**Preprocessing and Predictive Modeling of Diabetes Data**

Corey St Jean

Speed School of Engineering, University of Louisville

CSE-632: Data Mining

Dr. Mehmed Kantardzic

September 26, 2024

**Abstract:**

Diabetes is a condition impacting more individuals around the world each year. The true causes of most diabetes cases are still unknown, negatively impacting the ability to fully treat the disease. The explosion of the data science field has revealed never seen results in the field of diabetes when it comes to potential causes of the disease and how to best identify symptoms early. To explore how data science can be utilized for diabetes, a study was conducted in which three different data classification models were produced able to somewhat accurately classify individuals based on responses to a survey. The three models implemented include logistic regression, decision tree, and random forest with random forest obtaining the best results, followed by the decision tree. The logistic regression had errors that negatively impacted the results obtained from regression, meaning results from said model are obsolete.

**Problem Description:**

Diabetes is a chronic disease typically involving the pancreas in which the body is not able to effectively regulate blood sugar levels. Insulin, an enzyme created and regulated by the pancreas, is produced to lower blood sugar levels, so those with diabetes typically experience periods of hyperglycemia or high blood sugar. The total number of individuals with diabetes has increased drastically within the last few decades, positively correlated with a rise in obesity in first world countries, making new and more effective treatments as well as early detection critical in treatment. The reason early detection is so important is because as the disease progresses, nutrients are ineffectively moved to organs and tissues, which can cause permanent damage. Some common examples of damage caused by diabetes include blindness, loss of distal appendages, as well as loss of function of organs due to lack of nutrients. In some more extreme cases, untreated diabetes can also lead to coma or death.

Diabetes typically occurs because of two different reasons, and the cause of the diabetes determines the type of diabetes. Type I diabetes typically occurs during childhood and has unknown causes, but it occurs much more frequently in wealthy countries, potentially due to the ability to afford medical care or testing (WHO, 2023). Type II diabetes on the other hand has known causes, typically being attributed to genetics, obesity, lack of exercise, or a mixture of the three. Type II diabetes often is preventable, but is the much more common form, making up over 95% of diabetes cases (WHO, 2023).

Testing for diabetes is simple and inexpensive but requires high tech machines able to process and quantify blood samples often only available in first world countries and high population centers. To test, a blood sample is drawn, and the blood glucose levels are analyzed, and based on the levels of blood glucose (typically much lower or higher than average) the patient can be diagnosed with diabetes. Treatment of diabetes is much more complicated, and depending on the severity and type can be lifelong. For type I, treatment typically involves daily insulin injections and constant monitoring of blood glucose levels, and there are no known cures for Type I. Type II treatment can include insulin injections, but these are not always required and can be much less frequent. Treatment also includes prescription medicines that can manage blood glucose levels like metformin, sulfonylureas, and SGLT-2 inhibitors (WHO, 2023). Type II also does not have a known cure, but symptoms and treatments can lessen with weight loss, exercise, and diet management.

For this project, we are given three different files to start with, “train.csv”, “test.csv”, and “notes.txt”. The “Train” file contains over ten thousand samples, with the diabetes column already present. This dataset will be used to create classification models that are able to predict the value of the diabetes attribute accurately. The “Test” file contains the sample attributes as “Train”, except for the diabetes attribute. We are to use the classification models on this dataset to accurately predict if individuals have diabetes or not. “Notes” simply contains some information on various attributes and what their values mean for clarification. The entire process for creation of classification models involves first preprocessing the dataset then creating three different models from three different techniques and finally comparing accuracy of results obtained between techniques to figure out which technique performs best on this specific dataset. Finally, the model that performed best on the training data will be used on the testing data to see if the predictive model is accurate.

**Review of Publication:**

With the basics of the project as well as the diabetes disease understood, we can now examine the cross section between data mining and diabetes predictions or classifications. To gain some insight into current applications and approaches of using data mining to categorize and predict diabetes, three different academic articles are to be reviewed. To start, “Detection and Prediction of Diabetes Using Data Mining: A Comprehensive Review” discusses various techniques and their effectiveness when it comes to predicting diabetes and other related conditions (hypo/hyperglycemia) (Kahn et al., 2021). Firstly, data was collected from the National Health Insurance Research Database (NHIRD) from Taiwan, containing records for over 20 million individuals. This paper focuses on the different classification and predictions techniques possible within the data mining practice and breaks down their effectiveness when it comes to medical predictions. The techniques specifically covered are classification, hybrid, association, regression, sequential pattern mining, and clustering models and their effectiveness when it comes to predicting diabetes specifically (Kahn et al., 2021). Based on the analysis performed, Artificial Neural Networks showed the most promise when it comes to accurate and sensitive predictions of hypo/hyperglycemia and diabetes, but plenty of other techniques including regressions, decision trees, and random forests showed fairly accurate results for the effort they required (Kahn et al., 2021). Then, another study was conducted outlining the most effective process for data mining medical diagnoses. This study focused on outlining an effective plan from the data used, how the data should be prepared, to how to best choose or modify techniques for accurate classification and prediction. For the best results, preprocessing of the data is necessary, with some required steps including data selection, data cleaning, feature selection, dimensionality reduction, normalization, data transformation, and data integration. When it comes to prediction or classification techniques, hybrid models showed the best accuracy, meaning combinations of multiple models will provide the best results (Kahn et al., 2021).

The second research article explored, “Data-Mining Technologies for Diabetes: A Systematic Review” focused on the current applications of data mining for diabetes being used in research (Marinov et al., 2011). Specifically, seventeen research articles were gathered to perform a sort of survey on data mining techniques and their effectiveness. Based on the articles studied, data mining showed successful results in blood glucose prediction, identification of risk factors, genome analysis, adverse drug effect analysis, insurance fraud detection, and improved clinical guidelines for diabetes (Marinov et al., 2011). Data mining was found to be successful within the research of diabetes, with some major impacts being earlier detection of hyper/hypoglycemia allowing for diabetes prevention rather than treatment. Feature selection was also studied to gain insights into specific attributes that can provide important insights into diabetes and how the disease is impacting a specific individual. The features needed depends on the specific results or prediction to be obtained as well as the techniques applied. Blood glucose control’s feature selection includes age, diagnosis duration, insulin need, and blood glucose measurements for accurate predictions of effectiveness of blood glucose control (Marinov et al., 2011). When it comes to genome analysis, clustering showed great results to accurate classify genes, but did not provide great insight into what specifically made genes similar. Preprocessing and cleaning of data before predictive modeling was present in most studies, and often was associated with better results (Marinov et al., 2011).

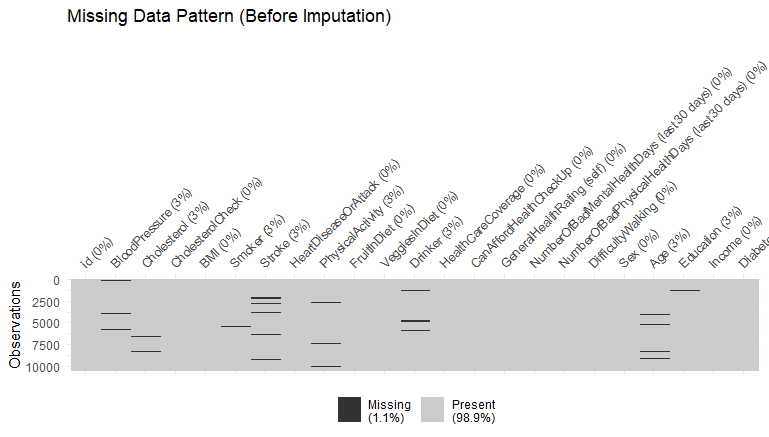
Finally, “Predicting Diabetes Mellitus using Data Mining Techniques” focuses on measuring the accuracy of five common and effective data mining techniques to measure their accuracy as well as processing time (Kavakiotis et al., 2017). The five data mining algorithms studied includes Naïve Bayes, logistic regression, C5.0 decision trees, support vector machines (SVM) and artificial neural networks (ANN) (Kavakiotis et al., 2017). Based on accuracy measures, logistic regression had the highest accuracy at 74.67%, followed closely by C5.0 decision tress at 74.63%. The next three in order were Naïve Bayes at 73.57%, ANN at 72.29%, and SVM at 72.17% (Kavakiotis et al., 2017). But all techniques were relatively similar with an accuracy from 72% to 75%. Processing time had some dramatic differences though with C5.0 and SVM both being under two minutes, Naïve Bayes and logistic regression under three minutes, and ANN at four and a half minutes (Kavakiotis et al., 2017). These results, while helpful, are not true across the board and mainly depend on the data being processed and the overall goals of the classifications and predictions, which is noted. Logistic regression is a supervised method for calculating the probability of certain attributes based on the other attributes. This technique focuses on calculating the probability of a successful classification over the probability of a failed classification.

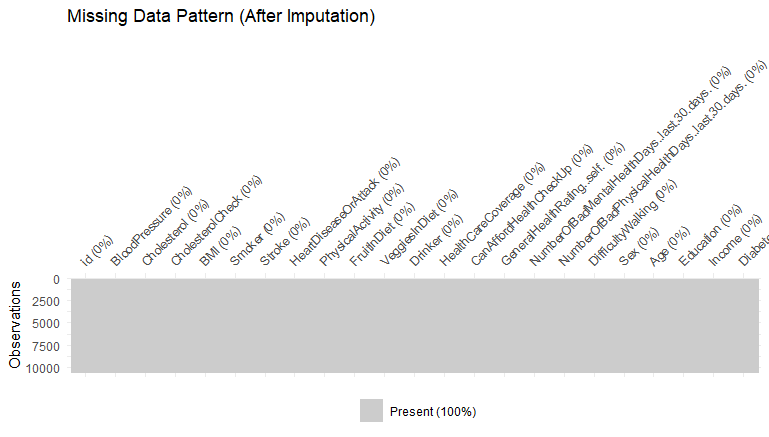
**Software tools:**

The main and only tool used is the R programming language. R is a great statistical and data processing language which plenty of machine learning libraries added by other users (Robinson & Burns, 2024). These libraries essentially hold functions that perform various data mining tasks in a simple to use and understand method. The R libraries used include Tidyverse, Naniar, gridExtra, caret, ggplot2, corrplot, rpart.plot, randomForest, and pROC. Tidyverse is a general library including lots of data science functions and some common visualization libraries (ggplot2). Naniar is a library with functions focused on imputing and describing missing values in a dataset. Caret is the library used for logistic regressions, specifically with functions for training and plotting regression models. randomForest adds the random forest algorithm to R and rpart.plot allows for creation and visualization of decision trees. The remaining libraries, gridExtra, ggplot2, and corrplot, are all used for graphing visualizations and improving legibility of said visualizations with ggplot2 being included in the TidyVerse package.

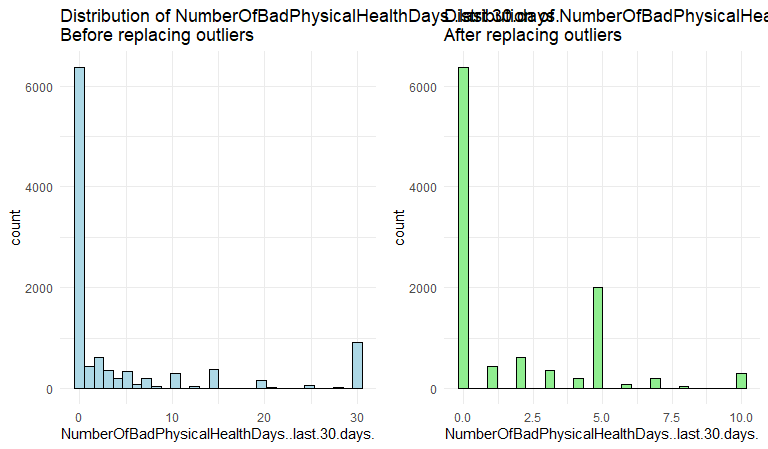
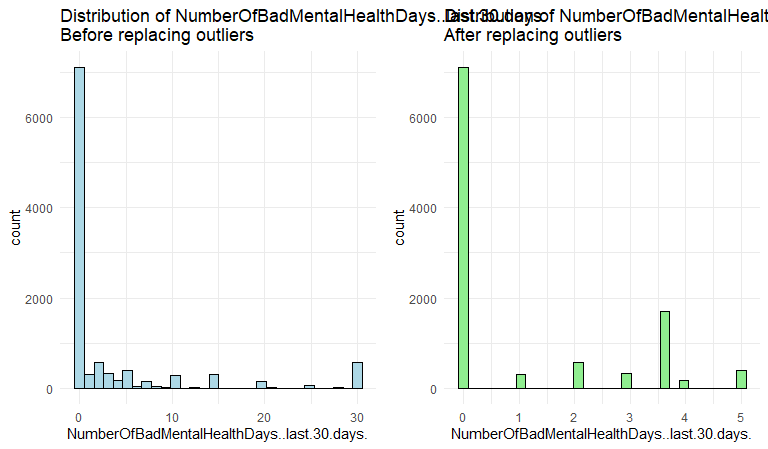
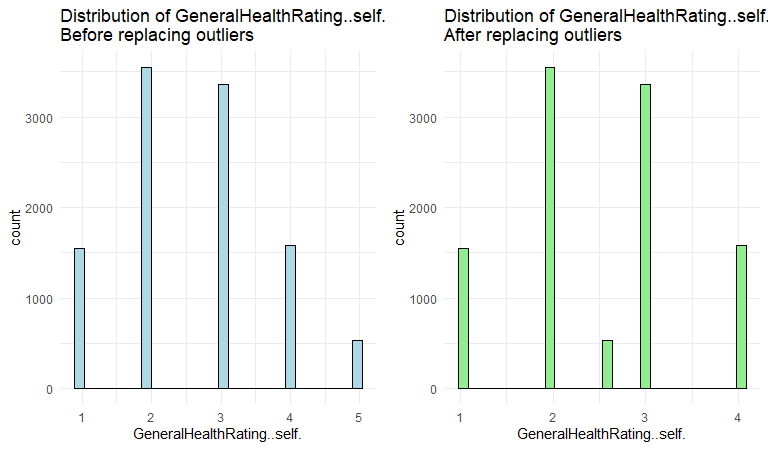
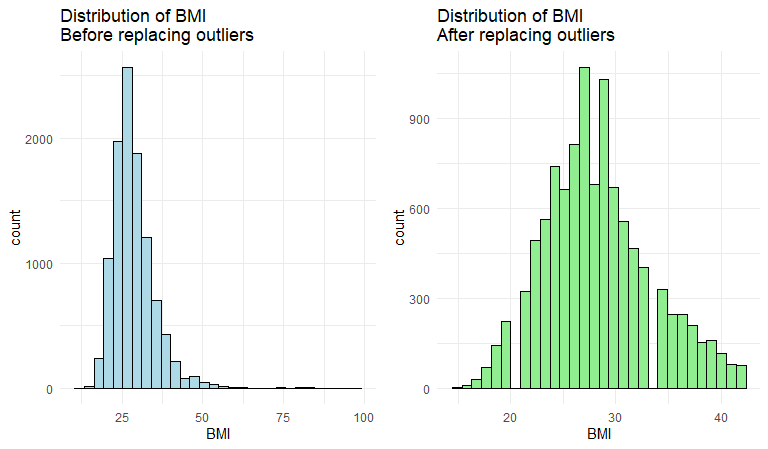
**Preprocessing:**

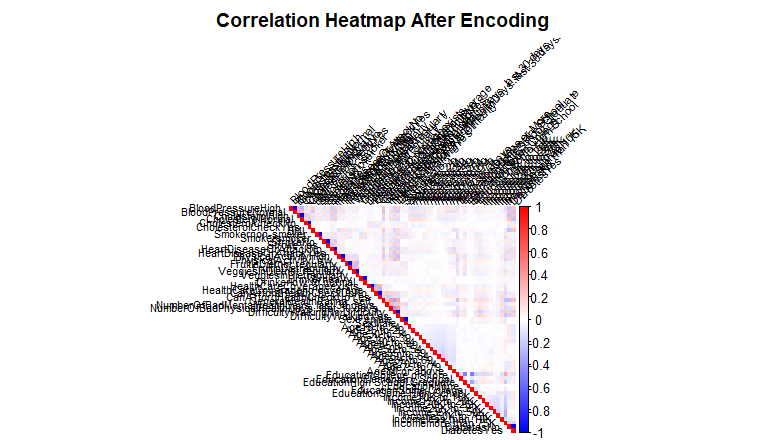
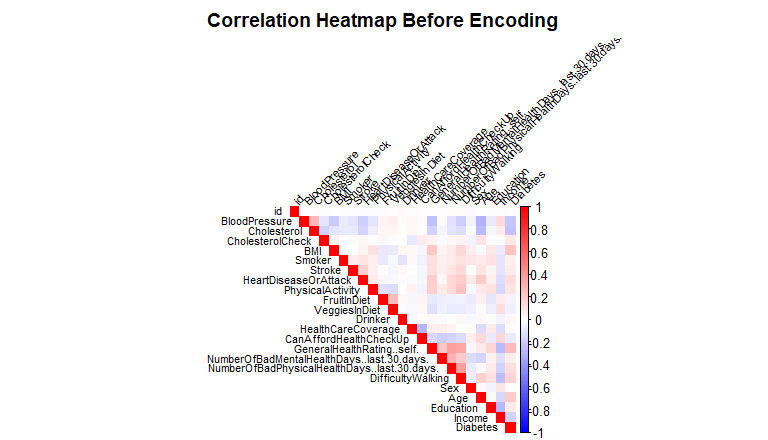
To start the data mining process, we first must load the dataset and then we can also save the diabetes column as its own unique dataset that can be referenced later. From here, missing values can now be detected and replaced with imputed values. For imputation, the mode will be used because it provides us with applicable values that are most common throughout the data. This essentially makes it so values we chose to fill will in no way be an outlier or be impacted by outliers throughout the data. This is important because outliers have yet to be dealt with, so using the mean or median may skew the results of the models. To calculate the mode, a simple function was created that finds all unique values for that feature and then adds up how many of each unique value is present in the dataset. From here, an ‘if … else’ statement is applied to each sample to check if the value is missing or not, if said value is missing, then it is replaced with the calculated mode from the function, else it is simply replaced with itself. To show how the data was impacted and that missing values were successfully filled, a visualization method for missing values was used from the Naniar library, this visualization shows how many values are missing for the entire dataset and well as each specific feature.

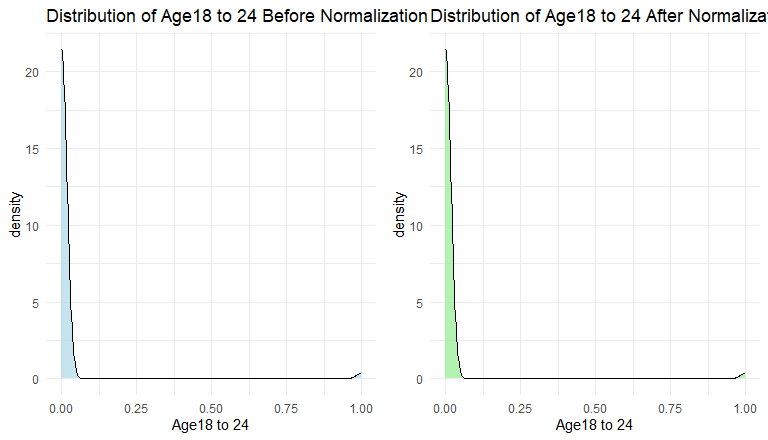
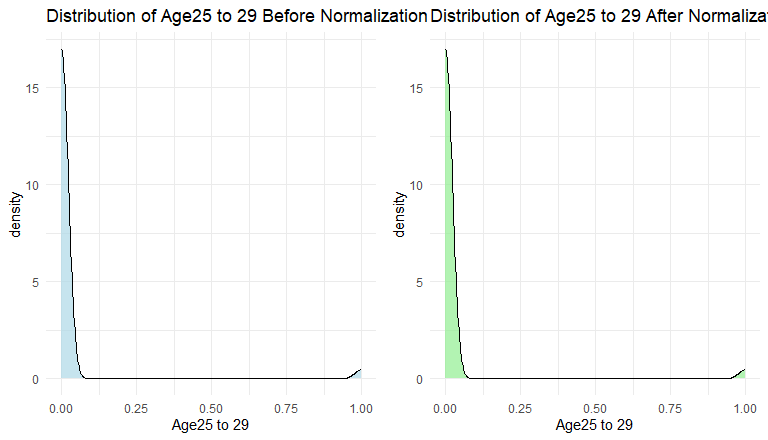
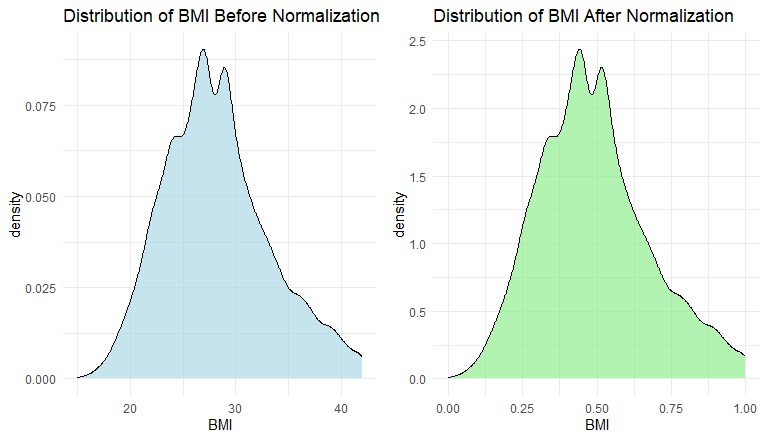
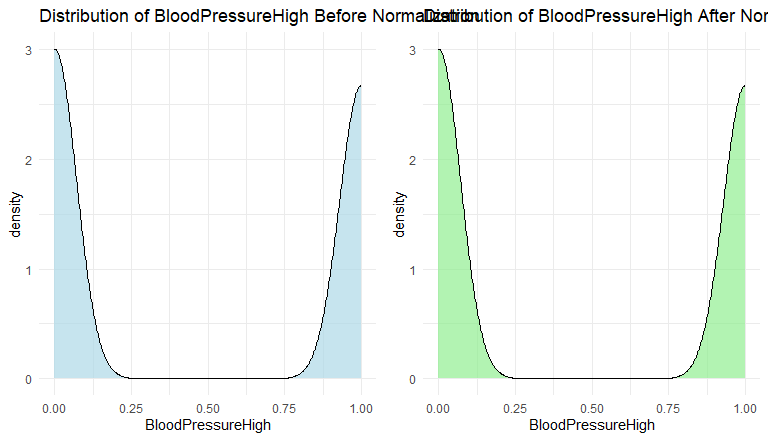




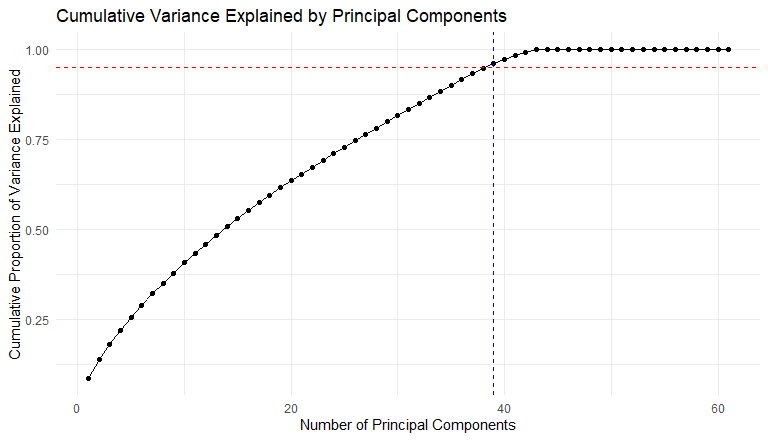
With missing data now filled, we can begin to find outliers and figure out how to best deal with said outliers. We also must figure out how categorical and numeric attributes are dealt with because the data type of the attribute determines how outliers are detected and dealt with. To detect categorical outliers, a function was created to calculate frequencies of categorical values within each feature, then a threshold frequency value of .01 was implemented. This essentially means that if a categorical value makes up one percent or less of the samples, then it can be considered an outlier. This function was then applied via a for loop over the dataset, and based on the results obtained, only one categorical outlier was detected within the Education feature. Individuals with zero education are considered outliers, but we cannot remove these individuals outright or replace them with another categorical value from that feature because this will ignore potential correlations for individuals with no education. This would mean our models would not be applicable for uneducated individuals, limiting the success of our models. Instead, this outlier is simply noted and ignored as to not negatively impact results. For the numeric attributes, we apply the Inter-Quartile Range (IQR) technique to detect outliers. IQR was chosen because it does great at detecting outliers regardless of any skew or the extreme outliers. IQR simply groups the data based on standard deviations and mean calculations into two categories, outliers and non-outliers. Then, the outliers detected from the IQR technique are replaced by the calculated mean values of the attributes. Mean is used in this case because we calculate it based on the non-outliers and ignore any values detected as an outlier in the dataset. To show the success of outlier detection and removal, the distribution of the four numeric attributes is shown before and after outlier replacement in the bar graphs below.

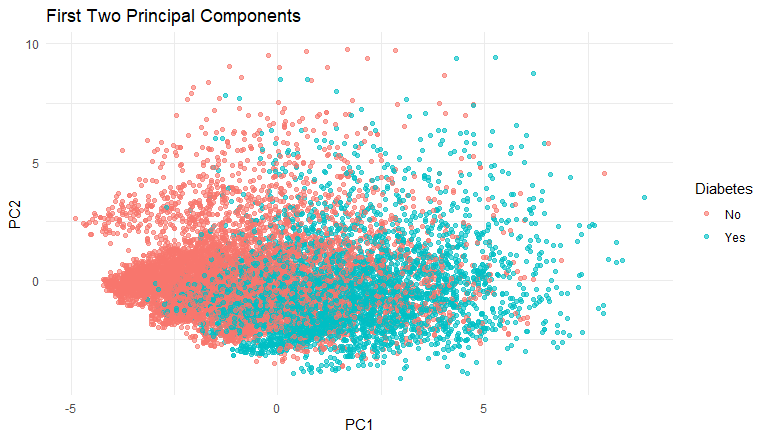


Following missing data and outlier replacement, we can begin to focus on methods of transforming the data so that models can be best applied to the datasets. To start, we cannot use categorical values for regression as they require numeric values, so we must encode our categorical values into a numeric value. To encode values, one-hot encoding was implemented. One-hot encoding essentially takes every possible discrete value from a categorical attribute and breaks it apart into different attributes based on each individual category. For example, if we had an attribute titled “Happy” in which possible values are “Yes” and “No” we can break this attribute into two attributes: “Happy Yes” and “Happy No” with unhappy samples having a one in the “Happy No” feature and a zero in the “Happy Yes” feature. To implement one-hot encoding, the id column was first removed as to not encode it, then all categorical columns are gathered, and their individual categories are saved. To apply encoding, a dummy model was created that simply takes the possible values for each category and adds them to the dataset while also removing the original category because we have its information as encoded features. To show the difference before and after encoding (mostly showing that encoding does not skew the data negatively and expands the number of features) correlation matrices are produced before and after encoding. Based on the difference, it is easy to see that our categorical features are now encoded as numeric features, but also it is clear that feature split up are inversely correlated proving that encoding worked correctly as well.

 With encoding completed, we can now normalize the database to help scale the data to a normal range and reduce the impact of any remaining or undetected outliers in the dataset. There are various types of normalization, but because one-hot encoding was used, in which out categorical values now have values of either zero or one, min-max normalization within the range [0-1] is used. This will essentially let us skip normalization on encoded values because they are already min-max normalized. To normalize in R, a function was created that simply applied the min-max normalization function to all samples and features in the dataset and replaced said samples with their normalized value. To show the effects of normalization, a few representative features’ distributions are plotted below, showing that distributions and scale remains the same, but the range is now between zero and one, proving successful normalization.

Our final step involves reducing the dimensions of the dataset so that our models can perform better and run faster, but we also need to ensure that removed dimensions have minimal impact on the overall variance of the dataset. To ensure that variance remains almost the same, principal component analysis (PCA) was chosen as the method to find the important features (principal components) and remove all unimportant features (non-principle components). PCA was implemented in R through the ‘prcomp’ function in which we ensure our PCA is based on the center of each data feature. We chose 95% variance as our target goal for our PCA, meaning we want to find the smallest number of features to include in a dataset that retains at least 95% of the variance of the dataset prior to dimensionality reduction. Included as visualization for this section is the cumulative variance line graph showing maximum number of principle components for 95% variance explained. Additionally, the correlation matrix before and after dimensionality reduction is shown to visualize which elements were removed and how the remaining elements relate to each other. Completed dimensionality reduction also completed our data preprocessing meaning we can begin to extract information and trends based on our preprocessing techniques and visualizations and then move to data mining with predictive modeling.



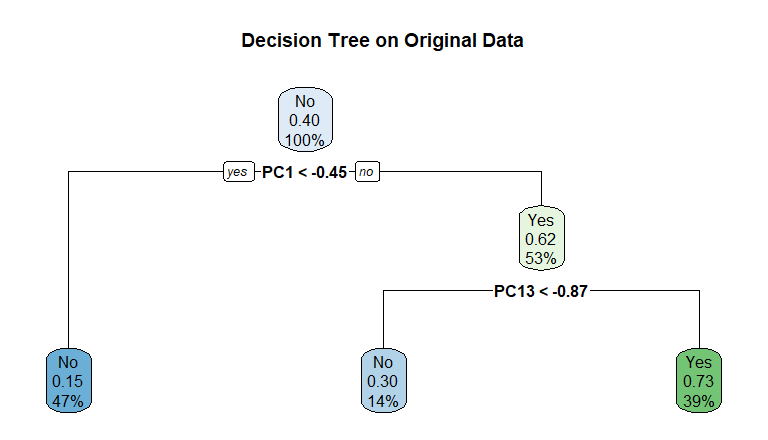


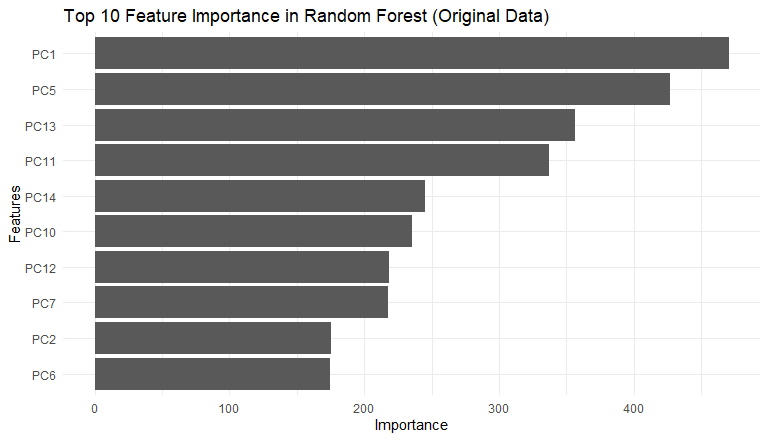
**Data Mining:**

After preprocessing the data, we can now begin classification techniques for data mining. These techniques will use the information preprocessed as well as the already filled out ‘Diabetes’ column to connect certain attributes and their values with diabetes results. This technique will essentially let our algorithms learn how data is connected to the diabetes column, allowing us to use these learnings on unlabeled data which will let us accurately classify users. The first data mining technique is logistic regression, implemented through the ‘caret’ package in R. This package allows us to train models, and then apply said models after learning on unlabeled data. First, a function was defined in which input is the data after preprocessing and the specific technique to be applied. This function then trains the model using the technique specified on the data, after which a summary of the model is printed. To calculate the F-score of the model, another function is defined in which 10-fold cross validation is applied to ensure that F-scores are accurate regardless of what data is selected as training or testing. Then once ten f-scores are calculated they are averaged into our final f-score for the model. This function essentially trains ten models, tests their accuracy, and then calculates the f-score by first creating a confusion matrix and extracting the precision (number of positive prediction values) and the recall (sensitivity) from said confusion matrix. These values are necessary for f-score calculation in which the formula is: f-score = [2 \* (precision \* recall)]/(precision \* recall).

Logistic regression is a data mining and statistics technique in which probabilities are calculated for specific attributes we plan on predicted, and then a Logit function is applied to said probabilities to compare success and failure rates. Based on said logit function, we can determine if an individual should or should not have diabetes simply based on calculated probabilities. For logistic regression, we can see which factors are most important based on their p-value. P-values greater than .5 are considered important leaving us with around 39 principal components (important components for logistic regression).

The next data mining technique applied was a decision tree. The decision tree algorithm takes attributes in a dataset and determines which attributes give us the most gain in reaching a final decision. This attribute is then chosen, and a cutoff point is determined, which is typically an inequality expression. This continues until we only have leaf nodes which are nodes that classify the sample (typically true/false). This technique was applied through the functions discussed in the first section of the ‘Data Mining’ chapter and the final decision tree created is included in the ‘results’ chapter. Based on said decision tree, we could easily see that PC1 is the gives the most gain and PC13 gives the second most. These two variables are the only necessary ones for a decently accurate decision tree that is easy to interpret for humans. This technique also completed first and took less than a minute.



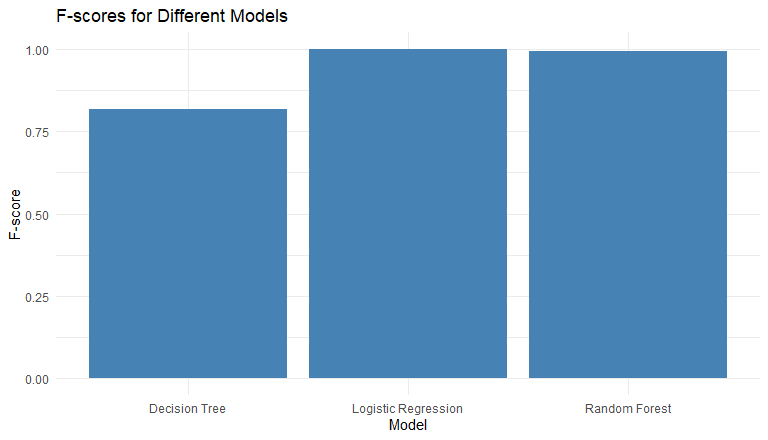
The final data mining technique applied was random forest in which a collection of decision trees is generated, all randomly created but all leading to truthful results. These trees are then collectively used to make predictions based on outputs from all decision trees within said collection. This technique allows for better results to be obtained than a single decision tree while also taking much more time and processing power. This technique took around an hour to complete. The number of trees to be created is defined in the randomForest package with a default value of 500 meaning 500 trees will be generated and will vote on results.

**Results:**

For logistic regression, the calculated f-score was 1.00 which essentially means that our model had 100% accuracy for all 10 folds. This score though does not seem accurate, mostly because over 50 warnings were included as output all stating that attributes had a p-value of 1 meaning that single value should be able to predict the diabetes column. This is very easy to disprove from the correlation matrices. So, while these results on the surface appear great, they are most likely due to an error in the model. While results are obtained from this model on the testing data, when compared to another model (random forest) with a high f-score, results were inconsistent. This means we cannot believe the results obtained from this model, meaning we will select the next best model for testing the data. The output of training the model was the trained model as well as basic information about the model including z-scores, p-values, and which values are most important for the regression.

The decision tree had an f-score of .8166631 essentially meaning it is accurate for around 82% of cases. This means that this model is fairly accurate and able to effectively classify most cases, but there are still around 18% of cases that will be misidentified (almost 1/5th!). This model took the shortest time to train, taking a couple minutes for the final output to be acquired. Output for the decision tree consists of the tree with specific attributes making decisions. One thing that impressed me about the decision tree was how simple it was while still being mostly accurate. Only two attributes were needed to effectively classify 82% of cases, meaning our decision tree has only two levels of decision with the third level being only leaves. This tree also specifies the threshold for each attribute determined to be the most important, but this threshold only applies after PCA is completed, making our tree a bit harder to understand. The two main attributes were PC1 and PC13 which represent High Blood Pressure and History of Heart Attacks/Conditions respectively. This means that individuals with a blood pressure over a threshold and BMI over another threshold most likely have diabetes. Thus far, this is the best results obtained, but the random forest expanded on these results and improved them greatly.

Random forest had a great f-score of .9915691 meaning it is accurate for around 99% of cases, which is, in practice, one of the best f-score obtainable. That means that in an extreme minority of cases this model will inaccurately classify diabetes. This model, by far, took the longest to train at around 45 minutes, but the results show that this processing and computation time is well worth it. This model as output includes the importance of each attribute, the kappa values for the model. The three features with the highest importance are PC1, PC5, and PC13 with PC1 and PC13 being high blood pressure and heart health as noted previously and PC5 represents BMI. The best kappa value obtained was .976 which describes the percentage of agreement between each of the decision trees generated in the forest. This, by far, was the best model meaning it will be the model used to classify the training data.



**Conclusions:**

Based on results obtained from the models, random forest and decision tree classification models were successfully created, while the logistic regression model had some difficulties or errors that impacted performance. The logistic regression application will be further explored to see where difficulties are arising. One potential source of errors in the regression model could be due to the encoding of categorical values. While they were successful encoding into numeric values, these values were simple (1’s and 0’s) which could negatively impact results obtained from said categorical values. To test this, one could remove all categorical values and see if the model has less errors and shows a more accurate and consistent result.

**Appendix:**

Kavakiotis, I., Tsave, O., Salifoglou, A., Maglaveras, N., Vlahavas, I., & Chouvarda, I. (2017). Machine learning and data mining methods in diabetes research. Computational and structural biotechnology journal, 15, 104-116.

Khan, F. A., Zeb, K., Al-Rakhami, M., Derhab, A., & Bukhari, S. A. C. (2021). Detection and prediction of diabetes using data mining: a comprehensive review. IEEE Access, 9, 43711-43735.

Marinov, M., Mosa, A. S. M., Yoo, I., & Boren, S. A. (2011). Data-mining technologies for diabetes: a systematic review. Journal of diabetes science and technology, 5(6), 1549-1556.

Robinson, S., & Burns, E. (2024, May 31). What is the R programming language?: Definition from TechTarget. TechTarget. https://www.techtarget.com/searchbusinessanalytics/definition/R-programming-language

World Health Organization. (2023, April 5). Diabetes. World Health Organization. https://www.who.int/news-room/fact-sheets/detail/diabetes

**Functions:**

[1] get\_mode(x):

get\_mode <- function(x) {

ux <- unique(x)

ux[which.max(tabulate(match(x, ux)))]

}

[2] check\_categorical\_outliers(x, threshold)

check\_categorical\_outliers <- function(x, threshold = 0.01) {

freq <- table(x) / length(x)

rare\_categories <- names(freq[freq < threshold])

if (length(rare\_categories) > 0) {

cat("Rare categories found:", paste(rare\_categories, collapse = ", "), "\n")

} else {

cat("No rare categories found.\n")

}

}

[3] detect\_outliers(x):

detect\_outliers\_iqr <- function(x) {

q1 <- quantile(x, 0.25, na.rm = TRUE)

q3 <- quantile(x, 0.75, na.rm = TRUE)

iqr <- q3 - q1

lower\_bound <- q1 - 1.5 \* iqr

upper\_bound <- q3 + 1.5 \* iqr

return(x < lower\_bound | x > upper\_bound)

}

[4] normalize(x):

normalize <- function(x) {

return((x - min(x)) / (max(x) - min(x)))

}