Lab 20

quan le

2022-11-01

Random Forests

```
library(caret)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```
library(titanic)
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
library(randomForest)
```

```
## randomForest 4.7-1.1
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(yardstick)
```

```
## For binary classification, the first factor level is assumed to be the event.
## Use the argument `event_level = "second"` to alter this as needed.
```

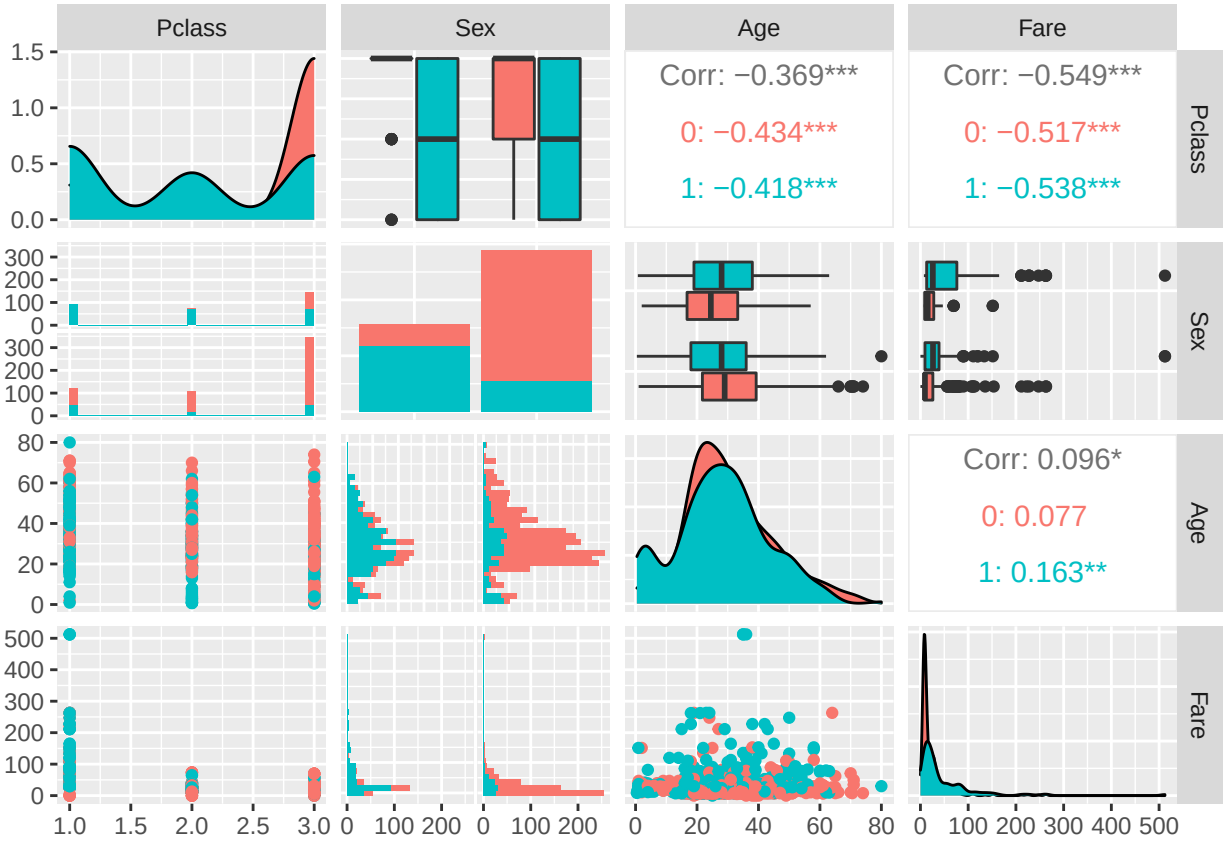
```
##
## Attaching package: 'yardstick'
```

```
## The following objects are masked from 'package:caret':
##
##     precision, recall, sensitivity, specificity
```

```
titanic_train$Survived = as.factor(titanic_train$Survived)

titanic_train %>%
  ggpairs(columns = c("Pclass",
                      "Sex",
                      "Age",
```

```
                      "Fare"),
         mapping = aes(color = Survived))
```

```
## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 177 rows containing missing values
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## Warning: Removed 177 rows containing non-finite values (stat_boxplot).
## Warning: Removed 177 rows containing missing values (geom_point).
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## Warning: Removed 177 rows containing non-finite values (stat_bin).
## Warning: Removed 177 rows containing non-finite values (stat_density).
## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 177 rows containing missing values
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## Warning: Removed 177 rows containing missing values (geom_point).
```



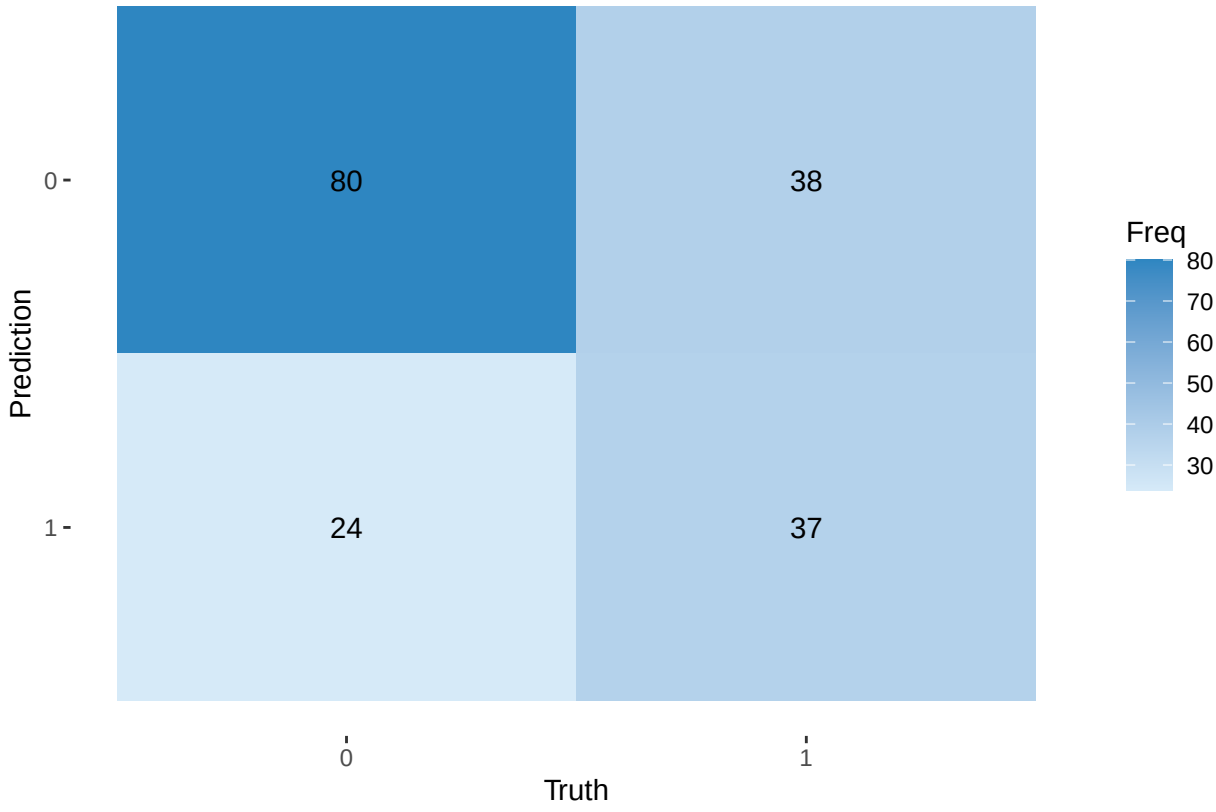

```
# Let us split into training and test sets
## 75% of the sample size
train.index <- createDataPartition(titanic_train$Survived, p = .75, list = FALSE)
train <- titanic_train[ train.index,]
test  <- titanic_train[-train.index,]
```

```
###################
# Random forest
# Need to take care of NAs
rfModel1 <- randomForest(Survived~Fare + Age,
                         data=train,
                         na.action=na.omit)

print(rfModel1)
```

```
##
## Call:
##  randomForest(formula = Survived ~ Fare + Age, data = train, na.action = na.omit)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 1
##
##         OOB estimate of  error rate: 34.21%
## Confusion matrix:
##     0   1 class.error
## 0 242  78   0.2437500
## 1 105 110   0.4883721
```

```
surv_pred <- predict(rfModel1, newdata=test, type='class')

trueAndPredFr <- data.frame(surv_pred, test$Survived)
confMat <- conf_mat(trueAndPredFr, truth=test.Survived, estimate=surv_pred)

autoplot(confMat, type = "heatmap") +
  scale_fill_gradient(low="#D6EAF8",high = "#2E86C1") +
  theme(legend.position = "right")
```

```
## Scale for 'fill' is already present. Adding another scale for 'fill', which
## will replace the existing scale.
```



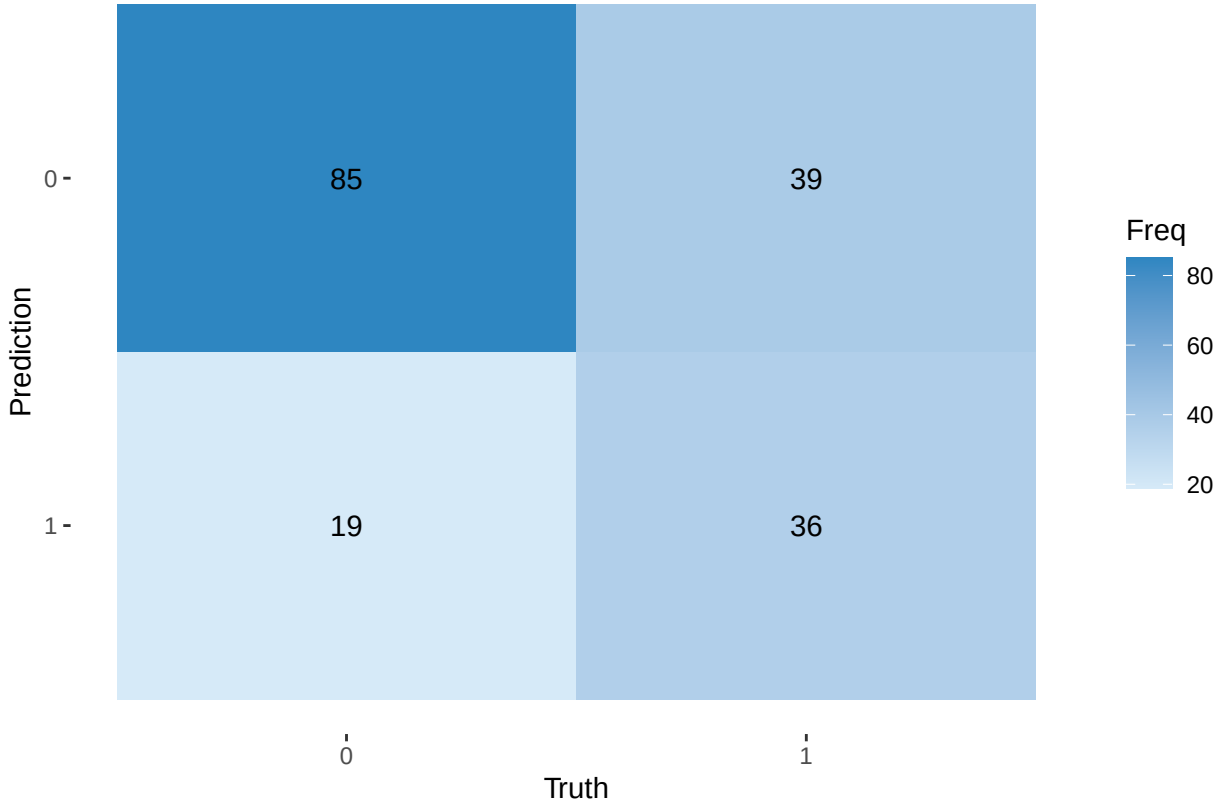

```
############################
# What is the complexity parameter here?
rfModel2 <- randomForest(Survived~Pclass + Age,
                         data=train,
                         na.action=na.omit,
                         ntree=10)

print(rfModel2)
```

```
##
## Call:
##  randomForest(formula = Survived ~ Pclass + Age, data = train,      ntree = 10, na.action = na.omit)
##                Type of random forest: classification
##                      Number of trees: 10
## No. of variables tried at each split: 1
##
##         OOB estimate of  error rate: 29.5%
## Confusion matrix:
##     0   1 class.error
## 0 262  53   0.1682540
## 1 101 106   0.4879227
```

```
surv_pred <- predict(rfModel2, newdata=test, type='class')

trueAndPredFr <- data.frame(surv_pred, test$Survived)
confMat <- conf_mat(trueAndPredFr, truth=test.Survived, estimate=surv_pred)

autoplot(confMat, type = "heatmap") +
  scale_fill_gradient(low="#D6EAF8",high = "#2E86C1") +
```

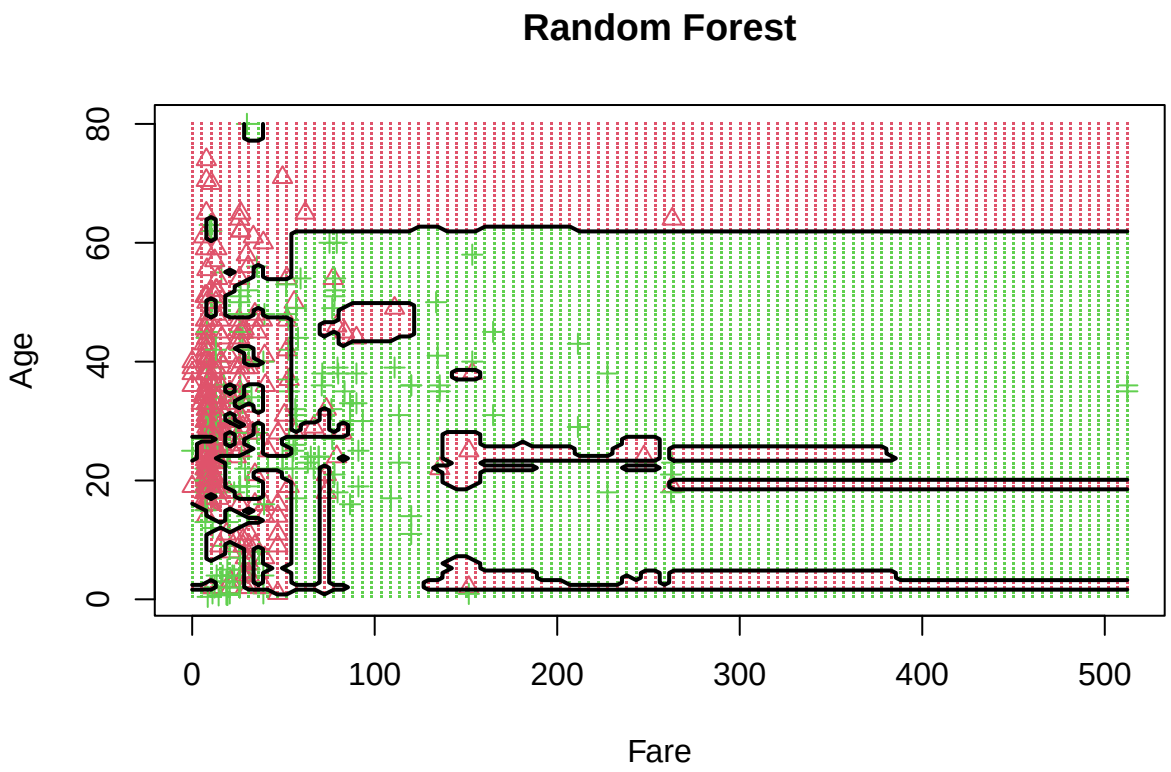
```
  theme(legend.position = "right")
```

```
## Scale for 'fill' is already present. Adding another scale for 'fill', which
## will replace the existing scale.
```

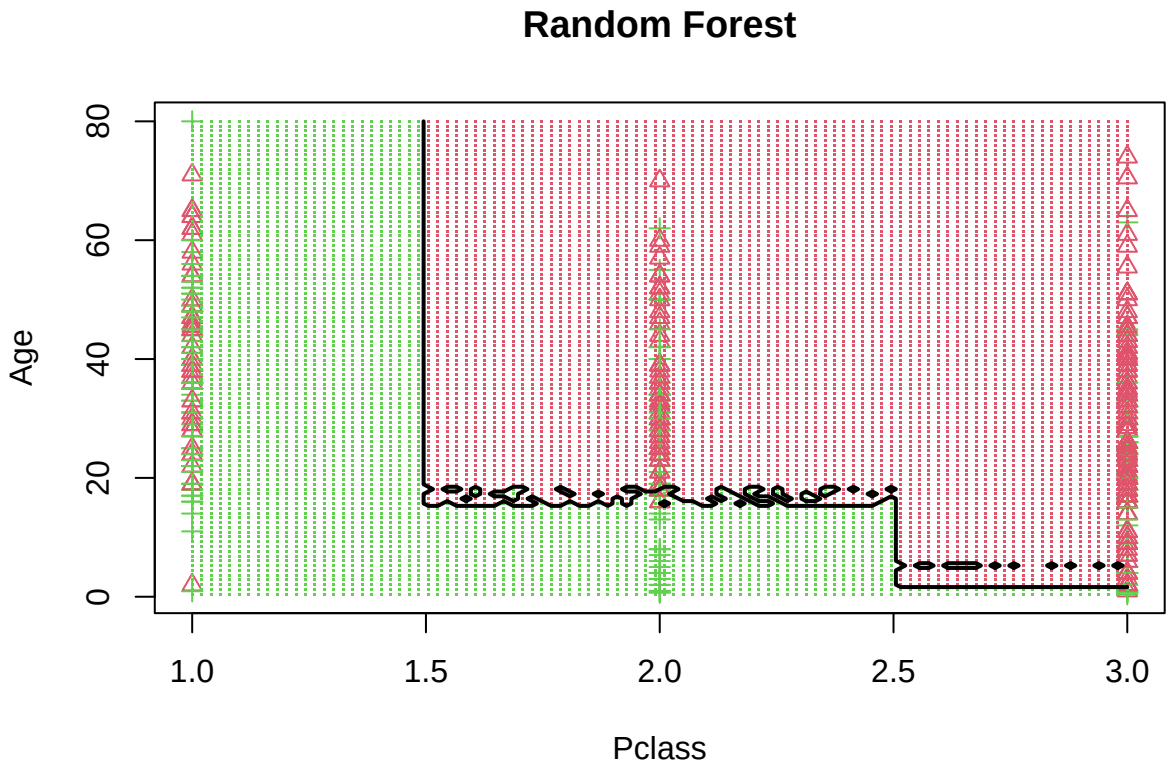



```
############################
# Decision boundary for random forest

# Function for plotting decision boundary
# Taken from https://michael.hahsler.net/SMU/EMIS7332/R/viz_classifier.html
decisionplot <- function(model, data, class = NULL, predict_type = "class",
                         resolution = 100, showgrid = TRUE, ...) {

  if(!is.null(class)) cl <- data[,class] else cl <- 1
  data <- data[,1:2]
  k <- length(unique(cl))

  plot(data, col = as.integer(cl)+1L, pch = as.integer(cl)+1L, ...)

  # make grid
  r <- sapply(data, range, na.rm = TRUE)
  xs <- seq(r[1,1], r[2,1], length.out = resolution)
  ys <- seq(r[1,2], r[2,2], length.out = resolution)
  g <- cbind(rep(xs, each=resolution), rep(ys, time = resolution))
  colnames(g) <- colnames(r)
  g <- as.data.frame(g)

  ### guess how to get class labels from predict
```

```
  ### (unfortunately not very consistent between models)
  p <- predict(model, g, type = predict_type)
  if(is.list(p)) p <- p$class
  p <- as.factor(p)

  if(showgrid) points(g, col = as.integer(p)+1L, pch = ".")

  z <- matrix(as.integer(p), nrow = resolution, byrow = TRUE)
  contour(xs, ys, z, add = TRUE, drawlabels = FALSE,
          lwd = 2, levels = (1:(k-1))+.5)

  invisible(z)
}

decisionplot(rfModel1, train[c("Fare", "Age", "Survived")], class = "Survived", main = "Random Forest")
```




```
decisionplot(rfModel2, train[c("Pclass", "Age", "Survived")], class = "Survived", main = "Random Forest
```



Feature Importance

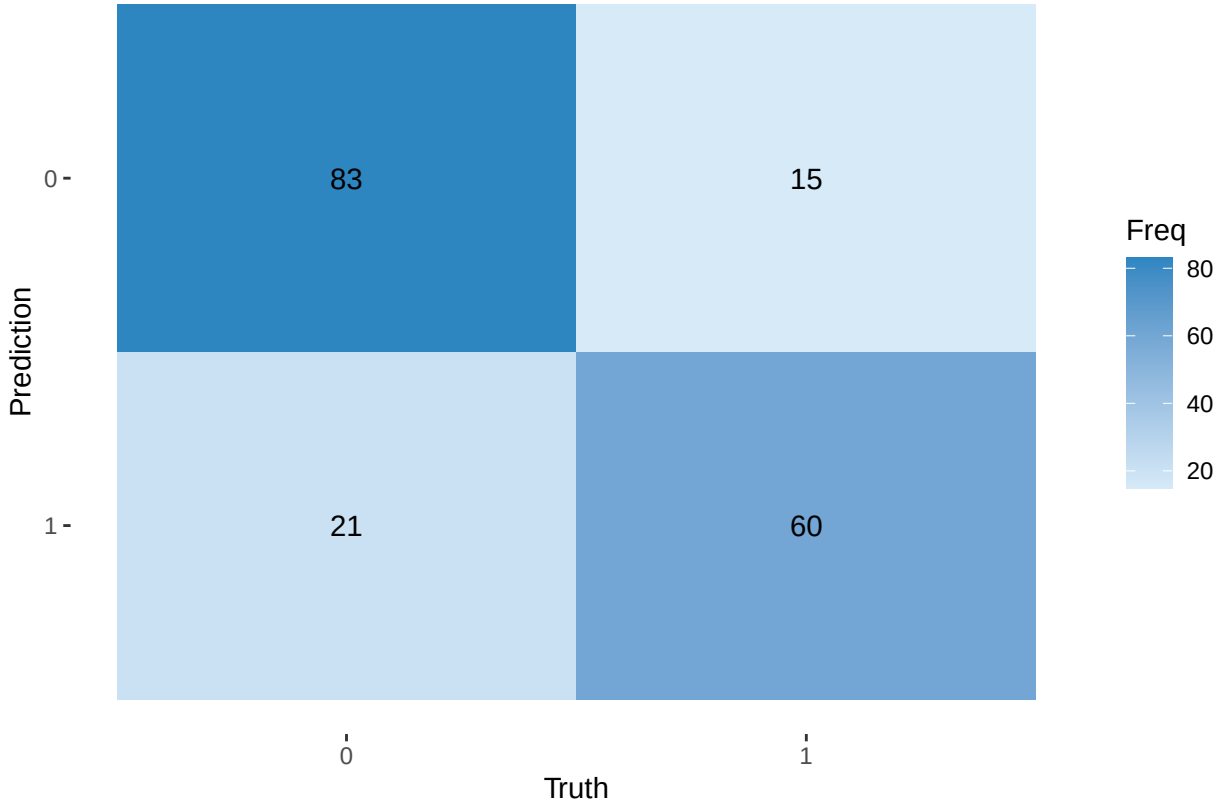
```
# What is the complexity parameter here?
rfModel3 <- randomForest(Survived~.,
                         data=train,
                         na.action=na.omit,
                         ntree=10)

print(rfModel3)
```

```
##
## Call:
##  randomForest(formula = Survived ~ ., data = train, ntree = 10,      na.action = na.omit)
##                Type of random forest: classification
##                      Number of trees: 10
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 23.86%
## Confusion matrix:
##     0   1 class.error
## 0 259  56   0.1777778
## 1  70 143   0.3286385
```

```
surv_pred <- predict(rfModel3, newdata=test, type='class')

trueAndPredFr <- data.frame(surv_pred, test$Survived)
confMat <- conf_mat(trueAndPredFr, truth=test.Survived, estimate=surv_pred)

autoplot(confMat, type = "heatmap") +
  scale_fill_gradient(low="#D6EAF8",high = "#2E86C1") +
```

```
  theme(legend.position = "right")
```

```
## Scale for 'fill' is already present. Adding another scale for 'fill', which
## will replace the existing scale.
```




```
importance(rfModel3)
```

```
##             MeanDecreaseGini
## PassengerId         23.265506
## Pclass              16.226184
## Name                24.881770
## Sex                 46.829685
## Age                 28.975196
## SibSp                7.981414
## Parch                8.783817
## Ticket              31.610045
## Fare                29.155427
## Cabin               23.706897
## Embarked             6.909356
```