Andrew Wolfe, Tej Seth, Ethan Goldstein
SI 370

Predicting Whether a Patient Will Have Diabetes or Not

I. Motivation

The overarching theme of this project was to be able to predict whether or not a patient has diabetes based on certain measurements that were included in the dataset. The first goal was a personal group goal as we wanted to grow our skills working with outcomes that have binary outcomes through data cleaning, feature engineering, and modeling techniques. The second goal was to build an accurate model on this dataset that included women older than 21 years old that are part of the Native American Pima People group that could eventually be applied to other groups with a similar technique. Since this was a Kaggle dataset there were already a lot of notebooks with high accuracy rates but we wanted to see if we could use our skills to find an edge that could help people working in biostatistics and/or healthcare analytics with these types of projects in the future.

II. Data Sources

diabetes.csv: The dataset consists of several medical predictor variables and one response variable. The predictor variables are all continuous numeric and are “Pregnancies”, “Glucose”, “BloodPressure”, “SkinThickness”, “Insulin”, “BMI”, “DiabetesPedigree” and “Age” while the response variable is a binary “Outcome” type. Each variable is described in detail below:

- **Pregnancies**: The number of times the patient got pregnant
- **Glucose:** Using an oral glucose tolerance test, this is the plasma glucose concentration in a 2 hour period
- **BloodPressure:** The diastolic blood pressure measured in mmHg
- **SkinThickness:** The skin fold thickness of each patient’s triceps measured in mm
- **Insulin:** The 2-hour serum insulin level of a patient measure in muU/ml
- **BMI:** A person's weight in kilograms divided by the their height in meters squared
- **DiabetesPedigree:** The score of how likely a patient is to get diabetes based on family history
- **Age:** Age in years
- **Outcome:** An indicator for if the patient has diabetes or not

III. Data Manipulation

When looking at the code section, the first thing that was needed was to load the necessary packages. “Sklearn”, ‘statsmodels’, ‘catboost’, ‘numpy’, ‘matplotlib’, ‘scipy’, and ‘xgboost’ packages needed to be imported and loaded so that the proper data manipulation, analysis, and modeling could be accomplished.

The next thing that we had to do was drop the columns ‘BloodPressure’ and ‘Skin Thickness’. This was necessary because after looking at the output of our correlation heatmap, we noticed that these two factors did not seem to have a large impact on if a person had diabetes or not.

After that, we looked to create a few additional columns through feature engineering that would increase the accuracy of our model. When creating these columns, our main objective was to find variables with clear correlations to whether or not someone has diabetes, and combine them in a manner that would result in an increased predictive validity for the new variable. To start this process, we first created a column called ‘Glucose_x_BMI’, equaling the ‘Glucose’ column multiplied by the ‘BMI’ column. This is because both ‘Glucose’ and ‘BMI’ had relatively higher correlation values (0.47 and 0.29, respectively) and when we combined them, the new ‘Glucose_x_BMI’ variable had a correlation of 0.49 with a person’s diabetes outcome. More importantly, this new variable helped to increase the accuracy of our model (particularly in the XGBoost classifier) from 75.9% to 77.3%.

Moreover, we created an additional column called ‘IsInsulinNonZero’, giving values of ‘1’ or ‘0’ to each column if they had an Insulin score of 0 or were diagnosed with Diabetes. While this column did not necessarily help improve the accuracy of our model, it was helpful to identify which rows had individuals with an insulin level of 0, as it seemed interesting that many rows had insulin levels of 0 and many others had insulin levels in the triple digits (up to a max of 846).

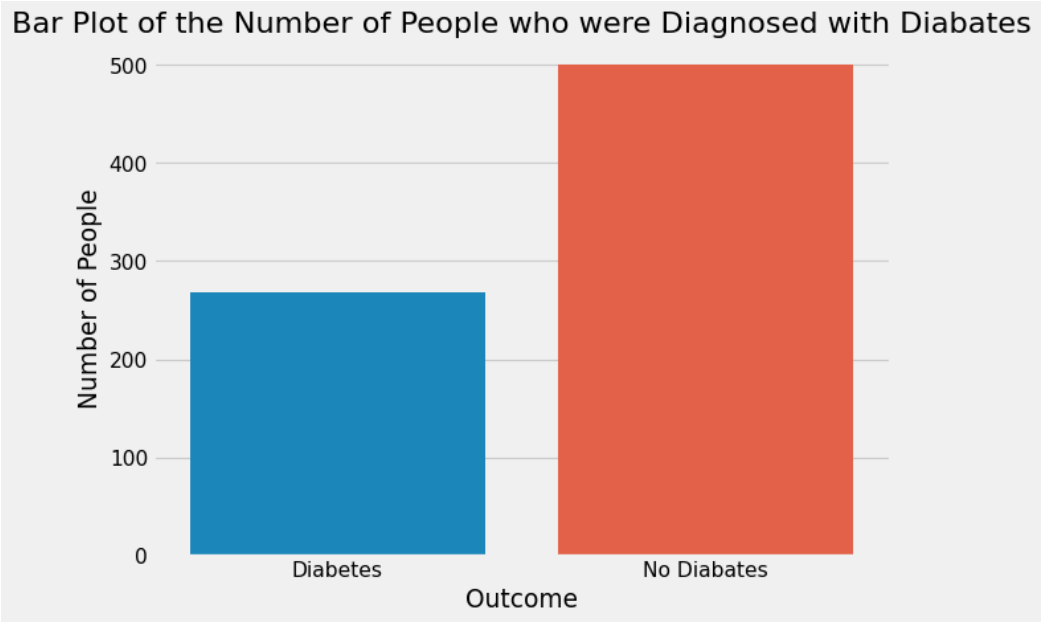
We tried to create a few other variables, such as a variable that factored in an individual’s number of pregnancies and age, but none of these variables seemed to improve the accuracy of the model, so we did not incorporate them into our final code.

Lastly, heatmap, bar plots, and boxplots were created to help emphasize which columns needed to be implemented in our modeling. To do this, matplotlib, numpy, and

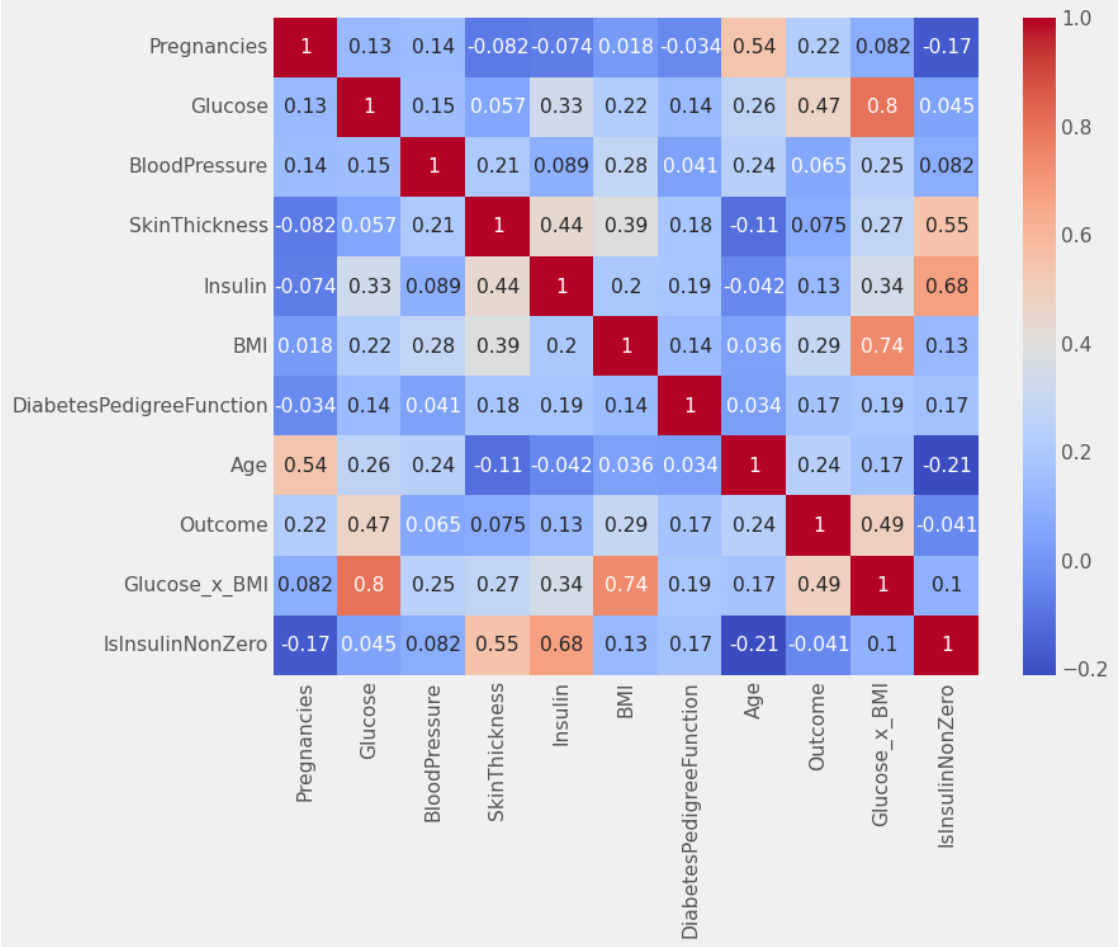
seaborn were used. This was extremely telling in sorting the importance of each of the variables in the dataset.

IV. Analysis and Visualization -

As it was laid out in Section III, the first thing that was checked in the dataset was how many of each outcome occurred. In the dataset that we were given to work with 268 of the samples had diabetes (indicated by a 1 in the "Outcome" column) and the other 500 didn't have diabetes (indicated by a 0 in the "Outcome" column). This means that 65.1% of patients in the dataset didn't have diabetes, giving us a baseline number for what our accuracy should be better than when we get to the modeling section.



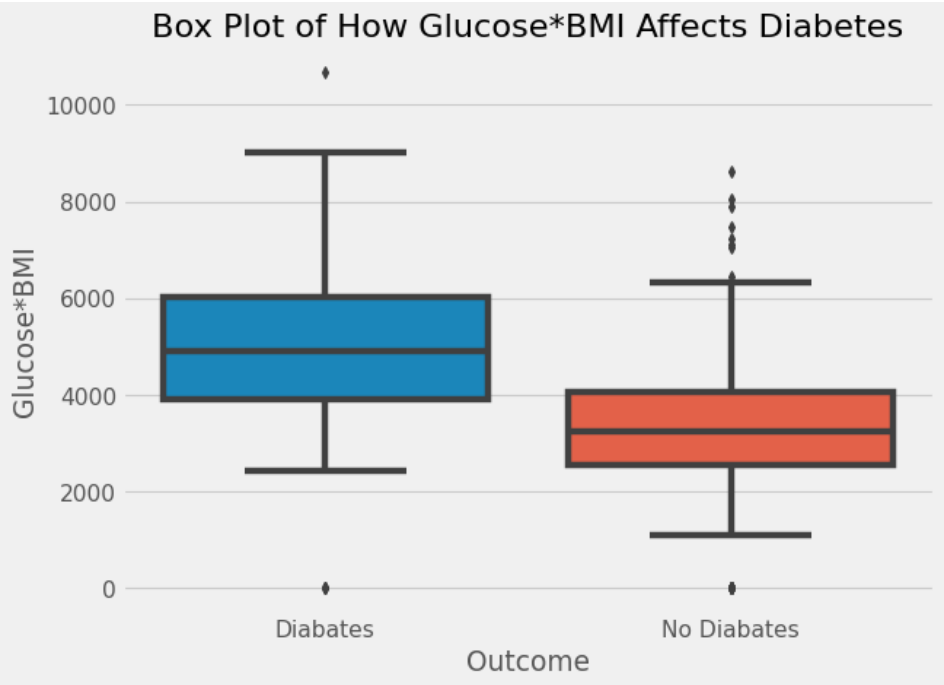
The next thing we can do is to take all of the original variables and the variables that were feature engineered and see how they correlate with each other and how they correlate with the response variable "Outcome".



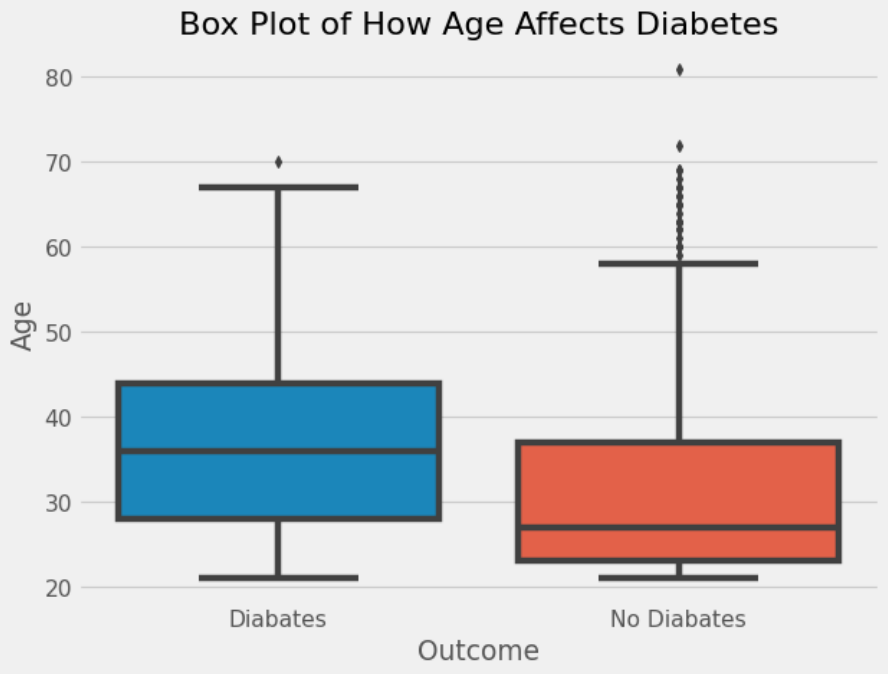
As seen in the correlation matrix, we can see why we would want to feature engineer Glucose and BMI by multiplying them together as “Glucose_x_BMI” correlates well with Glucose and BMI but not at a one-to-one level. It does have a 0.49 correlation with “Outcome” showing that when combining those two variables, there is a good signal in predicting whether or not someone has diabetes. Another set of variables that correlate strongly with each other is “Age” and “Pregnancies”, which makes intuitive sense as a woman’s age grows, so should the number of pregnancies as they get further in life. Also “Insulin” and “SkinThickness” have a pretty high correlation relative to the others which are useful to know as our priors didn’t consider those variables to be that similar.

The variables that strongly correlate with the “Outcome” response variable are “GLucose_x_BMI”, “Glucose”, “BMI”, “Age” and “Pregnancies” in that order. Glucose makes sense as the amount of sugar being measured in blood could be something that relates to diabetes as diabetes has a connection with sugar. BMI also checks out as a patient’s weight-to-height

ratio could affect their chance of getting diabetes. Lastly, as a person gets older their chance of getting diabetes should go up so age being there is directionally correct.



In this boxplot with “Outcome” on the x-axis and “Glucose*BMI” on the y-axis, we can see that there is a pretty significant difference between the Glucose*BMI values of those who have diabetes vs. those who don’t. Around 75% of patients that have diabetes have Glucose*BMI values over 4,000 while around 75% of patients that don’t have diabetes have Glucose*BMI values under 4,000.



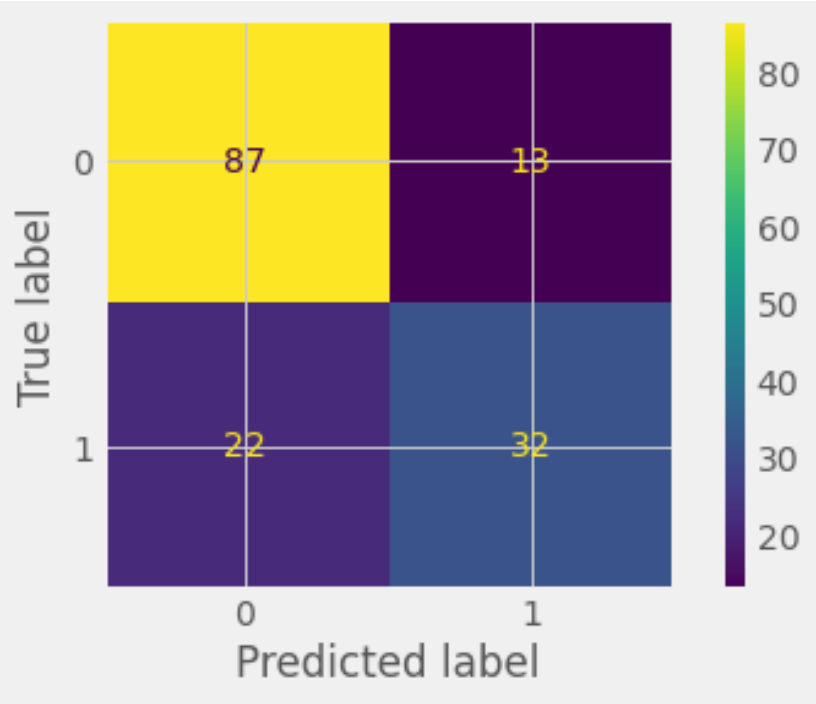
This boxplot, with "Outcome" on the x-axis and "Age" on the y-axis displays how older age can impact the likelihood of a patient getting diabetes. Out of the patients that didn't get diabetes, there appears to be outliers as most of the data is concentrated with patients under 40 years old but there is a wide right tail with it going all the way up to 80 years old.

With all this data analysis done, the data could be split up into training and testing datasets using a 80/20 split. Different modeling techniques were trained on the train dataset and their accuracy and F1 score was tracked on the test dataset. The table below shows the results for each type of classifier.

Model	Accuracy	F1 Score
Logistic Regression	0.740	0.59
Support Vector Machine	0.714	0.53
Random Forest	0.759	0.63
XGBoost	0.772	0.64
Tuned XGBoost	0.727	0.59
Gradient Boosted Classifier	0.720	0.57
LightGBM	0.759	0.64

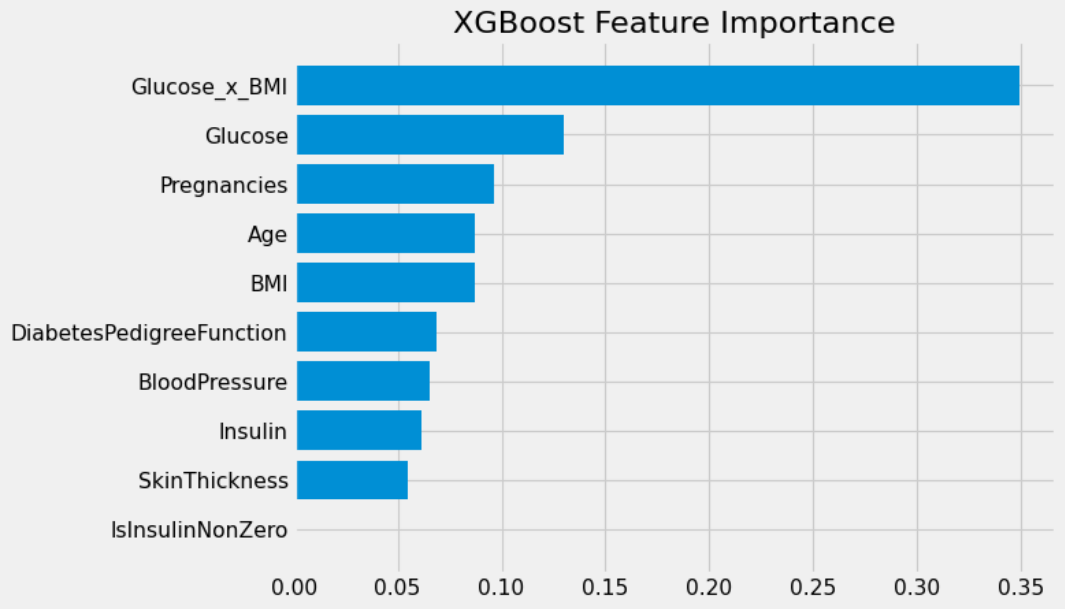
CatBoost	0.766	0.65

As seen, XGBoost gave the best accuracy on the test dataset, getting an accurate prediction on 77.2% of the predictions and scoring an F1 Score of 0.64. CatBoost had a slightly higher F1 Score but XGBoost will still be used as the final model as its accuracy was the highest. All in all, Logistic Regression, Random Forest, XGBoost, LightGBM or CatBoost could have all been used from an accuracy perspective but F1 Score shows a different story for some of them and places more emphasis on the positive class.



The confusion matrix for the XGBoost model shows that there are more false negatives than false positives but the model does a good job of predicting true negatives and true positives. It could be up for debate what is more important to focus on: accuracy on negatives or accuracy on positives.

We can then look at the feature importance for the XGBoost model to see which variables showed up as important and which didn't.



The variable that was feature-engineered, "Glucose_x_BMI", showed up as the most important. While there might be some collinearity with that variable and "Glucose" and "BMI", the reason why we chose a tree-based model was for that to not play as big of a factor as if we went with something like logistic regression. "Pregnancies" and "Age" also showed up with high feature importance which was laid out earlier in this section as having a strong correlation to the "Outcome" variable.

V. References

- Code: https://github.com/tejseth/diabetes-ml/blob/main/diabetes_prediction.ipynb
- Github: https://github.com/tejseth/diabetes-ml
- Pima Indians Diabetes Database: https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database
- Scikit-Learn Classification: https://scikit-learn.org/stable/supervised_learning.html#supervised-learning