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") %>%
  dplyr::select(-id, -Residence_type, -ever_married, -smoking_status, -work_type) %>%
  mutate(
    bmi = as.numeric(bmi),
    gender = as.factor(gender),
    hypertension = as.factor(hypertension),
    heart_disease = as.factor(heart_disease),
    stroke = as.factor(stroke)
  )


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

```
    gender = recode(
      gender,
      "Male" = 0,
      "Female" = 1
    ),
    stroke = recode(
      stroke,
     "0" = "No",
     "1" = "Yes"
    )
  ) %>%
  relocate(
    age, avg_glucose_level, bmi
  )
```

Exploratory analysis/ visualization

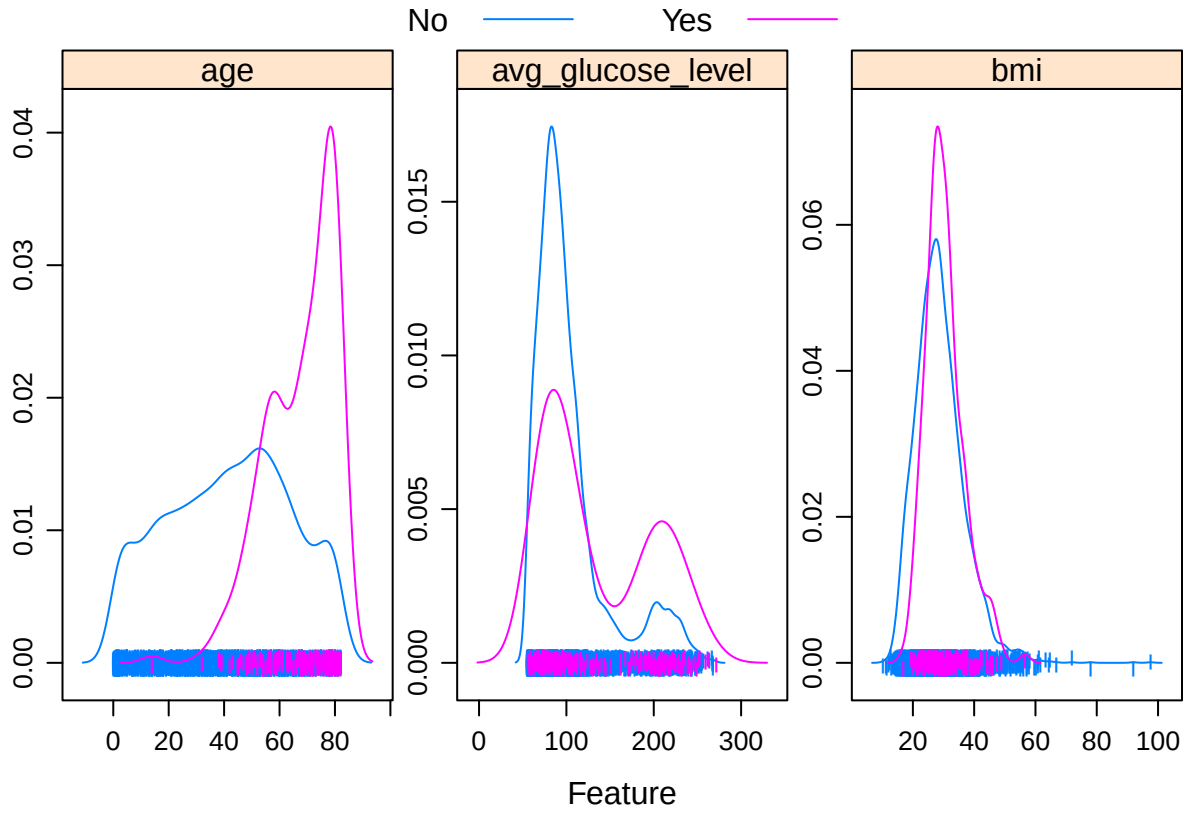
```
stats = tableby(stroke ~ gender + age + hypertension + heart_disease + avg_glucose_level + bmi, data = s

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
- 0	4429 (91.1%)	183 (73.5%)	4612 (90.3%)	
- 1	432 (8.9%)	66 (26.5%)	498 (9.7%)	
heart_disease				< 0.001
- 0	4632 (95.3%)	202 (81.1%)	4834 (94.6%)	
- 1	229 (4.7%)	47 (18.9%)	276 (5.4%)	
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	

```
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 = stroke1$stroke,
                                  p = 0.8,
                                  list = FALSE)

train = stroke1[indextrain, ]

test = stroke1[-indextrain, ]

x <- stroke1[indextrain, -c(7)]

y <- stroke1$stroke[indextrain] ###train

x2 <- stroke1[-indextrain, -c(7)]
```

```
y2 <- stroke1$stroke[-indextrain] ###test

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

GLM

```
model.glm = train(stroke ~ . ,
                  data = train,
                  method = 'glm',
                  metric = "ROC",
                  trControl = ctrl)

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

glm.pred = rep("No", length(glm.pred.prob))

glm.pred[glm.pred.prob < 0.9] = "Yes"

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

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  820  18
##        Yes 119  23
##
##                Accuracy : 0.8602
##                  95% CI : (0.8369, 0.8813)
##     No Information Rate : 0.9582
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1994
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.56098
##             Specificity : 0.87327
##          Pos Pred Value : 0.16197
##          Neg Pred Value : 0.97852
##              Prevalence : 0.04184
##          Detection Rate : 0.02347
##    Detection Prevalence : 0.14490
##       Balanced Accuracy : 0.71712
##
##        'Positive' Class : Yes
##
```

```
model.glm$bestTune
```

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

GLMN

```
glmn.grid <- expand.grid(.alpha = seq(0, 1, length = 6),
                         .lambda = exp(seq(-8, -2, length = 20)))

set.seed(1)

model.glmn <- train(x = data.matrix(x),
                    y = y,
                    method = "glmnet",
                    tuneGrid = glmn.grid,
                    metric = "ROC",
                    trControl = ctrl)

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

glmn.pred = rep("No", length(glmn.pred.prob))

glmn.pred[glmn.pred.prob < 0.9] = "Yes"

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

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  827  18
##        Yes 112  23
##
##                Accuracy : 0.8673
##                  95% CI : (0.8445, 0.888)
##     No Information Rate : 0.9582
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.2107
##
##  Mcnemar's Test P-Value : 3.445e-16
##
##             Sensitivity : 0.56098
##             Specificity : 0.88072
##          Pos Pred Value : 0.17037
##          Neg Pred Value : 0.97870
##              Prevalence : 0.04184
##          Detection Rate : 0.02347
```

```
##    Detection Prevalence : 0.13776
##       Balanced Accuracy : 0.72085
##
##        'Positive' Class : Yes
##
```

```
model.glmn$bestTune
```

```
##    alpha      lambda
## 68   0.6 0.003059592
```

MARS

```
set.seed(1)

model.mars <- train(stroke ~ . ,
                    data = train,
                    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("No", length(mars.pred.prob))

mars.pred[mars.pred.prob < 0.9] = "Yes"

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

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  820  18
##        Yes 119  23
##
##                Accuracy : 0.8602
##                  95% CI : (0.8369, 0.8813)
##     No Information Rate : 0.9582
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1994
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.56098
##             Specificity : 0.87327
```

```
##          Pos Pred Value : 0.16197
##          Neg Pred Value : 0.97852
##              Prevalence : 0.04184
##          Detection Rate : 0.02347
##    Detection Prevalence : 0.14490
##       Balanced Accuracy : 0.71712
##
##        'Positive' Class : Yes
##
```

```
model.mars$bestTune
```

```
##   nprune degree
## 7      8      1
```

GAM

```
set.seed(1)

model.gam <- train(stroke ~ . ,
                   data = train,
                   method = "gam",
                   metric = "ROC",
                   trControl = ctrl)

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

gam.pred = rep("No", length(gam.pred.prob))

gam.pred[gam.pred.prob < 0.9] = "Yes"

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

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  819  18
##        Yes 120  23
##
##                Accuracy : 0.8592
##                  95% CI : (0.8358, 0.8804)
##     No Information Rate : 0.9582
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1978
##
##  Mcnemar's Test P-Value : <2e-16
##
```

```
##             Sensitivity : 0.56098
##             Specificity : 0.87220
##          Pos Pred Value : 0.16084
##          Neg Pred Value : 0.97849
##              Prevalence : 0.04184
##          Detection Rate : 0.02347
##    Detection Prevalence : 0.14592
##       Balanced Accuracy : 0.71659
##
##        '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("No", length(lda.pred.prob))

lda.pred[lda.pred.prob < 0.9] = "Yes"

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

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  847  22
##        Yes  92  19
##
##                Accuracy : 0.8837
##                  95% CI : (0.8619, 0.9031)
##     No Information Rate : 0.9582
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.2012
##
```

```
##  Mcnemar's Test P-Value : 1.03e-10
##
##             Sensitivity : 0.46341
##             Specificity : 0.90202
##          Pos Pred Value : 0.17117
##          Neg Pred Value : 0.97468
##              Prevalence : 0.04184
##          Detection Rate : 0.01939
##    Detection Prevalence : 0.11327
##       Balanced Accuracy : 0.68272
##
##        'Positive' Class : Yes
##
```

```
model.lda$bestTune
```

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

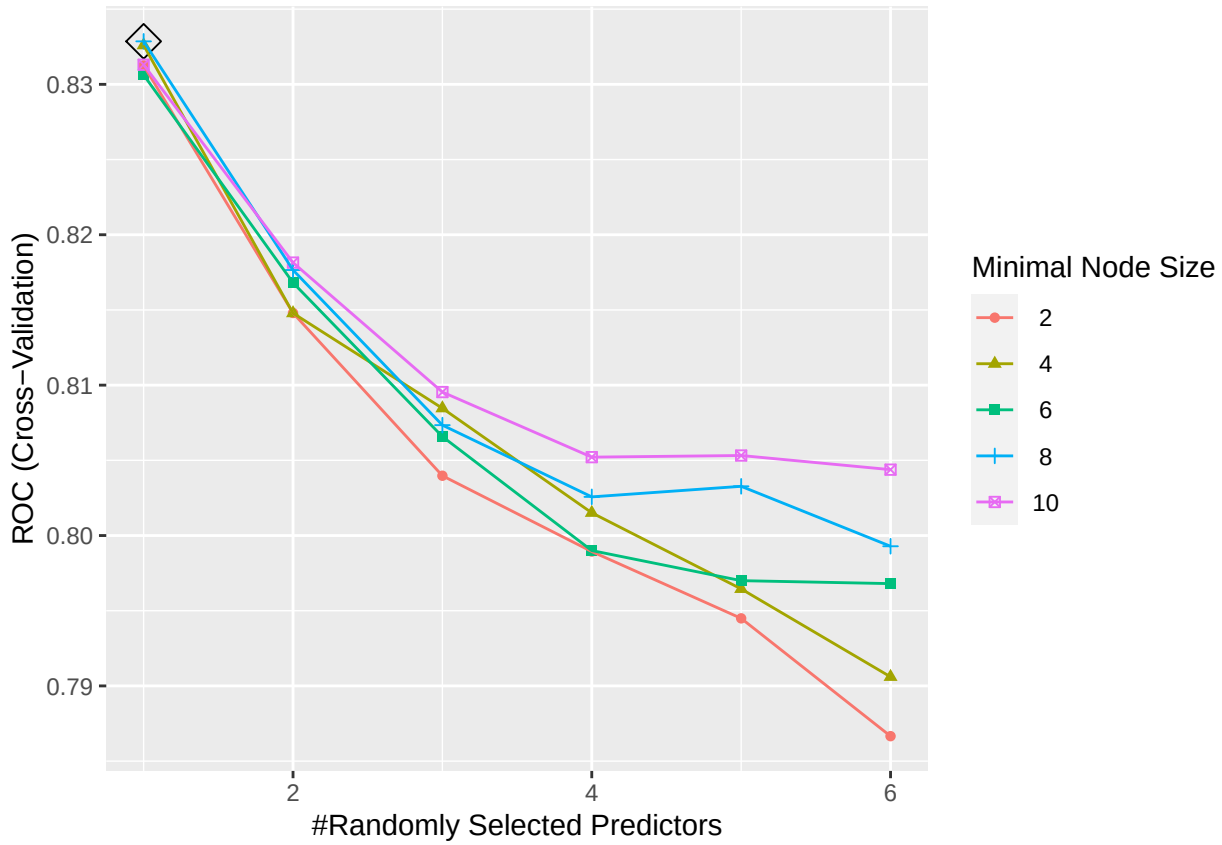
Random Forest

```
rf.grid <- expand.grid(mtry = 1:6,
                       splitrule = "gini",
                       min.node.size = seq(from = 2, to = 10, by = 2))

set.seed(1)

model.rf <- train(stroke ~ . ,
                  data = train,
                method = "ranger",
                tuneGrid = rf.grid,
                metric = "ROC",
                trControl = ctrl,
                importance = "permutation")

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




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

rf.pred = rep("No", length(rf.pred.prob))

rf.pred[rf.pred.prob < 0.9] = "Yes"

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

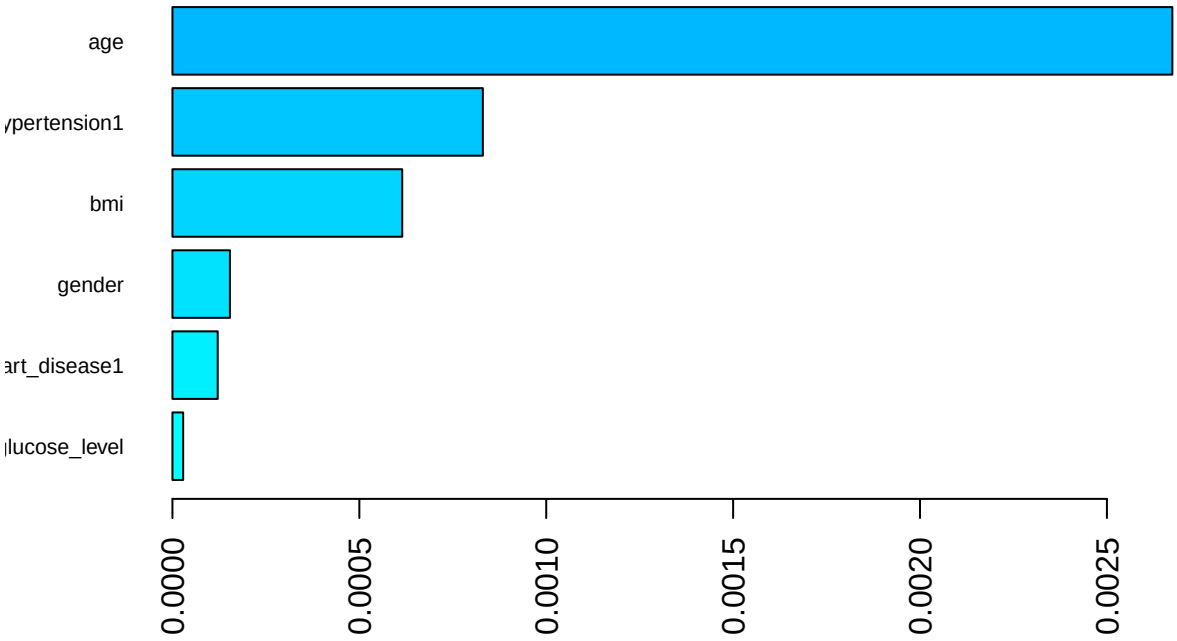
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  850  21
##        Yes  89  20
##
##                Accuracy : 0.8878
##                  95% CI : (0.8663, 0.9068)
##     No Information Rate : 0.9582
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.2192
##
##  Mcnemar's Test P-Value : 1.679e-10
```

```
##
##             Sensitivity : 0.48780
##             Specificity : 0.90522
##          Pos Pred Value : 0.18349
##          Neg Pred Value : 0.97589
##              Prevalence : 0.04184
##          Detection Rate : 0.02041
##    Detection Prevalence : 0.11122
##       Balanced Accuracy : 0.69651
##
##        'Positive' Class : Yes
##
```

```
model.rf$bestTune
```

```
##   mtry splitrule min.node.size
## 4    1      gini             8
```

```
barplot(sort(ranger::importance(model.rf$finalModel), decreasing = FALSE),
        las = 2, horiz = TRUE, cex.names = 0.7,
        col = colorRampPalette(colors = c("cyan","blue"))(19))
```



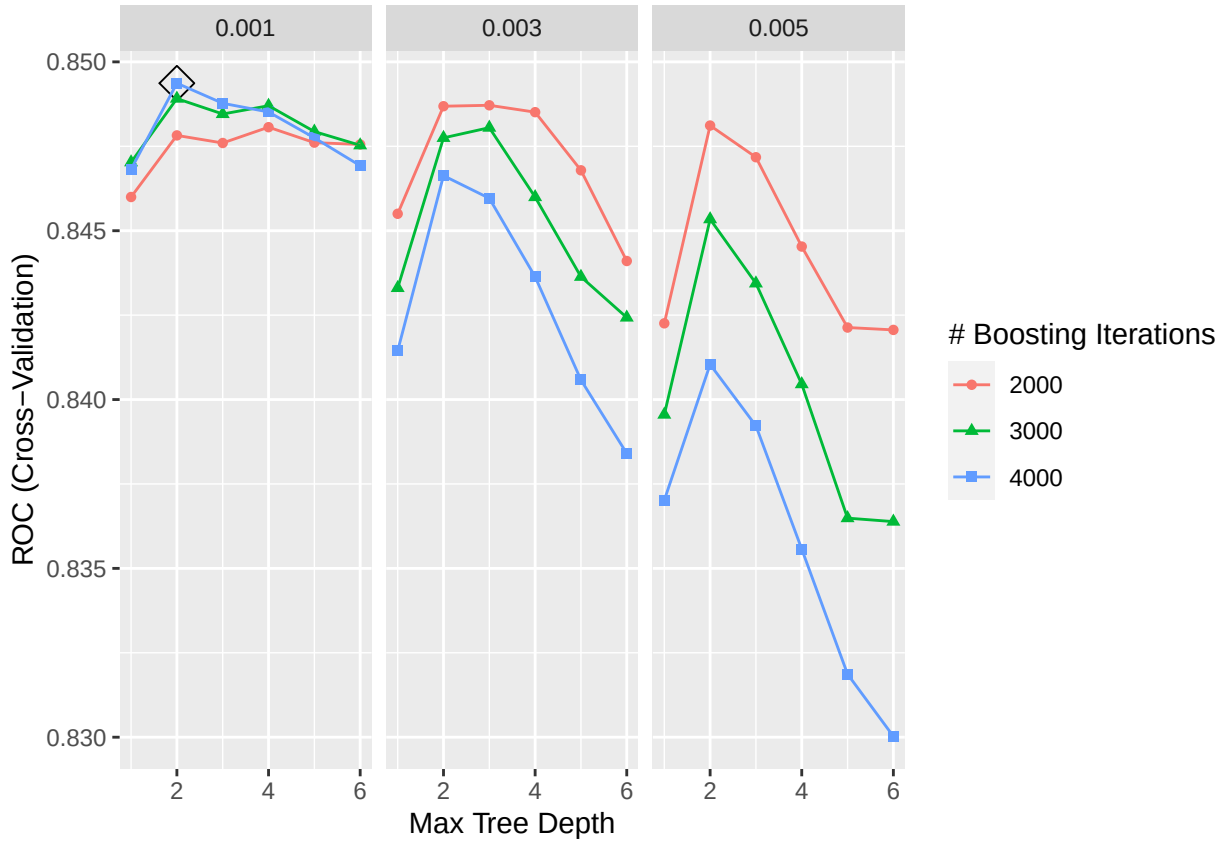
gbmA

```
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(stroke ~ . ,
                    data = train,
                    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("No", length(test.pred.prob))
```

```
test.pred[test.pred.prob < 0.9] = "Yes"

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

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  836  19
##        Yes 103  22
##
##                Accuracy : 0.8755
##                  95% CI : (0.8532, 0.8955)
##     No Information Rate : 0.9582
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.2156
##
##  Mcnemar's Test P-Value : 5.714e-14
##
##             Sensitivity : 0.53659
##             Specificity : 0.89031
##          Pos Pred Value : 0.17600
##          Neg Pred Value : 0.97778
##              Prevalence : 0.04184
##          Detection Rate : 0.02245
##    Detection Prevalence : 0.12755
##       Balanced Accuracy : 0.71345
##
##        'Positive' Class : Yes
##
```

```
model.gbma$bestTune
```

```
##   n.trees interaction.depth shrinkage n.minobsinnode
## 6    4000                 2     0.001              1
```

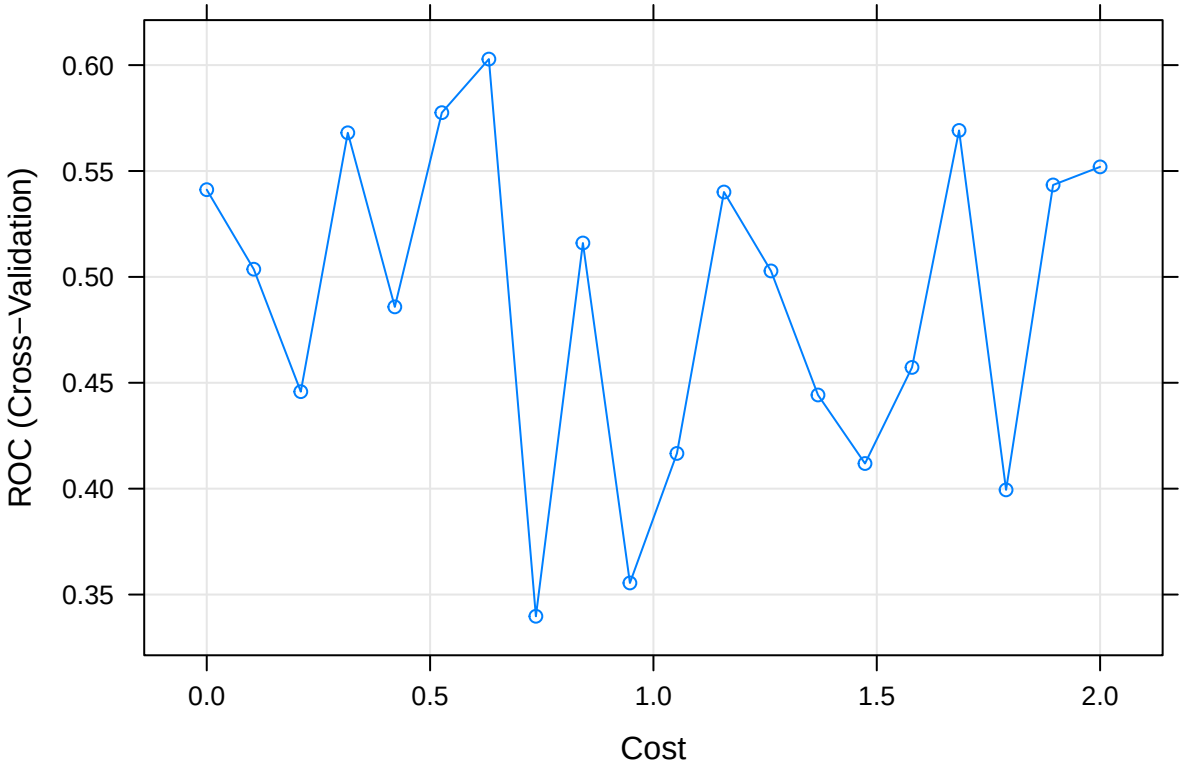
svml

```
set.seed(1)

model.svml <- train(stroke ~ . ,
                    data = train,
                    method = "svmLinear",
                    preProcess = c("center", "scale"),
                    tuneGrid = data.frame(C = exp(seq(0, 2,len = 20))),
                    metric = "ROC",
                    trControl = ctrl)
```

```
## maximum number of iterations reached 0.0006552409 0.0006552036maximum number of iterations reached 0
```

```
plot(model.svml, highlight = TRUE, xTrans = log)
```




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

svml.pred = rep("No", length(svml.pred.prob))

svml.pred[svml.pred.prob < 0.9] = "Yes"

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

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  939  41
##        Yes   0   0
##
##                Accuracy : 0.9582
##                  95% CI : (0.9437, 0.9698)
##     No Information Rate : 0.9582
##     P-Value [Acc > NIR] : 0.5414
```

```
##
##                   Kappa : 0
##
##  Mcnemar's Test P-Value : 4.185e-10
##
##             Sensitivity : 0.00000
##             Specificity : 1.00000
##          Pos Pred Value :     NaN
##          Neg Pred Value : 0.95816
##              Prevalence : 0.04184
##          Detection Rate : 0.00000
##    Detection Prevalence : 0.00000
##       Balanced Accuracy : 0.50000
##
##        'Positive' Class : Yes
##
```

```
model.svml$bestTune
```

```
##          C
## 7 1.880578
```

svmr

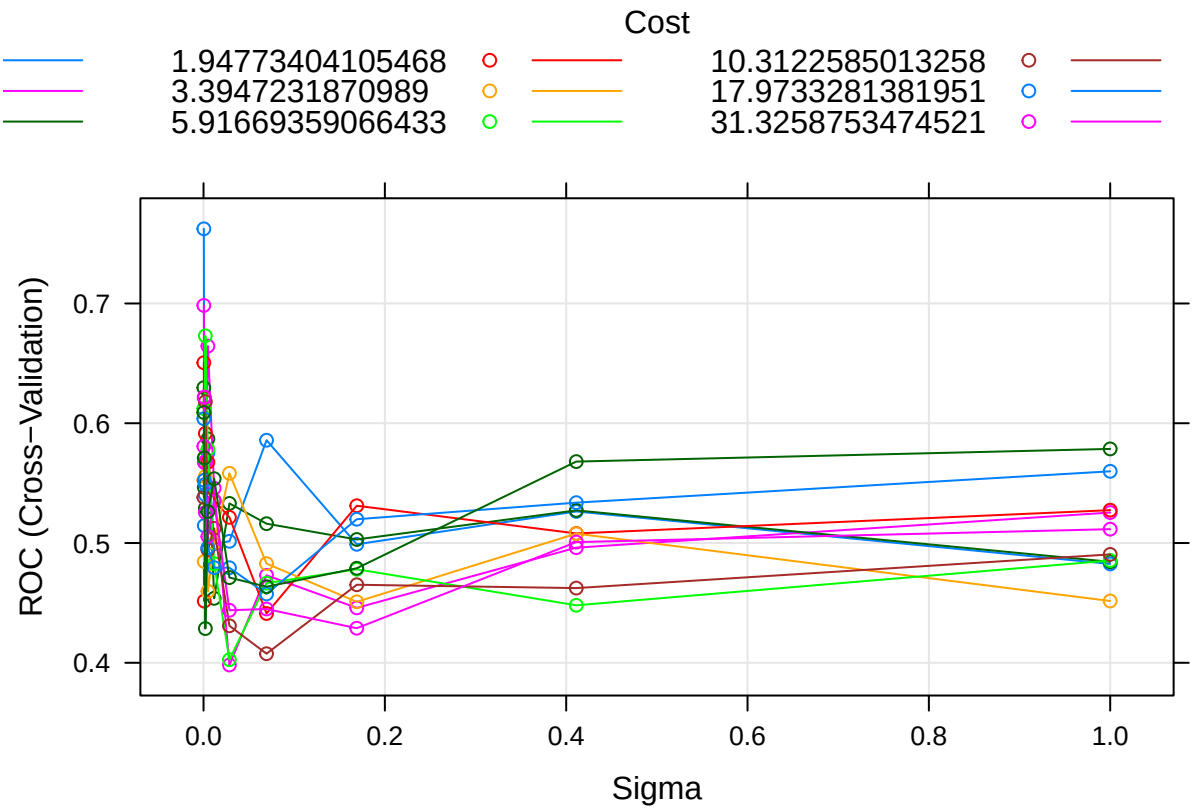
```
svmr.grid <- expand.grid(C = exp(seq(-1,4,len = 10)),
                         sigma = exp(seq(-8,0,len = 10)))

set.seed(1)

model.svmr <- train(stroke ~ . ,
                    data = train,
                    method = "svmRadialSigma",
                    preProcess = c("center", "scale"),
                    tuneGrid = svmr.grid,
                    trControl = ctrl)
```

```
## maximum number of iterations reached 0.004702465 0.004682068maximum number of iterations reached 0.0
```

```
plot(model.svmr, highlight = TRUE)
```




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

svmr.pred = rep("No", length(svmr.pred.prob))

svmr.pred[svmr.pred.prob < 0.9] = "Yes"

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

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  939  41
##        Yes   0   0
##
##                Accuracy : 0.9582
##                  95% CI : (0.9437, 0.9698)
##     No Information Rate : 0.9582
##     P-Value [Acc > NIR] : 0.5414
##
##                   Kappa : 0
##
##  Mcnemar's Test P-Value : 4.185e-10
```

```
##
##             Sensitivity : 0.00000
##             Specificity : 1.00000
##          Pos Pred Value :     NaN
##          Neg Pred Value : 0.95816
##              Prevalence : 0.04184
##          Detection Rate : 0.00000
##    Detection Prevalence : 0.00000
##       Balanced Accuracy : 0.50000
##
##        'Positive' Class : Yes
##
```

```
model.svmr$bestTune
```

```
##          sigma         C
## 1 0.0003354626 0.3678794
```

Comparison

```
res <- resamples(list(glm = model.glm,
                      glmn = model.glmn,
                      mars = model.mars,
                      gam = model.gam,
                      lda = model.lda,
                      rf = model.rf,
                      gbmA = model.gbma,
                      svml = model.svml,
                      svmr = model.svmr))

summary(res)
```

```
##
## Call:
## summary.resamples(object = res)
##
## Models: glm, glmn, mars, gam, lda, rf, gbmA, svml, svmr
## Number of resamples: 10
##
## ROC
##           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## glm  0.7850731 0.8110138 0.8408166 0.8480004 0.8759778 0.9245932    0
## glmn 0.7517209 0.8343608 0.8595070 0.8510521 0.8747653 0.8976846    0
## mars 0.7291927 0.8045421 0.8578448 0.8378447 0.8744329 0.9000313    0
## gam  0.7526596 0.8331948 0.8583532 0.8490858 0.8757431 0.8969024    0
## lda  0.7257509 0.8223952 0.8388220 0.8362690 0.8613892 0.9094180    0
## rf   0.7484355 0.8074546 0.8351651 0.8328594 0.8564612 0.9061327    0
## gbmA 0.7666615 0.8350648 0.8573217 0.8493674 0.8716755 0.8890801    0
## svml 0.3864205 0.5213939 0.5744241 0.6028463 0.6939338 0.8635795    0
## svmr 0.4946809 0.6954005 0.7973981 0.7623484 0.8372966 0.8878285    0
```

```
##
## Sens
##           Min.   1st Qu.    Median      Mean   3rd Qu. Max. NA's
## glm  1.0000000 1.0000000 1.0000000 1.0000000 1.0000000    1    0
## glmn 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000    1    0
## mars 0.9973404 1.0000000 1.0000000 0.9997340 1.0000000    1    0
## gam  1.0000000 1.0000000 1.0000000 1.0000000 1.0000000    1    0
## lda  0.9787234 0.9893617 0.9933511 0.9920213 0.9946809    1    0
## rf   1.0000000 1.0000000 1.0000000 1.0000000 1.0000000    1    0
## gbmA 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000    1    0
## svml 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000    1    0
## svmr 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000    1    0
##
## Spec
##      Min. 1st Qu. Median       Mean    3rd Qu.   Max. NA's
## glm     0       0      0 0.00000000 0.00000000 0.0000    0
## glmn    0       0      0 0.00000000 0.00000000 0.0000    0
## mars    0       0      0 0.00000000 0.00000000 0.0000    0
## gam     0       0      0 0.00000000 0.00000000 0.0000    0
## lda     0       0      0 0.03639706 0.05882353 0.1875    0
## rf      0       0      0 0.00000000 0.00000000 0.0000    0
## gbmA    0       0      0 0.00000000 0.00000000 0.0000    0
## svml    0       0      0 0.00000000 0.00000000 0.0000    0
## svmr    0       0      0 0.00000000 0.00000000 0.0000    0
```

```
bwplot(res, metric = "ROC")
```