problem 2

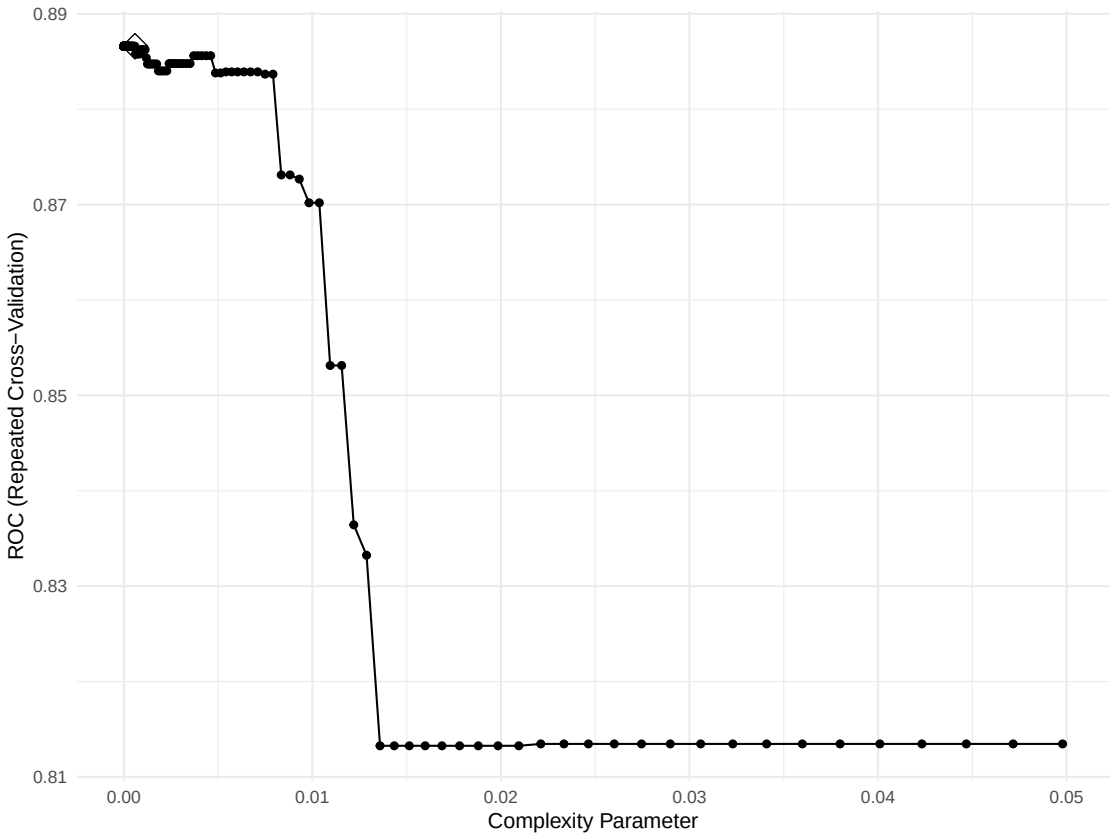
Hana Akbarnejad

4/26/2020

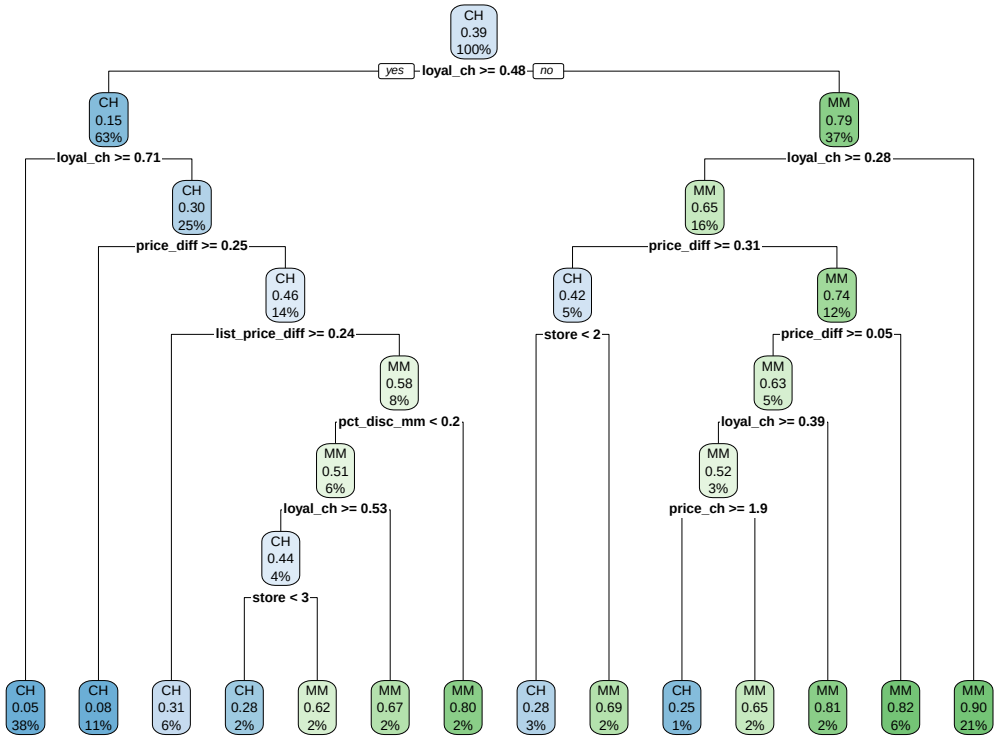
a

Fit a classification tree to the training set, with Purchase as the response and the other variables as predictors. Use cross-validation to determine the tree size and create a plot of the final tree. Predict the response on the test data. What is the test classification error rate?

```
set.seed(2020)

rpart.fit = train(purchase~., train_data,
                  method = "rpart",
                  tuneGrid = data.frame(cp = exp(seq(-30,-3, len = 500))),
                  trControl = ctrl2,
                  metric = "ROC")

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



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




```
tune_value = rpart.fit$finalModel$tuneValue


# prediction on test data
rpart_pred = predict(rpart.fit, newdata = test_data, type = "raw")
class_error = mean(rpart_pred != test_data$purchase)
```

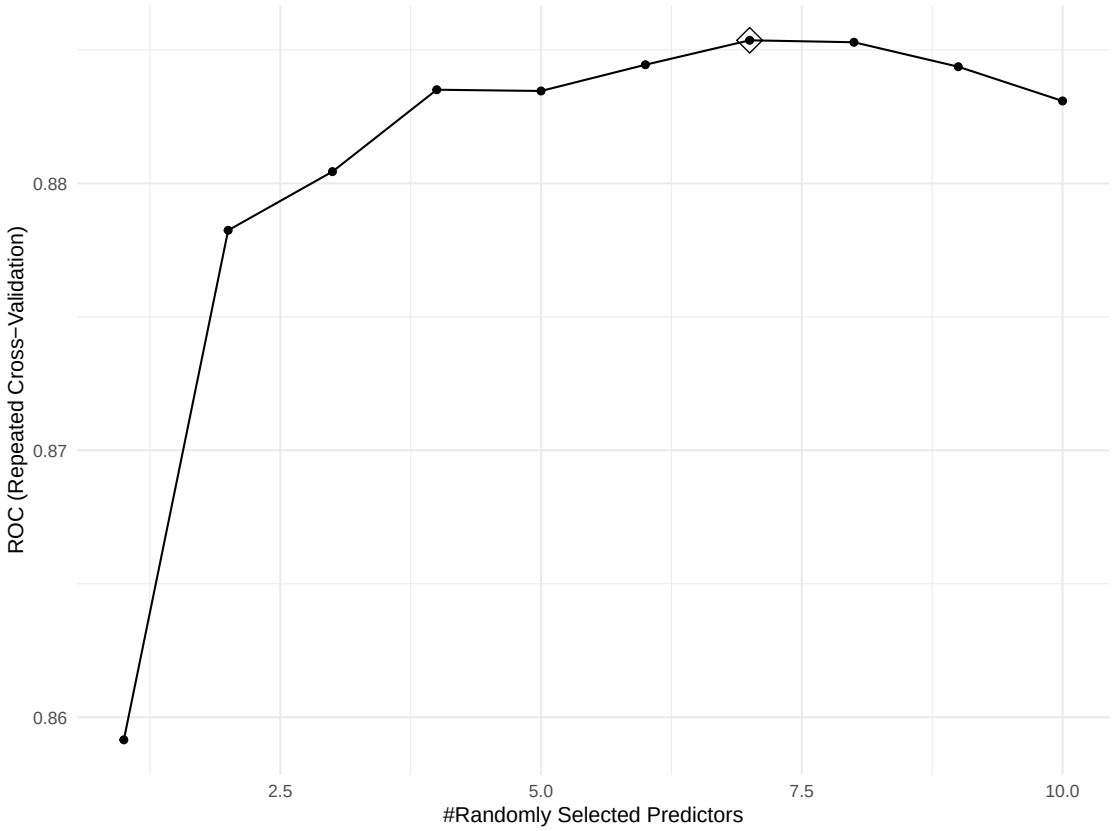
We can observe the final tree with 17 terminal nodes and the complexity (cp) of 6e-04. The test classification error rate is 21.11%.

b

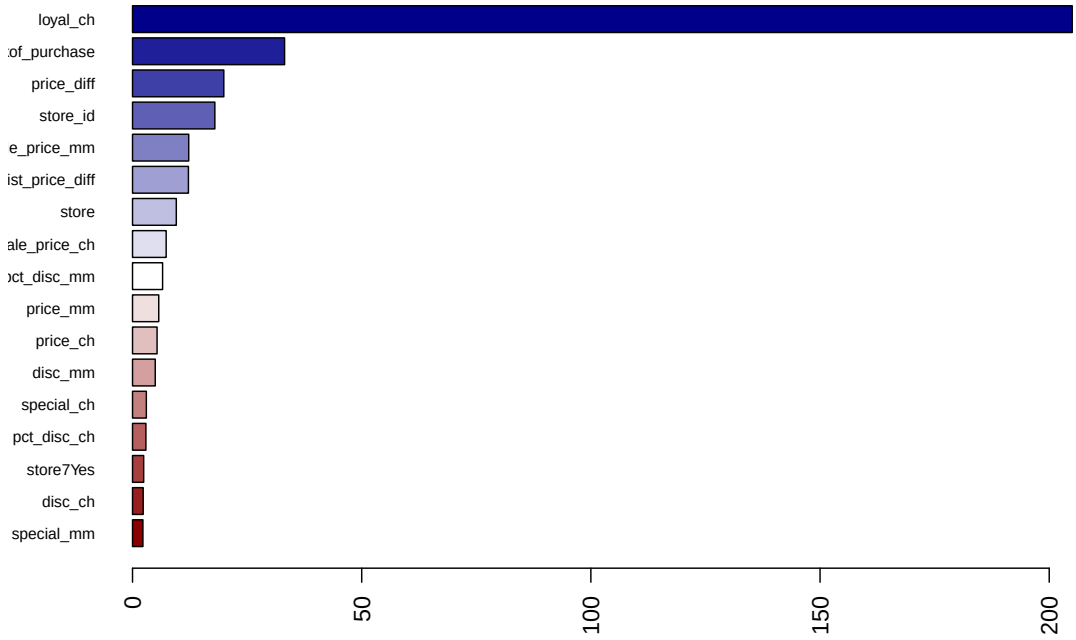
Perform random forests on the training set and report variable importance. What is the test error rate?

```
set.seed(2020)

rf.grid = expand.grid(mtry = 1:10,
                      splitrule = "gini",
                      min.node.size = 1)

rf.fit = train(purchase~., train_data,
               method = "ranger",
```

```
              tuneGrid = rf.grid,
              metric = "ROC",
              importance = "impurity",
              trControl = ctrl2)

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




```
barplot(sort(ranger::importance(rf.fit$finalModel), decreasing = FALSE),
                las = 2, horiz = TRUE, cex.names = 0.7,
                col = colorRampPalette(colors = c("darkred", "white", "darkblue"))(17))
```




```
rf.pred = predict(rf.fit, newdata = test_data, type = "raw")
rf_test_error = mean(rf.pred != test_data$purchase)
```

The plot above shows variable importance based on applying Random Forests ensambel method on the train data. We can see that the 5 most important variables are *loyal_ch*, *weekof_purchase*, *price_diff*, *store_id*, and *sale_price_mm*.

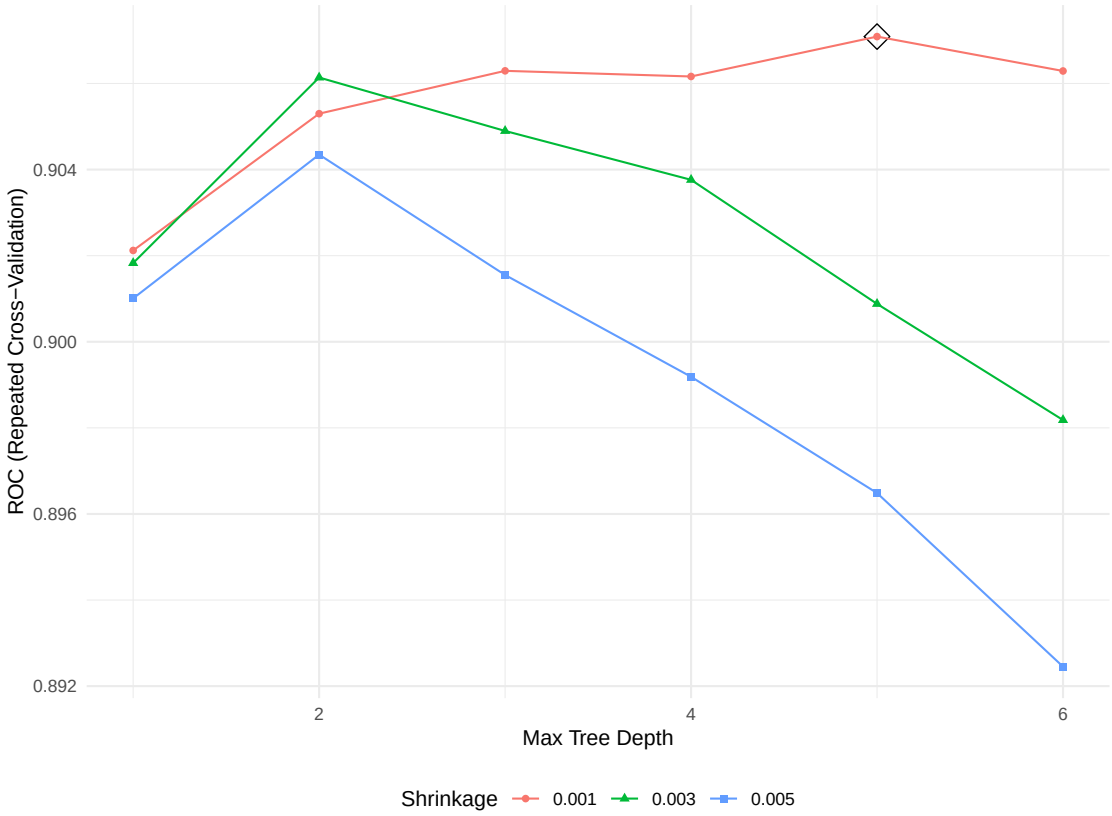
The test error rate is 21.11%.

c

Perform boosting on the training set and report variable importance. What is the test error rate?

```
class_gbm_grid = expand.grid(n.trees = 5000,
                         interaction.depth = 1:6,
                         shrinkage = c(0.001,0.003,0.005),
                         n.minobsinnode = 1)
set.seed(2020)
# Binomial loss function
class_gbm_fit = train(purchase~.,
                  data = train_data,
                  tuneGrid = class_gbm_grid,
                  trControl = ctrl2,
                  method = "gbm",
                  distribution = "bernoulli",
                  metric = "ROC",
```

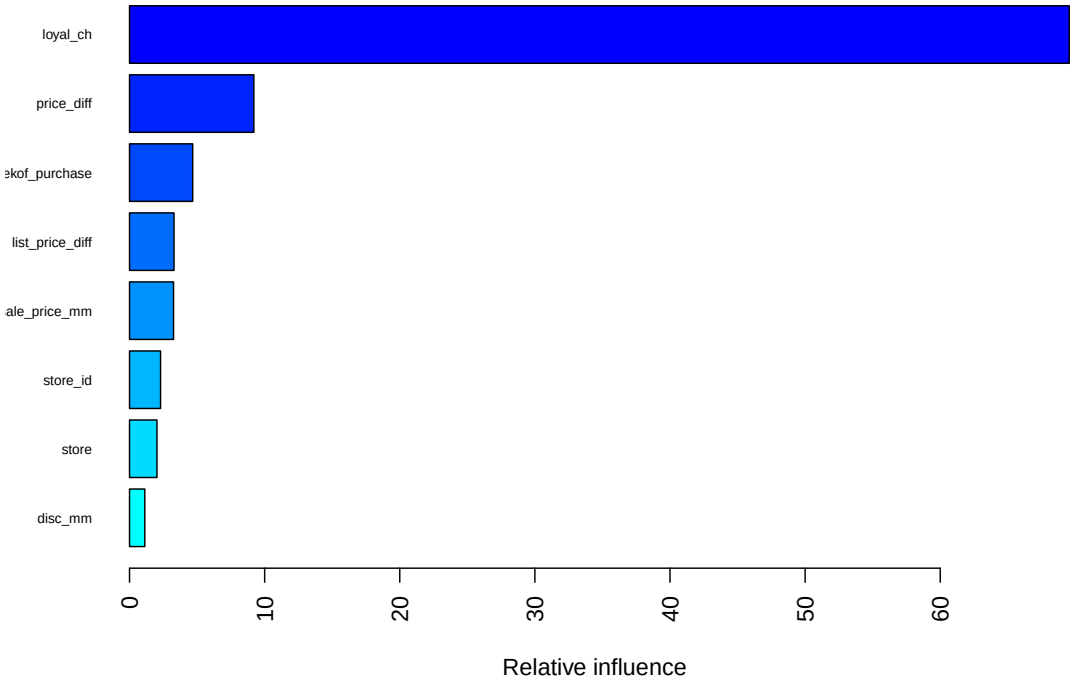
```
                  verbose = FALSE)

ggplot(class_gbm_fit, highlight = TRUE)
```




```
class_gbm_fit$finalModel$tuneValue
```

```
##   n.trees interaction.depth shrinkage n.minobsinnode
## 5    5000                 5     0.001              1
```

```
# variable importance
summary(class_gbm_fit$finalModel, las = 2, cBars = 8, cex.names = 0.6)
```




```
##                              var     rel.inf
## loyal_ch                loyal_ch 69.54702011
## price_diff            price_diff  9.19960460
## weekof_purchase weekof_purchase  4.67511873
## list_price_diff list_price_diff  3.29224523
## sale_price_mm     sale_price_mm  3.25715341
## store_id               store_id  2.29709733
## store                     store  2.03074027
## disc_mm                 disc_mm  1.13056082
## sale_price_ch     sale_price_ch  0.85709246
## price_mm               price_mm  0.76098492
## pct_disc_mm         pct_disc_mm  0.63732271
## special_ch           special_ch  0.61205135
## disc_ch                 disc_ch  0.60768982
## price_ch               price_ch  0.52948562
## special_mm           special_mm  0.37976693
## store7Yes             store7Yes  0.09329158
## pct_disc_ch         pct_disc_ch  0.09277410
```

```
# prediction and test error
class_gbm_pred = predict(class_gbm_fit, newdata = test_data, type = "raw")
gbm_test_error = mean(class_gbm_pred != test_data$purchase)
```

The plot above shows variable importance based on applying Boosting ensambel method on the train data. We can see that the 3 most important variables are *loyal_ch*, *price_diff*, and *sale_price_mm*.

The test error rate is 19.63%.