P8106\_final\_sd3731

Shuchen Dong

library(tidyverse)  
library(ggplot2)  
library(lattice)  
library(GGally)  
library(summarytools)  
library(corrplot)  
library(caret)  
library(vip)  
library(rpart.plot)  
library(ranger)  
library(gridExtra)  
library(e1071)  
library(pROC)

# import data  
load("./severity\_training.RData")  
load("./severity\_test.RData")

# train data  
trainData = training\_data |>   
 select(-id) |>   
 janitor::clean\_names() |>   
 mutate(  
 gender = factor(gender,levels = c("0","1"), labels = c("Female", "Male")),  
 race = factor(race,levels = c("1","2","3","4"), labels = c("White", "Asian","Black","Hispanic")),  
 smoking = factor(smoking,levels = c("0","1","2"), labels = c("Never\_smoked", "Former\_smoker", "Current\_smoker")),  
 hypertension = factor(hypertension,levels = c("0", "1"), labels = c("No", "Yes")),  
 diabetes = factor(diabetes,levels = c("0", "1"), labels = c("No", "Yes")),  
 vaccine = factor(vaccine,levels = c("0", "1"), labels = c("Not\_vaccinated", "Vaccinated")),  
 severity = factor(severity,levels = c("0", "1"), labels = c("not\_severe", "severe"))  
 )  
  
  
# test data  
testData = test\_data |>   
 select(-id) |>   
 janitor::clean\_names() |>  
 mutate(  
 gender = factor(gender,levels = c("0","1"), labels = c("Female", "Male")),  
 race = factor(race,levels = c("1","2","3","4"), labels = c("White", "Asian","Black","Hispanic")),  
 smoking = factor(smoking,levels = c("0","1","2"), labels = c("Never\_smoked", "Former\_smoker", "Current\_smoker")),  
 hypertension = factor(hypertension,levels = c("0", "1"), labels = c("No", "Yes")),  
 diabetes = factor(diabetes,levels = c("0", "1"), labels = c("No", "Yes")),  
 vaccine = factor(vaccine,levels = c("0", "1"), labels = c("Not\_vaccinated", "Vaccinated")),  
 severity = factor(severity,levels = c("0", "1"), labels = c("not\_severe", "severe"))  
 )

# summary  
skimr::skim(trainData)

Data summary

|  |  |
| --- | --- |
| Name | trainData |
| Number of rows | 800 |
| Number of columns | 14 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Column type frequency: |  |
| factor | 7 |
| numeric | 7 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Group variables | None |

**Variable type: factor**

| skim\_variable | n\_missing | complete\_rate | ordered | n\_unique | top\_counts |
| --- | --- | --- | --- | --- | --- |
| gender | 0 | 1 | FALSE | 2 | Fem: 410, Mal: 390 |
| race | 0 | 1 | FALSE | 4 | Whi: 521, Bla: 149, His: 80, Asi: 50 |
| smoking | 0 | 1 | FALSE | 3 | Nev: 467, For: 248, Cur: 85 |
| diabetes | 0 | 1 | FALSE | 2 | No: 679, Yes: 121 |
| hypertension | 0 | 1 | FALSE | 2 | No: 432, Yes: 368 |
| vaccine | 0 | 1 | FALSE | 2 | Vac: 464, Not: 336 |
| severity | 0 | 1 | FALSE | 2 | not: 514, sev: 286 |

**Variable type: numeric**

| skim\_variable | n\_missing | complete\_rate | mean | sd | p0 | p25 | p50 | p75 | p100 | hist |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| age | 0 | 1 | 60.03 | 4.30 | 46.0 | 57.00 | 60.0 | 63.0 | 72.0 | ▁▃▇▆▁ |
| height | 0 | 1 | 170.00 | 6.09 | 150.2 | 165.70 | 170.0 | 174.1 | 190.3 | ▁▅▇▃▁ |
| weight | 0 | 1 | 79.42 | 7.26 | 56.6 | 74.38 | 79.3 | 84.4 | 104.8 | ▁▅▇▃▁ |
| bmi | 0 | 1 | 27.54 | 2.74 | 19.6 | 25.78 | 27.6 | 29.1 | 37.4 | ▁▆▇▂▁ |
| sbp | 0 | 1 | 129.85 | 7.97 | 109.0 | 124.00 | 130.0 | 135.0 | 154.0 | ▁▆▇▃▁ |
| ldl | 0 | 1 | 110.25 | 20.05 | 41.0 | 98.00 | 111.0 | 123.0 | 174.0 | ▁▃▇▅▁ |
| depression | 0 | 1 | 6.91 | 2.12 | 0.0 | 5.00 | 7.0 | 8.0 | 13.0 | ▁▅▇▇▁ |

skimr::skim(testData)

Data summary

|  |  |
| --- | --- |
| Name | testData |
| Number of rows | 200 |
| Number of columns | 14 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Column type frequency: |  |
| factor | 7 |
| numeric | 7 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Group variables | None |

**Variable type: factor**

| skim\_variable | n\_missing | complete\_rate | ordered | n\_unique | top\_counts |
| --- | --- | --- | --- | --- | --- |
| gender | 0 | 1 | FALSE | 2 | Fem: 112, Mal: 88 |
| race | 0 | 1 | FALSE | 4 | Whi: 135, Bla: 35, His: 16, Asi: 14 |
| smoking | 0 | 1 | FALSE | 3 | Nev: 117, For: 65, Cur: 18 |
| diabetes | 0 | 1 | FALSE | 2 | No: 176, Yes: 24 |
| hypertension | 0 | 1 | FALSE | 2 | No: 104, Yes: 96 |
| vaccine | 0 | 1 | FALSE | 2 | Vac: 127, Not: 73 |
| severity | 0 | 1 | FALSE | 2 | not: 135, sev: 65 |

**Variable type: numeric**

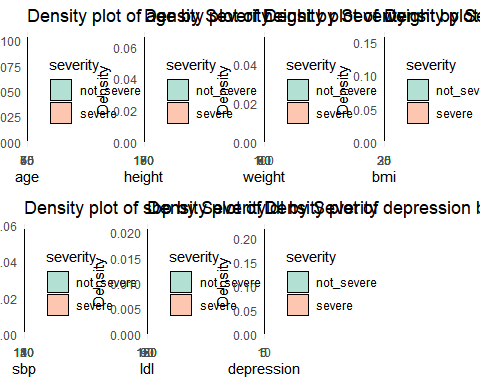
| skim\_variable | n\_missing | complete\_rate | mean | sd | p0 | p25 | p50 | p75 | p100 | hist |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| age | 0 | 1 | 60.24 | 4.18 | 49.0 | 58.00 | 60.00 | 63.00 | 71.0 | ▁▃▇▃▁ |
| height | 0 | 1 | 169.63 | 6.15 | 152.0 | 166.00 | 169.65 | 174.12 | 188.1 | ▁▅▇▅▁ |
| weight | 0 | 1 | 79.51 | 6.51 | 61.5 | 74.80 | 79.20 | 84.15 | 96.3 | ▁▅▇▅▂ |
| bmi | 0 | 1 | 27.72 | 2.72 | 20.4 | 26.05 | 27.50 | 29.70 | 35.3 | ▁▅▇▅▁ |
| sbp | 0 | 1 | 130.01 | 7.49 | 108.0 | 125.00 | 130.00 | 135.00 | 148.0 | ▁▃▇▆▂ |
| ldl | 0 | 1 | 111.30 | 18.45 | 70.0 | 98.75 | 111.50 | 124.00 | 165.0 | ▃▆▇▃▁ |
| depression | 0 | 1 | 6.72 | 2.21 | 2.0 | 5.00 | 7.00 | 8.00 | 12.0 | ▅▇▇▅▁ |

## Exploratory analysis and data visualization

con\_var = c("age", "height", "weight", "bmi", "sbp", "ldl", "depression")  
fac\_var = c("gender", "race", "smoking", "diabetes", "hypertension", "vaccine", "severity")

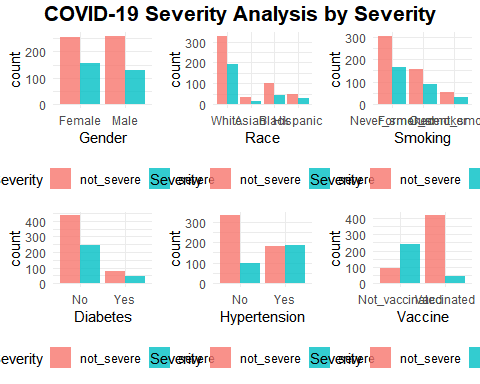
### continuous variable

plot\_con\_severity = lapply(con\_var, function(var) {  
 ggplot(trainData, aes\_string(x = var, fill = "severity")) +  
 geom\_density(alpha = 0.5) +  
 labs(title = paste("Density plot of", var, "by Severity"), x = var, y = "Density") +  
 scale\_fill\_manual(values = c("not\_severe" = "#66C2A5", "severe" = "#FC8D62")) +  
 theme\_minimal()  
})  
  
# Display all plots in a grid  
gridExtra::grid.arrange(grobs = plot\_con\_severity, nrow = 2, ncol = 4)



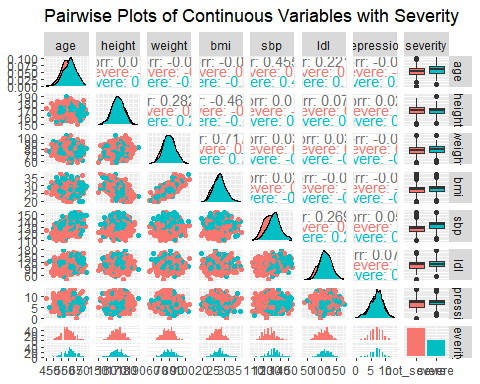
### categorical variable

# Bar Chart  
gender\_bar = trainData |>   
 ggplot(aes(x = gender, fill = severity)) +  
 geom\_bar(stat = "count",  
 position = "dodge",  
 alpha = 0.8) +  
 labs(x = "Gender", fill = "Severity") +  
 theme\_minimal() +  
 theme(legend.position = "bottom")  
  
race\_bar = trainData |>   
 ggplot(aes(x = race, fill = severity)) +  
 geom\_bar(stat = "count",  
 position = "dodge",  
 alpha = 0.8) +  
 labs(x = "Race", fill = "Severity") +  
 theme\_minimal() +  
 theme(legend.position = "bottom")  
  
smoking\_bar = trainData |>   
 ggplot(aes(x = smoking, fill = severity)) +  
 geom\_bar(stat = "count",  
 position = "dodge",  
 alpha = 0.8) +  
 labs(x = "Smoking", fill = "Severity") +  
 theme\_minimal() +  
 theme(legend.position = "bottom")  
  
diabetes\_bar = trainData |>   
 ggplot(aes(x = diabetes, fill = severity)) +  
 geom\_bar(stat = "count",  
 position = "dodge",  
 alpha = 0.8) +  
 labs(x = "Diabetes", fill = "Severity") +  
 theme\_minimal() +  
 theme(legend.position = "bottom")  
  
hypertension\_bar = trainData |>   
 ggplot(aes(x = hypertension, fill = severity)) +  
 geom\_bar(stat = "count",  
 position = "dodge",  
 alpha = 0.8) +  
 labs(x = "Hypertension", fill = "Severity") +  
 theme\_minimal() +  
 theme(legend.position = "bottom")  
  
vaccine\_bar = trainData |>   
 ggplot(aes(x = vaccine, fill = severity)) +  
 geom\_bar(stat = "count",  
 position = "dodge",  
 alpha = 0.8) +  
 labs(x = "Vaccine", fill = "Severity") +  
 theme\_minimal() +  
 theme(legend.position = "bottom")  
  
library(gridExtra)  
library(grid)  
grid.arrange(  
 arrangeGrob(  
 gender\_bar, race\_bar, smoking\_bar,   
 diabetes\_bar, hypertension\_bar, vaccine\_bar,  
 ncol = 3, nrow = 2  
 ),  
 top = textGrob("COVID-19 Severity Analysis by Severity", gp = gpar(fontsize = 16, fontface = "bold"))  
)



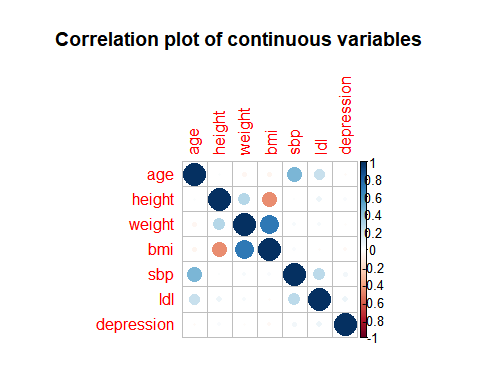
### correlation

ggpairs(trainData[, c(con\_var, "severity")],   
 mapping = aes(color = severity),  
 title = "Pairwise Plots of Continuous Variables with Severity")



ggsave("./figure/plot\_corr1.jpeg", dpi = 500)

corrplot(cor(trainData[,con\_var]), method = "circle", type = "full",   
 title = "Correlation plot of continuous variables",   
 mar = c(2, 2, 4, 2))



ggsave("./figure/plot\_corr2.jpeg", dpi = 500)

## Model training

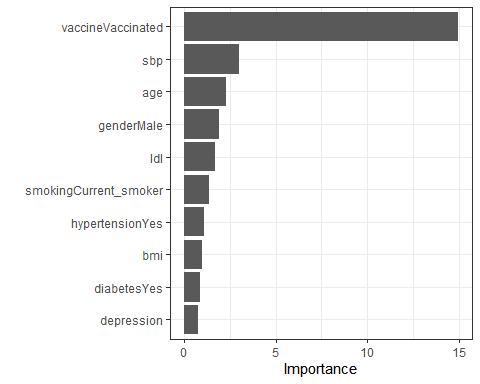
x = model.matrix(severity ~ . , trainData)[, -1]  
y = trainData[, "severity"]  
  
x2 = model.matrix(severity ~ . , testData)[, -1]  
y2 = testData$severity  
  
# cv  
set.seed(3731)  
ctrl = trainControl(method = "cv",   
 number = 10,  
 classProbs = TRUE,   
 allowParallel = TRUE,  
 summaryFunction = twoClassSummary,  
 savePredictions = "final")

### Logistic Regression

set.seed(3731)  
glm.fit = train(x,  
 y,  
 method = 'glm',  
 trControl = ctrl)  
coef(glm.fit$finalModel)

## (Intercept) age genderMale   
## -36.14267644 0.06479499 -0.40913157   
## raceAsian raceBlack raceHispanic   
## -0.20261995 0.01737165 -0.17462048   
## smokingFormer\_smoker smokingCurrent\_smoker height   
## 0.02496598 0.49239971 0.11171808   
## weight bmi diabetesYes   
## -0.13337473 0.53758507 0.25302775   
## hypertensionYes sbp ldl   
## 0.38092720 0.07081051 0.01002248   
## vaccineVaccinated depression   
## -3.61798671 -0.03796927

vip(glm.fit$finalModel) + theme\_bw()



#max(glm.fit$results$Accuracy)

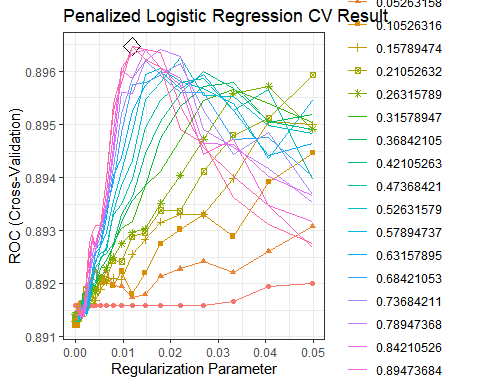
## Penalized logistic regression

Penalized logistic regression can be fitted using glmnet. We use the train function to select the optimal tuning parameters.

glmnGrid = expand.grid(.alpha = seq(0, 1, length = 20),  
 .lambda = exp(seq(-13, -3, length = 50)))  
set.seed(3731)  
glmn.fit = train(severity ~ .,  
 data = trainData,  
 method = "glmnet",  
 tuneGrid = glmnGrid,  
 trControl = ctrl)  
  
glmn.fit$bestTune

## alpha lambda  
## 993 1 0.01193152

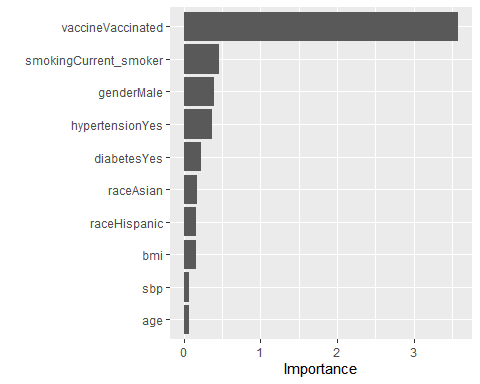
# plot  
myCol = rainbow(25)  
myPar = list(superpose.symbol = list(col = myCol),  
 superpose.line = list(col = myCol))  
ggplot(glmn.fit, highlight = TRUE) +   
 labs(title="Penalized Logistic Regression CV Result") +  
 theme\_bw()



ggsave("./figure/penal\_logi\_cv.jpeg", dpi = 500)  
  
# # Confusion matrix  
# glmn.pred.prob = predict(glmn.fit, newdata = testData, type = "prob")  
# glmn.pred = rep("not\_severe", nrow(testData))  
# glmn.pred[glmn.pred.prob[, "severe"] > 0.5] = "severe"  
#   
# confusionMatrix(data = as.factor(glmn.pred),  
# reference = y2,  
# positive = "severe")  
  
# Coefficients  
coef(glmn.fit$finalModel, glmn.fit$bestTune$lambda)

## 17 x 1 sparse Matrix of class "dgCMatrix"  
## s1  
## (Intercept) -13.748527613  
## age 0.048124759  
## genderMale -0.195662915  
## raceAsian .   
## raceBlack .   
## raceHispanic .   
## smokingFormer\_smoker .   
## smokingCurrent\_smoker 0.077493852  
## height .   
## weight .   
## bmi 0.117063103  
## diabetesYes .   
## hypertensionYes 0.290951440  
## sbp 0.061655434  
## ldl 0.004234011  
## vaccineVaccinated -3.169495661  
## depression .

vip(glmn.fit$finalModel)



### Elastic Net

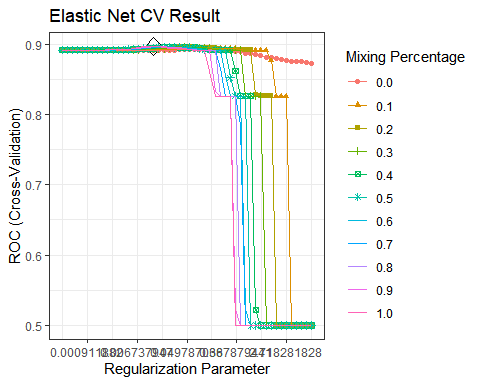
set.seed(3731)  
enet.fit = train(x,   
 y,  
 method = "glmnet",  
 tuneGrid = expand.grid(alpha = seq(0, 1, length = 11),  
 lambda = exp(seq(2,-8, length = 50))),  
 trControl = ctrl)  
  
enet.fit$bestTune

## alpha lambda  
## 519 1 0.01321331

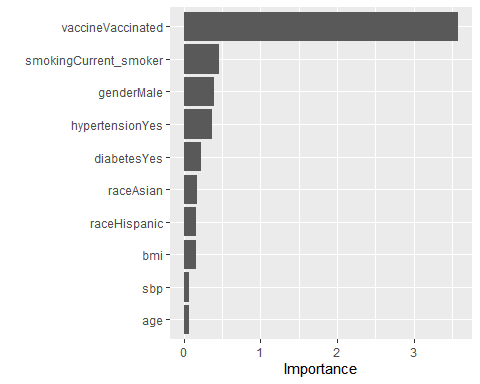
# Coefficients  
coef(enet.fit$finalModel, enet.fit$bestTune$lambda)

## 17 x 1 sparse Matrix of class "dgCMatrix"  
## s1  
## (Intercept) -13.377273440  
## age 0.046336268  
## genderMale -0.175609847  
## raceAsian .   
## raceBlack .   
## raceHispanic .   
## smokingFormer\_smoker .   
## smokingCurrent\_smoker 0.042685195  
## height .   
## weight .   
## bmi 0.112075355  
## diabetesYes .   
## hypertensionYes 0.281618275  
## sbp 0.060939569  
## ldl 0.003798565  
## vaccineVaccinated -3.133379258  
## depression .

# plot  
ggplot(enet.fit, highlight = TRUE) +   
 scale\_x\_continuous(trans='log', n.breaks = 6) +  
 labs(title ="Elastic Net CV Result") +   
 theme\_bw()



ggsave("./figure/enet\_cv.jpeg", dpi = 500)  
  
vip(enet.fit$finalModel)

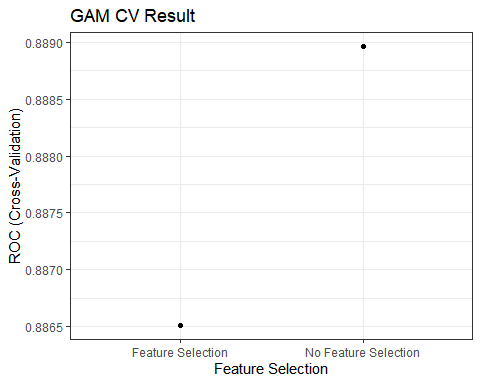


### Generalized Additive Model (GAM)

set.seed(3731)  
gam.fit = train(x,  
 y,  
 method = "gam",  
 metric = "ROC",  
 trControl = ctrl)  
gam.fit$bestTune

## select method  
## 1 FALSE GCV.Cp

ggplot(gam.fit) +  
 labs(title = "GAM CV Result") +  
 theme\_bw()



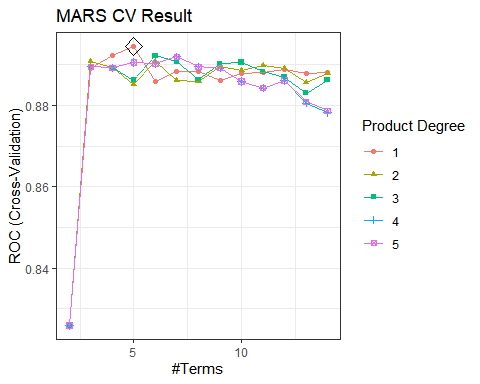
ggsave("./figure/gam\_cv.jpeg", dpi = 500)  
  
# coef(gam.fit$finalModel)  
gam.fit$finalModel

##   
## Family: binomial   
## Link function: logit   
##   
## Formula:  
## .outcome ~ genderMale + raceAsian + raceBlack + raceHispanic +   
## smokingFormer\_smoker + smokingCurrent\_smoker + diabetesYes +   
## hypertensionYes + vaccineVaccinated + s(depression) + s(age) +   
## s(sbp) + s(ldl) + s(bmi) + s(height) + s(weight)  
##   
## Estimated degrees of freedom:  
## 1.00 1.06 3.47 1.74 1.00 7.69 1.00   
## total = 26.97   
##   
## UBRE score: -0.2345662

# par(mfrow=c(2, 3))  
# plot(gam.fit$finalModel)  
# par(mfrow=c(1, 1))

### Multivariate Adaptive Regression Splines (MARS)

mars.grid = expand.grid(degree = 1:5,  
 nprune = 2:14)  
set.seed(3731)  
mars.fit = train(x,  
 y,  
 method = "earth",  
 tuneGrid = mars.grid,  
 trControl = ctrl)  
  
ggplot(mars.fit, highlight = TRUE)+   
 labs(title ="MARS CV Result") +  
 theme\_bw()



ggsave("./figure/mars\_cv.jpeg", dpi = 500)  
  
mars.fit$bestTune

## nprune degree  
## 4 5 1

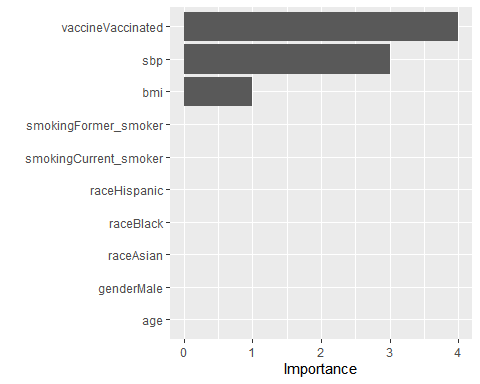
coef(mars.fit$finalModel)

## (Intercept) vaccineVaccinated h(sbp-139) h(139-sbp)   
## 1.98341761 -3.50798169 -0.01515556 -0.13557595   
## h(bmi-27)   
## 0.24293455

summary(mars.fit$finalModel)

## Call: earth(x=matrix[800,16], y=factor.object, keepxy=TRUE,  
## glm=list(family=function.object, maxit=100), degree=1, nprune=5)  
##   
## GLM coefficients  
## severe  
## (Intercept) 1.9834176  
## vaccineVaccinated -3.5079817  
## h(bmi-27) 0.2429345  
## h(139-sbp) -0.1355759  
## h(sbp-139) -0.0151556  
##   
## GLM (family binomial, link logit):  
## nulldev df dev df devratio AIC iters converged  
## 1043.15 799 605.753 795 0.419 615.8 5 1  
##   
## Earth selected 5 of 24 terms, and 3 of 16 predictors (nprune=5)  
## Termination condition: Reached nk 33  
## Importance: vaccineVaccinated, sbp, bmi, age-unused, genderMale-unused, ...  
## Number of terms at each degree of interaction: 1 4 (additive model)  
## Earth GCV 0.1246822 RSS 97.51412 GRSq 0.4585368 RSq 0.4693254

vip(mars.fit$finalModel)



### Linear Discriminant Analysis (LDA)

set.seed(3731)  
lda.fit = train(x,  
 y,  
 method = "lda",  
 metric = "ROC",  
 trControl = ctrl)

### Quadratic Discriminant Analysis (QDA)

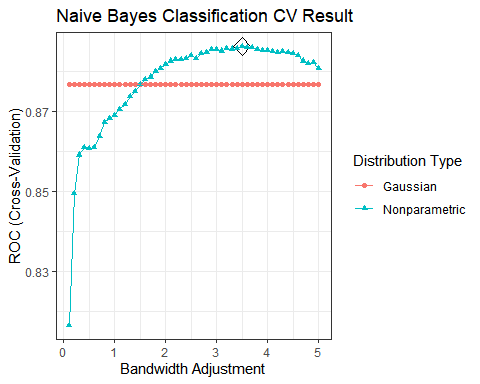
set.seed(3731)  
qda.fit = train(x,  
 y,  
 method = "qda",  
 metric = "ROC",  
 trControl = ctrl)

### Naive Bayes (NB)

nbGrid = expand.grid(usekernel = c(FALSE,TRUE),  
 fL = 1,  
 adjust = seq(0.1, 5, by = .1))  
set.seed(3731)  
nb.fit = train(x,  
 y,  
 method = "nb",  
 tuneGrid = nbGrid,  
 metric = "ROC",  
 trControl = ctrl)  
nb.fit$bestTune

## fL usekernel adjust  
## 85 1 TRUE 3.5

ggplot(nb.fit, highlight = TRUE) +   
 labs(title ="Naive Bayes Classification CV Result") +  
 theme\_bw()



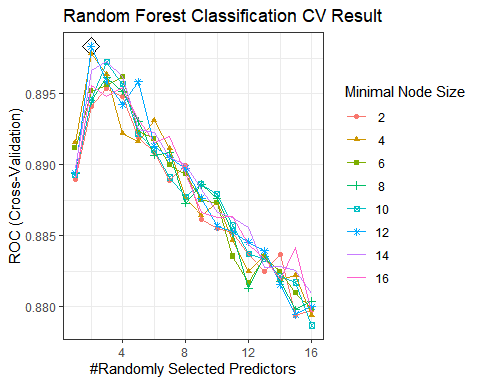
ggsave("./figure/nb\_cv.jpeg", dpi = 500)

### Random Forest

rf.grid2 = expand.grid(mtry = 1:ncol(x),  
 splitrule = "gini",  
 min.node.size = seq(from = 2, to = 16, by = 2))  
set.seed(3731)  
rf.fit2 = train(x,   
 y,  
 method = "ranger",  
 tuneGrid = rf.grid2,  
 trControl = ctrl)  
  
rf.fit2$bestTune

## mtry splitrule min.node.size  
## 14 2 gini 12

ggplot(rf.fit2, highlight = TRUE) +   
 labs(title = "Random Forest Classification CV Result") +   
 theme\_bw()



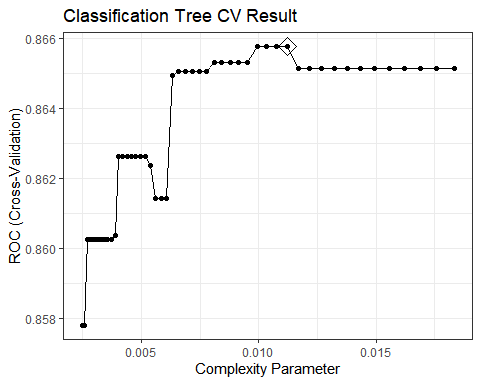
ggsave("./figure/rf\_classification\_cv.jpeg", dpi = 500)

### Classification Trees

rpart.grid = expand.grid(cp = exp(seq(-6,-4, len = 50)))  
set.seed(3731)  
rpart.fit = train(x,  
 y,  
 method = "rpart",  
 tuneGrid = rpart.grid,  
 trControl = ctrl)  
  
rpart.fit$bestTune

## cp  
## 38 0.01122293

ggplot(rpart.fit, highlight = TRUE) +  
 labs(title = "Classification Tree CV Result") +  
 theme\_bw()



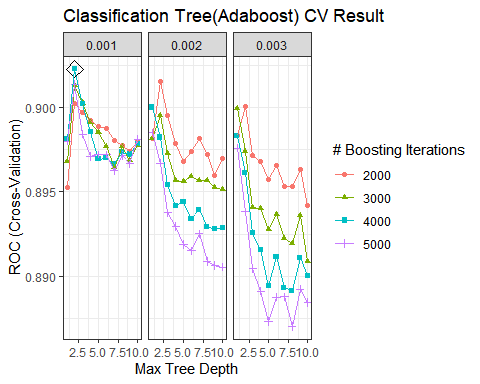
# ggsave("./figure/rpart\_cv.jpeg", dpi = 500)  
#   
# rpart.plot(rpart.fit$finalModel)  
#   
# jpeg("./figure/rpart.jpeg", width = 8, height = 6, units="in", res=500)  
# rpart.plot(rpart.fit$finalModel)  
# dev.off()

### Adaboost

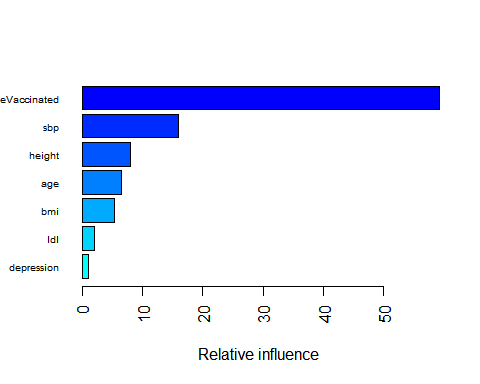
gbmA.grid = expand.grid(n.trees = c(2000, 3000, 4000, 5000),  
 interaction.depth = 1:10,  
 shrinkage = c(0.001, 0.002, 0.003),  
 n.minobsinnode = 1)  
set.seed(3731)  
gbmA.fit = train(x,  
 y,  
 method = "gbm",  
 tuneGrid = gbmA.grid,  
 trControl = ctrl,  
 distribution = "adaboost",  
 verbose = FALSE)  
gbmA.fit$bestTune

## n.trees interaction.depth shrinkage n.minobsinnode  
## 7 4000 2 0.001 1

# plot  
ggplot(gbmA.fit, highlight = TRUE) +  
 labs(title = "Classification Tree(Adaboost) CV Result") +   
 theme\_bw()



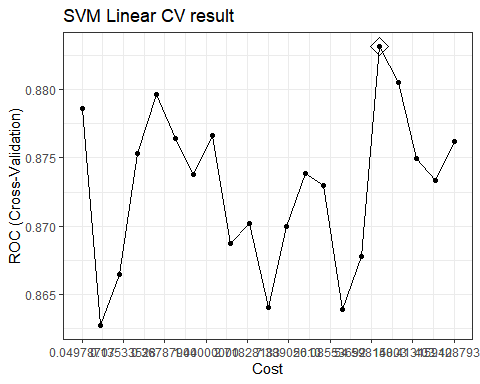
ggsave("./figure/gbmA\_cv.jpeg", dpi = 500)  
  
# Variable importance  
summary(gbmA.fit$finalModel, las = 2, cBars = 7, cex.names = 0.6)



## var rel.inf  
## vaccineVaccinated vaccineVaccinated 59.244643505  
## sbp sbp 15.984611726  
## height height 7.934349106  
## age age 6.450499807  
## bmi bmi 5.273480867  
## ldl ldl 1.994823637  
## depression depression 1.067582456  
## weight weight 0.970517584  
## hypertensionYes hypertensionYes 0.514722903  
## genderMale genderMale 0.276478309  
## smokingCurrent\_smoker smokingCurrent\_smoker 0.258412504  
## raceAsian raceAsian 0.010638798  
## diabetesYes diabetesYes 0.008363645  
## smokingFormer\_smoker smokingFormer\_smoker 0.008324171  
## raceHispanic raceHispanic 0.002550982  
## raceBlack raceBlack 0.000000000

### Support Vector Machine (SVM)

set.seed(3731)  
svml.fit = train(x,  
 y,  
 method = "svmLinear",  
 tuneGrid = data.frame(C = exp(seq(-3, 6, len = 21))),  
 trControl = ctrl)  
  
ggplot(svml.fit, highlight = TRUE) +   
 scale\_x\_continuous(trans='log',n.breaks = 10) +  
 labs(title = "SVM Linear CV result") +   
 theme\_bw()



ggsave("./figure/svml\_cv.jpeg", dpi = 500)

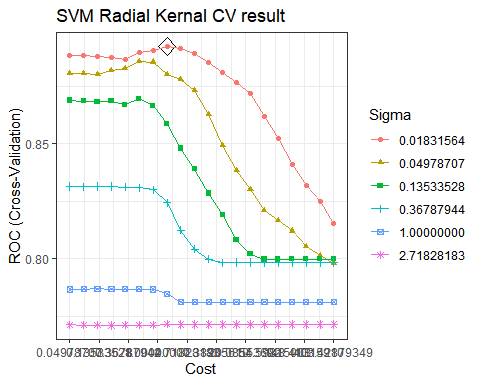
svmr.grid = expand.grid(C = exp(seq(-3, 6, len = 20)),  
 sigma = exp(seq(-4, 1, len = 6)))  
  
set.seed(3731)  
svmr.fit = train(x,  
 y,  
 method = "svmRadialSigma",  
 tuneGrid = svmr.grid,  
 trControl = ctrl)

## maximum number of iterations reached 0.001555028 0.001526522maximum number of iterations reached 0.0001049961 0.0001003757maximum number of iterations reached 0.001565161 0.00152748maximum number of iterations reached 0.0001555562 0.0001484596maximum number of iterations reached 0.001624193 0.00158779maximum number of iterations reached 0.001008161 0.0009672123maximum number of iterations reached 9.968407e-06 9.252759e-06maximum number of iterations reached 0.002522674 0.002475664maximum number of iterations reached 5.77933e-05 5.52207e-05maximum number of iterations reached 0.001071575 0.001035467maximum number of iterations reached 0.0002230658 0.0002129264maximum number of iterations reached 0.0005720601 0.0005568371maximum number of iterations reached 7.043732e-05 6.734821e-05maximum number of iterations reached 0.001051966 0.00102525maximum number of iterations reached 0.001943041 0.001873563maximum number of iterations reached 0.002437711 0.002383565maximum number of iterations reached 0.0001147904 0.0001097216maximum number of iterations reached 0.001801172 0.0017537maximum number of iterations reached 0.0006263894 0.0005984378maximum number of iterations reached 0.0002870176 0.000267116maximum number of iterations reached 0.002676939 0.00261618maximum number of iterations reached 0.0004819192 0.0004611776maximum number of iterations reached 1.891896e-05 1.755898e-05

svmr.fit$bestTune

## sigma C  
## 43 0.01831564 1.371342

# plot  
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 Kernal CV result") +   
 theme\_bw()



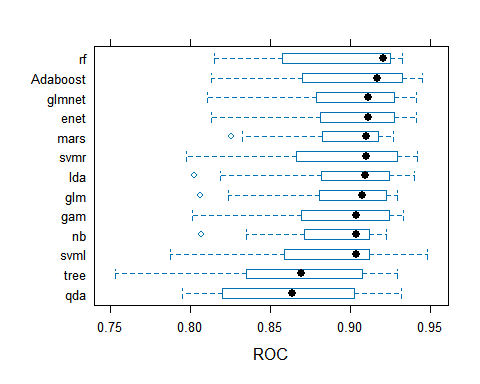
ggsave("./figure/svmr\_cv.jpeg", dpi = 500)

## Model Selection

set.seed(3731)  
resamp = resamples(list(glm = glm.fit,  
 glmnet = glmn.fit,  
 enet = enet.fit,  
 gam = gam.fit,  
 mars = mars.fit,  
 lda = lda.fit,  
 qda = qda.fit,  
 nb = nb.fit,  
 rf = rf.fit2,  
 tree = rpart.fit,  
 Adaboost = gbmA.fit,  
 svml = svml.fit,  
 svmr = svmr.fit))  
  
summary(resamp)

##   
## Call:  
## summary.resamples(object = resamp)  
##   
## Models: glm, glmnet, enet, gam, mars, lda, qda, nb, rf, tree, Adaboost, svml, svmr   
## Number of resamples: 10   
##   
## ROC   
## Min. 1st Qu. Median Mean 3rd Qu. Max. NA's  
## glm 0.8060224 0.8812879 0.9072129 0.8906537 0.9220274 0.9290061 0  
## glmnet 0.8109244 0.8846213 0.9112446 0.8964676 0.9254202 0.9409814 0  
## enet 0.8130252 0.8865160 0.9109195 0.8965507 0.9254202 0.9409814 0  
## gam 0.8011204 0.8721921 0.9038824 0.8889628 0.9231874 0.9330629 0  
## mars 0.8256303 0.8868742 0.9100563 0.8944594 0.9167082 0.9267241 0  
## lda 0.8025210 0.8843786 0.9088911 0.8929675 0.9238808 0.9396552 0  
## qda 0.7948179 0.8237412 0.8637593 0.8613904 0.8976164 0.9316976 0  
## nb 0.8067227 0.8739816 0.9038462 0.8862909 0.9111610 0.9222448 0  
## rf 0.8151261 0.8692951 0.9201681 0.8983231 0.9247347 0.9323867 0  
## tree 0.7535497 0.8387100 0.8693634 0.8657576 0.9045346 0.9290451 0  
## Adaboost 0.8130252 0.8804103 0.9166119 0.9022879 0.9320292 0.9449602 0  
## svml 0.7876944 0.8673622 0.9035666 0.8831536 0.9103641 0.9476127 0  
## svmr 0.7976190 0.8741684 0.9096119 0.8924048 0.9263016 0.9416446 0  
##   
## Sens   
## Min. 1st Qu. Median Mean 3rd Qu. Max. NA's  
## glm 0.8431373 0.8486991 0.8921569 0.8811840 0.9033748 0.9230769 0  
## glmnet 0.8235294 0.8503017 0.8823529 0.8792232 0.9033748 0.9230769 0  
## enet 0.8039216 0.8503017 0.8725490 0.8753017 0.9033748 0.9230769 0  
## gam 0.8431373 0.8627451 0.8834842 0.8869910 0.9171380 0.9230769 0  
## mars 0.8235294 0.8627451 0.8823529 0.8850679 0.9171380 0.9423077 0  
## lda 0.7843137 0.8116516 0.8446456 0.8442308 0.8647247 0.9230769 0  
## qda 0.8269231 0.8438914 0.8627451 0.8638009 0.8781109 0.9230769 0  
## nb 0.9038462 0.9414593 0.9607843 0.9553167 0.9756787 0.9807692 0  
## rf 0.8846154 0.9276018 0.9411765 0.9397059 0.9607843 0.9807692 0  
## tree 0.8627451 0.8745287 0.9127074 0.9104827 0.9366516 0.9615385 0  
## Adaboost 0.8627451 0.8921569 0.9215686 0.9142911 0.9371229 0.9615385 0  
## svml 0.7647059 0.7730015 0.8137255 0.8130845 0.8461538 0.8653846 0  
## svmr 0.8627451 0.8696267 0.9019608 0.8948341 0.9033748 0.9423077 0  
##   
## Spec   
## Min. 1st Qu. Median Mean 3rd Qu. Max. NA's  
## glm 0.6551724 0.6964286 0.7931034 0.7827586 0.8620690 0.8928571 0  
## glmnet 0.6206897 0.7500000 0.8103448 0.8038177 0.8851601 0.9310345 0  
## enet 0.6206897 0.7500000 0.8103448 0.8038177 0.8851601 0.9310345 0  
## gam 0.6206897 0.6875000 0.7758621 0.7513547 0.7931034 0.8620690 0  
## mars 0.5517241 0.7321429 0.7931034 0.7726601 0.8275862 0.8928571 0  
## lda 0.6551724 0.7857143 0.8602217 0.8352217 0.8965517 0.9310345 0  
## qda 0.6206897 0.6813424 0.7543103 0.7584975 0.8411330 0.8965517 0  
## nb 0.3928571 0.4482759 0.5178571 0.5379310 0.6083744 0.7931034 0  
## rf 0.5517241 0.6160714 0.7241379 0.7094828 0.7844828 0.8928571 0  
## tree 0.5862069 0.7142857 0.7758621 0.7584975 0.8275862 0.8571429 0  
## Adaboost 0.5862069 0.6785714 0.7758621 0.7548030 0.8275862 0.8928571 0  
## svml 0.6206897 0.7857143 0.8774631 0.8386700 0.8965517 0.9655172 0  
## svmr 0.6206897 0.7500000 0.8275862 0.8038177 0.8620690 0.9310345 0

bwplot(resamp, metric = "ROC")



Because the Adaboost model shows the highest median ROC value according to the resampling outcomes reflecting our models’ performance on the training group, my choice for predicting the severity response variable would be the **Adaboost** model.

## Training / Testing Error

# Adaboost error  
# training  
pred.gbmA.train = predict(gbmA.fit, newdata = x)  
confusionMatrix(data = pred.gbmA.train, reference = y, positive = "severe")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction not\_severe severe  
## not\_severe 481 66  
## severe 33 220  
##   
## Accuracy : 0.8762   
## 95% CI : (0.8514, 0.8983)  
## No Information Rate : 0.6425   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.7235   
##   
## Mcnemar's Test P-Value : 0.001299   
##   
## Sensitivity : 0.7692   
## Specificity : 0.9358   
## Pos Pred Value : 0.8696   
## Neg Pred Value : 0.8793   
## Prevalence : 0.3575   
## Detection Rate : 0.2750   
## Detection Prevalence : 0.3162   
## Balanced Accuracy : 0.8525   
##   
## 'Positive' Class : severe   
##

##Accuracy : 0.8762; Kappa : 0.7235  
  
# test  
pred.gbmA.test = predict(gbmA.fit, newdata = x2)  
confusionMatrix(data = pred.gbmA.test, reference = y2, positive = "severe")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction not\_severe severe  
## not\_severe 126 18  
## severe 9 47  
##   
## Accuracy : 0.865   
## 95% CI : (0.8097, 0.9091)  
## No Information Rate : 0.675   
## P-Value [Acc > NIR] : 5.597e-10   
##   
## Kappa : 0.6809   
##   
## Mcnemar's Test P-Value : 0.1237   
##   
## Sensitivity : 0.7231   
## Specificity : 0.9333   
## Pos Pred Value : 0.8393   
## Neg Pred Value : 0.8750   
## Prevalence : 0.3250   
## Detection Rate : 0.2350   
## Detection Prevalence : 0.2800   
## Balanced Accuracy : 0.8282   
##   
## 'Positive' Class : severe   
##

##Accuracy : 0.865; Kappa : 0.6809

### AUC

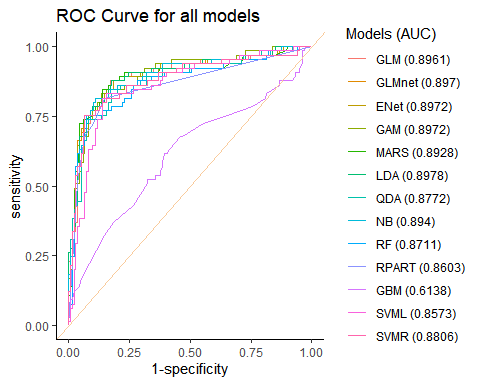
# AUC test  
glm.pred = predict(glm.fit, newdata = x2, type = "prob")[, 2]  
glmn.pred = predict(glmn.fit, newdata = testData, type = "prob")[, 2]  
enet.pred = predict(enet.fit, newdata = x2, type = "prob")[, 2]  
gam.pred = predict(gam.fit, newdata = x2, type = "prob")[, 2]  
mars.pred = predict(mars.fit, newdata = x2, type = "prob")[, 2]  
lda.pred = predict(lda.fit, newdata = x2, type = "prob")[, 2]  
qda.pred = predict(qda.fit, newdata = x2, type = "prob")[, 2]  
nb.pred = predict(nb.fit, newdata = x2, type = "prob")[, 2]  
rf.pred = predict(rf.fit2, newdata = x2, type = "prob")[, 2]  
rpart.pred = predict(rpart.fit, newdata = x2, type = "prob")[, 2]  
gbmA.pred = predict(gbmA.fit, newdata = testData, type = "prob")[, 2]  
svml.pred = predict(svml.fit, newdata = x2, type = "prob")[, 2]  
svmr.pred = predict(svmr.fit, newdata = x2, type = "prob")[, 2]  
  
  
roc.glm = roc(y2, glm.pred)  
roc.glmn = roc(y2, glmn.pred)  
roc.enet = roc(y2, enet.pred)  
roc.gam = roc(y2, gam.pred)  
roc.mars = roc(y2, mars.pred)  
roc.lda = roc(y2, lda.pred)  
roc.qda = roc(y2, qda.pred)  
roc.nb = roc(y2, nb.pred)  
roc.rf = roc(y2, rf.pred)  
roc.rpart = roc(y2, rpart.pred)  
roc.gbmA = roc(y2, gbmA.pred)  
roc.svml = roc(y2, svml.pred)  
roc.svmr = roc(y2, svmr.pred)  
  
  
auc = c(roc.glm$auc,  
 roc.glmn$auc,  
 roc.enet$auc,  
 roc.gam$auc,  
 roc.mars$auc,  
 roc.lda$auc,  
 roc.qda$auc,  
 roc.nb$auc,  
 roc.rf$auc,  
 roc.rpart$auc,  
 roc.gbmA$auc,  
 roc.svml$auc,  
 roc.svmr$auc)  
  
names(auc) = c("GLM", "GLMnet", "ENet", "GAM", "MARS", "LDA", "QDA", "NB", "RF", "RPART", "GBM", "SVML", "SVMR")  
auc

## GLM GLMnet ENet GAM MARS LDA QDA NB   
## 0.8960684 0.8969801 0.8972080 0.8972080 0.8927635 0.8977778 0.8771510 0.8940171   
## RF RPART GBM SVML SVMR   
## 0.8711111 0.8602849 0.6137892 0.8573219 0.8805698

modelNames = c("GLM", "GLMnet", "ENet", "GAM", "MARS", "LDA", "QDA", "NB", "RF", "RPART", "GBM", "SVML", "SVMR")  
# order auc  
auc\_data = data.frame(model = modelNames,  
 auc = auc)  
auc\_data[order(-auc\_data$auc), ]

## model auc  
## LDA LDA 0.8977778  
## GAM GAM 0.8972080  
## ENet ENet 0.8972080  
## GLMnet GLMnet 0.8969801  
## GLM GLM 0.8960684  
## NB NB 0.8940171  
## MARS MARS 0.8927635  
## SVMR SVMR 0.8805698  
## QDA QDA 0.8771510  
## RF RF 0.8711111  
## RPART RPART 0.8602849  
## SVML SVML 0.8573219  
## GBM GBM 0.6137892

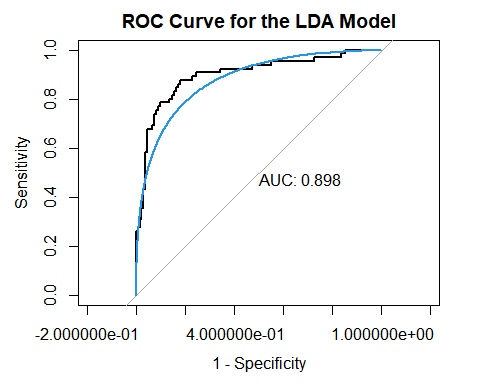
# plot auc  
ggroc(list(roc.glm, roc.glmn, roc.enet, roc.gam, roc.mars, roc.lda, roc.qda, roc.nb, roc.rf, roc.rpart, roc.gbmA, roc.svml, roc.svmr), legacy.axes = TRUE) +  
 scale\_color\_discrete(labels = paste0(modelNames, " (", round(auc, 4), ")"),  
 name = "Models (AUC)") +  
 geom\_abline(intercept = 0, slope = 1, color = "#f9cb9c") +  
 theme\_classic() +  
 labs(title = "ROC Curve for all models")



ggsave("./figure/roc\_test.jpeg", dpi = 500)

According to the model, the LDA model also has the highest auc value(**0.8977778**) among all the models. Thus, I prefer to choose the **LDA** model.

plot(roc.lda,  
 legacy.axes = TRUE,  
 print.auc = TRUE,  
 main = "ROC Curve for the LDA Model")  
plot(smooth(roc.mars), col = 4, add = TRUE)



# test data  
test.pred.prob = predict(lda.fit, newdata = x2, type = "prob")[, 2]  
test.pred = rep("not\_severe", length(test.pred.prob))  
test.pred[test.pred.prob > 0.5] = "severe"  
res = confusionMatrix(data = factor(test.pred, levels = c("not\_severe", "severe")),  
 reference = y2,  
 positive = "severe")  
res

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction not\_severe severe  
## not\_severe 115 13  
## severe 20 52  
##   
## Accuracy : 0.835   
## 95% CI : (0.7762, 0.8836)  
## No Information Rate : 0.675   
## P-Value [Acc > NIR] : 2.442e-07   
##   
## Kappa : 0.6341   
##   
## Mcnemar's Test P-Value : 0.2963   
##   
## Sensitivity : 0.8000   
## Specificity : 0.8519   
## Pos Pred Value : 0.7222   
## Neg Pred Value : 0.8984   
## Prevalence : 0.3250   
## Detection Rate : 0.2600   
## Detection Prevalence : 0.3600   
## Balanced Accuracy : 0.8259   
##   
## 'Positive' Class : severe   
##

1-0.835

## [1] 0.165

Also, the value of Accuracy is 0.835 and Kappa is 0.6341 in LDA for test data. In addition, we can obtain the misclassification error rate is 16.5% (1-0.835 = 0.165).