Predicting Income Capstone Project

Vadym Polishchuk

2022-05-12

Introduction

For my final capstone project I chose a data set that I found on Kaggle called "Adult Census Income". It has information about US citizens and their annual income. Income feature has two variations: over 50K and less than 50K, so I am going to predict whether a person earns more or less than 50 thousand US dollars a year. The original "Adult Census Income" data set has a few billion records, but the data set I am going to be using is reduced by grouping people with similar data. *fnlwgt* column represents the final weight of each record (group) in the original data set which doesn't really matter in this project. The goal of this project is to explore the data, clean it if needed, find patterns and insights and finally build a predictive model.

In this project I am going to use all the data science tools that I have learned over the HarvardX Data Science program to build predictive models, visualize and explore different features and find interesting insights in data. It is an opportunity for me to better understand the principles of working with data as the best way to learn something is to practice.

This report consists of 4 parts:

Data Set Overview and Cleaning

Adult Census Income data set I am going to analyze is a tibble with 32561 rows and 15 columns. Each row represents certain group of people with similar preferences and each column is some piece of personal information about people. Here is a structure of a data set.

```
str(data, give.attr=F)
```

```
## spec_tbl_df [32,561 x 15] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ age           : num [1:32561] 90 82 66 54 41 34 38 74 68 41 ...
##  $ workclass     : chr [1:32561] "?" "Private" "?" "Private" ...
##  $ fnlwgt        : num [1:32561] 77053 132870 186061 140359 264663 ...
##  $ education     : chr [1:32561] "HS-grad" "HS-grad" "Some-college" "7th-8th" ...
##  $ education.num : num [1:32561] 9 9 10 4 10 9 6 16 9 10 ...
##  $ marital.status: chr [1:32561] "Widowed" "Widowed" "Widowed" "Divorced" ...
##  $ occupation    : chr [1:32561] "?" "Exec-managerial" "?" "Machine-op-inspct" ...
##  $ relationship  : chr [1:32561] "Not-in-family" "Not-in-family" "Unmarried" "Unmarried" ...
##  $ race          : chr [1:32561] "White" "White" "Black" "White" ...
##  $ sex           : chr [1:32561] "Female" "Female" "Female" "Female" ...
##  $ capital.gain  : num [1:32561] 0 0 0 0 0 0 0 0 0 0 ...
##  $ capital.loss  : num [1:32561] 4356 4356 4356 3900 3900 ...
##  $ hours.per.week: num [1:32561] 40 18 40 40 40 45 40 20 40 60 ...
##  $ native.country: chr [1:32561] "United-States" "United-States" "United-States" "United-States" ...
##  $ income        : chr [1:32561] "<=50K" "<=50K" "<=50K" "<=50K" ...
```

There are 6 numerical features and 9 character features. We will transform some of the character features into binary or factored form later. But first let's check for the NA values.

```
sum(colSums(is.na(data)))
```

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

As we can see there are no NA's in the data set. Now let's have a look at unique values in each column.

```
sapply(data, function(x){length(unique(x))})
```

```
##            age      workclass         fnlwgt      education  education.num
##             73              9          21648             16             16
## marital.status     occupation   relationship           race            sex
##              7             15              6              5              2
##   capital.gain   capital.loss hours.per.week native.country         income
##            119             92             94             42              2
```

There are at least two features that can be converted into binary form now and many others that can be transformed into factors later, after some exploratory analysis. Income data and sex data have only 2 different variations of values, so let's create new binary columns *income_factor* and *sex_factor* where 0 means "<=50K" income or "Female" gender and 1 means ">50K" income or "Male" gender. *income* and *sex* columns are changed into 0's and 1's as well but without factors (maybe we can use them later).

```
# make income feature binary and transform it to factor
data$income[data$income == "<=50K"] <- 0
data$income[data$income == ">50K"] <- 1
data$income <- as.numeric(data$income)
data$income_factor <- factor(data$income, levels = c(0,1))

# make sex feature binary and transform it to factor
data$sex[data$sex == "Female"] <- 0
data$sex[data$sex == "Male"] <- 1
data$sex <- as.numeric(data$sex)
data$sex_factor <- factor(data$sex, levels = c(0,1))
```

Also, there are some undefined values in columns such as "?". Let's change them to "Other" to make it more readable.

```
data$workclass[data$workclass == "?"] <- "Other"
data$occupation[data$occupation == "?"] <- "Other"
data$native.country[data$native.country == "?"] <- "Other"
```

Finally, there some features that don't have any predictive value on income and so are useless in the analysis. First is *fnlwgt*. It shows the connection with the original data set (that has over 6 billion records). All the similar records from the original data set set were grouped and *fnlwgt* feature shows weight of each group. Second is *education.num*. It is basically the copy of *education* feature but with numbers instead of characters. Later in the exploratory analysis I am going to transform the *education* feature a bit and then convert it to numbers. So at this point *education.num* is another useless feature. Let's drop these features.

```
data <- select(data, -fnlwgt, -education.num)
```

After all the cleaning the data set looks like this with 15 columns.

```
str(data, give.attr=F)
```

```
## tibble [32,561 x 15] (S3: tbl_df/tbl/data.frame)
##  $ age           : num [1:32561] 90 82 66 54 41 34 38 74 68 41 ...
##  $ workclass     : chr [1:32561] "Other" "Private" "Other" "Private" ...
##  $ education     : chr [1:32561] "HS-grad" "HS-grad" "Some-college" "7th-8th" ...
##  $ marital.status: chr [1:32561] "Widowed" "Widowed" "Widowed" "Divorced" ...
##  $ occupation    : chr [1:32561] "Other" "Exec-managerial" "Other" "Machine-op-inspct" ...
##  $ relationship  : chr [1:32561] "Not-in-family" "Not-in-family" "Unmarried" "Unmarried" ...
##  $ race          : chr [1:32561] "White" "White" "Black" "White" ...
##  $ sex           : num [1:32561] 0 0 0 0 0 0 1 0 0 1 ...
##  $ capital.gain  : num [1:32561] 0 0 0 0 0 0 0 0 0 0 ...
##  $ capital.loss  : num [1:32561] 4356 4356 4356 3900 3900 ...
##  $ hours.per.week: num [1:32561] 40 18 40 40 40 45 40 20 40 60 ...
##  $ native.country: chr [1:32561] "United-States" "United-States" "United-States" "United-States" ...
##  $ income        : num [1:32561] 0 0 0 0 0 0 0 1 0 1 ...
##  $ income_factor : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 2 1 2 ...
##  $ sex_factor    : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 2 1 1 2 ...
```

Exploratory analysis

This section is the exploration of every feature of the data set and it's connection and effect on person's annual income. I am going to group values in some columns, remove some of the features that don't have any significant impact and build a lot of plots to see how different features affect income.
Let's start with the column we are going to predict (*income*) and it's distribution. (note: 0 means less than 50K, 1 - more than 50K)

```
table(data$income)
```

```
##
##     0     1
## 24720  7841
```

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

```
##
##         0         1
## 0.7591904 0.2408096
```

So 3/4 of people in the data set earn less than 50K$ a year.
Now let's look at the correlation between income and other numerical features.

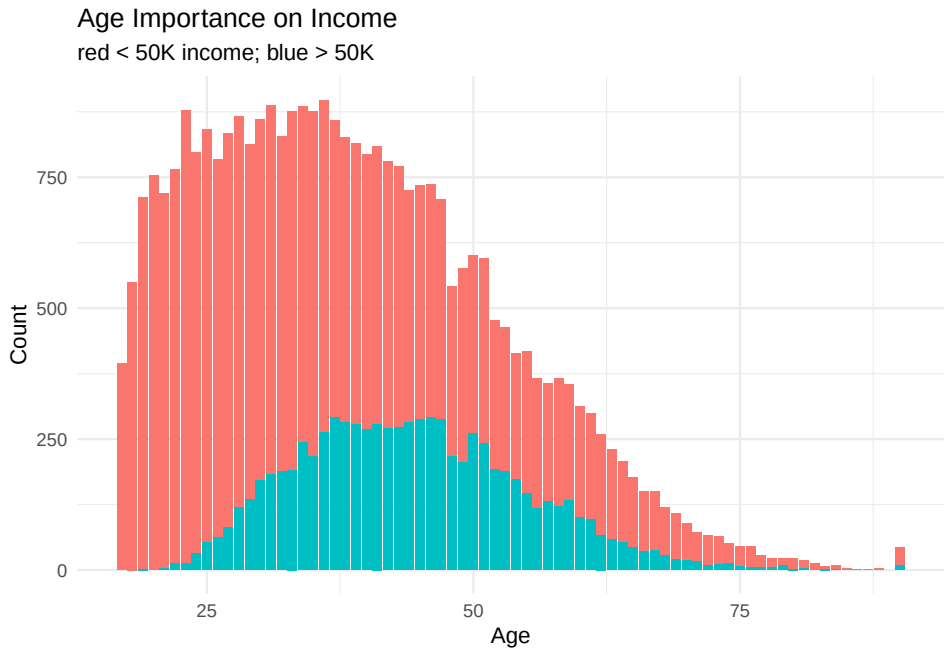
```
cor(data[,sapply(data, is.numeric)])[,"income"]
```

```
##            age            sex   capital.gain   capital.loss hours.per.week
##      0.2340371      0.2159802      0.2233288      0.1505263      0.2296891
##         income
##      1.0000000
```

We can see that all the numerical features except *capital.loss* have approximately the same impact on income. Later the *capital.loss* feature will be removed after we look at it closely. We can't see the correlation between income and character's features, but for that I am going to build plots. So now let's explore each feature separately.

Age Feature

Let's start with the age feature. Next plot shows the distribution of age feature with income division (note: red color represents people who have less than 50K annual income).

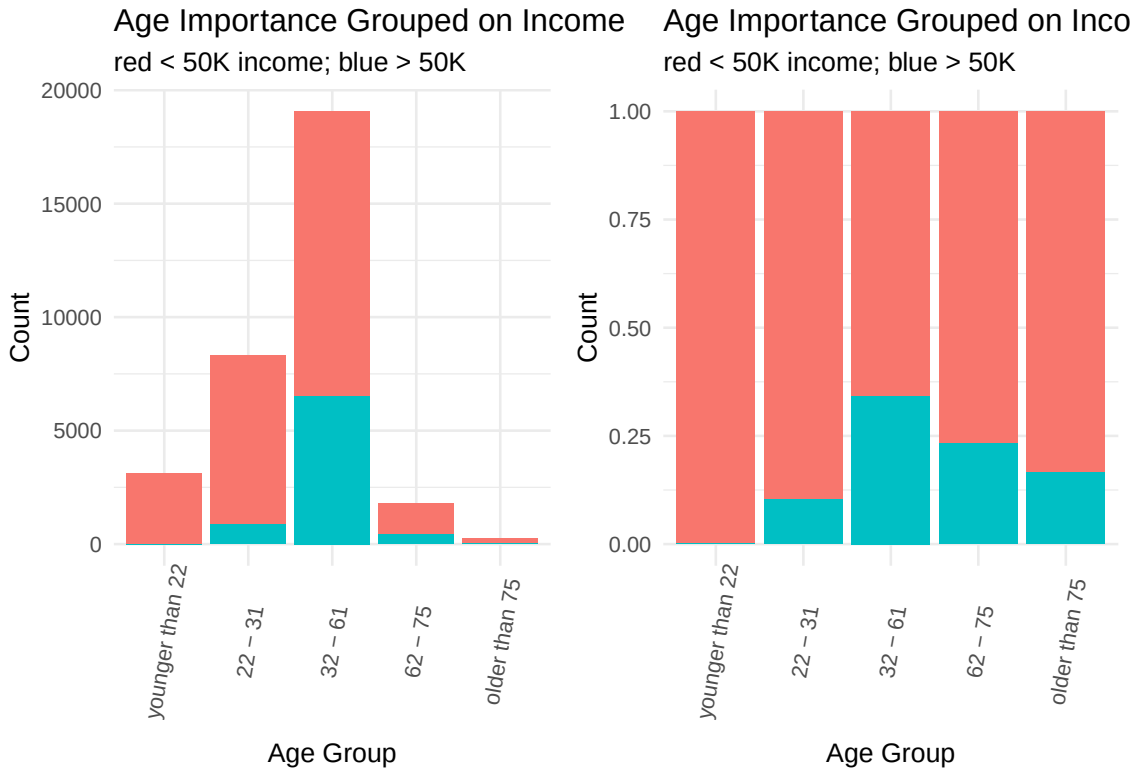


This plot shows only the amount but it doesn't show the proportion, so let's look at the next plot with proportions (for this I changed *position* to *"fill"* in *geom_histogram()*)

We can see that most people who earn more than 50K are between 30 and 60 years as these are the best working years for people. Also these plots show that young people are unlikely to earn much as they are just starting their careers and also there is a weird distribution for older people. Generally these plots can help us to group people who have roughly the same income.

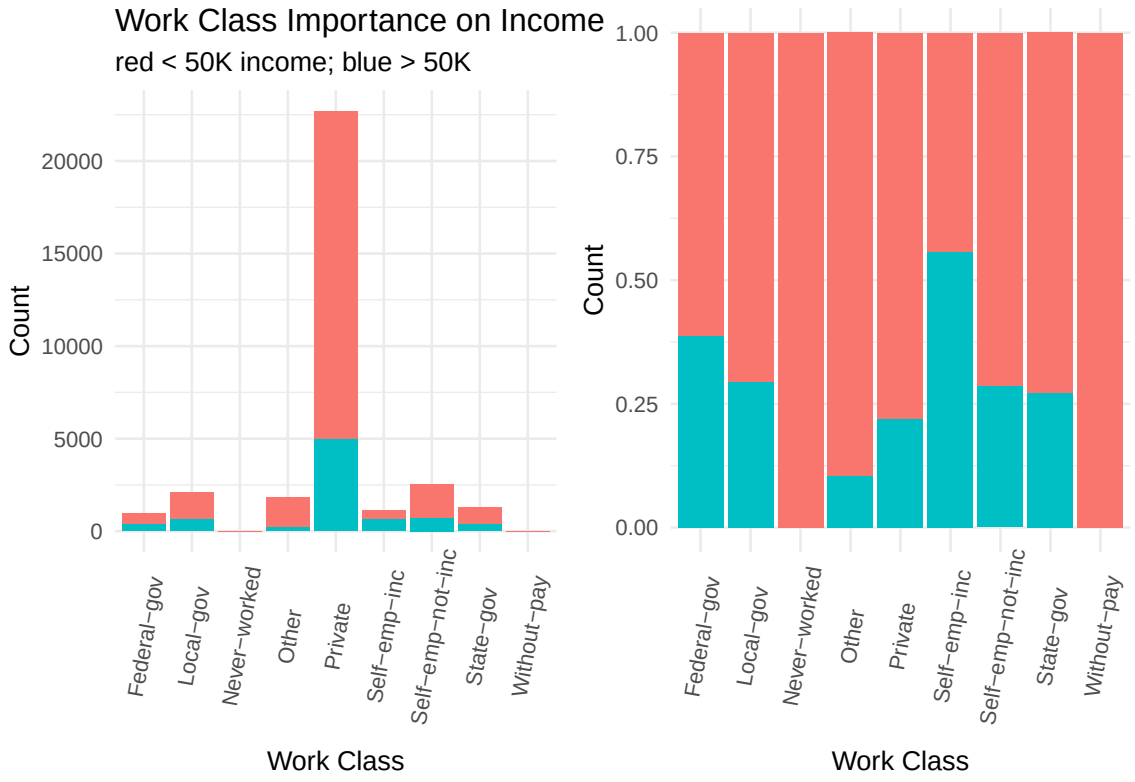
```
data <- data %>% add_column(age_factor = NA)
data$age_factor[data$age <= 21] <- "younger than 22"
data$age_factor[data$age > 21 & data$age <= 31] <- "22 - 31"
data$age_factor[data$age > 31 & data$age <= 61] <- "32 - 61"
data$age_factor[data$age > 61 & data$age <= 75] <- "62 - 75"
data$age_factor[data$age > 75] <- "older than 75"
data$age_factor <- factor(data$age_factor)
# relevel to make younger than 22 first
data$age_factor <- relevel(data$age_factor, "younger than 22")
```

Here is the distribution of created age groups. Next plots show us that people who are younger than 22 years old are likely to earn less than 50K a year which is quite logical and that people around 32-61 years earn more than other groups.



Workclass Feature

Workclass feature has 9 unique values.



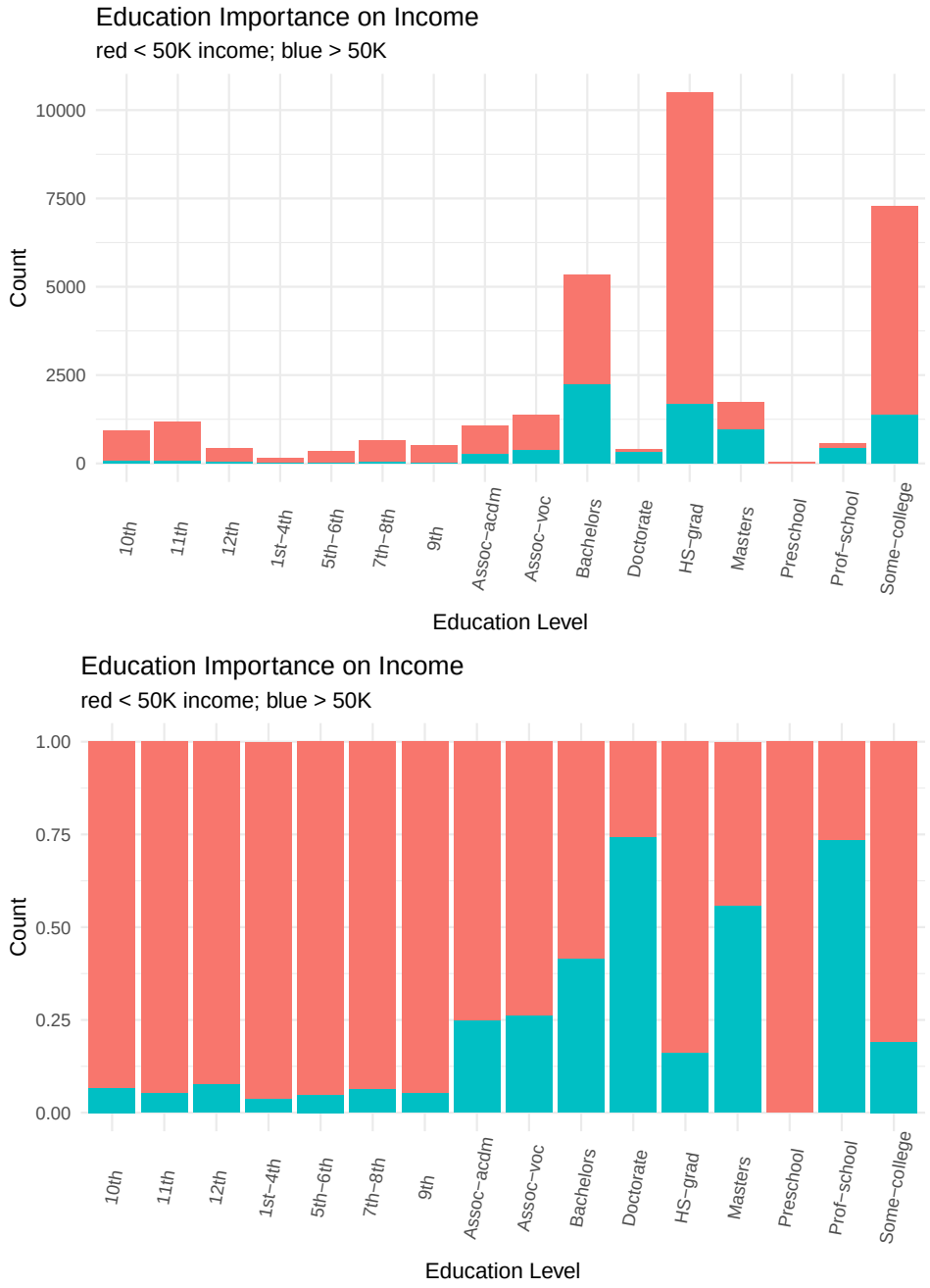
Looking at these two plots we can join *Never-worked* and *Without-pay* values as they both have the same income distribution and basically mean the same thing.

```
data$workclass[data$workclass == "Without-pay"] <- "Never-worked"
data$workclass <- factor(data$workclass)
```

This feature doesn't have significant impact on income as most of the records are in *Private* group (22696 rows) but still it's not that useless to remove it.

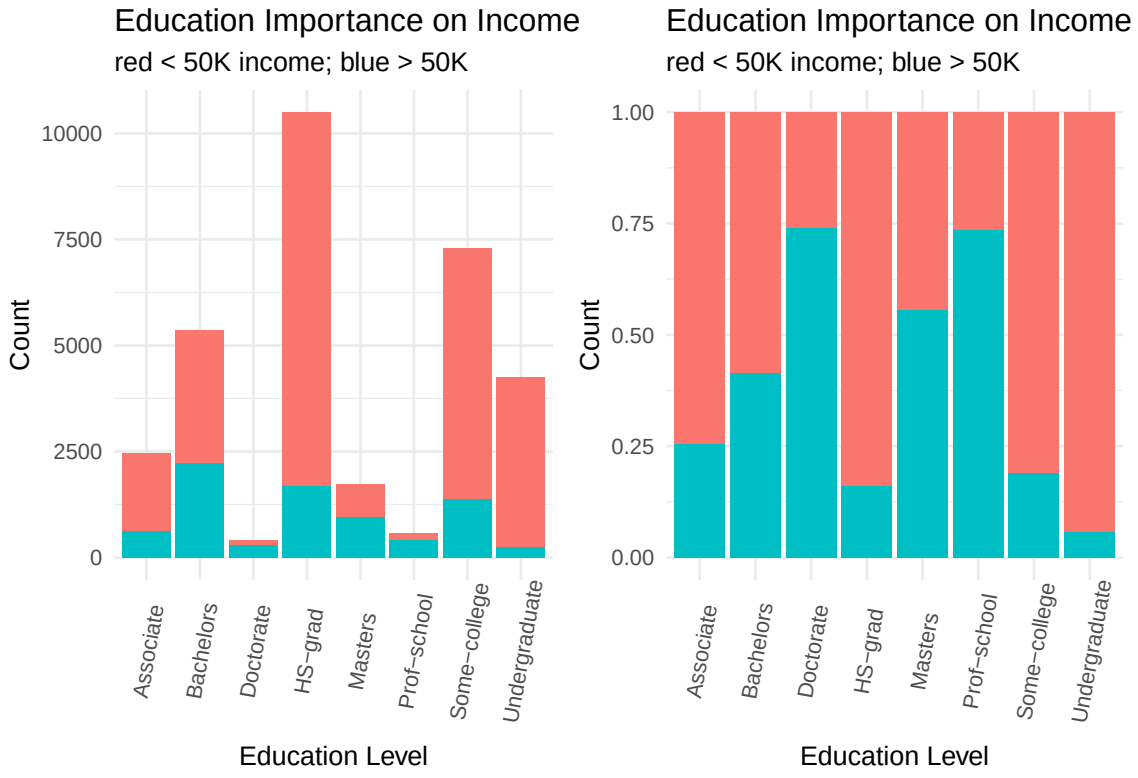
Education Feature

Now let's look at the education feature. It has 16 unique values.



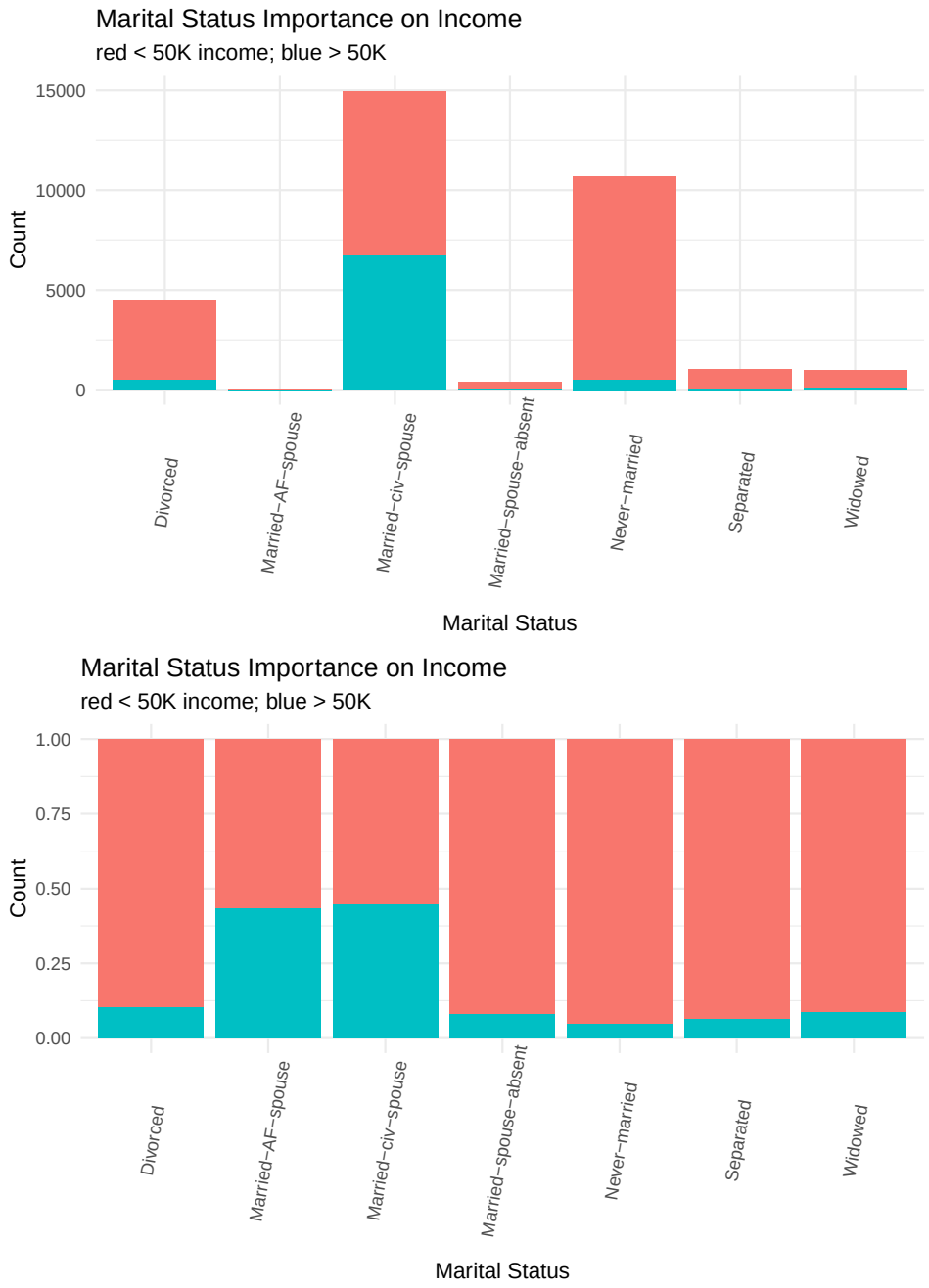
There are some similar values that can be joined together to make the modelling easier later. So let's create new group *Undergraduate* with *10th*, *11th*, *12th*, *1st-4th*, *5th-6th*, *7th-8th*, *9th*, *Preschool* values and also join *Assoc-acdm* and *Assoc-voc* into one group called *Associate*. (note: I'm not very familiar with the US education system so I might have misunderstood some of the terms, but all the transformation were made by intuition)

```
undergrad <- c("10th","11th","12th","1st-4th","5th-6th","7th-8th","9th","Preschool")
data$education[data$education %in% undergrad] <- "Undergraduate"
data$education[data$education %in% c("Assoc-acdm","Assoc-voc")] <- "Associate"
data$education <- factor(data$education)
```



Education feature is really important as the level of your education really helps to get better job and we can see it from the plots. People with Doctorate, Masters or Prof-school Degree have much higher income on average than people who only finished High-School or haven't finished even that.

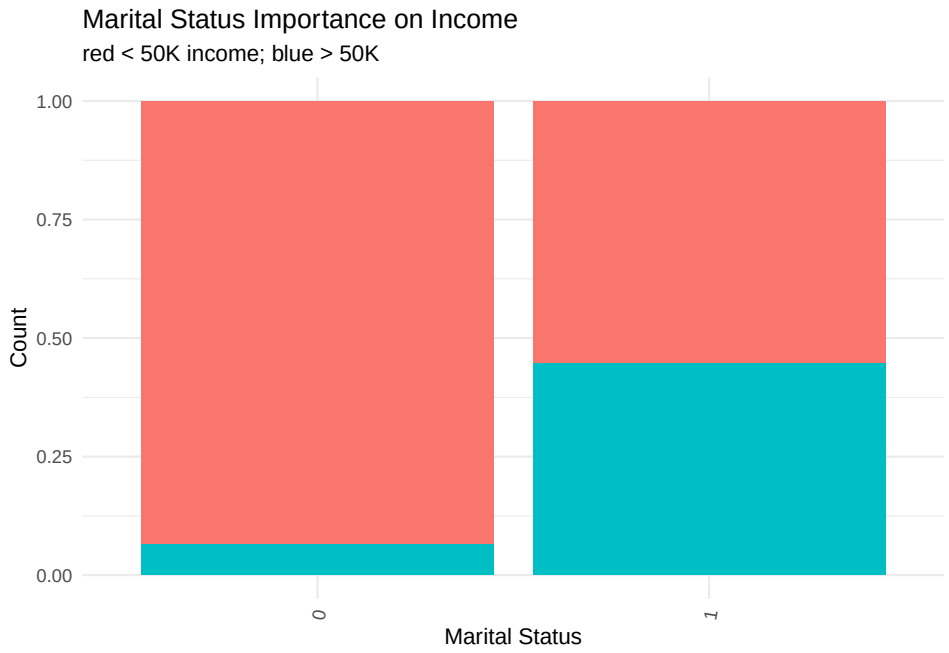
Marital Status Feature



From these plots we can see that there is 1 outlier (*Married-AF-spouse*) and some quite similar values. My idea is to make this feature binary with two values: married or not. (note: I'll consider *Married-spouse-absent* and *Separated* as not married because of similar income distribution)

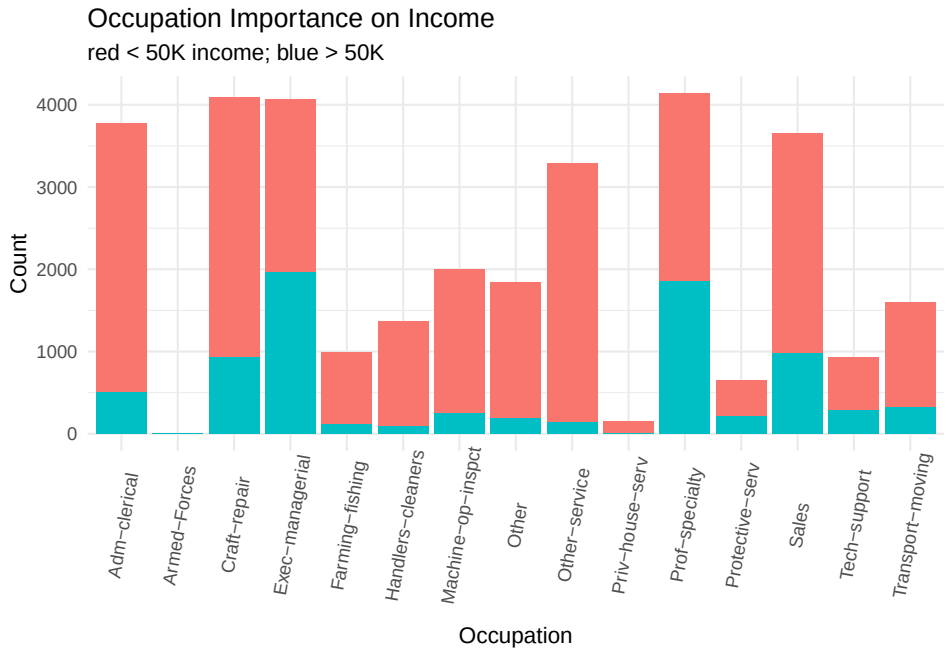
```
data$marital.status[data$marital.status %in% c("Married-civ-spouse", "Married-AF-spouse")] <- 1
data$marital.status[data$marital.status != 1] <- 0
data$marital.status <- factor(as.numeric(data$marital.status))
```

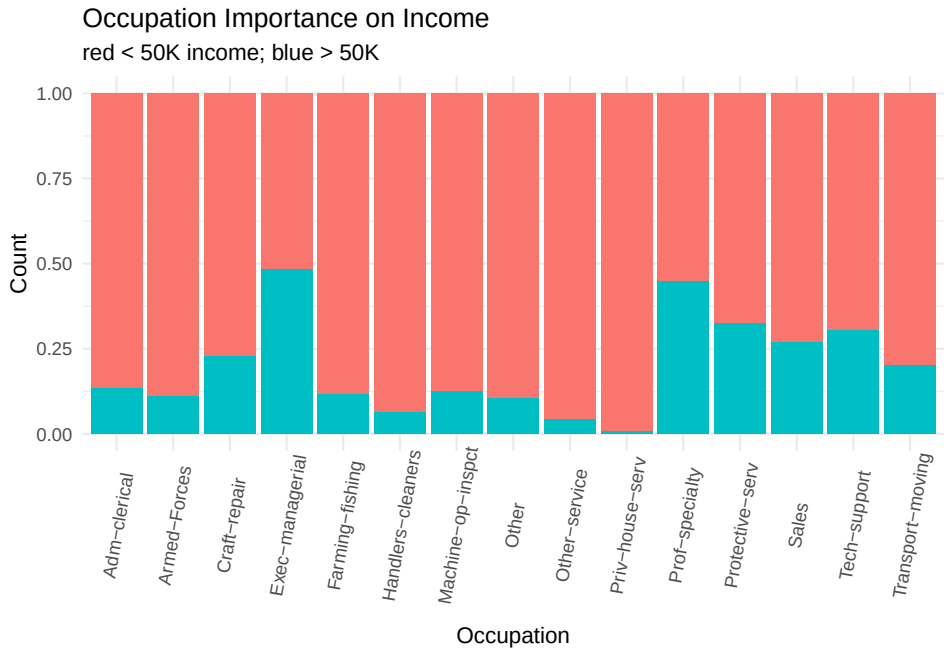
Here is the distribution of marital status after grouping. As we can see married people on average earn more, so this feature has significant impact on income and we should use it for predictive analysis.



Occupation feature

This feature represents the occupation of a person and has 15 unique values.



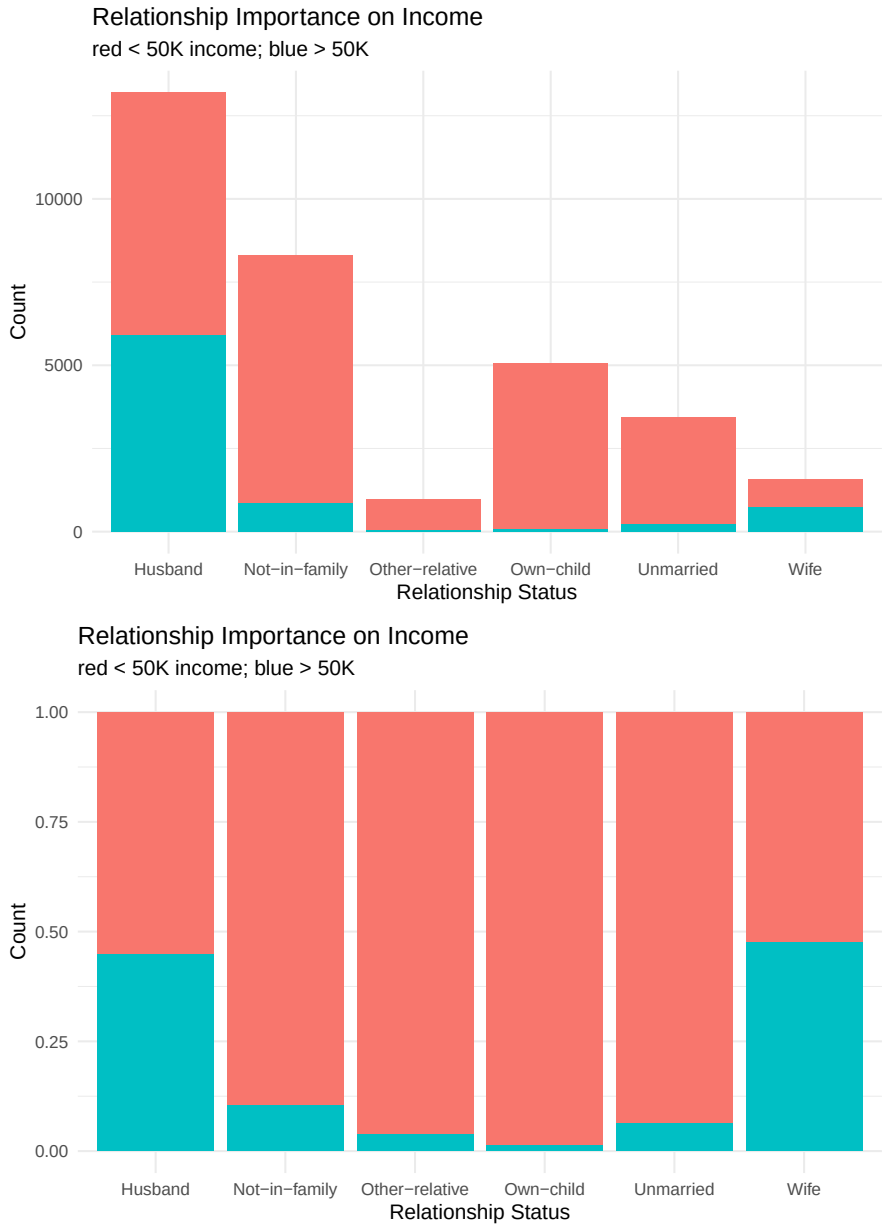


This feature is quite important as different jobs are paid differently and it has effect on income. There is nothing to group or change so let's just factor this feature.

```
data$occupation <- factor(data$occupation)
```

Relationship Feature

This feature represents the relationship status of a person and has 6 unique values.

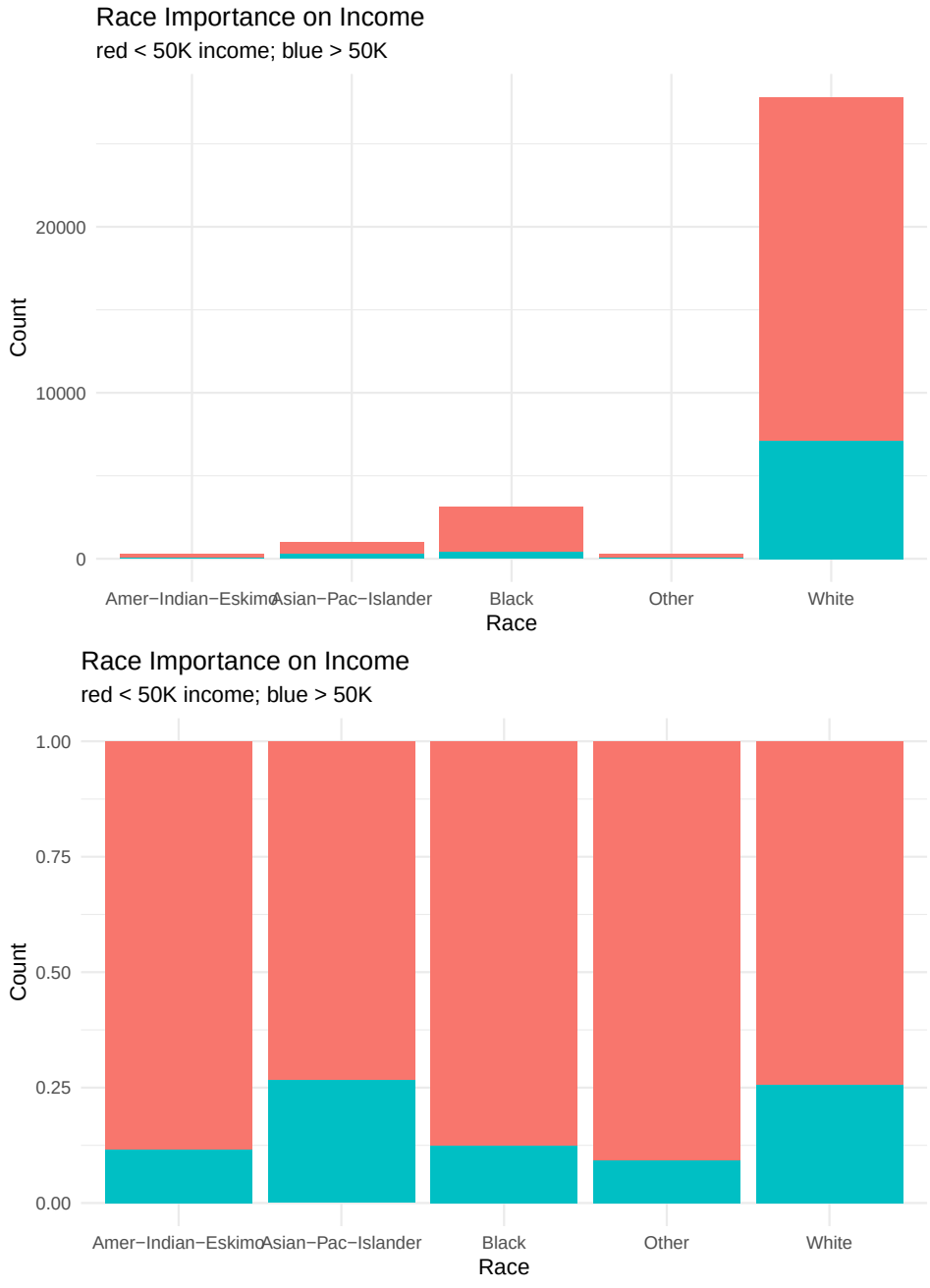


There is nothing to group or change so let's just factor this feature. As well as with the marital feature the relationship status has value on the income and it shows that married people earn much more than not married ones. Maybe we can drop one of these (*marital_status* or *relationship*) when conducting predictive analysis as they are quite similar.

```
data$relationship <- factor(data$relationship)
```

Race feature

This feature has 5 unique values.

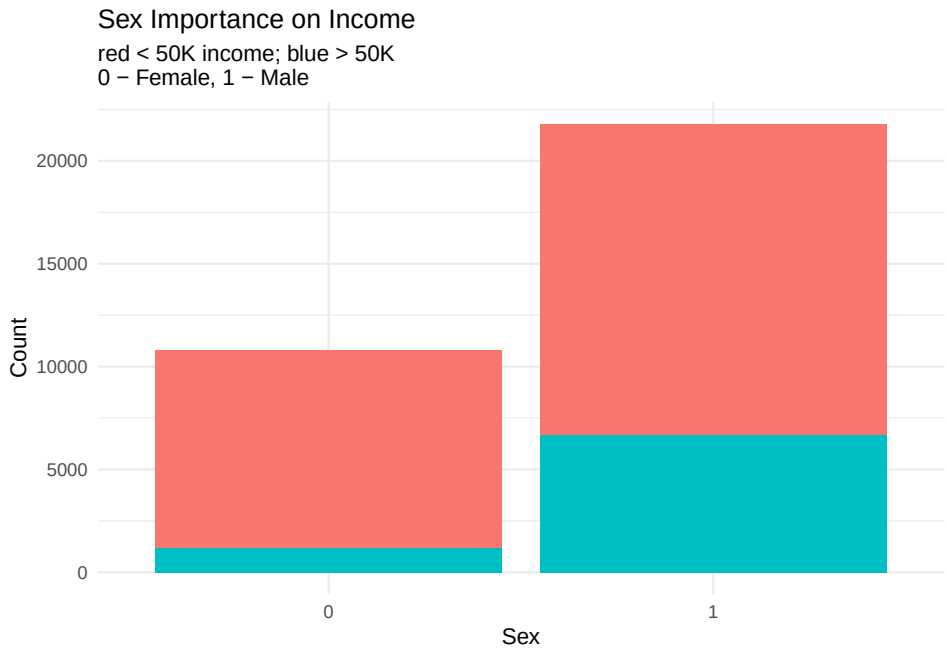


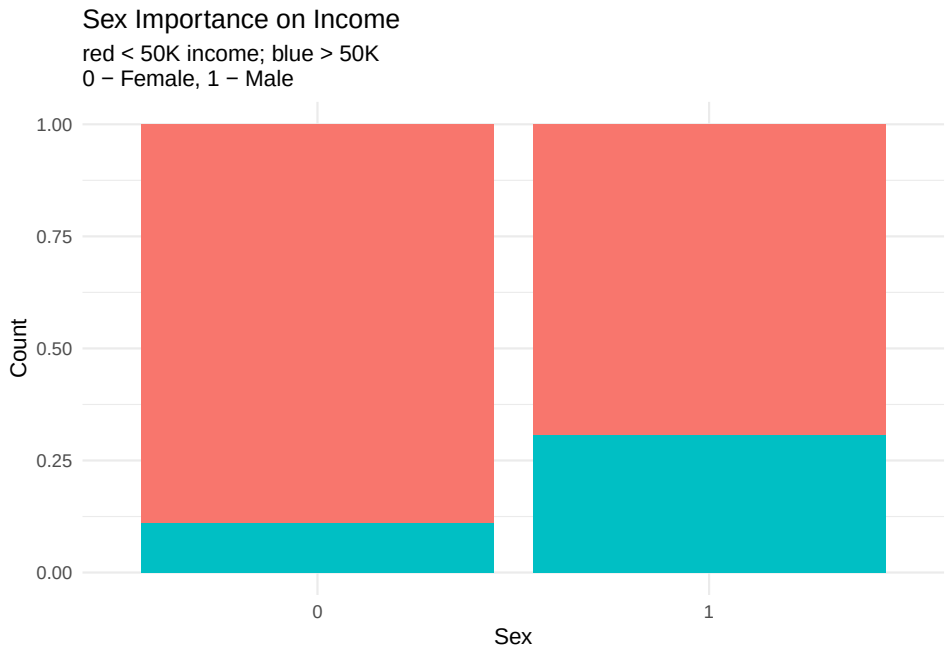
From these plots we can see that there is one main group (*White*) with 27816 rows and others that can be considered as outliers (because of small amount of records). So this feature has no significant value to us and we can just drop it.

```
data <- select(data, -race)
```

Sex factor

We have already made this feature binary so let's see it's effect on income. (note: 0 means female)

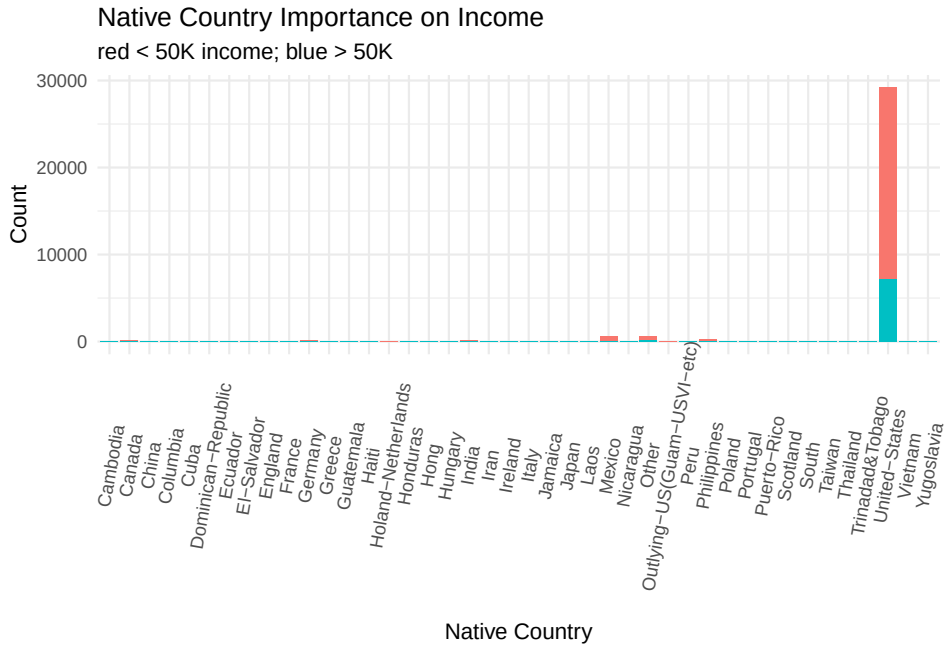


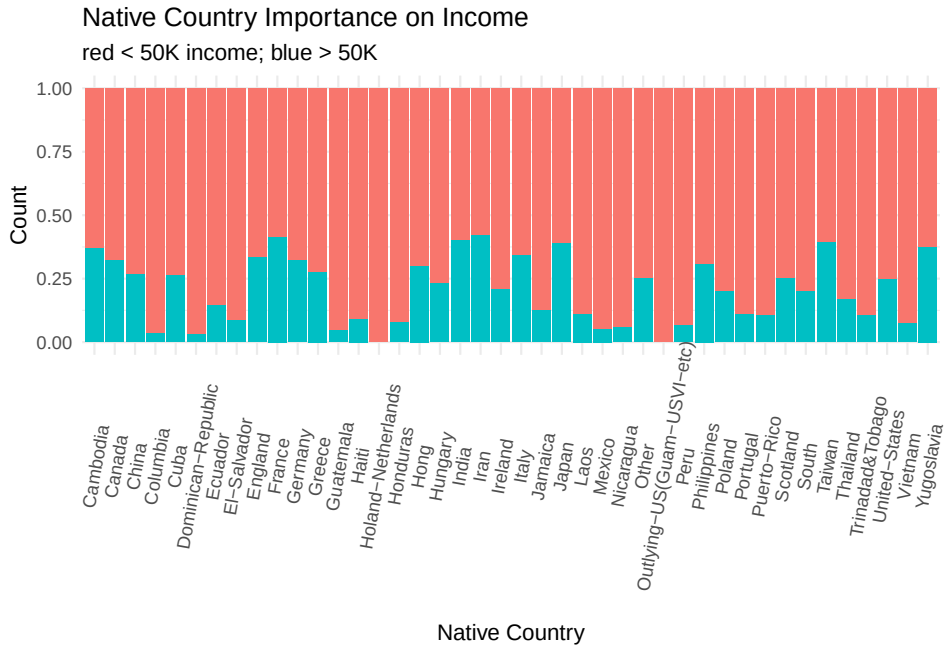


We can see that this feature is valuable as men earn more than women on average.

Native Country Feature

There are 42 unique countries.



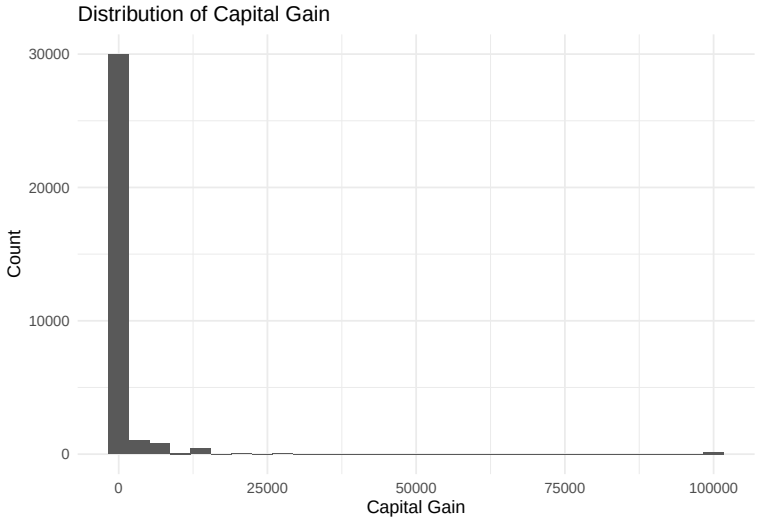


Most of the people in the data set are from US (29170 rows) so there is no significant value in this feature so we can just drop it.

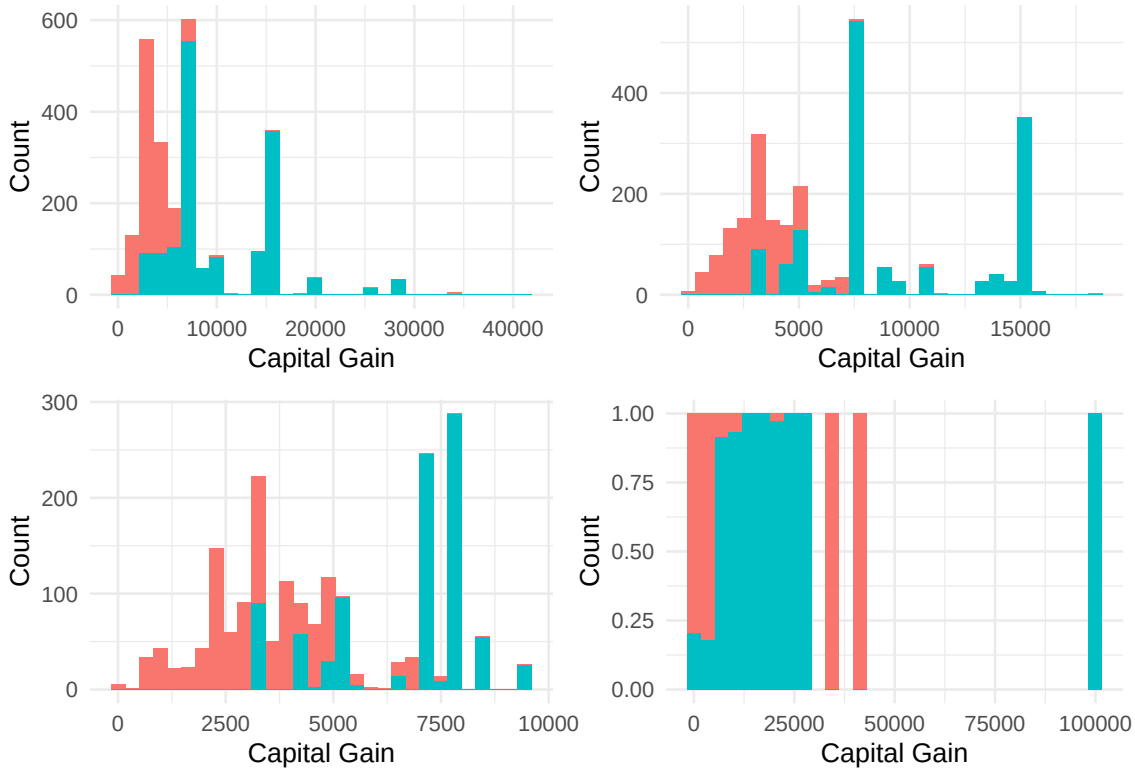
```
data <- select(data, -native.country)
```

Capital Gain Feature

Let's see the distribution of capital gain feature. Supposedly, it has a great predictive power as the more money the person gets outside of their job (I believe this is what this feature is) the more chances there are that this person earns more than 50K a year.



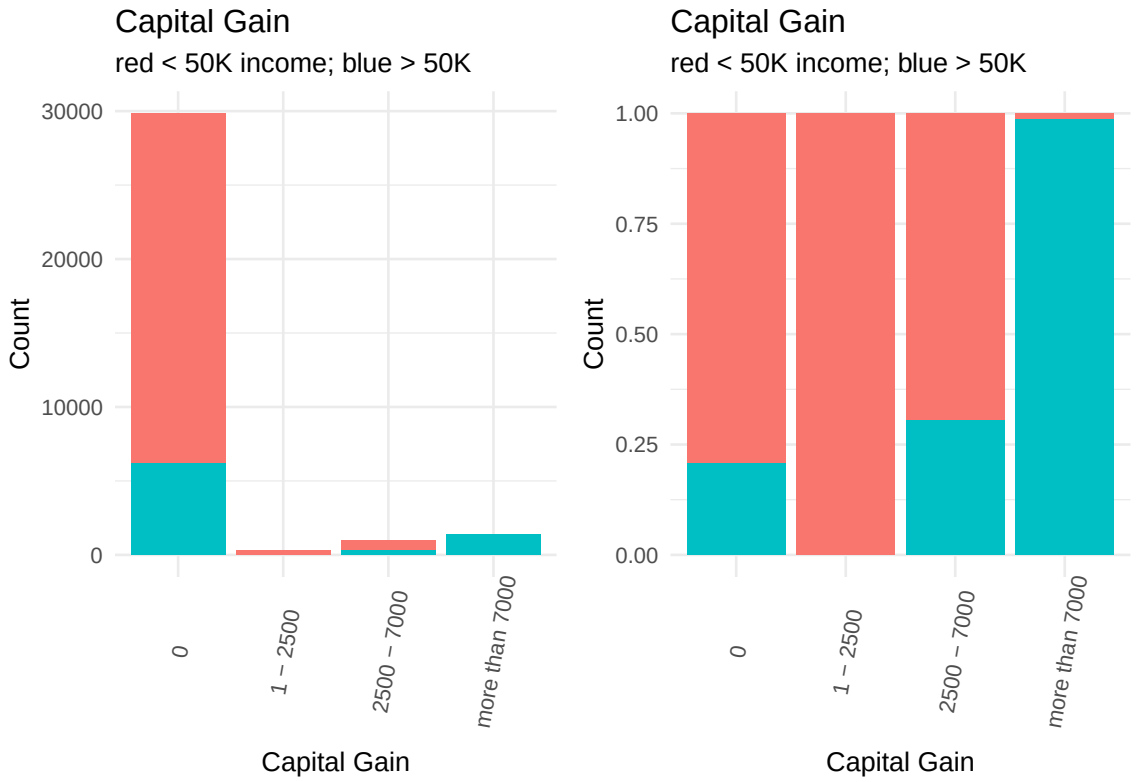
There is one outlier ($capital.gain = 99999$) this is probably some mistake in the data. Now let's look closer at the distribution of capital gain with division by income.



We can divide capital gain feature into 4 groups as it has significant value.

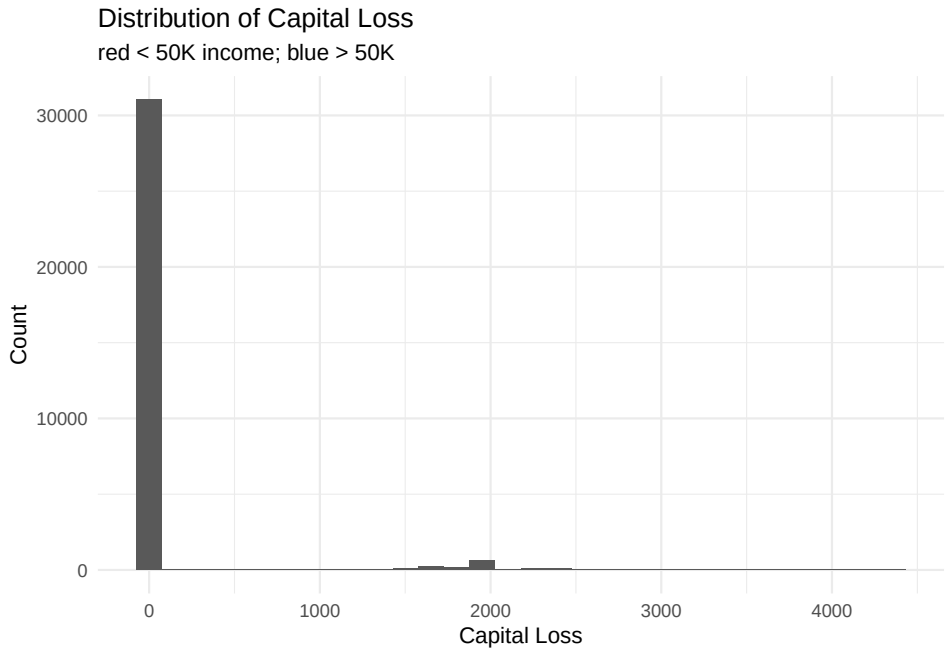
```
data <- data %>% add_column(capital.gain_factor = NA)
data$capital.gain_factor[data$capital.gain == 0] <- "0"
data$capital.gain_factor[data$capital.gain > 0 & data$age <= 2500] <- "1 - 2500"
data$capital.gain_factor[data$capital.gain > 2500 & data$age <= 7000] <- "2500 - 7000"
data$capital.gain_factor[data$capital.gain > 7000] <- "more than 7000"
data$capital.gain_factor <- factor(data$capital.gain_factor)
```

Let's see the distribution after we split the feature. We can see that people who have more than 7000$ of capital gain are almost 100% sure earning more than 50K$ a year. Even while most of the people have 0 capital gain, this feature is quite valuable because we can easily predict some people who earn more than 50K a year.



Capital Loss Feature

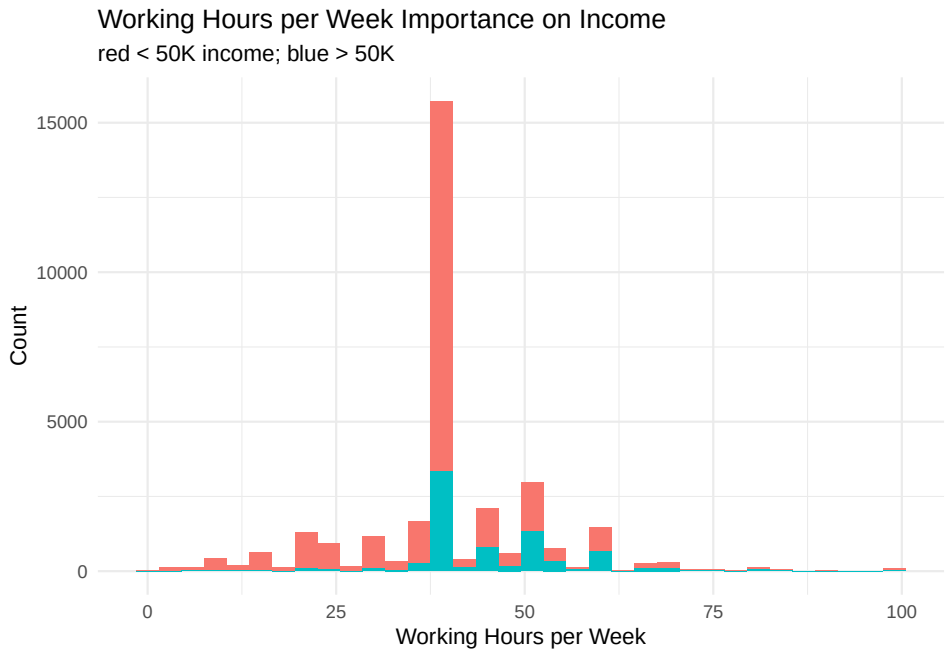
Capital loss feature has the lowest correlation coefficient with income and most of the values of this feature are zeroes as we can see from the distribution plot, so there is no significant value from this feature and we can drop it.

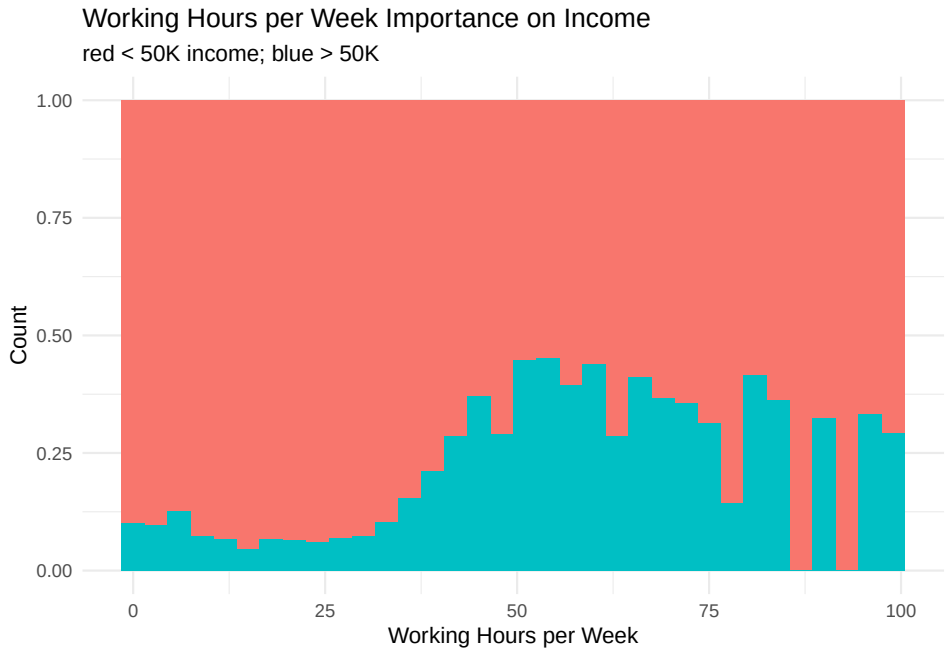



```
data <- select(data, -capital.loss)
```

Working Hours Feature

This feature must be quite valuable to us as the more people work the more they earn, don't they?





As we can see the more the person works the more money he gets, but this is true until around 60 working hours per week. After this number income becomes stable. This is probably because people who work a lot of time work on 2 or 3 different part-time low-paid jobs. Let's split this feature into 4 groups:

```
data <- data %>% add_column(hours.per.week_factor = NA)
data$hours.per.week_factor[data$hours.per.week < 35] <- "0-35"
data$hours.per.week_factor[data$hours.per.week >= 35 & data$hours.per.week < 50] <- "35-49"
data$hours.per.week_factor[data$hours.per.week >= 50 & data$hours.per.week <= 60] <- "50-60"
data$hours.per.week_factor[data$hours.per.week > 60] <- "61-100"
data$hours.per.week_factor <- factor(data$hours.per.week_factor)
```

Here is the distribution of groups.

As we can see people who work 50-60 hours per week earn more, and there is no need to work too much as the payments for such people are even lower than for people who work 50-60 hours. Also people who work less than 35 hours earn the least, as they probably have a part-time job.

Results of Exploratory Analysis

Last thing to do in our exploratory analysis is to remove columns that we don't need.

```
columns_to_drop <- c("age","capital.gain","sex","income","hours.per.week")
data <- data %>% select(-all_of(columns_to_drop))
```

After conducting exploratory analysis there are only 10 columns left, all factors. Here is the structure of the data set now.

```
str(data, give.attr=F)
```

```
## tibble [32,561 x 10] (S3: tbl_df/tbl/data.frame)
##  $ workclass            : Factor w/ 8 levels "Federal-gov",..: 4 5 4 5 5 5 5 8 1 5 ...
##  $ education            : Factor w/ 8 levels "Associate","Bachelors",..: 4 4 7 8 7 4 8 3 4 7 ...
##  $ marital.status       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ occupation           : Factor w/ 15 levels "Adm-clerical",..: 8 4 8 7 11 9 1 11 11 3 ...
##  $ relationship         : Factor w/ 6 levels "Husband","Not-in-family",..: 2 2 5 5 4 5 5 3 2 5 ...
##  $ income_factor        : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 2 1 2 ...
##  $ sex_factor           : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 2 1 1 2 ...
##  $ age_factor           : Factor w/ 5 levels "younger than 22",..: 5 5 4 3 3 3 3 4 4 3 ...
##  $ capital.gain_factor  : Factor w/ 4 levels "0","1 - 2500",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ hours.per.week_factor: Factor w/ 4 levels "0-35","35-49",..: 2 1 2 2 2 2 2 1 2 3 ...
```

So, we have explored every feature of the data set, defined important ones and removed useless ones and grouped every feature to make the modelling easier.

Model Building

Now let's build some models. I am are going to predict the income feature which is binary (either 0 or 1), and the best models for that are logistic regression, decision tree, random forest and knn method.

Split the Data

First we should split the data into train and test sets.

```
test_index <- createDataPartition(y = data$income_factor, times = 1, p = 0.1, list = FALSE)
train <- data[-test_index,]
test <- data[test_index,]
```

Logistic Regression

```
fitglm <- glm(income_factor~., data=train, family="binomial")
p_hat_logit <- predict(fitglm, newdata = test, type = "response")
y_hat_logit <- ifelse(p_hat_logit > 0.5, 1, 0) %>% factor

# accuracy
glm_accuracy <- confusionMatrix(y_hat_logit, test$income_factor)$overall[["Accuracy"]]
```

Table 1: Prediction Accuracy for Different Models

Model	Accuracy
Logistic Regression	0.86

Decision Tree

```
fitrpart <- train(income_factor ~ ., method = "rpart", data = train)
# accuracy
rpart_accuracy <- confusionMatrix(predict(fitrpart, test), test$income_factor)$overall["Accuracy"]
```

Table 2: Prediction Accuracy for Different Models

Model	Accuracy
Logistic Regression	0.860
Decision Tree	0.835

Random Forest

```
fitrf <- randomForest(income_factor~., data=train)

rf_accuracy <- confusionMatrix(predict(fitrf, test), test$income_factor)$overall["Accuracy"]
```

Table 3: Prediction Accuracy for Different Models

Model	Accuracy
Logistic Regression	0.860
Decision Tree	0.835
Random Forest	0.866

KNN

This method takes a lot of time to run, so I will use the accuracy that I calculated separately on my computer (that took me more than 20 minutes) but I will leave the code here.

```
#fitknn <- train(income_factor~., data=train, method="knn")

#accuracy <- confusionMatrix(predict(fitknn, test), test$income_factor)$overall["Accuracy"]
knn_accuracy <- 0.842
```

Table 4: Prediction Accuracy for Different Models

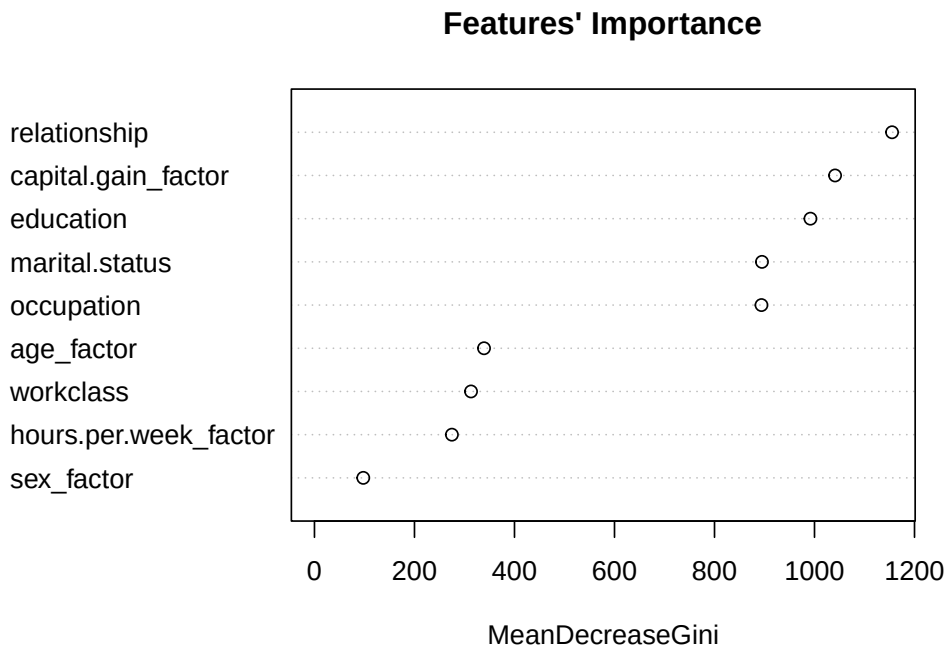
Model	Accuracy
Logistic Regression	0.860
Decision Tree	0.835
Random Forest	0.866
KNN Model	0.842

So, the best model for income prediction is random forest which gives us 86.613448% accuracy and the worst is decision tree.
Here are some other parameters of random forest model

Table 5: Parameters of Random Forest Model

Parameters	Value
Accuracy	0.866
Sensitivity	0.937
Specificity	0.642
Balanced Accuracy	0.790
Kappa	0.613

And also here is the plot of the most important variables.



The most important variables as we can see from the plot above are: *relationship*, *education*, *capital.gain_factor*, *marital.status* and *occupation*. The least important are: *age_factor*, *workclass*, *hours.per.week_factor* and *sex_factor*. These insights are quite surprising as I didn't think the *relationship* feature would be so important and also I hoped that *age_factor* and *hours.per.week_factor* would be more valuable.

Conclusions

After analyzing the 'Adult Census Income' data set I have explored every feature and its effect on annual income and built a model to predict whether a person would get more or less than 50 thousand US dollars of income a year. The analysis showed that the most important things in predicting the income are person's relationship status, education and place of work, while other factors have much lower impact. Also while visualizing data I have made some interesting insights, for example, that people who work more than 60 hours a week earn less than people who work 50-60 hours on average.

Also I have learned a lesson that before starting working on a project and analyzing data one should set a specific goal for a project. For example, to reach specific accuracy, or to get to specific sensitivity or specificity value without any significant drop in balanced accuracy and so on. This is very important because the job of a data scientist is not only about randomly analyzing data, it's also about answering questions and reaching the predefined objectives.

In general, while working on this project I have mastered a lot of skills, starting from data wrangling and visualizing up to building predictive models. I have enjoyed analyzing the data set as I have finally had a chance to do it on my own. Also, while choosing the data set for this project I have discovered Kaggle and I am looking forward to join Data Science community and work on other different projects. In the end, I would like to thank edX and HarvardX teams for providing such high-quality Data Science program and for helping people from Ukraine like myself during these bad times.