Detecting Stroke Using Machine Learning

Abstract

Stroke is the 2nd leading cause of death according to the World Health Organisation. A stroke occurs when the blood supply to the brain is interrupted, which will cause brain cells to die in a short period of time. A stroke is a medical emergency that causes a lot of people to pass away unexpectedly. However, stroke patients always shared some similar characteristics. Moreover, early action can reduce brain damage and prevent disability and death.

The dataset we used in this project collected a series of input parameters like gender, age, disease history, and smoking status. In this machine learning project, our objective is to choose the optimal model based on patients' data to successfully detect stroke in advance.

Introduction

Above we have described the severity of the consequences of being diagnosed as a stroke. However, about 80% of strokes are preventable according to medical research. High blood pressure is the single most important treatable risk factor for stroke. We can encourage patients to change their lifestyles and take medicine to reduce the risk of stroke.

In this background, we took measures to diagnose stroke in the machine learning dimension. We chose the topic of stroke prediction, which may effectively prevent physical risks and deaths caused by this serious disease. A machine learning approach may prove beneficial for establishing a relationship between the process parameters. Therefore, in this project, we have applied a machine learning classification-based approach.

More specifically, we collected the most distinguished factors related to stroke, built several statistical models, and tried to predict stroke occurrence, after which we also performed model selection in order to reach the optimal result.

Like most analysts do in the first step, this project began by giving an overview of the database. In the following data cleaning procedure, it is worth noting that after dealing with null values and transforming categorical variables into numeric data, we also used Synthetic Minority Oversampling Technique (SMOT), which synthesized new examples for the minority class. Then we respectively built Logistic Regression, SVM Model, Decision Tree, Random Forest, KNN Model, Neural Network, and XGBoost Model. Furthermore, we plotted evaluation statistics (accuracy, precision, recall, F1), ROC curves, and confusion matrices for every single model.

In the final evaluating stage, we focused on accuracy and recall. By comparing these two key targeted values, we concluded that the Decision Tree model has reached the best performance among the 6 models.

The main critical and developing point of this project is that it keeps up with the macroscopic trend of the world’s increasing attention on personal care and health issues. By selecting different models and giving a final answer, we give a rigorous analysis of this issue. However, there is still some room for improvement in our models due to the relatively small size of the dataset.

Task Description

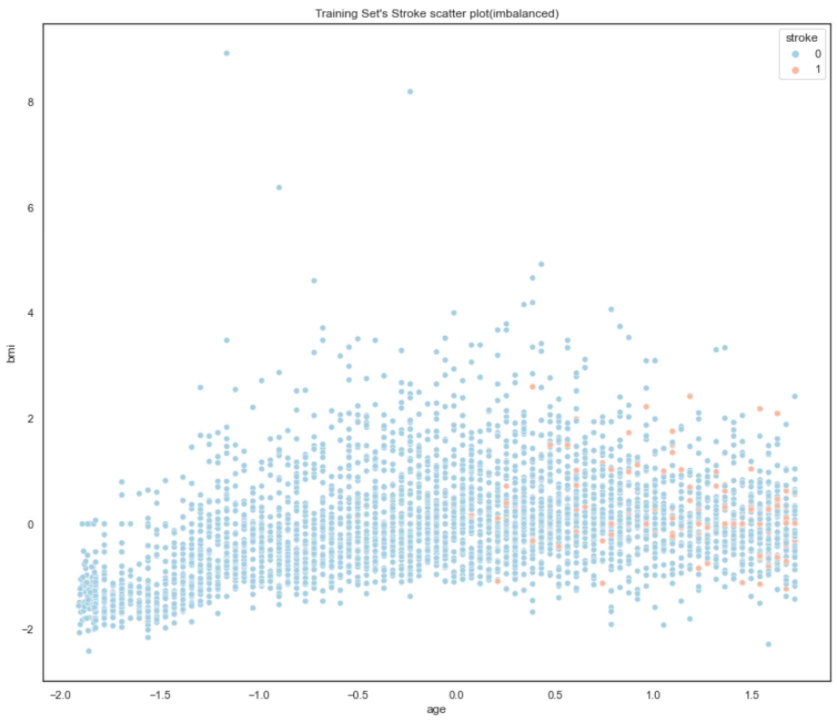
We want to finish a course project using machine learning techniques we learned this quarter from the ECON425 class. Given enough freedom to choose a topic, we want to have a deeper understanding of how machine learning is transforming clinical decisions. Among a wide variety of machine learning applications used in the industry of healthcare, we are impressed that machine learning actually effectively and efficiently provides countless people with care.

We found a comprehensive dataset related to our interest of study on Kaggle named "Stroke Prediction Dataset" which recorded 11 clinical features of 5110 patients. More information about this dataset will be covered in the Experiment part. Stroke has been regarded as a hard-to-diagnose disease. We hope to develop a trustworthy model to diagnose stroke and provide clinical treatment in advance with the help of intelligence and the convenience of machine learning solutions. In conclusion, the major task of our project is to find the optimal model with the best evaluation metrics to predict whether a patient will have a stroke or not.

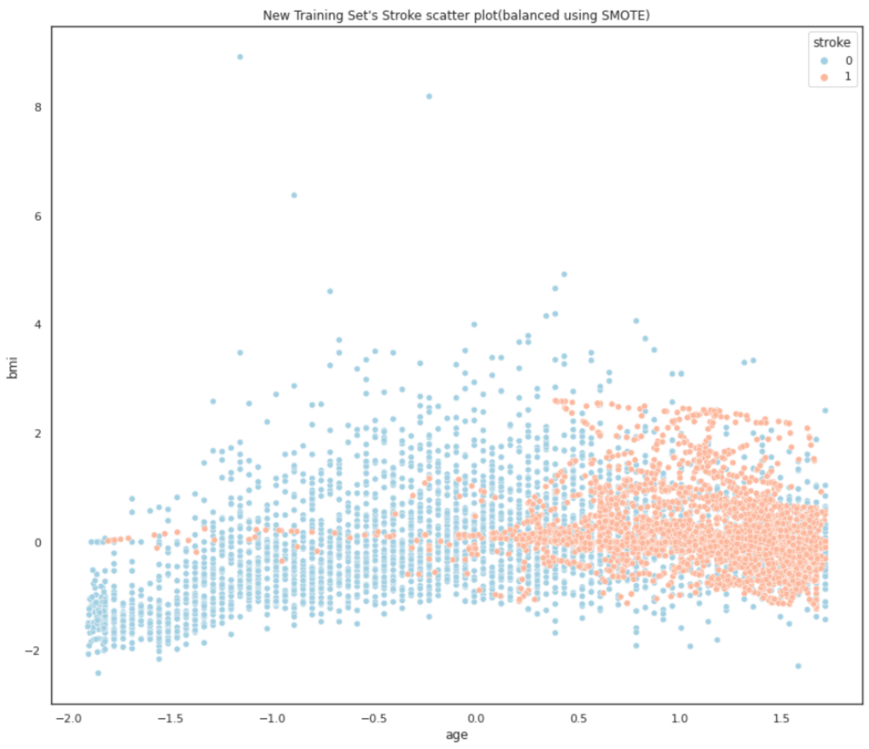
Major Challenges and Solutions

1. Challenge and solution in the data cleaning part

When our group was doing this machine learning project, we met several challenges. First of all, the data cleaning part was not processing as we expected. We handled missing values and drop irrelevant columns as usual. Then, we located categorical data and numeric data separately. As for the numeric data, we used StandardScaler to standardize numeric values. As for categorical data, we used LabelEncoder to transform the labels into a numeric form. After splitting the data into the training set and validation set, we disappointedly found that we had an imbalanced classification.



Our project was temporarily suspended, as the prediction won't be precise or accurate with such an imbalanced classification. We did research on how to balance data, and there was an article named "Overcoming Class Imbalance Using SMOTE Techniques". The SMOTE algorithm was used to randomly generate synthetic samples for the minority class, which addressed the issue of oversampling. We got inspiration from this article. After applying the SMOTE algorithm to further preprocess the dataset, we got a much more balanced scatter plot of stroke classification. The new plot is shown below. (https://www.analyticsvidhya.com/blog/2020/10/overcoming-class-imbalance-using-smote-techniques/).



2. Challenge and solution in the hyperparameter optimization

In the modeling part, we tried seven different models in total. They were logistic regression, SVM model, decision tree, random forest, KNN model, neural network, and XGBosst model.

At the very beginning, all of us agreed that we should use GridSearch to choose the optimal parameters for each model. After specifying a grid of parameter values, GridSearch exhaustively generated all possible parameter combinations. All parameter values are evaluated to tune the best combination.

Gradually, we realized that we needed to make some adjustments because the GridSearch method was too time-consuming. Although the result would be more accurate and precise, the run time negatively affected our group's efficiency. We faced a tradeoff between time and quality, and we all agreed that sometimes we were supposed to sacrifice some quality for more time. Thus, we decided to use the RandomizedSearch method for the SVM model and neural network model. The RandomizedSearch method might have slightly worse performance in hyperparameter optimization, but the run time was substantially lower than the GridSearch's. For the rest of the models, we decided to continue using the GridSearch method. In this way, we used either GridSearch or RandomizedSearch to choose the best parameters for each model in the modeling part to balance the time and quality.

(https://scikit-learn.org/stable/modules/grid_search.html)

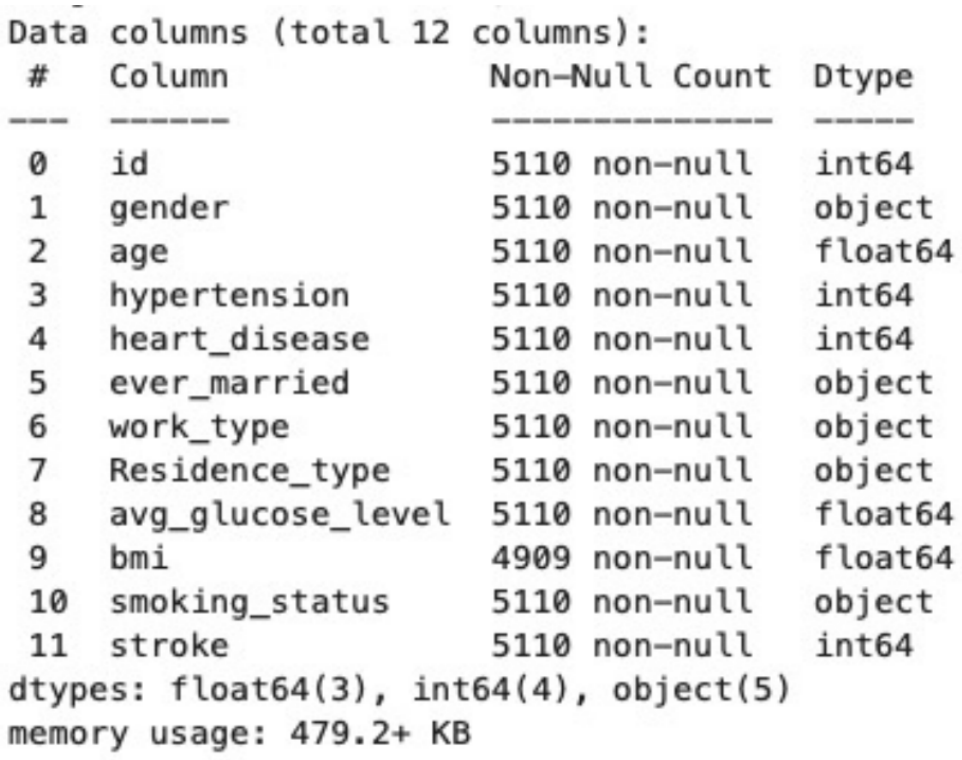
3. **Challenge and solution in the setup of neural network.**
In the process of setting up the neural network model, we got errors in the prediction part. We guessed the reason was that our categorical set was not properly converted or handled. We studied similar machine learning projects online, one common thing they did for the neural network model was to use the One-Hot Encoding algorithm to improve the classification accuracy. Thus, we did One-Hot Encoding to convert categorical data into a united form to have a better prediction.
(https://hackernoon.com/what-is-one-hot-encoding-why-and-when-do-you-have-to-use-it-e3c6186d008f)

Experiment Implementation

Dataset Description

https://www.kaggle.com/fedesoriano/stroke-prediction-dataset

This section reports several statistics about the data set. We used a dataset named "Stroke Prediction Dataset" in Kaggle. This dataset is used to predict whether a patient is likely to get a stroke based on the input parameters like gender, age, various diseases, and smoking status. As the table shows, each row in the data provides relevant information about the patient.



```
Data columns (total 12 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   id                 5110 non-null   int64
 1   gender             5110 non-null   object
 2   age                5110 non-null   float64
 3   hypertension       5110 non-null   int64
 4   heart_disease      5110 non-null   int64
 5   ever_married       5110 non-null   object
 6   work_type          5110 non-null   object
 7   Residence_type     5110 non-null   object
 8   avg_glucose_level  5110 non-null   float64
 9   bmi                4909 non-null   float64
 10  smoking_status     5110 non-null   object
 11  stroke             5110 non-null   int64
dtypes: float64(3), int64(4), object(5)
memory usage: 479.2+ KB
```

Kaggle also provides Attribute Information about the variables:
1) id: unique identifier
2) gender: "Male", "Female" or "Other"
3) age: age of the patient
4) hypertension: 0 if the patient doesn't have hypertension, 1 if the patient has hypertension

5) heart_disease: 0 if the patient doesn't have any heart diseases, 1 if the patient has a heart disease
6) ever_married: "No" or "Yes"
7) work_type: "children", "Govt_jov", "Never_worked", "Private" or "Self-employed"
8) Residence_type: "Rural" or "Urban"
9) avg_glucose_level: average glucose level in blood
10) BMI: body mass index
11) smoking_status: "formerly smoked", "never smoked", "smokes" or "Unknown"*
12) stroke: 1 if the patient had a stroke or 0 if not

*Note: "Unknown" in smoking_status means that the information is unavailable for this patient

In the actual study, we deleted the variable “id”, because it is irrelevant to our model building.

Model design

After selecting the dataset, we need to decide what models to use then.

In this project, our aim is mainly to make clinical decisions about if a person has a stroke or not using machine learning, so we basically will use the seven different models to deal with this problem:
Logistic Regression Model, SVM Model, Decision Tree Model, Random Forest Model, KNN Model, Neural Network Model, and XGBoost Model.

We also defined several functions for further use:
evaluation(model,xtest,ytest) and **nnevaluation(model,xtest,ytest)**: These two functions are to return each model’s accuracy, precision, recall and F1 score.
scores(model,xtest,ytest) and **nnscores(model,xtest,ytest)**: These two functions are to return model’s evaluation stats and confusion matrix.
roc(model,xtest,ytest): This function is to return each model’s ROC curve.
gridsearch(model,params_grids,xtest,ytest): This function is to do grid search on ammetersers of each model based on xtrain and ytrain data.
randomsearch(model,params_grids,xtest,ytest): This function is to do random search on parameters of each model based on xtrain and ytrain data.

Evaluation Metrics

While it is important to design the model, it is more crucial to properly evaluate the machine learning model using different metrics. By analyzing evaluation metrics, we will get feedback on how to improve the model to achieve better performance. In the case of detecting stroke for patients, we choose to use confusion matrix, F1 score, and AUC-ROC as the evaluation metrics to test models in this project.

As for the confusion matrix, the 2x2 matrix illustrates accuracy, precision, and recall. More importantly, we need to put more emphasis on both the accuracy score and the recall score. In the process of disease diagnosis, the cost of False Negative is very high. There is no doubt that we don’t want to wrongly label a stroke patient to be a non-stroke patient. The consequence of False Negative might cause deaths and injuries. However, if we label a non-stroke patient as a stroke patient, the cost is relatively low. In short, we need to look at the recall score to better detect a stroke.

As for the F1 score, it is the harmonic mean between precision and recall. The F1 score tells us whether the model has a precise and robust performance or not. Normally, we want to see a large F1 score. The greater the value of the F1 score, the better is the performance.

As for the AUC-ROC, it is one of the most popular evaluation metrics used in machine learning. The ROC curve has the true positive on the y-axis, and it has the false positive on the x-axis. The more the ROC curve will be occupied in the upper left corner, the better is the performance of the machine learning model, which also means a higher AUC value. In the same way, we want to see a large AUC value.

https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234

Result Analysis

1) Logistic Regression Model

Advantage: 1. The predicted result is a probability bounded between 0 and 1; 2. It can be applied to continuous and categorical independent variables; 3. Easy to use and explain.

Disadvantage: 1. It is more sensitive to the multicollinearity of independent variables in the model. For example, two highly correlated independent variables are put into the model at the same time, which may cause the regression symbol of the weaker independent variable to not meet expectations; 2. The prediction result is in an "S" shape, so the process of converting from log(odds) to probability is non-linear. At both ends, with the change of the log(odds) value, the probability changes very little, the marginal value is too small, the slope is too small, and the change in the intermediate probability is large and sensitive. The influence of variable changes in many intervals on the target probability has no discrimination, and the threshold value cannot be determined.

Our result using Logistic Regression Model:

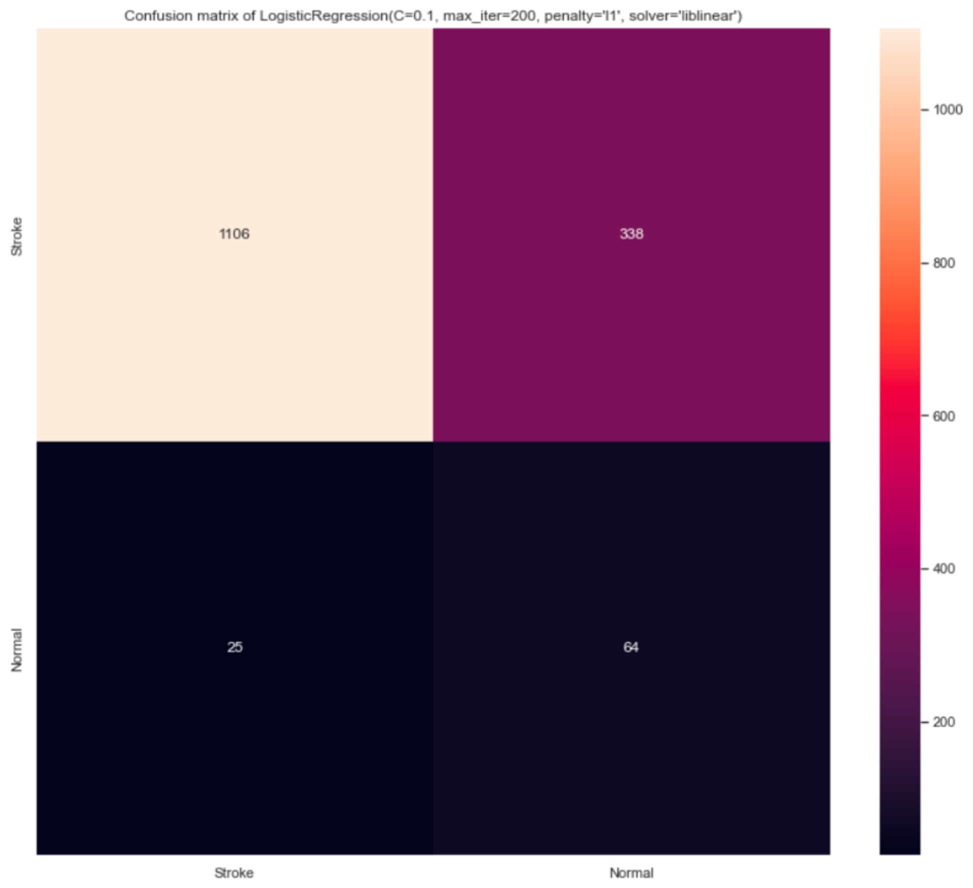
```
The Result of Logistic Regression is

Acuracy     ========>>> 0.743


Precision ========>>> 0.159


Recall    ========>>> 0.719


F1        ========>>> 0.261
```



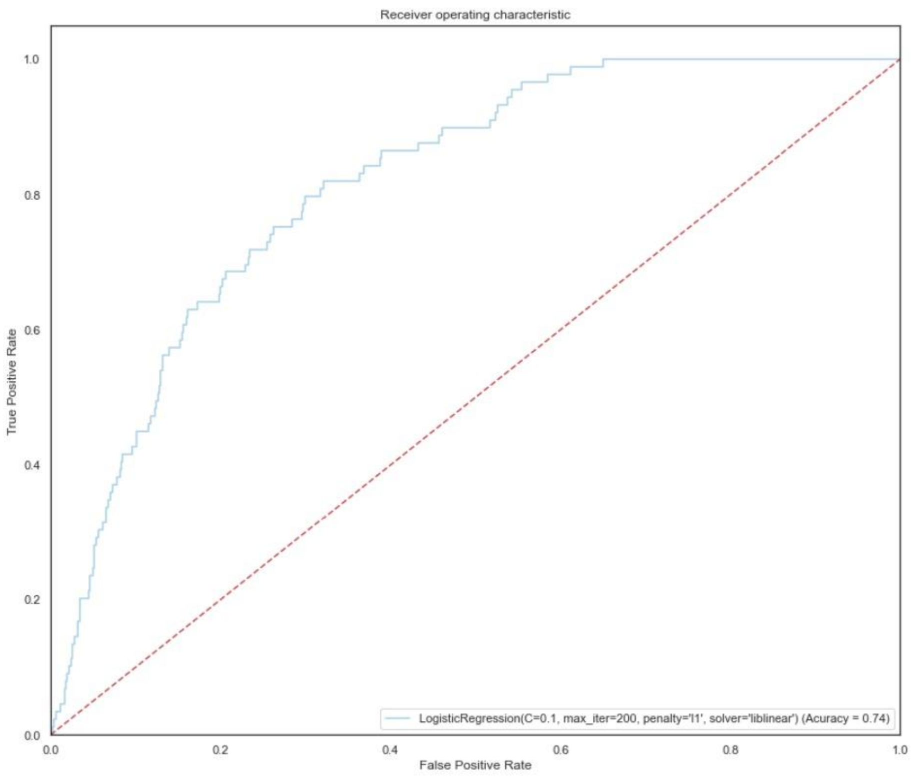
As can be seen from the result, the recall is 0.719, which means that it is about 71.9% that we can predict the patient has a stroke while he/she actually does have.

The accuracy is 0.743, which means the possibility that we made the correct prediction is around 74.3%.

The precision is 0.159, which means that from all the patients we have predicted to have a stroke, there are around 15.9% of patients actually do have a stroke.

And our Logistic Regression Model's F1 score, which takes both recall and precision into consideration, is 0.21.

The ROC curve of our Logistic Regression Model is as followed:



2) SVM Model

Advantage: 1. It can solve machine learning problems in the case of small samples; 2. It can improve the generalization performance; 3. It can solve high-dimensional problems; 4. It can solve nonlinear problems; 5. It can avoid the problem of neural network structure selection and local minimum points.

Disadvantage: 1. It is sensitive to missing data; 2. There is no general solution to nonlinear problems, Kernel function must be carefully selected to deal with it.

Our result using Logistic Regression Model:

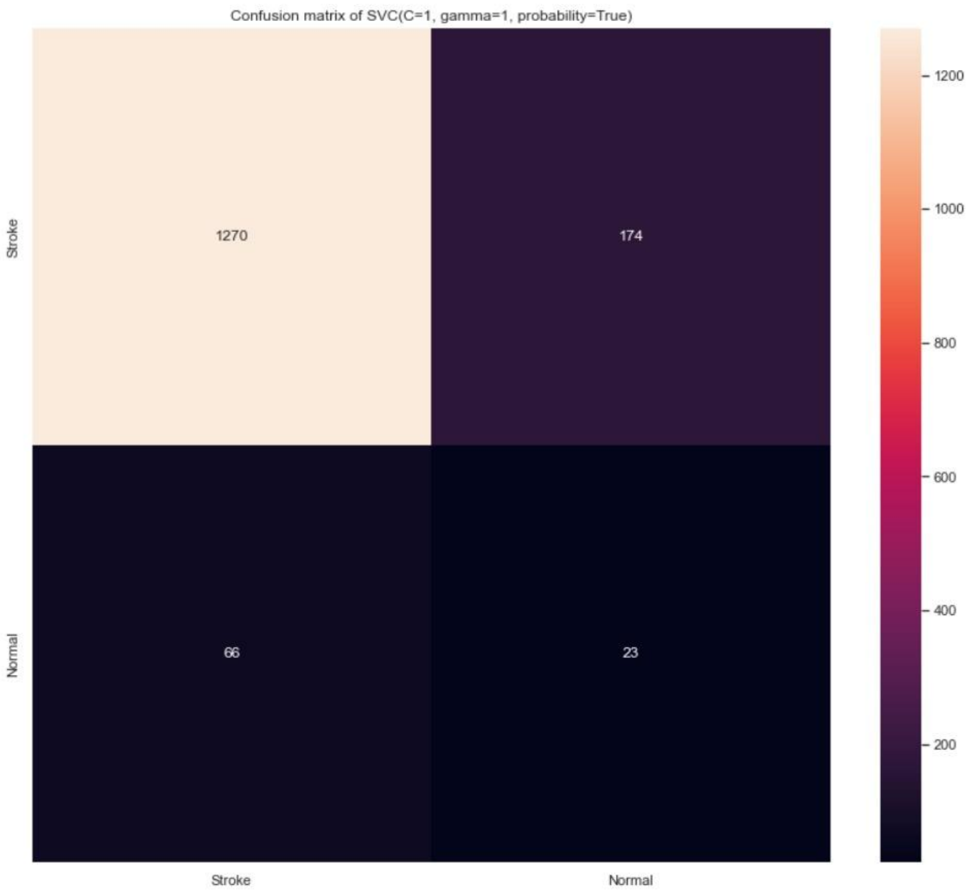
```
The Result of SVM Model is

Acuracy     ========>>> 0.569


Precision ========>>> 0.117


Recall    ========>>> 0.258


F1        ========>>> 0.161
```



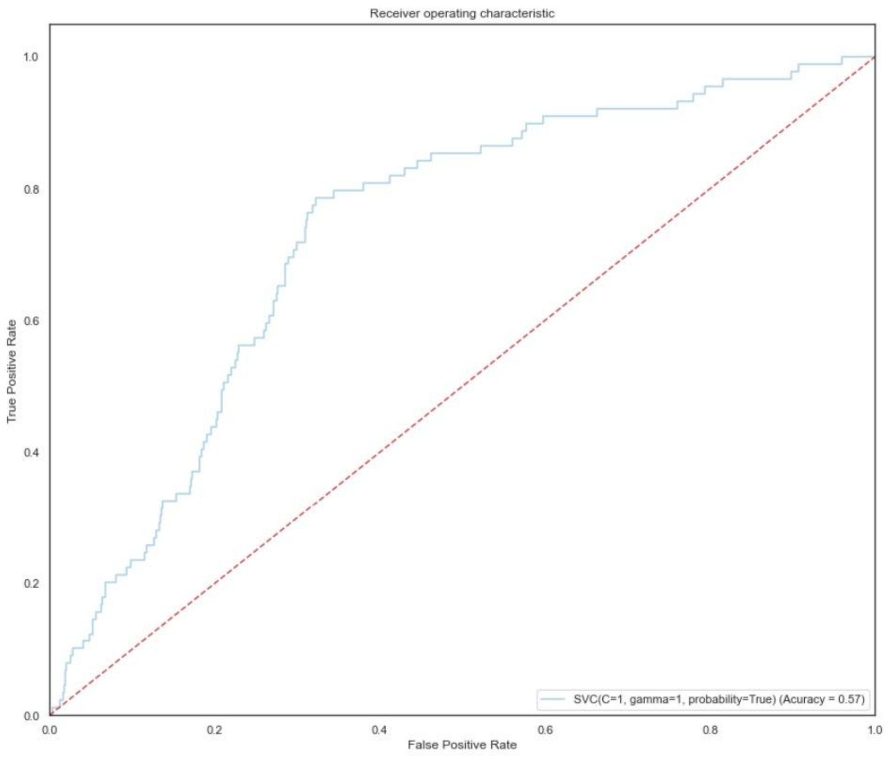
As can be seen from the result, the recall is 0.258, which means that it is about 25.8% that we can predict the patient has a stroke while he/she actually does have.

The accuracy is 0.569, which means the possibility that we made the correct prediction is around 56.9%.

The precision is 0.117, which means that from all the patients we have predicted to have a stroke, there are around 11.7% of patients actually do have a stroke.

And our Logistic Regression Model’s F1 score, which takes both recall and precision into consideration, is 0.161.

The ROC curve of our SVM Model is as followed:



3) Decision Tree Model

Advantage: 1. Decision trees are easy to understand and explain; 2. The data preparation is relatively simple; 3. It can handle both data-type and general-type attributes at the same time. Other techniques often require a single data attribute; 4. The decision tree is a white-box model. Given an observed model, it is easy to derive the corresponding logical expression from the resulting decision tree; 5. It is easy to evaluate the model through static testing. Indicates that it is possible to measure the confidence of the model; 6. Be able to produce feasible and effective results on large data sources in a relatively short period of time; 7. Decision trees can be constructed for datasets with many attributes; 8. Decision trees scale well to large databases, and their size is independent of the size of the database.

Disadvantage: 1. For data with inconsistent numbers of samples in each category, in the decision tree, the result of information gain is biased towards those features with more numerical values; 2. Difficulties in dealing with missing data; 3. The emergence of overfitting problem; 4. Ignore the correlation between attributes in the dataset.

Our result using Decision Tree Model:

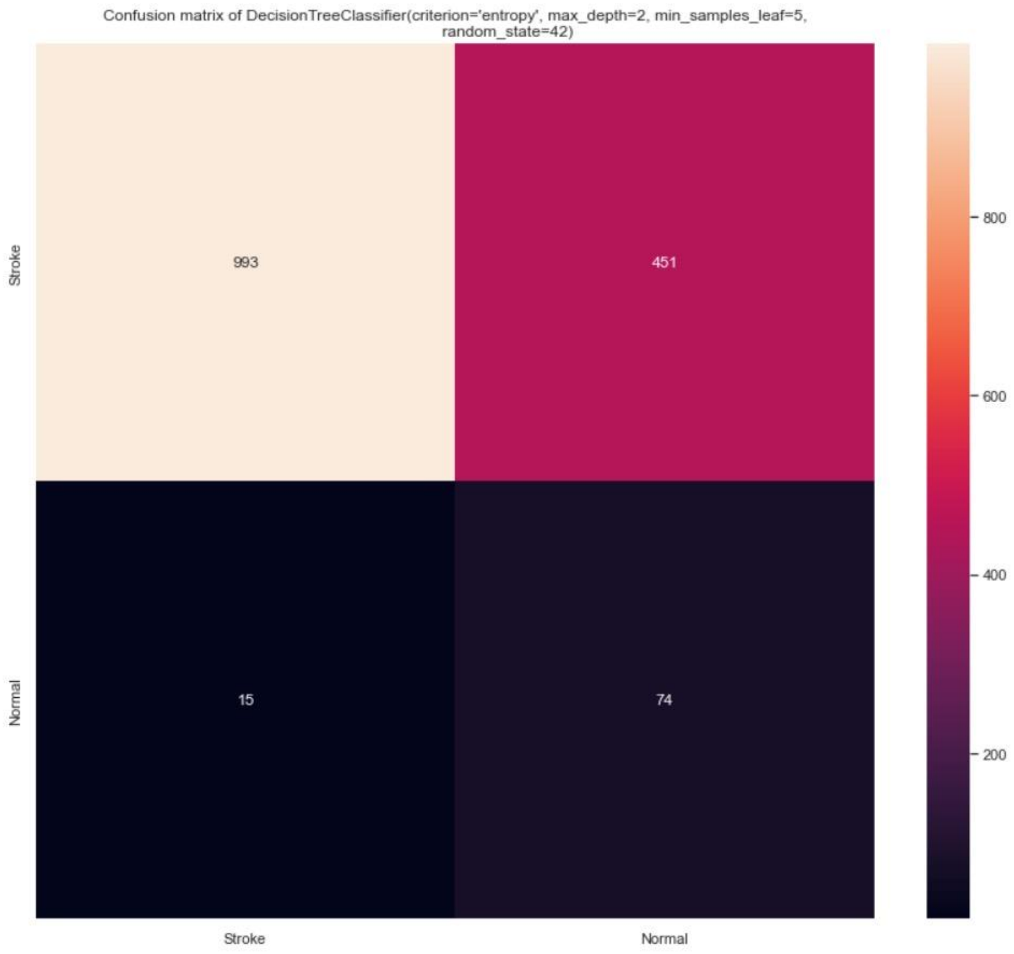
```
The Result of Decision Tree is

Acuracy     ========>>> 0.760


Precision ========>>> 0.141


Recall    ========>>> 0.831


F1        ========>>> 0.241
```



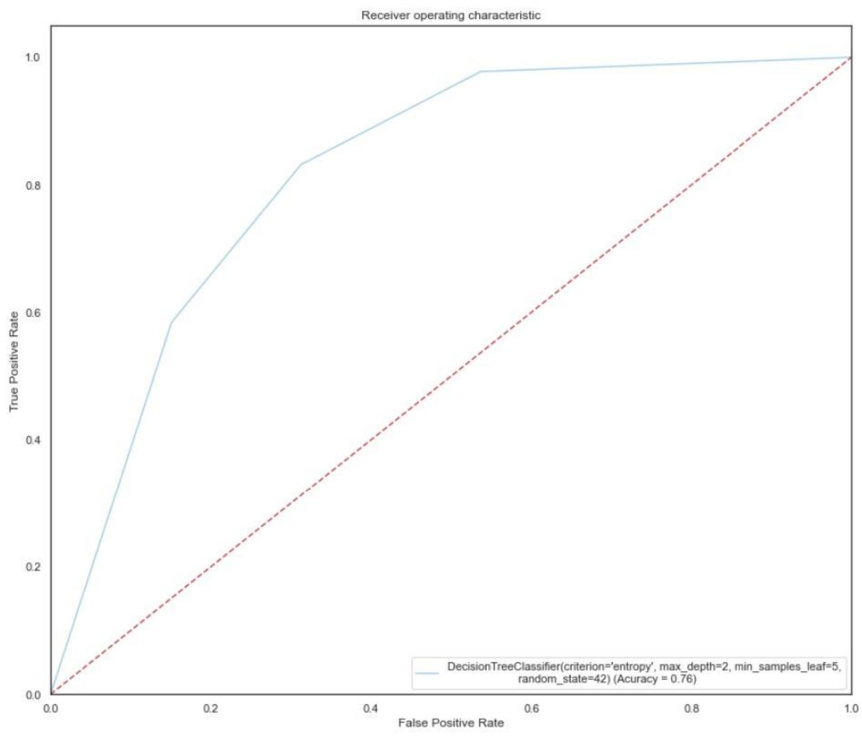
As can be seen from the result, the recall is 0.831, which means that it is about 83.1% that we can predict the patient has a stroke while he/she actually does have.

The accuracy is 0.76, which means the possibility that we made the correct prediction is around 76%.

The precision is 0.141, which means that from all the patients we have predicted to have a stroke, there are around 14.1% of patients actually do have a stroke.

And our Logistic Regression Model's F1 score, which takes both recall and precision into consideration, is 0.241.

The ROC curve of our Decision Tree Model is as followed:



4) Random Forest Model

Advantage: 1. On many current data sets, it has great advantages over other algorithms and performs well; 2. It can process data with very high dimensions (many features), and does not need to do feature selection; 3. After training, it can give which features are more important; 4. When creating a random forest, an unbiased estimate is used for the generalization error, and the model has a strong generalization ability; 5. The training speed is fast, and it is easy to make a parallel method; 6. During the training process, the interaction between features can be detected; 7. The implementation is relatively simple; 8. For imbalanced datasets, it can balance the error; 9. If a large part of the features is missing, the accuracy can still be maintained.

Disadvantage: 1. Random forests have been shown to overfit in some noisy classification or regression problems; 2. For data with attributes with different values, attributes with more value divisions will have a greater impact on random forests, so the attribute weights produced by random forests on such data are not credible.

Our result using Random Forest Model:

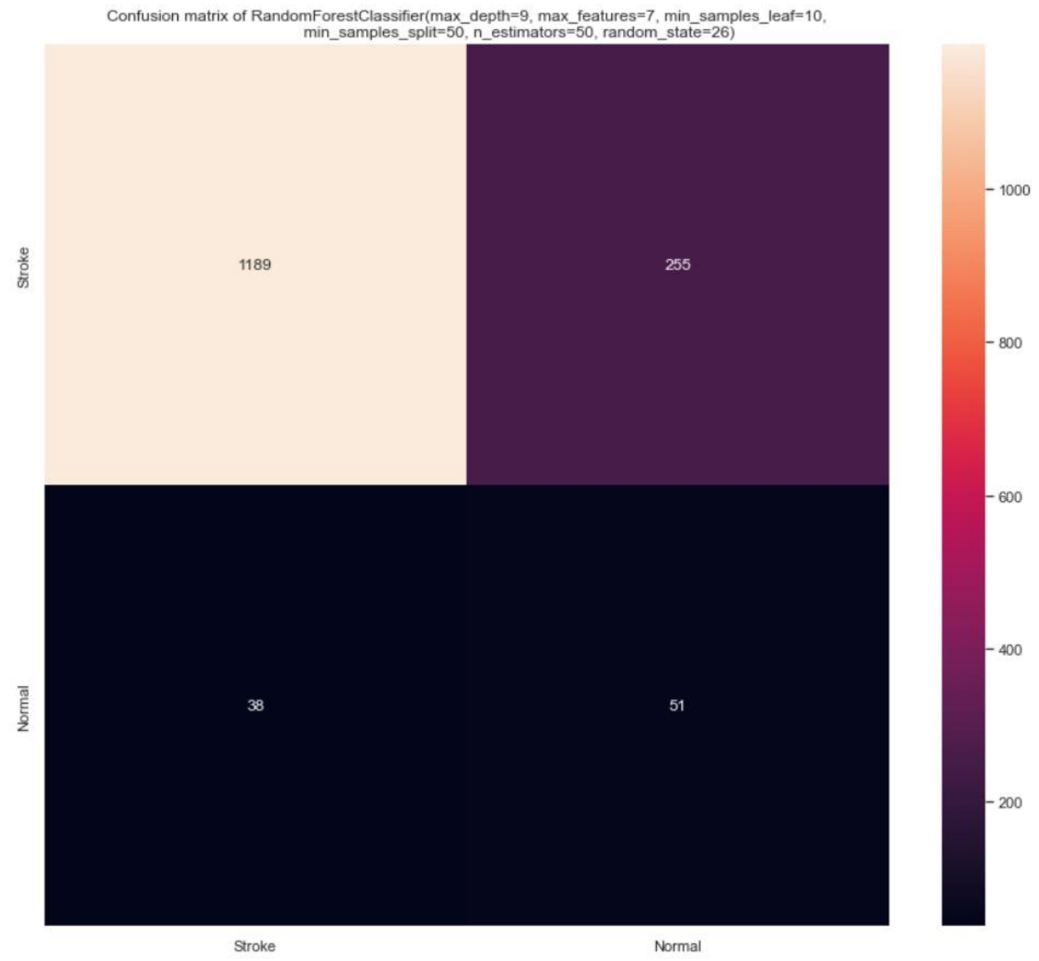
```
The Result of Random Forest is

Acuracy     ========>>> 0.698


Precision ========>>> 0.167


Recall    ========>>> 0.573


F1        ========>>> 0.258
```



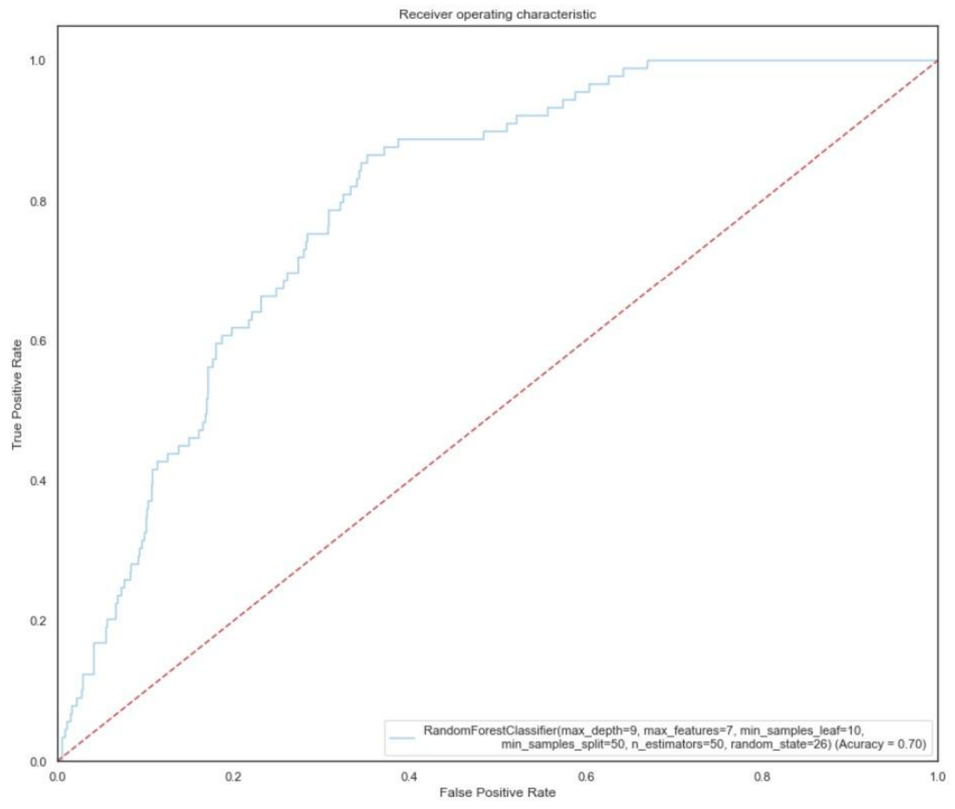
As can be seen from the result, the recall is 0.573, which means that it is about 57.3% that we can predict the patient has a stroke while he/she actually does have.

The accuracy is 0.698, which means the possibility that we made the correct prediction is around 69.8%.

The precision is 0.167, which means that from all the patients we have predicted to have a stroke, there are around 16.7% of patients actually do have a stroke.

And our Logistic Regression Model’s F1 score, which takes both recall and precision into consideration, is 0.258.

The ROC curve of our Random Forest Model is as followed:



5) KNN Model

Advantage: 1. Simple and effective; 2. The cost of retraining is low; 3. Computational time and space are linear to the size of the training set (not too large in some cases); 4. Since the KNN method mainly relies on the limited surrounding samples, rather than the method of discriminating the class domain to determine the class to which it belongs, for the sample set to be divided that has more intersections or overlaps in the class domain, the KNN method is better than other methods. the method is more suitable; 5. The algorithm is more suitable for automatic classification of the class domain with relatively large sample size.

Disadvantage: 1. KNN algorithm is a lazy learning method; 2. Category scores are not normalized; 3. The interpretability of the output is not strong; 4. When the samples are unbalanced, for example, the sample capacity of one class is large, while the sample capacity of other classes is very small, it may cause that when a new sample is an input, the samples of the large-capacity class among the K neighbors of the sample are the majority; 5. The amount of calculation is large.

Our result using KNN Model:

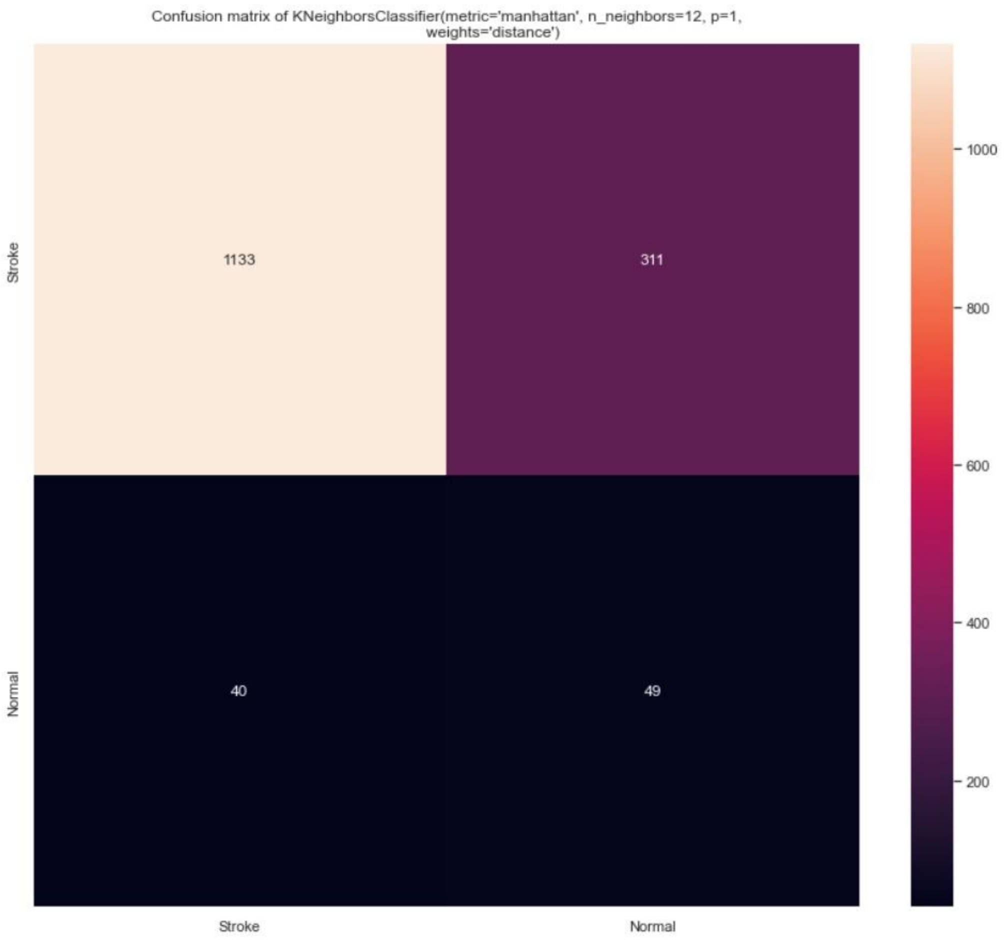
```
The Result of KNN Model is

Acuracy     ========>>> 0.668


Precision ========>>> 0.136


Recall    ========>>> 0.551


F1        ========>>> 0.218
```



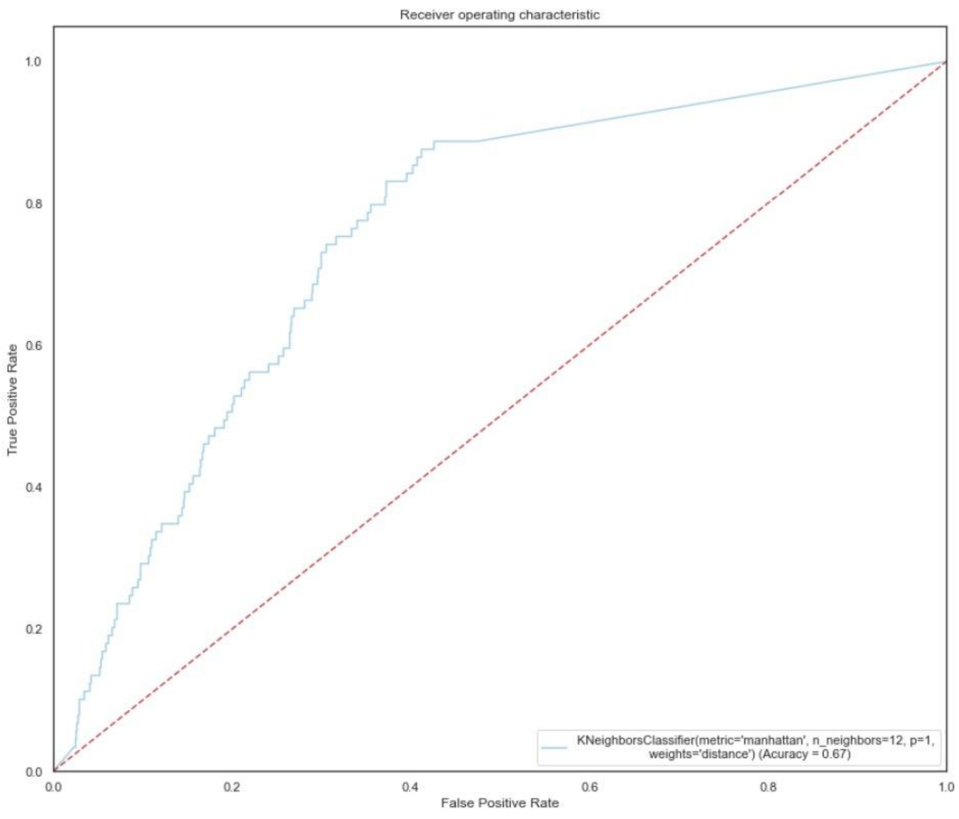
As can be seen from the result, the recall is 0.551, which means that it is about 55.1% that we can predict the patient has a stroke while he/she actually does have.

The accuracy is 0.668, which means the possibility that we made the correct prediction is around 66.8%.

The precision is 0.136, which means that from all the patients we have predicted to have a stroke, there are around 13.6% of patients actually do have a stroke.

And our Logistic Regression Model's F1 score, which takes both recall and precision into consideration, is 0.218.

The ROC curve of our KNN Model is as followed:



6) Neural Network Model

Advantage: It has high classification accuracy, strong parallel distributed processing ability, strong distributed storage and learning ability, strong robustness and fault tolerance to noise nerves, can fully approximate complex nonlinear relationships, and has the function of associative memory.

Disadvantage: Neural networks require a large number of parameters, such as network topology, weights, and initial values of thresholds; the learning process cannot be observed, and the output results are difficult to explain, which will affect the credibility and acceptability of the results; the learning time is too long, may not even achieve the purpose of learning.

Our result using Neural Network Model:

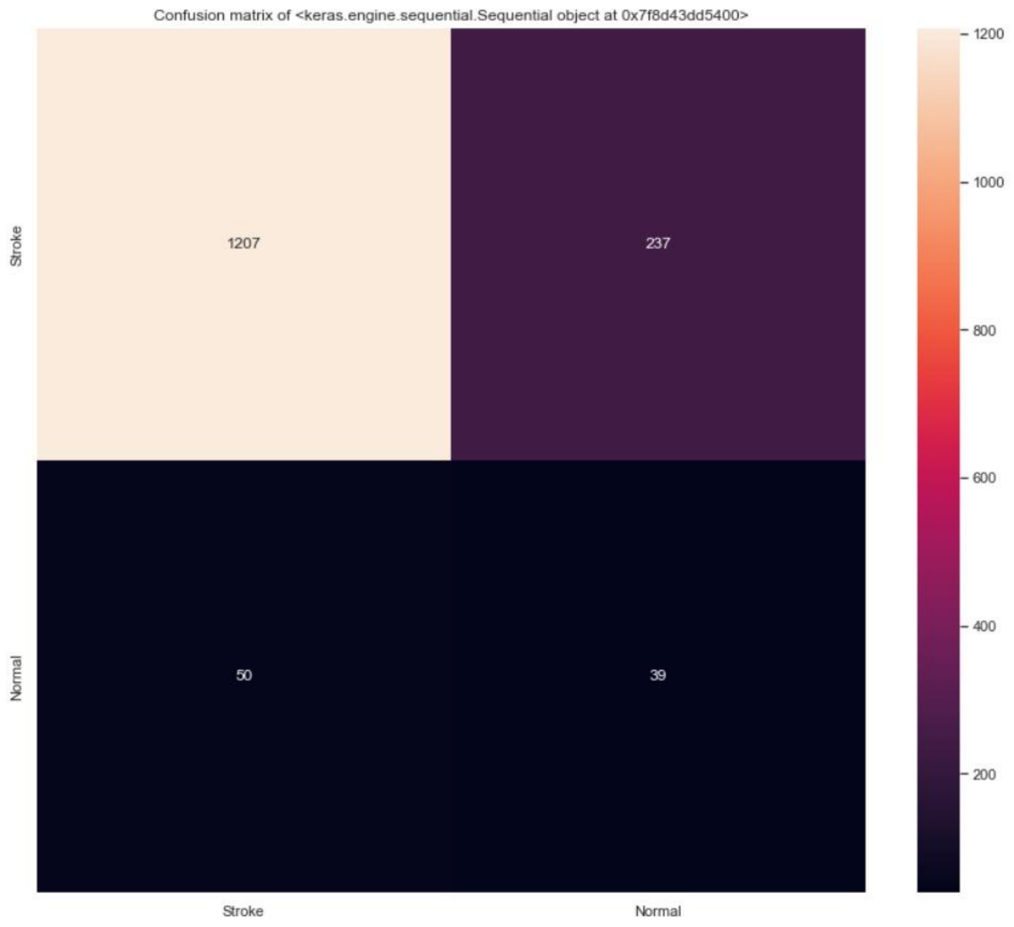
```
The Result of Neural Network is

Acuracy      ========>>> 0.637


Precision ========>>> 0.141


Recall     ========>>> 0.438


F1         ========>>> 0.214
```



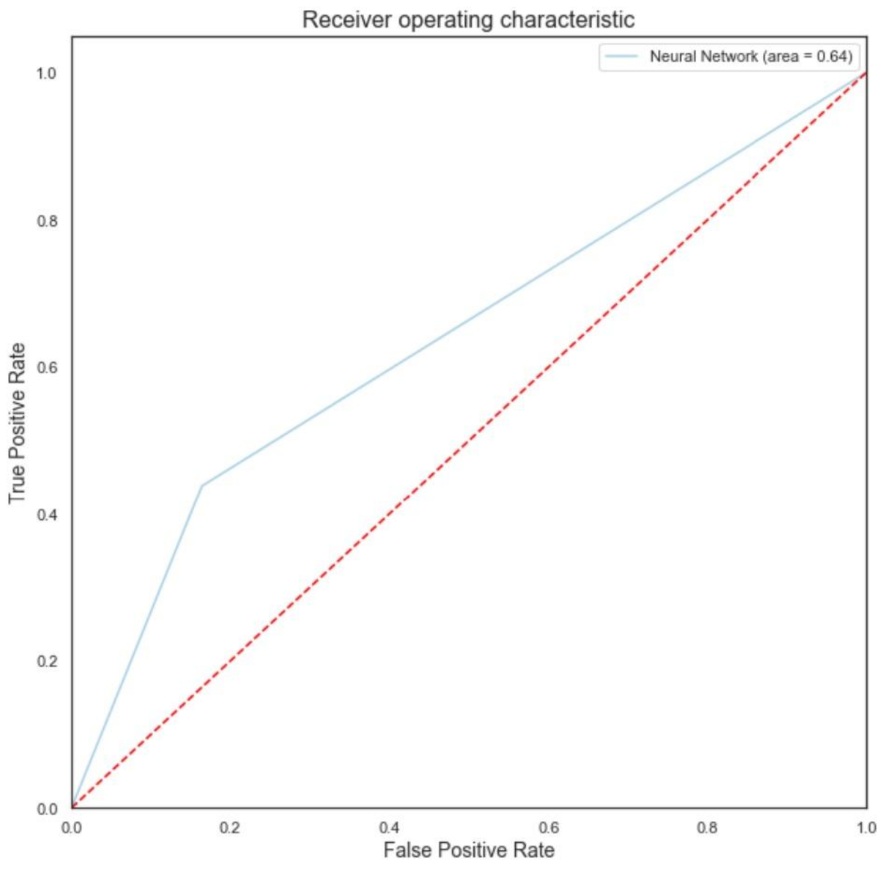
As can be seen from the result, the recall is 0.438, which means that it is about 43.8% that we can predict the patient has a stroke while he/she actually does have.

The accuracy is 0.637, which means the possibility that we made the correct prediction is around 63.7%.

The precision is 0.141, which means that from all the patients we have predicted to have a stroke, there are around 14.1% of patients actually do have a stroke.

And our Logistic Regression Model's F1 score, which takes both recall and precision into consideration, is 0.214.

The ROC curve of our Neural Network Model is as followed:



7) XGBoost Model

Our result using Neural Network Model:

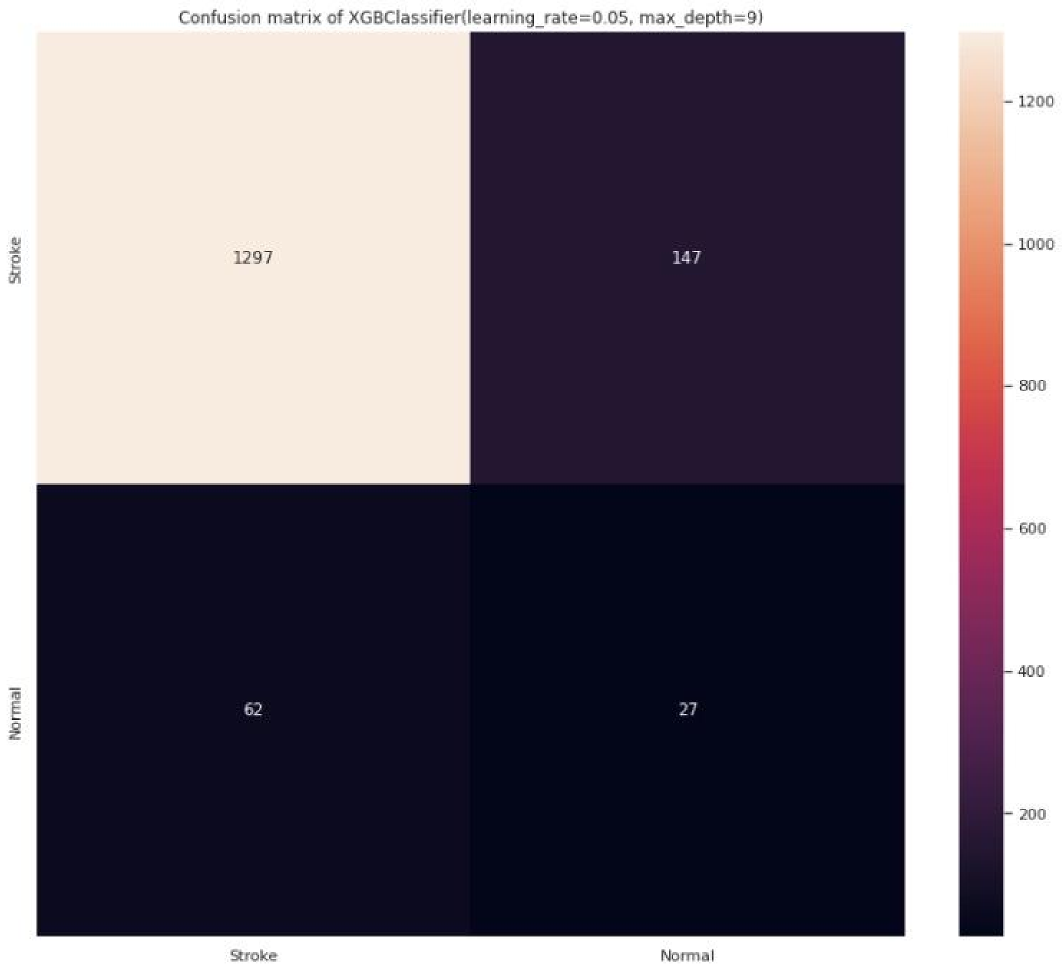
```
The Result of XGBoost is

Acuracy     ========>>> 0.601


Precision ========>>> 0.155


Recall    ========>>> 0.303


F1        ========>>> 0.205
```



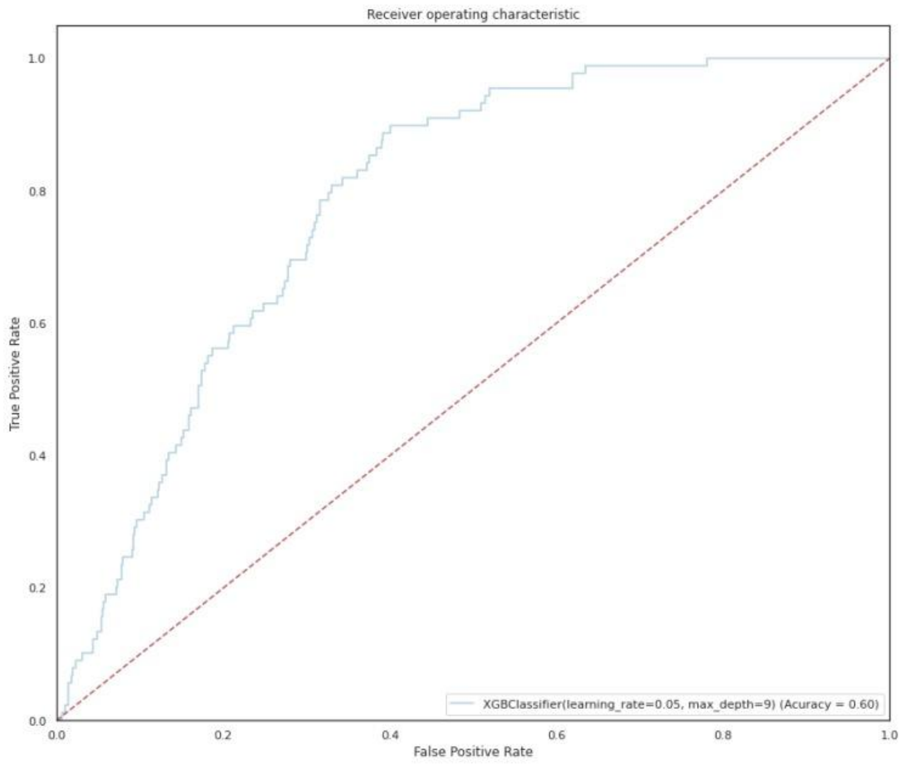
As can be seen from the result, the recall is 0.303, which means that it is about 30.3% that we can predict the patient has a stroke while he/she actually does have.

The accuracy is 0.601, which means the possibility that we made the correct prediction is around 60.1%.

The precision is 0.155, which means that from all the patients we have predicted to have a stroke, there are around 15.5% of patients actually do have a stroke.

And our Logistic Regression Model’s F1 score, which takes both recall and precision into consideration, is 0.205.

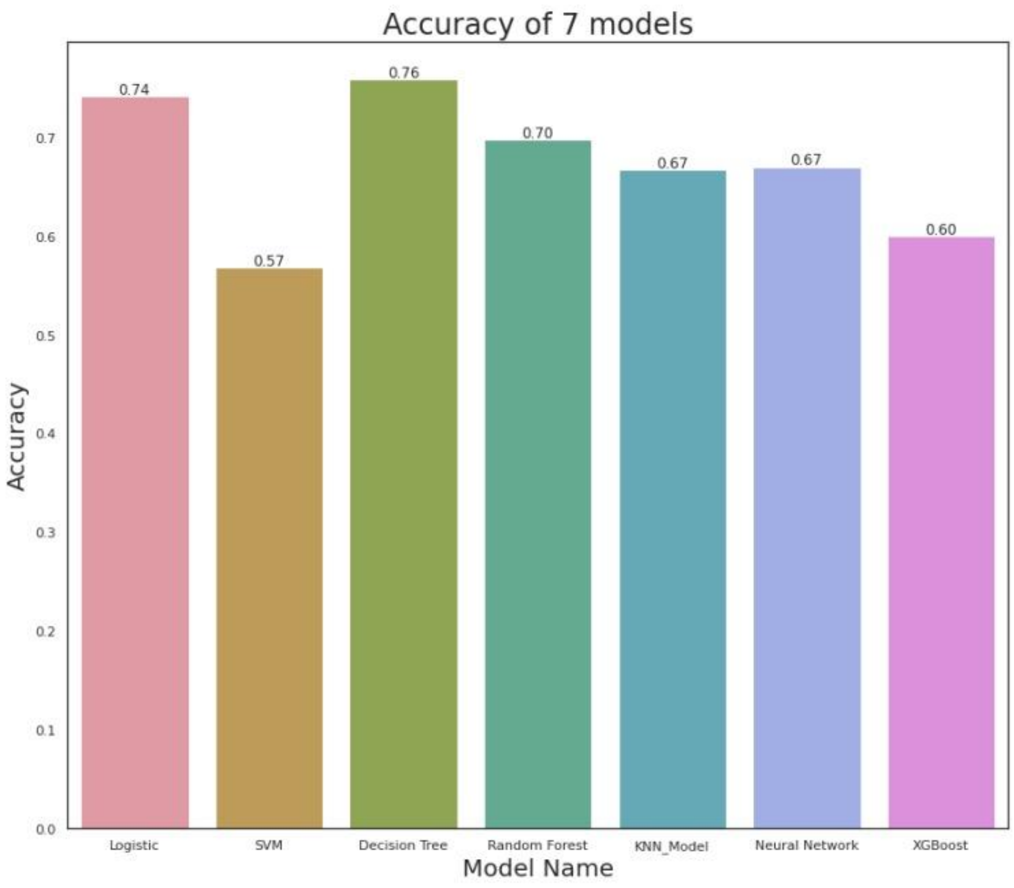
The ROC curve of our Neural Network Model is as followed:

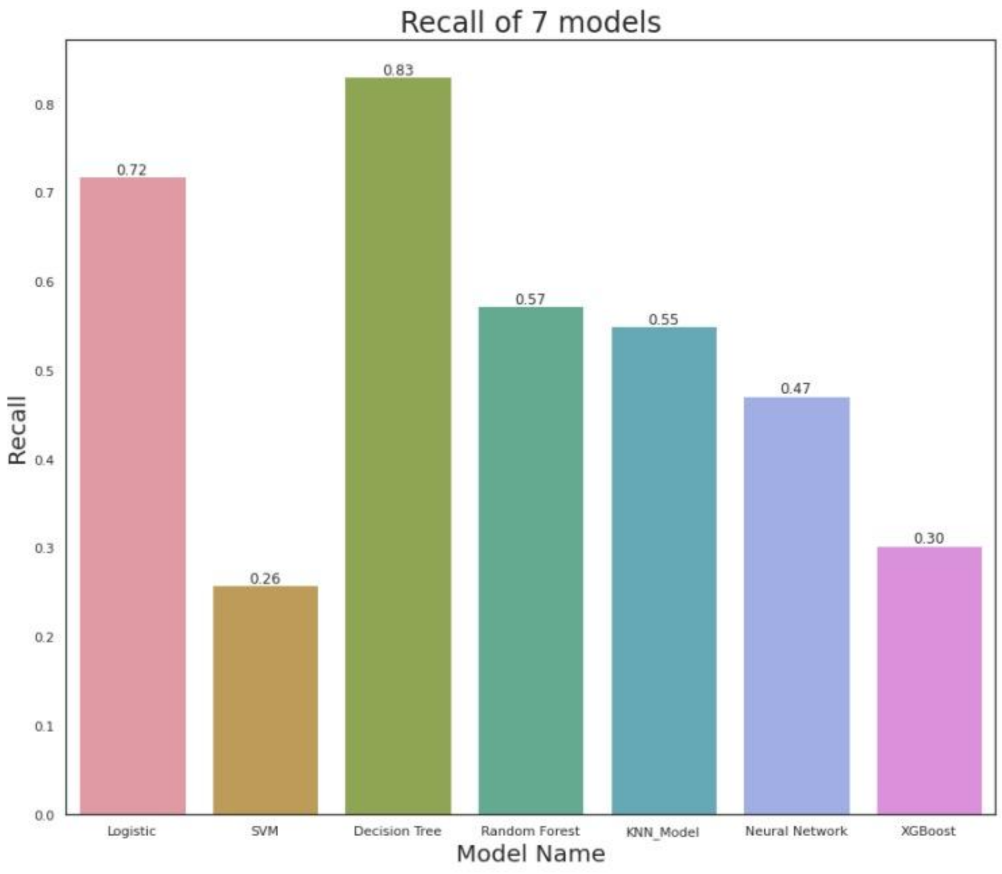


Conclusion and Future Works

Model Name	Auc	Precision	Recall	F1
Logistic	0.742515	0.159204	0.719101	0.260692
SVM	0.568964	0.116751	0.258427	0.160839
Decision Tree	0.759567	0.140952	0.831461	0.241042
Random Forest	0.698220	0.166667	0.573034	0.258228
KNN_Model	0.667594	0.136111	0.550562	0.218263
Neural Network	0.670858	0.182609	0.471910	0.263323
XGBoost	0.600785	0.155172	0.303371	0.205323

Among all the models stated above, we can clearly see that the Decision Tree model has the best Accuracy and Recall, and is much better than other models. The Random Forest model has the best Precision and F1 score, but just a little bit higher than the Decision Tree model. The ROC curve of the Neural Network Model is the worst, other models' ROC curves are all relatively good.


Accuracy of 7 models
Accuracy
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.74
0.57
0.76
0.70
0.67
0.67
0.60
Logistic
SVM
Decision Tree
Random Forest
KNN_Model
Neural Network
XGBoost
Model Name


Recall of 7 models
Recall
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.72
0.26
0.83
0.57
0.55
0.47
0.30
Logistic
SVM
Decision Tree
Random Forest
KNN_Model
Neural Network
XGBoost
Model Name

Since our task is to predict whether a patient has a stroke, it is more important to find out as soon as possible if the patient does have a stroke. In this case, the cost of false negatives is very high since if we don't fail to label a stroke patient, it will be extremely harmful. But if we label a non-patient as a patient, the cost is relatively low. So we think the most important metric here is the recall. As a result, we believe the Decision Tree model is the best model in this case.

There is one last thing needed to be noticed: The values of Precision in the above models are all very low. We believe this may be caused by the small size of the sample. So this will be the main focus for our future work.