

Final

Yuechen Liu, Mufeng Xu, Yanhao Li

Contents

Introduction	2
Load, clean, and tidy data	2
Exploratory analysis/ visualization	3
Models	5
Variable Importance	6
GLM	6
GLMN	7
MARS	8
GAM	9
LDA (from the midterm, LDA is the best among LDA, QDA and KNN)	10
Random Forest	11
gbmA	14
svml	15
svmr	17
Comparison	19

```

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

```

stroke = read_csv("./healthcare-dataset-stroke-data.csv") %>%
  mutate(
    bmi = as.numeric(bmi)
  ) %>%
  dplyr::select(-id)

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),
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)
) %>%
relocate(
  age, avg_glucose_level, bmi
)

```

Exploratory analysis/ visualization

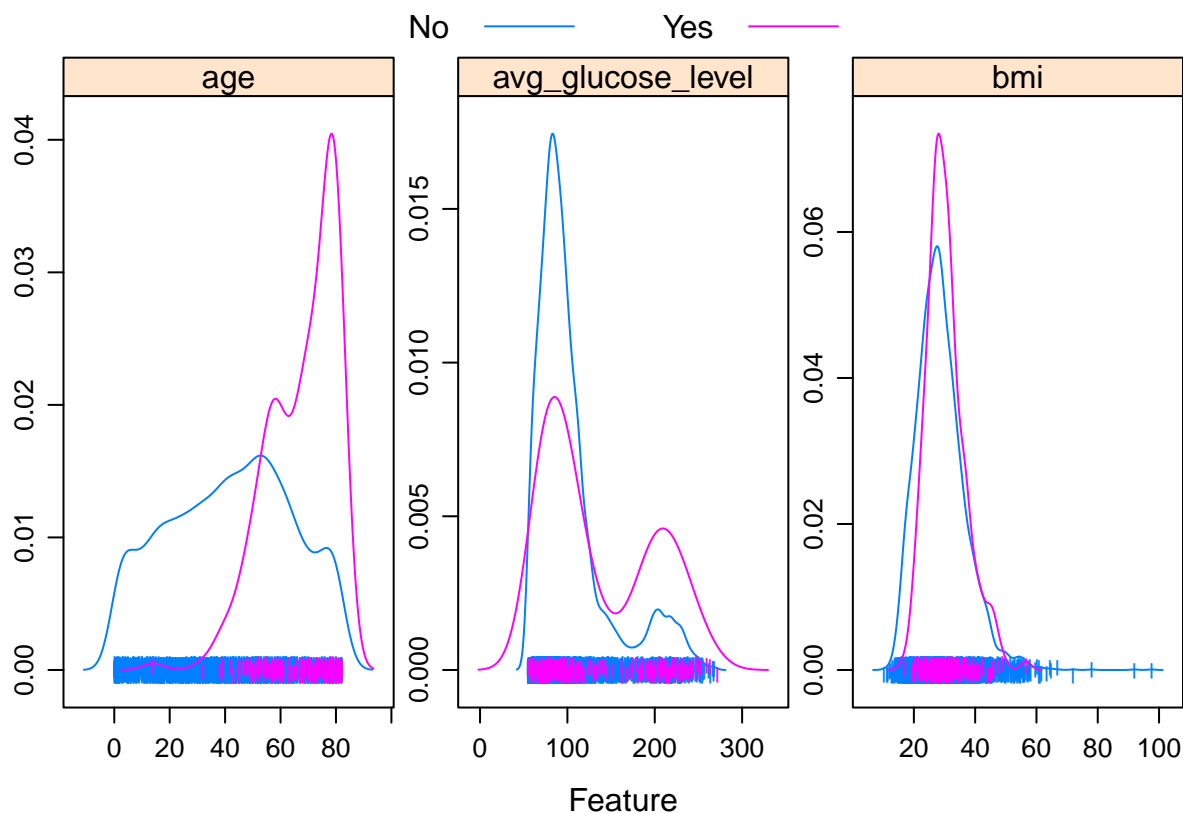
```

stats = tableby(stroke ~ gender + age + hypertension + heart_disease + ever_married + work_type + Residence_type)
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 = stroke1$stroke,
                                   p = 0.8,
                                   list = FALSE)

train = stroke1[indextrain, ]
test = stroke1[-indextrain, ]

x <- stroke1[indextrain, -c(11)]
y <- stroke1$stroke[indextrain] ###train

x2 <- stroke1[-indextrain, -c(11)]
y2 <- stroke1$stroke[-indextrain] ###test

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

Variable Importance

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("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

```

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("Yes", length(glmn.pred.prob))

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

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

```

```

## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##           No  231  30
##           Yes 708  11
##
##           Accuracy : 0.2469
##           95% CI : (0.2202, 0.2752)
##           No Information Rate : 0.9582
##           P-Value [Acc > NIR] : 1
##
##           Kappa : -0.0545
##
##           Mcnemar's Test P-Value : <2e-16
##
##           Sensitivity : 0.26829
##           Specificity : 0.24601
##           Pos Pred Value : 0.01530
##           Neg Pred Value : 0.88506
##           Prevalence : 0.04184
##           Detection Rate : 0.01122
##           Detection Prevalence : 0.73367
##           Balanced Accuracy : 0.25715
##
##           'Positive' Class : Yes
##

```

```
model.glmn$bestTune
```

```
##      alpha      lambda
## 88    0.8 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("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(stroke ~ . ,
                   data = train,
                   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
```

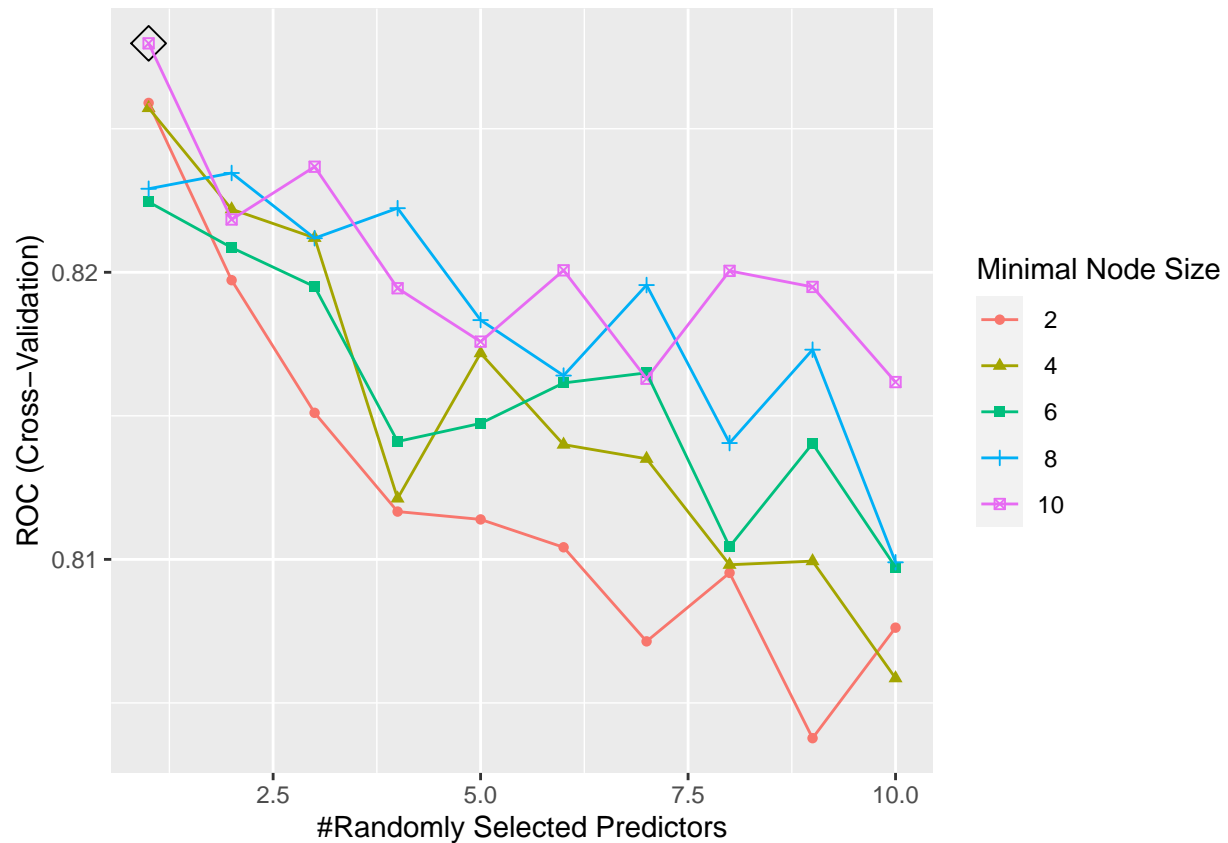
Random Forest

```
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(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("Yes", length(rf.pred.prob))
rf.pred[rf.pred.prob < 0.95] = "No"
confusionMatrix(data = as.factor(rf.pred),
                 reference = y2,
                 positive = "Yes")
```

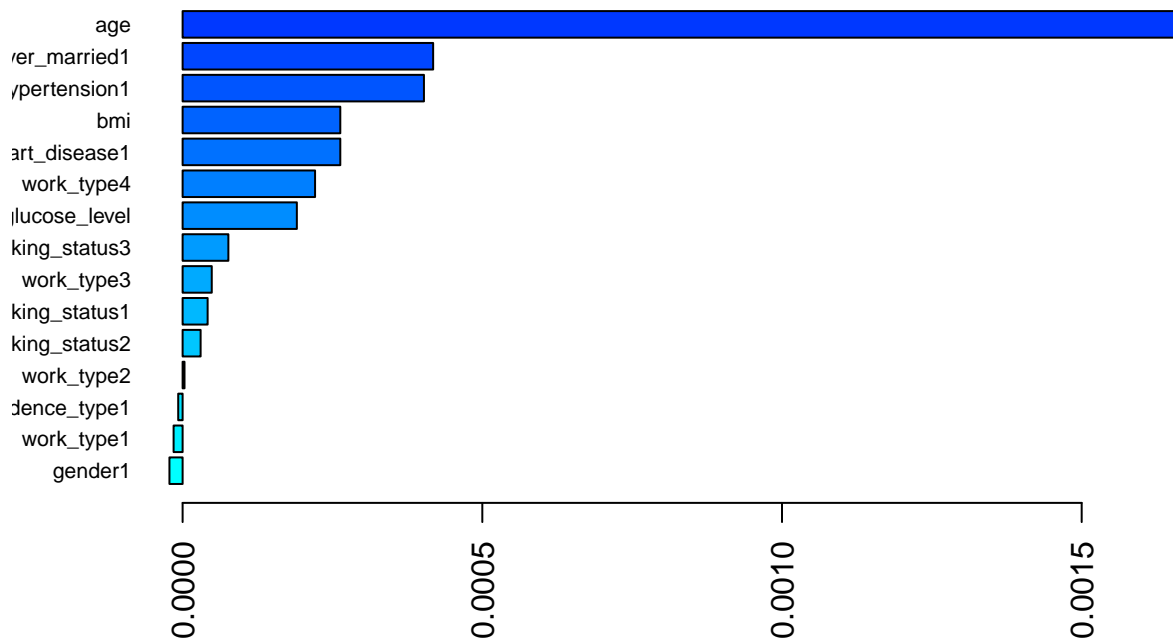
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No  Yes
##      No  227  31
##      Yes 712  10
##
##              Accuracy : 0.2418
##              95% CI : (0.2153, 0.2699)
##      No Information Rate : 0.9582
##      P-Value [Acc > NIR] : 1
##
##              Kappa : -0.0575
##
##      Mcnemar's Test P-Value : <2e-16
```

```
##
##      Sensitivity : 0.24390
##      Specificity : 0.24175
##      Pos Pred Value : 0.01385
##      Neg Pred Value : 0.87984
##      Prevalence : 0.04184
##      Detection Rate : 0.01020
##      Detection Prevalence : 0.73673
##      Balanced Accuracy : 0.24282
##
##      'Positive' Class : Yes
##
```

```
model.rf$bestTune
```

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

```
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

```

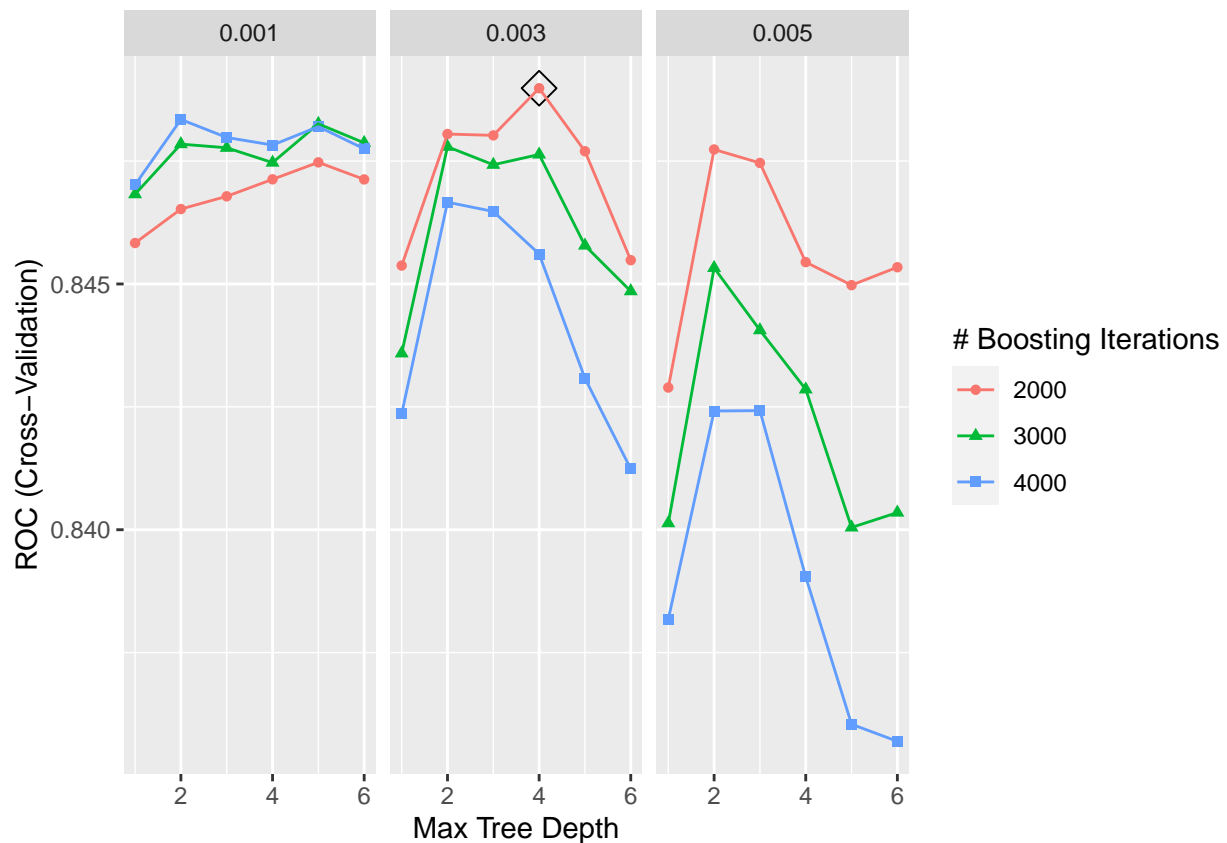
gbmA.grid <- expand.grid(n.trees = c(2000,3000,4000),
                        interaction.depth = 1:6,
                        shrinkage = c(0.001,0.003,0.005),
                        n.minobsinnode = 1)

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("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 194 28
##           Yes 745 13
##
##           Accuracy : 0.2112
##           95% CI : (0.1861, 0.2381)
##           No Information Rate : 0.9582
##           P-Value [Acc > NIR] : 1
##
##           Kappa : -0.0509
##
##           Mcnemar's Test P-Value : <2e-16
##
##           Sensitivity : 0.31707
##           Specificity : 0.20660
##           Pos Pred Value : 0.01715
##           Neg Pred Value : 0.87387
##           Prevalence : 0.04184
##           Detection Rate : 0.01327
##           Detection Prevalence : 0.77347
##           Balanced Accuracy : 0.26184
##
##           'Positive' Class : Yes
##
```

```
model.gbma$bestTune
```

```
##      n.trees interaction.depth shrinkage n.minobsinnode
## 28      2000                4      0.003              1
```

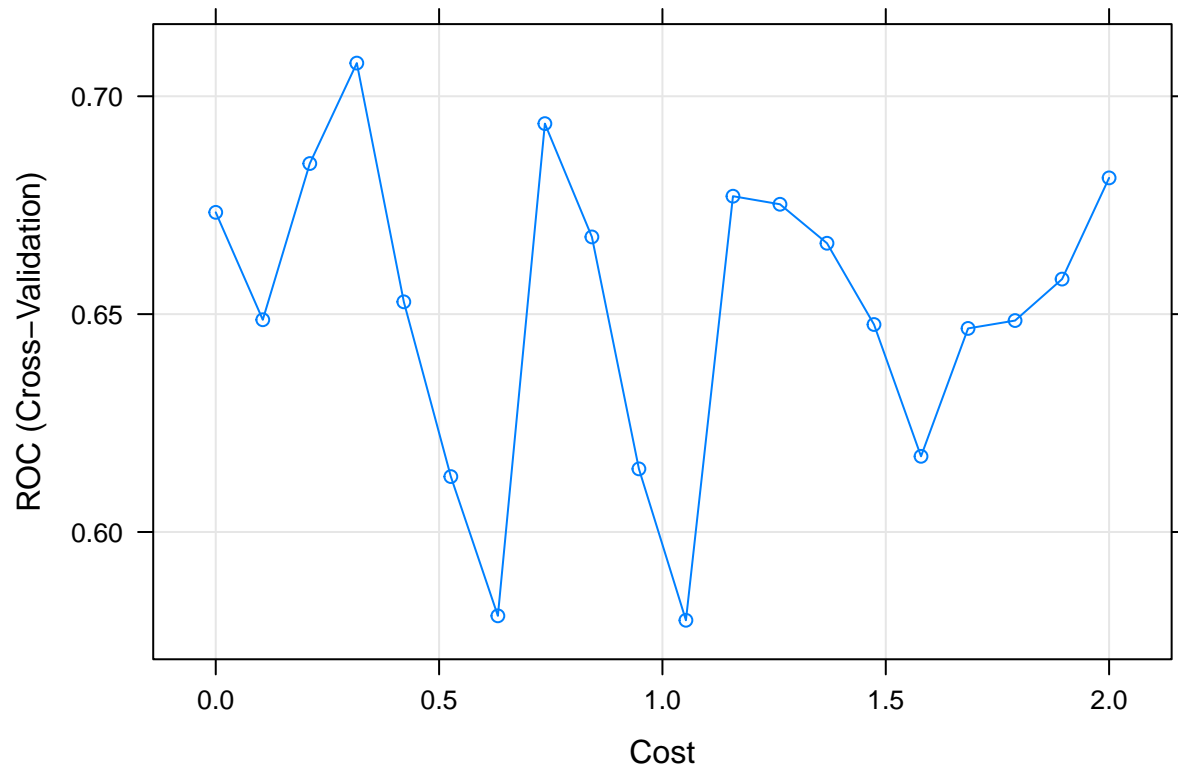
svml

```
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 0.003460544 0.003339861maximum number of iterations reached 0.0
```

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



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

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

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

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

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

```
##
```

```
##           Reference
```

```
## Prediction  No  Yes
```

```
##           No   98  15
```

```
##           Yes 841  26
```

```
##
```

```
##           Accuracy : 0.1265
```

```
##           95% CI : (0.1064, 0.149)
```

```
##           No Information Rate : 0.9582
```

```
##           P-Value [Acc > NIR] : 1
```



```
##
##           Kappa : -0.0246
##
## Mcnemar's Test P-Value : <2e-16
##
##           Sensitivity : 0.63415
##           Specificity : 0.10437
##           Pos Pred Value : 0.02999
##           Neg Pred Value : 0.86726
##           Prevalence : 0.04184
##           Detection Rate : 0.02653
##           Detection Prevalence : 0.88469
##           Balanced Accuracy : 0.36926
##
##           'Positive' Class : Yes
##
```

```
model.svml$bestTune
```

```
##           C
## 4 1.371342
```

svmr

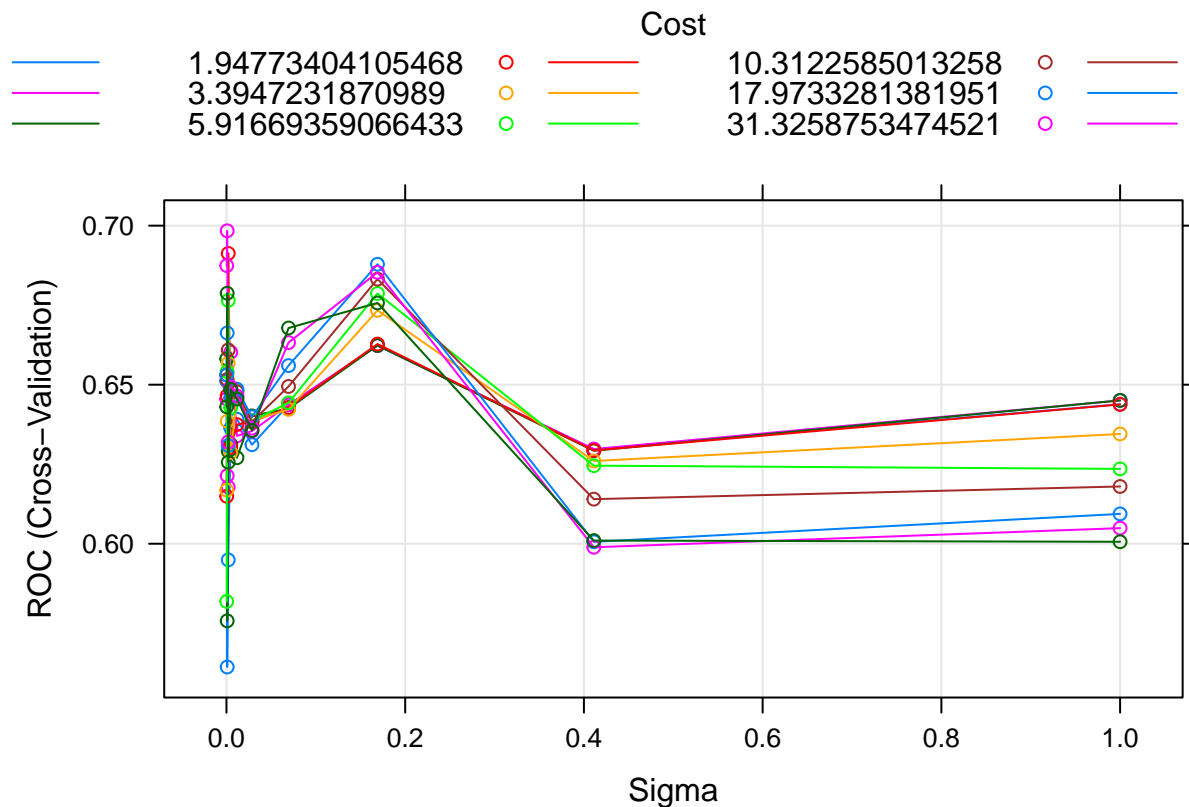
```
svmr.grid <- expand.grid(C = exp(seq(-1,4,len = 10)),
                        sigma = exp(seq(-8,0,len = 10)))
```

```
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 0.004409199 0.004323548maximum 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("Yes", length(svmr.pred.prob))
svmr.pred[svmr.pred.prob < 0.95] = "No"
confusionMatrix(data = as.factor(svmr.pred),
                 reference = y2,
                 positive = "Yes")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No  Yes
##      No  550  32
##      Yes 389   9
##
##           Accuracy : 0.5704
##           95% CI : (0.5387, 0.6017)
##      No Information Rate : 0.9582
##      P-Value [Acc > NIR] : 1
##
##           Kappa : -0.0377
##
##      McNemar's Test P-Value : <2e-16
```

```
##
##      Sensitivity : 0.219512
##      Specificity : 0.585729
##      Pos Pred Value : 0.022613
##      Neg Pred Value : 0.945017
##      Prevalence : 0.041837
##      Detection Rate : 0.009184
##      Detection Prevalence : 0.406122
##      Balanced Accuracy : 0.402621
##
##      'Positive' Class : Yes
##
```

```
model.svmr$bestTune
```

```
##      sigma      C
## 82 0.0008159878 31.32588
```

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,
                      svm1 = model.svm1,
                      svmr = model.svmr))
```

```
summary(res)
```

```
##
## Call:
## summary.resamples(object = res)
##
## Models: glm, glmn, mars, gam, lda, rf, gbma, svm1, svmr
## Number of resamples: 10
##
## ROC
##      Min.    1st Qu.    Median    Mean    3rd Qu.    Max. NA's
## glm  0.7814162 0.8122653 0.8404255 0.8461671 0.8712062 0.9224030 0
## glmn 0.7510951 0.8341188 0.8586270 0.8507343 0.8754302 0.8965895 0
## mars 0.7291927 0.7933061 0.8578448 0.8343779 0.8720862 0.8922090 0
## gam  0.7525031 0.8237641 0.8619368 0.8464438 0.8777647 0.8889237 0
## lda  0.7210576 0.8173865 0.8394380 0.8365564 0.8721449 0.8992491 0
## rf   0.7424906 0.8071417 0.8252503 0.8279744 0.8464951 0.9009700 0
## gbma 0.7701815 0.8286139 0.8539727 0.8489821 0.8731618 0.8890801 0
## svm1 0.5028160 0.6764828 0.7312265 0.7076101 0.7701815 0.8155507 0
## svmr 0.5979349 0.6596429 0.6952734 0.6983827 0.7425688 0.7952128 0
```

```
##
## Sens
##      Min.   1st Qu.   Median     Mean   3rd Qu.  Max. NA's
## glm  0.9973404 1.0000000 1.0000000 0.9997340 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 0.9827128 0.9906915 0.9890957 0.9920213    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
## svm1 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 0.0000000 0.0000000 0.0000000 0.0000    0
## glmn      0      0 0.0000000 0.0000000 0.0000000 0.0000    0
## mars      0      0 0.0000000 0.0000000 0.0000000 0.0000    0
## gam      0      0 0.0000000 0.0000000 0.0000000 0.0000    0
## lda      0      0 0.05882353 0.06029412 0.1038603 0.1875    0
## rf      0      0 0.0000000 0.0000000 0.0000000 0.0000    0
## gbmA      0      0 0.0000000 0.0000000 0.0000000 0.0000    0
## svm1      0      0 0.0000000 0.0000000 0.0000000 0.0000    0
## svmr      0      0 0.0000000 0.0000000 0.0000000 0.0000    0
```

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

