Data Science II Final Project

Yuechen Liu, Yanhao Li, Mufeng Xu

Contents

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.1.0     v dplyr   1.0.5
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(patchwork)
library(ggcorrplot)
library(ISLR)
library(caret)
```

```
## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(randomForest)
```

```
## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:dplyr':
##
##     combine

## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(MASS)
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:patchwork':
##
##     area

## The following object is masked from 'package:dplyr':
##
##     select
```

```
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
library(AppliedPredictiveModeling)
```

```
# Load and tidy
stroke_df = read_csv("healthcare-dataset-stroke-data.csv") %>%
  dplyr::select(gender:stroke) %>%
  filter(!bmi == "N/A") %>%
  na.omit() %>%
  janitor::clean_names() %>%
  mutate(
    gender = case_when(gender == "Male" ~ 0,
                       gender == "Female" ~ 1,
                       gender == "Other" ~ 2),
    ever_married = case_when(ever_married == "No" ~ 0,
                             ever_married == "Yes" ~ 1),
    work_type = case_when(work_type == "children" ~ 0,
                          work_type == "Govt_job" ~ 1,
                          work_type == "Never_worked" ~ 2,
                          work_type == "Private" ~ 3,
                          work_type == "Self-employed" ~ 4),
    residence_type = case_when(residence_type == "Rural" ~ 0,
                               residence_type == "Urban" ~ 1),
    smoking_status = case_when(smoking_status == "formerly smoked" ~ 0,
                               smoking_status == "never smoked" ~ 1,
                               smoking_status == "smokes" ~ 2,
                               smoking_status == "Unknown" ~ 3)
  ) %>%
  mutate(bmi = as.numeric(bmi),
         stroke = as.factor(stroke))
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
```

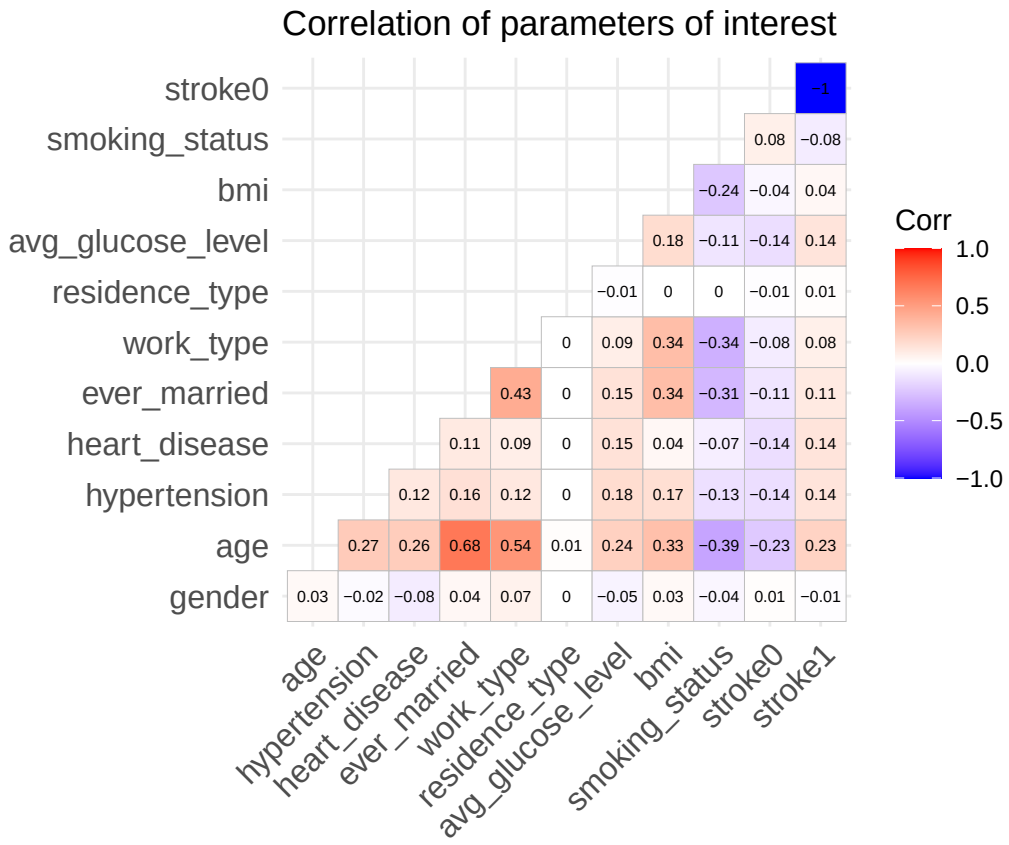
```
##   id = col_double(),
##   gender = col_character(),
##   age = col_double(),
##   hypertension = col_double(),
##   heart_disease = col_double(),
##   ever_married = col_character(),
##   work_type = col_character(),
##   Residence_type = col_character(),
##   avg_glucose_level = col_double(),
##   bmi = col_character(),
##   smoking_status = col_character(),
##   stroke = col_double()
## )
```

```
summary(stroke_df)
```

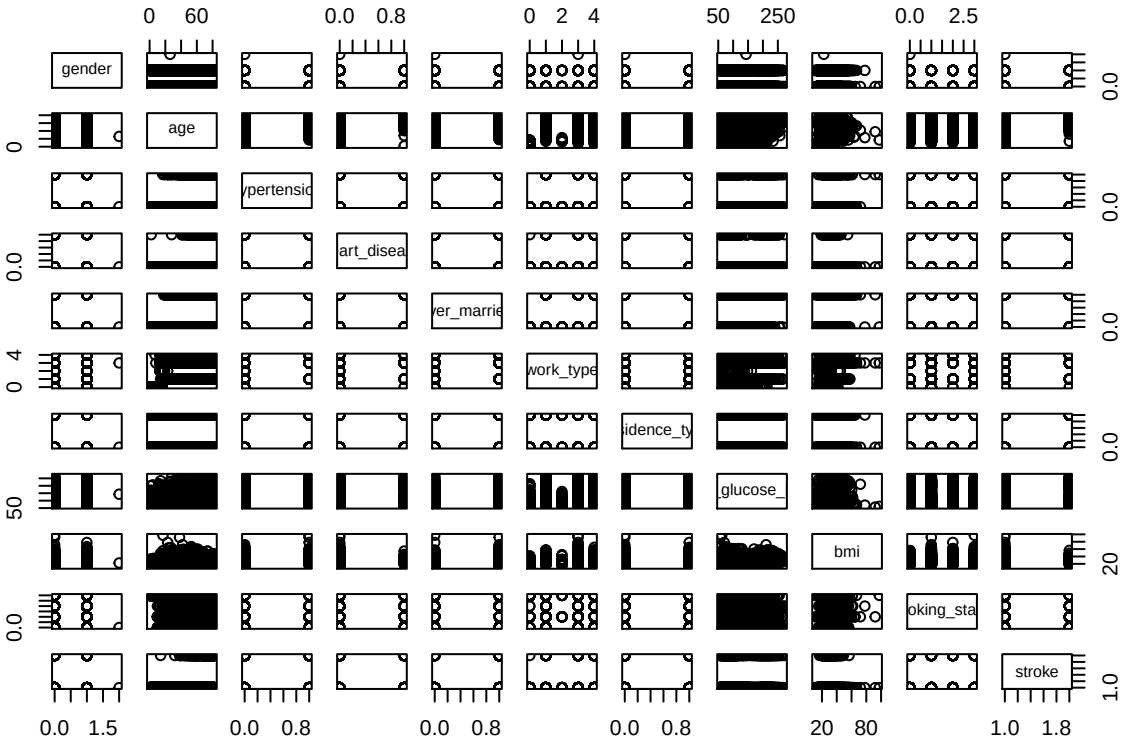
```
##      gender            age         hypertension     heart_disease
##  Min.   :0.0000   Min.   : 0.08   Min.   :0.00000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:25.00   1st Qu.:0.00000   1st Qu.:0.0000
##  Median :1.0000   Median :44.00   Median :0.00000   Median :0.0000
##  Mean   :0.5905   Mean   :42.87   Mean   :0.09187   Mean   :0.0495
##  3rd Qu.:1.0000   3rd Qu.:60.00   3rd Qu.:0.00000   3rd Qu.:0.0000
##  Max.   :2.0000   Max.   :82.00   Max.   :1.00000   Max.   :1.0000
##   ever_married      work_type     residence_type   avg_glucose_level
##  Min.   :0.0000   Min.   :0.000   Min.   :0.0000   Min.   : 55.12
##  1st Qu.:0.0000   1st Qu.:1.000   1st Qu.:0.0000   1st Qu.: 77.07
##  Median :1.0000   Median :3.000   Median :1.0000   Median : 91.68
##  Mean   :0.6527   Mean   :2.487   Mean   :0.5072   Mean   :105.31
##  3rd Qu.:1.0000   3rd Qu.:3.000   3rd Qu.:1.0000   3rd Qu.:113.57
##  Max.   :1.0000   Max.   :4.000   Max.   :1.0000   Max.   :271.74
##       bmi         smoking_status  stroke
##  Min.   :10.30   Min.   :0.000   0:4700
##  1st Qu.:23.50   1st Qu.:1.000   1: 209
##  Median :28.10   Median :1.000
##  Mean   :28.89   Mean   :1.584
##  3rd Qu.:33.10   3rd Qu.:3.000
##  Max.   :97.60   Max.   :3.000
```

EDA

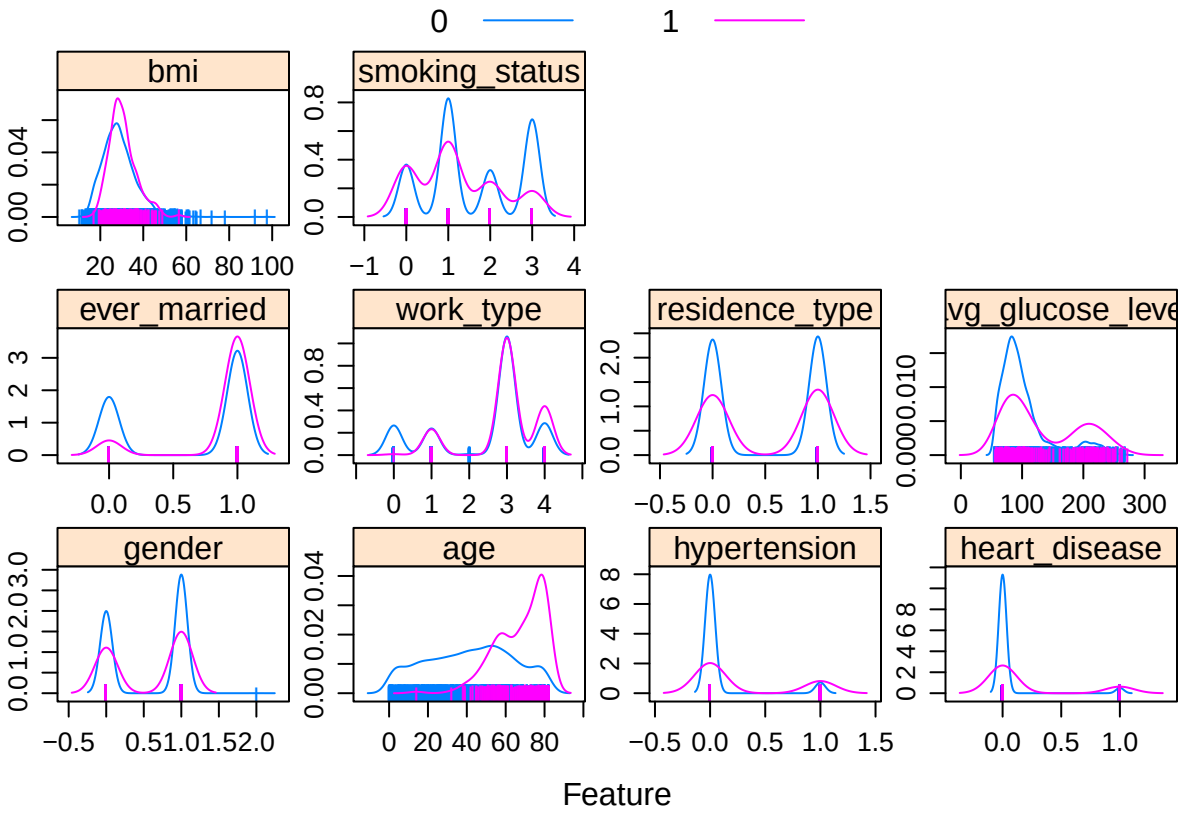
```
# Correlation Matrix
model.matrix(~0+., data = stroke_df) %>%
  cor(use = "pairwise.complete.obs") %>%
  ggcorrplot(show.diag = F, type = "lower",
             lab = TRUE, lab_size = 2,
             title = "Correlation of parameters of interest")
```




```
# scatter plots between variables
pairs(stroke_df) # pairs plot
```

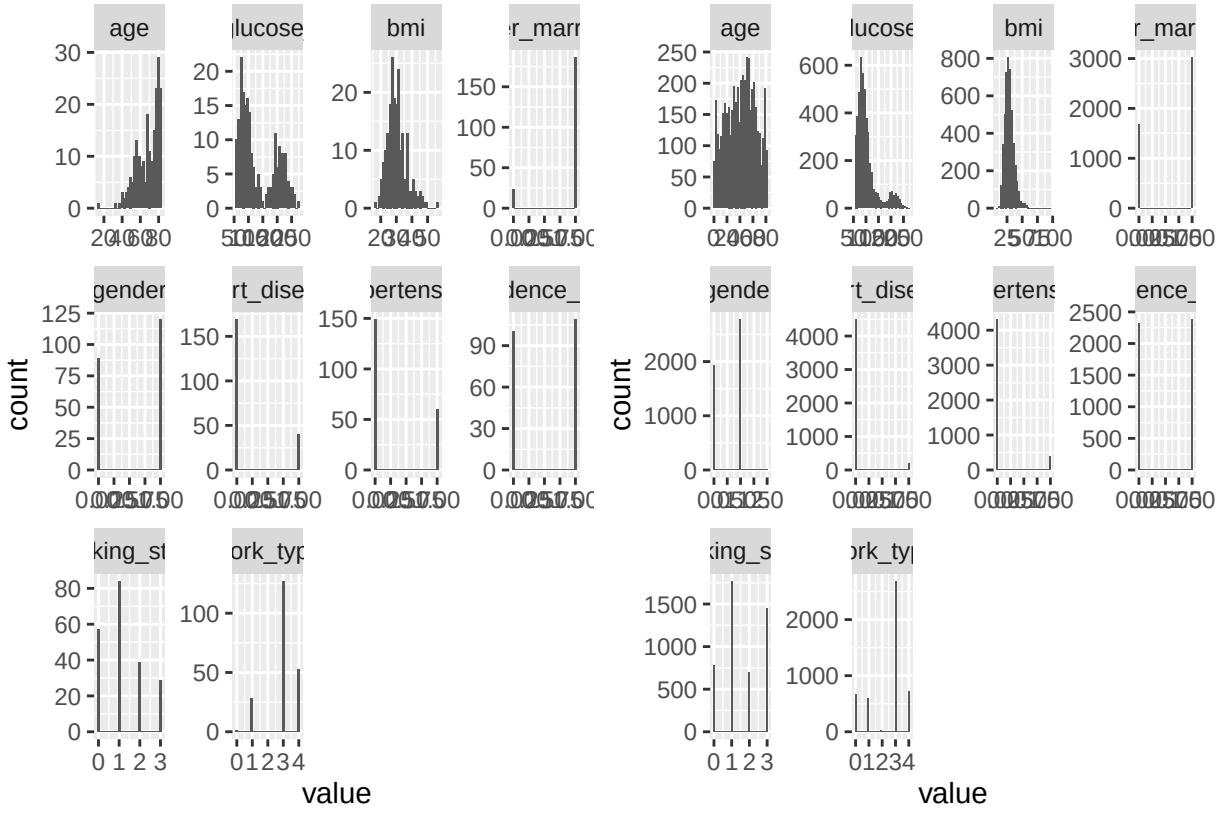



```
# density plots
featurePlot(x = stroke_df[, 1:10],
            y = stroke_df$stroke,
            scales = list(x = list(relation = "free"),
                          y = list(relation = "free")),
            plot = "density",
            pch = "|",
            auto.key = list(columns = 2),
            font = 2) # density plot
```




```
# Histogram
stroke_plot = stroke_df %>%
  filter(stroke == 1) %>%
  keep(is.numeric) %>%
  gather() %>%
  ggplot(aes(value)) +
  facet_wrap(~ key, scales = "free") +
  geom_histogram()

nonstroke_plot = stroke_df %>%
  filter(stroke == 0) %>%
  keep(is.numeric) %>%
  gather() %>%
  ggplot(aes(value)) +
  facet_wrap(~ key, scales = "free") +
  geom_histogram()

stroke_plot + nonstroke_plot
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Models

```
# split train vs. test
stroke_df = stroke_df %>%
  mutate(gender = as.factor(gender),
         hypertension = as.factor(hypertension),
         heart_disease = as.factor(heart_disease),
         ever_married = as.factor(ever_married),
         work_type = as.factor(work_type),
         residence_type = as.factor(residence_type),
         smoking_status = as.factor(smoking_status),
         stroke = as.factor(stroke)) %>%
  mutate(id = 1:nrow(stroke_df))

set.seed(2)
sample_size = floor(0.8 * nrow(stroke_df))
train = sample_n(stroke_df, size = sample_size)
test = anti_join(stroke_df, train, by = 'id') %>% dplyr::select(gender:stroke)
train = train %>% dplyr::select(gender:stroke)
stroke_df = stroke_df %>% dplyr::select(gender:stroke)
```

GLM

```
glm_fit = glm(stroke ~., data = train, family = binomial)
summary(glm_fit)
```

```
##
## Call:
## glm(formula = stroke ~ ., family = binomial, data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.1795  -0.2938  -0.1547  -0.0779   3.4928
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -7.105e+00  1.081e+00  -6.571 4.99e-11 ***
## gender1           -4.513e-02  1.738e-01  -0.260  0.79514
## gender2           -1.153e+01  2.400e+03  -0.005  0.99617
## age                6.914e-02  7.073e-03   9.776  < 2e-16 ***
## hypertension1      5.576e-01  1.966e-01   2.836  0.00456 **
## heart_disease1     4.300e-01  2.315e-01   1.858  0.06321 .
## ever_married1     -1.157e-01  2.677e-01  -0.432  0.66567
## work_type1        -5.661e-01  1.140e+00  -0.497  0.61941
## work_type2        -1.088e+01  5.216e+02  -0.021  0.98336
## work_type3        -3.952e-01  1.122e+00  -0.352  0.72457
## work_type4        -6.632e-01  1.144e+00  -0.580  0.56217
## residence_type1   -1.361e-02  1.689e-01  -0.081  0.93578
## avg_glucose_level  4.797e-03  1.464e-03   3.278  0.00105 **
## bmi               -8.128e-04  1.354e-02  -0.060  0.95213
## smoking_status1   -1.168e-01  2.165e-01  -0.540  0.58932
## smoking_status2    2.554e-01  2.586e-01   0.987  0.32347
## smoking_status3   -1.679e-01  2.704e-01  -0.621  0.53463
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1362.7  on 3926  degrees of freedom
## Residual deviance: 1082.1  on 3910  degrees of freedom
## AIC: 1116.1
##
## Number of Fisher Scoring iterations: 15
```

```
step_glm_fit = stepAIC(glm_fit, direction = "both", trace = FALSE)
summary(step_glm_fit)
```

```
##
## Call:
## glm(formula = stroke ~ age + hypertension + heart_disease + avg_glucose_level,
##     family = binomial, data = train)
##
## Deviance Residuals:
```

```
##     Min       1Q   Median       3Q      Max
## -1.1175  -0.2952  -0.1606  -0.0794   3.5631
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -7.537711   0.428935 -17.573  < 2e-16 ***
## age                0.065006   0.006195  10.493  < 2e-16 ***
## hypertension1      0.561298   0.194297   2.889 0.003866 **
## heart_disease1     0.495440   0.226841   2.184 0.028956 *
## avg_glucose_level  0.004863   0.001417   3.431 0.000602 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1362.7  on 3926  degrees of freedom
## Residual deviance: 1087.3  on 3922  degrees of freedom
## AIC: 1097.3
##
## Number of Fisher Scoring iterations: 7
```

```
test = test %>% dplyr::select(age, hypertension, heart_disease, avg_glucose_level, stroke)
train = train %>% dplyr::select(age, hypertension, heart_disease, avg_glucose_level, stroke)

# Confusion Matrix
prob = predict(step_glm_fit, newdata = test, type = "response")
pred = rep("0", length(prob))
pred[prob > 0.5] = "1"
confusionMatrix(data = as.factor(pred), reference = test$stroke, positive = "1")
```

```
## Warning in confusionMatrix.default(data = as.factor(pred), reference =
## test$stroke, : Levels are not in the same order for reference and data.
## Refactoring data to match.
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 937  45
##          1   0   0
##
##                Accuracy : 0.9542
##                  95% CI : (0.9392, 0.9664)
##     No Information Rate : 0.9542
##     P-Value [Acc > NIR] : 0.5395
##
##                   Kappa : 0
##
##  Mcnemar's Test P-Value : 5.412e-11
##
##             Sensitivity : 0.00000
##             Specificity : 1.00000
##          Pos Pred Value :     NaN
```

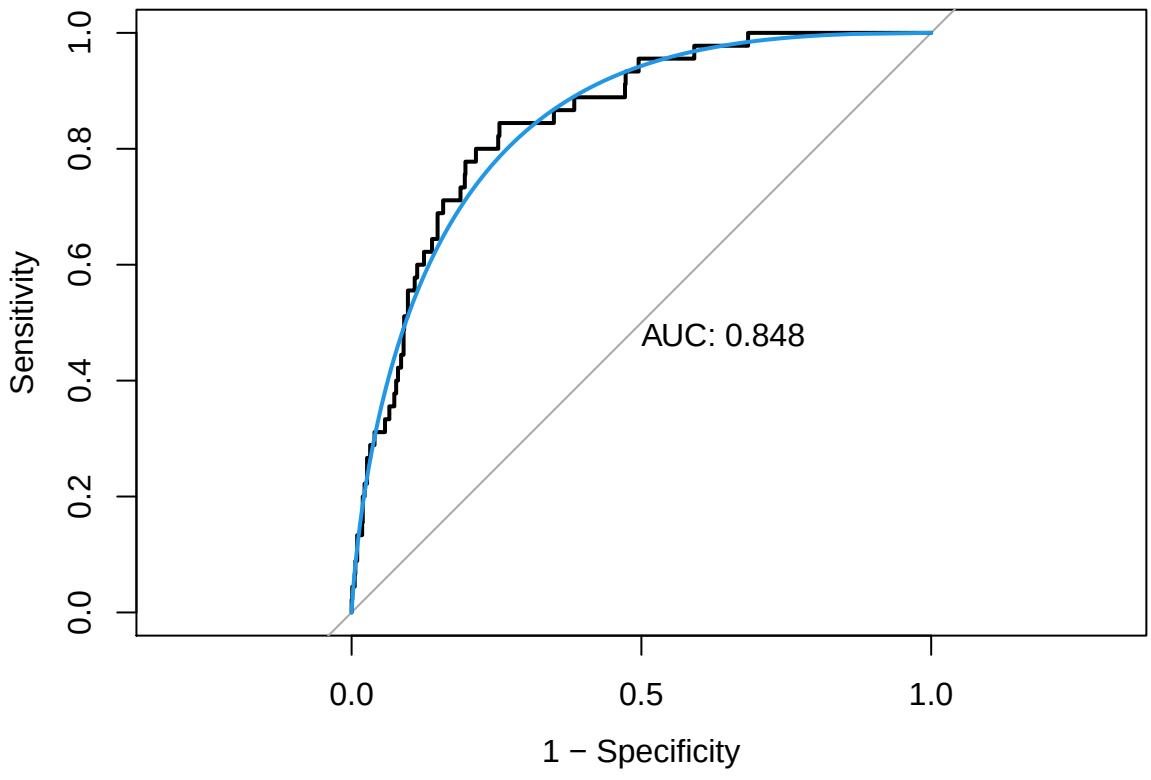
```
##          Neg Pred Value : 0.95418
##              Prevalence : 0.04582
##          Detection Rate : 0.00000
##    Detection Prevalence : 0.00000
##       Balanced Accuracy : 0.50000
##
##        'Positive' Class : 1
##
```

```
# ROC
glm_prob2 = predict(step_glm_fit, newdata = test, type = "response")
roc = roc(test$stroke, glm_prob2)
```

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

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

```
plot(roc, legacy.axes = TRUE, print.auc = TRUE)
plot(smooth(roc), col = 4, add = TRUE)
```

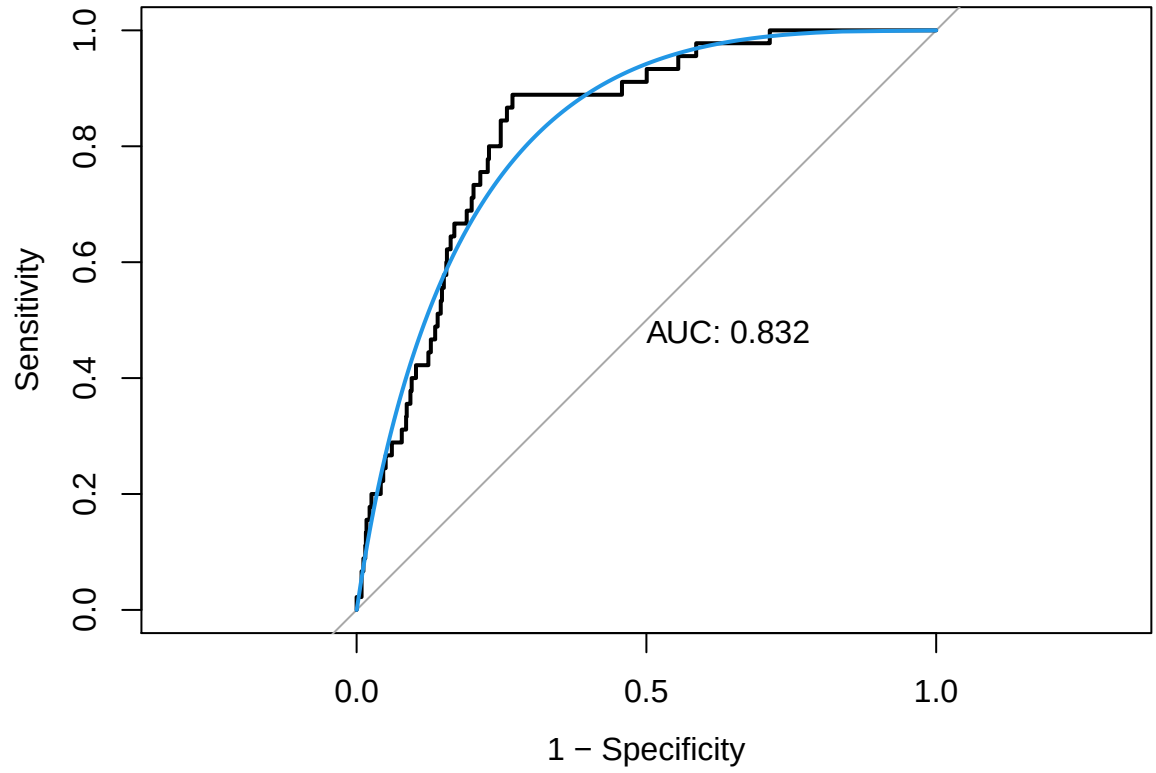


LDA

```
lda_fit = lda(stroke ~ age + hypertension + heart_disease + avg_glucose_level, data = train)
lda_pred = predict(lda_fit, newdata = test)
roc_lda = roc(test$stroke, lda_pred$posterior[,2], levels = c("0", "1"))
```

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

```
plot(roc_lda, legacy.axes = TRUE, print.auc = TRUE)
plot(smooth(roc_lda), col = 4, add = TRUE)
```

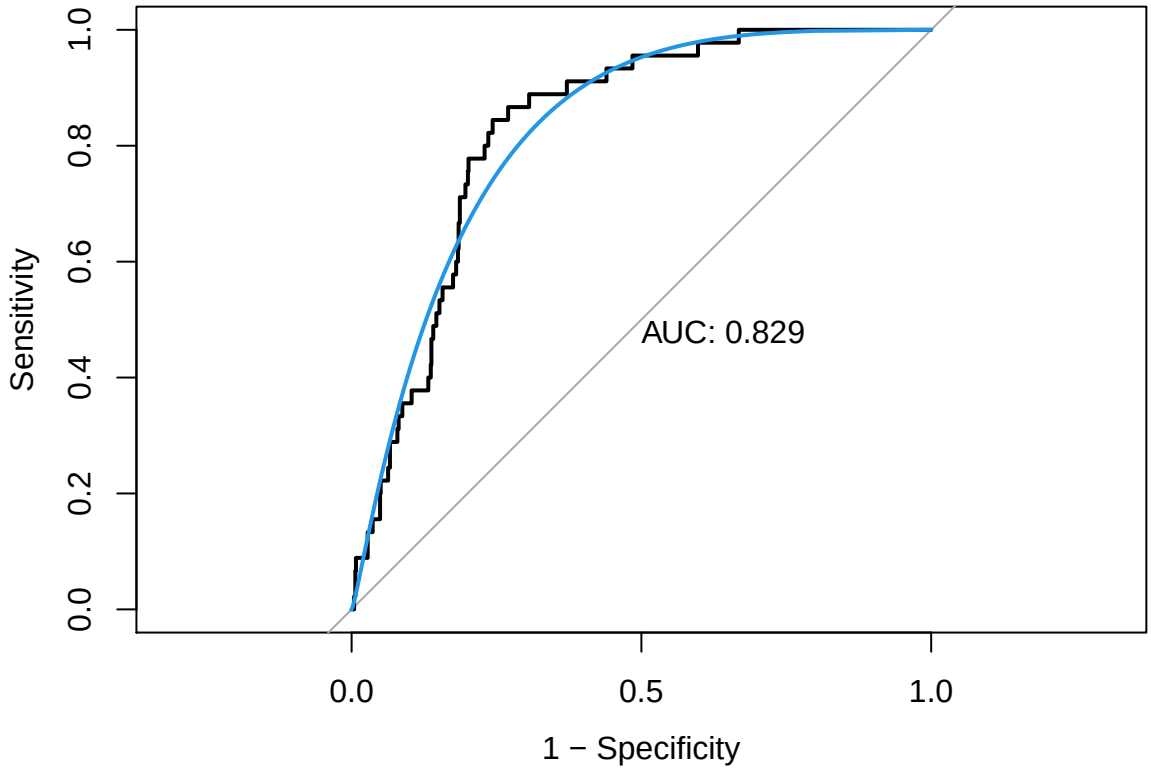


QDA

```
qda_fit = qda(stroke ~., data = train)
qda_pred = predict(qda_fit, newdata = test)
roc_qda = roc(test$stroke, qda_pred$posterior[,2], levels = c("0", "1"))
```

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

```
plot(roc_qda, legacy.axes = TRUE, print.auc = TRUE)
plot(smooth(roc_qda), col = 4, add = TRUE)
```



RandomForest

```
rfGrid = data.frame(
  .mtry = c(2,3),
  .splitrule = "gini",
  .min.node.size = 5
)

ctrl1 = trainControl(
    method = "cv",
    number = 10,
    verboseIter = TRUE
)

rf_model = train(
    stroke ~ age + hypertension + heart_disease + avg_glucose_level,
    data = train,
    method = "ranger",
    tuneLength = 3,
    tuneGrid = rfGrid,
    trControl = ctrl1
)
```

```
## + Fold01: mtry=2, splitrule=gini, min.node.size=5
```

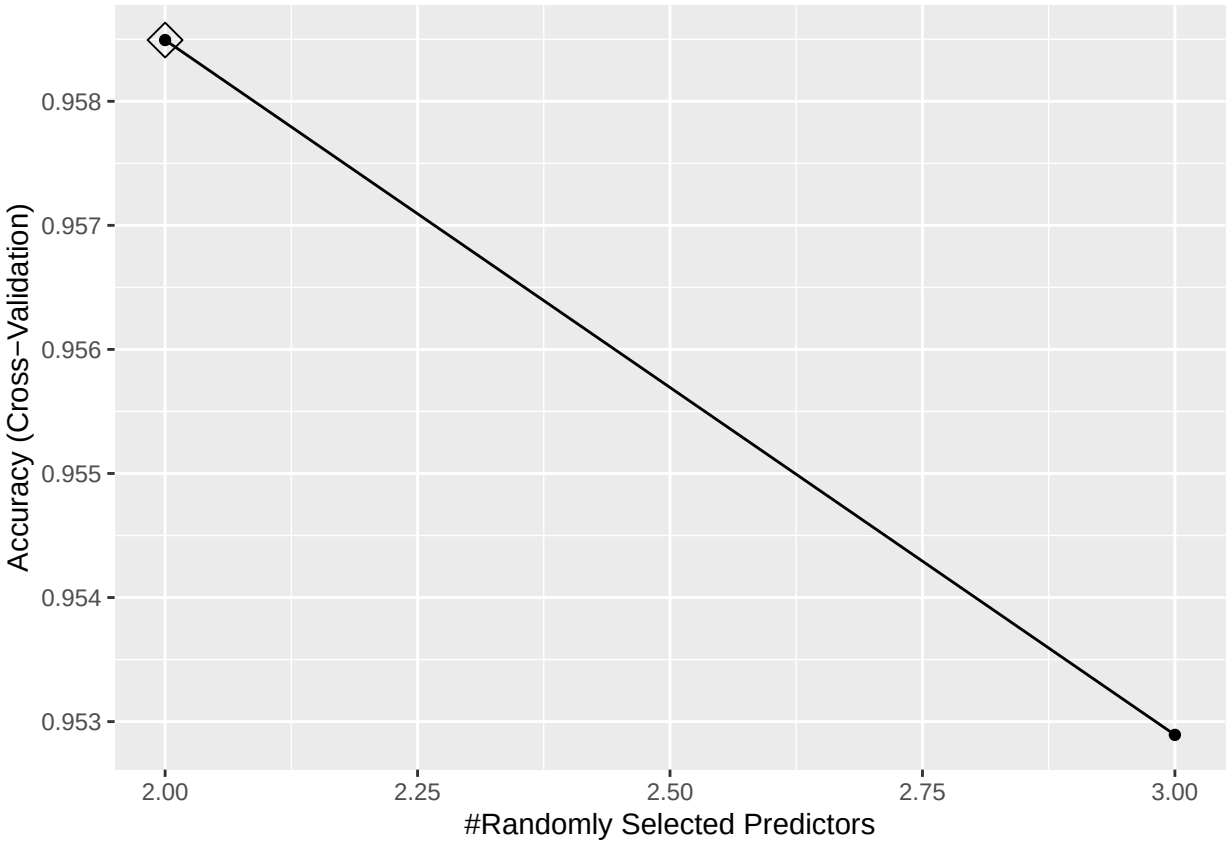
```
## - Fold01: mtry=2, splitrule=gini, min.node.size=5
## + Fold01: mtry=3, splitrule=gini, min.node.size=5
## - Fold01: mtry=3, splitrule=gini, min.node.size=5
## + Fold02: mtry=2, splitrule=gini, min.node.size=5
## - Fold02: mtry=2, splitrule=gini, min.node.size=5
## + Fold02: mtry=3, splitrule=gini, min.node.size=5
## - Fold02: mtry=3, splitrule=gini, min.node.size=5
## + Fold03: mtry=2, splitrule=gini, min.node.size=5
## - Fold03: mtry=2, splitrule=gini, min.node.size=5
## + Fold03: mtry=3, splitrule=gini, min.node.size=5
## - Fold03: mtry=3, splitrule=gini, min.node.size=5
## + Fold04: mtry=2, splitrule=gini, min.node.size=5
## - Fold04: mtry=2, splitrule=gini, min.node.size=5
## + Fold04: mtry=3, splitrule=gini, min.node.size=5
## - Fold04: mtry=3, splitrule=gini, min.node.size=5
## + Fold05: mtry=2, splitrule=gini, min.node.size=5
## - Fold05: mtry=2, splitrule=gini, min.node.size=5
## + Fold05: mtry=3, splitrule=gini, min.node.size=5
## - Fold05: mtry=3, splitrule=gini, min.node.size=5
## + Fold06: mtry=2, splitrule=gini, min.node.size=5
## - Fold06: mtry=2, splitrule=gini, min.node.size=5
## + Fold06: mtry=3, splitrule=gini, min.node.size=5
## - Fold06: mtry=3, splitrule=gini, min.node.size=5
## + Fold07: mtry=2, splitrule=gini, min.node.size=5
## - Fold07: mtry=2, splitrule=gini, min.node.size=5
## + Fold07: mtry=3, splitrule=gini, min.node.size=5
## - Fold07: mtry=3, splitrule=gini, min.node.size=5
## + Fold08: mtry=2, splitrule=gini, min.node.size=5
## - Fold08: mtry=2, splitrule=gini, min.node.size=5
## + Fold08: mtry=3, splitrule=gini, min.node.size=5
## - Fold08: mtry=3, splitrule=gini, min.node.size=5
## + Fold09: mtry=2, splitrule=gini, min.node.size=5
## - Fold09: mtry=2, splitrule=gini, min.node.size=5
## + Fold09: mtry=3, splitrule=gini, min.node.size=5
## - Fold09: mtry=3, splitrule=gini, min.node.size=5
## + Fold10: mtry=2, splitrule=gini, min.node.size=5
## - Fold10: mtry=2, splitrule=gini, min.node.size=5
## + Fold10: mtry=3, splitrule=gini, min.node.size=5
## - Fold10: mtry=3, splitrule=gini, min.node.size=5
## Aggregating results
## Selecting tuning parameters
## Fitting mtry = 2, splitrule = gini, min.node.size = 5 on full training set
```

```
x_test = test %>% dplyr::select(-stroke)

rf_pred = predict(rf_model, newdata = x_test)

confusionMatrix(rf_pred, factor(test[["stroke"]]), positive = "1")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 935  45
```

```
##          1   2   0
##
##                Accuracy : 0.9521
##                  95% CI : (0.9369, 0.9646)
##     No Information Rate : 0.9542
##     P-Value [Acc > NIR] : 0.6559
##
##                   Kappa : -0.0039
##
##  Mcnemar's Test P-Value : 8.993e-10
##
##             Sensitivity : 0.000000
##             Specificity : 0.997866
##          Pos Pred Value : 0.000000
##          Neg Pred Value : 0.954082
##              Prevalence : 0.045825
##          Detection Rate : 0.000000
##    Detection Prevalence : 0.002037
##       Balanced Accuracy : 0.498933
##
##        'Positive' Class : 1
##
```

```
ggplot(rf_model, highlight = TRUE)
```



```
rf = randomForest(stroke ~., data = train)
pred_rf = predict(rf, newdata = test)
```