fp

Gina Na (3032271139) & Taejin Jeong (23210427)

December 1, 2017

```r
library(ggplot2)  # Visualizing
library(knitr)  # Knitting
library(tree)  # Classification tree
```

```
## Warning: package 'tree' was built under R version 3.3.3
```

```r
library(randomForest)  # Random Forest
```

```
## Warning: package 'randomForest' was built under R version 3.3.3

## randomForest 4.6-12

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

##
## Attaching package: 'randomForest'

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

```r
library(caret)  # Finding importance
```

```
## Warning: package 'caret' was built under R version 3.3.3

## Loading required package: lattice

## Warning: package 'lattice' was built under R version 3.3.3
```

```r
library(rpart)  # Classification tree
```

```
## Warning: package 'rpart' was built under R version 3.3.3
```

```r
library(rpart.plot)  # Visualizing classification tree
```

```
## Warning: package 'rpart.plot' was built under R version 3.3.3
```

```r
library(ROCR)  # ROC curve
```

```
## Warning: package 'ROCR' was built under R version 3.3.3

## Loading required package: gplots

## Warning: package 'gplots' was built under R version 3.3.3

##
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
##
##     lowess
```

Outline of the project

- We are going to use the data, Cenesus Income Data, from UCI Machine Learning Repository, to answer our qustions :
- 1) **what features are decisive to determine whether people have income more than 50K or not?**
- 2) **what are the properties of people who make more than 50K income?** (big question)
- 3) **what are the properties of people who make less than 50K income?**
- This project consists of cleaning data, Analysis, and Conclusion. In the step of cleaning data, we first consider importance of all given variables and then drop unnecessary variables to build our models. With out data cleaned, we analyze using three different methods with each brief description: classification tree, bagged tree and random forest. In this process, we show imporant variables, confusion matrix, ROC and AUC of each mothod. At last, we sum up our reports, answer to our big questoin and discuss further work with new question.

```
train <- read.csv("https://raw.githubusercontent.com/ucb-stat154/stat154-fall-2017/master/problems/proj

test <- read.csv("https://raw.githubusercontent.com/ucb-stat154/stat154-fall-2017/master/problems/proje

names <-  c("age", "workclass", "fnlwgt", "education", "education_num", "marital_status", "occupation",

names(train) <- names
names(test) <- names

train <- na.omit(train)
test <- na.omit(test)

levels(test$income) = levels(train$income)

# To check that all the factors of columns in train data and test data, set check.level vector taking v

check.level <- c()
for (i in 1:ncol(test)){
  if (class(train[,i]) == "factor"){
    if (all.equal(levels(train[,i]), levels(test[,i])) == TRUE){
      check.level[i] = 1
    }else{
      check.level[i] = 0
    }
  }else{
    check.level[i] = 1
  }
}
```

```
## Warning in if (all.equal(levels(train[, i]), levels(test[, i])) == TRUE) {:
## the condition has length > 1 and only the first element will be used
```

```
check.level
```

```
##  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1
```

```
# It says that the level of factor of 14th column in train which is native_country does not match with

#Exclude the one who chose Holand-Netherlands as their native country and run random forest
```

```r
train <- train[-which(grepl(" Holand-Netherlands", train$native_country)),]
train$native_country = droplevels(train$native_country)
```

1) Drop unnecessary level in workclass

```r
train$workclass <- droplevels(train$workclass)
test$workclass <- droplevels(test$workclass)
```

2) Remove unnecessary variables

Education number

`Education` contains the highest level of education and `Education number` contains its numerical index. Those two variables share the same information and therefore, we consider that `Education` is preferable because number of eudcation sometimes does *not* match their final academic degrees.

Final weigh

We do not count `fnlwgt` because population does not affect building our model based on featues.

```r
train = train[,-c(3,5)]
test = test[,-c(3,5)]
```

Importance of variables

```r
raw.tree <- rpart(income~., data=train, method="class")
raw.bagged <- randomForest(income~., data = train, importance = TRUE, mtry = ncol(train) - 1)
raw.rf <- randomForest(income ~., data = train, importance = TRUE)

raw.tree$variable.importance
```

```
##   relationship marital_status   capital_gain      education            sex
##    2277.049587    2241.421945     972.695141     900.057529     746.538645
##     occupation            age hours_per_week native_country   capital_loss
##     672.925560     520.623278     313.917284      24.290709      16.049219
##           race
##       5.638915
```

```r
sort(importance(raw.bagged)[,3], decreasing = TRUE)
```

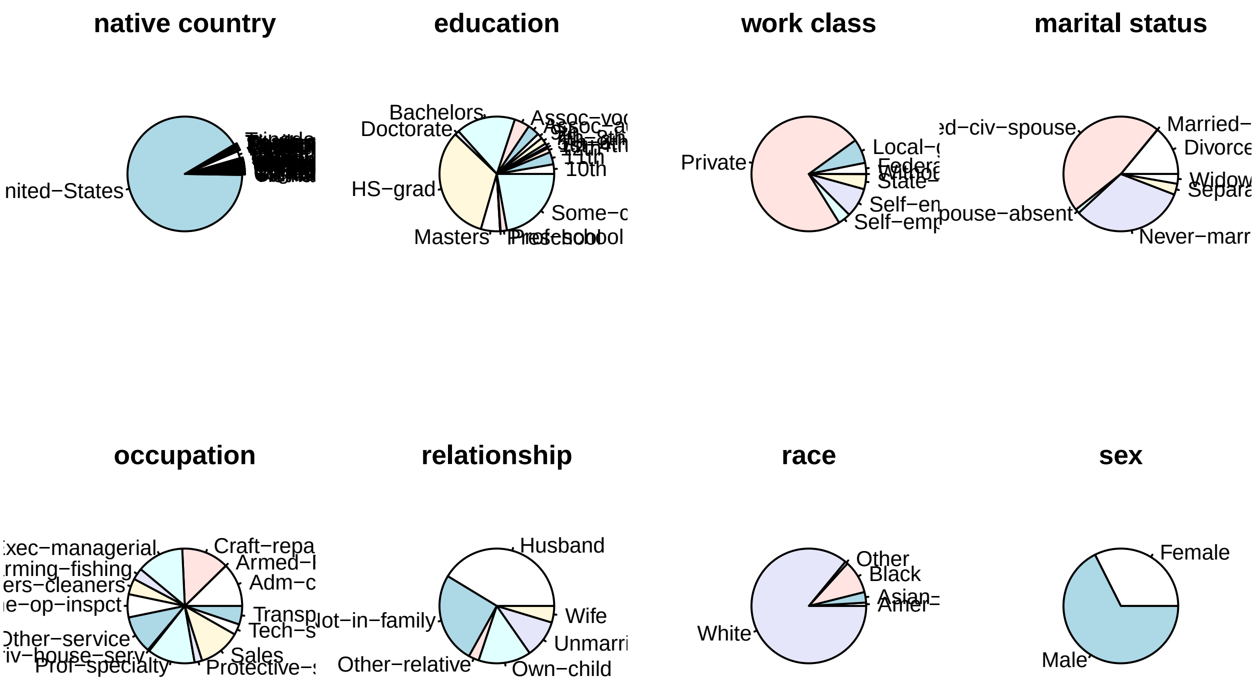
```
##    capital_gain   relationship marital_status   capital_loss            sex
##      192.588703      63.184533      61.768696      55.386432      44.103706
##             age     occupation hours_per_week      workclass      education
##       39.566650      37.482786      34.216238      32.794738      26.039222
##            race native_country
##        8.852819     -20.260443
```

```r
sort(importance(raw.rf)[,3], decreasing = TRUE)
```

```
##    capital_gain   capital_loss marital_status            age      education
##      117.931745      48.066434      40.991966      32.765316      31.299243
```

```
##      occupation   relationship hours_per_week      workclass            sex
##       30.355840      29.268049      28.444568      28.160485      24.455525
##            race native_country
##        7.056928     -22.092271
```

Categorical Variables



Excluded variables : `native country` and `race`. The pie chart of `native coury` and `race` indicates that the majority of observations are from United States and from white respectively; these two variables are not well-separated information. Moreover, based on reported importances of variables above, we observed that both `native country` and `race`are not significant enough to fit models. Therefore, we exclude the variables native country and race.

```
train = train[,-c(7,12)]
test = test[,-c(7,12)]
```

Capital gain & Capital loss

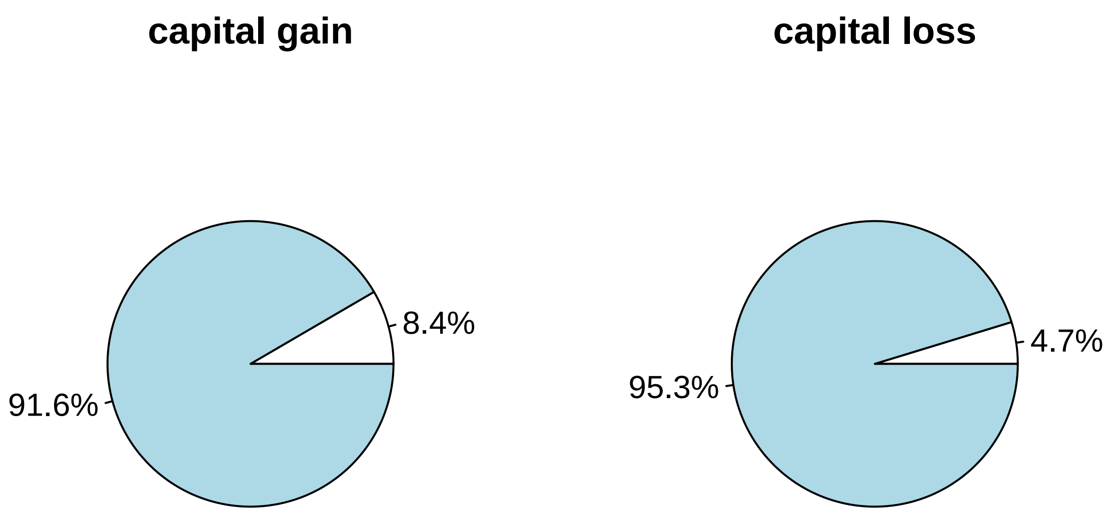
```
cap_gain <- ifelse(train$capital_gain == 0, "No gain", "gain")
cap_loss <- ifelse(train$capital_loss == 0, "No loss", "loss")

t1 <- table(cap_gain)
gain_pct <- paste0(round(100 * c(t1[1]/sum(t1), t1[2]/sum(t1)), 1) , "%") # percentage of gain & no gai

t2 <- table(cap_loss)
```

```r
loss_pct <- paste0(round(100 * c(t2[1]/sum(t2), t2[2]/sum(t2)), 1), "%") # percentage of loss & no loss


par(mfrow=c(1,2))
pie(table(cap_gain), labels=gain_pct, main="capital gain")
pie(table(cap_loss), labels=loss_pct, main="capital loss")
```



we observed that `capital gain` and `capital loss` variables are also lopsided; only 4.7% of whole have capital loss and and only 8.4% have capital gain. According to the summary. However, we will *not* exclude these values because those were reported as important variables from our first step.

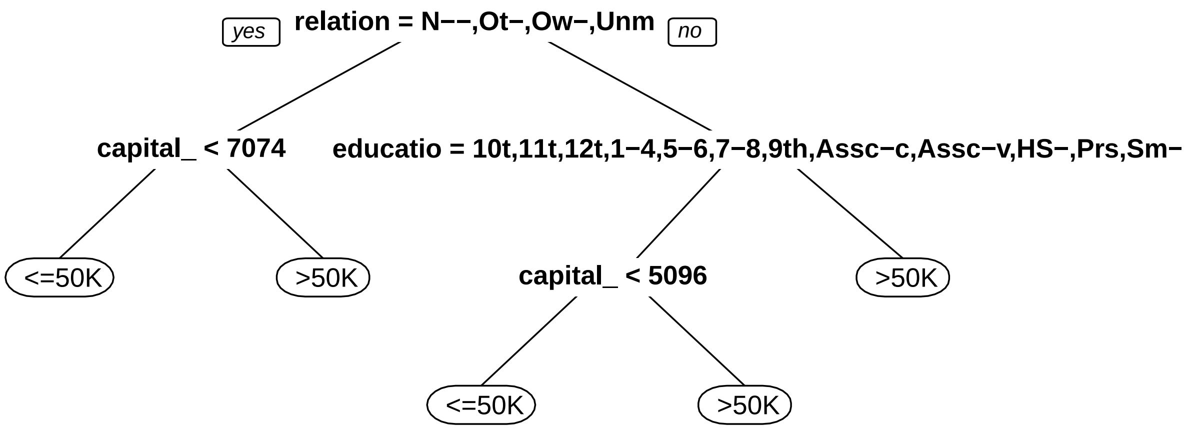
Even though they are reported by few people, We may assume that `capital gain` and `capital loss` have huge influence to whether individuals earn over 50K or not.

Classification tree

description : Classification Tree is a method to maintain the test cases in an efficient way in the aspect of cost and to visualize the testing scope more concrete by determining input domain and relevant factors affecting testing and partitioning input data into classes

```r
tree <- rpart(income~., data=train, method="class")#method="class" arg tells us to make classification

prp(tree)
```




```
# prune tree
## We prune back the tree to avoid the overfitting the data by assigning the complexity parameter with t

best_cp <- tree$cptable[which.min(tree$cptable[,"xerror"]),"CP"] #best_cp = 0.1

tree <- prune(tree, cp= best_cp)
```

1) most important variables

```
head(tree$variable.importance , 6)
```

```
##   relationship marital_status   capital_gain      education            sex
##      2277.0496      2241.4219       972.6951       900.0575       746.5386
##     occupation
##       672.9256
```

2) confusion matrix

```
tree.pred <- predict(tree, newdata=test, type="class")
tree.conf <- table(test$income, tree.pred)
tree.conf
```

```
##         tree.pred
##           <=50K  >50K
```

```
##    <=50K  10771   588
##    >50K    1837  1863
```

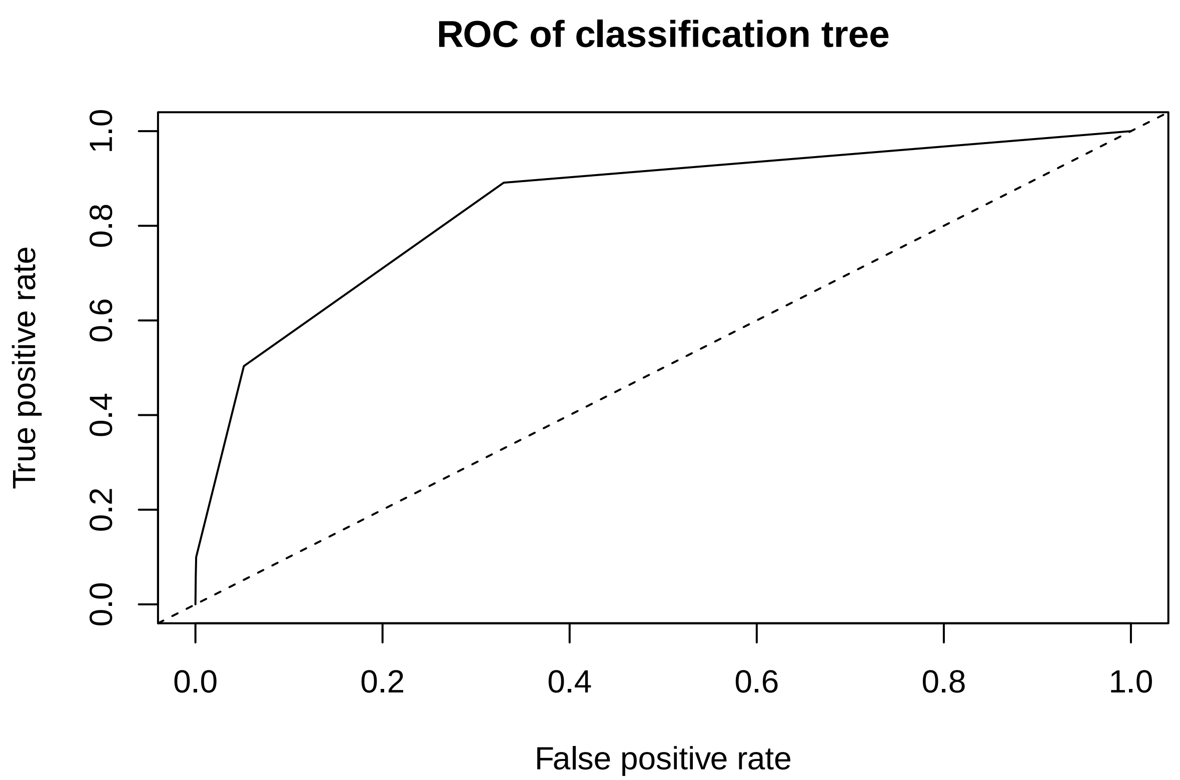
3) accurate & error rate

```r
tree.accurate <- (tree.conf[1,1] + tree.conf[2,2]) / sum(tree.conf)
tree.error <- 1 - tree.accurate
data.frame("accurate rate" = tree.accurate, "error rate"=tree.error)
```

```
##   accurate.rate error.rate
## 1     0.8389667  0.1610333
```

4)ROC curve and AUC

```r
tree.pred1 <- predict(tree, newdata=test)
tree.pred2 <- prediction(tree.pred1[,2], test$income)
tree.perf <- performance(tree.pred2, "tpr", "fpr")
plot(tree.perf, main="ROC of classification tree") + abline(0,1,lty=2)
```




```
## numeric(0)
```

```r
tree.auc <- performance(tree.pred2, measure="auc")@y.values[[1]]
tree.auc
```

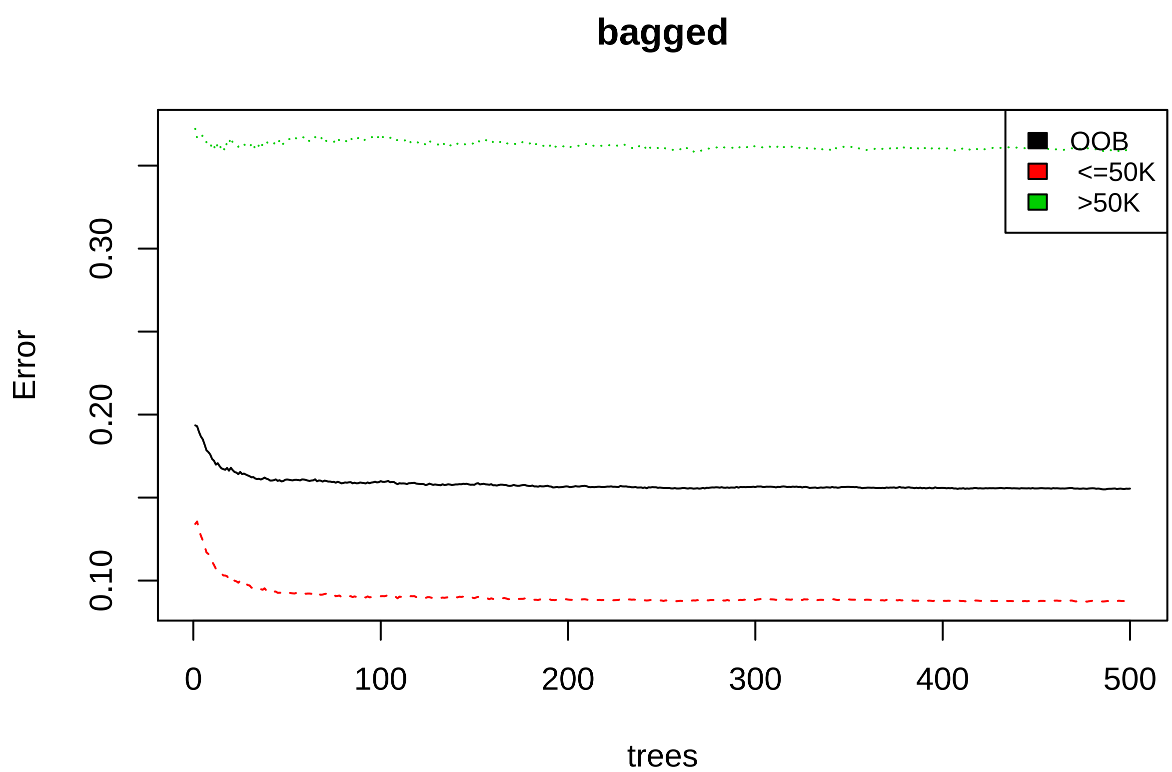
```
## [1] 0.8430644
```

Bagged Tree

description : Baged tree combines classifiers trained on bootstrap samples of the original data. It improves the classifications by reducing variance and avoiding overfitting.

1) most important variables

```
bagged <- randomForest(income~., data = train,
                       importance = TRUE, mtry = ncol(train) - 1)
plot(bagged)
legend("topright", colnames(bagged$err.rate),col=1:3,cex=0.8,fill=1:3)
```




```
## As shown in the graph, errors are almost constant after ntree=100, so we chose 100 for ntree and mak

bagged <- randomForest(income~., data = train, importance = TRUE, mtry = ncol(train) - 1, ntree = 100)
varImpPlot(bagged)
```

we chose`capital_gain`, `relationship`, `capital_loss`, `occupation` and `education` as our 5 most important variables.

2) confusion matrix

```
bagged.pred <- predict(bagged, newdata=test, type="class")
bagged.conf <- table(test$income, bagged.pred)
bagged.conf
```

```
##         bagged.pred
##            <=50K  >50K
##     <=50K  10311  1048
##     >50K    1359  2341
```

3) accurate & error rate

```
bagged.accurate <- (bagged.conf[1,1] + bagged.conf[2,2]) / sum(bagged.conf)
bagged.error <- 1 - bagged.accurate

data.frame("accurate rate" = bagged.accurate, "error rate"=bagged.error)
```

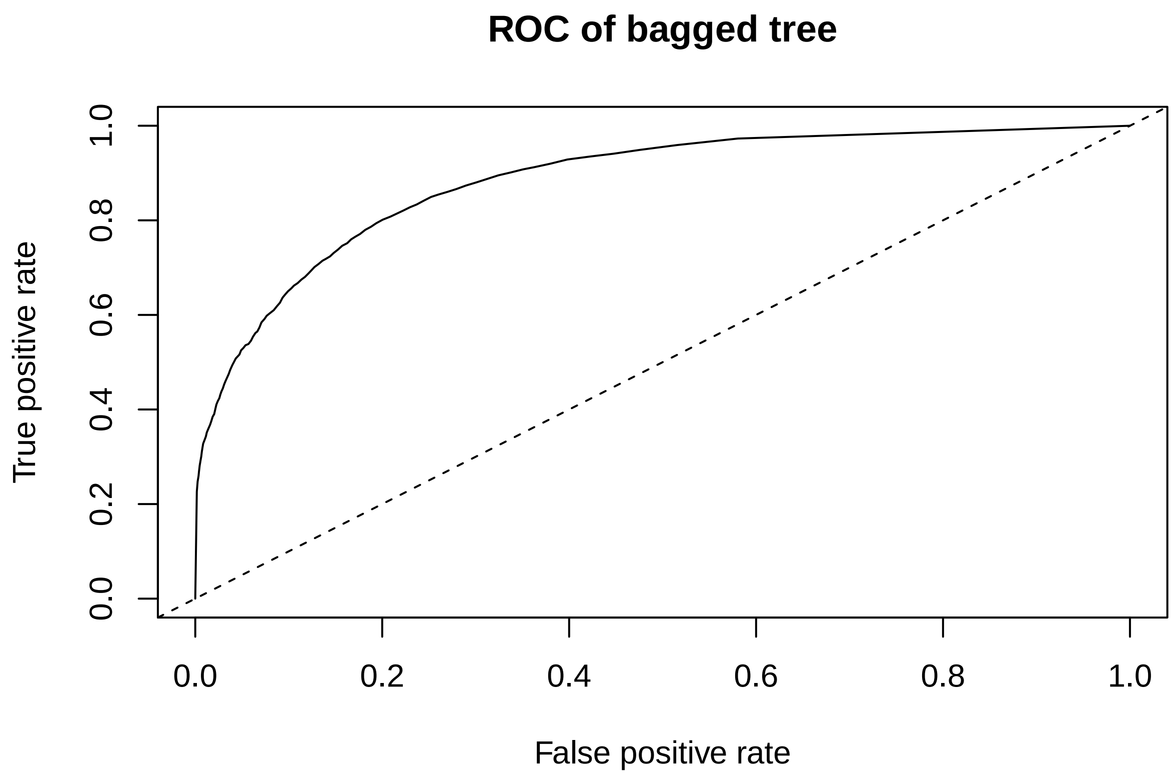
```
##   accurate.rate error.rate
## 1      0.840162   0.159838
```

4) ROC curve and AUC

```
bagged.pred1 <- predict(bagged, newdata=test, type="prob")
bagged.pred2 <- prediction(bagged.pred1[,2], test$income)

bagged.perf <- performance(bagged.pred2, "tpr", "fpr")

plot(bagged.perf, main="ROC of bagged tree") + abline(0,1,lty=2)
```




```
## numeric(0)
```

```
bagged.auc <- performance(bagged.pred2, measure="auc")@y.values[[1]]
bagged.auc
```

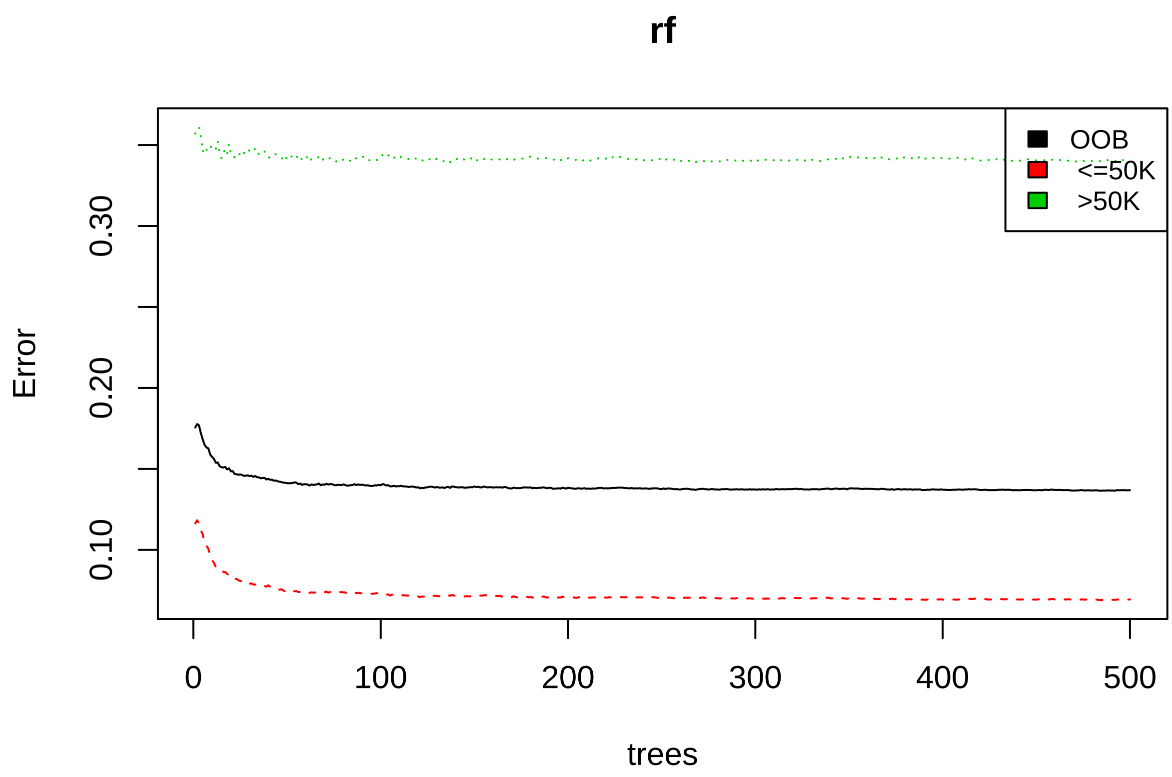
```
## [1] 0.8835624
```

Random Forest

description : Random Forest is a method to predict a target data by setting rules and splitting nodes to predict whether the target data looks like a model trained. It improves predictive accuracy by generating a large number of bootstrapped trees. Final predicted outcome is attained by combining the results across all of the trees

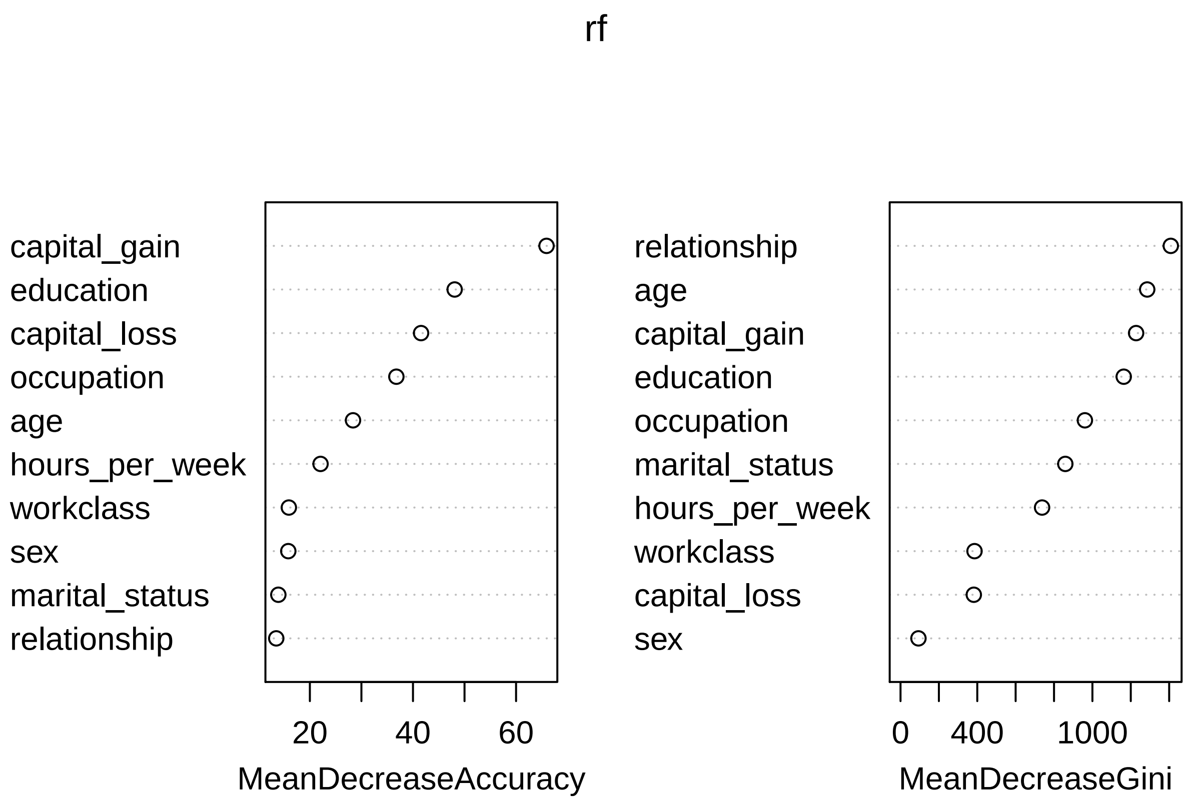
1) 5 most important variables

```
rf <- randomForest(income ~., data = train, importance = TRUE)
plot(rf)
legend("topright", colnames(rf$err.rate),col=1:3,cex=0.8,fill=1:3)
```




```
## As shown in the graph, errors are almost constant after ntree=50, so we chose 50 for ntree and make a

rf <- randomForest(income ~., data = train, importance = TRUE, ntree = 50)
varImpPlot(rf)
```



we chose `capital_gain`, `education`, `capital_loss`, `occupation` and `age` as our 5 most important variables.

2) confusion matrix

```
rf.pred <- predict(rf, newdata = test, type = "class")
rf.conf <- table(test$income, rf.pred)
rf.conf
```

```
##          rf.pred
##            <=50K  >50K
##     <=50K  10521   838
##     >50K    1321  2379
```

3) accurate & error rate

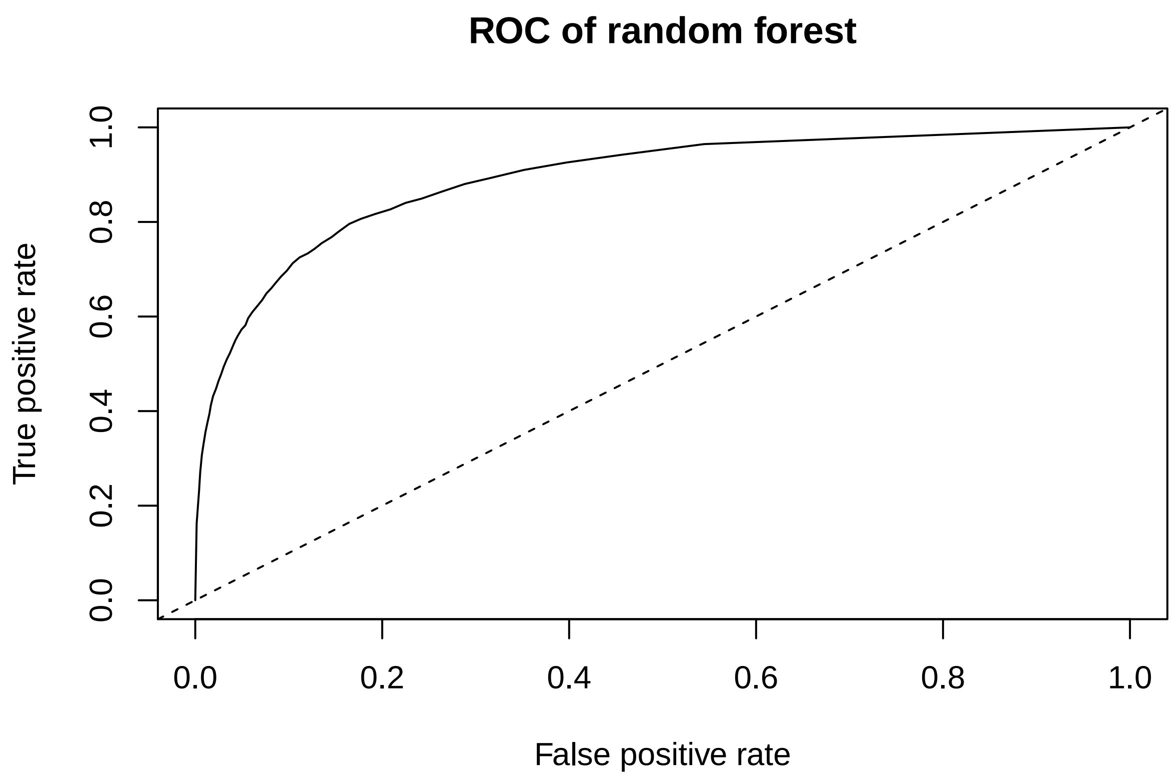
```
rf.accurate <- (rf.conf[1,1] + rf.conf[2,2]) / sum(rf.conf)
rf.error <- 1 - rf.accurate
data.frame("accurate rate" = rf.accurate, "error rate"=rf.error)
```

```
##   accurate.rate error.rate
## 1     0.8566306  0.1433694
```

4) ROC curve and AUC

```
rf.pred1 <- predict(rf, test, type="prob")
rf.pred2 <- prediction(rf.pred1[,2], test$income)
rf.perf <- performance(rf.pred2, "tpr", "fpr")

plot(rf.perf, main="ROC of random forest") + abline(0,1, lty=2)
```




```
## numeric(0)
```

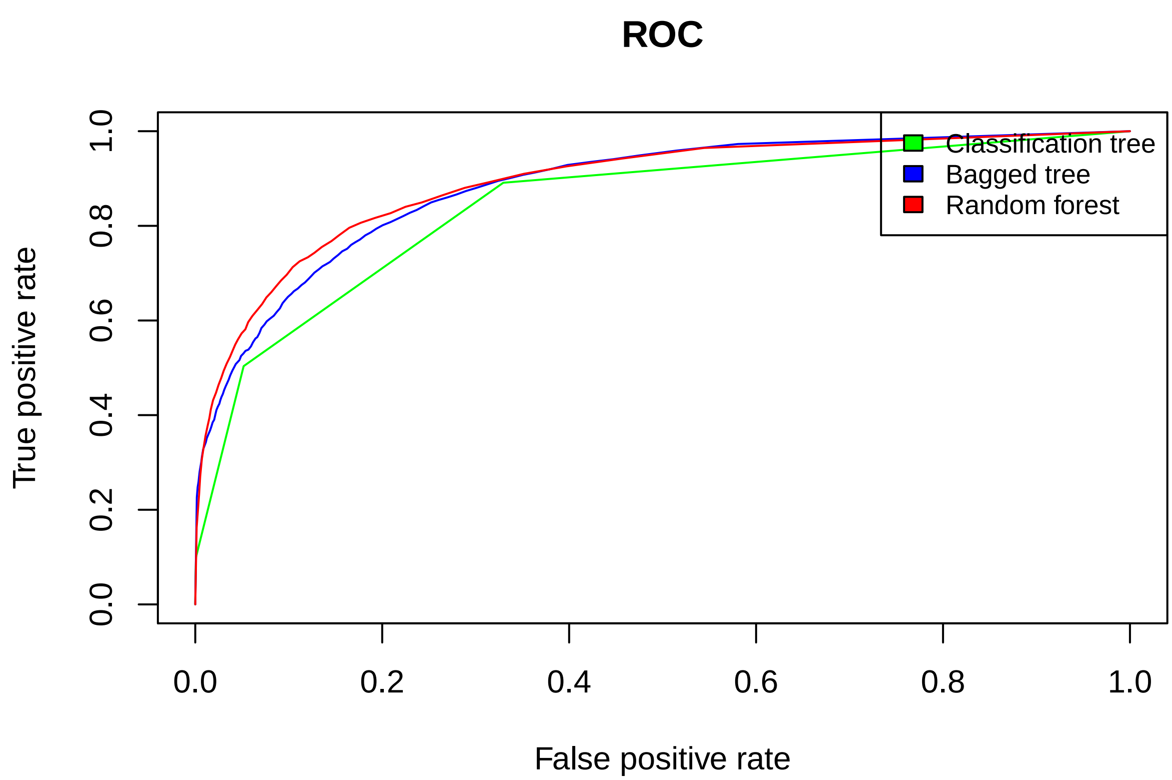
```
rf.auc <- performance(rf.pred2, measure="auc")@y.values[[1]]
rf.auc
```

```
## [1] 0.8913381
```

Model Selection

1) ROC and AUC

```
plot(tree.perf, col="green")
plot(bagged.perf, add=T, col="blue")
plot(rf.perf, add=T, col="red")
legend("topright", legend=c("Classification tree","Bagged tree","Random forest"), fill=c("green","blue"
title("ROC")
```




```
data.frame("Classification Tree" = tree.auc, "Bagged Tree" = bagged.auc, "Random Forest" = rf.auc, row.n
```

```
##     Classification.Tree Bagged.Tree Random.Forest
## AUC           0.8430644   0.8835624     0.8913381
```

Since the AUC of 'Random Forest' is the highest which implies Random Forest is the best model on the data, we chose 'Random Forest' to predict the test data.

2) Confusion matrix and TPR/TNR

```
rf.conf2 <- rf.conf[2:1, 2:1]
rf.conf2
```

```
##         rf.pred
##           >50K  <=50K
```

```
##    >50K   2379   1321
##    <=50K   838  10521
```

```
TPR <- rf.conf2[1,1] / ( rf.conf2[1,1] + rf.conf2[1,2])
TNR <- rf.conf2[2,2] / ( rf.conf2[2,1] + rf.conf2[2,2])
data.frame(TPR,TNR)
```

```
##        TPR       TNR
## 1 0.642973 0.9262259
```

3) Find features of >50k

```
new_data <- train[,c(1,3,5,8,9,11)]
new_data$capital_gain <- ifelse(new_data$capital_gain >0, "Gain", "None")
levels(new_data$capital_gain) <- c("Gain", "None")
new_data$capital_loss <- ifelse(new_data$capital_loss >0, "Loss", "None")
levels(new_data$capital_loss) <- c("Loss", "None")

new_data$income <- ifelse(new_data$income == " >50K", "Yes", "No")
levels(new_data$income) <- c("Yes","No")

new_data_split <- split(new_data, new_data$income)

new_data_split[[2]][,2] <- droplevels(new_data_split[[2]][,2])
new_data_split[[2]][,3] <- droplevels(new_data_split[[2]][,3])
levels(new_data_split[[2]][,4]) <- c("Gain","None")
levels(new_data_split[[2]][,5]) <- c("Loss","None")

n <- c()
Character <- c()
c <- c()

for (i in 2:5){
  n_split <- split(new_data_split[[2]][,c(i,6)], new_data_split[[2]][,i])
  for (h in 1:length(levels(new_data_split[[2]][,i]))){
    n[h] <- nrow(n_split[[h]])
    c <- order(n, decreasing = TRUE)[1]
  }
  Character[i] <- levels(new_data_split[[2]][,i])[c]
}

Character[1] <- round(mean(new_data_split[[2]][,1]))
Character
```

```
## [1] "44"                " Bachelors"        " Exec-managerial"
## [4] "None"              "None"
```

4) Find feature of <= 50K

```
new_data <- train[,c(1,3,5,8,9,11)]
new_data$capital_gain <- ifelse(new_data$capital_gain >0, "Gain", "None")
levels(new_data$capital_gain) <- c("Gain", "None")
```

```r
new_data$capital_loss <- ifelse(new_data$capital_loss >0, "Loss", "None")
levels(new_data$capital_loss) <- c("Loss", "None")

new_data$income <- ifelse(new_data$income == " >50K", "Yes", "No")
levels(new_data$income) <- c("Yes","No")

new_data_split <- split(new_data, new_data$income)

for (i in 1:2){
  new_data_split[[i]][,2] <- droplevels(new_data_split[[i]][,2])
  new_data_split[[i]][,2] <- droplevels(new_data_split[[i]][,2])
  new_data_split[[i]][,3] <- droplevels(new_data_split[[i]][,3])
  levels(new_data_split[[i]][,4]) <- c("Gain","None")
  levels(new_data_split[[i]][,5]) <- c("Loss","None")
}




n <- c()
Character_Y <- c()
c <- c()

for (i in 2:5){
  n_split <- split(new_data_split[[2]][,c(i,6)], new_data_split[[2]][,i])
  for (h in 1:length(levels(new_data_split[[2]][,i]))){
    n[h] <- nrow(n_split[[h]])
    c <- order(n, decreasing = TRUE)[1]
  }
  n <- c()
  Character_Y[i] <- levels(new_data_split[[2]][,i])[c]
}

Character_Y[1] <- round(mean(new_data_split[[2]][,1]))
Character_Y
```

```
## [1] "44"                " Bachelors"        " Exec-managerial"
## [4] "None"              "None"
```

```r
n <- c()
Character_N <- c()
c <- c()

for (i in 2:5){
  n_split <- split(new_data_split[[1]][,c(i,6)], new_data_split[[1]][,i])
  for (h in 1:length(levels(new_data_split[[1]][,i]))){
    n[h] <- nrow(n_split[[h]])
    c <- order(n, decreasing = TRUE)[1]
  }
  n <- c()
  Character_N[i] <- levels(new_data_split[[1]][,i])[c]
}
```

```
Character_N[1] <- round(mean(new_data_split[[1]][,1]))
Character_N
```

```
## [1] "37"            " HS-grad"      " Adm-clerical" "None"
## [5] "None"
```

Conclusion

answer the questions

1) **what features are decisive to determine whether people have income more than 50K or not?**: capital gain, education, capital_loss, occupation and age
2) **what are the properties of people who make more than 50K income?** (big question) : people who earn more than 50K are most likely in age of mid-40s and work in Exec-managerial area. They tend to have an education level of Bachelors, but have not reported capital loss or gain.
3) **what are the properties of people who make less than 50K income?** : peope who earn less than 50K are most likely in age of late-30s and have an eduacation level of High school graduation. They prone to have jobs in Adm-clerical field with no capital loss or gain.