885]]
```

Source: Authors.

Figure 23: Binary Validation Set Metrics and Confusion Matrix for Linear SVC



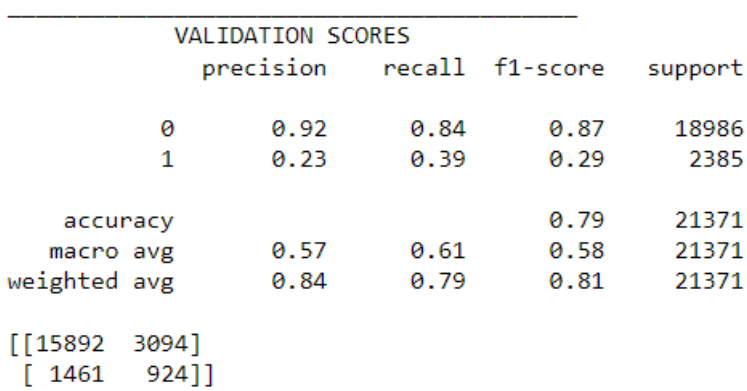
VALIDATION SCORES

	precision	recall	f1-score	support
0	0.92	0.83	0.87	18986
1	0.23	0.40	0.29	2385
accuracy			0.78	21371
macro avg	0.57	0.61	0.58	21371
weighted avg	0.84	0.78	0.81	21371

```
[[15768  3218]
 [ 1439   946]]
```

Source: Authors.

Figure 24: Binary Validation Set Metrics and Confusion Matrix for SGD



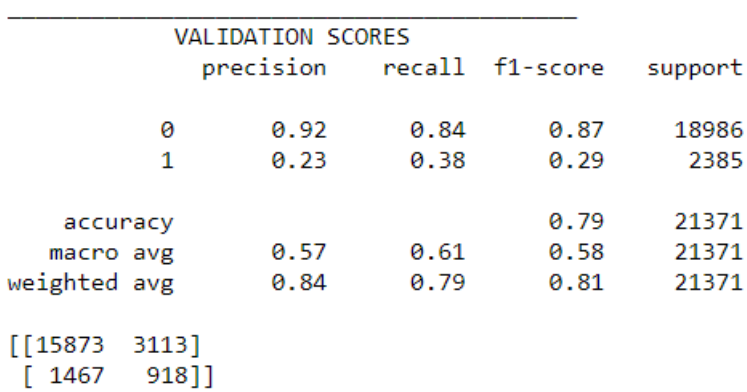
VALIDATION SCORES

	precision	recall	f1-score	support
0	0.92	0.84	0.87	18986
1	0.23	0.39	0.29	2385
accuracy			0.79	21371
macro avg	0.57	0.61	0.58	21371
weighted avg	0.84	0.79	0.81	21371

```
[[15892  3094]
 [ 1461   924]]
```

Source: Authors.

Figure 25: Binary Validation Set Metrics and Confusion Matrix for Random Forest



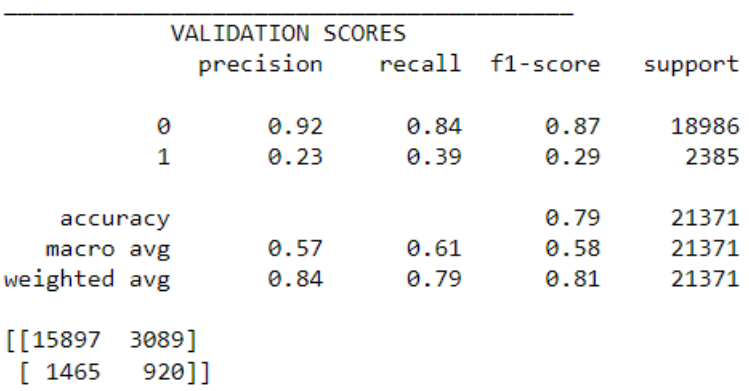
VALIDATION SCORES

	precision	recall	f1-score	support
0	0.92	0.84	0.87	18986
1	0.23	0.38	0.29	2385
accuracy			0.79	21371
macro avg	0.57	0.61	0.58	21371
weighted avg	0.84	0.79	0.81	21371

```
[[15873  3113]
 [ 1467   918]]
```

Source: Authors.

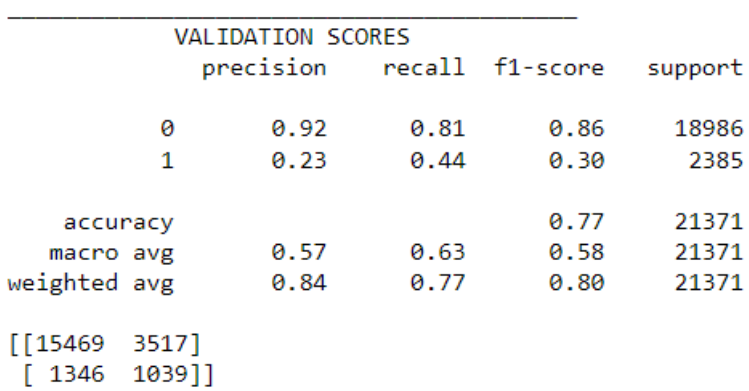
Figure 26: Binary Validation Set Metrics and Confusion Matrix for AdaBoost



```
_________________________________________
            VALIDATION SCORES
              precision    recall  f1-score   support

           0       0.92      0.84      0.87     18986
           1       0.23      0.39      0.29      2385

    accuracy                           0.79     21371
   macro avg       0.57      0.61      0.58     21371
weighted avg       0.84      0.79      0.81     21371

[[15897  3089]
 [ 1465   920]]
```

Source: Authors.

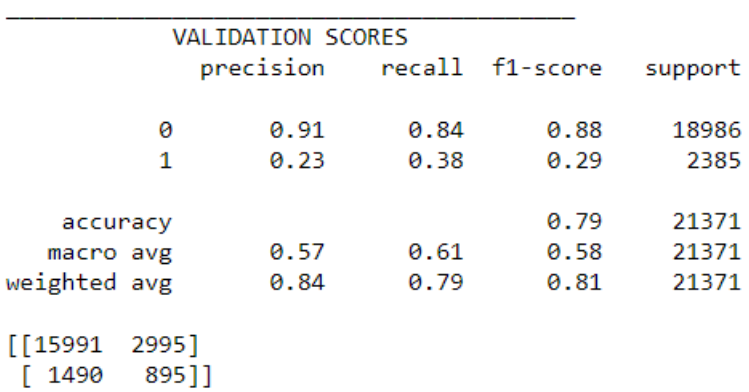
Figure 27: Binary Validation Set Metrics and Confusion Matrix for Bagging



```
_________________________________________
            VALIDATION SCORES
              precision    recall  f1-score   support

           0       0.92      0.81      0.86     18986
           1       0.23      0.44      0.30      2385

    accuracy                           0.77     21371
   macro avg       0.57      0.63      0.58     21371
weighted avg       0.84      0.77      0.80     21371

[[15469  3517]
 [ 1346  1039]]
```

Source: Authors.

Figure 28: Binary Validation Set Metrics and Confusion Matrix for Gradient Boosting



```
_________________________________________
            VALIDATION SCORES
              precision    recall  f1-score   support

           0       0.91      0.84      0.88     18986
           1       0.23      0.38      0.29      2385

    accuracy                           0.79     21371
   macro avg       0.57      0.61      0.58     21371
weighted avg       0.84      0.79      0.81     21371

[[15991  2995]
 [ 1490   895]]
```

Source: Authors.

Figure 29: Binary Validation Set Metrics and Confusion Matrix for Stacking

```
_________________________________________
            VALIDATION SCORES
              precision    recall  f1-score   support

           0       0.92      0.82      0.86     18986
           1       0.23      0.43      0.30      2385

    accuracy                           0.77     21371
   macro avg       0.57      0.62      0.58     21371
weighted avg       0.84      0.77      0.80     21371

[[15496  3490]
 [ 1357  1028]]
```

Source: Authors.

Figure 30: Binary Validation Set Metrics and Confusion Matrix for Voting

```
_________________________________________
            VALIDATION SCORES
              precision    recall  f1-score   support

           0       0.92      0.83      0.87     18986
           1       0.24      0.41      0.30      2385

    accuracy                           0.78     21371
   macro avg       0.58      0.62      0.59     21371
weighted avg       0.84      0.78      0.81     21371

[[15787  3199]
 [ 1402   983]]
```

Source: Authors.

Section R: Multiclass Validation Set Metrics and Confusion Matrix

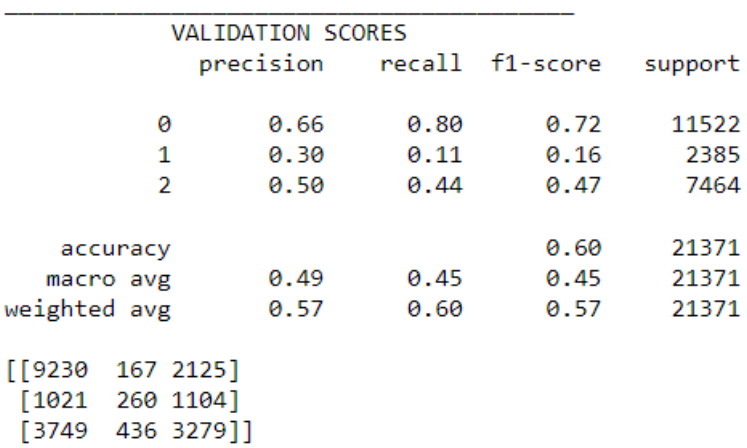
Once again, we display below the metrics and confusion matrices for all models tested in the multiclass chapter, allowing us to picture how class 1 gets lower scores compared to the remaining ones.

Figure 31: Multiclass Validation Set Metrics and Confusion Matrix for Logistic Regression

```
_________________________________________
            VALIDATION SCORES
              precision    recall  f1-score   support

           0       0.65      0.85      0.73     11522
           1       0.24      0.35      0.28      2385
           2       0.57      0.22      0.31      7464

    accuracy                           0.57     21371
   macro avg       0.49      0.47      0.44     21371
weighted avg       0.58      0.57      0.54     21371

[[9740 1009  773]
 [1135  831  419]
 [4168 1690 1606]]
```

Source: Authors.

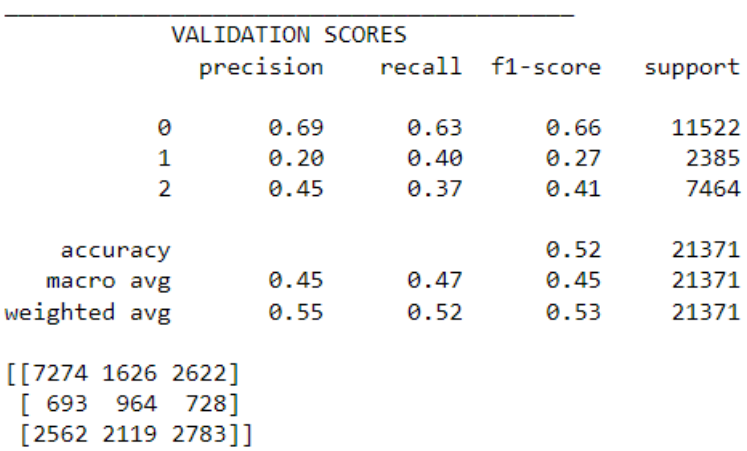
Figure 32: Multiclass Validation Set Metrics and Confusion Matrix for Decision Tree



```
_____________________________________________
            VALIDATION SCORES
              precision    recall  f1-score   support

           0       0.66      0.80      0.72     11522
           1       0.30      0.11      0.16      2385
           2       0.50      0.44      0.47      7464

    accuracy                           0.60     21371
   macro avg       0.49      0.45      0.45     21371
weighted avg       0.57      0.60      0.57     21371

[[9230  167 2125]
 [1021  260 1104]
 [3749  436 3279]]
```

Source: Authors.

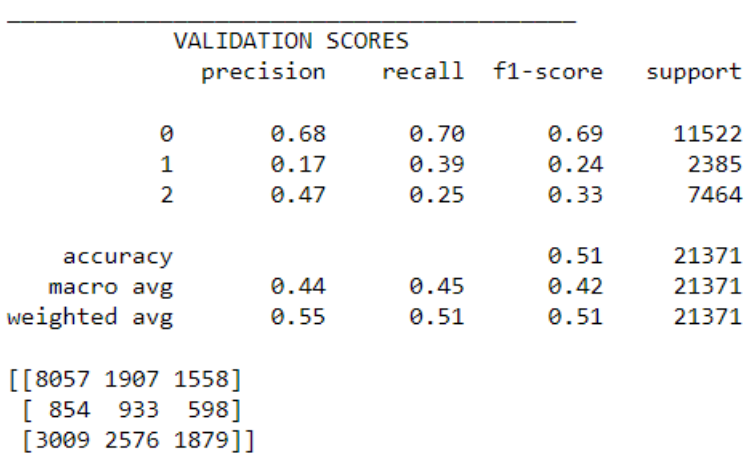
Figure 33: Multiclass Validation Set Metrics and Confusion Matrix for Gaussian Naive Bayes



```
_____________________________________________
            VALIDATION SCORES
              precision    recall  f1-score   support

           0       0.69      0.63      0.66     11522
           1       0.20      0.40      0.27      2385
           2       0.45      0.37      0.41      7464

    accuracy                           0.52     21371
   macro avg       0.45      0.47      0.45     21371
weighted avg       0.55      0.52      0.53     21371

[[7274 1626 2622]
 [ 693  964  728]
 [2562 2119 2783]]
```

Source: Authors.

Figure 34: Multiclass Validation Set Metrics and Confusion Matrix for Bernoulli Naive Bayes



```
_____________________________________________
            VALIDATION SCORES
              precision    recall  f1-score   support

           0       0.68      0.70      0.69     11522
           1       0.17      0.39      0.24      2385
           2       0.47      0.25      0.33      7464

    accuracy                           0.51     21371
   macro avg       0.44      0.45      0.42     21371
weighted avg       0.55      0.51      0.51     21371

[[8057 1907 1558]
 [ 854  933  598]
 [3009 2576 1879]]
```

Source: Authors.

Figure 35: Multiclass Validation Set Metrics and Confusion Matrix for Multinomial Naive Bayes

```
_________________________________________________
            VALIDATION SCORES
              precision    recall  f1-score   support

           0       0.63      0.89      0.73     11522
           1       0.21      0.37      0.27      2385
           2       0.63      0.07      0.13      7464

    accuracy                           0.55     21371
   macro avg       0.49      0.44      0.38     21371
weighted avg       0.58      0.55      0.47     21371

[[10233  1134   155]
 [ 1333   887   165]
 [ 4801  2110   553]]
```

Source: Authors.

Figure 36: Multiclass Validation Set Metrics and Confusion Matrix for Complement Naive Bayes

```
_________________________________________________
            VALIDATION SCORES
              precision    recall  f1-score   support

           0       0.62      0.89      0.73     11522
           1       0.21      0.40      0.28      2385
           2       0.64      0.03      0.05      7464

    accuracy                           0.54     21371
   macro avg       0.49      0.44      0.35     21371
weighted avg       0.58      0.54      0.44     21371

[[10300  1163    59]
 [ 1371   961    53]
 [ 4877  2392   195]]
```

Source: Authors.

Figure 37: Multiclass Validation Set Metrics and Confusion Matrix for Neural Networks

```
_________________________________________________
            VALIDATION SCORES
              precision    recall  f1-score   support

           0       0.66      0.82      0.73     11522
           1       0.22      0.20      0.21      2385
           2       0.55      0.35      0.43      7464

    accuracy                           0.59     21371
   macro avg       0.48      0.46      0.46     21371
weighted avg       0.57      0.59      0.57     21371

[[9496  695 1331]
 [1077  481  827]
 [3888  973 2603]]
```

Source: Authors.

Figure 38: Multiclass Validation Set Metrics and Confusion Matrix for Linear SVC

```
_________________________________________
            VALIDATION SCORES
              precision    recall  f1-score   support

           0       0.64      0.88      0.74     11522
           1       0.24      0.34      0.28      2385
           2       0.61      0.17      0.27      7464

    accuracy                           0.57     21371
   macro avg       0.50      0.46      0.43     21371
weighted avg       0.58      0.57      0.52     21371

[[10131   888   503]
 [ 1260   804   321]
 [ 4519  1653  1292]]
```

Source: Authors.

Figure 39: Multiclass Validation Set Metrics and Confusion Matrix for Random Forest

```
_________________________________________
            VALIDATION SCORES
              precision    recall  f1-score   support

           0       0.65      0.86      0.74     11522
           1       0.24      0.14      0.18      2385
           2       0.56      0.36      0.44      7464

    accuracy                           0.61     21371
   macro avg       0.48      0.45      0.45     21371
weighted avg       0.57      0.61      0.57     21371

[[9941  358 1223]
 [1150  333  902]
 [4099  689 2676]]
```

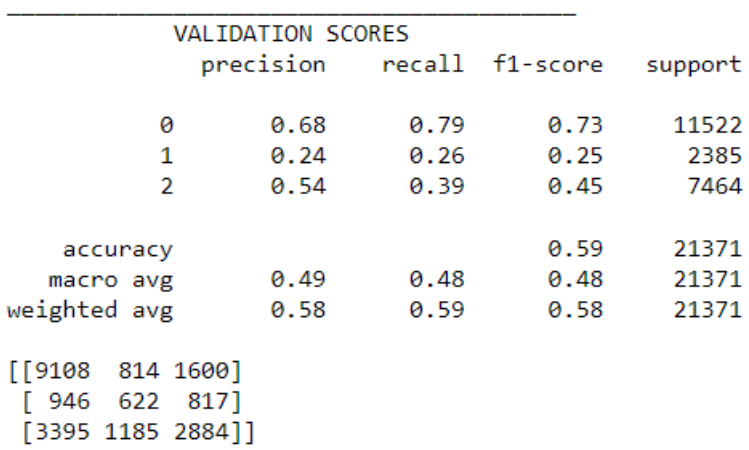
Source: Authors.

Figure 40: Multiclass Validation Set Metrics and Confusion Matrix for AdaBoost

```
_________________________________________
            VALIDATION SCORES
              precision    recall  f1-score   support

           0       0.64      0.89      0.74     11522
           1       0.27      0.04      0.07      2385
           2       0.55      0.36      0.44      7464

    accuracy                           0.61     21371
   macro avg       0.49      0.43      0.42     21371
weighted avg       0.57      0.61      0.56     21371

[[10280    67  1175]
 [ 1287    90  1008]
 [ 4588   174  2702]]
```

Source: Authors.

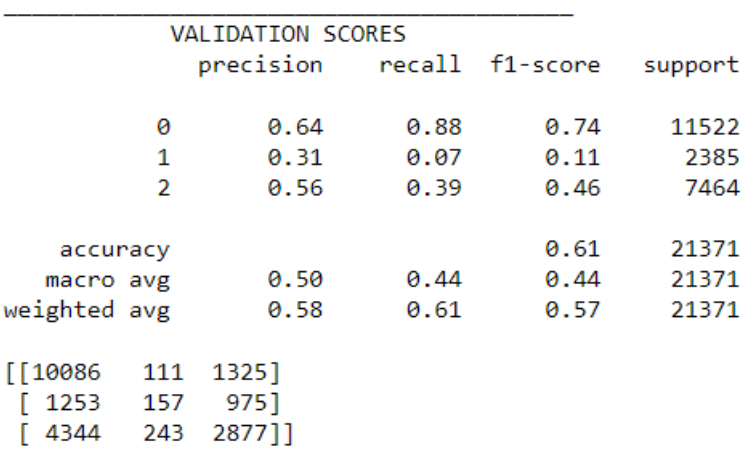
Figure 41: Multiclass Validation Set Metrics and Confusion Matrix for Bagging



```
                VALIDATION SCORES
                  precision    recall  f1-score   support

               0       0.68      0.79      0.73     11522
               1       0.24      0.26      0.25      2385
               2       0.54      0.39      0.45      7464

        accuracy                           0.59     21371
       macro avg       0.49      0.48      0.48     21371
    weighted avg       0.58      0.59      0.58     21371

[[9108  814 1600]
 [ 946  622  817]
 [3395 1185 2884]]
```

Source: Authors.

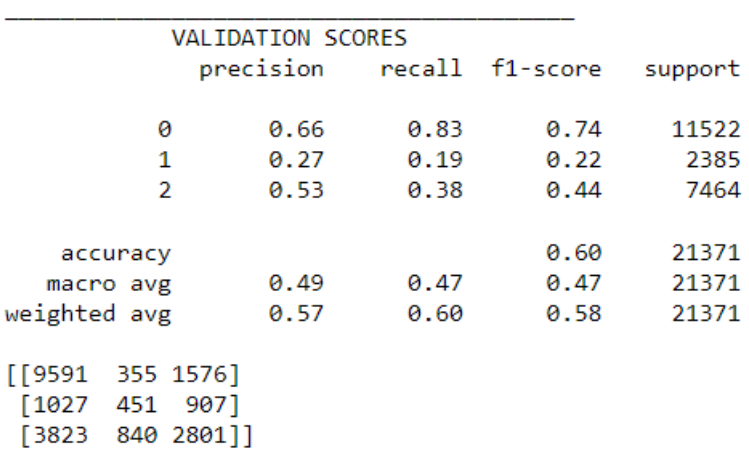
Figure 42: Multiclass Validation Set Metrics and Confusion Matrix for Gradient Boosting



```
                VALIDATION SCORES
                  precision    recall  f1-score   support

               0       0.64      0.88      0.74     11522
               1       0.31      0.07      0.11      2385
               2       0.56      0.39      0.46      7464

        accuracy                           0.61     21371
       macro avg       0.50      0.44      0.44     21371
    weighted avg       0.58      0.61      0.57     21371

[[10086   111  1325]
 [ 1253   157   975]
 [ 4344   243  2877]]
```

Source: Authors.

Figure 43: Multiclass Validation Set Metrics and Confusion Matrix for Stacking



```
                VALIDATION SCORES
                  precision    recall  f1-score   support

               0       0.66      0.83      0.74     11522
               1       0.27      0.19      0.22      2385
               2       0.53      0.38      0.44      7464

        accuracy                           0.60     21371
       macro avg       0.49      0.47      0.47     21371
    weighted avg       0.57      0.60      0.58     21371

[[9591  355 1576]
 [1027  451  907]
 [3823  840 2801]]
```

Source: Authors.

Figure 44: Multiclass Validation Set Metrics and Confusion Matrix for Voting

```
_____________________________________________
            VALIDATION SCORES
              precision    recall  f1-score   support

           0       0.66      0.85      0.74     11522
           1       0.28      0.14      0.18      2385
           2       0.56      0.40      0.46      7464

    accuracy                           0.61     21371
   macro avg       0.50      0.46      0.46     21371
weighted avg       0.58      0.61      0.58     21371

[[9834  285 1403]
 [1102  328  955]
 [3946  553 2965]]
```

Source: Authors.