

# Loan Sanction Amount Prediction Project

## Harvardx's Data Science Professional Certificate PH125.9x

Otman Cherrati

July 25, 2021

## Contents

|          |  |           |
|----------|--|-----------|
| <b>1</b> | <b>Intorduction :</b>                        | <b>2</b>  |
| 1.1      | Object :                                     | 2         |
| 1.2      | Packages used :                              | 2         |
| 1.3      | The Session Information :                    | 2         |
| 1.4      | Load the loan data set :                     | 3         |
| 1.5      | Dataset description .                        | 3         |
| <b>2</b> | <b>Data Exploration :</b>                    | <b>4</b>  |
| 2.1      | Initial exploration :                        | 4         |
| 2.2      | Dealing with nas                             | 6         |
| 2.3      | Exploring continuous variables               | 10        |
| 2.4      | Categorical features exploration             | 21        |
| <b>3</b> | <b>Data processing :</b>                     | <b>35</b> |
| 3.1      | Drop no needed features in models building   | 35        |
| 3.2      | Binarize categorical features                | 35        |
| 3.3      | Scale continuous features except the outcome | 35        |
| 3.4      | Split data into train and test set           | 35        |
| <b>4</b> | <b>Building models</b>                       | <b>36</b> |
| 4.1      | Linear regression                            | 36        |
| 4.2      | Rpart model                                  | 36        |
| 4.3      | Random forest                                | 37        |
| 4.4      | Xgboost model                                | 38        |
| <b>5</b> | <b>Conclusion :</b>                          | <b>40</b> |

# 1 Intorduction :

This report is the second part of the final capstone course of HarvardX Data Science Professional certificate taught by the famous professor Rafael Irizzary. The report is on loan sanction amount prediction.

Buying a house requires a lot of careful planning. Once you have finalized your budget and the house that you want to buy, you must ensure that you have sufficient funds to pay the seller.

With rising property rates, most people avail home loans to buy their dream houses. The bank only lends up to 80% of the total amount based on a person's finances (salary, outgoing expenses, existing loans, etc.). You will need to make the rest of the payment yourself after the bank tells you how much they can lend.

## 1.1 Object :

Our goal in this analysis is to predict the loan amount that can be sanctioned to customers who have applied for a home loan using the features provided in the dataset (by a XYZ Bank). We will do this by building a model capable of generating predictions of the loan amount to be approved for a client using a training set. We will start by taking a look on the dataset and then explore the different variables to see if there is any trends that can help us in the analysis. After cleaning the data we will start building and testing our models. We will discuss the result obtained by the different models and chose the one that performed the best. The evaluation is based on the RMSE - Root Mean Squared Error built under the caret package.

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$$

## 1.2 Packages used :

**tidyverse**  
**data.table**  
**caret**  
**Xgboost**  
**corrplot**  
**knitr**

## 1.3 The Session Information :

The output of sessionInfo() is placed here for reproducibility purposes.

```
## R version 4.0.5 (2021-03-31)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 19042)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United States.1252
## [2] LC_CTYPE=English_United States.1252
## [3] LC_MONETARY=English_United States.1252
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.1252
```

```

##
## attached base packages:
## [1] stats      graphics   grDevices utils      datasets   methods    base
##
## other attached packages:
## [1] xgboost_1.4.1.1  corrplot_0.88     knitr_1.33       data.table_1.14.0
## [5] caret_6.0-86     lattice_0.20-41  forcats_0.5.1   stringr_1.4.0
## [9] dplyr_1.0.5      purrr_0.3.4      readr_1.4.0     tidyverse_1.3.1
## [13] tibble_3.1.1     ggplot2_3.3.5     tidyverse_1.3.1
##
## loaded via a namespace (and not attached):
## [1] httr_1.4.2        jsonlite_1.7.2    splines_4.0.5
## [4] foreach_1.5.1     prodlim_2019.11.13 modelr_0.1.8
## [7] assertthat_0.2.1  stats4_4.0.5     cellranger_1.1.0
## [10] yaml_2.2.1        ipred_0.9-11     pillar_1.6.0
## [13] backports_1.2.1  glue_1.4.2      PROC_1.17.0.1
## [16] digest_0.6.27    rvest_1.0.0     colorspace_2.0-0
## [19] recipes_0.1.16   htmltools_0.5.1.1 Matrix_1.3-2
## [22] plyr_1.8.6       timeDate_3043.102 pkgconfig_2.0.3
## [25] broom_0.7.6     haven_2.4.1     scales_1.1.1
## [28] gower_0.2.2     lava_1.6.9      generics_0.1.0
## [31] ellipsis_0.3.1   withr_2.4.2     nnet_7.3-15
## [34] cli_2.5.0        survival_3.2-10 magrittr_2.0.1
## [37] crayon_1.4.1    readxl_1.3.1    evaluate_0.14
## [40] fs_1.5.0         fansi_0.4.2     nlme_3.1-152
## [43] MASS_7.3-53.1   xml2_1.3.2     class_7.3-18
## [46] tools_4.0.5     hms_1.0.0      lifecycle_1.0.0
## [49] munsell_0.5.0   reprex_2.0.0    compiler_4.0.5
## [52] rlang_0.4.10    grid_4.0.5     iterators_1.0.13
## [55] rstudioapi_0.13  rmarkdown_2.7   gtable_0.3.0
## [58] ModelMetrics_1.2.2.2 codetools_0.2-18 DBI_1.1.1
## [61] reshape2_1.4.4   R6_2.5.0      lubridate_1.7.10
## [64] utf8_1.2.1      stringi_1.5.3  Rcpp_1.0.6
## [67] vctrs_0.3.7     rpart_4.1-15   dbplyr_2.1.1
## [70] tidyselect_1.1.0 xfun_0.22

```

## 1.4 Load the loan data set :

## 1.5 Dataset description

This data set contains the information about the customers who applied for a home loan during an undefined period of time in an unknown bank. It was uploaded from kaggle on “<https://www.kaggle.com/zyper26/sanction-loan>”.

The downloaded zip file contains three data sets : “train”, “test” and “submission” but we will be using just the train file since the “test” data doesn’t contain the output variable and is used only for competition on kaggle just like the “submission” file.

We can get the zip file using this link. “<https://www.kaggle.com/zyper26/sanction-loan/download>”

## 2 Data Exploration :

### 2.1 Initial exploration :

Check the top six rows and six columns of the data set :

| Customer ID | Name              | Gender | Age | Income (USD) | Income Stability |
|-------------|-------------------|--------|-----|--------------|------------------|
| C-36995     | Frederica Shealy  | F      | 56  | 1933.05      | Low              |
| C-33999     | America Calderone | M      | 32  | 4952.91      | Low              |
| C-3770      | Rosetta Verne     | F      | 65  | 988.19       | High             |
| C-26480     | Zoe Chitty        | F      | 65  | NA           | High             |
| C-23459     | Afton Venema      | F      | 31  | 2614.77      | Low              |
| C-17688     | Polly Crumpler    | F      | 60  | 1234.92      | Low              |

### Inspect the dataset for dimensions

The loan data contains 30000 rows and 24 columns

Table 2: Number of rows and columns

| rows  | columns |
|-------|---------|
| 30000 | 24      |

### Check the structure of the data

Inspect the dataframe for the class of the different variables

```
## Classes 'data.table' and 'data.frame': 30000 obs. of 24 variables:
## $ Customer ID : chr "C-36995" "C-33999" "C-3770" "C-26480" ...
## $ Name : chr "Frederica Shealy" "America Calderone" "Rosetta Verne" "Zoe Chi
## $ Gender : chr "F" "M" "F" "F" ...
## $ Age : int 56 32 65 65 31 60 43 45 38 18 ...
## $ Income (USD) : num 1933 4953 988 NA 2615 ...
## $ Income Stability : chr "Low" "Low" "High" "High" ...
## $ Profession : chr "Working" "Working" "Pensioner" "Pensioner" ...
## $ Type of Employment : chr "Sales staff" NA NA NA ...
## $ Location : chr "Semi-Urban" "Semi-Urban" "Semi-Urban" "Rural" ...
## $ Loan Amount Request (USD) : num 72810 46837 45593 80058 113859 ...
## $ Current Loan Expenses (USD): num 241 496 172 299 491 ...
## $ Expense Type 1 : chr "N" "N" "N" "N" ...
## $ Expense Type 2 : chr "N" "Y" "Y" "Y" ...
## $ Dependents : num 3 1 1 2 NA 2 2 2 3 2 ...
## $ Credit Score : num 809 780 833 833 746 ...
## $ No. of Defaults : int 0 0 0 1 1 0 0 1 0 ...
## $ Has Active Credit Card : chr NA "Unpossessed" "Unpossessed" "Unpossessed" ...
## $ Property ID : int 746 608 546 890 715 491 227 314 241 883 ...
## $ Property Age : num 1933 4953 988 NA 2615 ...
## $ Property Type : int 4 2 2 2 4 2 1 2 4 2 ...
## $ Property Location : chr "Rural" "Rural" "Urban" "Semi-Urban" ...
## $ Co-Applicant : int 1 1 0 1 1 1 1 1 1 ...
## $ Property Price : num 119933 54791 72441 121442 208568 ...
## $ Loan Sanction Amount (USD) : num 54607 37470 36474 56041 74008 ...
## - attr(*, ".internal.selfref")=<externalptr>
```

The features of the data set are :

```
## [1] "Customer ID"           "Name"
## [3] "Gender"                 "Age"
## [5] "Income (USD)"          "Income Stability"
## [7] "Profession"             "Type of Employment"
## [9] "Location"               "Loan Amount Request (USD)"
## [11] "Current Loan Expenses (USD)" "Expense Type 1"
## [13] "Expense Type 2"         "Dependents"
## [15] "Credit Score"           "No. of Defaults"
## [17] "Has Active Credit Card" "Property ID"
## [19] "Property Age"           "Property Type"
## [21] "Property Location"     "Co-Applicant"
## [23] "Property Price"        "Loan Sanction Amount (USD)"
```

We can see that the data contains the customers names, it's obvious that they are not real names, but any data analyst should anonymize data before working on it. So we will drop the name column.

### Number of customer

We notice that all the customers in this data are unique.

```
## The number of customers is 30000
```

Count missing values in each column :

|                             | Number of NAs |
|-----------------------------|---------------|
| Customer ID                 | 0             |
| Gender                      | 53            |
| Age                         | 0             |
| Income (USD)                | 4576          |
| Income Stability            | 1683          |
| Profession                  | 0             |
| Type of Employment          | 7270          |
| Location                    | 0             |
| Loan Amount Request (USD)   | 0             |
| Current Loan Expenses (USD) | 172           |
| Expense Type 1              | 0             |
| Expense Type 2              | 0             |
| Dependents                  | 2493          |
| Credit Score                | 1703          |
| No. of Defaults             | 0             |
| Has Active Credit Card      | 1566          |
| Property ID                 | 0             |
| Property Age                | 4850          |
| Property Type               | 0             |
| Property Location           | 356           |
| Co-Applicant                | 0             |
| Property Price              | 0             |
| Loan Sanction Amount (USD)  | 340           |

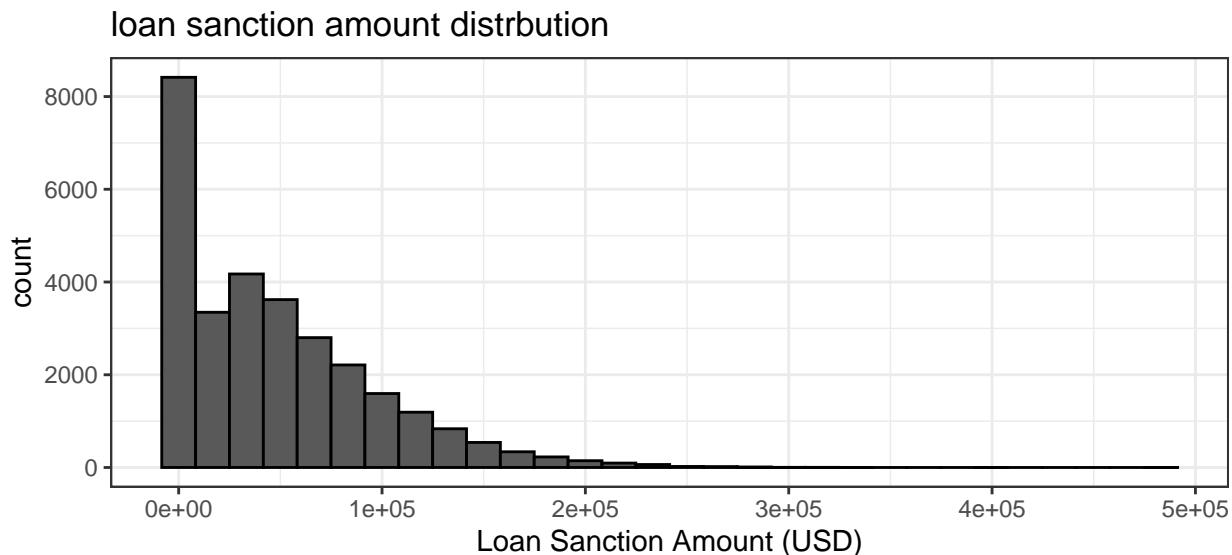
List all of the columns having missing values with percentage

|                             | percentage of NAs |
|-----------------------------|-------------------|
| Gender                      | 0.0017667         |
| Income (USD)                | 0.1525333         |
| Income Stability            | 0.0561000         |
| Type of Employment          | 0.2423333         |
| Current Loan Expenses (USD) | 0.0057333         |
| Dependents                  | 0.0831000         |
| Credit Score                | 0.0567667         |
| Has Active Credit Card      | 0.0522000         |
| Property Age                | 0.1616667         |
| Property Location           | 0.0118667         |
| Loan Sanction Amount (USD)  | 0.0113333         |

## 2.2 Dealing with nas

Since there is a lot of features with nas, we will first deal with those na's column by column, removing some and replacing the others

### 2.2.1 Loan sanction amount : the outcome variable



```
## The number of Na's is 340
## The number of values less than '0' is 338
```

There are 340 nas and 338 values less than 0 which doesn't make sense, we will remove the missing outcomes, and replace the negative ones by 0.

### 2.2.2 Gender

There is as much "Females" as "males" in the data set and the number of nas is small, so we will replace them equitably by the 2 classes "F" and "M".

| Gender | Frequency |
|--------|-----------|
| F      | 14718     |
| M      | 14890     |

```
## [1] "The number of missing values is 52"
```

### 2.2.3 Income stability

The classes low is predominant, we will replace the nas with the low class

| Income stab | Frequency |
|-------------|-----------|
| High        | 2544      |
| Low         | 25458     |

```
## The number of missing values is 1658
```

### 2.2.4 Type of employment

We have 18 different type of employment, and there is a great number of missing values most than any other class, it would be better to create a new class named “other” and assign the nas to it.

| Type of Employment    | Frequency |
|-----------------------|-----------|
| Laborers              | 5516      |
| Sales staff           | 3698      |
| Core staff            | 3189      |
| Managers              | 2464      |
| Drivers               | 1588      |
| Accountants           | 1363      |
| High skill tech staff | 1297      |
| Medicine staff        | 854       |
| Security staff        | 574       |
| Cooking staff         | 561       |
| Cleaning staff        | 338       |
| Private service staff | 337       |
| Secretaries           | 158       |
| Low-skill Laborers    | 154       |
| Waiters/barmen staff  | 148       |
| Realty agents         | 85        |
| IT staff              | 77        |
| HR staff              | 71        |

```
## The number of missing values is 7188
```

### 2.2.5 dependent

There is 14 classes (1 to 14) in the dependent variable, we can see that the class “2” is prevailing and that some classes have insignificant number (classes from 6 to 14), we will replace nas and classes that are almost insignificant in number with the class 2.

|    | dependents | Frequency |
|----|------------|-----------|
| 1  |            | 5488      |
| 2  |            | 12963     |
| 3  |            | 5653      |
| 4  |            | 2680      |
| 5  |            | 370       |
| 6  |            | 50        |
| 7  |            | 7         |
| 8  |            | 1         |
| 10 |            | 1         |
| 14 |            | 1         |

```
## The number of missing values is 2446
```

### 2.2.6 Has active card

We can see that the three classes of this variable have almost the same number of count, so We will split the nas equitably between the 3 classes.

|             | Has avtive card | Frequency |
|-------------|-----------------|-----------|
| Active      |                 | 9668      |
| Inactive    |                 | 9360      |
| Unpossessed |                 | 9086      |

```
## The number of missing values is 1546
```

### 2.2.7 Property location

As before the three classes have almost the same count, we will replace the nas by the three classes equitably.

|            | Peoperty location | Frequency |
|------------|-------------------|-----------|
| Rural      |                   | 9929      |
| Semi-Urban |                   | 10255     |
| Urban      |                   | 9129      |

```
## The number of missing values is 347
```

### 2.2.8 Income

Income summary

| Min.  | 1st Qu. | Median | Mean     | 3rd Qu. | Max.    | NA's |
|-------|---------|--------|----------|---------|---------|------|
| 377.7 | 1650.25 | 2223.3 | 2630.991 | 3090.33 | 1777460 | 4493 |

From the summary of this continuous feature we can see that there are big outliers which must affect the mean, in this case we will replace nas with the median.

## 2.2.9 Current loan expenses

Current loan expenses summary

| Min. | 1st Qu. | Median | Mean     | 3rd Qu. | Max.    | NA's |
|------|---------|--------|----------|---------|---------|------|
| -999 | 247.78  | 375.09 | 400.8938 | 521.05  | 3840.88 | 167  |

## the percentage of negative values is 0.005899705

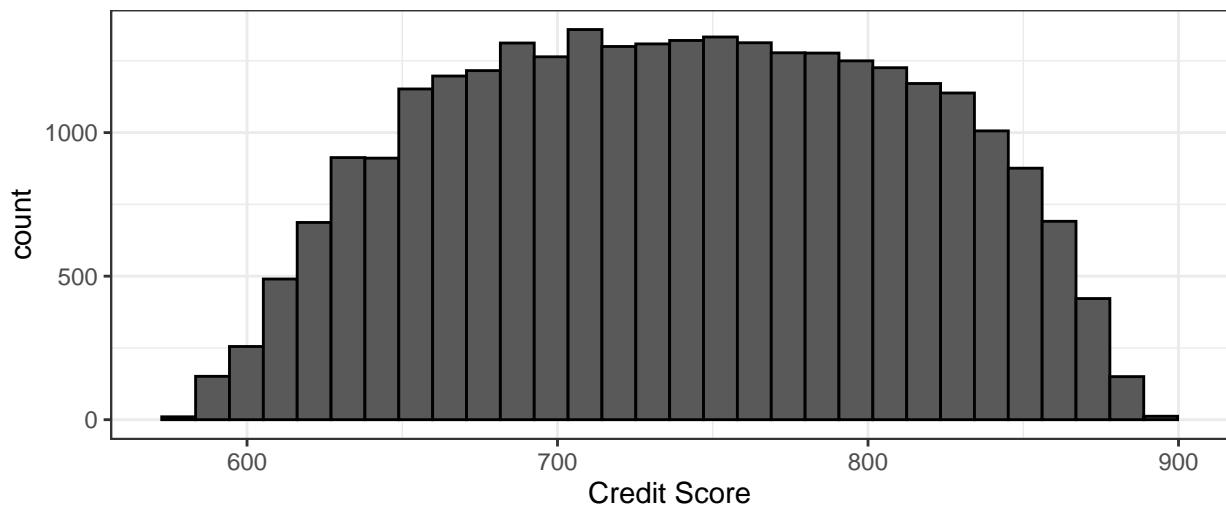
There number of nas is 167, and we notice the existence of some negative values(174), since the number is small we will consider those values outliers, and replace both nas and values less than 0 by the median.

## 2.2.10 Credit score

Credit score summary

| Min. | 1st Qu.  | Median  | Mean     | 3rd Qu. | Max.   | NA's |
|------|----------|---------|----------|---------|--------|------|
| 580  | 681.7325 | 739.815 | 739.8535 | 799.04  | 896.26 | 1670 |

Credit score dist



We can see that the distribution of this variable is almost square between the 1st and 3rd quartiles, then most values are between 680 and 800 with small variation, we replace nas with a sequence of 1670 (number of nas) values between 680 and 800 so as to not affect the distribution.

## 2.2.11 Property age

| Min.  | 1st Qu. | Median   | Mean     | 3rd Qu.  | Max.    | NA's |
|-------|---------|----------|----------|----------|---------|------|
| 377.7 | 1649.81 | 2223.965 | 2631.317 | 3090.832 | 1777460 | 4760 |

From the summary we notice that there is 4760 nas and there is some big outliers which may be driving the mean, so it would be better to replace nas by the median.

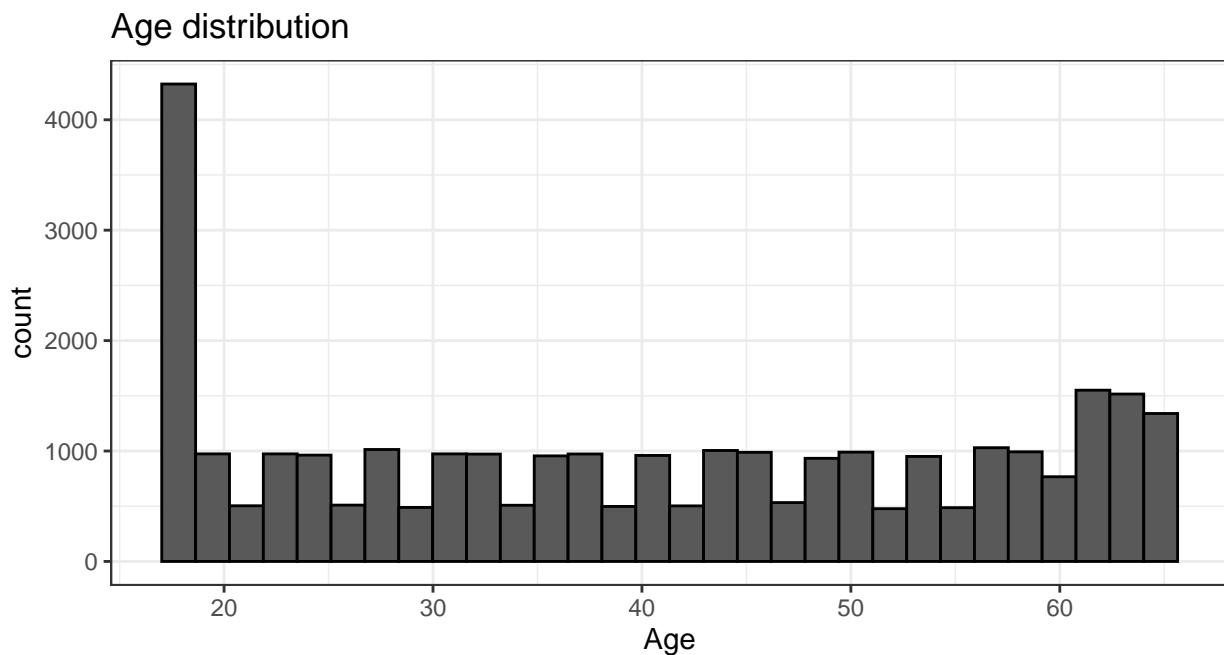
## 2.3 Exploring continuous variables

After some data cleaning, we can now go through variables exploration, we will start first with continuous features.

### 2.3.1 age

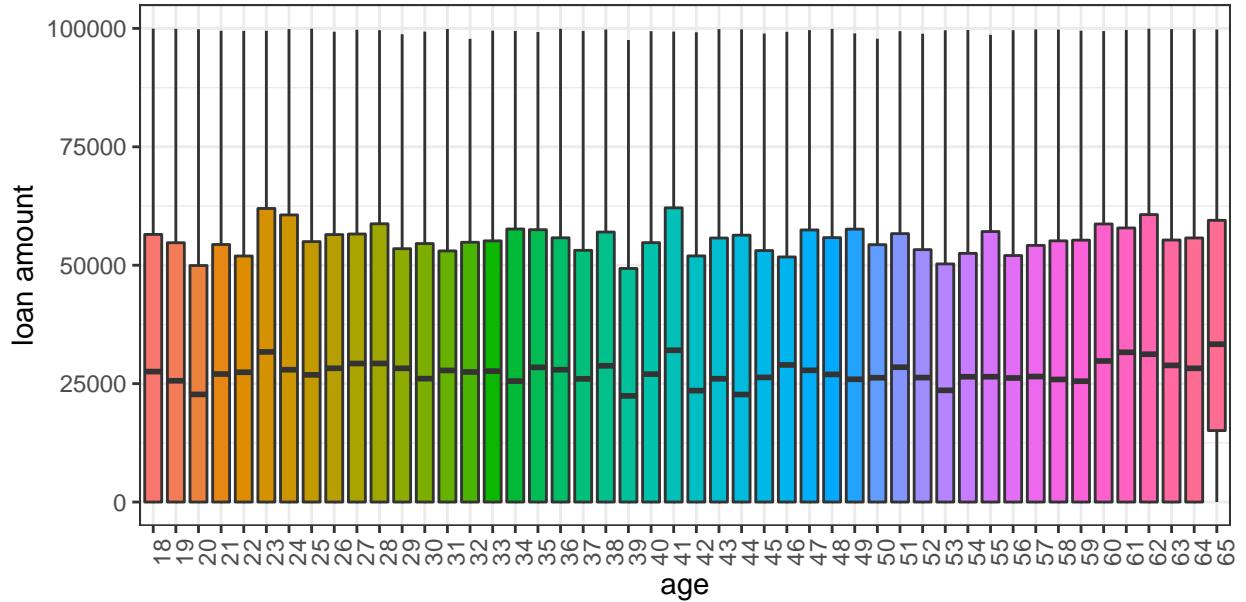
#### Summary

| Min. | 1st Qu. | Median | Mean     | 3rd Qu. | Max. |
|------|---------|--------|----------|---------|------|
| 18   | 25      | 40     | 40.09922 | 55      | 65   |



The variation of the age is almost stable, except the existence of a pick in the “18 age value” with 14% having the age of 18 years, the mean is 40 years and the range is 47 years.

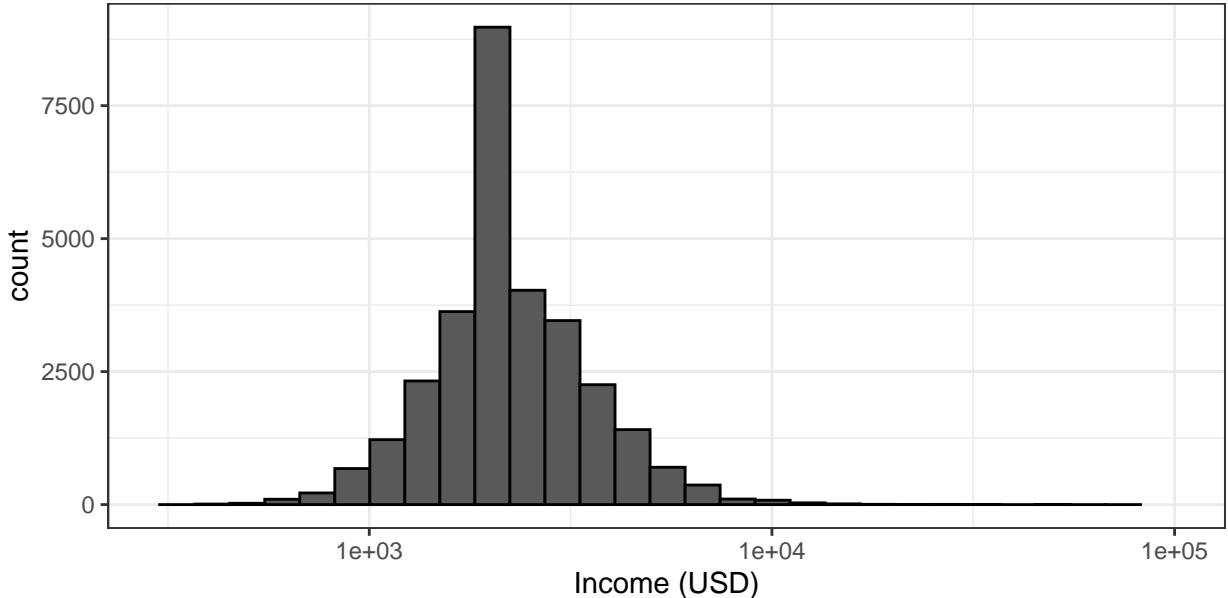
Boxplot of loan sanction amount vs age



From the boxplot we can see that customers with all ages has almost the same chances of getting loans, but ther are many outliers in the amount approved.

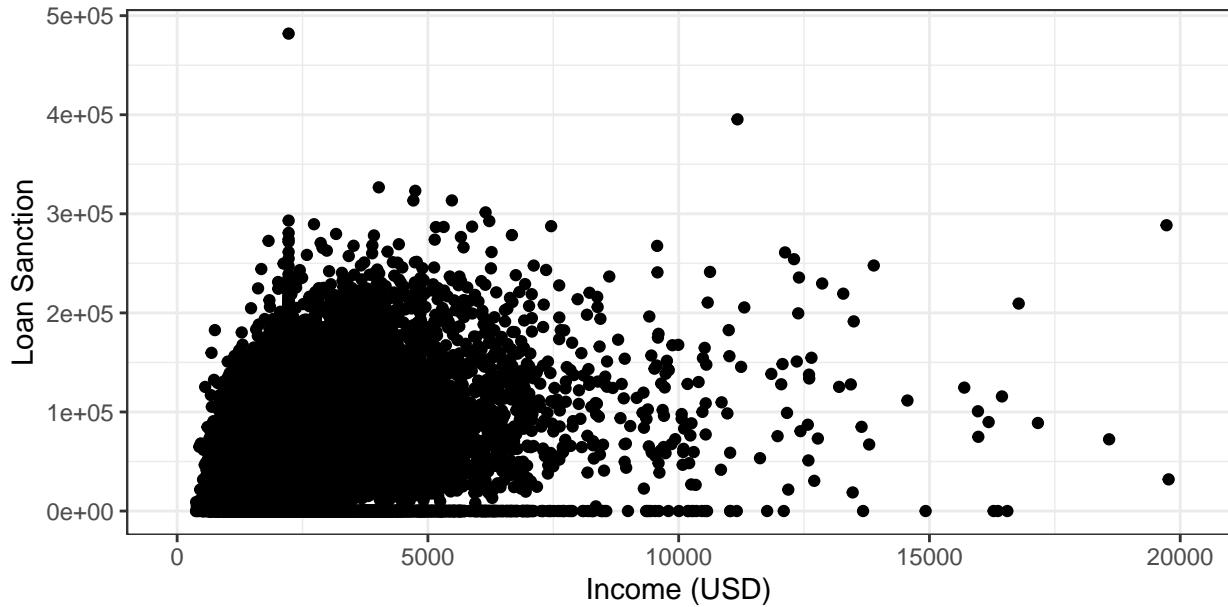
### 2.3.2 Income

Income distribution



From the plot we can notice that there is many outliers that are driving the distribution, excluding the those outliers the distribution is almost symmetric.

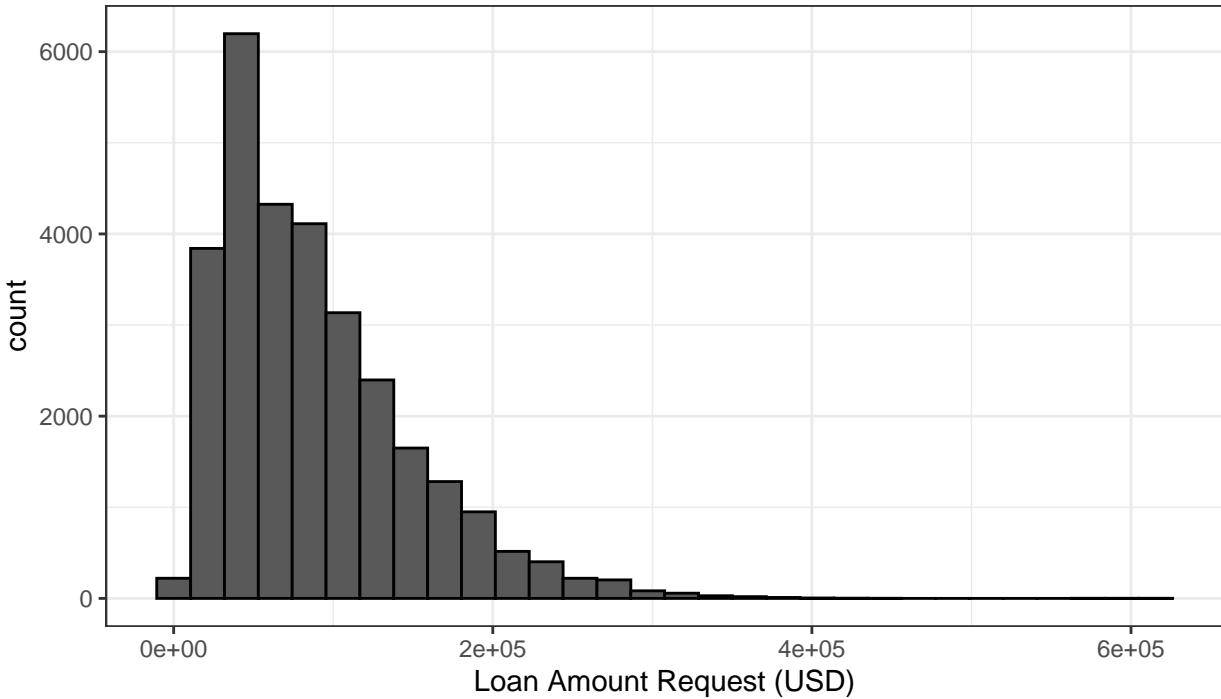
Loan sanction amount vs income



We notice a positive correlation between the income and the loan sanction amount except for null loan amount values and big outliers of income, using this variable in a model could improve the results.

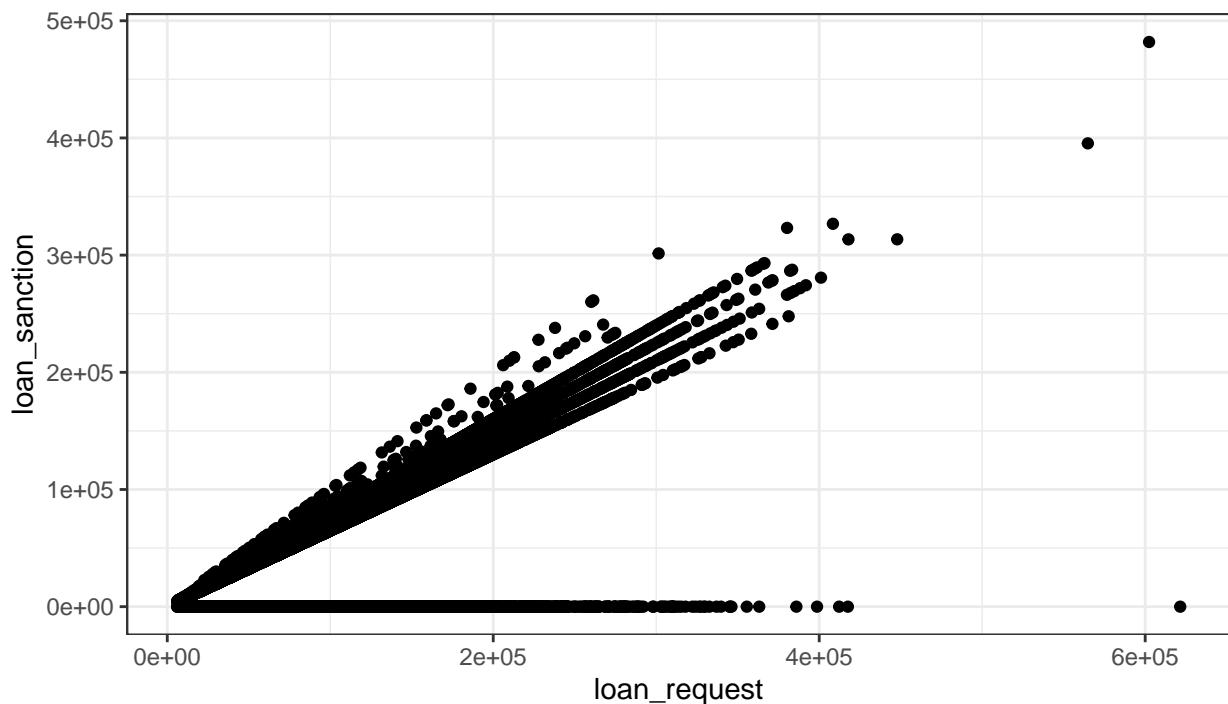
### 2.3.3 Loan amount request

Loan amount request distribution



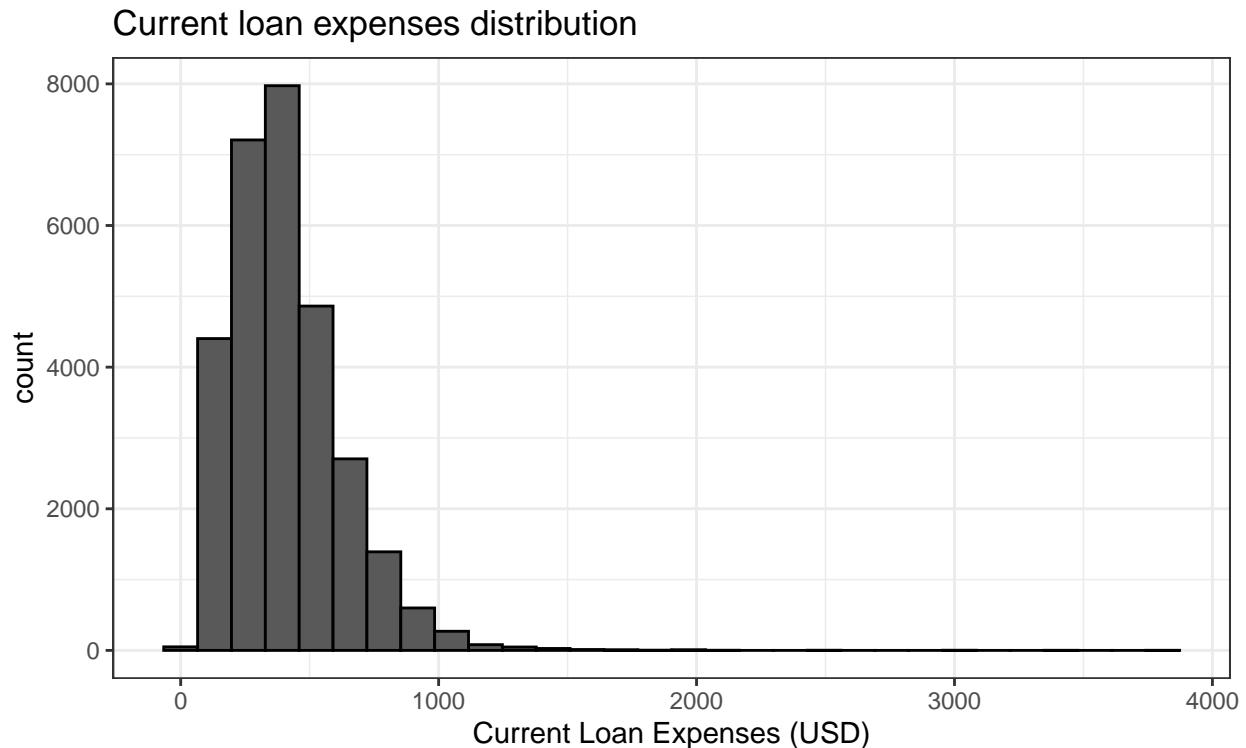
The distribution is right skewed, this means that most customers apply for big amounts of home loans.

Loan sanction amount vs loan request amount

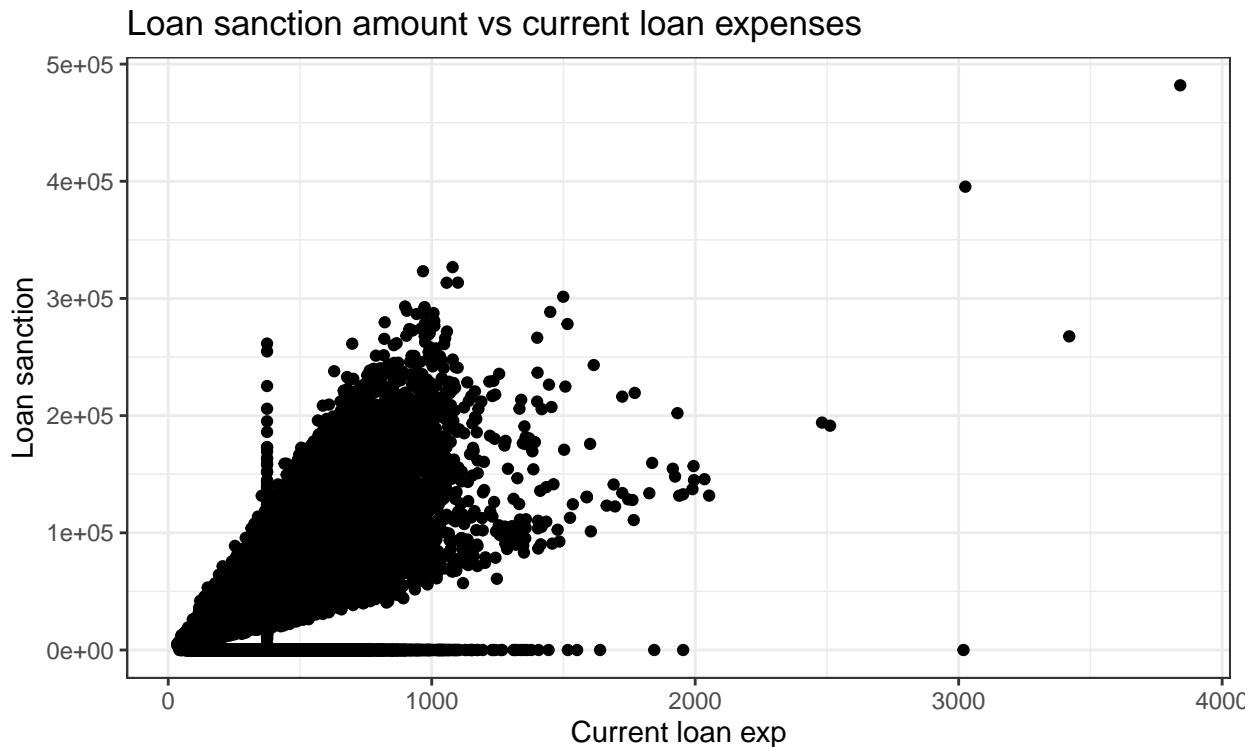


A part the null values representing the non approved loans we notice that there is a strong correlation between the amount requested and the amount approved, this may be very useful in building the predictions model.

#### 2.3.4 Current loan expenses

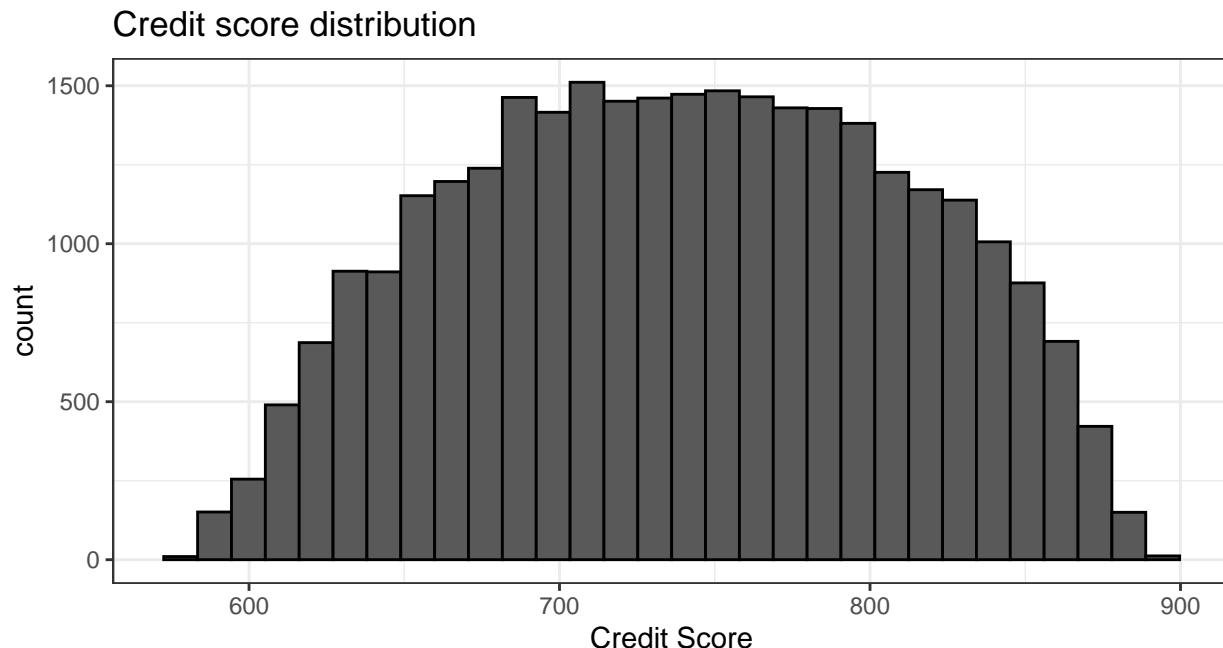


The distribution is right skewed, this means that most customers have less than the average of loan expenses.

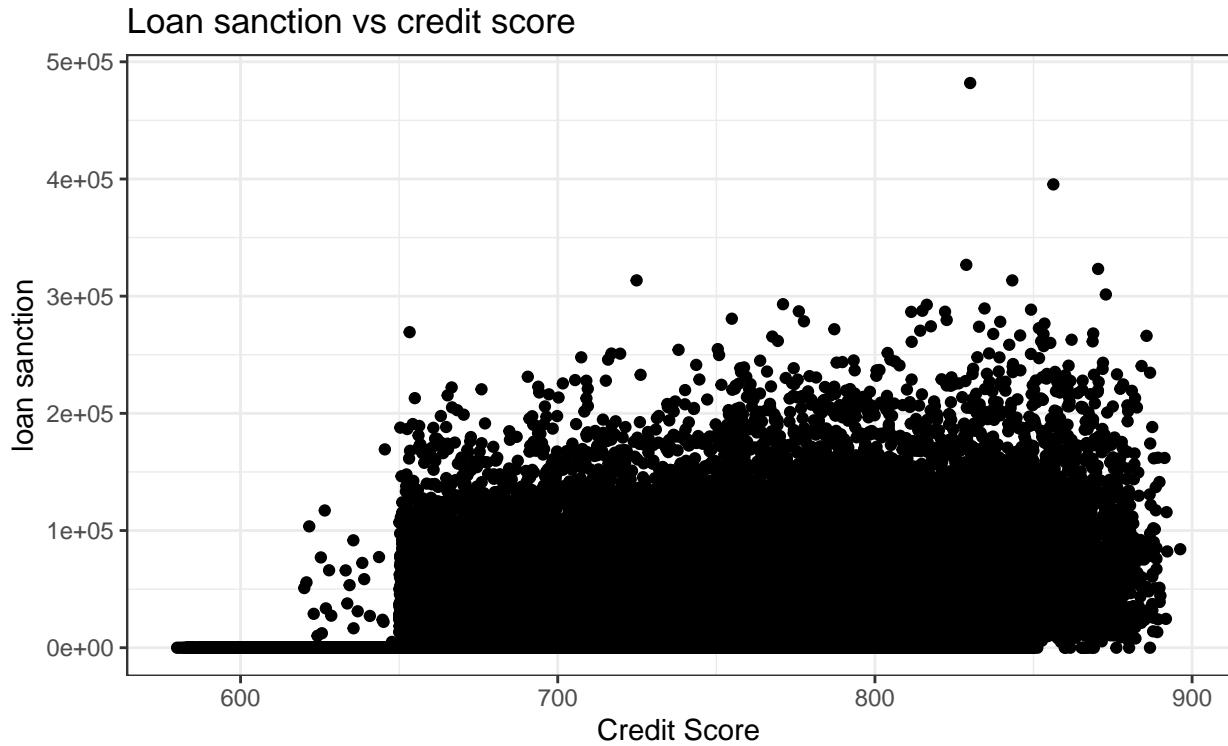


There is a strong correlation between the loan amount approved and the current loan expenses, this may mean that the bank tend to approve loans for customers who are already honoring their current loans and that customers who have big expenses tend to apply for big loans.

#### 2.3.5 Credit score



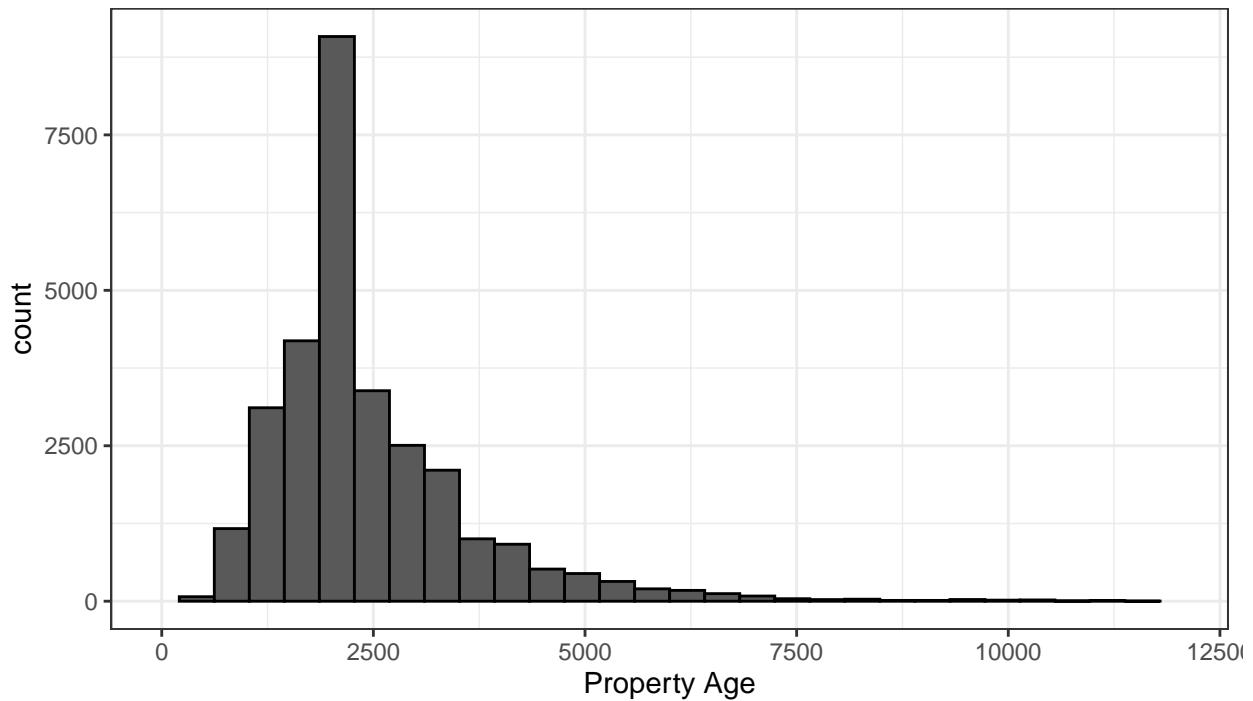
From the plot we can notice that the distribution is almost symmetric and that 80% of credit scores are between 620 and 820.



The correlation between the two variables is not strong enough to conclude that the credit score may affect the loan sanction amount, this is because the amount depends more on the amount requested which may not be correlated with the credit score. But the credit score still could be used to predict if a customer can get a loan or not at all.

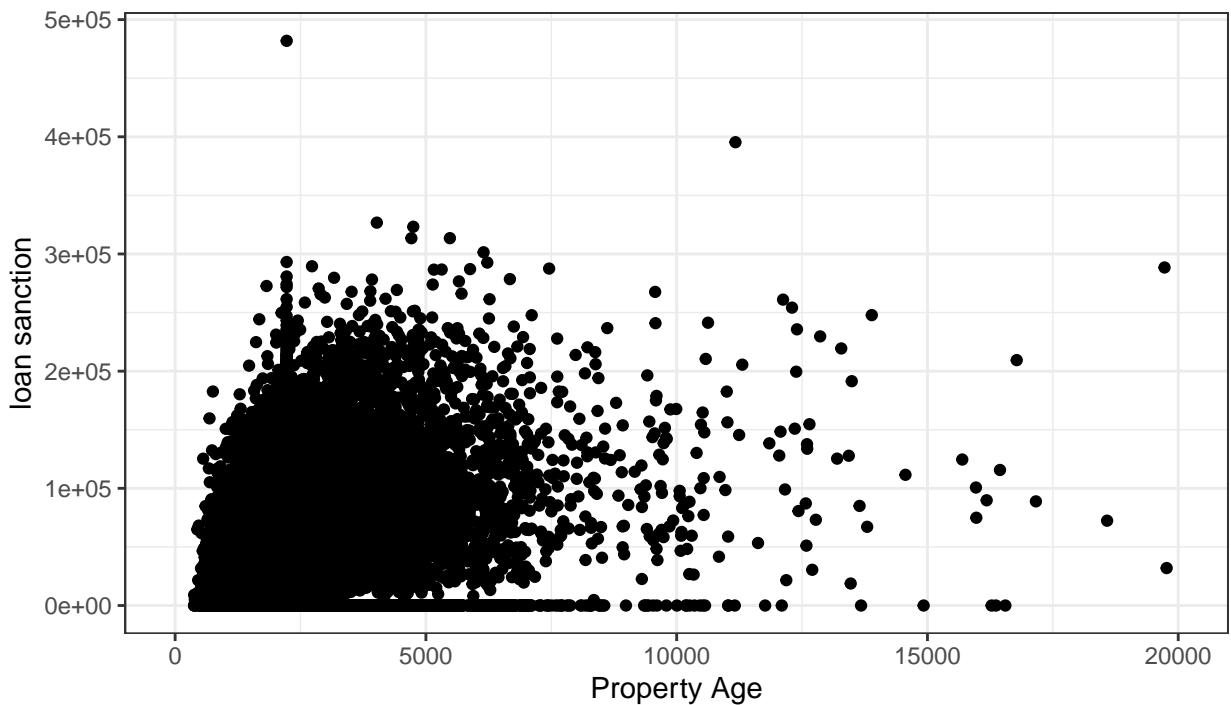
### 2.3.6 Property age

Property age distribution



The distribution contains big outliers and is right skewed, this means that most properties have an age below the mean.

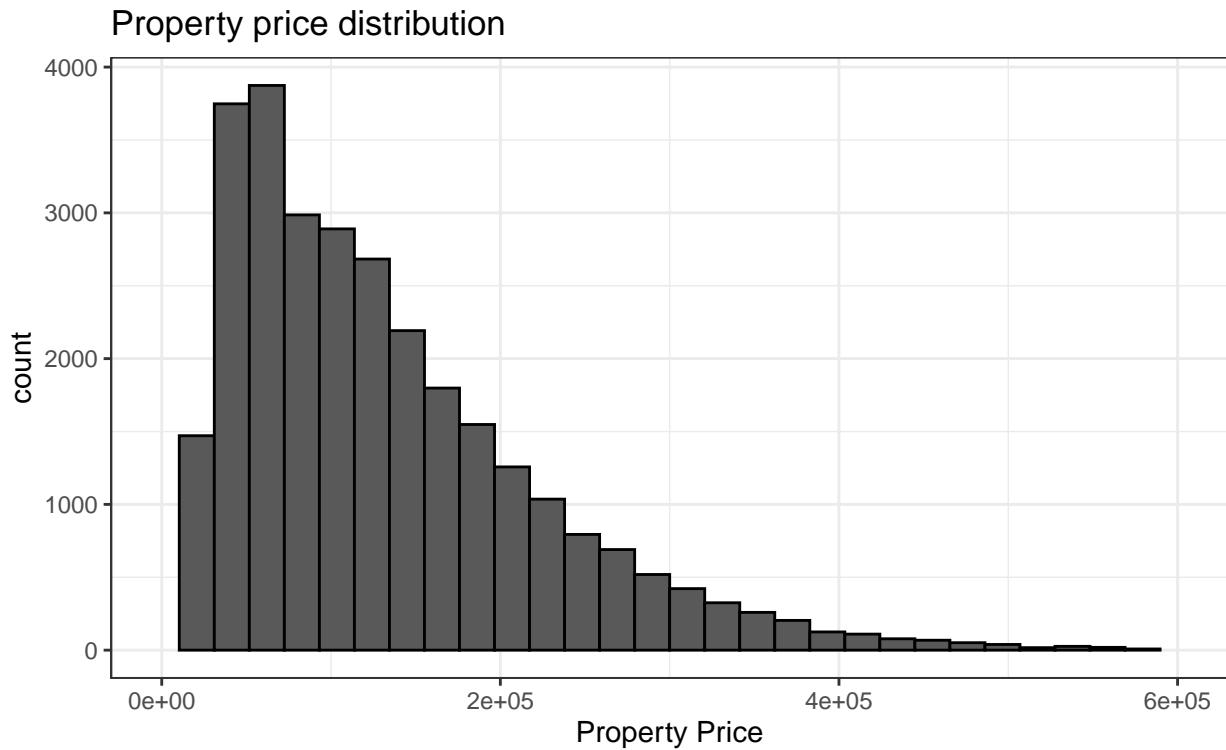
Loan sanction vs property age



We see that there is a correlation between the two variables, but this doesn't make sense for me, the property age should have a negative correlation with the loan amount. This feature would need more analysis if needed to be included in the model.

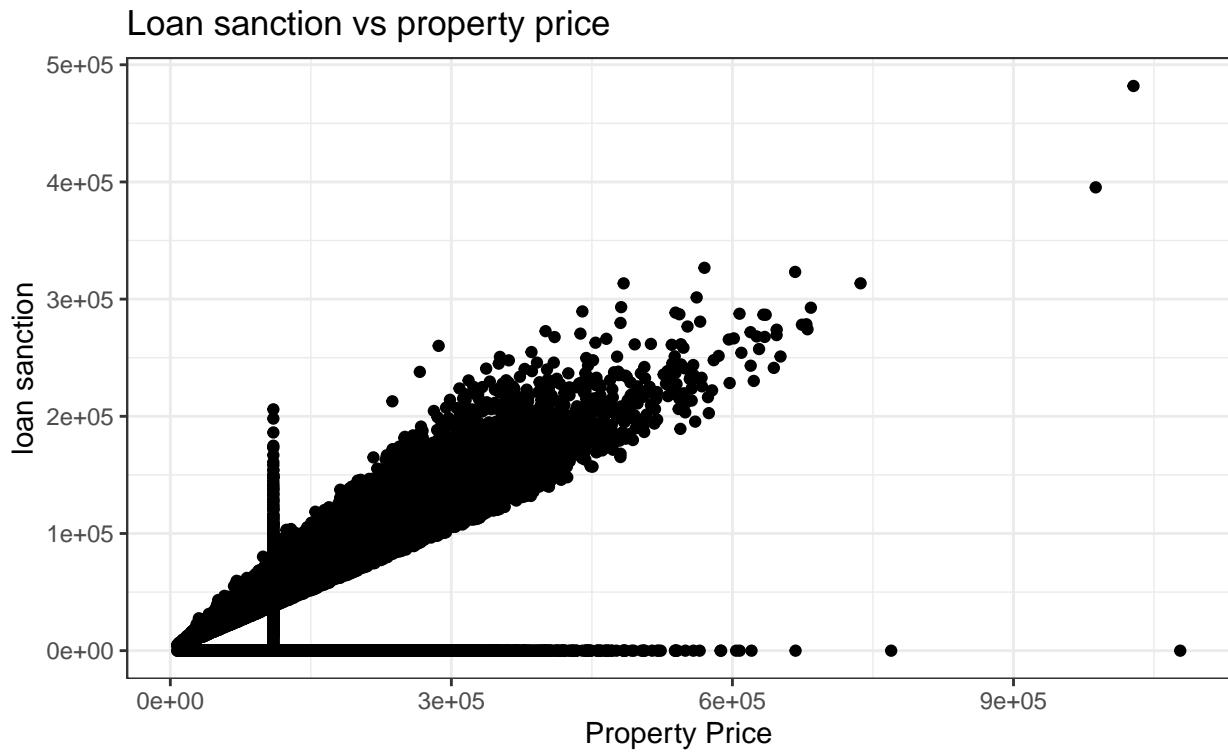
### 2.3.7 Property price

| Min. | 1st Qu.  | Median   | Mean     | 3rd Qu.  | Max.    |
|------|----------|----------|----------|----------|---------|
| -999 | 60658.74 | 109993.6 | 131733.5 | 178801.9 | 1077967 |



The distribution is right skewed, most prices are below the mean.

There is some negative prices, this may be an error so we will replace them with the median since the distribution is not symmetric.

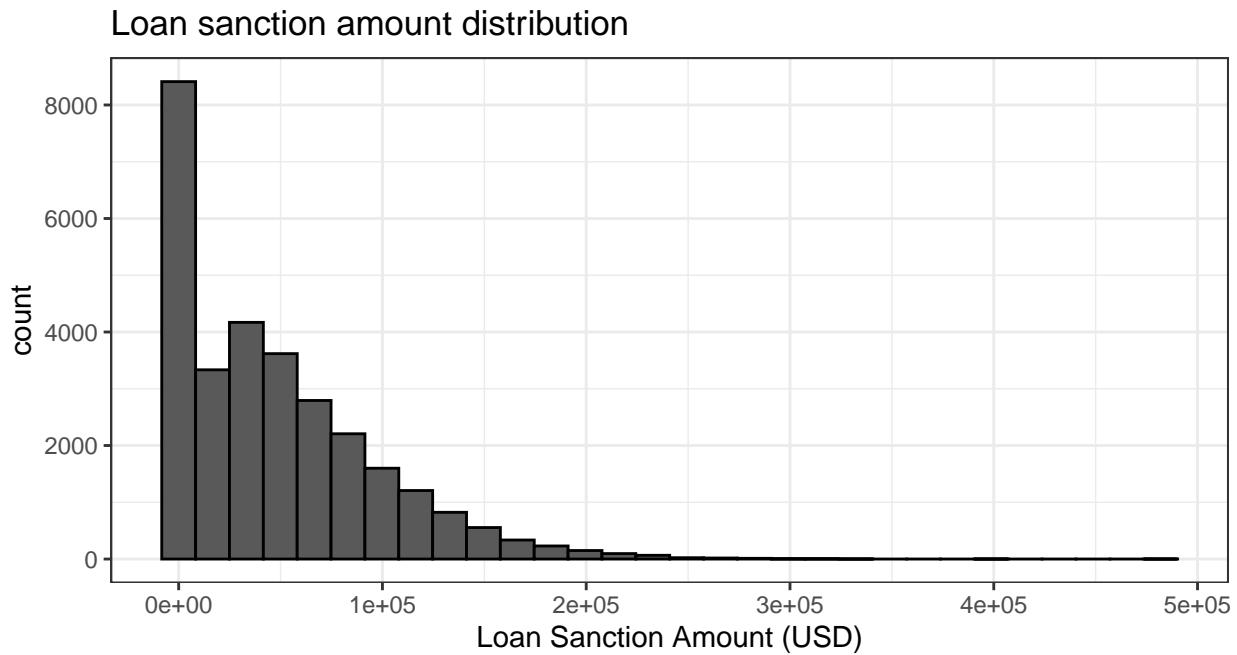


There is a strong correlation between the loan amount approved and property price, this is obvious since the amount requested depends on the property price. This correlation will be very useful in predicting the loan amount.

### 2.3.8 Loan sanction amount

#### Summary of the target variable

| Min. | 1st Qu. | Median   | Mean     | 3rd Qu.  | Max.     |
|------|---------|----------|----------|----------|----------|
| 0    | 0       | 35209.39 | 47660.73 | 74261.25 | 481907.3 |



We can see that the distribution is right skewed and that many customers (28%) didn't get their loans request approved.

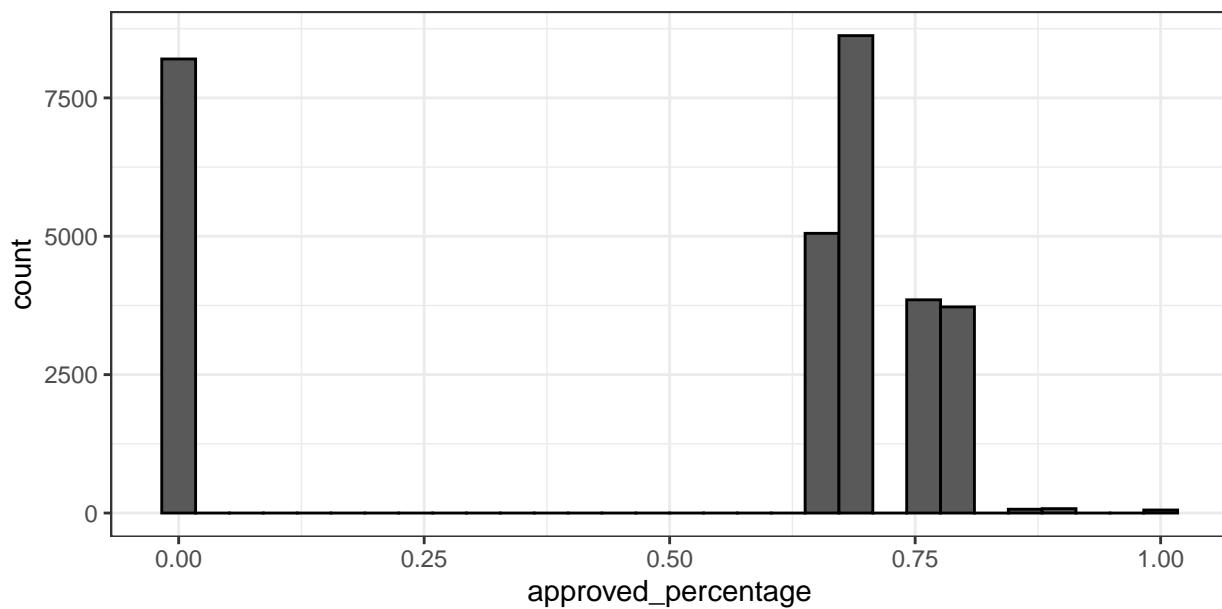
### 2.3.9 Percentage of approved loan amount:

The loan amount approved is calculated on the base of the requested loan, so it would be useful the make a variable in which we store the percentage approved of the requested loan.

#### Summary of approved percentage

| Min. | 1st Qu. | Median | Mean      | 3rd Qu.   | Max. |
|------|---------|--------|-----------|-----------|------|
| 0    | 0       | 0.7    | 0.5183715 | 0.7499999 | 1    |

Approved\_percentage distribution

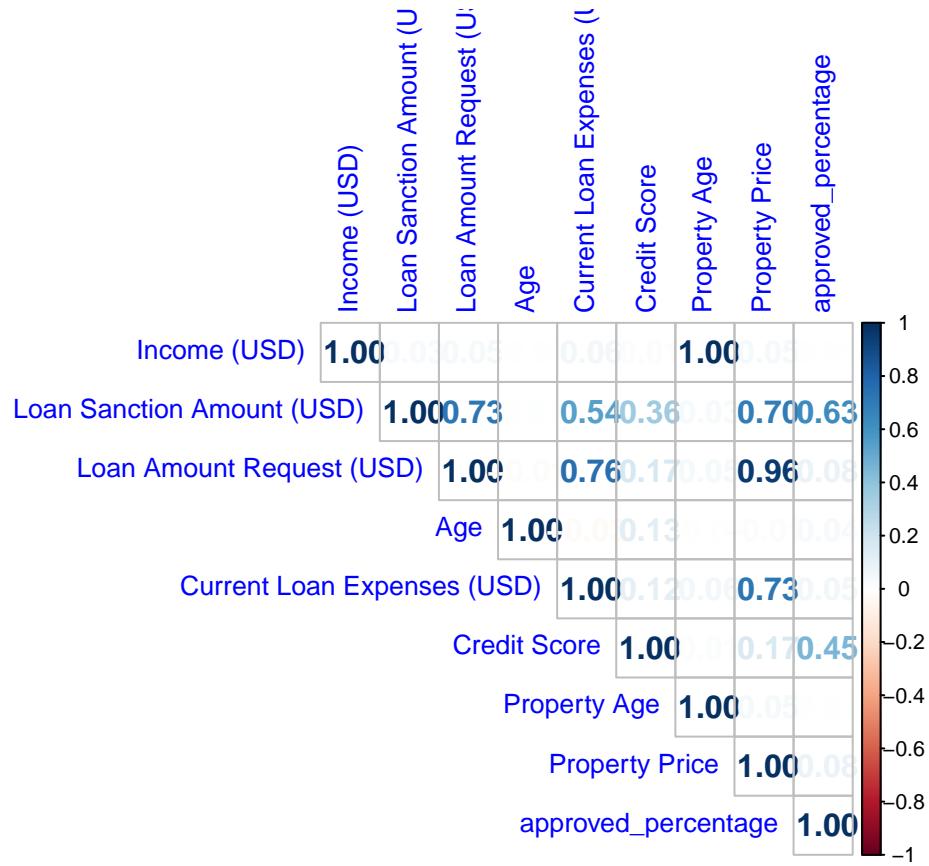


From the plot we can see that in general the customers either get nothing from their requested amount, or they get between 60% and 85 %.

#### 2.3.10 Correlation between continuous variables

Let's plot the correlation between all the continuous variables.

### Correlation between all continuous features



There is strong correlation between the sanction amount and : "the amount requested, the property price, the approved percentage, current loan expenses. there is also correlation with credit score even if not strong.

There is strong correlation between the amount requested and the property price.

The correlation between property age and income is 1, which is not normal, so let's drop the property age.

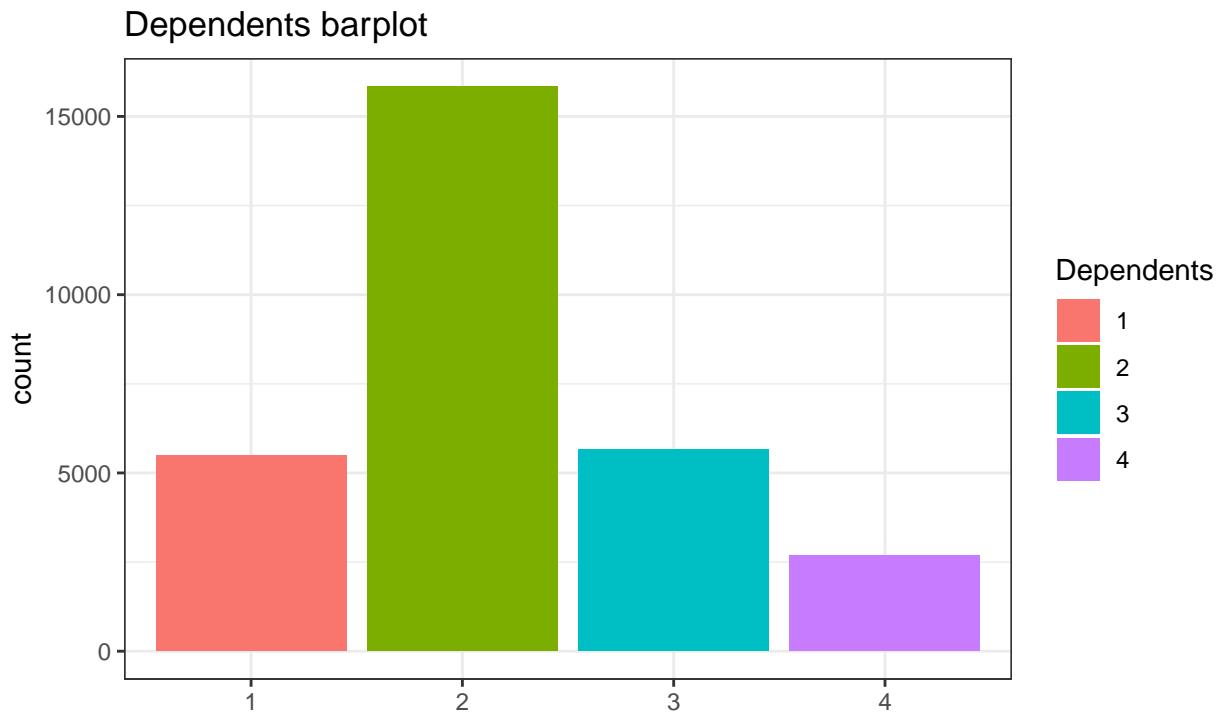
## 2.4 Categorical features exploration

Now we will deal with the categorical features, but first let's see how many classes we have in each variable.

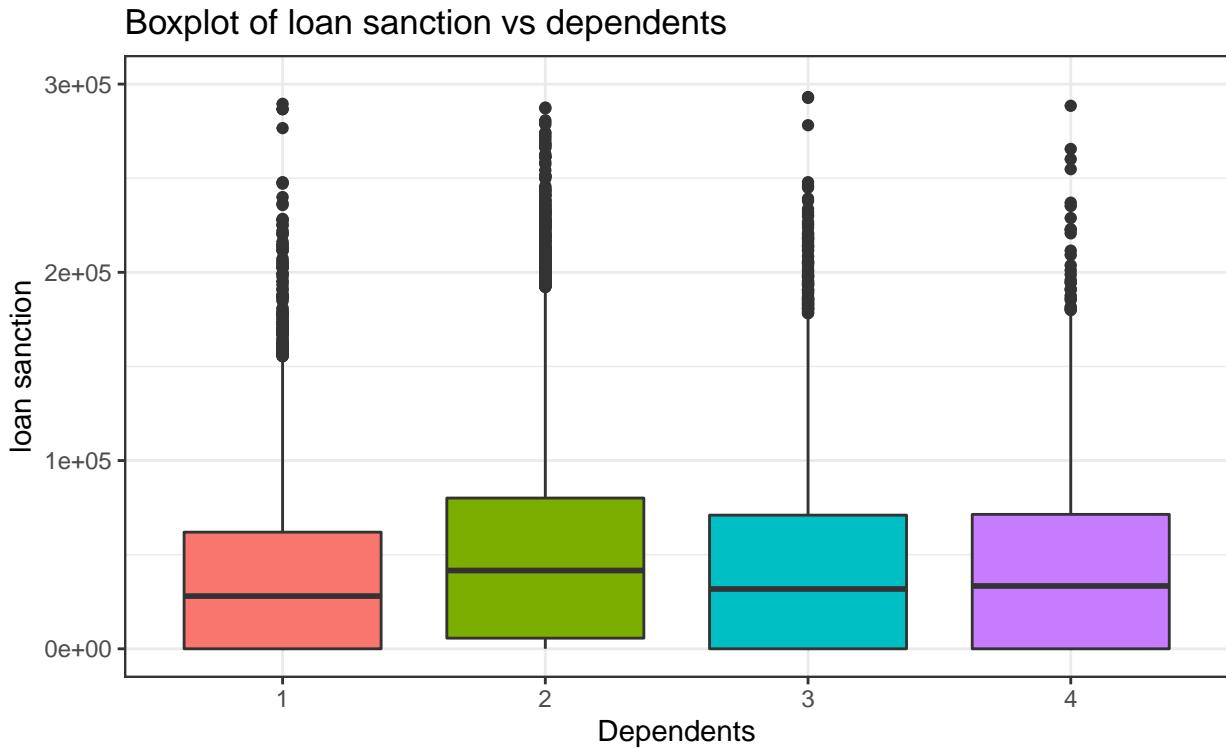
### Distinct classes in categorical columns

|                        | Number of classes |
|------------------------|-------------------|
| Gender                 | 2                 |
| Income Stability       | 2                 |
| Profession             | 8                 |
| Type of Employment     | 19                |
| Location               | 3                 |
| Expense Type 1         | 2                 |
| Expense Type 2         | 2                 |
| Has Active Credit Card | 3                 |
| Property Location      | 3                 |

#### 2.4.1 Dependents



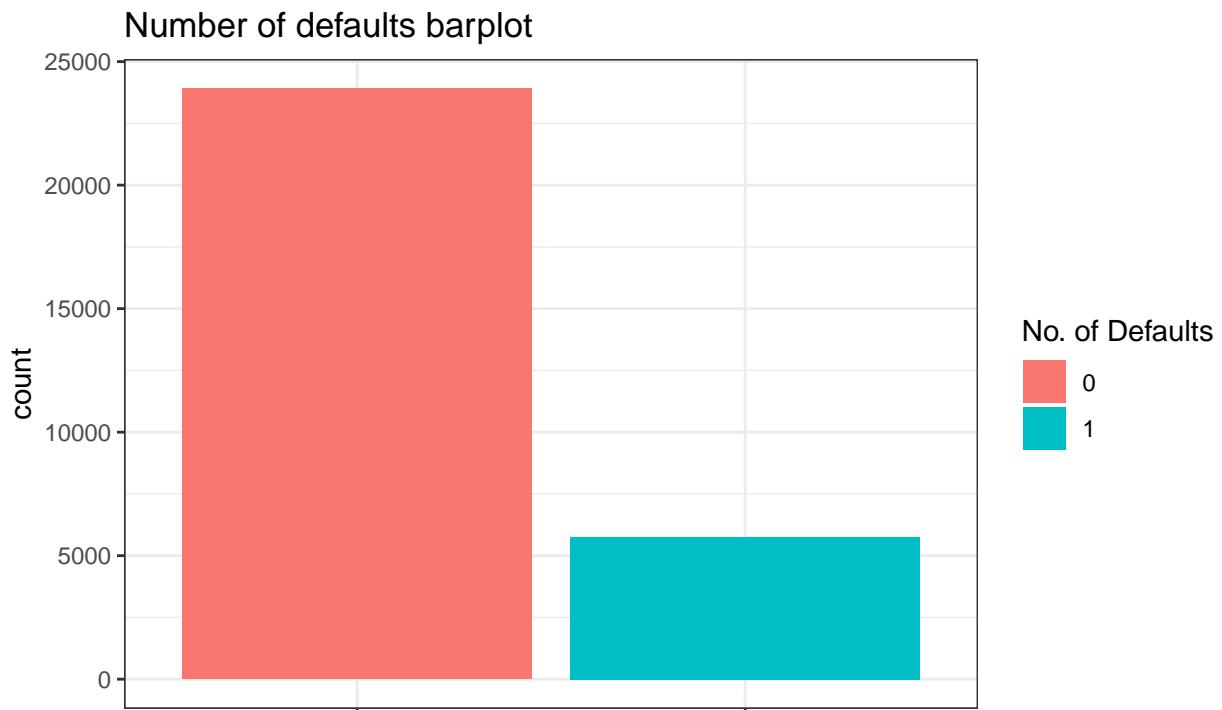
Most customers have 2 dependents.



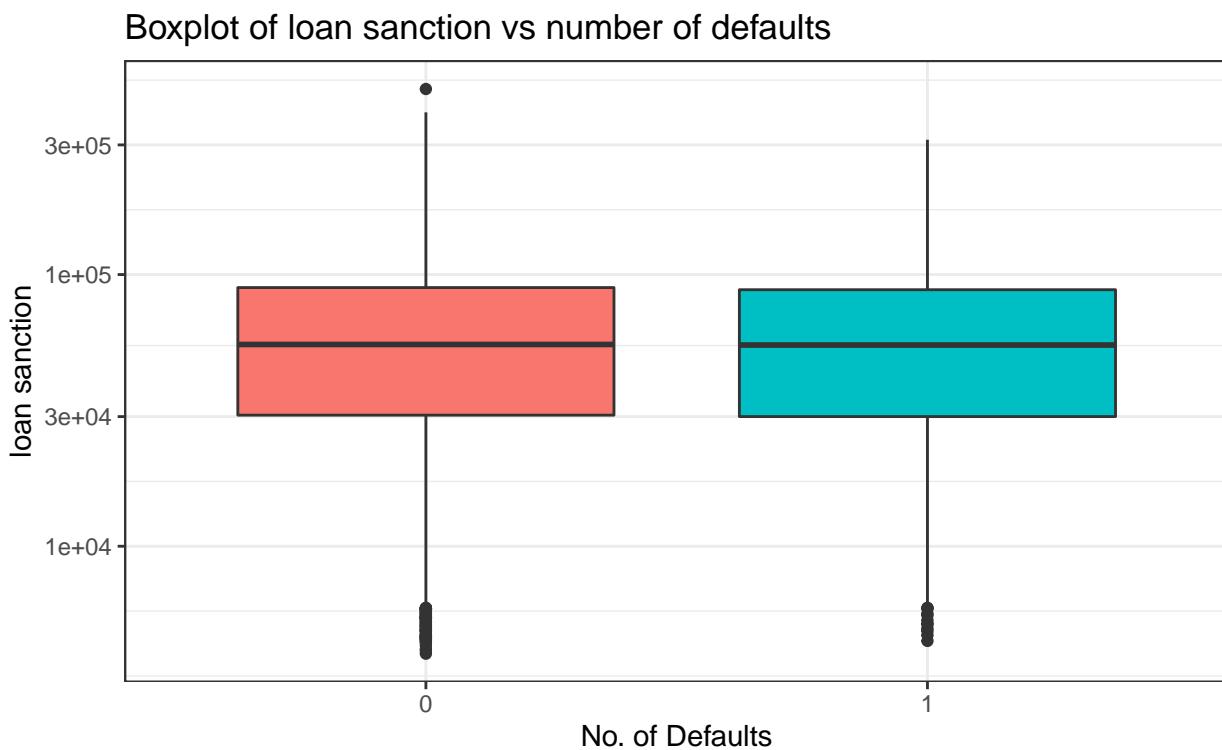
The clients with number of dependents 2 seem to have a bit more chances to get their request approved than

the others, their IQR doesn't overlap with 0 and they have the highest median loan amount.

#### 2.4.2 Number of defaults

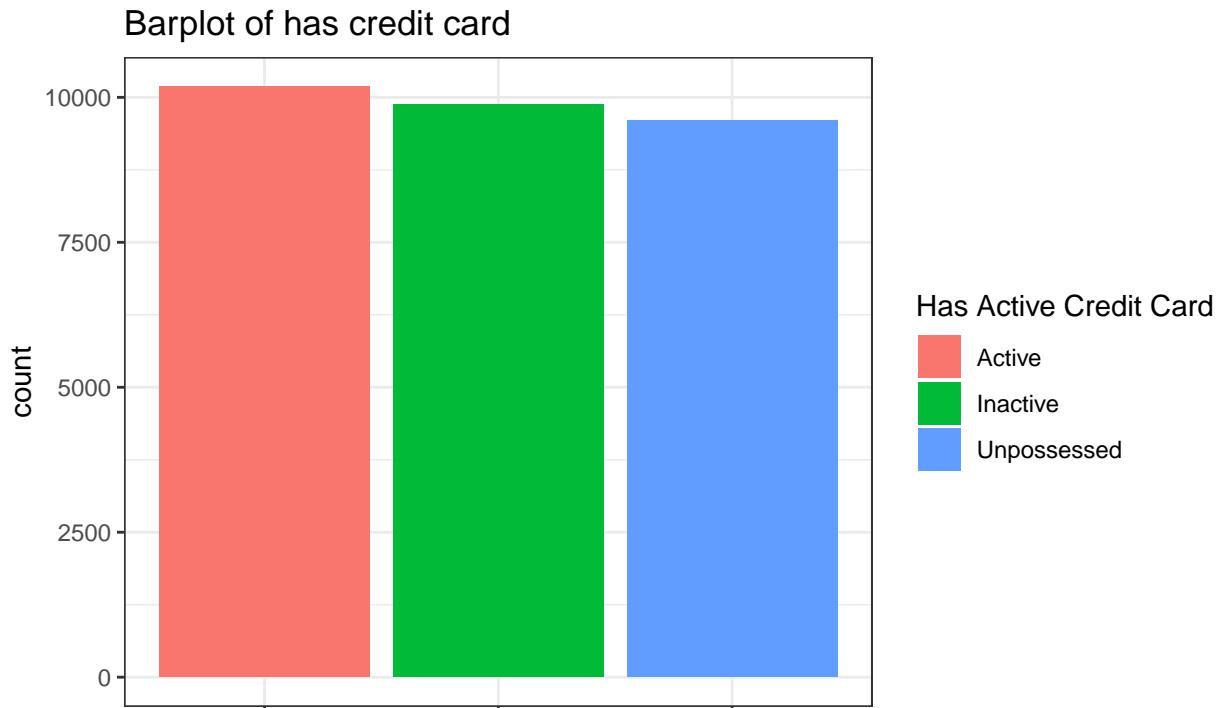


Most customers don't have past defaults.

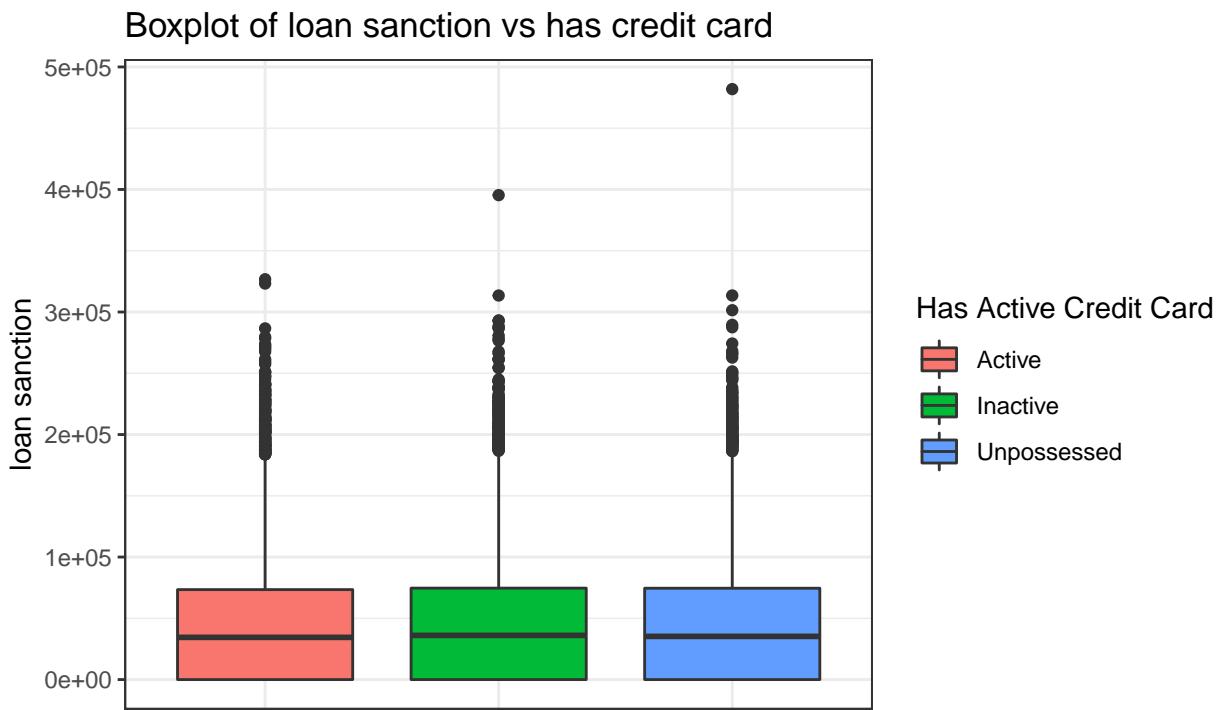


The number of defaults doesn't seem to highly affect the loan sanction amount.

#### 2.4.3 Has credit card

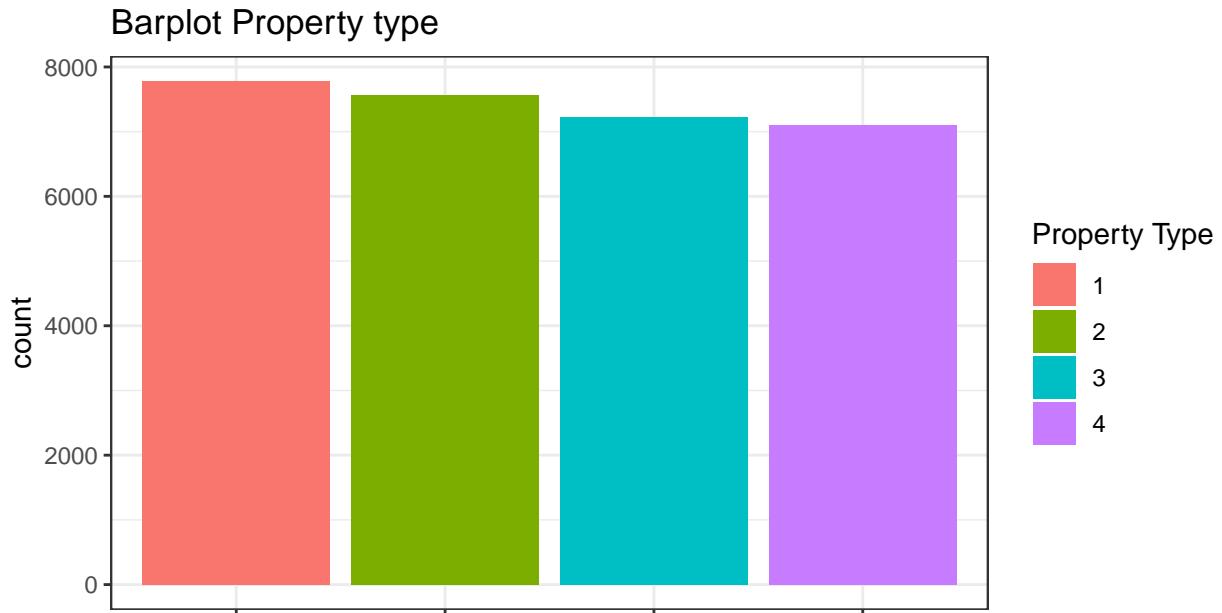


The number of clients who have or don't have credit card is close.

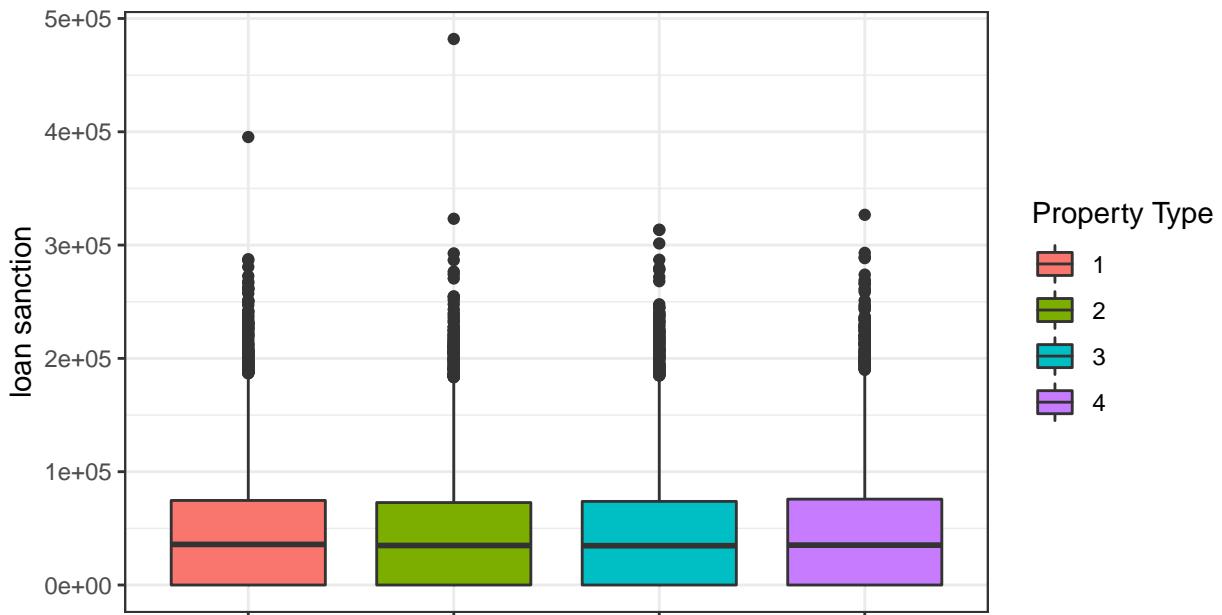


As we can see the fact of having or not having a credit card doesn't affect the amount of loan approved.

#### 2.4.4 Property type



Even if the property type “1” is on top, the number of the three classes is close.

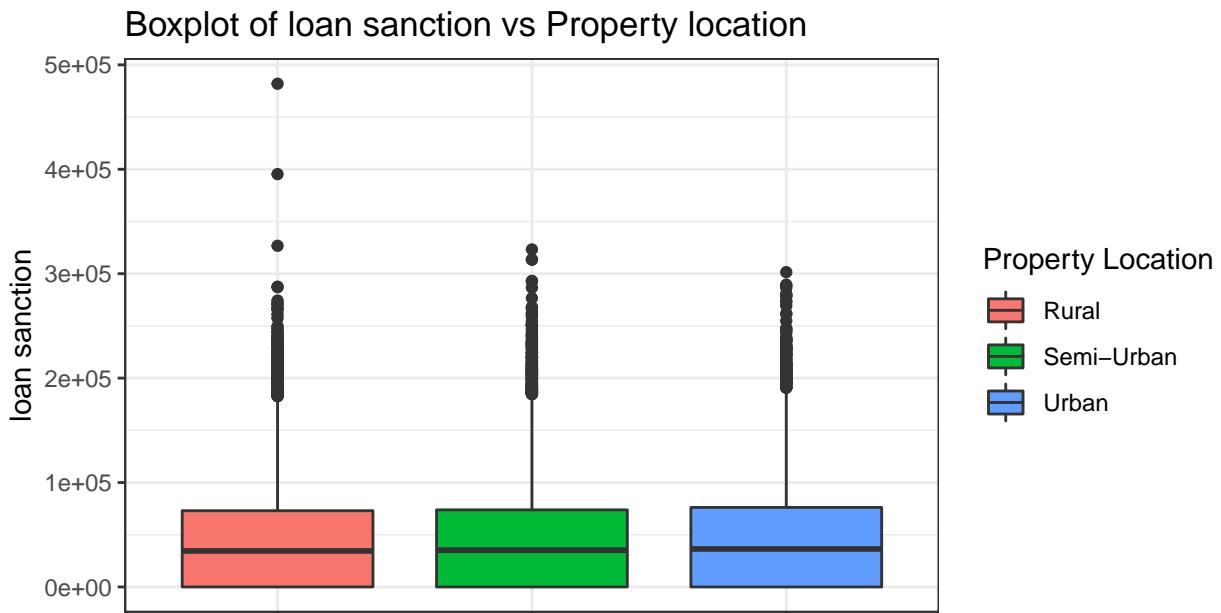


The property type on its own doesn't seem to affect the amount of approved loan.

#### 2.4.5 Property location

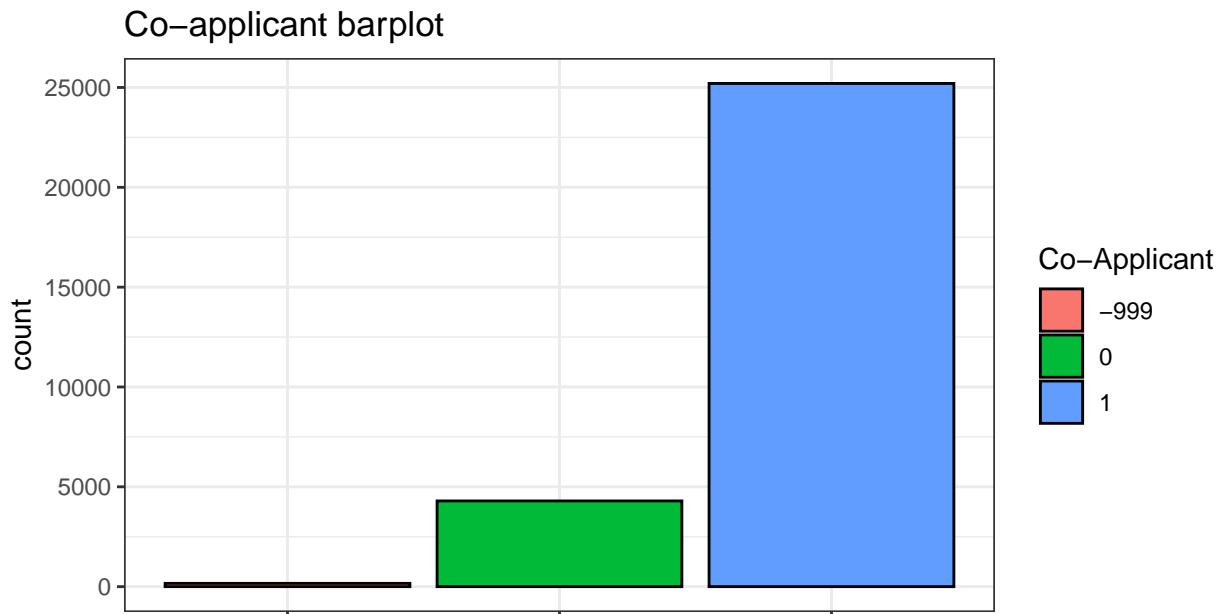


The number of property location classes is close.



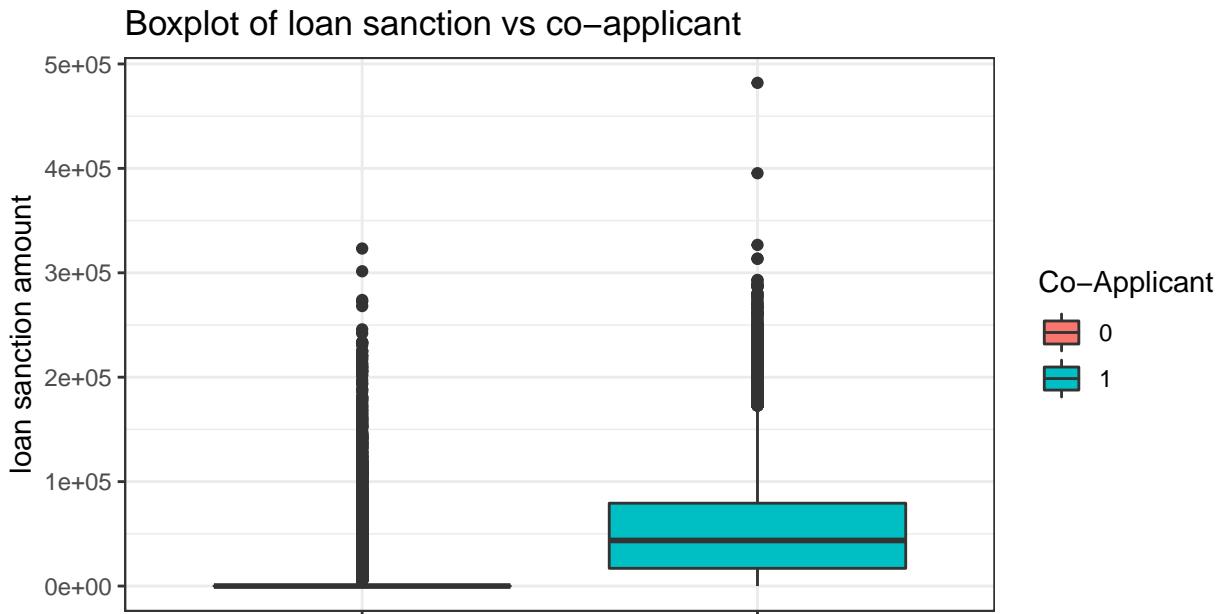
The property location doesn't seem to affect the loan sanction amount.

#### 2.4.6 Coapplicant



Most customers have one co-applicant for their requested loans.

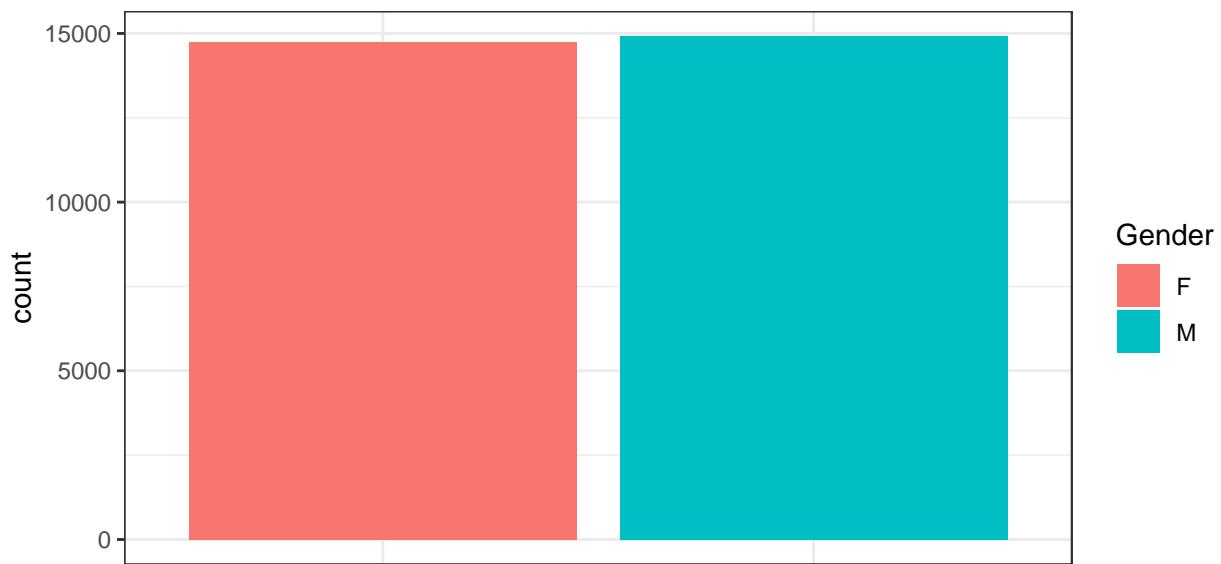
Co\_applicant must be a positive, so we assign negative value to the prevalent class “1”.



We notice that the most approved loans are the ones that had more than one applicant. The requested loan with no co-applicant seems to be rejected in general, this may be very useful in predictions.

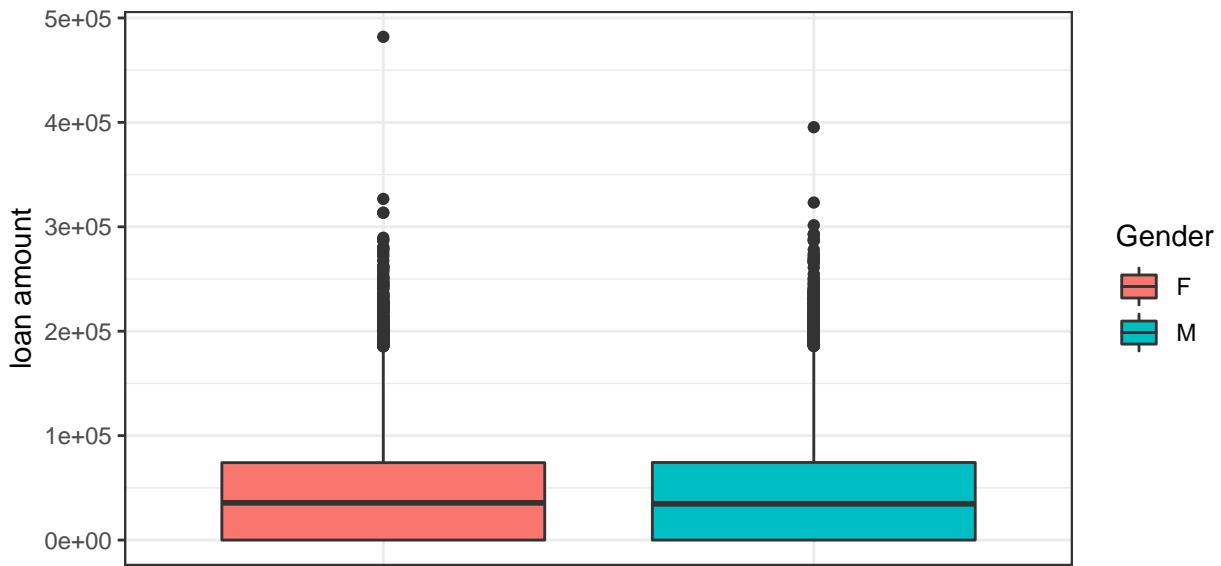
#### 2.4.7 Gender

Gender barplot



There is as many female as male customers in the data set.

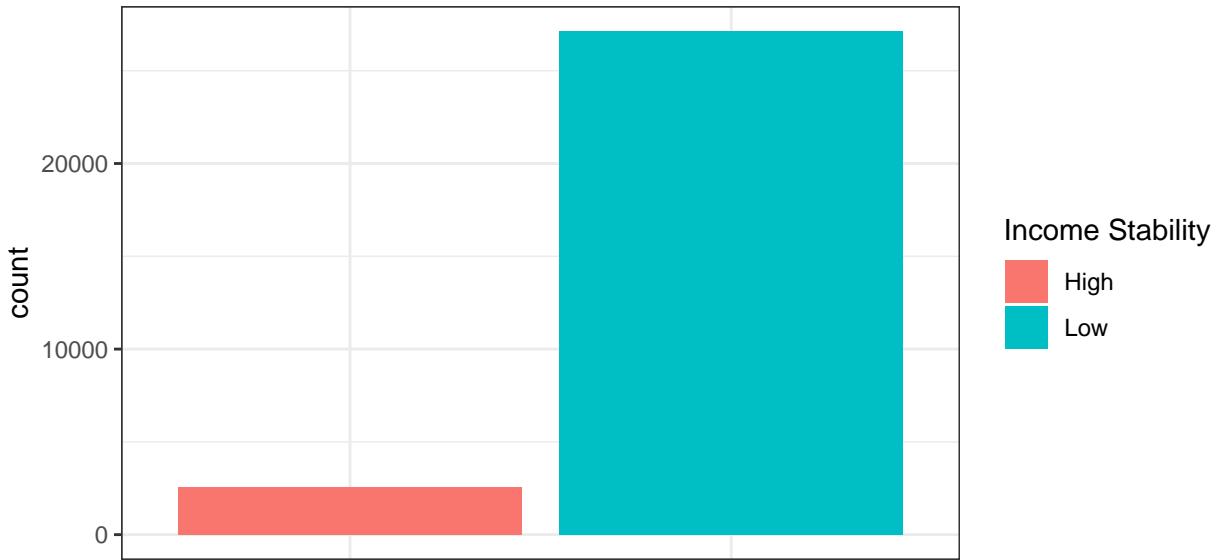
Boxplot of loan sanction vs gender



The fact of being a male or female doesn't seem to affect the amount of approved loan.

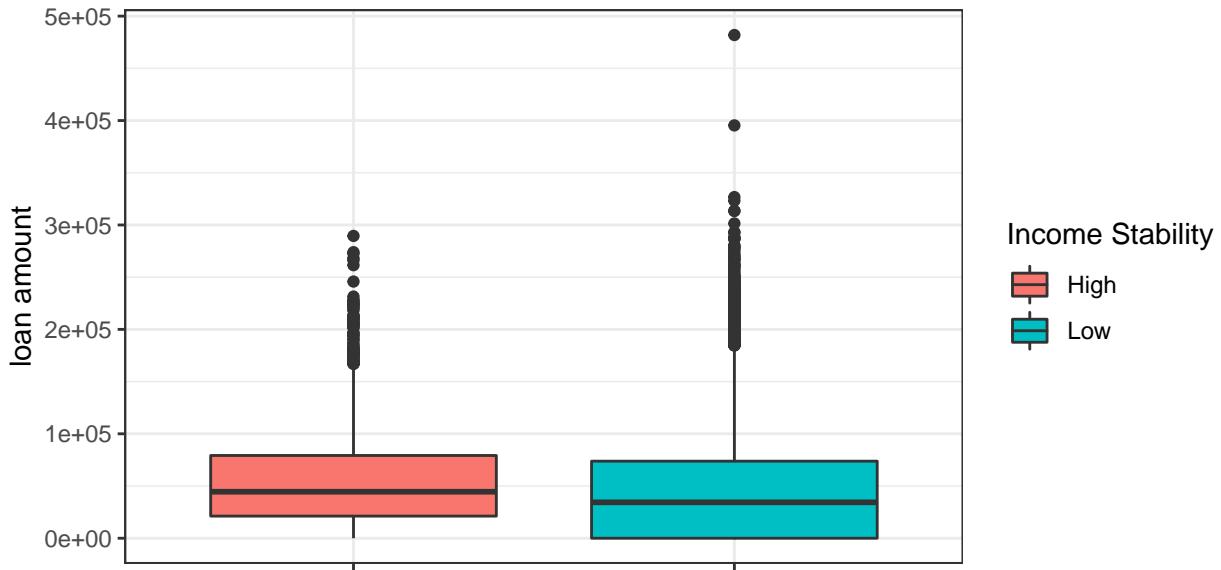
#### 2.4.8 income stability

Income stability barplot



We can notice that most customers have low income stability.

Boxplot of loan sanction vs income stability

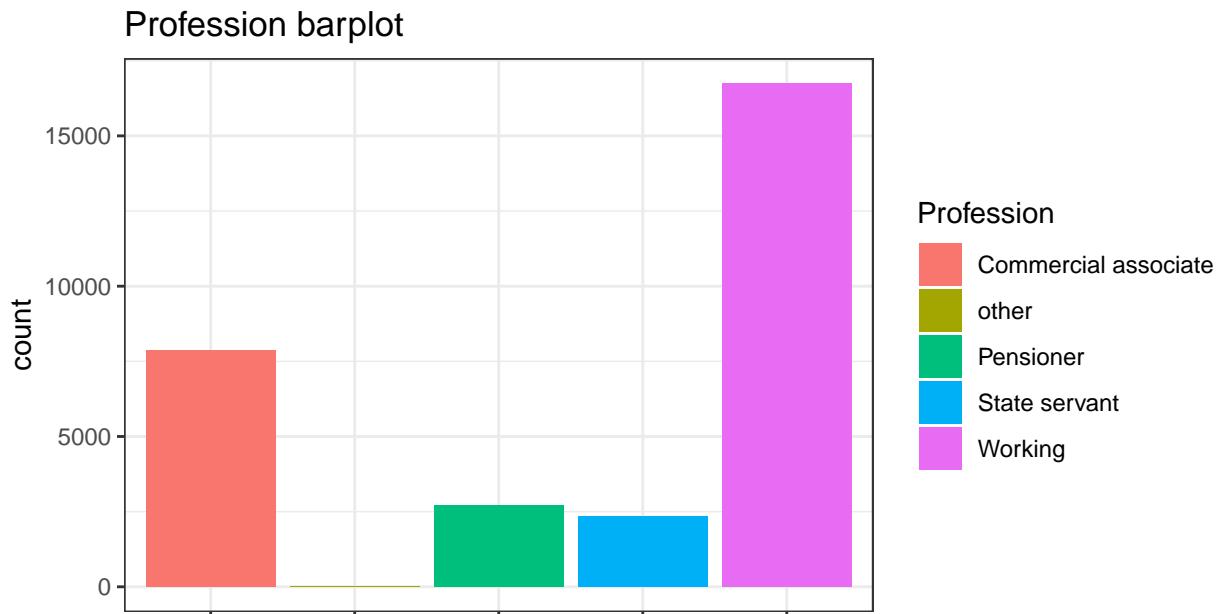


From the boxplot we can see that the fact of having a high income stability affect positively the loan amount approved, using this effect in building a model will improve the results.

#### 2.4.9 profession

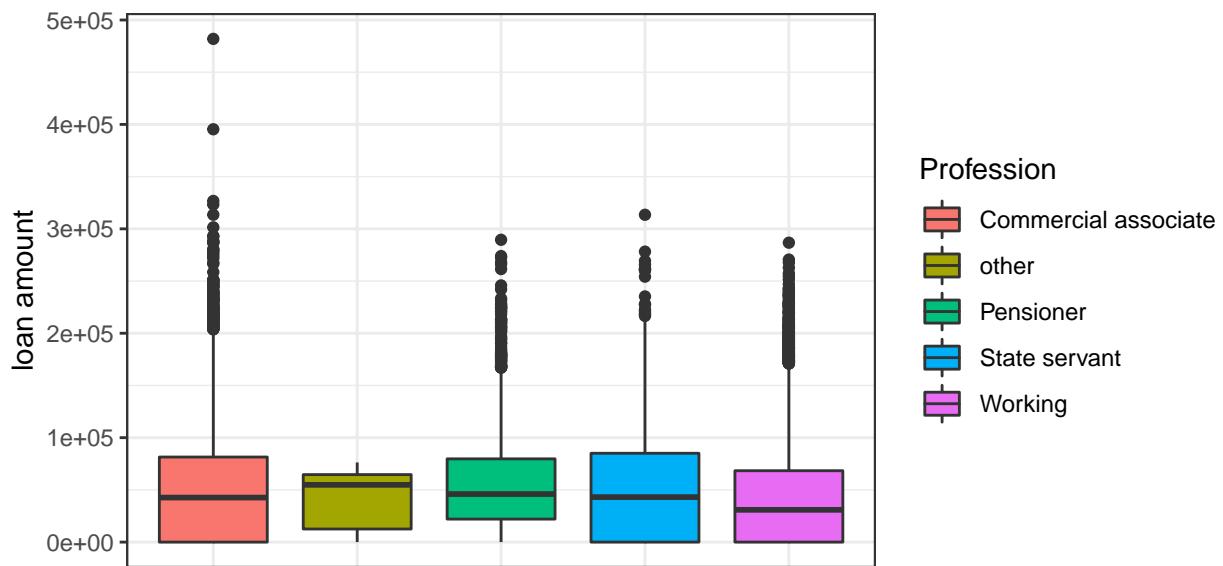
| .                    | Freq  |
|----------------------|-------|
| Working              | 16739 |
| Commercial associate | 7863  |
| Pensioner            | 2718  |
| State servant        | 2334  |
| Businessman          | 2     |
| Unemployed           | 2     |
| Maternity leave      | 1     |
| Student              | 1     |

There are 8 classes of profession in the data. Some classes have insignificant numbers, let's assign them to a new class called "other".



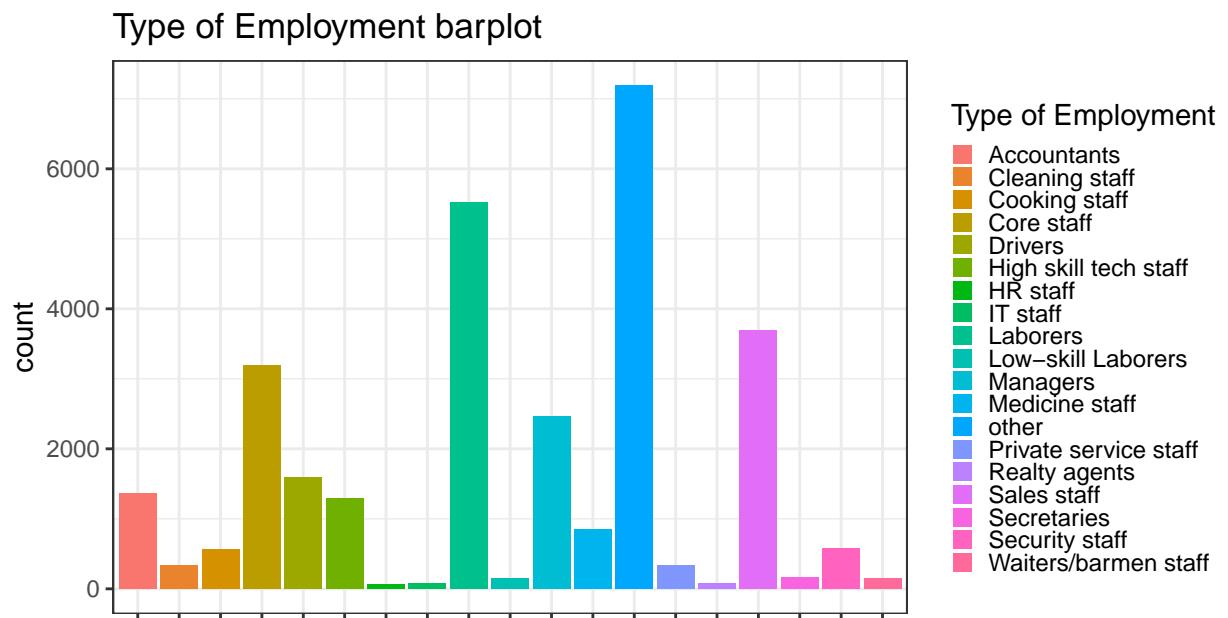
Most customers have the profession working, this means that this class is not a profession but a category that could contain many professions.

Boxplot of loan sanction vs Profession



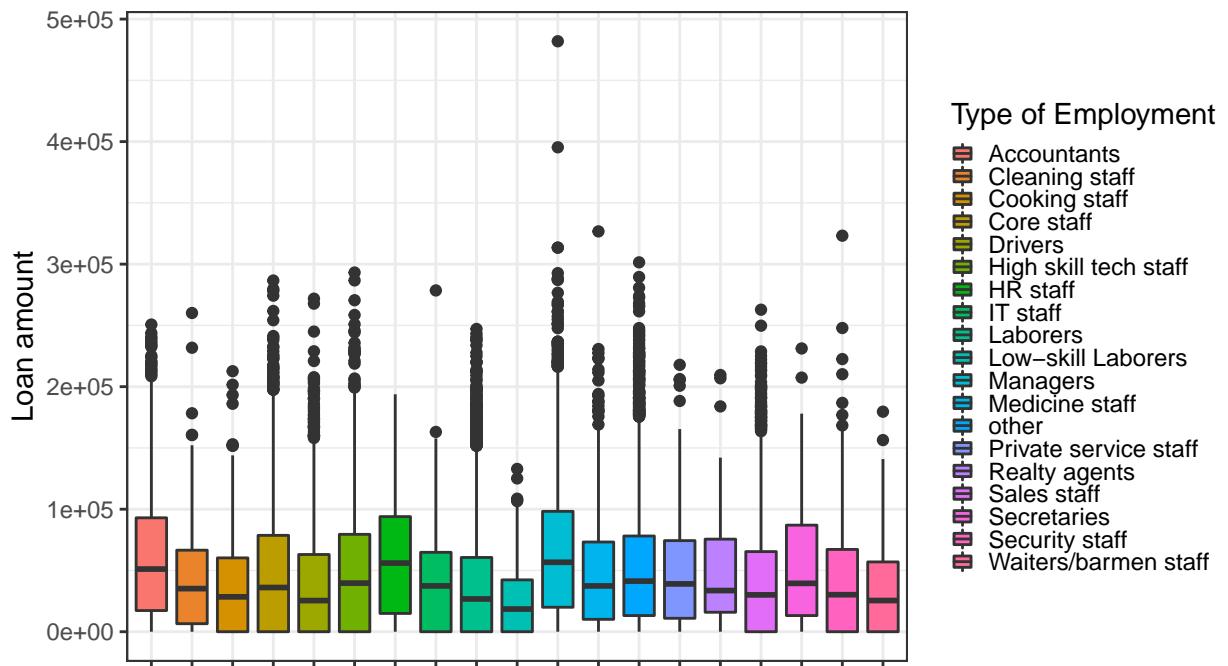
The pensioner and other class seem to have more chance to get their requested loan approved.

#### 2.4.10 Type of employment



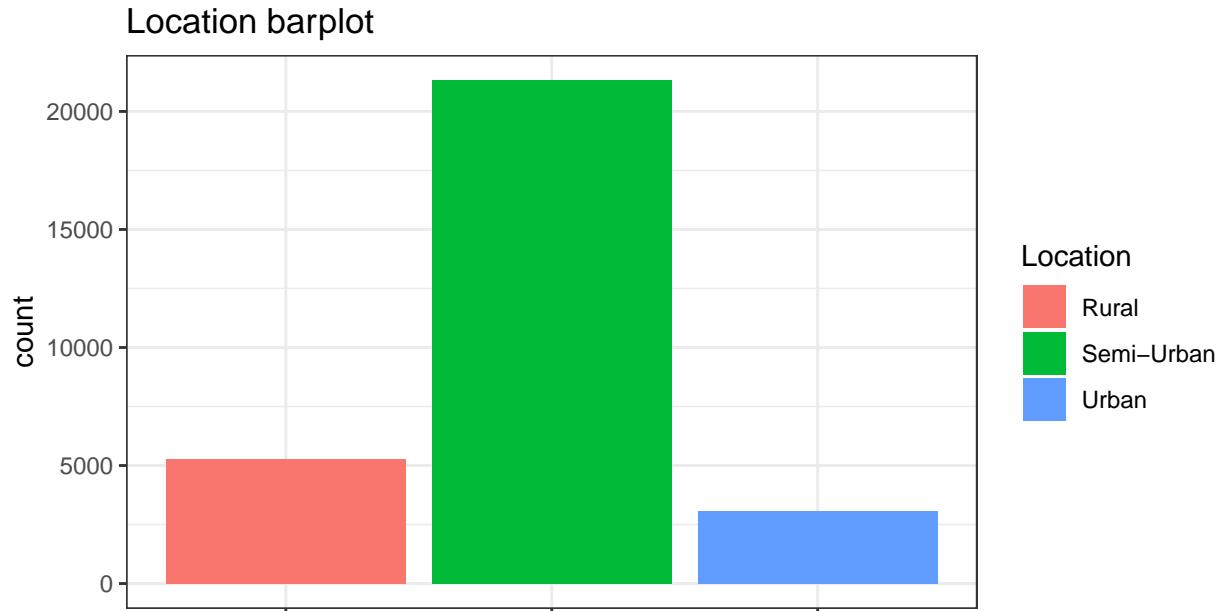
The number of applicants differs largely depending on the type of employment.

Boxplot of loan sanction vs type of employment



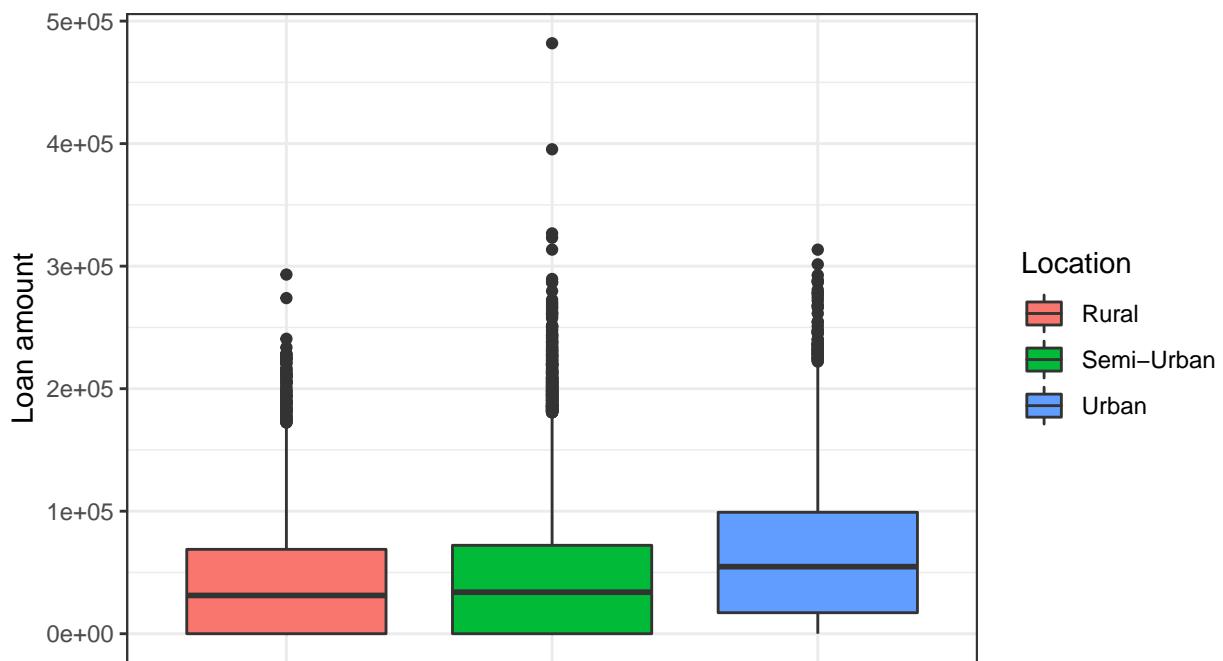
The distribution of the loan approved differs slightly from type of employment to other. We notice that some types have the chance of getting their request rejected more than others. This could help us define which customers are likely to have their loan sanction amount to be null.

#### 2.4.11 Location



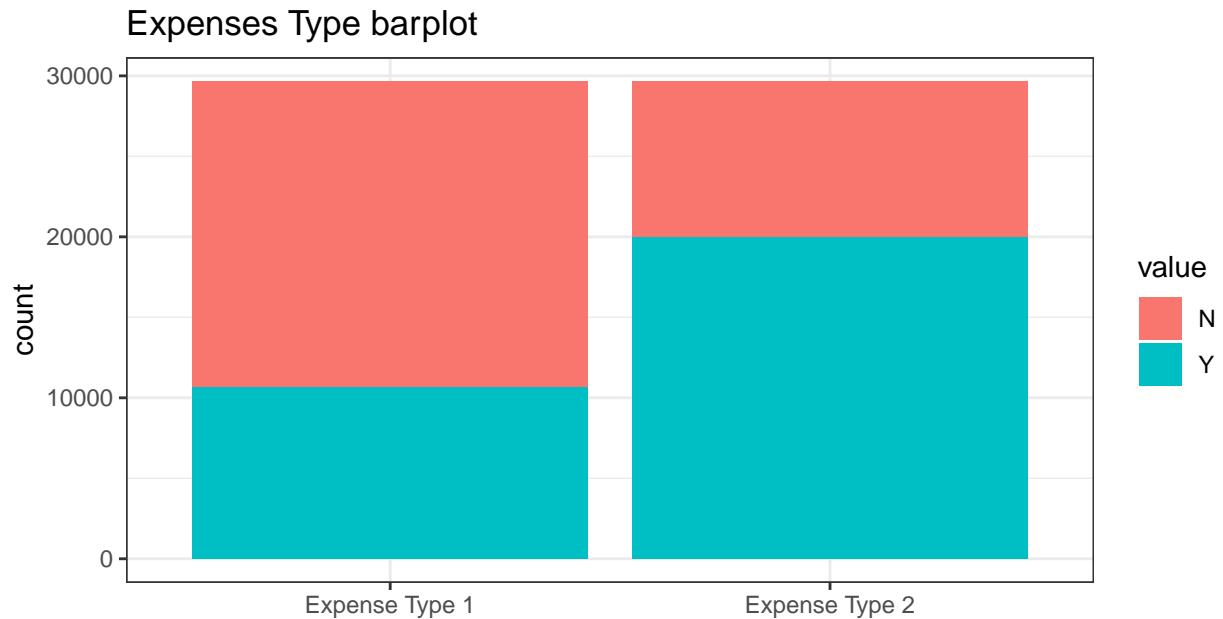
Most customers are located in semi\_urbain area.

Boxplot of loan sanction vs Location



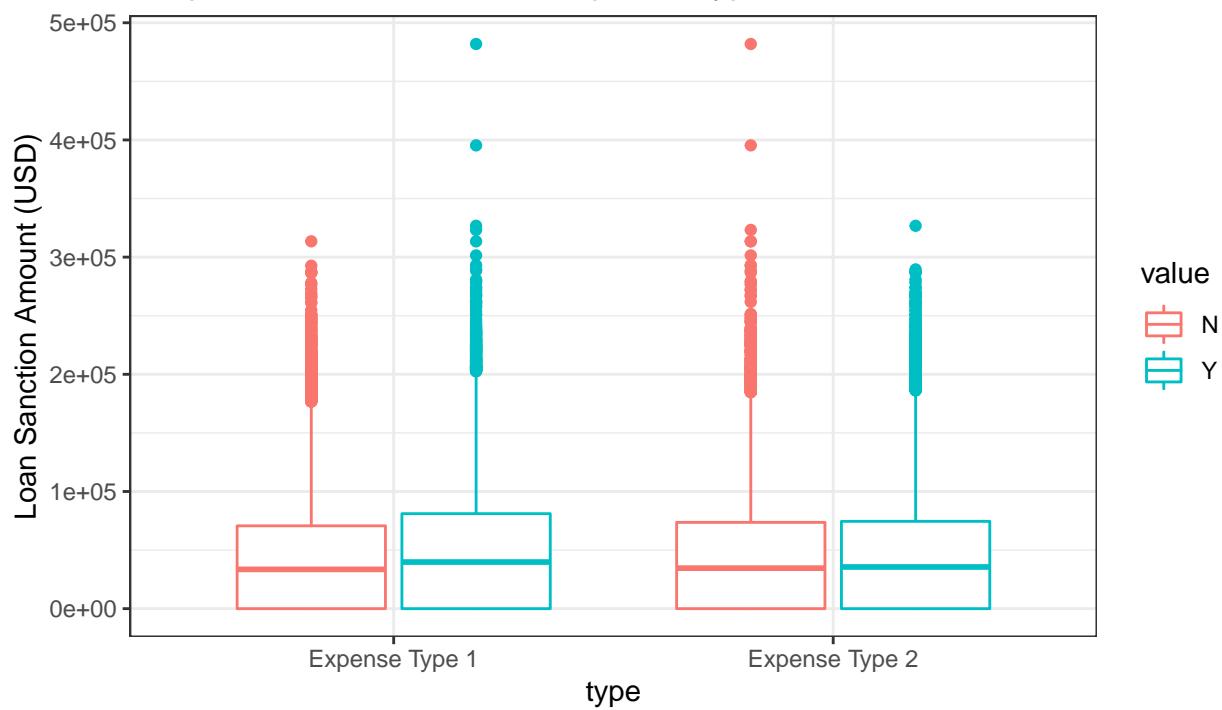
We see from the boxplot that customers which are located in Urban area are most likely to get their loan request approved and that with a higher loan amount.

#### 2.4.12 Expenses types :



We notice that Customers have expenses of type 2 more than they have of type1.

Boxplot of loan sanction vs Expense Type 1



As we see the type of expenses on its own doesn't seem to affect the outcome of the loan sanction amount.

### 3 Data processing :

#### 3.1 Drop no needed features in models building

As we don't need the property ID we will drop it for now. The customer ID is unique and sufficient to make final submission in the case we wanted to. The customer ID will be dropped while making train and test sets.

#### 3.2 Binarize categorical features

We have already cleaned our data, so now we will binarize the categorical features and scale the continuous ones.

The binarization of all the categorical data will make easy to work with a variety of models.

**After binarization the data now looks like**

| Customer ID | Current Loan Expenses (USD) | Cooking staff | HR staff | Managers | other | Sales staff | Security staff |
|-------------|-----------------------------|---------------|----------|----------|-------|-------------|----------------|
| C-36995     | 241.08                      | 0             | 0        | 0        | 0     | 1           | 0              |
| C-33999     | 495.81                      | 0             | 0        | 0        | 1     | 0           | 0              |
| C-3770      | 171.95                      | 0             | 0        | 0        | 1     | 0           | 0              |
| C-26480     | 298.54                      | 0             | 0        | 0        | 1     | 0           | 0              |
| C-23459     | 491.41                      | 0             | 0        | 0        | 0     | 0           | 0              |
| C-17688     | 181.48                      | 0             | 0        | 0        | 0     | 0           | 0              |

#### 3.3 Scale continuous features except the outcome

The scale function will permit us to center and scale continuous features. This is very important since it allows us to normalize the variation in the data.

##### 3.3.1 The data will look like this

| Customer ID | Income Stability | Loan Amount Request (USD) | Credit Score | IT staff | Security staff |
|-------------|------------------|---------------------------|--------------|----------|----------------|
| C-36995     | 0                | -0.2686989                | 0.98591465   | 0        | 0              |
| C-33999     | 0                | -0.7053444                | 0.57442182   | 0        | 0              |
| C-3770      | 1                | -0.7262659                | 1.32188211   | 0        | 0              |
| C-26480     | 1                | -0.1468391                | 1.31550567   | 0        | 0              |
| C-23459     | 0                | 0.4214259                 | 0.08060208   | 0        | 0              |
| C-17688     | 0                | -0.9138606                | -0.78985263  | 0        | 0              |

#### 3.4 Split data into train and test set

We will split the data in a training and testing sets and define the features and outcomes for each one. After splitting the data set we get :

train\_x : the training features used to train models

train\_y : the training outcome used for training

test\_x : the testing features used to get predictions

test\_y : the testing outcome used to measure the RMSE

## 4 Building models

**Metric used :** we will use the RMSE function of the caret package

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$$

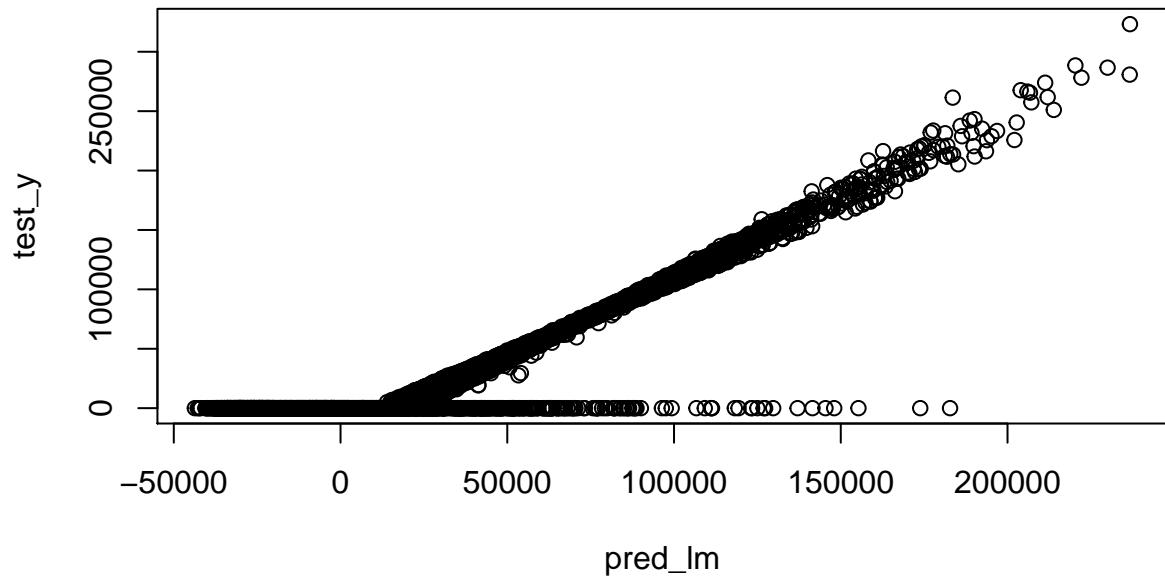
### 4.1 Linear regression

We will try a linear model with `lm()` under `caret::` using only the continuous features.

#### 4.1.1 Get results

After training the model with cross validation we apply it on the test set and get the rmse.

```
## The rmse with lm is 19247.46
```



we see that the predictions for values different than zero are good but for the ones near zero they are not, it would be better to fit a model to predict null targets first in the case we wanted to perform a linear model with good result.

## 4.2 Rpart model

### 4.2.1 Get results

After training the model with cross validation we apply it on the test set and get the rmse.

```
## The rmse with rpart model is 18176.69
```

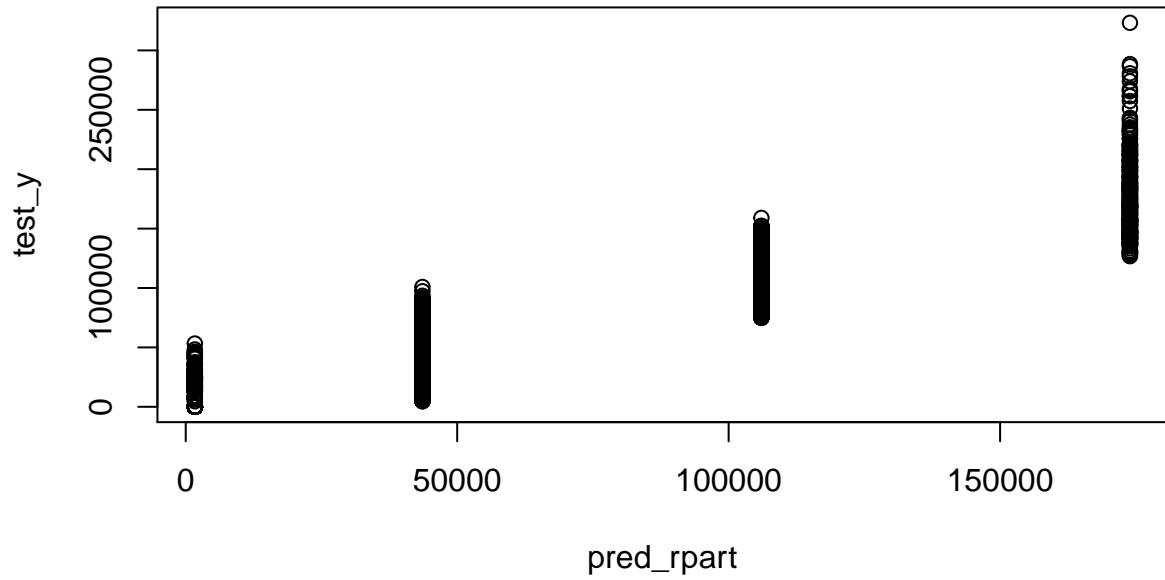


Table 23: Results

| method | rmse     |
|--------|----------|
| lm     | 19247.46 |
| rpart  | 18176.69 |

We see that the rmse has dropped less than lm but not enough, we are still making big mistakes.

### 4.3 Random forest

#### 4.3.1 Get results

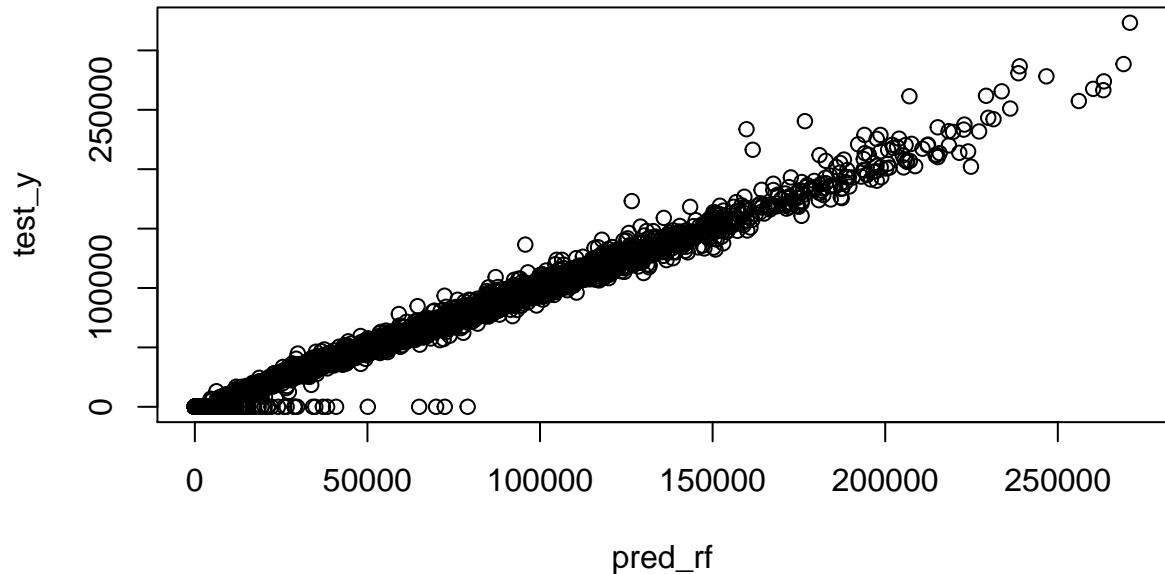
After training the model with cross validation we apply it with the optimal tunes on the test set and get the rmse. We could have tuned the model more but this would have taken much more time.

```
## The rmse with Random forest is 5134.205
```

Table 24: Results

| method | rmse      |
|--------|-----------|
| lm     | 19247.457 |
| rpart  | 18176.689 |
| rf     | 5134.205  |

We see that the results with Rf are much better, now we are not making big mistakes like the other models.



As we see the predictions now are correlated with the loan amount, but we are still making mistakes especially around the zero loan amount, can we do better.

## 4.4 Xgboost model

We will first process our data to fit the Xgboost model, we use cross-validation with Xgboost to find the best params, we train the model on the training set and finally apply it on the test set to get the predictions.

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

### 4.4.1 Get the best nround.

```
##      iter train_rmse_mean train_rmse_std test_rmse_mean test_rmse_std
## 1:  150       183.7088      6.037589     939.9707     477.4767
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

### 4.4.2 Get results

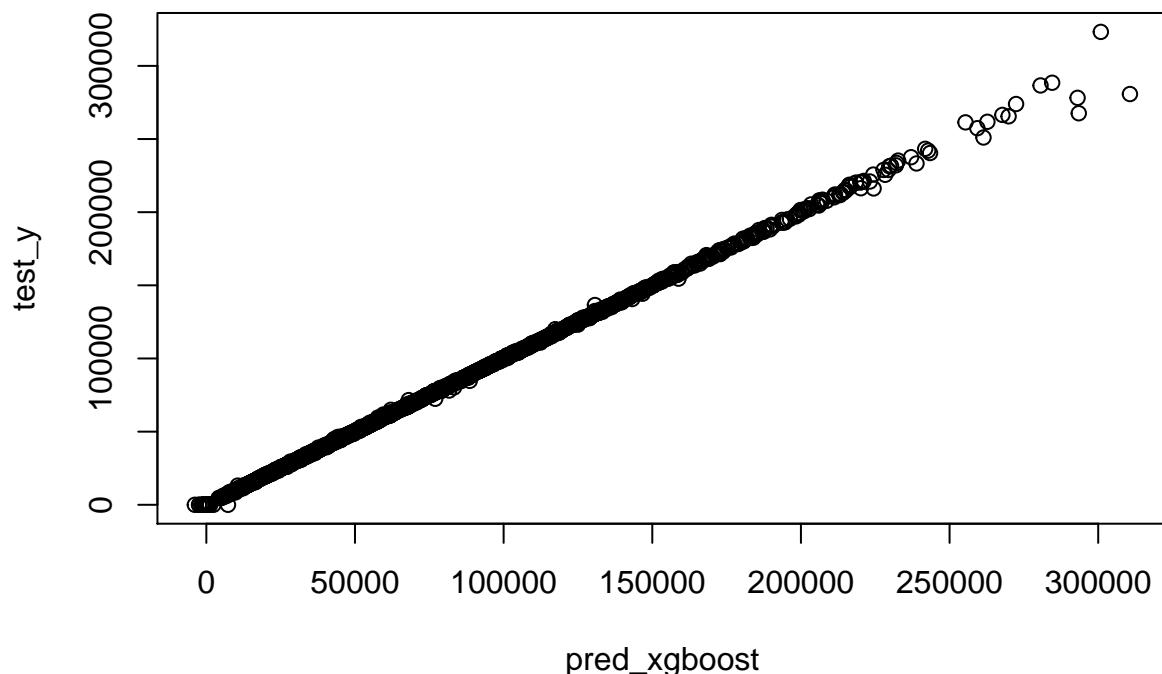
After applying the model on test set we get the predictions and rmse.

```
## The rmse with Xgboost is 765.672272548159
```

Table 25: Results

| method  | rmse       |
|---------|------------|
| lm      | 19247.4574 |
| rpart   | 18176.6894 |
| rf      | 5134.2053  |
| xgboost | 765.6723   |

We can see now clearly that with extreme gradinat boosting the results are far better than the other models. The rmse is now small and in the area of home loans is very good.



We can see that the predictions are close to the target, we are not making big mistakes anymore.

## **5 Conclusion :**

The project goal was to build a model that predict the amount home loan to be approved for a bank customer. Our approach undertaken here relied on an initial exploratory analysis with data visualizations, then we start building models based on the assumptions made in the last part.

The overall objective to training a machine learning algorithm of course resides in being able to generate predictions. With Xgboost model, we managed to reach an RMSE of 766, which is a very good results since the amounts of home loan are much more bigger.

We still could get better results if we tune more parameters. An approach based on making first a classification, of which customer could get a loan and which will have his request rejected, before applying regression to find the exact amount would make the results better.