Practical Homework 1 Decision Trees

Erdenetuya Namsrai

2025-04-06

1. Youth dataset

```
url <- "https://raw.githubusercontent.com/mendible/5322/main/Homework%201/youth_data.Rdata"
download.file(url, destfile = "youth_data.Rdata", mode = "wb")

loaded_dataset <- load("youth_data.Rdata")
youth_data_df <- get(loaded_dataset[1])

cleaned_youth_data <- na.omit(youth_data_df)
youth <- cleaned_youth_data
#youth
```

2.1 Binary classification (e.g. has or has not used cigarettes) –>Tree with MRJFLAG –> marijuana ever used (0 = never, 1 = ever)

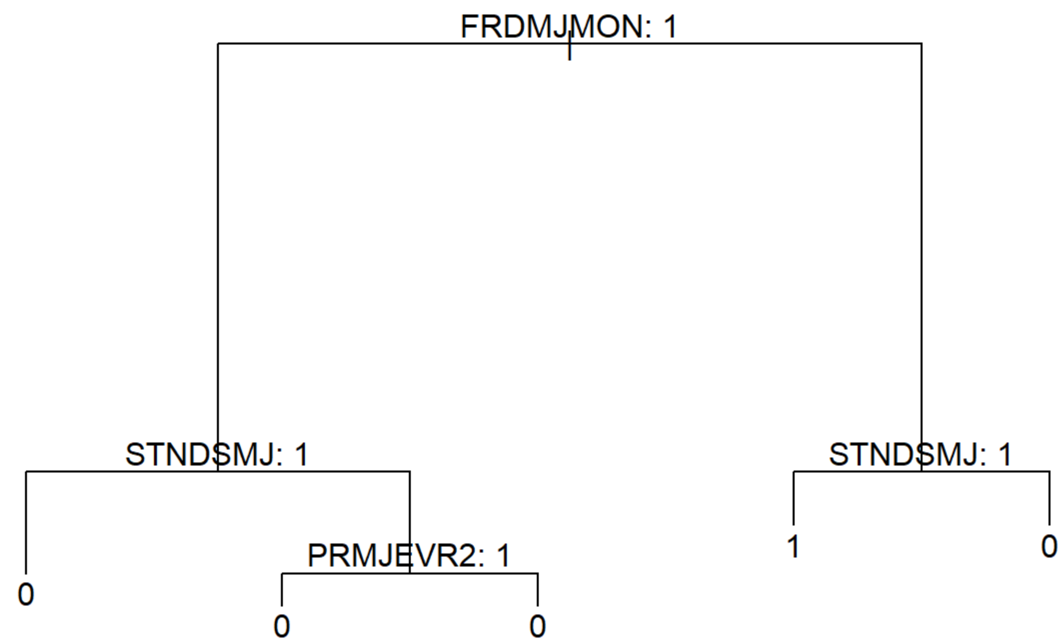
```
library(tree)
library(tidyverse)

youth_binary <- cleaned_youth_data[, c(demographic_cols, youth_experience_cols, "MRJFLAG")]
youth_binary$MRJFLAG <- as.factor(youth_binary$MRJFLAG)

set.seed(123)

training_set <- sample(1:nrow(youth_binary), 0.7 * nrow(youth_binary))
training_data <- youth_binary[training_set, ]
testing_data <- youth_binary[-training_set, ]

tree_youth <- tree(MRJFLAG ~ ., data = training_data)
summary(tree_youth)
```

```
##
## Classification tree:
## tree(formula = MRJFLAG ~ ., data = training_data)
## Variables actually used in tree construction:
## [1] "FRDMJMON" "STNDSMJ"  "PRMJEVR2"
## Number of terminal nodes:  5
## Residual mean deviance:  0.6121 = 3531 / 5769
## Misclassification error rate: 0.1299 = 750 / 5774
```

```
tree_youth
```

```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
##  1) root 5774 5084.0 0 ( 0.83963 0.16037 )
##    2) FRDMJMON: 1 4479 2212.0 0 ( 0.93257 0.06743 )
##      4) STNDSMJ: 1 905  911.2 0 ( 0.79779 0.20221 ) *
##      5) STNDSMJ: 2 3574 1044.0 0 ( 0.96670 0.03330 )
##       10) PRMJEVR2: 1 3271  707.1 0 ( 0.97738 0.02262 ) *
##       11) PRMJEVR2: 2 303  254.6 0 ( 0.85149 0.14851 ) *
##    3) FRDMJMON: 2 1295 1794.0 0 ( 0.51815 0.48185 )
##      6) STNDSMJ: 1 758 1010.0 1 ( 0.38391 0.61609 ) *
##      7) STNDSMJ: 2 537  649.0 0 ( 0.70764 0.29236 ) *
```

```
plot(tree_youth)
text(tree_youth, pretty = 0)
```




```
testing_data <- as.data.frame(testing_data)
test_pred <- predict(tree_youth, testing_data, type = "class")

confusion_matrix_dt <- table(Predicted = test_pred, Actual = testing_data$MRJFLAG)
confusion_matrix_dt
```

```
##          Actual
## Predicted    0    1
##         0 1941  215
##         1  121  198
```

```
accuracy_dt <- mean(test_pred == testing_data$MRJFLAG)
#(198+1941)/(198+1941+121+215)
test_error_rate_dt <- 1 - accuracy_dt
#(121+215)/2475

cat('Accuracy:', mean(test_pred == testing_data$MRJFLAG), "\n")
```

```
## Accuracy: 0.8642424
```

```
cat('Test Error Rate:', round(test_error_rate_dt, 4), "\n")
```

```
## Test Error Rate: 0.1358
```

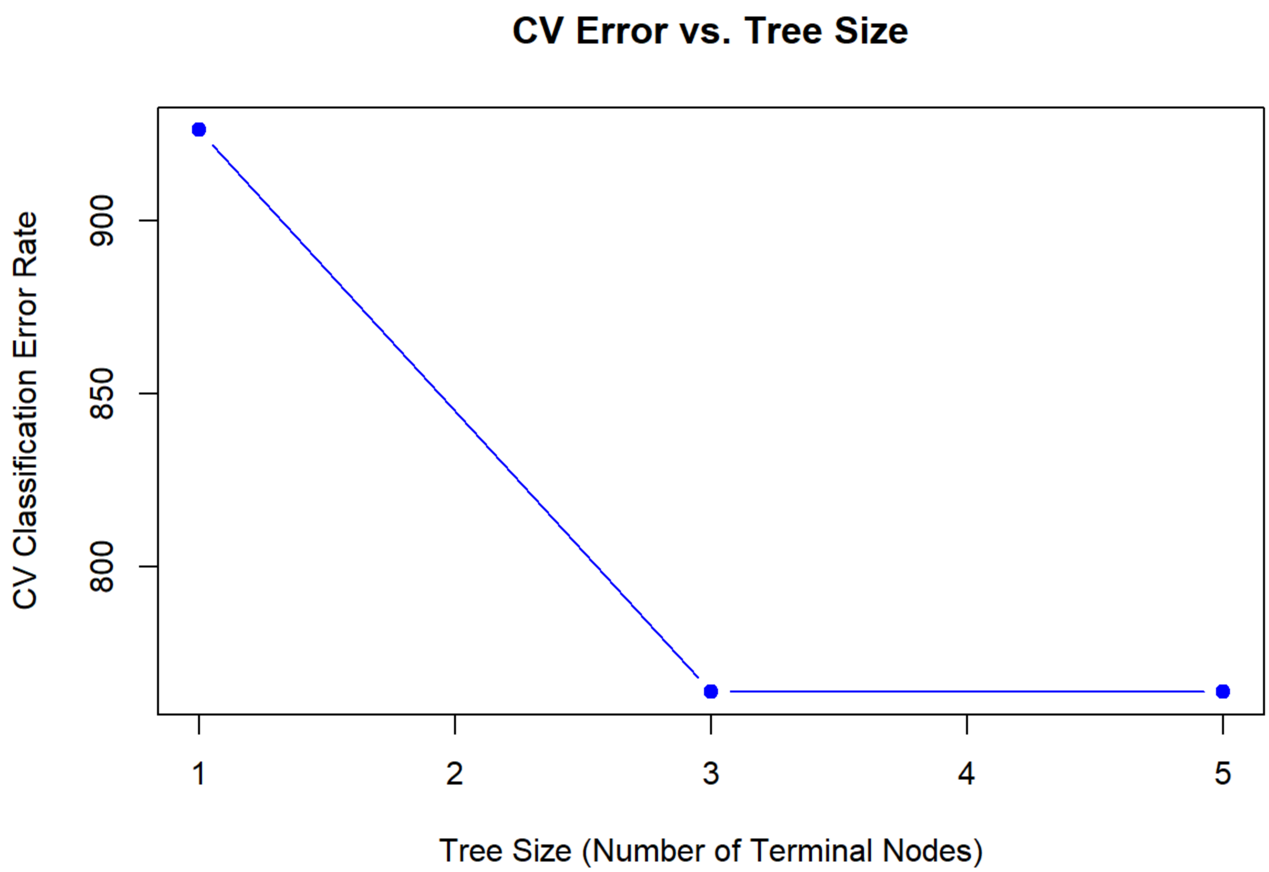
```
set.seed(7)

cv.youth <- cv.tree(tree_youth, FUN = prune.misclass)
names(cv.youth)
```

```
## [1] "size"   "dev"    "k"      "method"
```

```
cv.youth
```

```
## $size
## [1] 5 3 1
##
## $dev
## [1] 764 764 926
##
## $k
## [1] -Inf    0   88
##
## $method
## [1] "misclass"
##
## attr(,"class")
## [1] "prune"         "tree.sequence"
```

```
plot(cv.youth$size, cv.youth$dev, type = "b",
     xlab = "Tree Size (Number of Terminal Nodes)",
     ylab = "CV Classification Error Rate",
     main = "CV Error vs. Tree Size",
     pch = 19, col = "blue")
```



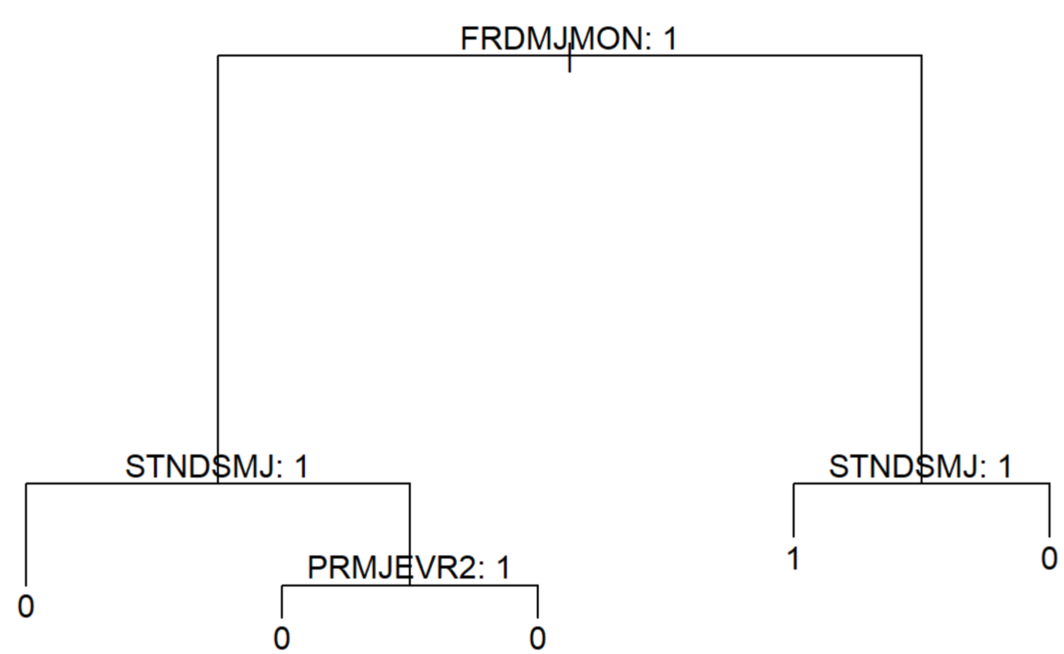

```
optimal_size_CV <- cv.youth$size[which.min(cv.youth$dev)]
optimal_size_CV
```

```
## [1] 5
```

```
pruned_tree <- prune.misclass(tree_youth, best = 5)
pruned_tree
```

```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
##  1) root 5774 5084.0 0 ( 0.83963 0.16037 )
##    2) FRDMJMON: 1 4479 2212.0 0 ( 0.93257 0.06743 )
##      4) STNDSMJ: 1 905  911.2 0 ( 0.79779 0.20221 ) *
##      5) STNDSMJ: 2 3574 1044.0 0 ( 0.96670 0.03330 )
##       10) PRMJEVR2: 1 3271  707.1 0 ( 0.97738 0.02262 ) *
##       11) PRMJEVR2: 2 303  254.6 0 ( 0.85149 0.14851 ) *
##    3) FRDMJMON: 2 1295 1794.0 0 ( 0.51815 0.48185 )
##      6) STNDSMJ: 1 758 1010.0 1 ( 0.38391 0.61609 ) *
##      7) STNDSMJ: 2 537  649.0 0 ( 0.70764 0.29236 ) *
```

```
plot(pruned_tree, main = "Pruned tree")
text(pruned_tree, pretty = 0)
```




```
testing_data <- as.data.frame(testing_data)
prune_pred_youth <- predict(pruned_tree, testing_data, type = "class")

confusion_matrix_pru <- table(Predicted = prune_pred_youth, Actual = testing_data$MRJFLAG)
confusion_matrix_pru
```

```
##          Actual
## Predicted    0    1
##         0 1941  215
##         1  121  198
```

```
accuracy_pru <- mean(prune_pred_youth == testing_data$MRJFLAG)
#(198+1941)/(198+1941+121+215)
test_error_rate_pru <- 1 - accuracy_pru
#(121+215)/2475

cat('Accuracy:', mean(prune_pred_youth == testing_data$MRJFLAG), "\n")
```

```
## Accuracy: 0.8642424
```

```
cat('Test Error Rate:', round(test_error_rate_pru, 4), "\n")
```

```
## Test Error Rate: 0.1358
```

2.2 Binary Classification –> Bagging –> with MRJFLAG –> marijuana ever used (0 = never, 1 = ever)

```
library(randomForest)

training_data_clean <- na.omit(training_data)
bag_youth = randomForest(MRJFLAG ~ ., data = training_data_clean, mtry = floor(sqrt(ncol(trainin
g_data_clean))), importance = TRUE)
bag_youth
```

```
##
## Call:
##  randomForest(formula = MRJFLAG ~ ., data = training_data_clean,      mtry = floor(sqrt(ncol
(training_data_clean))), importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 11.86%
## Confusion matrix:
##      0   1 class.error
## 0 4680 168  0.03465347
## 1  517 409  0.55831533
```

```
pred_bag_youth = predict(bag_youth, newdata = testing_data, type = 'class')

confusion_matrix_bag <- table(Predicted = pred_bag_youth, Actual = testing_data$MRJFLAG)
confusion_matrix_bag
```

```
##          Actual
## Predicted    0    1
##         0 1988  231
##         1   74  182
```

```
accuracy_bag <- mean(pred_bag_youth == testing_data$MRJFLAG)
#(180+1988)/(180+1988+74+233)
test_error_rate_bag <- 1 - accuracy_bag
#(74+233)/2475

cat('Accuracy:', mean(pred_bag_youth == testing_data$MRJFLAG), "\n")
```

```
## Accuracy: 0.8767677
```

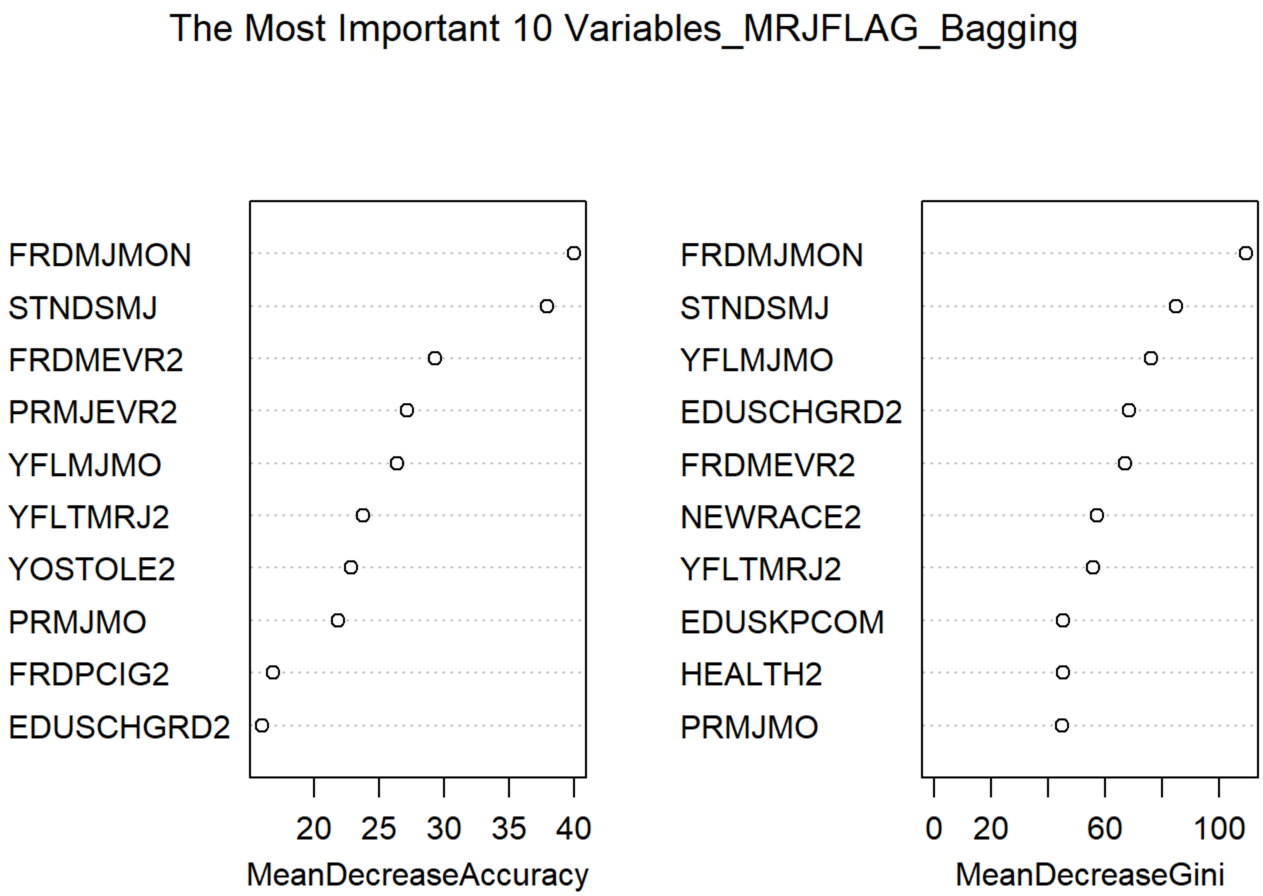
```
cat('Test Error Rate:', round(test_error_rate_bag, 4), "\n")
```

```
## Test Error Rate: 0.1232
```

```
top_10_bag_MRJFLAG = head(importance(bag_youth),10)
top_10_bag_MRJFLAG
```

```
##                     0         1 MeanDecreaseAccuracy MeanDecreaseGini
## IRSEX      -1.8737844  2.782412            0.1224023         19.57523
## NEWRACE2    7.2358409 -1.921402            5.1098328         57.06084
## HEALTH2    -0.6499873  8.251551            4.8961615         45.09863
## EDUSCHLGO   7.9006482  1.690225            8.2089011         14.12366
## EDUSCHGRD2  6.1800632 17.875633           16.0387093         68.15987
## EDUSKPCOM   4.3462588  1.978839            4.7195136         45.21221
## IMOTHER    -0.3355724  1.875191            0.7093558         12.64067
## IFATHER     4.5938223  3.105877            5.5076248         20.89775
## INCOME      6.7959102  1.260030            6.8911994         36.37135
## GOVTPROG    8.1346071  2.096600            8.5733163         17.03153
```

```
varImpPlot(bag_youth, n.var = 10, sort = TRUE, main = 'The Most Important 10 Variables_MRJFLAG_B
agging')
```



2.3 Binary Classification –> Random Forest –> with MRJFLAG –> marijuana ever used (0 = never, 1 = ever)

```
set.seed(1)
rf_youth = randomForest(MRJFLAG ~ ., data = training_data_clean, mtry = sqrt(ncol(training_data_
clean)), importance = TRUE)
rf_youth
```

```
##
## Call:
##  randomForest(formula = MRJFLAG ~ ., data = training_data_clean,      mtry = sqrt(ncol(traini
ng_data_clean)), importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 8
##
##         OOB estimate of  error rate: 12.09%
## Confusion matrix:
##      0   1 class.error
## 0 4660 188  0.03877888
## 1  510 416  0.55075594
```

```
yhat.rf <- predict(rf_youth, newdata = testing_data, type = 'class')

confusion_matrix_rf <- table(Predicted = yhat.rf, Actual = testing_data$MRJFLAG)
confusion_matrix_rf
```

```
##          Actual
## Predicted    0    1
##         0 1985  228
##         1   77  185
```

```
accuracy_rf <- mean(yhat.rf == testing_data$MRJFLAG, na.rm = TRUE)
test_error_rf <- 1 - accuracy_rf

cat("Accuracy:", round(accuracy_rf, 4), "\n")
```

```
## Accuracy: 0.8768
```

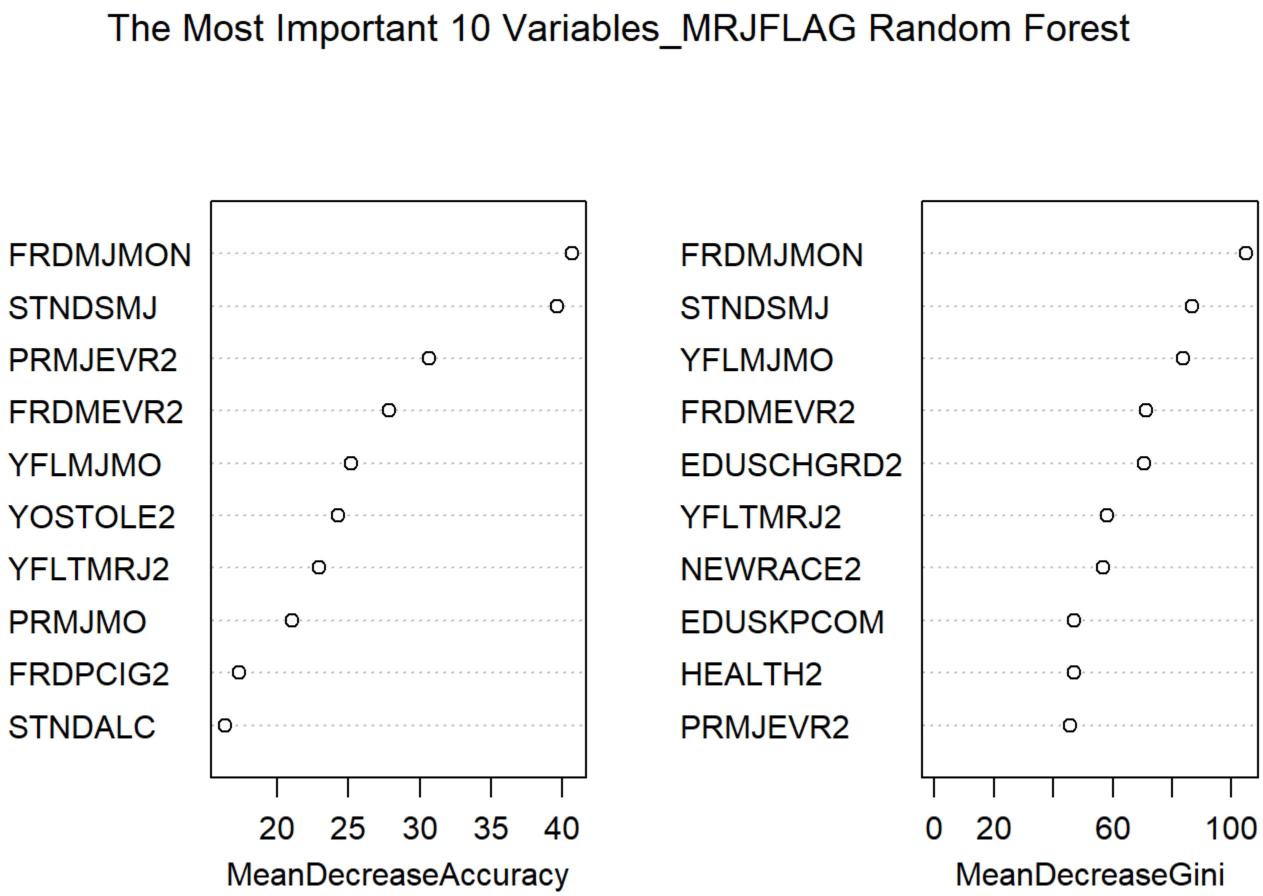
```
cat("Test Error Rate:", round(test_error_rf, 4), "\n")
```

```
## Test Error Rate: 0.1232
```

```
top_10_rf_MRJFLAG = head(importance(rf_youth),10)
top_10_rf_MRJFLAG
```

```
##                     0           1 MeanDecreaseAccuracy MeanDecreaseGini
## IRSEX      -0.1157842  3.58840417             2.081756         20.25110
## NEWRACE2    4.4962492  0.01980726             3.941679         56.91348
## HEALTH2    -1.7532805  8.92333215             4.130153         46.99917
## EDUSCHLGO   6.9505821  0.61237778             6.852772         13.36547
## EDUSCHGRD2  4.8814248 16.42028957            14.772170         70.73182
## EDUSKPCOM   4.8005257  1.03320065             4.599505         47.15655
## IMOTHER     0.0790352  2.08961310             1.271079         12.14550
## IFATHER     4.2265963  3.10415279             5.284194         20.48444
## INCOME      8.8768826  1.52806173             8.621981         37.65077
## GOVTPROG    6.9103234  6.37537191             9.724761         17.02415
```

```
varImpPlot(rf_youth, n.var = 10, sort = TRUE, main = 'The Most Important 10 Variables_MRJFLAG Ra
ndom Forest ')
```



3.1 Binary classification (e.g. has or has not used cigarettes) Tree with TOBFLAG –> any tobacco ever used (0 = never, 1 = ever)

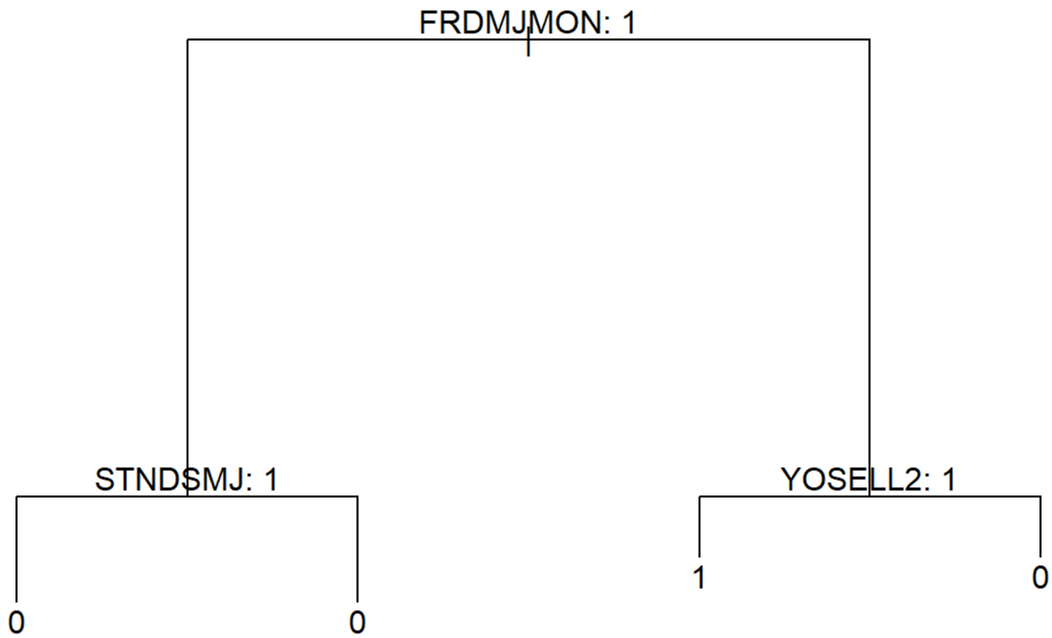
```
youth_binary <- cleaned_youth_data[, c(demographic_cols, youth_experience_cols, "TOBFLAG")]
youth_binary$TOBFLAG <- as.factor(youth_binary$TOBFLAG)

set.seed(123)

training_set <- sample(1:nrow(youth_binary), 0.7 * nrow(youth_binary))
training_data <- youth_binary[training_set, ]
testing_data <- youth_binary[-training_set, ]

tree_youth2 <- tree(TOBFLAG ~ ., data = training_data)
summary(tree_youth2)
```

```
##
## Classification tree:
## tree(formula = TOBFLAG ~ ., data = training_data)
## Variables actually used in tree construction:
## [1] "FRDMJMON" "STNDSMJ"  "YOSELL2"
## Number of terminal nodes:  4
## Residual mean deviance:  0.547 = 3156 / 5770
## Misclassification error rate: 0.09318 = 538 / 5774
```

```
tree_youth2
```

```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
## 1) root 5774 3677.00 0 ( 0.90301 0.09699 )
##   2) FRDMJMON: 1 4479 1831.00 0 ( 0.94798 0.05202 )
##     4) STNDSMJ: 1 905  665.70 0 ( 0.87956 0.12044 ) *
##     5) STNDSMJ: 2 3574 1077.00 0 ( 0.96530 0.03470 ) *
##   3) FRDMJMON: 2 1295 1464.00 0 ( 0.74749 0.25251 )
##     6) YOSELL2: 1 52   62.48 1 ( 0.28846 0.71154 ) *
##     7) YOSELL2: 2 1243 1350.00 0 ( 0.76669 0.23331 ) *
```

```
plot(tree_youth2)
text(tree_youth2, pretty = 0)
```




```
testing_data <- as.data.frame(testing_data)
test_pred <- predict(tree_youth2, testing_data, type = "class")

confusion_matrix_tobflag <- table(Predicted = test_pred, Actual = testing_data$TOBFLAG)
confusion_matrix_tobflag
```

```
##          Actual
## Predicted    0    1
##         0 2224  229
##         1    8   14
```

```
accuracy_tobflag <- mean(test_pred == testing_data$TOBFLAG)
#(2224+14)/(2224+299+8+14)
test_error_rate_tobflag <- 1 - accuracy_tobflag
#(8+229)/2475

cat('Accuracy:', mean(test_pred == testing_data$TOBFLAG), "\n")
```

```
## Accuracy: 0.9042424
```

```
cat('Test Error Rate:', round(test_error_rate_tobflag, 4), "\n")
```

```
## Test Error Rate: 0.0958
```

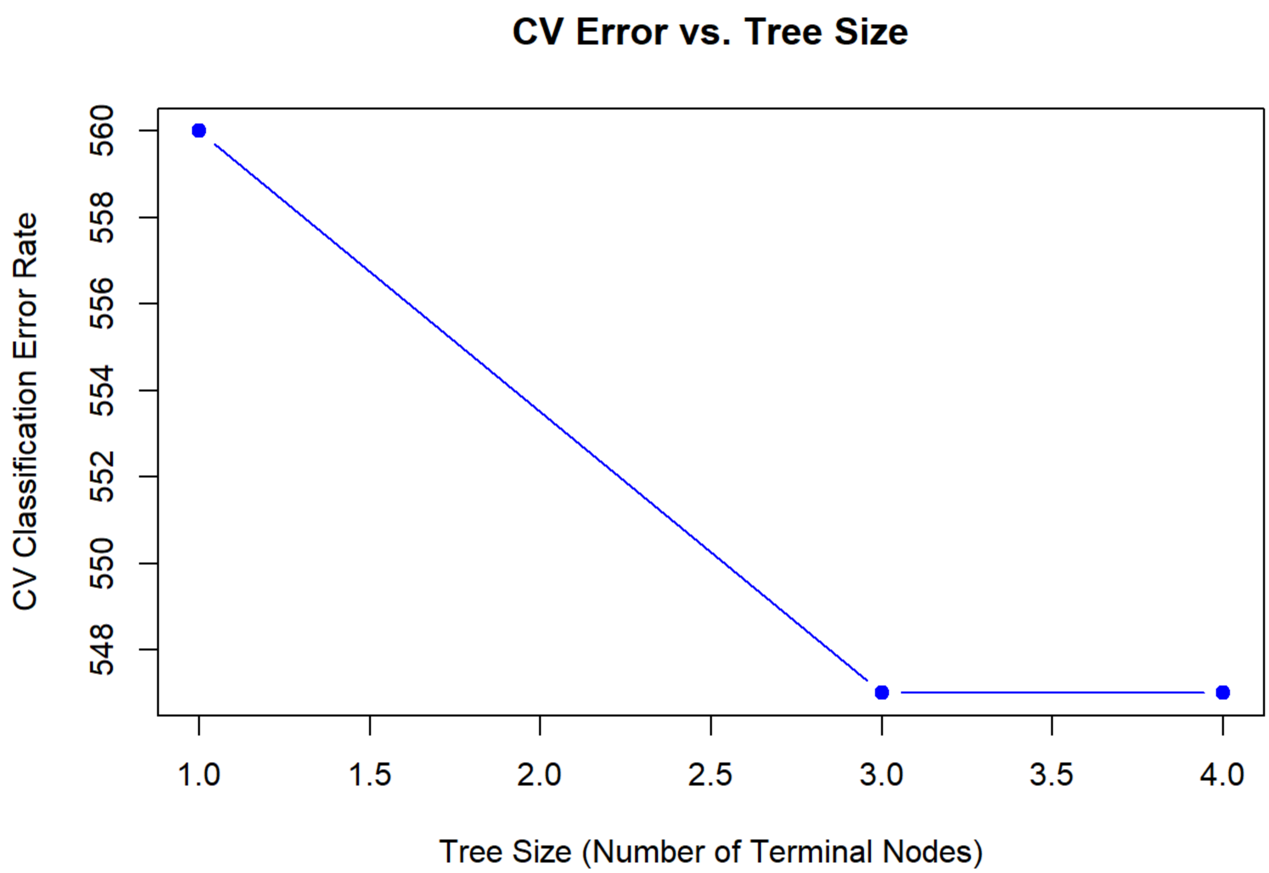
```
set.seed(7)

cv.youth2 <- cv.tree(tree_youth2, FUN = prune.misclass)
names(cv.youth2)
```

```
## [1] "size"   "dev"    "k"      "method"
```

```
cv.youth2
```

```
## $size
## [1] 4 3 1
##
## $dev
## [1] 547 547 560
##
## $k
## [1] -Inf    0   11
##
## $method
## [1] "misclass"
##
## attr(,"class")
## [1] "prune"         "tree.sequence"
```

```
plot(cv.youth2$size, cv.youth2$dev, type = "b",
     xlab = "Tree Size (Number of Terminal Nodes)",
     ylab = "CV Classification Error Rate",
     main = "CV Error vs. Tree Size",
     pch = 19, col = "blue")
```



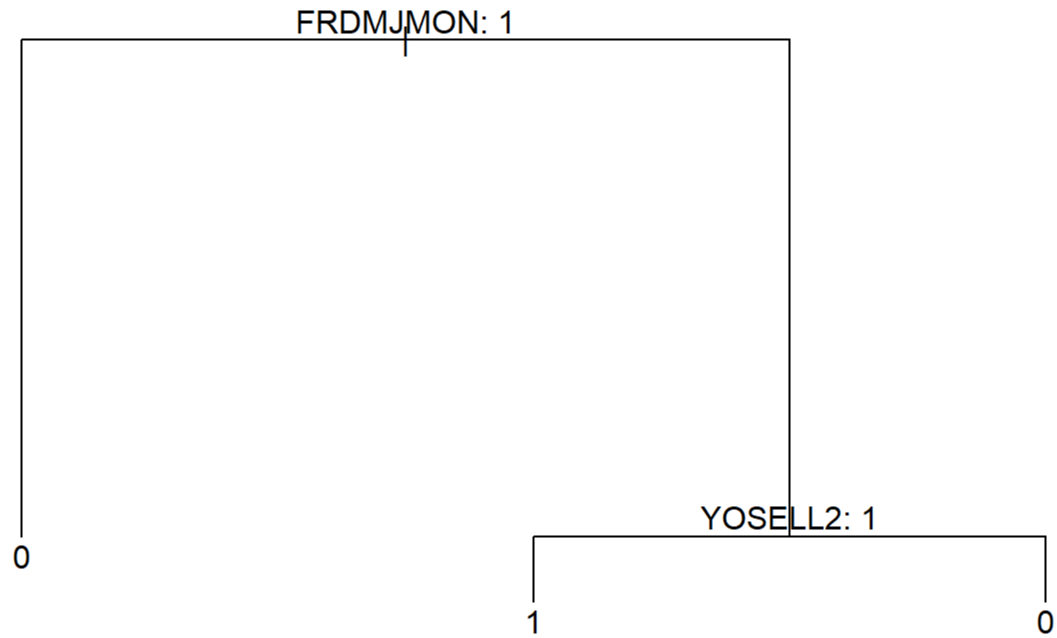

```
optimal_size_CV2 <- cv.youth2$size[which.min(cv.youth2$dev)]
optimal_size_CV2
```

```
## [1] 4
```

```
pruned_tree2 <- prune.misclass(tree_youth2, best = 3)
pruned_tree2
```

```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
## 1) root 5774 3677.00 0 ( 0.90301 0.09699 )
##   2) FRDMJMON: 1 4479 1831.00 0 ( 0.94798 0.05202 ) *
##   3) FRDMJMON: 2 1295 1464.00 0 ( 0.74749 0.25251 )
##     6) YOSELL2: 1 52   62.48 1 ( 0.28846 0.71154 ) *
##     7) YOSELL2: 2 1243 1350.00 0 ( 0.76669 0.23331 ) *
```

```
plot(pruned_tree2, main = "Pruned tree")
text(pruned_tree2, pretty = 0)
```




```
testing_data <- as.data.frame(testing_data)
prune_pred_youth2 <- predict(pruned_tree2, testing_data, type = "class")

confusion_matrix_pru2 <- table(Predicted = prune_pred_youth2, Actual = testing_data$TOBFLAG)
confusion_matrix_pru2
```

```
##          Actual
## Predicted    0    1
##         0 2224  229
##         1    8   14
```

```
accuracy_pru2 <- mean(prune_pred_youth2 == testing_data$TOBFLAG)
#(2002+89)/(2002+89+230+154)
test_error_rate_pru2 <- 1 - accuracy_pru2
#(230+154)/2475

cat('Accuracy:', mean(prune_pred_youth2 == testing_data$TOBFLAG), "\n")
```

```
## Accuracy: 0.9042424
```

```
cat('Test Error Rate:', round(test_error_rate_pru2, 4), "\n")
```

```
## Test Error Rate: 0.0958
```

3.2 Binary Classification –> Bagging –> with TOBFLAG –> any tobacco ever used (0 = never, 1 = ever)

```
library(randomForest)

training_data_clean <- na.omit(training_data)
bag_youth2 = randomForest(TOBFLAG ~ ., data = training_data_clean, mtry = floor(sqrt(ncol(traini
ng_data_clean))), importance = TRUE)
bag_youth2
```

```
##
## Call:
##  randomForest(formula = TOBFLAG ~ ., data = training_data_clean,      mtry = floor(sqrt(ncol
(training_data_clean))), importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 9.39%
## Confusion matrix:
##      0  1 class.error
## 0 5187 27 0.005178366
## 1  515 45 0.919642857
```

```
pred_bag_youth2 = predict(bag_youth2, newdata = testing_data, type = 'class')

confusion_matrix_bag2 <- table(Predicted = pred_bag_youth2, Actual = testing_data$TOBFLAG)
confusion_matrix_bag2
```

```
##          Actual
## Predicted    0    1
##         0 2224  227
##         1    8   16
```

```
accuracy_bag2 <- mean(pred_bag_youth2 == testing_data$TOBFLAG)
#(2224+16)/(2224+16+8+227)
test_error_rate_bag2 <- 1 - accuracy_bag2
#(8+227)/2475

cat('Accuracy:', mean(pred_bag_youth2 == testing_data$TOBFLAG), "\n")
```

```
## Accuracy: 0.9050505
```

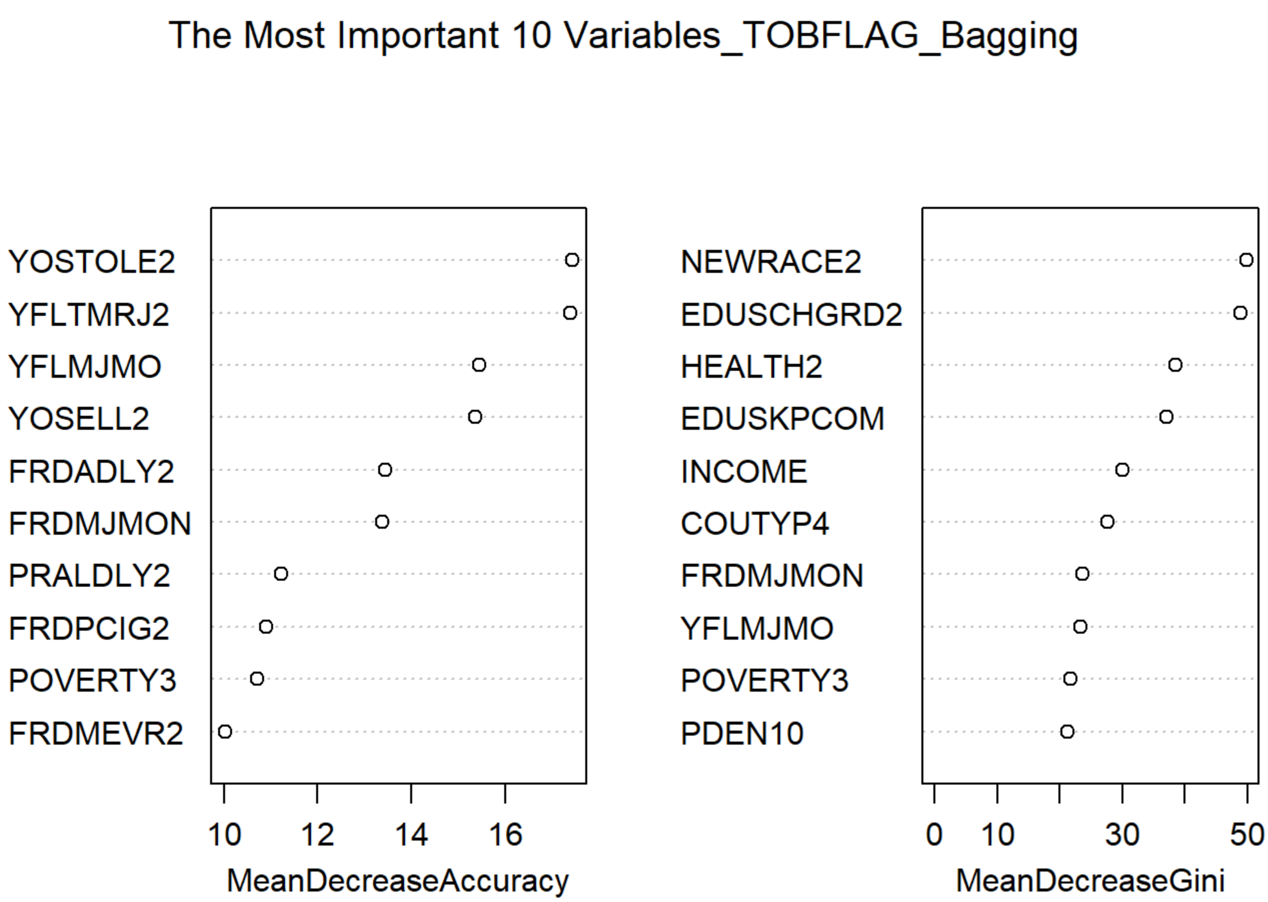
```
cat('Test Error Rate:', round(test_error_rate_bag2, 4), "\n")
```

```
## Test Error Rate: 0.0949
```

```
top_10_bag2_TOBFLAG = head(importance(bag_youth2),10)
top_10_bag2_TOBFLAG
```

```
##                    0          1 MeanDecreaseAccuracy MeanDecreaseGini
## IRSEX       0.5575283 -1.1552127           0.03638061         15.84773
## NEWRACE2    9.9692110  1.8222799           9.66997197         49.81190
## HEALTH2    -3.4647185  8.7210754           0.78696368         38.57313
## EDUSCHLGO   3.8651035 -0.7942530           3.47124749         11.17121
## EDUSCHGRD2  4.7203605  8.9304330           8.29280269         48.85338
## EDUSKPCOM   1.1928923  1.3925700           1.66021257         37.09268
## IMOTHER    -0.8581764  2.7383526           0.49398880         10.19009
## IFATHER     0.3966267 -2.1146757          -0.41032466         16.03109
## INCOME      8.1639593 -0.2730579           7.81041058         30.07543
## GOVTPROG    4.4013270  1.6264163           4.72209606         13.72608
```

```
varImpPlot(bag_youth2, n.var = 10, sort = TRUE, main = 'The Most Important 10 Variables_TOBFLAG_
Bagging')
```



3.3 Binary Classification –> Random Forest –> with TOBFLAG –> any tobacco ever used (0 = never, 1 = ever)

```
set.seed(1)
rf_youth2 = randomForest(TOBFLAG ~ ., data = training_data_clean, mtry = sqrt(ncol(training_data
_clean)), importance = TRUE)
rf_youth2
```

```
##
## Call:
##  randomForest(formula = TOBFLAG ~ ., data = training_data_clean,      mtry = sqrt(ncol(traini
ng_data_clean)), importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 8
##
##         OOB estimate of  error rate: 9.35%
## Confusion matrix:
##      0  1 class.error
## 0 5188 26 0.004986575
## 1  514 46 0.917857143
```

```
yhat.rf <- predict(rf_youth2, newdata = testing_data, type = 'class')

confusion_matrix_rf2 <- table(Predicted = yhat.rf, Actual = testing_data$TOBFLAG)
confusion_matrix_rf2
```

```
##          Actual
## Predicted    0    1
##         0 2223  227
##         1    9   16
```

```
accuracy_rf2 <- mean(yhat.rf == testing_data$TOBFLAG, na.rm = TRUE)
test_error_rate_rf2 <- 1 - accuracy_rf2

cat("Accuracy:", round(accuracy_rf2, 4), "\n")
```

```
## Accuracy: 0.9046
```

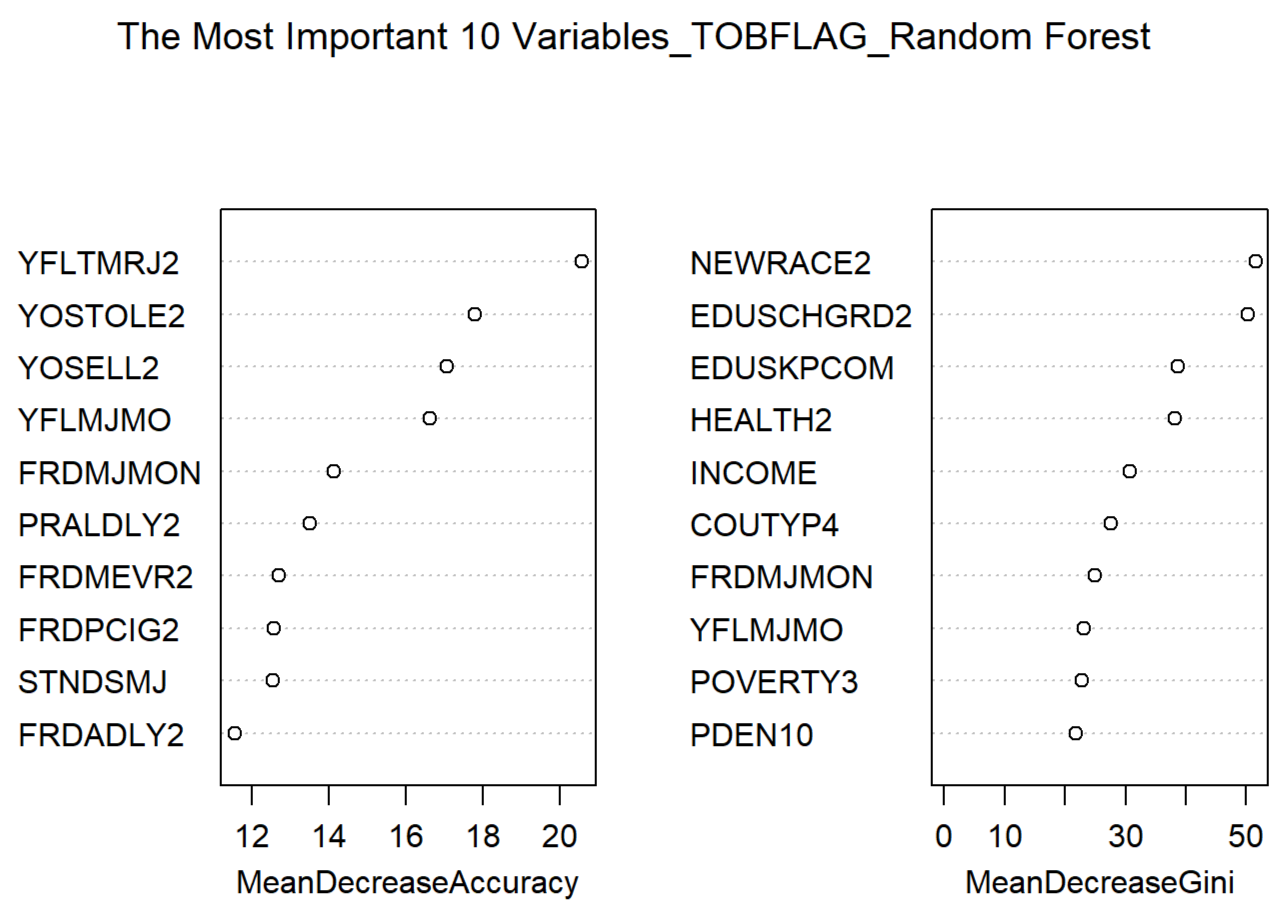
```
cat("Test Error Rate:", round(test_error_rate_rf2, 4), "\n")
```

```
## Test Error Rate: 0.0954
```

```
top_10_rf2_TOBFLAG = head(importance(rf_youth2),10)
top_10_rf2_TOBFLAG
```

```
##                   0          1 MeanDecreaseAccuracy MeanDecreaseGini
## IRSEX       2.2747632 -1.9374433            1.3610702         15.16408
## NEWRACE2   10.7706212  2.0182675           10.8896446         51.52574
## HEALTH2    -1.7244397  6.8033723            1.4588220         38.09800
## EDUSCHLGO   5.1332989 -0.5556501            4.9969606         10.77661
## EDUSCHGRD2  5.3339380  7.9335198            8.3243585         50.24594
## EDUSKPCOM   2.5890580  1.6953351            3.0580595         38.63693
## IMOTHER    -0.4774065  2.7254581            0.7538056         10.25463
## IFATHER     0.9609497  0.3411408            0.9917672         16.11981
## INCOME     10.0194664 -0.9962356            9.3224914         30.70777
## GOVTPROG    4.7735617  0.9069025            4.9611468         13.36711
```

```
varImpPlot(rf_youth2, n.var = 10, sort = TRUE, main = 'The Most Important 10 Variables_TOBFLAG_R
andom Forest')
```



4.1 Binary classification (e.g. has or has not used cigarettes) Tree with ALCFLAG –> alcohol ever used

(0 = never, 1 = ever)

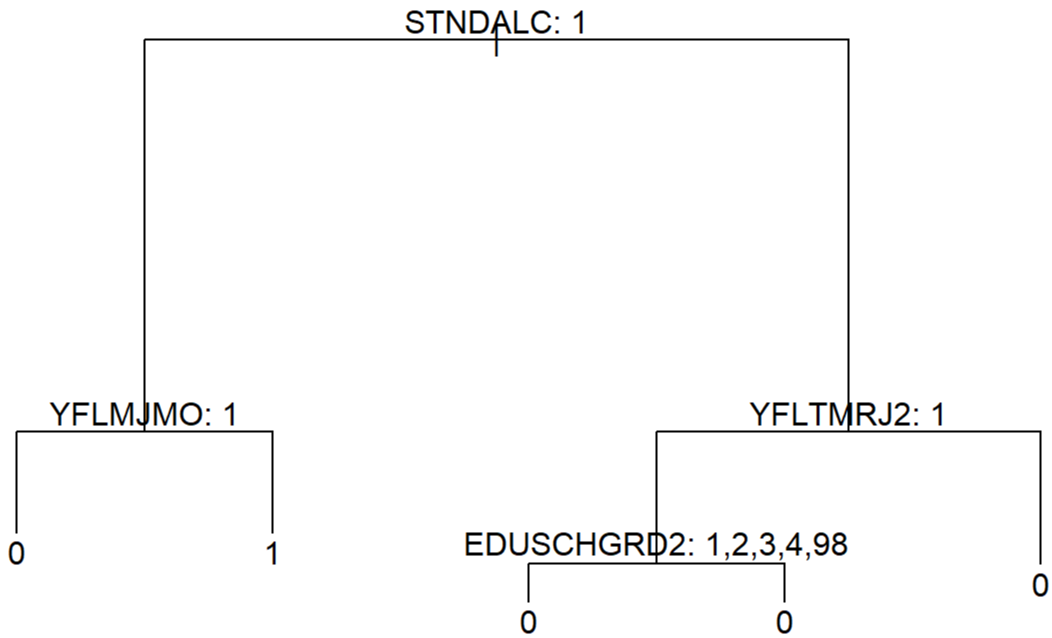
```
youth_binary3 <- cleaned_youth_data[, c(demographic_cols, youth_experience_cols, "ALCFLAG")]
youth_binary3$ALCFLAG <- as.factor(youth_binary3$ALCFLAG)

set.seed(123)

training_set <- sample(1:nrow(youth_binary3), 0.7 * nrow(youth_binary3))
training_data <- youth_binary3[training_set, ]
testing_data <- youth_binary3[-training_set, ]

tree_youth3 <- tree(ALCFLAG ~ ., data = training_data)
summary(tree_youth3)
```

```
##
## Classification tree:
## tree(formula = ALCFLAG ~ ., data = training_data)
## Variables actually used in tree construction:
## [1] "STNDALC"    "YFLMJMO"    "YFLTMRJ2"   "EDUSCHGRD2"
## Number of terminal nodes:  5
## Residual mean deviance:  0.9141 = 5273 / 5769
## Misclassification error rate: 0.2037 = 1176 / 5774
```

```
tree_youth3
```

```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
##  1) root 5774 6464.0 0 ( 0.75234 0.24766 )
##    2) STNDALC: 1 1681 2330.0 0 ( 0.50922 0.49078 )
##      4) YFLMJMO: 1 1017 1329.0 0 ( 0.64012 0.35988 ) *
##      5) YFLMJMO: 2 664  820.8 1 ( 0.30873 0.69127 ) *
##    3) STNDALC: 2 4093 3429.0 0 ( 0.85219 0.14781 )
##      6) YFLTMRJ2: 1 3409 2299.0 0 ( 0.89440 0.10560 )
##       12) EDUSCHGRD2: 1,2,3,4,98 1446  629.9 0 ( 0.94329 0.05671 ) *
##       13) EDUSCHGRD2: 5,6,7,8,9,10,99 1963 1601.0 0 ( 0.85838 0.14162 ) *
##      7) YFLTMRJ2: 2 684  892.4 0 ( 0.64181 0.35819 ) *
```

```
plot(tree_youth3)
text(tree_youth3, pretty = 0)
```




```
testing_data <- as.data.frame(testing_data)
test_pred <- predict(tree_youth3, testing_data, type = "class")

confusion_matrix_alcflag <- table(Predicted = test_pred, Actual = testing_data$ALCFLAG)
confusion_matrix_alcflag
```

```
##          Actual
## Predicted    0    1
##         0 1768  440
##         1   87  180
```

```
accuracy_alcflag <- mean(test_pred == testing_data$ALCFLAG)
#(1768+180)/(1768+180+87+440)
test_error_rate_alcflag <- 1 - accuracy_alcflag
#(87+440)/2475

cat('Accuracy:', mean(test_pred == testing_data$ALCFLAG), "\n")
```

```
## Accuracy: 0.7870707
```

```
cat('Test Error Rate:', round(test_error_rate_alcflag, 4), "\n")
```

```
## Test Error Rate: 0.2129
```

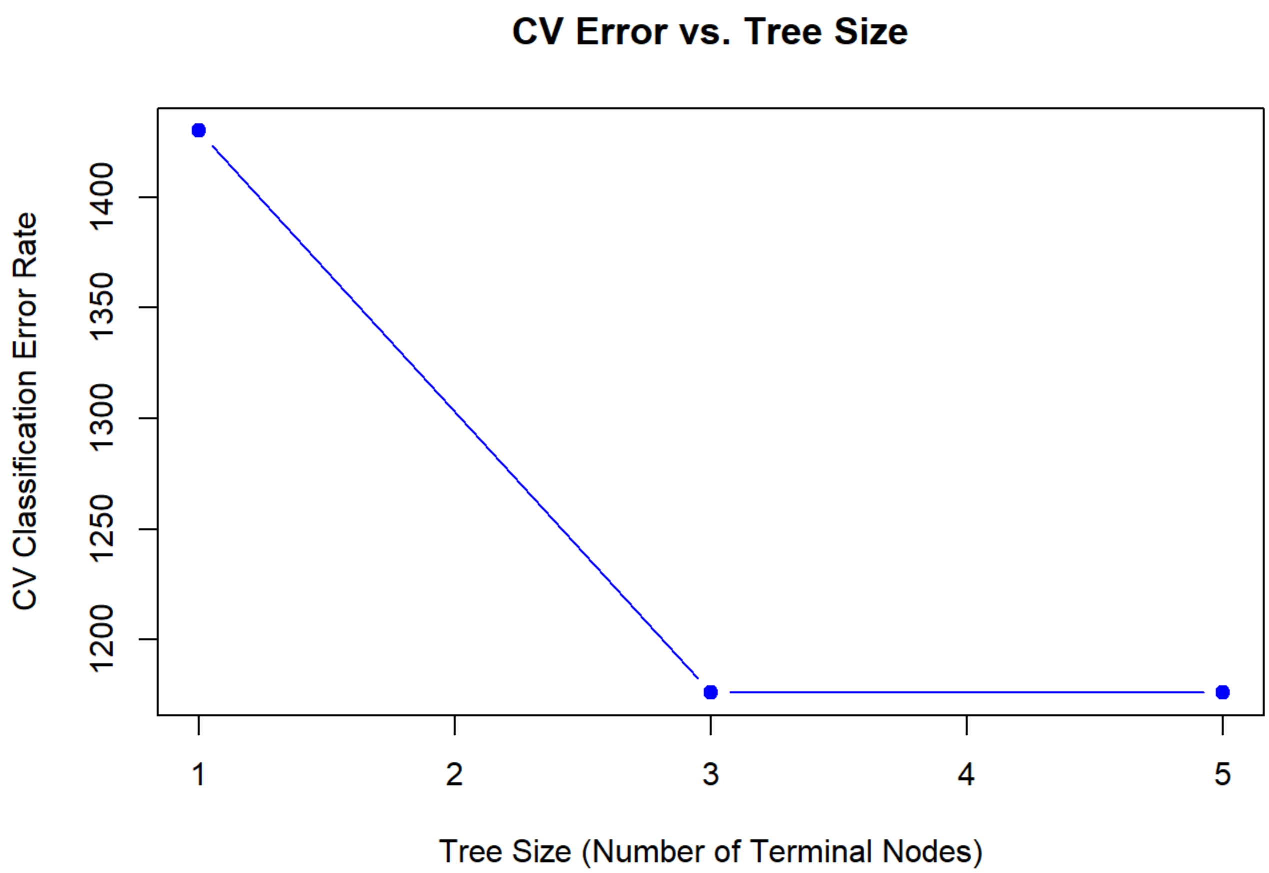
```
set.seed(7)

cv.youth3 <- cv.tree(tree_youth3, FUN = prune.misclass)
names(cv.youth3)
```

```
## [1] "size"   "dev"    "k"      "method"
```

```
cv.youth3
```

```
## $size
## [1] 5 3 1
##
## $dev
## [1] 1176 1176 1430
##
## $k
## [1] -Inf    0  127
##
## $method
## [1] "misclass"
##
## attr(,"class")
## [1] "prune"         "tree.sequence"
```

```
plot(cv.youth3$size, cv.youth3$dev, type = "b",
     xlab = "Tree Size (Number of Terminal Nodes)",
     ylab = "CV Classification Error Rate",
     main = "CV Error vs. Tree Size",
     pch = 19, col = "blue")
```



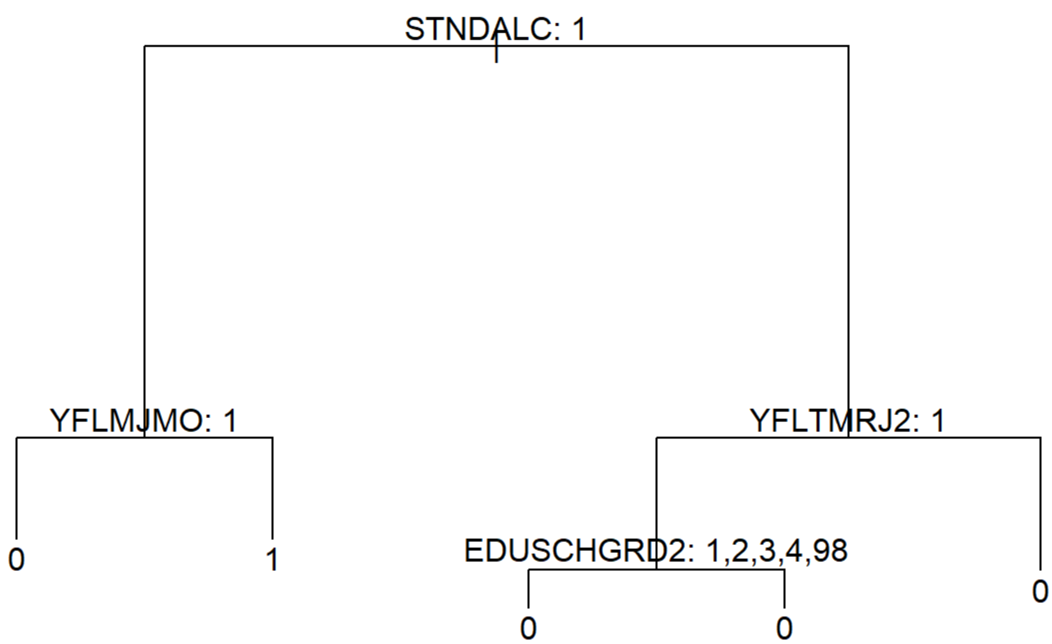

```
optimal_size_CV3 <- cv.youth3$size[which.min(cv.youth3$dev)]
optimal_size_CV3
```

```
## [1] 5
```

```
pruned_tree3 <- prune.misclass(tree_youth3, best = 5)
pruned_tree3 <- prune.misclass(tree_youth3, best = 5)
pruned_tree3
```

```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
##  1) root 5774 6464.0 0 ( 0.75234 0.24766 )
##    2) STNDALC: 1 1681 2330.0 0 ( 0.50922 0.49078 )
##      4) YFLMJMO: 1 1017 1329.0 0 ( 0.64012 0.35988 ) *
##      5) YFLMJMO: 2 664  820.8 1 ( 0.30873 0.69127 ) *
##    3) STNDALC: 2 4093 3429.0 0 ( 0.85219 0.14781 )
##      6) YFLTMRJ2: 1 3409 2299.0 0 ( 0.89440 0.10560 )
##       12) EDUSCHGRD2: 1,2,3,4,98 1446  629.9 0 ( 0.94329 0.05671 ) *
##       13) EDUSCHGRD2: 5,6,7,8,9,10,99 1963 1601.0 0 ( 0.85838 0.14162 ) *
##      7) YFLTMRJ2: 2 684  892.4 0 ( 0.64181 0.35819 ) *
```

```
plot(pruned_tree3, main = "Pruned tree")
text(pruned_tree3, pretty = 0)
```




```
testing_data <- as.data.frame(testing_data)
prune_pred_youth3 = predict(pruned_tree3, testing_data, type = 'class')

confusion_matrix_pru3 <- table(Predicted = prune_pred_youth3, Actual = testing_data$ALCFLAG)
confusion_matrix_pru3
```

```
##          Actual
## Predicted    0    1
##         0 1768  440
##         1   87  180
```

```
accuracy_pru3 <- mean(prune_pred_youth3 == testing_data$ALCFLAG)
#(1738+202)/(1738+202+117+418)
test_error_rate_pru3 <- 1 - accuracy_pru3
#(117+418)/2475

cat('Accuracy:', mean(prune_pred_youth3 == testing_data$ALCFLAG), "\n")
```

```
## Accuracy: 0.7870707
```

```
cat('Test Error Rate:', round(test_error_rate_pru3, 4), "\n")
```

```
## Test Error Rate: 0.2129
```

4.2 Binary Classification –> Bagging –> with ALCFLAG –> alcohol ever used (0 = never, 1 = ever)

```
library(randomForest)

training_data_clean = na.omit(training_data)
bag_youth3 = randomForest(ALCFLAG ~ ., data = training_data_clean, mtry = floor(sqrt(ncol(traini
ng_data_clean))), importance = TRUE)
bag_youth3
```

```
##
## Call:
##  randomForest(formula = ALCFLAG ~ ., data = training_data_clean,      mtry = floor(sqrt(ncol
(training_data_clean))), importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 19.61%
## Confusion matrix:
##      0   1 class.error
## 0 4075 269  0.06192449
## 1  863 567  0.60349650
```

```
pred_bag_youth3 = predict(bag_youth3, newdata = testing_data, type = 'class')

confusion_matrix_bag3 <- table(Predicted = pred_bag_youth3, Actual = testing_data$ALCFLAG)
confusion_matrix_bag3
```

```
##          Actual
## Predicted    0    1
##         0 1735  379
##         1  120  241
```

```
accuracy_bag3 <- mean(pred_bag_youth3 == testing_data$ALCFLAG)
#(2224+16)/(2224+16+8+227)
test_error_rate_bag3 <- 1 - accuracy_bag3
#(8+227)/2475

cat('Accuracy:', mean(pred_bag_youth3 == testing_data$ALCFLAG), "\n")
```

```
## Accuracy: 0.7983838
```

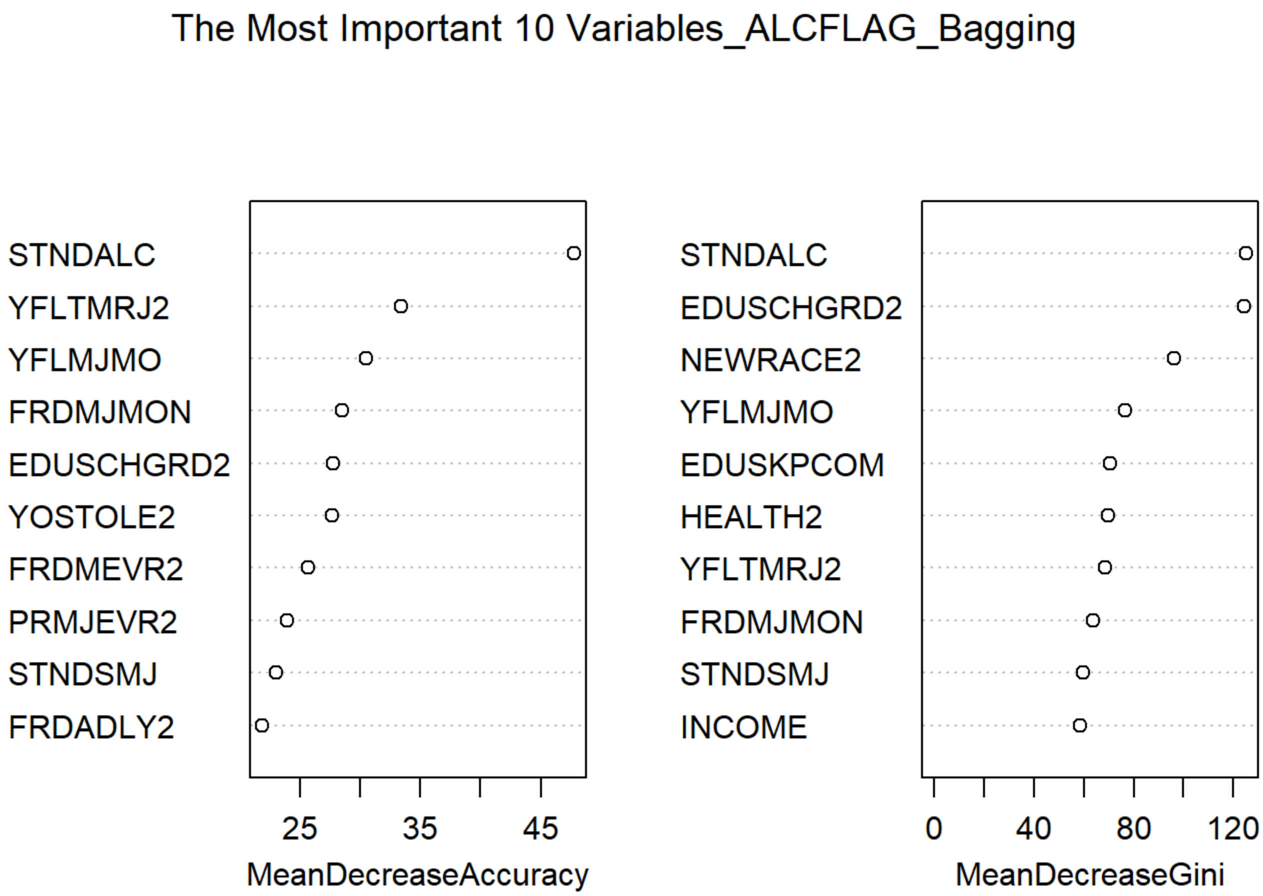
```
cat('Test Error Rate:', round(test_error_rate_bag3, 4), "\n")
```

```
## Test Error Rate: 0.2016
```

```
top_10_bag3_ALCFLAG = head(importance(bag_youth3),10)
top_10_bag3_ALCFLAG
```

```
##                     0            1 MeanDecreaseAccuracy MeanDecreaseGini
## IRSEX       0.8140106  1.752712436             1.703886         34.15393
## NEWRACE2   12.2530463  1.944899845            11.948808         96.32596
## HEALTH2     1.6482554  3.838190565             3.443447         69.87464
## EDUSCHLGO   8.9453423 -3.779807662             6.501898         17.71229
## EDUSCHGRD2 15.3954957 22.938638280            27.811631        124.40449
## EDUSKPCOM   7.2677393  1.485427503             7.322552         70.60417
## IMOTHER    -1.5786681 -1.353060887            -2.061123         19.21708
## IFATHER     8.5383233  0.311178438             7.774197         31.93798
## INCOME     11.1817173  5.011496717            13.528991         58.35479
## GOVTPROG   10.1807735 -0.001320414             9.104788         26.70772
```

```
varImpPlot(bag_youth3, n.var = 10, sort = TRUE, main = 'The Most Important 10 Variables_ALCFLAG_
Bagging')
```



4.3 Binary Classification –> Random Forest –> with ALCFLAG –> alcohol ever used (0 = never, 1 = ever)

```
set.seed(1)
rf_youth3 = randomForest(ALCFLAG ~ ., data = training_data_clean, mtry = sqrt(ncol(training_data
_clean)), importance = TRUE)
rf_youth3
```

```
##
## Call:
##  randomForest(formula = ALCFLAG ~ ., data = training_data_clean,      mtry = sqrt(ncol(traini
ng_data_clean)), importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 8
##
##         OOB estimate of  error rate: 19.64%
## Confusion matrix:
##      0   1 class.error
## 0 4074 270   0.0621547
## 1  864 566   0.6041958
```

```
yhat.rf <- predict(rf_youth3, newdata = testing_data, type = 'class')

confusion_matrix_rf3 <- table(Predicted = yhat.rf, Actual = testing_data$ALCFLAG)
confusion_matrix_rf3
```

```
##          Actual
## Predicted    0    1
##         0 1737  370
##         1  118  250
```

```
accuracy_rf3 <- mean(yhat.rf == testing_data$ALCFLAG, na.rm = TRUE)
test_error_rate_rf3 <- 1 - accuracy_rf3

cat("Accuracy:", round(accuracy_rf3, 4), "\n")
```

```
## Accuracy: 0.8028
```

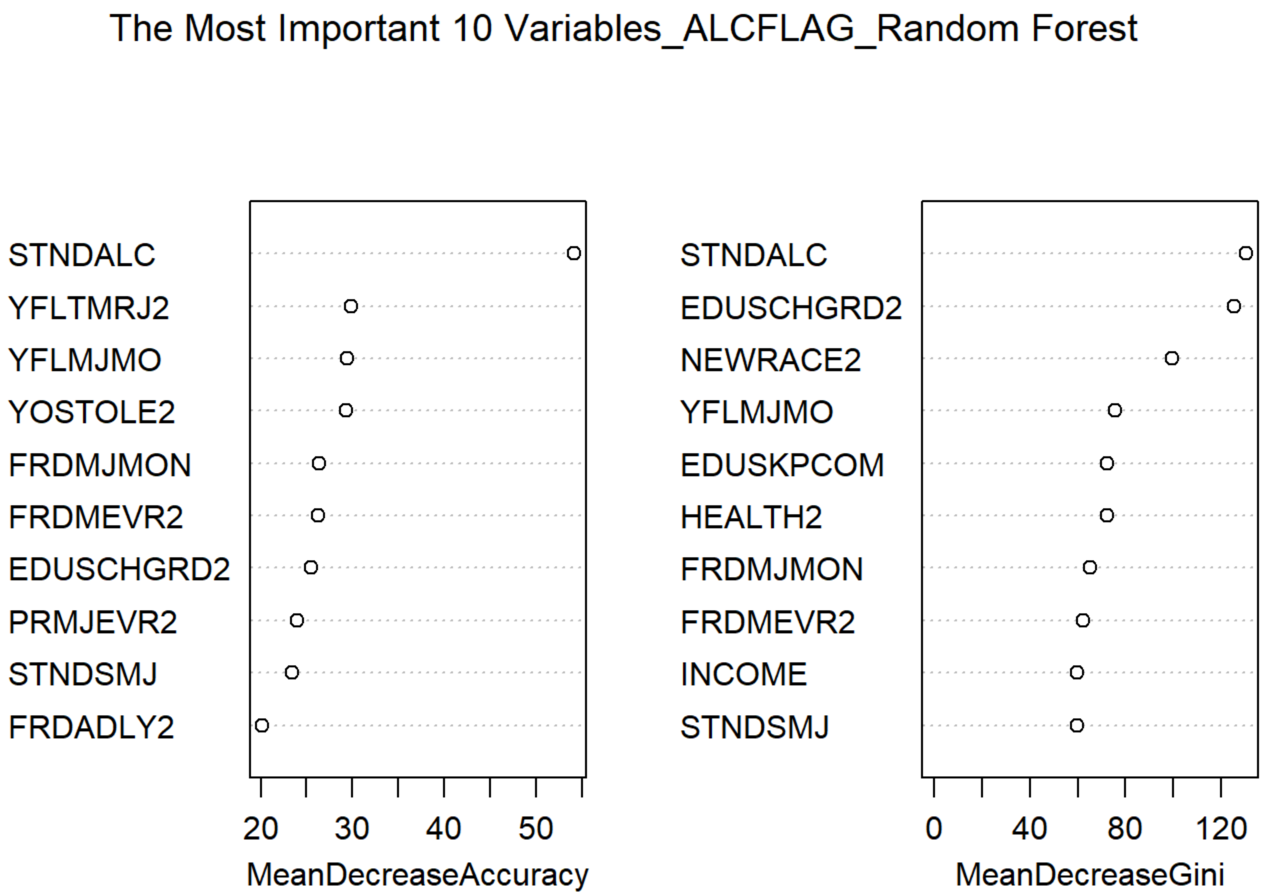
```
cat("Test Error Rate:", round(test_error_rate_rf3, 4), "\n")
```

```
## Test Error Rate: 0.1972
```

```
top_10_rf3_ALCFLAG = head(importance(rf_youth3),10)
top_10_rf3_ALCFLAG
```

```
##                     0          1 MeanDecreaseAccuracy MeanDecreaseGini
## IRSEX       1.5446554  2.3695415             2.572437         34.00565
## NEWRACE2   12.4431278  1.4848483            12.004452         99.48005
## HEALTH2     2.1927509  3.4885197             3.845355         72.32937
## EDUSCHLGO   8.3060529 -1.9704074             6.541364         17.58747
## EDUSCHGRD2 13.9505678 24.4614246            25.466823        125.63018
## EDUSKPCOM   6.8892816  1.7687471             6.940890         72.55806
## IMOTHER     0.7204869  1.2010987             1.304357         19.80173
## IFATHER     9.4159226 -0.9525001             8.055968         31.98522
## INCOME     12.2294913  4.7666007            13.825365         59.90346
## GOVTPROG   10.8882297  1.5089353            10.887644         26.26350
```

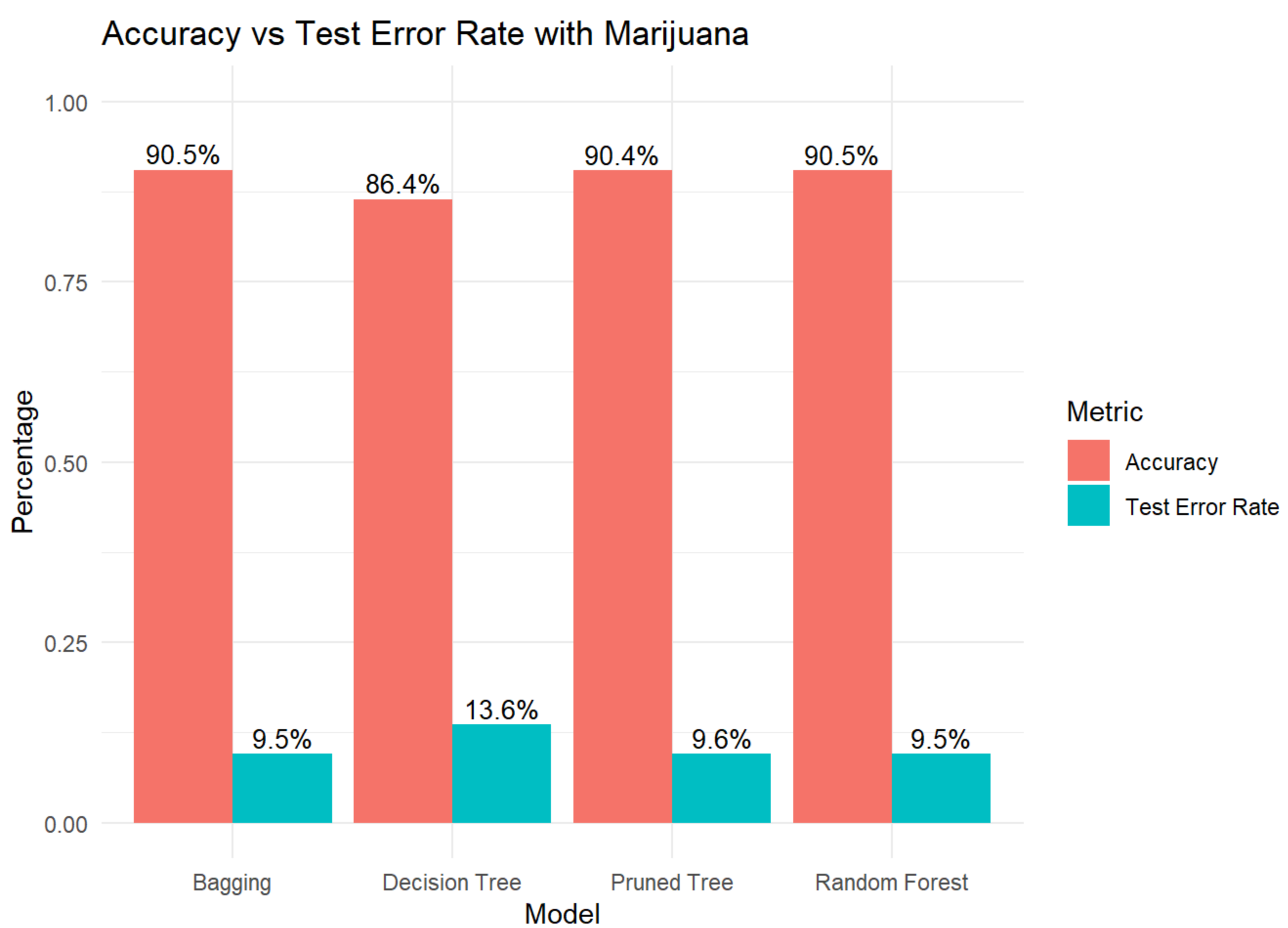
```
varImpPlot(rf_youth3, n.var = 10, sort = TRUE, main = 'The Most Important 10 Variables_ALCFLAG_R
andom Forest')
```



5.1 Compare Binary Classification Methods —> Tree with MRJFLAG –> Marijuana ever used (0 = never, 1

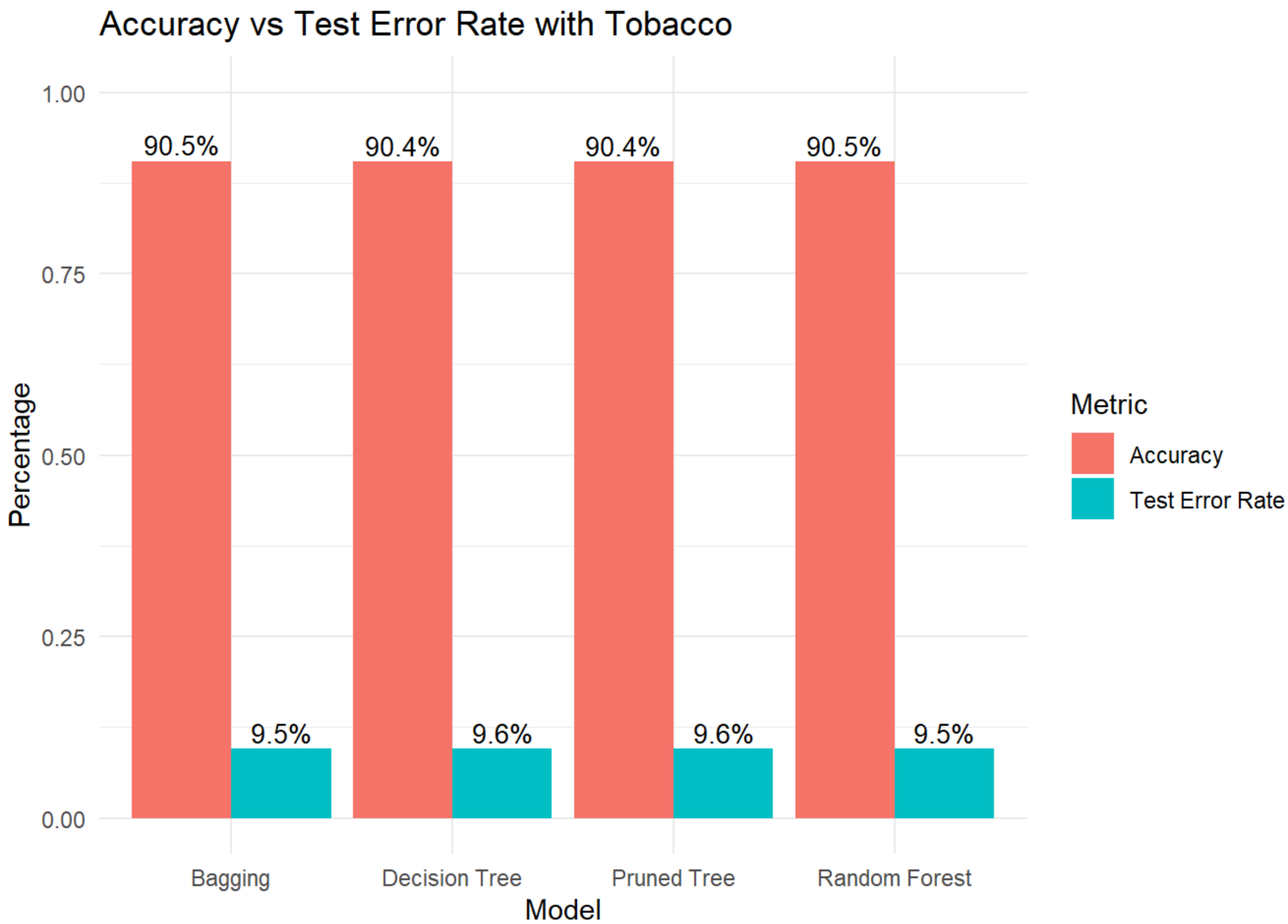
= ever)

```
library(ggplot2)

model_names <- c("Decision Tree", "Pruned Tree", "Bagging", "Random Forest")
accuracy_values <- c(accuracy_dt, accuracy_pru2, accuracy_bag2, accuracy_rf2)
error_rate_values <- c(test_error_rate_dt, test_error_rate_pru2, test_error_rate_bag2, test_erro
r_rate_rf2)

comparison_df <- data.frame(
  Model = rep(model_names, times = 2),
  Metric = rep(c("Accuracy", "Test Error Rate"), each = length(model_names)),
  Value = c(accuracy_values, error_rate_values)
)

ggplot(comparison_df, aes(x = Model, y = Value, fill = Metric)) +
  geom_bar(stat = "identity", position = "dodge") +
  geom_text(aes(label = paste0(round(Value * 100, 1), "%")),
            position = position_dodge(width = 0.9),
            vjust = -0.3, size = 3.5) +
  labs(title = "Accuracy vs Test Error Rate with Marijuana",
       x = "Model", y = "Percentage") +
  theme_minimal() +
  ylim(0, 1)
```



5.2 Compare Binary Classification Methods —> Tree with TOBFLAG –> any Tobacco ever used (0 = never, 1 = ever)

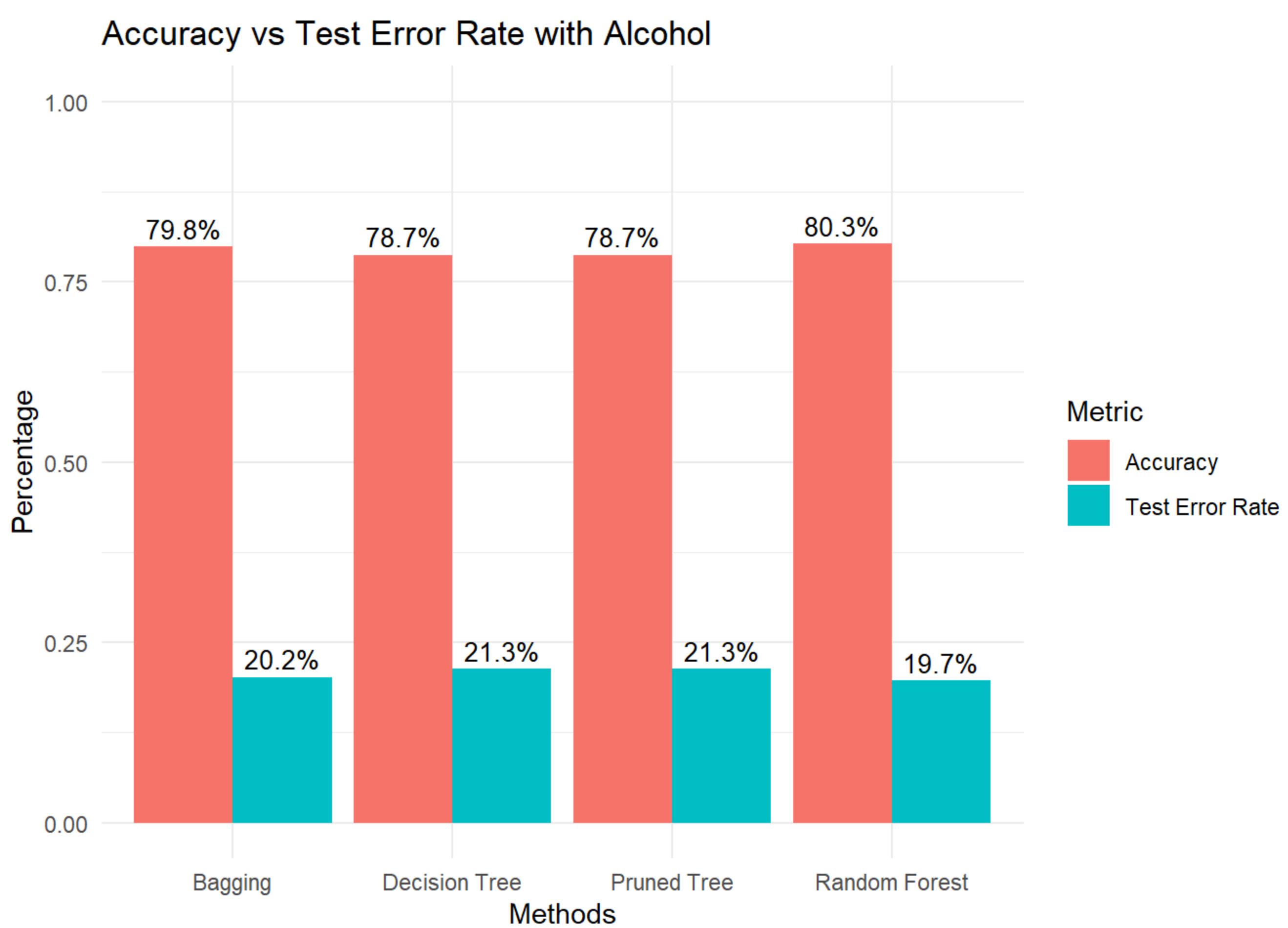
```
library(ggplot2)

model_names <- c("Decision Tree", "Pruned Tree", "Bagging", "Random Forest")
accuracy_values <- c(accuracy_tobflag, accuracy_pru2, accuracy_bag2, accuracy_rf2)
error_rate_values <- c(test_error_rate_tobflag, test_error_rate_pru2, test_error_rate_bag2, test
_error_rate_rf2)

comparison_df <- data.frame(
  Model = rep(model_names, times = 2),
  Metric = rep(c("Accuracy", "Test Error Rate"), each = length(model_names)),
  Value = c(accuracy_values, error_rate_values)
)

# Bar plot with percentage labels
ggplot(comparison_df, aes(x = Model, y = Value, fill = Metric)) +
  geom_bar(stat = "identity", position = "dodge") +
  geom_text(aes(label = paste0(round(Value * 100, 1), "%")),
            position = position_dodge(width = 0.9),
            vjust = -0.3, size = 3.5) +
  labs(title = "Accuracy vs Test Error Rate with Tobacco",
       x = "Model", y = "Percentage") +
  theme_minimal() +
  ylim(0, 1)
```



5.3 Compare Binary Classification Methods —> Tree with ALCFLAG –> Alcohol ever used (0 = never, 1 =

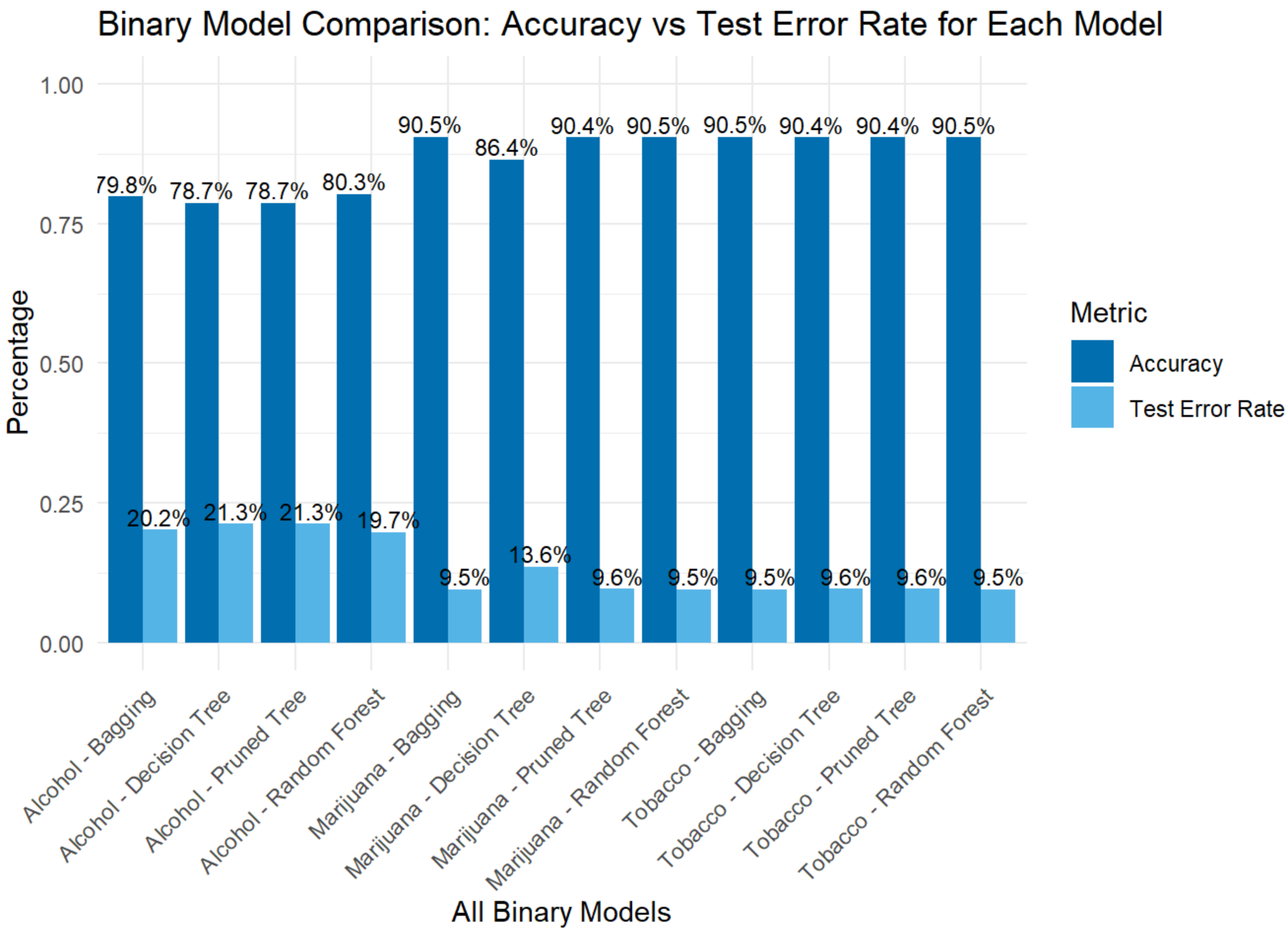
ever)

```
library(ggplot2)

model_names <- c("Decision Tree", "Pruned Tree", "Bagging", "Random Forest")
accuracy_values <- c(accuracy_alcflag, accuracy_pru3, accuracy_bag3, accuracy_rf3)
error_rate_values <- c(test_error_rate_alcflag, test_error_rate_pru3, test_error_rate_bag3, test
_error_rate_rf3)

comparison_df <- data.frame(
  Model = rep(model_names, times = 2),
  Metric = rep(c("Accuracy", "Test Error Rate"), each = length(model_names)),
  Value = c(accuracy_values, error_rate_values)
)

ggplot(comparison_df, aes(x = Model, y = Value, fill = Metric)) +
  geom_bar(stat = "identity", position = position_dodge()) +
  geom_text(aes(label = paste0(round(Value * 100, 1), "%")),
            position = position_dodge(width = 0.9),
            vjust = -0.4, size = 3.5) +
  labs(title = "Accuracy vs Test Error Rate with Alcohol",
       x = "Methods", y = "Percentage") +
  theme_minimal() +
  ylim(0, 1)
```



5.4 Binary Model Comparison: Accuracy vs Test Error Rate for Each Model

```
library(ggplot2)

model_names <- c("Decision Tree", "Pruned Tree", "Bagging", "Random Forest")

accuracy_marijuana <- c(accuracy_dt, accuracy_pru2, accuracy_bag2, accuracy_rf2)
error_marijuana    <- c(test_error_rate_dt, test_error_rate_pru2, test_error_rate_bag2, test_err
or_rate_rf2)

accuracy_tobacco <- c(accuracy_tobflag, accuracy_pru2, accuracy_bag2, accuracy_rf2)
error_tobacco    <- c(test_error_rate_tobflag, test_error_rate_pru2, test_error_rate_bag2, test_
error_rate_rf2)

accuracy_alcohol <- c(accuracy_alcflag, accuracy_pru3, accuracy_bag3, accuracy_rf3)
error_alcohol    <- c(test_error_rate_alcflag, test_error_rate_pru3, test_error_rate_bag3, test_
error_rate_rf3)

substances <- c("Marijuana", "Tobacco", "Alcohol")
x_labels <- paste(rep(substances, each = length(model_names)),
                  rep(model_names, times = length(substances)),
                  sep = " - ")

comparison_df <- data.frame(
  Method = rep(x_labels, times = 2),
  Metric = rep(c("Accuracy", "Test Error Rate"), each = length(x_labels)),
  Value = c(accuracy_marijuana, accuracy_tobacco, accuracy_alcohol,
            error_marijuana, error_tobacco, error_alcohol)
)

custom_palette <- c("Accuracy" = "#0072B2", "Test Error Rate" = "#56B4E9")

# Plot
ggplot(comparison_df, aes(x = Method, y = Value, fill = Metric)) +
  geom_bar(stat = "identity", position = position_dodge()) +
  geom_text(aes(label = paste0(round(Value * 100, 1), "%")),
            position = position_dodge(width = 0.9),
            vjust = -0.25, size = 3) +
  scale_fill_manual(values = custom_palette) +
  labs(title = "Binary Model Comparison: Accuracy vs Test Error Rate for Each Model",
       x = "All Binary Models", y = "Percentage") +
  theme_minimal() +
  ylim(0, 1) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



6. The Subset of Best 22 features from the Binary

Models

```
url <- "https://raw.githubusercontent.com/mendible/5322/main/Homework%201/youth_data.Rdata"
download.file(url, destfile = "youth_data.Rdata", mode = "wb")

loaded_dataset <- load("youth_data.Rdata")
youth_data_df <- get(loaded_dataset[1])

cleaned_youth_data <- na.omit(youth_data_df)
youth <- cleaned_youth_data

if (!"PRADLY2" %in% names(youth)) {
  youth$PRADLY2 <- NA
}

important_vars <- c(
  "YFLMJMO", "FRDMJMON", "YFLTMRJ2", "EDUSCHGRD2", "STNDSMJ",
  "FRDMEVR2", "YOSTOLE2", "NEWRACE2", "HEALTH2", "EDUSKPCOM",
  "PRMJEVR2", "STNDALC", "INCOME", "FRDPCIG2", "FRDADLY2",
  "POVERTY3", "PRMJMO", "PDEN10", "YOSELL2", "COUTYP4",
  "PRALDLY2", "PRADLY2"
)

target_vars <- c("MRJFLAG", "TOBFLAG", "ALCFLAG")

all_vars <- unique(c(important_vars, target_vars))
existing_vars <- all_vars[all_vars %in% names(youth)]

youth_fltrd <- youth[, existing_vars]
#youth_fltrd
```

7.1 Multi-class Classification —> (differentiate between seldom, sometimes, and frequent marijuana use) —> MRJYDAYS —> Number of days of marijuana in past month (1-4 categories, 5 = none)

```
youth_multi <- df[,c(demographic_cols, youth_experience_cols,'CIGMDAYS')]
youth_multi <- na.omit(youth_multi)
youth_multi$CIGMDAYS <- as.factor(youth_multi$CIGMDAYS)
#youth_multi
```

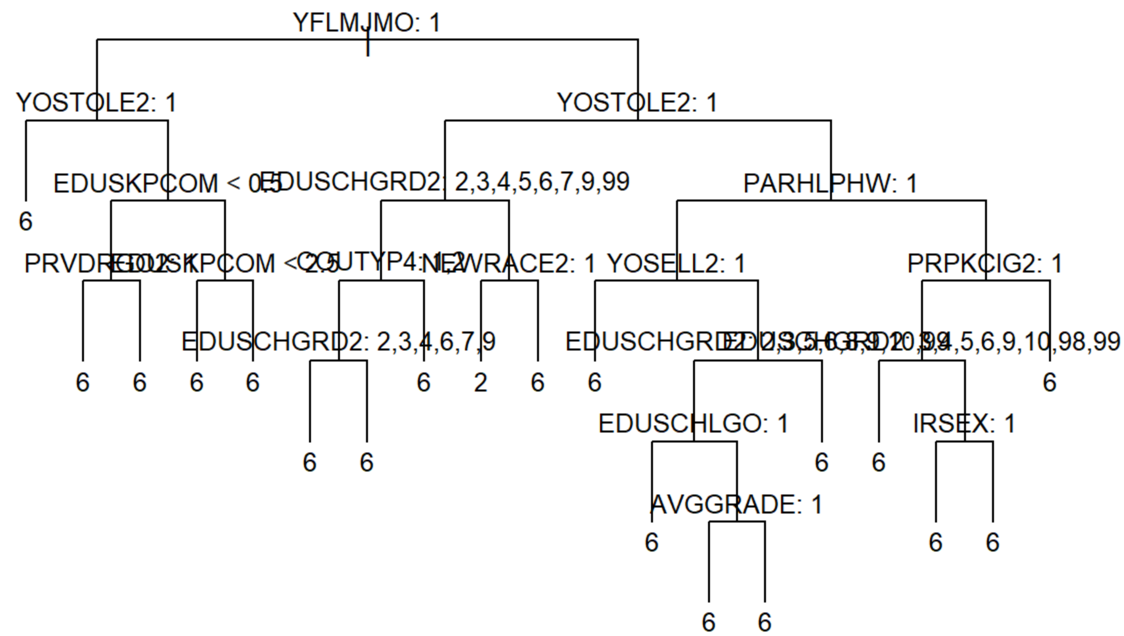
```
train_indices <- sample(1:nrow(youth_multi), 0.7*nrow(youth_multi))
train_multi <- youth_multi[train_indices,]
test_multi <- youth_multi[-train_indices,]
```

```
tree_multi <- tree(CIGMDAYS ~., data = train_multi)
tree_multi
```

```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
##   1) root 5774 1093.000 6 ( 0.0079667 0.0032906 0.0022515 0.0006928 0.0005196 0.9852788 )
##     2) YFLMJMO: 1 4436  356.700 6 ( 0.0036069 0.0004509 0.0013526 0.0000000 0.0002254 0.99436
43 )
##       4) YOSTOLE2: 1 116   61.250 6 ( 0.0431034 0.0000000 0.0172414 0.0000000 0.0000000 0.939
6552 ) *
##       5) YOSTOLE2: 2 4320  270.700 6 ( 0.0025463 0.0004630 0.0009259 0.0000000 0.0002315 0.99
58333 )
##        10) EDUSKPCOM < 0.5 2582   68.070 6 ( 0.0003873 0.0007746 0.0003873 0.0000000 0.000000
0 0.9984508 )
##          20) PRVDRGO2: 1 206   37.920 6 ( 0.0048544 0.0048544 0.0048544 0.0000000 0.0000000
0.9854369 ) *
##          21) PRVDRGO2: 2 2376   17.550 6 ( 0.0000000 0.0004209 0.0000000 0.0000000 0.0000000
0.9995791 ) *
##        11) EDUSKPCOM > 0.5 1738  184.100 6 ( 0.0057537 0.0000000 0.0017261 0.0000000 0.000575
4 0.9919448 )
##          22) EDUSKPCOM < 2.5 428   75.260 6 ( 0.0046729 0.0000000 0.0070093 0.0000000 0.00233
64 0.9859813 ) *
##          23) EDUSKPCOM > 2.5 1310   97.520 6 ( 0.0061069 0.0000000 0.0000000 0.0000000 0.0000
000 0.9938931 ) *
##     3) YFLMJMO: 2 1338  639.600 6 ( 0.0224215 0.0127055 0.0052317 0.0029895 0.0014948 0.95515
70 )
##       6) YOSTOLE2: 1 125  157.400 6 ( 0.0560000 0.0400000 0.0320000 0.0240000 0.0000000 0.848
0000 )
##        12) EDUSCHGRD2: 2,3,4,5,6,7,9,99 114  111.000 6 ( 0.0614035 0.0263158 0.0175439 0.0087
719 0.0000000 0.8859649 )
##          24) COUTYP4: 1,2 97   61.430 6 ( 0.0515464 0.0103093 0.0000000 0.0103093 0.0000000
0.9278351 )
##            48) EDUSCHGRD2: 2,3,4,6,7,9 60   10.170 6 ( 0.0000000 0.0166667 0.0000000 0.000000
0 0.0000000 0.9833333 ) *
##            49) EDUSCHGRD2: 5,99 37   38.210 6 ( 0.1351351 0.0000000 0.0000000 0.0270270 0.000
0000 0.8378378 ) *
##          25) COUTYP4: 3 17   35.260 6 ( 0.1176471 0.1176471 0.1176471 0.0000000 0.0000000 0.6
470588 ) *
##        13) EDUSCHGRD2: 8,98 11   28.340 6 ( 0.0000000 0.1818182 0.1818182 0.1818182 0.0000000
0.4545455 )
##          26) NEWRACE2: 1 5   10.550 2 ( 0.0000000 0.4000000 0.2000000 0.4000000 0.0000000 0.0
000000 ) *
##          27) NEWRACE2: 2,7 6    5.407 6 ( 0.0000000 0.0000000 0.1666667 0.0000000 0.0000000
0.8333333 ) *
##       7) YOSTOLE2: 2 1213  449.600 6 ( 0.0189613 0.0098928 0.0024732 0.0008244 0.0016488 0.96
61995 )
##        14) PARHLPHW: 1 848  218.200 6 ( 0.0106132 0.0094340 0.0000000 0.0000000 0.0023585 0.9
775943 )
##          28) YOSELL2: 1 23   21.250 6 ( 0.0000000 0.1739130 0.0000000 0.0000000 0.0000000 0.8
260870 ) *
##          29) YOSELL2: 2 825  177.800 6 ( 0.0109091 0.0048485 0.0000000 0.0000000 0.0024242 0.
9818182 )
##            58) EDUSCHGRD2: 2,3,5,6,8,9,10,99 579   74.410 6 ( 0.0034542 0.0069085 0.0000000
0.0000000 0.0000000 0.9896373 )
```

```
##             116) EDUSCHLGO: 1 489   14.380 6 ( 0.0000000 0.0020450 0.0000000 0.0000000 0.0000
000 0.9979550 ) *
##             117) EDUSCHLGO: 2,11 90   45.350 6 ( 0.0222222 0.0333333 0.0000000 0.0000000 0.00
00000 0.9444444 )
##               234) AVGGRADE: 1 13   20.550 6 ( 0.0769231 0.2307692 0.0000000 0.0000000 0.0000
000 0.6923077 ) *
##               235) AVGGRADE: 2 77   10.670 6 ( 0.0129870 0.0000000 0.0000000 0.0000000 0.0000
000 0.9870130 ) *
##            59) EDUSCHGRD2: 4,7,98 246   86.750 6 ( 0.0284553 0.0000000 0.0000000 0.0000000 0.
0081301 0.9634146 ) *
##        15) PARHLPHW: 2 365  210.700 6 ( 0.0383562 0.0109589 0.0082192 0.0027397 0.0000000 0.9
397260 )
##          30) PRPKCIG2: 1 313  121.900 6 ( 0.0287540 0.0031949 0.0031949 0.0031949 0.0000000
0.9616613 )
##            60) EDUSCHGRD2: 3,4,5,6,9,10,98,99 166   12.220 6 ( 0.0000000 0.0060241 0.0000000
0.0000000 0.0000000 0.9939759 ) *
##            61) EDUSCHGRD2: 7,8 147   91.400 6 ( 0.0612245 0.0000000 0.0068027 0.0068027 0.000
0000 0.9251701 )
##             122) IRSEX: 1 64   20.570 6 ( 0.0000000 0.0000000 0.0156250 0.0156250 0.0000000
0.9687500 ) *
##             123) IRSEX: 2 83   56.980 6 ( 0.1084337 0.0000000 0.0000000 0.0000000 0.0000000
0.8915663 ) *
##          31) PRPKCIG2: 2 52   71.510 6 ( 0.0961538 0.0576923 0.0384615 0.0000000 0.0000000 0.
8076923 ) *
```

```
plot(tree_multi,type = 'uniform')
text(tree_multi, pretty=0, cex  = 0.8)
```




```
pred_multi <- predict(tree_multi, test_multi, type = 'class')

confusion_matrix_dt <- table(Predicted = pred_multi, Actual = test_multi$CIGMDAYS)
confusion_matrix_dt
```

```
##          Actual
## Predicted    1    2    3    4    5    6
##         1    0    0    0    0    0    0
##         2    0    0    0    1    0    2
##         3    0    0    0    0    0    0
##         4    0    0    0    0    0    2
##         5    0    0    0    0    0    0
##         6   24    4    3    3    1 2435
```

```
accuracy_dt <- mean(pred_multi == test_multi$CIGMDAYS)
test_error_rate_dt <- 1 - accuracy_dt

cat('Accuracy:', round(accuracy_dt, 4), "\n")
```

```
## Accuracy: 0.9838
```

```
cat('Test Error Rate:', round(test_error_rate_dt, 4), "\n")
```

```
## Test Error Rate: 0.0162
```

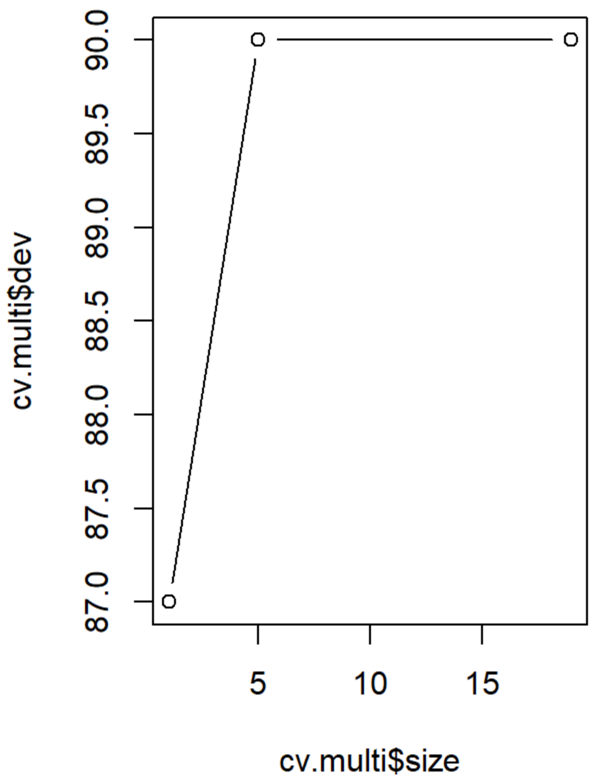
```
cv.multi = cv.tree(tree_multi, FUN = prune.misclass)
names(cv.multi)
```

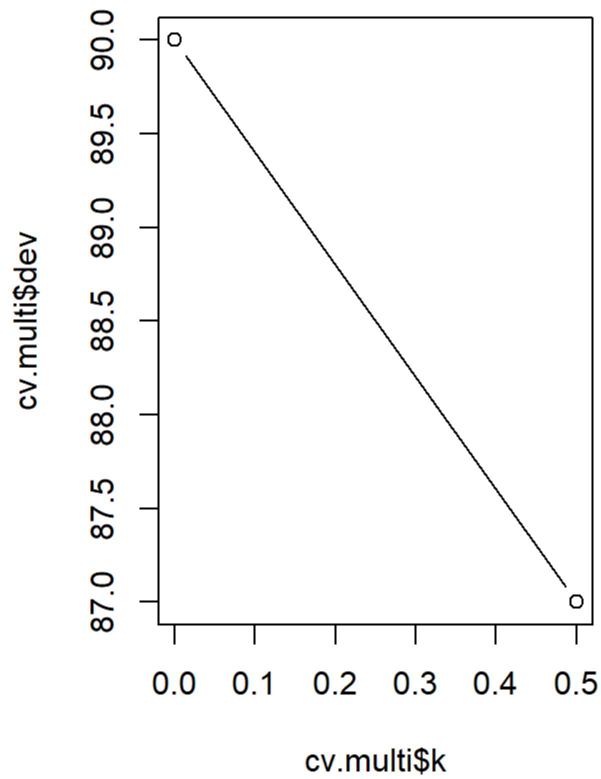
```
## [1] "size"   "dev"    "k"      "method"
```

```
cv.multi
```

```
## $size
## [1] 19  5  1
##
## $dev
## [1] 90 90 87
##
## $k
## [1] -Inf  0.0  0.5
##
## $method
## [1] "misclass"
##
## attr(,"class")
## [1] "prune"         "tree.sequence"
```

```
par(mfrow = c(1, 2))
plot(cv.multi$size, cv.multi$dev, type = "b")
plot(cv.multi$k, cv.multi$dev, type = "b")
```

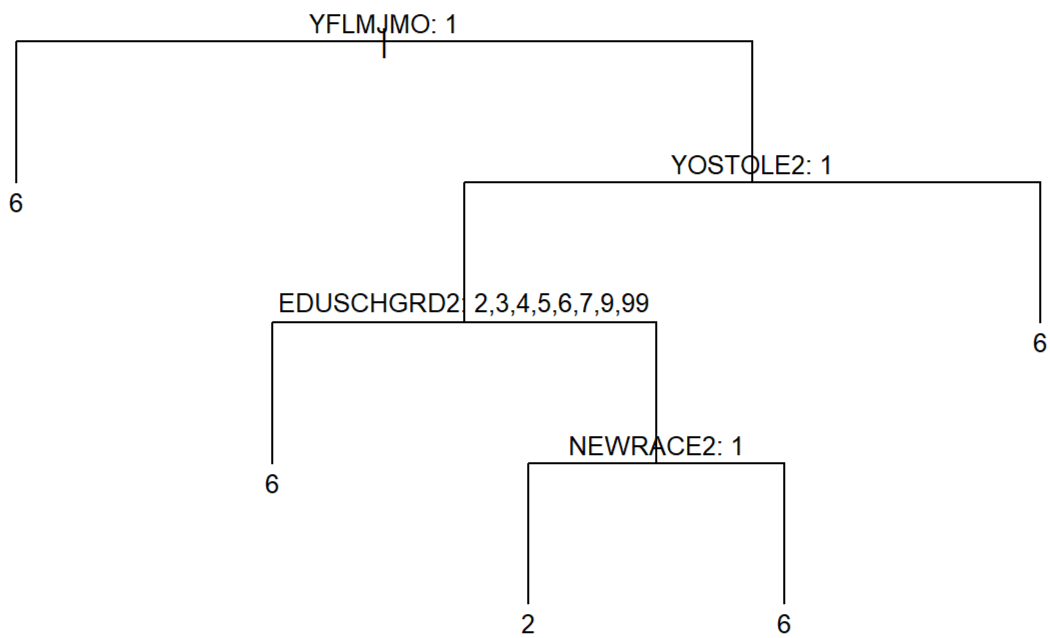





```
prune.multi = prune.misclass(tree_multi, best = 3)
prune.multi
```

```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
##  1) root 5774 1093.000 6 ( 0.0079667 0.0032906 0.0022515 0.0006928 0.0005196 0.9852788 )
##    2) YFLMJMO: 1 4436  356.700 6 ( 0.0036069 0.0004509 0.0013526 0.0000000 0.0002254 0.994364
3 ) *
##    3) YFLMJMO: 2 1338  639.600 6 ( 0.0224215 0.0127055 0.0052317 0.0029895 0.0014948 0.955157
0 )
##      6) YOSTOLE2: 1 125  157.400 6 ( 0.0560000 0.0400000 0.0320000 0.0240000 0.0000000 0.8480
000 )
##       12) EDUSCHGRD2: 2,3,4,5,6,7,9,99 114  111.000 6 ( 0.0614035 0.0263158 0.0175439 0.00877
19 0.0000000 0.8859649 ) *
##       13) EDUSCHGRD2: 8,98 11   28.340 6 ( 0.0000000 0.1818182 0.1818182 0.1818182 0.0000000
0.4545455 )
##         26) NEWRACE2: 1 5   10.550 2 ( 0.0000000 0.4000000 0.2000000 0.4000000 0.0000000 0.00
00000 ) *
##         27) NEWRACE2: 2,7 6    5.407 6 ( 0.0000000 0.0000000 0.1666667 0.0000000 0.0000000 0.
8333333 ) *
##      7) YOSTOLE2: 2 1213  449.600 6 ( 0.0189613 0.0098928 0.0024732 0.0008244 0.0016488 0.966
1995 ) *
```

```
plot(prune.multi, type ='uniform')
text(prune.multi, pretty = 0, cex = 0.8)
```




```
prune_pred_multi <- predict(prune.multi, test_multi, type = 'class')

confusion_matrix_pru <- table(Predicted = prune_pred_multi, Actual = test_multi$CIGMDAYS)
confusion_matrix_pru
```

```
##          Actual
## Predicted    1    2    3    4    5    6
##         1    0    0    0    0    0    0
##         2    0    0    0    0    0    1
##         3    0    0    0    0    0    0
##         4    0    0    0    1    0    3
##         5    0    0    0    0    0    0
##         6   24    4    3    3    1 2435
```

```
accuracy_pru <- mean(prune_pred_multi == test_multi$CIGMDAYS)
cat('Accuracy:', round(accuracy_pru, 4), "\n")
```

```
## Accuracy: 0.9842
```

```
test_error_rate_pru <- 1 - accuracy_pru
cat('Test Error Rate:', round(test_error_rate_pru, 4), "\n")
```

```
## Test Error Rate: 0.0158
```

7.2 Multi-class Classification —> Bagging —> CIGMDAYS

```
library(randomForest)

train_multi_clean = na.omit(train_multi)
bag_multi = randomForest(CIGMDAYS ~ ., data = train_multi_clean, mtry = (sqrt(ncol(train_multi_c
lean))), importance = TRUE)
bag_multi
```

```
##
## Call:
##  randomForest(formula = CIGMDAYS ~ ., data = train_multi_clean,      mtry = (sqrt(ncol(train_
multi_clean))), importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 8
##
##         OOB estimate of  error rate: 1.47%
## Confusion matrix:
##   1 2 3 4 5    6 class.error
## 1 0 0 0 0 0   46           1
## 2 0 0 0 0 0   19           1
## 3 0 0 0 0 0   13           1
## 4 0 0 0 0 0    4           1
## 5 0 0 0 0 0    3           1
## 6 0 0 0 0 0 5689           0
```

```
pred_bag_multi <- predict(bag_multi, newdata = test_multi, type = 'class')

confusion_matrix_bag <- table(Predicted = pred_bag_multi, Actual = test_multi$CIGMDAYS)
confusion_matrix_bag
```

```
##          Actual
## Predicted    1    2    3    4    5    6
##         1    0    0    0    0    0    0
##         2    0    0    0    0    0    0
##         3    0    0    0    0    0    0
##         4    0    0    0    0    0    0
##         5    0    0    0    0    0    0
##         6   24    4    3    4    1 2439
```

```
accuracy_bag <- mean(pred_bag_multi == test_multi$CIGMDAYS, na.rm = TRUE)
cat('Accuracy:', round(accuracy_bag, 4), "\n")
```

```
## Accuracy: 0.9855
```

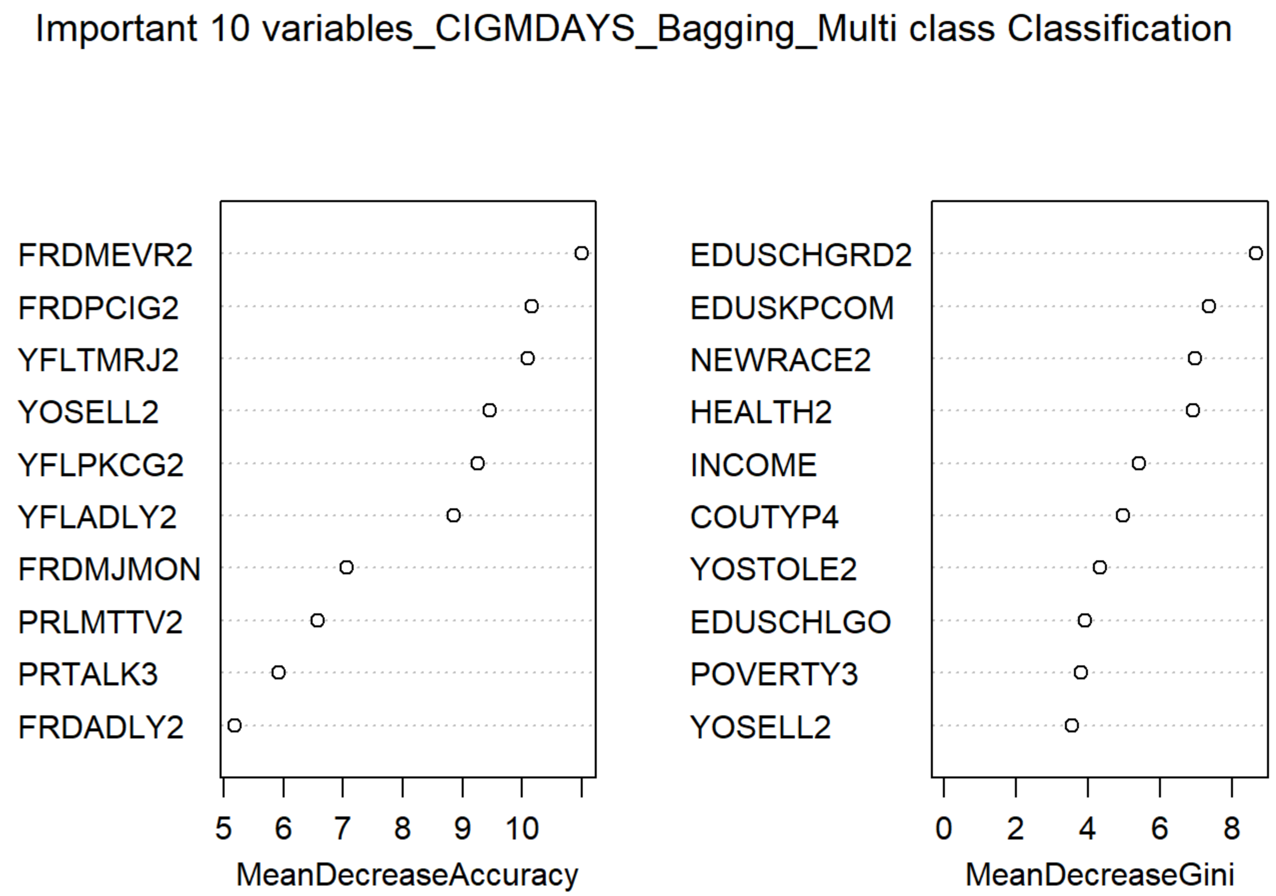
```
test_error_rate_bag <- 1 - accuracy_bag
cat('Test Error Rate:', round(test_error_rate_bag, 4), "\n")
```

```
## Test Error Rate: 0.0145
```

```
top_10_bag_CIGMDAYS = head(importance(bag_multi),10)
top_10_bag_CIGMDAYS
```

```
##                      1           2          3          4         5          6
## IRSEX      -0.88728815 -1.52299120  1.0010015  1.0010015  0.000000  1.0689759
## NEWRACE2    2.13667619  0.51854586 -0.1280390  0.0000000  0.000000  0.6770423
## HEALTH2    -0.76699898  1.54949343  0.4774432  0.0000000  0.000000  0.1536375
## EDUSCHLGO  -0.25269949  1.24225999 -1.0010015  0.0000000  0.000000  0.5226216
## EDUSCHGRD2  1.99920207 -0.09943165  2.4402337  1.4170505  0.000000 -0.2244138
## EDUSKPCOM   1.59853550 -0.98084246 -1.4128599  1.4170505  0.000000  3.2792677
## IMOTHER     0.64543131 -2.13309751 -1.7135103  1.9042342  0.000000 -3.0755575
## IFATHER     2.05094574 -1.23545479  0.0000000  0.0000000  0.000000 -0.7132478
## INCOME      0.19102726 -0.48904922 -1.2834337 -0.3333704 -1.344062  2.1853153
## GOVTPROG    0.08038355  1.36183411  1.0010015  0.0000000  0.000000  0.1284930
##            MeanDecreaseAccuracy MeanDecreaseGini
## IRSEX                0.99379865         2.496852
## NEWRACE2             0.98244976         6.961897
## HEALTH2              0.14212444         6.908483
## EDUSCHLGO            0.55078452         3.909716
## EDUSCHGRD2           0.06446117         8.645055
## EDUSKPCOM            3.36331738         7.354988
## IMOTHER             -3.12511815         2.787450
## IFATHER             -0.55499518         2.214909
## INCOME               2.01786627         5.412274
## GOVTPROG             0.27413595         2.069601
```

```
varImpPlot(bag_multi, n.var = 10, sort = TRUE, main = 'Important 10 variables_CIGMDAYS_Bagging_M
ulti class Classification')
```



7.3 Multi-class Classification —> RandomForest —> CIGMDAYS

```
set.seed(1)
rf_multi = randomForest(CIGMDAYS ~ ., data = train_multi_clean, mtry = sqrt(ncol(train_multi_cle
an)), importance = TRUE)
rf_multi
```

```
##
## Call:
##  randomForest(formula = CIGMDAYS ~ ., data = train_multi_clean,      mtry = sqrt(ncol(train_m
ulti_clean)), importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 8
##
##         OOB estimate of  error rate: 1.47%
## Confusion matrix:
##   1 2 3 4 5    6 class.error
## 1 0 0 0 0 0   46           1
## 2 0 0 0 0 0   19           1
## 3 0 0 0 0 0   13           1
## 4 0 0 0 0 0    4           1
## 5 0 0 0 0 0    3           1
## 6 0 0 0 0 0 5689           0
```

```
yhat.rf <- predict(rf_multi, newdata = test_multi, type = 'class')

confusion_matrix_rf <- table(Predicted = yhat.rf, Actual = test_multi$CIGMDAYS)
confusion_matrix_rf
```

```
##          Actual
## Predicted    1    2    3    4    5    6
##         1    0    0    0    0    0    0
##         2    0    0    0    0    0    0
##         3    0    0    0    0    0    0
##         4    0    0    0    0    0    0
##         5    0    0    0    0    0    0
##         6   24    4    3    4    1 2439
```

```
accuracy_rf <- mean(yhat.rf == test_multi$CIGMDAYS, na.rm = TRUE)
cat('Accuracy:', round(accuracy_rf, 4), "\n")
```

```
## Accuracy: 0.9855
```

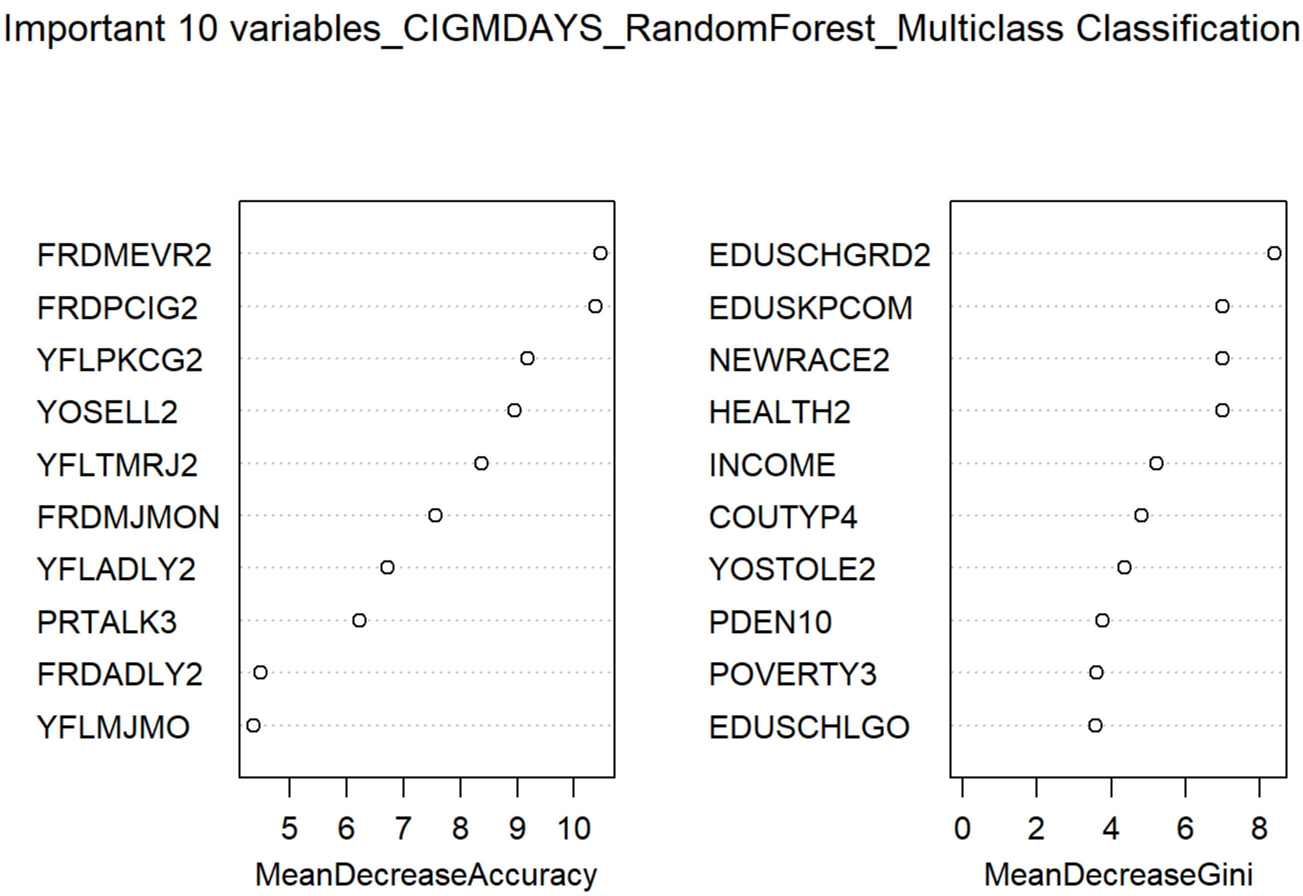
```
test_error_rate_rf <- 1 - accuracy_rf
cat('Test Error Rate:', round(test_error_rate_rf, 4), "\n")
```

```
## Test Error Rate: 0.0145
```

```
top_10_rf_CIGMDAYS = head(importance(rf_multi),10)
top_10_rf_CIGMDAYS
```

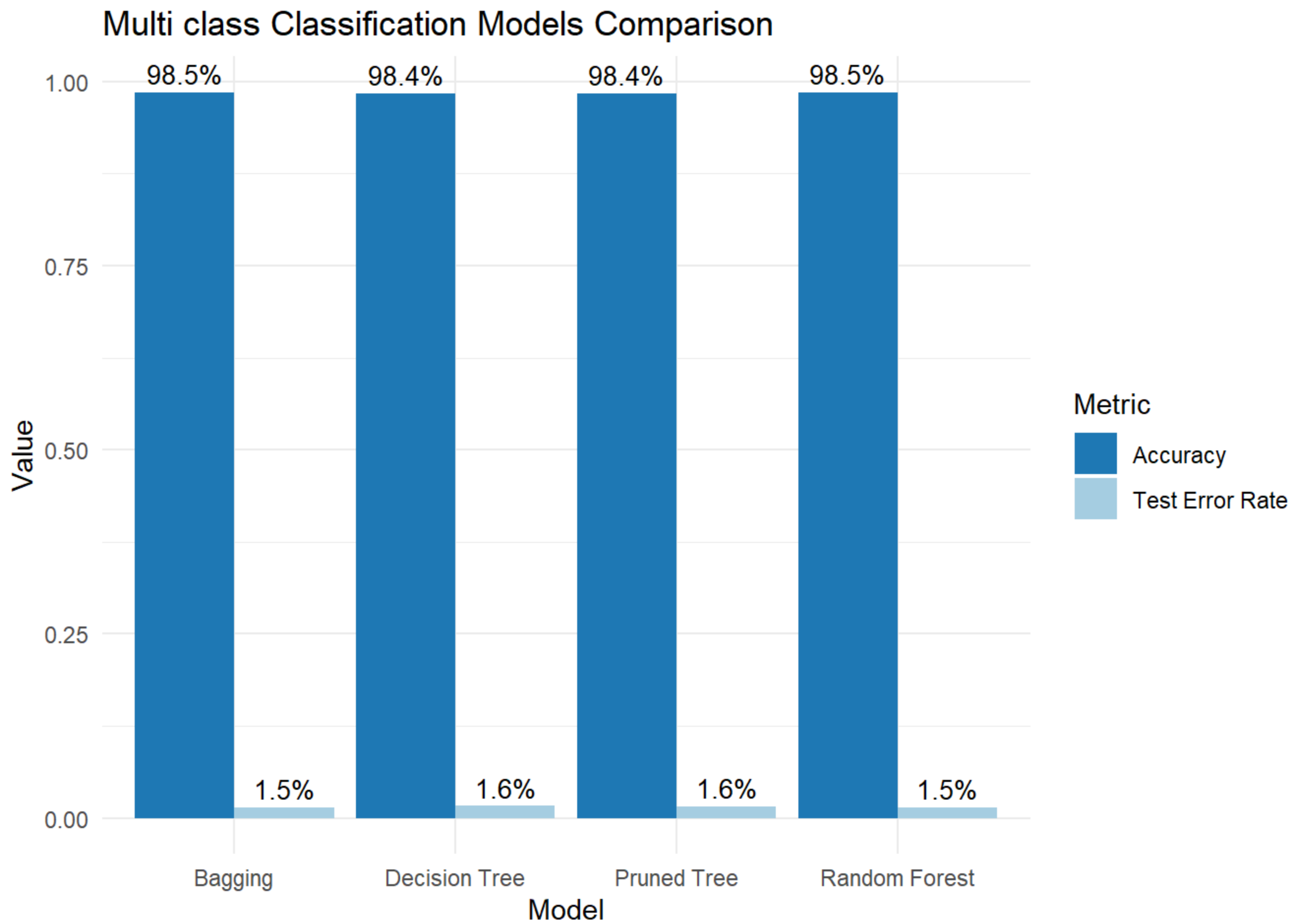
```
##                      1           2           3          4         5          6
## IRSEX      -0.63961160 -0.07475622 -1.41285987  0.0000000  0.000000 -2.1110812
## NEWRACE2   -0.06430056 -0.57544850  1.00100150  0.0000000  1.904234 -1.7756153
## HEALTH2     0.70402629  1.13155122  1.44867141 -1.0010015  0.000000 -0.3320352
## EDUSCHLGO  -1.14968313 -0.37061147 -1.00100150  0.0000000  0.000000  3.0697411
## EDUSCHGRD2  1.60672139  1.04543234  1.05173535  1.0010015  0.000000  3.5622050
## EDUSKPCOM   1.49427774 -0.37175697 -1.41285987 -0.8170415  1.417051  2.1072905
## IMOTHER     0.75661646  0.41041638 -0.82255090  0.0000000  0.000000 -2.8465231
## IFATHER    -0.19958832 -0.39709593  0.00000000 -1.0010015  0.000000  1.6300513
## INCOME     -0.08170820 -1.50508061  0.08774398  0.0000000 -1.001002  2.9499131
## GOVTPROG   -1.35351624  1.08161038 -0.20000800  0.0000000  1.344062 -0.9525729
##            MeanDecreaseAccuracy MeanDecreaseGini
## IRSEX                -2.1952872         2.325348
## NEWRACE2             -1.7142229         7.004674
## HEALTH2              -0.1451907         7.000369
## EDUSCHLGO             2.9657259         3.591197
## EDUSCHGRD2            3.7993762         8.403823
## EDUSKPCOM             2.1252031         7.017269
## IMOTHER              -2.8170829         2.707809
## IFATHER               1.5168669         2.106479
## INCOME                2.8391406         5.238096
## GOVTPROG             -0.9270224         2.043119
```

```
varImpPlot(rf_multi, n.var = 10, sort = TRUE, main = 'Important 10 variables_CIGMDAYS_RandomFore
st_Multiclass Classification')
```


Important 10 variables_CIGMDAYS_RandomForest_Multiclass Classification
FRDMEVR2
FRDPCIG2
YFLPKCG2
YOSELL2
YFLTMRJ2
FRDMJMON
YFLADLY2
PRTALK3
FRDADLY2
YFLMJMO
5 6 7 8 9 10
MeanDecreaseAccuracy
EDUSCHGRD2
EDUSKPCOM
NEWRACE2
HEALTH2
INCOME
COUTYP4
YOSTOLE2
PDEN10
POVERTY3
EDUSCHLGO
0 2 4 6 8
MeanDecreaseGini

7.4 Compare Multi class Classification Models

```
library(ggplot2)

model_names <- c("Decision Tree", "Pruned Tree", "Bagging", "Random Forest")

accuracy_values <- c(accuracy_dt, accuracy_pru, accuracy_bag, accuracy_rf)
error_rate_values <- c(test_error_rate_dt, test_error_rate_pru, test_error_rate_bag, test_error_
rate_rf)

comparison_df <- data.frame(
  Model = rep(model_names, times = 2),
  Metric = rep(c("Accuracy", "Test Error Rate"), each = length(model_names)),
  Value = c(accuracy_values, error_rate_values)
)

comparison_df$Label <- paste0(round(comparison_df$Value * 100, 1), "%")

ggplot(comparison_df, aes(x = Model, y = Value, fill = Metric)) +
  geom_bar(stat = "identity", position = position_dodge()) +
  geom_text(aes(label = Label),
            position = position_dodge(width = 0.9),
            vjust = -0.4, size = 3.5) +
  scale_fill_manual(values = c("Accuracy" = "#1f78b4", "Test Error Rate" = "#a6cee3")) +
  labs(title = "Multi class Classification Models Comparison",
       y = "Value", x = "Model") +
  theme_minimal()
```



8.1 Regression —> Decision Tree —> IRCIGAGE —> Cigarette age of first use (1-55), 991=never used

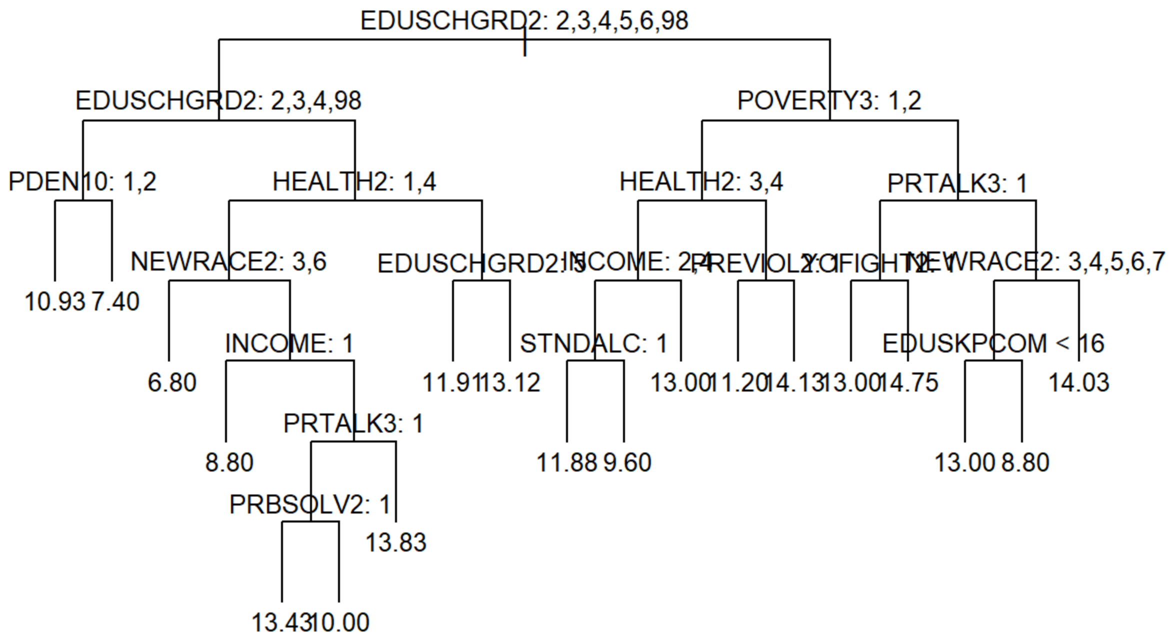
```
library(dplyr)

youth_reg <- df[, c(demographic_cols, youth_experience_cols, 'IRCIGAGE')] %>%
  filter(!is.na(IRCIGAGE) & IRCIGAGE != 991) %>%
  na.omit()
#youth_reg
```

```
train_indices <- sample(1:nrow(youth_reg), 0.7*nrow(youth_reg))
train_reg <- youth_reg[train_indices,]
test_reg <- youth_reg[-train_indices,]
```

```
library(tree)

tree_reg <- tree(IRCIGAGE ~., data = train_reg)
tree_reg
```

```
## node), split, n, deviance, yval
##       * denotes terminal node
##
##   1) root 443 3155.000 12.73
##     2) EDUSCHGRD2: 2,3,4,5,6,98 191 1225.000 11.78
##       4) EDUSCHGRD2: 2,3,4,98 48  271.800 10.56
##         8) PDEN10: 1,2 43  180.800 10.93 *
##         9) PDEN10: 3 5   35.200  7.40 *
##       5) EDUSCHGRD2: 5,6 143  857.900 12.19
##        10) HEALTH2: 1,4 40  378.400 11.20
##          20) NEWRACE2: 3,6 5   10.800  6.80 *
##          21) NEWRACE2: 1,2,5,7 35  257.000 11.83
##            42) INCOME: 1 5   18.800  8.80 *
##            43) INCOME: 2,3,4 30  184.700 12.33
##              86) PRTALK3: 1 18  128.000 11.33
##               172) PRBSOLV2: 1 7    5.714 13.43 *
##               173) PRBSOLV2: 2 11   72.000 10.00 *
##              87) PRTALK3: 2 12   11.670 13.83 *
##        11) HEALTH2: 2,3 103  425.200 12.57
##          22) EDUSCHGRD2: 5 47  219.700 11.91 *
##          23) EDUSCHGRD2: 6 56  168.100 13.12 *
##     3) EDUSCHGRD2: 7,8,9,99 252 1626.000 13.45
##       6) POVERTY3: 1,2 112  786.500 12.76
##        12) HEALTH2: 3,4 68  541.500 12.09
##          24) INCOME: 2,4 35  302.200 11.23
##            48) STNDALC: 1 25  162.600 11.88 *
##            49) STNDALC: 2 10  102.400  9.60 *
##          25) INCOME: 1,3 33  186.000 13.00 *
##        13) HEALTH2: 1,2 44  167.200 13.80
##          26) PREVIOL2: 1 5   10.800 11.20 *
##          27) PREVIOL2: 2 39  118.400 14.13 *
##       7) POVERTY3: 3 140  743.000 14.01
##        14) PRTALK3: 1 86  281.200 14.44
##          28) YOFIGHT2: 1 15   88.000 13.00 *
##          29) YOFIGHT2: 2 71  155.400 14.75 *
##        15) PRTALK3: 2 54  419.600 13.31
##          30) NEWRACE2: 3,4,5,6,7 17  179.100 11.76
##            60) EDUSKPCOM < 16 12   34.000 13.00 *
##            61) EDUSKPCOM > 16 5   82.800  8.80 *
##          31) NEWRACE2: 1,2 37  181.000 14.03 *
```

```
plot(tree_reg,type = 'uniform')
text(tree_reg, pretty=0, cex = 0.8)
```




```
pred_reg <- predict(tree_reg, test_reg)

mse_tree <- mean((pred_reg - test_reg$IRCIGAGE)^2)
cat("Mean Squared Error (MSE):", round(mse_tree, 4), "\n")
```

```
## Mean Squared Error (MSE): 6.8087
```

```
rmse_tree <- sqrt(mse_tree)
cat("Root Mean Squared Error (RMSE):", round(rmse_tree, 4), "\n")
```

```
## Root Mean Squared Error (RMSE): 2.6093
```

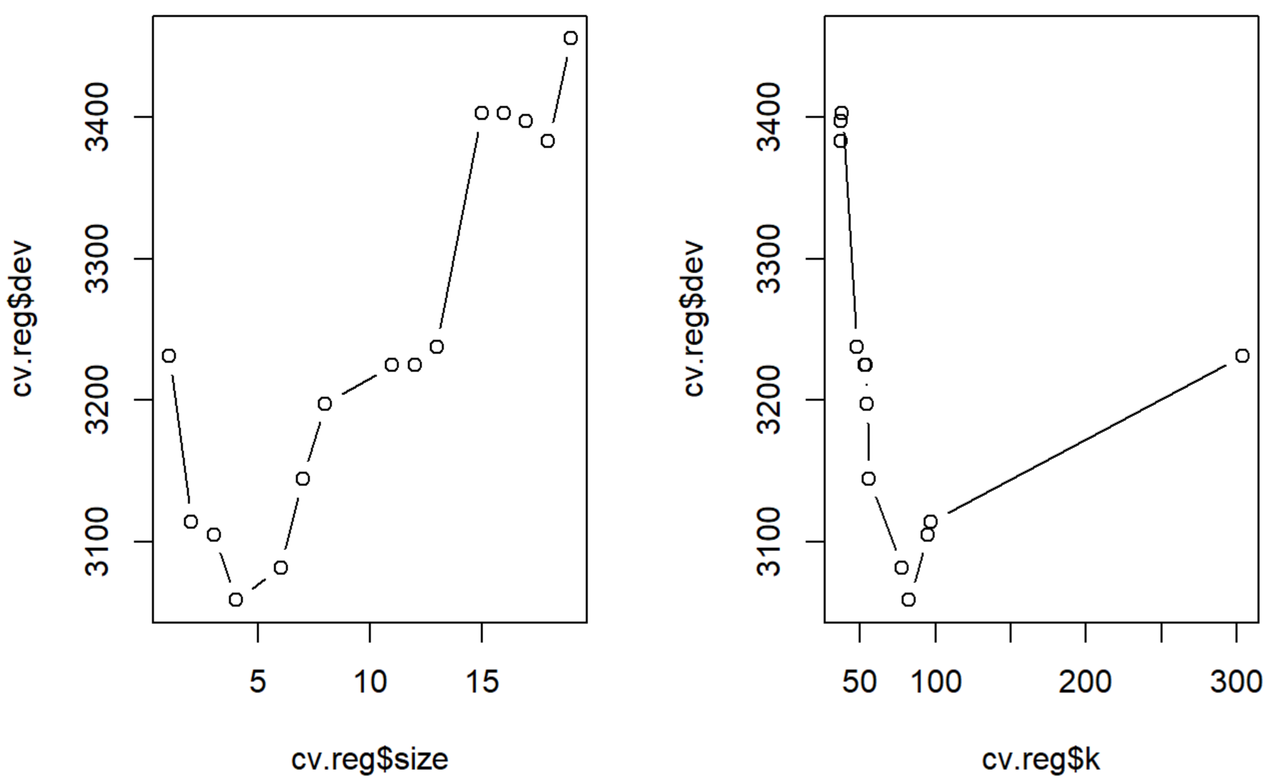
```
cv.reg = cv.tree(tree_reg, FUN = prune.tree)
names(cv.reg)
```

```
## [1] "size"   "dev"    "k"      "method"
```

```
cv.reg
```

```
## $size
##  [1] 19 18 17 16 15 13 12 11  8  7  6  4  3  2  1
##
## $dev
##  [1] 3455.430 3383.086 3397.218 3402.898 3402.898 3237.762 3224.978 3224.978
##  [9] 3197.455 3144.343 3081.615 3058.962 3105.241 3113.849 3231.313
##
## $k
##  [1]      -Inf  37.13143  37.41931  37.77268  38.00012  47.64286  53.29916
##  [8]  53.50476  54.67019  55.82180  77.86139  82.46339  95.04980  96.94464
## [15] 303.84089
##
## $method
## [1] "deviance"
##
## attr(,"class")
## [1] "prune"         "tree.sequence"
```

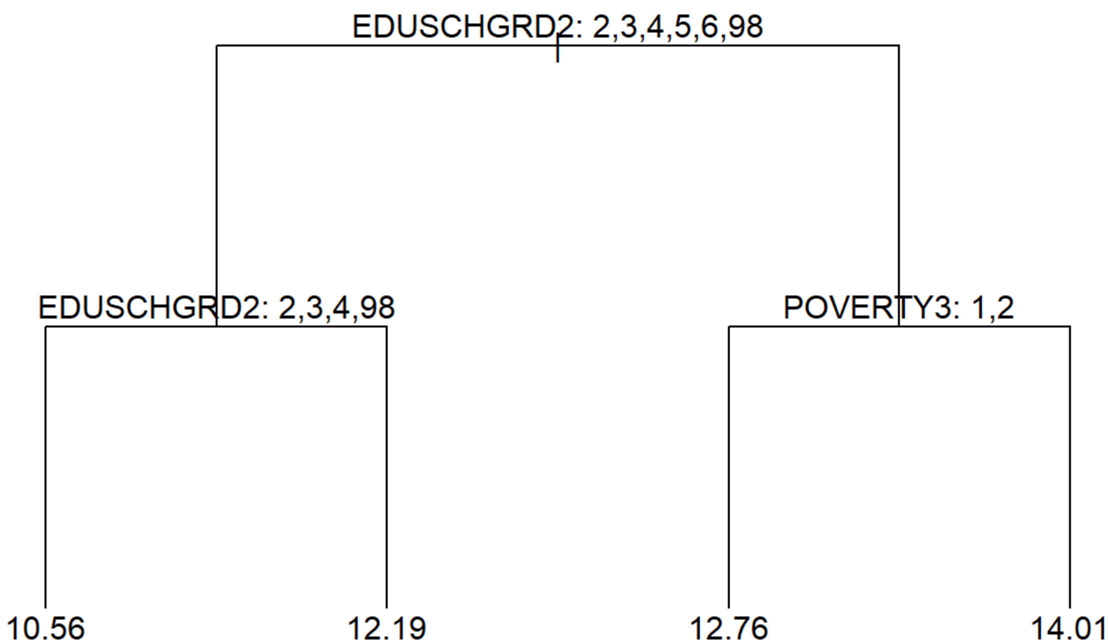
```
par(mfrow = c(1, 2))
plot(cv.reg$size, cv.reg$dev, type = "b")
plot(cv.reg$k, cv.reg$dev, type = "b")
```



```
prune.reg_tree = prune.tree(tree_reg, best = 4)
prune.reg_tree
```

```
## node), split, n, deviance, yval
##       * denotes terminal node
##
## 1) root 443 3155.0 12.73
##   2) EDUSCHGRD2: 2,3,4,5,6,98 191 1225.0 11.78
##     4) EDUSCHGRD2: 2,3,4,98 48  271.8 10.56 *
##     5) EDUSCHGRD2: 5,6 143  857.9 12.19 *
##   3) EDUSCHGRD2: 7,8,9,99 252 1626.0 13.45
##     6) POVERTY3: 1,2 112  786.5 12.76 *
##     7) POVERTY3: 3 140  743.0 14.01 *
```

```
plot(prune.reg_tree, type ='uniform')
text(prune.reg_tree, pretty = 0)
```




```
prune_pred_reg <- predict(prune.reg_tree, test_reg)

mse_prune <- mean((prune_pred_reg - test_reg$IRCIGAGE)^2)
cat("Mean Squared Error (MSE):", round(mse_prune, 4), "\n")
```

```
## Mean Squared Error (MSE): 5.8645
```

```
rmse_prune <- sqrt(mse_prune)
cat("Root Mean Squared Error (RMSE):", round(rmse_prune, 4), "\n")
```

```
## Root Mean Squared Error (RMSE): 2.4217
```

8.2 Regression —> Bagging —> IRCIGAGE —> Cigarette age of first use (1-55), 991=never used

```
library(randomForest)

train_reg_clean = na.omit(train_reg)
bag_reg = randomForest(IRCIGAGE ~ ., data = train_reg_clean, mtry = floor(ncol(train_reg)/3), im
portance = TRUE)
bag_reg
```

```
##
## Call:
##  randomForest(formula = IRCIGAGE ~ ., data = train_reg_clean,      mtry = floor(ncol(train_re
g)/3), importance = TRUE)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 20
##
##           Mean of squared residuals: 6.113398
##                     % Var explained: 14.16
```

```
pred_bag_reg <- predict(bag_reg, newdata = test_reg)

mse_bag <- mean((pred_bag_reg - test_reg$IRCIGAGE)^2, na.rm = TRUE)
cat("Mean Squared Error (MSE):", round(mse_bag, 4), "\n")
```

```
## Mean Squared Error (MSE): 5.7653
```

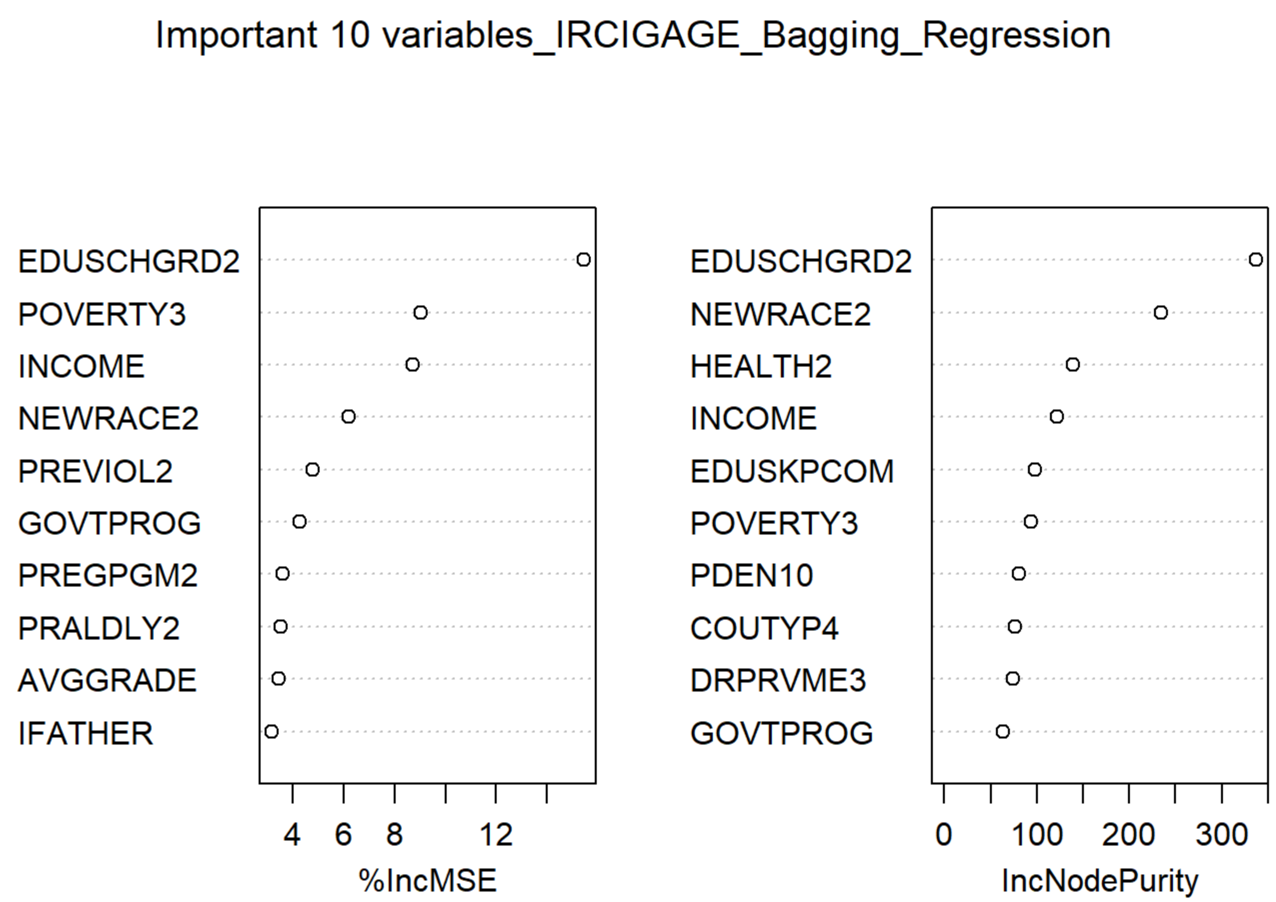
```
rmse_bag <- sqrt(mse_bag)
cat("Root Mean Squared Error (RMSE):", round(rmse_bag, 4), "\n")
```

```
## Root Mean Squared Error (RMSE): 2.4011
```

```
top_10_bag_IRCIGAGE = head(importance(bag_reg),10)
top_10_bag_IRCIGAGE
```

```
##              %IncMSE IncNodePurity
## IRSEX       0.3642390      34.96658
## NEWRACE2    6.1868160     234.85201
## HEALTH2     0.9762210     138.78330
## EDUSCHLGO   2.4141007      48.82703
## EDUSCHGRD2 15.4333234     336.68997
## EDUSKPCOM   1.8831359      98.54486
## IMOTHER    -0.4354211      24.70289
## IFATHER     3.1731748      50.04534
## INCOME      8.7278803     121.33630
## GOVTPROG    4.2599748      63.12329
```

```
varImpPlot(bag_reg, n.var = 10, sort = TRUE, main = 'Important 10 variables_IRCIGAGE_Bagging_Reg
ression')
```



8.3 Regression —> RandomForest —> IRCIGAGE —> Cigarette age of first use (1-55), 991=never used

```
set.seed(1)
rf_reg = randomForest(IRCIGAGE  ~ ., data = train_reg_clean, mtry = floor(ncol(train_reg)/3), im
portance = TRUE)
rf_reg
```

```
##
## Call:
##  randomForest(formula = IRCIGAGE ~ ., data = train_reg_clean,      mtry = floor(ncol(train_re
g)/3), importance = TRUE)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 20
##
##           Mean of squared residuals: 6.082042
##                     % Var explained: 14.6
```

```
yhat.rf <- predict(rf_reg, newdata = test_reg)

mse_rf <- mean((yhat.rf - test_reg$IRCIGAGE)^2, na.rm = TRUE)
cat("Mean Squared Error (MSE):", round(mse_rf, 4), "\n")
```

```
## Mean Squared Error (MSE): 5.7427
```

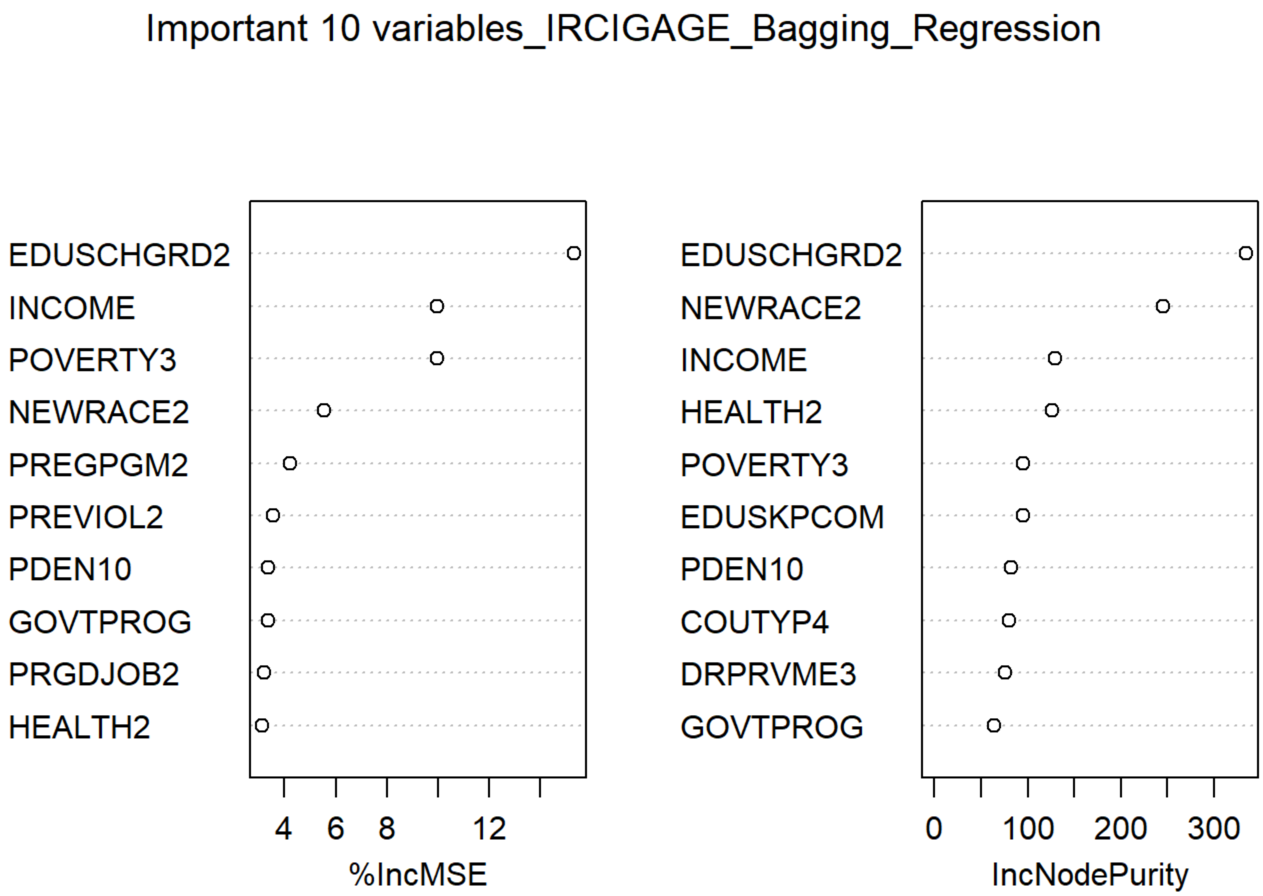
```
rmse_rf <- sqrt(mse_rf)
cat("Root Mean Squared Error (RMSE):", round(rmse_rf, 4), "\n")
```

```
## Root Mean Squared Error (RMSE): 2.3964
```

```
top_10_bag_IRCIGAGE = head(importance(rf_reg),10)
top_10_bag_IRCIGAGE
```

```
##               %IncMSE IncNodePurity
## IRSEX      -0.5157858      34.74347
## NEWRACE2    5.5309712     245.26809
## HEALTH2     3.1387545     126.51167
## EDUSCHLGO   2.8850530      43.67124
## EDUSCHGRD2 15.2946670     333.95377
## EDUSKPCOM   0.4490891      94.80535
## IMOTHER    -1.4440255      25.25111
## IFATHER     2.3699464      47.32725
## INCOME      9.9763670     129.15845
## GOVTPROG    3.3740437      63.53119
```

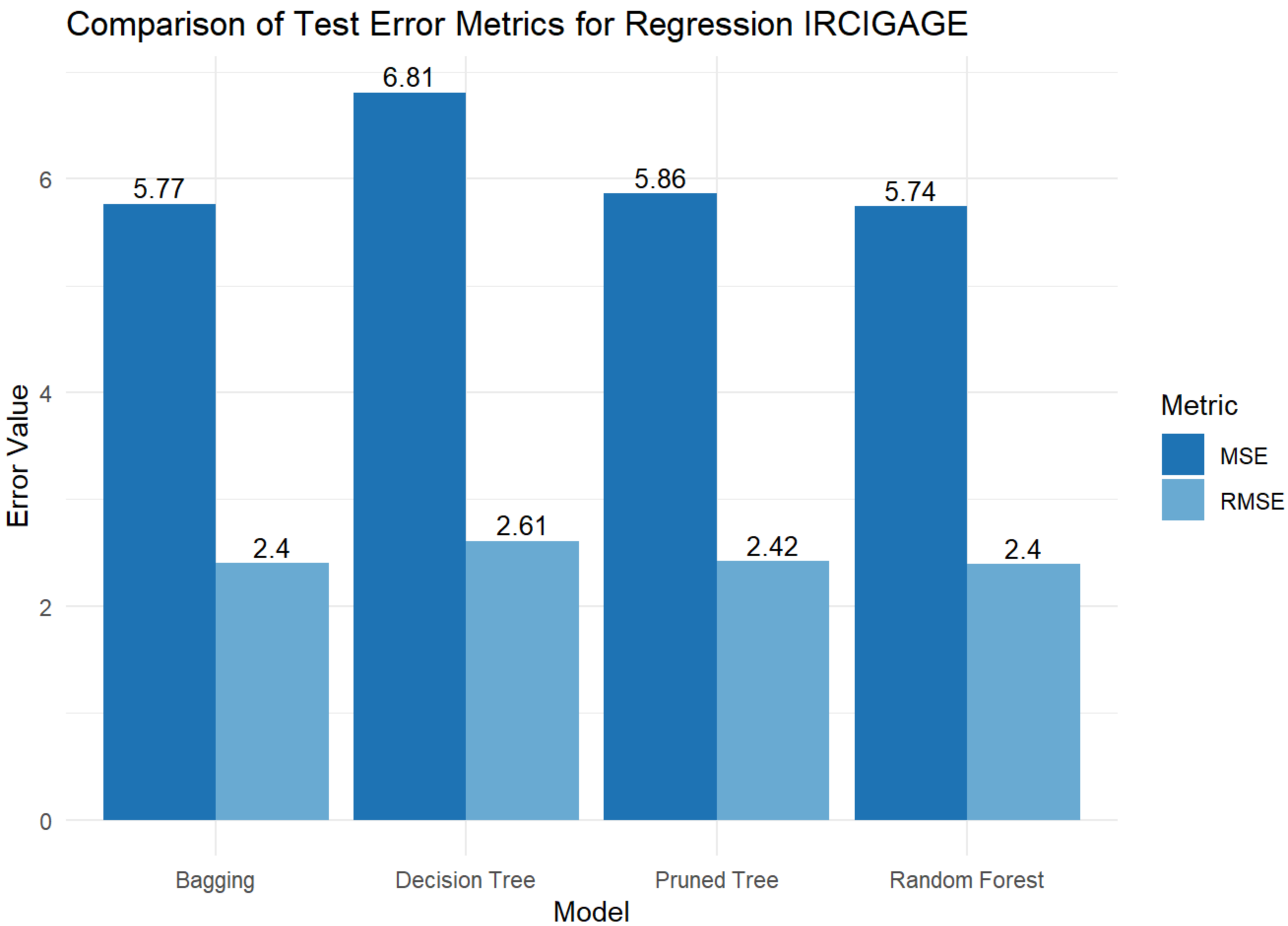
```
varImpPlot(rf_reg, n.var = 10, sort = TRUE, main = 'Important 10 variables_IRCIGAGE_Bagging_Regr
ession')
```



8.4 Compare Regression Methods —> IRCIGAGE —

> Cigarette age of first use (1-55), 991=never used

```
library(ggplot2)

model_names <- c("Decision Tree", "Pruned Tree", "Bagging", "Random Forest")

mse_values <- c(mse_tree, mse_prune, mse_bag, mse_rf)
rmse_values <- c(rmse_tree, rmse_prune, rmse_bag, rmse_rf)

error_df <- data.frame(
  Model = rep(model_names, times = 2),
  Metric = factor(rep(c("MSE", "RMSE"), each = length(model_names)),
                  levels = c("MSE", "RMSE")),
  Value = c(mse_values, rmse_values)
)

ggplot(error_df, aes(x = Model, y = Value, fill = Metric)) +
  geom_bar(stat = "identity", position = position_dodge()) +
  geom_text(aes(label = round(Value, 2)),
            position = position_dodge(width = 0.9),
            vjust = -0.3, size = 3.5) +
  labs(title = "Comparison of Test Error Metrics for Regression IRCIGAGE",
       x = "Model",
       y = "Error Value") +
  scale_fill_manual(values = c("MSE" = "#1f77b4", "RMSE" = "#6baed6")) +  # Custom blues
  theme_minimal()
```


Comparison of Test Error Metrics for Regression IRCIGAGE
Error Value
0
2
4
6
5.77
2.4
6.81
2.61
5.86
2.42
5.74
2.4
Bagging
Decision Tree
Pruned Tree
Random Forest
Model
Metric
MSE
RMSE