Aidan Zhou

Professor Koehler

Data Bootcamp

16 December 2024

Final Project Write Up

Introduction: Overview of the data, predictive task, and summary findings.

For my final project, I decided to build classification models to predict whether or not a patient has diabetes based on multiple diagnostic measurements. The independent variables include number of pregnancies, glucose, blood pressure, skin thickness, insulin, BMI, Diabetes pedigree function, and age. A classification model based on these variables could be useful to patients and doctors alike who want to determine if they are at risk of developing diabetes and potentially lead to further testing based on the results. The models I created are a logistic regression with and without polynomial features, k nearest neighbors, and a random forest. For my additional model, I used the XGBoost library to create a model using XGBClassifier. Looking at the overall accuracy of the model and the recall score on the test data, the best performing model was XGBoost with an accuracy of ~78% and a recall of 69%.

Data Description: Data source and description

For this binary classification task the dataset I will be using comes from: https://www.kaggle.com/datasets/mathchi/diabetes-data-set. The data is originally sourced from the National Institute of Diabetes and Digestive and Kidney Diseases and contains 768 samples of females at least 21 years old of Pima Indian heritage. In addition to the 8 variables given in the dataset I feature engineered two additional independent variables: Glucose to Insulin Ratio and BMI * Age. My rationale for creating these engineered variables is that the level of glucose in the blood relative to the level of insulin is more important than the individual values by themselves. There have been a few studies: (https://pubmed.ncbi.nlm.nih.gov/9709933/, https://pubmed.ncbi.nlm.nih.gov/11397901/) that have found that the glucose to insulin ratio is potentially helpful in predicting a patient's insulin resistance which is closely related to type 2 diabetes. For the BMI and Age interaction, my logic was that as you get older the bigger of an impact the BMI will have on your health and likelihood of diabetes.

Most of the given variables are pretty self explanatory but the definition of the Diabetes Pedigree Function is not as clear. The DPF is a formula developed by the original authors of the paper (https://pmc.ncbi.nlm.nih.gov/articles/PMC2245318/) from which the dataset was sourced and it looks at the history of diabetes in each person's family and the genetic relationship of those relatives to the subject. The output of the pedigree function is proportional to the number of relatives diagnosed with diabetes divided by the number of family members without diabetes. Family members that are more closely related (siblings, parents) are weighted more heavily than cousins and uncles/aunts.

The correlation heatmap shows that glucose, BMI * Age, and BMI has the highest correlation with a diabetes diagnosis. Surprisingly, the Glucose Insulin Ratio has a near 0 correlation. The high correlation of glucose and diabetes makes a lot of sense as the level of glucose in blood is one of the primary diagnostic tools for identifying diabetes. Type 2 diabetes is caused by insulin resistance which prevents cells from processing glucose leading to chronic hyperglycemia. I would have thought that the glucose to insulin ratio would be more useful in predicting diabetes since insulin resistance is a hallmark of Type 2 diabetes, and the ratio of glucose to insulin could reflect how effectively the body is responding to insulin. A high glucose-insulin ratio might suggest that the body requires excessive insulin production to manage blood glucose levels, indicating the presence of insulin resistance. On the other hand, a low ratio could imply that the body is not producing enough insulin, which is a common issue in advanced stages of diabetes. But with this data, it does not seem to be that useful with our data in this case compared to looking at glucose and insulin by themselves.

A number of the rows of data contained zero measurements and to remedy this I decided to replace the zeros in Glucose, BloodPressure, SkinThickness, Insulin, BMI with the mean of the remaining data in order to preserve the integrity of the dataset while reducing data loss due to missing values. This may hurt the predictive power of my model and introduce some bias into the data as it assumes that the missing values are evenly distributed and not related to the target variable.

Looking at the histograms and boxplots of each of the variables, we can see a few relationships that warrant some investigation. Those with diabetes have higher numbers of pregnancies, glucose levels, blood pressure, skin thickness, insulin, BMI, diabetes pedigree function, age. Only the glucose to insulin ratio histogram seems to follow the same distribution for those with and without diabetes.

Models and Methods: Overview of models and implementation

Prior to modeling I created a train test split of the data with 20% in the test set. All of the models created will be trained and tested on the same split in order to allow for comparability between models. The baseline of predicting zero for all rows would provide an accuracy of roughly 65% and all of the models tested performed better than that.

The first model I used was a multivariate logistic regression using all 10 of the variables. To do this model, I created a pipeline of a standard scaler and the logistic regression. The logistic regression provided an accuracy of 77.5% on the training data and a slightly lower 73.3% on the testing data which doesn't raise any major red flags regarding overfitting on training data.

Next, building on the previous model, I implemented another logistic regression this time with polynomial features of degree 2 which creates 66 independent variables in total. I created another pipeline starting with scaling the data again, creating the polynomial features, and running the logistic regression again. For this regression, I set the solver to liblinear instead of the default of lbfgs since it handles larger datasets better and we have a large number of variables in the model due to the polynomial features. The default number of iterations is only 100 and that wasn't able to converge on a minimum, so I increased the max iterations to a thousand to ensure convergence. Interestingly, the accuracy of the polynomial logistic regression gave us a training accuracy of 82% but the exact same test accuracy as the normal logistic regression indicating that there was some more overfitting with this model.

The third model in this project was a K-nearest neighbors classifier. The model works by comparing the distance between a new data point to the existing labeled data points in the training set. The k part of the model can be chosen and it represents the number of neighbors to consider when evaluating test data. With a small k value, the model will be more sensitive to noise in the data while a large k will lead to underfitting. In order to find the optimal k value for my data, I utilized a grid search that tests k values from 1 to 15. After scaling the data, the grid search selected a k of 9 which resulted in a training accuracy of 80% but a testing accuracy of only 72%.

Similarly, I used a grid search for the implementation of the random forest model in order to select the max depth of the trees and number of estimators. A random forest is a collection of decision tree classifiers that are trained on different samples of the training data and the prediction is the average of the trees. Since scaling the data is not necessary for random forests, I ran the grid search and fit it directly on the training data. The chosen parameters were a max depth of 6 and 150 estimators in the forest. This model performed the best out of all so far with a training accuracy of 90% and a testing accuracy of 77%. This disparity may indicate that the model is overfit on the training data but it still performs better than the rest so far.

Last but not least, I implemented a classifier using XGBoost, a library that can create a gradient boosting decision trees model. A random forest and a GBDT model are similar but gradient boosting trains its trees iteratively and after each iteration it uses the residual errors from the previous and a loss function in order to make an improved tree. For this model I also used a grid search in order to select the number of estimators, max depth, and learning rate of the model. The selected parameters were a max depth of 4, 300 estimators, and a learning rate of 0.01. The XGBoost model had a lower training accuracy than the random forest with 87.6% but a higher testing accuracy of 77.9%. Additionally, the feature importance of the model lists glucose, Diabetes Pedigree Function, and the BMI * Age as the most important which lines up with my intuition.

Results and Interpretation: Review of modeling results and interpretation of performance

When evaluating the performance of the various models developed for predicting diabetes, it is crucial to consider the most appropriate metric for comparison. While accuracy is often the go-to measure, I would argue that it is not the most suitable choice in this particular context. Accuracy simply represents the overall proportion of correct predictions, encompassing both true positive and true negative cases. However, in the case of diabetes prediction, our primary concern should be identifying as many positive cases as possible, even at the expense of slightly lower accuracy in identifying negative cases. False negatives, in this context, represent individuals who have diabetes but are incorrectly classified as not having the condition. The consequences of missing these cases can be severe, as delayed diagnosis and treatment can lead to serious complications and health risks. Looking at the recall scores and confusion matrices for each of the models we can see that the XGBoost model still performs the best with a recall of 69.2%. The next best was the random forest with a score of 67.3%, followed by the k nearest neighbors, logistic regression, and polynomial logistic regression all tied at 59.6% coincidentally.

Looking more closely at the best performing model, XGBoost, I created plots of the ROC Curve and Precision Recall Curve. The ROC Curve plots the true positive against the false positive rate at various classification thresholds. The TPR is the proportion of actual positive instances that are correctly identified by the model while the FPR is the proportion of actual negative instances that are incorrectly identified as positive by the model. By varying the threshold, the model's performance can be adjusted to prioritize either higher TPR or lower FPR. The ROC curve is created by plotting the TPR on the y-axis against the FPR on the x-axis for different threshold values. A diagonal line represents a random classifier and as the area under the curve (AUC) increases the performance of the classifier increases. For our model we have an AUC of 0.83 which is certainly better than just randomly guessing but still means that there is potentially room for improvement.

The precision recall curve looks at the model's precision and recall at different classification thresholds. As the threshold decreases, recall increases (more actual positives are captured), but precision may decrease (more false positives are introduced), and vice versa. Similarly we can look at the area under the curve to get a sense of the model's performance with 1 being the most desirable. Our model achieves an AP of 0.74. For our use case, we can look at the precision recall curve to potentially select a threshold level that will maximize the recall of the model in order to prioritize the reduction of false negatives.

Conclusion and Next Steps: Summary of models and next steps for further analysis

The best performing model was XGBoost, achieving an accuracy of approximately 78% and a recall of 69.2% on the test set. This outperformed the other models, with random forest being the next best with 77% accuracy and 67.3% recall. When evaluating the models, recall was prioritized over accuracy, as the goal is to minimize false negatives (diabetic patients incorrectly classified as non-diabetic). False negatives are more detrimental in this context, as failing to identify diabetes cases can lead to delayed treatment and health complications. While the XGBoost model had the highest recall, there is still room for improvement in capturing all positive cases. XGBoost is also the most complex model and took the longest to train, perhaps making it less desirable if we were to increase the dataset. It is important to consider the fact that the dataset used in this project is relatively limited in scope and observations. The data comes from a limited demographic which may limit the generalizability of the models. Furthermore, the dataset contains only 768 observations, which is a relatively small sample size. While the models showed promising performance on this limited data, their ability to apply to larger and more diverse populations is uncertain. For next steps, I would prioritize gathering a more diverse dataset that includes a wider range of gender, ages, ethnicities, and ages. Additionally, there are many more variables that could be collected and trained on that could potentially contribute to the prediction including exercise, smoking status, socioeconomic indicators.