Incomes predicter

Jesús Aguerri

23/9/2019

1. INTRODUCTION

The goal of this project is to develop and algorithm that could predict if someone wins less than 50.000 (50K) dollars per year. We are going to use for that the data set "Adult census income", avaible in the website kaggle.
https://www.kaggle.com/uciml/adult-census-income/kernels
This data-set was extracted from the 1994 Census bureau database by Ronny Kohavi and Barry Becker and contain information about union states of america citizens older than 16 year old. This data could look a bit older, however this data-set have as adventage that it´s almost ready to train algorithm and that many people have used it to build their own models, which give us a lot of usefull information to try to bulid our own algorithm.
In addition, even though this project will finish when the algorithm is built, the real utility of this project is that the developed algorithm could be used in other datasets with sociodemografic information.
In consequence, first of all we will present the data and we will analyse them, looking for the varaibles that could be usefulls to develop the algorithm. Later, we will test different models and finally we will pick that with a better performance. We will hightlight the model accuracy, but always having in mind the balance between recall and precision. In aditton, the algorithm interpretabillity will be also really important for us, becasuse to know what varaibles could explain the major varaiabillity across the data are going to be even as important as the algorithm accuracy.

2. TO GET AND TO PRESENT THE DATA

During this project we will use the following packages:

```
#Tidycerse
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
#Caret
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
```

The data can be download from Jesús Aguerri Git-hub account using ths code:

```
dl <- tempfile()
download.file("https://raw.githubusercontent.com/jcaguerri/adult-income-project/master/adultdataset.csv
, dl)
adults <- read.csv(dl)
```

Using dim() we can see how many rows (cases) and how many variables have the dataset.

```
dim(adults)
```

```
## [1] 32561    15
```

And using the functions head and str we can see the adults data-set structure

```r
head(adults) #first rows of the dataset
```

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

```r
str(adults) #varaible structure
```

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

Beffore go deeply we have to see that there are somo "?" working as values, so we´re going to replace it for "N/A" values. This let us omit them later and to avoid confusions. "N/A" values are easy to count and, as we can see below, there are more than 4.000 missing values.

```r
adults <- adults %>% na_if("?")
sum(is.na(adults)) #there are more than 4000 missing values
```

```
## [1] 4262
```

Finally we have to hightlight that the main variable, that we want to predict is the varaiable income. In the next section of the project we are going to explore what variables look related with the incomes, but we must to have always in mind the distribution of the incomes acros he population.

```
adults  %>%
  summarise("less50K" = mean(income== "<=50K"),
            "more50K" = mean(income == ">50K"),
            "total" = n())
```

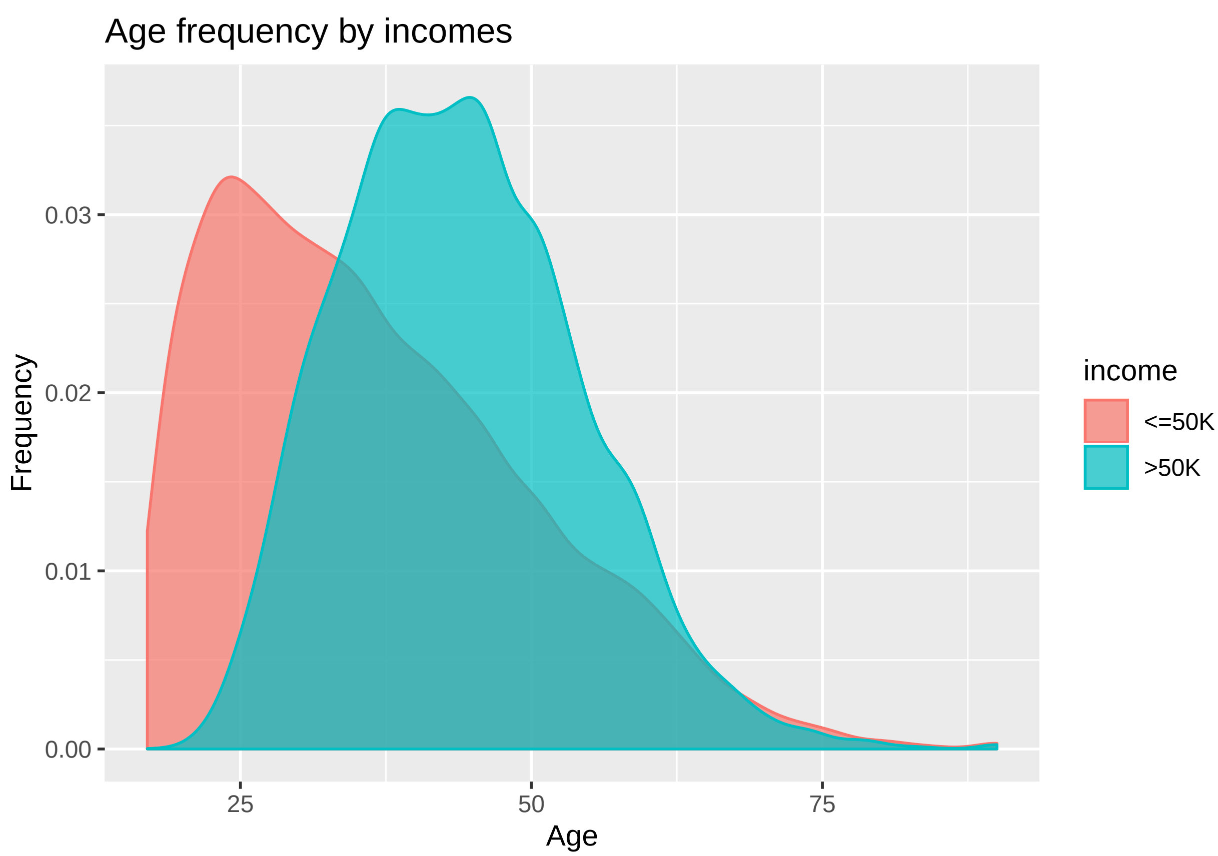
```
##     less50K   more50K total
## 1 0.7591904 0.2408096 32561
```

The proportion of people that earn more or less than 50k per year will be reference that able us to say if a variable loos related or not related with the incomes.

3. ANALYZING THE VARIABLES

3.1. Incomes by age

```
adults %>% na.omit() %>%
  ggplot(aes(age, color= income, fill= income)) +
  geom_density(alpha= 0.7) +
  labs(x = "Age", y = "Frequency", title = "Age frequency by incomes")
```



As the plot shows, youngest people seems earn less than 50K with more frequency that older people.

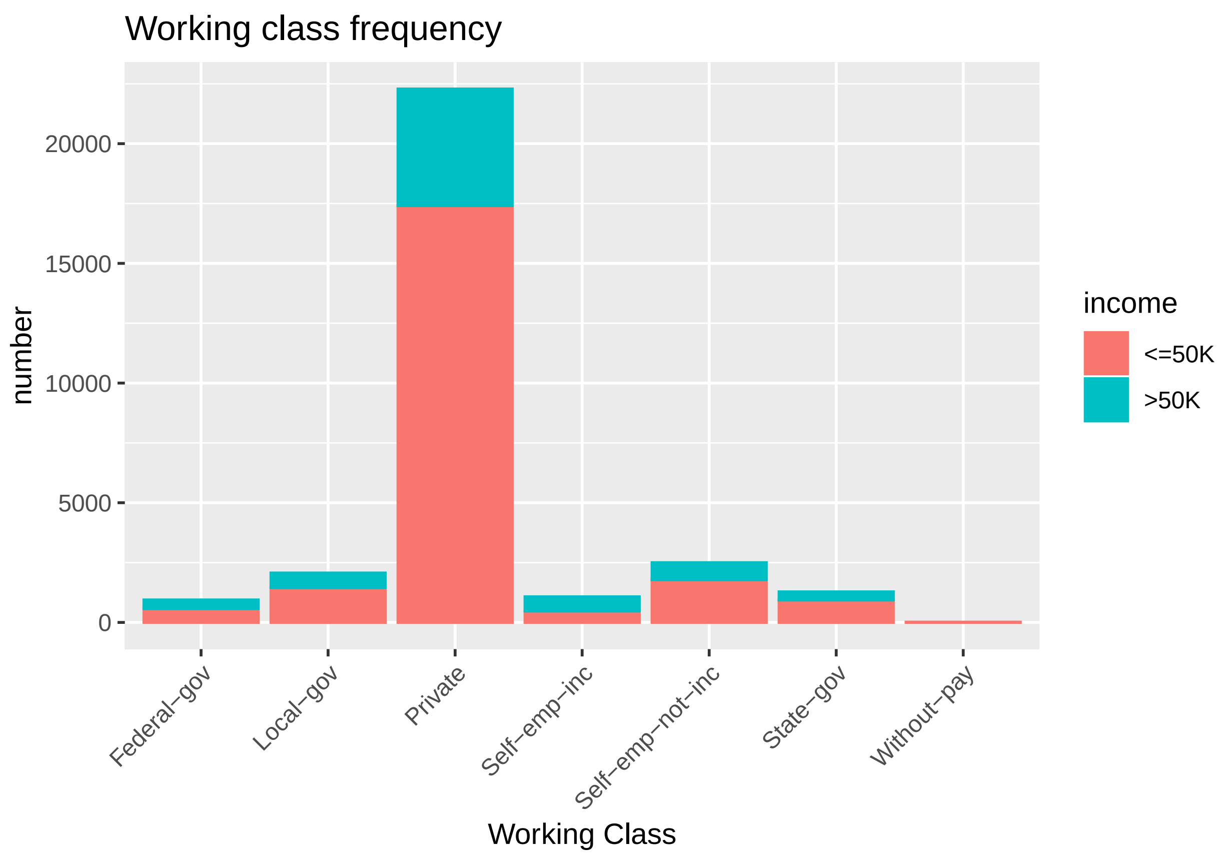
3.2 Incomes by workclass

As the following table and the following plot show, there are some labour market sector where to earn more than 50k is more usual than in the general population. Those sectors are formed by self employers and persons who works for the federal gouvernment.

```
adults  %>%
  na.omit() %>%
  group_by(workclass) %>%
  summarise("less50K" = mean(income== "<=50K"),
            "more50K" = mean(income == ">50K"),
            "total" = n()) %>%
  arrange(desc(more50K)) %>%
  knitr::kable()
```

workclass	less50K	more50K	total
Self-emp-inc	0.4413408	0.5586592	1074
Federal-gov	0.6129374	0.3870626	943
Local-gov	0.7053701	0.2946299	2067
Self-emp-not-inc	0.7142857	0.2857143	2499
State-gov	0.7310399	0.2689601	1279
Private	0.7812079	0.2187921	22286
Without-pay	1.0000000	0.0000000	14

```
adults %>% na.omit() %>%
  ggplot(aes(workclass, color= income, fill= income)) +
  geom_bar(position = position_stack(reverse = TRUE)) +
  labs(x = "Working Class", y = "number", title = "Working class frequency")+
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



3.3. Incomes by education

```
#Table
adults  %>%
  na.omit() %>%
  group_by(education) %>%
  summarise("less50K" = mean(income== "<=50K"),
            "more50K" = mean(income == ">50K"),
            "total" = n()) %>%
  knitr::kable()
```

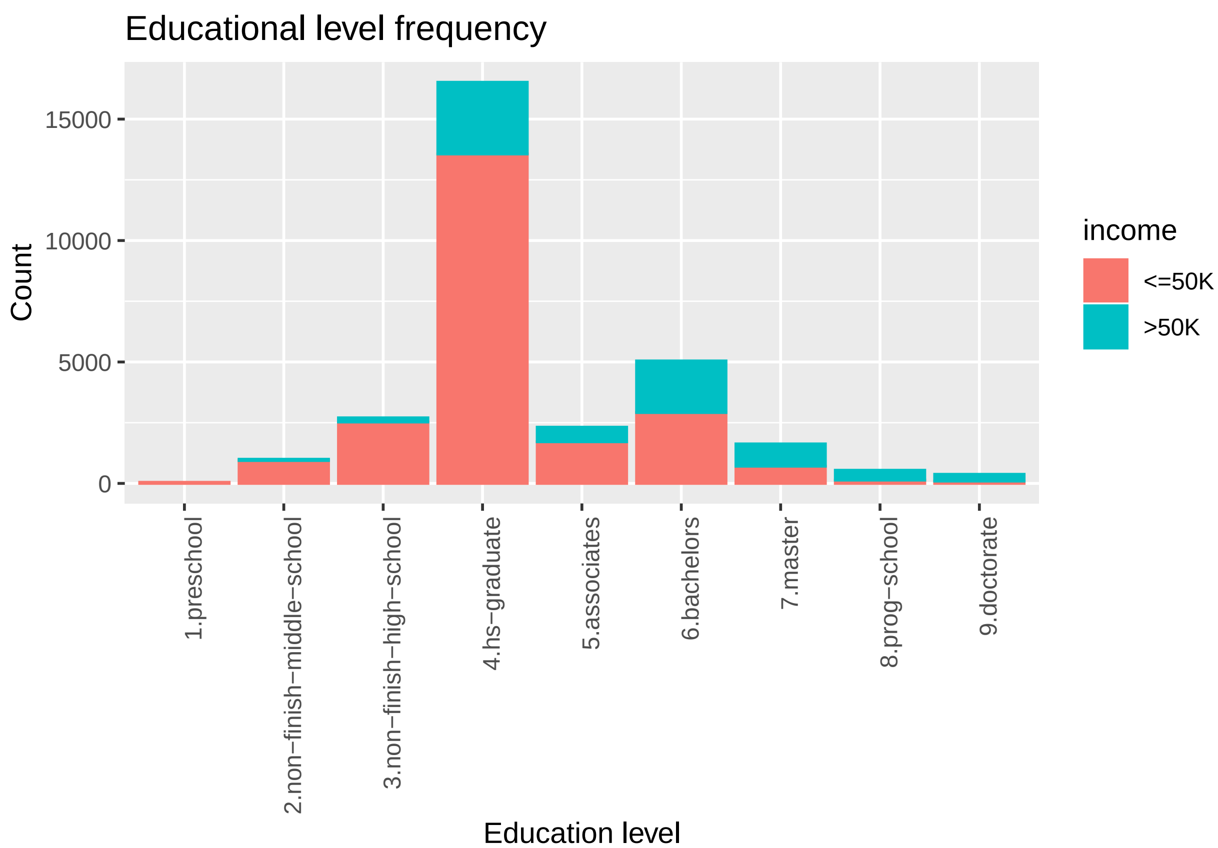
education	less50K	more50K	total
10th	0.9280488	0.0719512	820
11th	0.9437023	0.0562977	1048
12th	0.9230769	0.0769231	377
1st-4th	0.9602649	0.0397351	151
5th-6th	0.9583333	0.0416667	288
7th-8th	0.9371634	0.0628366	557
9th	0.9450549	0.0549451	455
Assoc-acdm	0.7460317	0.2539683	1008
Assoc-voc	0.7368018	0.2631982	1307
Bachelors	0.5785091	0.4214909	5044
Doctorate	0.2533333	0.7466667	375

education	less50K	more50K	total
HS-grad	0.8356707	0.1643293	9840
Masters	0.4357714	0.5642286	1627
Preschool	1.0000000	0.0000000	45
Prof-school	0.2509225	0.7490775	542
Some-college	0.7999401	0.2000599	6678

The table and the plot show that this variable have 15 different levels, some of them have just a few cases and many have really close relations with the income variable. So we are going to group this variable by educational levels in order to get more robust predictions and to get a better performance of our algorithm.

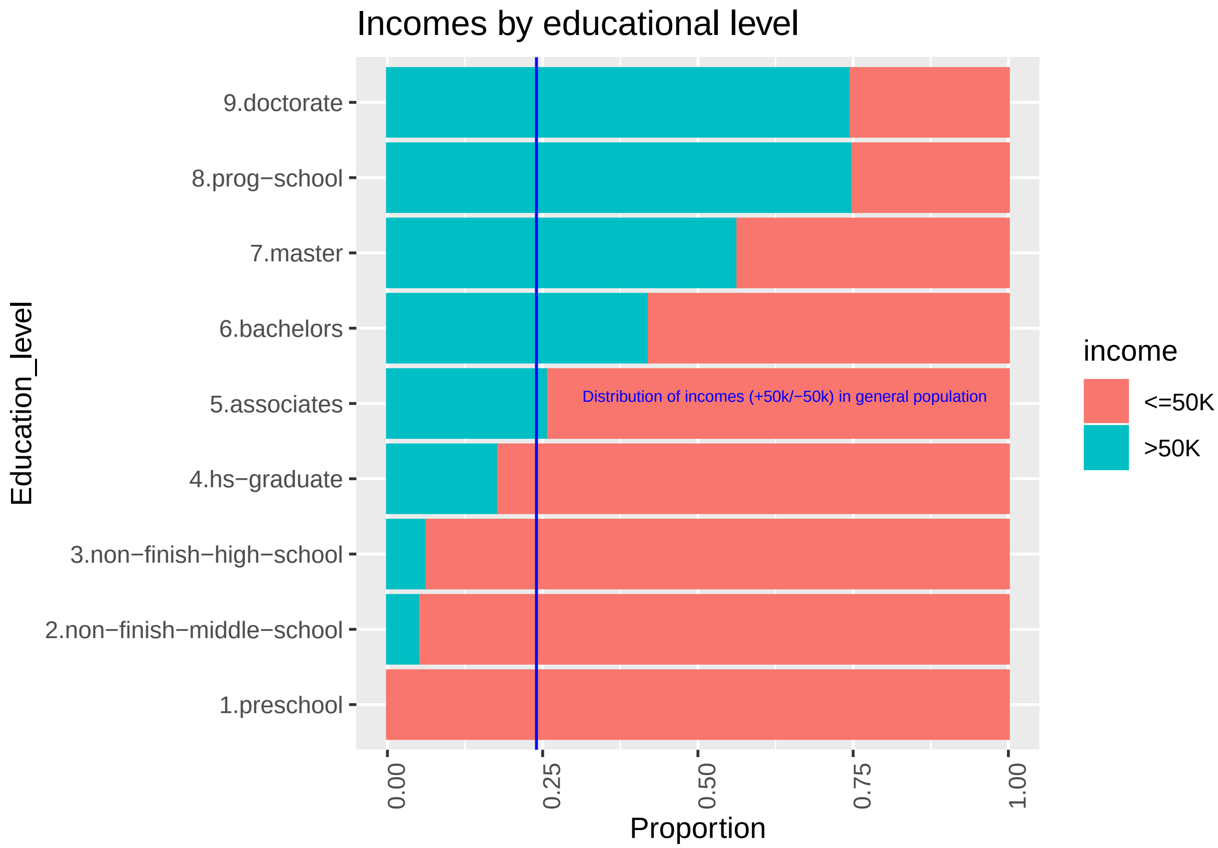
```
adults <- adults %>% mutate(education_level = case_when(
  .$education %in% c("1st-4th", "5th-6th", "7th-8th")~"2.non-finish-middle-school",
  .$education %in% c("9th", "10th", "11th", "12th")~"3.non-finish-high-school",
  .$education %in% c("Assoc-acdm", "Assoc-voc")~"5.associates",
  .$education %in% c("Some-college", "HS-grad")~"4.hs-graduate",
  .$education %in% "Bachelors" ~ "6.bachelors",
  .$education %in% "Prof-school" ~ "8.prog-school",
  .$education %in% "Preschool" ~ "1.preschool",
  .$education %in% "Doctorate" ~ "9.doctorate",
  .$education %in% "Masters" ~"7.master"))
```

```
#Plot(it shows mainly the distribution of educational levels across the population):
adults %>% na.omit() %>%
  ggplot(aes(education_level, color= income, fill= income)) +
  geom_bar(position = position_stack(reverse = TRUE)) +
  labs(x = "Education level",
       y = "Count",
       title = "Educational level frequency") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



The grouped variable able us to produce the following plot, which shows pretty well the positive relation between educational level and incomes

```
adults %>% na.omit() %>%
  ggplot(aes(education_level , color= income, fill= income)) +
  geom_bar(position = "fill") +
  coord_flip() +
  labs(x = "Education_level",
       y = "Proportion",
       title = "Incomes by educational level")+
  theme(axis.text.x = element_text(angle = 90, hjust = 1))+
  geom_hline(yintercept=0.24, col="blue") +
  annotate("text", x= 5.1, y= 0.64, size= 2, color="blue",
           label="Distribution of incomes (+50k/-50k) in general population")
```



3.4. Marital status

This varaiable have 7 labels, two of them are:
-Married-AF-spouse: Married armed forces spouse
-Married-civ-spouse: Married civilian spouse
Both levels can be grouped into the group "married".
In addition, the level "married-spouse-absent" could be joined to the label "Separated".

```
adults <- adults %>% mutate(marital_status = case_when(
  .$marital.status %in% c("Married-AF-spouse", "Married-civ-spouse")~"Married",
  .$marital.status %in% c("Separated", "Married-spouse-absent", "11th")~"separated",
  .$marital.status %in% "Divorced" ~ "divorced",
  .$marital.status %in% "Widowed" ~ "widowed",
  .$marital.status %in% "Never-married" ~ "never-married"))
```

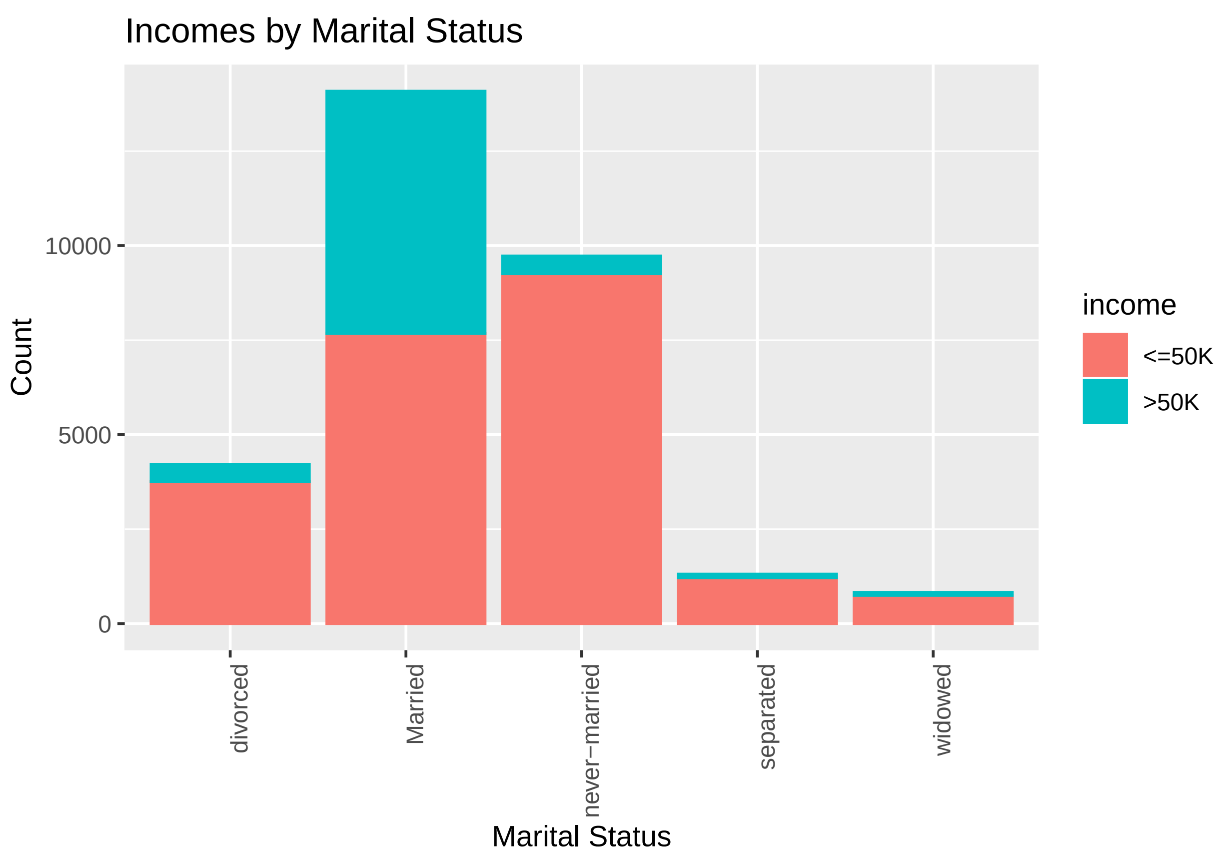
Now we can summarize the relation between the marital status and the incomes in the next table.

```
adults  %>%
  na.omit() %>%
  group_by(marital_status) %>%
  summarise("less50K" = mean(income== "<=50K"),
            "more50K" = mean(income == ">50K"),
            "total" = n()) %>%
  arrange(desc(more50K))%>%
  knitr::kable()
```

marital_status	less50K	more50K	total
Married	0.5450092	0.4549908	14086
divorced	0.8927385	0.1072615	4214
widowed	0.9032648	0.0967352	827
separated	0.9258976	0.0741024	1309
never-married	0.9516759	0.0483241	9726

And like the next plot shows, the married persons earn more frequnetly than other groups more tahn 50K per year

```
adults %>% na.omit() %>%
  ggplot(aes(marital_status, color= income, fill= income)) +
  geom_bar(position = position_stack(reverse = TRUE)) +
  labs(x = "Marital Status", y = "Count", title = "Incomes by Marital Status")+
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



3.5. Occupation

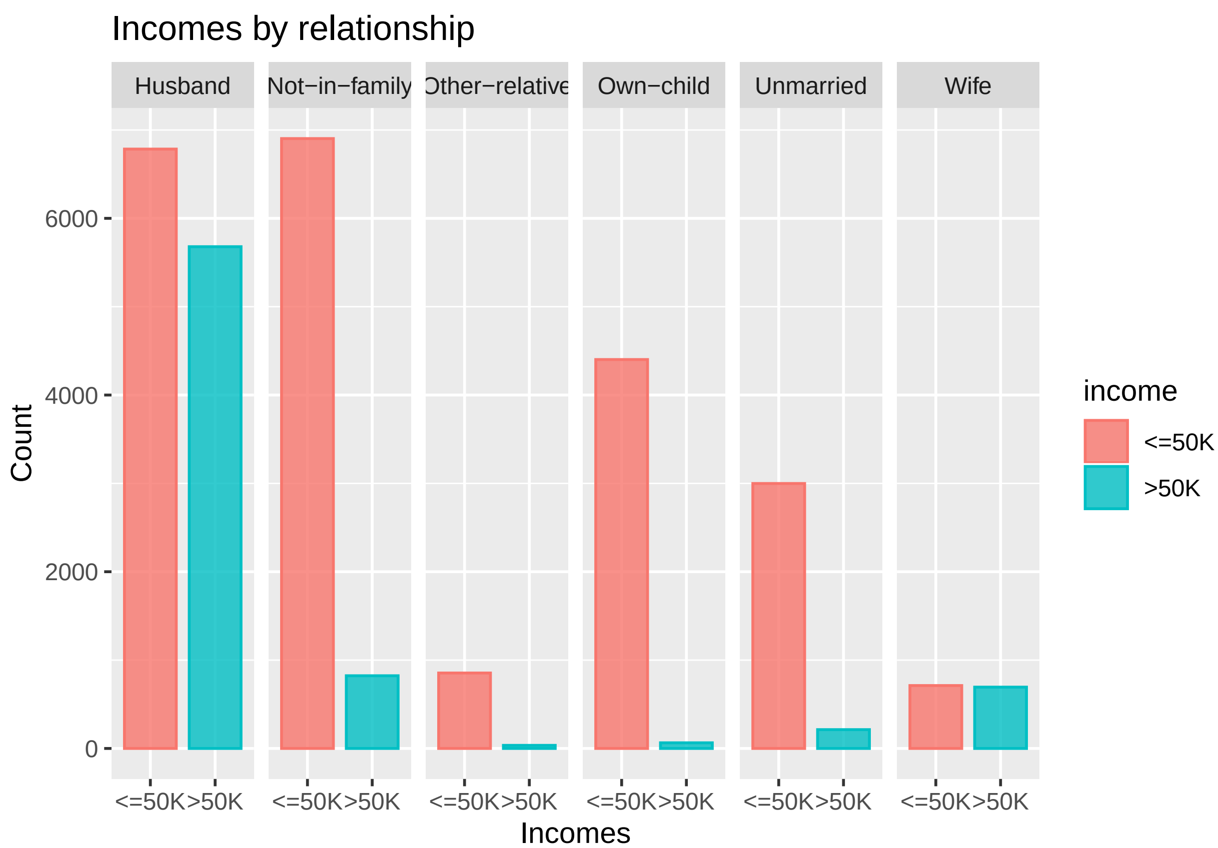
As the next plot show, there are some profesions in which looks more frequent earn more tahn 50k, like executives and proffesionals, an there anothers with the opposite situacion, escecially in the private hause service.

```
adults %>% na.omit() %>%
  ggplot(aes(occupation, color= income, fill= income)) +
  geom_bar(position = "fill") +
  coord_flip() +
  labs(x = "Ocupation", y = "Proportion", title = "Incomes by occupation")+
  theme(axis.text.x = element_text(angle = 90, hjust = 1))+
  geom_hline(yintercept=0.24, col="blue") +
  annotate("text", x= 5.1, y= 0.64, size= 2, color="blue",
           label="Distribution of incomes (+50k/-50k) in general population")
```

3.6. Relationship (rol into the family)

The following plot shows us that husbands and wifes have the highest proportions of individuals who earn more than 50k each year.

```
adults %>% na.omit() %>%
  ggplot(aes(income, color= income, fill= income)) +
  geom_bar( alpha = 0.8, width = 0.8) +
  facet_grid(~relationship)+
  labs(x = "Incomes", y = "Count", title = "Incomes by relationship")
```



3.7. Race

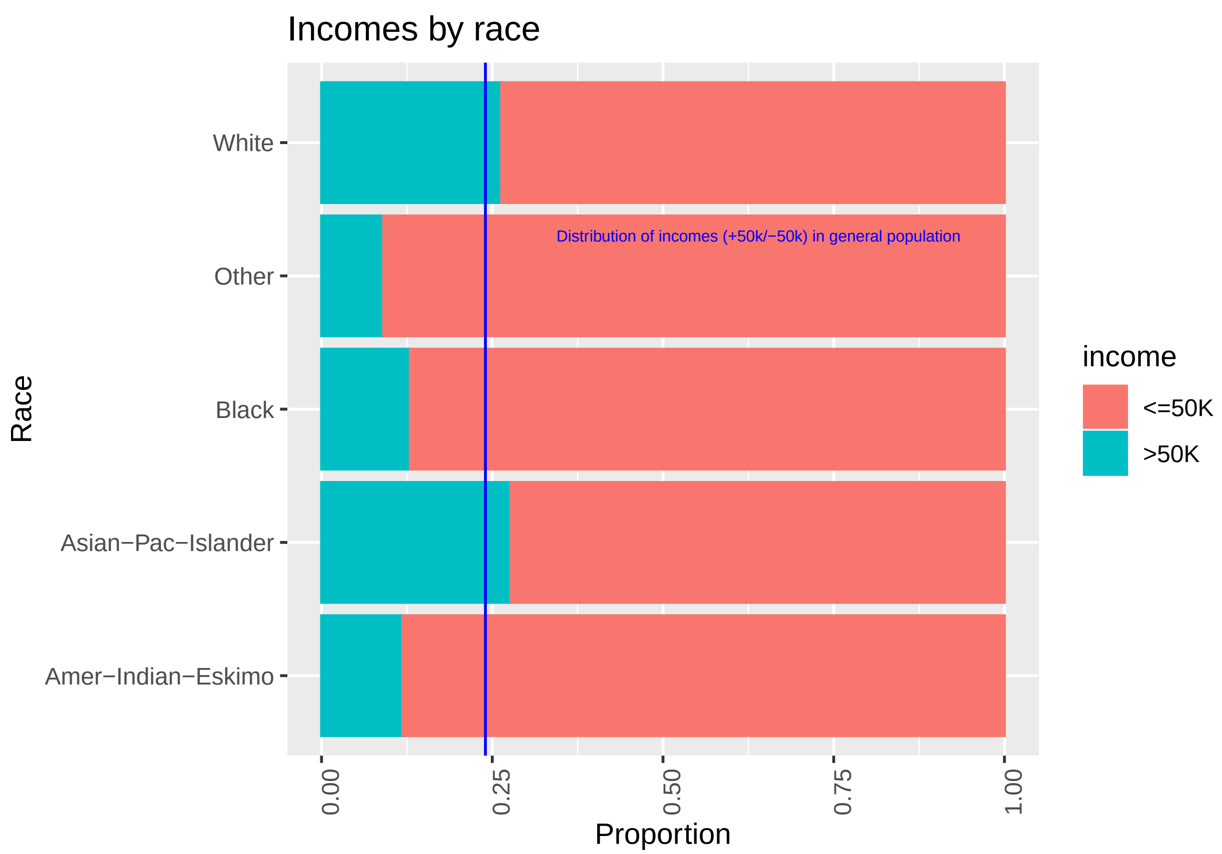
```
adults  %>% na.omit() %>%
  group_by(race) %>%
  summarise("less50K" = mean(income== "<=50K"),
            "more50K" = mean(income == ">50K"),
            "total" = n()) %>%
  arrange(desc(more50K)) %>%
  knitr::kable()
```

race	less50K	more50K	total
Asian-Pac-Islander	0.7229050	0.2770950	895
White	0.7362820	0.2637180	25933
Black	0.8700745	0.1299255	2817
Amer-Indian-Eskimo	0.8811189	0.1188811	286
Other	0.9090909	0.0909091	231

There seems to be a bias in income attending the race

```
adults %>% na.omit() %>%
  ggplot(aes(race, color= income, fill= income)) +
  geom_bar(position = "fill") +
```

```r
  coord_flip() +
  labs(x = "Race", y = "Proportion", title = "Incomes by race")+
  theme(axis.text.x = element_text(angle = 90, hjust = 1))+
  geom_hline(yintercept=0.24, col="blue") +
  annotate("text", x= 4.3, y= 0.64, size= 2, color="blue",
           label="Distribution of incomes (+50k/-50k) in general population")
```



Black and "Amer-Indio-Eskima" earn less than 50k more frequently than the general population. However, the percent of whites and "Asian-Pac-Islander" that earn more than 50k is over the general population average.

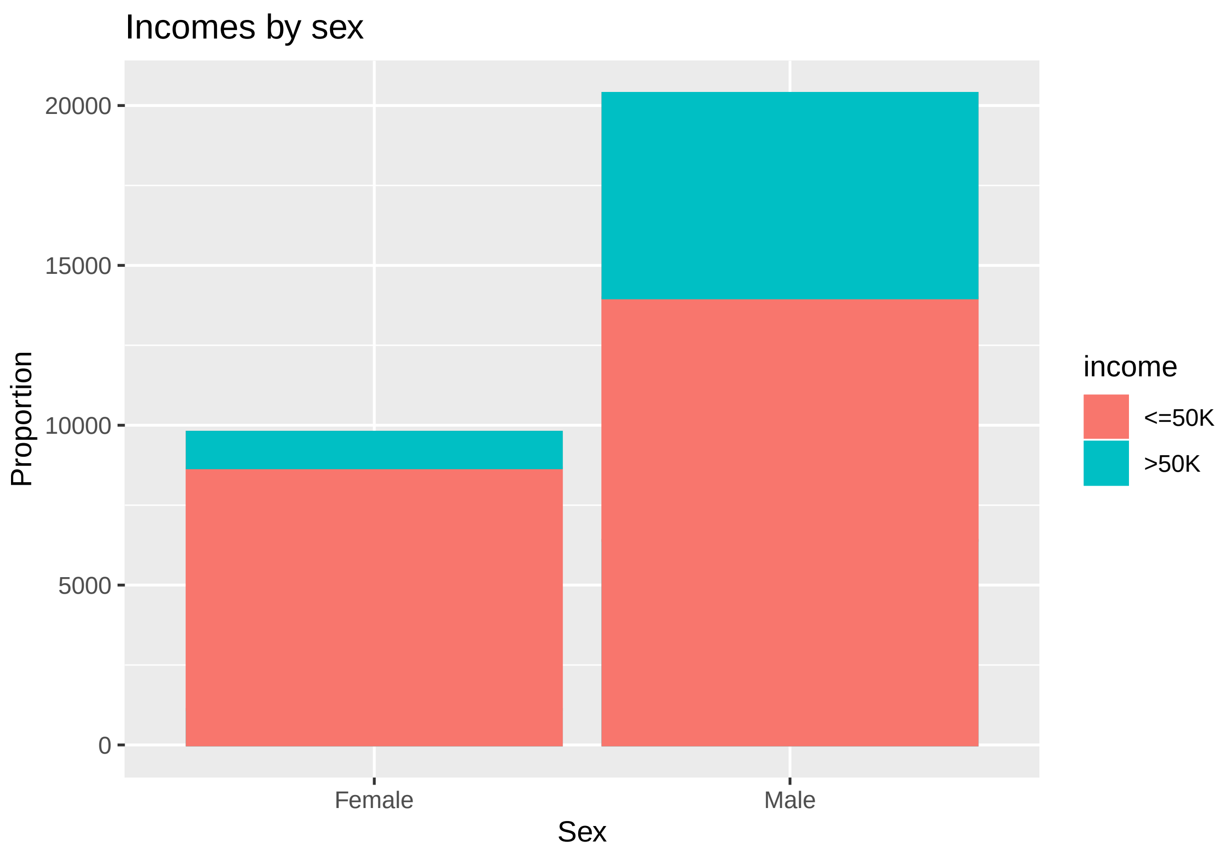
3.8. Gender

There seems to be a gender bias also. The proportion of womans that earn more than 50k is quite low that the proportion of mens that do it.

```r
adults  %>% na.omit() %>%
  group_by(race) %>%
  summarise("less50K" = mean(income== "<=50K"),
            "more50K" = mean(income == ">50K"),
            "total" = n()) %>%
  arrange(desc(more50K)) %>%
  knitr::kable()
```

race	less50K	more50K	total
Asian-Pac-Islander	0.7229050	0.2770950	895
White	0.7362820	0.2637180	25933
Black	0.8700745	0.1299255	2817
Amer-Indian-Eskimo	0.8811189	0.1188811	286
Other	0.9090909	0.0909091	231

```
adults %>% na.omit() %>%
  ggplot(aes(sex, color= income, fill= income)) +
  geom_bar() +
  labs(x = "Sex", y = "Proportion", title = "Incomes by sex")+
  geom_bar(position = position_stack(reverse = TRUE))
```



3.9. Capital Variations

We are goint to mix the varaibles capital.loss and capital.gain into a unique varaible with the capital variation.

```
adults <- adults %>% mutate(capital_variation = capital.gain - capital.loss)
```

```
adults  %>% na.omit() %>%
  group_by(income) %>%
  summarise("capital_var_mean" = mean(capital_variation),
            "capital_var_min" = min(capital_variation),
```

```
            "capital_var_max" = max(capital_variation),
            "total" = n()) %>%
  knitr::kable()
```

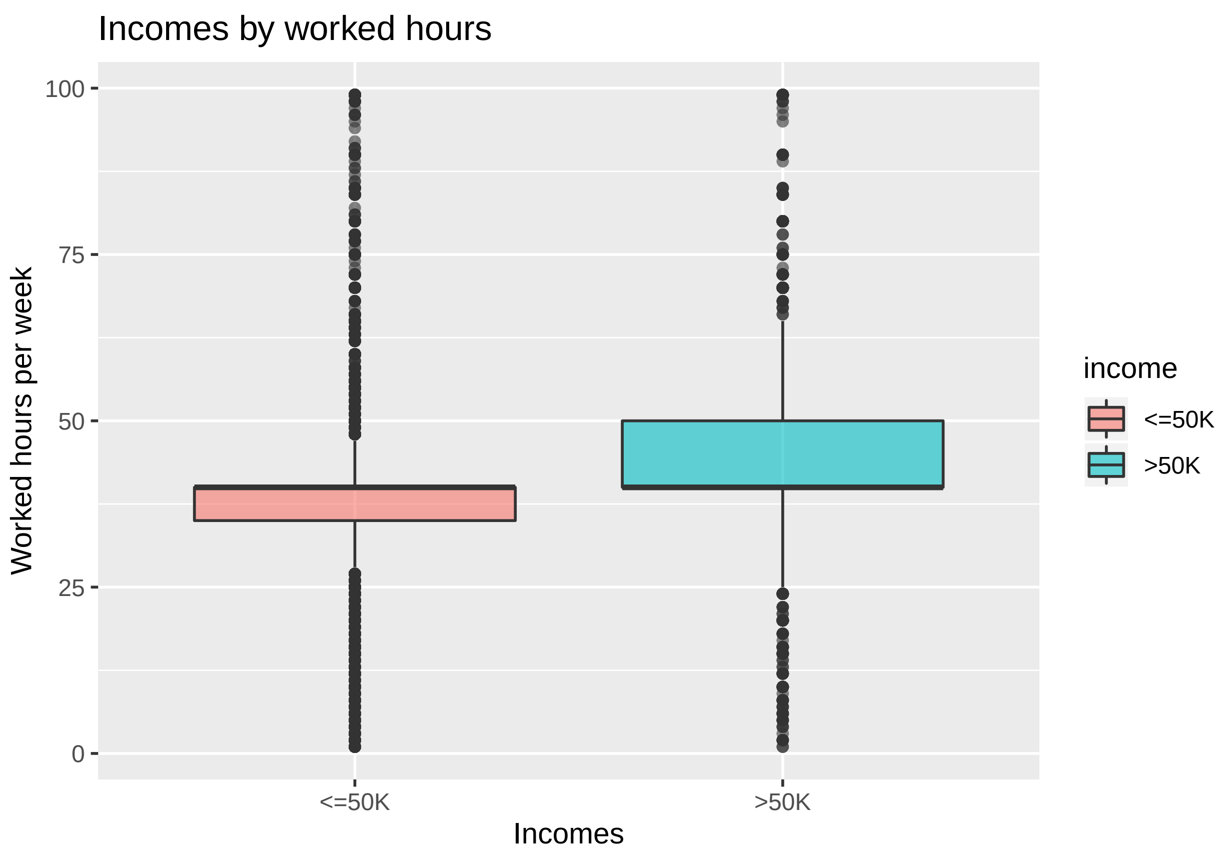
income	capital_var_mean	capital_var_min	capital_var_max	total
<=50K	95.44584	-4356	41310	22654
>50K	3743.92914	-3683	99999	7508

3.10. Incomes by worked hour per week

```
adults  %>% na.omit() %>%
  group_by(income) %>%
  summarise("Mean hours per week" = mean(hours.per.week),
            "Standard deviatrion" = sd(hours.per.week)) %>%
  knitr::kable()
```

income	Mean hours per week	Standard deviatrion
<=50K	39.34859	11.95077
>50K	45.70658	10.73699

```
adults %>% ggplot(aes(income, hours.per.week, fill=income))+
  geom_boxplot(alpha= 0.6)+
  labs(x = "Incomes", y = "Worked hours per week", title = "Incomes by worked hours")
```



As the table and the boxplot shows those who earn more than 50k work more hours. However, we must have in mind that there are a lot of ouliers and that the standard deviation is quite high in both groups.

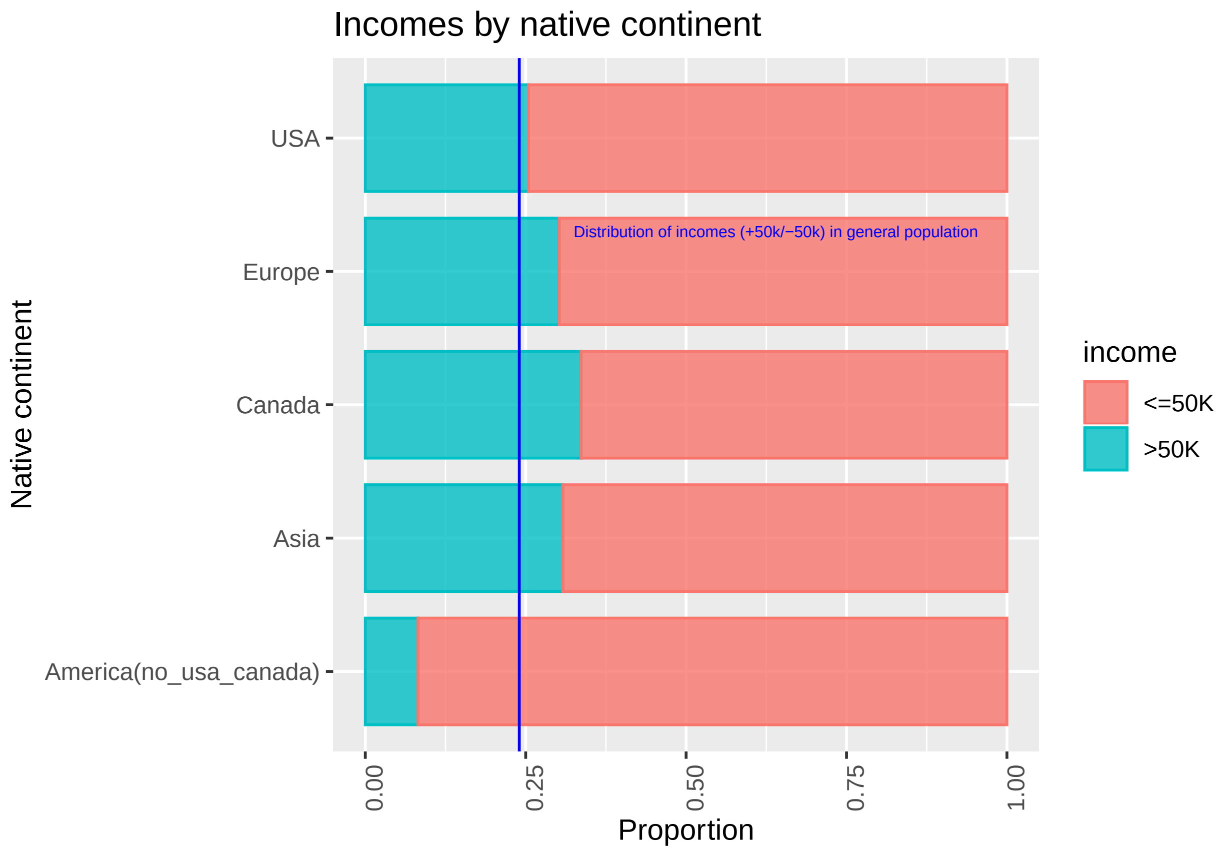
3.11. Incomes by origin

There are 40 differnet native countries and some of them have just a few cases, for instance, honduras have just 10 cases. So we´re going to group the countries by continent. This will give us a more robust approach to incomes variability across national origin.

```
adults <- adults %>% mutate(native_continent = case_when(
  .$native.country %in% c("France", "Greece", "Hungary", "Italy", "Portugal",
                          "Scotland", "England", "Germany", "Holand-Netherlands",
                          "Ireland", "Poland", "Yugoslavia")~"Europe",
  .$native.country %in% c("Columbia", "Dominican-Republic", "El-Salvador", "Haiti",
                          "Honduras", "Mexico", "Outlying-US(Guam-USVI-etc)",
                          "Cuba", "Ecuador", "Guatemala", "Jamaica", "Nicaragua",
                          "Peru", "Puerto-Rico", "Trinadad&Tobago")~"America(no_usa_canada)",
  .$native.country %in% c("Iran", "Japan", "Philippines", "Taiwan", "Vietnam", "Cambodia",
                          "China", "Hong", "India", "Laos", "South", "Thailand")~"Asia",
  .$native.country %in% "United-States" ~ "USA",
  .$native.country %in% "Canada" ~"Canada"))
```

The following plot shows some bias acording to continental origin. It´s specially interesting to see that those who came from other countries in America (omited Canada) have the lowest chance to earn more than 50K per year.

```r
adults %>% na.omit() %>%
  ggplot(aes(native_continent, color= income, fill= income)) +
  geom_bar(position = "fill", alpha = 0.8, width = 0.8) +
  coord_flip() +
  labs(x = "Native continent", y = "Proportion", title = "Incomes by native continent")+
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  geom_hline(yintercept=0.24, col="blue") +
    annotate("text", x= 4.3, y= 0.64, size= 2, color="blue",
           label="Distribution of incomes (+50k/-50k) in general population")
```



4. METHODS AND MODELS

First of all we have to prepare our data, so we are going to select these varaibles that we have created and to drop those that we aren´t going to use.

```r
adults <- adults %>% select(age, workclass, education_level, marital_status,
                             occupation, relationship, race,
                            sex, capital_variation,
                             hours.per.week, native_continent, income)
head(adults)
```

```
##   age workclass            education_level marital_status
## 1  90      <NA>              4.hs-graduate        widowed
```

```
## 2  82   Private              4.hs-graduate        widowed
## 3  66      <NA>              4.hs-graduate        widowed
## 4  54   Private 2.non-finish-middle-school       divorced
## 5  41   Private              4.hs-graduate      separated
## 6  34   Private              4.hs-graduate       divorced
##          occupation  relationship  race    sex capital_variation
## 1              <NA> Not-in-family White Female             -4356
## 2   Exec-managerial Not-in-family White Female             -4356
## 3              <NA>     Unmarried Black Female             -4356
## 4 Machine-op-inspct     Unmarried White Female             -3900
## 5    Prof-specialty     Own-child White Female             -3900
## 6     Other-service     Unmarried White Female             -3770
##   hours.per.week native_continent income
## 1             40              USA  <=50K
## 2             18              USA  <=50K
## 3             40              USA  <=50K
## 4             40              USA  <=50K
## 5             40              USA  <=50K
## 6             45              USA  <=50K
```

Those are the varaiables that we are going to use for try to predict the ouput income. We have droped those varaibles that we have modificated, and the varaibles education.num (the education level but showed as numerical variable) and fnlwgt. This second varaible represent the weight or each case in the population, it could help to improve this project. However, we have decided to do our work ignoring this variable, because even without it, we can get results statistically significants.
Second we have to remove the N/A cases.

```r
sum(is.na(adults))
```

```
## [1] 4262
```

```r
adults <- na.omit(adults)
```

So the size of our data have been reduced to 30.162 cases

```r
dim(adults)
```

```
## [1] 30162    12
```

Finally, we have to create a data partition into two sets. the train set will be used to train our models, and the test set(composed by 20% of the data) will be used to test our models performance.

```r
set.seed(1)
test_index <- createDataPartition(y = adults$income, times = 1, p = 0.2, list = FALSE)
train <- adults[-test_index,]
test <- adults[test_index,]
```

Now that we have already prepared our data, we can start to test different models.

4.1. Logistic regression (GLM)

Logistic regression is a linear model suitable for predict categorical outcomes.

```
set.seed(1)
glm_model <- train(income~., data=train, method= "glm",
                   trControl = trainControl(method = "cv", number=5, p=0.9)) #Chosen crossvalidation
y_hat_glm <- predict(glm_model, test, type = "raw") #Predict
confusionMatrix(y_hat_glm, test$income) #Show the results
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction <=50K >50K
##      <=50K  4196  628
##      >50K    335  874
##
##                Accuracy : 0.8404
##                  95% CI : (0.8309, 0.8495)
##     No Information Rate : 0.751
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5434
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9261
##             Specificity : 0.5819
##          Pos Pred Value : 0.8698
##          Neg Pred Value : 0.7229
##              Prevalence : 0.7510
##          Detection Rate : 0.6955
##    Detection Prevalence : 0.7996
##       Balanced Accuracy : 0.7540
##
##        'Positive' Class : <=50K
##
```

The model accuracy is 0.8404, which is quite good. In addition, sensitivity is really good, 0.9261. However, specificity isn´t so good, because it´s just 0.5819

4.2. K-nearest neighbor (KNN)

KNN is another supervised machine learning algorithm. This algotihm have a tunning parameter K, so we have tested different parameters k.

```
set.seed(1)
knn_model <- train(income~., data=train, method = "knn",
                   tuneGrid = data.frame(k = seq(3, 20, 2)),#test differents k
                   trControl = trainControl(method = "cv", number=5, p=0.9)) #5-folds
knn_model$bestTune
```

```
##   k
## 2 5
```

The parameter K that maximeize the accuracy is K=13. Using this K we can to test our model traying to predict the results

```
y_hat_knn <- predict(knn_model, test, type = "raw") #Predict
confusionMatrix(y_hat_knn, test$income) #show results
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction <=50K >50K
##      <=50K  4132  579
##      >50K    399  923
##
##                Accuracy : 0.8379
##                  95% CI : (0.8283, 0.8471)
##     No Information Rate : 0.751
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5484
##
##  Mcnemar's Test P-Value : 1.042e-08
##
##             Sensitivity : 0.9119
##             Specificity : 0.6145
##          Pos Pred Value : 0.8771
##          Neg Pred Value : 0.6982
##              Prevalence : 0.7510
##          Detection Rate : 0.6849
##    Detection Prevalence : 0.7809
##       Balanced Accuracy : 0.7632
##
##        'Positive' Class : <=50K
##
```

4.3. Classification tree

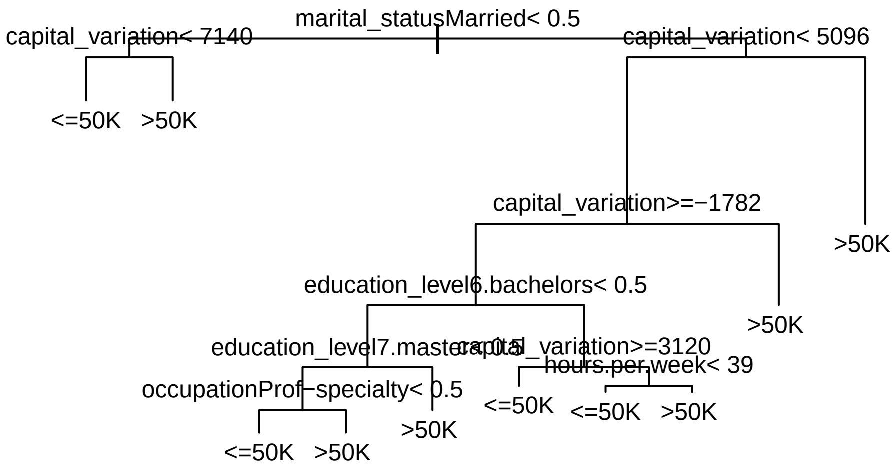
Classification tree is a supervised machine learning algorithm that predict categorical outcomes. It don´t have the best accuracy, but, this algorithm able to visualice the results in a plot that give information about the importance of each varaible.

```
set.seed(1)
rpart_model <- train(income ~., data = train, method = "rpart",
                   tuneGrid = data.frame(cp = seq(0, 0.05, 0.005)),
                   trControl = trainControl(method = "cv",
                                            number=5,
                                            p=0.9)) #Choose crossvalidation with
                                                    #5 k-folds)
y_hat_rpart <- predict(rpart_model, test)#Predict
confusionMatrix(y_hat_rpart, test$income)#Show results
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction <=50K >50K
##      <=50K  4273  715
##      >50K    258  787
##
##                Accuracy : 0.8387
##                  95% CI : (0.8292, 0.8479)
##     No Information Rate : 0.751
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5199
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9431
##             Specificity : 0.5240
##          Pos Pred Value : 0.8567
##          Neg Pred Value : 0.7531
##              Prevalence : 0.7510
##          Detection Rate : 0.7083
##    Detection Prevalence : 0.8268
##       Balanced Accuracy : 0.7335
##
##        'Positive' Class : <=50K
##
```

To plot the tree able us to see what varaibles have the mains rol. Those varaibles are the capital variation, the marital status and the education level.

```r
plot(rpart_model$finalModel, margin=0.1)
text(rpart_model$finalModel, cex = 0.75)
```



4.4. Ramdom forest

Ramdom forest is a powerfull supervised machine learning algorithm. It have two parameters that can be optimized: the mtry (the number of variables that are randomly collected to be sampled at each split time), and the number of trees that the algorithm is going to built.
The default behavoir of "rf" in the train function is to built 500 hundreed trees, to test a long number of mtry and to use 25 bootstraps. This produce that the code take several hours for run. By mistake, this was our first approach to the algorithm.

```
rf_mode <- train(income~., data=train, method = "rf")
rf_mode$bestTune
rf_mode$finalModel
```

An algorithm that needs several hours for run it´s not usufull (at least in this case). However, this first approach able us to pick the mtry and to reduce the number of trees that we can built improving the performance but remaining the same error. So we have built a best ramdom forest model using 50 trees and mtry= 27. In addition, change bootstrap for crossvalidation with 5 k folds have also let us to make the algorithm more usefull.

```
rf_fit_mod <- train(income~., data= train, method= "rf",
                    tuneGrid = data.frame(mtry = 27),
                    trControl= trainControl(method="cv",
                                            number=5), #CV with 5 folds
                    ntree = 50 )
```

```r
y_hat_rf <- predict(rf_fit_mod, test, type = "raw") #Predict
confusionMatrix(y_hat_rf, test$income) #Show results
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction <=50K >50K
##      <=50K  4173  582
##      >50K    358  920
##
##                Accuracy : 0.8442
##                  95% CI : (0.8348, 0.8533)
##     No Information Rate : 0.751
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5615
##
##  Mcnemar's Test P-Value : 3.504e-13
##
##             Sensitivity : 0.9210
##             Specificity : 0.6125
##          Pos Pred Value : 0.8776
##          Neg Pred Value : 0.7199
##              Prevalence : 0.7510
##          Detection Rate : 0.6917
##    Detection Prevalence : 0.7882
##       Balanced Accuracy : 0.7668
##
##        'Positive' Class : <=50K
##
```

As we can see, the accuracy is now 0.846, the sensitivity is 0.9225, and the scecificity have improved to 0.6152.
Ramdom forest also able us to see which are the more important varaibles

```r
varImp(rf_fit_mod) #varaible importance
```

```
## rf variable importance
##
##   only 20 most important variables shown (out of 51)
##
##                                Overall
## capital_variation              100.000
## age                             90.577
## marital_statusMarried           87.682
## hours.per.week                  50.791
## marital_statusnever-married     21.263
## occupationProf-specialty        15.557
## education_level6.bachelors      14.294
## occupationExec-managerial       13.581
## workclassPrivate                 7.634
## education_level7.master          7.552
```

```
## education_level4.hs-graduate   6.876
## sexMale                        6.746
## workclassSelf-emp-not-inc      6.426
## occupationSales                5.067
## raceWhite                      5.067
## occupationCraft-repair         4.762
## workclassLocal-gov             4.610
## native_continentUSA            4.530
## education_level5.associates    4.217
## relationshipWife               4.140
```

5. RESULTS

Now that we have already trained 4 differents algorithms is time to compare the results.

```
con_glm <- confusionMatrix(y_hat_glm, test$income)
con_knn <- confusionMatrix(y_hat_knn, test$income)
con_rpart <- confusionMatrix(y_hat_rpart, test$income)
con_rf <- confusionMatrix(y_hat_rf, test$income)
  #To build the table with the results:
models <- c("glm", "knn", "rpart", "rf")
```

```
      #Accuracies
accuracies <- c(con_glm$overall["Accuracy"], con_knn$overall["Accuracy"],
                con_rpart$overall["Accuracy"], con_rf$overall["Accuracy"])
```

```
resoults <- data.frame(models, accuracies)
resoults %>%knitr::kable()
```

models	accuracies
glm	0.8403779
knn	0.8378916
rpart	0.8387204
rf	0.8441903

Accuracies are quite high, however we must also have in mind the sensitivity (or Recall) and the specificity (precision).

```
#Acuracies, precision, recall and balanced accuracy.
resoults <- resoults %>%
  mutate(sensitivity = c(con_glm$byClass["Sensitivity"],
                         con_knn$byClass["Sensitivity"],
                         con_rpart$byClass["Sensitivity"],
                         con_rf$byClass["Sensitivity"]),
         specificity =  c(con_glm$byClass["Specificity"],
                          con_knn$byClass["Specificity"],
                          con_rpart$byClass["Specificity"],
                          con_rf$byClass["Specificity"]),
         balanced_accuracy = c(con_glm$byClass["Balanced Accuracy"],
```

```r
                                con_knn$byClass["Balanced Accuracy"],
                                con_rpart$byClass["Balanced Accuracy"],
                                con_rf$byClass["Balanced Accuracy"]))
resoults %>%knitr::kable()
```

models	accuracies	sensitivity	specificity	balanced_accuracy
glm	0.8403779	0.9260649	0.5818908	0.7539778
knn	0.8378916	0.9119400	0.6145140	0.7632270
rpart	0.8387204	0.9430589	0.5239680	0.7335135
rf	0.8441903	0.9209887	0.6125166	0.7667527

The table shows us that the all the sensitivities are highs, while the specificitys are lower. However, the model built by ramdom forest have a specificity better than the other models and also a quite good balanced accuracy. So we can affirm that the model built using ramdom forest reach our goal.

6. CONCLUSION

This porject aim was develop a model that let us predict if someone earn less than 50k per year. We have use different supervised machine learning algorithms to try to reach our goal. As we have already see, the model fit by ramdom forest have give us a good accuracy -0.846- and also a good balanced accuracy, so our model is also "good" predicting if someone earns more than 50k.
In addition, ramdom forest let us to know which are the more important varaibles to do the prediction. The three most important varaibles are tha capital variation, the age and the marital status. However, we have to highlight that these variables aren´t explicative variables, are just variables with certain correlation with the incomes and that are ussefulls to predict it. It is not our goal in this project, but if we want an explicative model we could re-run the ramdom forest selecting those varaibles that can be explicatives (this exercise is done in the annexed).

ANNEXED

```r
#RAMDOM FOREST JUST WITH EXPLICATIVE VARIABLES
rf_exp_mod <- train(income~ age + workclass+
                      education_level +occupation +
                      race + sex +hours.per.week +
                      native_continent ,
                    data= train,
                    method= "rf",
                    trControl= trainControl(method="cv",
                                            number=5), #CV with 5 folds
                    ntree = 100 )
y_hat_rf_exp <- predict(rf_exp_mod, test, type = "raw") #Predict
confusionMatrix(y_hat_rf_exp, test$income) #Show results
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction <=50K >50K
```

```
##      <=50K  4081  779
##      >50K    450  723
##
##                Accuracy : 0.7963
##                  95% CI : (0.7859, 0.8064)
##     No Information Rate : 0.751
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4122
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9007
##             Specificity : 0.4814
##          Pos Pred Value : 0.8397
##          Neg Pred Value : 0.6164
##              Prevalence : 0.7510
##          Detection Rate : 0.6764
##    Detection Prevalence : 0.8056
##       Balanced Accuracy : 0.6910
##
##        'Positive' Class : <=50K
##
```

```r
varImp(rf_exp_mod) #varaible importance
```

```
## rf variable importance
##
##   only 20 most important variables shown (out of 41)
##
##                                         Overall
## age                                     100.000
## hours.per.week                           52.462
## sexMale                                  16.363
## occupationExec-managerial                12.022
## occupationProf-specialty                 11.695
## education_level6.bachelors                8.261
## education_level4.hs-graduate              8.025
## workclassPrivate                          6.892
## workclassSelf-emp-not-inc                 5.616
## raceWhite                                 4.982
## education_level7.master                   4.760
## education_level3.non-finish-high-school   4.717
## occupationSales                           4.445
## native_continentUSA                       4.398
## education_level5.associates               4.104
## workclassLocal-gov                        4.028
## occupationCraft-repair                    3.863
## workclassSelf-emp-inc                     3.745
## raceBlack                                 3.678
## workclassState-gov                        3.498
```