Decision Tree vs Random Forest in R

```
# Disable warning messages globally
options(warn = - 1)
```

Library for data wrangling

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

Library for timing

```
library(tictoc)
```

Library for decision tree

```
library(rpart)
library(rpart.plot)
```

Library for random forest

```
library(randomForest)
```

```
## randomForest 4.6-14
```

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

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

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

Library for plotting

```
library(ggplot2)
```

```
##
## Attaching package: 'ggplot2'
```

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

Importing the data

```
data = read.csv('C:\\Users\\Dell\\OneDrive\\College_2nd\\R Lab\\adult.csv')
```

Looking at the data

```
head(data)
```

```
##   age workclass fnlwgt    education education.num marital.status
## 1  90         ?  77053      HS-grad             9        Widowed
## 2  82   Private 132870      HS-grad             9        Widowed
## 3  66         ? 186061 Some-college            10        Widowed
## 4  54   Private 140359      7th-8th             4       Divorced
## 5  41   Private 264663 Some-college            10      Separated
## 6  34   Private 216864      HS-grad             9       Divorced
##          occupation  relationship  race    sex capital.gain capital.loss
## 1                 ? Not-in-family White Female            0         4356
## 2   Exec-managerial Not-in-family White Female            0         4356
## 3                 ?     Unmarried Black Female            0         4356
## 4 Machine-op-inspct     Unmarried White Female            0         3900
## 5    Prof-specialty     Own-child White Female            0         3900
## 6     Other-service     Unmarried White Female            0         3770
##   hours.per.week native.country income
## 1             40  United-States  <=50K
## 2             18  United-States  <=50K
## 3             40  United-States  <=50K
## 4             40  United-States  <=50K
## 5             40  United-States  <=50K
## 6             45  United-States  <=50K
```

Statistical summary

```
summary(data)
```

```
##       age         workclass            fnlwgt         education
##  Min.   :17.00   Length:32561       Min.   :  12285   Length:32561
##  1st Qu.:28.00   Class :character   1st Qu.: 117827   Class :character
##  Median :37.00   Mode  :character   Median : 178356   Mode  :character
##  Mean   :38.58                      Mean   : 189778
##  3rd Qu.:48.00                      3rd Qu.: 237051
##  Max.   :90.00                      Max.   :1484705
##  education.num   marital.status      occupation        relationship
##  Min.   : 1.00   Length:32561       Length:32561       Length:32561
##  1st Qu.: 9.00   Class :character   Class :character   Class :character
##  Median :10.00   Mode  :character   Mode  :character   Mode  :character
##  Mean   :10.08
##  3rd Qu.:12.00
##  Max.   :16.00
##      race               sex             capital.gain    capital.loss
##  Length:32561       Length:32561       Min.   :    0   Min.   :   0.0
##  Class :character   Class :character   1st Qu.:    0   1st Qu.:   0.0
##  Mode  :character   Mode  :character   Median :    0   Median :   0.0
##                                        Mean   : 1078   Mean   :  87.3
##                                        3rd Qu.:    0   3rd Qu.:   0.0
##                                        Max.   :99999   Max.   :4356.0
##  hours.per.week  native.country        income
##  Min.   : 1.00   Length:32561       Length:32561
##  1st Qu.:40.00   Class :character   Class :character
##  Median :40.00   Mode  :character   Mode  :character
##  Mean   :40.44
##  3rd Qu.:45.00
##  Max.   :99.00
```

Structure of the data

```
str(data)
```

```
## 'data.frame':    32561 obs. of  15 variables:
##  $ age           : int  90 82 66 54 41 34 38 74 68 41 ...
##  $ workclass     : chr  "?" "Private" "?" "Private" ...
##  $ fnlwgt        : int  77053 132870 186061 140359 264663 216864 150601 88638 422013 70037
...
##  $ education     : chr  "HS-grad" "HS-grad" "Some-college" "7th-8th" ...
##  $ education.num : int  9 9 10 4 10 9 6 16 9 10 ...
##  $ marital.status: chr  "Widowed" "Widowed" "Widowed" "Divorced" ...
##  $ occupation    : chr  "?" "Exec-managerial" "?" "Machine-op-inspct" ...
##  $ relationship  : chr  "Not-in-family" "Not-in-family" "Unmarried" "Unmarried" ...
##  $ race          : chr  "White" "White" "Black" "White" ...
##  $ sex           : chr  "Female" "Female" "Female" "Female" ...
##  $ capital.gain  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ capital.loss  : int  4356 4356 4356 3900 3900 3770 3770 3683 3683 3004 ...
##  $ hours.per.week: int  40 18 40 40 40 45 40 20 40 60 ...
##  $ native.country: chr  "United-States" "United-States" "United-States" "United-States"
...
##  $ income        : chr  "<=50K" "<=50K" "<=50K" "<=50K" ...
```

Checking for Nulls

```
any(is.na(data))
```

```
## [1] FALSE
```

Converting to factors:

```
data$workclass <- factor(data$workclass, exclude = c("", NA))

data$education <- factor(data$education, exclude = c("", NA))

data$education.num <- factor(data$education.num, exclude = c("", NA))

data$marital.status <- factor(data$marital.status, exclude = c("", NA))

data$occupation <- factor(data$occupation, exclude = c("", NA))

data$relationship <- factor(data$relationship, exclude = c("", NA))

data$race <- factor(data$race, exclude = c("", NA))

data$sex <- factor(data$sex, exclude = c("", NA))

data$income <- factor(data$income, exclude = c("", NA))

data$native.country <- factor(data$native.country, exclude = c("", NA))
```

Looking at structure again

```
str(data)
```

```
## 'data.frame':    32561 obs. of  15 variables:
##  $ age           : int  90 82 66 54 41 34 38 74 68 41 ...
##  $ workclass     : Factor w/ 9 levels "?","Federal-gov",..: 1 5 1 5 5 5 5 8 2 5 ...
##  $ fnlwgt        : int  77053 132870 186061 140359 264663 216864 150601 88638 422013 70037
...
##  $ education     : Factor w/ 16 levels "10th","11th",..: 12 12 16 6 16 12 1 11 12 16 ...
##  $ education.num : Factor w/ 16 levels "1","2","3","4",..: 9 9 10 4 10 9 6 16 9 10 ...
##  $ marital.status: Factor w/ 7 levels "Divorced","Married-AF-spouse",..: 7 7 7 1 6 1 6 5 1
5 ...
##  $ occupation    : Factor w/ 15 levels "?","Adm-clerical",..: 1 5 1 8 11 9 2 11 11 4 ...
##  $ relationship  : Factor w/ 6 levels "Husband","Not-in-family",..: 2 2 5 5 4 5 5 3 2 5
...
##  $ race          : Factor w/ 5 levels "Amer-Indian-Eskimo",..: 5 5 3 5 5 5 5 5 5 5 ...
##  $ sex           : Factor w/ 2 levels "Female","Male": 1 1 1 1 1 1 2 1 1 2 ...
##  $ capital.gain  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ capital.loss  : int  4356 4356 4356 3900 3900 3770 3770 3683 3683 3004 ...
##  $ hours.per.week: int  40 18 40 40 40 45 40 20 40 60 ...
##  $ native.country: Factor w/ 42 levels "?","Cambodia",..: 40 40 40 40 40 40 40 40 40 1 ...
##  $ income        : Factor w/ 2 levels "<=50K",">50K": 1 1 1 1 1 1 1 2 1 2 ...
```

Dropping unwanted columns

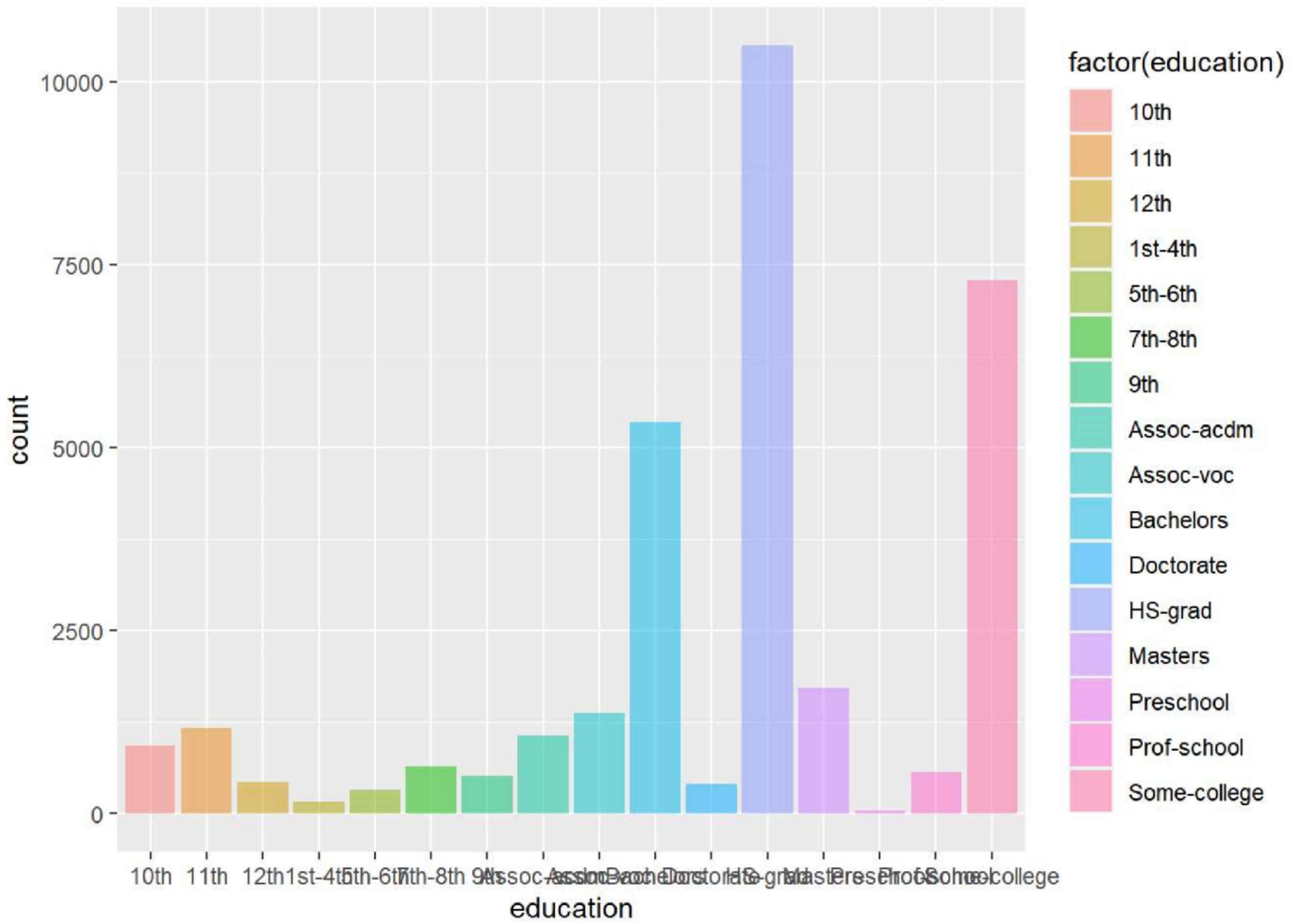
```
# Drop variables
df <- data %>% select(-c(fnlwgt, capital.gain))
```

```
glimpse(df)
```

```
## Rows: 32,561
## Columns: 13
## $ age            <int> 90, 82, 66, 54, 41, 34, 38, 74, 68, 41, 45, 38, 52, 32,~
## $ workclass      <fct> ?, Private, ?, Private, Private, Private, Private, Stat~
## $ education      <fct> HS-grad, HS-grad, Some-college, 7th-8th, Some-college, ~
## $ education.num  <fct> 9, 9, 10, 4, 10, 9, 6, 16, 9, 10, 16, 15, 13, 14, 16, 1~
## $ marital.status <fct> Widowed, Widowed, Widowed, Divorced, Separated, Divorce~
## $ occupation     <fct> ?, Exec-managerial, ?, Machine-op-inspct, Prof-specialt~
## $ relationship   <fct> Not-in-family, Not-in-family, Unmarried, Unmarried, Own~
## $ race           <fct> White, White, Black, White, White, White, White, White,~
## $ sex            <fct> Female, Female, Female, Female, Female, Female, Male, F~
## $ capital.loss   <int> 4356, 4356, 4356, 3900, 3900, 3770, 3770, 3683, 3683, 3~
## $ hours.per.week <int> 40, 18, 40, 40, 40, 45, 40, 20, 40, 60, 35, 45, 20, 55,~
## $ native.country <fct> United-States, United-States, United-States, United-Sta~
## $ income         <fct> <=50K, <=50K, <=50K, <=50K, <=50K, <=50K, <=50K, >50K, ~
```

Plotting Education

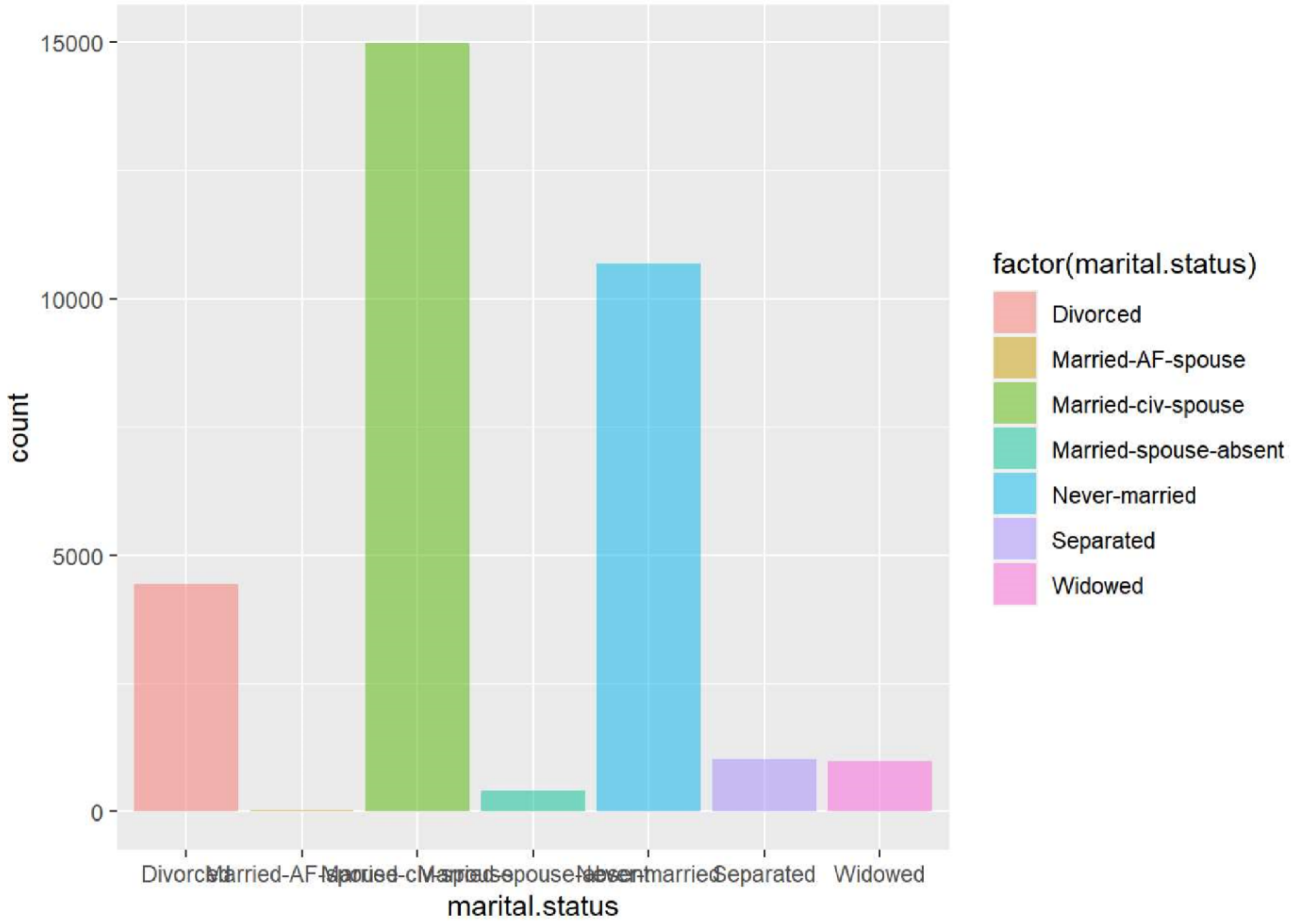
```
ggplot(df,aes(education))+
        geom_bar(aes(fill=factor(education)),alpha=0.5)
```



We can see that most of them are high school graduates

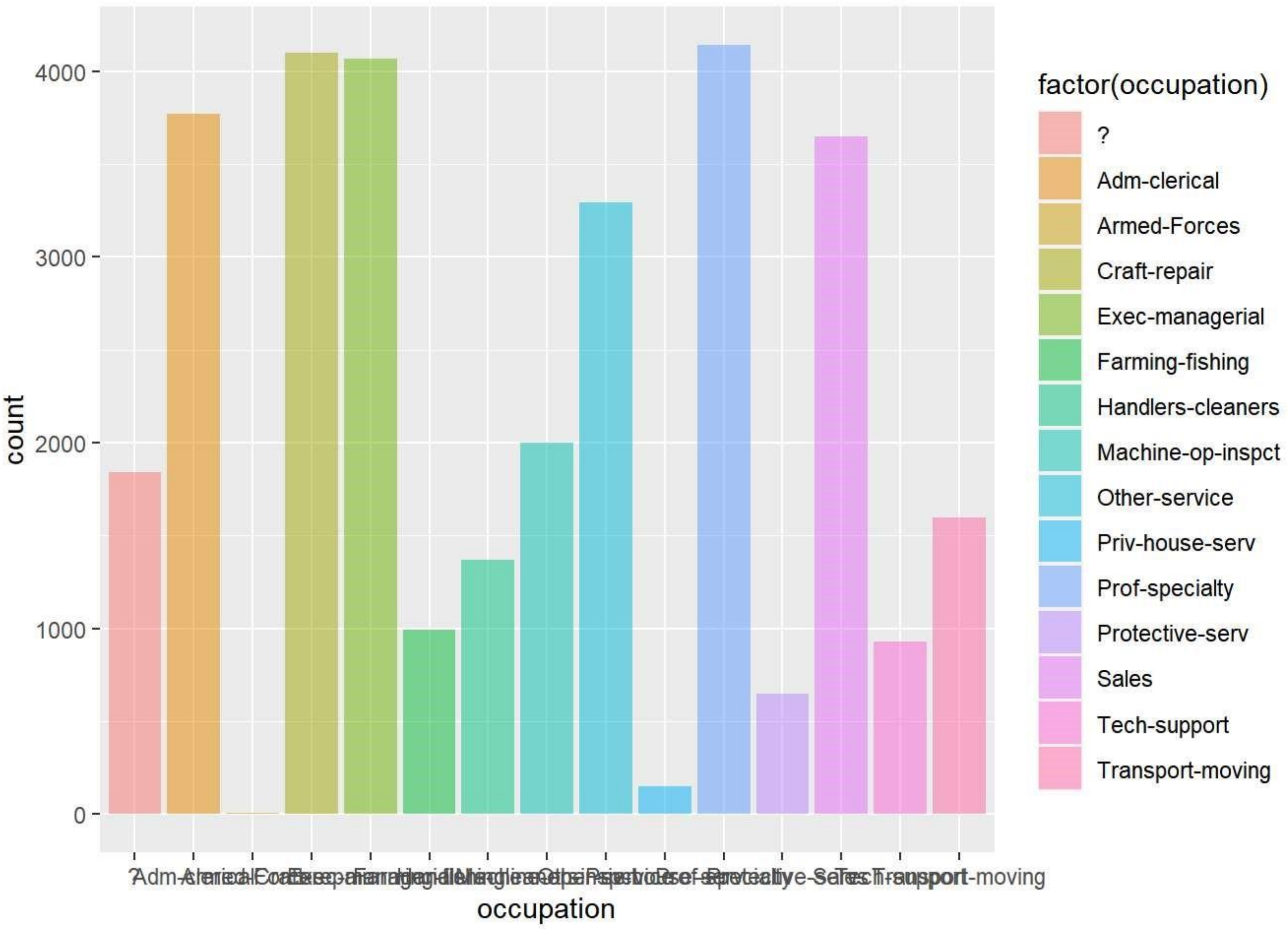
Plotting Marital status

```
ggplot(df,aes(marital.status))+
        geom_bar(aes(fill=factor(marital.status)),alpha=0.5)
```



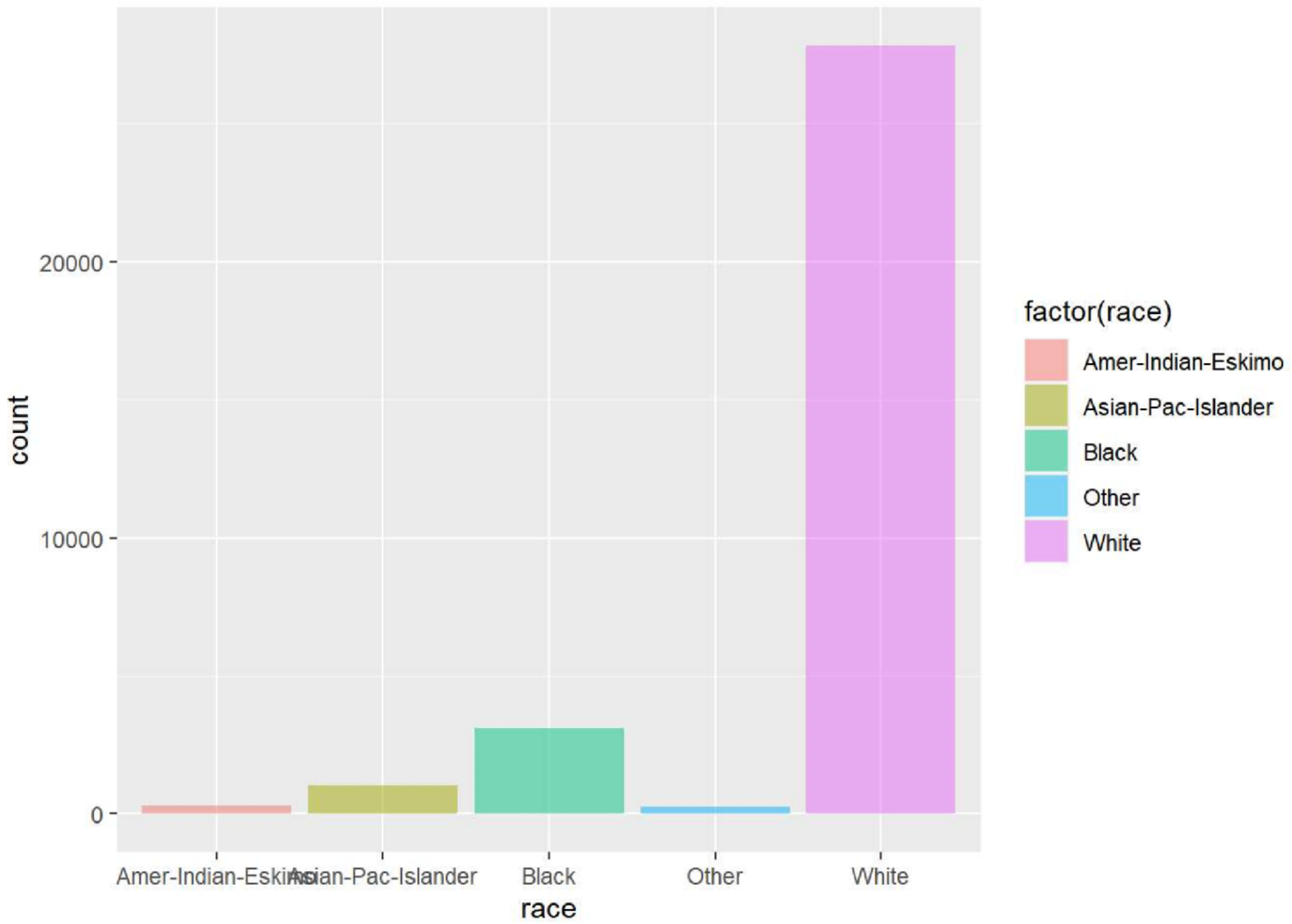
Plotting Occupation

```
ggplot(df,aes(occupation))+
        geom_bar(aes(fill=factor(occupation)),alpha=0.5)
```



Plotting Race

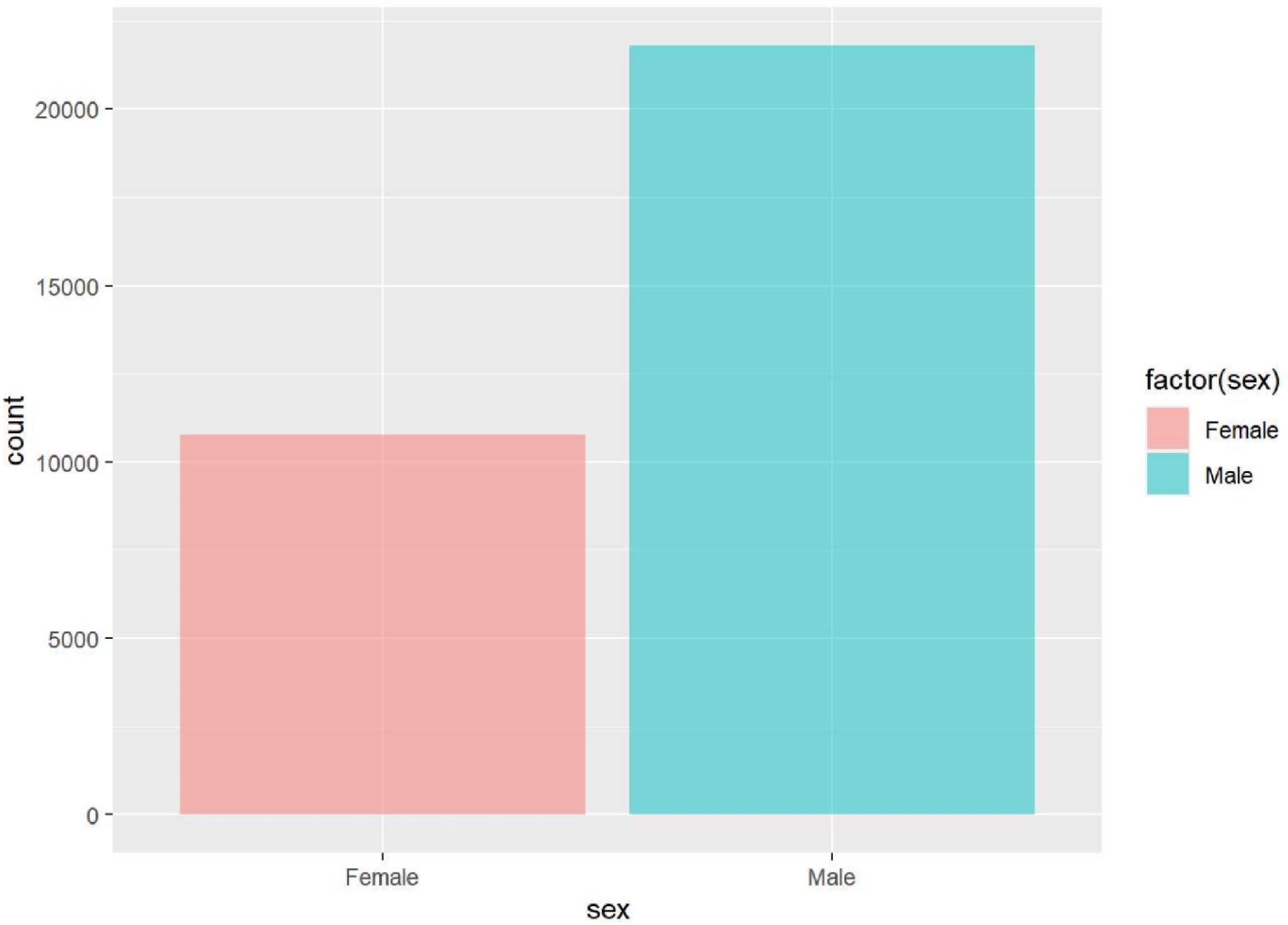
```
ggplot(df,aes(race))+
        geom_bar(aes(fill=factor(race)),alpha=0.5)
```



The data is clearly unbalanced and biased

Plotting sex

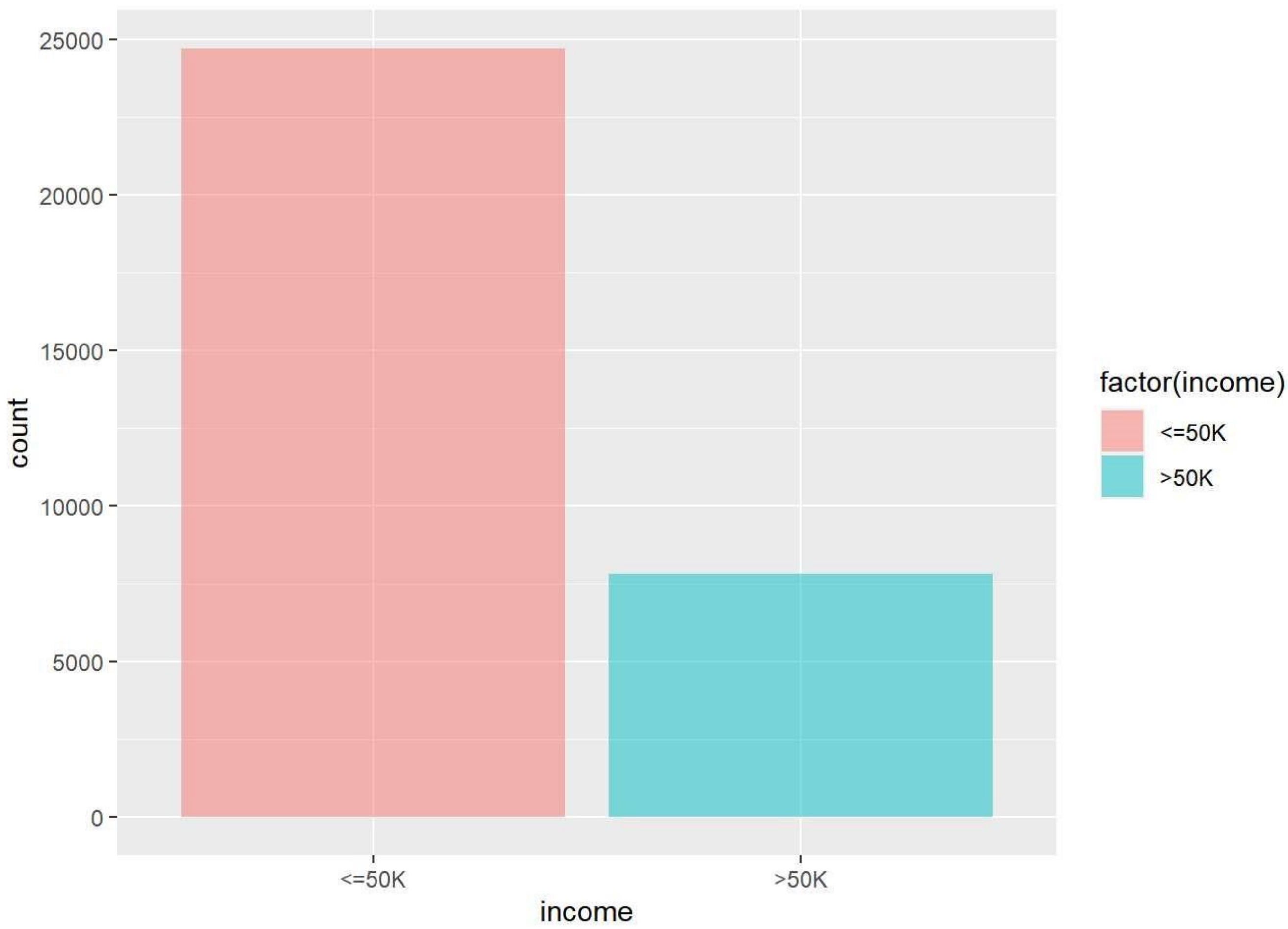
```
ggplot(df,aes(sex))+
        geom_bar(aes(fill=factor(sex)),alpha=0.5)
```



Again we can see that the classes are imbalanced

Plotting income

```
ggplot(df,aes(income))+
        geom_bar(aes(fill=factor(income)),alpha=0.5)
```



The target class is highly imbalanced

Data pre process for Modelling

Create test and train

```
create_train_test <- function(data, size = 0.8, train = TRUE)
{
  #' create_train_test(df, size = 0.8, train = TRUE)
  #' arguments:
  #' @param df: Dataset used to train the model.
  #' @param size: Size of the split. By default, 0.8. Numerical value
  #' @param train: If set to `TRUE`, the function creates the train set, otherwise the test s
et. Default value sets to `TRUE`. Boolean value.You      need to add a Boolean parameter beca
use R does not allow to return two data frames simultaneously.

  #' @return test/train data


    n_row = nrow(data)
    total_row = size * n_row
    train_sample <- 1: total_row
    if (train == TRUE) {
        return (data[train_sample, ])
    } else {
        return (data[-train_sample, ])
    }
}
```

##Getting data

```
data_train <- create_train_test(df, 0.8, train = TRUE)
data_test <- create_train_test(df, 0.8, train = FALSE)
dim(data_train)
```

```
## [1] 26048    13
```

Seeing propotion of data

```
prop.table(table(data_train$income))
```

```
##
##     <=50K      >50K
## 0.7470439 0.2529561
```

```
prop.table(table(data_test$income))
```

```
##
##     <=50K      >50K
## 0.8077691 0.1922309
```

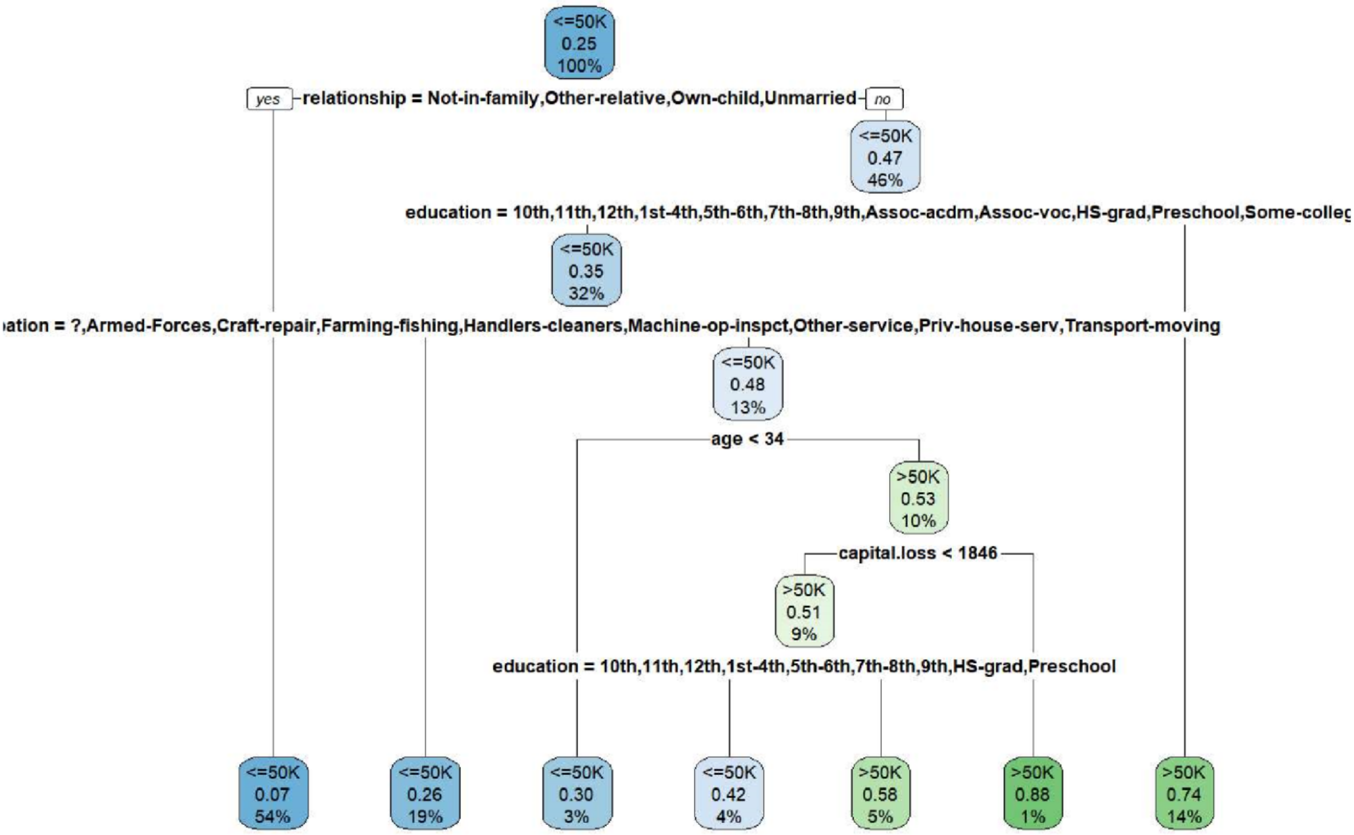
Decision Tree

```
tic("Running the decision tree: ")
fit_dt <- rpart(income~., data = data_train, method = 'class')
toc()
```

```
## Running the decision tree: : 0.8 sec elapsed
```

Plotting the tree,

```
rpart.plot(fit_dt, extra = 106)
```




```
predict_unseen <-predict(fit_dt, data_test, type = 'class')
```

```
table_mat <- table(data_test$income, predict_unseen)
table_mat
```

```
##        predict_unseen
##          <=50K >50K
##   <=50K   4859  402
##   >50K     573  679
```

Evaluations on Decision tree

```
accuracy_Test <- sum(diag(table_mat)) / sum(table_mat)
```

```
n = sum(table_mat) # number of instances
nc = nrow(table_mat) # number of classes
diag = diag(table_mat) # number of correctly classified instances per class
rowsums = apply(table_mat, 1, sum) # number of instances per class
colsums = apply(table_mat, 2, sum) # number of predictions per class
p = rowsums / n # distribution of instances over the actual classes
q = colsums / n # distribution of instances over the predicted classes
```

```
precision = diag / colsums
recall = diag / rowsums
f1 = 2 * precision * recall / (precision + recall)
```

Printing Metrics for Decision tree

```
print(table_mat)
```

```
##        predict_unseen
##         <=50K >50K
##   <=50K  4859  402
##   >50K    573  679
```

```
print(paste(accuracy_Test, "is the accuracy"))
```

```
## [1] "0.850299401197605 is the accuracy"
```

```
print(paste(precision, "is the precision"))
```

```
## [1] "0.894513991163476 is the precision" "0.628122109158187 is the precision"
```

```
print(paste(recall, "is the recall"))
```

```
## [1] "0.923588671355256 is the recall" "0.542332268370607 is the recall"
```

```
print(paste(f1, "is the f1"))
```

```
## [1] "0.908818853455532 is the f1" "0.582083154736391 is the f1"
```

Random Forest

Making the model

```
tic("Running random Forest: ")
model_rf <- randomForest(income ~ ., data = data_train, importance = TRUE)
toc()
```

e

```
## Running random Forest: : 67.87 sec elapsed
```

```
predict_unseen <-predict(model_rf, data_test, type = 'class')
```

```
predict_unseen = as.data.frame(predict_unseen)
```

```
table_mat <- table(data_test$income, predict_unseen$predict_unseen)
```

Evaluations on Random Forest

```
accuracy_Test <- sum(diag(table_mat)) / sum(table_mat)
```

```
 n = sum(table_mat) # number of instances
 nc = nrow(table_mat) # number of classes
 diag = diag(table_mat) # number of correctly classified instances per class
 rowsums = apply(table_mat, 1, sum) # number of instances per class
 colsums = apply(table_mat, 2, sum) # number of predictions per class
 p = rowsums / n # distribution of instances over the actual classes
 q = colsums / n # distribution of instances over the predicted classes
```

```
 precision = diag / colsums
 recall = diag / rowsums
 f1 = 2 * precision * recall / (precision + recall)
```

```
print(table_mat)
```

```
##
##         <=50K >50K
##   <=50K  4810  451
##   >50K    505  747
```

```
print(paste(accuracy_Test, "is the accuracy"))
```

```
## [1] "0.853216643635805 is the accuracy"
```

```
print(paste(precision, "is the precision"))
```

```
## [1] "0.904985888993415 is the precision" "0.623539232053422 is the precision"
```

```
print(paste(recall, "is the recall"))
```

```
## [1] "0.914274852689603 is the recall" "0.596645367412141 is the recall"
```

```
print(paste(f1, "is the f1"))
```

e

```
## [1] "0.909606656580938 is the f1" "0.609795918367347 is the f1"
```

INFERENCE:

First, let us see the score of random forest and decision tree

The different evaluations that were done

Decision Tree :

1. accuracy -> 85.03 | This means that the decision tree has correctly predicted the class on the test data 94.09% of the time
2. precision -> 89.45 | The model gave correct predictions for a class 1, the model predicted correctly
3. precision -> 62.81 | The model gave correct predictions for a class 2, the model predicted correctly
4. recall -> 0.9 | This is the fraction of instances of a class 1 that were correctly predicted, that is 0.9
5. recall -> 0.5 | This is the fraction of instances of a class 2 that were correctly predicted, that is 0.9
6. f1 -> 90.88 | This is the harmonic mean of precision and recall, for class 1
7. f1 -> 58.21 | This is the harmonic mean of precision and recall, for class 2
8. time -> $0.98s$

Random Forest :

1. accuracy -> 85.47 | This means that the decision tree has correctly predicted the class on the test data 94.09% of the time
2. precision -> 90.60 | The model gave correct predictions for a class 1, the model predicted correctly
3. precision -> 62.77 | The model gave correct predictions for a class 2, the model predicted correctly
4. recall -> 0.9 | This is the fraction of instances of a class 1 that were correctly predicted, that is 0.9
5. recall -> 0.6 | This is the fraction of instances of a class 2 that were correctly predicted, that is 0.9
6. f1 -> 91.05 | This is the harmonic mean of precision and recall, for class 1
7. f1 -> 61.39 | This is the harmonic mean of precision and recall, for class 2
8. time -> $88.98s$

Since we can see that we did not get any dramatic change while using random forest and the difference in time in huge. We can use random forest when we are suffering with overfitting. In this case, random forest did slightly better than decision tree.