

Final Secondary Analysis

Tianshu Liu, Lincole Jiang, Jiong Ma

Contents

1	Model Training	2
1.1	Secondary Analysis	2
1.1.1	Logistic Regression	2
1.1.2	Penalized Logistic Regression	3
1.1.3	Generalized Additive Model (GAM) for classification	5
1.1.4	Multivariate Adaptive Regression Splines (MARS) for classification	7
1.1.5	Linear Discriminant Analysis (LDA)	10
1.1.6	Quadratic Discriminant Analysis (QDA)	10
1.1.7	Naive Bayes (NB)	10
1.1.8	Bagging	11
1.1.9	Random Forest	13
1.1.10	Boosting	15
1.1.11	Classification Trees	17
1.1.12	Support Vector Machine (SVM)	19
1.2	Model Selection	23
1.3	Training / Testing Error	25

```
library(tidyverse)
library(summarytools)
library(corrplot)
library(caret)
library(vip)
library(rpart.plot)
library(ranger)
library(e1071)
```

1 Model Training

1.1 Secondary Analysis

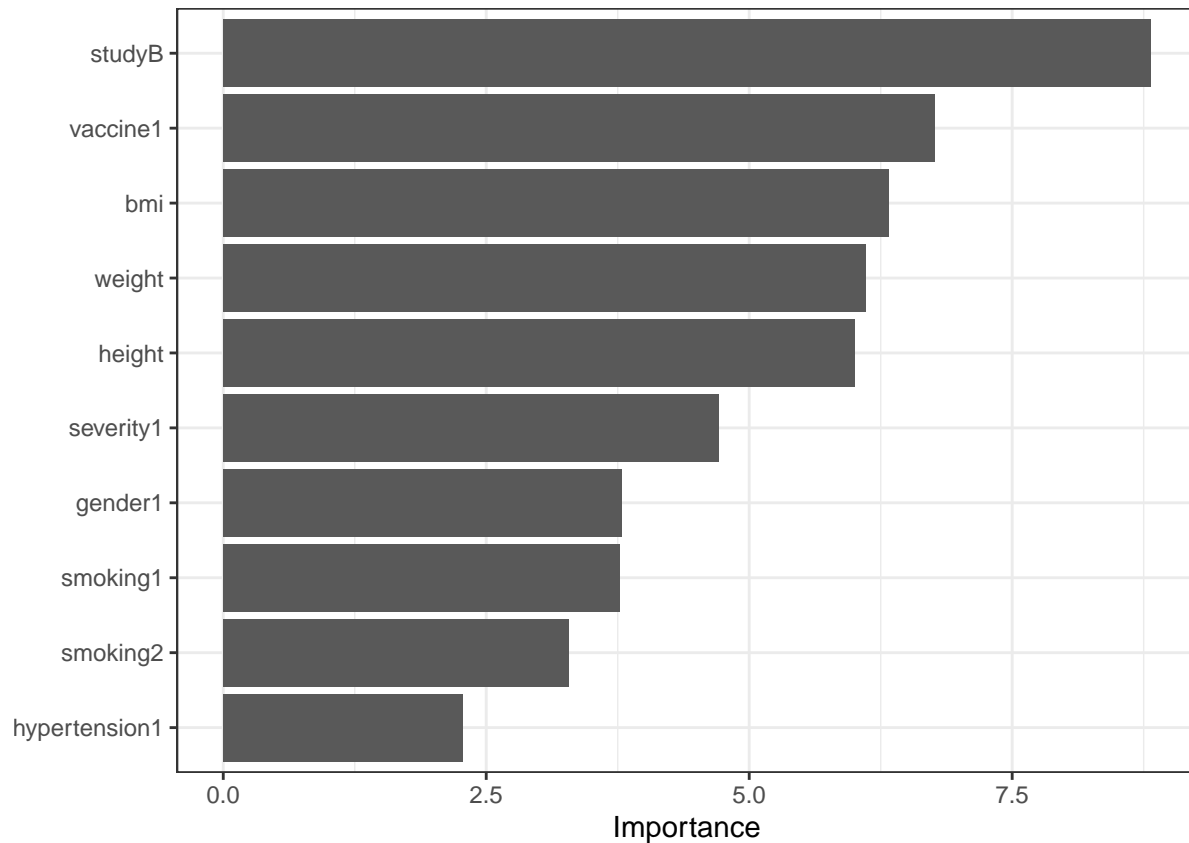
```
ctrl1 <- trainControl(method = "cv", number = 10)
```

1.1.1 Logistic Regression

```
set.seed(2023)
glm.fit <- train(x = train.x,
                 y = train.bin.y,
                 method = 'glm',
                 trControl = ctrl1)
coef(glm.fit$finalModel)
```

```
##      (Intercept)          age      gender1      race2      race3
## -85.313351100    0.014271893  -0.323467202  -0.104376256  -0.039351201
##           race4      smoking1      smoking2      height      weight
##   0.003550318    0.367190652    0.502782618    0.502637208  -0.545510559
##           bmi hypertension1    diabetes1      SBP      LDL
##   1.634689792    0.325697328  -0.070567897  -0.005832901  -0.001829020
##      vaccine1      severity1      studyB      studyC
##  -0.600151829    0.761039467  -1.066825060  -0.031460504
```

```
vip(glm.fit$finalModel) + theme_bw()
```



1.1.2 Penalized Logistic Regression

```

glmnetGrid <- expand.grid(.alpha = seq(0, 1, length = 21),
                          .lambda = exp(seq(-10, -5, length = 15)))
set.seed(2023)
glmnet.fit <- train(train.x,
                    train.bin.y,
                    method = 'glmnet',
                    tuneGrid = glmnetGrid,
                    trControl = ctrl1)

glmnet.fit$bestTune

```

```

##      alpha      lambda
## 292  0.95 0.0003869779

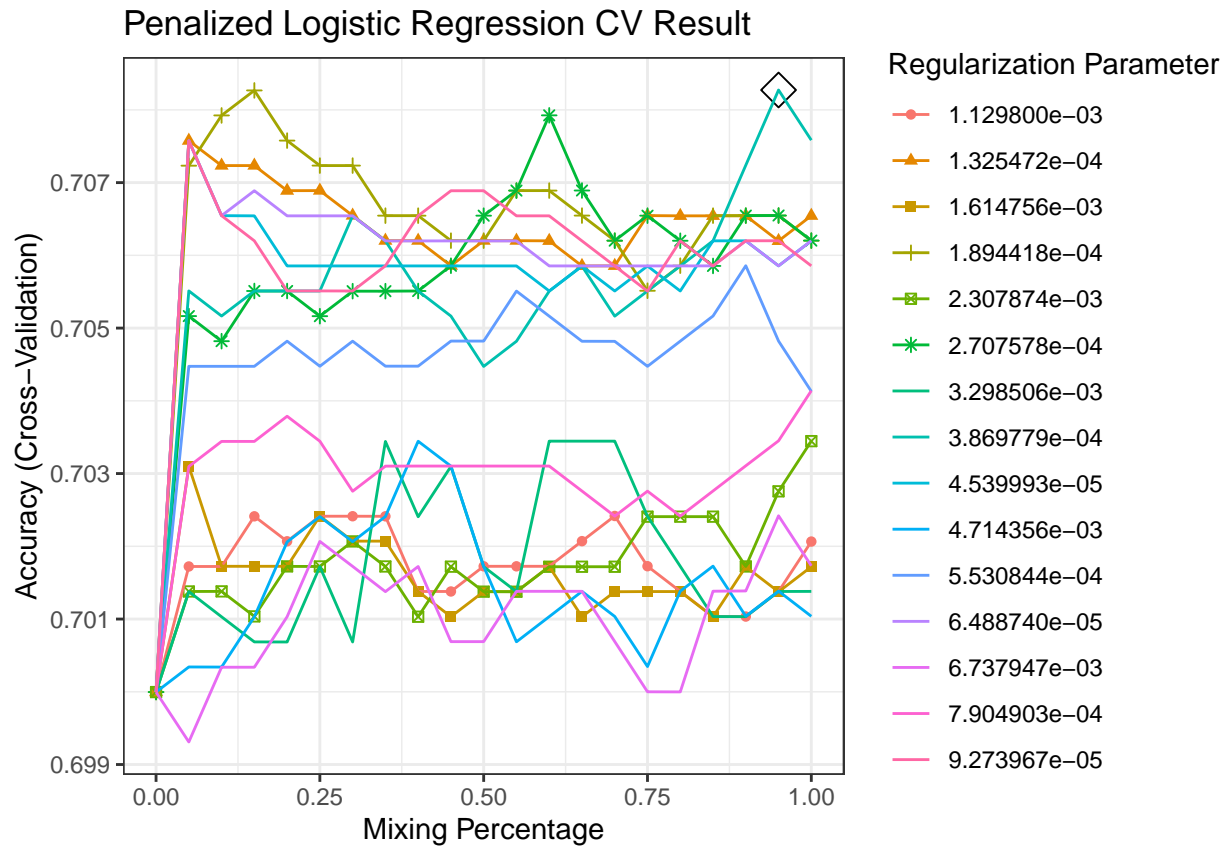
```

```

myCol <- rainbow(25)
myPar <- list(superpose.symbol = list(col = myCol),
              superpose.line = list(col = myCol))

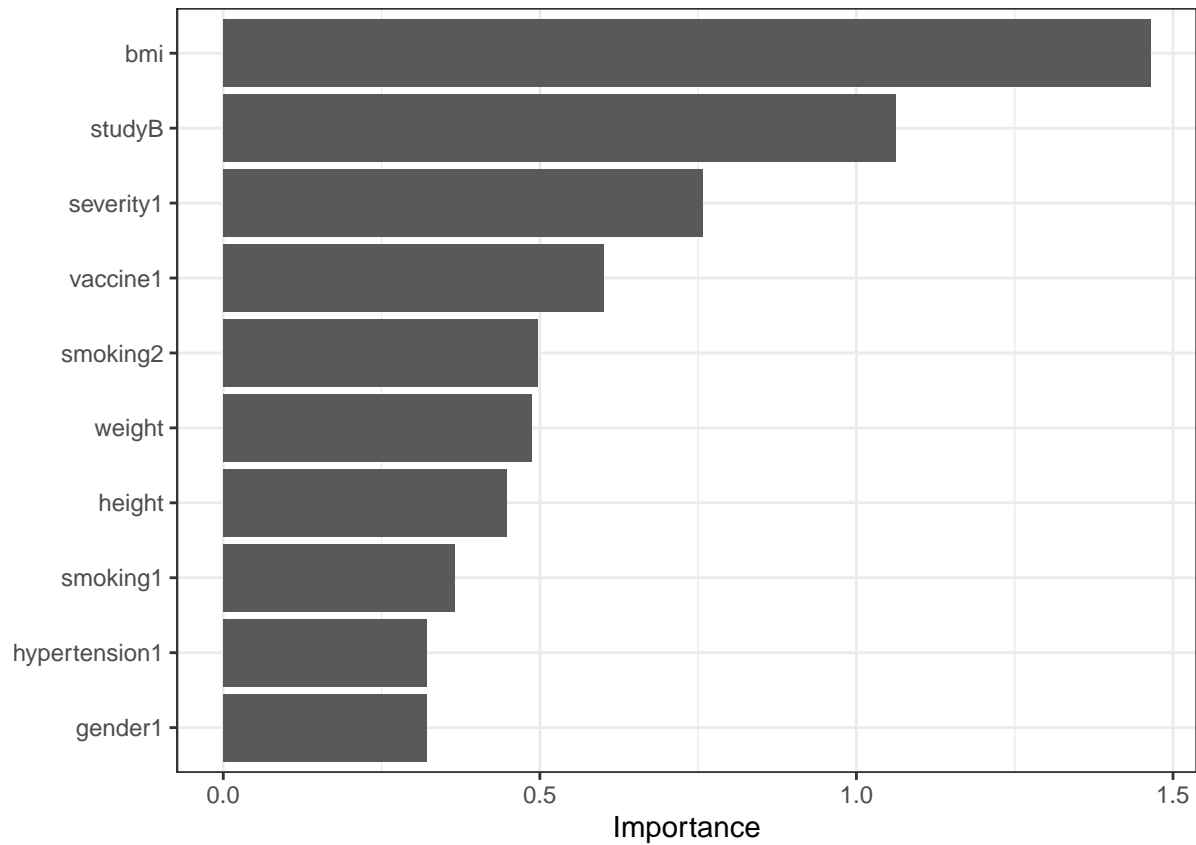
ggplot(glmnet.fit, highlight = TRUE) +
  labs(title="Penalized Logistic Regression CV Result") +
  theme_bw()

```



```
ggsave("./figure/penal_logi_cv.jpeg", dpi = 500)
```

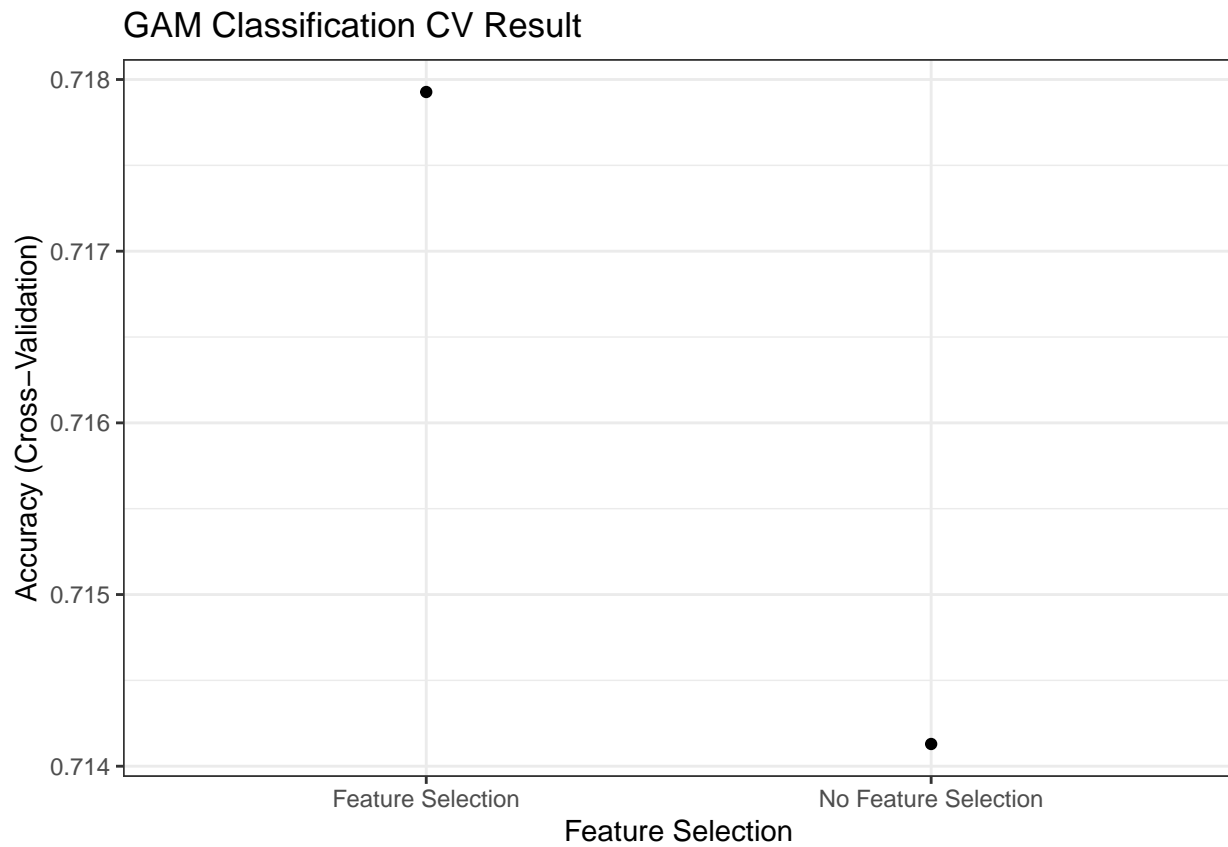
```
#coef(glmn.fit$finalModel)
vip(glmn.fit$finalModel) + theme_bw()
```



1.1.3 Generalized Additive Model (GAM) for classification

```
set.seed(2023)
gam.bin.fit <- train(train.x,
                     train.bin.y,
                     method = "gam",
                     trControl = ctrl1)

ggplot(gam.bin.fit) +
  labs(title = "GAM Classification CV Result") +
  theme_bw()
```



```
ggsave("./figure/gam_binned_cv.jpeg", dpi = 500)
```

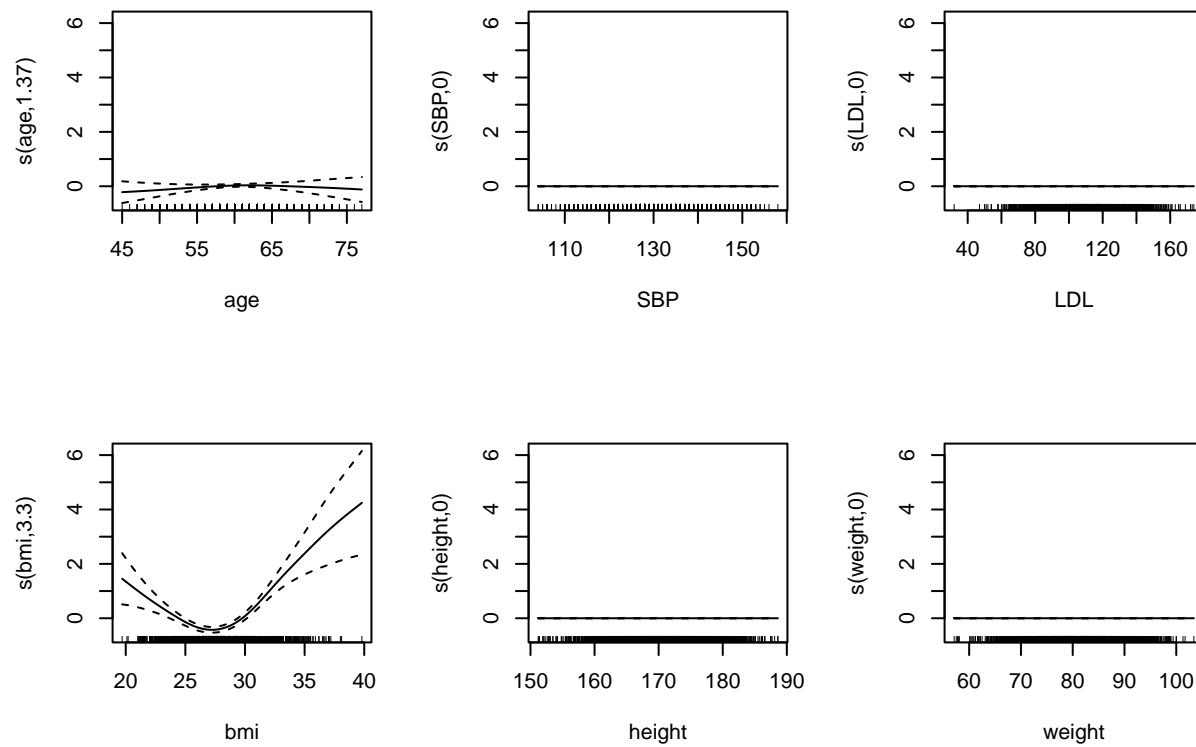
```
gam.bin.fit$bestTune
```

```
## select method
```

```
## 2 TRUE GCV.Cp
```

```
par(mfrow=c(2, 3))
```

```
plot(gam.bin.fit$finalModel)
```



```
par(mfrow=c(1, 1))
```

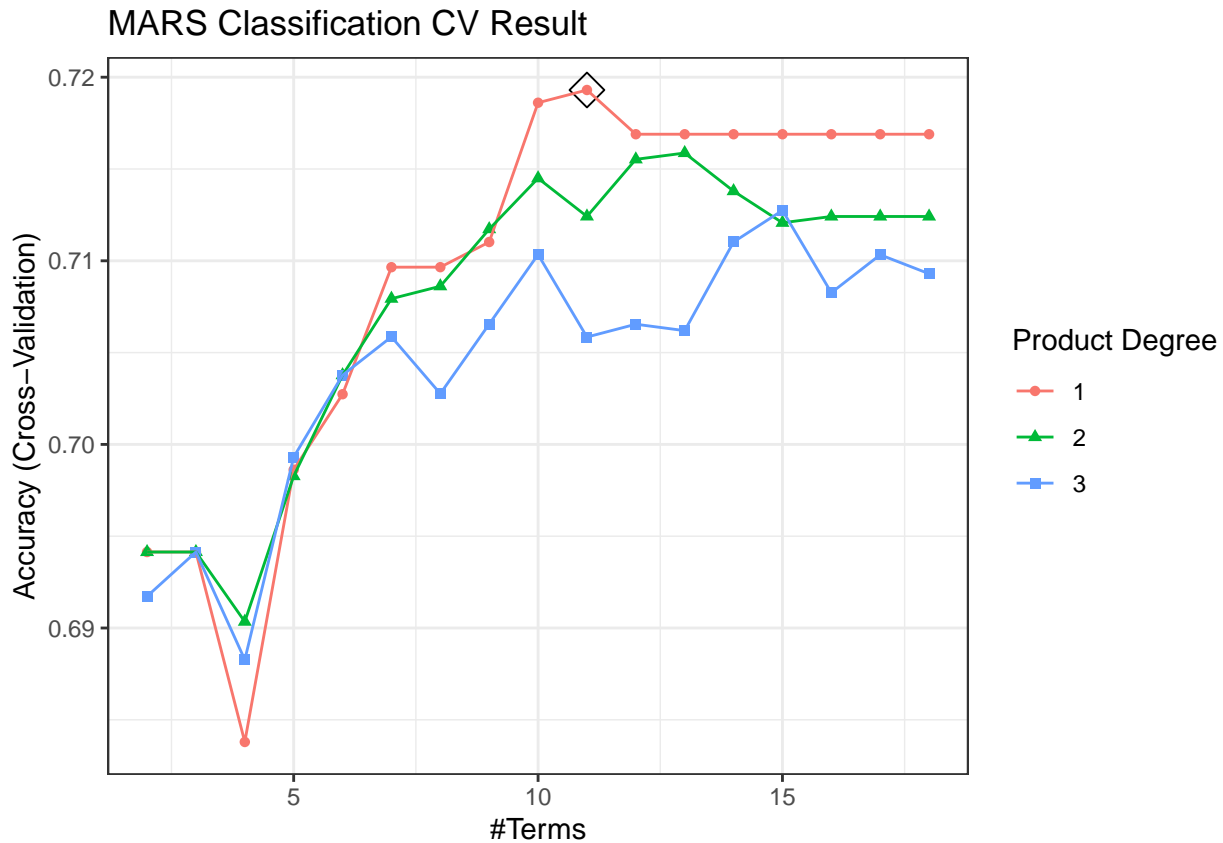
1.1.4 Multivariate Adaptive Regression Splines (MARS) for classification

```
set.seed(2023)
mars.bin.fit <- train(train.x,
                      train.bin.y,
                      method = "earth",
                      tuneGrid = expand.grid(degree = 1:3,
                                             nprune = 2:ncol(train.x)),
                      trControl = ctrl1)

mars.bin.fit$bestTune

##      nprune degree
## 10       11      1

ggplot(mars.bin.fit, highlight = TRUE) +
  labs(title = "MARS Classification CV Result") +
  theme_bw()
```



```
ggsave("./figure/mars_binned_cv.jpeg", dpi = 500)
```

```
mars.bin.fit$bestTune
```

```
##      nprune degree
```

```
## 10      11      1
```

```
coef(mars.bin.fit$finalModel) %>%
```

```
  broom::tidy() %>%
```

```
  knitr::kable()
```

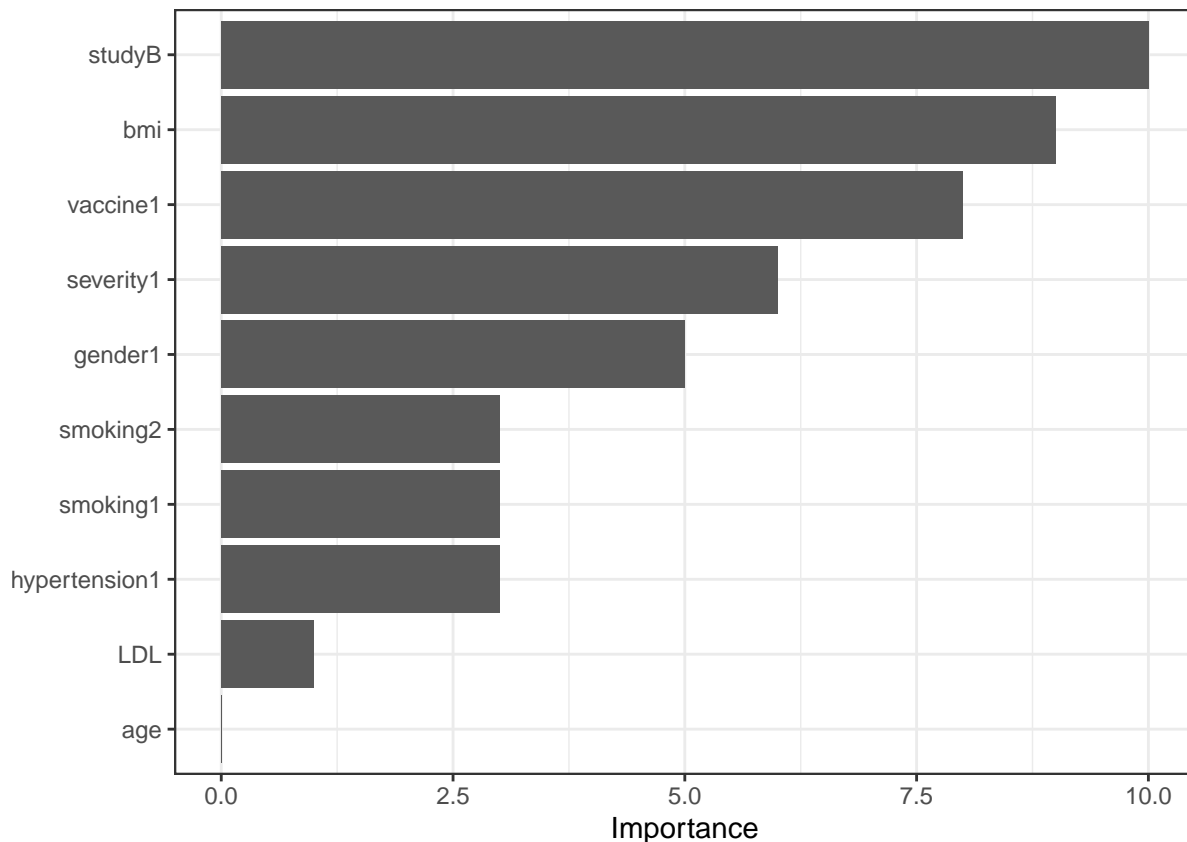
names	x
(Intercept)	1.1011705
studyB	-1.0779091
h(bmi-26.9)	0.2900212
h(26.9-bmi)	0.2935615
vaccine1	-0.6217928
severity1	0.7969230
gender1	-0.3261333
hypertension1	0.3099788
smoking1	0.3912885
smoking2	0.5358382
h(LDL-157)	-0.1512777

```
summary(mars.bin.fit$finalModel)
```

```
## Call: earth(x=matrix[2900,18], y=factor.object, keepxy=TRUE,
```



```
##          glm=list(family=function.object, maxit=100), degree=1, nprune=11)
##
## GLM coefficients
##          gt30
## (Intercept)  1.1011705
## gender1      -0.3261333
## smoking1      0.3912885
## smoking2      0.5358382
## hypertension1 0.3099788
## vaccine1     -0.6217928
## severity1     0.7969230
## studyB       -1.0779090
## h(26.9-bmi)   0.2935615
## h(bmi-26.9)   0.2900212
## h(LDL-157)    -0.1512777
##
## GLM (family binomial, link logit):
## nulldev  df      dev  df  devratio    AIC iters converged
## 3571.35 2899  3204.42 2889    0.103   3226    4          1
##
## Earth selected 11 of 14 terms, and 9 of 18 predictors (nprune=11)
## Termination condition: RSq changed by less than 0.001 at 14 terms
## Importance: studyB, bmi, vaccine1, severity1, gender1, smoking1, smoking2, ...
## Number of terms at each degree of interaction: 1 10 (additive model)
## Earth GCV 0.1906834  RSS 545.0022  GRSq 0.1024844  RSq 0.1148255
vip(mars.bin.fit$finalModel) + theme_bw()
```



1.1.5 Linear Discriminant Analysis (LDA)

```
set.seed(2023)
lda.fit <- train(train.x,
                 train.bin.y,
                 method = "lda",
                 trControl = ctrl1)
```

1.1.6 Quadratic Discriminant Analysis (QDA)

```
set.seed(2023)
qda.fit <- train(train.x,
                 train.bin.y,
                 method = "qda",
                 trControl = ctrl1)
```

1.1.7 Naive Bayes (NB)

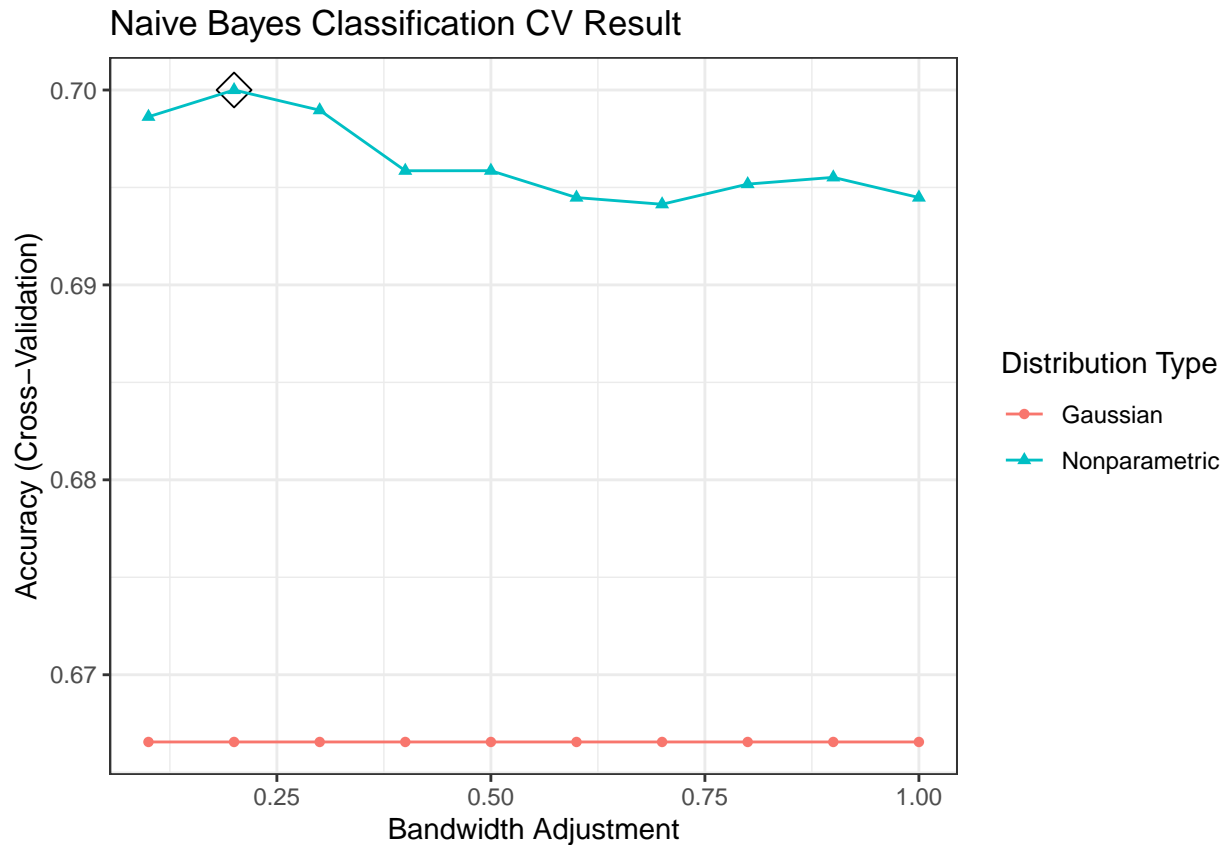
```
nbGrid <- expand.grid(usekernel = c(FALSE, TRUE),
                     fL = 1,
                     adjust = seq(0.1, 1, by = .1))
```

```
set.seed(2023)
nb.fit <- train(train.x,
               train.bin.y,
               method = "nb",
               tuneGrid = nbGrid,
               trControl = ctrl1)

nb.fit$bestTune
```

```
##    fL usekernel adjust
## 12  1      TRUE    0.2
```

```
ggplot(nb.fit, highlight = TRUE) +
  labs(title = "Naive Bayes Classification CV Result") +
  theme_bw()
```



```
ggsave("./figure/nb_cv.jpeg", dpi = 500)
```

1.1.8 Bagging

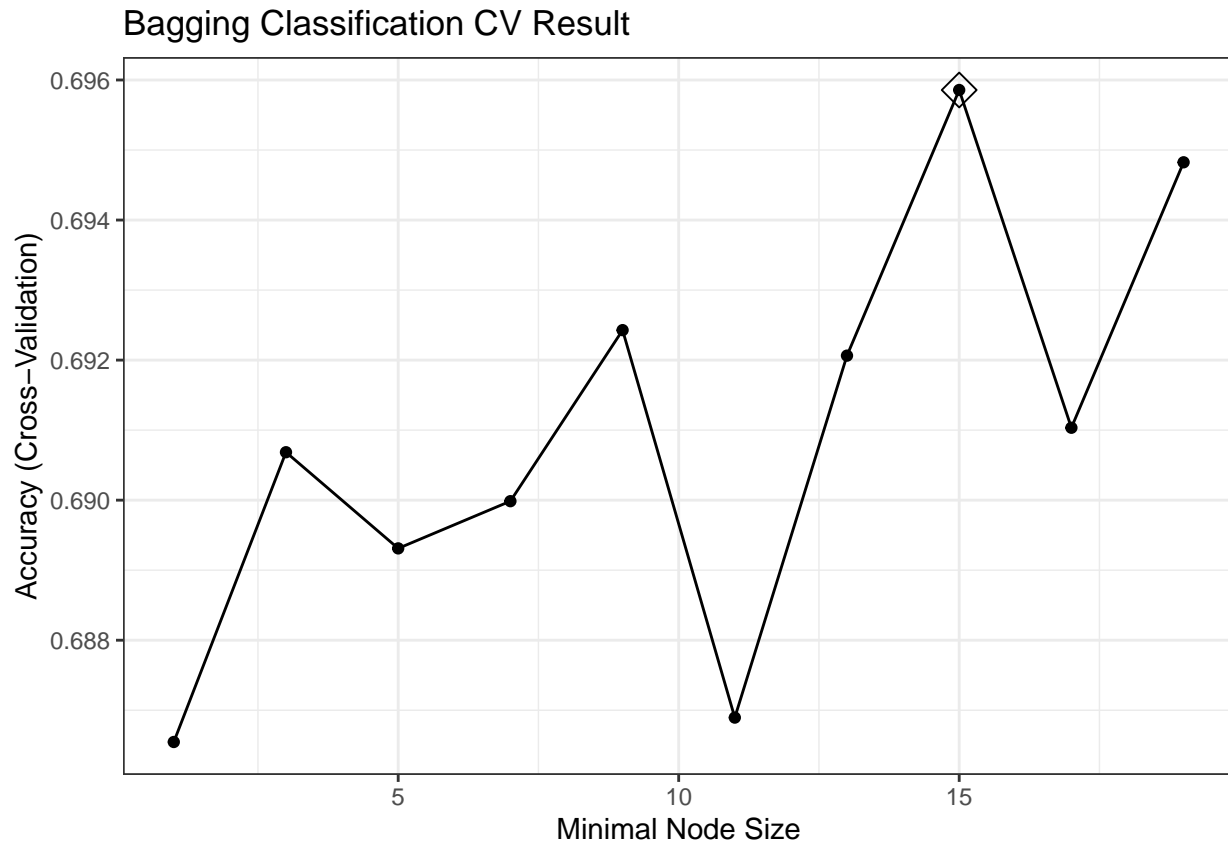
```
bag.grid2 <- expand.grid(mtry = ncol(train.x),
                        splitrule = "gini",
                        min.node.size = seq(1, 19, by = 2))

set.seed(2023)
bag.fit2 <- train(train.x,
                  train.bin.y,
                  method = "ranger",
                  tuneGrid = bag.grid2,
                  trControl = ctrl1)

bag.fit2$bestTune

##  mtry splitrule min.node.size
##  8    18      gini           15

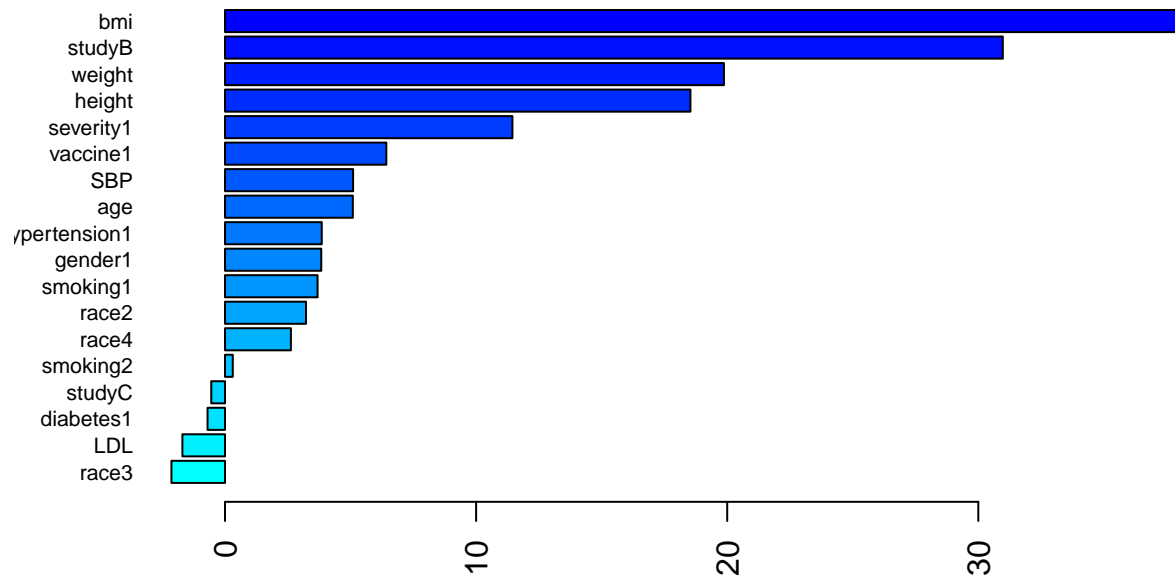
ggplot(bag.fit2, highlight = TRUE) +
  labs(title = "Bagging Classification CV Result") +
  theme_bw()
```



```
ggsave("./figure/bagging_classification_cv.jpeg", dpi = 500)
```

```
bag.final.per2 <- ranger(recovery_time ~ .,
  data = train.bin.dat.matrix,
  mtry = ncol(train.x),
  splitrule = "gini",
  min.node.size = bag.fit2$bestTune[[3]],
  importance = "permutation",
  scale.permutation.importance = TRUE)
```

```
barplot(sort(ranger::importance(bag.final.per2),
  decreasing = FALSE),
  las = 2, horiz = TRUE, cex.names = 0.7,
  col = colorRampPalette(colors = c("cyan", "blue"))(ncol(train.x)))
```



1.1.9 Random Forest

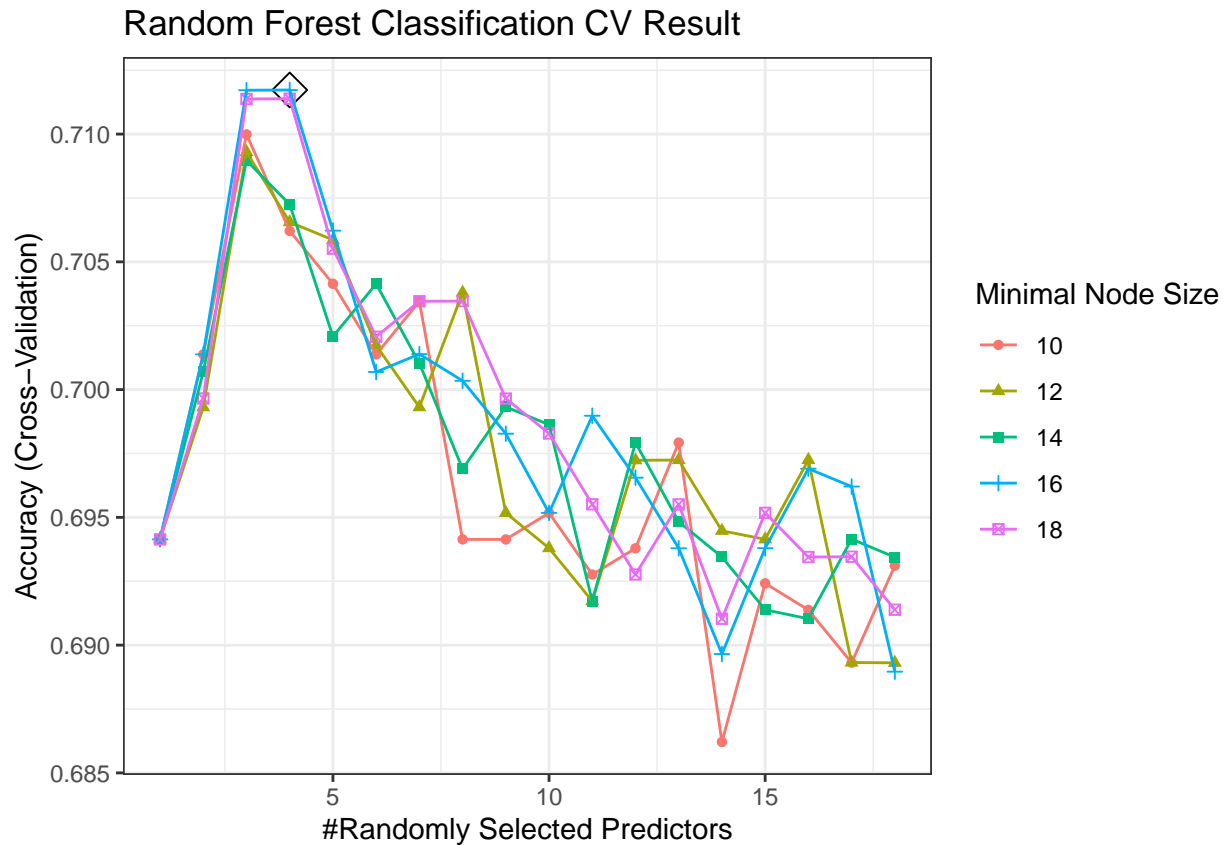
```
rf.grid2 <- expand.grid(mtry = 1:ncol(train.x),
                      splitrule = "gini",
                      min.node.size = seq(10, 18, by = 2))

set.seed(2023)
rf.fit2 <- train(train.x,
                train.bin.y,
                method = "ranger",
                tuneGrid = rf.grid2,
                trControl = ctrl1)

rf.fit2$bestTune

##      mtry splitrule min.node.size
## 19      4      gini             16

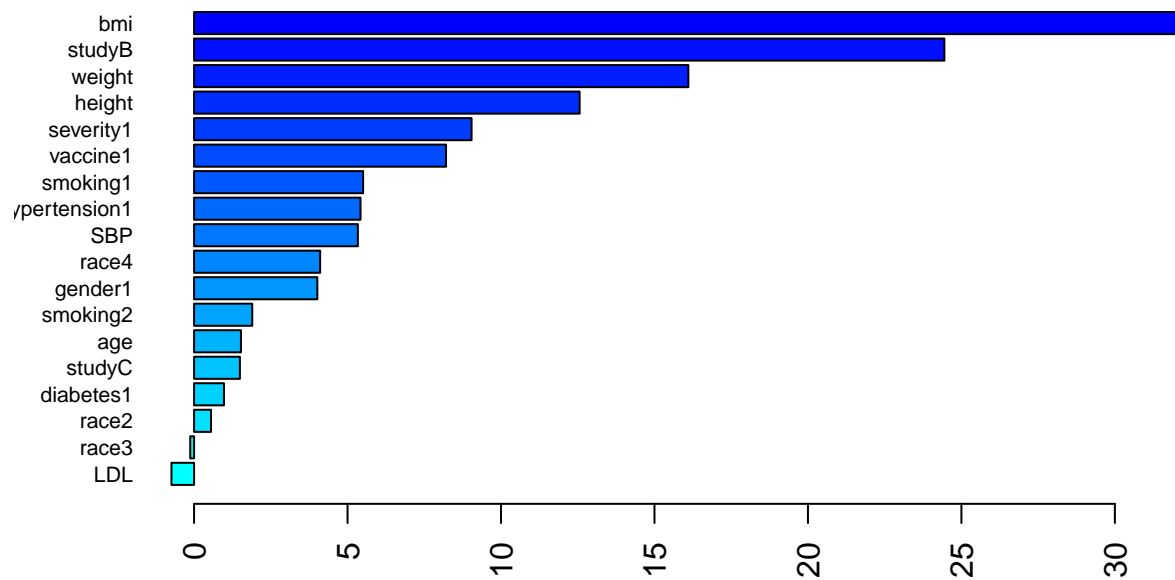
ggplot(rf.fit2, highlight = TRUE) +
  labs(title = "Random Forest Classification CV Result") +
  theme_bw()
```



```
ggsave("./figure/rf_classification_cv.jpeg", dpi = 500)

rf.final.per2 <- ranger(recovery_time ~ .,
  data = train.bin.dat.matrix,
  mtry = rf.fit2$bestTune[[1]],
  splitrule = "gini",
  min.node.size = rf.fit2$bestTune[[3]],
  importance = "permutation",
  scale.permutation.importance = TRUE)

barplot(sort(ranger::importance(rf.final.per2), decreasing = FALSE),
  las = 2, horiz = TRUE, cex.names = 0.7,
  col = colorRampPalette(colors = c("cyan", "blue"))(ncol(train.x)))
```



1.1.10 Boosting

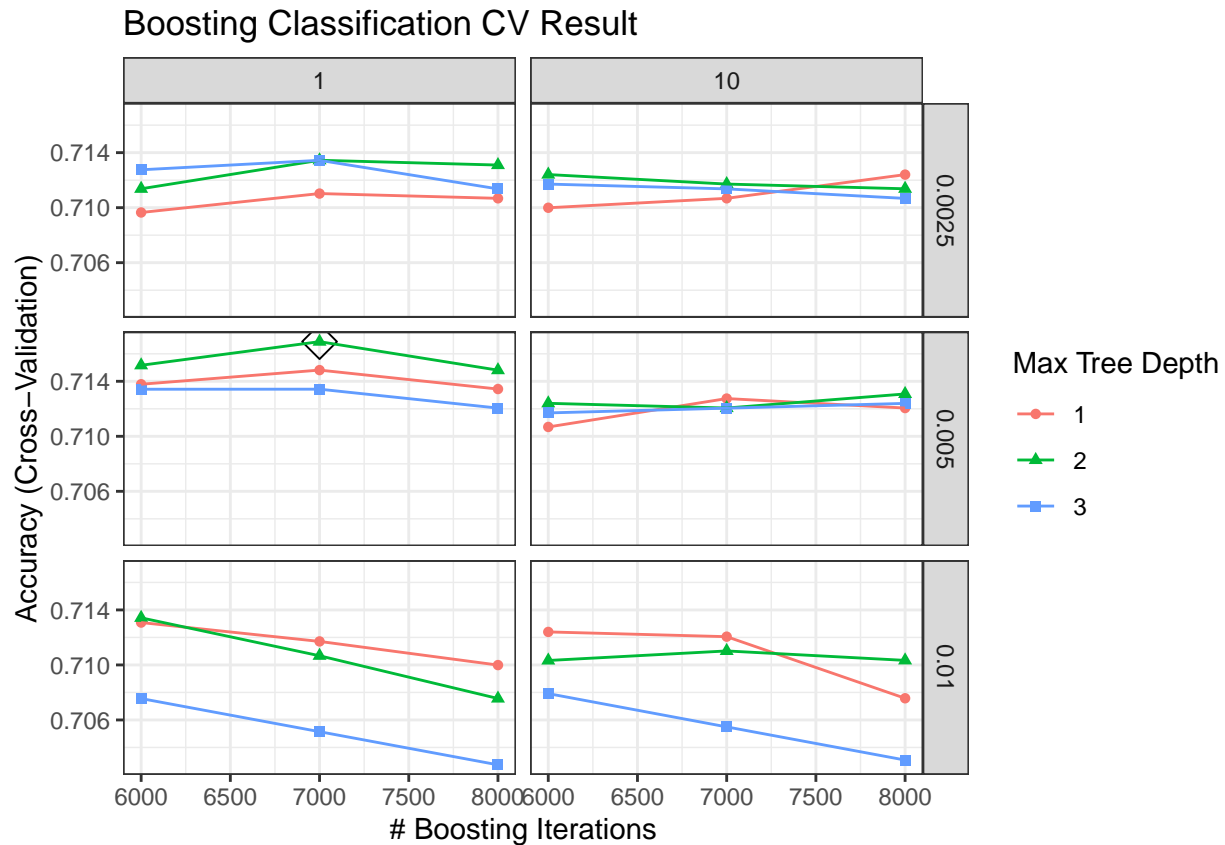
```
set.seed(2023)
bst.grid2 <- expand.grid(n.trees = c(6000, 7000, 8000),
                        interaction.depth = 1:3,
                        shrinkage = c(0.0025, 0.005, 0.01),
                        n.minobsinnode = c(1,10))

bst.fit2 <- train(train.x,
                  train.bin.y,
                  method = "gbm",
                  tuneGrid = bst.grid2,
                  trControl = ctrl1,
                  verbose = FALSE)

bst.fit2$bestTune

##      n.trees interaction.depth shrinkage n.minobsinnode
## 26      7000                2      0.005              1

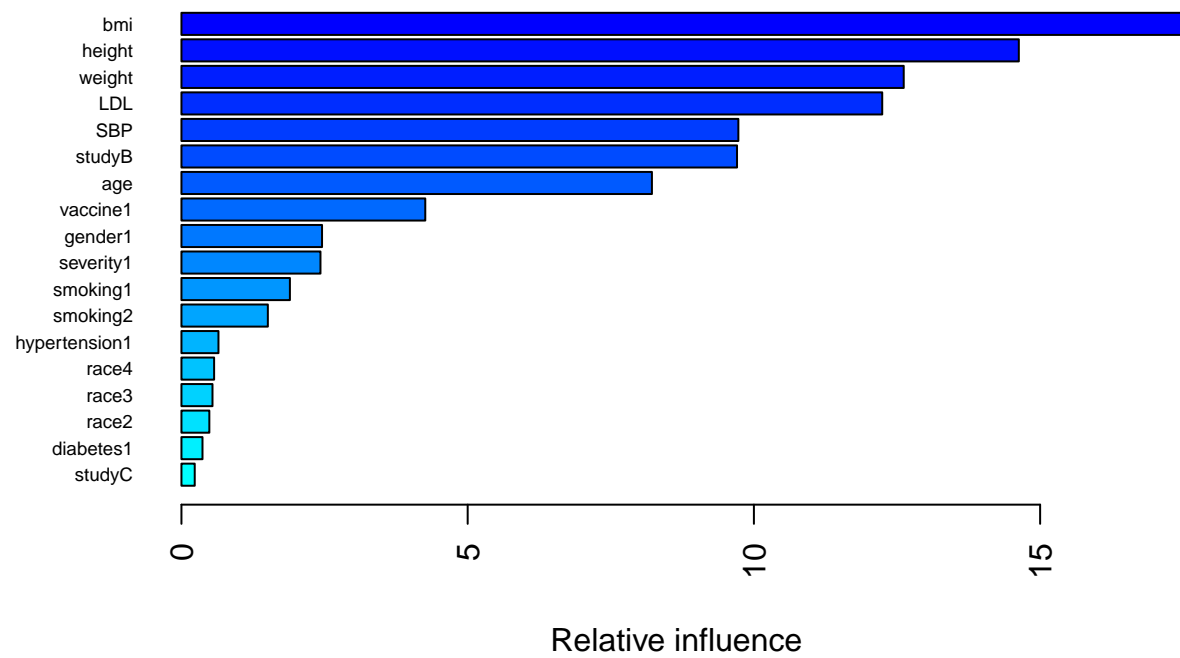
ggplot(bst.fit2, highlight = TRUE) +
  labs(title = "Boosting Classification CV Result") +
  theme_bw()
```



```
ggsave("./figure/boosting_classification_cv.jpeg", dpi = 500)
```

```
# Variable Importance
```

```
summary(bst.fit2$finalModel, las = 2, cBars = ncol(train.x), cex.names = 0.6)
```



```
##          var    rel.inf
```



```
## bmi                bmi 17.4678617
## height             height 14.6274336
## weight             weight 12.6168495
## LDL                LDL 12.2417343
## SBP                SBP 9.7283254
## studyB             studyB 9.7051849
## age                age 8.2186384
## vaccine1           vaccine1 4.2592937
## gender1            gender1 2.4561002
## severity1          severity1 2.4286230
## smoking1           smoking1 1.8945366
## smoking2           smoking2 1.5093649
## hypertension1      hypertension1 0.6466999
## race4              race4 0.5705586
## race3              race3 0.5423003
## race2              race2 0.4871927
## diabetes1          diabetes1 0.3675255
## studyC             studyC 0.2317767
```

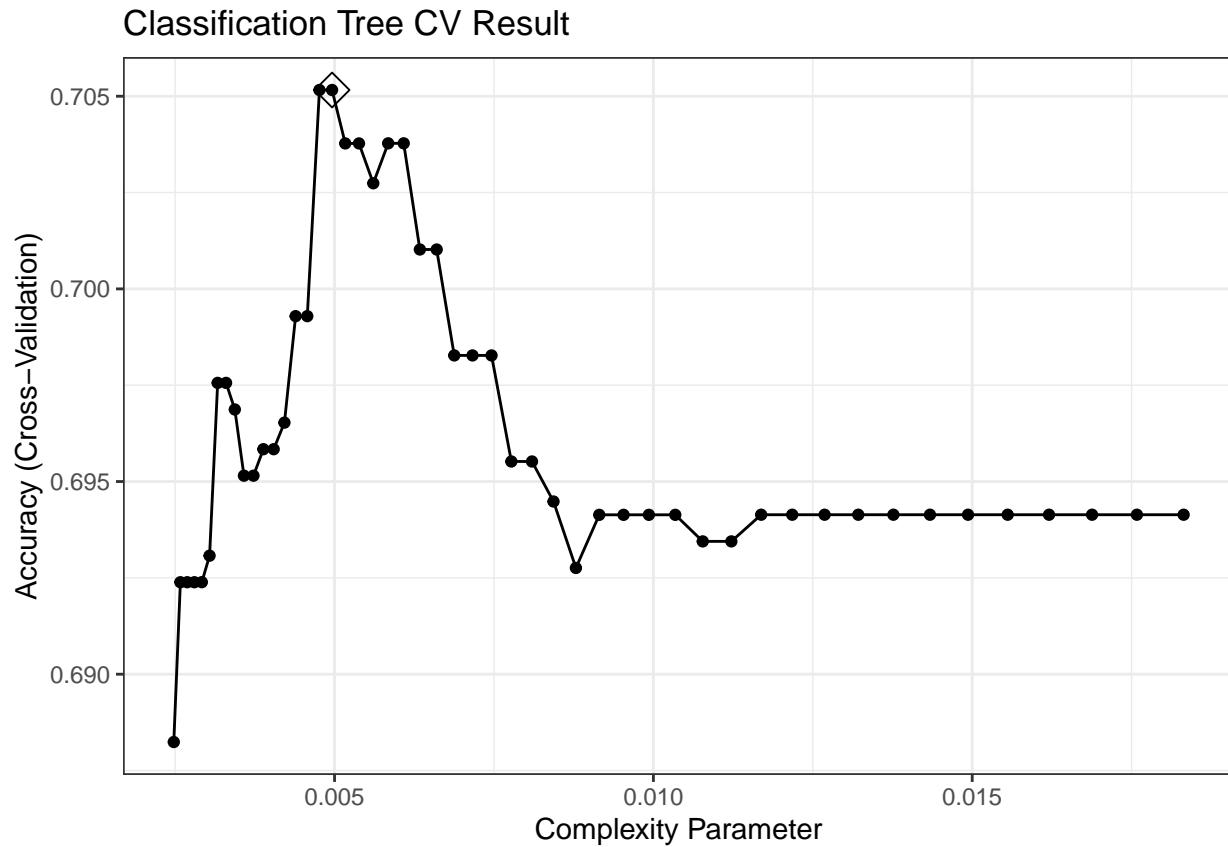
1.1.11 Classification Trees

```
rpart.grid = expand.grid(cp = exp(seq(-6,-4, len = 50)))
set.seed(2023)
rpart.fit2 <- train(train.x,
                    train.bin.y,
                    method = "rpart",
                    tuneGrid = rpart.grid,
                    trControl = ctrl1)

rpart.fit2$bestTune
```

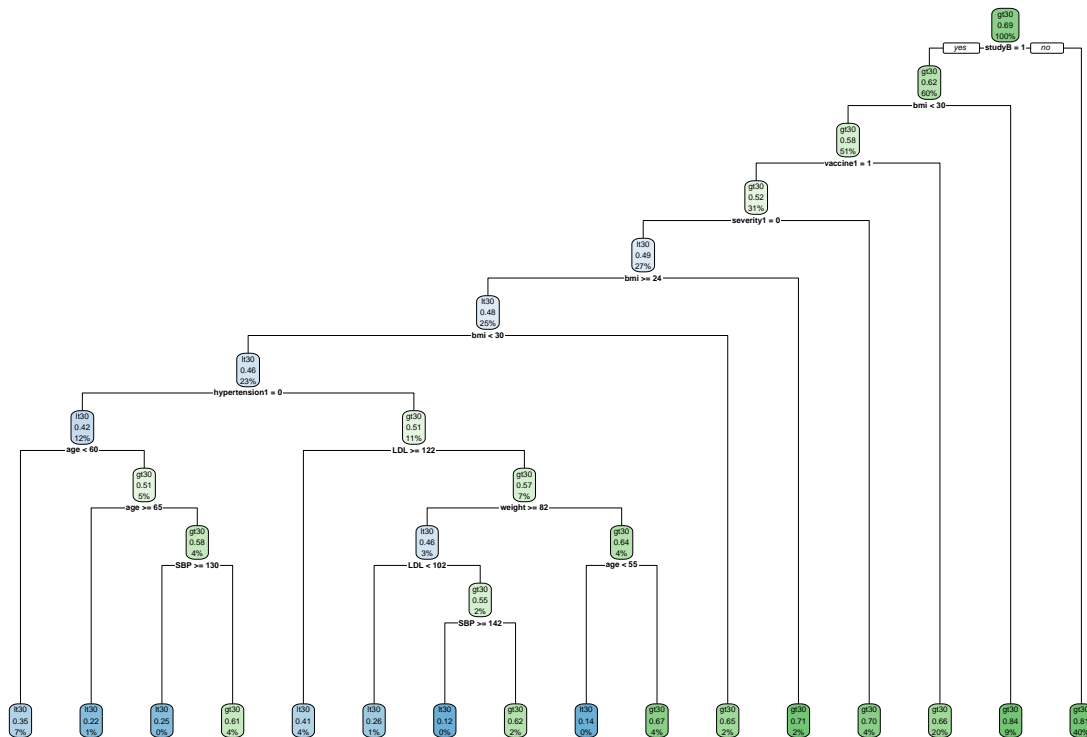
```
##                cp
## 18 0.004961126

ggplot(rpart.fit2, highlight = TRUE) +
  labs(title = "Classification Tree CV Result") +
  theme_bw()
```



```
ggsave("./figure/rpart2_cv.jpeg", dpi = 500)
```

```
rpart.plot(rpart.fit2$finalModel)
```

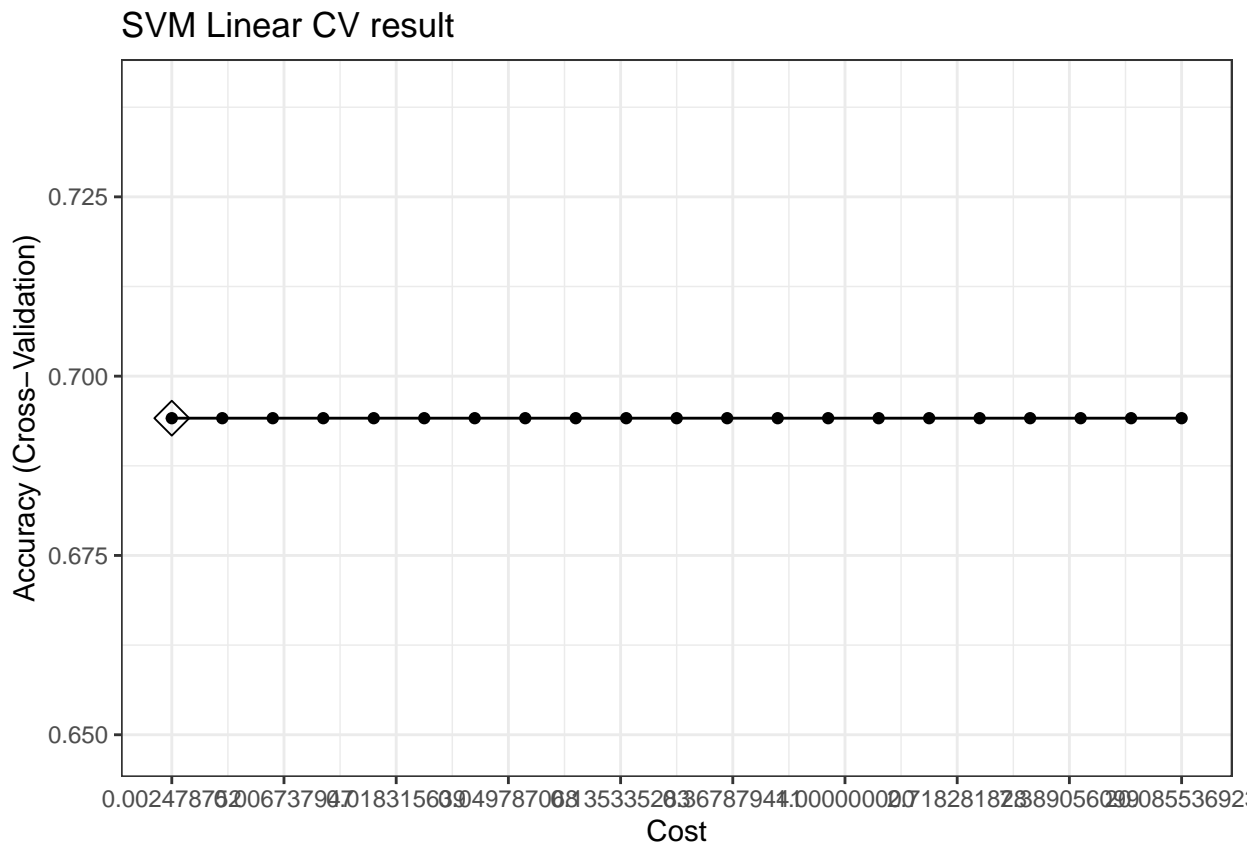


```
jpeg("./figure/rpart2.jpeg", width = 8, height = 6, units="in", res=500)
rpart.plot(rpart.fit2$finalModel)
dev.off()
```

```
## pdf
## 2
```

1.1.12 Support Vector Machine (SVM)

```
set.seed(2023)
svml.fit <- train(train.x,
                  train.bin.y,
                  method = "svmLinear",
                  tuneGrid = data.frame(C = exp(seq(-6, 3, len = 21))),
                  trControl = ctrl1)
ggplot(svml.fit, highlight = TRUE) +
  scale_x_continuous(trans='log', n.breaks = 10) +
  labs(title = "SVM Linear CV result") +
  theme_bw()
```



```
svmr.grid <- expand.grid(C = exp(seq(-2, 5, len = 20)),
                        sigma = exp(seq(-4, 1, len = 6)))
```

```
set.seed(2023)
svmr.fit <- train(train.x,
                  train.bin.y,
                  method = "svmRadialSigma",
```

```

tuneGrid = svmr.grid,
trControl = ctrl1)

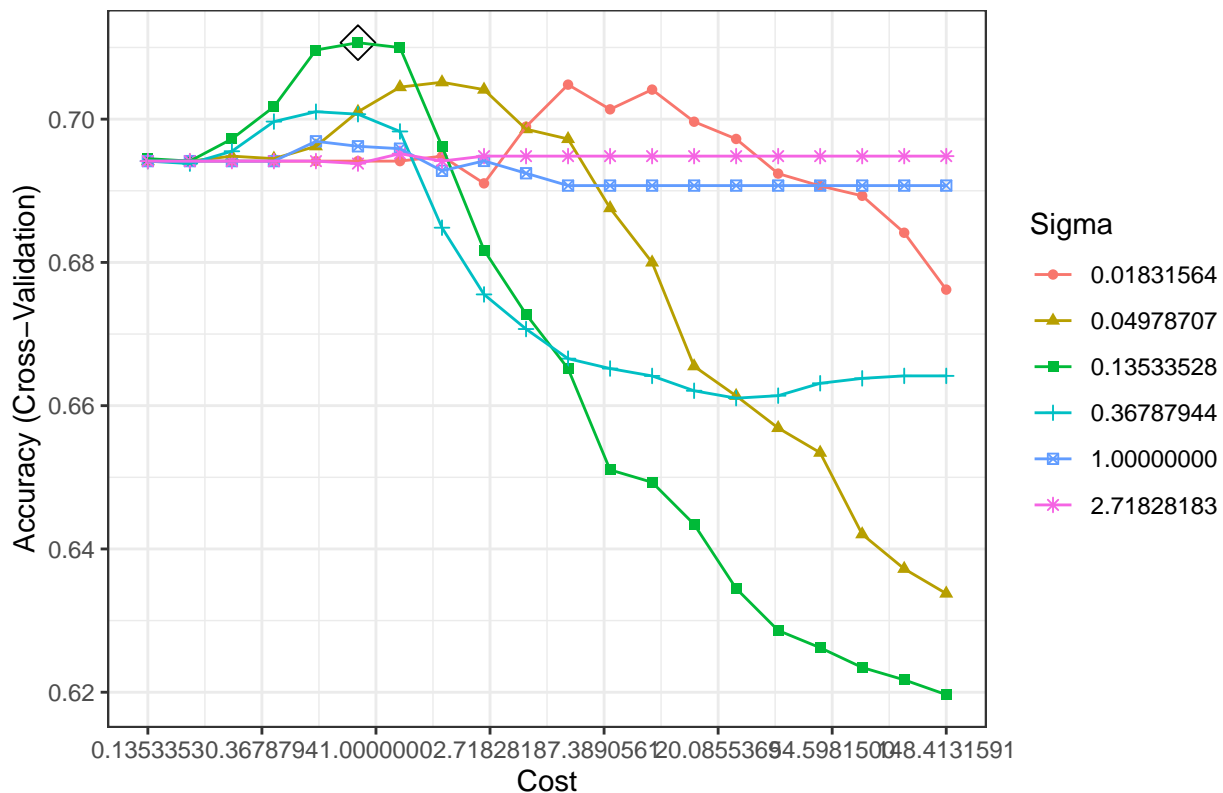
svmr.fit$bestTune

##      sigma      C
## 33 0.1353353 0.8539397

myCol<- rainbow(25)
myPar <- list(superpose.symbol = list(col = myCol),
              superpose.line = list(col = myCol))
ggplot(svmr.fit, highlight = TRUE, par.settings = myPar) +
  scale_x_continuous(trans='log',n.breaks = 10) +
  labs(title = "SVM Radial Kernel CV result") +
  theme_bw()

```

SVM Radial Kernel CV result



```

ggsave("./figure/svmr_cv.jpeg", dpi = 500)

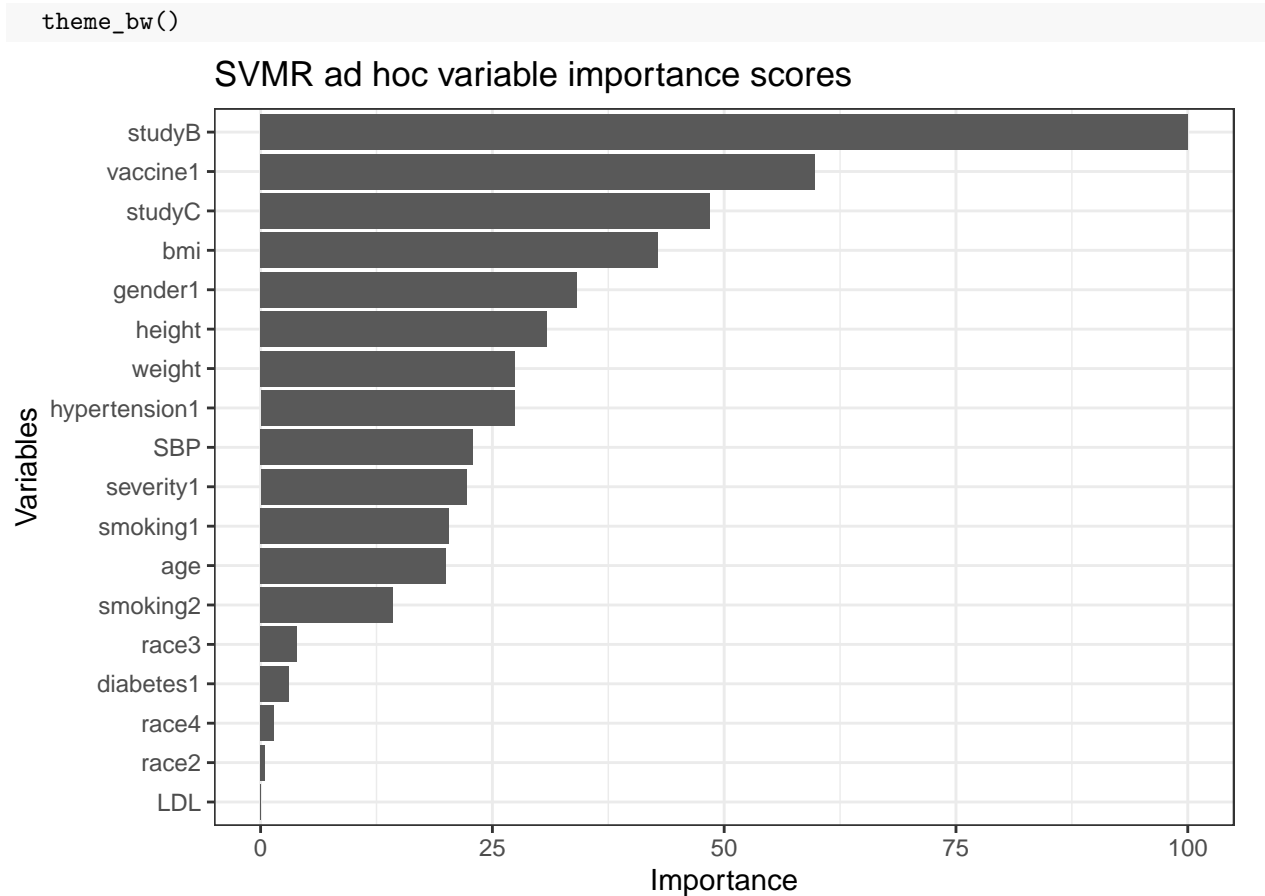
importance <- varImp(svmr.fit)$importance

vip_data <- data.frame(Variables = row.names(importance), Importance = importance$lt30)

vip_data <- vip_data[order(vip_data$Importance, decreasing = TRUE), ]

ggplot(vip_data, aes(x = Importance, y = reorder(Variables, Importance))) +
  geom_bar(stat = "identity") +
  labs(x = "Importance", y = "Variables", title = "SVMR ad hoc variable importance scores") +

```



```
ggsave("./figure/svmr_vip.jpeg", dpi = 500)

svmr.fit2 = svm(recovery_time ~ .,
               data = train.bin.dat.matrix,
               kernel = "radial",
               cost = svmr.fit$bestTune$C,
               gamma = svmr.fit$bestTune$sigma,
               probability = TRUE)
summary(svmr.fit2)

##
## Call:
## svm(formula = recovery_time ~ ., data = train.bin.dat.matrix, kernel = "radial",
##      cost = svmr.fit$bestTune$C, gamma = svmr.fit$bestTune$sigma,
##      probability = TRUE)
##
##
## Parameters:
##   SVM-Type:  C-classification
##   SVM-Kernel: radial
##      cost:  0.8539397
##
## Number of Support Vectors:  2247
##
## ( 1378 869 )
```

```
##
##
## Number of Classes: 2
##
## Levels:
## 1t30 gt30

pdp.studyB <- pdp::partial(svmr.fit2, pred.var = c("studyB"), grid.resolution = 10) %>% autoplot() +
  theme_bw()

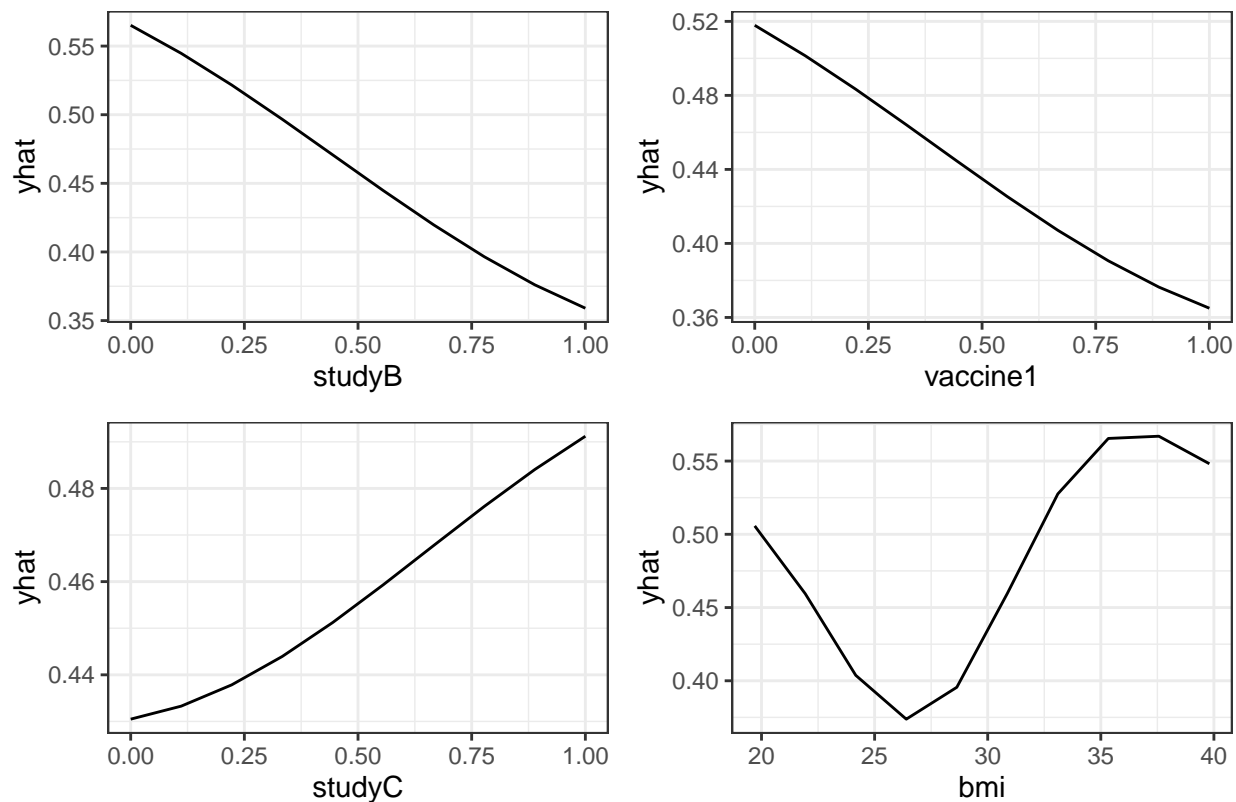
pdp.vaccine1 <- pdp::partial(svmr.fit2, pred.var = c("vaccine1"), grid.resolution = 10) %>% autoplot() +
  theme_bw()

pdp.studyC <- pdp::partial(svmr.fit2, pred.var = c("studyC"), grid.resolution = 10) %>% autoplot() +
  theme_bw()

pdp.bmi <- pdp::partial(svmr.fit2, pred.var = c("bmi"), grid.resolution = 10) %>% autoplot() +
  theme_bw()

gridExtra::grid.arrange(pdp.studyB, pdp.vaccine1, pdp.studyC, pdp.bmi, ncol = 2, top = "Partial Dependence
```

Partial Dependence Plots of SVMR Model



```
jpeg("./figure/svmr_pdp.jpeg", width = 8, height=6, units="in", res=500)
gridExtra::grid.arrange(pdp.studyB, pdp.vaccine1, pdp.studyC, pdp.bmi, ncol = 2, top = "Partial Dependence",
  dev.off())
```

```
## pdf
## 2
```

1.2 Model Selection

```

set.seed(2023)
resamp2 <- resamples(list(glm = glm.fit,
                          glmnet = glmn.fit,
                          gam = gam.bin.fit,
                          mars = mars.bin.fit,
                          lda = lda.fit,
                          qda = qda.fit,
                          nb = nb.fit,
                          bagging = bag.fit2,
                          rf = rf.fit2,
                          boosting = bst.fit2,
                          tree = rpart.fit2,
                          svm1 = svm1.fit,
                          svmr = svmr.fit))

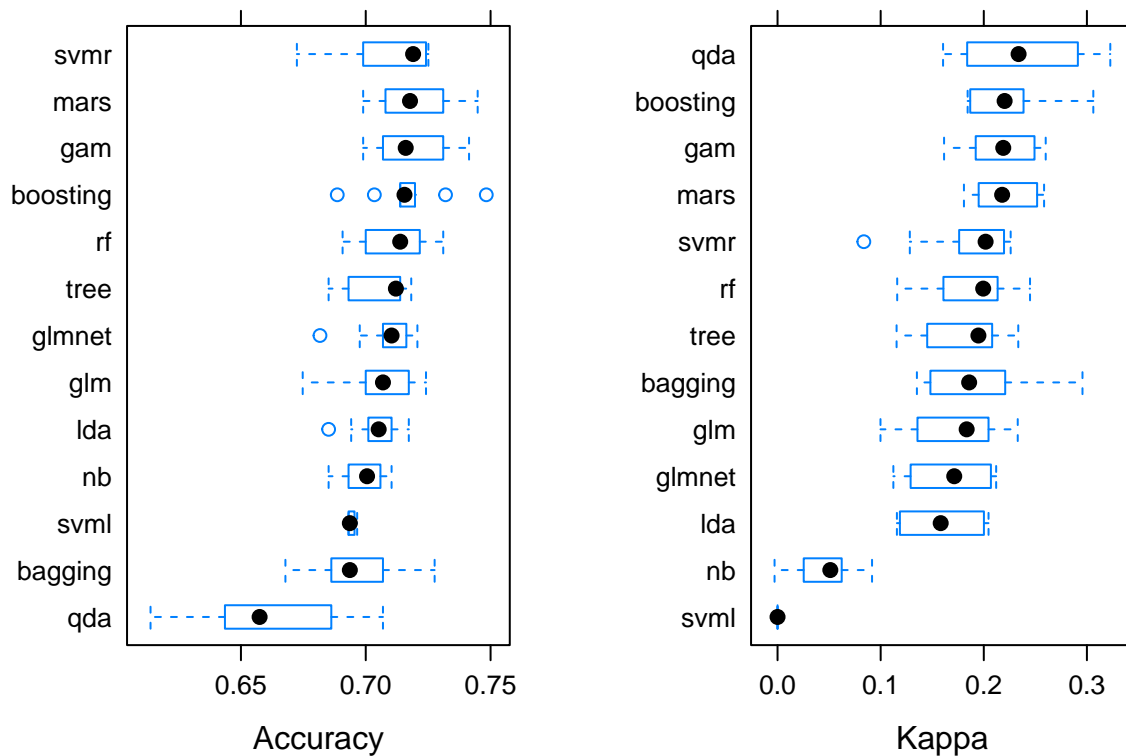
summary(resamp2)

##
## Call:
## summary.resamples(object = resamp2)
##
## Models: glm, glmnet, gam, mars, lda, qda, nb, bagging, rf, boosting, tree, svm1, svmr
## Number of resamples: 10
##
## Accuracy
##           Min.   1st Qu.   Median     Mean   3rd Qu.     Max. NA's
## glm      0.6747405 0.7008621 0.7068966 0.7062019 0.7157661 0.7241379    0
## glmnet   0.6816609 0.7071484 0.7103448 0.7082733 0.7156455 0.7206897    0
## gam      0.6989619 0.7075081 0.7160031 0.7179273 0.7301724 0.7413793    0
## mars     0.6989619 0.7085140 0.7177272 0.7193030 0.7293103 0.7448276    0
## lda      0.6851211 0.7016353 0.7051724 0.7041377 0.7100943 0.7172414    0
## qda      0.6137931 0.6456300 0.6574831 0.6617249 0.6830401 0.7068966    0
## nb       0.6851211 0.6948276 0.7005155 0.6999997 0.7052738 0.7103448    0
## bagging  0.6678201 0.6868005 0.6936308 0.6958570 0.7060345 0.7275862    0
## rf       0.6907216 0.7025862 0.7137931 0.7117251 0.7214095 0.7310345    0
## boosting 0.6885813 0.7137931 0.7155172 0.7168821 0.7193457 0.7482759    0
## tree     0.6851211 0.6939655 0.7120715 0.7051615 0.7137931 0.7182131    0
## svm1     0.6931034 0.6931034 0.6936308 0.6941389 0.6951658 0.6965517    0
## svmr     0.6724138 0.7000835 0.7189655 0.7106728 0.7235158 0.7250859    0
##
## Kappa
##           Min.   1st Qu.   Median     Mean   3rd Qu.     Max.
## glm      0.099807807 0.13883240 0.18339368 0.17256251 0.20348169 0.23291936
## glmnet   0.112313034 0.13880153 0.17136252 0.16787994 0.19888520 0.21200875
## gam      0.161494386 0.19398754 0.21887062 0.21564943 0.24436296 0.26005307
## mars     0.180770377 0.19596335 0.21778595 0.22123773 0.25078783 0.25846579
## lda      0.115743112 0.12766456 0.15822405 0.15973057 0.19212331 0.20459740
## qda      0.160506591 0.18959640 0.23375702 0.23794797 0.28596378 0.32268598
## nb       -0.002898219 0.02675247 0.05118034 0.04369503 0.06220707 0.09165486
## bagging  0.135151078 0.15295507 0.18580343 0.19152328 0.21778297 0.29572702
## rf       0.116038882 0.16147317 0.19942793 0.19013744 0.21328968 0.24484209
## boosting 0.184344290 0.19435635 0.22020503 0.22719300 0.23638392 0.30622010

```

```
## tree      0.115434149 0.15719160 0.19478063 0.18525932 0.20594295 0.23344205
## svm1      0.000000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## svmr      0.083682565 0.17761275 0.20175711 0.18642362 0.21956318 0.22606383
##          NA's
## glm       0
## glmnet    0
## gam       0
## mars      0
## lda       0
## qda       0
## nb        0
## bagging   0
## rf        0
## boosting  0
## tree      0
## svm1      0
## svmr      0
```

```
p1=bwplot(resamp2, metric = "Accuracy")
p2=bwplot(resamp2, metric = "Kappa")
grid.arrange(p1, p2, ncol=2)
```



```
jpeg("./figure/resample2.jpeg", width = 8, height=6, units="in", res=500)
p1=bwplot(resamp2, metric = "Accuracy")
p2=bwplot(resamp2, metric = "Kappa")
grid.arrange(p1, p2, ncol=2)
dev.off()
```

```
## pdf
## 2
```


1.3 Training / Testing Error

```
# svmr error
# training
pred.svmr.train <- predict(svmr.fit, newdata = train.x)
confusionMatrix(data = pred.svmr.train, reference = train.bin.y)
```

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction lt30 gt30
##      lt30  373   90
##      gt30  514 1923
##
##              Accuracy : 0.7917
##              95% CI : (0.7765, 0.8064)
##      No Information Rate : 0.6941
##      P-Value [Acc > NIR] : < 2.2e-16
##
##              Kappa : 0.4338
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##      Sensitivity : 0.4205
##      Specificity : 0.9553
##      Pos Pred Value : 0.8056
##      Neg Pred Value : 0.7891
##      Prevalence : 0.3059
##      Detection Rate : 0.1286
##      Detection Prevalence : 0.1597
##      Balanced Accuracy : 0.6879
##
##      'Positive' Class : lt30
##
```

```
# test
pred.svmr.test <- predict(svmr.fit, newdata = test.x)
confusionMatrix(data = pred.svmr.test, reference = test.bin.y)
```

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction lt30 gt30
##      lt30   62   39
##      gt30  153  469
##
##              Accuracy : 0.7344
##              95% CI : (0.7006, 0.7663)
##      No Information Rate : 0.7026
##      P-Value [Acc > NIR] : 0.03255
##
##              Kappa : 0.2498
##
##  Mcnemar's Test P-Value : 3.49e-16
##
```

```
##          Sensitivity : 0.28837
##          Specificity : 0.92323
##          Pos Pred Value : 0.61386
##          Neg Pred Value : 0.75402
##          Prevalence : 0.29737
##          Detection Rate : 0.08575
##          Detection Prevalence : 0.13970
##          Balanced Accuracy : 0.60580
##
##          'Positive' Class : 1t30
##
```