**Disease X Detection**

**Project Description**

The data set given for our project includes 90 predictors and 110,000 samples. The predictors are different physiological factors, and the response variable is whether the individual is infected with disease X. Class 1 corresponds to an infected individual and class 0 means the individual does not have the disease. Although the predictors are not defined, the physiological factors could be related to blood pressure, body mass index, cholesterol level, weight, height, or blood pressure. The data values range from high negatives to high positives, but without more context about the predictors not much else can be inferred from the dataset.

**Data Visualization**

The first plot we made to visualize the dataset was a bar graph depicting the response variable. As shown in **Figure 1**, more individuals did not have disease X, but the two classes were close. 60,000 people we not infected and about 50,000 people were infected. The next graph we chose to make were boxplots of the predictors. **Figure 2** is a boxplot of all 90 predictors, and it shows that X14-X27 have the largest range in the whole dataset. We then made a boxplot of predictors X1–X30 to investigate this further. **Figure 3** depicts these boxplots clearer and shows the high variation of data within X14-X27.

***Figure 2:*** *Boxplot of all Predictors*

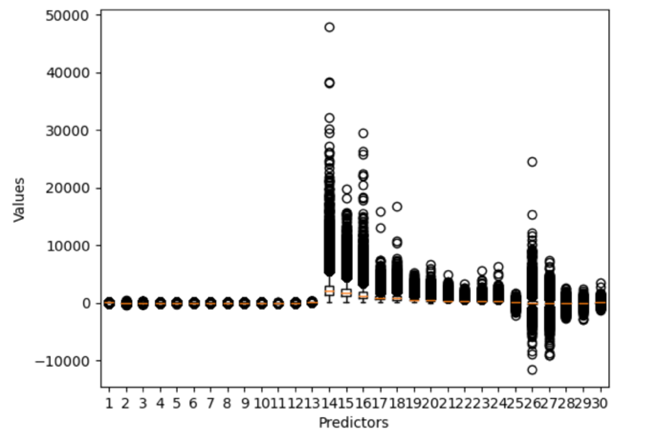
***Figure 1:*** *Graph of Infected (1) and non-Infected (0)*

A graph of a graph

Description automatically generated with medium confidenceA red and blue bar graph

Description automatically generated

***Figure 2:*** *Boxplot of all Predictors*



The correlation plot shown in **Figure 4**, was used to visualize the correlation between different variables in a dataset. It shows a color-coded matrix, where each cell represents the correlation coefficient between two variables. Strong correlations (positive or negative) are indicated by dark orange and dark blue, aiding in the exploration of multivariate datasets. We took out predictors X18, X22, and X23 to begin pre-processing because they were highly correlated with each other with values above 0.8. In the end, the model accuracy with and without these predicates were essential the same, so we ended up keeping them.

***Figure 4:*** *Correlation Plot*

**A screen shot of a chart

Description automatically generated**

**Data Pre-Processing/Data Splitting Techniques**

For pre-processing, we decided to scale the data to standardize the independent variables. This will ensure that one predictor will not dominate the others and help the model perform better. We also split the dataset into test and train data by using 80% of the data for training and 20% for the test data. We used a Z-score of 1.96 to calculate the outliers and once they were removed, the regression model had an accuracy of 77.15% as shown in **Figure 5.** Our SVC model had an accuracy of 78.8% as shown in **Figure 6** and **Figure 7** shows our neural network model accuracy is 74.5%. Before we took out the outliers, the model had around a 73% accuracy. A Z-score of 1.96 means the confidence level is 95%. We used the Z-score to help us identify outliers above this threshold and remove them from the dataset.

***Figure 5:*** *Model Accuracy*

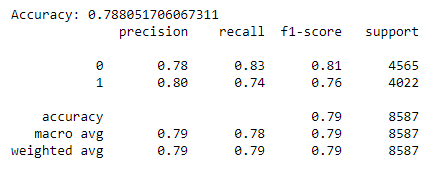
A screenshot of a computer code

Description automatically generated

***Figure 6:*** *Support Vector Classifier Accuracy*

***Figure 7:*** *Neural Network Model Accuracy*

A screenshot of a computer

Description automatically generated

For deliverable 3, we used SMOTE but the model didn’t change much, it made the results worse. This is a result of the training data set being fairly balanced already.

**Description of Models**

**a. Model Selection**

The first model we chose to analyze the dataset was logistic regression. We chose this classification model because disease detection involves the binary classification of two classes, positive (1) or negative (0). Logistic regression is specifically designed to model the probability of an event occurring as a function of one or more predictor variables. The logistic regression model uses the logistic function (sigmoid function) to transform a linear combination of predictor variables into probabilities, ensuring that the predicted probabilities fall between 0 and 1.

The random forests method is an ensemble algorithm used for classification and regression tasks. It builds multiple decision trees, introducing randomness by training on random subsets of data and features. We also decided to use this method to create a model on the Disease X dataset. As discussed in the following sections, this method did not perform as well as logistic regression, but both methods produced an accuracy above 75%.

The third model we chose to analyze the dataset with was support vector classifier. Support Vector Classifier (SVC) is a supervised machine learning algorithm used for classification and regression tasks. Its primary objective is to find the optimal hyperplane that best separates data points into different classes in a high-dimensional space. This hyperplane maximizes the margin, which is the distance between the hyperplane and the closest data points from each class, known as support vectors.

We then created a model using neural networks (NN). This is a machine learning model inspired by the human brains structure. It consists of interconnected nodes organized into layers. Each node processes input data, applies an activation function, and passes the result to the next layer. Neural networks excel at modeling complex, non-linear relationships in data, automatically learning relevant features, and scaling to large datasets.

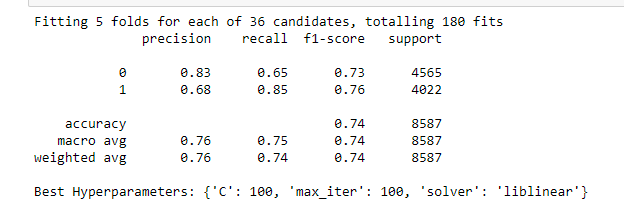
In deliverable 3, we were given a new test data set to create more predictions given that the new test data set has more noise than our previous set, so we would need to use a more robust model. Knowing that SVC was the best model we’ve used so far, we used that and hyperparameter tuned the model. We switched through 10 different kernels, penalty values, and gammas, but it was too computationally expensive and wouldn’t run to completion. We simplified it by choosing the best parameters of three different models. We tested the kernels in SVC like RBF, poly, sigmoid, and linear kernels. We used RBF in deliverable 2 because we don’t have much context behind the data we are modeling. Even after testing the other three kernels, RBF still came out to be the best. We used RBF and then tested different penalty parameters (larger parameters could lead to overfitting) to get the best accuracy. When we tested larger penalty values, c=100 and c=50 for example, the accuracy dropped. We chose to use the c value of 1 and we then received an accuracy of 78.81%, which was best.

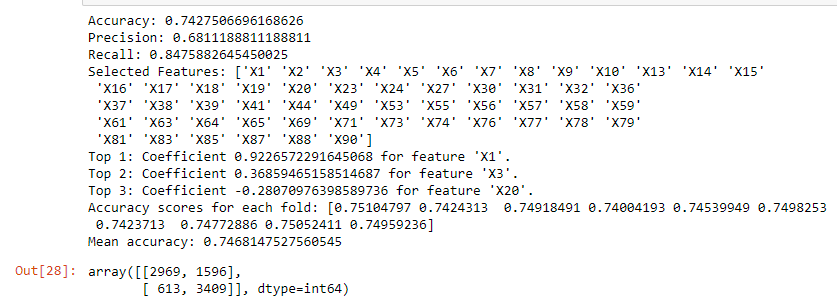
The Huber loss function is a type of loss function usually used in regression tasks, particularly in machine learning models where robustness to outliers is desired. It is used to train the model parameters (e.g., neural network weights) by minimizing the difference between predicted and true values while effectively handling outliers. To make this useful for our classification case, we used the function Stochastic Gradient Descent Classifier in python. It is a linear classifier that optimizes the classification objective using stochastic gradient descent (SGD). After coding this for our data, we got an accuracy of 72.9% as shown in **Figure 8**.

A screen shot of a computer program

Description automatically generated ***Figure 8****: Huber Loss*

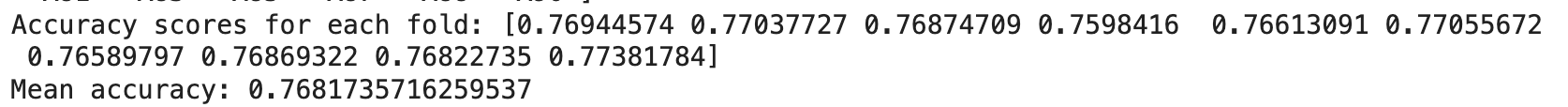
Lastly, we decided to go back to the Logistic Regression model to try hyperparameter tuning so we could get a more effective model because we are testing it against more noise. It is more likely to perform better than the SVC. **Figure 10** shows the accuracy of the model after using Logistic Regression again after giving more importance to false negatives. We did this because it’s generally better to reduce false negative when it comes to disease detection. Class weights were added to the model making class 1 a 2 and class 0 a 1 in order to predict more 1s than 0’s.

** *Figure 9****:* *Hyperparameter Tuning*

***Figure 10****: Logistic Regression Accuracy, CV, & Confusion Matrix*

**b. Resampling Method Utilized During Training**

We chose to use 10-fold cross validation to limit problems like over-fitting or selection bias. The benefits of using cross-validation includes a more accurate assessment of the model's performance, reduced risk of overfitting, and reduced variability. Using 10-fold, the dataset is divided into 10 subsets, the learning algorithm is trained 10 times, and each time a different fold is used as the validation set while the remaining 9 folds are used as the training set. **Figure 9** shows after validating our logistic regression model 10 times, the output numbers were very close to our final accuracy of 77.15%.

** *Figure 9:*** *10-Fold Cross Validation Values*

For the support vector classifier and neural network models we decided to use 5-fold cross validation so the python code wouldn’t take as long to run. **Figure 10** shows the values from using 5-fold cross validation for our support vector classifier model. As shown, the mean was about 78% which is very close to its accuracy of 78.8%.

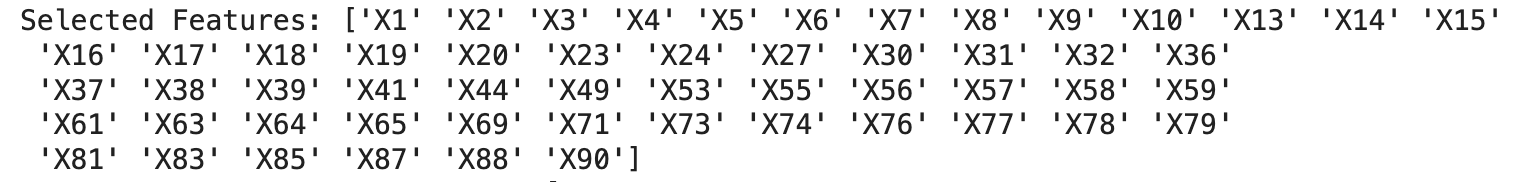
***Figure 10:*** *5-Fold Cross Validation Values*

image

**c. Performance Metric Used to Gauge Prediction Accuracy of the Models**

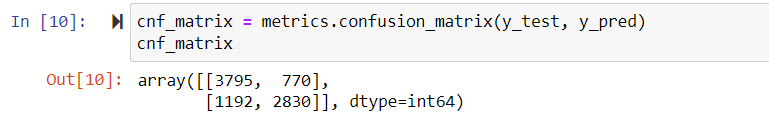
Our initial logistic regression model from deliverable 1 produced an accuracy of 66% when we used all 90 predictors. We were not happy with this number, so we decided to take out some predictors to improve the model’s accuracy. After testing differing amounts of predictors with the python package function called recursive feature elimination, using 55 predictors resulted in the highest accuracy of 77.15%. We kept the same preprocessing we did in deliverable 1 for this deliverable. **Figure 11** lists the predictors that resulted in the highest accuracy which we used in all our models.

***Figure 11:*** *Chosen Predictors*

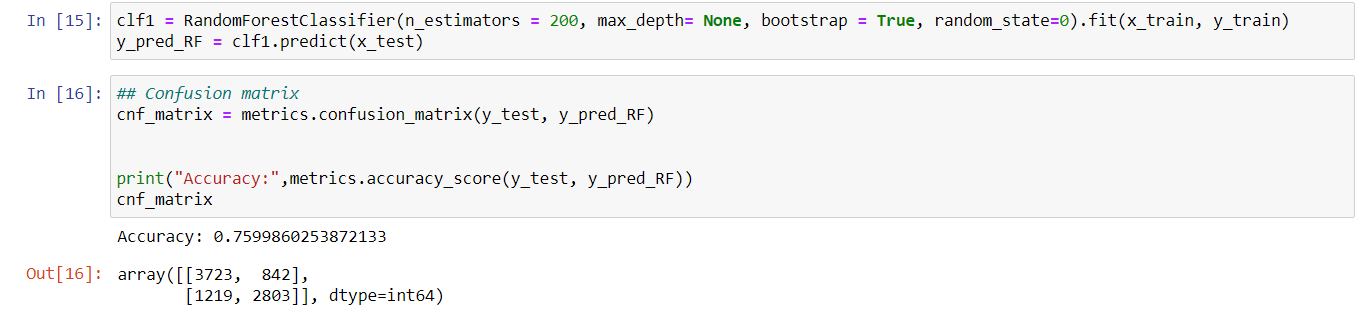
****

**Visual Representation** **of Results**

A confusion matrix is a table used to evaluate the prediction performance of a classification algorithm. The matrix has four entries: True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN). As shown in **Figure 12**, our regression model accurately classified 3795 people as TP and accurately classified 2830 people as TN. **Figure 13** depicts the accuracy and confusion matrix for our random forest model. It has a slightly lower accuracy than the regression model, therefore it predicted more false positives and negatives. The ROC curve below was produced from our logistic regression model and shows good performance because the area under the curve value is higher than 0.5 and close to 1.

***Figure 12:*** *Confusion Matrix (Logistic Regression)*

***Figure 13:*** *Confusion Matrix (Random Forest)*



***Figure 14:*** *Logistic R**ROC Curve*

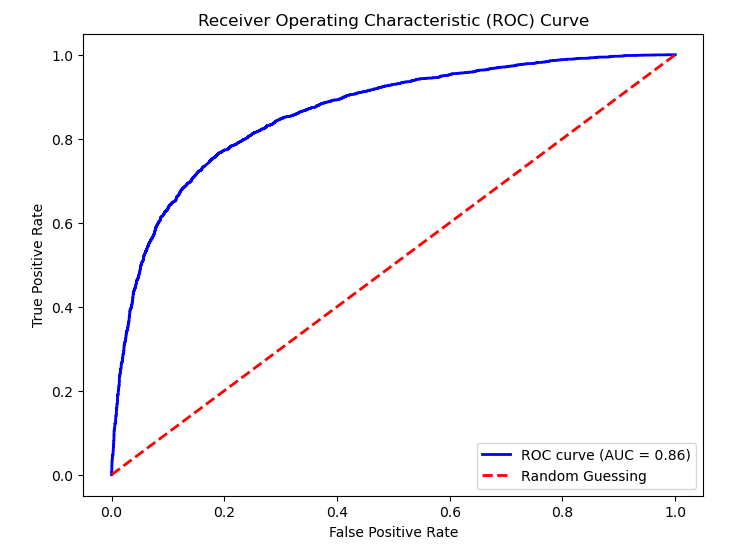
**A graph of a positive curve

Description automatically generated with medium confidence**

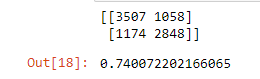
As shown in **Figure 15**, our support vector classifier model accurately classified 3808 people as TP and accurately classified 2959 people as TN. **Figure 16** depicts the accuracy and confusion matrix for our neural network model. As shown, it has a slightly lower accuracy than the SVC, therefore it predicted more false positives and negatives. **Figure 17,** the ROC curve shown below was produced from our support vector classifier model and shows good performance with an area under the curve of 0.86.

***Figure 17:*** *SVC**ROC Curve*

***Figure 15:*** *Confusion Matrix (Support Vector Classifier)*



***Figure 16:*** *Confusion Matrix (Neural Network)*



**Conclusion**

The four classification models we chose to analyze our dataset were logistic regression, random forests, support vector classifier, and neural network. After scaling the data, removing outliers, and reducing the number of predictors from 90 to 55, our support vector classifier model resulted in the highest accuracy. It had an accuracy of 78.8% and the second highest was our regression model which has 77.15% accuracy. Our random forests model was 76% accurate and our neural network model was 74.5% accurate at making predictions. The most significant predictors were X1, X3, and X20 because they were the predictors with the largest regression coefficients. We created visualizations to better interpret the data and implemented 10-fold and 5-fold cross-validation to test model reliability.

For deliverable 3, we addressed the new test dataset with increased noise by refining our logistic regression model. We chose logistic regression because SVC is a complex model that would not perform well with the noise. As stated before, class weights were added to this model so the 1’s are now 2 and the 0’s are now 1. Therefore, it will predict more 1’s than 0’s. Even though this gave us a lower accuracy, it is better to over predict positives than negatives when it comes to disease detection, which is our objective.