Final

Yuechen Liu, Mufeng Xu, Yanhao Li

Contents

```r
library(tidyverse)
library(caret)
library(glmnet)
library(ISLR)
library(pls)
library(AppliedPredictiveModeling)
library(MASS)
library(e1071)
library(mlbench)
library(pROC)
library(arsenal)
library(visdat)
library(pdp)
library(vip)
library(randomForest)
library(ranger)
library(gbm)
library(e1071)
library(kernlab)
```

Introduction

Stroke is a serious life-threatening medical condition. According to the World Health Organization, stroke is the second leading cause of death globally. To better understand which factors correlate to the stroke event, our group find a stroke prediction dataset. This dataset contains twelve columns. The first column labels the unique identifier of the patient. The last column records the occurrence of stroke by 1 (Yes) or 0 (No). The other ten columns contain the observations of possible predictors.

Load, clean, and tidy data

```r
stroke = read_csv("./healthcare-dataset-stroke-data.csv") %>%
  mutate(
    bmi = as.numeric(bmi)
  )

stroke1 = stroke %>%
  janitor::clean_names() %>%
  na.omit() %>%
  filter(
    bmi != "N/A",
    gender != "Other"
  ) %>%
  mutate(
    gender = recode(
      gender,
      "Male" = 0,
      "Female" = 1
    ),
    ever_married = recode(
```

```
      ever_married,
      "No" = 0,
      "Yes" = 1
    ),
    work_type = recode(
      work_type,
      "children" = 0,
      "Govt_job" = 1,
      "Never_worked" = 2,
      "Private" = 3,
      "Self-employed" = 4
    ),
    residence_type = recode(
      residence_type,
      "Rural" = 0,
      "Urban" = 1
    ),
    smoking_status = recode(
      smoking_status,
      "formerly smoked" = 0,
      "never smoked" = 1,
      "smokes" = 2,
      "Unknown" = 3
    ),
    stroke = recode(
      stroke,
     "0" = "No",
     "1" = "Yes"
    ),
    stroke = as.factor(stroke)
  ) %>%
  relocate(
    age, avg_glucose_level, bmi
  )

stroke2 = stroke1 %>%
  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)
    )
```

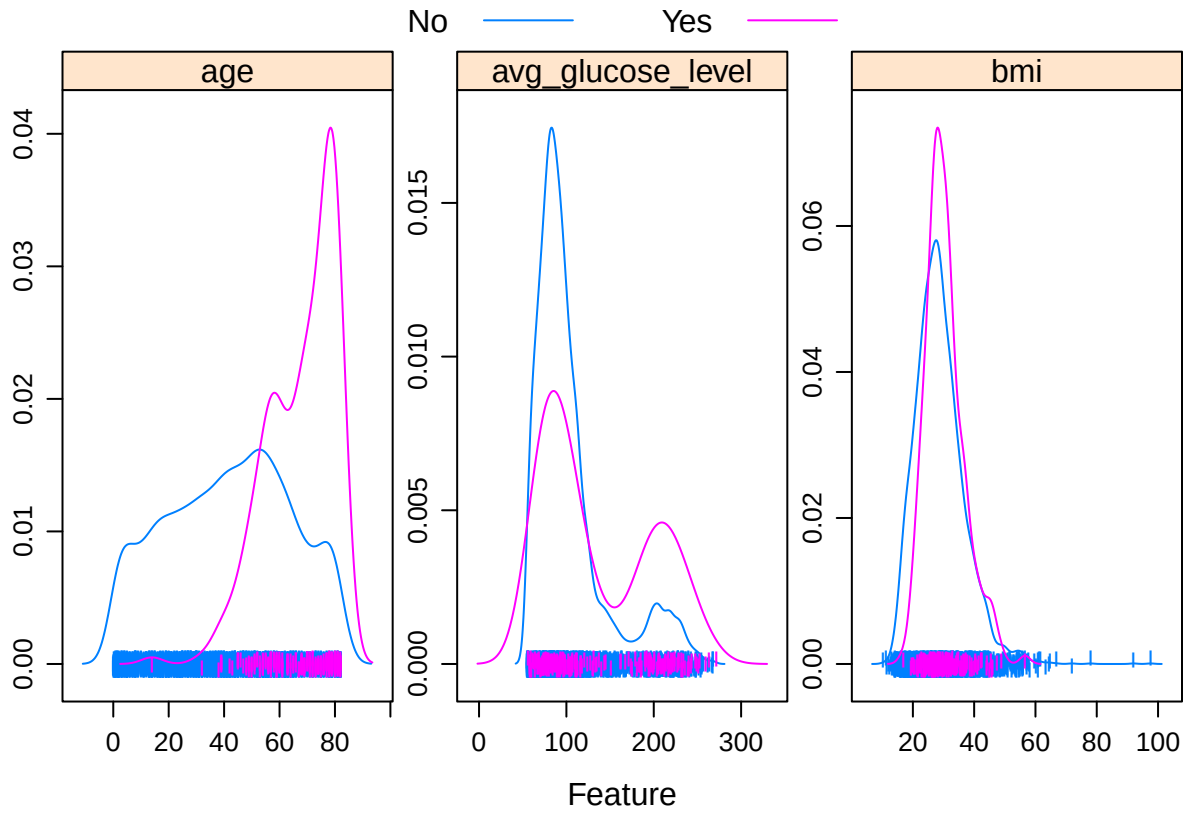
Exploratory analysis/ visualization

```
stats = tableby(stroke ~ gender + age + hypertension + heart_disease + ever_married + work_type + Reside

summary(stats, text = TRUE) %>% knitr::kable()
```

	0 (N=4861)	1 (N=249)	Total (N=5110)	p value
gender				0.790
- Female	2853 (58.7%)	141 (56.6%)	2994 (58.6%)	
- Male	2007 (41.3%)	108 (43.4%)	2115 (41.4%)	
- Other	1 (0.0%)	0 (0.0%)	1 (0.0%)	
age				< 0.001
- Mean (SD)	41.972 (22.292)	67.728 (12.727)	43.227 (22.613)	
- Range	0.080 - 82.000	1.320 - 82.000	0.080 - 82.000	
hypertension				< 0.001
- Mean (SD)	0.089 (0.285)	0.265 (0.442)	0.097 (0.297)	
- Range	0.000 - 1.000	0.000 - 1.000	0.000 - 1.000	
heart_disease				< 0.001
- Mean (SD)	0.047 (0.212)	0.189 (0.392)	0.054 (0.226)	
- Range	0.000 - 1.000	0.000 - 1.000	0.000 - 1.000	
ever_married				< 0.001
- No	1728 (35.5%)	29 (11.6%)	1757 (34.4%)	
- Yes	3133 (64.5%)	220 (88.4%)	3353 (65.6%)	
work_type				< 0.001
- children	685 (14.1%)	2 (0.8%)	687 (13.4%)	
- Govt_job	624 (12.8%)	33 (13.3%)	657 (12.9%)	
- Never_worked	22 (0.5%)	0 (0.0%)	22 (0.4%)	
- Private	2776 (57.1%)	149 (59.8%)	2925 (57.2%)	
- Self-employed	754 (15.5%)	65 (26.1%)	819 (16.0%)	
Residence_type				0.269
- Rural	2400 (49.4%)	114 (45.8%)	2514 (49.2%)	
- Urban	2461 (50.6%)	135 (54.2%)	2596 (50.8%)	
avg_glucose_level				< 0.001
- Mean (SD)	104.796 (43.846)	132.545 (61.921)	106.148 (45.284)	
- Range	55.120 - 267.760	56.110 - 271.740	55.120 - 271.740	
bmi				0.003
- N-Miss	161	40	201	
- Mean (SD)	28.823 (7.908)	30.471 (6.329)	28.893 (7.854)	
- Range	10.300 - 97.600	16.900 - 56.600	10.300 - 97.600	
smoking_status				< 0.001
- formerly smoked	815 (16.8%)	70 (28.1%)	885 (17.3%)	
- never smoked	1802 (37.1%)	90 (36.1%)	1892 (37.0%)	
- smokes	747 (15.4%)	42 (16.9%)	789 (15.4%)	
- Unknown	1497 (30.8%)	47 (18.9%)	1544 (30.2%)	

```
featurePlot(x = stroke1[, 1:3],
            y = stroke1$stroke,
            scales = list(x = list(relation = "free"),
                          y = list(relation = "free")),
            plot = "density",
            pch = "|",
            auto.key = list(columns = 2),
            font = 2)
```



Models

```
set.seed(1)

indextrain <- createDataPartition(y = stroke2$stroke,
                                  p = 0.8,
                                  list = FALSE)

x <- stroke2[indextrain, -c(4, 12)]

y <- stroke2$stroke[indextrain] ###train

x2 <- stroke2[-indextrain, -c(4, 12)]

y2 <- stroke2$stroke[-indextrain] ###test

ctrl <- trainControl(method = "cv",
                     summaryFunction = twoClassSummary,
                     classProbs = TRUE)
```

GLM

```
model.glm = train(x = x,
                  y = y,
                  method = 'glm',
                  metric = "ROC",
                  trControl = ctrl)

glm.pred.prob = predict(model.glm, newdata = x2, type = "prob")[,1]

glm.pred = rep("Yes", length(glm.pred.prob))

glm.pred[glm.pred.prob < 0.95] = "No"

confusionMatrix(data = as.factor(glm.pred),
                reference = y2,
                positive = "Yes")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  219  29
##        Yes 720  12
##
##                Accuracy : 0.2357
##                  95% CI : (0.2095, 0.2636)
##     No Information Rate : 0.9582
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : -0.0523
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.29268
##             Specificity : 0.23323
##          Pos Pred Value : 0.01639
##          Neg Pred Value : 0.88306
##              Prevalence : 0.04184
##          Detection Rate : 0.01224
##    Detection Prevalence : 0.74694
##       Balanced Accuracy : 0.26295
##
##        'Positive' Class : Yes
##
```

```
model.glm$bestTune
```

```
##   parameter
## 1      none
```

MARS

```
set.seed(1)

model.mars <- train(x = x,
                    y = y,
                    method = "earth",
                    tuneGrid = expand.grid(degree = 1:3,
                                           nprune = 2:15),
                    metric = "ROC",
                    trControl = ctrl)

mars.pred.prob = predict(model.mars, newdata = x2, type = "prob")[,1]

mars.pred = rep("Yes", length(mars.pred.prob))

mars.pred[mars.pred.prob < 0.95] = "No"

confusionMatrix(data = as.factor(mars.pred),
                reference = y2,
                positive = "Yes")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  199  29
##        Yes 740  12
##
##                Accuracy : 0.2153
##                  95% CI : (0.1899, 0.2424)
##     No Information Rate : 0.9582
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : -0.0533
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.29268
##             Specificity : 0.21193
##          Pos Pred Value : 0.01596
##          Neg Pred Value : 0.87281
##              Prevalence : 0.04184
##          Detection Rate : 0.01224
##    Detection Prevalence : 0.76735
##       Balanced Accuracy : 0.25231
##
##        'Positive' Class : Yes
##
```

```
model.mars$bestTune
```

```
##   nprune degree
```

```
## 7      8      1
```

GAM

```
set.seed(1)

model.gam <- train(x = x,
                   y = y,
                   method = "gam",
                   metric = "ROC",
                   trControl = ctrl)

gam.pred.prob = predict(model.gam, newdata = x2, type = "prob")[,1]

gam.pred = rep("Yes", length(gam.pred.prob))

gam.pred[gam.pred.prob < 0.95] = "No"

confusionMatrix(data = as.factor(gam.pred),
                reference = y2,
                positive = "Yes")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  240  31
##        Yes 699  10
##
##                Accuracy : 0.2551
##                  95% CI : (0.2281, 0.2836)
##     No Information Rate : 0.9582
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : -0.0569
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.24390
##             Specificity : 0.25559
##          Pos Pred Value : 0.01410
##          Neg Pred Value : 0.88561
##              Prevalence : 0.04184
##          Detection Rate : 0.01020
##    Detection Prevalence : 0.72347
##       Balanced Accuracy : 0.24975
##
##        'Positive' Class : Yes
##
```

```
model.gam$bestTune
```

```
##   select method
## 1  FALSE GCV.Cp
```

LDA (from the midterm, LDA is the best among LDA, QDA and KNN)

```
set.seed(1)

model.lda = train(x = data.matrix(x),
                  y = y,
                  method = "lda",
                  metric = "ROC",
                  trControl = ctrl)

lda.pred.prob = predict(model.lda, newdata = data.matrix(x2), type = "prob")[,1]

lda.pred = rep("Yes", length(lda.pred.prob))

lda.pred[lda.pred.prob < 0.95] = "No"

confusionMatrix(data = as.factor(lda.pred),
                reference = y2,
                positive = "Yes")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  200  28
##        Yes 739  13
##
##                Accuracy : 0.2173
##                  95% CI : (0.1919, 0.2445)
##     No Information Rate : 0.9582
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : -0.0506
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.31707
##             Specificity : 0.21299
##          Pos Pred Value : 0.01729
##          Neg Pred Value : 0.87719
##              Prevalence : 0.04184
##          Detection Rate : 0.01327
##    Detection Prevalence : 0.76735
##       Balanced Accuracy : 0.26503
##
##        'Positive' Class : Yes
```

```
##
```

```
model.lda$bestTune
```

```
##   parameter
## 1      none
```

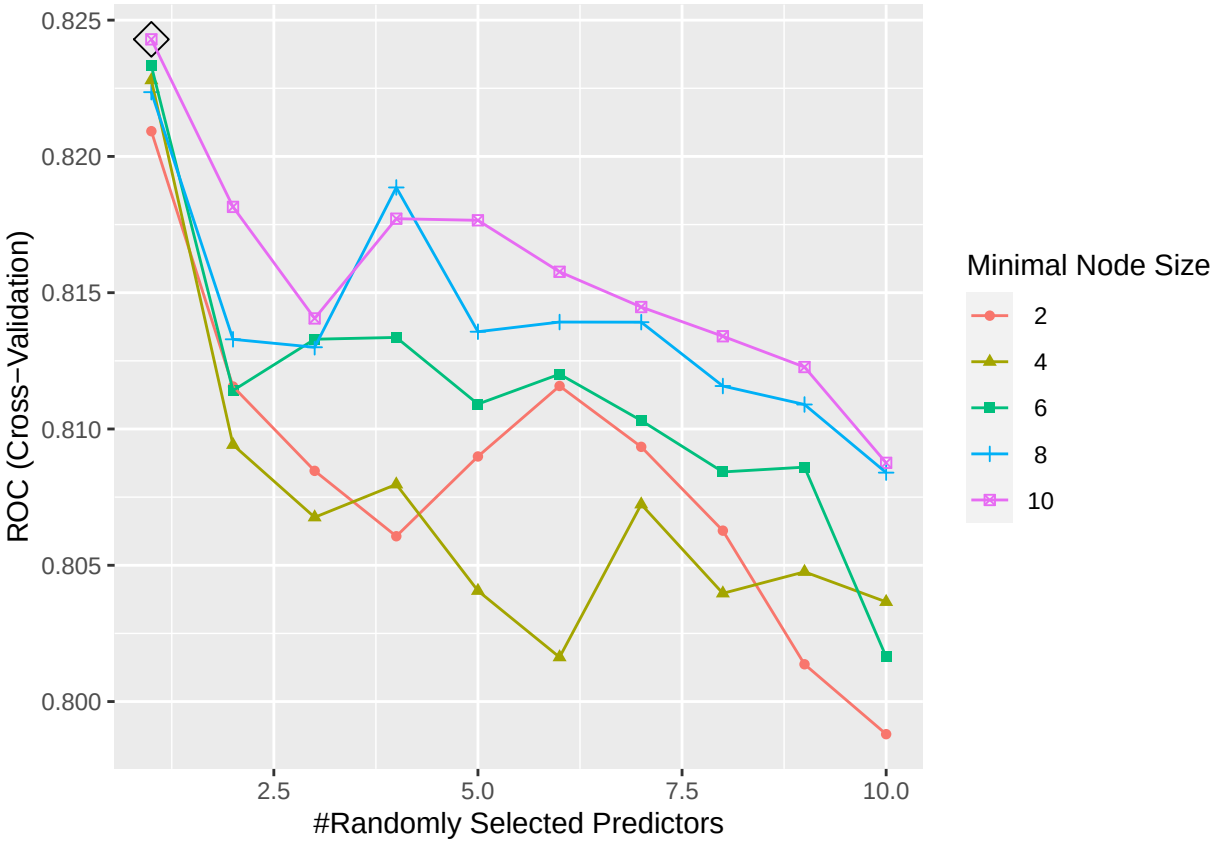
Random Forest

```
rf.grid <- expand.grid(mtry = 1:10,
                       splitrule = "gini",
                       min.node.size = seq(from = 2, to = 10, by = 2))

set.seed(1)

model.rf <- train(x = x,
                  y = y,
                 method = "ranger",
                 tuneGrid = rf.grid,
                 metric = "ROC",
                 trControl = ctrl)

ggplot(model.rf, highlight = TRUE)
```



```
rf.pred.prob = predict(model.rf, newdata = x2, type = "prob")[,1]

rf.pred = rep("Yes", length(rf.pred.prob))

rf.pred[rf.pred.prob < 0.95] = "No"

confusionMatrix(data = as.factor(rf.pred),
                reference = y2,
                positive = "Yes")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  222  31
##        Yes 717  10
##
##                Accuracy : 0.2367
##                  95% CI : (0.2104, 0.2646)
##     No Information Rate : 0.9582
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : -0.0577
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.24390
##             Specificity : 0.23642
##          Pos Pred Value : 0.01376
##          Neg Pred Value : 0.87747
##              Prevalence : 0.04184
##          Detection Rate : 0.01020
##    Detection Prevalence : 0.74184
##       Balanced Accuracy : 0.24016
##
##        'Positive' Class : Yes
##
```

```
model.rf$bestTune
```

```
##   mtry splitrule min.node.size
## 5    1      gini            10
```

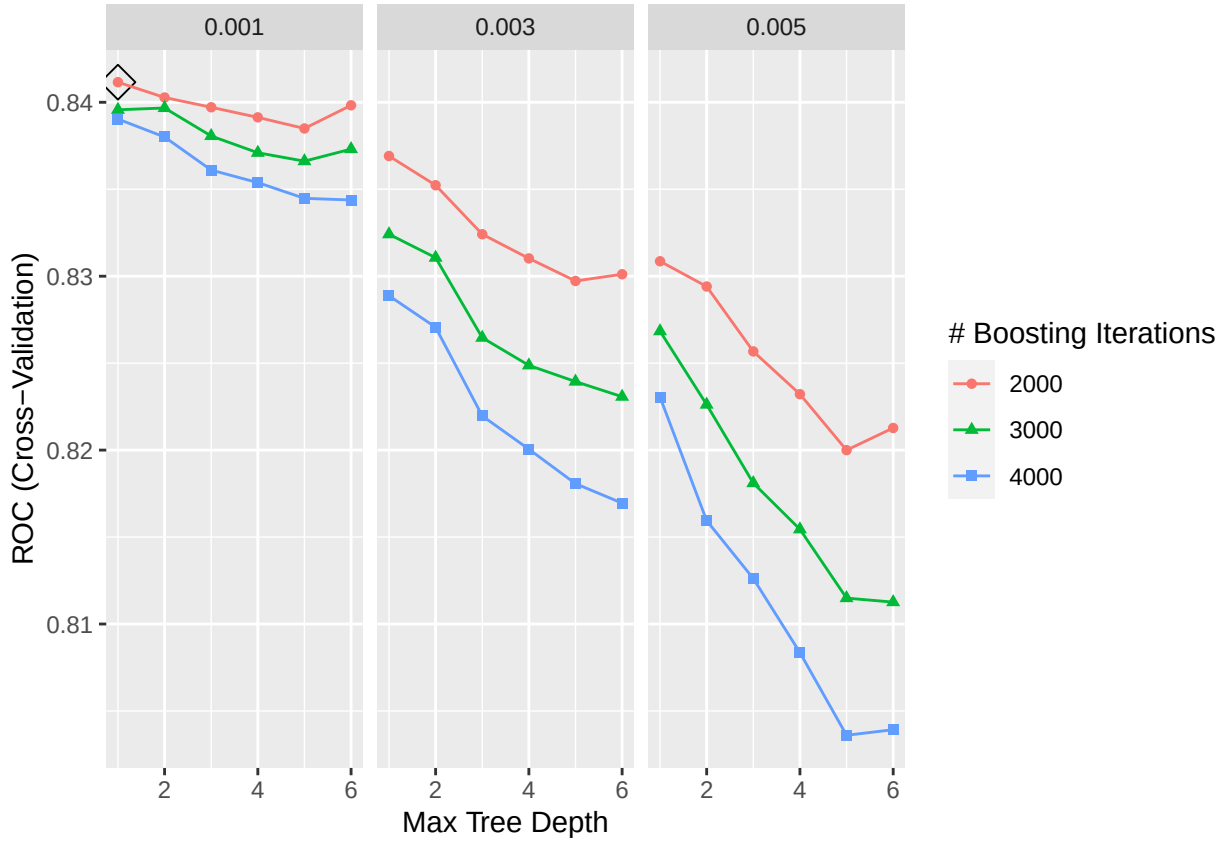
gbmA

```
gbmA.grid <- expand.grid(n.trees = c(2000,3000,4000),
                         interaction.depth = 1:6,
                         shrinkage = c(0.001,0.003,0.005),
                         n.minobsinnode = 1)
```

```
set.seed(1)

model.gbma <- train(x = x,
                    y = y,
                    tuneGrid = gbmA.grid,
                    trControl = ctrl,
                    method = "gbm",
                    distribution = "adaboost",
                    metric = "ROC",
                    verbose = FALSE)

ggplot(model.gbma, highlight = TRUE)
```




```
test.pred.prob = predict(model.gbma, newdata = x2, type = "prob")[,1]

test.pred = rep("Yes", length(test.pred.prob))

test.pred[test.pred.prob < 0.95] = "No"

confusionMatrix(data = as.factor(test.pred),
                reference = y2,
                positive = "Yes")
```

```
## Confusion Matrix and Statistics
##
```

```
##           Reference
## Prediction  No Yes
##        No  271  31
##        Yes 668  10
##
##                Accuracy : 0.2867
##                  95% CI : (0.2586, 0.3162)
##     No Information Rate : 0.9582
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : -0.0555
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.24390
##             Specificity : 0.28860
##          Pos Pred Value : 0.01475
##          Neg Pred Value : 0.89735
##              Prevalence : 0.04184
##          Detection Rate : 0.01020
##    Detection Prevalence : 0.69184
##       Balanced Accuracy : 0.26625
##
##        'Positive' Class : Yes
##
```

```
model.gbma$bestTune
```

```
##   n.trees interaction.depth shrinkage n.minobsinnode
## 1    2000                 1     0.001              1
```

svml

svmr