Final Project

Yueyi Xu

2024-05-09

```r
training_data <- training_data %>%
  janitor::clean_names() %>%
  dplyr::select(-id) %>%
  mutate(severity = case_match(as.numeric(severity),
                               1 ~ "Not Severe",
                               2 ~ "Severe"),
         severity = factor(severity),
         gender = case_match(gender,
                             1 ~ "Male",
                             0 ~ "Female"),
         race = case_match(as.numeric(race),
                           1 ~ "White",
                           2 ~ "Asian",
                           3 ~ "Black",
                           4 ~ "Hispanic"),
         smoking = case_match(as.numeric(smoking),
                              1 ~ "Never",
                              2 ~ "Former",
                              3 ~ "Current"),
         hypertension = case_match(hypertension,
                                   0 ~ "No",
                                   1 ~ "Yes"),
         diabetes = case_match(diabetes,
                               0 ~ "No",
                               1 ~ "Yes"),
         vaccine = case_match(vaccine,
                              0 ~ "Not Vaccinated",
                              1 ~ "Vaccinated")
         )
```

```r
test_data <- test_data %>%
  janitor::clean_names() %>%
  dplyr::select(-id) %>%
  mutate(severity = case_match(as.numeric(severity),
                               1 ~ "Not Severe",
                               2 ~ "Severe"),
         severity = factor(severity),
         gender = case_match(gender,
                             1 ~ "Male",
                             0 ~ "Female"),
         race = case_match(as.numeric(race),
                           1 ~ "White",
```

```r
                        2 ~ "Asian",
                        3 ~ "Black",
                        4 ~ "Hispanic"),
         smoking = case_match(as.numeric(smoking),
                           1 ~ "Never",
                           2 ~ "Former",
                           3 ~ "Current"),
         hypertension = case_match(hypertension,
                                0 ~ "No",
                                1 ~ "Yes"),
         diabetes = case_match(diabetes,
                            0 ~ "No",
                            1 ~ "Yes"),
         vaccine = case_match(vaccine,
                           0 ~ "Not Vaccinated",
                           1 ~ "Vaccinated")
         )
```

Exploratory analysis and data visualization

We will create box plots for continuous predictors such as Age, Height, Weight, BMI, Systolic blood pressure (SBP), LDL cholesterol (LDL), and Depression. These plots will show how these metrics vary with the severity of COVID-19.

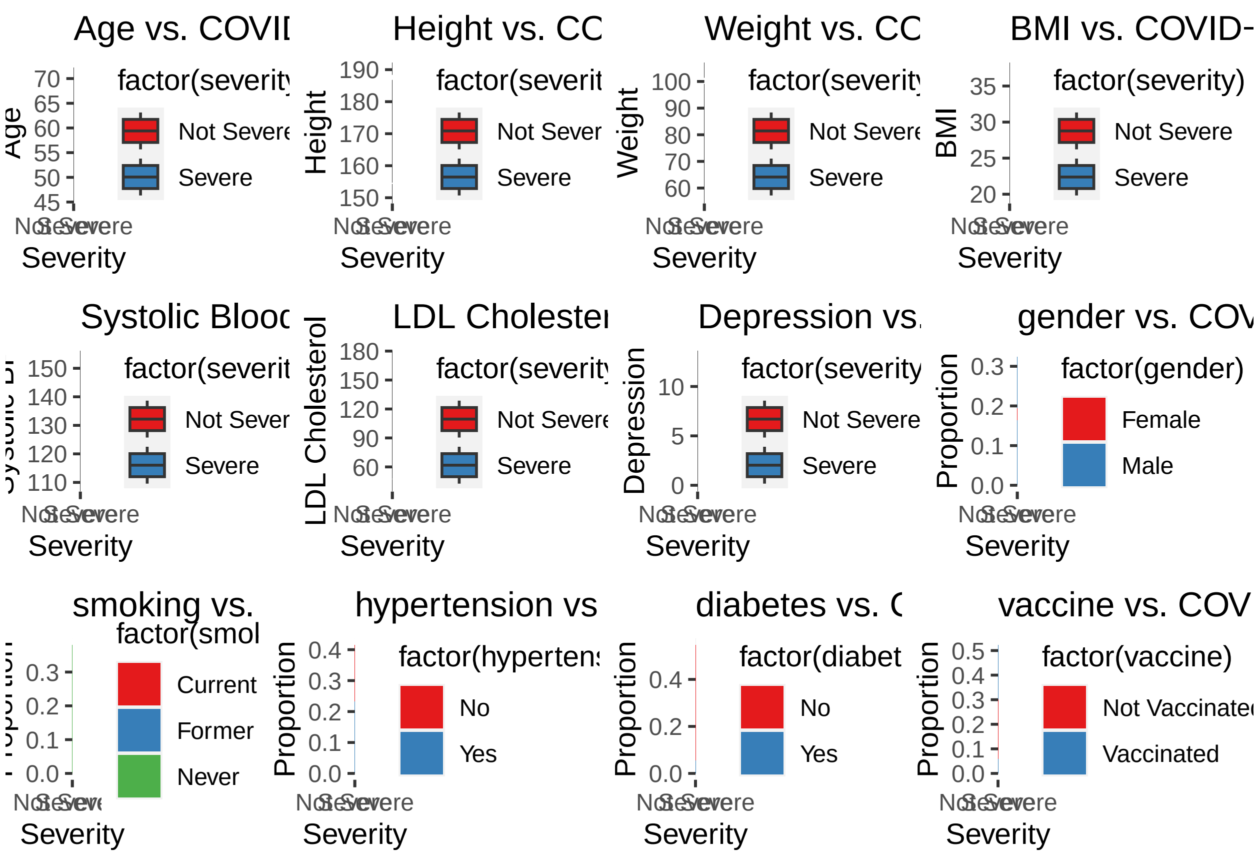
```r
# Boxplot for Age vs. Severity
p1 <- ggplot(training_data, aes(x = factor(severity), y = age, fill = factor(severity))) +
  geom_boxplot() +
  labs(title = "Age vs. COVID-19 Severity", x = "Severity", y = "Age") +
  scale_fill_brewer(palette = "Set1")

# Boxplot for Height vs. Severity
p2 <- ggplot(training_data, aes(x = factor(severity), y = height, fill = factor(severity))) +
  geom_boxplot() +
  labs(title = "Height vs. COVID-19 Severity", x = "Severity", y = "Height") +
  scale_fill_brewer(palette = "Set1")

# Boxplot for Weight vs. Severity
p3 <- ggplot(training_data, aes(x = factor(severity), y = weight, fill = factor(severity))) +
  geom_boxplot() +
  labs(title = "Weight vs. COVID-19 Severity", x = "Severity", y = "Weight") +
  scale_fill_brewer(palette = "Set1")

# Boxplot for BMI vs. Severity
p4 <- ggplot(training_data, aes(x = factor(severity), y = bmi, fill = factor(severity))) +
  geom_boxplot() +
  labs(title = "BMI vs. COVID-19 Severity", x = "Severity", y = "BMI") +
  scale_fill_brewer(palette = "Set1")

# Boxplot for SBP vs. Severity
p5 <- ggplot(training_data, aes(x = factor(severity), y = sbp, fill = factor(severity))) +
  geom_boxplot() +
  labs(title = "Systolic Blood Pressure vs. COVID-19 Severity", x = "Severity", y = "Systolic BP") +
```

```r
  scale_fill_brewer(palette = "Set1")

# Boxplot for LDL vs. Severity
p6 <- ggplot(training_data, aes(x = factor(severity), y = ldl, fill = factor(severity))) +
  geom_boxplot() +
  labs(title = "LDL Cholesterol vs. COVID-19 Severity", x = "Severity", y = "LDL Cholesterol") +
  scale_fill_brewer(palette = "Set1")

# Boxplot for Depression vs. Severity
p7 <- ggplot(training_data, aes(x = factor(severity), y = depression, fill = factor(severity))) +
  geom_boxplot() +
  labs(title = "Depression vs. COVID-19 Severity", x = "Severity", y = "Depression") +
  scale_fill_brewer(palette = "Set1")
```

We will visualize the relationship between categorical predictors and severity. We will focus on gender, smoking status, hypertension, diabetes, and vaccination status. We will use bar plots showing the proportion within each severity category.

```r
# Function to create proportion bar plots
create_prop_plot <- function(data, varname) {
  plot <- data %>%
    group_by(severity, !!rlang::sym(varname)) %>%
    summarise(Count = n(), .groups = 'drop') %>%
    mutate(Prop = Count / sum(Count)) %>%
    ggplot(aes(x = factor(severity), y = Prop, fill = factor(!!rlang::sym(varname)))) +
    geom_bar(stat = "identity", position = position_dodge()) +
    labs(title = paste(varname, "vs. COVID-19 Severity"), x = "Severity", y = "Proportion") +
    scale_fill_brewer(palette = "Set1")

  return(plot)
}

# Proportion Bar Plots
p8 <- create_prop_plot(training_data, "gender")
p9 <- create_prop_plot(training_data, "smoking")
p10 <- create_prop_plot(training_data, "hypertension")
p11 <- create_prop_plot(training_data, "diabetes")
p12 <- create_prop_plot(training_data, "vaccine")
```

```r
plots_list <- list(p1, p2, p3, p4, p5, p6, p7, p8, p9, p10, p11, p12)
combined_plot <- grid.arrange(grobs = plots_list, ncol = 4, nrow = 3)
```




```
png("combined_plots.png", width = 2000, height = 1500)
grid.arrange(grobs = plots_list, ncol = 4, nrow = 3)
dev.off()
```

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

```
dat_continuous <- training_data %>%
  dplyr::select("age", "height", "weight", "bmi", "sbp", "ldl", "depression")
png("continuous_corrplot.png", width = 400, height = 300)

corrplot(cor(dat_continuous), method = 'number', type = 'lower')

dev.off()
```

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

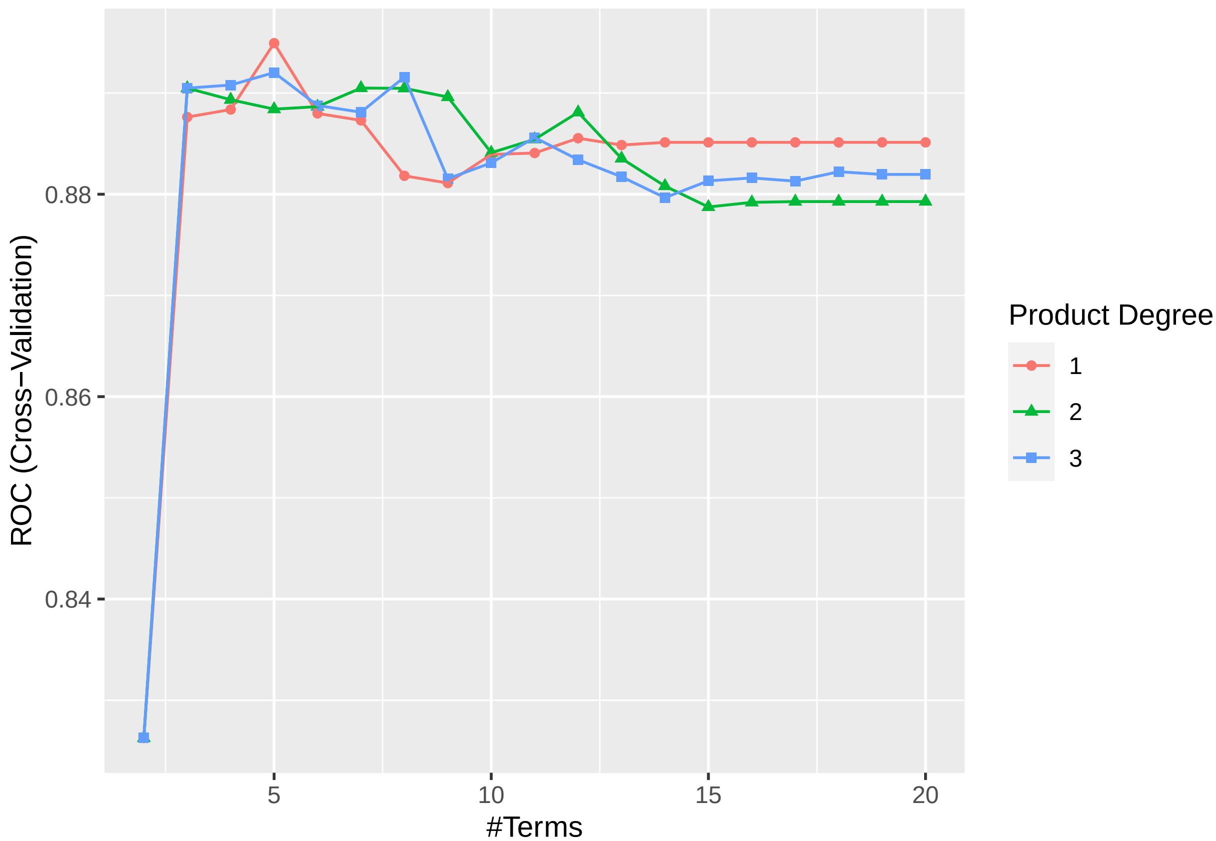
Model Training

MARS

```r
set.seed(1)
mars_grid <- expand.grid(degree = 1:3,
                         nprune = 2:20)

ctrl <- trainControl(method = "cv",
                     number = 10,
                     summaryFunction = twoClassSummary,
                     classProbs = TRUE)

mars.fit <- train(make.names(severity) ~ .,
                  data = training_data,
                  method = "earth",
                  tuneGrid = mars_grid,
                  metric = "ROC",
                  trControl = ctrl)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
ggplot(mars.fit)
```




```r
mars.fit$bestTune
```

```
##   nprune degree
## 4      5      1
```

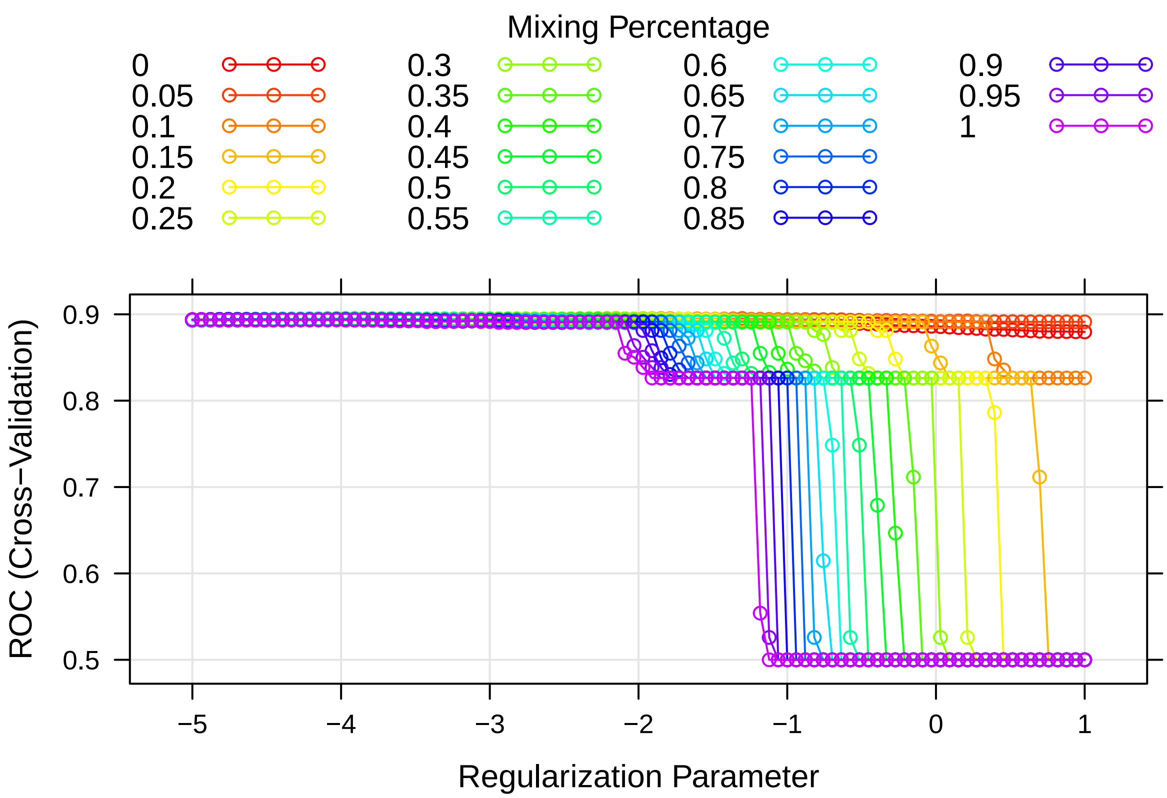
```r
coef(mars.fit$finalModel)
```

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

Penalized Logistic Regression

```r
set.seed(1)
glmnGrid <- expand.grid(.alpha = seq(0, 1, length = 21),
                        .lambda = exp(seq(-5, 1, length = 100)))

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

```r
                      summaryFunction = twoClassSummary,
                      classProbs = TRUE)

glmn.fit <- train(make.names(severity) ~.,
                  data = training_data,
                  method = "glmnet",
                  tuneGrid = glmnGrid,
                  metric = "ROC",
                  trControl = ctrl)

myCol <- rainbow(25)
myPar <- list(superpose.symbol = list(col = myCol),
              superpose.line = list(col = myCol))
plot(glmn.fit, par.settings = myPar, xTrans = function(x) log(x))
```




```r
glmn.fit$bestTune
```

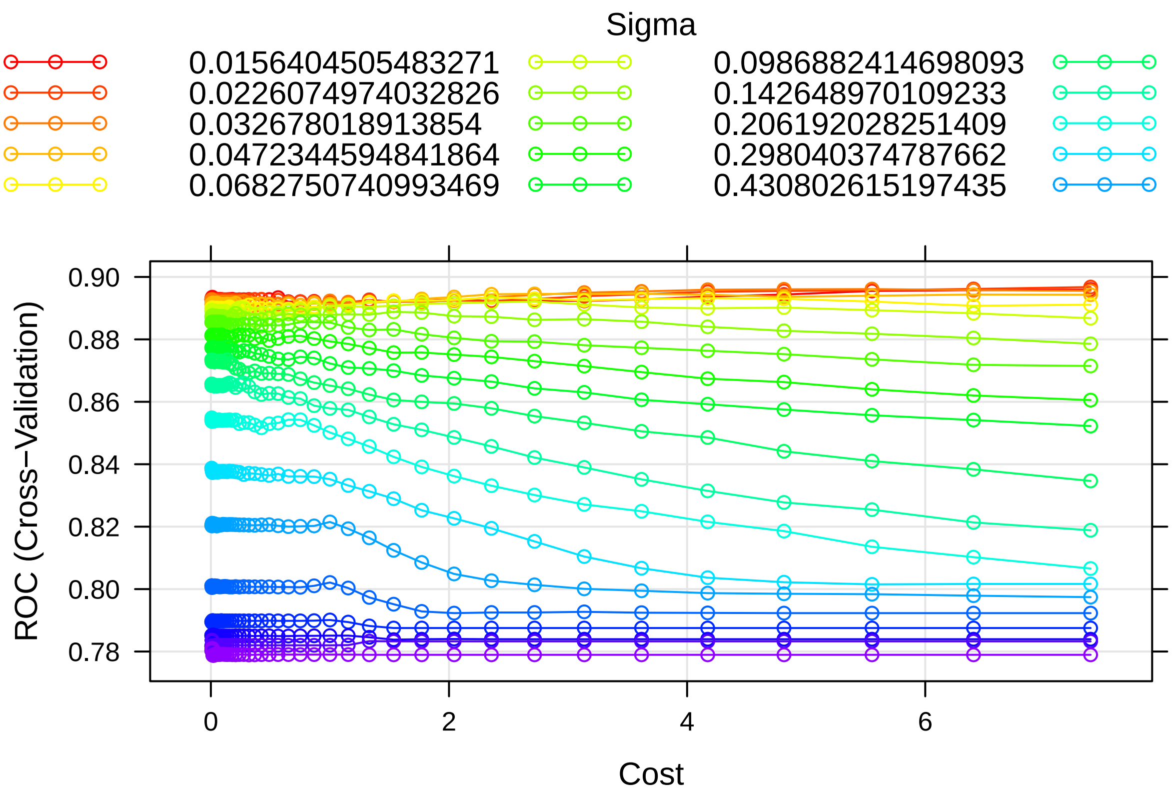
```
##     alpha     lambda
## 441   0.2 0.07609615
```

SVM

```r
set.seed(1)
svmr.grid <- expand.grid(C = exp(seq(-5, 2, len = 50)),
                         sigma = exp(seq(-6, 1, len = 20)))

ctrl <- trainControl(method = "cv",
                     number = 10,
                     summaryFunction = twoClassSummary,
                     classProbs = TRUE)

svmr.fit <- train(make.names(severity) ~ . ,
                  data = training_data,
                  method = "svmRadialSigma",
                  tuneGrid = svmr.grid,
                  metric = "ROC",
                  trControl = ctrl)
```

```
## maximum number of iterations reached 9.44953e-05 9.418661e-05maximum number of iterations reached 1.0
```

```r
myCol <- rainbow(25)
myPar <- list(superpose.symbol = list(col = myCol),
              superpose.line = list(col = myCol))
plot(svmr.fit, highlight = TRUE, par.settings = myPar)
```

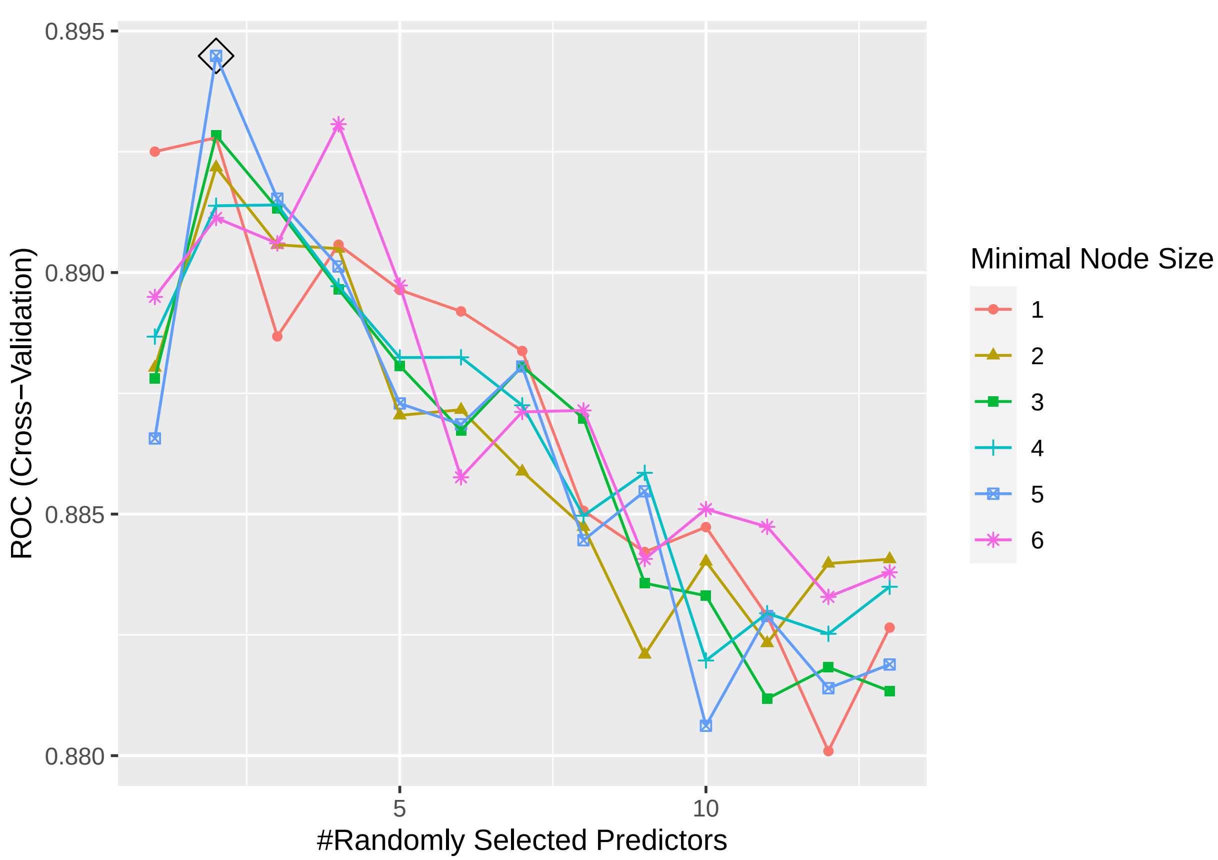


Random Forest

```r
set.seed(1)
rf.grid <- expand.grid(mtry = 1:13,
                       splitrule = "gini",
                       min.node.size = 1:6)

ctrl <- trainControl(method = "cv",
                     number = 10,
                     summaryFunction = twoClassSummary,
                     classProbs = TRUE)

rf.fit <- train(make.names(severity) ~ .,
                data = training_data,
                method = "ranger",
                tuneGrid = rf.grid,
                metric = "ROC",
                trControl = ctrl)
rf.fit$bestTune
```

```
##    mtry splitrule min.node.size
## 11    2      gini             5
```

```r
ggplot(rf.fit, highlight = TRUE)
```



LDA

```
set.seed(1)

ctrl <- trainControl(method = "cv",
                     number = 10,
                     summaryFunction = twoClassSummary,
                     classProbs = TRUE)

lda.fit <- train(make.names(severity) ~.,
                 data = training_data,
                 method = "lda",
                 metric = "ROC",
                 trControl = ctrl)

lda.fit$finalModel
```

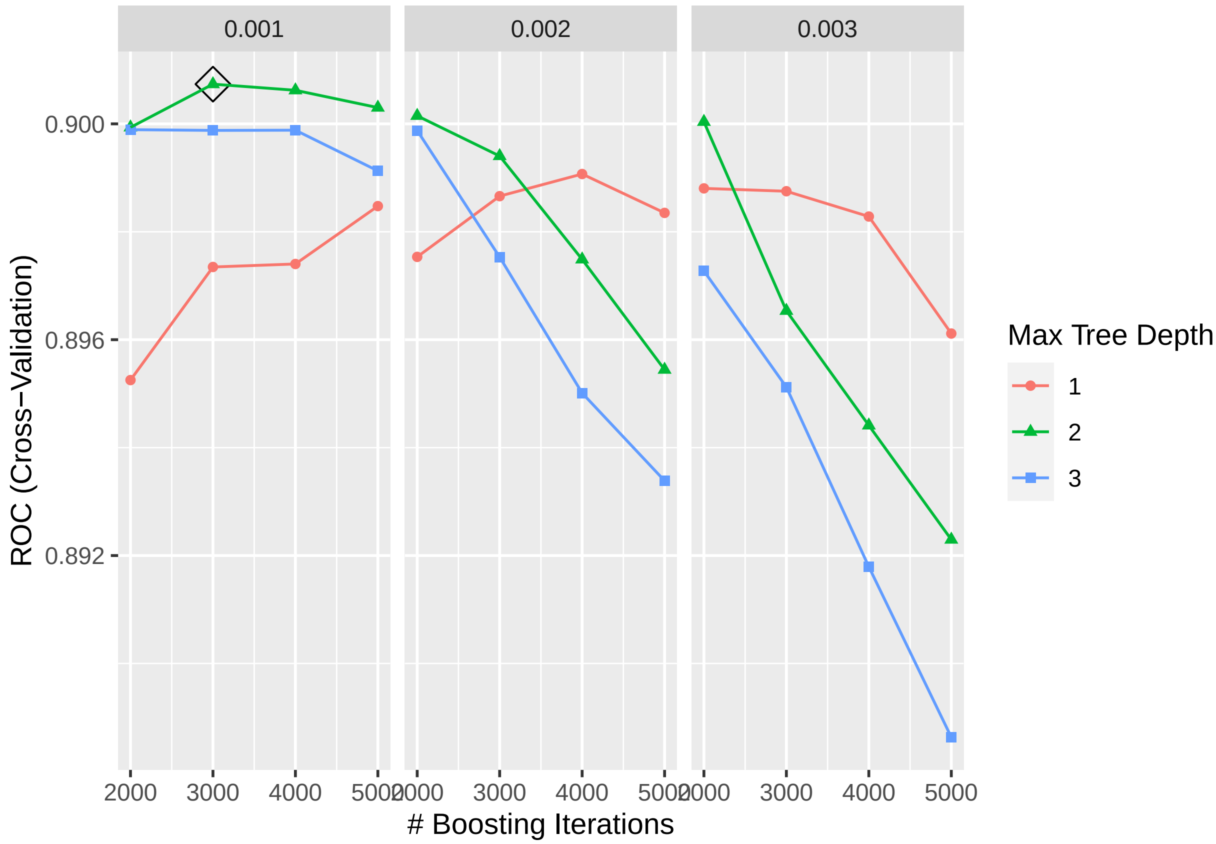
```
## Call:
## lda(x, grouping = y)
##
## Prior probabilities of groups:
## Not.Severe     Severe
##     0.6425     0.3575
##
## Group means:
##                 age genderMale raceBlack raceHispanic raceWhite smokingFormer
## Not.Severe 59.46887  0.5038911 0.2003891   0.09533074 0.6381323     0.3054475
## Severe     61.04545  0.4580420 0.1608392   0.10839161 0.6748252     0.3181818
##            smokingNever   height   weight      bmi diabetesYes hypertensionYes
## Not.Severe    0.5914397 170.1516 79.04125 27.35331   0.1498054       0.3540856
## Severe        0.5699301 169.7269 80.10245 27.86993   0.1538462       0.6503497
##                 sbp      ldl vaccineVaccinated depression
## Not.Severe 128.0272 108.4689         0.8132296   6.912451
## Severe     133.1224 113.4580         0.1608392   6.902098
##
## Coefficients of linear discriminants:
##                            LD1
## age                0.034385337
## genderMale        -0.236369596
## raceBlack          0.133771275
## raceHispanic       0.011221117
## raceWhite          0.126074229
## smokingFormer     -0.220509810
## smokingNever      -0.248324764
## height             0.067561771
## weight            -0.077322293
## bmi                0.301287096
## diabetesYes        0.144018177
## hypertensionYes    0.215120572
## sbp                0.036100853
## ldl                0.004588793
## vaccineVaccinated -2.478676582
## depression        -0.010600382
```

AdaBoost

```
set.seed(1)
gbmA.grid <- expand.grid(n.trees = c(2000,3000,4000,5000),
                         interaction.depth = 1:3,
                         shrinkage = c(0.001,0.002,0.003),
                         n.minobsinnode = 1)

ctrl <- trainControl(method = "cv",
                     number = 10,
                     summaryFunction = twoClassSummary,
                     classProbs = TRUE)

gbmA.fit <- train(make.names(severity) ~ .,
                  data = training_data,
                  method = "gbm",
                  tuneGrid = gbmA.grid,
                  metric = "ROC",
                  trControl = ctrl,
                  distribution = "adaboost",
                  verbose = FALSE)
gbmA.fit$bestTune
```

```
##   n.trees interaction.depth shrinkage n.minobsinnode
## 6    3000                 2     0.001              1
```

```
ggplot(gbmA.fit, highlight = TRUE)
```



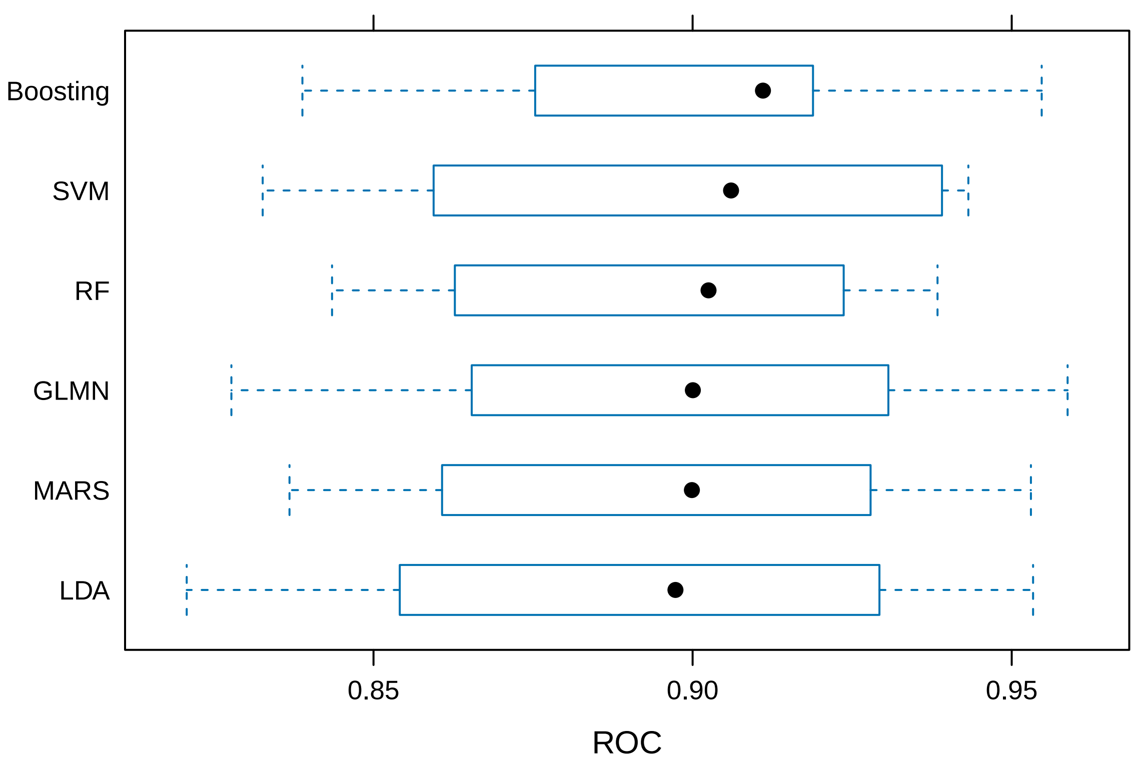
Compare all models

```
resamp <- resamples(list(MARS = mars.fit,
                         GLMN = glmn.fit,
                         SVM = svmr.fit,
                         RF = rf.fit,
                         LDA = lda.fit,
                         Boosting = gbmA.fit))
summary(resamp)
```

```
##
## Call:
## summary.resamples(object = resamp)
##
## Models: MARS, GLMN, SVM, RF, LDA, Boosting
## Number of resamples: 10
##
## ROC
##               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## MARS     0.8368347 0.8617374 0.8998836 0.8949289 0.9242048 0.9530088    0
## GLMN     0.8277311 0.8667531 0.9000364 0.8953483 0.9279149 0.9587559    0
## SVM      0.8326331 0.8639673 0.9060241 0.8967610 0.9359244 0.9432049    0
## RF       0.8435013 0.8629141 0.9024939 0.8944851 0.9221477 0.9383754    0
## LDA      0.8207283 0.8569460 0.8973124 0.8924337 0.9280664 0.9533469    0
```

```
## Boosting 0.8388594 0.8755656 0.9110222 0.9007355 0.9184898 0.9546991    0
##
## Sens
##               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## MARS     0.8431373 0.8461538 0.8725490 0.8834087 0.9128959 0.9607843    0
## GLMN     0.8461538 0.8634050 0.8823529 0.8970965 0.9226998 0.9803922    0
## SVM      0.7692308 0.8125000 0.8448341 0.8505279 0.8823529 0.9411765    0
## RF       0.8653846 0.9082768 0.9411765 0.9360106 0.9754902 0.9807692    0
## LDA      0.7500000 0.8076923 0.8350302 0.8407994 0.8725490 0.9411765    0
## Boosting 0.8653846 0.8829186 0.9019608 0.9164781 0.9558824 0.9807692    0
##
## Spec
##               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## MARS     0.6785714 0.7327586 0.7721675 0.7758621 0.8143473 0.8620690    0
## GLMN     0.6428571 0.7521552 0.7586207 0.7656404 0.8143473 0.8620690    0
## SVM      0.7142857 0.8001847 0.8423645 0.8284483 0.8608374 0.8965517    0
## RF       0.5714286 0.6813424 0.7413793 0.7270936 0.7789409 0.8275862    0
## LDA      0.7142857 0.8017241 0.8571429 0.8352217 0.8620690 0.8965517    0
## Boosting 0.6428571 0.6982759 0.7413793 0.7480296 0.7912562 0.8620690    0
```

```r
bwplot(resamp, metric = "ROC")
```




```r
resamp_summary <- summary(resamp)

roc_summary <- as.data.frame(resamp_summary$statistics$ROC)
```

```r
roc_summary$Model <- rownames(roc_summary)
write.csv(roc_summary, "Roc_summary.csv", row.names = FALSE)
```

```r
png("Roc_boxplot.png", width = 400, height = 300)
bwplot(resamp, metric = "ROC")
dev.off()
```

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

```r
boosting.pred <- predict(gbmA.fit, newdata = test_data, type = "prob")
boosting.pred
```

```
##      Not.Severe     Severe
## 1    0.89630340 0.10369660
## 2    0.43313344 0.56686656
## 3    0.48536884 0.51463116
## 4    0.09701847 0.90298153
## 5    0.90786982 0.09213018
## 6    0.16864968 0.83135032
## 7    0.29994593 0.70005407
## 8    0.21075008 0.78924992
## 9    0.13076766 0.86923234
## 10   0.82659319 0.17340681
## 11   0.88841781 0.11158219
## 12   0.13667853 0.86332147
## 13   0.80671339 0.19328661
## 14   0.87902215 0.12097785
## 15   0.90451779 0.09548221
## 16   0.76508389 0.23491611
## 17   0.43040267 0.56959733
## 18   0.85820468 0.14179532
## 19   0.83302007 0.16697993
## 20   0.90749853 0.09250147
## 21   0.93204529 0.06795471
## 22   0.66431513 0.33568487
## 23   0.90224590 0.09775410
## 24   0.63228775 0.36771225
## 25   0.46271132 0.53728868
## 26   0.36734190 0.63265810
## 27   0.59463071 0.40536929
## 28   0.81730975 0.18269025
## 29   0.88712291 0.11287709
## 30   0.86823897 0.13176103
## 31   0.90730277 0.09269723
## 32   0.39872172 0.60127828
## 33   0.38333235 0.61666765
## 34   0.92901686 0.07098314
## 35   0.92442802 0.07557198
## 36   0.82882768 0.17117232
## 37   0.92975988 0.07024012
## 38   0.89737049 0.10262951
```

```
## 39  0.92412334 0.07587666
## 40  0.90447153 0.09552847
## 41  0.80911791 0.19088209
## 42  0.89197911 0.10802089
## 43  0.16119579 0.83880421
## 44  0.86505699 0.13494301
## 45  0.90463468 0.09536532
## 46  0.25120471 0.74879529
## 47  0.83158037 0.16841963
## 48  0.86855693 0.13144307
## 49  0.89904276 0.10095724
## 50  0.16234205 0.83765795
## 51  0.91067025 0.08932975
## 52  0.89465317 0.10534683
## 53  0.37343253 0.62656747
## 54  0.90610908 0.09389092
## 55  0.12278553 0.87721447
## 56  0.82262376 0.17737624
## 57  0.15771395 0.84228605
## 58  0.88992615 0.11007385
## 59  0.82794851 0.17205149
## 60  0.41472213 0.58527787
## 61  0.47594334 0.52405666
## 62  0.93578565 0.06421435
## 63  0.59317092 0.40682908
## 64  0.83680774 0.16319226
## 65  0.83380199 0.16619801
## 66  0.83583076 0.16416924
## 67  0.84462906 0.15537094
## 68  0.82533962 0.17466038
## 69  0.82769313 0.17230687
## 70  0.33682737 0.66317263
## 71  0.28624634 0.71375366
## 72  0.15022671 0.84977329
## 73  0.09694790 0.90305210
## 74  0.83721026 0.16278974
## 75  0.84263824 0.15736176
## 76  0.92562460 0.07437540
## 77  0.82965066 0.17034934
## 78  0.84491196 0.15508804
## 79  0.16538680 0.83461320
## 80  0.79523000 0.20477000
## 81  0.91967321 0.08032679
## 82  0.91734443 0.08265557
## 83  0.89476862 0.10523138
## 84  0.89444763 0.10555237
## 85  0.82213997 0.17786003
## 86  0.33064473 0.66935527
## 87  0.29481632 0.70518368
## 88  0.16952340 0.83047660
## 89  0.56554200 0.43445800
## 90  0.84461564 0.15538436
## 91  0.16653051 0.83346949
## 92  0.29974965 0.70025035
```

```
## 93  0.08620767 0.91379233
## 94  0.08550628 0.91449372
## 95  0.70556311 0.29443689
## 96  0.91067020 0.08932980
## 97  0.79940976 0.20059024
## 98  0.52096980 0.47903020
## 99  0.52738486 0.47261514
## 100 0.87533982 0.12466018
## 101 0.45085086 0.54914914
## 102 0.12700413 0.87299587
## 103 0.83689636 0.16310364
## 104 0.86151238 0.13848762
## 105 0.91107545 0.08892455
## 106 0.29365365 0.70634635
## 107 0.57086494 0.42913506
## 108 0.79878612 0.20121388
## 109 0.92424837 0.07575163
## 110 0.86046557 0.13953443
## 111 0.08615355 0.91384645
## 112 0.85219908 0.14780092
## 113 0.08603372 0.91396628
## 114 0.91194087 0.08805913
## 115 0.21730042 0.78269958
## 116 0.33543170 0.66456830
## 117 0.88932934 0.11067066
## 118 0.92045660 0.07954340
## 119 0.82824881 0.17175119
## 120 0.92836296 0.07163704
## 121 0.91000058 0.08999942
## 122 0.10885632 0.89114368
## 123 0.92837357 0.07162643
## 124 0.92734810 0.07265190
## 125 0.28608311 0.71391689
## 126 0.86104339 0.13895661
## 127 0.90264339 0.09735661
## 128 0.90530990 0.09469010
## 129 0.33398536 0.66601464
## 130 0.91513714 0.08486286
## 131 0.90832988 0.09167012
## 132 0.92321434 0.07678566
## 133 0.23457417 0.76542583
## 134 0.82805458 0.17194542
## 135 0.90947823 0.09052177
## 136 0.63393663 0.36606337
## 137 0.93141038 0.06858962
## 138 0.87155771 0.12844229
## 139 0.89921140 0.10078860
## 140 0.91955694 0.08044306
## 141 0.80501464 0.19498536
## 142 0.80820967 0.19179033
## 143 0.85839121 0.14160879
## 144 0.91764995 0.08235005
## 145 0.81786168 0.18213832
## 146 0.92854100 0.07145900
```

```
## 147 0.33320542 0.66679458
## 148 0.80193426 0.19806574
## 149 0.91545423 0.08454577
## 150 0.89709530 0.10290470
## 151 0.92254355 0.07745645
## 152 0.89502023 0.10497977
## 153 0.83439419 0.16560581
## 154 0.28949786 0.71050214
## 155 0.18781695 0.81218305
## 156 0.92054712 0.07945288
## 157 0.92957568 0.07042432
## 158 0.82460084 0.17539916
## 159 0.56796791 0.43203209
## 160 0.30046659 0.69953341
## 161 0.90844882 0.09155118
## 162 0.93377849 0.06622151
## 163 0.25026066 0.74973934
## 164 0.53685511 0.46314489
## 165 0.34890735 0.65109265
## 166 0.54557646 0.45442354
## 167 0.92047548 0.07952452
## 168 0.25322062 0.74677938
## 169 0.91998302 0.08001698
## 170 0.85804436 0.14195564
## 171 0.93023826 0.06976174
## 172 0.93159371 0.06840629
## 173 0.90705164 0.09294836
## 174 0.78445812 0.21554188
## 175 0.87013877 0.12986123
## 176 0.91727599 0.08272401
## 177 0.92543856 0.07456144
## 178 0.89979656 0.10020344
## 179 0.80527319 0.19472681
## 180 0.64309591 0.35690409
## 181 0.89487274 0.10512726
## 182 0.84814624 0.15185376
## 183 0.82740004 0.17259996
## 184 0.62368353 0.37631647
## 185 0.34434075 0.65565925
## 186 0.83839183 0.16160817
## 187 0.83510160 0.16489840
## 188 0.77502233 0.22497767
## 189 0.53967743 0.46032257
## 190 0.80847454 0.19152546
## 191 0.23514521 0.76485479
## 192 0.20449541 0.79550459
## 193 0.21511544 0.78488456
## 194 0.33885739 0.66114261
## 195 0.92400030 0.07599970
## 196 0.83279144 0.16720856
## 197 0.92323815 0.07676185
## 198 0.89392083 0.10607917
## 199 0.86127466 0.13872534
## 200 0.40622057 0.59377943
```