CS 178: Final Project

An investigation of classification methods for hospital readmission using diabetes dataset

Team Name: **at&t**

June 10, 2024

Names	Teresa Hempen Tiffany Nguyen Aashi Singh
IDs	thempen tqnguye6 aashis
Dataset	Diabetes
Classification Methods	kNN Logistic regression Feed forward neural network Random forest

1 Summary

This project investigates various classification methods for hospital readmissions using the diabetes dataset from UC Irvine's Machine Learning Repository. Our classification methods included kNN, logistic regression, feedforward neural networks, and random forest. We found that random forest counters the problem of class imbalance in the diabetes dataset with its effective feature selection. Even so, recall was a persistent challenge for all classification methods due to the underrepresentation of readmitted patients.

2 Data Description

The diabetes dataset covers 10 years of hospital records across 130 US hospitals. Each of the more than 100,000 rows corresponds to the record of a patient diagnosed with diabetes. The dataset contains 50 categorical and integer features, and the target classification is readmission to the hospital, identified as readmitted within 30 days, readmitted after 30 days, or not readmitted after discharge.

This dataset was introduced by Strack et al. in the 2014 paper "Impact of HbA1c Measurement on Hospital Readmission Rates" in which multivariable logistic regression was used to measure the relationship between HbA1C measurement and early readmission. Subsequent studies have focused on resolving the class imbalance challenge presented by readmission data using feature selection algorithms (Du et al., 2020). Class imbalance refers to the fact that positive cases are underrepresented in the dataset, with early readmission within 30 days representing only 11% of patients.

We cleaned and preprocessed that data largely following the methodology originally described by Stack et al. This included dropping sparse features (**weight, payer_code, medical_specialty, max_glu_serum**), including only the first encounter per patient, and ignoring patients for whom readmission is not applicable (patients who died or were discharged to hospice). **A1Cresult** and **age** were mapped to integer values so they could be used when in kNN and logistic regression. All numeric values were also scaled. A patient is labeled 0 if they were not readmitted and labeled 1 if they were readmitted at any time, as the class imbalance was too extreme when considering readmission within only 30 days. The original dataset contained over 100,000 patient records. Rows with missing data were dropped, and **A1Cresult** was fairly sparse, so we were left with 12,845 patient records after cleaning the data, 38% of whom were readmitted to the hospital. This slight class imbalance can be seen in **Figure 1.** Other data exploration included summary statistics and pairplots for numeric features, which can be found in **Appendix A.**

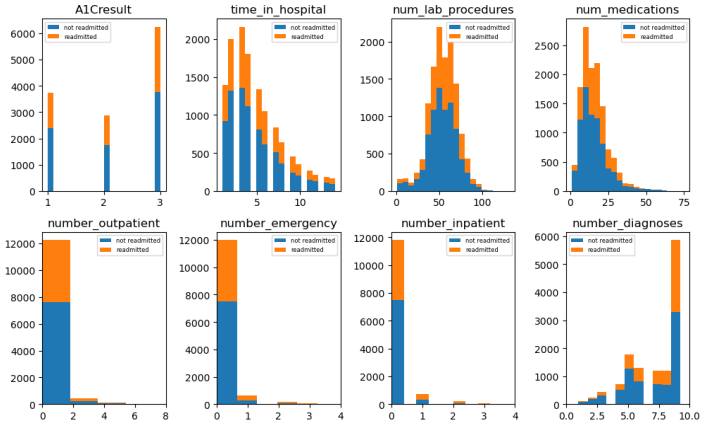

Figure 1. Stacked bar charts for numeric features.

3 Classifiers

Below is a list of the classifiers explored. The scikit-learn package was used for all methods.

kNN Classifier

Given a set of training data X and labels y, the k-Nearest Neighbors classifier assigns an unlabeled feature vector x based on the labels of its k-nearest neighbors. There is no training required for kNN classifiers. For prediction, the distance from the new vector to all training vectors is calculated, the k smallest distances are found, and the majority label of the corresponding points is assigned to the new point. An odd-valued k is typically chosen to avoid random tie-breaking for the majority label. The primary hyperparameter for tuning is the value of k, with k and classifier complexity being inversely related.

Logistic Classifier

The logistic regression classifier is widely used for binary classification tasks. Logistic regression estimates the probability that a given input point belongs to a certain class by fitting a logistic function (sigmoid) to a linear combination of the input features. The main hyperparameter for logistic regression is the regularization strength, C, which controls the trade-off between fitting the training data well and keeping the model parameters small to prevent overfitting.

Feedforward Neural Network

The feedforward neural network passes the input data to the input layer, through the hidden layer, and lastly to the output layer. In a feedforward neural network, there is only one hidden layer, which is composed of hidden units. Increasing the number of hidden units leads to more complex decision boundaries. A unit applies a non-linear activation function to the input. For training, stochastic gradient descent can be used. The main hyperparameters for neural networks include number of layers, number of hidden units, type of activation function, regularization weight, learning rate, and batch size for stochastic gradient descent.

Random Forest Classifier

A random forest classifier takes advantage of bagging and randomness to decrease the high variance typical of decision trees. Given a set of training data X and labels y, a single decision tree will learn a set of nested if-then-else statements, which form a tree: internal nodes branch on a feature at a threshold value determined by some metric, such as the Gini index, and leaf nodes represent the label prediction. While the output of decision trees is easily explainable, they can tend to overfit to the training data as the depth of the tree increases. Therefore, random forests use bootstrap aggregating, or bagging. The training data X is sampled with replacement B times, a decision tree is trained on each sample b, and the trees are aggregated for classification with the predicted label being that with the highest vote count from the B trees. The primary hyperparameters for tuning are the number of trees (corresponding to the number of bootstrap samples), the number of features randomly selected for search at each node, the maximum depth of each tree, and the minimum number of samples.

4 Experimental Setup

A random three-way split was used to create separate training, validation, and test sets, containing 60%, 20%, and 20% of the original dataset respectively. We assessed the models using several metrics, including classification accuracy, precision, and recall score, which provides a comprehensive evaluation of the model's performance across different thresholds. Specifically, classification accuracy was used to measure the proportion of correctly predicted instances, while precision and recall were used to evaluate the quality of the positive predictions. Hyperparameters were ultimately chosen based on highest validation accuracy.

kNN Classifier

We trained and tuned our kNN classifier using various subsets of the numeric features. As such, we used euclidean distance rather than defining any custom distance metrics. We swept over the values k (10, 50, 100, 150), first considering all numeric features. However, the curse of dimensionality applies to kNN: the feature space becomes increasingly sparse as the number of features grows, decreasing the effectiveness of Euclidean distance as a classification metric. Therefore, we then swept over the same k values for all pairwise combinations of the numeric features. The best feature pair and k value were selected based on the highest validation accuracy.

Logistic Classifier

To determine the optimal hyperparameters, particularly the regularization strength, C, we employed a systematic approach. We tested multiple values of C (0.01, 0.1, 1, 10, 50) using the training and validation sets. For each value of C, we trained the logistic regression model with an L1 penalty using the 'liblinear' solver, and recorded the training and validation accuracies. The best performing hyperparameter was selected based on the highest validation accuracy.

Feedforward Neural Network

For the feedforward neural network, we tested multiple values for the following hyperparameters: hidden_layer_sizes (50, 100, 150, 200, 250, 300), alpha (1, 0.1, 0.01, 0.001), learning_rate_init (1, 0.1, 0.01, 0.001, 0.0001), max_iter (50, 100, 150, 200, 250, 300), and batch_size (50, 100, 150, 200, 250, 300, 500). The hyperparameters activation = relu and solver = sgd were used.

Random Forest Classifier

We performed random search over the following values to find the best hyperparameters for random forest: numbers of trees (10, 50, 100, 200, 500, 1000), max_depth (2, 8, 32, 512, None), and min_samples_leaf (1, 2, 4, 8, 16). The remaining scikit-learn default values were used, including max_features = sqrt(N_features) and criterion = gini.

5 Experimental Results

Classifier	Val Accuracy	Test Accuracy	Recall	Precision
kNN	63.65%	62.98%	0.19	0.61
Logistic	64.69%	63.16%	0.35	0.67
Neural Network	63.44%	61.19%	0.50	0.50
Random Forest	65.95%	63.61%	0.20	0.63

Figure 2. Summary of results for each of the four classifiers.

kNN Classifier

The kNN classifier performed the best with the feature combination **num_lab_procedures** and **number_visits,** a feature derived from the sum of **num_inpatient** and **num_outpatient** visits. As the value of k increases and the complexity of the model decreases, both the training and validation accuracy improve dramatically. This pattern was exhibited across nearly all feature pairs, and the validation accuracy approached 62%. This reflects the proportion of patients who were readmitted (P(y=1) = 0.62), meaning that the model behaves about as well as the majority classifier. Accuracy for various k values, **Figure 5**; a decision boundary, **Figure 6**; and confusion matrix, **Figure 7**; for this classifier can be found in **Appendix B**.

Logistic Classifier

The logistic regression model's performance remained relatively stable across different regularization strengths, with training accuracies ranging from 63.13% to 63.23% and testing accuracies from 63.40% to 63.65%. This suggests that the model's ability to generalize to unseen data is consistent regardless of the regularization strength applied. For the precision and recall metrics, the model exhibited higher precision for class 1 but lower recall, indicating that while the model is good at correctly identifying readmitted patients, it misses a significant number of them. Specifically, the precision for class 0 is around 0.62, and for class 1 it is 0.67. The recall for class 0 is high at 0.86, but for class 1, it drops to 0.35. The F1-scores reflect this imbalance, with 0.72 for class 0 and 0.46 for class 1. The validation accuracy was slightly higher at 64.69%, indicating a potential slight overfit on the validation set compared to the test set accuracy of 63.16%. The results suggest that while the logistic regression model is reasonably effective at identifying non-readmitted patients, it struggles more with accurately identifying readmitted patients. This could be due to an imbalance in the dataset or inherent difficulty in predicting readmissions. Accuracy for various C values **Figure 8**; for this classifier can be found in **Appendix B**.

Feedforward Neural Network

We found that the best hyperparameters for the feedforward neural network are as follows: { 'hidden_layer_sizes': 155, 'activation': 'relu', 'solver': 'sgd', 'alpha': 0.1, 'learning_rate_init': 0.001, 'max_iter': 145, 'n_iter_no_change': 100, 'batch_size': 190 }. This set of hyperparameters provided the highest accuracy with training accuracy being 63.99%, validation accuracy being 63.45%, and test

accuracy being 61.19%. For label 0 (not readmitted), the precision was 65% and the recall was 86%. For label 1 (readmitted), the precision was 59% and the recall was 31%. Accuracy as a function of training set size, **Figure 9**; for this classifier can be found in **Appendix B**.

Random Forest Classifier

The random forest model achieved the best validation accuracy with the parameters n_estimators=1000, max_depth=512, and min_samples_leaf=8. The validation accuracy was 65.95% and a test accuracy of 63.61%. The random forest model exhibited a tendency to overfit the training data, as indicated by its perfect training accuracy across all configurations. Despite this overfitting, the test accuracies remained relatively consistent around 60%, suggesting that the model was able to generalize reasonably well to unseen data. Feature importance analysis from the random forest indicated that **num_lab_procedures**, **num_medications**, and **time_in_hospital** are critical for predicting readmission, providing insights into potential areas for targeted interventions. The model struggled with recall for the readmitted class, indicating difficulty in correctly identifying patients who would be readmitted. Accuracy as a function of number of trees, **Figure 10**; feature importances table, **Figure 11**; and first three layers of the first decision tree, **Figure 12**; for this classifier can be found in **Appendix B**.

6 Insights

One of the most significant challenges was dealing with class imbalance, as the majority of patients were not readmitted. This imbalance led to models that tended to behave like majority classifiers, resulting in high precision but low recall for the readmitted class. This suggests that while models could identify non-readmitted patients accurately, they struggled to correctly identify those who would be readmitted.

Throughout our experiments, certain features consistently emerged as important predictors of readmission. For example, **num_lab_procedures**, **num_medications**, and **time_in_hospital** were highlighted by both kNN (manually during tuning) and random forest. This insight can guide healthcare providers in focusing on these areas to predict and potentially reduce readmission rates.

The kNN classifier struggled due to the curse of dimensionality, where the effectiveness of euclidean distance decreased with more features. It performed best with specific feature combinations but did not significantly outperform the majority classifier. The logistic regression model showed stable performance across different regularization strengths, with decent precision but poor recall for the readmitted class. This indicates it was better at identifying non-readmitted patients. The feedforward neural network had a moderate performance, with hyperparameters tuned to achieve the best accuracy. However, it also faced challenges with recall, similar to the logistic model. The random forest model performed well in terms of feature importance and validation accuracy, but it tended to overfit the training data. Despite this, its test accuracy suggested reasonable generalization.

The persistent issue across all models was the low recall for the readmitted class. This reflects the difficulty of predicting readmission due to the underrepresentation of positive cases. This could also be attributed to the complexity and variability in patient health conditions and hospital practices.

Dropping rows with missing values, particularly **A1Cresult** measurements, significantly reduced the dataset size and slightly altered the readmission rate. This highlights the importance of careful data preprocessing and the potential impact of missing data on model performance.

7 Contributions

Teresa: Concentrated on the kNN and random forest classifiers, evaluating feature combinations and tuning model parameters for optimal results.

Tiffany: Worked on developing and optimizing the feedforward neural network, exploring different regularization strengths and tree numbers.

Aashi: Focused on tuning the logistic regression and random forest classifiers, conducting hyperparameter optimization to improve their performance.

8 References

Du, Guodong, et al. “Joint Imbalanced Classification and Feature Selection For Hospital Readmissions.” Knowledge-Based Systems, Elsevier, 18 May 2020, www.sciencedirect.com/science/article/abs/pii/S095070512030318X?casa_token=pr3MQkQasQwAAAAA%3ASlA4AkGmzsT0_xCTpXdFz7bOzb5dRTT1uwVLKQ7dBWrmbcekHu6NIlHOgyOi9NZnC8oI_9abzQ.

Hephzibah Munnangi, MS and Dr . Goutam Chakraborty. “Predicting Readmission of Diabetic Patients using the high performance Support Vector Machine algorithm of SAS® Enterprise Miner™.” (2015).

Strack, Beata, et al. “Impact of HbA1c Measurement on Hospital Readmission Rates: Analysis of 70,000 Clinical Database Patient Records.” BioMed Research International, U.S. National Library of Medicine, 3 Apr. 2014, onlinelibrary.wiley.com/doi/10.1155/2014/781670.

9 Appendix

Appendix A

Summary Statistics for Numeric Features								
VARIABLE	**MEAN**	**STD. DEV.**	**MIN.**	**25%**	**50%**	**75%**	**MAX.**	**MED.**
A1Cresult	2.19447	0.85976	1	1	2	3	3	2
time_in_hospital	4.73126	3.06945	1	2	4	6	14	4
num_lab_procedures	54.26672	16.67474	1	44	54	65	132	54
num_medications	15.98949	8.67717	1	10	14	20	75	14
number_outpatient	0.21051	0.94627	0	0	0	0	36	0
number_emergency	0.09763	0.45744	0	0	0	0	13	0
number_inpatient	0.12573	0.51541	0	0	0	0	8	0
number_diagnoses	7.18824	2.09593	1	5	8	9	16	8
readmitted	0.38451	0.48650	0	0	0	1	1	0

Figure 3. Summary statistics for numeric features.

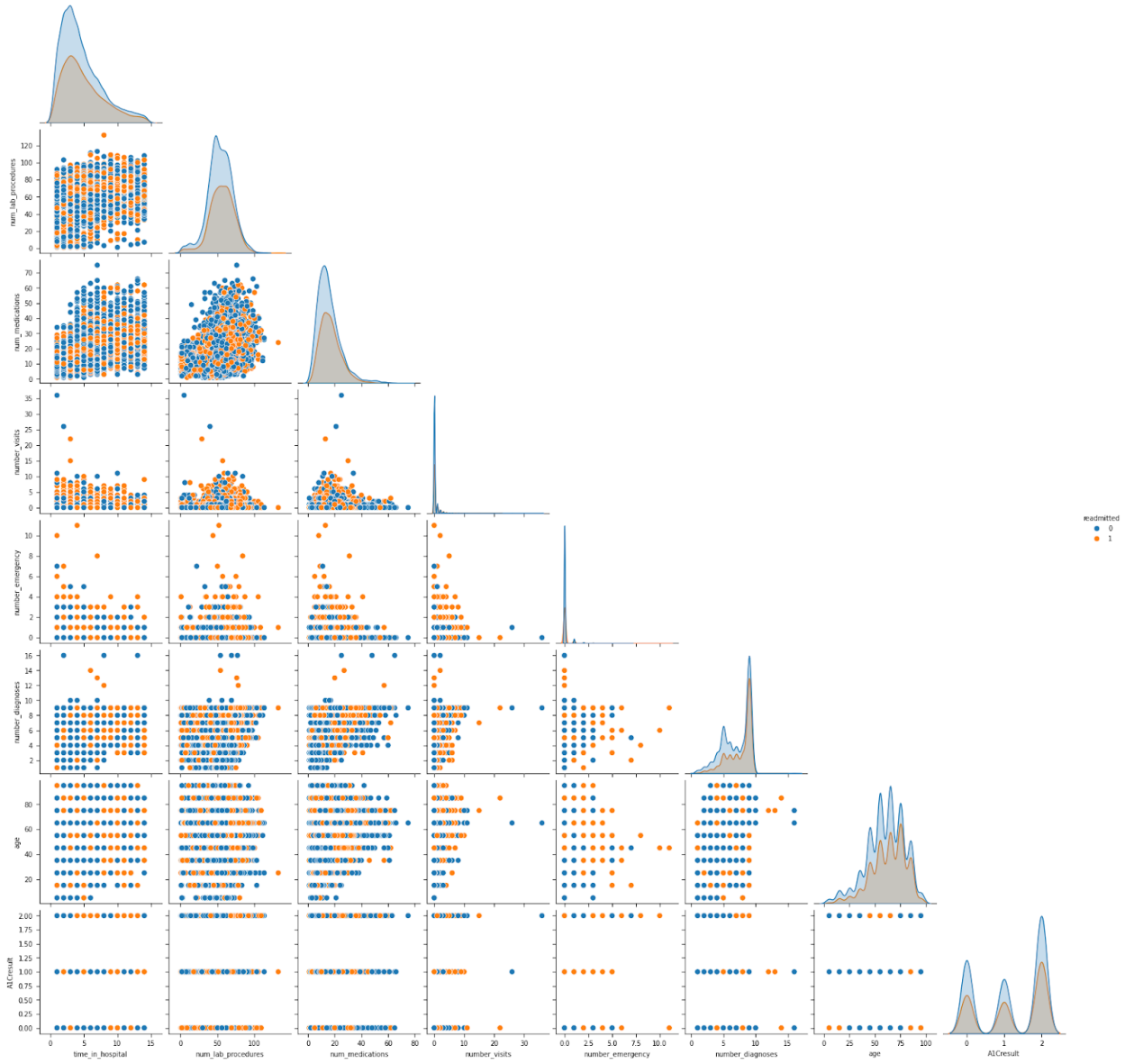

Figure 4. Pair plots for numeric features.

Appendix B

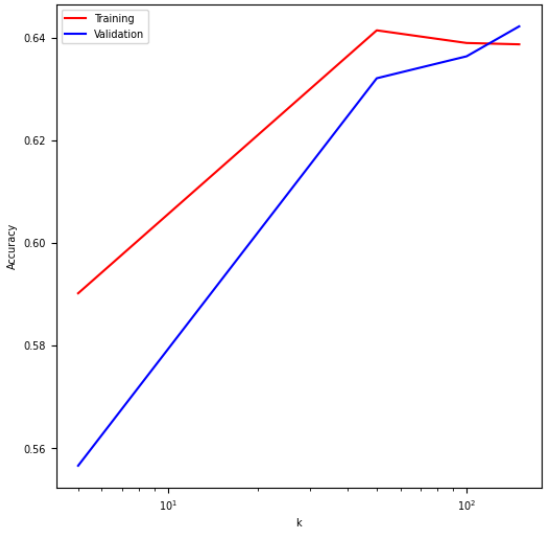

Figure 5. Accuracy as a function of k for num_lab_procedures and number_visits.

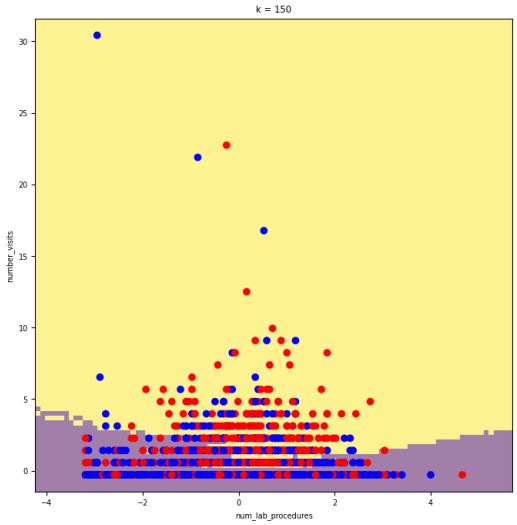

Figure 6. Decision boundary for final kNN classifier.

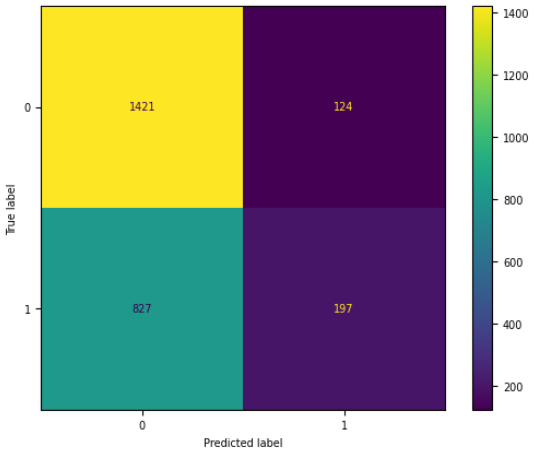

Figure 7. Accuracy as a function of k for num_lab_procedures and number_visits.

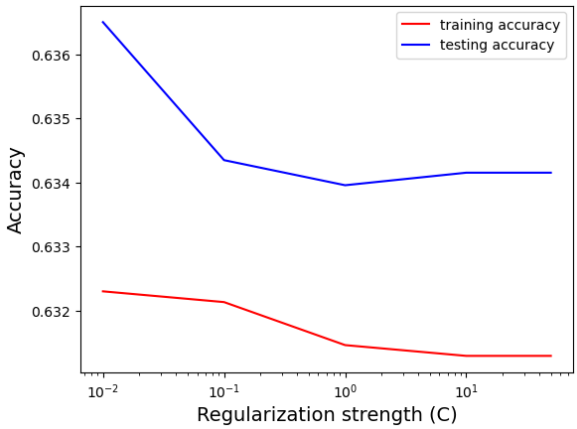

Figure 8. Accuracy for various C values for the logistic classifier.

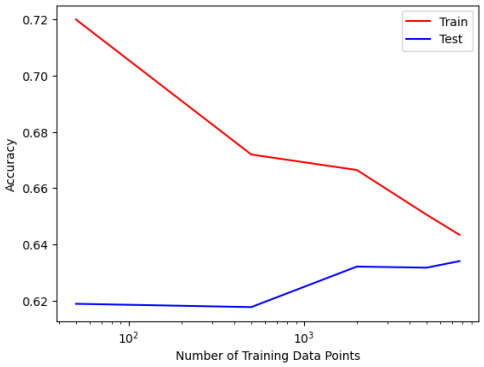

Figure 9. Accuracy as a function of training set size for the feedforward neural network.

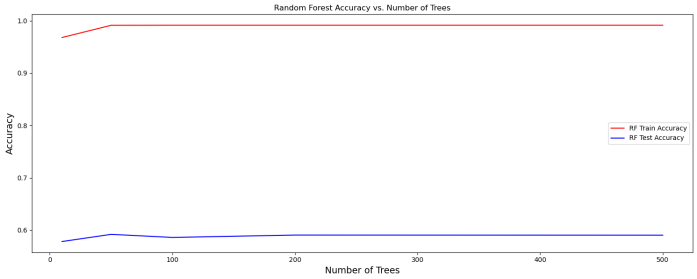

Figure 10. The number of trees used in random forest does not significantly affect the accuracy of the model for larger numbers.

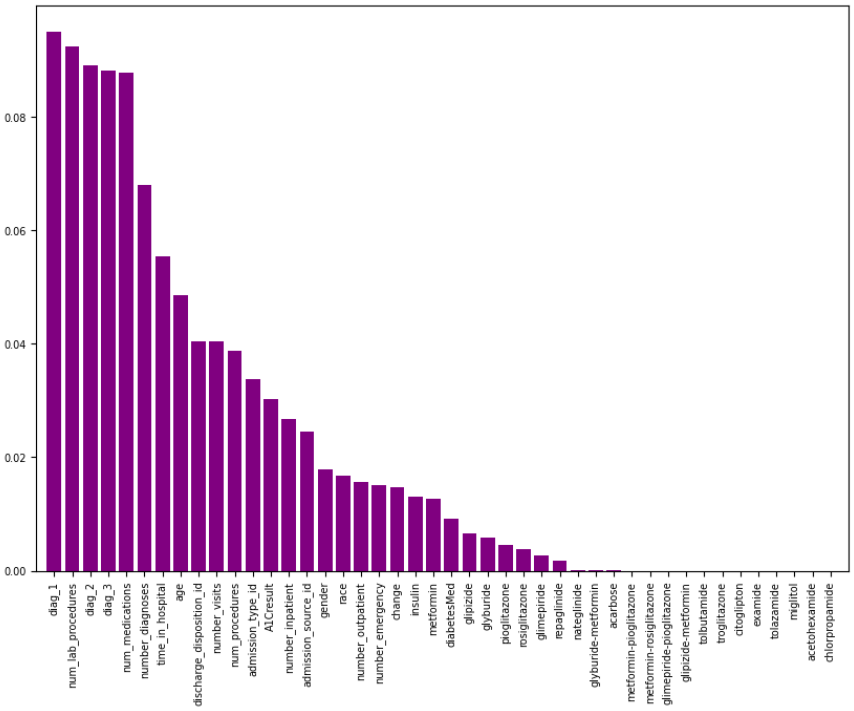

Figure 11. Feature importances determined by the final random forest classifier.

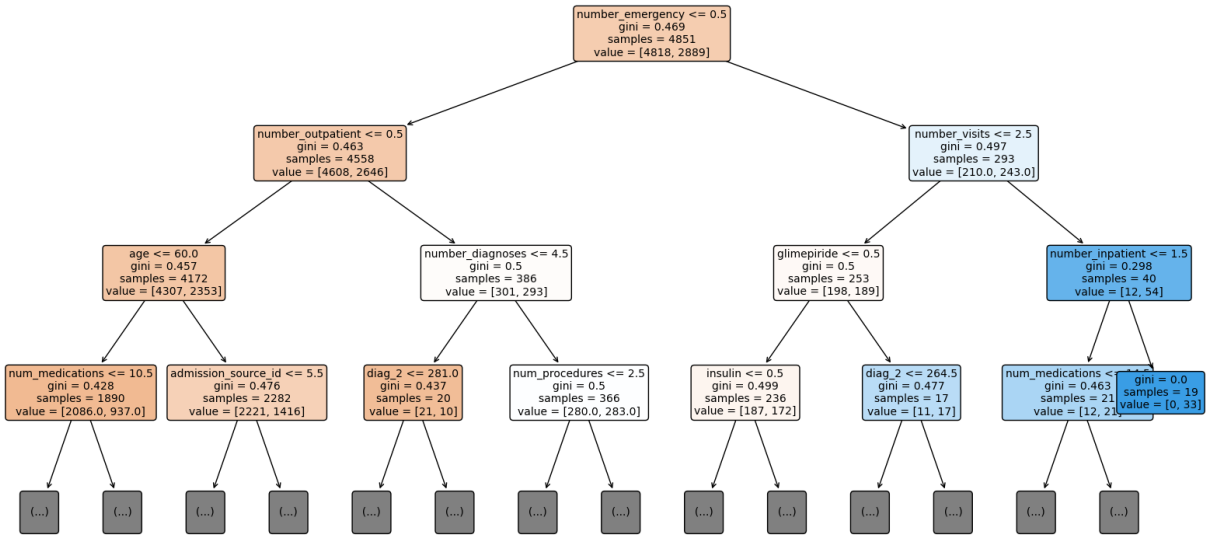

Figure 12. First three layers of the first decision tree used in the final random forest classifier.