Final_Project_Adult_Dataset

Tong Wu

2022-12-08

Final project - Adult Census Income Level Prediction Reproduce and Research of Different Approach

Initialise - Import Adult Dataset and Data Cleaning

Initialise Functions

```
library(OneR)
```

```
## Warning:   'OneR' R 4.2.2
```

```
library(ggplot2)
library(dplyr)
```

```
##
##     'dplyr'

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

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

```
# Function of draw the confusion matrix
draw_confusion_matrix <- function(cm, title, class1, class2) {

  layout(matrix(c(1,1,2)))
  par(mar=c(2,2,2,2))
  plot(c(100, 345), c(300, 450), type = "n", xlab="", ylab="", xaxt='n', yaxt='n')
  title(title, cex.main=2)

  # create the matrix
  rect(150, 430, 240, 370, col='#3F97D0')
  text(195, 435, class1, cex=1.2)
  rect(250, 430, 340, 370, col='#F7AD50')
```

```
  text(295, 435, class2, cex=1.2)
  text(125, 370, 'Predicted', cex=1.3, srt=90, font=2)
  text(245, 450, 'Actual', cex=1.3, font=2)
  rect(150, 305, 240, 365, col='#F7AD50')
  rect(250, 305, 340, 365, col='#3F97D0')
  text(140, 400, class1, cex=1.2, srt=90)
  text(140, 335, class2, cex=1.2, srt=90)

  # add in the cm results
  res <- as.numeric(cm$table)
  text(195, 400, res[1], cex=1.6, font=2, col='white')
  text(195, 335, res[2], cex=1.6, font=2, col='white')
  text(295, 400, res[3], cex=1.6, font=2, col='white')
  text(295, 335, res[4], cex=1.6, font=2, col='white')

  # add in the specifics
  plot(c(100, 0), c(100, 0), type = "n", xlab="", ylab="", main = "DETAILS", xaxt='n', yaxt='n')
  text(10, 85, names(cm$byClass[1]), cex=1.2, font=2)
  text(10, 70, round(as.numeric(cm$byClass[1]), 3), cex=1.2)
  text(30, 85, names(cm$byClass[2]), cex=1.2, font=2)
  text(30, 70, round(as.numeric(cm$byClass[2]), 3), cex=1.2)
  text(50, 85, names(cm$byClass[5]), cex=1.2, font=2)
  text(50, 70, round(as.numeric(cm$byClass[5]), 3), cex=1.2)
  text(70, 85, names(cm$byClass[6]), cex=1.2, font=2)
  text(70, 70, round(as.numeric(cm$byClass[6]), 3), cex=1.2)
  text(90, 85, names(cm$byClass[7]), cex=1.2, font=2)
  text(90, 70, round(as.numeric(cm$byClass[7]), 3), cex=1.2)

  # add in the accuracy information
  text(30, 35, names(cm$overall[1]), cex=1.5, font=2)
  text(30, 20, round(as.numeric(cm$overall[1]), 3), cex=1.4)
  text(70, 35, names(cm$overall[2]), cex=1.5, font=2)
  text(70, 20, round(as.numeric(cm$overall[2]), 3), cex=1.4)
}
draw_confusion_matrix_simple <- function(cm, title, class1, class2) {

  layout(matrix(c(1,1,2)))
  par(mar=c(2,2,2,2))
  plot(c(100, 345), c(300, 450), type = "n", xlab="", ylab="", xaxt='n', yaxt='n')
  title(title, cex.main=2)

  # create the matrix
  rect(150, 430, 240, 370, col='#3F97D0')
  text(195, 435, class1, cex=1.2)
  rect(250, 430, 340, 370, col='#F7AD50')
  text(295, 435, class2, cex=1.2)
  text(125, 370, 'Predicted', cex=1.3, srt=90, font=2)
  text(245, 450, 'Actual', cex=1.3, font=2)
  rect(150, 305, 240, 365, col='#F7AD50')
  rect(250, 305, 340, 365, col='#3F97D0')
  text(140, 400, class1, cex=1.2, srt=90)
  text(140, 335, class2, cex=1.2, srt=90)
```

```
  # add in the cm results
  res <- as.numeric(cm)
  text(195, 400, res[1], cex=1.6, font=2, col='white')
  text(195, 335, res[2], cex=1.6, font=2, col='white')
  text(295, 400, res[3], cex=1.6, font=2, col='white')
  text(295, 335, res[4], cex=1.6, font=2, col='white')

  # add in the specifics
  plot(c(100, 0), c(100, 0), type = "n", xlab="", ylab="", main = "DETAILS", xaxt='n', yaxt='n')
  accuracy = (res[1] + res[4]) / (res[1] + res[2] + res[3] + res[4])
  # add in the accuracy information
  text(50, 35, 'accuracy', cex=1.5, font=2)
  text(50, 20, round(accuracy, 3), cex=1.4)
}
```

Read Dataset

```
#Read the adult data set
adult <- read.csv(url("https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data"), hea

# Apply column names
colname<-c("age", "type_employer", "fnlwgt", "education",
                "education_num","marital", "occupation", "relationship", "race","sex",
                "capital_gain", "capital_loss", "hr_per_week","country", "income")
colnames(adult)<-colname

# Continue Variable to char
varcontinue <- c("age","fnlwgt","education_num","capital_gain","capital_loss","hr_per_week")
adult <- cbind(lapply(adult[,varcontinue],function(x) as.numeric(as.character(x))),adult[,setdiff(colnar
head(adult)
```

```
##   age fnlwgt education_num capital_gain capital_loss hr_per_week
## 1  39  77516            13         2174            0          40
## 2  50  83311            13            0            0          13
## 3  38 215646             9            0            0          40
## 4  53 234721             7            0            0          40
## 5  28 338409            13            0            0          40
## 6  37 284582            14            0            0          40
##       type_employer  education             marital         occupation
## 1         State-gov  Bachelors       Never-married       Adm-clerical
## 2  Self-emp-not-inc  Bachelors  Married-civ-spouse    Exec-managerial
## 3           Private    HS-grad            Divorced  Handlers-cleaners
## 4           Private       11th  Married-civ-spouse  Handlers-cleaners
## 5           Private  Bachelors  Married-civ-spouse     Prof-specialty
## 6           Private    Masters  Married-civ-spouse    Exec-managerial
##     relationship   race     sex        country income
## 1  Not-in-family  White    Male  United-States  <=50K
## 2        Husband  White    Male  United-States  <=50K
## 3  Not-in-family  White    Male  United-States  <=50K
## 4        Husband  Black    Male  United-States  <=50K
## 5           Wife  Black  Female           Cuba  <=50K
## 6           Wife  White  Female  United-States  <=50K
```

Data Cleaning

```
# Show variable type
str(adult)
```

```
## 'data.frame':    32561 obs. of  15 variables:
##  $ age          : num  39 50 38 53 28 37 49 52 31 42 ...
##  $ fnlwgt       : num  77516 83311 215646 234721 338409 ...
##  $ education_num: num  13 13 9 7 13 14 5 9 14 13 ...
##  $ capital_gain : num  2174 0 0 0 0 ...
##  $ capital_loss : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ hr_per_week  : num  40 13 40 40 40 40 16 45 50 40 ...
##  $ type_employer: Factor w/ 9 levels " ?"," Federal-gov",..: 8 7 5 5 5 5 5 7 5 5 ...
##  $ education    : Factor w/ 16 levels " 10th"," 11th",..: 10 10 12 2 10 13 7 12 13 10 ...
##  $ marital      : Factor w/ 7 levels " Divorced"," Married-AF-spouse",..: 5 3 1 3 3 3 4 3 5 3 ...
##  $ occupation   : Factor w/ 15 levels " ?"," Adm-clerical",..: 2 5 7 7 11 5 9 5 11 5 ...
##  $ relationship : Factor w/ 6 levels " Husband"," Not-in-family",..: 2 1 2 1 6 6 2 1 2 1 ...
##  $ race         : Factor w/ 5 levels " Amer-Indian-Eskimo",..: 5 5 5 3 3 5 3 5 5 5 ...
##  $ sex          : Factor w/ 2 levels " Female"," Male": 2 2 2 2 1 1 1 2 1 2 ...
##  $ country      : Factor w/ 42 levels " ?"," Cambodia",..: 40 40 40 40 6 40 24 40 40 40 ...
##  $ income       : Factor w/ 2 levels " <=50K"," >50K": 1 1 1 1 1 1 1 2 2 2 ...
```

```
# Check the missing value
# table(adult$type_employer)

# Seems like the missing value is replaced by "?"
# Change the missing value to NA, which is readable for R
adult$type_employer[adult$type_employer==" ?"]<-NA
adult$occupation[adult$occupation==" ?"]<-NA
adult$country[adult$country==" ?"]<-NA

# Delete the missing value
adult_cl <- na.omit(adult)
print(nrow(adult))
```

```
## [1] 32561
```

```
print(nrow(adult_cl))
```

```
## [1] 30162
```

```
str(adult_cl)
```

```
## 'data.frame':    30162 obs. of  15 variables:
##  $ age          : num  39 50 38 53 28 37 49 52 31 42 ...
##  $ fnlwgt       : num  77516 83311 215646 234721 338409 ...
##  $ education_num: num  13 13 9 7 13 14 5 9 14 13 ...
##  $ capital_gain : num  2174 0 0 0 0 ...
##  $ capital_loss : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ hr_per_week  : num  40 13 40 40 40 40 16 45 50 40 ...
```

```
##  $ type_employer: Factor w/ 9 levels " ?"," Federal-gov",..: 8 7 5 5 5 5 5 7 5 5 ...
##  $ education    : Factor w/ 16 levels " 10th"," 11th",..: 10 10 12 2 10 13 7 12 13 10 ...
##  $ marital      : Factor w/ 7 levels " Divorced"," Married-AF-spouse",..: 5 3 1 3 3 3 4 3 5 3 ...
##  $ occupation   : Factor w/ 15 levels " ?"," Adm-clerical",..: 2 5 7 7 11 5 9 5 11 5 ...
##  $ relationship : Factor w/ 6 levels " Husband"," Not-in-family",..: 2 1 2 1 6 6 2 1 2 1 ...
##  $ race         : Factor w/ 5 levels " Amer-Indian-Eskimo",..: 5 5 5 3 3 5 3 5 5 5 ...
##  $ sex          : Factor w/ 2 levels " Female"," Male": 2 2 2 2 1 1 1 2 1 2 ...
##  $ country      : Factor w/ 42 levels " ?"," Cambodia",..: 40 40 40 40 6 40 24 40 40 40 ...
##  $ income       : Factor w/ 2 levels " <=50K"," >50K": 1 1 1 1 1 1 1 2 2 2 ...
##  - attr(*, "na.action")= 'omit' Named int [1:2399] 15 28 39 52 62 70 78 94 107 129 ...
##   ..- attr(*, "names")= chr [1:2399] "15" "28" "39" "52" ...
```

```
# More than 2k data containing missing value has been deleted.
# Data clean finished
```

Reproduce paper result

1 - Using decision tree classifier to predict income levels [1]

Bekena using random forest to predit the income level of adult, and analysis the income level of an individual with certain attributes, and the key features determining incoming level. This section is trying to reproduce the result comes from Bekena by using the same approach.

Copy from Bekena's paper about the method section: "A supervised machine learning approach of Random Forest Classifier is used for the study. Random forest classifier is chosen due to two reasons. First since the outcome (target) variable is binary variable (income level >50K or not), using classification algorithms is better than regression algorithms. This is because the target having only values of 0 and 1, regression algorithms will perform less due to less variation in the target variable. Secondly, random forest classifier is found to have better accuracy score compared to gaussian naive baise classifier. Random forest and decisionTree Classifier gave accuracy score of 85% while GaussianNB gave accuracy of 78%. Random forest is preferred to decision tree since using results from many decision trees will avoid the over fitting problem associated with using a decision tree classifier. The results from the model show that both random forest and decision tree gave similar results. This is shown by the fact that the top 7 features in importance are the same in the two models"

```
# Different attributes correlation with income
# Bekena Paper Page 8
# install.packages("PerformanceAnalytics")
library("PerformanceAnalytics")
```

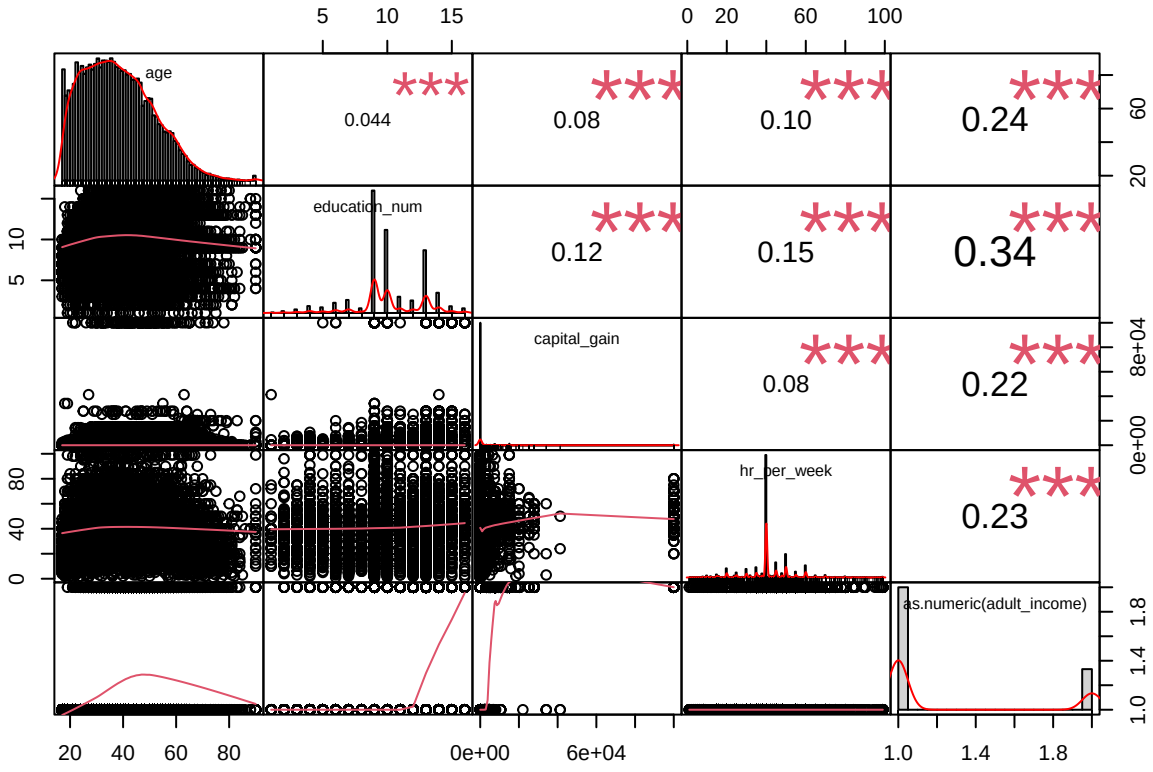
Data exploration

```
## Warning:   'PerformanceAnalytics' R 4.2.2

##      xts

## Warning:   'xts' R 4.2.2

##      zoo

## Warning:   'zoo' R 4.2.2

##
##    'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

##
##    'xts'

## The following objects are masked from 'package:dplyr':
##
##     first, last
```

```
##
##     'PerformanceAnalytics'

## The following object is masked from 'package:graphics':
##
##     legend
```

```
# Choose age, education_num, capital_gain, hr_per_week, income_level
adult_cor <- adult_cl[, c("age", "education_num", "capital_gain","hr_per_week")]
# str(adult_cor)
adult_income <- adult_cl[, c("income")]
# str(adult_income)
adult_cor <- cbind(adult_cor, as.numeric(adult_income))
# str(adult_cor)

# Correlation graph
chart.Correlation(adult_cor, histogram=TRUE, pch=19)
```

```
## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter
```



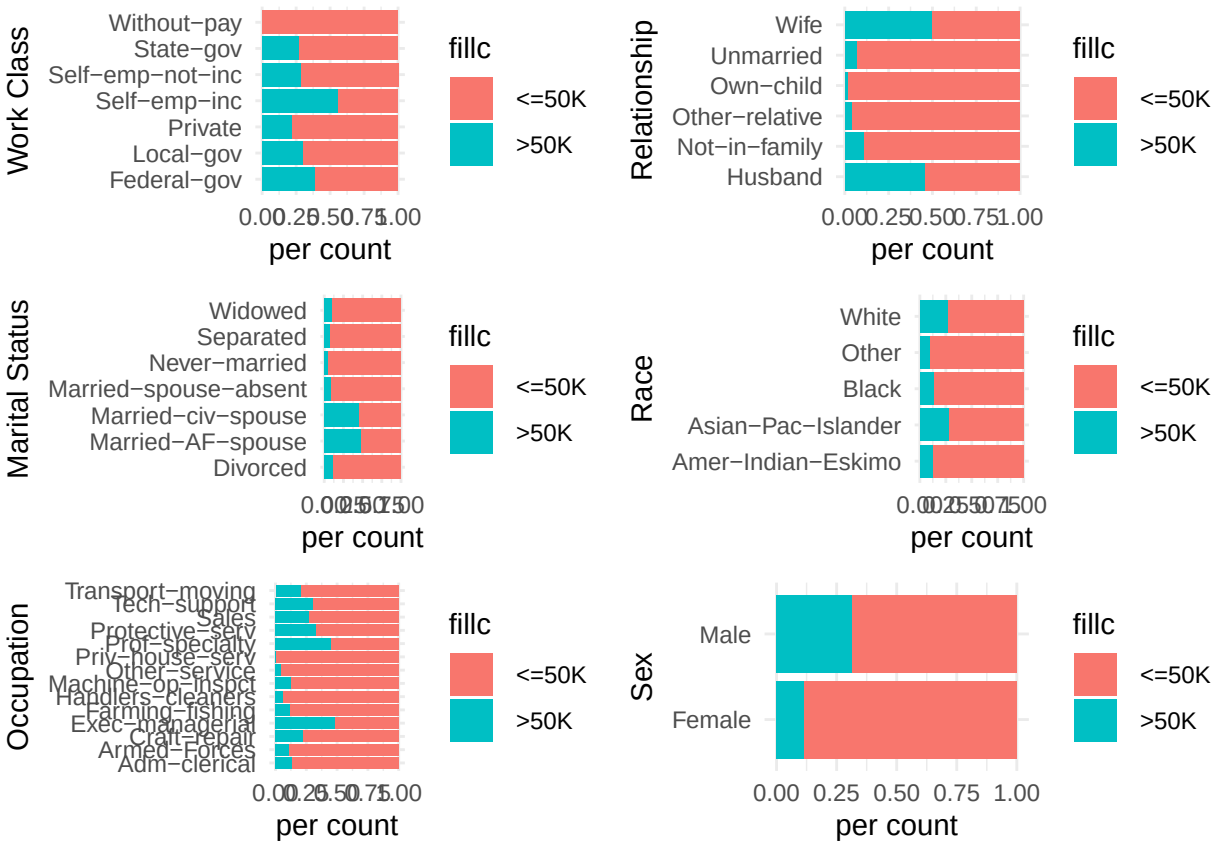
The correlation graph is same as Bekena's, where the education has the highest corrlation +0.34 with income. Capital gain, age and hours worked per week are also positively correlated with income with around +0.20.

```
# Plots of individuals with low and high income by different variables
# Bekena Paper Page 9

# Uncomment below if the rlang is required for higher version
## remove.packages('rlang')
## install.packages("rlang")

# Uncomment below if the Rmisc package is not found
## install.packages("Rmisc")
## install.packages("multiplot")
library(ggplot2)
library(Rmisc)
```

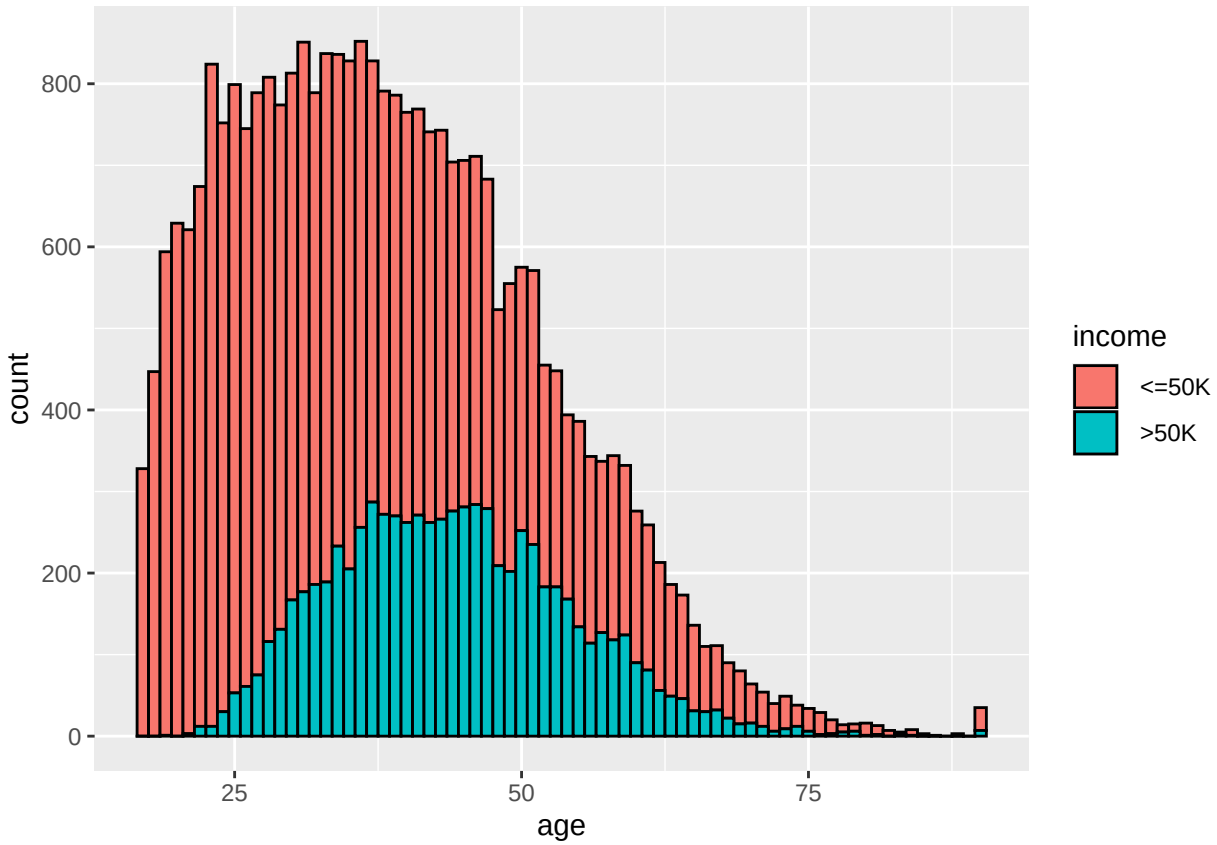
```
## Warning:   'Rmisc' R 4.2.2

##      lattice

##      plyr

## Warning:   'plyr' R 4.2.2

## ------------------------------------------------------------------------------
```

```
## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)

## ------------------------------------------------------------------------------

##
##     'plyr'

## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```
library(plyr)
plot_indrel <- function(data, ylab, fillc, pos, xname, yname) {
  ggplot(data, aes(ylab, fill = fillc)) +
    geom_bar(position = pos) +
    labs(x = xname, y = yname) +
    coord_flip() +
    theme_minimal()
}
# Work class
p1 <- plot_indrel(data = adult_cl, ylab = adult_cl$type_employer, fillc = adult_cl$income,
              pos = 'fill',  xname = 'Work Class', yname = 'per count')
# Marital status
p2 <- plot_indrel(data = adult_cl, ylab = adult_cl$marital, fillc = adult_cl$income,
              pos = 'fill',  xname = 'Marital Status', yname = 'per count')
# Occupation
p3 <- plot_indrel(data = adult_cl, ylab = adult_cl$occupation, fillc = adult_cl$income,
              pos = 'fill',  xname = 'Occupation', yname = 'per count')
# Relationship
p4 <- plot_indrel(data = adult_cl, ylab = adult_cl$relationship, fillc = adult_cl$income,
              pos = 'fill',  xname = 'Relationship', yname = 'per count')
# Race
p5 <- plot_indrel(data = adult_cl, ylab = adult_cl$race, fillc = adult_cl$income,
              pos = 'fill',  xname = 'Race', yname = 'per count')
# Sex
p6 <- plot_indrel(data = adult_cl, ylab = adult_cl$sex, fillc = adult_cl$income,
              pos = 'fill',  xname = 'Sex', yname = 'per count')
# plot(p1)
# plot(p2)
# plot(p3)
# plot(p4)
# plot(p5)
# plot(p6)
multiplot(p1, p2, p3, p4, p5, p6, cols=2)
```




```
# Histogram of ages, classified by income level
ggplot(adult_cl, aes(age)) + geom_histogram(aes(fill = income), color = "black", binwidth = 1)
```


800
600
400
200
0
count
25
50
75
age
income
<=50K
>50K

```
# Build test set and train set
income <- adult_income
## Discard fnlwgt and income, cbind numeric income
### adult_rfc <- adult_cl[, c(1, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15)]
train_rfc=sample(1:nrow(adult_cl),0.8*nrow(adult_cl))
adult_train_rfc <- adult_cl[train_rfc,]
adult_test_rfc <- adult_cl[-train_rfc,]
```

```
# Uncomment below if package randomForest and caret is not found
## install.packages("randomForest")
## install.packages("caret")
library(randomForest)
```

Random Forest Classifier

```
## Warning:   'randomForest' R 4.2.2
```

```
## randomForest 4.7-1.1
```

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

```
##
##     'randomForest'
```

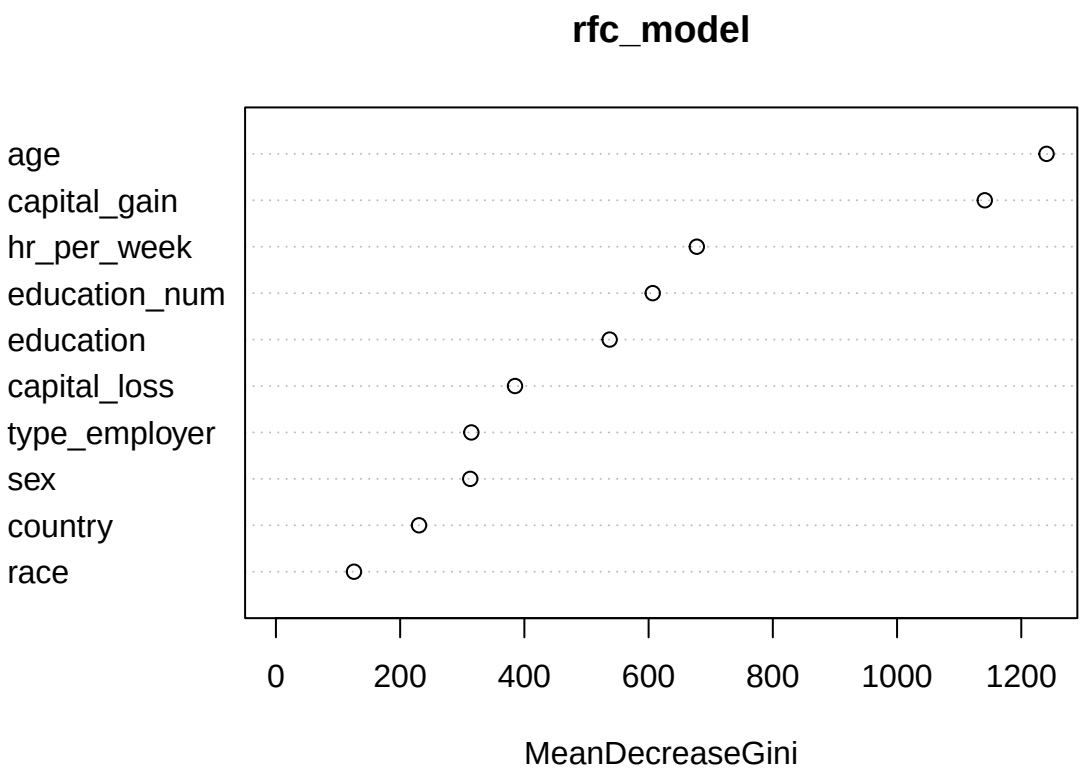
```
## The following object is masked from 'package:dplyr':
##
##     combine
```

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

```
library(caret)
```

```
## Warning:   'caret' R 4.2.2
```

```
# Start training
## Set seed avoiding randomise
set.seed(6690)
rfc_model <- randomForest(income~.-fnlwgt-relationship-marital-occupation, data = adult_train_rfc)
rfc_pred <- predict(rfc_model, adult_train_rfc, type = 'response')
varImpPlot(rfc_model)
```




```
library(dplyr)
# Importance score
importance_score <- varImp(rfc_model, conditional=TRUE)
# print(importance_score)
## sum(importance_score$Overall)
proportions <- importance_score$Overall/sum(importance_score$Overall)
percentages <- proportions*100
cbind(importance_score, percentages)
```

```
##                 Overall percentages
## age           1240.6595   22.266227
## education_num  606.6595   10.887773
## capital_gain  1141.2420   20.481973
## capital_loss   384.9071    6.907962
## hr_per_week    677.5878   12.160730
## type_employer  314.6384    5.646844
## education      537.2752    9.642526
## race           125.6927    2.255819
## sex            312.8727    5.615154
## country        230.3989    4.134990
```

Get the quite same result as Bekena's.

```
set.seed(6690)
rfc_model2 <- randomForest(income~.-fnlwgt-relationship-occupation, data = adult_train_rfc)
rfc_pred2 <- predict(rfc_model2, adult_train_rfc, type = 'response')
## varImpPlot(rfc_model2)
# Importance score
importance_score2 <- varImp(rfc_model2, conditional=TRUE)
# print(importance_score2)
## sum(importance_score$Overall)
proportions2 <- importance_score2$Overall/sum(importance_score2$Overall)
percentages2 <- proportions2*100
cbind(importance_score2, percentages2)
```

```
##                 Overall percentages2
## age            934.8870    15.086857
## education_num  595.9400     9.617057
## capital_gain  1003.4862    16.193886
## capital_loss   320.2147     5.167506
## hr_per_week    546.4961     8.819150
## type_employer  292.9933     4.728217
## education      580.3806     9.365965
## marital       1455.1305    23.482353
## race           106.8782     1.724761
## sex            159.9162     2.580667
## country        200.3752     3.233581
```

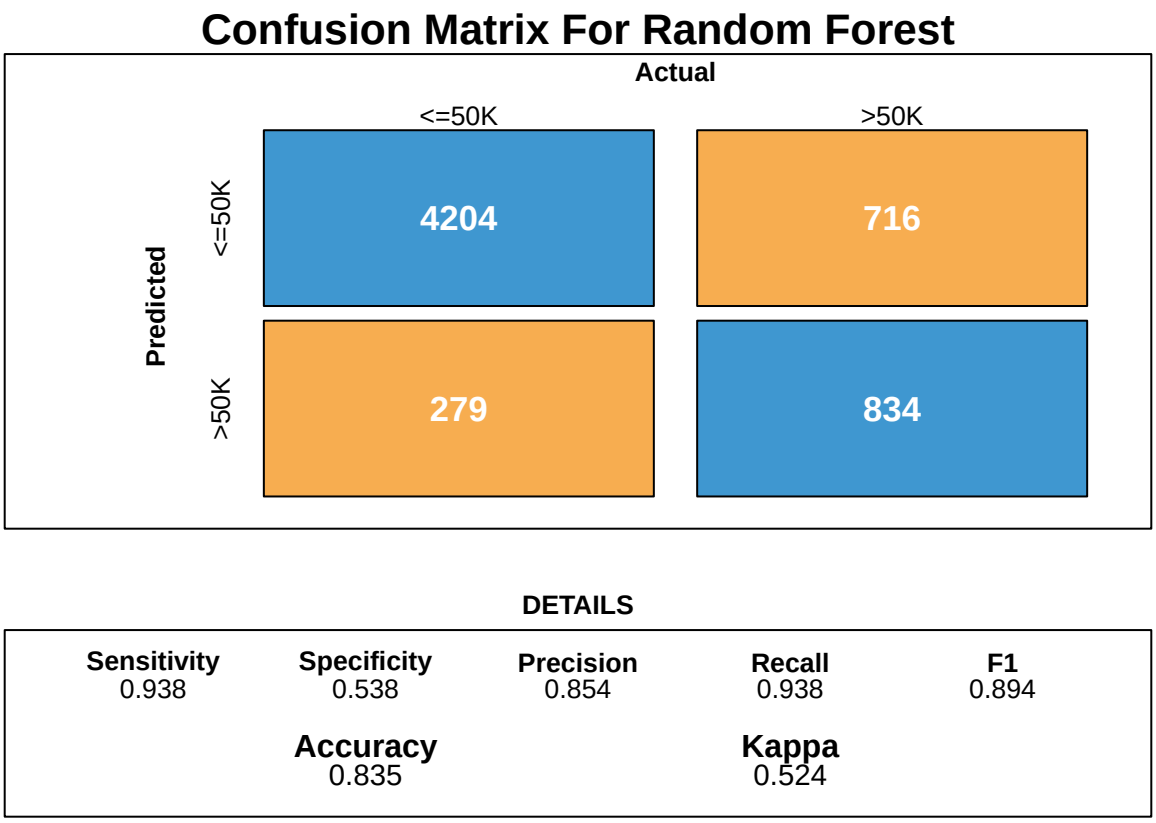
The reason that the attribute "marital" is higher than expected is that, in Bekena's paper, marital status "married" is listed separated, which cause the different value with my result.

```
# Validate through test data set
rfc_pred_test <- predict(rfc_model, adult_test_rfc, type = 'response')
eval_model(rfc_pred_test, adult_test_rfc)
```

```
##
## Confusion matrix (absolute):
##            Actual
## Prediction  <=50K  >50K  Sum
##      <=50K   4204   716 4920
##      >50K     279   834 1113
##     Sum      4483  1550 6033
##
## Confusion matrix (relative):
##            Actual
## Prediction  <=50K  >50K  Sum
##      <=50K   0.70  0.12 0.82
##      >50K    0.05  0.14 0.18
##     Sum      0.74  0.26 1.00
##
## Accuracy:
## 0.8351 (5038/6033)
##
## Error rate:
## 0.1649 (995/6033)
```

```
##
## Error rate reduction (vs. base rate):
## 0.3581 (p-value < 2.2e-16)
```

```
rfc_cm <- confusionMatrix(rfc_pred_test, adult_test_rfc$income)
draw_confusion_matrix(rfc_cm, "Confusion Matrix For Random Forest", "<=50K", ">50K")
```




```
## rfc_pred_test
## adult_test_rfc
```

2 - A Comparative Study of Classification Techniques On Adult Data Set [2]

Deepajothi and Selvarajan used multiple classification algorithms to predict the income level using adult data set[2], including Naive Bayesian, KNN, Random forest and Zero R. The result they provided is that the naive bayesian should had the highest accuracy. Now we are trying to reproduce the result. Note that the RF algorithm is reproduced in the previous paper, so in this section other three algorithms will be reproduced.

```
library(naivebayes)
```

Naive Bayesian

```
## Warning:  'naivebayes' R 4.2.2
```

```
## naivebayes 0.9.7 loaded
```

```
# Random split the data set into 0.8 -> training, 0.2 -> testing
set.seed(6690)
train_nb=sample(1:nrow(adult_cl),0.8*nrow(adult_cl))
adult_train_nb <- adult_cl[train_nb,]
adult_test_nb <- adult_cl[-train_nb,]
# Delete the column fnlwgt
adult_train_nb <- adult_train_nb[,-2]
adult_test_nb <- adult_test_nb[,-2]
# Naive Bayes Algorithm
nb_model <- naive_bayes(income~., data = adult_train_nb, laplace = 1)
## print(nb_model)
## plot(nb_model)
nb_pred <- predict(nb_model, adult_train_nb, type = 'prob')
```

```
## Warning: predict.naive_bayes(): more features in the newdata are provided as
## there are probability tables in the object. Calculation is performed based on
## features to be found in the tables.
```

```
# Validate on testing data set
nb_pred_test <- predict(nb_model, adult_test_nb)
```

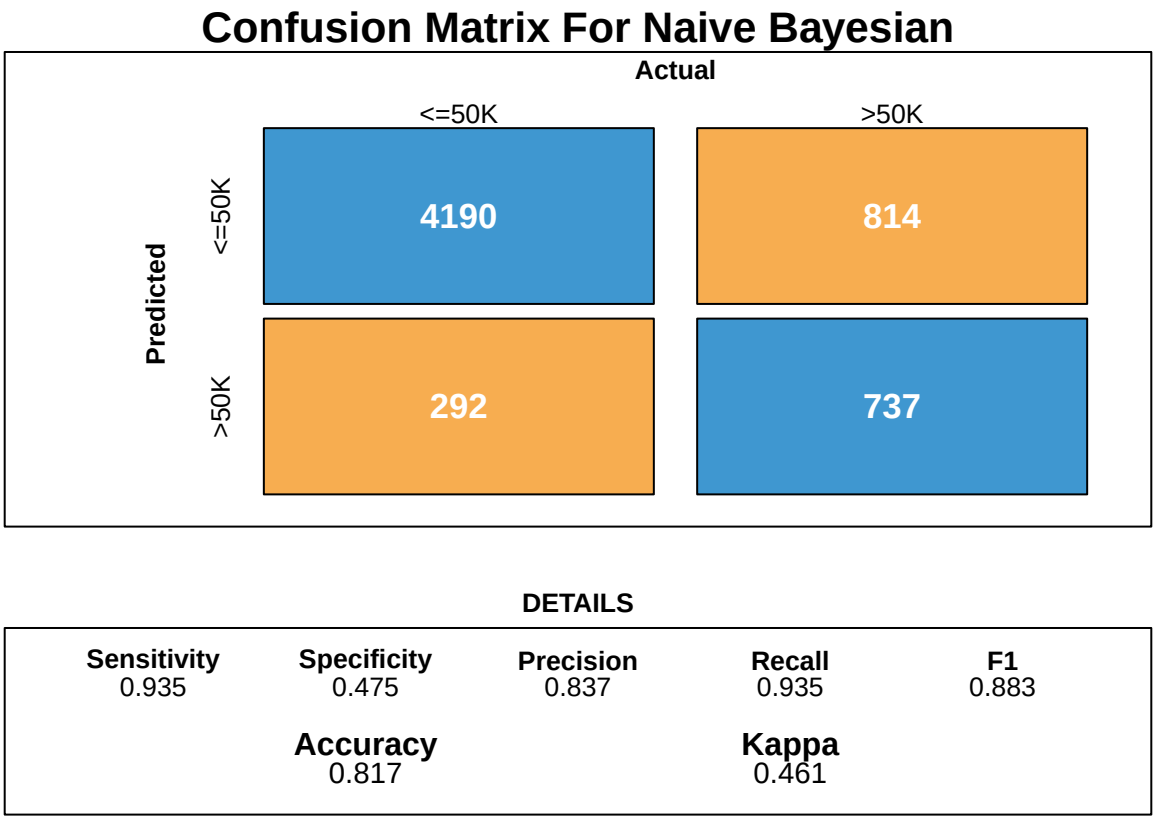
```
## Warning: predict.naive_bayes(): more features in the newdata are provided as
## there are probability tables in the object. Calculation is performed based on
## features to be found in the tables.
```

```
eval_model(nb_pred_test, adult_test_nb)
```

```
##
## Confusion matrix (absolute):
##           Actual
## Prediction  <=50K  >50K  Sum
##      <=50K   4190   814 5004
##      >50K     292   737 1029
```

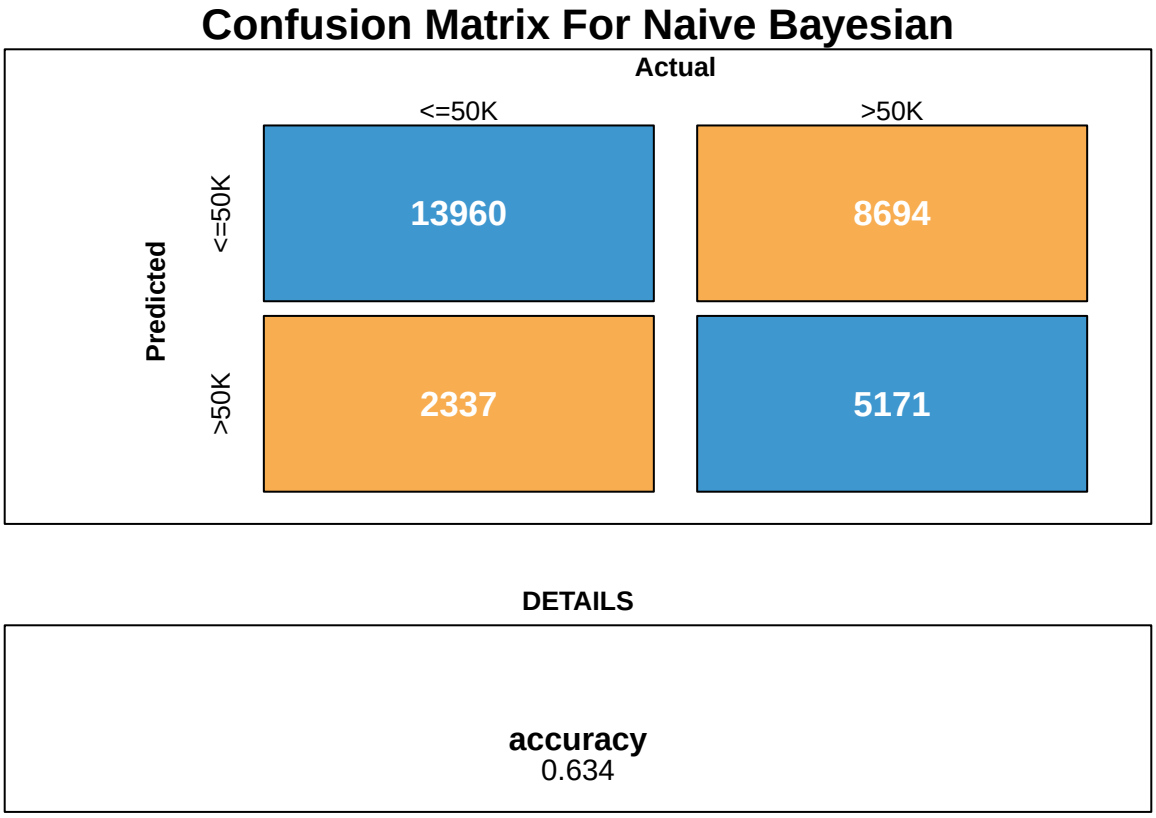
```
##     Sum      4482  1551 6033
##
## Confusion matrix (relative):
##           Actual
## Prediction  <=50K  >50K  Sum
##      <=50K   0.69  0.13 0.83
##      >50K    0.05  0.12 0.17
##     Sum      0.74  0.26 1.00
##
## Accuracy:
## 0.8167 (4927/6033)
##
## Error rate:
## 0.1833 (1106/6033)
##
## Error rate reduction (vs. base rate):
## 0.2869 (p-value < 2.2e-16)
```

```
# Confusion Matrix
nb_cm <- confusionMatrix(data=nb_pred_test, reference=adult_test_nb$income)
draw_confusion_matrix(nb_cm, "Confusion Matrix For Naive Bayesian", "<=50K", ">50K")
```



K-means clustering (K Star) In the paper [2], they are using an algorithm called "K Star", which I can only found several information in Google. Luckily, the chatGPT tells me that the K star is a variant of the KNN algorithm, where K Star is used for clustering and KNN is used for classification. After more digging the example code from chatGPT I found that, the K star "should be" the previous name of K-means algorithm. So in this section, I will use K-means clustering to reproduce the result.

```
library(cluster)
library(dplyr)
# Random split the data set into 0.8 -> training, 0.2 -> testing
set.seed(6690)
adult_km <- adult_cl[,c(-2, -4, -5, -15)]
adult_km <- adult_km %>% mutate_at(c('type_employer', 'education', 'marital', 'occupation', 'relationshi
km_model <- kmeans(adult_km, centers = 2, nstart = 30)
km_cm <- table(adult_cl$income, factor(km_model$cluster))
draw_confusion_matrix_simple(km_cm, "Confusion Matrix For Naive Bayesian", "<=50K", ">50K")
```



The K-means algorithm is unsupervised, so no need to split training and testing subset.

```
# To see the plot, un-comment the code below, but DO NOT try to knit with this chunk
## plot(adult_km,col=km_model$cluster)
# Here is the screenshot of the code above
```

```
knitr::include_graphics("./src/K-Means_plot.jpg")
```

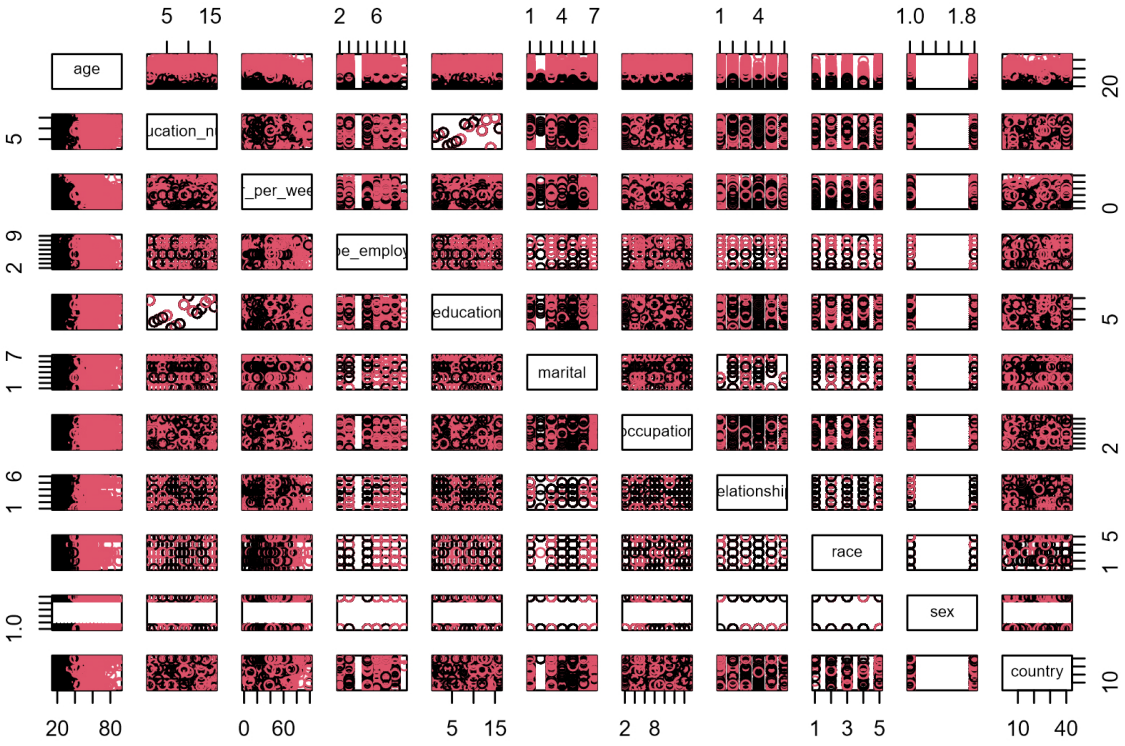


Figure 1: K-Means plot

```
sprintf("The accuracy for predicting income <=50K: %f", km_cm[1]/22654)
```

```
## [1] "The accuracy for predicting income <=50K: 0.616227"
```

```
sprintf("The accuracy for predicting income >50K: %f", 1-km_cm[2]/7508)
```

```
## [1] "The accuracy for predicting income >50K: 0.688732"
```

```
sprintf("The total accuracy for predicting income: %f", (km_cm[1]+7508-km_cm[2])/30162)
```

```
## [1] "The total accuracy for predicting income: 0.634275"
```

```
library(OneR)
adult_zr <- adult_cl[,c(-2)]
train_zr=sample(1:nrow(adult_zr),0.8*nrow(adult_zr))
adult_train_zr <- adult_zr[train_zr,]
adult_test_zr <- adult_zr[-train_zr,]
zr_model <- OneR(cbind(dummy = TRUE, adult_train_zr))
```

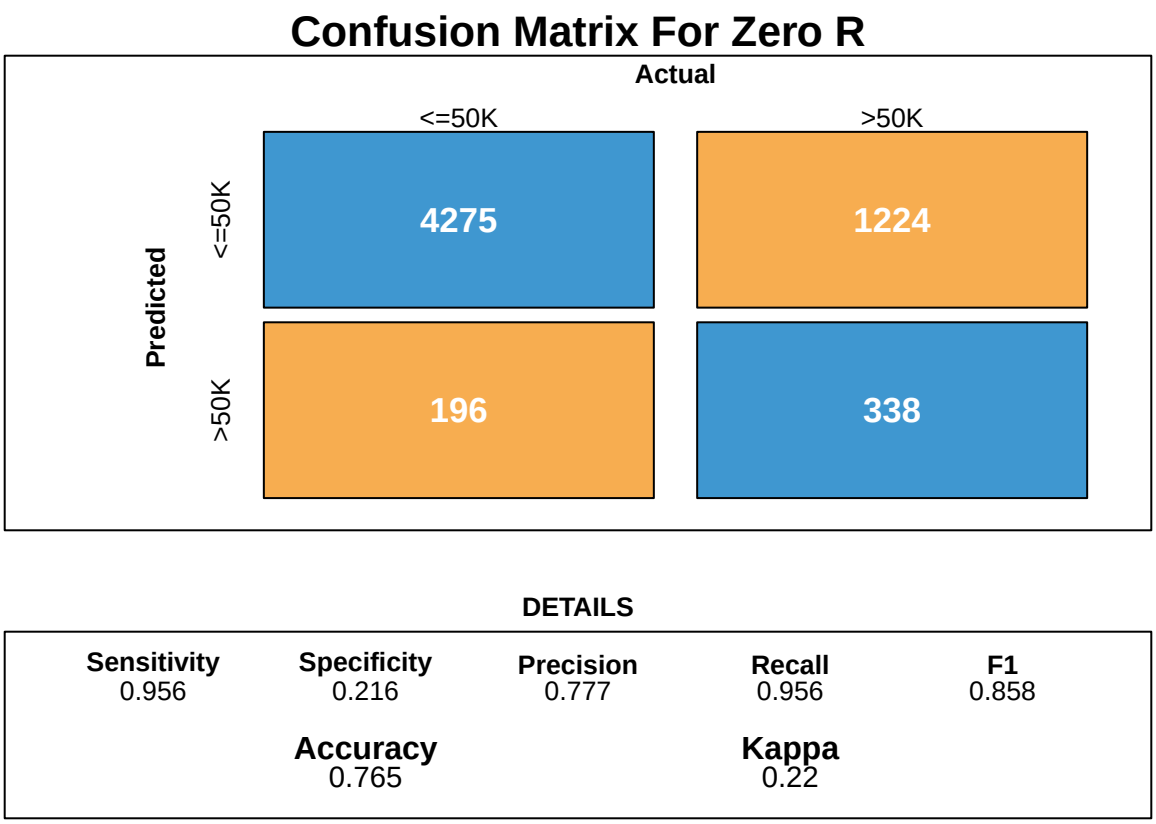
Zero R

```
## Warning in OneR.data.frame(cbind(dummy = TRUE, adult_train_zr)): data contains
## unused factor levels
```

```
class(zr_model) <- "OneR"
zr_pred <- predict(zr_model, cbind(dummy = TRUE, adult_test_zr))
eval_model(zr_pred, adult_test_zr)
```

```
##
## Confusion matrix (absolute):
##            Actual
## Prediction  <=50K  >50K  Sum
##      <=50K   4275  1224 5499
##      >50K     196   338  534
##     Sum      4471  1562 6033
##
## Confusion matrix (relative):
##            Actual
## Prediction  <=50K  >50K  Sum
##      <=50K   0.71  0.20 0.91
##      >50K    0.03  0.06 0.09
##     Sum      0.74  0.26 1.00
##
## Accuracy:
## 0.7646 (4613/6033)
##
## Error rate:
## 0.2354 (1420/6033)
##
## Error rate reduction (vs. base rate):
## 0.0909 (p-value = 1.338e-05)
```

```
zr_cm <- confusionMatrix(zr_pred, adult_test_zr$income)
draw_confusion_matrix(zr_cm, "Confusion Matrix For Zero R", "<=50K", ">50K")
```




```
sprintf("Accuracy for Random Forest: %f", rfc_cm$overall[1])
```

```
## [1] "Accuracy for Random Forest: 0.835074"
```

```
sprintf("Accuracy for Naive Bayesian: %f", nb_cm$overall[1])
```

```
## [1] "Accuracy for Naive Bayesian: 0.816675"
```

```
sprintf("Accuracy for K-Means: %f", (km_cm[1]+7508-km_cm[2])/30162)
```

```
## [1] "Accuracy for K-Means: 0.634275"
```

```
sprintf("Accuracy for Zero R: %f", zr_cm$overall[1])
```

```
## [1] "Accuracy for Zero R: 0.764628"
```

Got the similar result with S.deepajothi, D. s selvarajan [2], where Random Forest and Naive Bayesian has the relative higher accuracy than K-means and Zero R. The result may because of the high accuracy for Naive Bayesian and Random Forest classifier when dealing with the large scale data set. However, K-means as unsupervised learning, learn from un-tagged dataset. Although dimensionality reduction was already done in the code, the high dimension still drag the accuracy. I also tried to implement K-means on the dataset with few dimensions, the accuracy is much higher than more dimensions. Zero R is a naive approach to classify a datase, which based on target and ignores other independent attributes, so it brings fast computing with relative low accuracy.

Another Approach - Logistic Regression and SVM

Logistic Regression

Logistic Regression use several independent variables to predict categorical dependent variables. In this section, I will use Logistic Regression to build the model and predict the income level in order to see the accuracy.

```
library(caret)
# Random split the data set into 0.8 -> training, 0.2 -> testing
train_lr=sample(1:nrow(adult_cl),0.8*nrow(adult_cl))
adult_train_lr <- adult_cl[train_lr,]
adult_test_lr <- adult_cl[-train_lr,]
# Delete the column fnlwgt
adult_train_lr <- adult_train_lr[,-2]
adult_test_lr <- adult_test_lr[,-2]
# Train the logistic regression model
lr_model <- glm(income ~., data = adult_train_lr, family = binomial('logit'))
```

```
## Warning: glm.fit:
```

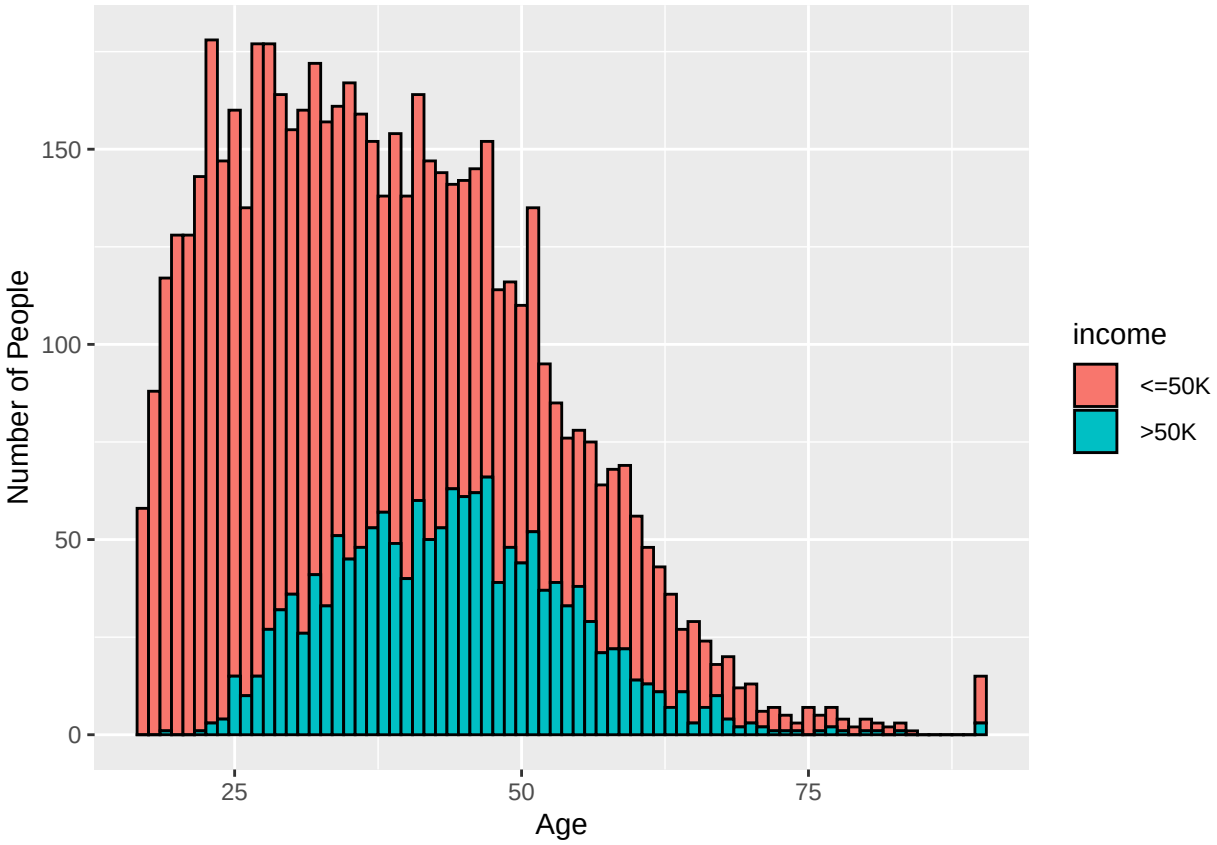
```
summary(lr_model)
```

```
##
## Call:
## glm(formula = income ~ ., family = binomial("logit"), data = adult_train_lr)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -4.3802  -0.5188  -0.1925  -0.0007   3.7106
##
## Coefficients: (1 not defined because of singularities)
##                                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)                      -8.219e+00  9.188e-01  -8.945  < 2e-16 ***
## age                               2.511e-02  1.908e-03  13.158  < 2e-16 ***
## education_num                     2.792e-01  4.485e-02   6.225 4.80e-10 ***
## capital_gain                      3.260e-04  1.214e-05  26.855  < 2e-16 ***
## capital_loss                      5.934e-04  4.303e-05  13.790  < 2e-16 ***
## hr_per_week                       2.983e-02  1.901e-03  15.690  < 2e-16 ***
## type_employer Local-gov          -7.299e-01  1.259e-01  -5.795 6.83e-09 ***
## type_employer Private            -5.743e-01  1.046e-01  -5.488 4.07e-08 ***
## type_employer Self-emp-inc       -3.630e-01  1.386e-01  -2.620 0.008796 **
## type_employer Self-emp-not-inc   -1.032e+00  1.227e-01  -8.411  < 2e-16 ***
## type_employer State-gov          -9.250e-01  1.407e-01  -6.573 4.95e-11 ***
## type_employer Without-pay        -1.429e+01  3.713e+02  -0.038 0.969298
## education 11th                   -1.503e-01  2.123e-01  -0.708 0.479139
## education 12th                   -1.296e-01  2.720e-01  -0.476 0.633786
## education 1st-4th                 8.866e-01  6.000e-01   1.478 0.139500
## education 5th-6th                 4.698e-01  4.926e-01   0.954 0.340216
## education 7th-8th                 8.522e-02  3.337e-01   0.255 0.798425
## education 9th                    -2.435e-03  3.338e-01  -0.007 0.994180
## education Assoc-acdm             -3.931e-01  1.485e-01  -2.647 0.008120 **
## education Assoc-voc              -1.161e-01  1.120e-01  -1.037 0.299856
```

```
## education Bachelors                -8.565e-03  1.569e-01  -0.055 0.956473
## education Doctorate                 2.702e-01  3.338e-01   0.809 0.418231
## education HS-grad                  -6.001e-03  6.512e-02  -0.092 0.926587
## education Masters                   4.922e-02  2.101e-01   0.234 0.814812
## education Preschool                -1.852e+01  1.670e+02  -0.111 0.911671
## education Prof-school               3.682e-01  2.801e-01   1.315 0.188633
## education Some-college                     NA         NA      NA       NA
## marital Married-AF-spouse           2.762e+00  5.902e-01   4.679 2.88e-06 ***
## marital Married-civ-spouse          2.124e+00  3.000e-01   7.079 1.46e-12 ***
## marital Married-spouse-absent       5.776e-02  2.672e-01   0.216 0.828845
## marital Never-married              -5.093e-01  9.988e-02  -5.099 3.41e-07 ***
## marital Separated                  -4.167e-02  1.829e-01  -0.228 0.819774
## marital Widowed                     1.681e-01  1.781e-01   0.944 0.345090
## occupation Armed-Forces            -1.083e+00  1.498e+00  -0.723 0.469655
## occupation Craft-repair             1.153e-01  9.031e-02   1.277 0.201516
## occupation Exec-managerial          7.959e-01  8.708e-02   9.140  < 2e-16 ***
## occupation Farming-fishing         -9.067e-01  1.528e-01  -5.933 2.97e-09 ***
## occupation Handlers-cleaners       -6.773e-01  1.615e-01  -4.193 2.76e-05 ***
## occupation Machine-op-inspct       -2.798e-01  1.146e-01  -2.442 0.014602 *
## occupation Other-service           -8.461e-01  1.345e-01  -6.291 3.16e-10 ***
## occupation Priv-house-serv         -4.228e+00  1.764e+00  -2.397 0.016533 *
## occupation Prof-specialty           4.936e-01  9.209e-02   5.360 8.30e-08 ***
## occupation Protective-serv          6.058e-01  1.408e-01   4.303 1.69e-05 ***
## occupation Sales                    3.188e-01  9.285e-02   3.434 0.000595 ***
## occupation Tech-support             7.607e-01  1.242e-01   6.125 9.09e-10 ***
## occupation Transport-moving        -2.796e-02  1.107e-01  -0.253 0.800539
## relationship Not-in-family          5.037e-01  2.964e-01   1.699 0.089243 .
## relationship Other-relative        -3.372e-01  2.748e-01  -1.227 0.219795
## relationship Own-child             -5.896e-01  2.948e-01  -2.000 0.045501 *
## relationship Unmarried              3.767e-01  3.140e-01   1.200 0.230279
## relationship Wife                   1.384e+00  1.183e-01  11.704  < 2e-16 ***
## race Asian-Pac-Islander             9.046e-01  3.226e-01   2.804 0.005048 **
## race Black                          6.720e-01  2.710e-01   2.479 0.013164 *
## race Other                          2.146e-01  4.145e-01   0.518 0.604623
## race White                          7.158e-01  2.593e-01   2.760 0.005772 **
## sex Male                            8.642e-01  9.045e-02   9.554  < 2e-16 ***
## country Canada                     -5.710e-01  7.309e-01  -0.781 0.434625
## country China                      -1.659e+00  7.411e-01  -2.239 0.025154 *
## country Columbia                   -2.619e+00  1.252e+00  -2.091 0.036489 *
## country Cuba                       -5.867e-01  7.504e-01  -0.782 0.434351
## country Dominican-Republic         -2.541e+00  1.252e+00  -2.029 0.042434 *
## country Ecuador                    -1.088e+00  9.989e-01  -1.089 0.276015
## country El-Salvador                -1.474e+00  8.473e-01  -1.739 0.081948 .
## country England                    -8.797e-01  7.689e-01  -1.144 0.252564
## country France                     -3.663e-01  8.767e-01  -0.418 0.676088
## country Germany                    -4.871e-01  7.203e-01  -0.676 0.498898
## country Greece                     -1.577e+00  8.994e-01  -1.753 0.079544 .
## country Guatemala                  -1.044e+00  1.018e+00  -1.025 0.305158
## country Haiti                      -9.595e-01  9.700e-01  -0.989 0.322564
## country Holand-Netherlands         -1.246e+01  1.455e+03  -0.009 0.993169
## country Honduras                   -2.107e+00  2.682e+00  -0.786 0.432109
## country Hong                       -1.557e+00  1.020e+00  -1.527 0.126819
## country Hungary                    -6.871e-01  1.087e+00  -0.632 0.527132
## country India                      -1.716e+00  7.138e-01  -2.404 0.016207 *
```

```
## country Iran                       -1.333e+00  8.268e-01  -1.613 0.106800
## country Ireland                    -1.452e+00  1.138e+00  -1.276 0.202035
## country Italy                      -2.038e-01  7.578e-01  -0.269 0.787991
## country Jamaica                    -9.993e-01  8.238e-01  -1.213 0.225070
## country Japan                      -5.625e-01  8.025e-01  -0.701 0.483299
## country Laos                       -1.517e+00  1.077e+00  -1.408 0.158997
## country Mexico                     -1.518e+00  7.118e-01  -2.133 0.032951 *
## country Nicaragua                  -1.598e+00  1.047e+00  -1.525 0.127168
## country Outlying-US(Guam-USVI-etc) -1.426e+01  3.584e+02  -0.040 0.968255
## country Peru                       -1.534e+00  1.093e+00  -1.403 0.160528
## country Philippines                -6.984e-01  6.829e-01  -1.023 0.306479
## country Poland                     -1.669e+00  8.638e-01  -1.932 0.053325 .
## country Portugal                   -7.638e-01  9.359e-01  -0.816 0.414440
## country Puerto-Rico                -1.313e+00  7.953e-01  -1.651 0.098725 .
## country Scotland                   -9.427e-01  1.160e+00  -0.813 0.416301
## country South                      -2.232e+00  8.037e-01  -2.777 0.005490 **
## country Taiwan                     -1.306e+00  7.943e-01  -1.644 0.100186
## country Thailand                   -1.243e+00  1.204e+00  -1.032 0.301874
## country Trinadad&Tobago            -2.373e+00  1.304e+00  -1.820 0.068807 .
## country United-States              -8.269e-01  6.678e-01  -1.238 0.215606
## country Vietnam                    -2.253e+00  9.316e-01  -2.419 0.015569 *
## country Yugoslavia                 -5.254e-01  9.776e-01  -0.537 0.590997
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 27026  on 24128  degrees of freedom
## Residual deviance: 15674  on 24034  degrees of freedom
## AIC: 15864
##
## Number of Fisher Scoring iterations: 14
```

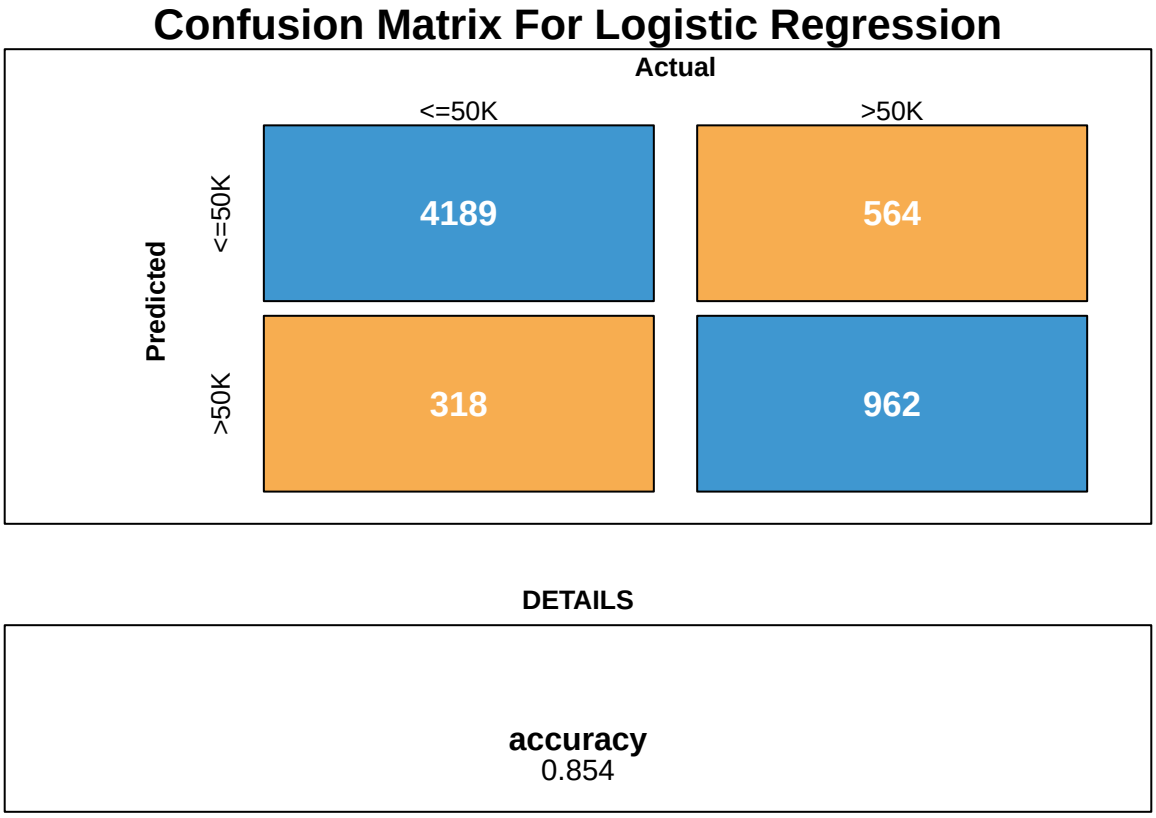
```
library(ggplot2)
# Print the income level percentage histogram of test data set
ggplot(adult_test_lr) + aes(x=as.numeric(age), group=income, fill=income) +
  geom_histogram(binwidth=1, color='black')+labs(x="Age",y="Number of People")
```



```r
# Predict the result from test data set
lr_pred <- predict(lr_model, adult_test_lr, type = 'response')
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
##
```

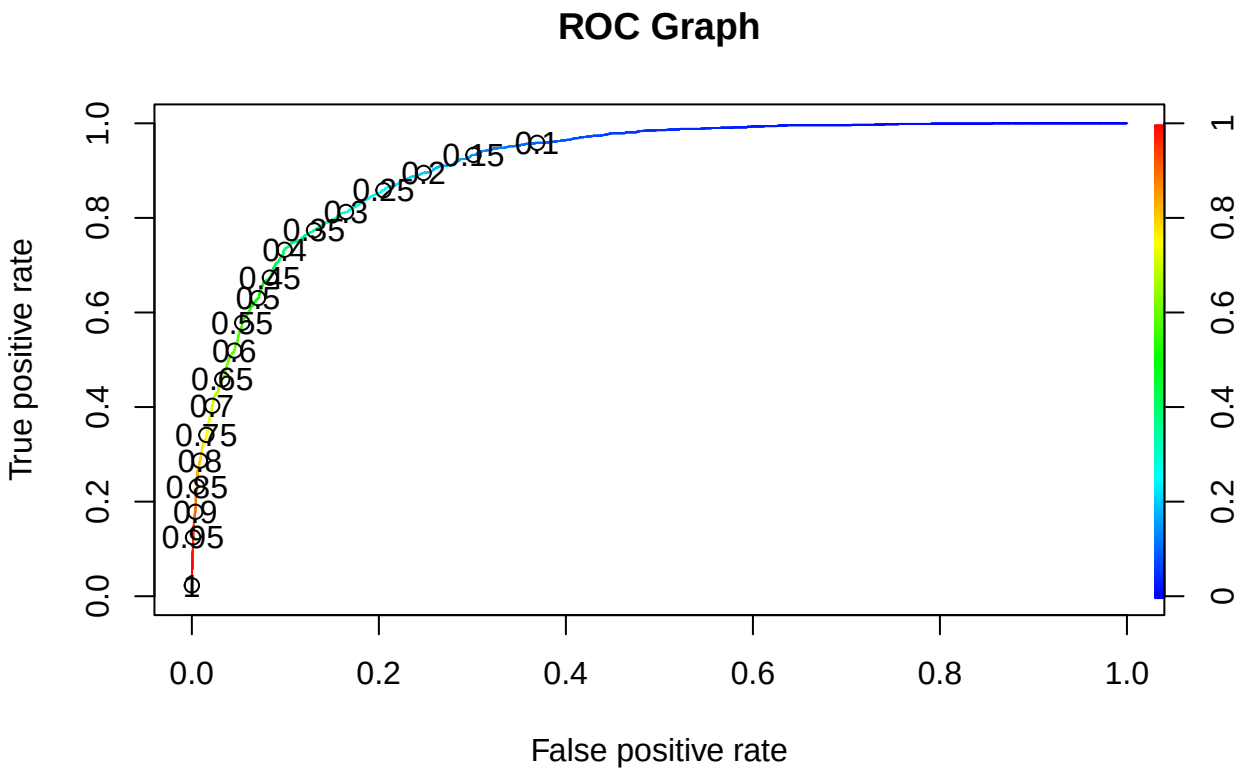
```r
lr_y_pred <- rep('<=50K', length(lr_pred))
lr_y_pred[lr_pred>=.5] <- '>50K'
# confusion matrix
lr_cm<- table(lr_y_pred, adult_test_lr$income)
draw_confusion_matrix_simple(lr_cm, "Confusion Matrix For Logistic Regression", "<=50K", ">50K")
```



```
library(ROCR)
```

```
## Warning:   'ROCR' R 4.2.2
```

```
# Plot ROC curve
lr_roc_pred <- prediction(lr_pred, adult_test_lr$income)
lr_performance <- performance(lr_roc_pred, measure = "tpr", x.measure = "fpr")
plot(lr_performance,colorize=T,print.cutoffs.at=seq(0.1,by=0.05),main="ROC Graph")
```



SVM

The Support Vector Machine can be used to classify the dataset into two groups, which is fit for the adult dataset. The SVM with the principle of a maximum marginal classifier, and a multi-dimensional space to classify data points since the SVM generates a hyperplane in an multi-dimensional space. In this section, I will use the SVM classifier to build the model and predict the income level in order to see the accuracy of SVM.

```
library(ISLR)
library(e1071)
```

```
## Warning:   'e1071' R 4.2.2
```

```
##
##    'e1071'
```

```
## The following objects are masked from 'package:PerformanceAnalytics':
##
##     kurtosis, skewness
```

```
set.seed(6690)
# Random split the data set into 0.8 -> training, 0.2 -> testing
train_svm=sample(1:nrow(adult_cl),0.8*nrow(adult_cl))
adult_train_svm <- adult_cl[train_svm,]
adult_test_svm <- adult_cl[-train_lr,]
# Delete the column fnlwgt
adult_train_svm <- adult_train_svm[,-2]
adult_test_svm <- adult_test_svm[,-2]
# Train the SVM model
svm_model = svm(income ~ .,data = adult_train_svm)
summary(svm_model)
```

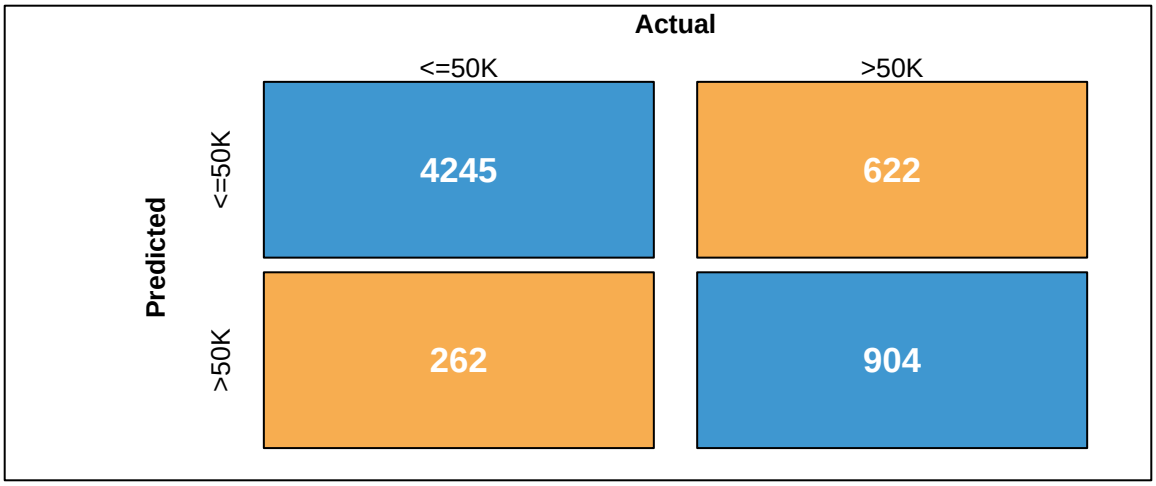
```
##
## Call:
## svm(formula = income ~ ., data = adult_train_svm)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1
##
## Number of Support Vectors:  8675
##
##  ( 4348 4327 )
##
##
## Number of Classes:  2
##
## Levels:
##   <=50K  >50K
```

```r
# Prediction
svm_pred = predict(svm_model, newdata=adult_test_svm, type="response")
eval_model(svm_pred, adult_test_svm)
```

```
##
## Confusion matrix (absolute):
##           Actual
## Prediction  <=50K  >50K  Sum
##      <=50K   4245   622 4867
##      >50K     262   904 1166
##     Sum      4507  1526 6033
##
## Confusion matrix (relative):
##           Actual
## Prediction  <=50K  >50K  Sum
##      <=50K   0.70  0.10 0.81
##      >50K    0.04  0.15 0.19
##     Sum      0.75  0.25 1.00
##
## Accuracy:
## 0.8535 (5149/6033)
##
## Error rate:
## 0.1465 (884/6033)
##
## Error rate reduction (vs. base rate):
## 0.4207 (p-value < 2.2e-16)
```

```r
# confusion matrix
svm_cm <- confusionMatrix(svm_pred, adult_test_svm$income)
draw_confusion_matrix(svm_cm, "Confusion Matrix For SVM", "<=50K", ">50K")
```

Confusion Matrix For SVM



DETAILS

Sensitivity	Specificity	Precision	Recall	F1
0.942	0.592	0.872	0.942	0.906

Accuracy	Kappa
0.853	0.579

References

[1] Sisay Menji Bekena:"Using decision tree classifier to predict income levels", Munich Personal RePEc Archive 30th July, 2017

[2] S.deepajothi, D. s selvarajan. (2012). A Comparative Study of Classification Techniques On Adult Data Set. International Journal of Engineering Research, 1(8), 8.