Hyper-parameter Tuning of Untrained Classifiers via Nested Cross-Validation for the Classification of Healthy or Diabetic Patients

MLCB: Assignment 2

Glykeria Spyrou

Abstract

Diabetes is a prevalent chronic disease with significant health implications, making early diagnosis crucial for effective management and treatment. This study focuses on employing machine learning classifiers for the binary classification of diabetes based on medical features. Through a nested cross-validation approach, hyperparameters of various classifiers are optimized for improved model performance. Additionally, preprocessing steps and class balancing techniques are explored to assess their impact on classification. The study finds that Logistic Regression is the best performing classifier, demonstrating resilience against outliers and robustness across different datasets and balancing methods. This research contributes to the field by providing insights into the effectiveness of machine learning techniques for diabetes classification.

Keywords: diabetes, classification, machine learning, nested cross-validation, Logistic Regression, LDA, k-NN, Gaussian Naïve Bayes, SVM, class balancing

1 Introduction

Diabetes is a chronic disease that affects millions of people worldwide. It is categorized in Type 1, Type 2 and gestational diabetes. It accounts for a condition during which the pancreas does not produce enough insulin or even when the organism cannot manage effectively the insulin produced, leading to high levels of blood glucose [1]. This disease can have a huge toll in one's overall health, and according to a report on diabetes by the WHO, 0.5 million people died from diabetes in 2012 only at a global scale [2]. Overall, diabetes can cause numerous complications, even blindness, and its incidence rate is growing bigger every year. Therefore, prevention and early diagnosis are key factors towards providing proper medical intervention and treatment. In order to achieve that, many sectors under the biomedical umbrella are required to research for the discovery of diabetes biomarkers not only essential for its diagnosis but also for proper classification within the spectrum of diabetes. in this study the challenge of classifying patients into either healthy or diabetic is addressed, based on several medical features.

Accurate classification of diabetes can have a significant impact on patient outcomes and healthcare costs. Early detection and proper management of diabetes can prevent complications such as heart disease, kidney failure, and vision loss. Furthermore, advancements in machine learning and data analytics have opened up new opportunities for improving the accuracy and efficiency of diabetes classification.

Previous research in this field has explored various machine learning approaches for the binary classification of diabetes, including logistic regression, decision trees, and support vector machines, Naïve Bayes approaches, decision tree, random forest, linear descriminant analysis, and quadratic discriminant analysis [3]–[6]. While these methods show promise, they can sometimes struggle with overfitting or fail to generalize well to unseen data. Furthermore, developing machine learning models is a long process and the community continues exploring the relationship among clinical and how they can appropriately be captured in the proper model.

In this study, we employ nested cross-validation (nCV) to optimize the hyperparameters of various classifiers for binary classification of a diabetes dataset. Nested cross-validation is a robust method for hyperparameter tuning that uses an outer loop for model selection and an inner loop for hyperparameter optimization. This approach seeks to iden-

tify the best-performing model based on our selected evaluation metric. Once identified, the final model is trained on the complete dataset and saved for future use and testing on unseen data.

2 Material & Methods

2.1 Dataset Description

The diabetes dataset contains information related to diabetes classification downloaded from a GitHub repository. It includes 506 samples with 10 columns: nine features and one target variable. The target variable indicates whether an individual is healthy (class 0) or diabetic (class 1). The features and their full description, as well as they're ranges and mean values are shown on the table 1. All columns contain non-null values, indicating no missing data. However, there were numerous zero (0) values in fields that are considered to be out of range, for example, glucose concentration, blood pressure, skin thickness, insulin, and BMI. In this dataset, there are 329 samples of target 0 (healthy) accounting for 65.02% of the whole dataset and 177 samples of target 1 (diabetes) that account for the 34.98%. Therefore, the healthy class (0) is more prevalent than the diabetic class (1) and was taken into account throughout the training and performance evaluation methods of this assignment.

2.2 Data Exploration

For a better comprehension and exploration of the given diabetes dataset, several methods were employed. First of all, visualizations, such as box plots, were generated in order to explore the distributions of the features, allowing us to better observe the range and variance of each feature. Through these graphs we were able to identify skewness and potential outliers present in the data. Furthermore, Principal Component Analysis (PCA) was performed to reduce the dataset to two principal components, providing a lower-dimensional representation of the data. Scatter plots of the PCA results revealed the spread of the 2-class data in the 2-dimensional space and their overlap. A correlation heatmap was also plotted so as to examine the relationships that the features in the data have. This correlation analysis helped us have a better overview on the associations of the the features.

The key findings of this exploratory analysis were that even though there were no missing values in the data there was a big amount of zero values (0) as mentioned in the previous subsection. Because of the noise in the data, there was no distinct separation of each class in the 2D space. Additionally, through the box plots we discovered that there were multiple outlier values in the most of the features.

2.3 Preprocessing

Several preprocessing techniques were applied to prepare the data for analysis and modeling. Firstly, the dataset was filtered to remove rows with zero values in columns where zeros are considered irrational, such as Glucose, BloodPressure, SkinThickness, Insulin, and BMI. This filtering ensures that the data more accurately reflects true measurements. Additionally, any ID column present was removed as it is not useful for modeling. To address potential outliers in the data, an interquartile range (IQR) method was used with a threshold of 1.5. This method identified and removed data points outside the defined IQR range, thereby eliminating extreme values and improving data quality. No normalization or scaling techniques were applied to the dataset in this study. Further details on the data splitting process for training, validation, and testing will be described in the next subsection regarding nested cross-validation (nCV).

2.4 Nested Cross-Validation Pipeline Structure

The nested cross-validation (nCV) pipeline is a robust approach for model evaluation and hyperparameter optimization, providing unbiased model selection and performance estimation. The pipeline consists of two main loops: the outer loop and the inner loop, whose schema is depicted in the figure 1. The outer loop involves 5-fold stratified cross-validation. This loop serves as the primary evaluation phase, using different training and test splits in each iteration. It provides a more realistic estimate of the model's performance by simulating different scenarios of training and testing. Within each iteration of the outer loop, the inner loop consists of 3-fold stratified cross-validation. This inner loop is used to perform hyperparameter optimization for each classifier using *Optuna* [7]. By conducting multiple trials and evaluating the model performance using the Receiver Operating Characteristic - Area Under the Curve, (ROC-AUC) metric, the optimal hyperparameters for each classifier are selected.

The pipeline utilizes a diverse set of classifiers including Logistic Regression, Gaussian Naïve Bayes, k-Nearest Neighbors, Linear Discriminant Analysis, and Support Vector Machines. Each classifier has a defined hyperparameter search space based on its unique requirements and capabilities.

For the evaluation of each classifier's performance, such as Matthews Correlation Coefficient (MCC),

Feature	Description	Range	Mean
Pregnancies	Number of times a patient has been pregnant	0 - 17	3.88
Glucose	Plasma glucose concentration	0 - 198	120.5
BloodPressure	Diastolic blood pressure (mm Hg)	0 - 122	69.4
SkinThickness	Triceps skinfold thickness (mm)	0 - 60	20.48
Insulin	Serum insulin (mu U/ml)	0 - 744	76.67
BMI	Body Mass Index	0 - 67.1	32.17
DiabetesPedigreeFunction	Function representing diabetes pedigree	0.084 - 2.329	0.48
Age	Patient age (years)	21 - 81	33.27
ID	Identifier for the sample	N/A	N/A

Table 1: Feature Descriptions, ranges and mean values across the diabetes dataset.

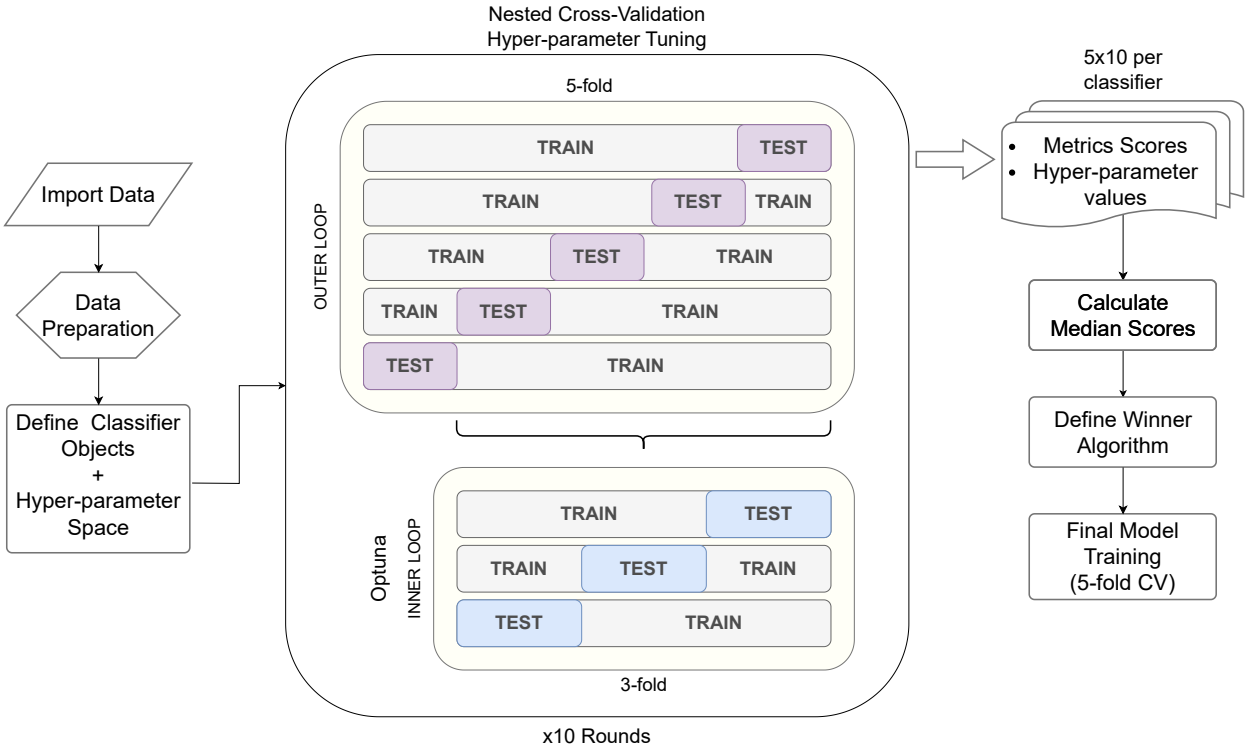

Figure 1: This flowchart illustrates the structure of the nested cross-validation pipeline, including data import, data preparation, classifier definition and hyperparameter optimization using *Optuna*, nested cross-validation with outer and inner loops, performance evaluation, and final model selection.

Balanced Accuracy, F1 Score, Precision, Recall, Average Precision, Specificity, Negative Predictive Value (NPV), and Precision-Recall AUC (PR-AUC). These metrics provide a comprehensive view of model performance, especially when dealing with imbalanced data. A priority list based on Recall, Precision, and PR-AUC is used for model selection, emphasizing metrics that are crucial for imbalanced datasets and binary classification. The final model is chosen based on the highest sum of median scores across the selected metrics.

2.5 Pipeline Stages

For the nested cross-validation implementation two separate categories of experiments were employed for hyperparameter tuning. The first category involved using the raw dataset directly for tuning the hyperparameters of each classifier, with a sole preprocessing of removing the 'ID' feature. The second category applied the preprocessing steps detailed in the previous subsection before tuning the hyperparameters. This allowed for a comparison of model performance when trained on the raw dataset versus the preprocessed data, providing insights into the impact of data preparation on the model's effectiveness.

The training of the model occurred within the inner loop of the nested cross-validation process. Using `OptunaSearchCV` for hyperparameter optimization, each classifier's hyperparameter space was explored. Stratified K-fold cross-validation (with 3 folds) was used to split the training data, and the models were trained using a specified scoring metric (ROC-AUC). This process allowed the selection of the best parameters for each classifier within each outer loop fold. Additionally, the nCV process involved 10 rounds of nested cross-validation with a 5-fold outer loop and a 3-fold inner loop for each classifier to ensure robust evaluation and tuning.

In the outer loop of the nested cross-validation process, each classifier was evaluated using the best parameters determined in the inner loop. Various performance metrics were calculated, including balanced accuracy, F1 score, recall, precision, average precision, and area under the curve of the precision-recall curve (PR-AUC), among others. These metrics provided a comprehensive assessment of each classifier's performance. ROC-AUC was used as the primary scoring metric because it effectively measures a model's ability to discriminate between the binary classes (healthy and diabetic), even when the classes are imbalanced. This metric is particularly useful for assessing model performance across a range of decision thresholds, making it well-suited for this binary classification problem.

The model selection stage involved determining the best classifier based on the evaluation results. Metrics such as median performance across rounds and confidence intervals for each classifier were considered. A priority list of evaluation metrics was established to prioritize certain metrics for imbalanced data, with recall, precision, and PR-AUC being the most important. This allowed for the selection of the best classifier based on priority scores and the confidence intervals for the top classifiers. The overall best classifier was identified by assessing the models' performance across all metrics and choosing the one with the highest overall score.

2.6 Bonus Task: Class Balancing

In addition to the baseline model, the impact of class balancing methods on the classification task by applying undersampling and oversampling techniques to the dataset was explored [8]. These methods aimed to address the class imbalance in the data and evaluate how they could improve the results. Four experiments were conducted: hyperparameter tuning on raw data with undersampling, raw data with oversampling, preprocessed data with undersampling, and preprocessed data with oversampling.

For undersampling, the *RandomUnderSampler* [9] object from the *imblearn* library was utilized. This method reduces the number of samples in the majority class to match the number of samples in the minority class. The *RandomUnderSampler* was initialized with a random state of 42 for reproducibility and a sampling strategy set to "majority." For oversampling, the Synthetic Minority Over-sampling Technique (SMOTE) [10] from the *imblearn* library was applied. *SMOTE* generates synthetic samples for the minority class to balance the class distribution, ensuring that the classes were equalized in size.

For evaluating the classifiers in this class balance category, the same approach was used as before, summing the scores of three priority metrics. This time, the chosen priority metrics were balanced accuracy, recall, and precision, which are appropriate given the now-balanced data.

3 Results & Discussion

3.1 Exploratory Data Analysis

The Exploratory Data Analysis (EDA) stage provides valuable insights into the characteristics of the dataset and how they change after preprocessing. Box plots were used to visualize the distribution of each feature, highlighting any skewness, outliers, or

extreme values present in the data. Comparing the box plots before and after preprocessing allowed for an assessment of the impact of data cleaning and outlier removal on the features. In the figures below, are depicted the boxplots before (Figure 2) and after (Figure 3) the preprecessing steps.

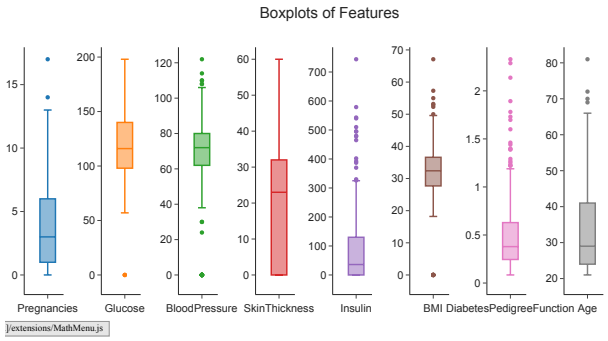

Figure 2: Box plots of the feature distributions of the raw dataset before any curation.

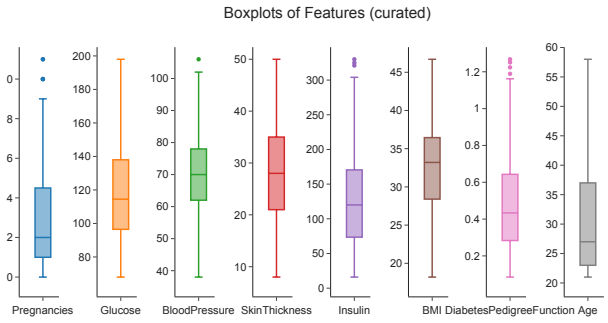

Figure 3: Box plots of the feature distributions of the dataset after handling zero values and removing outliers.

Principal Component Analysis (PCA) plots offered an overview of how the features are related, both before and after preprocessing. This analysis helped identify patterns and potential redundancies in the dataset, as well as the extent to which the preprocessing affected the relationships between features. In the figures below, are depicted the PCA plots before (Figure 4) and after (Figure 5) the preprecessing steps. In the raw dataset, it is obvious that in the space defined by the first two principal components that the two classes are not distinguishable and separable. However, in the figure after performing the preprocessing steps mentioned we can see that the classes are more distinguishable in the 2D space and that there is a small overlapping space.

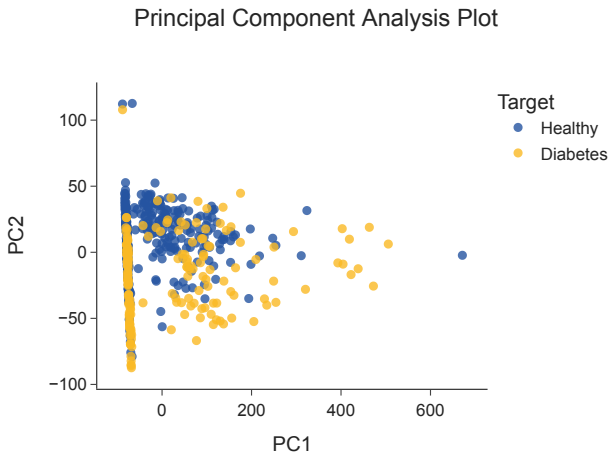

Figure 4: PCA plot of the samples as data points coloured by class label of the raw dataset before any curation.

[h] Additionally, correlation heat maps were generated to visualize the strength of relationships between the features in the dataset. This allowed for the identification of highly correlated features, which can impact model performance. Comparing the heatmaps before and after preprocessing provided insights into how the data cleaning process influenced these relationships. In the figures below, are depicted the correlation heatmaps of before (Figure 6) and after (Figure 7) the preprecessing steps. As seen by the figures, there are several feature pairs like age-pregnancies, skin thickness-BMI, glucose-insulin, whose correlation coefficient has grown stronger in a positive direction after preprocessing.

3.2 *Nested Cross-Validation Results*

After the execution of the 10 rounds of nested cross-validation for each of the five classifiers, the metric scores of the outer loops were used to visualize their distributions into boxplots that are shown in the figures 12 & 13. As we can see from the subplots of the two figures, the overall ranges in all of the classifiers broadens in the case of the preprocessed dataset, even though in several metrics the scores were higher. For the unprocessed dataset the winner model was LDA, and for the preprocessed dataset the winner was LDA again. Additionally, the worst performing model is apparently the SVM. In the figures 14 & 15 we have plotted only the medians of the scores in order to have a closer look and compare more efficiently the median metrics across all 10

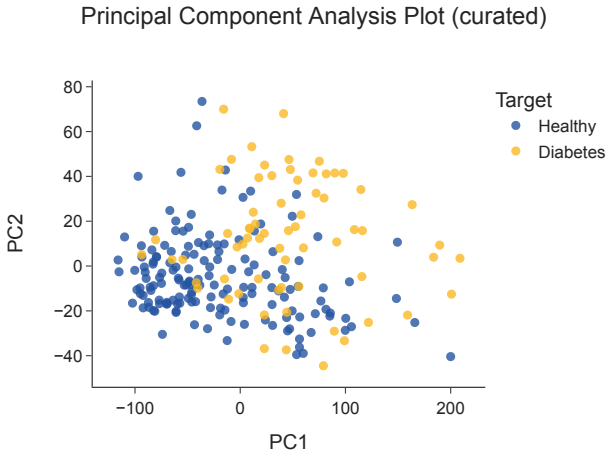

Figure 5: PCA plot of the samples as data points coloured by class label after handling zero values and removing outliers.

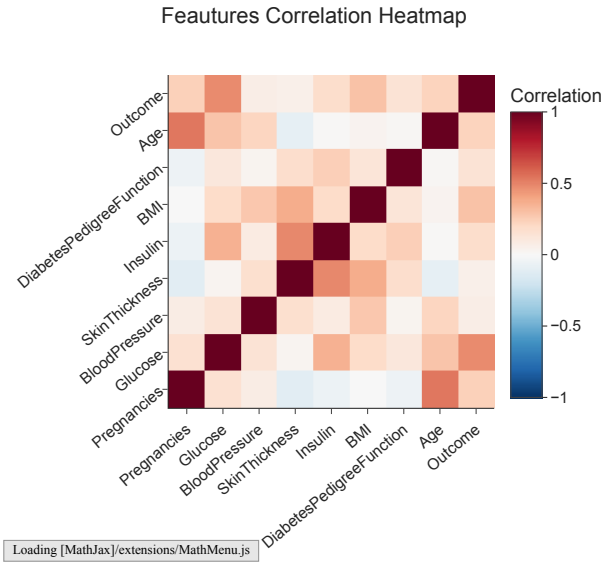

Figure 6: Correlation heatmap of the features of the raw dataset before any curation.

nCV rounds of hyperparameter tuning of the classifiers. By comparing the two plots, we can see that the overall performance of most classifiers dropped when trained using the preprocessed dataset.

3.3 Bonus Task: Class Balancing

The same pipeline as before was applied for the undersampled and the oversampled datasets that were generated. Therefore, the same visualizations were created in order to evaluate and assess the performance of each classifier in the 4 different scenarios that we created. In the figures 8 % 9 are depicted the two categories for which the class was balanced using the undersampling method. On the contrary, in the figures 10 % 11 are depicted the two categories for which the class was balanced using the oversampling method. The winner algorithms for these four occasions were the Logistic Regression and k-NN respectively for the undersampling and oversampling methods.

In the next figures (16 & 17) we have the barplots of the median values across of the metrics of the five classifiers for the undersampling method. While in the figures 18 and 19 we have the barplots of the median values across of the metrics of the five classifiers for the oversampling method. As for the performance of the classifiers between these two different methods for handling class imbalance, we saw that overall undersampling oversampling have very similar performance, but certainly classifier performed better than in the experiments were imbalanced data were involved.

3.4 Final Model Selection

As shown above, the experiments in which the models managed to showcase an overall better performance across the majority of the metrics were the once were data preprocessing was applied. More specifically, better results were achieved upon using the oversampling method for class balancing. For the evaluation of each classifier, the sum of the priority metrics was used to compare their performance. As seen above, across the different experiments, there was not a consensus for the best algorithm. However, LDA appear one time (2), k-NN two times (2), Logistic Regression two times (2). SVM was one of the algorithms that had the worst performances and is the first to be eliminated. Logistic Regression and LDA appear to have very similar metric scores throughout the different experiments. Finally, Logistic Regression is selected as a final model to be trained, since it shows a very good overall performace across all different experiments, and is suspected to yield better results on unseen data. Furthermore Logistic Regression managed to receive first place in both curated and uncurated experiments, explaining probability less sensitivity to outliers in a dataset, making it more robust. Overall, Logistic Regression doesn't assume a linear relationship between the features of the dataset, while it can also avoid overfitting through the L1 and L2 regularization terms.

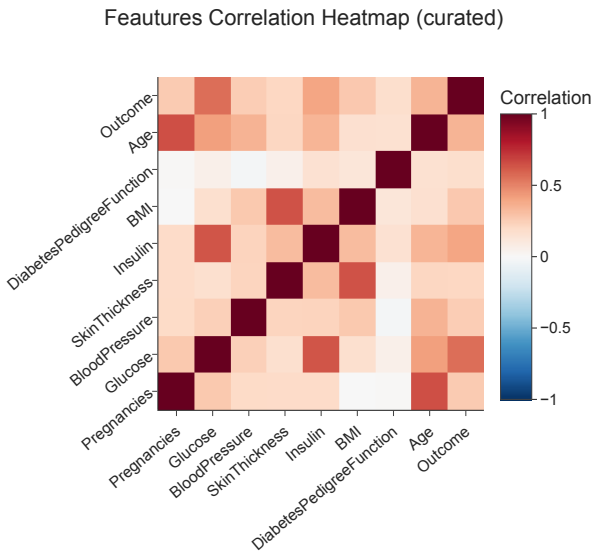

Figure 7: Correlation heatmap of the features after handling zero values and removing outliers.

4 Conclusions

In the present study, various machine learning classifiers were evaluated for their performance in the binary classification of diabetes using nested cross-validation and hyperparameter tuning. Data preprocessing, including the removal of outliers and zero values, didn't seem to improve the overall performance of the classifiers. Additionally, the use of class balancing methods such as undersampling and oversampling further enhanced classifier performance, with oversampling showing slightly better results overall. Logistic Regression emerged as the best-performing model across different datasets and balancing methods, demonstrating robustness and resilience against outliers. This classifier was chosen as the final model due to its consistent performance and resistance to overfitting, making it a reliable choice for unseen data. The study illustrates the potential of machine learning in diabetes classification and emphasizes the importance of proper data preparation and model selection. The chosen model can serve as a strong foundation for future work in the early diagnosis and management of diabetes.

Disclosure

ChatGPT-3.5 was used as an aid for the execution of some tasks of this assignment. Since such models are prone to errors when given a very specific task, and sometimes enter infinity loops of suggesting bug fixes that were proven to not work out in the end, this aforementioned model was used for building a generalized corpus for the nCV class, for documentation in custom functions, code debugging, and finally grammar check. The appropriate intervention in the generated code was applied in order to best meet not only the needs of each task but also the needs and preferences of the author. It was also used for general chatting around the general topic of this assignment.

References

[1] M. Z. Banday, A. S. Sameer, and S. Nissar, "Pathophysiology of diabetes: An overview," *Avicenna J Med*, vol. 10, no. 4, pp. 174–188, 2020, ISSN: 2231-0770, 2249-4464. DOI: 10.4103/ajm.ajm_53_20.

[2] G. Roglic, "WHO Global report on diabetes: A summary," *International Journal of Noncommunicable Diseases*, vol. 1, no. 1, p. 3, –Jun. 2016, ISSN: 2468-8827. DOI: 10.4103/2468-8827.184853.

[3] S. G. Rabiha, A. Wibowo, Lukas, and Y. Heryadi, "Diabetes Classification Using Support Vector Machine : Binary Classification Model," in *2021 4th International Conference on Information and Communications Technology (ICOIACT)*, 2021, pp. 280–284. DOI: 10.1109/ICOIACT53268.2021.9563994.

[4] N. P. Tigga and S. Garg, "Prediction of Type 2 Diabetes using Machine Learning Classification Methods," *Procedia Computer Science*, International Conference on Computational Intelligence and Data Science, vol. 167, pp. 706–716, 2020, ISSN: 1877-0509. DOI: 10.1016/j.procs.2020.03.336.

[5] Md. Maniruzzaman, N. Kumar, Md. Menhazul Abedin, *et al.*, "Comparative approaches for classification of diabetes mellitus data: Machine learning paradigm," *Computer Methods and Programs in Biomedicine*, vol. 152, pp. 23–34, 2017, ISSN: 0169-2607. DOI: 10.1016/j.cmpb.2017.09.004.

[6] Md. Maniruzzaman, M. J. Rahman, B. Ahammed, and M. M. Abedin, "Classification and prediction of diabetes disease using machine learning paradigm," *Health Inf Sci Syst*, vol. 8, no. 1, p. 7, 2020, ISSN: 2047-2501. DOI: 10.1007/s13755-019-0095-z.

[7] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama. "Optuna: A Next-generation Hyperparameter Optimization Framework." arXiv: 1907.10902 [cs, stat]. (2019), preprint.

[8] guest_blog. "10 Techniques to Solve Imbalanced Classes in Machine Learning (Updated 2024)," Analytics Vidhya. (2020), [Online]. Available: https://www.analyticsvidhya.com/blog/2020/07/10-techniques-to-deal-with-class-imbalance-in-machine-learning/.

[9] "RandomUnderSampler — Version 0.12.2." (), [Online]. Available: https://imbalanced-learn.org/stable/references/generated/imblearn.under_sampling.RandomUnderSampler.html.

[10] R. Blagus and L. Lusa, "SMOTE for high-dimensional class-imbalanced data," *BMC Bioinformatics*, vol. 14, no. 1, p. 106, 2013, ISSN: 1471-2105. DOI: 10.1186/1471-2105-14-106.

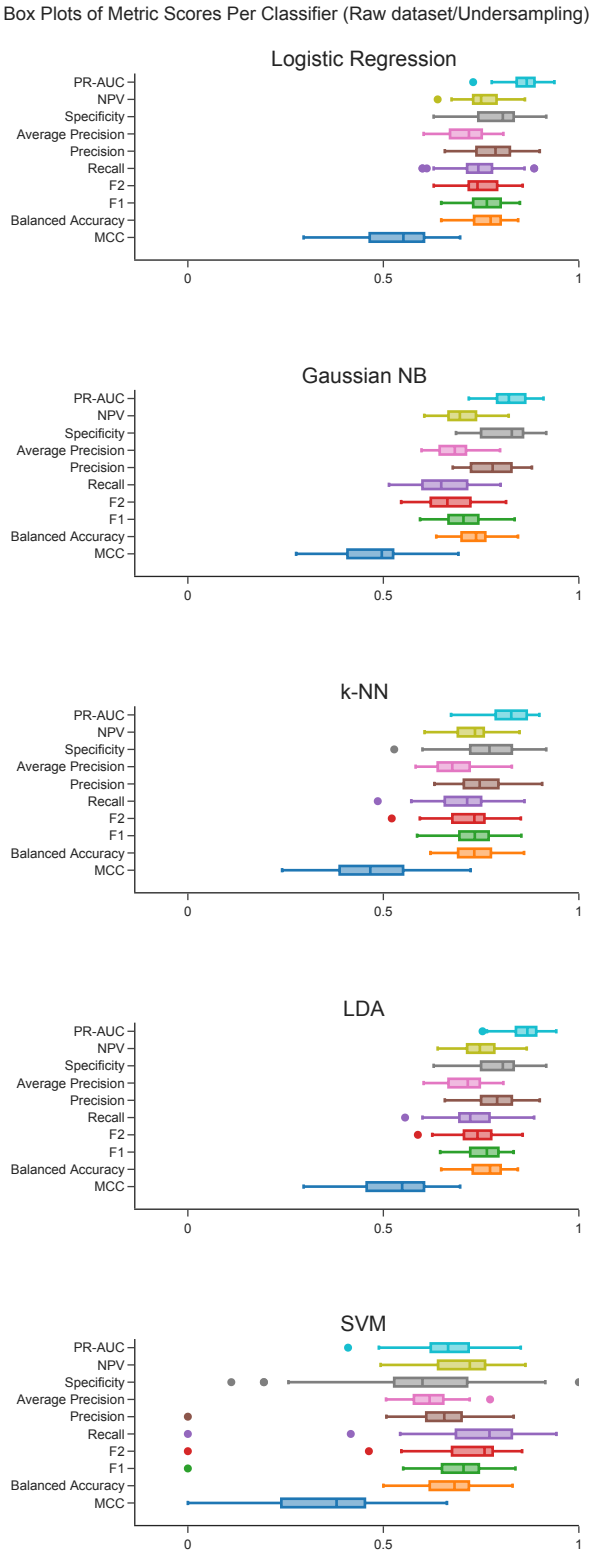

Figure 8: Boxplots of all metric scores per classifier across all 10 rounds of nested cross-validation for the raw dataset using the undersampling method for class balance.

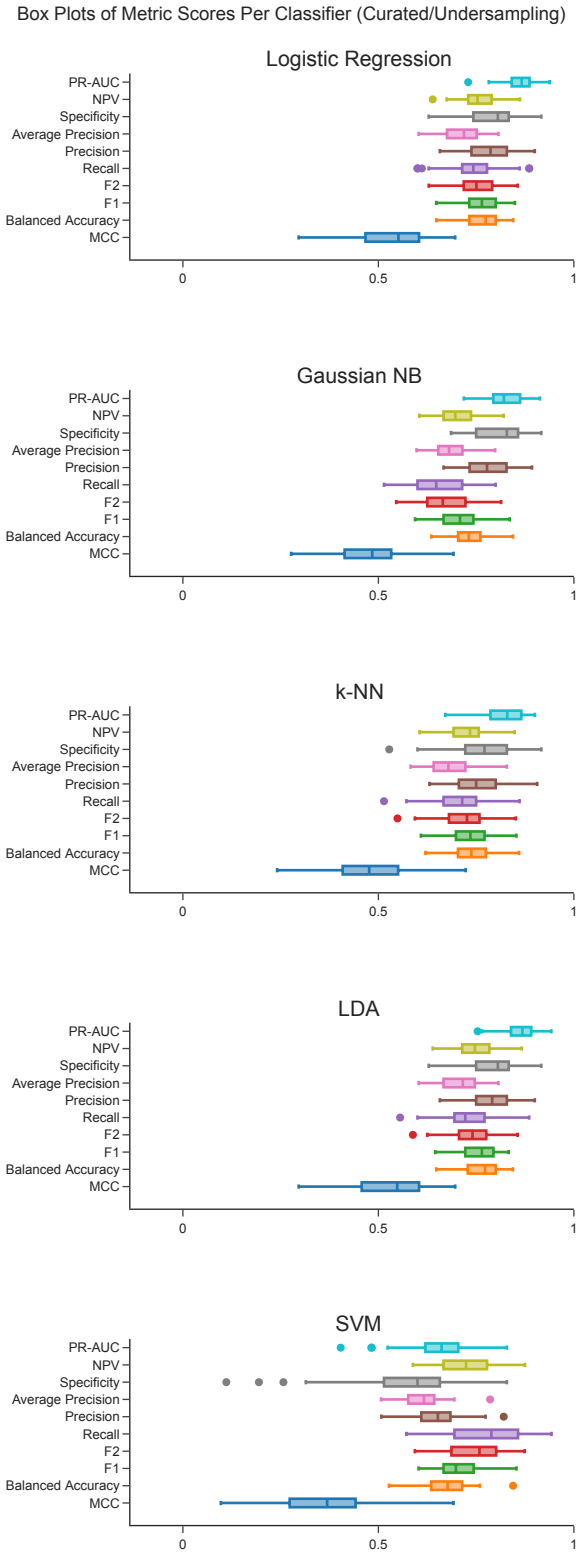

Figure 9: Boxplots of all metric scores per classifier across all 10 rounds of nested cross-validation for the preprocessed dataset using the undersampling method for class balance.

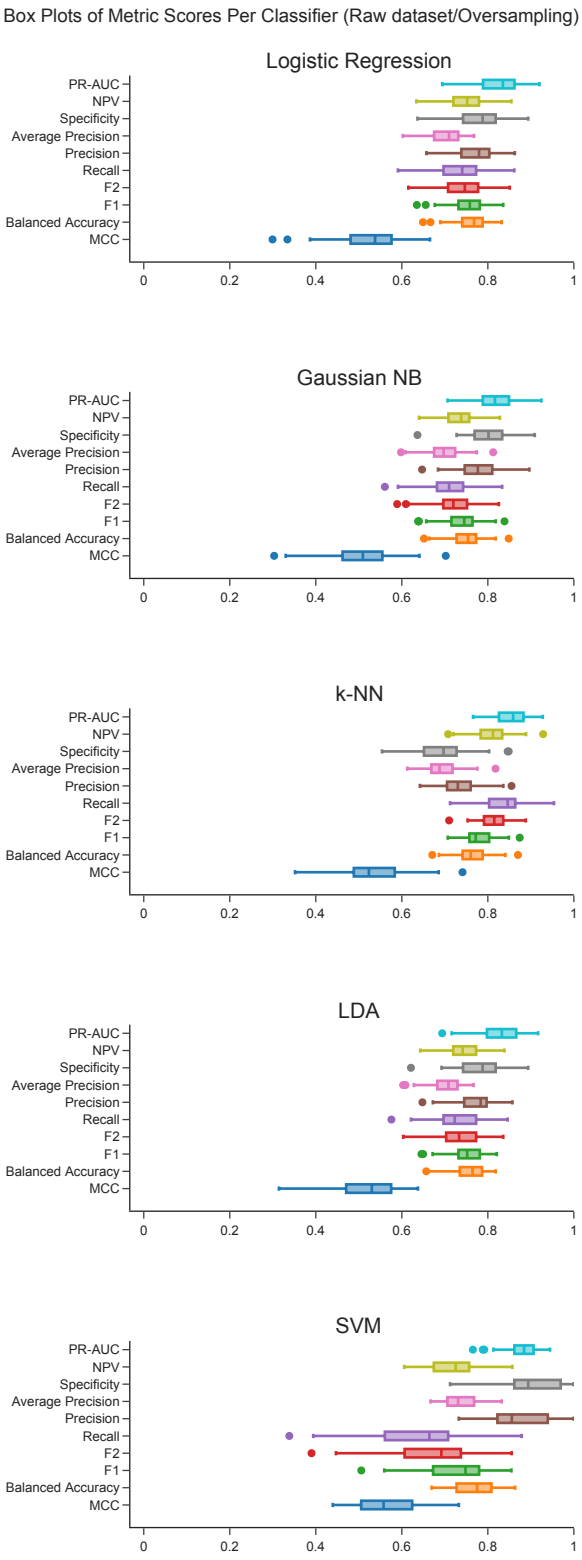

Figure 10: Boxplots of all metric scores per classifier across all 10 rounds of nested cross-validation foracross all 10 rounds of nested cross-validation for the raw dataset using the oversampling method forthe preprocessed dataset using the oversampling class balance. method for class balance.

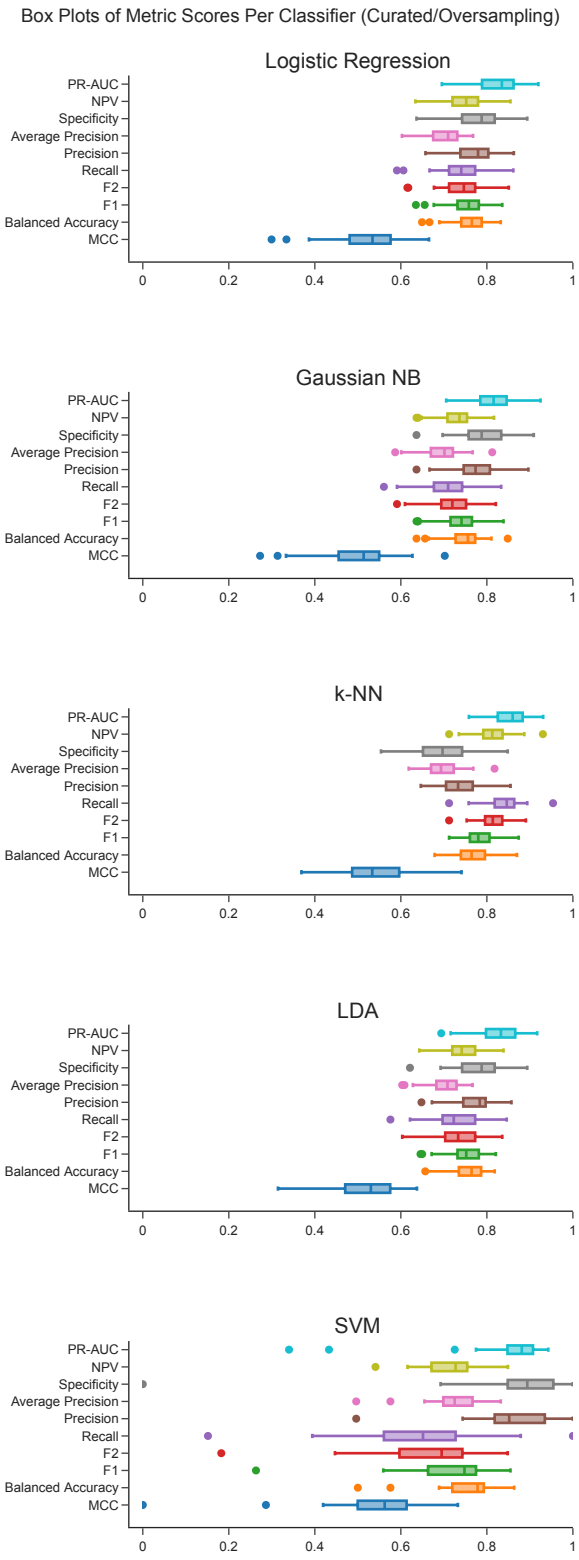

Figure 11: Boxplots of all metric scores per classifier

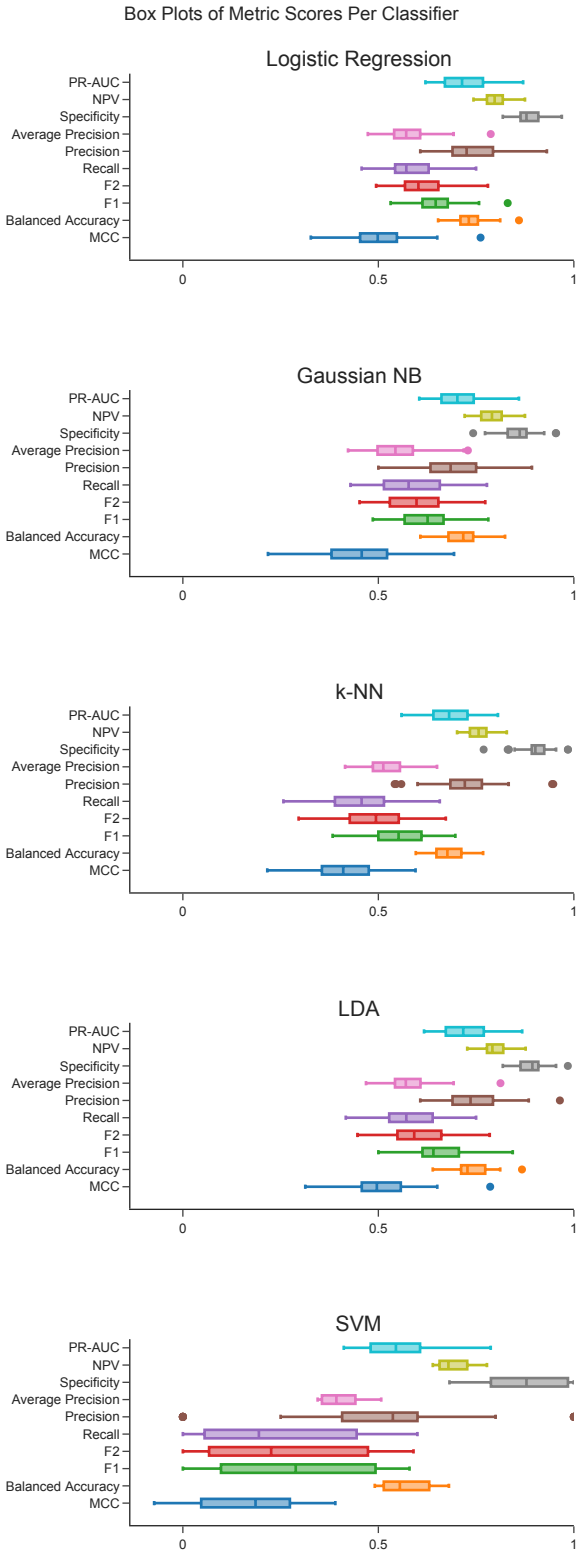

Figure 12: Boxplots of all metric scores per classifier across all 10 rounds of nested cross-validation for the raw dataset.

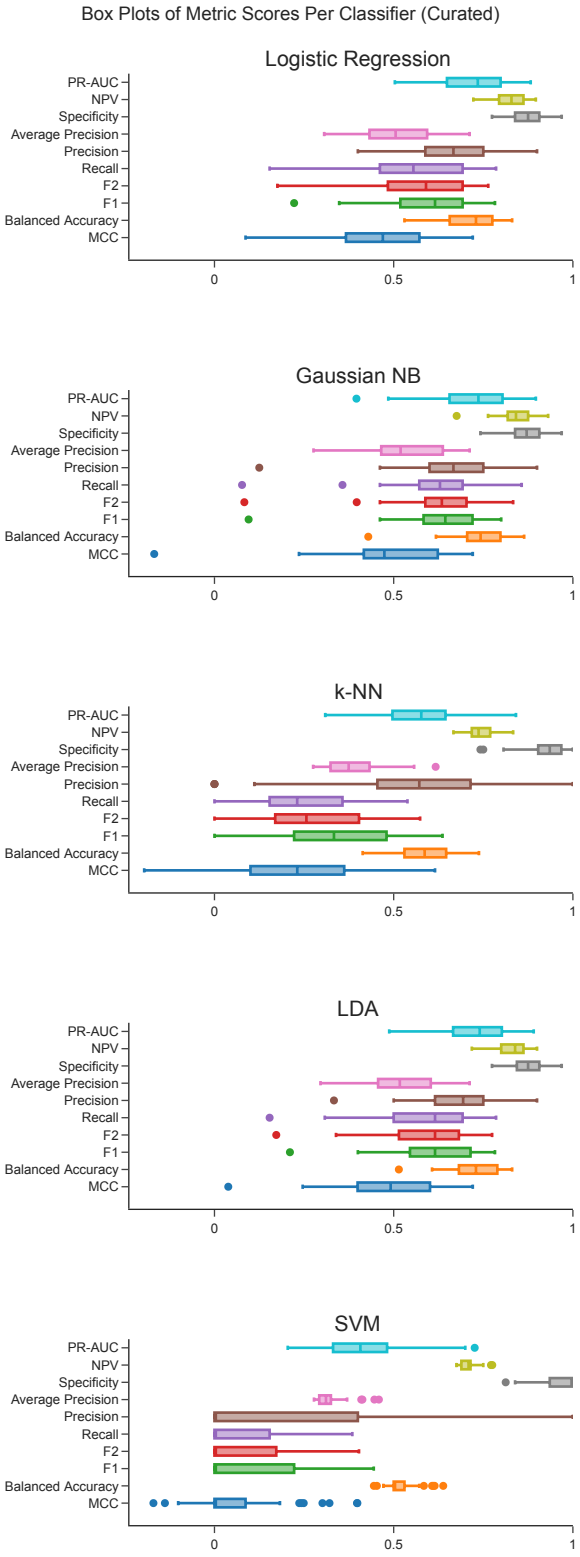

Figure 13: Boxplots of all metric scores per classifier across all 10 rounds of nested cross-validation for the preprocessed dataset.

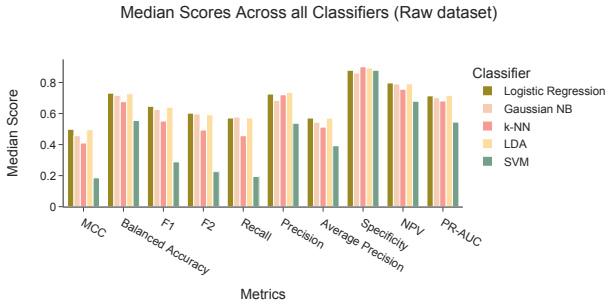

Figure 14: Barplots of the median scores of the classifiers of the raw dataset.

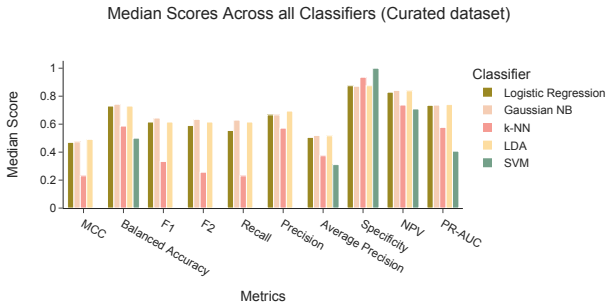

Figure 15: Barplots of the median scores of the classifiers of the preprocessed dataset.

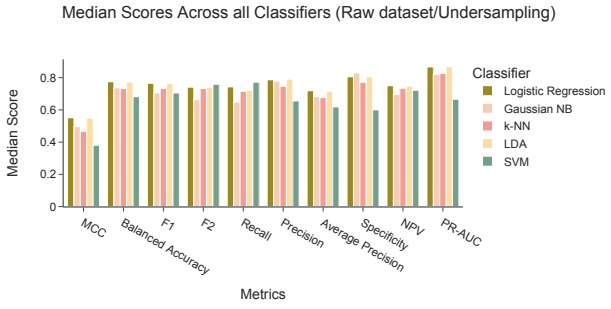

Figure 16: Barplots of the median scores of the classifiers of the raw dataset, while using the undersampling method for class balance.

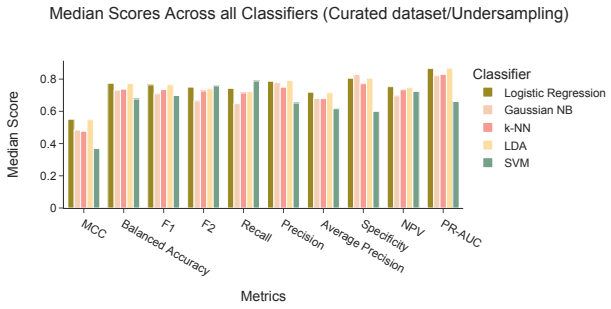

Figure 17: Barplots of the median scores of the classifiers of the preprocessed dataset, while using the undersampling method for class balance.

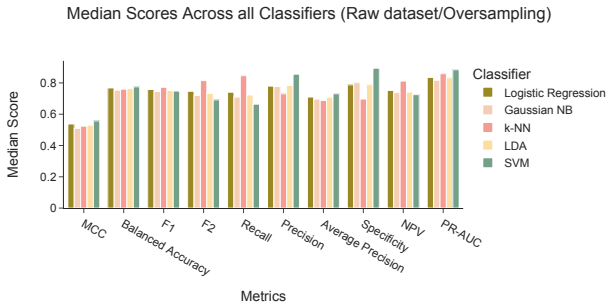

Figure 18: Barplots of the median scores of the classifiers of the raw dataset, while using the oversampling method for class balance.

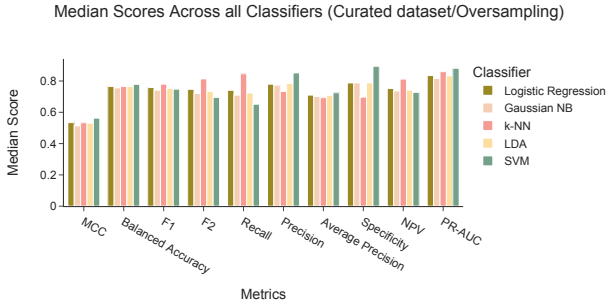

Figure 19: Barplots of the median scores of the classifiers of the preprocessed dataset, while using the oversampling method for class balance.