SACRAMENTO STATE
COLLEGE OF ENGINEERING & COMPUTER SCIENCE

CSC 215-01 Artificial Intelligence (Spring 2023)

Project 1: Predicting Acute Kidney Injury after Liver Cancer Resection

By
Rajvee Modi
Sac State ID - 219712052

Maryam Siddique
Sac State ID – 219485345

Due at 3 pm, Monday, February 20, 2023

Predicting Acute Kidney Injury after Liver Cancer Resection

Abstract:

Artificial intelligence is increasingly being used in disease prediction by analyzing large amounts of data to identify patterns and risk factors that may indicate the presence of a disease. In this project, we design various AI models to predict the likelihood of Acute Kidney Injury (AKI), a common postoperative complication among surgical patients. We used machine learning models to test the presence of AKI by regarding this problem as a binary classification problem. Thus, we implemented models such as Logistic Regression, Nearest Neighbor, Support Vector Machine, and classification Neural Network model using Tensor Flow. For this project, we compare the accuracy of the best classification neural network model obtained with per classification model achieved in the Scikit learn model.

Problem Statement:

Generally, a postoperative complication of surgery known as Acute Kidney Injury (AKI) accounts for almost 18% - 47% of hospitalized AKI Patients. AKI can lead to an increased risk of in-hospital mortality and chronic kidney diseases. Thus, it makes the timely diagnosis of AKI crucial for the treatment. To solve the problem, we can treat the issue as an AI binary classification problem. We will implement machine learning models such as Logistic Regression, Nearest Neighbor, Support Vector Machine, and classification neural network model using TensorFlow and compare the accuracy, precision, recall, and F1 Score for the prediction of AKI.

Methodology:

1. **Data Pre-Processing:**

   We attempted to model our classification in three steps. The first step involved processing the data. In data processing, we ensured that the data was fully loaded into the Python notebook and contained zero null rows. By running the isnull() method, our dataset contained zero null values thus no rows were removed. We further checked our dataset for duplicate data rows by running the duplicated() method and found zero duplicate rows for our dataset. Then, we proceeded to encode categorical features and normalized numeric features. To normalize, we dropped the KDIGO column as it was not really important to compare the final outcome of the model. Additionally, we used the zscore and also tried min-max scalar and to_xy methods for neural network model to normalize the dataset. This normalized the dataset to obtain a cleaner dataset.

2. **Train Test and Split the Dataset**

   The next step included splitting the dataset to train and test the machine-learning models. We chose our train-test ratio to be 70:30, and 70% of the data was dedicated to training and 30% dedicated to testing our classification models using train_test_split() method.

3. Data Source

The AKI dataset obtained from the resource article is used to have been used in this AI process of which 70% is treated as training data and 30% is considered as testing data. The dataset has various attributes used in this empirical study. Different configurations of this data set are available with variations in the number of instances but the number of attributes in each case is the same. The attribute labeled KDIGO tells us whether the patient in the data set is affected by the symptoms. For data cleaning, we removed the redundant and missing rows, however, our data was already clean and consisted of zero redundancy. Further, to treat this problem as a binary classification problem we have encoded the good connections as "0" and the bad connections as "1".

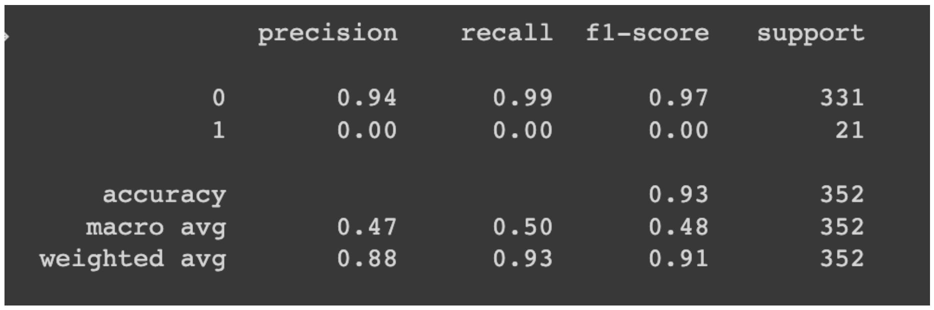
4. Machine learning Models

We used four models to evaluate our classification problem. While processing the dataset we realized that the dataset provided was unbalanced so we had to balance the set for 0's and 1's where 0 indicates the absence of symptoms and 1 indicates that a patient is affected. We divided our project into three subsections: unbalanced dataset, balanced dataset for top 10 features and balanced data set for all features. Following is the process that we used to evaluate our problem for 5 major categories: ROC curve, accuracy, f1-score, precision recall and confusion matrix.

1. Logistic Regression:

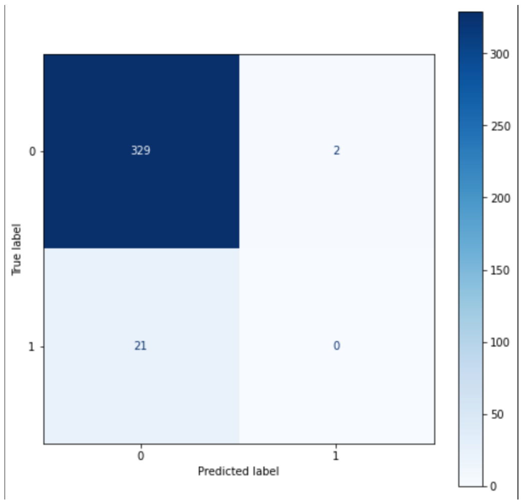
Logistic regression is intended for binary classification problems so it was perfectly suitable for our problem. Using sklearn we implemented a logistic regression model to predict the probability of an instance belonging to the default class, which can be snapped into a 0 or 1 classification. Further we created a confusion matrix to evaluate our model.

For LogisticRegression we calculated , the actual and predicted values, ROC Curve, accuracy, f1-score, precision, recall, confusion matrix and classification report as follows:

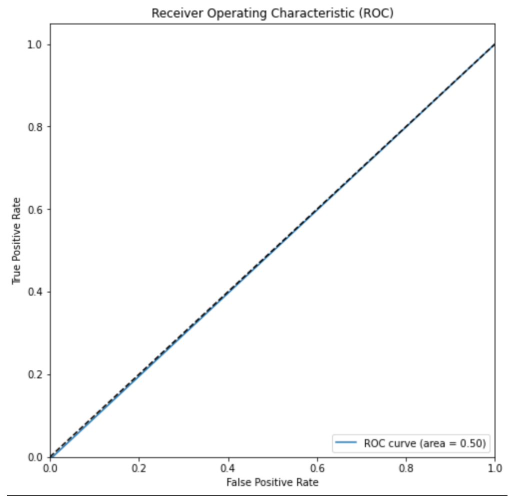


```
              precision    recall  f1-score   support

           0       0.94      0.99      0.97       331
           1       0.00      0.00      0.00        21

    accuracy                           0.93       352
   macro avg       0.47      0.50      0.48       352
weighted avg       0.88      0.93      0.91       352
```

Confusion matrix:

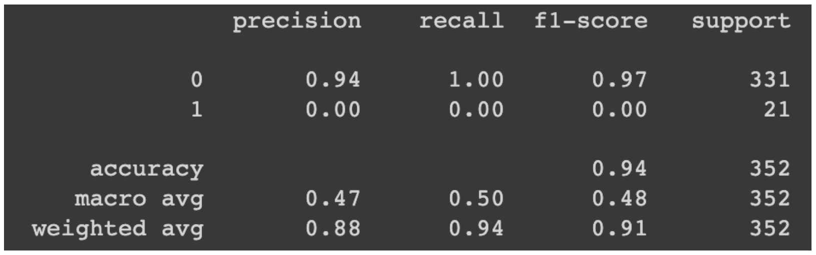


ROC Curve for logistic regression:



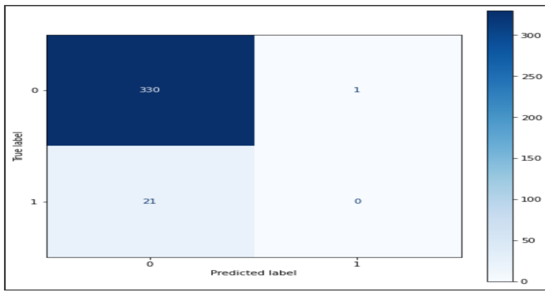
2. Nearest Neighbor:

In a nearest neighbor an input object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors where k is a small positive integer. If k = 1, then the object is simply assigned to the class of that single nearest neighbor. We implemented this model using the sklearn library and calculated the following categories with the unbalanced dataset.

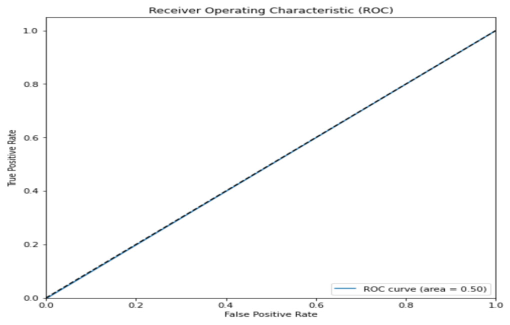


```
              precision    recall  f1-score   support

           0       0.94      1.00      0.97       331
           1       0.00      0.00      0.00        21

    accuracy                           0.94       352
   macro avg       0.47      0.50      0.48       352
weighted avg       0.88      0.94      0.91       352
```

Confusion matrix:

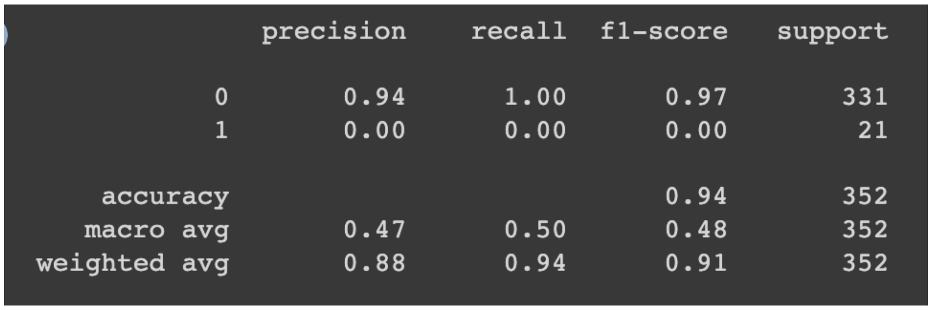


ROC Curve for Nearest Neighbor

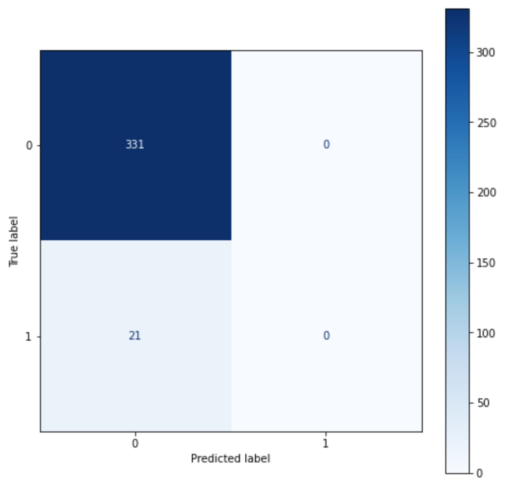
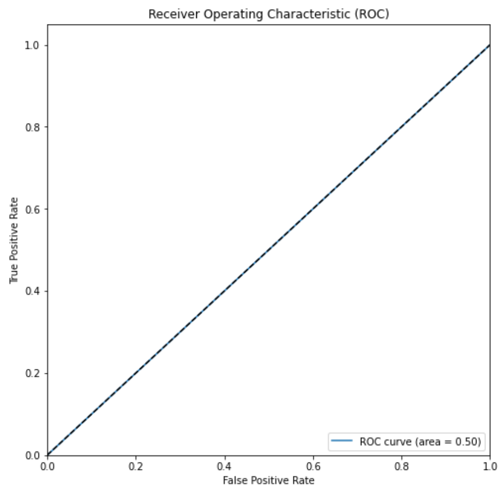


3. Support vector model (SVM)

Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier.

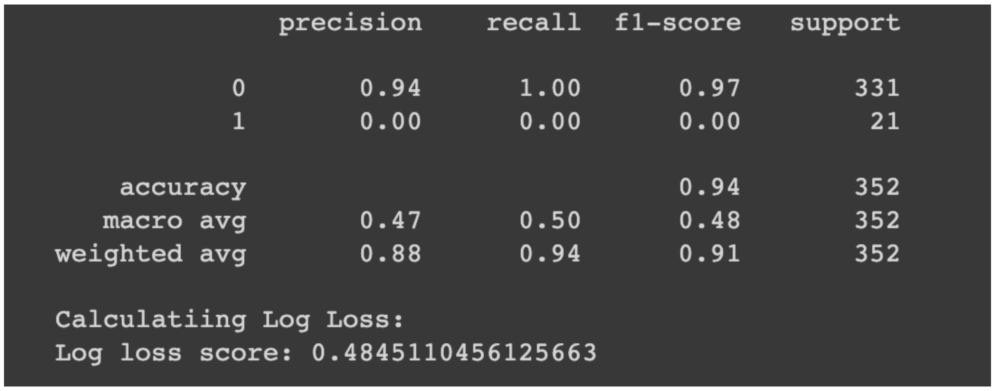


```
              precision    recall  f1-score   support

           0       0.94      1.00      0.97       331
           1       0.00      0.00      0.00        21

    accuracy                           0.94       352
   macro avg       0.47      0.50      0.48       352
weighted avg       0.88      0.94      0.91       352
```

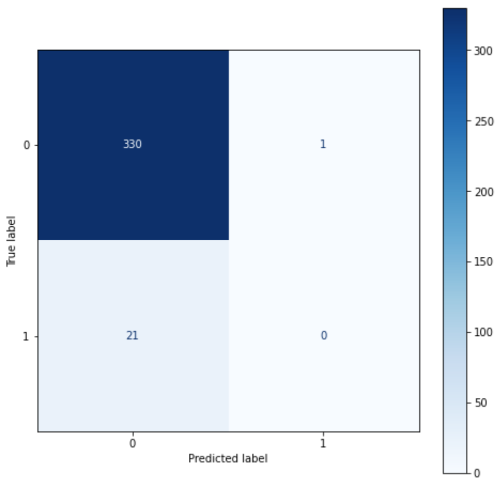
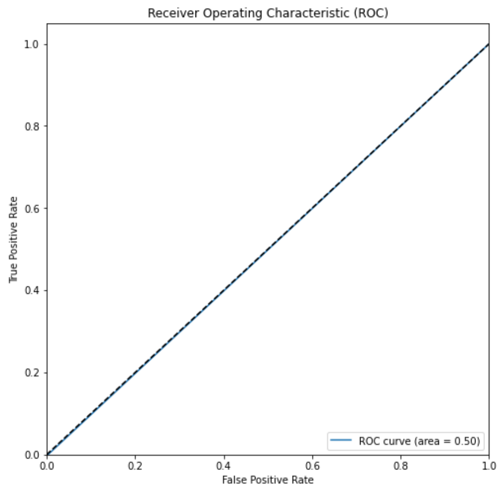
Confusion Matrix	ROC Curve for SVM
	

4. Fully Connected Neural Network

Finally, we implemented a Neural network to obtain the intended calculations. For this model we re-processed our data and splitted the dataset for training and testing again with a different to_xy method. We used Relu as our activation function for the hidden layer and softmax for our output layer. The number of layers were a multiple of 2 for our model and we used the adam optimizer to minimize the loss in our model. We chose Relu as our activation function because it is used in almost all the convolutional neural networks or deep learning. We further used theTensorboard to visualize the hidden models. We calculated the following parameters using the Tensorflow API.



```
              precision    recall  f1-score   support

           0       0.94      1.00      0.97       331
           1       0.00      0.00      0.00        21

    accuracy                           0.94       352
   macro avg       0.47      0.50      0.48       352
weighted avg       0.88      0.94      0.91       352

Calculatiing Log Loss:
Log loss score: 0.4845110456125663
```

Confusion Matrix	ROC Curve for NN
	

`

Experimental Results and Analysis:

As mentioned above we divided our project into three section to evaluate the best model based on the five categories: ROC curve, accuracy, f1-score, precision recall and confusion matrix. Bellow are table to indicate the values for all the models.

UNBALANCED DATASET INCL. ALL FEATURES

Machine learning Model	Precision	Recall	F1-score	Accuracy
Logistic Regression	0 : 0.94 1 : 0.00	0 : 0.99 1 : 0.00	0 : 0.97 1 : 0.00	0.94 for 352 support

Nearest Neighbor	0 : 0.94 1 : 0.00	0 : 1.00 1 : 0.00	0 : 0.97 1 : 0.00	0.94 for 352 support
Support Vector	0 : 0.94 1 : 0.00	0 : 1.00 1 : 0.00	0 : 0.97 1 : 0.00	0.94 for 352 support
Fully connected Neural Network	0 : 0.94 1 : 0.00	0 : 1.00 1 : 0.00	0 : 0.97 1 : 0.00	0.94 for 352 support

BALANCED DATASET INCL. ALL FEATURES

Machine learning Model	Precision	Recall	F1-score	Accuracy
Logistic Regression	0 : 0.76 1 : 0.67	0 : 0.64 1 : 0.79	0 : 0.70 1 : 0.72	0.71 for 658 support
Nearest Neighbor	0 : 1.00 1 : 0.85	0 : 0.84 1 : 1.00	0 : 0.91 1 : 0.82	0.91 for 658 support
Support Vector	0 : 0.98 1 : 0.84	0 : 0.83 1 : 0.98	0 : 0.90 1 : 0.91	0.90 for 658 support
Fully connected Neural Network	0 : 0.94 1 : 0.93	0 : 0.94 1 : 0.94	0 : 0.94 1 : 0.93	0.94 for 658 support

BALANCED DATASET INCL. TOP 10 FEATURES

Machine learning Model	Precision	Recall	F1-score	Accuracy
Logistic Regression	0 : 0.66 1 : 0.62	0 : 0.63 1 : 0.64	0 : 0.64 1 : 0.63	0.64 for 658 support
Nearest Neighbor	0 : 0.91 1 : 0.85	0 : 0.85 1 : 0.91	0 : 0.88 1 : 0.88	0.88 for 658 support
Support Vector	0 : 0.73 1 : 0.67	0 : 0.66 1 : 0.74	0 : 0.69 1 : 0.70	0.70 for 658 support
Fully connected Neural Network	0 : 0.86 1 : 0.78	0 : 0.78 1 : 0.86	0 : 0.82 1 : 0.82	0.82 for 658 support

From the data above we configured that the unbalanced data set was evaluated almost similarly for all four models and for the balanced dataset with all features the best model was the neural network with the best division of the confusion matrix and the f-1 score out of the other three. Finally, the balanced database for top 10 features was also evaluated best with the neural network as well. Overall, the accuracy score was higher for all models because the dataset for training and testing was relatively small so the models did not have enough rows to learn during the process to evaluate the accuracy.

Task Division and Project Reflection

Rajvee Modi

1. Analysis of the dataset and figuring out the attributes that are needed to be considered and finalize the approach of how to solve the problem.
2. Implementation of data cleaning and pre-processing.
3. Implementation of normalization of dataset and Implementation of SVM, Neural Network with Hyperparameter Tuning
4. Strategy and implementation of the additional feature of feature selection using logistic regression and balancing the data using SMOTE algorithm.
5. Comparison of models implemented.

Maryam Siddique:

1. Analysis of the dataset and figuring out the attributes that are needed to be considered and finalize the approach of how to solve the problem.
2. Implementation of data cleaning and pre-processing.
3. Implementation of Logistic Regression and KNN
4. Implementation of Tensorboard.
5. Comparison of models implemented.

Challenges encountered and their solution:

For this project we faced two major challenges: one was to understand the data and balancing the original dataset. We resolved this challenge by studying different methods and which would work the best for balancing the dataset. Second was data preprocessing as we were confused in using zscore for normalization as well as minmax scalar. Later we decided to try both and with zscore we received better accuracy.

Learning:

This project provided us with a broad picture of how to begin with pre-processing and data cleaning. The hands-on activities during class lectures helped us map out ideas and put them into action during our project. We learned about various data mining techniques and were able to share this knowledge and improve our project ideas as a team. It was a good learning experience for the team to gain a thorough understanding of the topics covered in class.

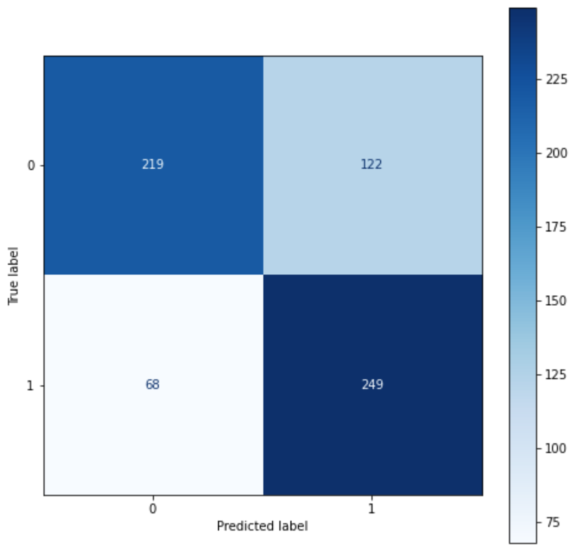
Additional Features

Balance DataSet:

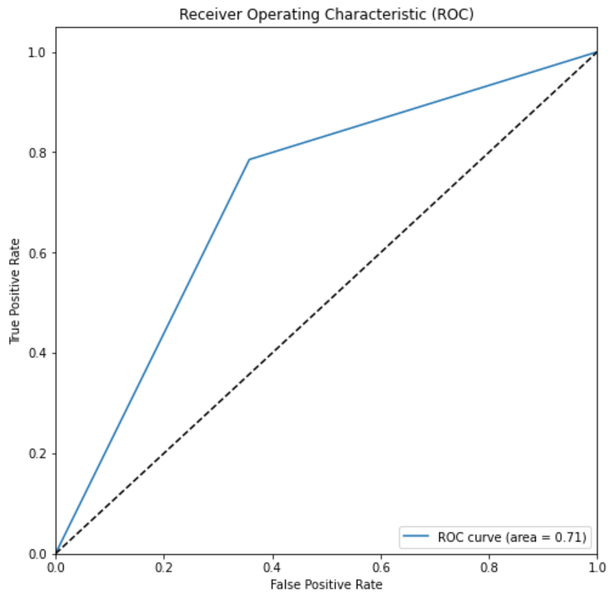
The original dataset contained many 0's leading to inefficient results, we balanced out the dataset to obtain balance between 0's and 1's. To do this we used a predefined function SMOTE(). The SMOTE function allows us to address the imbalanced datasets by oversampling the minority class. The simplest approach involves duplicating examples in the minority class and this function allows us to do the oversampling for balancing the dataset.

For the additional features, we balanced the original dataset to obtain optimized results. We also implemented the hyperparameter tuning for the neural networks. Additionally we tested each model with top 10 features to visualize optimal results and compare all models. Below are the ROC curves and the confusion matrices for all four models with the balanced dataset.

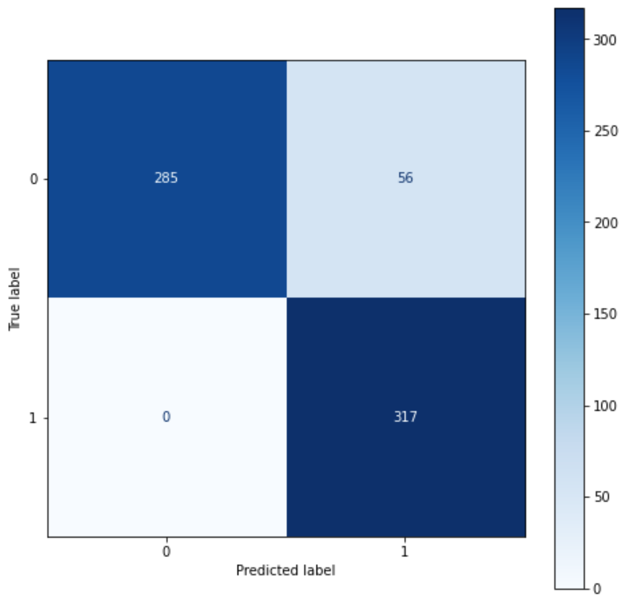
Confusion matrix for logistic regression (balanced):



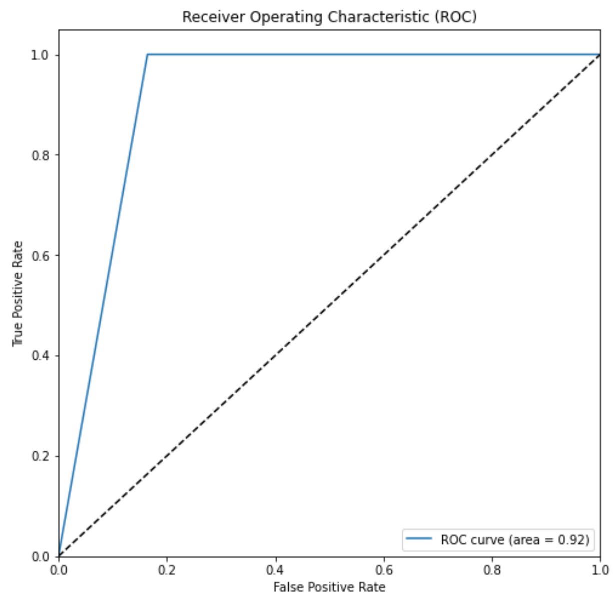
ROC Curve for logistic regression (balanced):



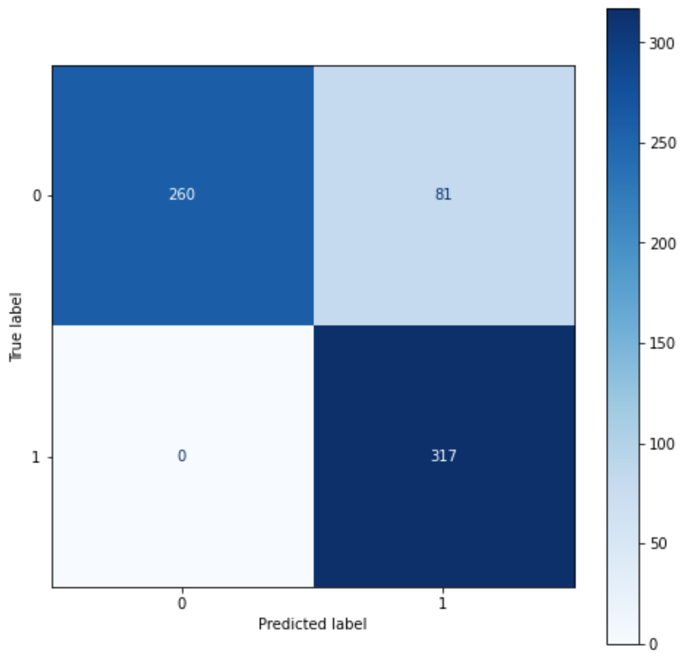
Confusion matrix for Nearest Neighbor (balanced):



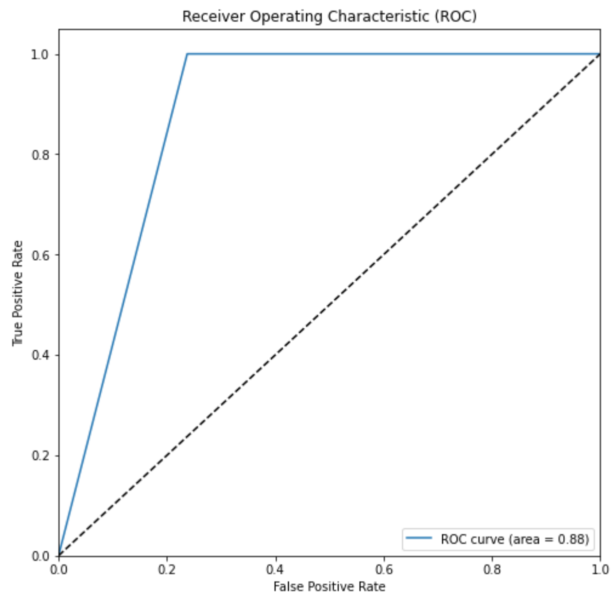
ROC Curve for Nearest Neighbor (balanced):

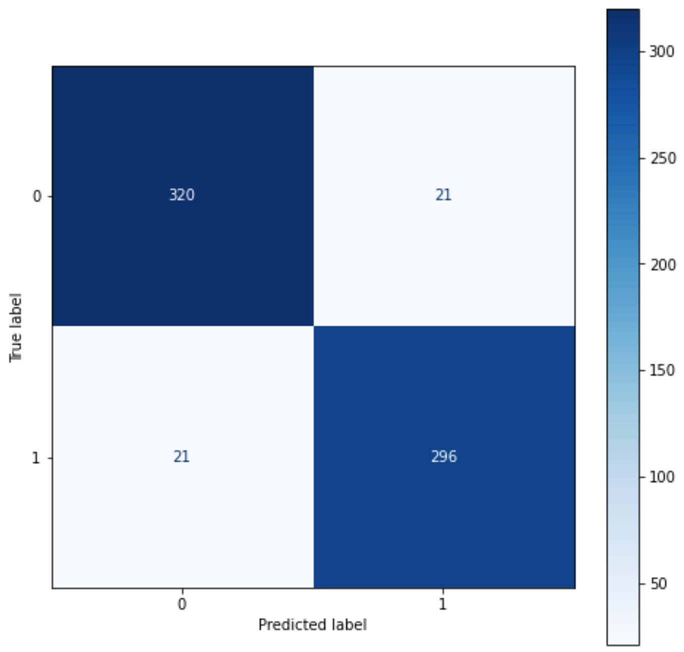
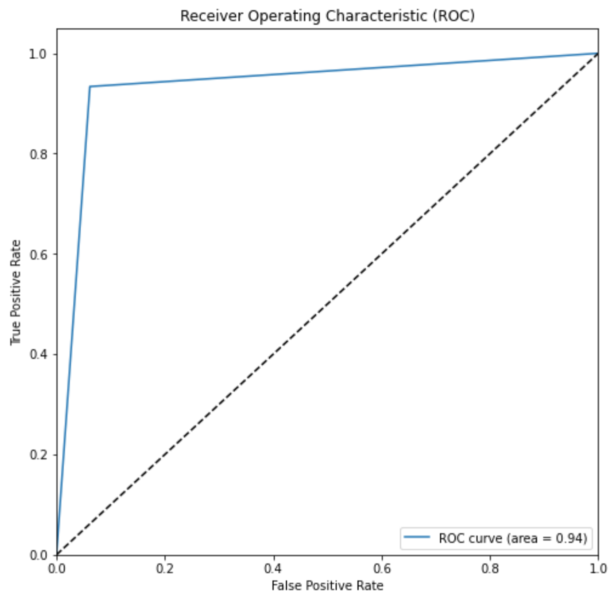


Confusion matrix for SVM(balanced):



ROC Curve for SVM (balanced):

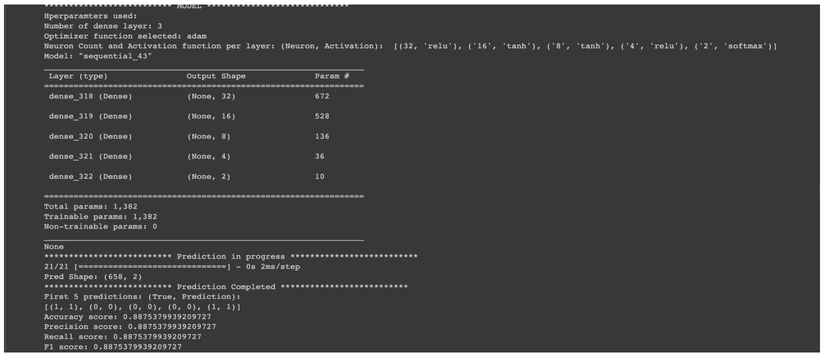


Confusion matrix for Neural Network (balanced):	**ROC Curve for Neural Network (balanced):**
	

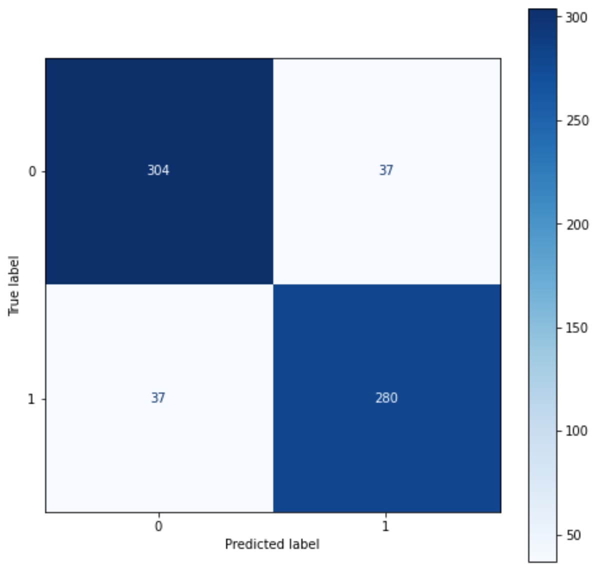
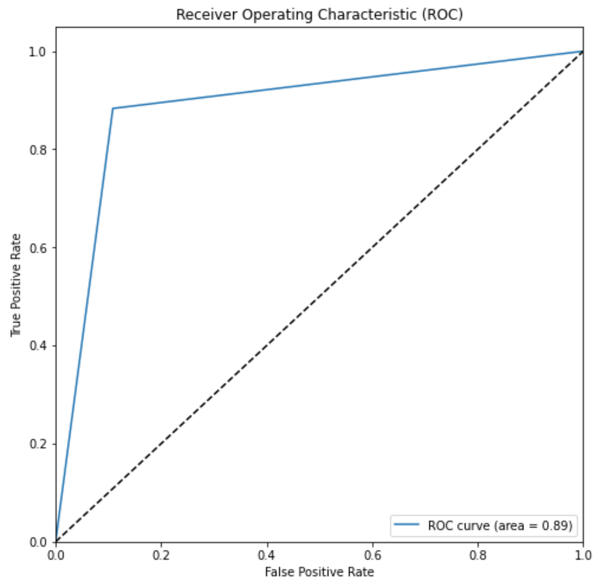
The above four tables show us the drastic change in the confusion matrices and the ROC curves after the dataset was balanced. Notice that with the original dataset the presence of 1 (patients with the symptoms) was 0 and after we balanced the dataset the spread was relatively different and gave us a better ROC curve for all models. We repeated the similar process for top 10 features and obtained similar optimum results for all fours models.

Logistic Regression:
We used logistic regression to obtain the top 10 features and evaluate these features for all our models. We also performed hyperparameter tuning for the top 10 features

Hyperparameter Tuning:
For the neural network models we did hyperparameter tuning to obtain balanced and optimal results. We implemented hyperparameter tuning for both balanced and unbalanced dataset and it gave us optimum results. Below is a visualization of our results after tuning with the balanced dataset.



```
************************** MODEL ******************************
Hperparamters used:
Number of dense layer: 3
Optimizer function selected: adam
Neuron Count and Activation function per layer: (Neuron, Activation):  [(32, 'relu'), ('16', 'tanh'), ('8', 'tanh'), ('4', 'relu'), ('2', 'softmax')]
Model: "sequential_43"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_318 (Dense)           (None, 32)                672

 dense_319 (Dense)           (None, 16)                528

 dense_320 (Dense)           (None, 8)                 136

 dense_321 (Dense)           (None, 4)                 36

 dense_322 (Dense)           (None, 2)                 10

=================================================================
Total params: 1,382
Trainable params: 1,382
Non-trainable params: 0
_________________________________________________________________
None
************************** Prediction in progress **************************
21/21 [==============================] - 0s 2ms/step
Pred Shape: (658, 2)
************************** Prediction Completed **************************
First 5 predictions: (True, Prediction):
[(1, 1), (0, 0), (0, 0), (0, 0), (1, 1)]
Accuracy score: 0.8875379939209727
Precision score: 0.8875379939209727
Recall score: 0.8875379939209727
F1 score: 0.8875379939209727
```

Confusion Matrix with Hyperparameter Tuning	**ROC Curve with Hyperparameter Tuning**
	

Overall, in this project we implemented logistic regression, nearest neighbor, SVM and Neural Network with an unbalanced and a balanced dataset while implementing hyperparameter tuning for neural networks. We evaluated each model based on 5 categories: ROC curve, accuracy, f1-score, precision recall and confusion matrices. We also used a tensorboard to display each dense hidden layer.