Final Project

Yueyi Xu

2024-05-08

```
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")
         )
```

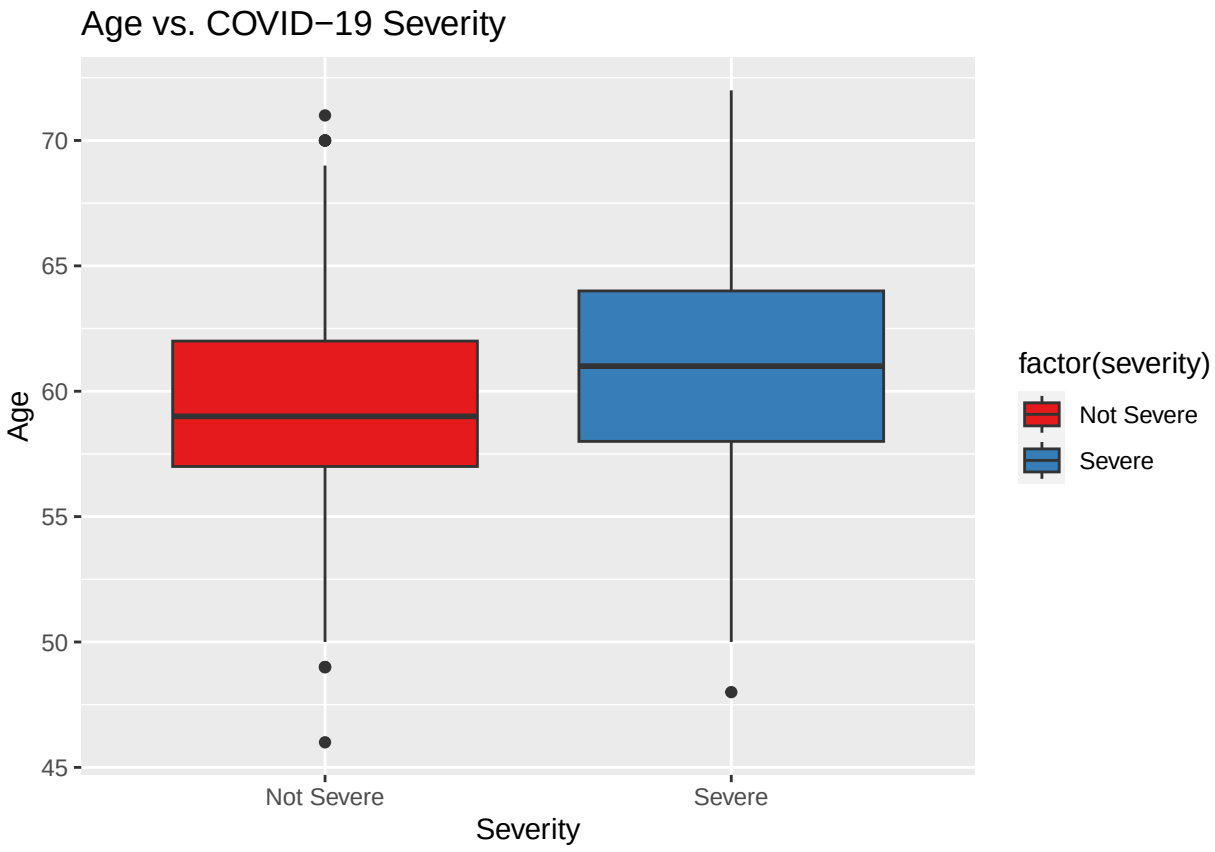
```
test_data <- test_data %>%
  janitor::clean_names() %>%
  dplyr::select(-id) %>%
  mutate(severity = case_match(as.numeric(severity),
                               1 ~ "Not Severe",
                               2 ~ "Severe"),
         severity = factor(severity),
         gender = case_match(gender,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

```
                          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")
         )
```

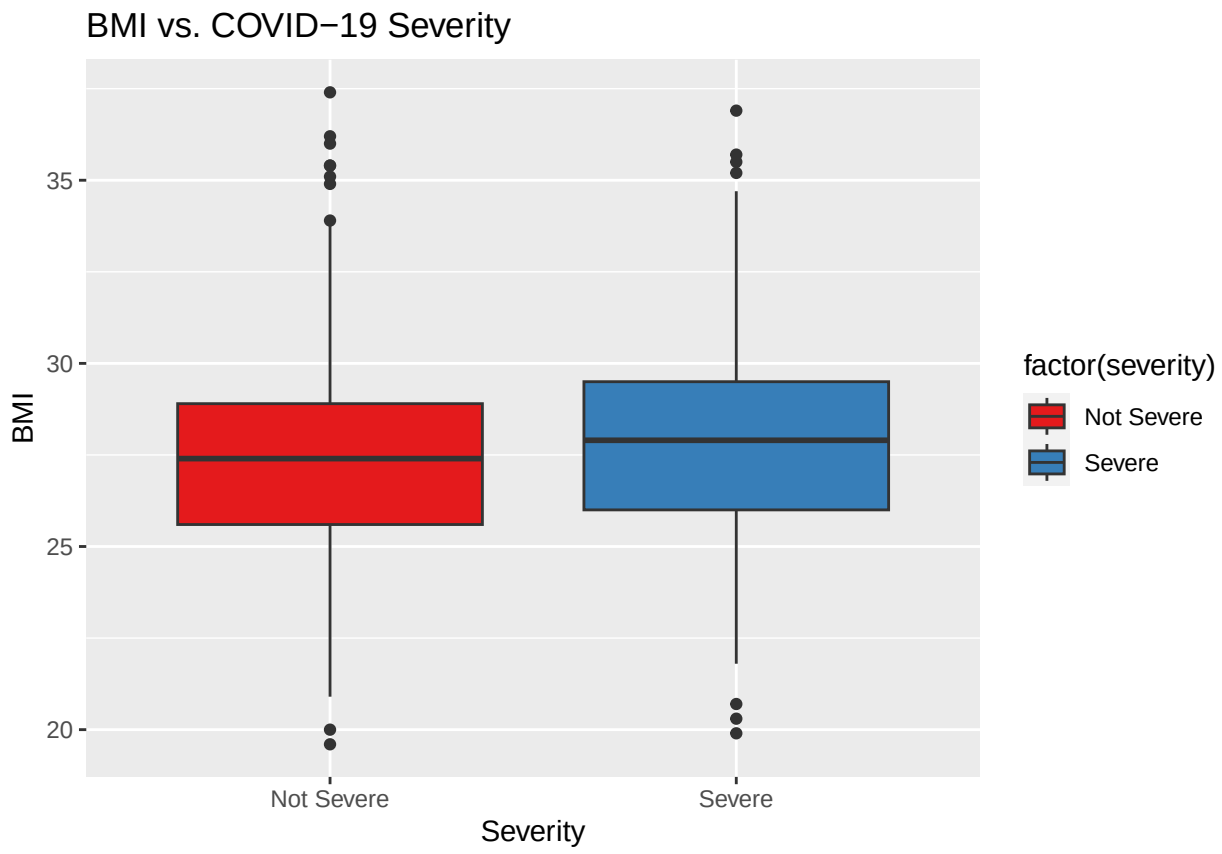
Exploratory analysis and data visualization

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

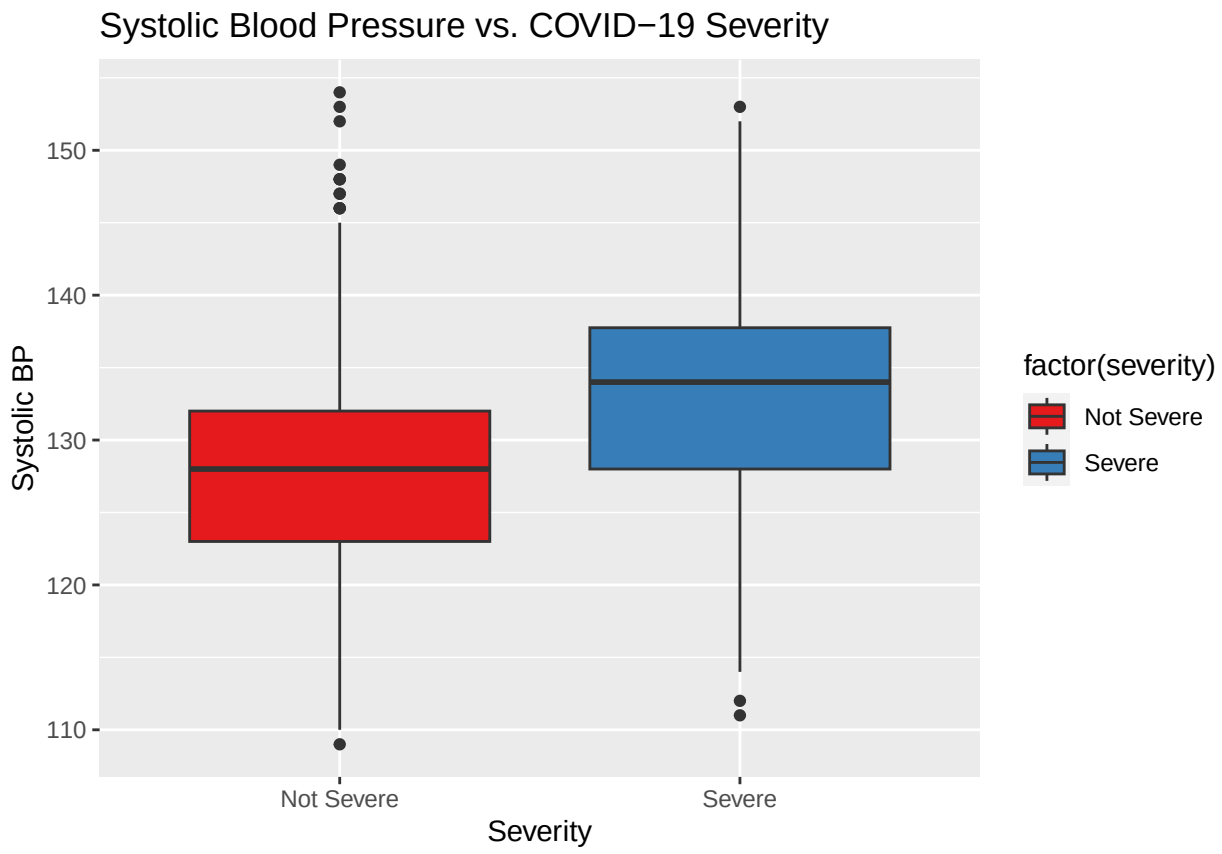
```
# Boxplot for Age vs. Severity
ggplot(training_data, aes(x = factor(severity), y = age, fill = factor(severity))) +
  geom_boxplot() +
  labs(title = "Age vs. COVID-19 Severity", x = "Severity", y = "Age") +
  scale_fill_brewer(palette = "Set1")
```



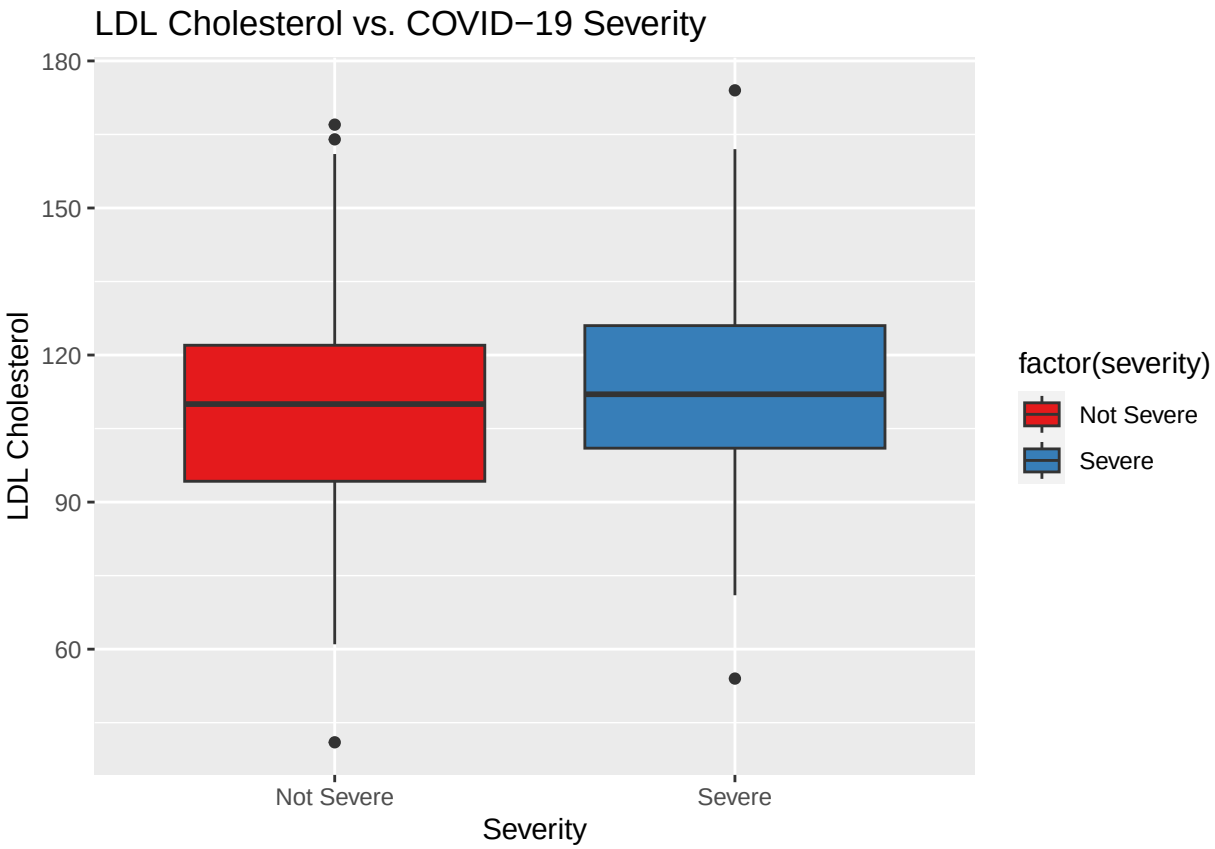

```
# Boxplot for BMI vs. Severity
ggplot(training_data, aes(x = factor(severity), y = bmi, fill = factor(severity))) +
  geom_boxplot() +
  labs(title = "BMI vs. COVID-19 Severity", x = "Severity", y = "BMI") +
  scale_fill_brewer(palette = "Set1")
```




```
# Boxplot for SBP vs. Severity
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") +
  scale_fill_brewer(palette = "Set1")
```

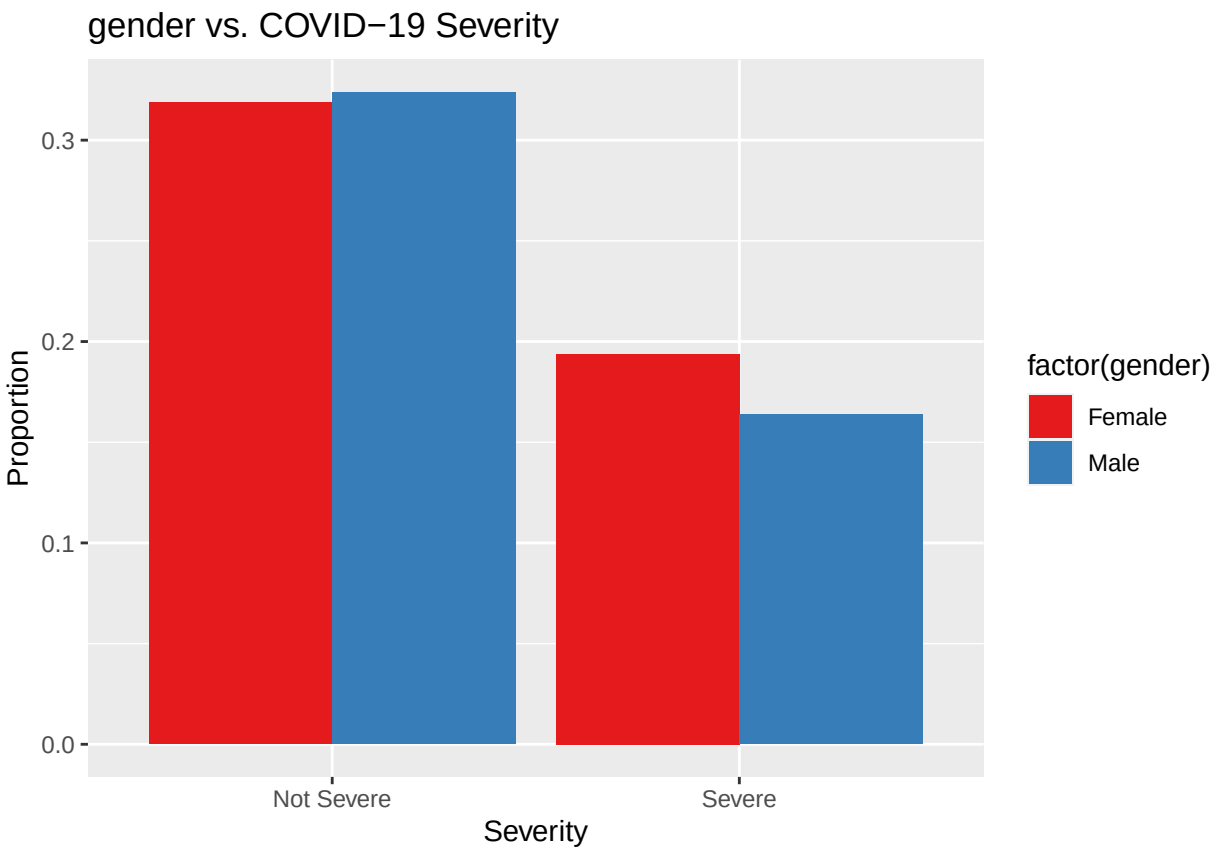



```
# Boxplot for LDL vs. Severity
ggplot(training_data, aes(x = factor(severity), y = ldl, fill = factor(severity))) +
  geom_boxplot() +
  labs(title = "LDL Cholesterol vs. COVID-19 Severity", x = "Severity", y = "LDL Cholesterol") +
  scale_fill_brewer(palette = "Set1")
```

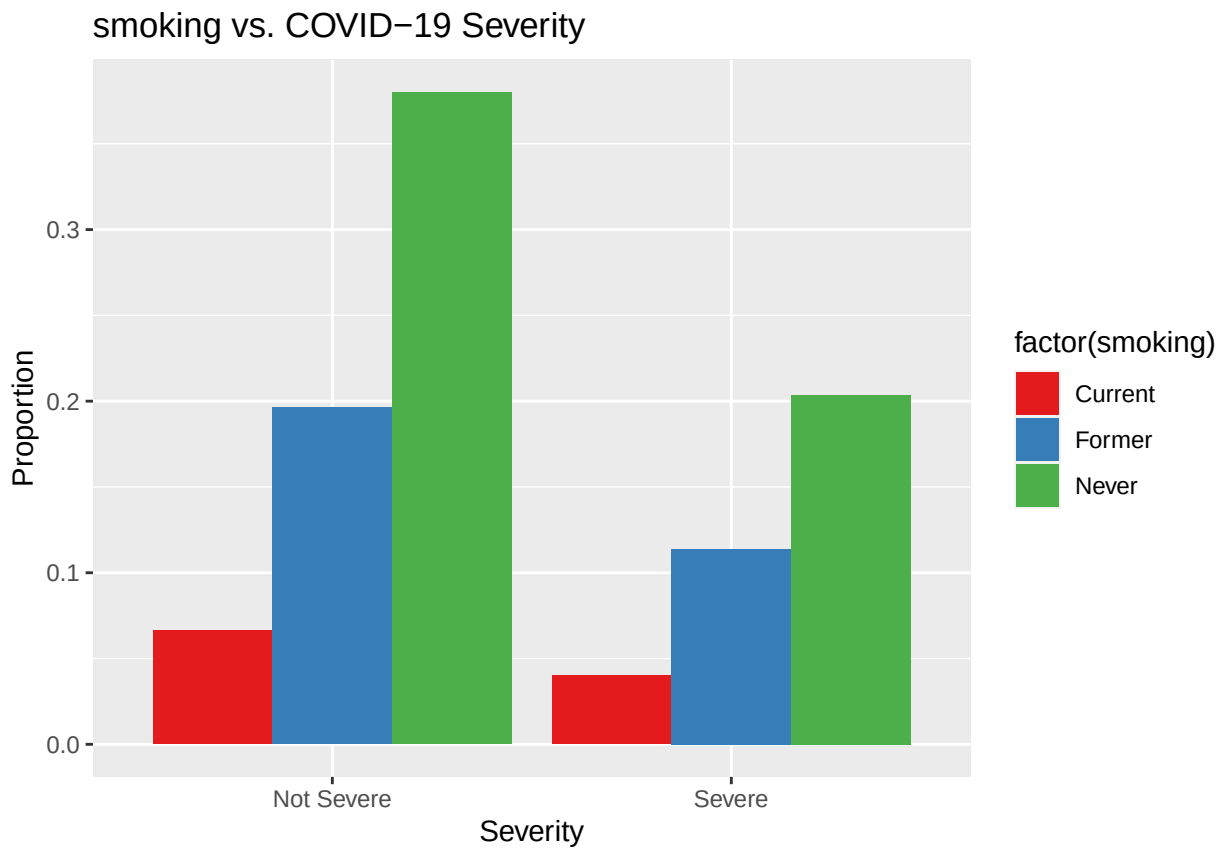


We will visualize the relationship between categorical predictors and severity. We will focus on gender, smoking status, hypertension, diabetes, and vaccination status. We will use bar plots showing the proportion within each severity category.

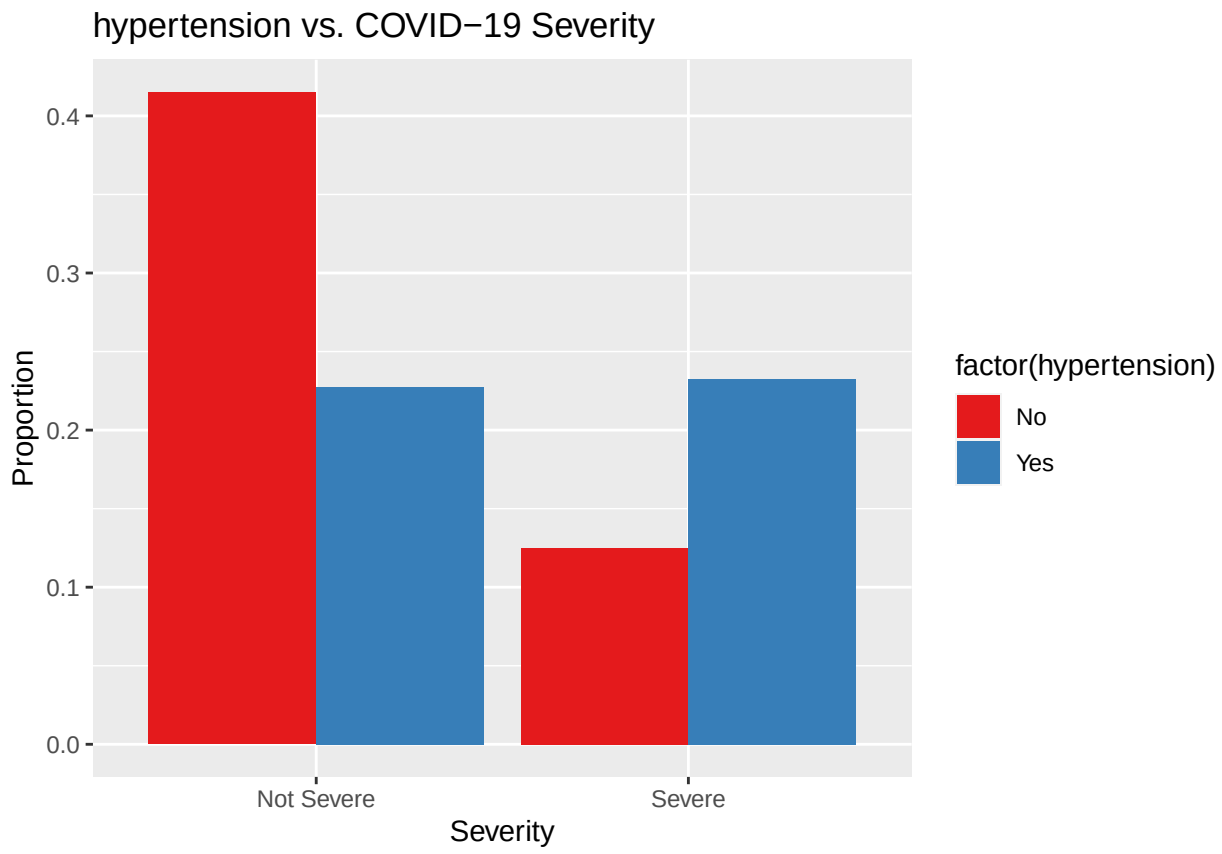
```r
# Function to create proportion bar plots
create_prop_plot <- function(data, varname) {
  data %>%
    group_by(severity, !!sym(varname)) %>%
    summarise(Count = n(), .groups = 'drop') %>%
    mutate(Prop = Count / sum(Count)) %>%
    ggplot(aes(x = factor(severity), y = Prop, fill = factor(!!sym(varname))))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

# Generate plots
create_prop_plot(training_data, "gender")
```



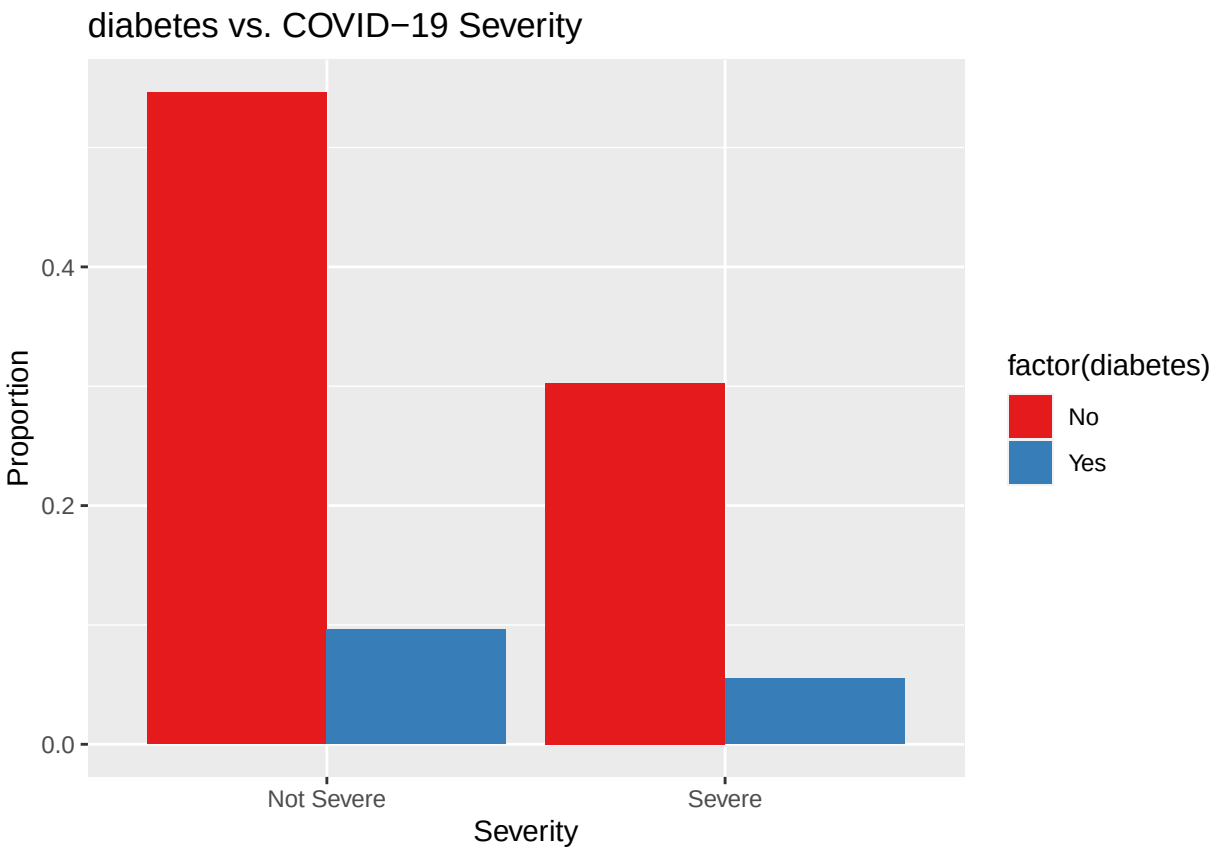

```
create_prop_plot(training_data, "smoking")
```



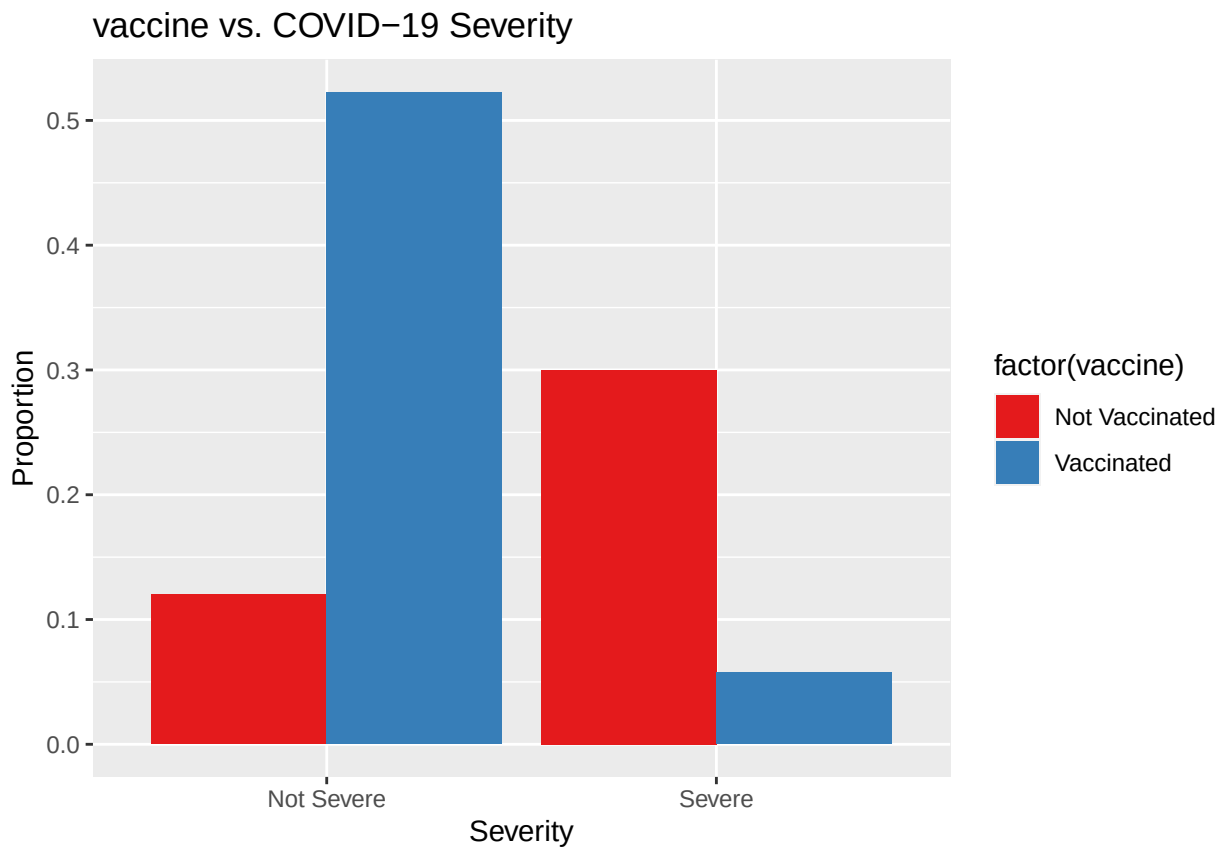

```
create_prop_plot(training_data, "hypertension")
```



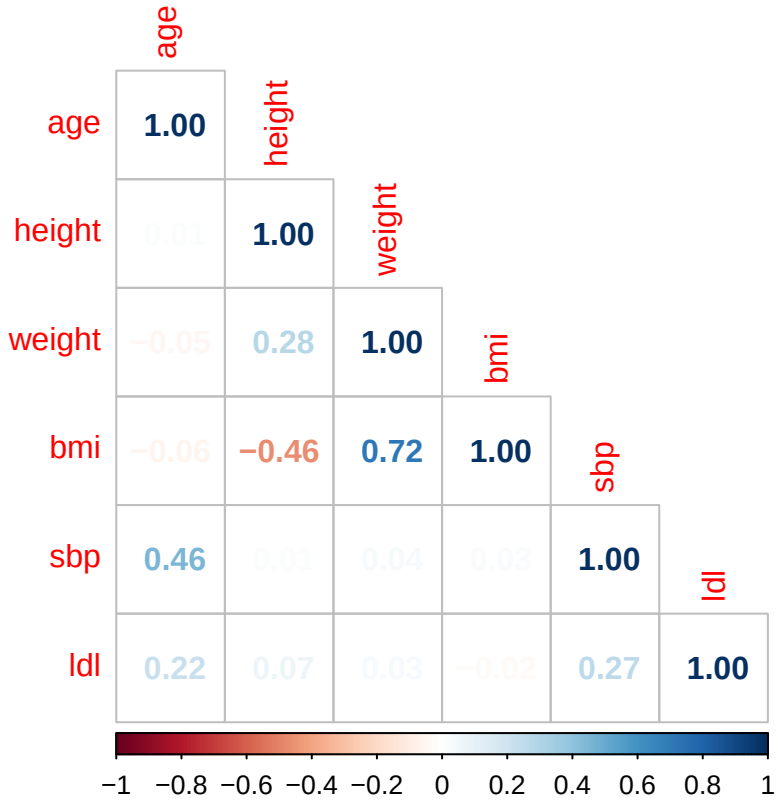

```
create_prop_plot(training_data, "diabetes")
```




```
create_prop_plot(training_data, "vaccine")
```




```
dat_continuous <- training_data %>%
  dplyr::select("age", "height", "weight", "bmi", "sbp", "ldl")

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



Conclusion:

1. Age (box plot) Age has shown a distinct distribution between severe and non-severe COVID-19 cases, with older individuals tending to have more severe outcomes. Given the visible difference in the age distributions between severity groups, age is a crucial variable that could help the model capture risk stratification more effectively.

2. SBP (box plot) The boxplot showing higher SBP in severe COVID-19 cases supports its inclusion, reflecting the impact of cardiovascular health on disease outcomes. High blood pressure is known to compromise vascular integrity and could exacerbate COVID-19 severity.

3. Diabetes (bar plot) Diabetes has a notable impact on immune system efficiency. The higher proportion of severe cases among diabetics in your data supports the inclusion of this variable, reflecting the metabolic and immune challenges posed by this condition.

4. Vaccine (bar plot) There's a significant difference in COVID-19 severity between vaccinated and unvaccinated individuals, as seen in the bar plots. Including vaccination status can help quantify the protective effect of vaccines against severe COVID-19, which is crucial for the model.

5. Age + Height + Weight + BMI + SBP + LDL (correlation plot) Age-Related Increases in SBP and LDL: The moderate correlations of age with SBP and LDL highlight common age-related health risks. Height, Weight, and BMI: The relationships among these three are as expected, with height inversely related to BMI when weight is constant, but weight strongly driving increases in BMI. BMI and SBP: The strong positive correlation is significant from a health perspective, reinforcing the importance of weight management in controlling or preventing hypertension. Therefore, building models could help quantify the impact of these variables on each other, particularly useful for predicting health outcomes based on changes in BMI, SBP, or age.

Model Training

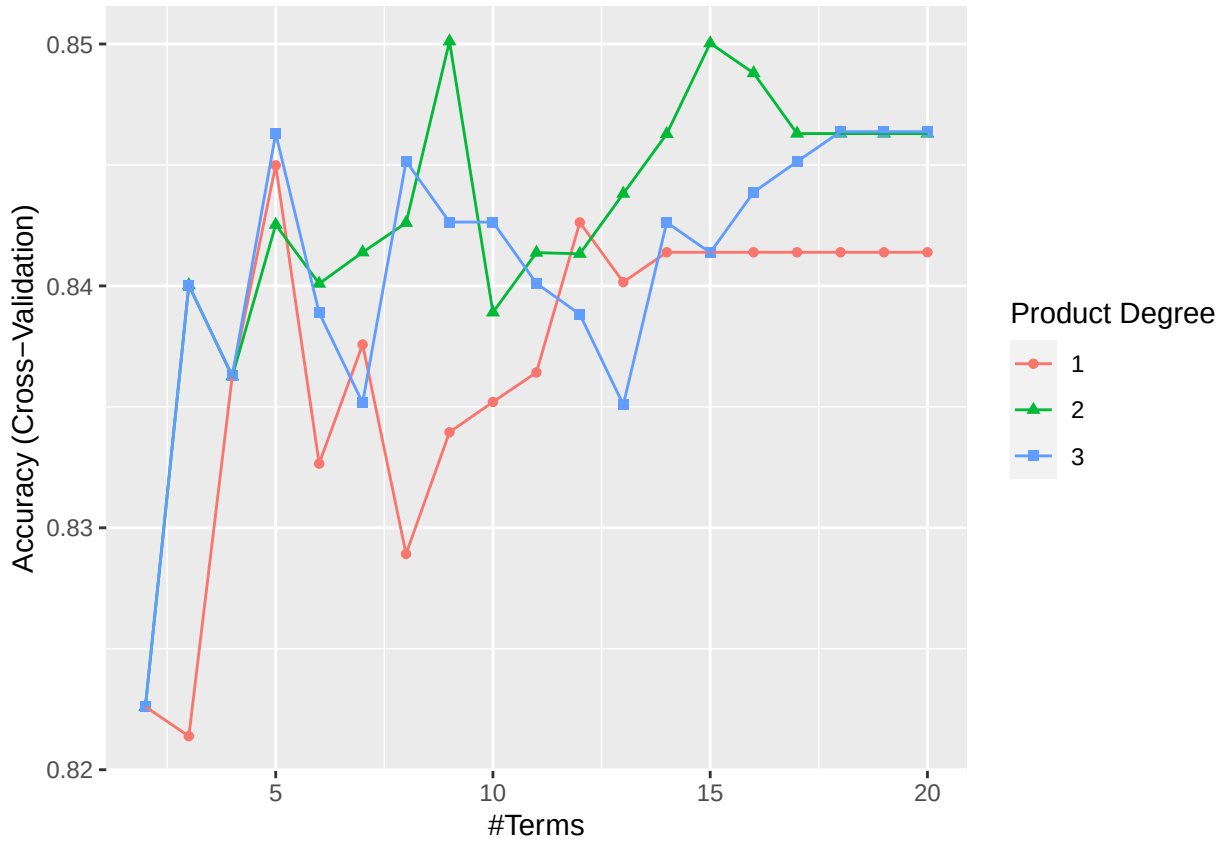
```
ctrl <- trainControl(method = "cv",
                     number = 10,
                     savePredictions = "final")
```

MARS

```
set.seed(1)
mars_grid <- expand.grid(degree = 1:3,
                         nprune = 2:20)
mars.fit <- train(severity ~ .,
                  data = training_data,
                  method = "earth",
                  tuneGrid = mars_grid,
                  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
```

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




```
mars.fit$bestTune
```

```
##    nprune degree
## 27      9      2
```

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

```
##                         (Intercept)                   vaccineVaccinated
##                           2.95296706                          -3.66937278
##                           h(139-sbp)    h(sbp-124) * vaccineVaccinated
##                          -0.19903730                          -0.05079822
##    h(124-sbp) * vaccineVaccinated                          h(bmi-27.2)
##                           0.24710679                           0.34990771
## h(height-175) * vaccineVaccinated     h(bmi-27.2) * hypertensionYes
##                           0.43088146                          -0.22462393
##                     h(height-174.6)
##                          -0.30640724
```

```
mars.pred <- predict(mars.fit, newdata = test_data)
confusionMatrix(data = as.factor(mars.pred),
                reference = test_data$severity,
                positive = "Severe")
```

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

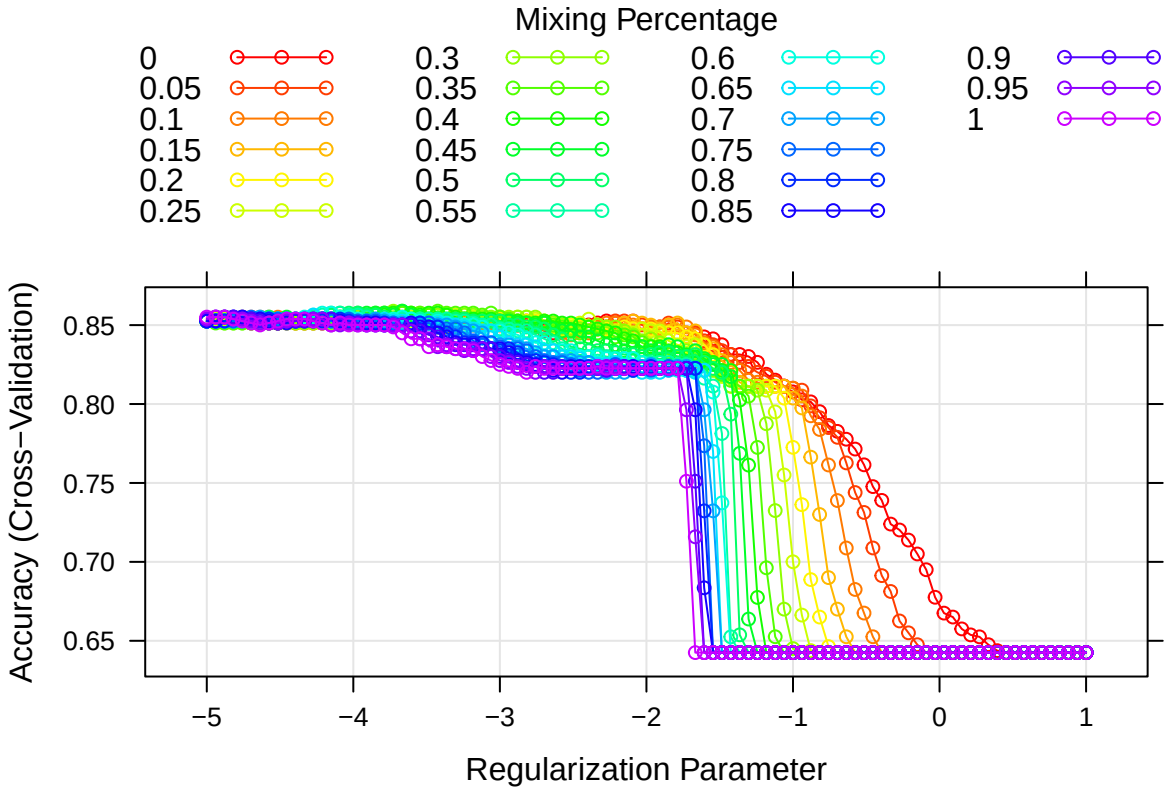
```
##
##             Reference
## Prediction   Not Severe Severe
##   Not Severe        124     18
##   Severe             11     47
##
##                Accuracy : 0.855
##                  95% CI : (0.7984, 0.9007)
##     No Information Rate : 0.675
##     P-Value [Acc > NIR] : 4.95e-09
##
##                   Kappa : 0.66
##
##  Mcnemar's Test P-Value : 0.2652
##
##             Sensitivity : 0.7231
##             Specificity : 0.9185
##          Pos Pred Value : 0.8103
##          Neg Pred Value : 0.8732
##              Prevalence : 0.3250
##          Detection Rate : 0.2350
##    Detection Prevalence : 0.2900
##       Balanced Accuracy : 0.8208
##
##        'Positive' Class : Severe
##
```

Penalized Logistic Regression

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

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

print(glmn.fit$bestTune)
```

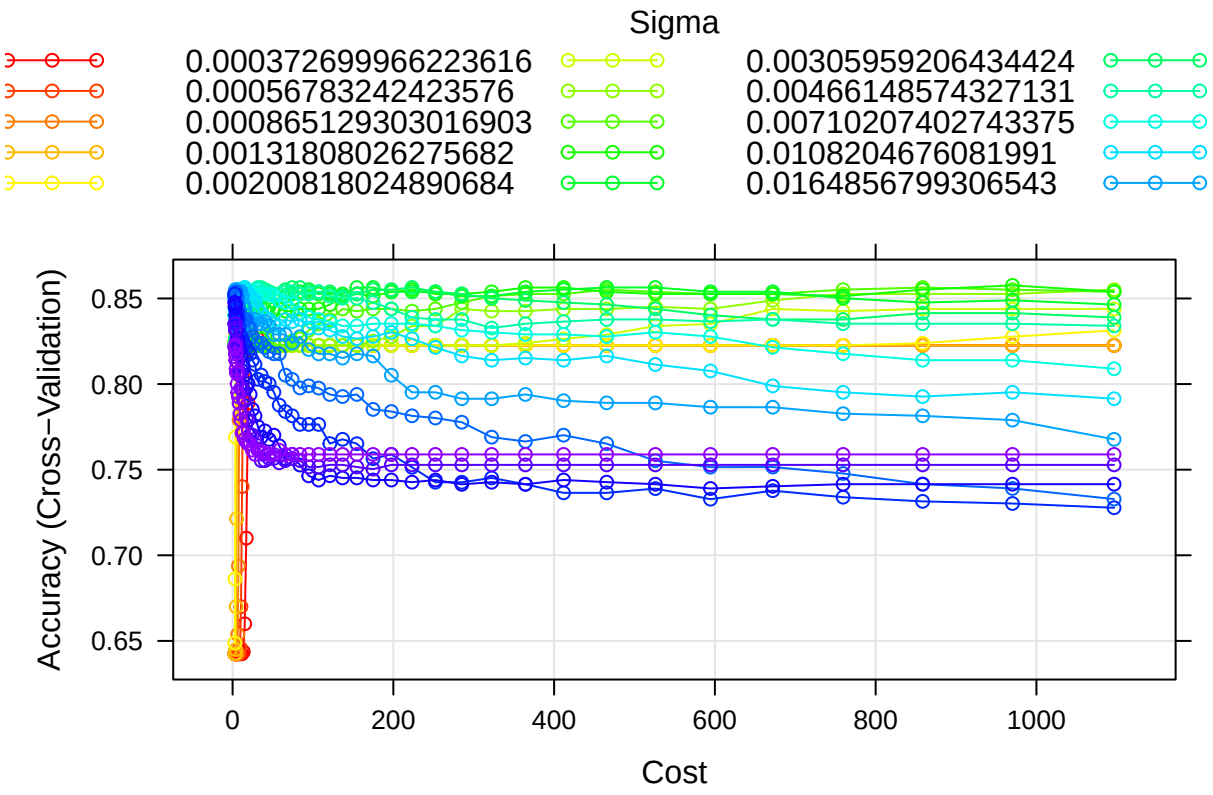
```
##     alpha     lambda
## 727  0.35 0.03257395
```

```
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))
```



SVM

```
svmr.grid <- expand.grid(C = exp(seq(1, 7, len = 50)),
                         sigma = exp(seq(-10, -2, len = 20)))

set.seed(1)
svmr.fit <- train(severity ~ . ,
                  data = training_data,
                  method = "svmRadialSigma",
                  tuneGrid = svmr.grid,
                  trControl = ctrl)

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




```
svmr.pred <- predict(svmr.fit, newdata = test_data)

confusionMatrix(data = as.factor(svmr.pred),
                reference = test_data$severity,
                positive = "Severe")
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   Not Severe Severe
##   Not Severe        125     16
##   Severe             10     49
##
##                Accuracy : 0.87
##                  95% CI : (0.8153, 0.9133)
##     No Information Rate : 0.675
##     P-Value [Acc > NIR] : 1.77e-10
##
##                   Kappa : 0.6964
##
##  Mcnemar's Test P-Value : 0.3268
##
##             Sensitivity : 0.7538
##             Specificity : 0.9259
##          Pos Pred Value : 0.8305
```

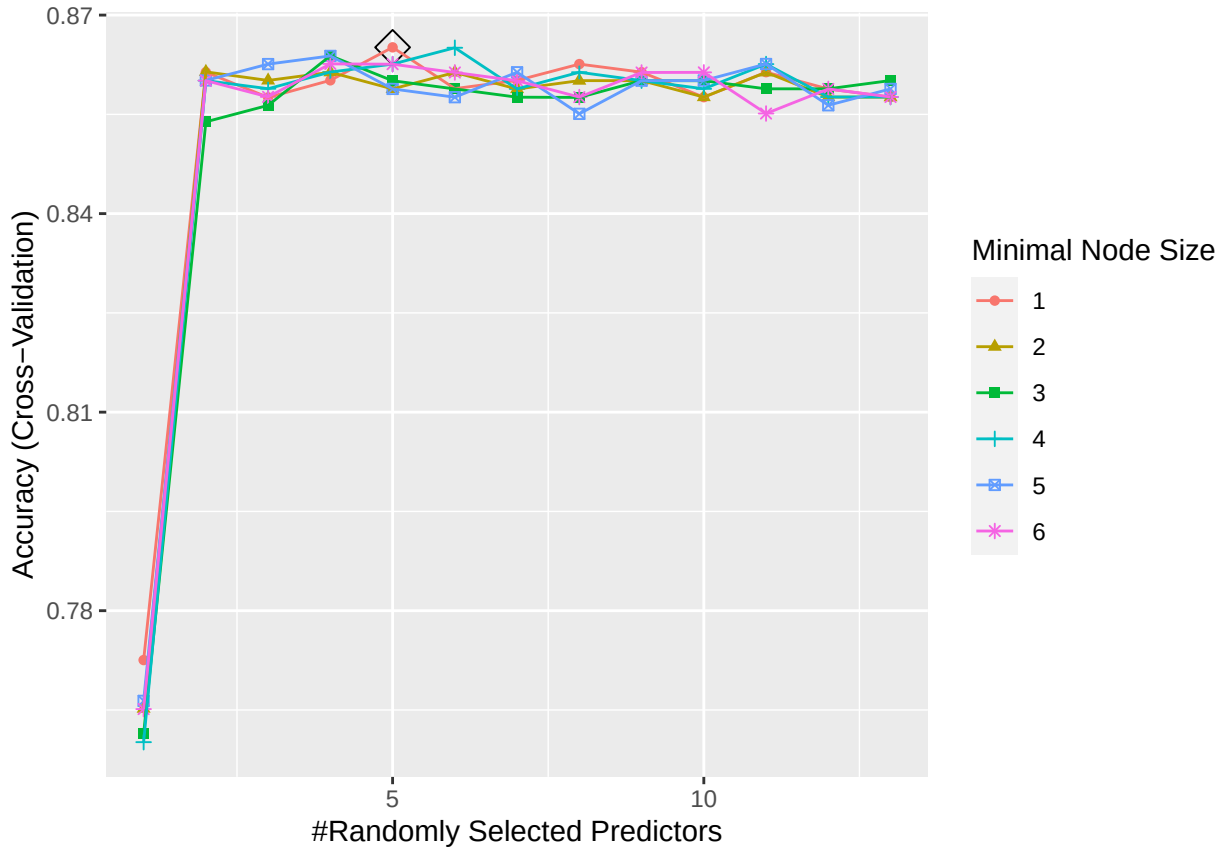
```
##          Neg Pred Value : 0.8865
##              Prevalence : 0.3250
##          Detection Rate : 0.2450
##    Detection Prevalence : 0.2950
##       Balanced Accuracy : 0.8399
##
##        'Positive' Class : Severe
##
```

Random Forest

```
rf.grid <- expand.grid(mtry = 1:13,
                       splitrule = "gini",
                       min.node.size = 1:6)
set.seed(1)
rf.fit <- train(severity ~ .,
                  data = training_data,
                  method = "ranger",
                  tuneGrid = rf.grid,
                  trControl = ctrl)
rf.fit$bestTune
```

```
##    mtry splitrule min.node.size
## 25    5      gini             1
```

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




```
rf.pred <- predict(rf.fit, newdata = test_data)

confusionMatrix(data = as.factor(rf.pred),
                reference = test_data$severity,
                positive = "Severe")
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   Not Severe Severe
##   Not Severe        124     18
##   Severe             11     47
##
##                Accuracy : 0.855
##                  95% CI : (0.7984, 0.9007)
##     No Information Rate : 0.675
##     P-Value [Acc > NIR] : 4.95e-09
##
##                   Kappa : 0.66
##
##  Mcnemar's Test P-Value : 0.2652
##
##             Sensitivity : 0.7231
##             Specificity : 0.9185
##          Pos Pred Value : 0.8103
```

```
##          Neg Pred Value : 0.8732
##              Prevalence : 0.3250
##          Detection Rate : 0.2350
##    Detection Prevalence : 0.2900
##       Balanced Accuracy : 0.8208
##
##        'Positive' Class : Severe
##
```

LDA

```
set.seed(1)
lda.fit <- train(severity ~.,
                 data = training_data,
                 method = "lda",
                 metric = "Accuracy",
                 trControl = ctrl)

lda.pred2 <- predict(lda.fit, newdata = test_data)

confusionMatrix(data = as.factor(lda.pred2),
                reference = test_data$severity,
                positive = "Severe")
```

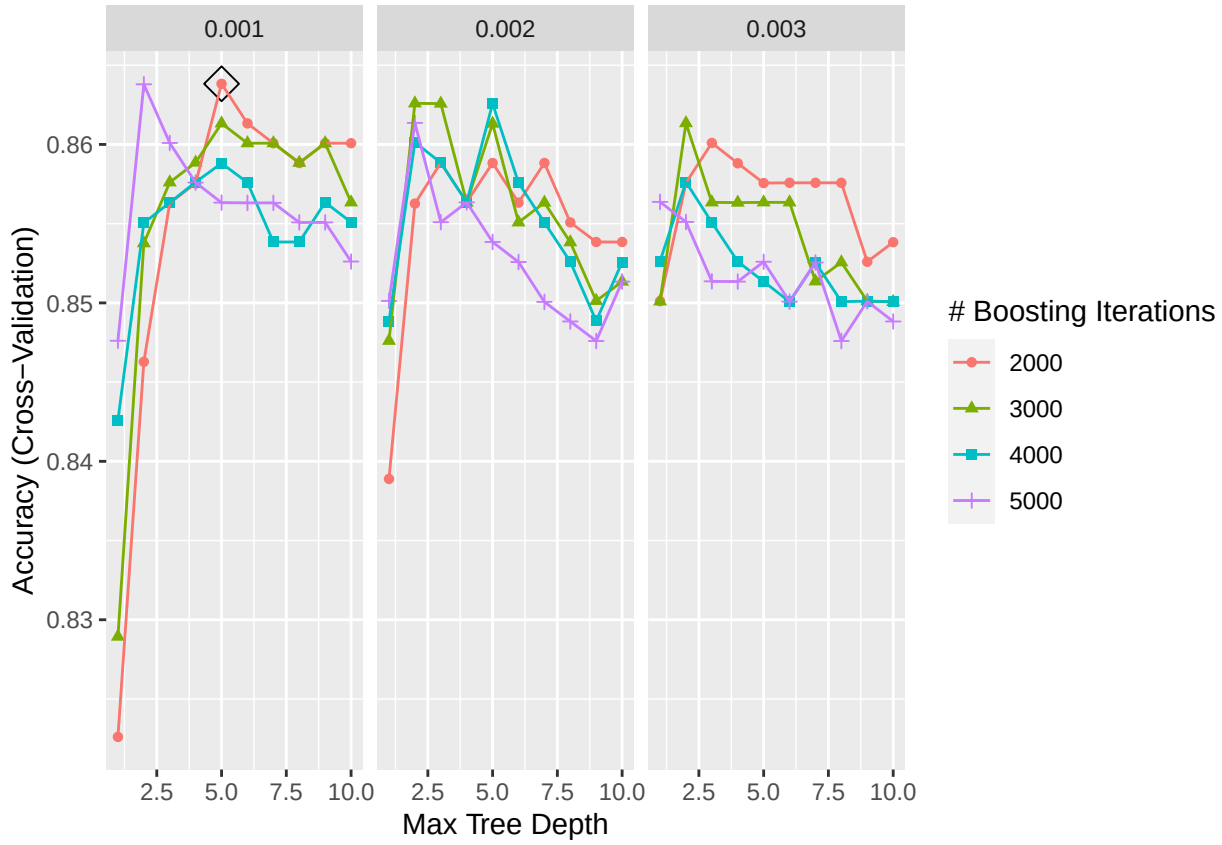
```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   Not Severe Severe
##   Not Severe        115     13
##   Severe             20     52
##
##                Accuracy : 0.835
##                  95% CI : (0.7762, 0.8836)
##     No Information Rate : 0.675
##     P-Value [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
##
```

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(1)
gbmA.fit <- train(severity ~ .,
                  data = training_data,
                  method = "gbm",
                  tuneGrid = gbmA.grid,
                  trControl = ctrl,
                  distribution = "adaboost",
                  verbose = FALSE)
gbmA.fit$bestTune
```

```
##    n.trees interaction.depth shrinkage n.minobsinnode
## 17    2000                 5     0.001              1
```

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




```
gbmA.pred <- predict(gbmA.fit, newdata = test_data)
```

```r
confusionMatrix(data = as.factor(gbmA.pred),
                reference = test_data$severity,
                positive = "Severe")
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   Not Severe Severe
##   Not Severe        124     18
##   Severe             11     47
##
##                Accuracy : 0.855
##                  95% CI : (0.7984, 0.9007)
##     No Information Rate : 0.675
##     P-Value [Acc > NIR] : 4.95e-09
##
##                   Kappa : 0.66
##
##  Mcnemar's Test P-Value : 0.2652
##
##             Sensitivity : 0.7231
##             Specificity : 0.9185
##          Pos Pred Value : 0.8103
##          Neg Pred Value : 0.8732
##              Prevalence : 0.3250
##          Detection Rate : 0.2350
##    Detection Prevalence : 0.2900
##       Balanced Accuracy : 0.8208
##
##        'Positive' Class : Severe
##
```

Compare all models

```r
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
##
## Accuracy
##               Min.   1st Qu.    Median      Mean   3rd Qu.   Max. NA's
```

```
## MARS     0.7901235 0.8276108 0.8490506 0.8501094 0.8725973 0.9250    0
## GLMN     0.8024691 0.8328704 0.8616297 0.8588447 0.8722994 0.9375    0
## SVM      0.8024691 0.8302469 0.8616297 0.8576414 0.8750000 0.9125    0
## RF       0.8024691 0.8327136 0.8813096 0.8651264 0.8871440 0.9125    0
## LDA      0.7901235 0.8043835 0.8323302 0.8388266 0.8644383 0.9250    0
## Boosting 0.8024691 0.8295886 0.8734177 0.8638293 0.8885417 0.9250    0
##
## Kappa
##                Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## MARS     0.5468904 0.6180663 0.6693116 0.6701090 0.7223409 0.8327526    0
## GLMN     0.5767472 0.6398793 0.7038020 0.6933999 0.7217518 0.8616874    0
## SVM      0.5767472 0.6339254 0.6992158 0.6889469 0.7242651 0.8092643    0
## RF       0.5636364 0.6238195 0.7358226 0.7001029 0.7485697 0.8033708    0
## LDA      0.5574230 0.5890651 0.6482706 0.6592446 0.7033939 0.8377282    0
## Boosting 0.5767472 0.6175304 0.7123710 0.6978178 0.7573462 0.8295455    0
```

```
bwplot(resamp, metric = "Accuracy")
```