MLPH R

2024-05-13

```
clean_output <- function(x) {
  gsub("\b", "", x)
}
```

```
library(readr)
brain_stroke <- read_csv("/Users/v/Desktop/brain_stroke.csv")
```

```
## Rows: 4981 Columns: 11
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (5): gender, ever_married, work_type, Residence_type, smoking_status
## dbl (6): age, hypertension, heart_disease, avg_glucose_level, bmi, stroke
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
#recode
# check the missing value
sum(is.na(brain_stroke))
```

```
## [1] 0
```

```
#marry
table(brain_stroke$ever_married)
```

```
##
##   No  Yes
## 1701 3280
```

```
brain_stroke$ever_married <- ifelse(brain_stroke$ever_married == "No", 0, ifelse(brain_stroke$ever_marri
table(brain_stroke$ever_married)
```

```
##
##    0    1
## 1701 3280
```

```
#Residence_type
table(brain_stroke$Residence_type)
```

```
##
## Rural Urban
##  2449  2532
```

```
brain_stroke$Residence_type <- ifelse(brain_stroke$Residence_type == "Urban", 0, ifelse(brain_stroke$Res
table(brain_stroke$Residence_type)
```

```
##
##    0    1
## 2532 2449
```

```r
#gender
table(brain_stroke$gender)
```

```
##
## Female   Male
##   2907   2074
```

```r
brain_stroke$gender <- ifelse(brain_stroke$gender == "Male", 0, ifelse(brain_stroke$gender == "Female",
table(brain_stroke$gender)
```

```
##
##    0    1
## 2074 2907
```

```r
# one_hot encoding for smoking and work type
table(brain_stroke$smoking_status)
```

```
##
## formerly smoked    never smoked          smokes         Unknown
##             867            1838             776            1500
```

```r
table(brain_stroke$work_type )
```

```
##
##      children      Govt_job       Private Self-employed
##           673           644          2860           804
```

```r
one_hot_encoded <- model.matrix(~ smoking_status + work_type    - 1, data = brain_stroke)
head(one_hot_encoded)
```

```
##   smoking_statusformerly smoked smoking_statusnever smoked smoking_statussmokes
## 1                             1                           0                     0
## 2                             0                           1                     0
## 3                             0                           0                     1
## 4                             0                           1                     0
## 5                             1                           0                     0
## 6                             0                           1                     0
##   smoking_statusUnknown work_typeGovt_job work_typePrivate
## 1                     0                0                1
## 2                     0                0                1
## 3                     0                0                1
## 4                     0                0                0
## 5                     0                0                1
## 6                     0                0                1
##   work_typeSelf-employed
## 1                      0
## 2                      0
## 3                      0
## 4                      1
## 5                      0
## 6                      0
```

```r
df2 <- cbind(brain_stroke, one_hot_encoded)
df2_new <- subset(df2, select = -c(work_type, smoking_status))

colnames(df2_new)[colnames(df2_new) == 'smoking_statusformerly smoked'] <- 'smoking_statusformerly_smok
colnames(df2_new)[colnames(df2_new) == 'smoking_statusnever smoked'] <- 'smoking_statusnever_smoked'
```

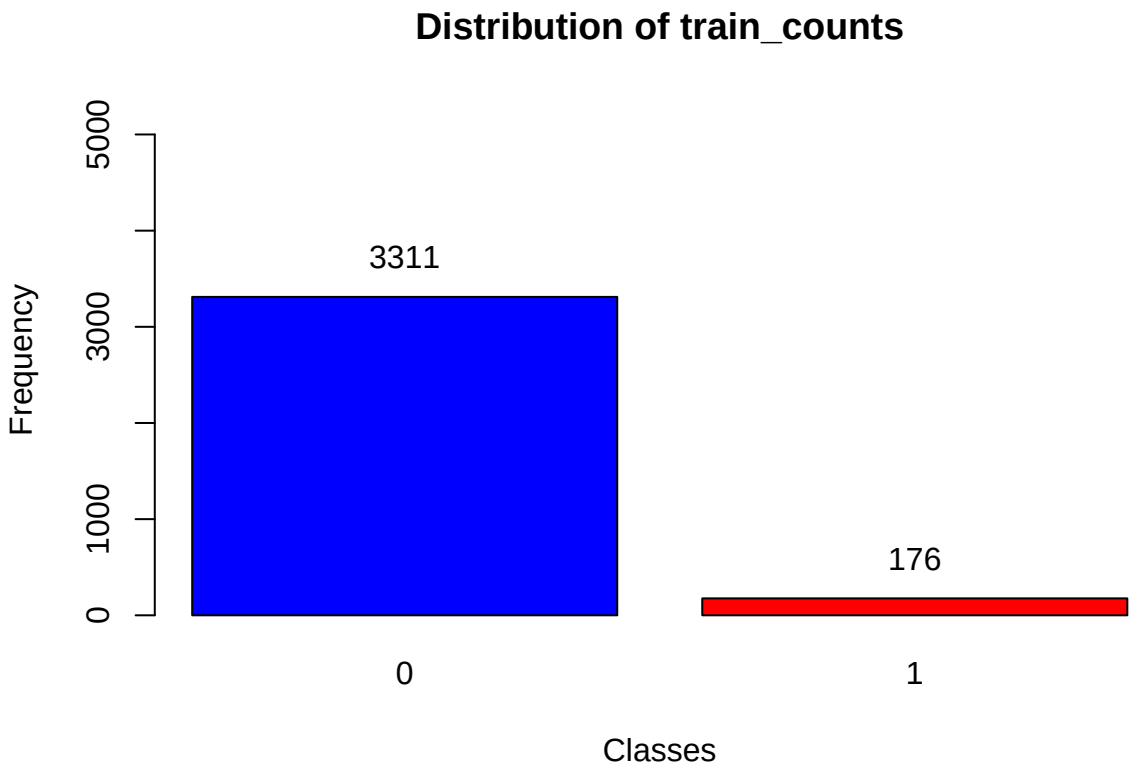
```r
colnames(df2_new)[colnames(df2_new) == 'work_typeSelf-employed'] <- 'work_typeSelf_employed'
```

```r
#split the data
library(caret)
set.seed(123)
train_indices<- createDataPartition(df2_new$stroke, p = .7, list = FALSE, times = 1)
train_data <- df2_new[train_indices, ]
test_data <- df2_new[-train_indices, ]

## imblance
# data imblance
train_counts <- table(train_data$stroke)
train_counts
```

```
##
##    0    1
## 3311  176
```

```r
barplot(train_counts,
        main = "Distribution of train_counts",
        xlab = "Classes",
        ylab = "Frequency",
        col = c("blue", "red"),
        ylim = c(0, max(train_counts) + 2000))
midpoints <- barplot(train_counts, plot = FALSE)
text(midpoints, train_counts + 100, labels = train_counts, pos = 3)
```

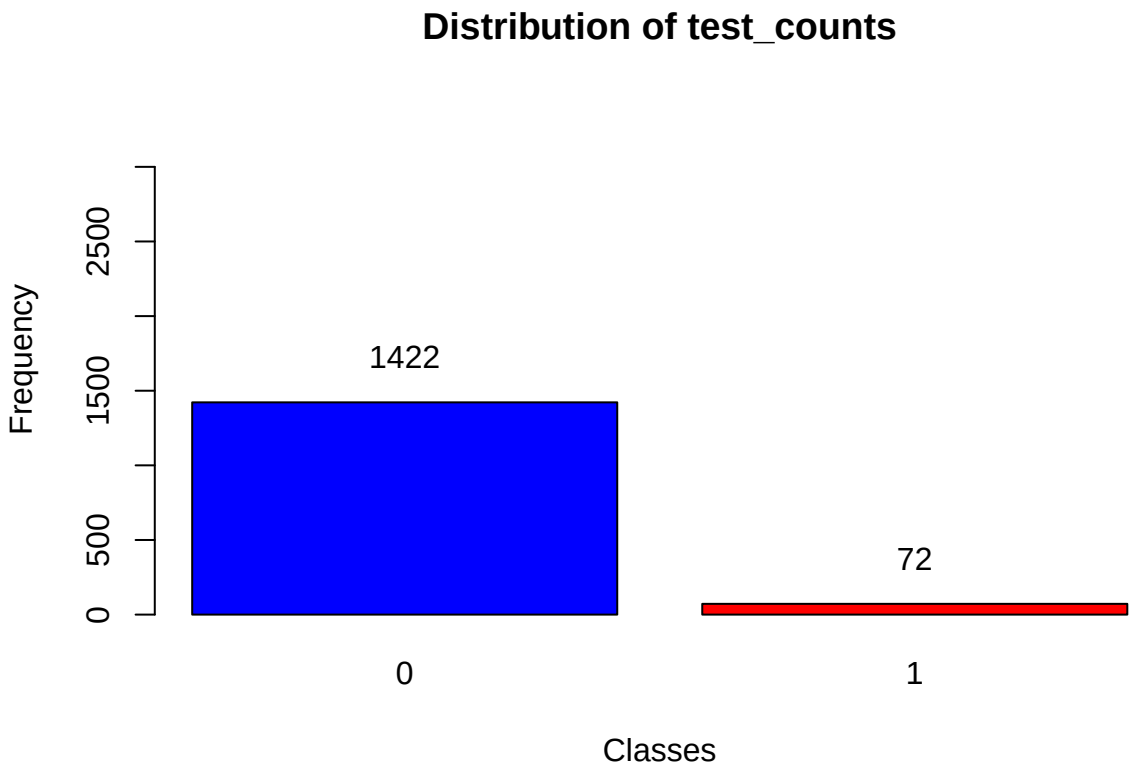



```r
test_counts <- table(test_data$stroke)
test_counts
```

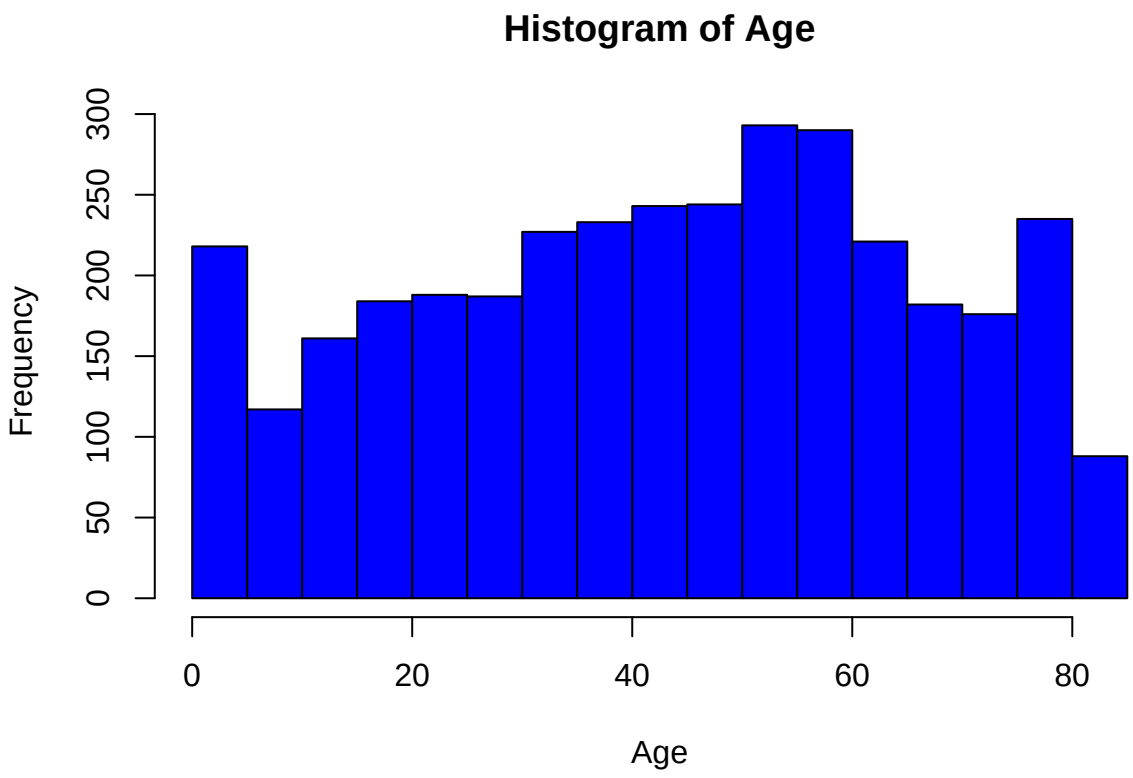
```
##
```

```
##    0    1
## 1422   72
```

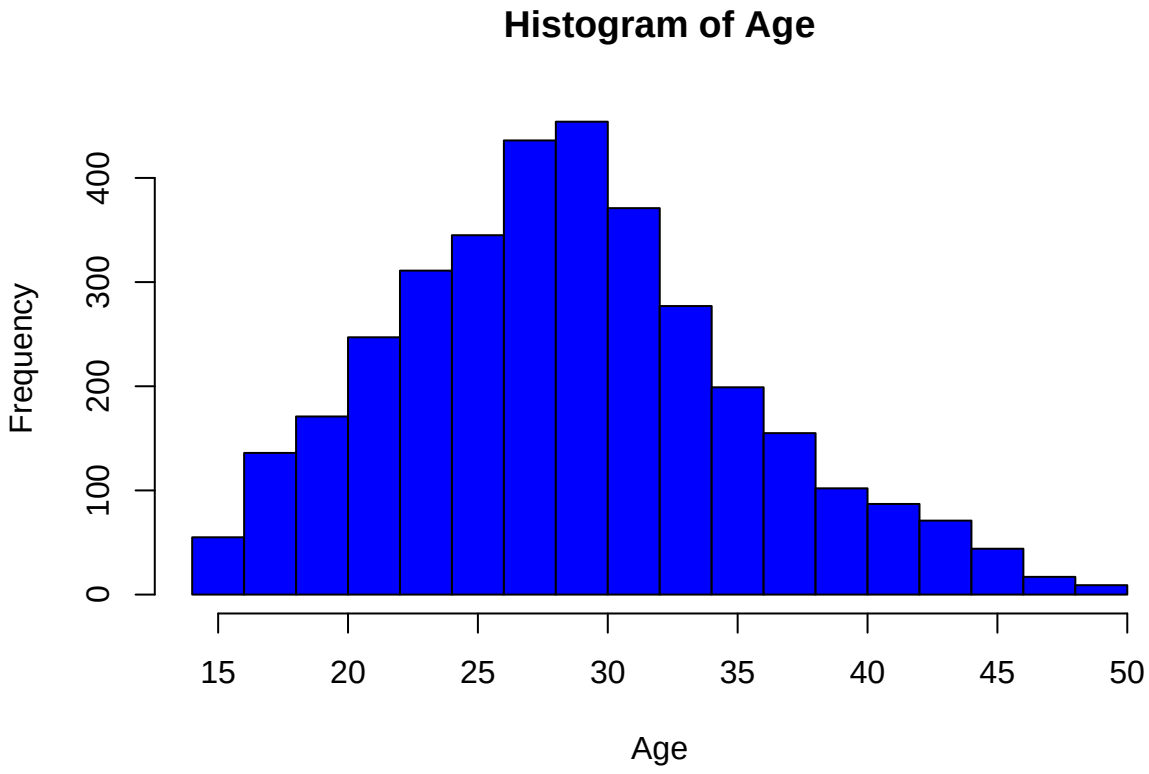
```r
barplot(test_counts,
        main = "Distribution of test_counts",
        xlab = "Classes",
        ylab = "Frequency",
        col = c("blue", "red"),
        ylim = c(0, max(test_counts) + 2000))
midpoints <- barplot(test_counts, plot = FALSE)
text(midpoints, test_counts + 100, labels = test_counts, pos = 3)
```



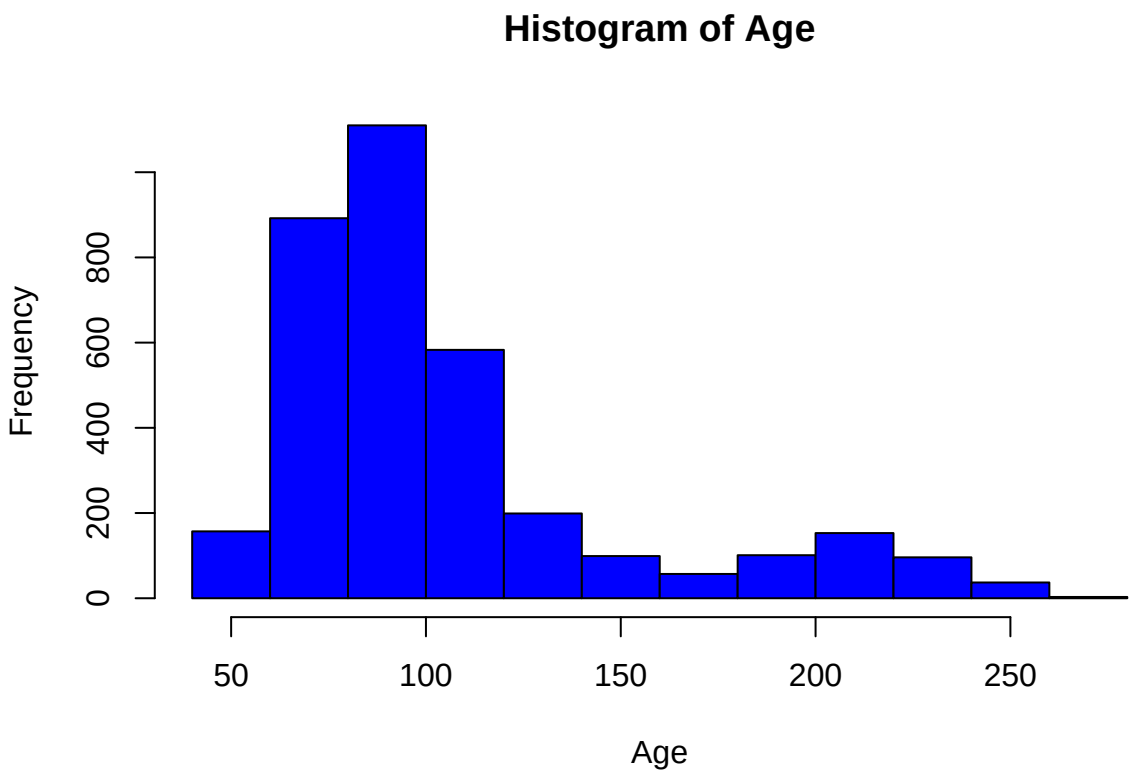

```r
#plot
hist(train_data$age, main = "Histogram of Age", xlab = "Age", col = "blue", border = "black")
```




```
hist(train_data$bmi, main = "Histogram of Age", xlab = "Age", col = "blue", border = "black")
```




```
hist(train_data$avg_glucose_level, main = "Histogram of Age", xlab = "Age", col = "blue", border = "bla
```




```
#_standardized- not for tree model!!
train_data_standardized <-train_data
train_data_standardized [, 2] <- scale(train_data [, 2], center = TRUE, scale = TRUE)
train_data_standardized[, 7:8] <- scale(train_data [, 7:8], center = TRUE, scale = TRUE)
head(train_data_standardized )
```

```
##    gender       age hypertension heart_disease ever_married Residence_type
## 1       0 1.0329007            0             1            1              0
## 2       0 1.6070420            0             1            1              1
## 3       1 0.2379357            0             0            1              0
## 5       0 1.6512068            0             0            1              0
## 7       1 1.1212301            0             0            0              0
## 10      1 0.7679124            0             1            1              1
##    avg_glucose_level         bmi stroke smoking_statusformerly_smoked
## 1        2.739041848  1.18343794      1                             1
## 2        0.006155042  0.58000632      1                             0
## 3        1.459969796  0.85964536      1                             0
## 5        1.793427842  0.06488176      1                             1
## 7       -0.250505256 -0.84762459      1                             0
## 10       0.329818594  1.21287363      1                             0
##    smoking_statusnever_smoked smoking_statussmokes smoking_statusUnknown
## 1                           0                    0                    0
## 2                           1                    0                    0
## 3                           0                    1                    0
## 5                           0                    0                    0
## 7                           1                    0                    0
## 10                          0                    1                    0
##    work_typeGovt_job work_typePrivate work_typeSelf_employed
## 1                  0                1                      0
## 2                  0                1                      0
```

```
## 3                  0               1                     0
## 5                  0               1                     0
## 7                  0               1                     0
## 10                 1               0                     0
```

```
test_data_standardized <-test_data
test_data_standardized [, 2] <- scale(test_data [, 2], center = TRUE, scale = TRUE)
test_data_standardized[, 7:8] <- scale(test_data [, 7:8], center = TRUE, scale = TRUE)
head(test_data_standardized )
```

```
##    gender       age hypertension heart_disease ever_married Residence_type
## 4       1 1.5864376            1             0            1              1
## 6       0 1.3662816            1             1            1              1
## 8       1 1.5424064            0             0            1              0
## 9       1 1.6745000            1             0            1              1
## 16      1 0.7498448            0             0            0              0
## 23      1 0.9700008            0             0            1              1
##    avg_glucose_level        bmi stroke smoking_statusformerly_smoked
## 4          1.4850806 -0.6423349      1                             0
## 6         -0.8045148 -0.1409512      1                             0
## 8         -1.0580584 -0.6128418      1                             0
## 9         -0.5769419  0.1982201      1                             0
## 16        -0.3834828  1.3926930      1                             0
## 23        -0.1246571 -0.0229786      1                             1
##    smoking_statusnever_smoked smoking_statussmokes smoking_statusUnknown
## 4                           1                    0                    0
## 6                           1                    0                    0
## 8                           0                    0                    1
## 9                           1                    0                    0
## 16                          1                    0                    0
## 23                          0                    0                    0
##    work_typeGovt_job work_typePrivate work_typeSelf_employed
## 4                  0               0                     1
## 6                  0               1                     0
## 8                  0               1                     0
## 9                  0               1                     0
## 16                 0               1                     0
## 23                 0               1                     0
```

```
table(train_data_standardized$stroke)
```

```
##
##    0    1
## 3311  176
```

```
## how to fix imblance smote? weight on model?
## Smote -- for no standardizaton model
str(train_data)
```

```
## 'data.frame':    3487 obs. of  16 variables:
##  $ gender                       : num  0 0 1 0 1 1 1 1 1 0 ...
##  $ age                          : num  67 80 49 81 69 61 54 79 50 64 ...
##  $ hypertension                 : num  0 0 0 0 0 0 0 0 1 0 ...
##  $ heart_disease                : num  1 1 0 0 0 1 0 1 0 1 ...
##  $ ever_married                 : num  1 1 1 1 0 1 1 1 1 1 ...
##  $ Residence_type               : num  0 1 0 0 0 1 0 0 1 0 ...
```

```
##  $ avg_glucose_level            : num  228.7 105.9 171.2 186.2 94.4 ...
##  $ bmi                          : num  36.6 32.5 34.4 29 22.8 36.8 27.3 28.2 30.9 37.5 ...
##  $ stroke                       : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ smoking_statusformerly_smoked: num  1 0 0 1 0 0 0 0 0 0 ...
##  $ smoking_statusnever_smoked   : num  0 1 0 0 1 0 0 1 1 0 ...
##  $ smoking_statussmokes         : num  0 0 1 0 0 1 1 0 0 1 ...
##  $ smoking_statusUnknown        : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ work_typeGovt_job            : num  0 0 0 0 0 1 0 0 0 0 ...
##  $ work_typePrivate             : num  1 1 1 1 1 0 1 1 0 1 ...
##  $ work_typeSelf_employed       : num  0 0 0 0 0 0 0 0 1 0 ...
```

```
summary(train_data$stroke)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.00000 0.00000 0.00000 0.05047 0.00000 1.00000
```

```
library(smotefamily)
smote_output <- SMOTE(train_data, train_data$stroke, K = 5, dup_size = 17.8)
balanced_data <- smote_output$data
balanced_data_new <- subset(balanced_data, select = -c(class))
table(balanced_data_new$stroke)
```

```
##
##    0    1
## 3311 3168
```

```
## standardizaton
smote_output_standardizaton <- SMOTE(train_data_standardized, train_data_standardized$stroke, K = 5, du
balanced_data_standardizaton <- smote_output_standardizaton$data
balanced_data_std_new <- subset(balanced_data_standardizaton, select = -c(class))
table(balanced_data_std_new$stroke)
```

```
##
##    0    1
## 3311 3168
```

```
## XGboost

train_data_xgboost <- as.matrix(balanced_data_new[, -which(names(balanced_data_new) == "stroke")])
train_label <- balanced_data_new$stroke


test_data_xgboost <- as.matrix(test_data[, -which(names(test_data) == "stroke")])
test_label <- test_data$stroke
library(xgboost)

# Create DMatrix for training and testing
dtrain <- xgb.DMatrix(data = train_data_xgboost, label = train_label)
dtest <- xgb.DMatrix(data = test_data_xgboost, label = test_label)

params <- list(
  booster = "gbtree",
  eta = 0.1,
  max_depth = 6,
  min_child_weight = 1,
  subsample = 0.5,
  colsample_bytree = 0.7,
```

```
  objective = "binary:logistic",
  eval_metric = "auc"
)

num_rounds <- 100
watchlist <- list(train = dtrain, test = dtest)
model <- xgb.train(
  params = params,
  data = dtrain,
  nrounds = num_rounds,
  watchlist = watchlist,
  early_stopping_rounds = 10,
  verbose = 1
)
```

```
## [1]  train-auc:0.958777  test-auc:0.621308
## Multiple eval metrics are present. Will use test_auc for early stopping.
## Will train until test_auc hasn't improved in 10 rounds.
##
## [2]  train-auc:0.976684  test-auc:0.709501
## [3]  train-auc:0.987773  test-auc:0.795442
## [4]  train-auc:0.988830  test-auc:0.804174
## [5]  train-auc:0.989354  test-auc:0.805404
## [6]  train-auc:0.989880  test-auc:0.806649
## [7]  train-auc:0.990415  test-auc:0.819039
## [8]  train-auc:0.990269  test-auc:0.826457
## [9]  train-auc:0.990761  test-auc:0.823376
## [10] train-auc:0.991417  test-auc:0.821432
## [11] train-auc:0.991433  test-auc:0.822018
## [12] train-auc:0.991785  test-auc:0.821701
## [13] train-auc:0.992175  test-auc:0.818902
## [14] train-auc:0.992543  test-auc:0.819874
## [15] train-auc:0.992790  test-auc:0.824011
## [16] train-auc:0.992909  test-auc:0.822863
## [17] train-auc:0.993054  test-auc:0.823259
## [18] train-auc:0.993206  test-auc:0.823688
## Stopping. Best iteration:
## [8]  train-auc:0.990269  test-auc:0.826457
```

```
pred_probs <- predict(model, dtest)
pred_class <- ifelse(pred_probs > 0.5, 1, 0)
library(pROC)
conf_matrix <- confusionMatrix(factor(pred_class, levels = c(0, 1)),
                               factor(test_label, levels = c(0, 1)))

accuracy_xgboost <- conf_matrix$overall['Accuracy']
precision_xgboost <- conf_matrix$byClass['Precision']
recall_xgboost <- conf_matrix$byClass['Sensitivity']


f1_score_xgboost <- 2 * (precision_xgboost * recall_xgboost) / (precision_xgboost + recall_xgboost)

print(accuracy_xgboost)
```

```
##  Accuracy
```

```
## 0.9464525
```

```
print(precision_xgboost)
```

```
## Precision
## 0.9564626
```

```
print(recall_xgboost)
```

```
## Sensitivity
##   0.9887482
```

```
print(f1_score_xgboost)
```

```
## Precision
## 0.9723375
```

```
roc_obj_xgboost <- roc(response = factor(test_label, levels = c(0, 1)), predictor = pred_probs)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

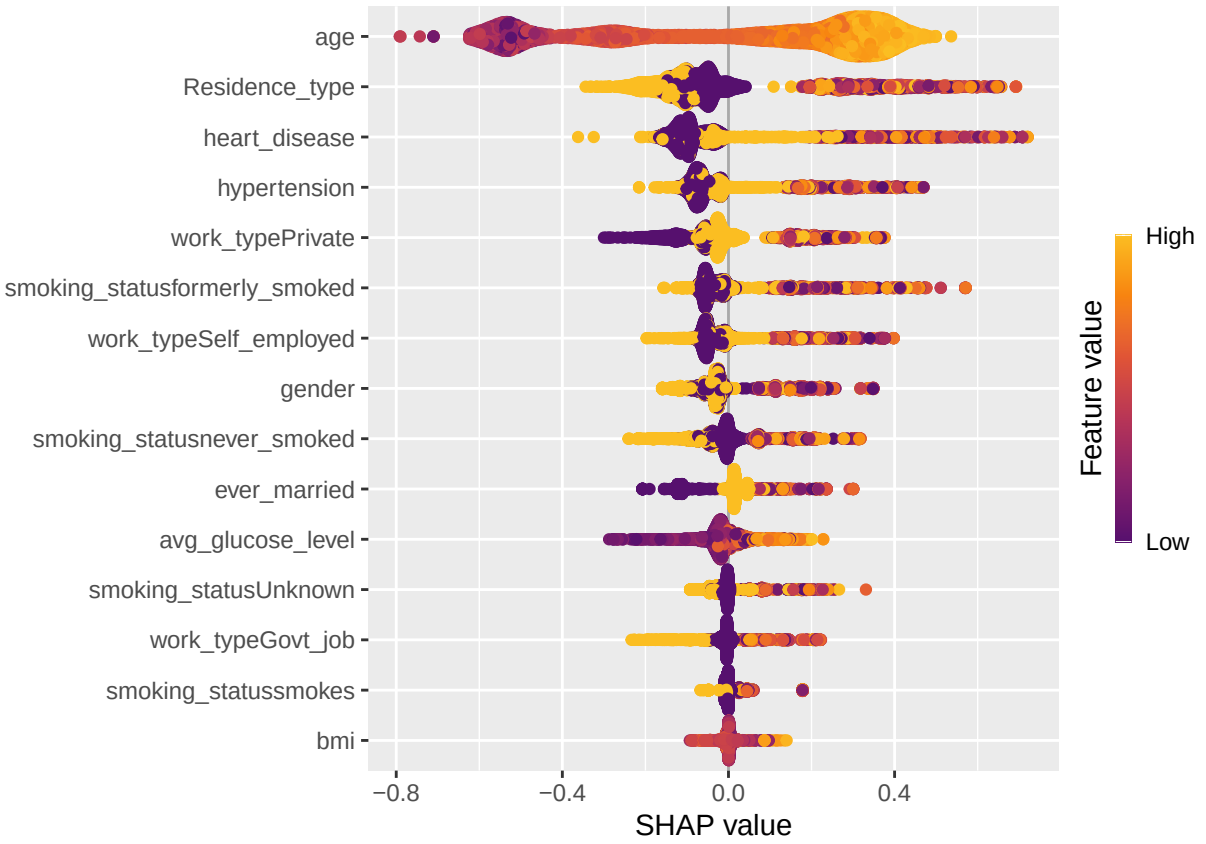
```
auc_value_xgboost<- auc(roc_obj_xgboost)
roc_obj_xgboost <- roc(response = factor(test_label, levels = c(0, 1)), predictor = pred_probs)
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

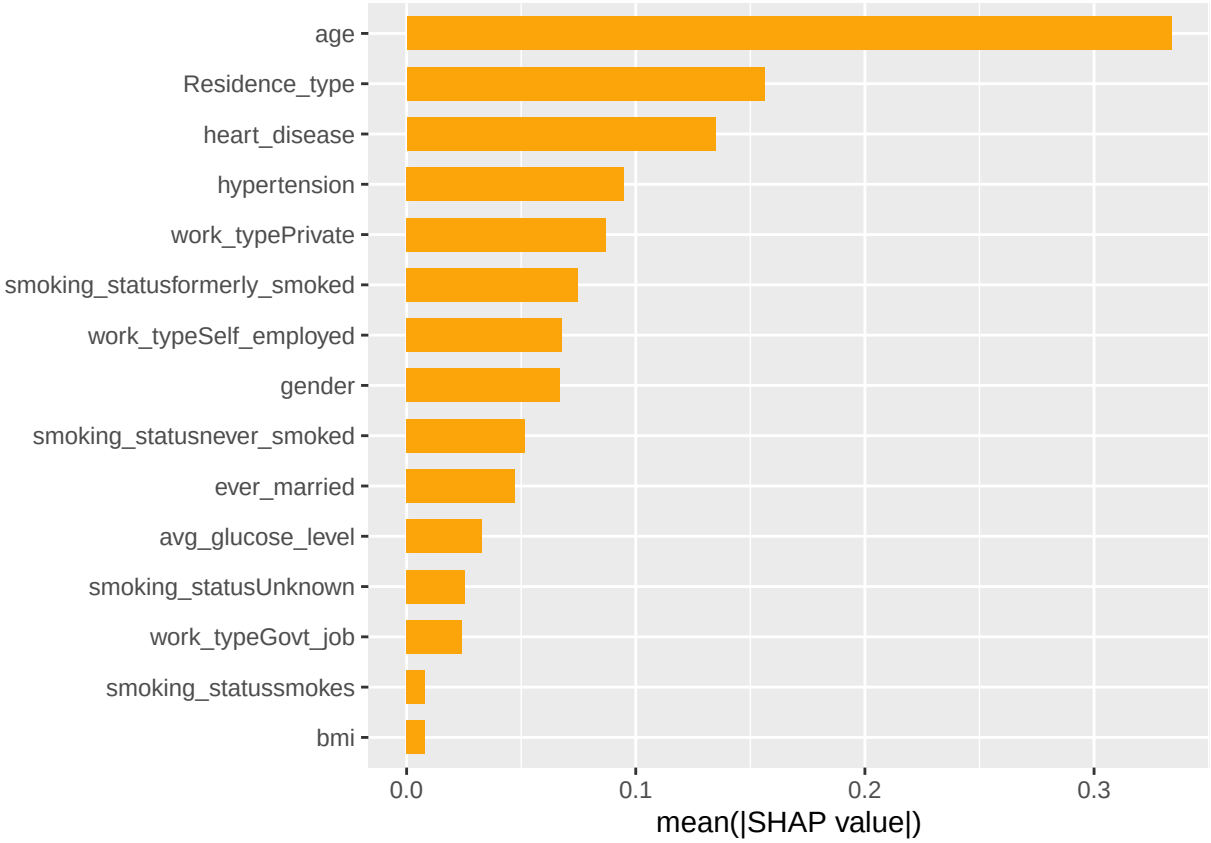
```
auc_value_xgboost <- auc(roc_obj_xgboost)
print(auc_value_xgboost)
```

```
## Area under the curve: 0.8265
```

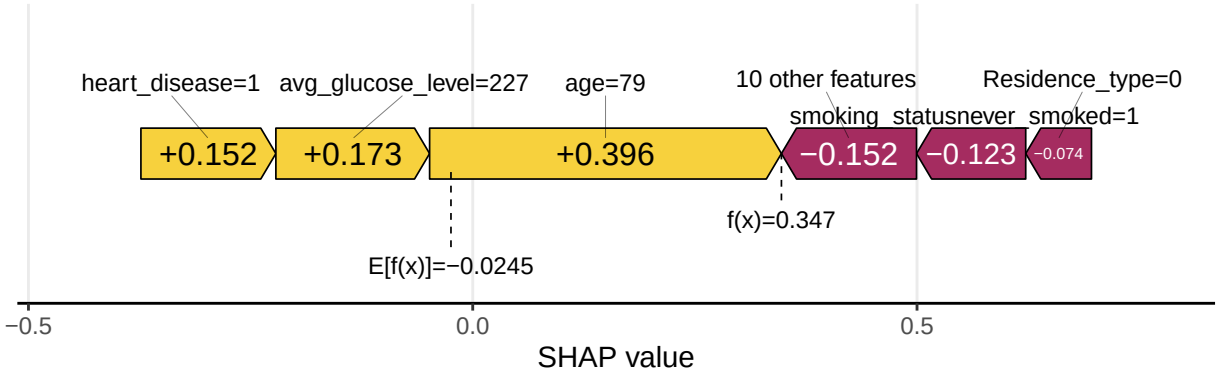
```
#shap plot
library(shapviz)
shp <- shapviz(model, train_data_xgboost )
sv_importance(shp, kind = "beeswarm")
```




```
sv_importance(shp)
```




```
sv_force(shp, row_id = 1)
```

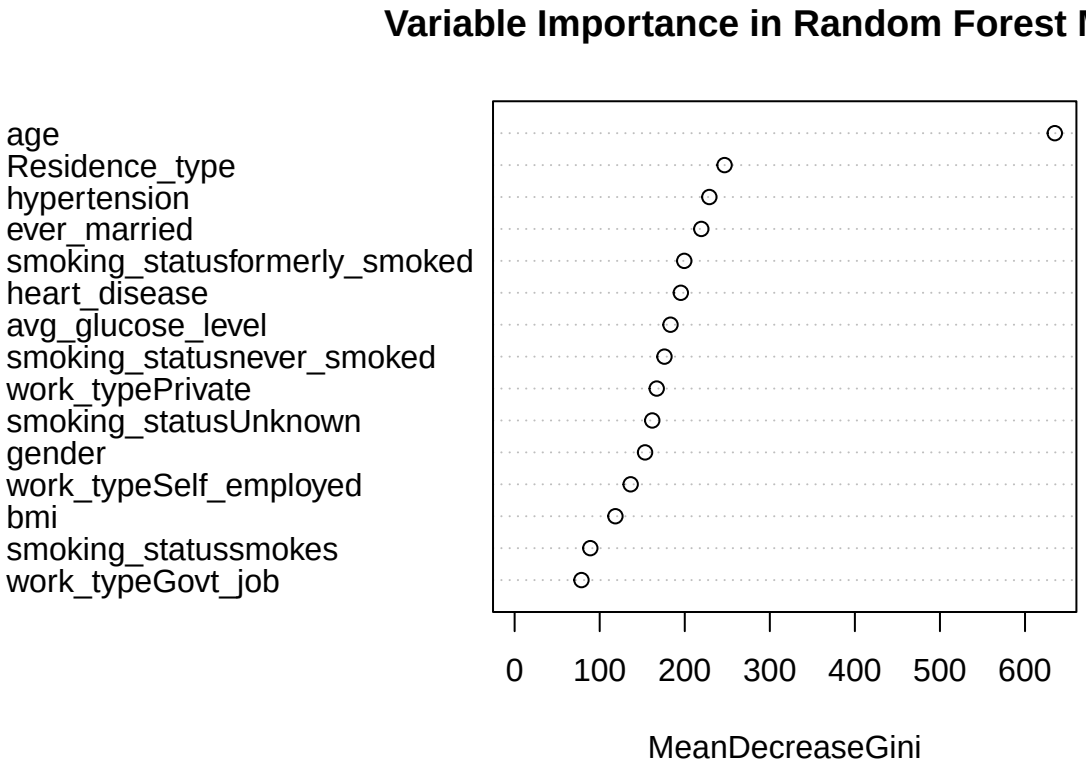



```
## Random forest
library(randomForest)
```

```
## randomForest 4.7-1.1
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```r
balanced_data_new$stroke <- as.factor(balanced_data_new$stroke )
test_data$stroke <- as.factor(test_data$stroke)

# RF-Train
rf_train <- randomForest(stroke ~ ., data = balanced_data_new, ntree = 500, importance = TRUE)
rf_predictions <- predict(rf_train, newdata = test_data)
rf_predictions_auc <- predict(rf_train, newdata = test_data, type = "prob")[,2]
varImpPlot(rf_train, type = 2, main = "Variable Importance in Random Forest Model")
```




```r
conf_matrix_rf<- confusionMatrix(rf_predictions, test_data$stroke)
print(conf_matrix_rf)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1410   70
##          1   12    2
##
##                Accuracy : 0.9451
##                  95% CI : (0.9323, 0.9561)
##     No Information Rate : 0.9518
##     P-Value [Acc > NIR] : 0.8959
##
##                   Kappa : 0.0313
##
##  Mcnemar's Test P-Value : 3.082e-10
##
##             Sensitivity : 0.99156
##             Specificity : 0.02778
```

```
##          Pos Pred Value : 0.95270
##          Neg Pred Value : 0.14286
##              Prevalence : 0.95181
##          Detection Rate : 0.94378
##    Detection Prevalence : 0.99063
##       Balanced Accuracy : 0.50967
##
##        'Positive' Class : 0
##
```

```
accuracy_rf <- conf_matrix_rf$overall['Accuracy']
precision_rf <- conf_matrix_rf$byClass['Precision']
recall_rf <- conf_matrix_rf$byClass['Sensitivity']
f1_score_rf <- 2 * (precision_rf * recall_rf) / (precision_rf + recall_rf)
print(accuracy_rf)
```

```
##  Accuracy
## 0.9451138
```

```
print(precision_rf)
```

```
## Precision
## 0.9527027
```

```
print(recall_rf)
```

```
## Sensitivity
##   0.9915612
```

```
print(f1_score_rf)
```

```
## Precision
## 0.9717436
```

```
roc_obj_rf <- roc(test_data$stroke, rf_predictions_auc)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
auc_value_rf <- auc(roc_obj_rf)
```

```
#knn
train_data <- balanced_data_std_new[, -which(names(balanced_data_std_new) == "stroke")]
train_labels <- factor(balanced_data_std_new$stroke)
test_data_without_stroke <- test_data_standardized[, -which(names(test_data_standardized) == "stroke")]
test_labels <- factor(test_data_standardized$stroke)

y_pre <- knn(train = train_data,
             test = test_data_without_stroke,
             cl = train_labels,
             k = 3)


y_pre <- factor(y_pre, levels = levels(test_labels))

c_m <- confusionMatrix(data = y_pre, reference = test_labels)
print(c_m)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1231   42
##          1  191   30
##
##                Accuracy : 0.844
##                  95% CI : (0.8246, 0.8621)
##     No Information Rate : 0.9518
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1424
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.8657
##             Specificity : 0.4167
##          Pos Pred Value : 0.9670
##          Neg Pred Value : 0.1357
##              Prevalence : 0.9518
##          Detection Rate : 0.8240
##    Detection Prevalence : 0.8521
##       Balanced Accuracy : 0.6412
##
##        'Positive' Class : 0
##
```

```
accuracy_knn <- c_m$overall['Accuracy']
precision_knn <- c_m$byClass['Precision']
recall_knn <- c_m$byClass['Sensitivity']
f1_score_knn <- 2 * (precision_knn * recall_knn) / (precision_knn + recall_knn)
print(paste("Accuracy:", accuracy_knn))
```

```
## [1] "Accuracy: 0.844042838018742"
```

```
print(paste("Precision:", precision_knn))
```

```
## [1] "Precision: 0.96700706991359"
```

```
print(paste("Recall:", recall_knn))
```

```
## [1] "Recall: 0.865682137834037"
```

```
print(paste("F1 Score:", f1_score_knn))
```

```
## [1] "F1 Score: 0.913543599257885"
```

```
knn_prob <- as.numeric(y_pre == "1")
roc_knn <- roc(response = as.numeric(test_labels), predictor = knn_prob)
```

```
## Setting levels: control = 1, case = 2
```

```
## Setting direction: controls < cases
```

```
auc_knn <- auc(roc_knn)
```

```
#SVM
library(e1071)
```

```r
library(caret)
library(pROC)

classifier <- svm(formula = stroke ~ .,
                  data = balanced_data_std_new,
                  type = 'C-classification',
                  kernel = 'linear')

test_data_without_stroke <- test_data_standardized[, -which(names(test_data_standardized) == "stroke")]
y_pre_svm <- predict(classifier, newdata = test_data_without_stroke)
test_labels <- factor(test_data_standardized[, "stroke"])
y_pre_svm2 <- factor(y_pre_svm, levels = levels(test_labels))


c_m <- confusionMatrix(data = y_pre, reference = test_labels)
print(c_m)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1231   42
##          1  191   30
##
##                Accuracy : 0.844
##                  95% CI : (0.8246, 0.8621)
##     No Information Rate : 0.9518
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1424
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.8657
##             Specificity : 0.4167
##          Pos Pred Value : 0.9670
##          Neg Pred Value : 0.1357
##              Prevalence : 0.9518
##          Detection Rate : 0.8240
##    Detection Prevalence : 0.8521
##       Balanced Accuracy : 0.6412
##
##        'Positive' Class : 0
##
```

```r
accuracy_svm <- c_m$overall['Accuracy']
precision_svm <- c_m$byClass['Precision']
recall_svm <- c_m$byClass['Sensitivity']
f1_score_svm <- 2 * (precision_svm * recall_svm) / (precision_svm + recall_svm)
print(paste("Accuracy:", accuracy_svm))
```

```
## [1] "Accuracy: 0.844042838018742"
```

```r
print(paste("Precision:", precision_svm))
```

```
## [1] "Precision: 0.96700706991359"
```

```
print(paste("Recall:", recall_svm))
```

```
## [1] "Recall: 0.865682137834037"
```

```
print(paste("F1 Score:", f1_score_svm))
```

```
## [1] "F1 Score: 0.913543599257885"
```

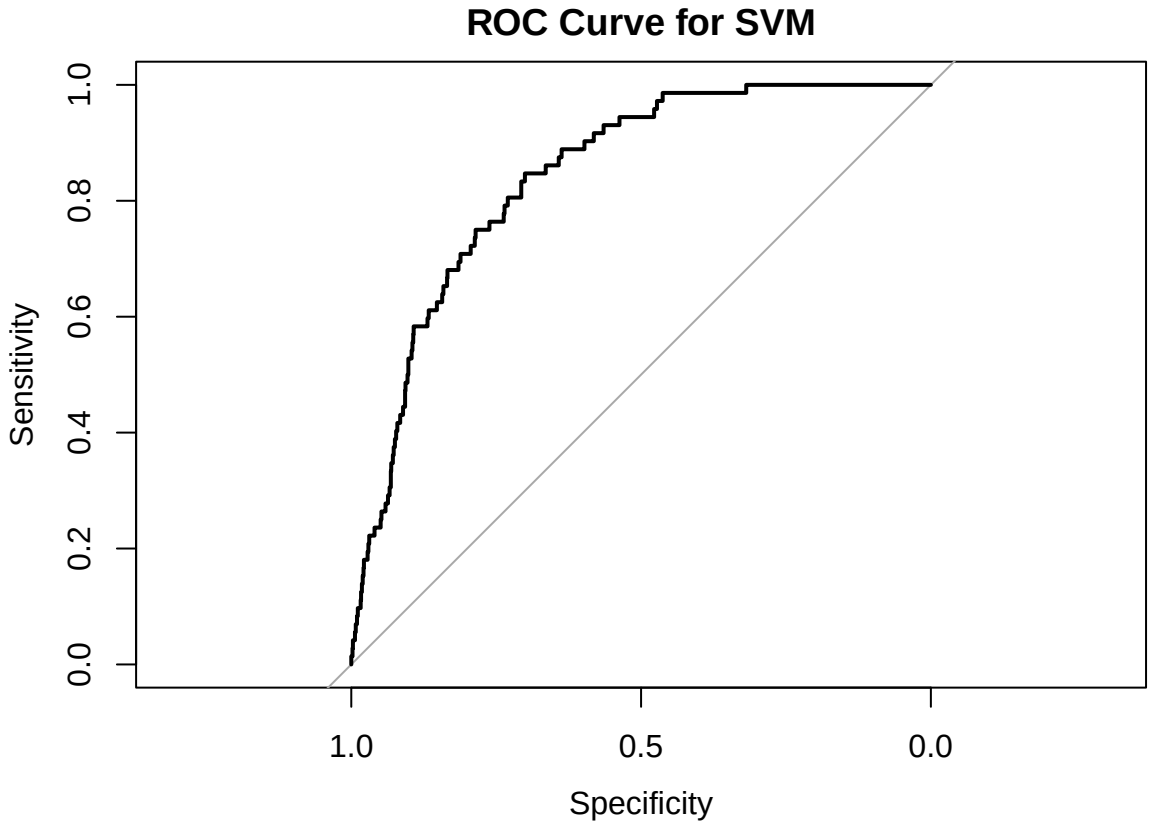
```
svm_scores <- predict(classifier, newdata = test_data_without_stroke, decision.values = TRUE)
decision_values <- attributes(svm_scores)$decision.values
roc_svm <- roc(response = as.numeric(test_labels), predictor = decision_values)
```

```
## Setting levels: control = 1, case = 2
```

```
## Warning in roc.default(response = as.numeric(test_labels), predictor =
## decision_values): Deprecated use a matrix as predictor. Unexpected results may
## be produced, please pass a numeric vector.
```

```
## Setting direction: controls < cases
```

```
auc_svm <- auc(roc_svm)
plot(roc_svm, main="ROC Curve for SVM")
```




```
print(paste("AUC for SVM:", auc_svm))
```

```
## [1] "AUC for SVM: 0.84223120800125"
```

```
#logistic Regression
logistic_model <- glm(stroke ~ . - smoking_statusUnknown, family = binomial(), data = balanced_data_std
summary(logistic_model)
```

```
##
## Call:
## glm(formula = stroke ~ . - smoking_statusUnknown, family = binomial(),
##     data = balanced_data_std_new)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7092  -0.6505  -0.1734   0.7713   2.6996
##
## Coefficients:
##                                Estimate Std. Error z value Pr(>|z|)
## (Intercept)                     0.57083    0.25594   2.230  0.02573 *
## gender                          0.18430    0.07044   2.616  0.00888 **
## age                             2.09987    0.06498  32.314  < 2e-16 ***
## hypertension                    0.15846    0.09755   1.624  0.10429
## heart_disease                   0.05521    0.11856   0.466  0.64144
## ever_married                   -0.03543    0.11917  -0.297  0.76624
## Residence_type                 -0.21041    0.06813  -3.088  0.00201 **
## avg_glucose_level               0.18199    0.02923   6.227 4.77e-10 ***
## bmi                             0.21193    0.04282   4.949 7.45e-07 ***
## smoking_statusformerly_smoked   0.17364    0.10315   1.683  0.09231 .
## smoking_statusnever_smoked     -0.29061    0.09629  -3.018  0.00254 **
## smoking_statussmokes            0.16992    0.11199   1.517  0.12919
## work_typeGovt_job              -2.48445    0.30253  -8.212  < 2e-16 ***
## work_typePrivate               -1.82418    0.29047  -6.280 3.38e-10 ***
## work_typeSelf_employed         -2.50577    0.30920  -8.104 5.32e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 8978.6  on 6478  degrees of freedom
## Residual deviance: 6009.3  on 6464  degrees of freedom
## AIC: 6039.3
##
## Number of Fisher Scoring iterations: 5
```

```
logi_pred <- predict(logistic_model, type = "response")
roc_logistic <- roc(balanced_data_std_new$stroke,logi_pred )
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
auc_logi <- auc(roc_logistic)
auc_logi
```

```
## Area under the curve: 0.851
```

```
pred_classes <- ifelse(logi_pred > 0.5, 1, 0)
accuracy <- sum(pred_classes == balanced_data_std_new$stroke) / nrow(balanced_data_std_new)
print(paste("Accuracy:", accuracy))
```

```
## [1] "Accuracy: 0.787621546534959"
```

```
## lasso
library(glmnet)
```

```r
library(pROC)
x <- model.matrix(stroke ~ . - 1, data = balanced_data_std_new)
summary(x)
```

```
##      gender            age             hypertension    heart_disease
##  Min.   :0.0000   Min.   :-1.92260   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:-0.09383   1st Qu.:0.0000   1st Qu.:0.0000
##  Median :1.0000   Median : 0.71797   Median :0.0000   Median :0.0000
##  Mean   :0.5722   Mean   : 0.48986   Mean   :0.1472   Mean   :0.1051
##  3rd Qu.:1.0000   3rd Qu.: 1.29753   3rd Qu.:0.0000   3rd Qu.:0.0000
##  Max.   :1.0000   Max.   : 1.69537   Max.   :1.0000   Max.   :1.0000
##   ever_married    Residence_type   avg_glucose_level      bmi
##  Min.   :0.0000   Min.   :0.0000   Min.   :-1.1247   Min.   :-2.14279
##  1st Qu.:1.0000   1st Qu.:0.0000   1st Qu.:-0.5733   1st Qu.:-0.40609
##  Median :1.0000   Median :0.1835   Median :-0.2091   Median : 0.06587
##  Mean   :0.7768   Mean   :0.4584   Mean   : 0.2877   Mean   : 0.12957
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.: 0.8560   3rd Qu.: 0.58018
##  Max.   :1.0000   Max.   :1.0000   Max.   : 3.6087   Max.   : 2.99373
##  smoking_statusformerly_smoked smoking_statusnever_smoked smoking_statussmokes
##  Min.   :0.0000                Min.   :0.0000             Min.   :0.000
##  1st Qu.:0.0000                1st Qu.:0.0000             1st Qu.:0.000
##  Median :0.0000                Median :0.0000             Median :0.000
##  Mean   :0.2377                Mean   :0.3623             Mean   :0.163
##  3rd Qu.:0.2690                3rd Qu.:1.0000             3rd Qu.:0.000
##  Max.   :1.0000                Max.   :1.0000             Max.   :1.000
##  smoking_statusUnknown work_typeGovt_job work_typePrivate
##  Min.   :0.0000        Min.   :0.0000    Min.   :0.0000
##  1st Qu.:0.0000        1st Qu.:0.0000    1st Qu.:0.0000
##  Median :0.0000        Median :0.0000    Median :1.0000
##  Mean   :0.2370        Mean   :0.1279    Mean   :0.6082
##  3rd Qu.:0.1171        3rd Qu.:0.0000    3rd Qu.:1.0000
##  Max.   :1.0000        Max.   :1.0000    Max.   :1.0000
##  work_typeSelf_employed
##  Min.   :0.0000
##  1st Qu.:0.0000
##  Median :0.0000
##  Mean   :0.1898
##  3rd Qu.:0.0000
##  Max.   :1.0000
```

```r
y <- balanced_data_std_new$stroke
set.seed(123)
cv_model <- cv.glmnet(x, y, alpha = 1, family = "binomial")
cv_model
```

```
##
## Call:  cv.glmnet(x = x, y = y, alpha = 1, family = "binomial")
##
## Measure: Binomial Deviance
##
##       Lambda Index Measure       SE Nonzero
## min 0.000246    77  0.9327 0.011566      14
## 1se 0.009250    38  0.9438 0.009609      11
```

```r
lasso_pred <- predict(cv_model, newx = x, type = "response", s = cv_model$lambda.min)
roc_lasso <- roc(response = y, predictor = as.vector(lasso_pred))
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

```r
auc_lasso <- auc(roc_lasso)
print(auc_lasso)
```

```
## Area under the curve: 0.8509
```

```r
## deep neutral network
train_indices<- createDataPartition(df2_new$stroke, p = .7, list = FALSE, times = 1)
train_data <- df2_new[train_indices, ]
test_data <- df2_new[-train_indices, ]

library(caret)

scale01 <- function(x) {
  return ((x - min(x)) / (max(x) - min(x)))
}
train_data_normalized <- train_data
train_data_normalized[, 2] <- scale01(train_data[, 2])
train_data_normalized[, 7:8] <- apply(train_data[, 7:8], 2, scale01)

head(train_data_normalized)
```

```
##   gender       age hypertension heart_disease ever_married Residence_type
## 1      0 0.8167155            0             1            1              0
## 2      0 0.9755621            0             1            1              1
## 3      1 0.5967742            0             0            1              0
## 6      0 0.9022483            1             1            1              1
## 7      1 0.8411535            0             0            0              0
## 9      1 0.9877810            1             0            1              1
##   avg_glucose_level       bmi stroke smoking_statusformerly_smoked
## 1        0.81626223 0.6475645      1                             1
## 2        0.23890143 0.5300860      1                             0
## 3        0.54604026 0.5845272      1                             0
## 6        0.07040068 0.3839542      1                             0
## 7        0.18467833 0.2521490      1                             0
## 9        0.11902746 0.4498567      1                             0
##   smoking_statusnever_smoked smoking_statussmokes smoking_statusUnknown
## 1                          0                    0                    0
## 2                          1                    0                    0
## 3                          0                    1                    0
## 6                          1                    0                    0
## 7                          1                    0                    0
## 9                          1                    0                    0
##   work_typeGovt_job work_typePrivate work_typeSelf_employed
## 1                 0                1                      0
## 2                 0                1                      0
## 3                 0                1                      0
## 6                 0                1                      0
## 7                 0                1                      0
## 9                 0                1                      0
```

```
test_data_normalized <- test_data
test_data_normalized[, 2] <- scale01(test_data[, 2])
test_data_normalized[, 7:8] <- apply(test_data[, 7:8], 2, scale01)

head(test_data_normalized)
```

```
##    gender       age hypertension heart_disease ever_married Residence_type
## 4       1 0.9633789            1             0            1              1
## 5       0 0.9877930            0             0            1              0
## 8       1 0.9511719            0             0            1              0
## 17      1 0.8657227            0             0            1              1
## 19      1 0.9633789            0             0            1              0
## 22      0 0.9755859            0             0            1              1
##    avg_glucose_level       bmi stroke smoking_statusformerly_smoked
## 4         0.54912013 0.2853026      1                             0
## 5         0.60496051 0.4293948      1                             1
## 8         0.01542654 0.2910663      1                             0
## 17        0.64066325 0.2391931      1                             0
## 19        0.80121011 0.3602305      1                             0
## 22        0.22580943 0.2708934      1                             0
##    smoking_statusnever_smoked smoking_statussmokes smoking_statusUnknown
## 4                           1                    0                    0
## 5                           0                    0                    0
## 8                           0                    0                    1
## 17                          0                    1                    0
## 19                          1                    0                    0
## 22                          1                    0                    0
##    work_typeGovt_job work_typePrivate work_typeSelf_employed
## 4                  0                0                      1
## 5                  0                1                      0
## 8                  0                1                      0
## 17                 1                0                      0
## 19                 0                0                      1
## 22                 0                0                      1
```

```
table(train_data_normalized$stroke)
```

```
##
##    0    1
## 3326  161
```

```
#smote
library(smotefamily)
smote_output_nor <- SMOTE(train_data_normalized, train_data_normalized$stroke, K = 5, dup_size = 17.8)
balanced_data_normalized <- smote_output_nor$data
balanced_data_nor_new <- subset(balanced_data_normalized, select = -c(class))
table(balanced_data_normalized$stroke)
```

```
##
##    0    1
## 3326 2898
```

```
x_train <- as.matrix(subset(balanced_data_nor_new, select = -stroke))
y_train <-balanced_data_nor_new$stroke
x_test <- as.matrix(subset(test_data_normalized, select = -stroke))
y_test <- test_data_normalized$stroke
```