UNIVERSITY of WASHINGTON

A Predictive Analytics Approach towards Early Detection of Stroke

Masters Of Science in Information Systems: Introduction to Data Mining and Analytics

MSIS 510B

Purple Team 9 – Srikar, Khushboo, Sneha, Ning, Faraz

December 2021

FOSTER SCHOOL OF BUSINESS
UNIVERSITY of WASHINGTON

BE BOUNDLESS

Introduction

Project Purpose - Using Machine Learning and Data Science to predict Stroke in a human body

Stroke is a medical disorder by which arteries in the blood are ruptured, causing brain damage. When the supply of blood and the other nutrients to the brain is interrupted, there will be development of symptoms.

According to the World Health Organization (WHO), Stroke is the greatest cause of death and disability globally. Early recognition of various warning signs of a stroke can help reduce the severity of the stroke. This project uses a range of physiological parameters and machine learning algorithm's, such as Logistic Regression (LR), Random Forrest (RF) and Neural Networks to build three different models to achieve accurate and detailed analysis for prediction.

Data Description

The dataset used in the development of this method from the open-source Stroke Prediction Dataset using 11 clinical features available on Kaggle.

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

Dataset Attributes:

```
'data.frame':   5110 obs. of  12 variables:
 $ id               : int  9046 51676 31112 60182 1665 56669 53882 10434 27419 60491 ...
 $ gender           : chr  "Male" "Female" "Male" "Female" ...
 $ age              : num  67 61 80 49 79 81 74 69 59 78 ...
 $ hypertension     : int  0 0 0 0 1 0 1 0 0 0 ...
 $ heart_disease    : int  1 0 1 0 0 0 1 0 0 0 ...
 $ ever_married     : chr  "Yes" "Yes" "Yes" "Yes" ...
 $ work_type        : chr  "Private" "Self-employed" "Private" "Private" ...
 $ Residence_type   : chr  "Urban" "Rural" "Rural" "Urban" ...
 $ avg_glucose_level: num  229 202 106 171 174 ...
 $ bmi              : chr  "36.6" "N/A" "32.5" "34.4" ...
 $ smoking_status   : chr  "formerly smoked" "never smoked" "never smoked" "smokes" ...
 $ stroke           : int  1 1 1 1 1 1 1 1 1 1 ...
```

Data Preparation and Exploratory Analysis

Data Preprocessing

Before building a model, data preprocessing is required to remove unwanted noise and outliers from the dataset that could lead the model to depart from its intended training.

In our case we performed the following procedures:

- To begin with, the column id is omitted since its presence has no bearing on model construction
- The dataset is then inspected for null values and filled if any are detected.
- The null values in the column BMI are filled using the data column's mean in this case. The age was categorized into buckets of 10 through which the BMI mean was taken for each bucket and then used to replace the null values present in the BMI Column with respect to the age bucket.

Data Exploration

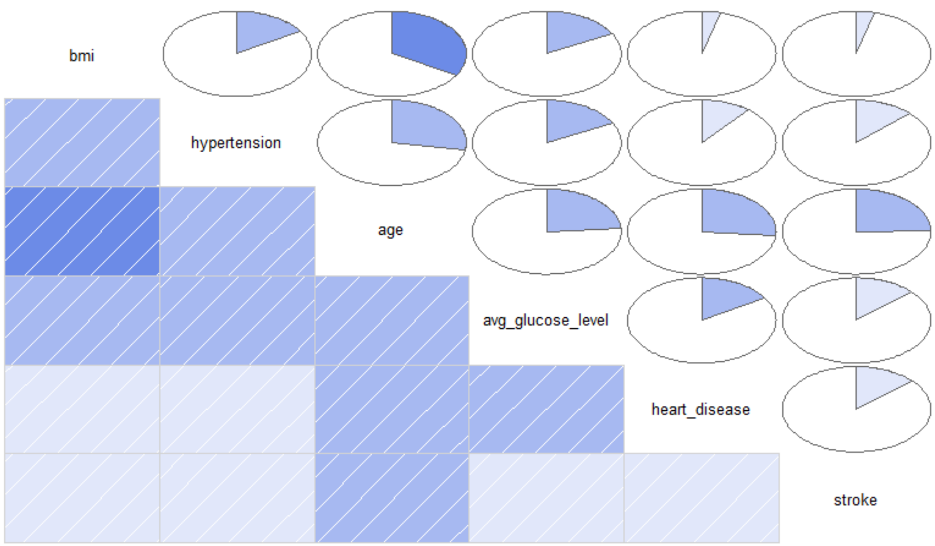
After the data has been preprocessed an exploratory analysis has to be performed to understand the correlation between the target prediction and the other relevant attributes.

We used three functions to perform our exploratory analysis

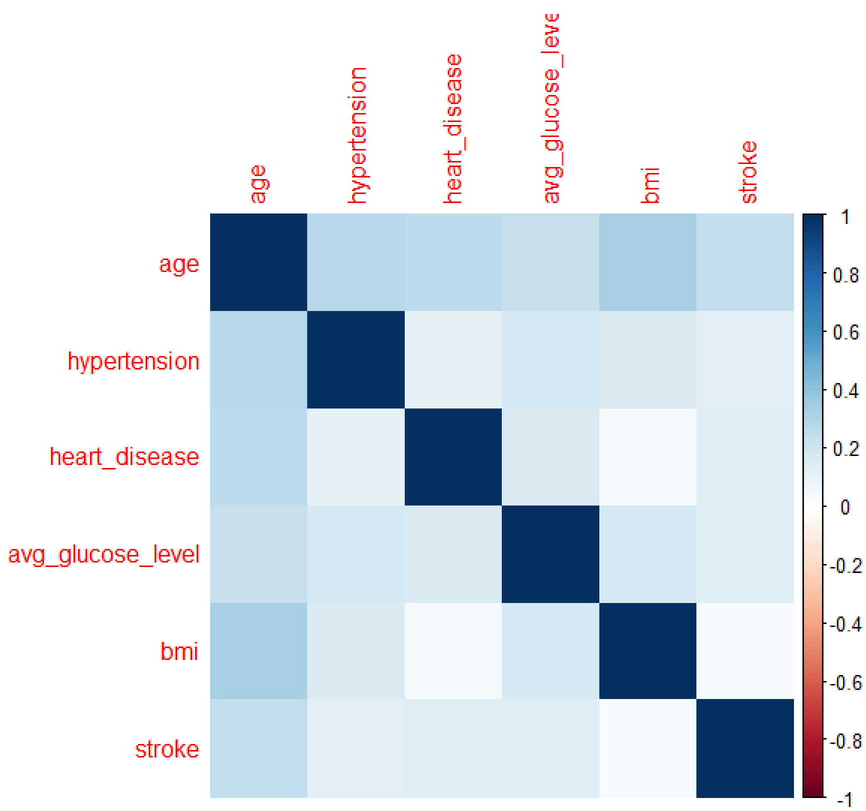
- "Corrgram" Function was used to produce a graphical display of the correlation matrix. The cells in the matrix are the relevant numerical attributes from the dataset depicting age, hypertension, heart disease, average glucose level and BMI to understand the relationship with stroke. To understand relevancy amongst the above attributes, we added pie chart to each attribute and clearly BMI, Age, Hypertension supersedes other attributes. The study of correlation helped us in gauging our visualization and prediction technique.
- Corrplot package allowed us to create a visual of a correlation matrix that helps to automatically reorder and help detect patterns amongst the datasets.
- Ggpairs makes a similar matrix plot using the dataset to determine correlation.

The use of three different exploratory techniques allowed to determine the most significant attributes.

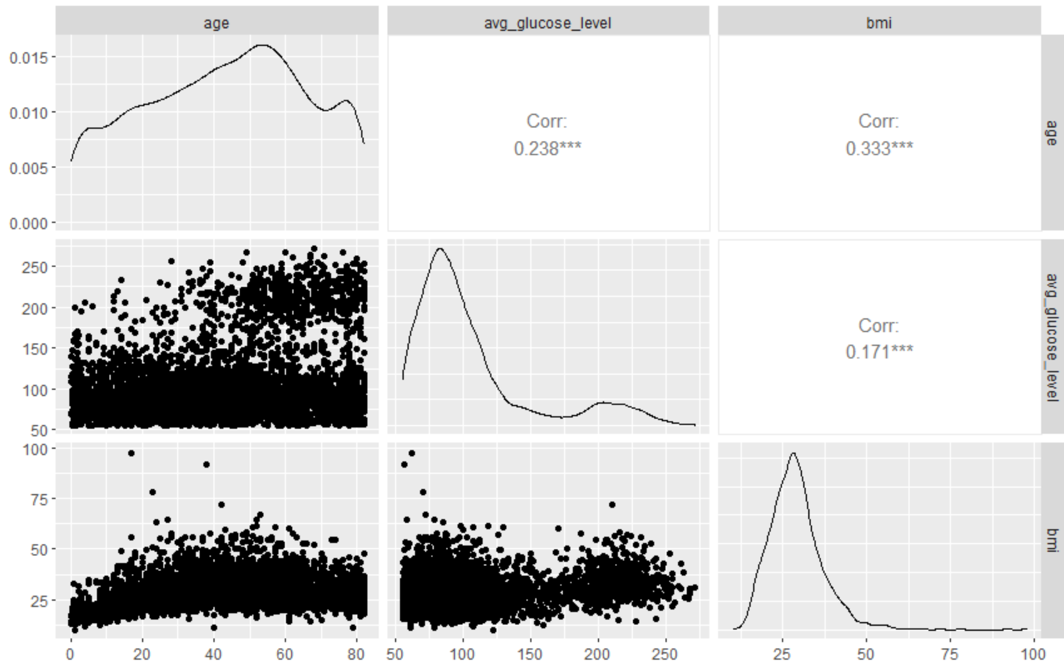
Corrgram Analysis:



Corrplot Analysis:



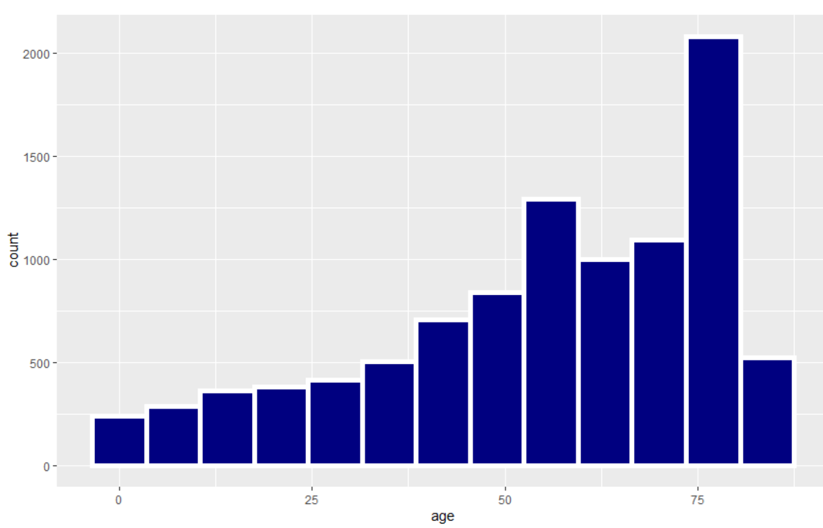
Ggpairs Plot Analysis:



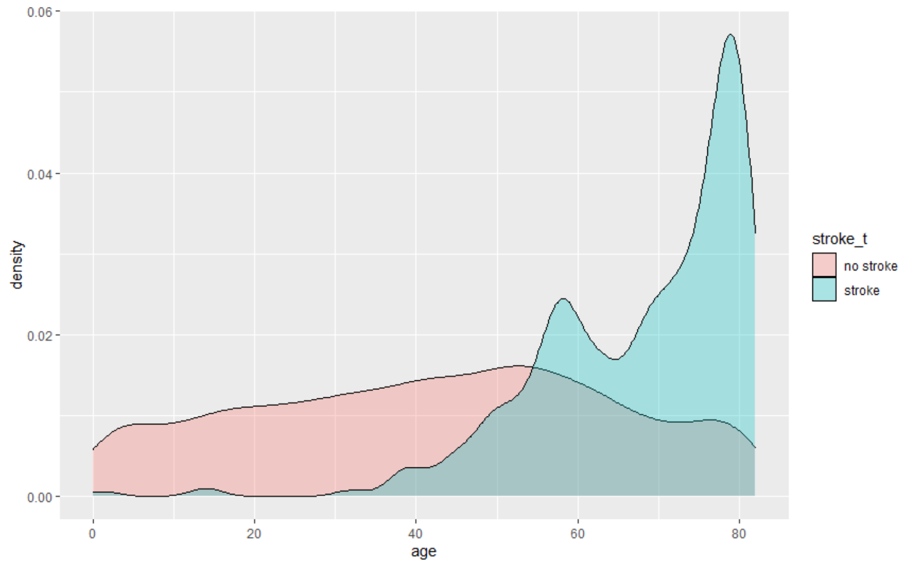
Data Visualization

After the exploratory analysis we used data visualization to understand the impact of each predictor and the data underlying to gauge the implication on stroke prediction before training our prediction models.

Dataset Age Distribution:

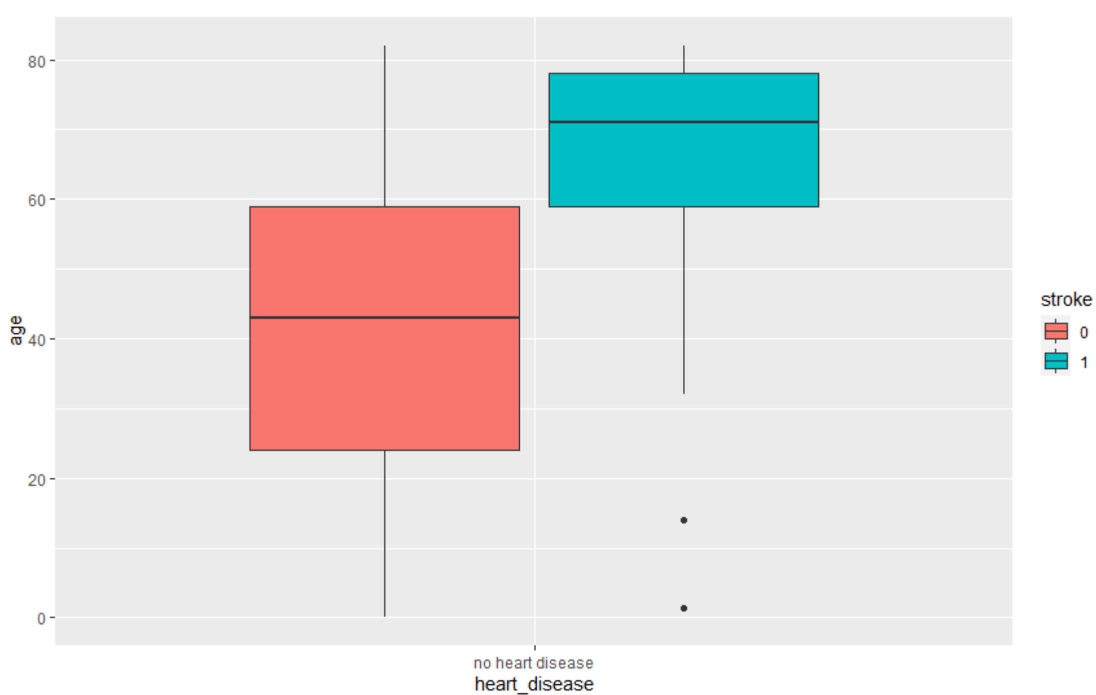


Reportedly Stroke Chances increases with Age:

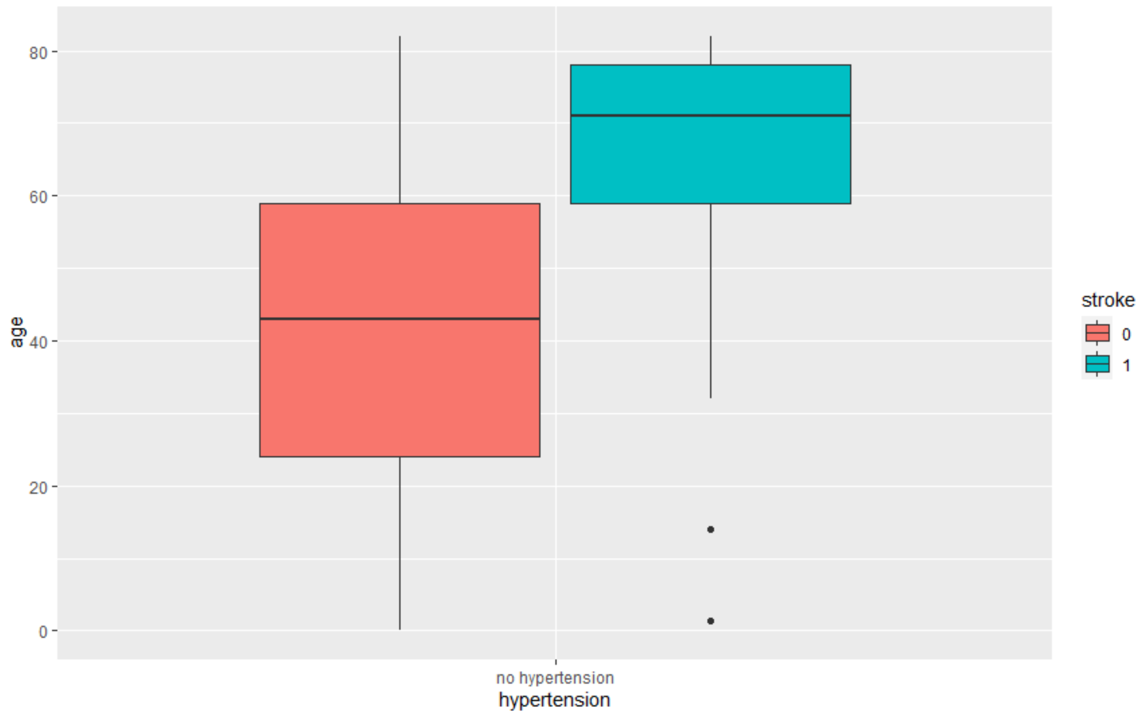


Age leads to other heart, glucose and hypertension issues directly adding to the stroke chances which can be concluded from below visualizations:

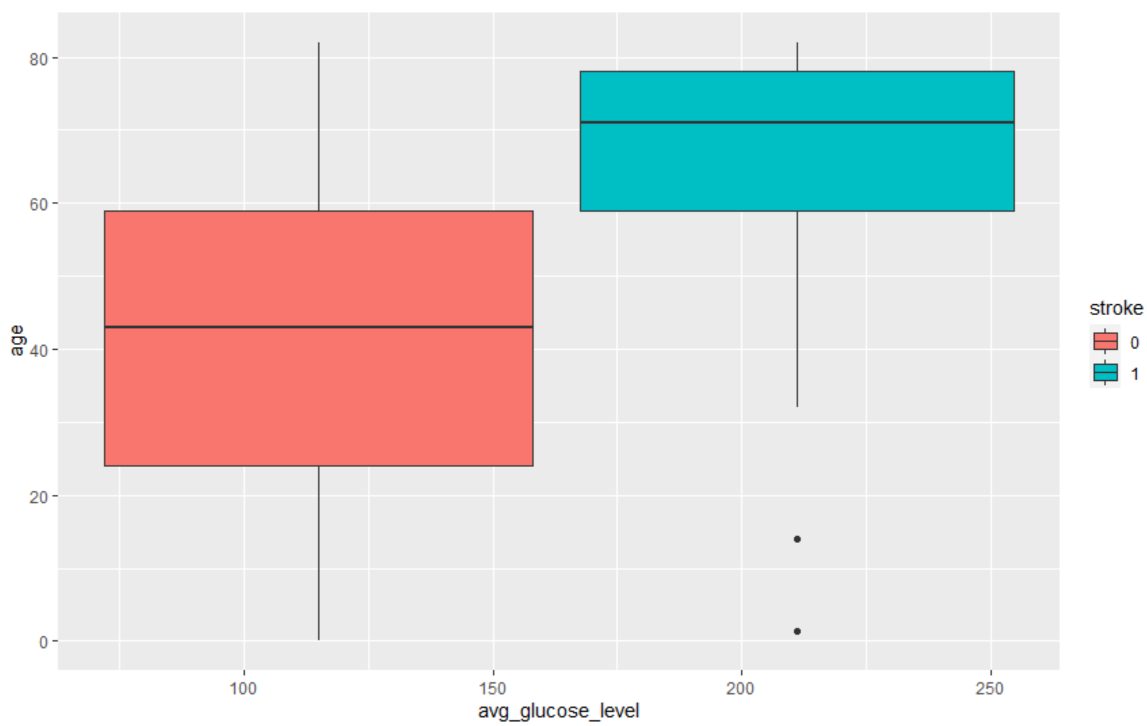
- **Age/Heart Disease/Stroke**



- **Age/Hypertension/Stroke**

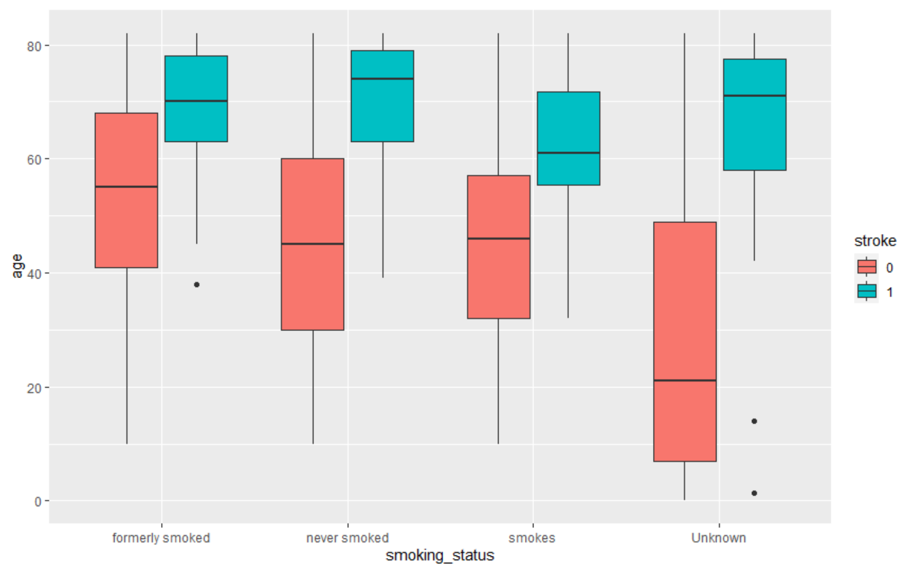


- **Age/Average Glucose Level /Stroke**



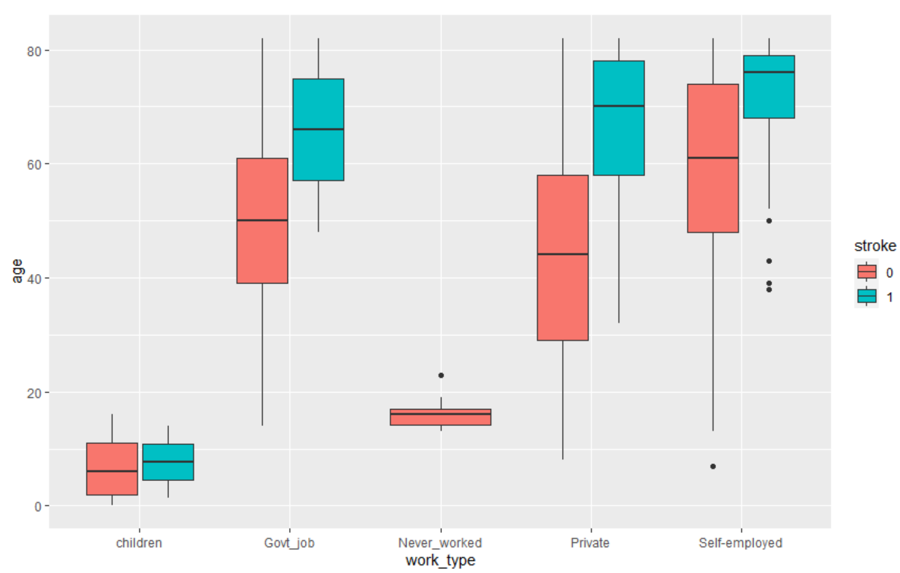
Stroke may be avoided by leading a healthy and balanced lifestyle that includes abstaining from unhealthy behaviors, such as smoking and drinking.

Impact of Smoking:



Even workstyle and age combination can be an interesting thing to consider when training your dataset for Stroke Prediction:

Impact of Workstyle:



Predictive Analytics Methodology

Upsampling

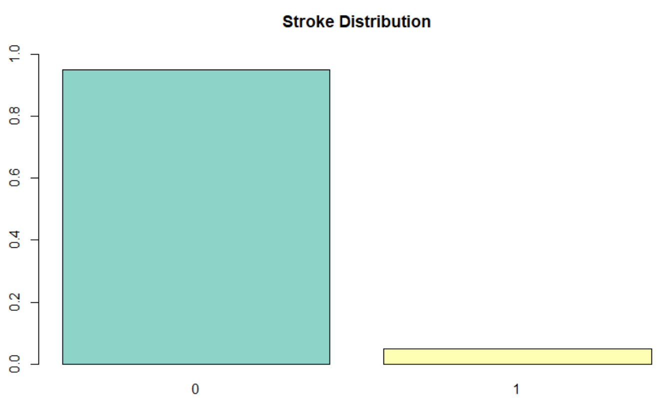
The dataset present in Kaggle for stroke prediction was very imbalanced. The dataset has a total of 5110 rows, with 249 rows indicating the possibility of a stroke and 4861 rows confirming the lack of a stroke.

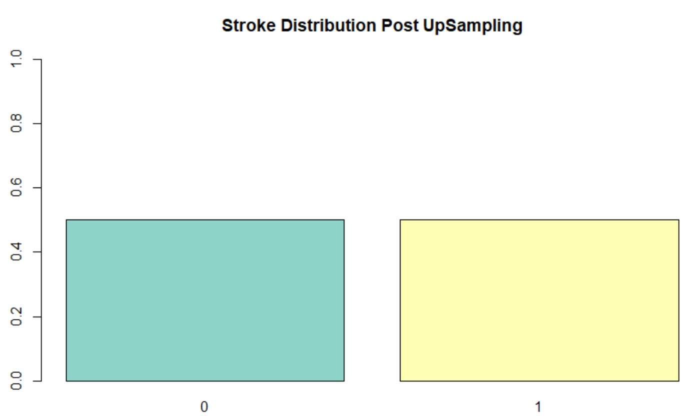
Whilst using such data to train a machine-level model may result in accuracy, other accuracy measures such as precision and recall are inadequate.

To train our algorithms in an efficient manner we followed the approach

- Split the data in 6:4, 60 % Train Data and 40 % Test Data.
- Train Data is then upsampled using upsample function in CARET library.

Stroke distribution before and after Upsampling in the training data:





Now that we have an up sampled version of our trained dataset where both sample outcomes are uniformly distributed, the below predictive models which we will try to train will have a better chance of picking up on the details that define stroke individuals from those who are negative for a stroke.

Logistic Regression

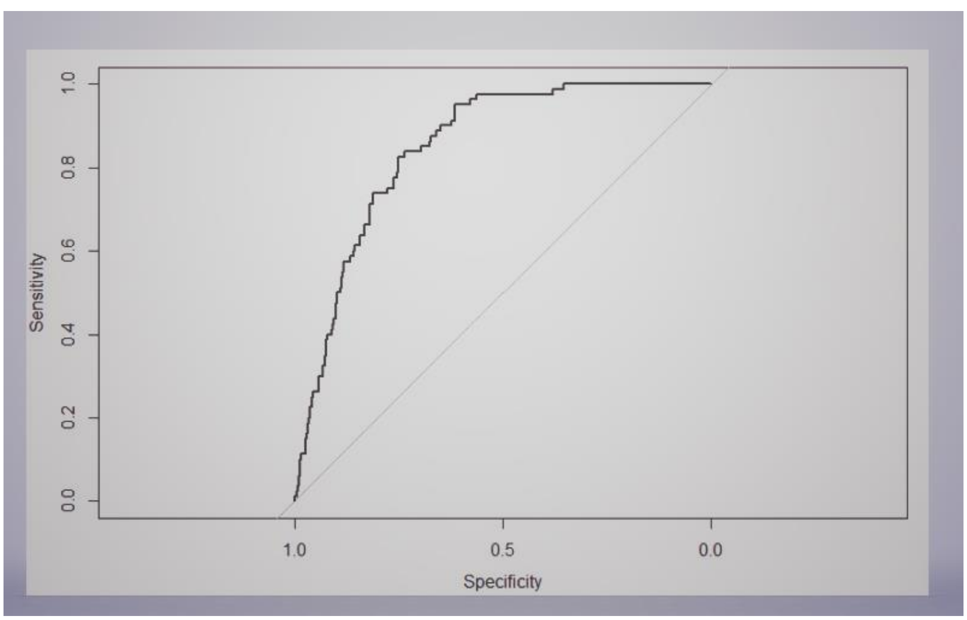
Logistic Regression is a method of supervised learning algorithm which can be used for predicting the probability of the target variable. This algorithm is usually deemed best fit if the output is binary (0 or 1).

Logistic Regression was chosen as the output attribute in the dataset has only two possible values (0 - No stroke predicted, 1- Stroke Predicted).

After performing this algorithm, the accuracy obtained was **75.34%**. Efficiency of this algorithm can also be found by using the metrics precision score and recall score. The specificity score is **82.5%** and the sensitivity score is **75.05%** showing that this is very close to the accuracy.

The other alternative for looking at the combination of Precision and recall score as this takes both the false positives and the false negatives into account. F1 score therefore is more useful, especially in cases where there would be an uneven class distribution. The F1 Score obtained with this algorithm is **78.59%**.

ROC Curve through Logistic Regression:



Confusion Matrix and Statistics through Logistic Regression:

```
> coords(r, x = "best")
  threshold specificity sensitivity
1 0.5225281   0.7505092       0.825
> coords(r, x = c(0.1, 0.2, 0.5))
  threshold specificity sensitivity
1       0.1   0.3350305      1.0000
2       0.2   0.4765784      0.9750
3       0.5   0.7331976      0.8375
> pred <- ifelse(logit.reg.pred > 0.522, 1, 0)
>
> library(caret)
> confusionMatrix(factor(pred), factor(valid.df$stroke), positive = "1")
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1474   14
         1  490   66

               Accuracy : 0.7534
                 95% CI : (0.7341, 0.772)
    No Information Rate : 0.9609
    P-Value [Acc > NIR] : 1

                  Kappa : 0.1493

 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.82500
            Specificity : 0.75051
         Pos Pred Value : 0.11871
         Neg Pred Value : 0.99059
             Prevalence : 0.03914
         Detection Rate : 0.03229
   Detection Prevalence : 0.27202
      Balanced Accuracy : 0.78775

       'Positive' Class : 1
```

Random Forrest

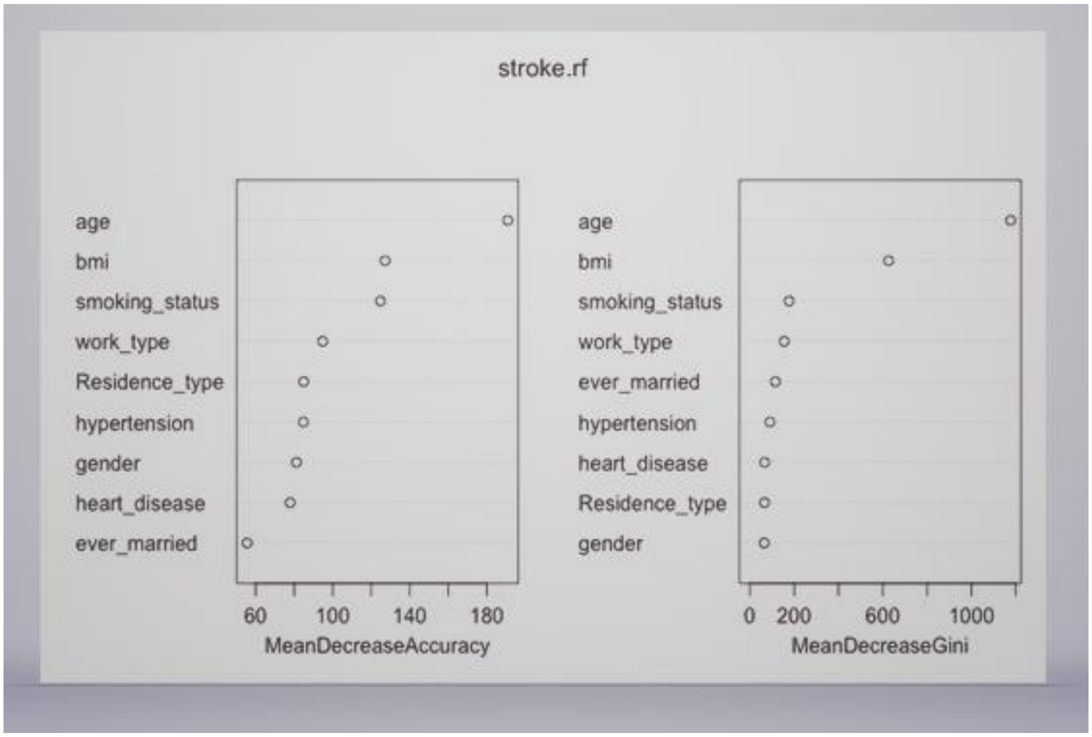
Random Forest is an ensemble of decision trees. It builds multiple decision trees and combines them altogether to get accurate results. It is a classification algorithm. It is called Random as it chooses the predictors randomly and combines results of multiple tress, so it is a forest.

After performing this algorithm, the accuracy obtained was 92.66%. Efficiency of this algorithm can also be found by using the metrics precision score and recall score. The specificity score is 96.61% and the sensitivity score is 13.40%. The other alternative for looking at the combination of Precision and recall score as this takes both the false positives and the false negatives into account. F1 score therefore is more useful, especially in cases where there would be an uneven class distribution. The F1 Score obtained with this algorithm is 13.40% which is comparatively less.

The model is implemented by choosing 3 random predictors and generating 500 trees.

Below diagrams show a few visual results of Random Forest Implementation.

Predictor's Accuracy & Gini Index Comparison's Random Forest:

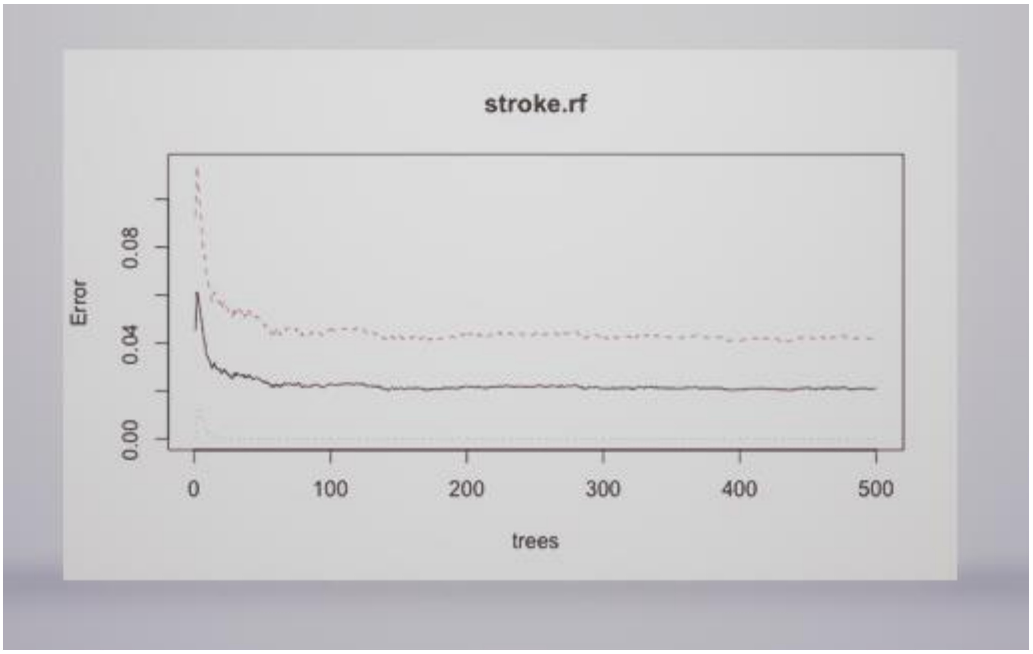


Predictor's Importance in Random Forest:

```
> importance(stroke.rf)
                        0         1 MeanDecreaseAccuracy MeanDecreaseGini
gender          2.7883690  84.69051             83.63493         64.06076
age            53.4191713 179.25867            182.08534       1220.34328
hypertension    5.6906329  69.81097             68.60656         76.15901
heart_disease   5.3912546  75.18726             73.83379         65.98229
ever_married   -3.7604331  61.04824             59.98307        151.16593
work_type      -2.3608825  90.55944             90.81188        167.82993
Residence_type -0.2080452  91.12064             88.90071         75.43688
bmi             7.7471312 131.37371            129.07697        642.23788
smoking_status -0.3761202 110.38851            108.80966        159.02022
```

Error rate through Random Forest:

The graph illustrates the error behavior for 0 to 500 trees choosing 3 predictors at a time in Random Forest Implementation.



Neural Networks

Neural networks can learn and model non-linear and complex relationships, which is important because in real-life, many of the relationships between inputs and outputs are non-linear as well as complex. The purpose we use neural networks model for this dataset is to improve the overall performance of the prediction.

After several times modifying and adjusting, we did a **feature scaling** for each of the variables and set the **3 hidden layers and 32 nodes** for each of the layers. This settlement which has a small layer number and four times the node number made the model less complex to avoid overfitting and keep the relatively high accuracy. A deeper model may give us higher accuracy but, in the meanwhile, it will result in low sensitivity which is unacceptable for the result.

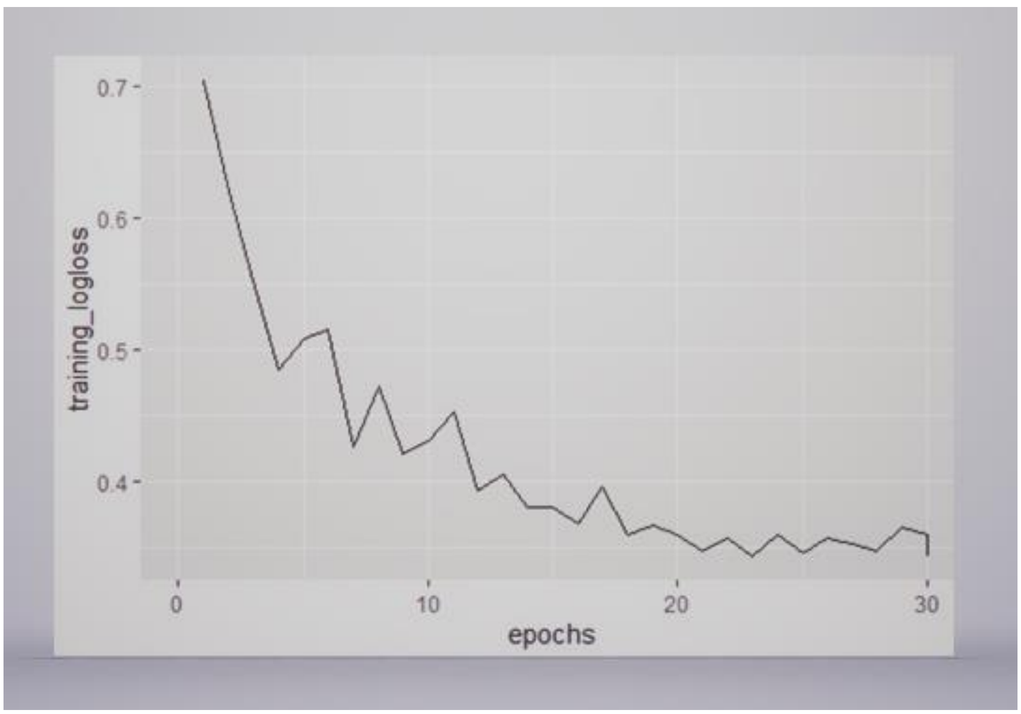
For improvement purpose, we implemented the '**RectifierWithDropout** 'as the activation function, which gave us more flexibility in controlling the regularization process. We do not have much noise in the input, so we will only adjust the hidden dropout ratio which can help us improve generalization.

After compared to the F1 and accuracy result, we set **0.48, 0.47, 0.43 as final hidden dropout ratio** for each layer.

After performing this algorithm, the accuracy obtained is **83.51%**. The sensitivity score is **62.28%** and the specificity score is **84.76**%.

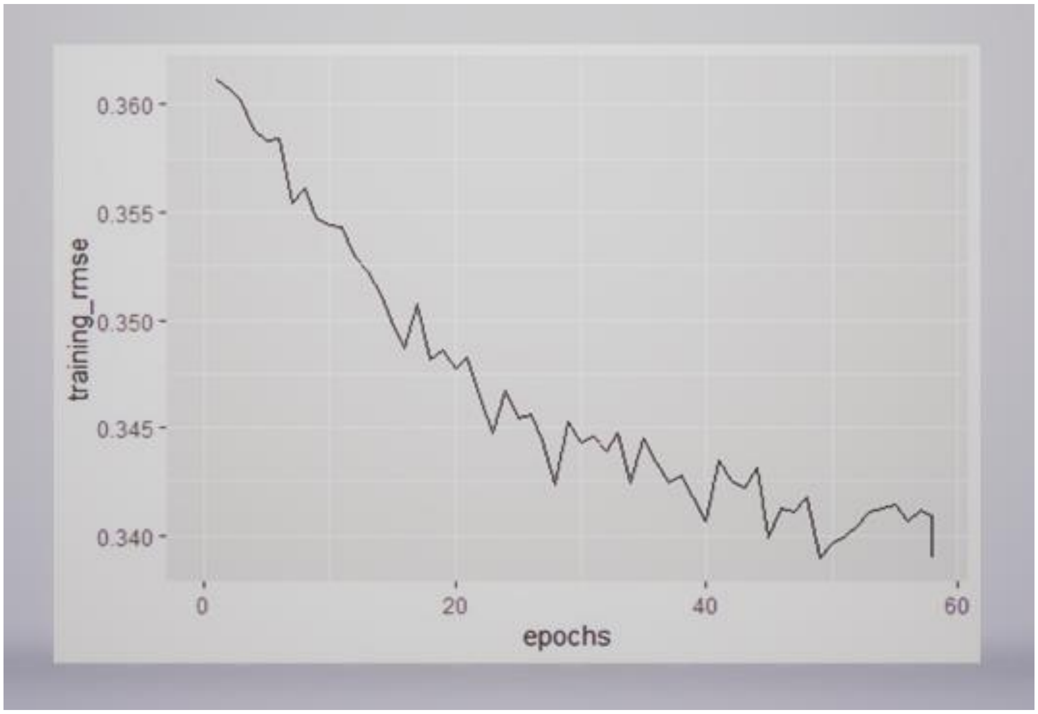
Log loss through each iteration of the Neural Network:

Log-loss is indicative of how close the prediction probability is to the corresponding actual/true value (0 or 1 in case of binary classification). The less the predicted probability diverges from the actual value, the lower is the log-loss value. In this graph, we can see that the log loss drops slowly after the twentieth iteration.



Root-mean-square deviation through each iteration of the Neural Network:

The RMSE drops rapidly between 0 and 40 and goes smoothly from 50 to 60 epochs.



Conclusion

Observations

The development of an ML model could aid in the early detection of stroke and the subsequent mitigation of its severe consequences. The effectiveness of several ML algorithms in properly predicting stroke based on several physiological variables is investigated in this study.

Comparing accuracy, sensitivity, specificity values for the algorithm we used in defining the model, we believe Logistic Regression is the most sensitivity way to predict whether the body will have stroke or not. Logistic Regression outperforms the other methods tested with a sensitivity score of 82.50 percent. In a real-life scenario, neural network would perform better depending on the dataset, in this scenario the data set was a small data set with an Up-sampling procedure performed to improve the training performance.

Based on the models applied on the current data set we can incur some key observations:

1. People in the elder age group are more vulnerable to stroke
2. Significant difference in stroke frequency between groups of people in age of 50-65 with and without heart disease and hypertension.
3. Factor of smoking need to be investigated in a detailed approach.
4. People with a male gender were more prone to stroke based on the data observation.

Improvements

The ability to predict stroke can be a gamechanger, globally over 13 million have stroke each year and around 5.5 million people die of stroke with these numbers going up significantly year on year. There are many other risk factors which influence stroke like tobacco usage, physical inactivity, unhealthy diet, harmful use of alcohol, atrial fibrillation, raised blood lipid level, genetic disposition, and psychological factors. Access to this data allows the models to be trained better as the influx of other attributes which lead to stroke would help derive further correlations. An accurate data set in this aspect would allow researchers to further enhance their models and prepare a list of people in real time who might have a high probability of having a stroke in the future. Access to fast primary response will help reduce the fatality rate in low- and middle-income countries where it is said that two out of three people suffer from a stroke.

References

- https://www.hindawi.com/journals/jhe/2021/7633381/
- https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9349502
- https://towardsdatascience.com/deep-neural-networks-for-regression-problems-81321897ca33
- https://www.kaggle.com/fedesoriano/stroke-prediction-dataset

Appendix

#1 Training Data before and after Upsampling:

```
> str(valid.df)
'data.frame':   2044 obs. of  11 variables:
 $ id               : int  9046 31112 56669 27419 56112 70630 13861 4219 61843 33879 ...
 $ gender           : Factor w/ 3 levels "Female","Male",..: 2 2 2 1 2 1 1 2 2 2 ...
 $ age              : num  67 80 81 59 64 71 52 71 58 42 ...
 $ hypertension     : int  0 0 0 0 0 0 1 0 0 0 ...
 $ heart_disease    : int  1 1 0 0 1 0 0 0 0 0 ...
 $ ever_married     : Factor w/ 2 levels "No","Yes": 2 2 2 2 2 2 2 2 2 2 ...
 $ work_type        : Factor w/ 5 levels "children","Govt_job",..: 4 4 4 4 4 2 5 4 4 4 ...
 $ Residence_type   : Factor w/ 2 levels "Rural","Urban": 2 1 2 1 2 1 2 2 1 1 ...
 $ bmi              : num  36.6 32.5 29 30.6 37.5 ...
 $ smoking_status   : Factor w/ 4 levels "formerly smoked",..: 1 2 1 4 3 3 2 1 4 4 ...
 $ stroke           : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...
> str(train_old.df)
'data.frame':   3066 obs. of  11 variables:
 $ id               : int  35106 21850 507 38070 2580 62715 17771 35838 17951 24736 ...
 $ gender           : Factor w/ 3 levels "Female","Male",..: 2 2 1 1 2 2 1 1 2 1 ...
 $ age              : num  3 58 28 56 66 82 64 1.16 27 4 ...
 $ hypertension     : int  0 0 0 0 0 0 1 0 0 0 ...
 $ heart_disease    : int  0 0 0 0 1 1 0 0 0 0 ...
 $ ever_married     : Factor w/ 2 levels "No","Yes": 1 2 2 2 1 2 2 1 1 1 ...
 $ work_type        : Factor w/ 5 levels "children","Govt_job",..: 1 2 4 4 2 4 2 1 5 1 ...
 $ Residence_type   : Factor w/ 2 levels "Rural","Urban": 2 2 1 1 2 2 2 2 1 2 ...
 $ bmi              : num  17.7 31.4 23.1 29.6 34.5 27.5 22 17 29.5 14 ...
 $ smoking_status   : Factor w/ 4 levels "formerly smoked",..: 4 4 3 2 2 2 2 4 3 4 ...
 $ stroke           : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
> str(train.df)
'data.frame':   5846 obs. of  10 variables:
 $ gender           : Factor w/ 3 levels "Female","Male",..: 2 2 1 1 2 2 1 1 2 1 ...
 $ age              : num  3 58 28 56 66 82 64 1.16 27 4 ...
 $ hypertension     : int  0 0 0 0 0 0 1 0 0 0 ...
 $ heart_disease    : int  0 0 0 0 1 1 0 0 0 0 ...
 $ ever_married     : Factor w/ 2 levels "No","Yes": 1 2 2 2 1 2 2 1 1 1 ...
 $ work_type        : Factor w/ 5 levels "children","Govt_job",..: 1 2 4 4 2 4 2 1 5 1 ...
 $ Residence_type   : Factor w/ 2 levels "Rural","Urban": 2 2 1 1 2 2 2 2 1 2 ...
 $ bmi              : num  17.7 31.4 23.1 29.6 34.5 27.5 22 17 29.5 14 ...
 $ smoking_status   : Factor w/ 4 levels "formerly smoked",..: 4 4 3 2 2 2 2 4 3 4 ...
 $ stroke           : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
```

#2 Coefficient Analysis through Logistic Regression

```
Call:
glm(formula = stroke ~ ., family = "binomial", data = train.df)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.3523  -0.7439   0.1573   0.7562   2.5640

Coefficients:
                             Estimate Std. Error z value Pr(>|z|)
(Intercept)                 -3.528510   0.269929 -13.072  < 2e-16 ***
genderMale                  -0.079587   0.068550  -1.161 0.245638
genderOther                -11.915151 882.743390  -0.013 0.989231
age                          0.076206   0.002554  29.837  < 2e-16 ***
hypertension                 0.419827   0.088982   4.718 2.38e-06 ***
heart_disease                0.424707   0.110727   3.836 0.000125 ***
ever_marriedYes              0.292572   0.115276   2.538 0.011148 *
work_typeGovt_job           -1.458202   0.293354  -4.971 6.67e-07 ***
work_typeNever_worked      -12.339994 254.043860  -0.049 0.961259
work_typePrivate            -1.529561   0.283495  -5.395 6.84e-08 ***
work_typeSelf-employed      -1.722045   0.299876  -5.743 9.33e-09 ***
Residence_typeUrban          0.046218   0.065511   0.705 0.480501
bmi                          0.019009   0.005137   3.700 0.000216 ***
smoking_statusnever smoked  -0.430208   0.087430  -4.921 8.63e-07 ***
smoking_statussmokes        -0.189851   0.105387  -1.801 0.071630 .
smoking_statusUnknown       -0.216948   0.100183  -2.166 0.030348 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 8032.2  on 5793  degrees of freedom
Residual deviance: 5701.3  on 5778  degrees of freedom
AIC: 5733.3

Number of Fisher Scoring iterations: 13
```

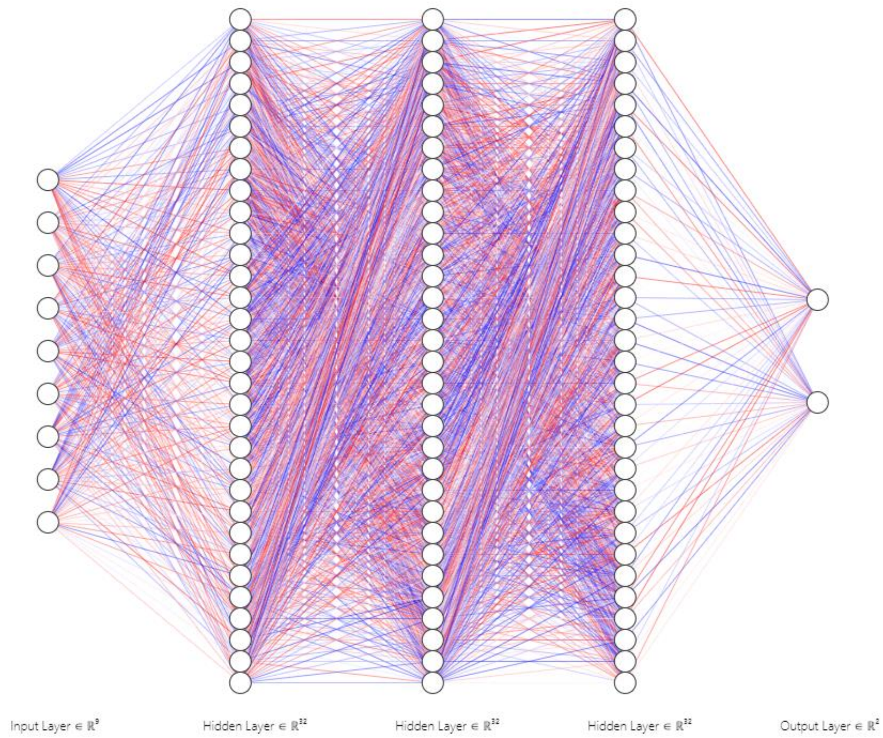
#3 Confusion Matrix for Random Forrest

```
> confusionMatrix(factor(rf.pred), factor(valid.df$stroke), positive = "1")
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1881   84
         1   66   13

               Accuracy : 0.9266
                 95% CI : (0.9144, 0.9375)
    No Information Rate : 0.9525
    P-Value [Acc > NIR] : 1.0000

                  Kappa : 0.1098

 Mcnemar's Test P-Value : 0.1651

            Sensitivity : 0.13402
            Specificity : 0.96610
         Pos Pred Value : 0.16456
         Neg Pred Value : 0.95725
             Prevalence : 0.04746
         Detection Rate : 0.00636
   Detection Prevalence : 0.03865
      Balanced Accuracy : 0.55006

       'Positive' Class : 1
```

#4 Confusion Matrix for Neural Network

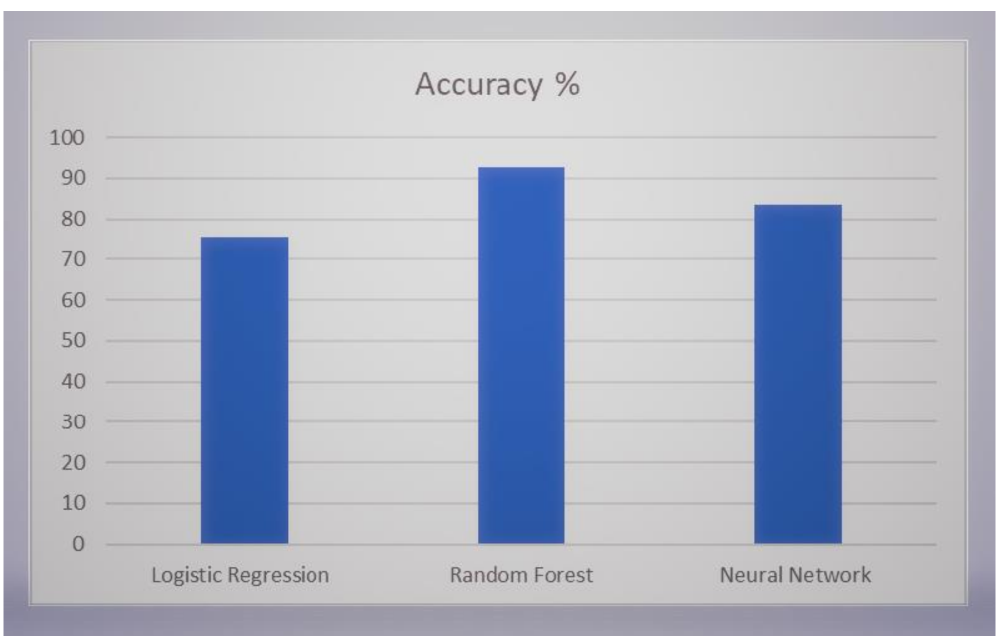
```
> confusionMatrix(factor(y_pred),factor(valid.df[, 10]),positive = '1')
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1636   43
         1  294   71

               Accuracy : 0.8351
                 95% CI : (0.8183, 0.851)
    No Information Rate : 0.9442
    P-Value [Acc > NIR] : 1

                  Kappa : 0.2311

 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.62281
            Specificity : 0.84767
         Pos Pred Value : 0.19452
         Neg Pred Value : 0.97439
             Prevalence : 0.05577
         Detection Rate : 0.03474
   Detection Prevalence : 0.17857
      Balanced Accuracy : 0.73524

       'Positive' Class : 1
```

#4 Visualization Graph for Neural Network

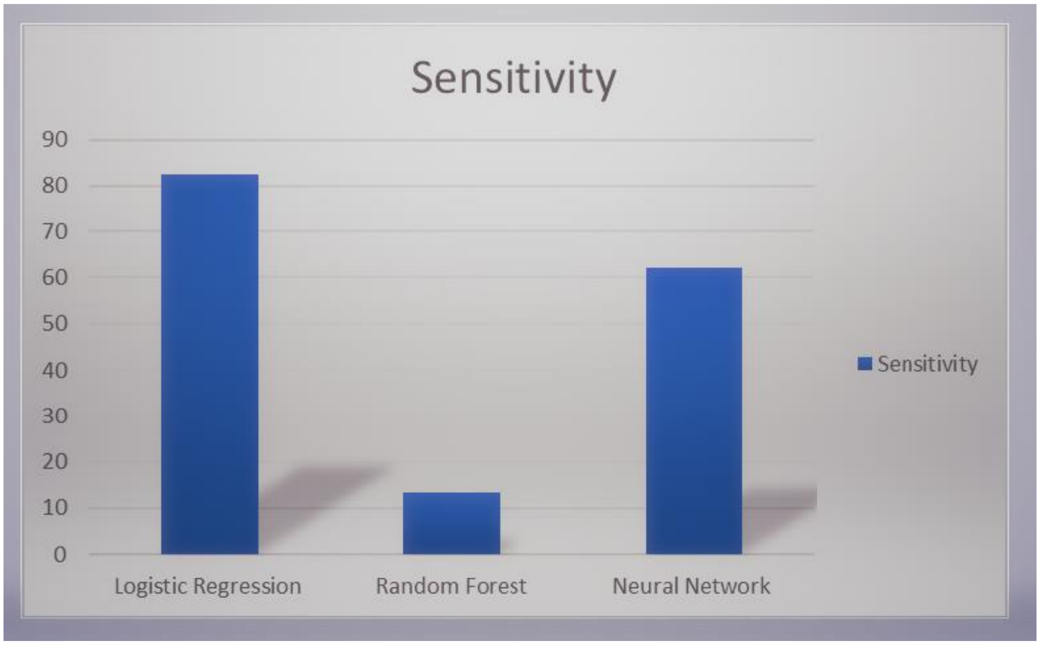


#5 Algorithm Comparison based on Accuracy, Sensitivity and Specificity:

#5.1 Accuracy %



#5.2 Sensitivity %



#5.3 Specificity %