Will your income exceed $50k/year?

Ishita Gupta, Praveena P., Rohith Krishna, Shravanth J, Vijiyashree S.B.

14 April 2020

Contents

Will your income exceed $50k/year?

Ishita Gupta, Praveena P., Rohith Krishna, Shravanth J, Vijiyashree S.B.

14 April 2020

Abstract

We investigate the role of several demographic factors in determining income categorisation setting a level of differentiation at $50,000/year using classification techniques such as logistic regression,linear discriminant analysis, decision trees, gradient boosting and support vector machines.

1 Introduction

In this project, we will use a standard imbalanced machine learning dataset referred to as the "adult" dataset. The dataset is credited to Ronny Kohavi and Barry Becker and was drawn from the 1994 United States Census Bureau data and involves using personal details such as education level, employment status, etc. to predict whether an individual will earn more or less than $50,000 per year. We observe that there are more cases of less than or equal to $50k/ year as compared to more than $50k/year.

The dataset provides 14 input variables that are a mixture of categorical, ordinal, and numerical data types. The complete list of variables is as follows:

- `sal` - Income level of the respondent
- `age` - age of the person
- `wrk` - employment status
- `fwt` - final number of how much of the population it represents
- `edu` - education level
- `enum` - number of years of education
- `mar` - marital status of the respondent
- `occ` - occupation of the respondent
- `rel` - what the individual is relative to others
- `race` - race of the respondent
- `sex` - gender of the respondent
- `gain` - capital gains earn

- `loss` - capital losses incurred
- `hours` - average number of working hours per week
- `cntry` - country of origin

2 Definitions

Age

A period of human life, measured by years from birth. It is a continuous variable.

Workclass

A general term to represent the employment status of an individual. It is a categorical variable with the following categories: Private, self-emp-not-inc, self-emp-inc, Federal-gov, local-gov, State-gov, without-pay and never-worked.

Final Weight

This is the number of people that the census board believes to be representing the population. It is a continuous variable.

Education

The highest level of education achieved by an individual. It is a categorical variable with the following categories: Preschool, 1st-4th, 5th-6th,7th-8th,9th,10th, 11th, 12th, HS-grad, prof-school, Bachelors, some-college, assoc-acdm, assoc-voc, Masters, Doctorate.

Years of Education

The highest level of education achieved by an individual in numerical form. It is a continuous variable.

Marital Status

A term which represents whether one is single, married, separated, divorced, or widowed. Married-civ-spouse corresponds to a civilian spouse while Married-AF-spouse is a spouse in the Armed Forces. It is a categorical variable with the following categories: Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse.

Occupation

A person's usual or principal work or business, especially as a means of earning a living. It is a categorical variable with the following categories: Tech-support, Craft-repair, Other-service,

Sales, Exec-managerial, Prof-specialty, Handlers-cleaners, Machine-op-inspct, Adm-clerical, Farming-fishing, Transport-moving, Priv-house-serv, Protective-serv, Armed-Forces.

Relationship

It represents what this individual is relative to others. For example an individual could be a Husband. Each entry only has one relationship attribute. It is a categorical variable with the following categories: Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried

Race

A group, especially of people, with particular similar physical characteristics, who are considered as belonging to the same type, or the fact of belonging to such a group. It is a categorical variable with the following categories: White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black.

Sex

The biological sex of the individual. It is a categorical variable with binary categorisation (male, female).

Capital Gain and Loss

Income from investment sources other than salary/wages. It is a continuous variable.

Hours-per-week

The average number of hours an individual has reported to work per week.It is a continuous variable.

Native Country

The country someone is born in or native to. It is a categorical variable with the following categories: United-States, Cambodia, England, Puerto-Rico, Canada, Germany, Outlying-US(Guam-USVI-etc), India, Japan, Greece, South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal, Ireland, France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Yugoslavia, El- Salvador, Trinadad&Tobago, Peru, Hong, Holand-Netherlands.

3 Data Preprocessing

3.1 Handling Dataframes

We shall be first loading R with the `adult` dataframe and referring to it as `df`. With the given size of the dataframe at 32000 rows, although the compiling of `R codes` are possible in short time, we predict that the knitting of the `R Markdown` files may take a significantly longer duration (of the order of minutes per single knit). The diligent user is therefore advised to clock up the processing capabilities of the system before moving forward with compiling and knitting on `R Markdown`.

```
filename <- "adult.csv"
library(data.table)
df <- fread(filename,sep=",", header=T, na.strings=c("?"), stringsAsFactors=T)
df0 <- df #Preseving the initial dataframe as df0
```

Since we observe missing values of the form of '?' in the dataframe, we use the `na.strings=c("?")` command for reading these as `NA`. We use the `data.table` package to load data instead of the typical `read.csv()` function. Thus enables the question marks to be converted to proper `NA` in `df`.

The data is summarized using the `summary` function as seen below. This results in the summary list such as min, max, mean etc. for numerical variables and the frequency in class for categorical variables.

```
summary(df0)
```

```
##       age                        wrk             fwt
##  Min.   :17.00   Private         :22696   Min.   :  12285
##  1st Qu.:28.00   Self-emp-not-inc: 2541   1st Qu.: 117827
##  Median :37.00   Local-gov       : 2093   Median : 178356
##  Mean   :38.58   State-gov       : 1298   Mean   : 189778
##  3rd Qu.:48.00   Self-emp-inc    : 1116   3rd Qu.: 237051
##  Max.   :90.00   (Other)         :  981   Max.   :1484705
##                  NA's            : 1836
##            edu             enum                            mar
##  HS-grad     :10501   Min.   : 1.00   Divorced                 : 4443
##  Some-college: 7291   1st Qu.: 9.00   Married-AF-spouse        :   23
##  Bachelors   : 5355   Median :10.00   Married-civ-spouse       :14976
##  Masters     : 1723   Mean   :10.08   Married-spouse-absent    :  418
##  Assoc-voc   : 1382   3rd Qu.:12.00   Never-married            :10683
##  11th        : 1175   Max.   :16.00   Separated                : 1025
##  (Other)     : 5134                   Widowed                  :  993
```

```
##               occ                    rel                          race
##  Prof-specialty : 4140   Husband       :13193   Amer-Indian-Eskimo:  311
##  Craft-repair   : 4099   Not-in-family : 8305   Asian-Pac-Islander: 1039
##  Exec-managerial: 4066   Other-relative:  981   Black              : 3124
##  Adm-clerical   : 3770   Own-child     : 5068   Other              :  271
##  Sales          : 3650   Unmarried     : 3446   White              :27816
##  (Other)        :10993   Wife          : 1568
##  NA's           : 1843
##      sex             gain            loss            hours
##  Female:10771   Min.   :    0   Min.   :   0.0   Min.   : 1.00
##  Male  :21790   1st Qu.:    0   1st Qu.:   0.0   1st Qu.:40.00
##                 Median :    0   Median :   0.0   Median :40.00
##                 Mean   : 1078   Mean   :  87.3   Mean   :40.44
##                 3rd Qu.:    0   3rd Qu.:   0.0   3rd Qu.:45.00
##                 Max.   :99999   Max.   :4356.0   Max.   :99.00
##
##            cntry         salary
##  United-States:29170   <=50K:24720
##  Mexico       :  643   >50K : 7841
##  Philippines  :  198
##  Germany      :  137
##  Canada       :  121
##  (Other)      : 1709
##  NA's         :  583
```

3.2 Classification of missing data

There are two types of missing data:

- **MCAR: *missing completely at random.*** There is no way to determine what the value should have been, here. Random imputation methods as well as any method such as predictive mean matching, logistic regression imputation etc. can for MCAR. This is the desirable scenario in case of missing data. There must be a strong theoretical foundation for the assumption that the missing values of the given dataframe are in fact MCAR.
- **MNAR: *missing not at random.*** Missing not at random data is a more serious issue and in this case it might be wise to check the data gathering process in order to understand the reason for the lack of information. For instance, in a survey, if most participants did not answer a certain question, one must ponder as to why they chose not answer. Perhaps, it might be the case that the question was unclear, or that the question was tricky and people felt uncomfortable answering. IN another instance, if there were a block of missing values in the data generated by the Lunar Orbiter, it might the case of signal loss or non-functioning of a particular sensor, etc. If MNAR is indeed the case then missing values should not be imputed with constants or at random because these values will not reflect the actual values. We have no way of statistically guessing plausible values.
- **MAR: *missing at random.*** Under MAR, values are missing because of their relationship to another column in a dataset. There is an important difference between MCAR and MAR. While MCAR is preferable, MAR is again something that must be dealt with theoretical introspection. MAR implies a conditional relationship between the missing value and other variables. The missing value itself isn't known and cannot be known, but it is missing because of another observed value. For example, in a survey with optional questions, participants might have omitted one question by favoring another.

3.3 Summary of Missing Values

From our summary on `df`, we obtained missing values in the following columns:

- 1836 missing values in `wrk`
- 1843 missing values in `occ`
- 583 missing values in `cntry`

Even under the assumption that the missing values in `df` is MCAR, too much missing data can be a problem too. Usually a safe maximum threshold is 10% of the total for large datasets. If missing data for a certain feature or sample is more than 10% then the feature must be left out. We therefore check for features (columns) and samples (rows) where more than 10% of the data is missing using the following function.

```r
pMiss <- function(x){sum(is.na(x))/length(x)*100}
apply(df0,2,pMiss) #apply(df,1,pMiss)
```

```
##      age      wrk      fwt      edu     enum      mar      occ      rel
## 0.000000 5.638647 0.000000 0.000000 0.000000 0.000000 5.660146 0.000000
##     race      sex     gain     loss    hours    cntry   salary
## 0.000000 0.000000 0.000000 0.000000 0.000000 1.790486 0.000000
```

We observe that of the features, `wrk` and `occ` has about 5.6%, and `country` has about 1.8% missing values. Since these are less than a 10% threshold, these values may be omitted. Nonetheless, we shall attempt various methods to visualize the missing data, prior to their treatment. We shall also attempt imputation and its possible consequences.

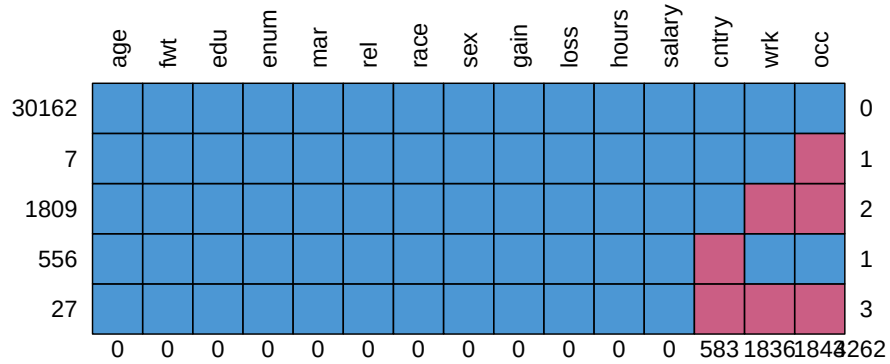
3.4 Visualizing missing values

3.4.1 Using package `mice`

The MICE in `mice package` stands for ***multivariate imputation using chained equations***. According to the R-Documentation website [1] the mice package implements a method to deal with missing data. The package creates multiple imputations (replacement values) for multivariate missing data. The method is based on Fully Conditional Specification, where each incomplete variable is imputed by a separate model. The MICE algorithm imputes mixes of continuous, binary, unordered categorical and ordered categorical data. In addition, MICE also imputes continuous two-level data, and maintain consistency between imputations by means of passive imputation.

We shall first visualize the missing data using the `pattern` function.

```r
md.pattern(df0, plot = TRUE, rotate.names = TRUE)
```




```
##       age fwt edu enum mar rel race sex gain loss hours salary cntry  wrk  occ
```

[1] `https://www.rdocumentation.org/packages/mice/versions/3.8.0/topics/mice-` Official R Documentation.

```
## 30162    1   1   1    1   1   1    1   1    1    1    1    1    1    1    1
## 7        1   1   1    1   1   1    1   1    1    1    1    1    1    1    0
## 1809     1   1   1    1   1   1    1   1    1    1    1    1    1    0    0
## 556      1   1   1    1   1   1    1   1    1    1    1    1    0    1    1
## 27       1   1   1    1   1   1    1   1    1    1    1    1    0    0    0
##          0   0   0    0   0   0    0   0    0    0    0    0  583 1836 1843
##
## 30162     0
## 7         1
## 1809      2
## 556       1
## 27        3
##        4262
```

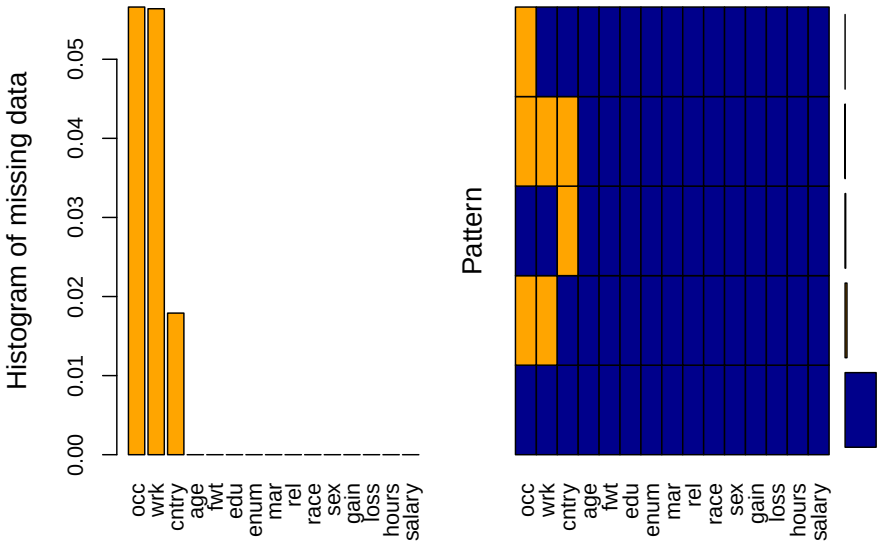
The 1's in the above output denote that the particular column has no missing values, while the 0's indicate the presence of missing values. The inferences from the above table are:

- The first row above has all 1's. This shows that there are 30162 rows in `df` that are completely filled (that is filled in all columns).
- The second row above indicates that there are only 7 rows which misses `occ` alone. These people have refused to state their occupation. Contrarily, they have stated their working class.
- The third row is our major chunk of 1809 missing values in both `occ` and `wrk` columns. These are the people who have refused to state their working class and occupation.
- The fourth row above indicates that there are 556 missing values in the `cntry` column.
- Finally, in the fifth row above, there are 27 rows in `df` which misses out all three - `occ, wrk, cntry`.

Now we shall try out another plot to visualize missing data using the `VIM package` on R, using the `aggr()` function, for this plots the histogram of all missing data divided over various columns in the x axis and also allows better graphing customization options.

```
aggr_plot <- aggr(df0, col=c('darkblue','orange'),
                  numbers=TRUE, sortVars=TRUE, labels=names(df0),
                  cex.axis=0.9,gap=3, ylab=c("Histogram of missing data","Pattern"))
```

```
## Warning in plot.aggr(res, ...): not enough horizontal space to display
## frequencies
```

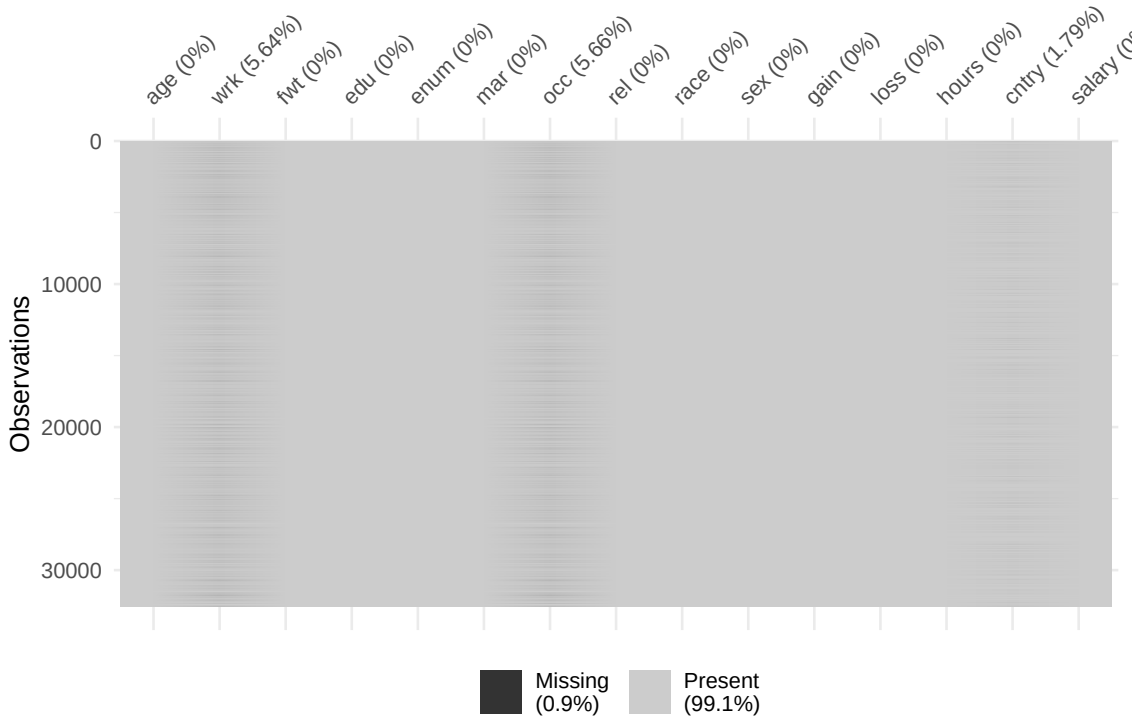



```
##
##  Variables sorted by number of missings:
##  Variable      Count
##       occ 0.05660146
##       wrk 0.05638647
##     cntry 0.01790486
##       age 0.00000000
##       fwt 0.00000000
##       edu 0.00000000
##      enum 0.00000000
##       mar 0.00000000
##       rel 0.00000000
##      race 0.00000000
##       sex 0.00000000
##      gain 0.00000000
##      loss 0.00000000
##     hours 0.00000000
##    salary 0.00000000
```

3.5 Exploration of missing values using `naniar`

To visualize where in the data is missing.

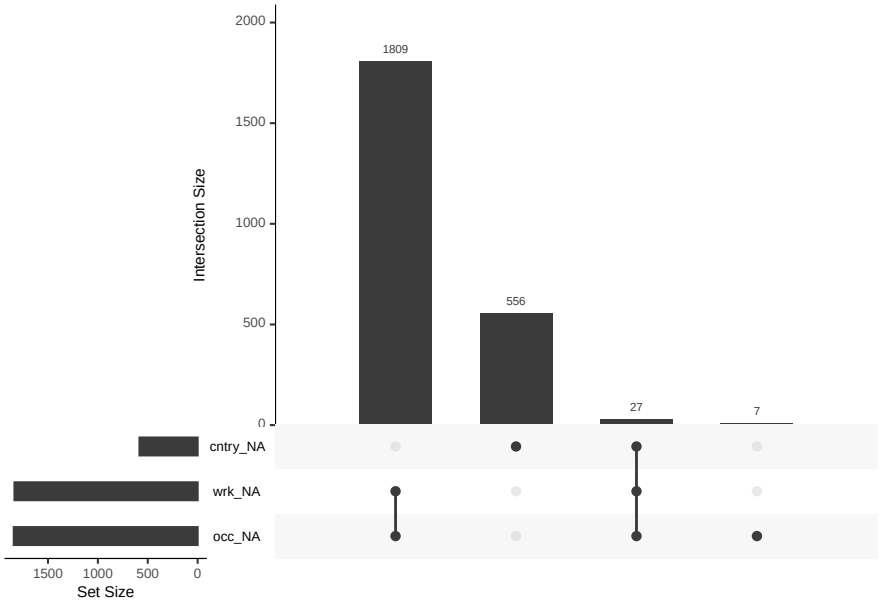
```
library(naniar)
vis_miss(df0)
```



3.5.1 Exploring patterns with `UpSetR`

An ***upset plot*** from the `UpSetR package` can be used to visualise the patterns of *missingness*, or rather the combinations of missingness across cases. To see combinations of missingness and intersections of missingness amongst variables, we use the :

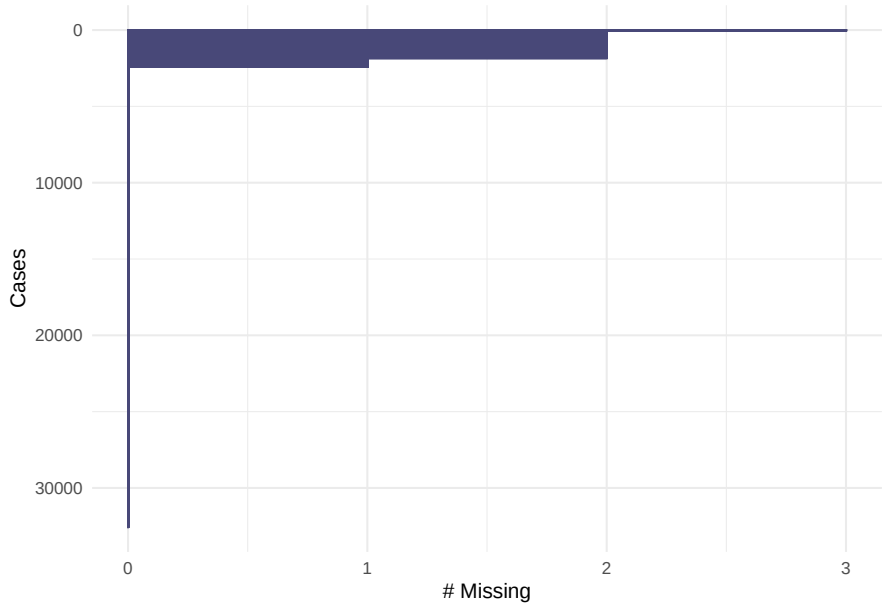
```
gg_miss_upset(df0, nsets = n_var_miss(df0))
```



3.5.2 Missing cases

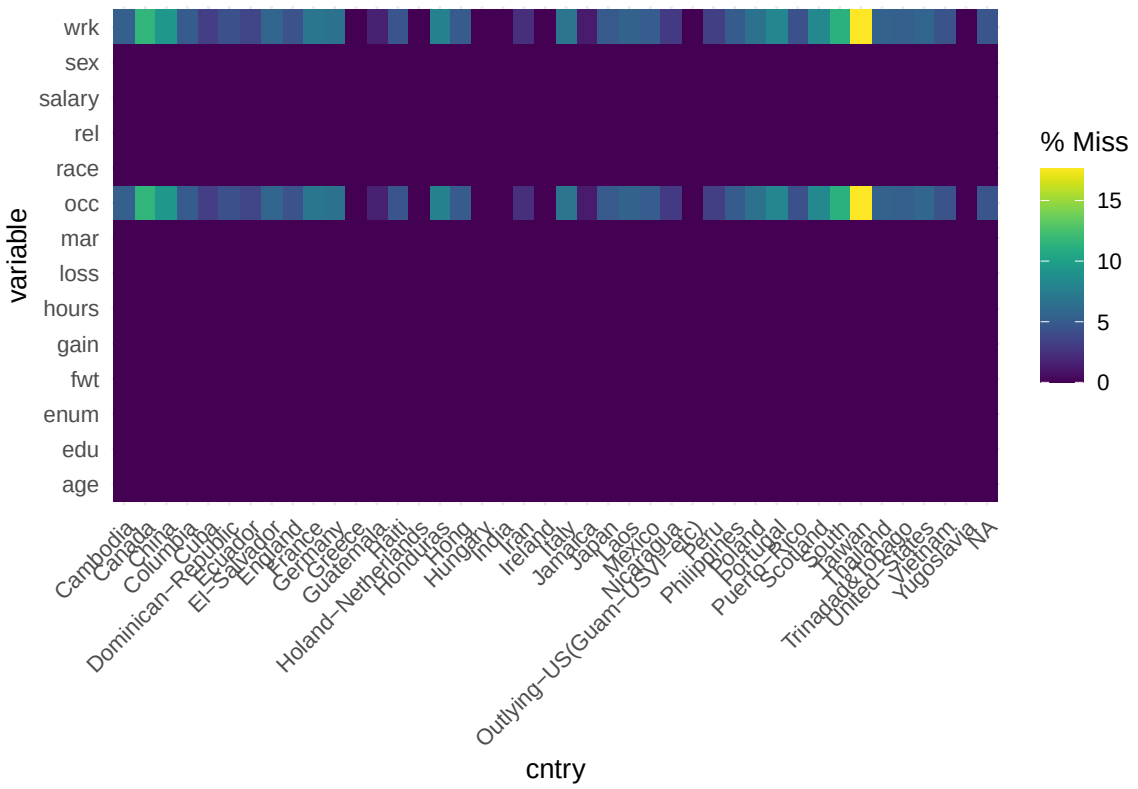
The number of observations is very valuable to any data analysis. The number of cases lost as a result of omission should therefore be proved to be insignificant before proceeding further. We use the and functions.

```
gg_miss_case(df0)
```



3.5.3 Missing cases by country

```
gg_miss_fct(x = df0, fct = cntry)
```



3.6 Removing missing values

We therefore see that the missing values are concentrated in the variables `occ`, `wrk`, and `cntry`. Of these we observe that occupation and workclass have 1809 common missing values. This makes sense because, those individuals who have refused to provide details of their occupation have also done so when it came to the question of workclass. This is due to the similar nature of these two variable. Hence our assumption of MCAR is not valid for these missing values. In fact these are of the type MAR, missing values in occupation often alludes to missing values in workclass.

Secondly, from the countries' plot we see that the distribution of missing values is not uniform and hence not of the MCAR type. Moreover predictive mean or median imputations do not make sense, for these are factors whose levels are not ordered.

Further, given the case that less than 10% of the values are only missing, we omit these values, rather than imputing.

```
# After the missing values section
df[df == "?"] <- NA
df <- na.omit(df)
```

4 Data Exploration and Preparation

4.1 Converting categorical variables to factors

```
df$wrk <- factor(df$wrk)
df$occ <- factor(df$occ)
df$cntry<- factor(df$cntry)
```

```
df$salary <- factor(df$salary, labels=c("<=50k", ">50k"))
#Checking the levels
levels(df$salary)
```

```
## [1] "<=50k" ">50k"
```

```
# Changing the factors of Workforce
levels(df$wrk)
```

```
## [1] "Federal-gov"      "Local-gov"        "Private"          "Self-emp-inc"
## [5] "Self-emp-not-inc" "State-gov"        "Without-pay"
```

```
levels(df$wrk)[c(1,2,6)] <- "Government"
levels(df$wrk)[c(3,4)] <- "Self_Employed"
levels(df$wrk)
```

```
## [1] "Government"    "Private"       "Self_Employed" "Without-pay"
```

```
# Changing the factors of Marital Status
levels(df$mar)
```

```
## [1] "Divorced"              "Married-AF-spouse"     "Married-civ-spouse"
## [4] "Married-spouse-absent" "Never-married"         "Separated"
## [7] "Widowed"
```

```
levels(df$mar)[c(1,4,5,6,7)]<- "Single"
levels(df$mar)
```

```
## [1] "Single"             "Married-AF-spouse"  "Married-civ-spouse"
```

```
levels(df$mar)[c(2,3)]<-"Married"
levels(df$mar)
```

```
## [1] "Single"  "Married"
```

```
levels(df$edu)
```

```
##  [1] "10th"         "11th"         "12th"         "1st-4th"      "5th-6th"
##  [6] "7th-8th"      "9th"          "Assoc-acdm"   "Assoc-voc"    "Bachelors"
## [11] "Doctorate"    "HS-grad"      "Masters"      "Preschool"    "Prof-school"
## [16] "Some-college"
```

```r
# Reordering the education levels for data set
df$edu <- factor(df$edu,levels(df$edu)[c(14, 4:7, 1:3, 12, 15, 8:9, 16, 10, 13, 11)])
levels(df$edu)
```

```
##  [1] "Preschool"    "1st-4th"      "5th-6th"      "7th-8th"      "9th"
##  [6] "10th"         "11th"         "12th"         "HS-grad"      "Prof-school"
## [11] "Assoc-acdm"   "Assoc-voc"    "Some-college" "Bachelors"    "Masters"
## [16] "Doctorate"
```

```r
# Combining Husband and Wife in Relationship to form a single variable, "Spouse"
levels(df$rel)
```

```
## [1] "Husband"        "Not-in-family"  "Other-relative" "Own-child"
## [5] "Unmarried"      "Wife"
```

```r
levels(df$rel)[c(1,6)]<-"Spouse"
levels(df$rel)
```

```
## [1] "Spouse"         "Not-in-family"  "Other-relative" "Own-child"
## [5] "Unmarried"
```

```r
# Combining all native country types apart from US
levels(df$cntry)
```

```
##  [1] "Cambodia"                  "Canada"
##  [3] "China"                     "Columbia"
##  [5] "Cuba"                      "Dominican-Republic"
##  [7] "Ecuador"                   "El-Salvador"
##  [9] "England"                   "France"
## [11] "Germany"                   "Greece"
## [13] "Guatemala"                 "Haiti"
## [15] "Holand-Netherlands"        "Honduras"
## [17] "Hong"                      "Hungary"
## [19] "India"                     "Iran"
## [21] "Ireland"                   "Italy"
## [23] "Jamaica"                   "Japan"
## [25] "Laos"                      "Mexico"
## [27] "Nicaragua"                 "Outlying-US(Guam-USVI-etc)"
## [29] "Peru"                      "Philippines"
## [31] "Poland"                    "Portugal"
## [33] "Puerto-Rico"               "Scotland"
## [35] "South"                     "Taiwan"
## [37] "Thailand"                  "Trinadad&Tobago"
## [39] "United-States"             "Vietnam"
## [41] "Yugoslavia"
```

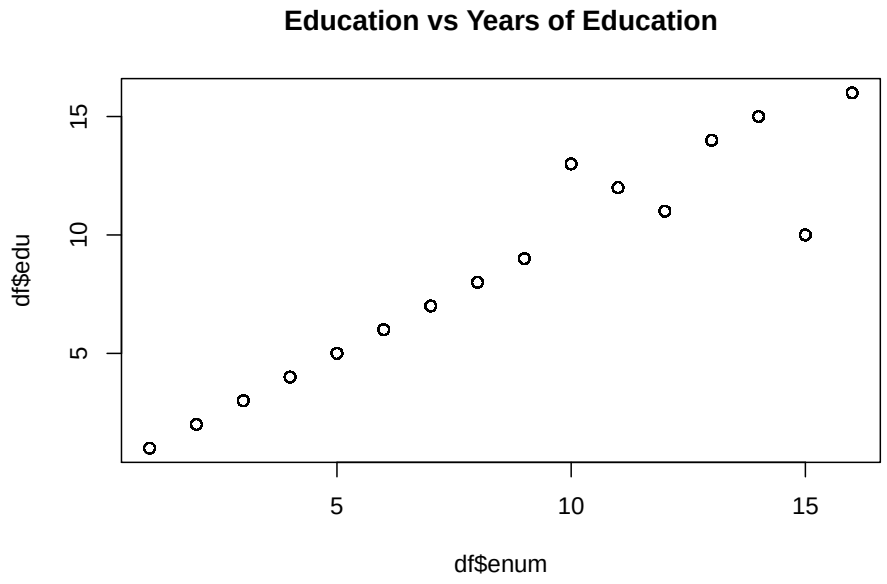
```r
levels(df$cntry)[c(1:38, 40,41)]<-"Non-US"
levels(df$cntry)
```

```
## [1] "Non-US"        "United-States"
```

```r
str(df)
```

```
## Classes 'data.table' and 'data.frame':   30162 obs. of  15 variables:
##  $ age   : int  39 50 38 53 28 37 49 52 31 42 ...
##  $ wrk   : Factor w/ 4 levels "Government","Private",..: 1 3 2 2 2 2 2 3 2 2 ...
##  $ fwt   : int  77516 83311 215646 234721 338409 284582 160187 209642 45781 159449 ...
##  $ edu   : Factor w/ 16 levels "Preschool","1st-4th",..: 14 14 9 7 14 15 5 9 15 14 ...
##  $ enum  : int  13 13 9 7 13 14 5 9 14 13 ...
##  $ mar   : Factor w/ 2 levels "Single","Married": 1 2 1 2 2 2 1 2 1 2 ...
##  $ occ   : Factor w/ 14 levels "Adm-clerical",..: 1 4 6 6 10 4 8 4 10 4 ...
##  $ rel   : Factor w/ 5 levels "Spouse","Not-in-family",..: 2 1 2 1 1 1 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 ...
##  $ gain  : int  2174 0 0 0 0 0 0 0 14084 5178 ...
##  $ loss  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ hours : int  40 13 40 40 40 40 16 45 50 40 ...
##  $ cntry : Factor w/ 2 levels "Non-US","United-States": 2 2 2 2 1 2 1 2 2 2 ...
##  $ salary: Factor w/ 2 levels "<=50k",">50k": 1 1 1 1 1 1 1 2 2 2 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```r
# Plotting education against education.num
plot(df$enum, df$edu, main="Education vs Years of Education")
```




```r
# abline(lm(edu ~ enum, data = df),col="blue")
# Linear relationship between the two, hence, we can drop education (16 factors).
```

4.2 Feature Creation

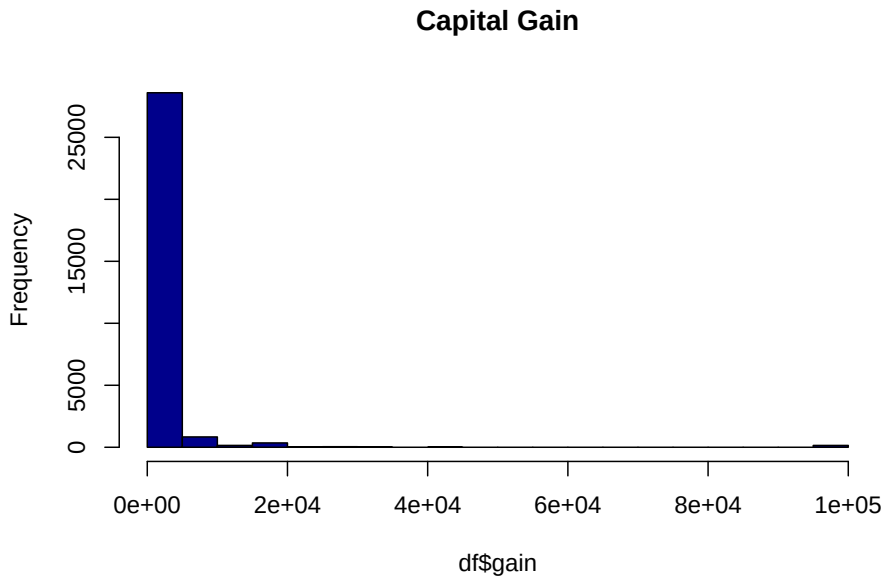
Since capital gain and capital loss are similar financial variables, we create a new feature, termed as net capital, which will account for both the variables.

```
# Net capital = Capital gain - capital loss
df$net.capital <- df$gain-df$loss
```

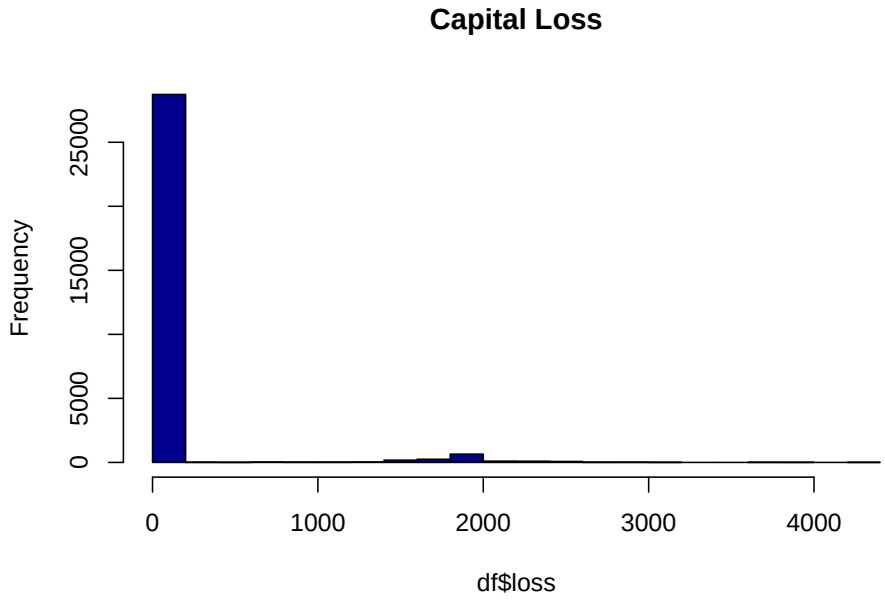
4.2.1 Univariate Analysis

Histograms are used to perform analysis when only one single variable is under consideration. The distributions of capital gain and loss, and hence that of the newly created feature net capital are plotted below.

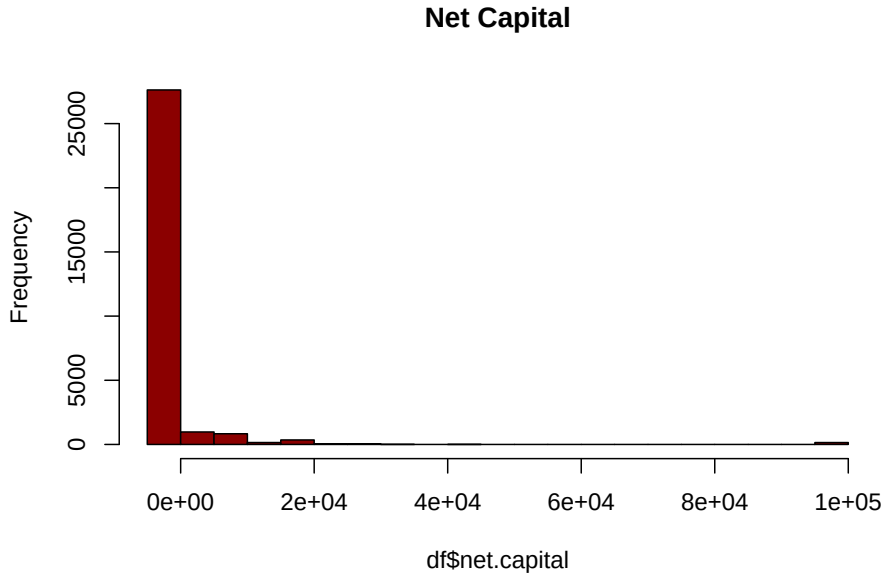
```
hist(df$gain, col="darkblue", main="Capital Gain")
```




```
hist(df$loss, col="darkblue", main="Capital Loss")
```




```r
hist(df$net.capital, col="darkred", main="Net Capital")
```



5 Data Visualization

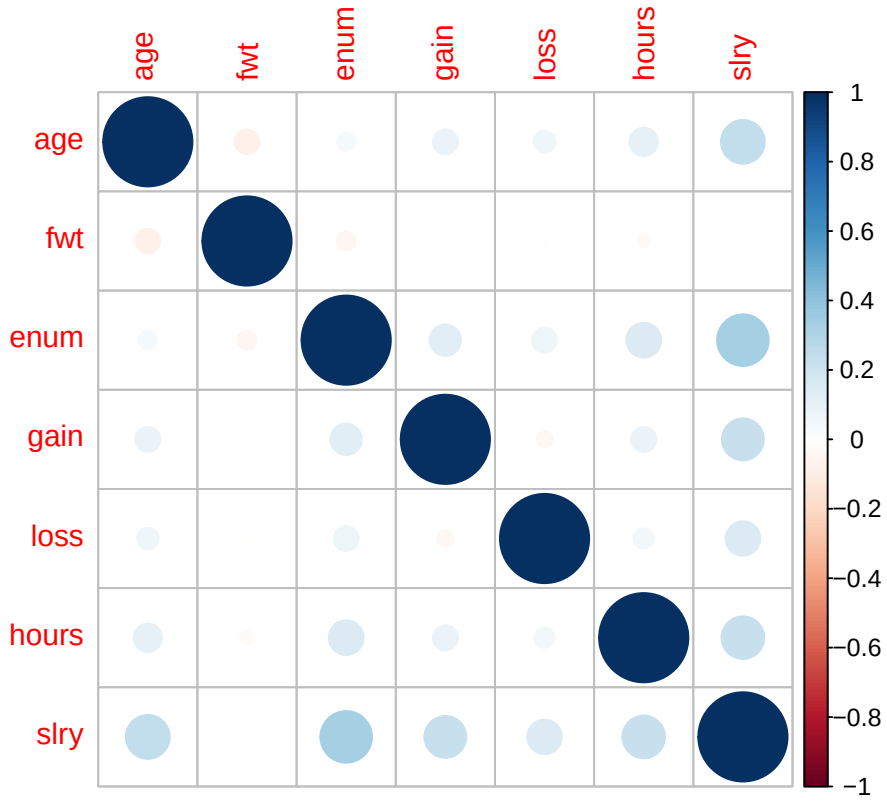
5.1 Correlation amongst numeric variables

```
slry <- as.numeric(df$salary)-1
numvar <- c(1, 3, 5, 11:13)
data <- cbind(df[, ..numvar],slry)
```

```
#Correlation plot amongst all numerical variables
library(corpcor)
library(corrplot)
```

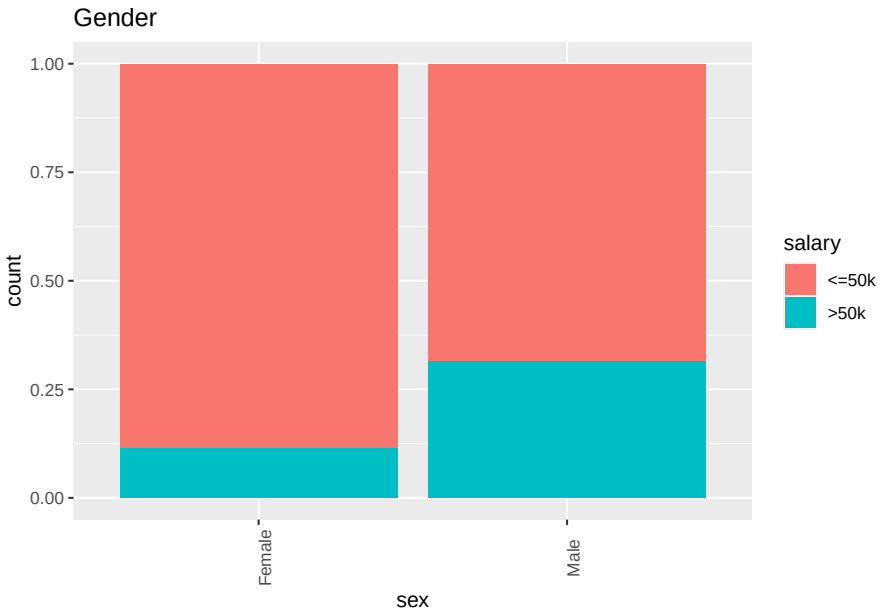
```
## corrplot 0.84 loaded
```

```
corrplot(cor(data))
```

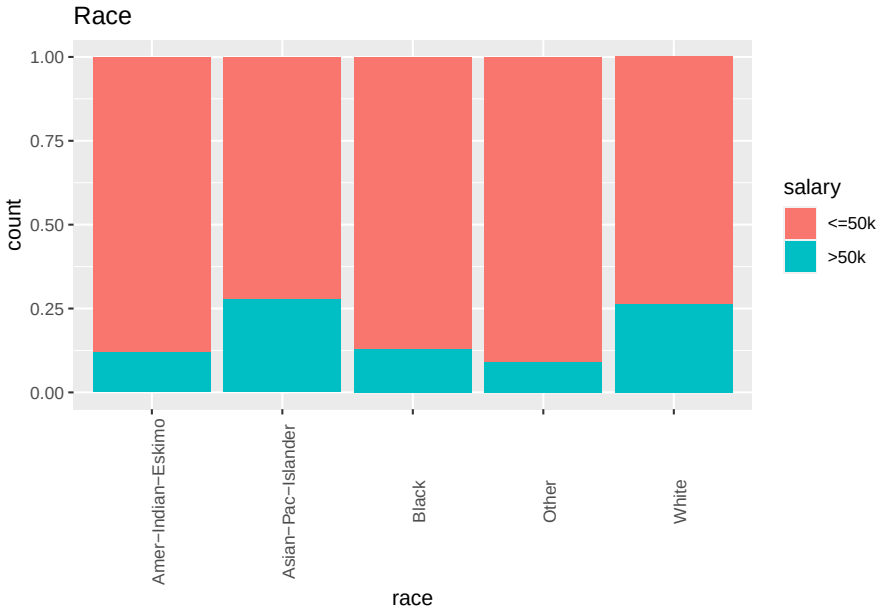


5.2 Visualising categorical variables as per salary

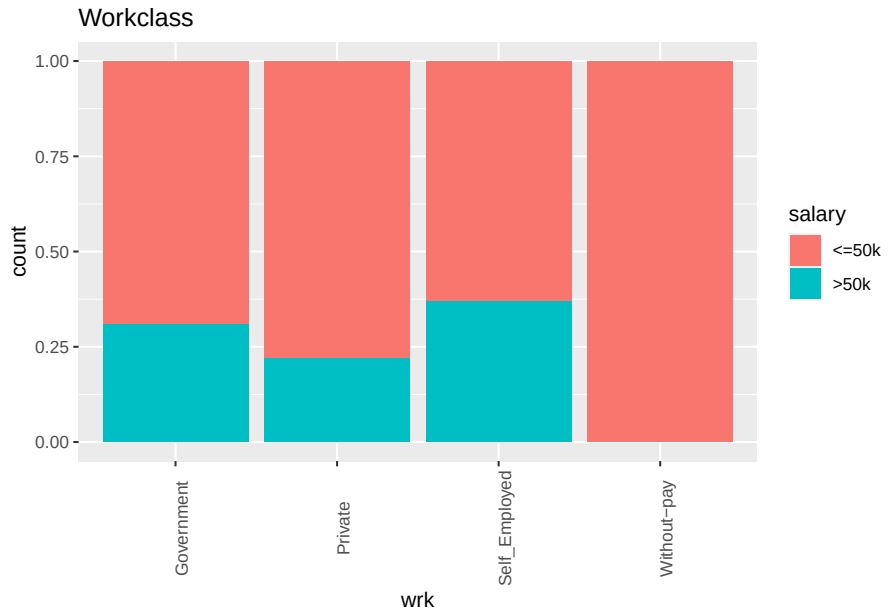
```
library(ggplot2)
# Sex
ggplot(df, aes(x = sex, fill = salary)) + geom_bar(position="fill") +
  theme(axis.text.x = element_text(angle = 90)) + ggtitle("Gender")
```



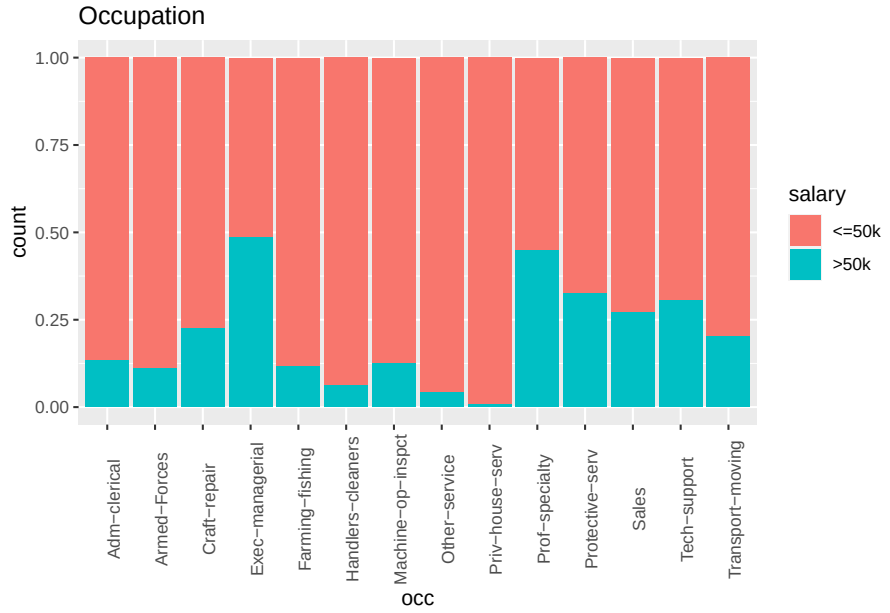

```
# Race
ggplot(df, aes(x = race, fill = salary)) + geom_bar(position="fill") +
  theme(axis.text.x = element_text(angle = 90)) + ggtitle("Race")
```



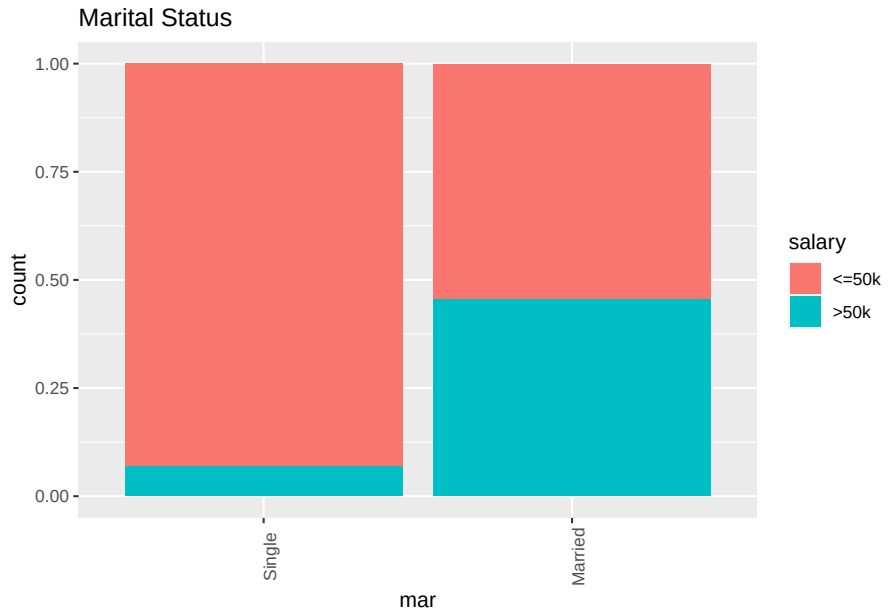

```
#Workclass
ggplot(df, aes(x = wrk, fill = salary)) + geom_bar(position="fill") +
  theme(axis.text.x = element_text(angle = 90)) + ggtitle("Workclass")
```



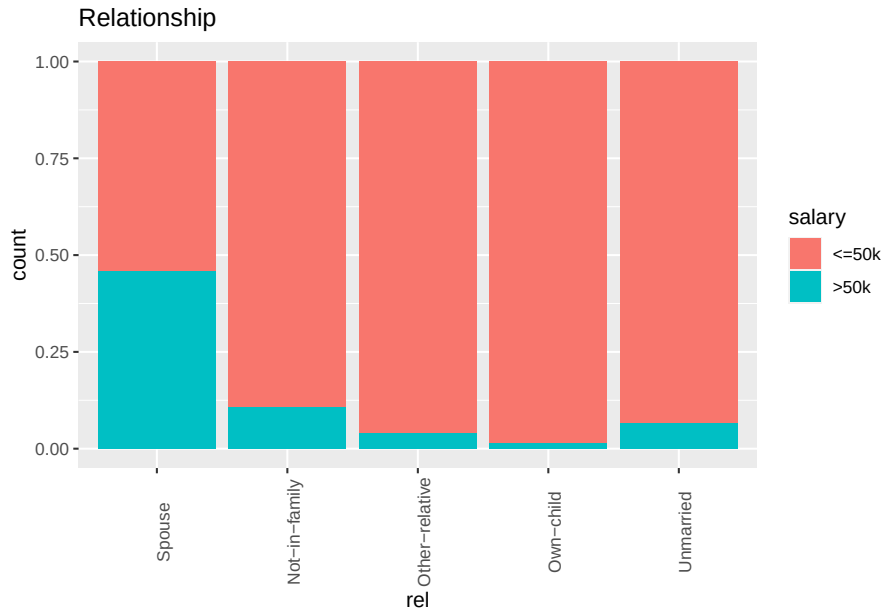

```
# Occupation
ggplot(df, aes(x = occ, fill = salary)) + geom_bar(position="fill") +
  theme(axis.text.x = element_text(angle = 90)) + ggtitle("Occupation")
```



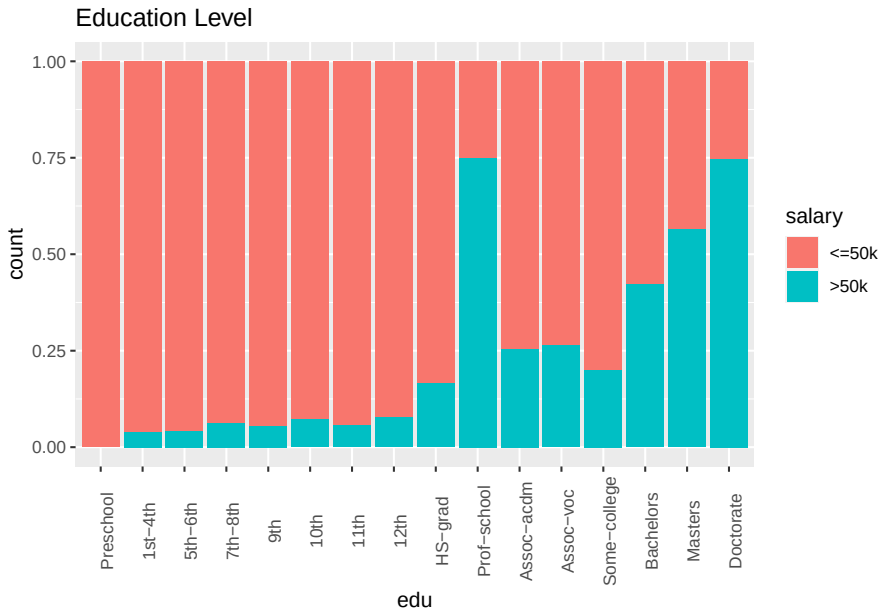

```
# Marital Status
ggplot(df, aes(x = mar, fill = salary)) + geom_bar(position="fill") +
  theme(axis.text.x = element_text(angle = 90)) + ggtitle("Marital Status")
```



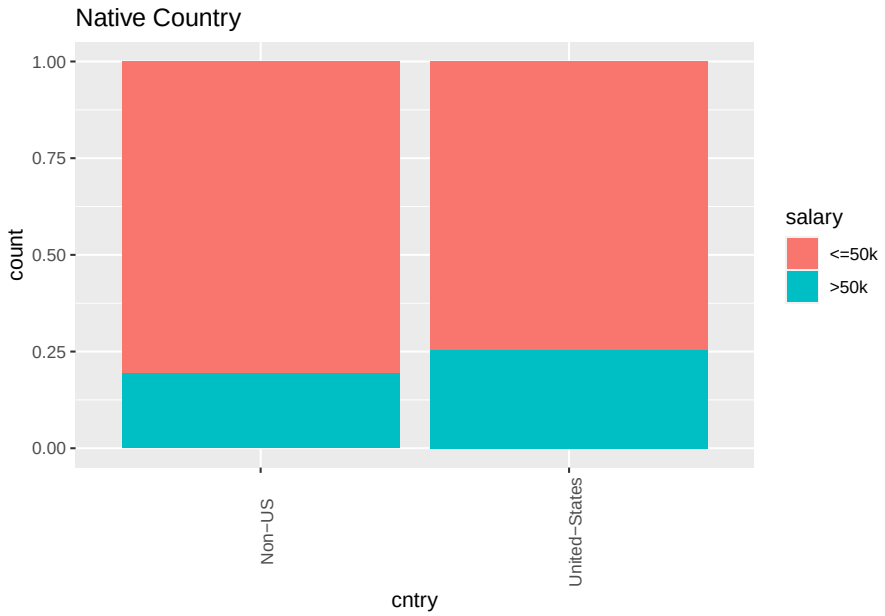

```
# Relationship
ggplot(df, aes(x = rel, fill = salary)) + geom_bar(position="fill") +
  theme(axis.text.x = element_text(angle = 90)) + ggtitle("Relationship")
```




```
# Education
ggplot(df, aes(x = edu, fill = salary)) + geom_bar(position="fill") +
  theme(axis.text.x = element_text(angle = 90)) + ggtitle("Education Level")
```




```
# Native Country
ggplot(df, aes(x = cntry, fill = salary)) + geom_bar(position="fill")+
  theme(axis.text.x = element_text(angle = 90)) + ggtitle("Native Country")
```



6 Building the final model

6.1 Variable Selection

As we have already seen, education and number of years of education are linearly related. Thus, it makes sense to remove `edu` as a predictor variable from the model. Owing to the similar nature of occupation and workclass, we remove, `occ` from our model. Moreover, our feature creation yielded net capital, thereby making capital gain and loss redundant. We shall promptly remove `gain` and `loss` from our model.

```
# Removing fwt, edu, occ, capital gain, capital loss (to form net capital)
df2 <- df[, -c(3, 4, 7,11,12)]
df2 <- df2[,c(10, 1:9,11)]
```

6.2 Training and Test datasets

```
# Creating the test and training sets
set.seed(1) # To run random numbers
train <- sample(1:length(df2$salary), length(df2$salary)*0.75)
df2.train <- df2[train,]
df2.test <- df2[-train,]
```

7 Logistic Regression

Multiple logistic regression concerns with predicting a binary response Y using multiple predictors X. Our model is given by:

$$\log\left(\frac{p(>50)}{(1-p(>50))}\right) = \beta_0 + \beta_1(age) + \beta_2(wrk) + \beta_3(enu) + \beta_4(mar) + \beta_5(rel)$$
$$+\beta_6(race) + \beta_7(sex) + \beta_8(hours) + \beta_9(cntry) + \beta_{10}(net.capital)$$

Here, p is the probability that the chosen individual has a salary greater than \$50k per year. Using logistic regression our objective is to determine $\hat{p}(>50)$.

$$\hat{p}(>50) = \frac{e^{\hat{\beta_0}+\hat{\beta_1}age+\hat{\beta_2}wrk\cdots+\hat{\beta_{10}}net.capital}}{1+e^{\hat{\beta_0}+\hat{\beta_1}age+\hat{\beta_2}wrk\cdots+\hat{\beta_{10}}net.capital}}$$

```
glmfit <- glm(df2.train$salary~., data=df2.train, family=binomial)
summary(glmfit)
```

```
##
## Call:
## glm(formula = df2.train$salary ~ ., family = binomial, data = df2.train)
##
## Deviance Residuals:
```

```
##     Min       1Q   Median       3Q      Max
## -4.7587  -0.5753  -0.2411   0.0000   3.8356
##
## Coefficients:
##                           Estimate Std. Error z value Pr(>|z|)
## (Intercept)            -9.705e+00  4.425e-01 -21.935  < 2e-16 ***
## age                     2.987e-02  1.764e-03  16.938  < 2e-16 ***
## wrkPrivate             -1.967e-02  5.530e-02  -0.356 0.722034
## wrkSelf_Employed       -3.100e-01  7.250e-02  -4.276 1.90e-05 ***
## wrkWithout-pay         -1.263e+01  1.269e+02  -0.099 0.920766
## enum                    3.670e-01  9.169e-03  40.026  < 2e-16 ***
## marMarried              2.227e+00  3.204e-01   6.951 3.63e-12 ***
## relNot-in-family        4.632e-02  3.240e-01   0.143 0.886326
## relOther-relative      -1.170e+00  2.877e-01  -4.066 4.79e-05 ***
## relOwn-child           -1.291e+00  3.276e-01  -3.941 8.10e-05 ***
## relUnmarried           -1.712e-01  3.347e-01  -0.512 0.608978
## raceAsian-Pac-Islander  5.859e-01  2.699e-01   2.171 0.029939 *
## raceBlack               2.858e-01  2.490e-01   1.148 0.251071
## raceOther              -1.543e-01  4.050e-01  -0.381 0.703179
## raceWhite               4.848e-01  2.372e-01   2.044 0.040961 *
## sexMale                 7.715e-02  5.683e-02   1.358 0.174595
## hours                   3.000e-02  1.834e-03  16.353  < 2e-16 ***
## cntryUnited-States      3.050e-01  8.812e-02   3.461 0.000538 ***
## net.capital             2.354e-04  1.002e-05  23.498  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 25526  on 22620  degrees of freedom
## Residual deviance: 16047  on 22602  degrees of freedom
## AIC: 16085
##
## Number of Fisher Scoring iterations: 12
```

From the above, our model reduces to

$$\log\left(\frac{p(>50)}{(1-p(>50))}\right) = -9.705 + 0.029(age) - 0.19(private) - 0.367(SelfEmployed)$$
$$-0.12(Withoutpay) + 3.67(enu) + 2.27(mar) + 0.046(Notinfamily)$$
$$-1.17(Otherrelative) - 1.29(ownchild) - 0.172(Unmarried) + 5.859(Asian)$$
$$+2.858(Black) - 0.1543(Other) + 4.848(White) + 0.77(Male)$$
$$+0.03(hours) + 3.05(US) + 0.023(Net.Capital)$$

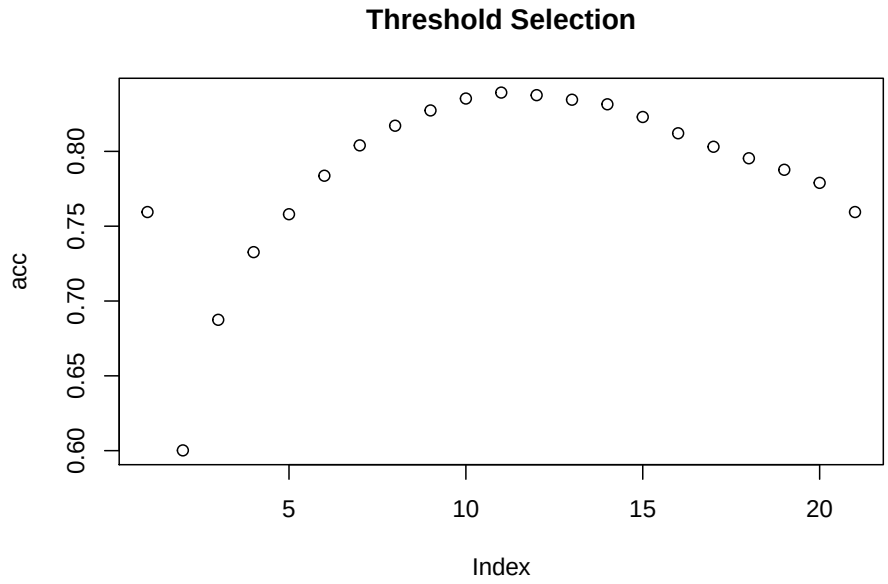
```
contrasts(df2$salary)
```

```
##        >50k
## <=50k     0
## >50k      1
```

7.1 Threshold Selection

Thresholds are the levels at which the user defines the categorisation for the response variable, say, for our model if we define the threshold as 0.5, this means that if the probability turns out to be beyond 0.5, the variable will take the value >50k. Similarly, if the threshold was 0.4, if the probability comes out to be beyond 0.4, the variable would take value >50k. The threshold plots in our codes are to decide which level of threshold maximises accuracy for our data.

```
threshold <- seq(0, 1, 0.05) # Various thresholds (Different thresholds
# give varying accuracy levels)
acc <- rep(0, length(threshold))
for (i in 1:length(threshold)) {
  glmprobs <- predict(glmfit, newdata=df2.test, type = "response")
  glmpred <- rep("<=50k", length(df2.test$salary))
  glmpred[glmprobs > threshold[i]] = ">50k"
  glmtable <- table(glmpred, df2.test$salary)
  acc[i] <- sum(diag(glmtable))/sum(glmtable)
}
plot(acc, main="Threshold Selection")
```




```
# To see where the plot maximises accuracy
threshold[which.max(acc)]
```

```
## [1] 0.5
```

```
# accuracy
log.acc <- acc[which.max(acc)]
# To convert the accuracy into log of accuracy to get the probability
log.acc
```

```
## [1] 0.8392786
```

```
#Confusion matrix for logistic regression
library(caret)
```

```
## Loading required package: lattice
```

```
logpred <- predict(glmfit, newdata = df2.test, type = "response")
estimatedResponses=ifelse(logpred>0.5,">50k","<=50k")
trueResponses=ifelse(df2.test$salary==">50k",">50k","<=50k")
log.table <-table(estimatedResponses,trueResponses)
log.table
```

```
##                   trueResponses
## estimatedResponses <=50k >50k
##              <=50k  5309  794
##              >50k    418 1020
```

From the contrasts matrix, we know that our positive case is one with `salary>50k`. Thus we define our True and False cases accordingly. The accuracy from the confusion matrix again comes out to $(5309+1020)/7541 = 83.92\%$.

```
# prob that income >50k
log.inc <- (log.table[2,2])/7541
log.inc
```

```
## [1] 0.1352606
```

Therefore, as per the logistic regression model, the probability that one's income is greater than \$50k is 13.5%.

7.2 Baseline Model

A baseline result is the simplest possible prediction model. We set a criteria for setting our baseline model and comparing other models against this level. For our model, the baseline model is where it predicts everyone will make below \$50,000 as the baseline, 0.752735.

```
#Baseline = Everyone makes below 50k
baseline <- acc[length(threshold)]
baseline
```

```
## [1] 0.7594483
```

7.3 Interpretations

- Age: The model showed a positive relationship with age i.e. an increase in age indicated an increase in the log of odds for the respondent earning more than $50k.
- WorkClass: The model showed a negative relationship in case of private workclass, self employed as well as without pay. Thus, the log of odds for the respondent earning more than $50k is higher in case of Government than in any other working class.
- Education Years: As expected, with an increase in the number of education years there's an increase in the log of odds of the respondent earning more than $50k/year.
- Marital Status: A married respondent is more likely to earn more than $50k as compared to a single person.
- Relationship: Own child, other relatives, and own children are less likely to earn more than $50k/ year as compared to someone 'not in family' or a spouse.
- Race: An Asian, Black or White is more likely to earn more than $50k/year as compared to other races.
- Sex: Males are more likely to earn more than $50k/year as compared to females.
- Number of working hours: An increase in the number of working hours will increase the likelihood of earning more than $50k/year.
- Native Country: A US citizen is more likely to earn $50k/year as compared to citizens with other countries of origin.
- Net capital: A respondent is more likely to earn $50k/year if he has a positive net capital flow.
- The model fits with an accuracy of 83.92%.

8 Decision Trees

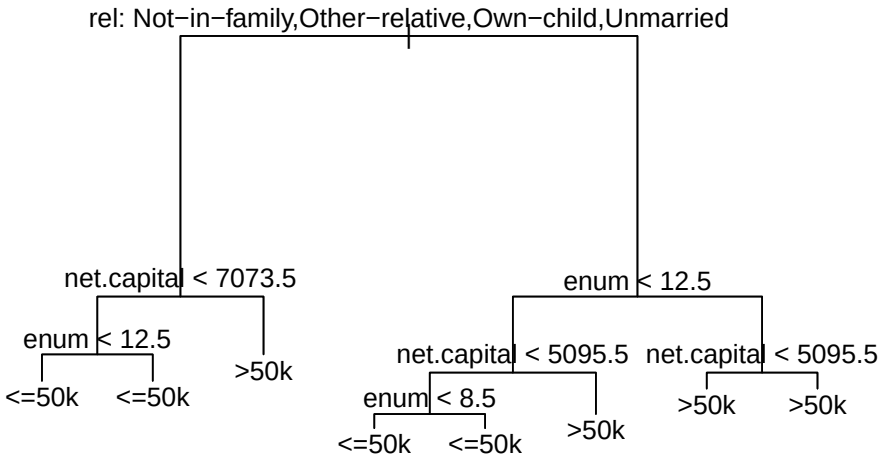
8.1 Introduction

Decision trees involve stratifying or segmenting the predictor space into a number of simple regions. In order to make a prediction for a given observation, we typically use the mean or the mode of the training observations in the region to which it belongs. Since the set of splitting rules used to segment the predictor space can be summarized in a tree, these types of approaches are known as decision tree methods. Tree-based methods are simple and useful for interpretation. However, they typically are not competitive with the best supervised learning approaches. Approaches like bagging, random forests and boosting involve producing multiple trees which are then combined to yield a single consensus prediction. In our model, we begin with the simple decision tree,and then move on to the better approach of boosting. For a classification tree, we predict that each observation belongs to the most commonly occurring class of training observations in the region to which it belongs.

```
library(tree)
tree.income <- tree(df2.train$salary~., data=df2.train)
tree.pred <- predict(tree.income, newdata=df2.test, type="class")
summary(tree.income)
```

```
##
## Classification tree:
## tree(formula = df2.train$salary ~ ., data = df2.train)
## Variables actually used in tree construction:
## [1] "rel"         "net.capital" "enum"
## Number of terminal nodes:  8
## Residual mean deviance:  0.723 = 16350 / 22610
## Misclassification error rate: 0.1607 = 3636 / 22621
```

```
plot(tree.income)
text(tree.income, pretty=0)
```



Since we plan to assign an observation in a given region to the most commonly occurring

error rate class of training observations in that region, the classification error rate is simply the fraction of the training observations in that region that do not belong to the most common class:

```
# Confusion Matrix
# Tree classification comes out to be according to Relationship,Net Capital,
# Number of education years, Age,
tree.table <- table(tree.pred, df2.test$salary)
tree.table
```

```
##
## tree.pred <=50k >50k
##     <=50k  5455  886
##     >50k    272  928
```

```
tree.acc <- mean(tree.pred == df2.test$salary)
tree.acc
```

```
## [1] 0.8464395
```

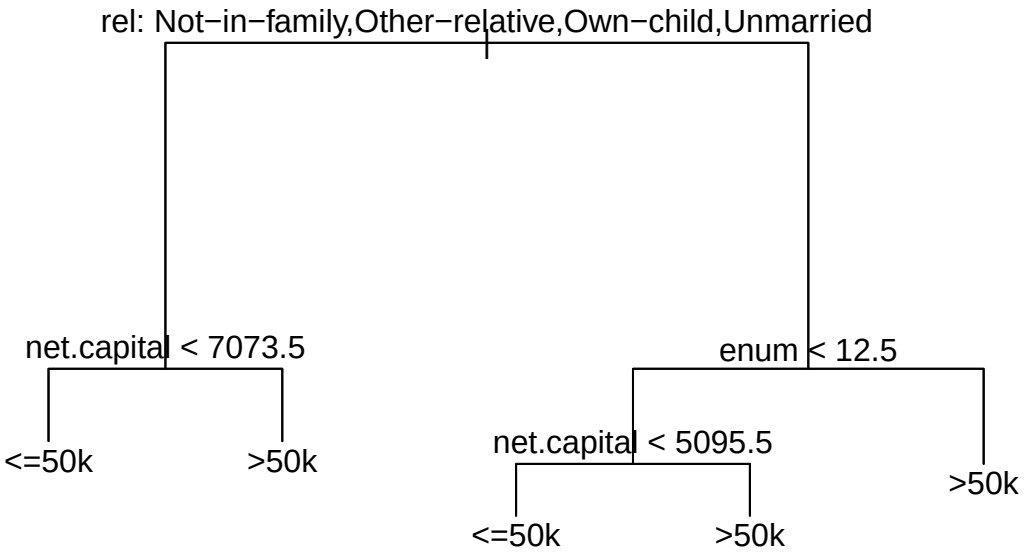
8.2 Pruning

```
# To check the desired level of pruning
cv.income = cv.tree(tree.income, FUN = prune.misclass)
cv.income
```

```
## $size
## [1] 8 5 4 3 1
##
## $dev
## [1] 3636 3636 3835 4193 5694
##
## $k
## [1]  -Inf   0.0 199.0 358.0 750.5
##
## $method
## [1] "misclass"
##
## attr(,"class")
## [1] "prune"         "tree.sequence"
```

Thus, we see that pruning the tree is redundant, as it does not improve the classification error rate. However, we can check at 5 nodes in order to verify the residual mean deviance.

```
prune.income=prune.misclass(tree.income,best=5)
plot(prune.income )
text(prune.income,pretty=0)
```




```
tree.pred1=predict(prune.income,df2.test,type="class")
table(tree.pred1 ,df2.test$salary)
```

```
##
## tree.pred1 <=50k >50k
##      <=50k  5455  886
##      >50k    272  928
```

```
tree.acc1<- mean(tree.pred1==df2.test$salary)
summary(prune.income)
```

```
##
## Classification tree:
## snip.tree(tree = tree.income, nodes = c(4L, 7L, 12L))
## Variables actually used in tree construction:
## [1] "rel"         "net.capital" "enum"
## Number of terminal nodes:  5
## Residual mean deviance:  0.7697 = 17410 / 22620
## Misclassification error rate: 0.1607 = 3636 / 22621
```

```
tree.acc1
```

```
## [1] 0.8464395
```

We observe that after pruning, the residual mean deviance actually increases while the classification accuracy stays the same. Hence, we conform to our original classification.

```
# prob that income >50k
tree.inc<- (tree.table[2,2])/7541
tree.inc
```

```
## [1] 0.1230606
```

Therefore, as per decision trees model, the probability that one's income is greater than $50k is 12.3%.

8.3 Interpretation

- The model bifurcated the data on the basis of the variable 'Relationship' (`rel`), where the categories, 'Not in Family', 'Other Relative', 'Own Child', and 'Unmarried' were clubbed into the first category and 'Spouse' was the second category.
- The first category was further bifurcated on the basis of `net.capital` less than $7073.5.
- Respondents with a net capital income of less than $7073.5 were further divided on the basis of the number of education years (`enum`) and age (`age`).
- Respondents with more than a `net.capital` income of $7073.5 are likely to have a final income of more than $50k/year.
- The second category i.e, 'Spouse', was also further categorised on the basis of `net.capital`, `enum`.
- The model had 8 terminal nodes i.e. the data can be finally concluded in 8 kinds of respondents. Eg. Spouse with a net capital income of more than $5095.5 is likely to have a salary of more than $50k,etc.
- The model fits with an accuracy of 84.64%.

9 Boosting

Boosting is a method that substantially improves the predictive performance of trees. Other methods include bagging, random forests etc. Each of these approaches involves producing multiple trees which are then combined to yield a single consensus prediction. Boosting particularly is an approach for improving the predictions resulting from a decision tree. Boosting does not involve bootstrap sampling, instead each tree is fit on a modified version of the original data set. It allows one to transform weak predictors into stronger ones and also focuses on minimizing bias.

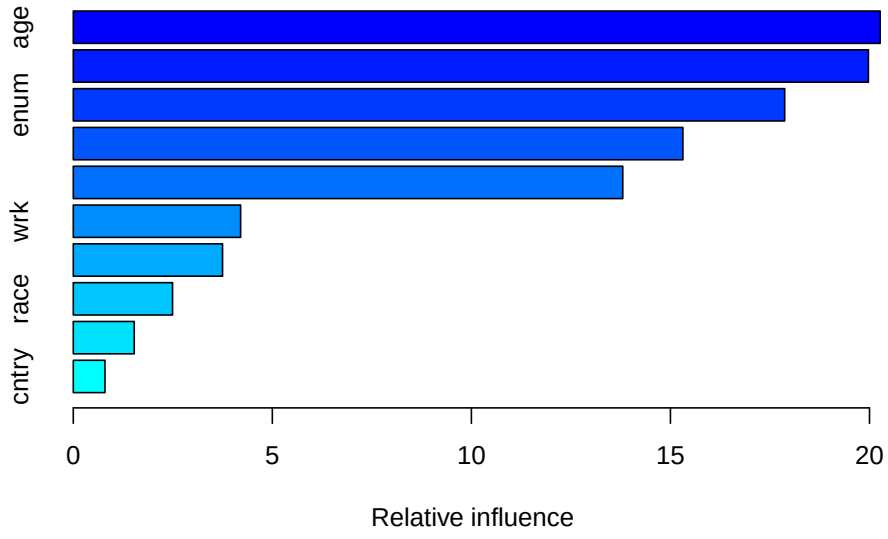
We use gradient boosting model function in R in order to find out the accuracy. It employs three tuning parameters - One, the number of trees, B; two, the shrinkage parameter λ, and three, the number dp of splits in each tree that is interaction depth y. The number of trees to build the boosting model is set to 5000 here. Thus in the `gbm()` function, `n.trees=5000`. Since this is both a regression problem and a binary classification problem we use `distribution = bernoulli` in the function.

```
df3<-df2 # The dataset is transferred to df3.
df3.train<- df2.train
df3.test<-df2.test
```

```
library(gbm)
```

```
## Loaded gbm 2.1.5
```

```
boost.fit <- gbm(as.numeric(salary)-1~.,data=df3.train, distribution =
                   "bernoulli", n.trees = 5000 , interaction.depth = 4)
summary(boost.fit)
```



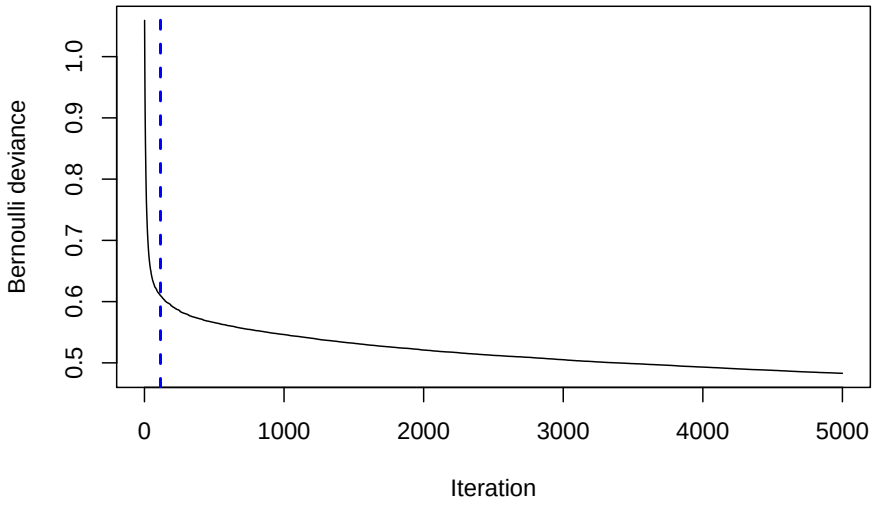

```
##                   var    rel.inf
## age               age 20.2675894
```

```
## rel                 rel 19.9731372
## enum               enum 17.8697771
## net.capital net.capital 15.3132983
## hours             hours 13.8022551
## wrk                 wrk  4.2033486
## mar                 mar  3.7496041
## race               race  2.4938263
## sex                 sex  1.5313603
## cntry             cntry  0.7958035
```

The following graph indicates the optimum number of trees based on the respective technique used. The blue dotted line points the optimum number of iterations. The computational constraints in performing boosting has led to the low number of iterations.

```
gbm.perf(boost.fit)
```

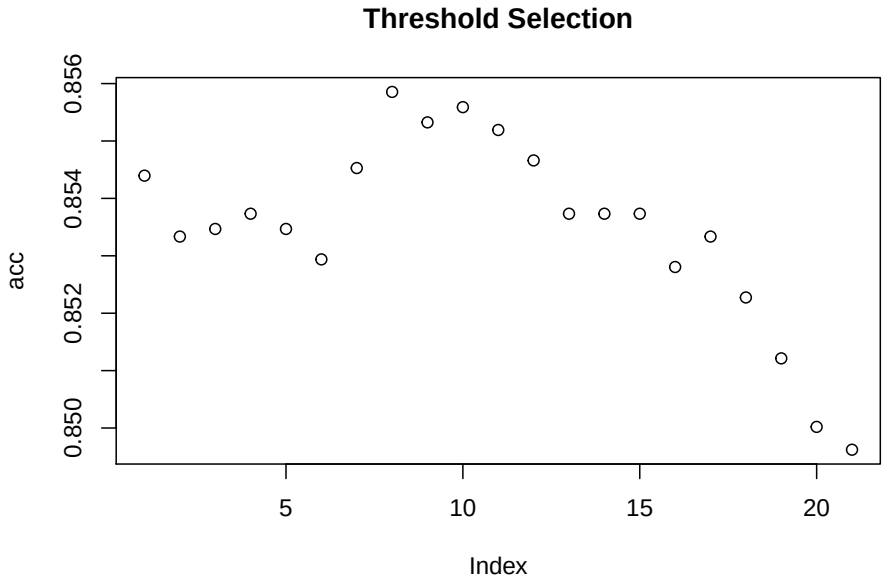
```
## OOB generally underestimates the optimal number of iterations although predictive per
```




```
## [1] 115
## attr(,"smoother")
## Call:
## loess(formula = object$oobag.improve ~ x, enp.target = min(max(4,
##     length(x)/10), 50))
##
## Number of Observations: 5000
## Equivalent Number of Parameters: 39.99
## Residual Standard Error: 0.0006121
```

9.1 Threshold selection

```
df3.train$salary<-as.numeric(df3.train$salary)+1
threshold <- seq(0, 1, 0.05)
acc <- rep(0, length(threshold))
for (i in 1:length(threshold)) {
  boost.probs <- predict(boost.fit, df3.test, n.trees = 5000)
  boost.pred <- ifelse(boost.probs > threshold[i], 1, 0)
  acc[i] <- mean(boost.pred == as.numeric(df3.test$salary)-1)
}
plot(acc, main="Threshold Selection")
```




```
threshold[which.max(acc)]
```

```
## [1] 0.35
```

```
boost.table <- table(boost.pred, df2.test$salary)
boost.table
```

```
##
## boost.pred <=50k >50k
##          0  5592  999
##          1   135  815
```

```
boost.acc <- acc[which.max(acc)]
boost.acc
```

```
## [1] 0.8558547
```

```
# prob that income >50k
boost.inc<- (boost.table[2,2])/7541
boost.inc
```

```
## [1] 0.1080759
```

Therefore, as per gradient boosting model, the probability that one's income is greater than $50k is 10.7%.

9.2 Interpretation

The most important variable with the highest influence was found to be age and relationship while the least important variable turned out to be sex and country in classifing salary level. In this case, we can say that salary increases with relationship and decreases with age. We also find that a total of 6418 observations were classifier correctly, with an accuracy of 85.68%

10 Linear Discriminant Analysis

10.1 Classification Method

Linear Discriminant Analysis, or LDA for short, uses Bayes' theorem in predicting the class of the response variable. Consider the Y variable to be classified into K classes and assume there are p predictor or explanatory variables, such that each X could themselves be divided into classes or be a numerical variable. If π_k is the overall prior probability that a randomly chosen observation comes from the k^{th} class. If $f_k(x) = Pr(X = x|Y = k)$ is the probability density function that the randomly chosen observation (row) comes from the k^{th} class of Y, then we can use Bayes' theorem to state that:

$$Pr(Y = k|X = x) = \frac{\pi_k f_k(x)}{\sum_{l=1}^{K} \pi_l f_l(x)}$$

This simply means that, given we know the prior probabilities through from the fraction of a variable that falls under a particular class and given we are aware of the probability density function for the univariate distribution of values in the x variable, then we can use Bayes' theorem to predict the class of the Y variable conditioned to the given X variable. Once we have computed the posterior probability, $Pr(Y = k|X = x)$, then we can classify each observation x into that class of Y which has the highest prior probability. If p is the number of discriminants, d the squared Mahalanobis distance from point x to the class centroid and S is the covariance matrix, the probability density function of x given that $Y = k$ is

$$f_k(x) = Pr(X = x|Y = k) = \frac{e^{-d/2}}{(2\pi)^{p/2}\sqrt{|S|}}$$

10.2 LDA applied to `df`

In the current dataframe, we have multiple predictors - 10 to be precise. Hence $p = 10$. The Y variable is split into only two classes,, which are `>50k` and \mo{<=50k). Hence $K = 2$. Further, we see that from the univariate analysis, the distributions of the predictors are known. Thus LDA is very suitable for classification in this problem, owning to the high number of discrete variables.

```
print(length(df2.train$salary))
```

```
## [1] 22621
```

```
summary(df2.train$salary)
```

```
## <=50k  >50k
## 16927  5694
```

From the above code we can calculate the prior probabilities of the 2 classes in Y, the `salary` variable. For the class `>50k`, we have $p = 5694/22621 = 0.2517$ and thus for class

\mo{<=50k), we have $1 - p = 16927/22621 = 0.7482$

We now initialize the `MASS package` in `R` which has the `lda` function to carry this out. Further we use the `predict` function to predict income in the test data.

```
library(MASS)
ldafit <- lda(df2.train$salary~., data=df2.train)
ldapred <- predict(ldafit, newdata=df2.test)
ldafit
```

```
## Call:
## lda(df2.train$salary ~ ., data = df2.train)
##
## Prior probabilities of groups:
##    <=50k     >50k
## 0.748287 0.251713
##
## Group means:
##            age wrkPrivate wrkSelf_Employed wrkWithout-pay      enum marMarried
## <=50k 36.63620  0.7675312       0.09972234   0.0007089266  9.634962  0.3404029
## >50k  44.02125  0.6543730       0.16982789   0.0000000000 11.621707  0.8524763
##       relNot-in-family relOther-relative relOwn-child relUnmarried
## <=50k        0.3009393       0.039227270  0.193300644   0.13422343
## >50k         0.1088865       0.004390587  0.008429926   0.03020724
##       raceAsian-Pac-Islander  raceBlack   raceOther raceWhite   sexMale
## <=50k             0.02800260 0.10899746 0.009452354 0.8423229 0.6197791
## >50k              0.03424658 0.04882332 0.002634352 0.9088514 0.8489638
##          hours cntryUnited-States net.capital
## <=50k 39.37360          0.9068943    93.69628
## >50k  45.68282          0.9316825  3737.74535
##
## Coefficients of linear discriminants:
##                                  LD1
## age                       0.0177742535
## wrkPrivate               -0.0125655779
## wrkSelf_Employed         -0.1787699035
## wrkWithout-pay           -1.2027743479
## enum                      0.2285940702
## marMarried                0.5493271731
## relNot-in-family         -0.9983101656
## relOther-relative        -0.8933882387
## relOwn-child             -0.9041422150
## relUnmarried             -0.9922583033
## raceAsian-Pac-Islander    0.1269226781
```

```
## raceBlack                0.0905183053
## raceOther               -0.0627683917
## raceWhite                0.1984066453
## sexMale                  0.1003936834
## hours                    0.0150617925
## cntryUnited-States       0.0009397101
## net.capital              0.0000383816
```

```
# Confusion Matrix
lda.table <- table(ldapred$class, df2.test$salary)
lda.table
```

```
##
##         <=50k >50k
##   <=50k  5280  862
##   >50k    447  952
```

```
# Accuracy
lda.acc <- mean(ldapred$class==df2.test$salary)
lda.acc
```

```
## [1] 0.8264156
```

```
# prob that income >50k
lda.inc<- (lda.table[2,2])/7541
lda.inc
```

```
## [1] 0.1262432
```

Therefore, as per the linear discriminant analysis model, the probability that one's income is greater than \$50k is 12.6%.

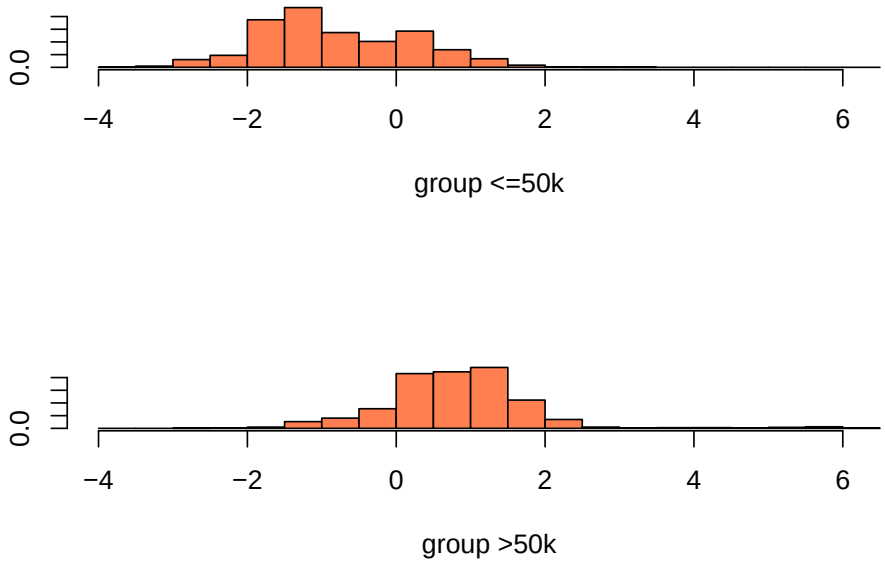
10.3 Interpretations

- Since there are only two classes in Y, there is only one linear discriminant `LD1` used to classify x into these classes.
- Although LDA can be used for dimensionality reduction it is not the case with our dataset `df2.train`.
- With only two classes, only a single score is required per observation for that is all that's needed. This is because the probability of being in one class is the complement of the probability of being in the other.
- If we multiply each value of `LDA1` by the corresponding elements of the predictor variables and sum them `(0.0177742535)age + (-0.0125655779)wrkPrivate + ... + (0.0000383816)net.capital` we get a score for each observation x.

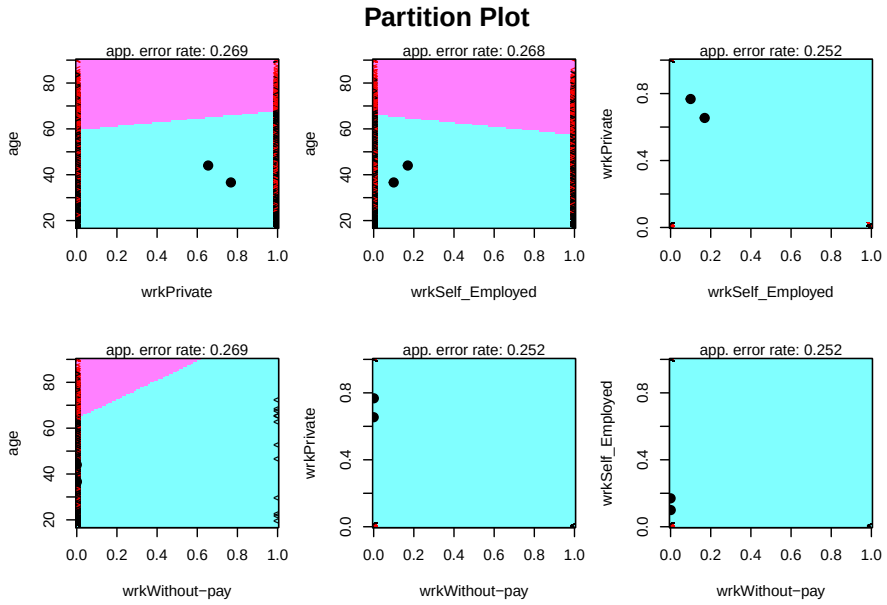
- This score along with the prior probability calculated earlier are used to compute the posterior probability of class membership.
- Classification is then made on the basis of posterior probability, with observations predicted to be in the class for which they have the highest probability.
- The confusion matrix for LDA is observed as shown below. We see that of the 7541 test values, the LDA algorithm has correctly predicted 6232 values. Note that this is the sum of 5280+952, which includes cases where salary is $<= 50k$ and has been predicted correctly and salary is $> 50k$ and is predicted so.
- Thus the accuracy of our model is estimated at 82.64% which is fairly high, but not as high as our other classifiers. LDA is less accurate than the others

We obtain a stacked histogram of the LDA values for both the classes in the predictor variables.

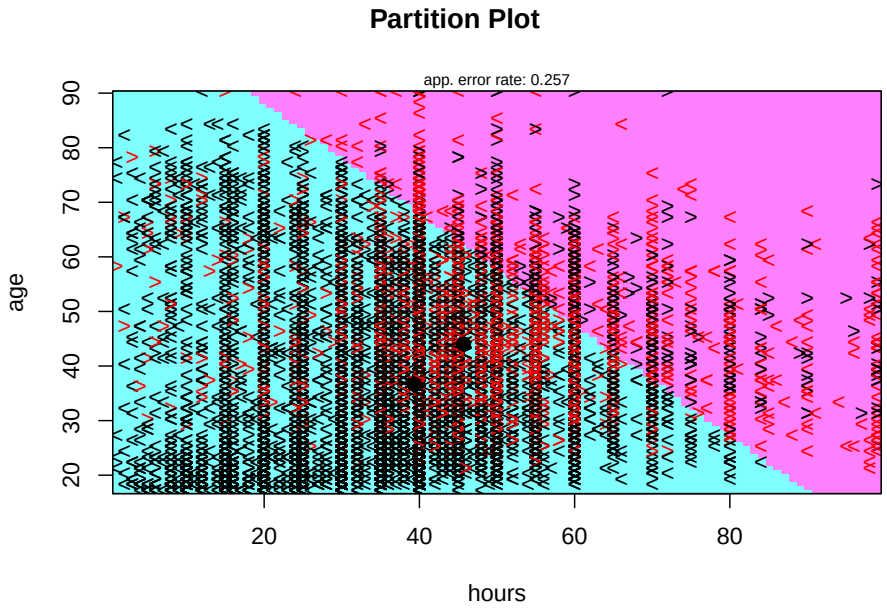
```
plot(ldafit, col="coral", main="LDA plot")
```



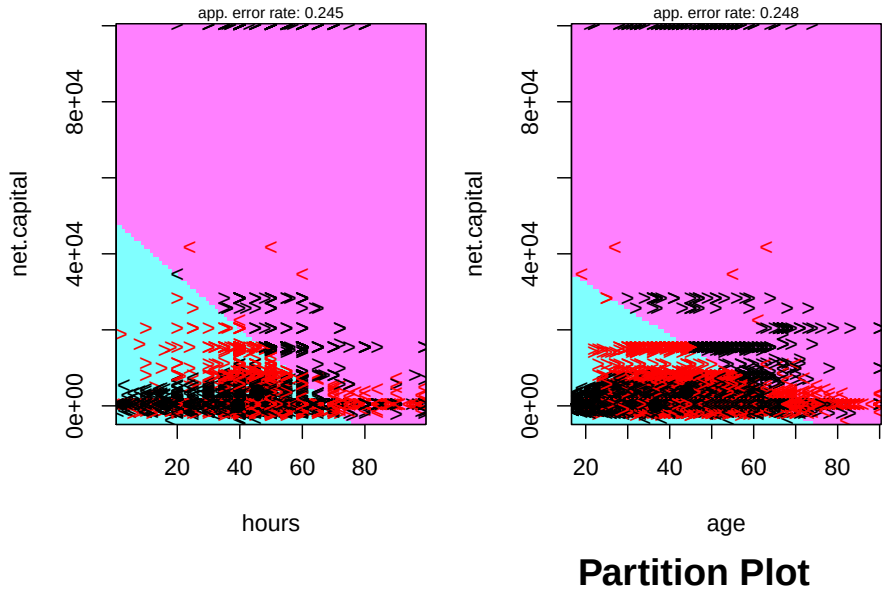

```
library(klaR)
partimat(salary ~ age+wrk, data = df2.train, method = "lda")
```

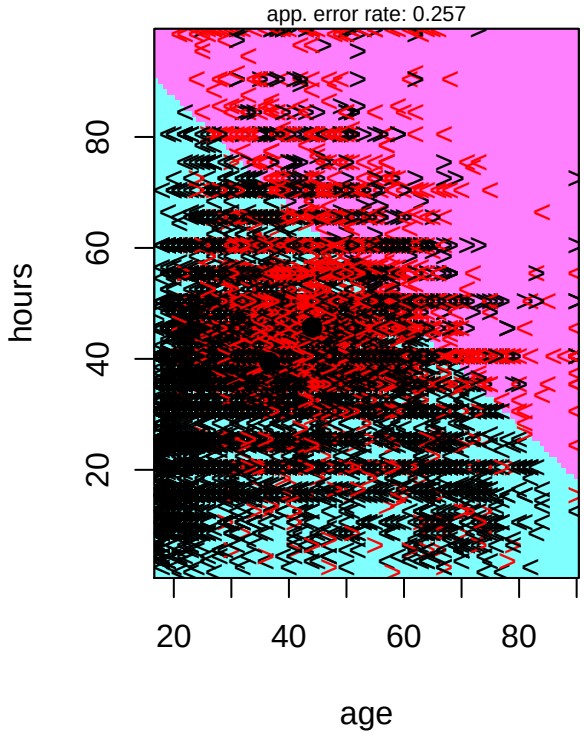



```
library(klaR)
partimat(salary ~ age+hours, data = df2.train, method = "lda")
```

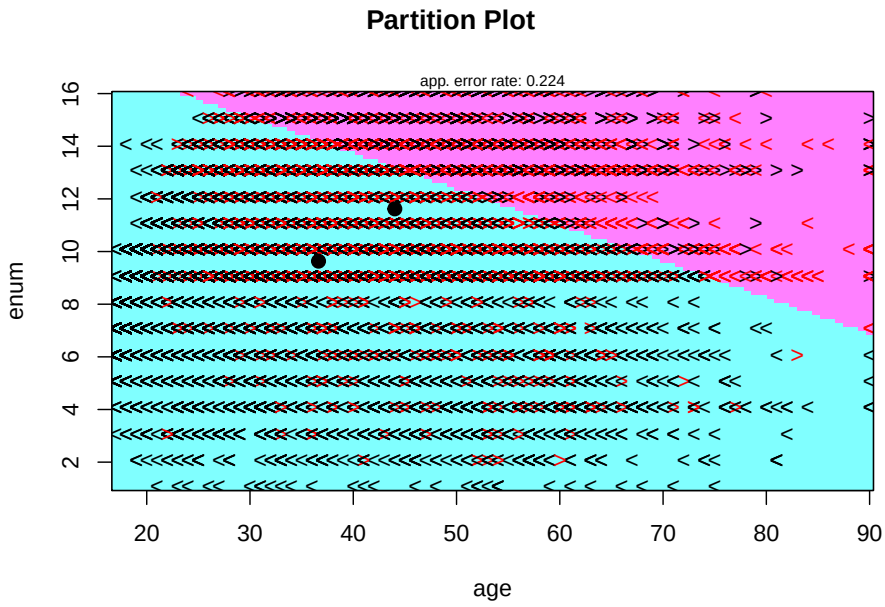



```
library(klaR)
partimat(salary ~ net.capital+hours+age, data = df2.train, method = "lda")
```

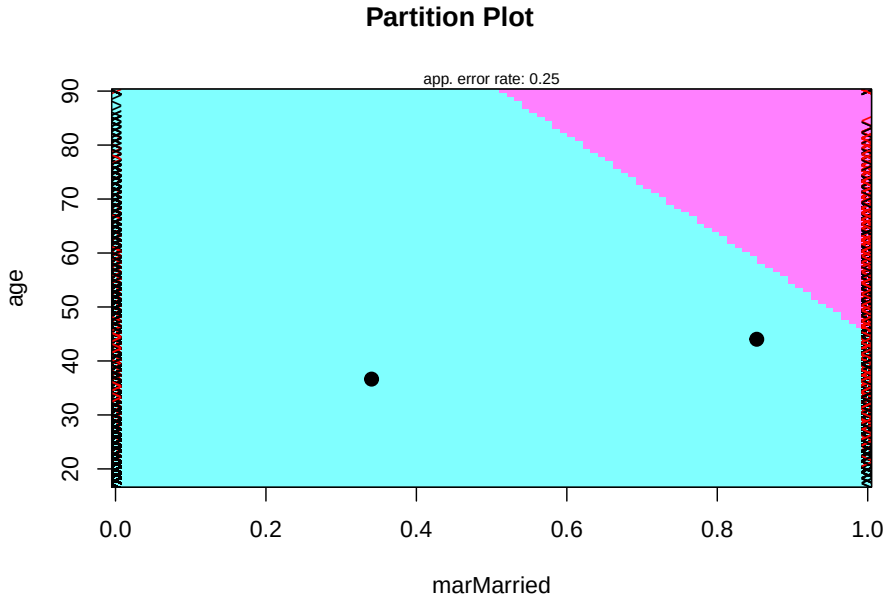




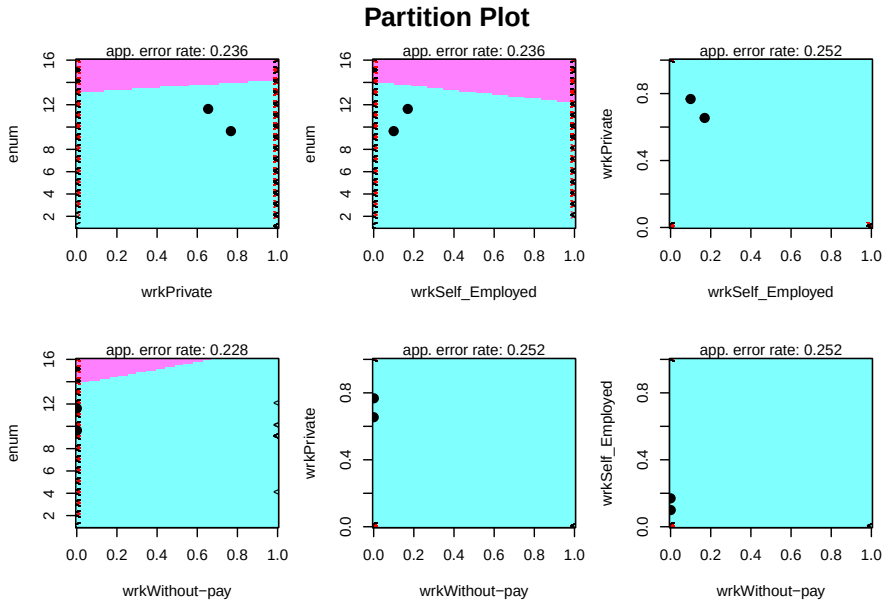

```
library(klaR)
partimat(salary ~ enum+age, data = df2.train, method = "lda")
```




```
library(klaR)
partimat(salary ~ age+mar, data = df2.train, method = "lda")
```




```
library(klaR)
partimat(salary ~ enum+wrk, data = df2.train, method = "lda")
```


Partition Plot
app. error rate: 0.236
app. error rate: 0.236
app. error rate: 0.252
app. error rate: 0.228
app. error rate: 0.252
app. error rate: 0.252
enum
wrkPrivate
wrkSelf_Employed
wrkWithout−pay

11 Support Vector Machines

Support Vector Machines, or SVM in short, is a very efficient algorithm for classification. The basic concept of SVM is the hyperplane. A hyperplane is a flat affine (that need not pass through the origin) which divides the plane into two halves.The distance of the points from the hyperplane should be maximum from both the sides which is the optimum hyperplane. We use two SVM classifiers here - one with a linear kernel and the other with radial.

11.1 Using a linear kernel

```
library(e1071)
# running the svm algorithm
classifier <- svm(formula = df2.train$salary~.,
                  data = df2.train,
                  type = 'C-classification',
                  kernel = 'linear')
print(classifier)
```

```
##
## Call:
## svm(formula = df2.train$salary ~ ., data = df2.train, type = "C-classification",
##     kernel = "linear")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  1
##
## Number of Support Vectors:  8823
```

```
# using the trained classifier to predict
svmpredict = predict(classifier, newdata = df2.test)
```

```
#Making the confusion matrix
cm = table(df2.test$salary,svmpredict)
cm
```

```
##        svmpredict
##         <=50k >50k
##   <=50k  5354  373
##   >50k    832  982
```

```
summary(classifier)
```

```
##
## Call:
## svm(formula = df2.train$salary ~ ., data = df2.train, type = "C-classification",
##     kernel = "linear")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  1
##
## Number of Support Vectors:  8823
##
##  ( 4404 4419 )
##
##
## Number of Classes:  2
##
## Levels:
##  <=50k >50k
```

11.2 Using a radial kernel

```
svm.model<- svm(df2.train$salary~., data = df2.train,kernel = "radial", cost = 1, gamma
svm.predict <- predict(svm.model, df2.test)
```

```
library(caret)
svm.table<-confusionMatrix(df2.test$salary, svm.predict)
svm.table
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction <=50k >50k
##      <=50k  5432  295
##      >50k    863  951
##
##                Accuracy : 0.8464
##                  95% CI : (0.8381, 0.8545)
##     No Information Rate : 0.8348
##     P-Value [Acc > NIR] : 0.003111
```

```
##
##                  Kappa : 0.5294
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.8629
##             Specificity : 0.7632
##          Pos Pred Value : 0.9485
##          Neg Pred Value : 0.5243
##              Prevalence : 0.8348
##          Detection Rate : 0.7203
##    Detection Prevalence : 0.7594
##       Balanced Accuracy : 0.8131
##
##        'Positive' Class : <=50k
##
```

```
# prob that income >50k
svm.inc<- (svm.table$table[2,2])/7541
svm.inc
```

```
## [1] 0.1261106
```

Therefore, as per the support vector machine model, the probability that one's income is greater than $50k is 12.6%.

```
svm.acc <- mean(svm.predict == df2.test$salary)
svm.acc
```

```
## [1] 0.8464395
```

```
summary(svm.model)
```

```
##
## Call:
## svm(formula = df2.train$salary ~ ., data = df2.train, kernel = "radial",
##     cost = 1, gamma = 0.1)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1
##
## Number of Support Vectors:  8130
```

```
##
##  ( 4007 4123 )
##
##
## Number of Classes:  2
##
## Levels:
##  <=50k >50k
```

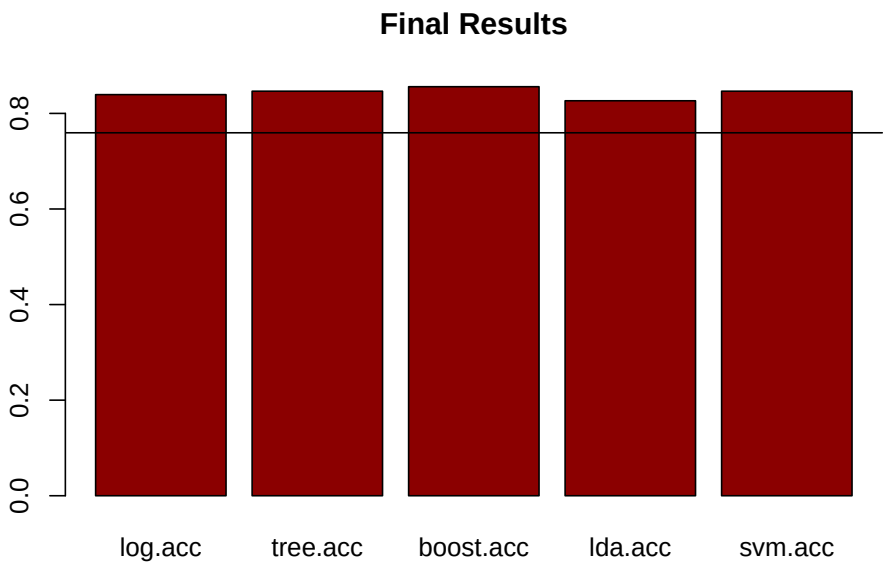
11.3 Interpretations

We observe that both linear as well as radial kernels classify the observations with the same accuracy of about 84.6%. The Cost parameter in SVM indicates the soft margin for classification. A small cost indicates large margins and allows mis-classification. This also implies that the support vectors have a huge influence on a hyperplane if it existed. In our model the cost is 1, which is low. It implies that there are chances of misclassification. The high number of observations, and most predictors being categorical variables could be a possible reason. This is evidenced by the fact that the number of support vectors is enormous.

12 Conclusions

12.1 Accuracy levels

```
# Combining all the accuracy levels.
all.acc <- cbind(log.acc, tree.acc, boost.acc, lda.acc, svm.acc)
barplot(all.acc,  col = "darkred", main="Final Results")
abline(h=baseline)
```

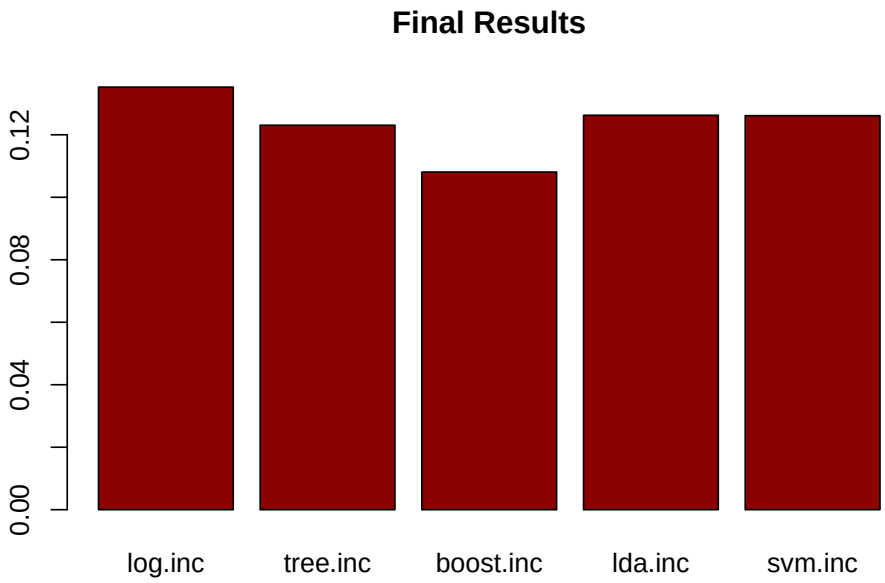


12.2 Income Prediction

```
base.table <- table(df2.test$salary)
base.inc <- base.table[2]/7541
base.inc
```

```
##      >50k
## 0.2405517
```

```
# Combining income level predictions
all.inc <- cbind(log.inc, tree.inc, boost.inc, lda.inc, svm.inc)
barplot(all.inc,  col = "darkred", main="Final Results")
abline(h=base.inc)
```



We looked at many important determinants of income in our model including variables such as age, employment status, race, etc. Age, education, and number of working hours were significant features in classifying whether an individual will earn more than $50,000/year in the year 1994. We used several classification techniques to predict the likelihood of a respondent to fall under the desired category. In these, we attained an accuracy rate of more than 80% for all the models.

Also, to answer the question - ***Will your income exceed $50k/year?***; Our classifiers suggest that approximately, there is a 12% likelihood that a citizen in the US in 1994, would earn more $50000 per year.

References

[1] R Markdown Guide. https://rmarkdown.rstudio.com/
[2] Predicting if income exceeds $50,000 per year based on 1994 US Census Data with Simple Classification Techniques.
http://cseweb.ucsd.edu/classes/sp15/cse190-c/reports/sp15/048.pdf
[3] A Guide to Machine Learning in R for Beginners: Logistic Regression. https://medium.com/analytics-vidhya/a-guide-to-machine-learning-in-r-for-beginners-part-5-4c00f2366b90
[4] Source for dataset: http://archive.ics.uci.edu/ml/datasets/Adult