Regression Analysis

Leo Artoni, Riccardo Paolantoni

19/1/2023

1. Describe the dataset

The dataset that we have chosen is "BrazHousesRent.cvs", it contains information on 10692 houses in various Brazilian cities. As already specified by the description of the dataset the data has been gathered through a *web-crawler* therefore it presents some complications that will be further discussed later. With the function *glimpse()* we can take an actual glimpse at the data and at the different types of variables that we are working with. By looking at the summary we can already notice some particular numbers in some of the variables such as the big leap between the 3rd quartile and the maximum value of the variables *hoa*, *property_tax* and *area*. We also checked for the amount of zeros in the dataset and displayed the percentage of each variable, here we faced another problem, the variables *parking_space*, *hoa* and *property_tax* have a high percentage of zeros (~25%, ~22% and ~15%), this substantial presence of zeros could be explained by the fact that some houses may not have any parking spaces and others may not be in any Homeowners association, but we are not sure how to explain this phenomenon with regards to property tax, this could be due to difficulties that the web-crawler had in order to get this type of information. This being said, we decided to remove any observations with zeros in both *hoa* **and** *property_tax*. After checking for NAs and removing some duplicates a new variable called *data* is created, this will be the variable used along all steps of the analysis.

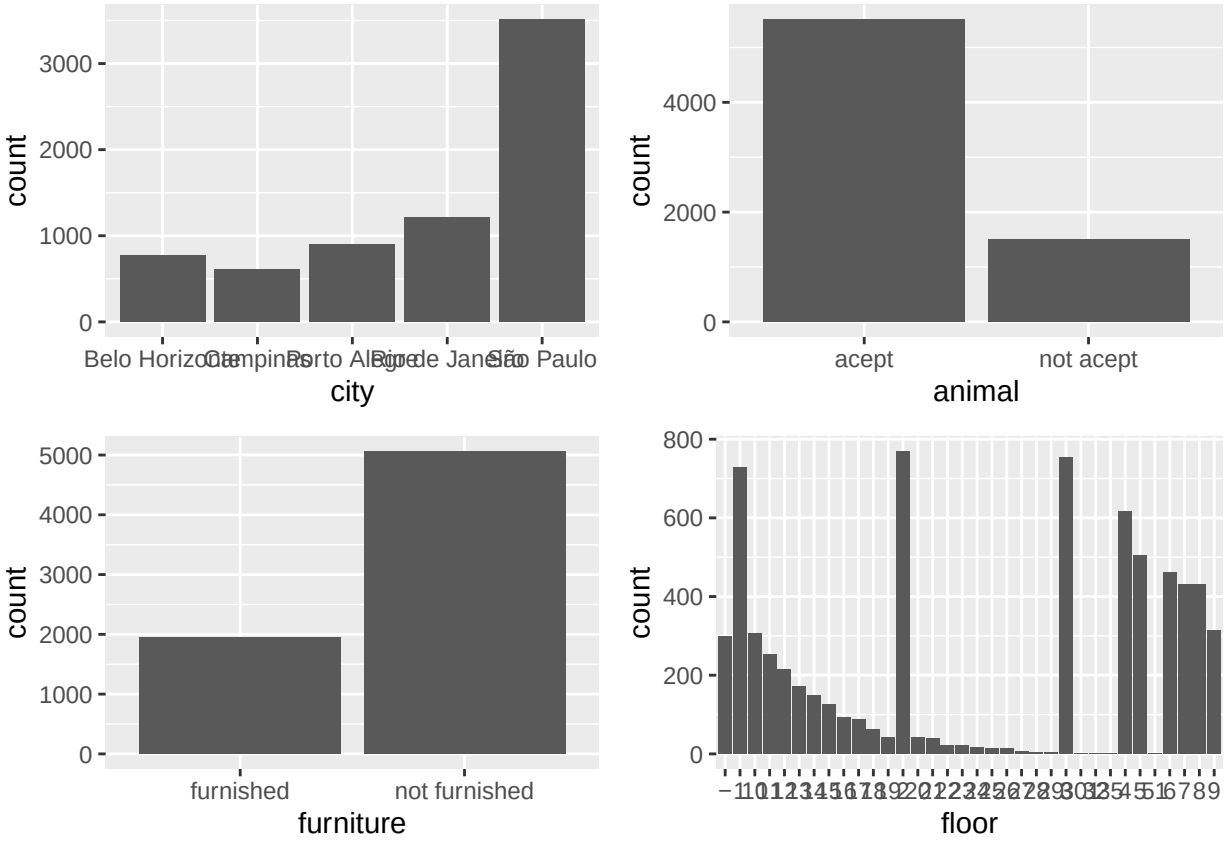
```
## Rows: 10,692
## Columns: 12
## $ city           <chr> "São Paulo", "São Paulo", "Porto Alegre", "Porto Alegre~
## $ area           <int> 70, 320, 80, 51, 25, 376, 72, 213, 152, 35, 26, 46, 36,~
## $ rooms          <int> 2, 4, 1, 2, 1, 3, 2, 4, 2, 1, 1, 1, 1, 1, 2, 4, 2, 2, 2~
## $ bathroom       <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
## $ parking_spaces <int> 1, 0, 2, 0, 0, 2, 0, 2, 1, 0, 0, 2, 0, 2, 2, 2, 1, 1, 0~
## $ floor          <chr> "7", "20", "6", "2", "1", "-", "7", "4", "3", "2", "2",~
## $ animal         <chr> "acept", "acept", "acept", "acept", "not acept", "acept~
## $ furniture      <chr> "furnished", "not furnished", "not furnished", "not fur~
## $ hoa            <int> 2065, 1200, 1000, 270, 0, 0, 740, 2254, 1000, 590, 470,~
## $ rent_amount    <int> 3300, 4960, 2800, 1112, 800, 8000, 1900, 3223, 15000, 2~
## $ property_tax   <int> 211, 1750, 0, 22, 25, 834, 85, 1735, 250, 35, 150, 43, ~
## $ fire_insurance <int> 42, 63, 41, 17, 11, 121, 25, 41, 191, 30, 27, 8, 27, 54~
```

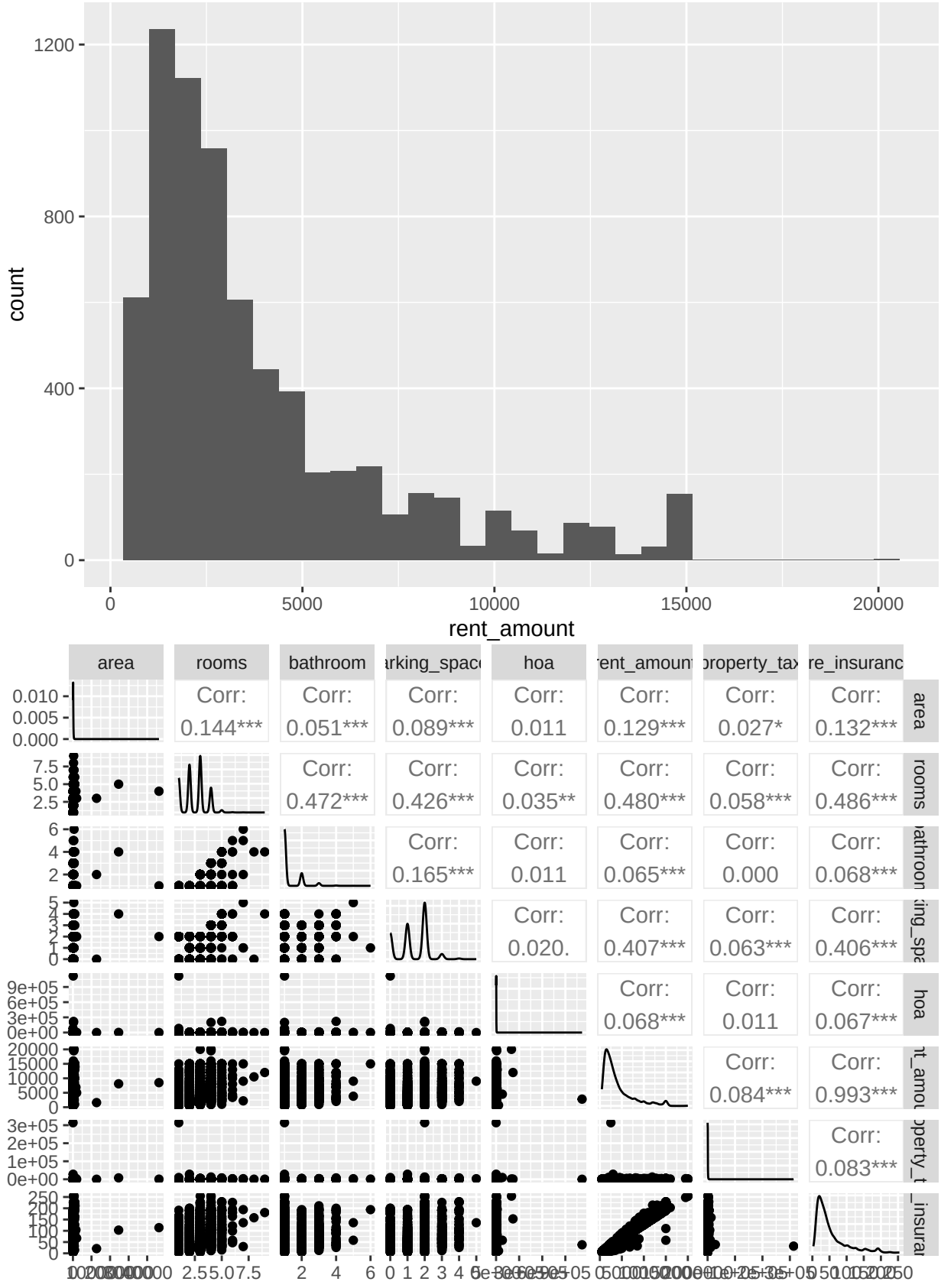
```
##      city                area             rooms           bathroom      
##  Length:10692       Min.   :   11.0   Min.   : 1.000   Min.   :1.000  
##  Class :character   1st Qu.:   56.0   1st Qu.: 2.000   1st Qu.:1.000  
##  Mode  :character   Median :   90.0   Median : 2.000   Median :1.000  
##                     Mean   :  149.2   Mean   : 2.506   Mean   :1.275  
##                     3rd Qu.:  182.0   3rd Qu.: 3.000   3rd Qu.:1.000  
##                     Max.   :46335.0   Max.   :13.000   Max.   :9.000  
##  parking_spaces    floor              animal           furniture        
##  Min.   :0.00   Length:10692       Length:10692       Length:10692      
```

```
##  1st Qu.:0.00   Class :character   Class :character   Class :character
##  Median :1.00   Mode  :character   Mode  :character   Mode  :character
##  Mean   :1.31
##  3rd Qu.:2.00
##  Max.   :8.00
##       hoa           rent_amount     property_tax      fire_insurance
##  Min.   :      0   Min.   :  450   Min.   :     0.0   Min.   :  3.0
##  1st Qu.:    170   1st Qu.: 1530   1st Qu.:    38.0   1st Qu.: 21.0
##  Median :    560   Median : 2661   Median :   125.0   Median : 36.0
##  Mean   :   1174   Mean   : 3896   Mean   :   366.7   Mean   : 53.3
##  3rd Qu.:   1238   3rd Qu.: 5000   3rd Qu.:   375.0   3rd Qu.: 68.0
##  Max.   :1117000   Max.   :45000   Max.   :313700.0   Max.   :677.0
```

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

```
## [1] 194
```





In the plots above *ggplot* and *gridExtra* are used in order to better visualize the categorical variables and the

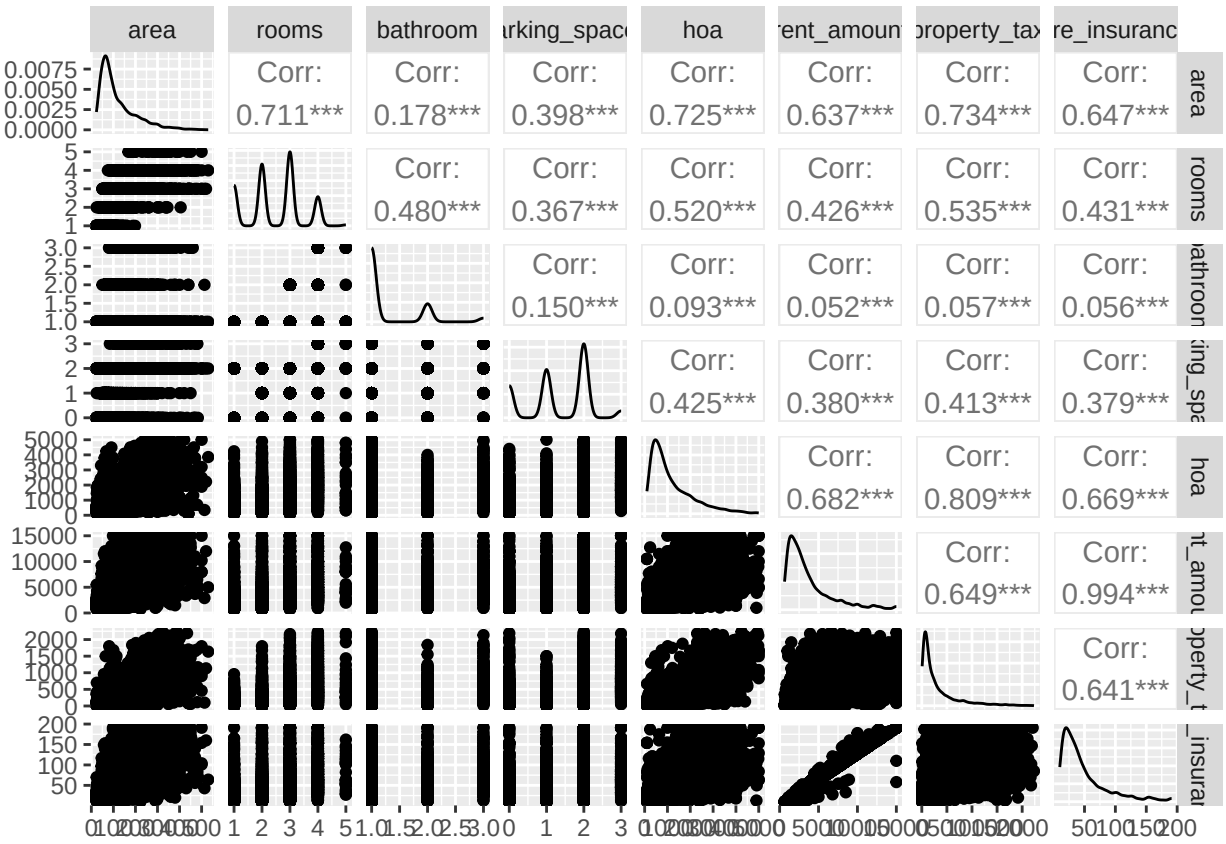
target variable that we are interested in. From these plots it is possible to observe a disparity in the variable *city* where the city *São Paulo* is by far the most predominant observation, and that the *floor* variable is quite messy with most of its observations being that the houses are placed at the ground floor. We also used the function *ggpairs()* on all numerical variables in order to better visualize the distribution of the observations, this allowed us to notice that there are many outliers that skew the distribution.

2. Data manipulation

In the data manipulation phase we have first converted all the *character variables* into *factor*, removed the outliers by using the library *dataPreparation;* the function *remove_percentile_outlier()* has also been used here. (note that *remove_sd_outliers()* could have been used in this context but both functions lead to similar results). We have then decided to create a new variable called *house_size* that simply takes the values from *rooms*, *bathroom*, *parking_spaces* and adds them together and assigns them to a category depending on the magnitude of the value (ex. 3 = "very small" and 30 = "very big").

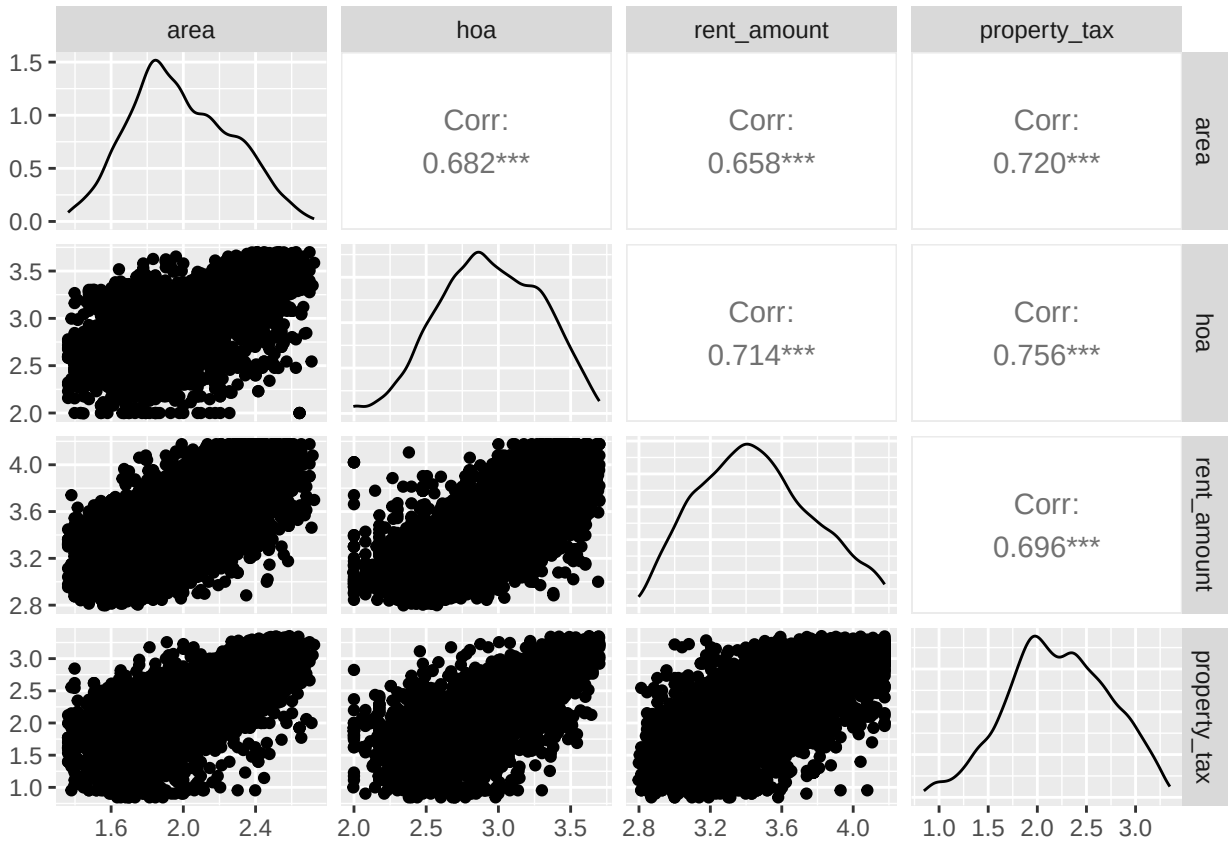
```
## [1] "city"      "floor"     "animal"    "furniture"
## [1] "remove_percentile_outlier: c(\"city\", \"floor\", \"animal\", \"furniture\") aren't columns of t
## [1] "remove_percentile_outlier: I start to filter categorical rare events"
## [1] "remove_percentile_outlier: dropped 126 row(s) that are rare event on area."
## [1] "remove_percentile_outlier: dropped 9 row(s) that are rare event on rooms."
## [1] "remove_percentile_outlier: dropped 20 row(s) that are rare event on bathroom."
## [1] "remove_percentile_outlier: dropped 19 row(s) that are rare event on parking_spaces."
## [1] "remove_percentile_outlier: dropped 136 row(s) that are rare event on hoa."
## [1] "remove_percentile_outlier: dropped 68 row(s) that are rare event on rent_amount."
## [1] "remove_percentile_outlier: dropped 133 row(s) that are rare event on property_tax."
## [1] "remove_percentile_outlier: dropped 95 row(s) that are rare event on fire_insurance."
## [1] "remove_percentile_outlier: 606 have been dropped. It took 0.01 seconds. "
```

```
## house_size
## Very_Small      Small     Normal        Big   Very_Big
##        962       1948       2125        853        510
```



3. Variables selection and dealing with Skewness

```
##         area          hoa  rent_amount property_tax
##     1.542718     1.406848     1.681585     2.135134
```



Looking at the *ggpairs* it is possible to detect an unusual pattern relating *fire_insurance* and *rent_amount* (with a correlation of almost 99%), this has led us to remove it from the variables; along with *rooms*, *bathroom*, *parking_spaces:* that are now stored inside *house_size;* and the variable *floor*. We have then employed the library *moments* to compute the skewness of the variables. As we can see the variables are mostly positively skewed (right skewed) this is why in the next step a **square root transformation** has been applied to the variables. Parametric methods, assume that there is an approximately normally distributed dataset. This is why we can consider transformations of either the independent or dependent variable or both in order to obtain a linear relationship. After the transformation we see an improved distribution across all variables.

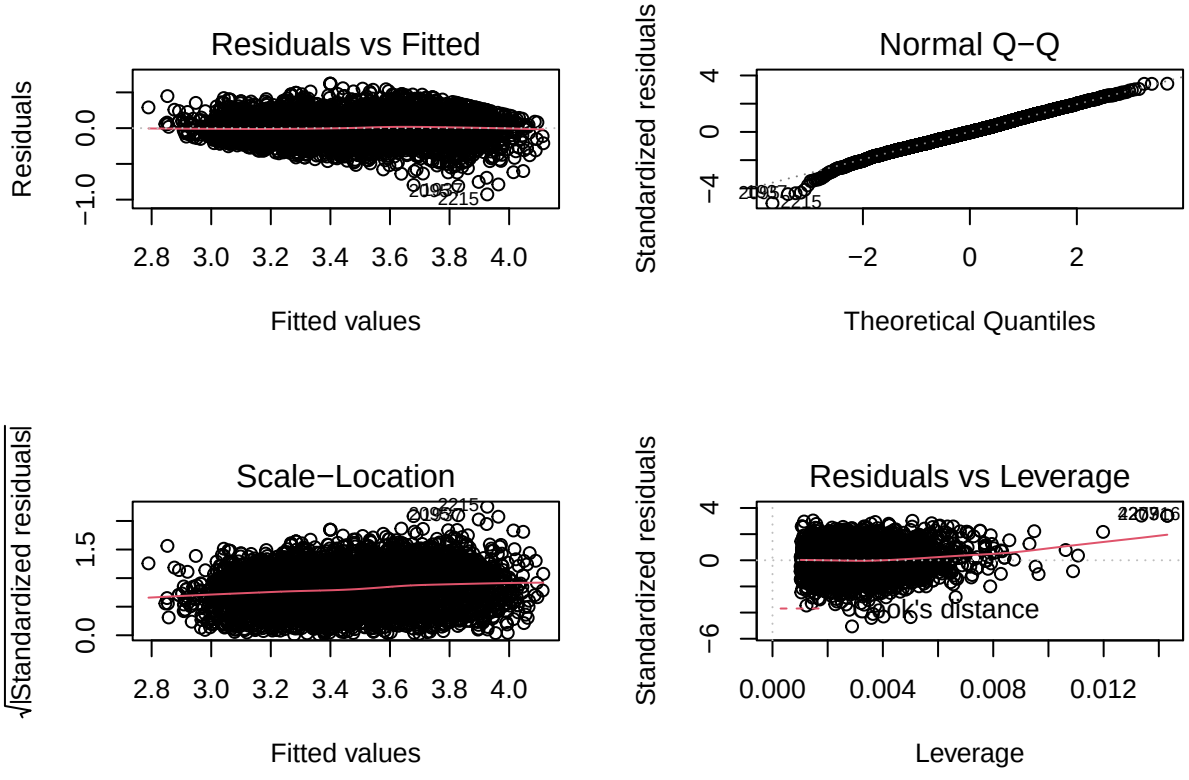
4. Build a predictive model

This task requires to build a predictive model in order to find out the rent amount according to the house specifics. First of all, we have decided to build a *Multiple linear regression* model as a "*benchmark*". In order to validate the results of the prediction we used as a resampling method the *validation set* approach and divided the dataset into two "*subsets*": 70% of the data has been used as **training set** and the remaining 30% will be the **validation** or **test set**.

```
##
## Call:
## lm(formula = rent_amount ~ ., data = data.train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.92659 -0.11913 -0.00329  0.12221  0.62268
##
## Coefficients:
```

```
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 1.874482   0.033424  56.081  < 2e-16 ***
## cityCampinas               -0.048288   0.013009  -3.712 0.000208 ***
## cityPorto Alegre           -0.004235   0.011839  -0.358 0.720557
## cityRio de Janeiro          0.050065   0.011748   4.262 2.07e-05 ***
## citySão Paulo               0.119439   0.010228  11.677  < 2e-16 ***
## area                        0.372820   0.018055  20.649  < 2e-16 ***
## animalnot acept             0.019524   0.006812   2.866 0.004173 **
## furniturenot furnished     -0.133757   0.006310 -21.197  < 2e-16 ***
## hoa                         0.196016   0.014345  13.664  < 2e-16 ***
## property_tax                0.122159   0.009401  12.994  < 2e-16 ***
## house_sizeSmall             0.019864   0.009111   2.180 0.029300 *
## house_sizeNormal            0.029867   0.010808   2.764 0.005741 **
## house_sizeBig               0.031726   0.013032   2.434 0.014954 *
## house_sizeVery_Big          0.029937   0.015471   1.935 0.053049 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1835 on 4464 degrees of freedom
## Multiple R-squared:  0.6573, Adjusted R-squared:  0.6564
## F-statistic: 658.8 on 13 and 4464 DF,  p-value: < 2.2e-16
```

```
## [1] 0.03377791
```



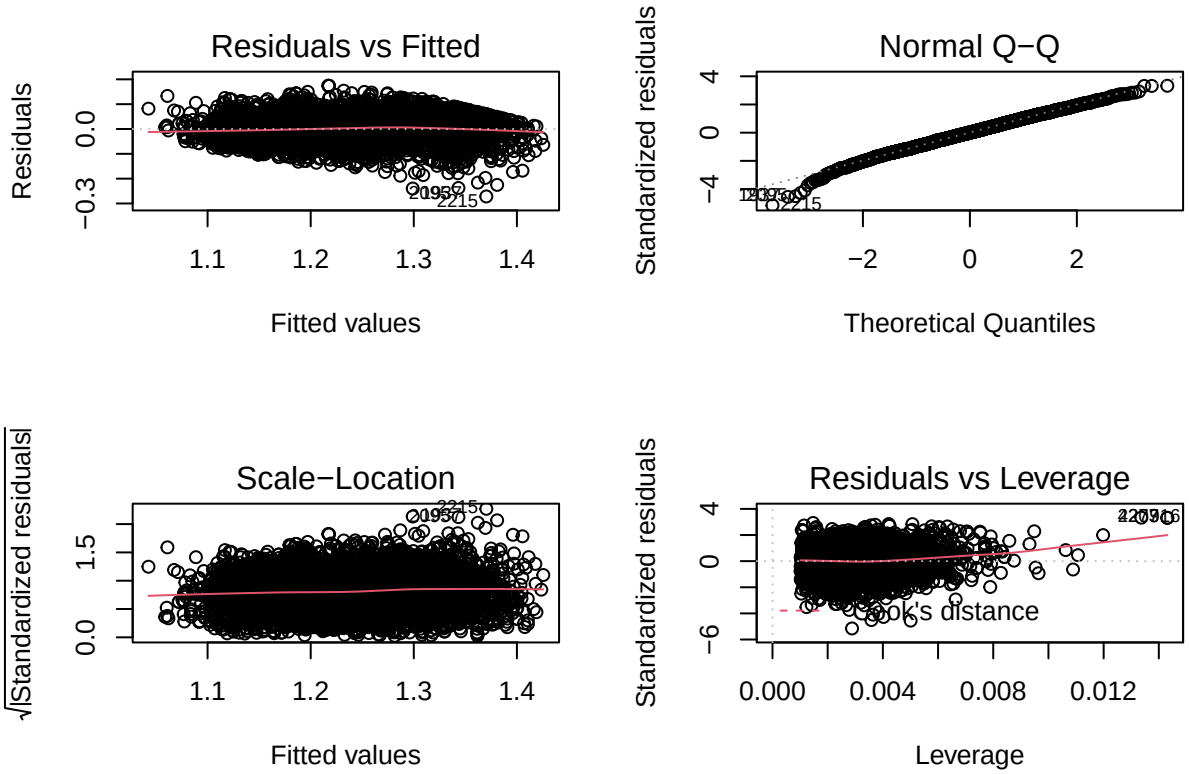

```
##                  GVIF Df GVIF^(1/(2*Df))
## city         1.735533  4        1.071344
## area         3.330220  1        1.824889
## animal       1.038458  1        1.019048
```

```
## furniture    1.072217  1        1.035479
## hoa          3.267009  1        1.807487
## property_tax 3.053277  1        1.747363
## house_size   2.533845  4        1.123240
```

As we can see from the summary of the model the majority of the coefficients are significant (with a small p-value) except for **city** which has the value *Porto Alegre* that is not significant, this may be related to the fact that there are not many observation in *Porto Alegre*. Both the **R-squared** and the Adjusted **R-squared** are somewhat good considering the nature of the dataset with 65% of variance explained. We then computed the MSE by using the values of rent amount stored in *data.valid* (the test set) and the predicted values from the model resulting in a good value of 0.033. In addition to the model we decided to check for **multicollinearity** since the variables are all highly correlated. By using the function *vif()* we computed the variance inflation factor, which measures how much the variance of a regression coefficient is inflated due to multicollinearity in the model and concluded that the values are acceptable in order to assume that there is no multicollinearity. If we look at the plot of the residuals is it possible to notice a slight **funnel shape** that may suggests **heteroscedasticity**.

```
##
## Call:
## lm(formula = log(rent_amount) ~ ., data = data.train)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.271721 -0.034755  0.000435  0.036070  0.174536
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)               0.788146   0.009618  81.941  < 2e-16 ***
## cityCampinas             -0.015174   0.003744  -4.053 5.14e-05 ***
## cityPorto Alegre         -0.001832   0.003407  -0.538 0.590888
## cityRio de Janeiro        0.016409   0.003381   4.854 1.25e-06 ***
## citySão Paulo             0.035747   0.002943  12.145  < 2e-16 ***
## area                      0.103170   0.005196  19.857  < 2e-16 ***
## animalnot acept           0.005711   0.001960   2.913 0.003592 **
## furniturenot furnished -0.038903   0.001816 -21.424  < 2e-16 ***
## hoa                       0.055487   0.004128  13.442  < 2e-16 ***
## property_tax              0.035229   0.002705  13.022  < 2e-16 ***
## house_sizeSmall           0.008066   0.002622   3.076 0.002108 **
## house_sizeNormal          0.012428   0.003110   3.996 6.54e-05 ***
## house_sizeBig             0.013106   0.003750   3.495 0.000479 ***
## house_sizeVery_Big        0.013501   0.004452   3.032 0.002440 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.05281 on 4464 degrees of freedom
## Multiple R-squared:  0.6588, Adjusted R-squared:  0.6578
## F-statistic:   663 on 13 and 4464 DF,  p-value: < 2.2e-16
```

```
## [1] 4.982108
```



After performing a logarithmic transformation of the dependent variable there is no observable significant change in the shape or in the results of the model, other than a very small increase of the R-squared.

We then fitted a **Ridge regression**. The ridge regression needs a *model matrix* in order to work since it only takes numerical values (it transforms all categorical variables into dummy variables). [The *glmnet()* function takes two specific arguments: the **alpha** argument which specifies the type of model that we want to implement (ridge or lasso) and **lambda**, a parameter unique to lasso and Ridge regressions. In this case, we have implemented a **grid** of values for lambda that goes from -2 to 10.

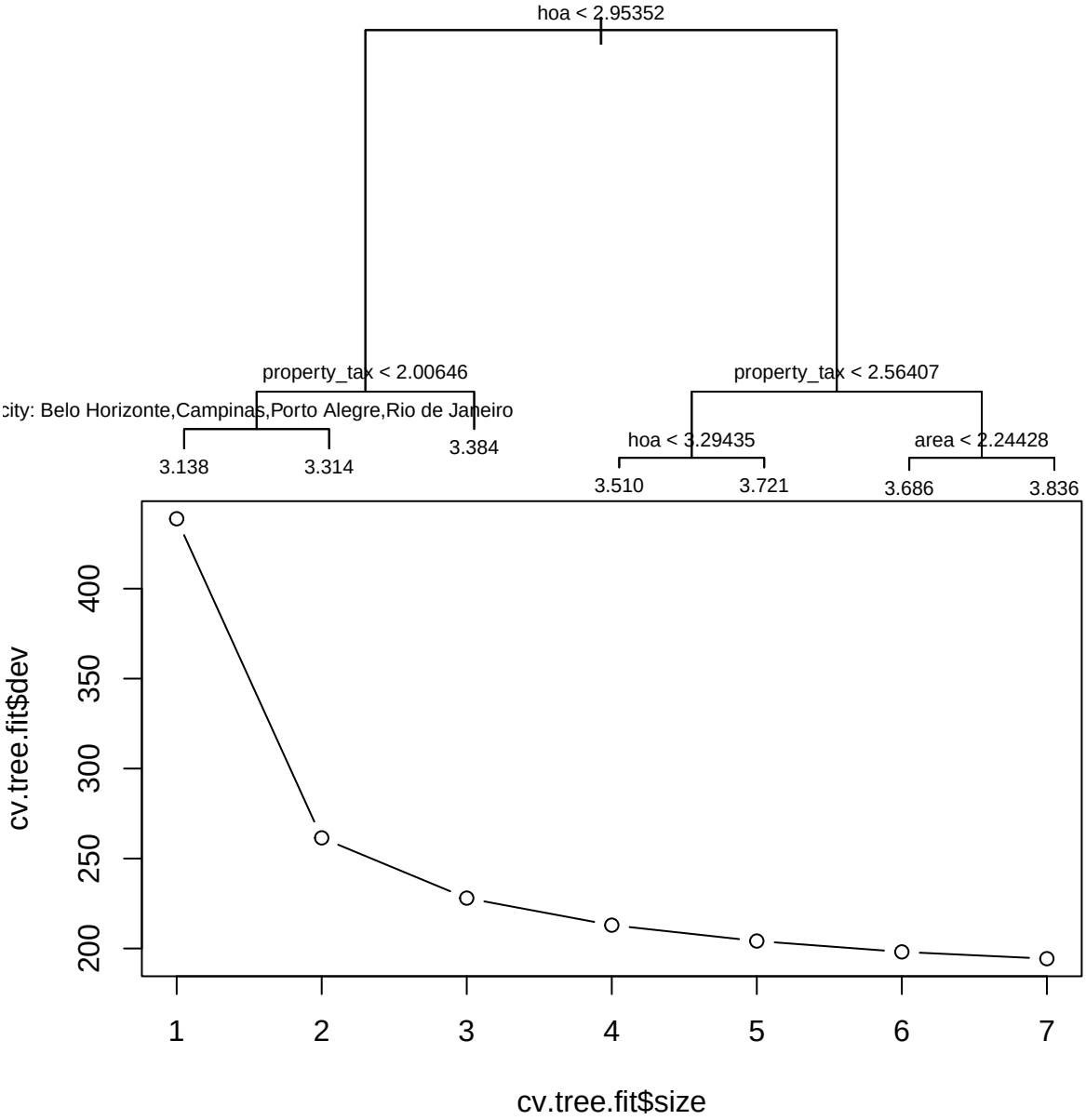
```
## [1] 0.02225211
```

```
## [1] 0.03381409
```

```
## 14 x 1 sparse Matrix of class "dgCMatrix"
##                                    s1
## (Intercept)                1.94252361
## cityCampinas              -0.06124813
## cityPorto Alegre          -0.02060942
## cityRio de Janeiro         0.02960327
## citySão Paulo              0.10056243
## area                       0.33002751
## animalnot acept            0.01694600
## furniturenot furnished    -0.12553545
## hoa                        0.20467453
## property_tax               0.12356031
## house_sizeSmall            0.01300950
## house_sizeNormal           0.03165089
## house_sizeBig              0.03434370
## house_sizeVery_Big         0.03386524
```

The results of the Ridge regression are slightly better than the standard multiple linear regression (as we can see from the MSE). In this code the optimal value for **lambda** was computed through cross-validation and is displayed above (~0.022). We decided to implement the Ridge regression (with respect to the lasso regression) because we have already done some variable selection and we considered the remaining ones to be significant.

```
##
## Regression tree:
## tree(formula = rent_amount ~ ., data = data.train)
## Variables actually used in tree construction:
## [1] "hoa"          "property_tax" "city"         "area"
## Number of terminal nodes:  7
## Residual mean deviance:  0.04145 = 185.3 / 4471
## Distribution of residuals:
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -0.836900 -0.137500 -0.004095  0.000000  0.136000  0.883600
```



```
## [1] 0.04156978
```

After implementing the Ridge regression, we decided to implement both a **regression tree model** and a **Random forest model**. As one can see from the summary, the resulting tree is composed of 7 *leafs*, and only 4 variables are actually used in the model: *hoa* ,that is also the first split (minimizes the most the RSS), *property_tax*, *city* and *area*. The plot gives us insight on the splits of the tree and the ending leaves. We decided to not implement *pruning* because the function **cv.tree()** selects (via cross-validation) the most complex tree composed of 7 ending leafs. The performance of the model is quite good on the validation set, positioning at the second position, after the multiple linear regression with an MSE = 0.0415.

```
##
## Call:
##  randomForest(formula = rent_amount ~ ., data = data.train, mtry = 3,      importance = T, ntree = 1
##                Type of random forest: regression
##                      Number of trees: 1000
## No. of variables tried at each split: 3
##
##           Mean of squared residuals: 0.0295459
##                     % Var explained: 69.84
```

```
## [1] 0.02983786
```

Finally, we have the **random forest**. We built a forest composed of 1000 trees and only used 3 variables for each split since it is standard for regression random forests to use the number of variables divided by 3 (p/3 where p is the number of variables), using all variables (**bagging**) led to worse results (less variance and worst MSE) which may be due to overfitting in the model. As one could guess, the results of the random forest are the best overall.

Conclusions

As a conclusion, the model with the best results is definitely the **Random Forest** but it is important to note that the **variance explained** and the **MSE** are not too far from the way simpler models of **multiple linear regression** and **ridge regression**, this may be due to the way the dataset is constructed and how the data is scaled/manipulated (tree models perform significantly better on scaled data). In the previous analysis, any dimensional reduction methods such as best subset selection or step-wise selection are not present since an heavy *a priori* variable selection was made in order to clean the dataset. After observing the selected variables in the different models, we can first say that **regression tree** shows the 7 most significant ones. This being said, if we were to reduce this number, the most important characteristics in order to determine the apartments with the highest rent prizes in Brazil are the *Monthly Homeowners Association Tax* and the *Property Tax*