Homework 2

Chike Odenigbo

November 6, 2022

```r
get_accuracy <- function(predicted, actual){
  confusion_table = table(predicted, actual)
  TP = confusion_table[2,2]
  TN = confusion_table[1,1]
  FN = confusion_table[1,2]
  FP = confusion_table[2,1]
  accuracy = round((TP + TN) / sum(TP,FP,TN,FN), 2)
  return(accuracy)
}

get_classification_error_rate <- function(predicted, actual){
  confusion_table = table(predicted, actual)
  TP = confusion_table[2,2]
  TN = confusion_table[1,1]
  FN = confusion_table[1,2]
  FP = confusion_table[2,1]
  classification_error_rate = round((FP + FN) / sum(TP,FP,TN,FN),2)
  return(classification_error_rate)
}

get_precision <- function(predicted, actual){
  confusion_table = table(predicted, actual)
  TP = confusion_table[2,2]
  TN = confusion_table[1,1]
  FN = confusion_table[1,2]
  FP = confusion_table[2,1]
  precision = round(TP / (TP + FP), 2)
  return(precision)
}


get_sensitivity <- function(predicted, actual){
  confusion_table = table(predicted, actual)
  TP = confusion_table[2,2]
  TN = confusion_table[1,1]
  FN = confusion_table[1,2]
  FP = confusion_table[2,1]
  sensitivity = round(TP / (TP + FN), 2)
  return(sensitivity)
}

get_specificity <- function(predicted, actual){
  confusion_table = table(predicted, actual)
  TP = confusion_table[2,2]
  TN = confusion_table[1,1]
  FN = confusion_table[1,2]
  FP = confusion_table[2,1]
  specificity = round(TN / (TN + FP), 2)
  return(specificity)
}

get_f1_score <- function(predicted, actual){
  confusion_table = table(predicted, actual)
  TP = confusion_table[2,2]
  TN = confusion_table[1,1]
  FN = confusion_table[1,2]
  FP = confusion_table[2,1]

  precision = round(TP / (TP + FP), 2)
  sensitivity = round(TP / (TP + FN), 2)
  f1_score = round((2 * precision * sensitivity) / (precision + sensitivity), 2)
  return(f1_score)
}

get_false_positive_rate <- function(predicted, actual){
  confusion_table = table(predicted, actual)
  TP = confusion_table[2,2]
  TN = confusion_table[1,1]
  FN = confusion_table[1,2]
  FP = confusion_table[2,1]

  fpr = round(FP / (FP + TN), 2)
  return(fpr)
}

get_false_negative_rate <- function(predicted, actual){
  confusion_table = table(predicted, actual)
  TP = confusion_table[2,2]
  TN = confusion_table[1,1]
  FN = confusion_table[1,2]
  FP = confusion_table[2,1]

  fnr = round(FN / (FN + TP), 2)
  return(fnr)
}
```

Section 1

Question 1.1

Create a training set of 800 observations, and a test set containing the rest using the following code:

```
# ```{r pressure, echo=FALSE}
n <- nrow(OJ)
set.seed(1234)
id.train=sample(1:n,size=800)
```

```
id.test=setdiff(1:n,id.train)
OJ$Purchase = as.factor(OJ$Purchase)
OJ$Store7 = as.factor(OJ$Store7)
as.data.frame(sapply(OJ,class))
```

```
##                 sapply(OJ, class)
## Purchase                   factor
## WeekofPurchase            numeric
## StoreID                   numeric
## PriceCH                   numeric
## PriceMM                   numeric
## DiscCH                    numeric
## DiscMM                    numeric
## SpecialCH                 numeric
## SpecialMM                 numeric
## LoyalCH                   numeric
## SalePriceMM               numeric
## SalePriceCH               numeric
## PriceDiff                 numeric
## Store7                     factor
## PctDiscMM                 numeric
## PctDiscCH                 numeric
## ListPriceDiff             numeric
## STORE                     numeric
```

```
OJ.train=OJ[id.train,-3]
OJ.test=OJ[id.test,-3]
```

Question 1.2

Construct an unpruned classification tree to predict the variable purchase using the available predictors. Calculate the false positive rate, false negative rate and overall error rate of this tree on the test data (note: you can use your code from the previous assignment directly for this question).

```
# ```{r pressure, echo=FALSE}
tree <- rpart(OJ.train$Purchase ~ . ,data = OJ.train, method = 'class')
#summary(tree)

pred = predict(tree,newdata = OJ.test,type = c("class"))

print(paste0("False Positive Rate: ", get_false_positive_rate(OJ.test$Purchase,pred)))
```

```
## [1] "False Positive Rate: 0.14"
```

```
print(paste0("Error Rate: ", get_classification_error_rate(OJ.test$Purchase,pred)))
```

```
## [1] "Error Rate: 0.19"
```

```
print(paste0("False Negative Rate: ", get_false_negative_rate(OJ.test$Purchase,pred)))
```

```
## [1] "False Negative Rate: 0.27"
```

Question 1.3

Using all the data (training and test put together), use the bagging approach to do the same analysis again and compare the results of the different error rates.

```
set.seed(1)
n_pred <- ncol(OJ.train) - 1

bag.OJ <- randomForest(OJ.train$Purchase ~ .,
                           data = OJ.train,
                           mtry=n_pred,
                           importance=TRUE)

bag.pred = predict(bag.OJ,newdata = OJ.test,type = c("class"))

print(paste0("False Positive Rate: ", get_false_positive_rate(OJ.test$Purchase,bag.pred)))
```

```
## [1] "False Positive Rate: 0.15"
```

```
print(paste0("Error Rate: ", get_classification_error_rate(OJ.test$Purchase,bag.pred)))
```

```
## [1] "Error Rate: 0.21"
```

```
print(paste0("False Negative Rate: ", get_false_negative_rate(OJ.test$Purchase,bag.pred)))
```

```
## [1] "False Negative Rate: 0.31"
```

Question 1.4

Using all the data (training and test put together), use the random forest approach to redo the same analysis and compare the results of the different error rates.

```
set.seed(1)

rf.OJ <- randomForest(OJ.train$Purchase ~ .,
                           data = OJ.train,
                           #mtry=n_pred,
                           importance=TRUE)

rf.pred = predict(rf.OJ,newdata = OJ.test,type = c("class"))

print(paste0("False Positive Rate: ", get_false_positive_rate(OJ.test$Purchase,rf.pred)))
```

```
## [1] "False Positive Rate: 0.17"
```

```
print(paste0("Error Rate: ", get_classification_error_rate(OJ.test$Purchase,rf.pred)))
```

```
## [1] "Error Rate: 0.21"
```

```
print(paste0("False Negative Rate: ", get_false_negative_rate(OJ.test$Purchase,rf.pred)))
```

```
## [1] "False Negative Rate: 0.28"
```

Question 1.5

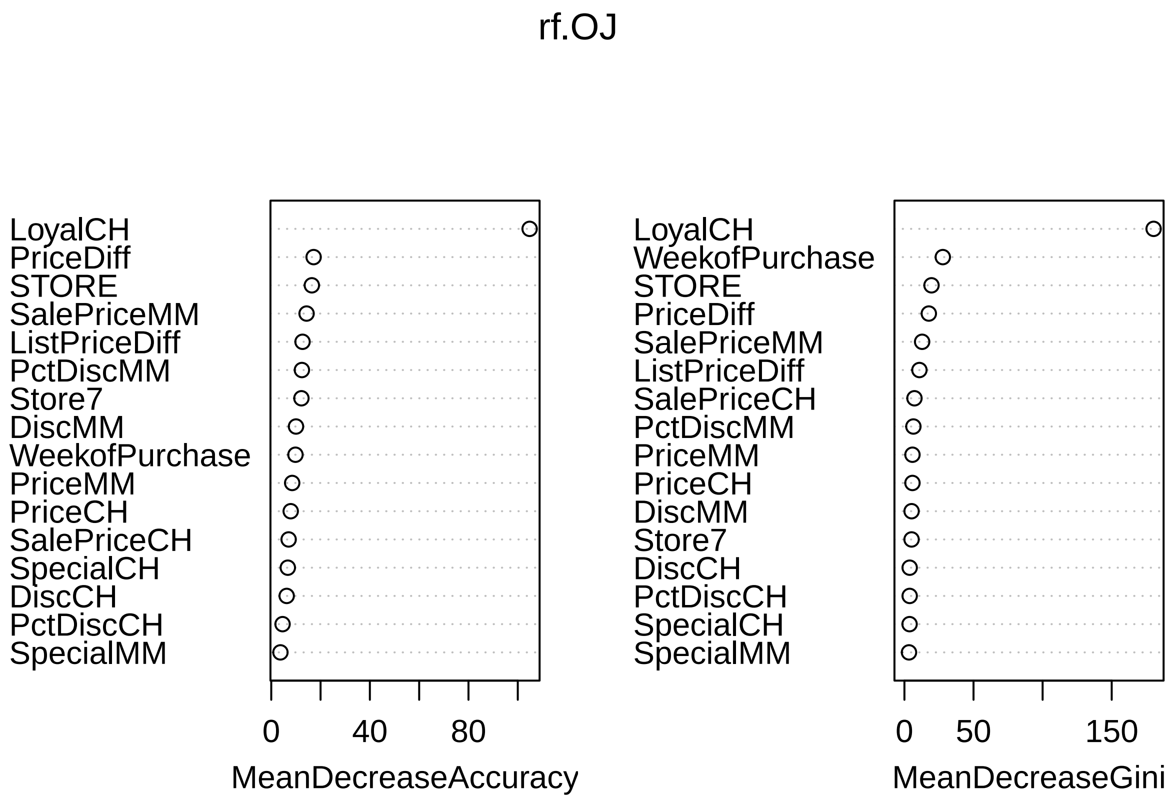
Calculate the importance of the variables in the classification tree constructed in 2) and the forest constructed in 4). Compare.

```
set.seed(1)

as.data.frame(importance(rf.OJ))
```

```
##                          CH        MM MeanDecreaseAccuracy MeanDecreaseGini
## WeekofPurchase    6.2370043  6.263470             9.821831        27.774264
## PriceCH         6.4129690  3.347700             7.894530         5.809040
## PriceMM         3.6162856  7.199457             8.496137         5.816289
## DiscCH          3.2356695  5.311119             6.259811         3.814248
## DiscMM          5.1714256  6.529393            10.040448         5.196968
## SpecialCH       2.1385953  5.812328             6.715932         3.727776
## SpecialMM      -0.4787197  4.741747             3.719254         3.312229
## LoyalCH        73.7608676 98.090066           104.804821       180.327296
## SalePriceMM     6.9341087 11.630595            14.330011        12.811005
## SalePriceCH     5.3705321  3.347173             7.054213         7.263904
## PriceDiff       9.6259636 12.248643            17.187297        17.667201
## Store7          4.4216587 12.366033            12.239986         5.174581
## PctDiscMM       7.1044541  8.672862            12.428228         6.497433
## PctDiscCH       2.3521748  4.010164             4.593497         3.812999
## ListPriceDiff   8.0987352  8.405751            12.687143        10.860433
## STORE           7.2910690 15.037911            16.467642        19.595198
```

```
varImpPlot(rf.OJ)
```



```
set.seed(1)

as.data.frame(tree$variable.importance)
```

```
##                 tree$variable.importance
## LoyalCH                       170.416755
## PriceDiff                      21.478485
## SalePriceMM                    20.474434
## ListPriceDiff                  15.244049
## PctDiscMM                         13.723186
## DiscMM                            12.957004
## PriceMM                            7.244905
## SpecialMM                          7.151039
## WeekofPurchase                     5.708685
## SpecialCH                          5.376632
## STORE                              5.077294
## PriceCH                            2.477877
## PctDiscCH                          1.722996
## DiscCH                             1.656462
## SalePriceCH                        1.369650
```

```
#varImpPlot(tree)
```

Question 1.6

Using all the data (training and test put together), use the boosting approach to do the same analysis again and compare the results of the error rates between all tested methods.

```
set.seed(1)

boost.OJ = boosting(Purchase ~ .,
                    data=OJ.train,
                    #mfinal = 100,
                    coeflearn = 'Freund')#,
                    #control=rpart.control(maxdepth=10))
boost.pred = predict(boost.OJ,newdata = OJ.test,type = c("class"))

print(boost.pred)
```

```
## $formula
## Purchase ~ .
##
## $votes
##             [,1]      [,2]
##   [1,] 26.988109 17.935172
##   [2,] 26.278240 18.645042
##   [3,] 28.569733 16.353549
##   [4,] 29.231073 15.692209
##   [5,] 29.047458 15.875823
##   [6,] 24.859462 20.063820
##   [7,] 27.778991 17.144290
##   [8,] 26.160923 18.762358
##   [9,] 32.280933 12.642348
##  [10,] 28.122220 16.801061
##  [11,] 25.176539 19.746742
##  [12,] 31.478518 13.444763
##  [13,] 33.807036 11.116245
##  [14,] 33.385855 11.537426
##  [15,] 27.713492 17.209789
##  [16,] 24.867392 20.055889
##  [17,] 24.757789 20.165493
##  [18,] 26.886510 18.036771
##  [19,] 22.820437 22.102844
##  [20,] 22.751186 22.172095
## [21,] 22.087702 22.835579
## [22,] 29.431332 15.491949
## [23,] 26.418127 18.505154
## [24,] 24.077486 20.845795
## [25,] 29.572247 15.351035
## [26,] 20.464208 24.459074
## [27,] 20.149681 24.773600
## [28,] 20.019135 24.904147
## [29,] 27.364655 17.558626
## [30,] 22.529830 22.393451
## [31,] 26.887398 18.035883
## [32,] 29.167827 15.755455
## [33,] 29.412501 15.510781
## [34,] 27.009738 17.913543
## [35,] 31.185581 13.737700
## [36,] 29.948379 14.974902
## [37,] 21.818304 23.104977
## [38,] 24.858046 20.065236
## [39,] 31.423671 13.499610
## [40,] 28.141141 16.782141
## [41,] 28.439899 16.483382
## [42,] 21.895139 23.028142
## [43,] 22.973984 21.949297
## [44,] 17.690840 27.232441
## [45,] 22.171483 22.751798
## [46,] 29.442413 15.480868
## [47,] 30.223678 14.699604
## [48,] 26.886510 18.036771
## [49,] 24.251687 20.671595
## [50,] 32.587111 12.336170
## [51,] 31.198783 13.724498
## [52,] 31.048586 13.874695
## [53,] 28.171564 16.751717
## [54,] 27.801036 17.122246
## [55,] 30.269148 14.654133
## [56,] 29.960095 14.963186
## [57,] 31.666185 13.257096
## [58,] 31.791142 13.132139
## [59,] 31.696320 13.226961
## [60,] 29.529083 15.394198
## [61,] 30.111821 14.811461
## [62,] 30.426722 14.496559
## [63,] 31.034125 13.889156
## [64,] 17.736948 27.186334
## [65,] 23.994526 20.928755
## [66,] 23.494137 21.429145
## [67,] 25.538772 19.384509
## [68,] 28.077690 16.845591
## [69,] 28.543877 16.379404
## [70,] 31.669581 13.253700
## [71,] 32.952685 11.970596
## [72,] 25.386795 19.536487
## [73,] 20.966811 23.956471
## [74,] 24.495321 20.427960
##  [75,] 28.272142 16.651139
##  [76,] 15.339919 29.583362
##  [77,] 14.519771 30.403510
##  [78,] 16.743730 28.179551
##  [79,]  9.663242 35.260040
##  [80,] 21.864634 23.058647
##  [81,] 21.519313 23.403968
##  [82,] 20.385056 24.538225
##  [83,] 20.237911 24.685370
##  [84,] 26.251963 18.671318
##  [85,] 28.429980 16.493301
##  [86,] 28.476096 16.447185
##  [87,] 35.095039  9.828243
##  [88,] 28.586409 16.336873
##  [89,] 23.221018 21.702263
##  [90,] 20.576708 24.346573
##  [91,] 24.856724 20.066557
##  [92,] 22.522734 22.400548
##  [93,] 21.516498 23.406783
##  [94,] 25.675106 19.248176
##  [95,] 21.496247 23.427035
##  [96,] 20.738850 24.184432
##  [97,] 33.272101 11.651180
##  [98,] 20.001135 24.922147
##  [99,] 19.942848 24.980433
## [100,] 25.097807 19.825474
## [101,] 19.770007 25.153274
## [102,] 19.098258 25.825023
## [103,] 15.005350 29.917931
## [104,] 19.114733 25.808548
## [105,] 15.136937 29.786345
## [106,] 21.403400 23.519881
## [107,] 14.497688 30.425593
## [108,] 21.117672 23.805610
## [109,] 20.056135 24.867146
## [110,] 23.159257 21.764024
## [111,] 20.604459 24.318823
## [112,] 26.192243 18.731038
## [113,] 29.647082 15.276200
## [114,] 25.096209 19.827072
## [115,] 28.127017 16.796264
## [116,] 17.041861 27.881420
## [117,] 30.061292 14.861989
## [118,] 33.358603 11.564679
## [119,] 29.469235 15.454047
## [120,] 14.348673 30.574608
## [121,] 28.470083 16.453198
## [122,] 25.950880 18.972402
## [123,] 21.076268 23.847013
## [124,] 26.514981 18.408301
## [125,] 27.026229 17.897052
## [126,] 25.753571 19.169710
## [127,] 31.951050 12.972231
## [128,] 30.859299 14.063982
## [129,] 20.067219 24.856063
## [130,] 24.814489 20.108792
## [131,] 26.846595 18.076686
## [132,] 25.720956 19.202325
## [133,] 32.245204 12.678078
## [134,] 34.217438 10.705843
## [135,] 19.361249 25.562032
## [136,] 16.768932 28.154349
## [137,] 16.873376 28.049905
## [138,] 17.693737 27.229544
## [139,] 20.276853 24.646428
## [140,] 16.963911 27.959370
## [141,] 25.452683 19.470598
## [142,] 22.973984 21.949297
## [143,] 29.607493 15.315789
## [144,] 19.446832 25.476449
## [145,] 18.684747 26.238534
## [146,] 27.665139 17.258143
## [147,] 26.906086 18.017195
## [148,] 24.771509 20.151773
## [149,] 24.313494 20.609787
## [150,] 30.719951 14.203330
## [151,] 32.389987 12.533295
## [152,] 18.549946 26.373336
## [153,] 29.728451 15.194830
## [154,] 31.869267 13.054014
## [155,] 27.073143 17.850138
## [156,] 27.458454 17.464827
## [157,] 23.979640 20.943642
## [158,] 26.852862 18.070420
## [159,] 18.327842 26.595439
## [160,] 27.657003 17.266278
## [161,] 30.172963 14.750319
## [162,] 27.132196 17.791085
## [163,] 17.150962 27.772319
## [164,] 17.220277 27.703004
## [165,] 14.886618 30.036663
## [166,] 16.171542 28.751740
## [167,] 18.062976 26.860305
## [168,] 17.974966 26.948315
## [169,] 17.183181 27.740100
## [170,] 15.456897 29.466384
## [171,] 16.219489 28.703792
## [172,] 16.743730 28.179551
## [173,] 16.604263 28.319018
## [174,] 18.318208 26.605073
## [175,] 20.322022 24.601259
## [176,] 28.309221 16.614060
## [177,] 30.660099 14.263183
## [178,] 19.520460 25.402821
## [179,] 25.415127 19.508154
## [180,] 20.237530 24.685752
## [181,] 22.973984 21.949297
## [182,] 23.293416 21.629865
## [183,] 19.356220 25.567062
## [184,] 23.157986 21.765295
## [185,] 18.776845 26.146437
## [186,] 20.415493 24.507788
## [187,] 28.253104 16.670177
## [188,] 29.625345 15.297936
## [189,] 21.790850 23.132431
## [190,] 27.715360 17.207921
## [191,] 19.599763 25.323519
## [192,] 20.407861 24.515420
## [193,] 13.624485 31.298796
## [194,] 29.123141 15.800141
## [195,] 27.401002 17.522279
## [196,] 18.835447 26.087834
## [197,] 19.751918 25.171364
## [198,] 22.315936 22.607345
## [199,] 18.776845 26.146437
## [200,] 18.908103 26.015178
## [201,] 14.812665 30.110617
## [202,] 19.849233 25.074049
## [203,] 14.802373 30.120908
## [204,] 26.380690 18.542592
## [205,] 26.568233 18.355048
## [206,] 33.707680 11.215601
## [207,] 24.984286 19.938995
## [208,] 19.550431 25.372850
## [209,] 25.431735 19.491546
## [210,] 19.900006 25.023276
## [211,] 20.237911 24.685370
## [212,] 12.931778 31.991503
## [213,] 12.803911 32.119371
## [214,] 26.645443 18.277838
## [215,] 31.218747 13.704535
## [216,] 20.056135 24.867146
## [217,] 31.565800 13.357481
## [218,] 28.181532 16.741750
## [219,] 20.186892 24.736389
## [220,] 22.961861 21.961420
## [221,] 26.034866 18.888415
## [222,] 27.051183 17.872098
## [223,] 23.996907 20.926374
## [224,] 29.460084 15.463198
## [225,] 32.256940 12.666342
## [226,] 33.105952 11.817330
## [227,] 29.721397 15.201884
## [228,] 18.177190 26.746092
## [229,] 26.392910 18.530372
## [230,] 26.976739 17.946543
## [231,] 24.110133 20.813148
## [232,] 29.993247 14.930034
## [233,] 23.681981 21.241300
## [234,] 15.212499 29.710783
## [235,] 19.021559 25.901722
## [236,] 13.722718 31.200564
## [237,] 13.930475 30.992806
## [238,] 14.164122 30.759159
## [239,] 15.783610 29.139671
## [240,] 24.115018 20.808263
## [241,] 24.771504 20.151777
## [242,] 19.359352 25.563929
## [243,] 23.324965 21.598316
## [244,] 23.852542 21.070739
## [245,] 24.937836 19.985445
## [246,] 23.579880 21.343402
## [247,] 21.988601 22.934681
## [248,] 14.794677 30.128604
## [249,] 13.060865 31.862416
## [250,] 15.465286 29.457995
## [251,] 22.262641 22.660640
## [252,] 17.789111 27.134170
## [253,] 16.917236 28.006045
## [254,] 25.269989 19.653292
## [255,] 17.331459 27.591823
## [256,] 24.080202 20.843080
## [257,] 26.665272 18.258009
## [258,] 26.312940 18.610341
## [259,] 25.853300 19.069982
## [260,] 29.746146 15.177135
## [261,] 24.857037 20.066244
## [262,] 25.503112 19.420169
## [263,] 30.684617 14.238664
## [264,] 25.892319 19.030962
## [265,] 26.238705 18.684576
## [266,] 27.137071 17.786210
## [267,] 25.269989 19.653292
## [268,] 17.982801 26.940480
## [269,] 19.322754 25.600527
## [270,] 29.438855 15.484426
##
## $prob
##             [,1]      [,2]
##   [1,] 0.6007600 0.3992400
##   [2,] 0.5849582 0.4150418
##   [3,] 0.6359672 0.3640328
##   [4,] 0.6506887 0.3493113
##   [5,] 0.6466014 0.3533986
##   [6,] 0.5533759 0.4466241
##   [7,] 0.6183651 0.3816349
##   [8,] 0.5823467 0.4176533
##   [9,] 0.7185791 0.2814209
##  [10,] 0.6260055 0.3739945
##  [11,] 0.5604341 0.4395659
##  [12,] 0.7007172 0.2992828
##  [13,] 0.7525505 0.2474495
##  [14,] 0.7431749 0.2568251
##  [15,] 0.6169071 0.3830929
##  [16,] 0.5535524 0.4464476
##  [17,] 0.5511127 0.4488873
## [18,] 0.5984984 0.4015016
## [19,] 0.5079869 0.4920131
## [20,] 0.5064453 0.4935547
## [21,] 0.4916761 0.5083239
## [22,] 0.6551465 0.3448535
## [23,] 0.5880721 0.4119279
## [24,] 0.5359690 0.4640310
## [25,] 0.6582833 0.3417167
## [26,] 0.4555368 0.5444632
## [27,] 0.4485354 0.5514646
## [28,] 0.4456294 0.5543706
## [29,] 0.6091420 0.3908580
## [30,] 0.5015179 0.4984821
## [31,] 0.5985181 0.4014819
## [32,] 0.6492809 0.3507191
## [33,] 0.6547273 0.3452727
## [34,] 0.6012414 0.3987586
## [35,] 0.6941964 0.3058036
## [36,] 0.6666561 0.3333439
## [37,] 0.4856792 0.5143208
## [38,] 0.5533444 0.4466556
## [39,] 0.6994963 0.3005037
## [40,] 0.6264266 0.3735734
## [41,] 0.6330771 0.3669229
## [42,] 0.4873896 0.5126104
## [43,] 0.5114049 0.4885951
## [44,] 0.3938012 0.6061988
## [45,] 0.4935410 0.5064590
## [46,] 0.6553932 0.3446068
## [47,] 0.6727843 0.3272157
## [48,] 0.5984984 0.4015016
## [49,] 0.5398467 0.4601533
## [50,] 0.7253947 0.2746053
## [51,] 0.6944903 0.3055097
## [52,] 0.6911469 0.3088531
## [53,] 0.6271039 0.3728961
## [54,] 0.6188559 0.3811441
## [55,] 0.6737965 0.3262035
## [56,] 0.6669169 0.3330831
## [57,] 0.7048947 0.2951053
## [58,] 0.7076763 0.2923237
## [59,] 0.7055656 0.2944344
## [60,] 0.6573225 0.3426775
## [61,] 0.6702943 0.3297057
## [62,] 0.6773041 0.3226959
## [63,] 0.6908250 0.3091750
## [64,] 0.3948275 0.6051725
## [65,] 0.5341223 0.4658777
## [66,] 0.5229835 0.4770165
## [67,] 0.5684975 0.4315025
## [68,] 0.6250142 0.3749858
## [69,] 0.6353916 0.3646084
## [70,] 0.7049703 0.2950297
## [71,] 0.7335325 0.2664675
##  [72,] 0.5651144 0.4348856
##  [73,] 0.4667248 0.5332752
##  [74,] 0.5452701 0.4547299
##  [75,] 0.6293428 0.3706572
##  [76,] 0.3414692 0.6585308
##  [77,] 0.3232126 0.6767874
##  [78,] 0.3727183 0.6272817
##  [79,] 0.2151054 0.7848946
##  [80,] 0.4867105 0.5132895
##  [81,] 0.4790236 0.5209764
##  [82,] 0.4537749 0.5462251
##  [83,] 0.4504994 0.5495006
##  [84,] 0.5843732 0.4156268
##  [85,] 0.6328563 0.3671437
##  [86,] 0.6338828 0.3661172
##  [87,] 0.7812216 0.2187784
##  [88,] 0.6363384 0.3636616
##  [89,] 0.5169039 0.4830961
##  [90,] 0.4580411 0.5419589
##  [91,] 0.5533150 0.4466850
##  [92,] 0.5013599 0.4986401
##  [93,] 0.4789610 0.5210390
##  [94,] 0.5715323 0.4284677
##  [95,] 0.4785102 0.5214898
##  [96,] 0.4616504 0.5383496
##  [97,] 0.7406427 0.2593573
##  [98,] 0.4452287 0.5547713
##  [99,] 0.4439312 0.5560688
## [100,] 0.5586815 0.4413185
## [101,] 0.4400838 0.5599162
## [102,] 0.4251305 0.5748695
## [103,] 0.3340217 0.6659783
## [104,] 0.4254973 0.5745027
## [105,] 0.3369508 0.6630492
## [106,] 0.4764434 0.5235566
## [107,] 0.3227210 0.6772790
## [108,] 0.4700830 0.5299170
## [109,] 0.4464530 0.5535470
## [110,] 0.5155291 0.4844709
## [111,] 0.4586588 0.5413412
## [112,] 0.5830439 0.4169561
## [113,] 0.6599492 0.3400508
## [114,] 0.5586459 0.4413541
## [115,] 0.6261123 0.3738877
## [116,] 0.3793548 0.6206452
## [117,] 0.6691696 0.3308304
## [118,] 0.7425683 0.2574317
## [119,] 0.6559903 0.3440097
## [120,] 0.3194039 0.6805961
## [121,] 0.6337490 0.3662510
## [122,] 0.5776711 0.4223289
## [123,] 0.4691614 0.5308386
## [124,] 0.5902280 0.4097720
## [125,] 0.6016085 0.3983915
## [126,] 0.5732789 0.4267211
## [127,] 0.7112359 0.2887641
## [128,] 0.6869333 0.3130667
## [129,] 0.4466998 0.5533002
## [130,] 0.5523748 0.4476252
## [131,] 0.5976098 0.4023902
## [132,] 0.5725529 0.4274471
## [133,] 0.7177838 0.2822162
## [134,] 0.7616861 0.2383139
## [135,] 0.4309848 0.5690152
## [136,] 0.3732793 0.6267207
## [137,] 0.3756043 0.6243957
## [138,] 0.3938656 0.6061344
## [139,] 0.4513663 0.5486337
## [140,] 0.3776196 0.6223804
## [141,] 0.5665811 0.4334189
## [142,] 0.5114049 0.4885951
## [143,] 0.6590679 0.3409321
## [144,] 0.4328898 0.5671102
## [145,] 0.4159257 0.5840743
## [146,] 0.6158308 0.3841692
## [147,] 0.5989341 0.4010659
## [148,] 0.5514181 0.4485819
## [149,] 0.5412226 0.4587774
## [150,] 0.6838314 0.3161686
## [151,] 0.7210067 0.2789933
## [152,] 0.4129250 0.5870750
## [153,] 0.6617605 0.3382395
## [154,] 0.7094154 0.2905846
## [155,] 0.6026528 0.3973472
## [156,] 0.6112299 0.3887701
## [157,] 0.5337909 0.4662091
## [158,] 0.5977493 0.4022507
## [159,] 0.4079809 0.5920191
## [160,] 0.6156497 0.3843503
## [161,] 0.6716554 0.3283446
## [162,] 0.6039674 0.3960326
## [163,] 0.3817834 0.6182166
## [164,] 0.3833263 0.6166737
## [165,] 0.3313787 0.6686213
## [166,] 0.3599813 0.6400187
## [167,] 0.4020850 0.5979150
## [168,] 0.4001259 0.5998741
## [169,] 0.3825006 0.6174994
## [170,] 0.3440732 0.6559268
## [171,] 0.3610486 0.6389514
## [172,] 0.3727183 0.6272817
## [173,] 0.3696138 0.6303862
## [174,] 0.4077665 0.5922335
## [175,] 0.4523717 0.5476283
## [176,] 0.6301682 0.3698318
## [177,] 0.6824991 0.3175009
## [178,] 0.4345288 0.5654712
## [179,] 0.5657451 0.4342549
## [180,] 0.4504909 0.5495091
## [181,] 0.5114049 0.4885951
## [182,] 0.5185155 0.4814845
## [183,] 0.4308728 0.5691272
## [184,] 0.5155008 0.4844992
## [185,] 0.4179758 0.5820242
## [186,] 0.4544524 0.5455476
## [187,] 0.6289190 0.3710810
## [188,] 0.6594653 0.3405347
## [189,] 0.4850681 0.5149319
## [190,] 0.6169487 0.3830513
## [191,] 0.4362941 0.5637059
## [192,] 0.4542825 0.5457175
## [193,] 0.3032834 0.6967166
## [194,] 0.6482861 0.3517139
## [195,] 0.6099510 0.3900490
## [196,] 0.4192803 0.5807197
## [197,] 0.4396811 0.5603189
## [198,] 0.4967566 0.5032434
## [199,] 0.4179758 0.5820242
## [200,] 0.4208976 0.5791024
## [201,] 0.3297325 0.6702675
## [202,] 0.4418473 0.5581527
## [203,] 0.3295034 0.6704966
## [204,] 0.5872387 0.4127613
## [205,] 0.5914135 0.4085865
## [206,] 0.7503388 0.2496612
## [207,] 0.5561545 0.4438455
## [208,] 0.4351960 0.5648040
## [209,] 0.5661148 0.4338852
## [210,] 0.4429776 0.5570224
## [211,] 0.4504994 0.5495006
## [212,] 0.2878636 0.7121364
## [213,] 0.2850173 0.7149827
## [214,] 0.5931322 0.4068678
## [215,] 0.6949347 0.3050653
## [216,] 0.4464530 0.5535470
## [217,] 0.7026602 0.2973398
## [218,] 0.6273258 0.3726742
## [219,] 0.4493637 0.5506363
## [220,] 0.5111350 0.4888650
## [221,] 0.5795406 0.4204594
## [222,] 0.6021640 0.3978360
## [223,] 0.5341753 0.4658247
## [224,] 0.6557866 0.3442134
## [225,] 0.7180450 0.2819550
## [226,] 0.7369442 0.2630558
## [227,] 0.6616034 0.3383966
## [228,] 0.4046274 0.5953726
## [229,] 0.5875107 0.4124893
## [230,] 0.6005069 0.3994931
## [231,] 0.5366957 0.4633043
## [232,] 0.6676549 0.3323451
## [233,] 0.5271650 0.4728350
## [234,] 0.3386328 0.6613672
## [235,] 0.4234232 0.5765768
## [236,] 0.3054701 0.6945299
## [237,] 0.3100948 0.6899052
## [238,] 0.3152958 0.6847042
## [239,] 0.3513459 0.6486541
## [240,] 0.5368045 0.4631955
## [241,] 0.5514180 0.4485820
## [242,] 0.4309425 0.5690575
## [243,] 0.5192178 0.4807822
## [244,] 0.5309617 0.4690383
## [245,] 0.5551205 0.4448795
## [246,] 0.5248922 0.4751078
## [247,] 0.4894700 0.5105300
## [248,] 0.3293321 0.6706679
## [249,] 0.2907371 0.7092629
## [250,] 0.3442599 0.6557401
## [251,] 0.4955702 0.5044298
## [252,] 0.3959887 0.6040113
## [253,] 0.3765806 0.6234194
## [254,] 0.5625143 0.4374857
## [255,] 0.3858013 0.6141987
## [256,] 0.5360294 0.4639706
## [257,] 0.5935736 0.4064264
## [258,] 0.5857306 0.4142694
## [259,] 0.5754989 0.4245011
## [260,] 0.6621544 0.3378456
## [261,] 0.5533219 0.4466781
## [262,] 0.5677037 0.4322963
## [263,] 0.6830449 0.3169551
## [264,] 0.5763675 0.4236325
## [265,] 0.5840781 0.4159219
## [266,] 0.6040759 0.3959241
## [267,] 0.5625143 0.4374857
## [268,] 0.4003003 0.5996997
## [269,] 0.4301278 0.5698722
## [270,] 0.6553140 0.3446860
##
## $class
##   [1] "CH" "CH" "CH" "CH" "CH" "CH" "CH" "CH" "CH" "CH" "CH" "CH" "CH" "CH" "CH"
##  [16] "CH" "CH" "CH" "CH" "CH" "MM" "CH" "CH" "CH" "CH" "MM" "MM" "MM" "CH" "CH"
##  [31] "CH" "CH" "CH" "CH" "CH" "CH" "MM" "CH" "CH" "CH" "CH" "MM" "CH" "MM" "MM"
##  [46] "CH" "CH" "CH" "CH" "CH" "CH" "CH" "CH" "CH" "CH" "CH" "CH" "CH" "CH" "CH"
##  [61] "CH" "CH" "CH" "MM" "CH" "CH" "CH" "CH" "CH" "CH" "CH" "CH" "MM" "CH" "CH"
##  [76] "MM" "MM" "MM" "MM" "MM" "MM" "MM" "MM" "CH" "CH" "CH" "CH" "CH" "CH" "MM"
##  [91] "CH" "CH" "MM" "CH" "MM" "MM" "CH" "MM" "MM" "CH" "MM" "MM" "MM" "MM" "MM"
## [106] "MM" "MM" "MM" "MM" "CH" "MM" "CH" "CH" "CH" "CH" "MM" "CH" "CH" "CH" "MM"
## [121] "CH" "CH" "MM" "CH" "CH" "CH" "CH" "CH" "MM" "CH" "CH" "CH" "CH" "CH" "MM"
## [136] "MM" "MM" "MM" "MM" "MM" "CH" "CH" "CH" "MM" "MM" "CH" "CH" "CH" "CH" "CH"
## [151] "CH" "MM" "CH" "CH" "CH" "CH" "CH" "CH" "MM" "CH" "CH" "CH" "MM" "MM" "MM"
## [166] "MM" "MM" "MM" "MM" "MM" "MM" "MM" "MM" "MM" "MM" "CH" "CH" "MM" "CH" "MM"
## [181] "CH" "CH" "MM" "CH" "MM" "MM" "CH" "CH" "MM" "CH" "MM" "MM" "MM" "CH" "CH"
## [196] "MM" "MM" "MM" "MM" "MM" "MM" "MM" "MM" "CH" "CH" "CH" "CH" "MM" "CH" "MM"
## [211] "MM" "MM" "MM" "CH" "CH" "MM" "CH" "CH" "MM" "CH" "CH" "CH" "CH" "CH" "CH"
## [226] "CH" "CH" "MM" "CH" "CH" "CH" "CH" "CH" "MM" "MM" "MM" "MM" "MM" "MM" "CH"
## [241] "CH" "MM" "CH" "CH" "CH" "CH" "MM" "MM" "MM" "MM" "MM" "MM" "MM" "CH" "MM"
## [256] "CH" "CH" "CH" "CH" "CH" "CH" "CH" "CH" "CH" "CH" "CH" "CH" "MM" "MM" "CH"
##
## $confusion
##                 Observed Class
## Predicted Class  CH  MM
##               CH 138  29
##               MM  32  71
##
## $error
## [1] 0.2259259
```

```
print(paste0("False Positive Rate: ", get_false_positive_rate(OJ.test$Purchase,boost.pred$class)))
```

```
## [1] "False Positive Rate: 0.17"
```

```
print(paste0("Error Rate: ", get_classification_error_rate(OJ.test$Purchase,boost.pred$class)))
```

```
## [1] "Error Rate: 0.23"
```

```
print(paste0("False Negative Rate: ", get_false_negative_rate(OJ.test$Purchase,boost.pred$class)))
```

```
## [1] "False Negative Rate: 0.31"
```