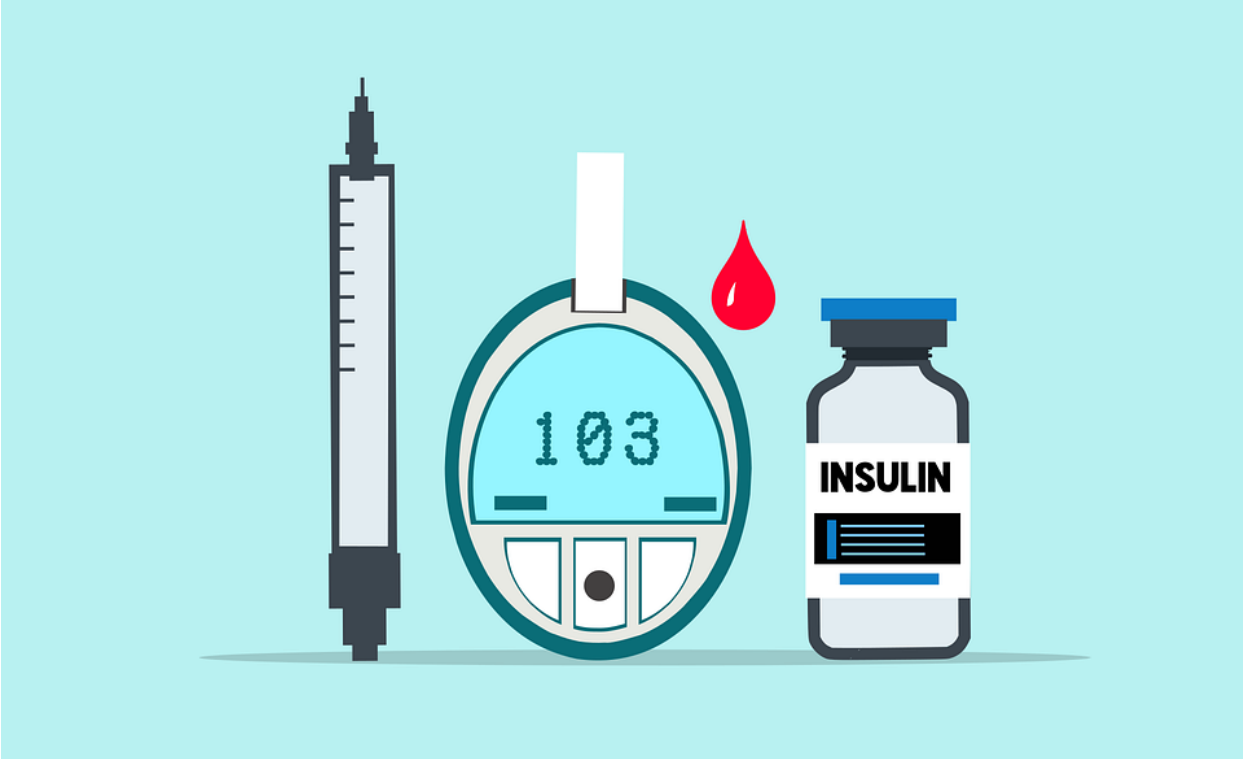
Diabetes Indicator

Data Analysis



Carmen Lee, Mason Maviglia, Cicelia Siu, Kevin-Ty Sweitzer

04 May 2022

CS 422 - Intro to Machine Learning

MOTIVATION

We wanted to challenge our machine learning abilities with a challenge that could have a huge impact on the world. Diabetes is a disease known for being undiagnosed for years when people have it. Since Diabetes affects 400 million people in the world, the goal of this project was to use machine learning to determine the most effective way of detecting whether or not an individual has diabetes or not.

According to the CDC, Diabetes is a chronic (long-lasting) health condition that affects how your body turns food into energy. The human body naturally breaks down the food you eat into a sugar called glucose, and releases the glucose into your bloodstream. Then the pancreas releases insulin which allows your cells to receive the glucose in your bloodstream. However, people with diabetes either cannot produce insulin or the insulin being created does not work. This leaves too high of blood sugar levels in their bloodstream which can later cause heart disease, vision loss, and kidney disease.

A motiviation for one of the members to specifically try a diabetes dataset is because their family member has diabetes. Anything they can do to help fight this struggle became their motiviation for this project.

DATASET

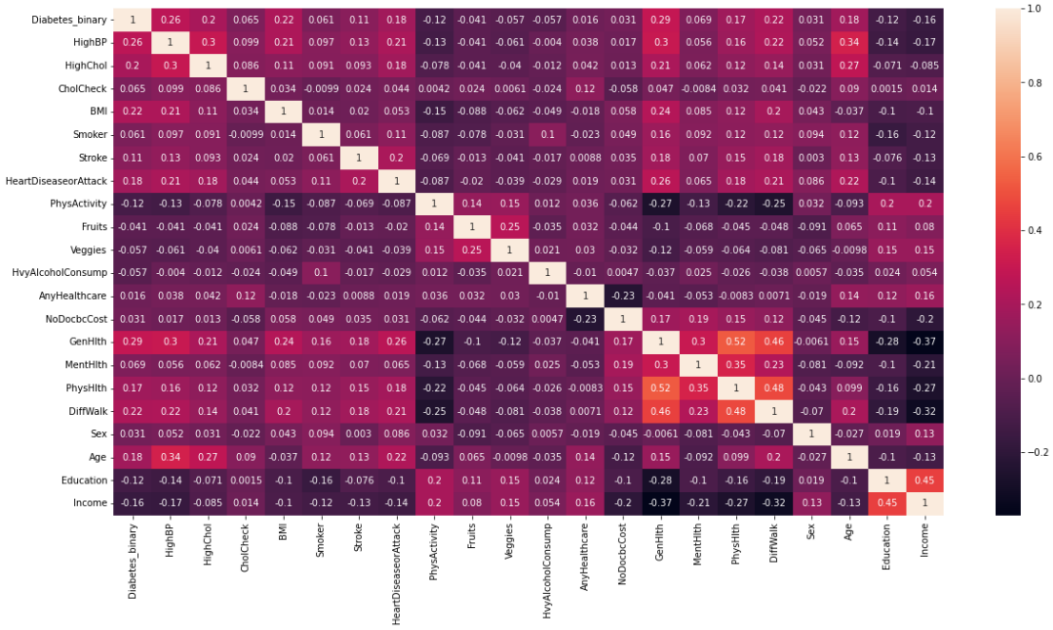
The dataset used for this project was found on Kaggle.com. The survey was performed via telephone and was conducted by the CDC in 2015 that asked participants numerous health related questions. The original survey resulted in 330 features in total with 253,680 responses. The modified dataset found on Kaggle is a reduced version with only 21 features and contains only two classes for the target variable: 0 for no diabetes, and 1 for prediabetes or diabetes. It is important to note that the dataset is not balanced.

The features that are included in the modified dataset are as follows: high blood pressure, high cholesterol, cholesterol check[ed], BMI, smoker, stroke, heart disease or attack, physical activity, fruits, vegetables, consumes heavy alcohol, any healthcare, afford doctor's visit, general health, physical health, mental health, difficulty walking, sex, age, education, and income. The 21 features were chosen because of their potential relevance to detecting diabetes in an individual, and range from health conditions to lifestyle choices and factors.

OBSERVATIONS

In our initial observation, we found that high blood pressure, high cholesterol, BMI, heart disease [or] attack, general health, and age all have significant correlation with the target variable. In addition, there is a slightly noticeable negative correlation with physical activity, education, and

income with the outcome variable, meaning that diabetes is less prominent in individuals where these features are higher.



Initial observations from the dataset visualized as a heatmap

MACHINE LEARNING ALGORITHMS

When it comes to creating a model for a binary classification problem, such as this one, there are many suitable and popular algorithms. We chose to use k-Nearest Neighbors (KNN), Logistic Regression, and Neural Network to create a diabetes classifier, all which are very different, because we wanted to see how they would compare to one another in performance and accuracy.

EXPERIMENT

K-Nearest Neighbors (KNN)

KNN is a simple, but very powerful supervised classification algorithm that classifies based on a similarity measure. KNN utilizes “lazy” learning because the model does not actually learn anything. Instead, when a new data is introduced, we find its k-nearest neighbor from the training data. This makes it very important to have a balanced and consistent training data.

To make our diabetes predictions with the KNN model, a new data is compared with the entire training data and takes the output of its nearest neighbor (in this case, the mode). The nearest neighbor is measured by Euclidean distance. The formula for Euclidean distance is defined as:

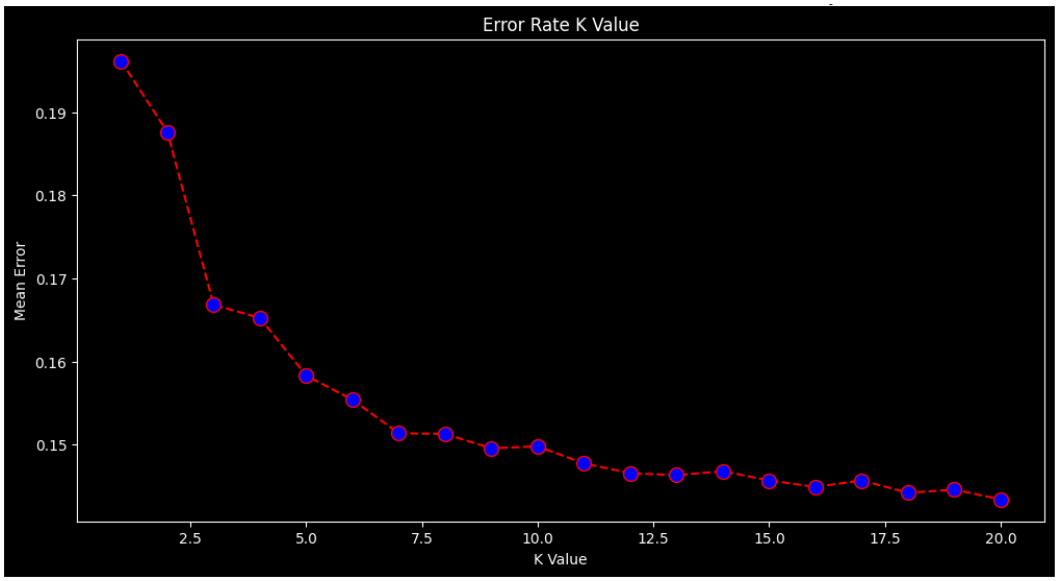
$$dist = \sqrt{\sum_{k=1}^{p}(a_k - b_k)^2}$$

, where p is the number of features and a_k and b_k are, respectively, the k^{th} feature of a and b.

Choosing the most optimal k is important to achieve maximum accuracy. We found that when k is less than or equal to 10, it leads to unstable decision boundaries, and for k greater than 10, the decision boundaries were smoother. For our KNN algorithm, we tested every k from 1 to 20. The optimal parameter for k would be where the error rate is at its lowest. Shown in graph below, the most optimal parameter was between 16 to 20. Due to our problem being a binary classification problem, 17, an odd k value, was chosen.

```
for i in range(1, 21):
    knn = KNeighborsClassifier(n_neighbors=i, weights = 'distance')
    knn.fit(X_train, y_train)
    pred_i = knn.predict(X_test)
    error.append(np.mean(pred_i != y_test))
```

Calculating error for k-values between 1 and 20

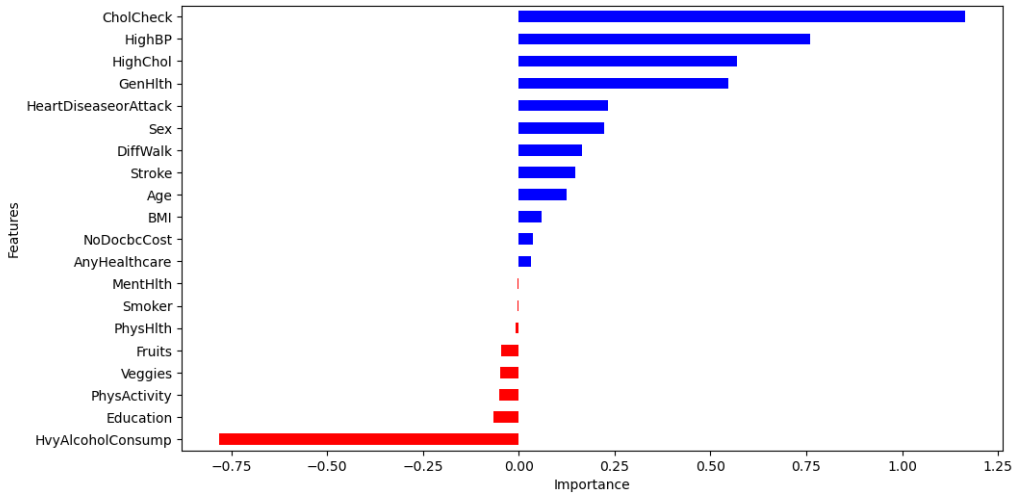


Convergence of mean error as k increases

Logistic Regression

Logistic Regression is another supervised machine learning algorithm we used to determine the probability of an individual having diabetes. We wanted to determine the correlation of each feature with the target variable and see how they could affect the model. In the initial interpretation of the correlation coefficient of each feature, we can see that CholCheck, HighBP,

GenHlth, and HighChol have significant influence on the model, whereas features like MentHlth and Smoker have little to no significance on the model.



Relationship between features and diabetes presence

The model uses a logistic function to model the conditional probability of the label Y (prediabetic or diabetic) variables X (the 21 features). The continuous result is then mapped to a closed set [0, 1] using a sigmoid function, which is used to further interpret probability. In order to classify the continuous value, we rounded the probability to 0 or 1 to make a prediction.

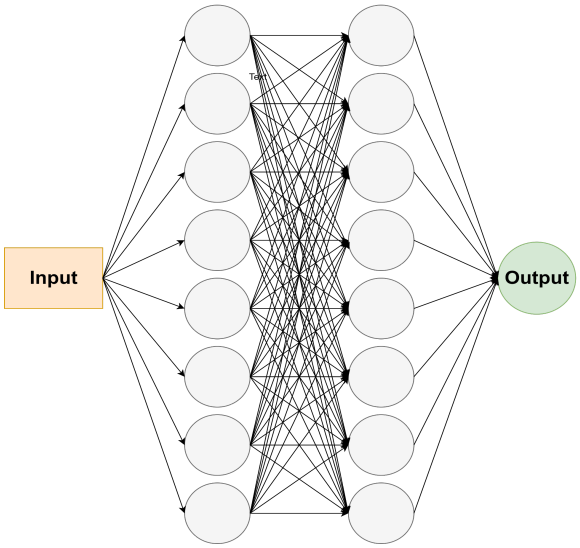
$$P(Y|X)$$

The conditional probability

Neural Network

In our artificial neural network (ANN), we use deep learning to perform binary classification on our dataset. This algorithm is completely different from our Logistic Model because it uses a backward propagation of errors with respect to the neural network's weights.

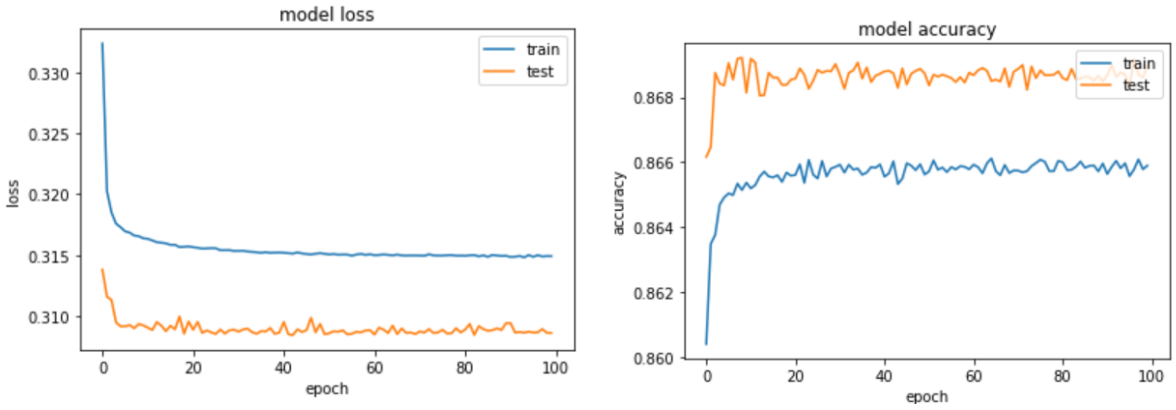
In our initial experimentation, we use a simple ANN architecture: one input layer of 20 nodes, two hidden layers of 8 nodes each that use the Rectified Linear Unit (ReLu) activation function, and one output layer of 1 node that uses Sigmoid activation function. The ReLu activation function outputs the input directly if it is positive and zero otherwise to effectively help the model learn faster, while the Sigmoid function reduces the output to a value from 0.0 to 1.0 representing a probability.



Initial ANN architecture

To optimize our ANN, we used a grid test algorithm of various hyperparameters to find the optimal parameters for the neural network. We experimented with different optimizers, loss functions, learning rates, batch sizes, number of nodes, and epochs which resulted in the most optimal hyperparameters: Adam optimizer with the default learning rate, Binary Cross Entropy loss function, the default batch size, 8 nodes for each hidden layer, and a total of 100 epochs.

We plotted the accuracy and loss to epochs to see how the model was improving each iteration. The elbow-shaped graph for the training data in the model loss plot shows that maximum accuracy was achieved, which corresponds with the model accuracy plot.



Model loss and accuracy plots for ANN

RESULTS

Confusion Matrix and Classification Report

The Confusion Matrix and Classification report gives a number to show the overall accuracy of the models. The confusion matrix shows the number of true/false positives and true/false negatives. Comparing the confusion matrices between the models, each model got approximately the same number of true positives and negatives, meaning that they were able to achieve around the same accuracy. We can also see from the confusion matrices that the models made a significant amount of false negative guesses compared to false positives. This shows that the model was more likely to guess negative if the model was unsure.

The classification report further proves these observations. Looking at the precisions -the ratio of the true model guesses to the sum of true and false model guesses for a specific class- of the models you can see each model had a very high precision in guessing negative cases with the testing dataset. The recall, which shows the percentage of instances the model was able to find in the class, also shows how well the models were able to find negative cases since each model was able to correctly find over 97% of all negative instances compared to only finding at least 14% of the positive instances.

KNN Results

Confusion Matrix

	POSITIVE	NEGATIVE
POSITIVE	10576	383
NEGATIVE	1451	274

Classification Report

	PRECISION	RECALL	F1-SCORE
0.0	0.88	0.97	0.92
1.0	0.42	0.16	0.23
ACCURACY			0.86

Logistic Regression Results

Confusion Matrix

	POSITIVE	NEGATIVE
POSITIVE	10706	253
NEGATIVE	1452	273

Classification Report

	PRECISION	RECALL	F1-SCORE
0.0	0.88	0.98	0.93
1.0	0.52	0.16	0.24
ACCURACY			0.87

Neural Network Results

Confusion Matrix

	POSITIVE	NEGATIVE
POSITIVE	10749	210
NEGATIVE	1486	239

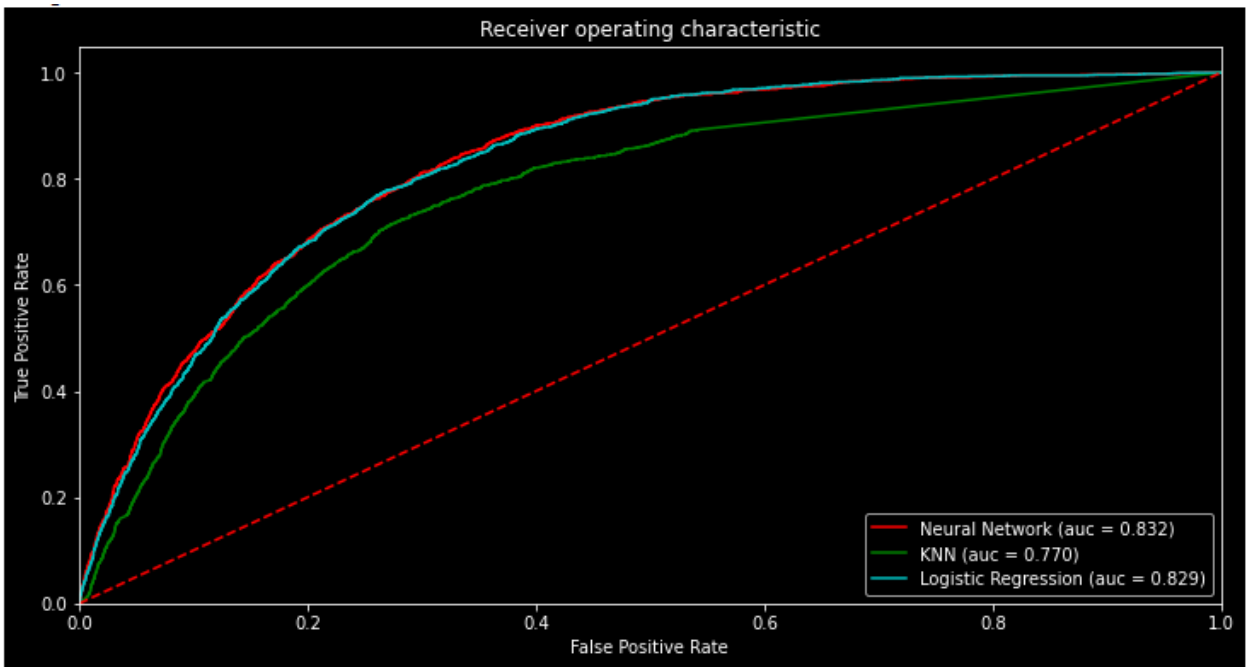
Classification Report

	PRECISION	RECALL	F1-SCORE
0.0	0.88	0.98	0.93
1.0	0.53	0.14	0.22
ACCURACY			0.87

ROC Curve

The ROC curve represents the discrimination between the classes that the models made. Having an ROC curve that is significantly different from a straight line shows that the classifier model is not making random guesses. Looking at the ROC curve from all of our models we can clearly see that our models were not making random guesses, and were using the training data to make inferences about the testing data. The area under the curve (AUC) shows how well the classifier is able to distinguish between the different classes. Looking at each models AUC, we can see that the Neural Network and Logistic Regression models were able to distinguish between the positive and negative classes much more than the KNN algorithm

Based on a rough classifying system, AUC can be interpreted as follows: 90 -100 = excellent; 80 - 90 = good; 70 - 80 = fair; 60 - 70 = poor; 50 - 60 = fail



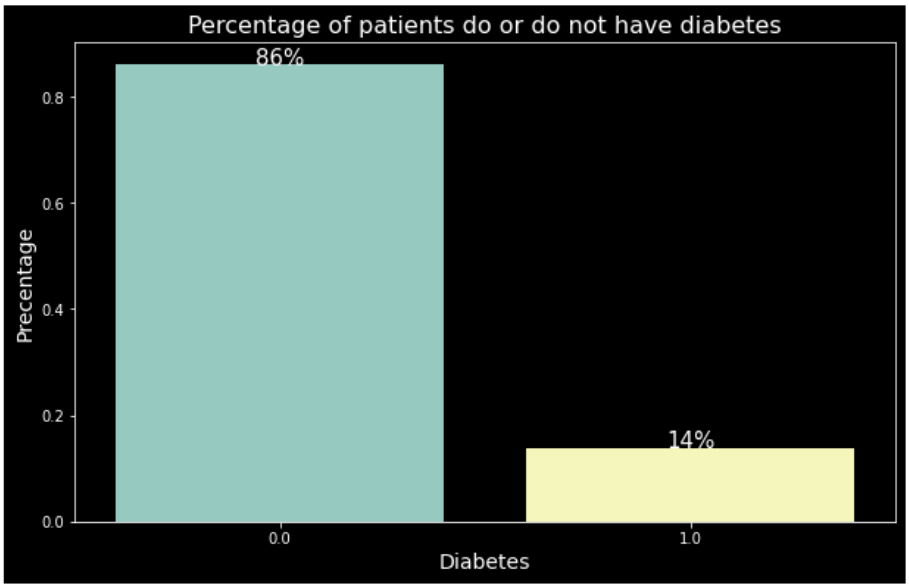
CONCLUSION

Through the experiment we have found that Logistic Regression was the best technique to use for binary classification. It was more effective than KNN should in the ROC/AUC and the F1 score. Since Neural Network has very similar scores in the Confusion Matrix, Classification Report, and AUC, we chose to compare the time it took to run each technique. We found that it was more efficient to use Logistic Regression in comparison to Neural Network. In the end, we found that Logistic Regression is the best technique between these three when it comes to binary classification.

MISCELLANEOUS

The one big obstacle we faced was scheduling. We were all busy people with jobs and school. Some of us even drove while trying to attend the meeting or made sure to get on the meeting right after getting home.

The next issue we faced was from the beginning of our experiment, when we didn't look at what our dataset consisted of. We first looked at the features and the number of samples, but we didn't consider the imbalance in the dataset. Near the end when the team was looking at the uneven F1 score from the classification report, we felt that there was something wrong. We searched up why the uneven F1 score would happen and found that an imbalanced dataset would do that. Looking at the percentage from each class, 14% of the samples has diabetes and the other 86% did not have diabetes.



Percentage of patients that do or do not have diabetes.

To fix the imbalance dataset, we added class weights to Logistic Regression. Logistic Regression was chosen over KNN due to its higher precision and was chosen over Neural Network due to the time that it took to find the optimal parameters. The confusion matrix and F1 score seemed to match the dataset more and shown below. Although the class weights didn't give the optimal F1 score, it was much more evenly distributed and the accuracy seemed to match our model much better. Had we done this again, we could have added custom class weights to further optimize the model.

Logistic Regression using Class Weights

Confusion Matrix

	POSITIVE	NEGATIVE
POSITIVE	7908	3051
NEGATIVE	380	1345

Classification Report

	PRECISION	RECALL	F1-SCORE
0.0	0.95	0.72	0.82
1.0	0.30	0.78	0.44
ACCURACY			0.73

MEMBER PARTICIPATION

All the members meet online, twice a week starting from when the project was announced till the end of writing the report.

Carmen Lee

Lee agreed to work on the diseases and illness topic. She looked over the proposal and proofread it to make sure we were following what was required. Lee and Sweitzer worked together on the Logistic Regression algorithm and code. Lee started the Neural Networking algorithm and code. Most of the individual graphs were made by Lee, including the heat map, the logistics regression importance bar chart and more. The graphs on the poster and presentation, the professional look of the poster, and the revision of the poster's and presentations text were done by Lee. In addition, parts of the presentation including the Logistics Regression and Neural Network algorithms were completed by Lee. Lee completed her part of the oral presentation and sent it to Siu. Lee drafted the first half of the report, added in most of the visuals, and made it look professional.

Mason Maviglia

Maviglia agreed on the diseases and illness topic and traded datasets with Siu and Sweitzer to find the best datasets to work on. After concluding on the dataset, Maviglia wrote about half the proposal. Maviglia and Siu worked together on the KNN algorithm and code. Maviglia created the gridtest to find the optimal parameters for the Neural Network. After splitting the test parameters with the other members, he ran the grid test. On the poster, Mavilga filled in the parts of the context for the poster and presentation including, Motivation and Dataset and Algorithm of KNN. Maviglia completed his part of the oral presentation and sent it to Siu. Maviglia edited and revised the report.

Cicelia Siu

Once agreeing to work on diseases and illnesses, Siu traded possible datasets with Mavilglia and Sweitzer. Siu wrote part of the proposal and set due dates for each part of the project. Maviglia and Siu worked together on the KNN algorithm and code. After splitting the test parameters with the other members, she ran the grid test for NN. Siu found an outline for the poster project and the presentation and noted which parts were crucial to add in both. Siu ran the code for any parameters that needed to be run again. Anything that was changed was readjusted to the poster and the presentation. After the poster presentation, Siu added class weights to logistic regression, combined the three ROC curves into one graph, and added both to the presentation. Siu completed her part of the oral presentation and edited the presentation video. She completed the membership participation and the Misc. portion of the report.

Kevin-Ty Sweitzer

Sweitzer was the first person to start sending ideas that he wanted to accomplish. Most of which were dealing with diseases or illnesses. Sweitzer, along with Maviglia and Siu, traded datasets found on Kaggle to vote on the best one. Once a dataset was found, he started writing about half the proposal. Lee and Sweitzer worked together on the Logistic Regression algorithm and code. After splitting the test parameters with the other members, he ran the grid test for NN. On the poster, Mavilga and Sweitzer filled the majority of the context for the poster. Sweitzer completed his part of the oral presentation and sent it to Siu. He outlined what the final report should have and began to add context into the report. Sweitzer completed the results portion (classification report, confusion matrix, ROC curve), and the motivation section of the report.

REFERENCES

Teboul, A., Centers for Disease Control and Prevention (2011). Diabetes Health Indicators Dataset. Retrieved March 24, 2022 from https://www.kaggle.com/datasets/alexteboul/diabetes-health-indicators-dataset