Taiwan Real Estate

Pierre FLINE

15/02/2022

Contents

1 Introduction

This report is produced as the final report to complete the *Data Science Professional Certificate*, an online program provided by Harvard University through edx platform.

The assignment chosen is to create a model to predict the price of real estate properties in Taiwan with the best possible accuracy. The accuracy is here measured as the Residual Mean Squared Error (RMSE), which we can calculate this way:

$$RMSE = \sqrt{\frac{1}{N}\sum^{N}(y - y')^2}$$

where:
`y`: real price (outcome)
`y'`: predicted price (prediction)
N: number of prices (observations)

The original data set is provided by the University of California (Irvine Campus).

2 Data creation

The original data set is downloaded from the website of the University of California (Irvine Campus), on this webpage: https://archive.ics.uci.edu/ml/datasets/Real+estate+valuation+data+set#.

3 Data exploration/wrangling/visualisation

There is no NA in the data set.

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

Structure of the data set:

```
## tibble [414 x 8] (S3: tbl_df/tbl/data.frame)
##  $ No                                    : num [1:414] 1 2 3 4 5 6 7 8 9 10 ...
##  $ X1 transaction date                   : num [1:414] 2013 2013 2014 2014 2013 ...
##  $ X2 house age                          : num [1:414] 32 19.5 13.3 13.3 5 7.1 34.5 20.3 31.7 17.9 .
##  $ X3 distance to the nearest MRT station: num [1:414] 84.9 306.6 562 562 390.6 ...
##  $ X4 number of convenience stores       : num [1:414] 10 9 5 5 5 3 7 6 1 3 ...
##  $ X5 latitude                           : num [1:414] 25 25 25 25 25 ...
##  $ X6 longitude                          : num [1:414] 122 122 122 122 122 ...
##  $ Y house price of unit area            : num [1:414] 37.9 42.2 47.3 54.8 43.1 32.1 40.3 46.7 18.8
```

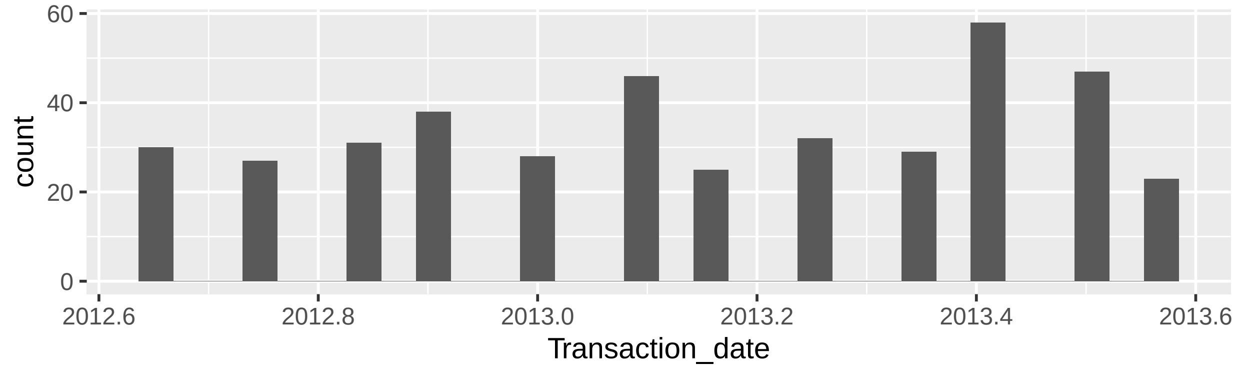
We have 414 observations and 8 columns:
- the observation number (No)
- 6 predictors (X1, X2… X6)
- the outcome (Y)

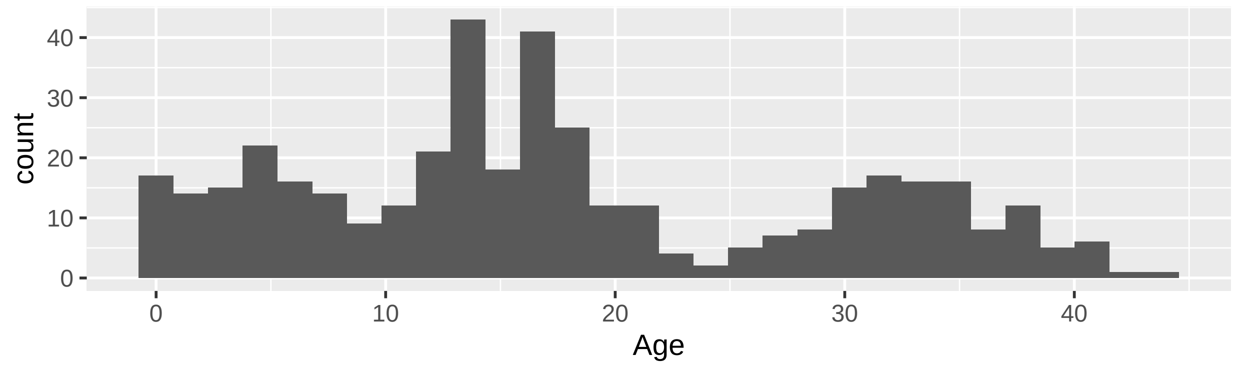
We remove the column with the observation number (No) as it is not a useful information to predict the price.
We also change the columns names to be more succinct.

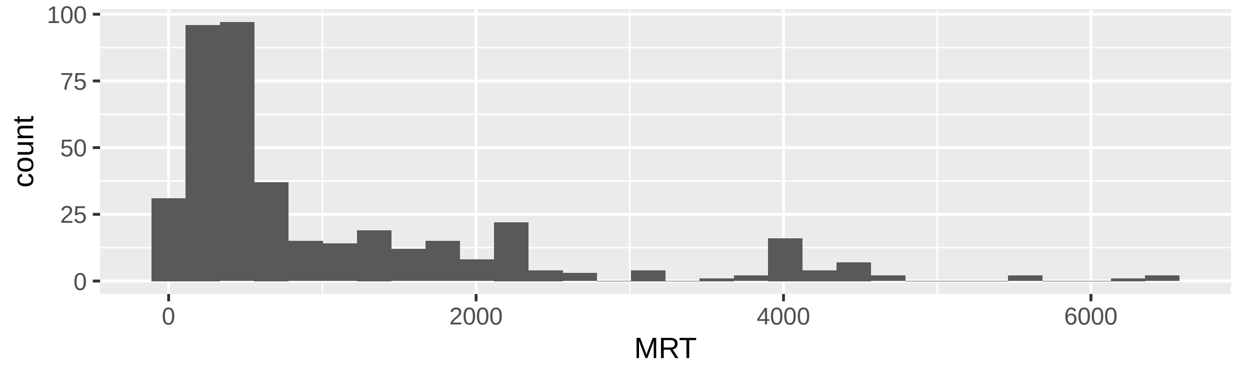
Here is a summary of the various variables:

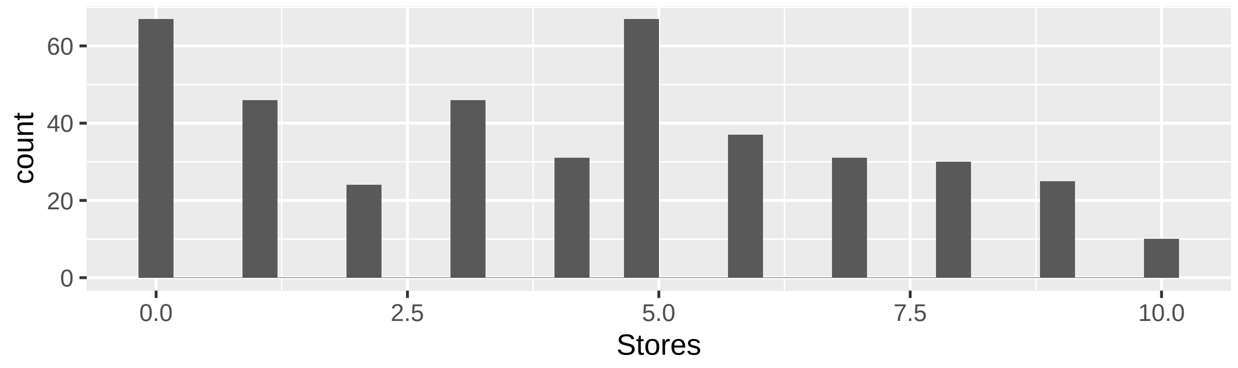
```
##  Transaction_date      Age              MRT              Stores
##  Min.   :2013     Min.   : 0.00   Min.   :  23.4   Min.   : 0.00
##  1st Qu.:2013     1st Qu.: 9.03   1st Qu.: 289.3   1st Qu.: 1.00
##  Median :2013     Median :16.10   Median : 492.2   Median : 4.00
##  Mean   :2013     Mean   :17.71   Mean   :1083.9   Mean   : 4.09
##  3rd Qu.:2013     3rd Qu.:28.15   3rd Qu.:1454.3   3rd Qu.: 6.00
##  Max.   :2014     Max.   :43.80   Max.   :6488.0   Max.   :10.00
##     latitude      longitude       Price
##  Min.   :24.9   Min.   :121   Min.   :  7.6
##  1st Qu.:25.0   1st Qu.:122   1st Qu.: 27.7
##  Median :25.0   Median :122   Median : 38.5
##  Mean   :25.0   Mean   :122   Mean   : 38.0
##  3rd Qu.:25.0   3rd Qu.:122   3rd Qu.: 46.6
##  Max.   :25.0   Max.   :122   Max.   :117.5
```

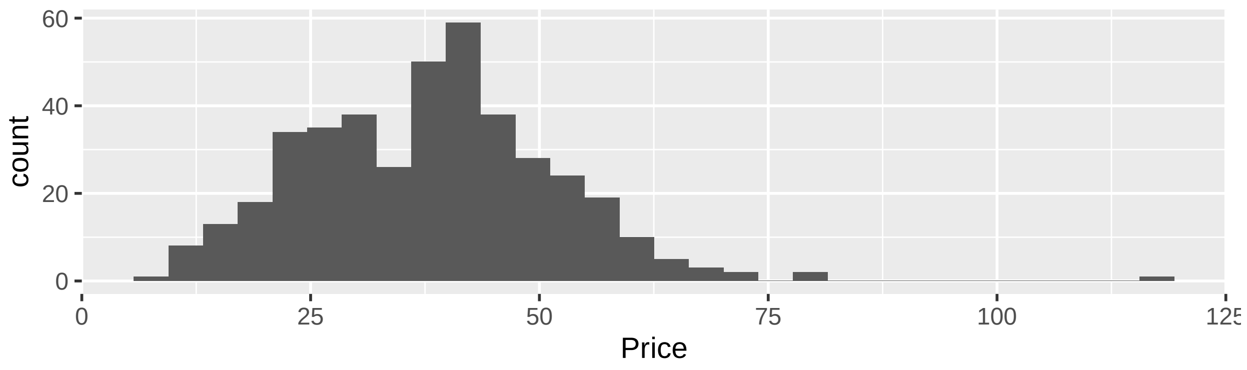
Here are the distributions of the variables (geographic coordinates excluded):











We notice that the transactions were made more or less at the same period of time (2012 and 2013). Apart from if there has been a real estate crisis at that period, we don't expect the prices to depend much on this parameter.

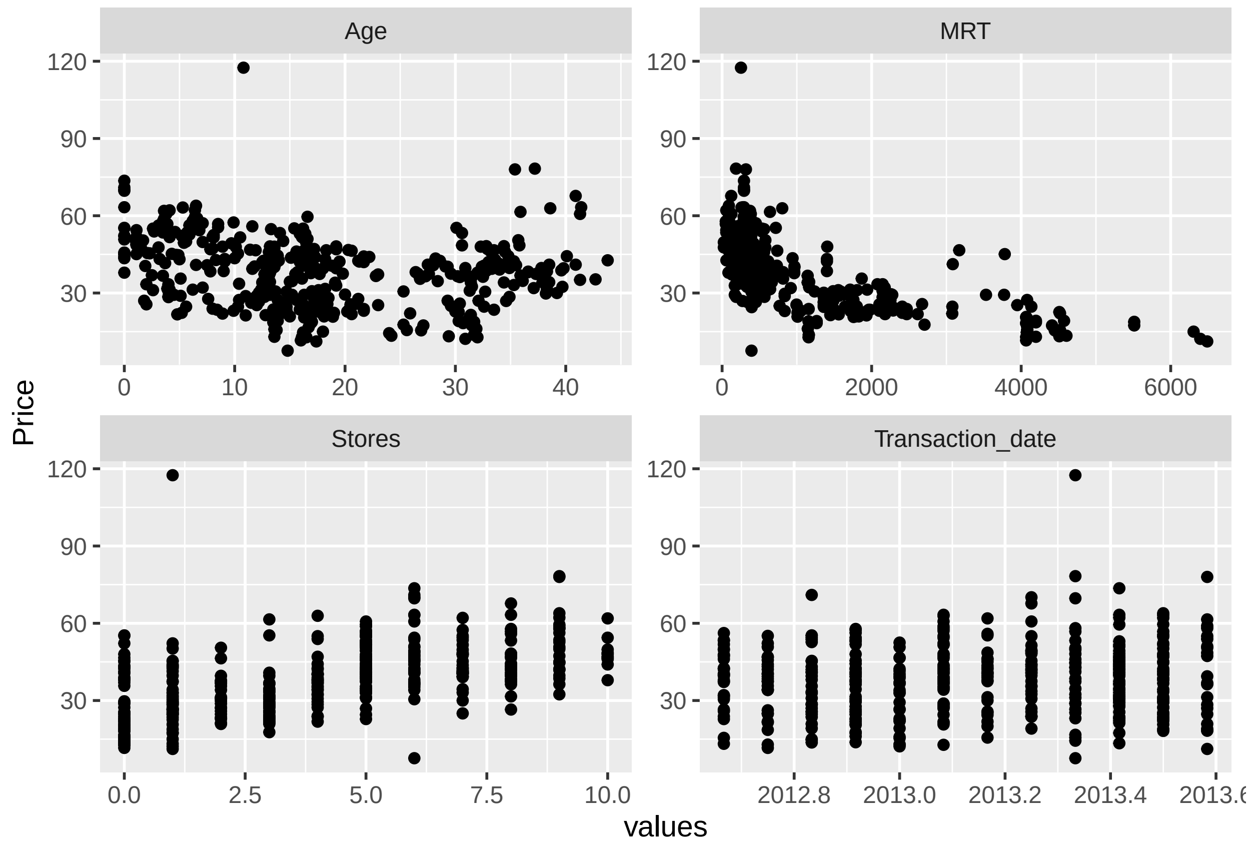
MRT and Price variables contain some isolated values (MRT>5000 and Price>90).

Let's look at the correlation coefficients of Price VS the other parameters (*Pearson* coefficient):

```
##                        [,1]
## Transaction_date  0.087529
## Age              -0.210567
## MRT              -0.673613
## Stores            0.571005
## latitude          0.546307
## longitude         0.523287
## Price             1.000000
```

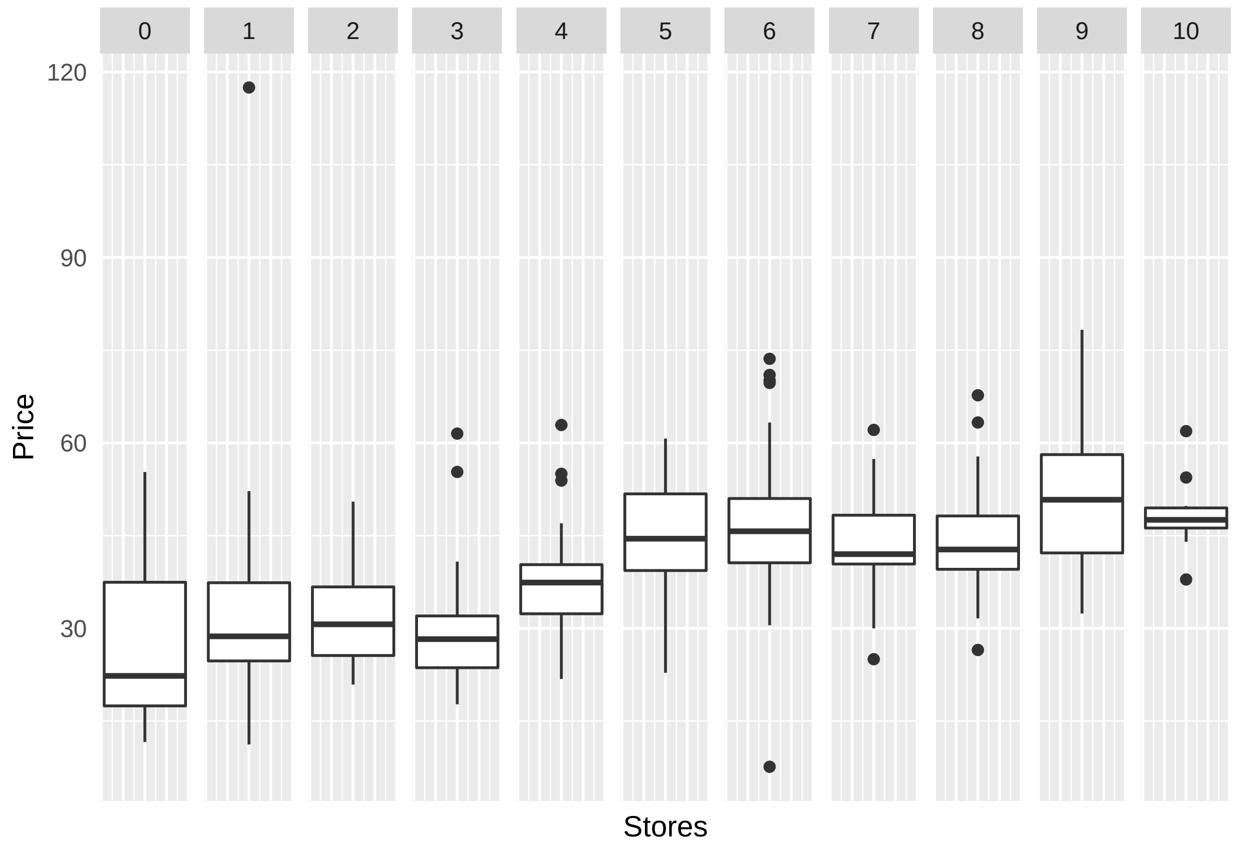
We see that the coefficient of Transaction_date is rather close to zero, which might confirm the fact that the transaction date does not have a strong influence on the price.
MRT, Stores, and the geographical coordinates have coefficients above 0.5 or below -0.5, they may be considered as more significant predictors than the others.
MRT has the coefficient closest to 1 or -1.

We plot the price VS the other variables (geographic coordinates excluded):



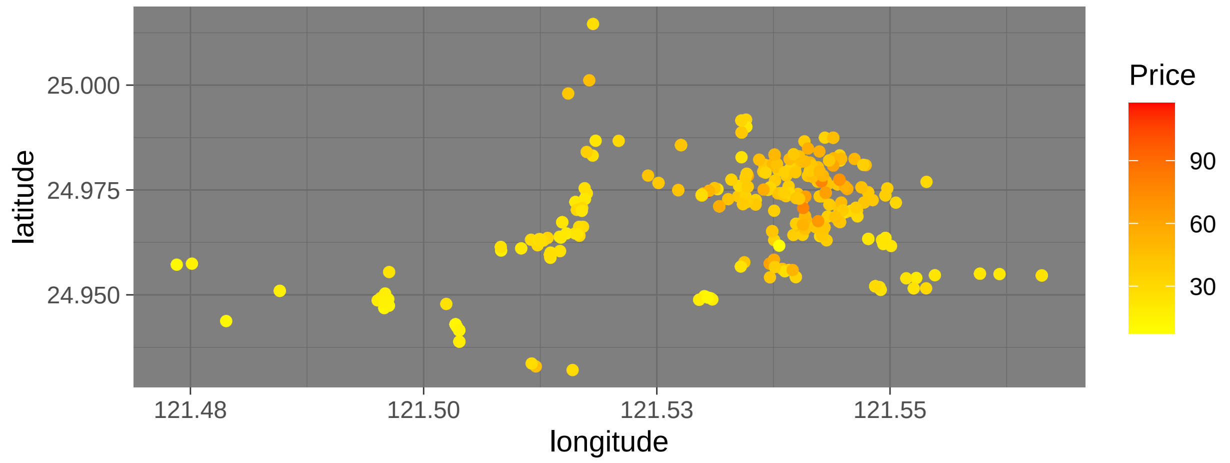
We mostly see a trend on Price VS MRT, which does not seem linear.

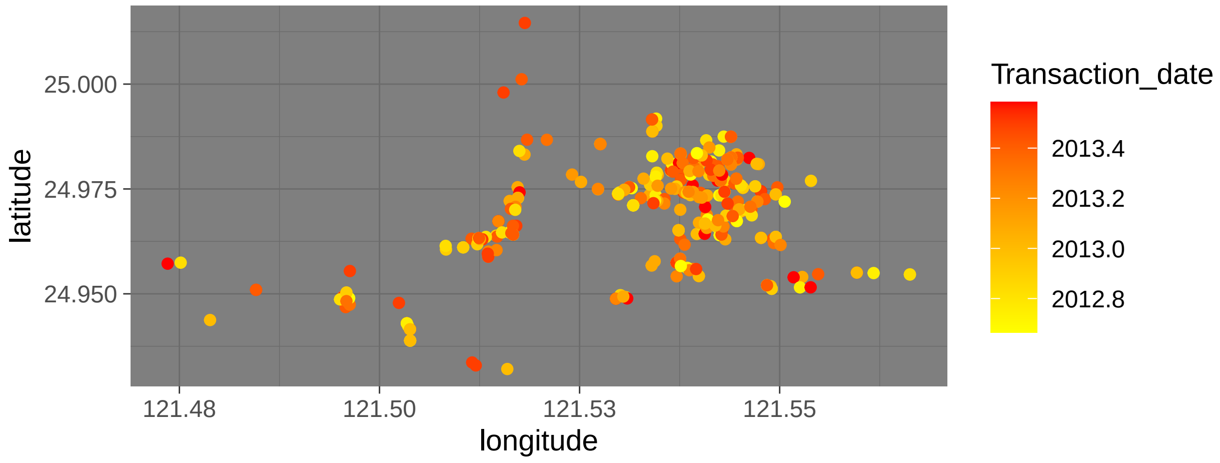
We focus on the Price VS Stores relation by ploting the boxplots of Prices for each amount of Stores:

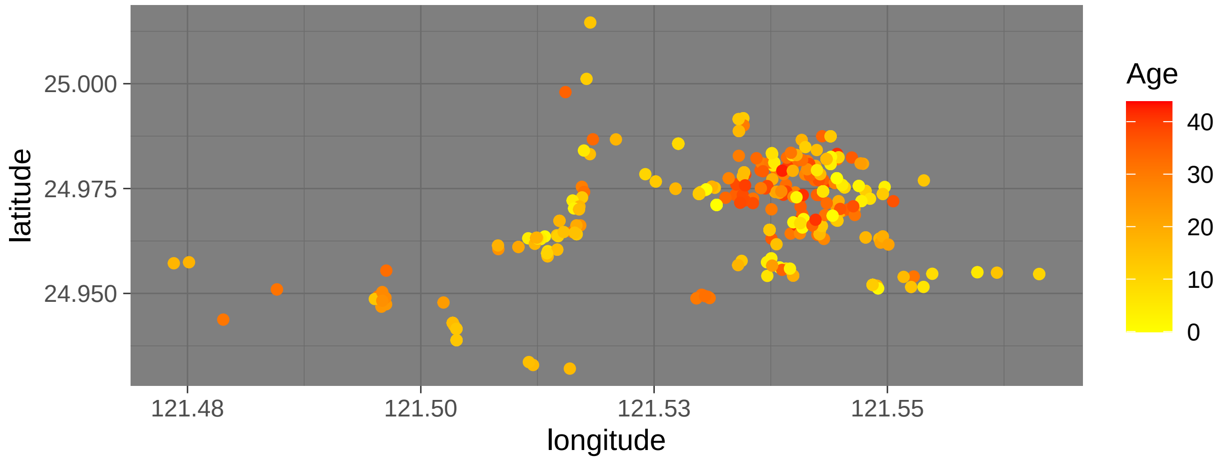


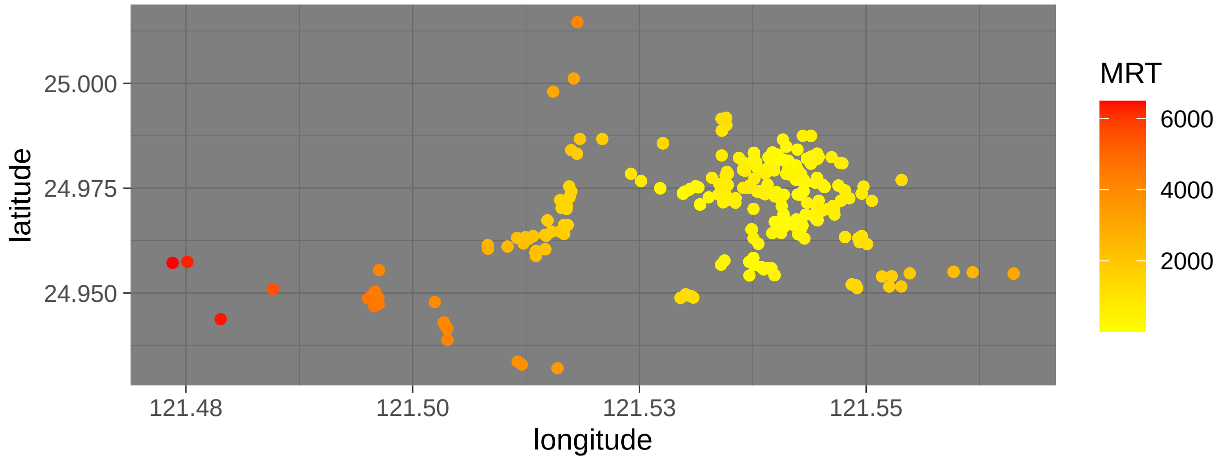
When looking at the means, this could look like a linear model.

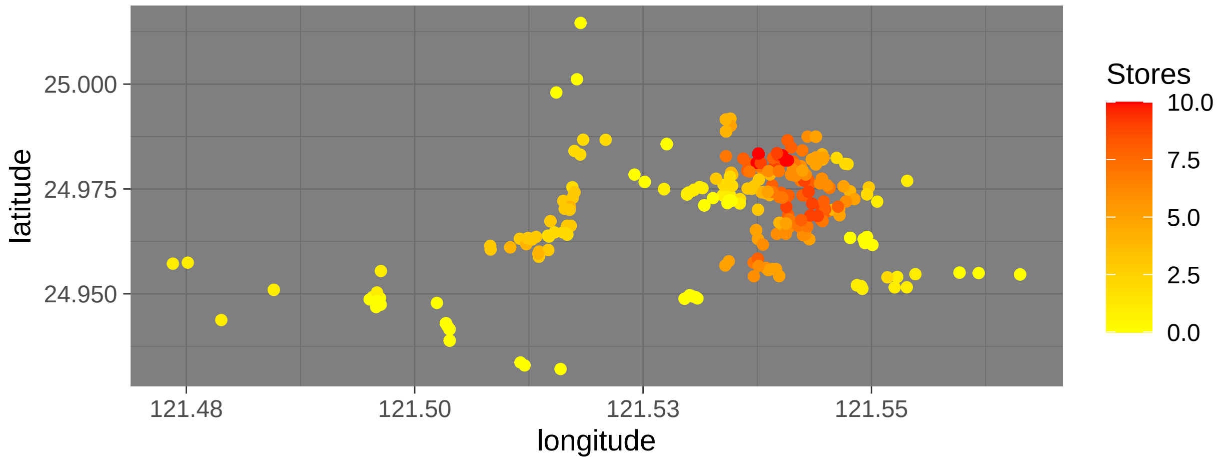
We plot the geographic repartition of the various variables:




Transaction_date
2013.4
2013.2
2013.0
2012.8
25.000
24.975
24.950
latitude
121.48
121.50
121.53
121.55
longitude


Age
40
30
20
10
0
25.000
24.975
24.950
latitude
121.48
121.50
121.53
121.55
longitude


MRT
6000
4000
2000
25.000
24.975
24.950
latitude
121.48
121.50
121.53
121.55
longitude



The strongest disparities with geographic gathering are on MRT and Stores variables. # Data split

We randomly split the data set into 2 sets:
- a **train_set** (80% of the observations): this is intended to train our algorithms
- a **test_set** (20% of the observations): this is intended to test our algorithms

This split choice is tricky as we need enough observations on the **train_set** in order to build a robust enough model, but we also need enough observations on the **test_set** to correctly assess the model. We choose to favor the **train_set** (i.e. the quality of our training) by allowing it more observations than the **test_set**, as the whole data set does not contain many observations (414).

The **train_set** and the **test_set** do not contain the observation numbers of the original data set (No).

4 Methods/Analysis

4.1 Multivariate linear regression (lm)

As we saw in the previous part:
- the variations of the Price VS the number of Stores could look like a linear model.
- the variations of the Price VS the other parameters (individually considered), did not seem to look like a linear model.

It still can be considered interesting to try a linear regression and see which result we get.
For this we use the lm() function.

We use the **train_set** to make a regression of Price VS all the other variables.
Then we make a prediction on the **test_set**. Here is the resulting RMSE:

Method	RMSE
Multivariate linear regression (lm)	8.1

As we saw earlier, the transaction date did not seem to be very relevant (correlation coefficient closest to zero).
We make another linear regression, but without considering the transaction date this time.
Here is the resulting RMSE:

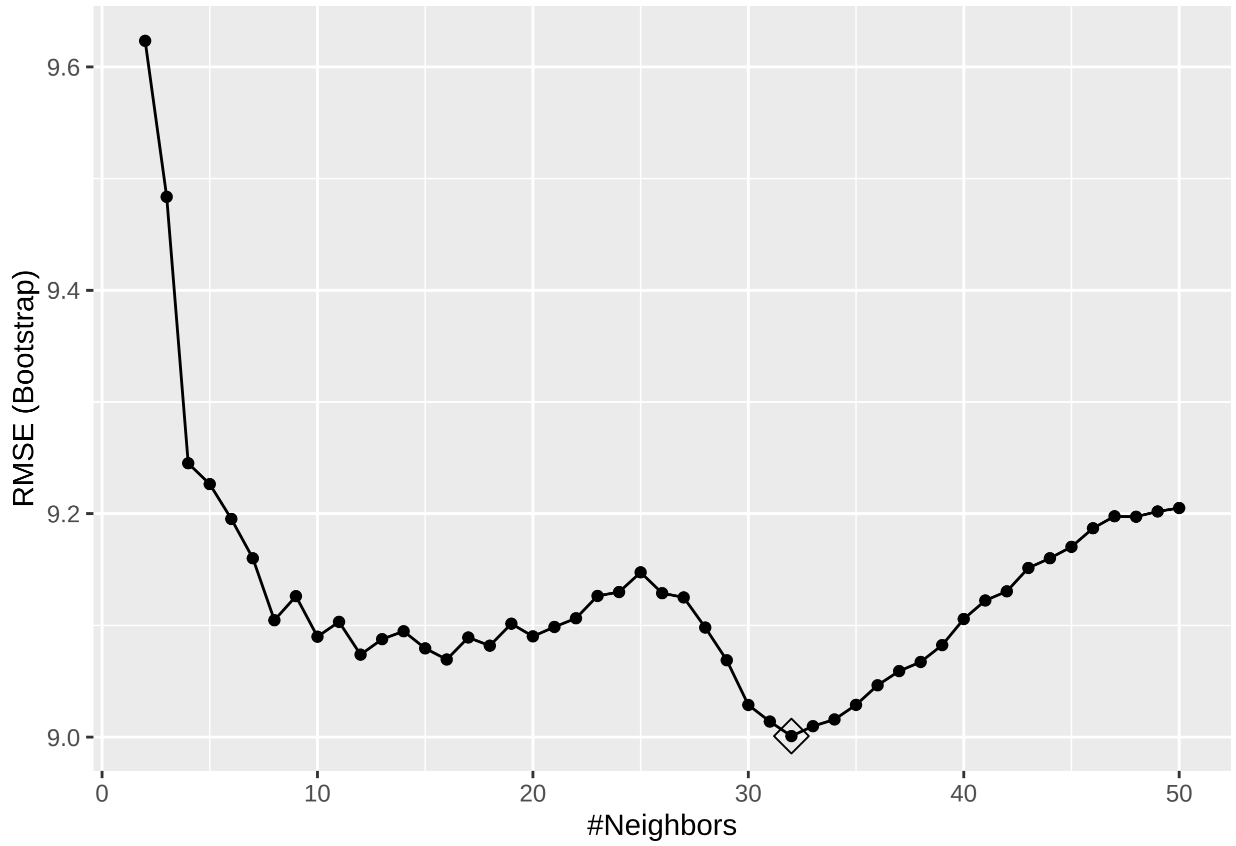
Method	RMSE
Multivariate linear regression (lm) without transaction date	7.9

We get a better result when not considering the transaction date.

4.2 k-nearest neighbours (knn)

The knn method seems appropriate to our assignment as one can easily imagine that properties in the same neighborhood and with the same characteristics are supposed to get the same value.

For this we train our data with the knn model from the caret package.
The tuning parameter is the number of neighbours (k) that we consider. To fix it, we train the algorithm with k from 2 to 50 and see the resulting RMSEs.
Here is the plot of the RMSE VS the number of neighbours (k):



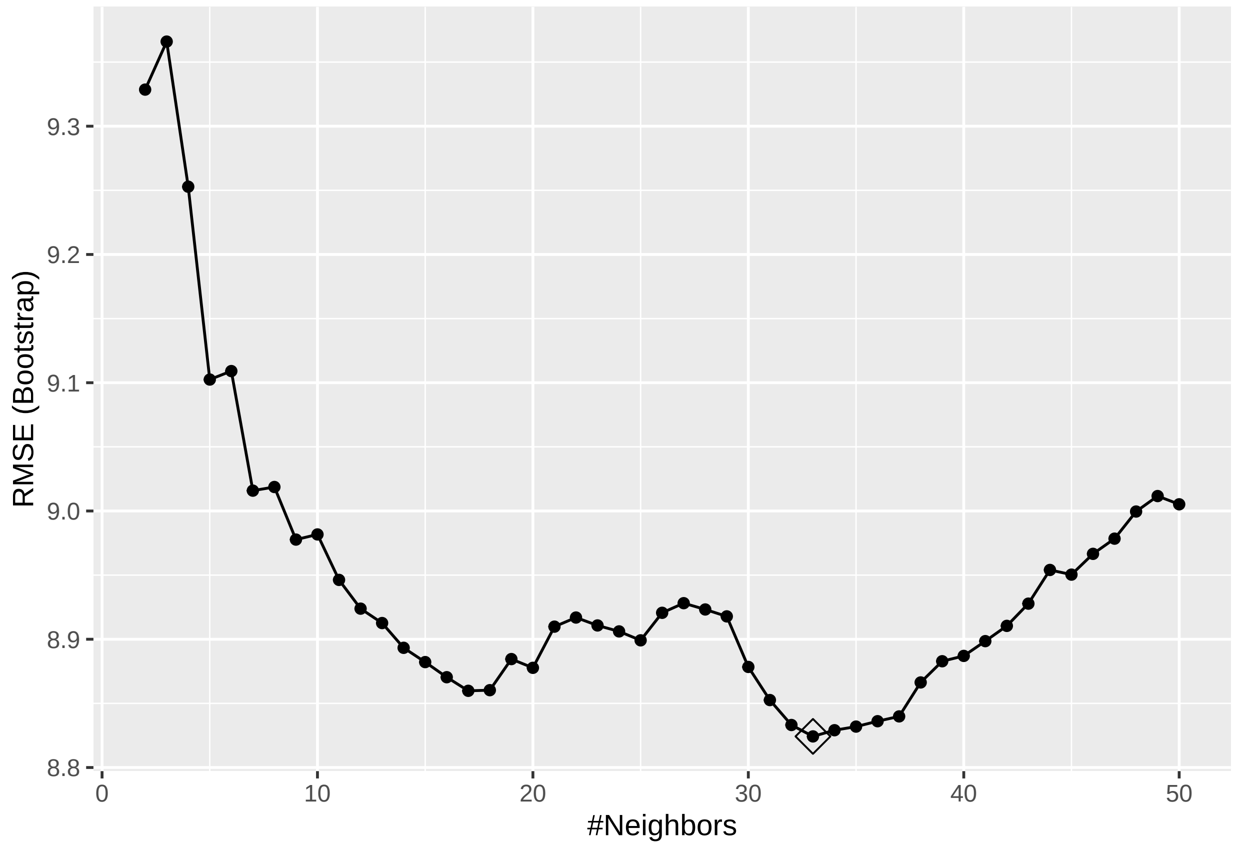
The best tune is the following k:

```
##     k
## 31 32
```

And we reach the following RMSE on the **test_set**:

Method	RMSE
k-nearest neighbours (knn)	8.4

As in the linear regression model, we now try the knn model but without considering the transaction date. Here is the plot of the RMSE VS the number of neighbours (k):



The best tune is the following k:

```
##     k
## 32 33
```

And we reach the following RMSE on the **test_set**:

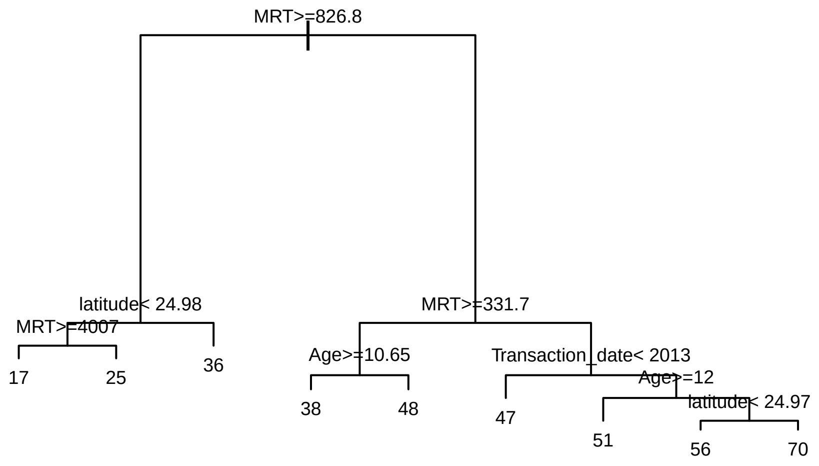
Method	RMSE
k-nearest neighbours (knn) without transaction date	8.4

We don't get a better result (at least no more than 0.1 better).

4.3 Regression Trees

As our outcome (Price) is continuous, we can make regressions with what we call *regression trees* with the rpart package.

Let's plot a tree with the default parameters:



This is interesting to see that the MRT predictor is the first one. Remember that among all the predictors, we previously observed that MRT had the *Pearson* coefficient closest to 1 or -1. This may confirm that this parameter has the strongest influence on the Price.

With the default parameters we get the following RMSE on the **test_set**:

```
## [1] 8.2676
```

To go further we can set 3 various parameters on the model, literally defined as follows in the rpart library:
- cp (complexity parameter): any split that does not decrease the overall lack of fit by a factor of cp is not attempted
- minsplit: the minimum number of observations that must exist in a node in order for a split to be attempted
- minbucket: the minimum number of observations in any terminal node

We compute regression trees for the following ranges combinations:
- cp: from 0 to 2 by 0.01
- minsplit: from 1 to 60 by 1
- minbucket: from 1 to 60 by 1

The lowest RMSE on the **test_set** is obtained for the following parameters:

```
##        Minsplit Minbucket cp  RMSE
## 473758       40        18  1 7.959
```

Here is the lowest RMSE on the **test_set**:

Method	RMSE
Regression Trees	8

As in the linear regression model, we now try the *regression trees* model but without considering the transaction date.

The lowest RMSE on the **test_set** is obtained for the following parameters:

```
##      Minsplit Minbucket cp   RMSE
## 2011        1        11  1 6.7024
```

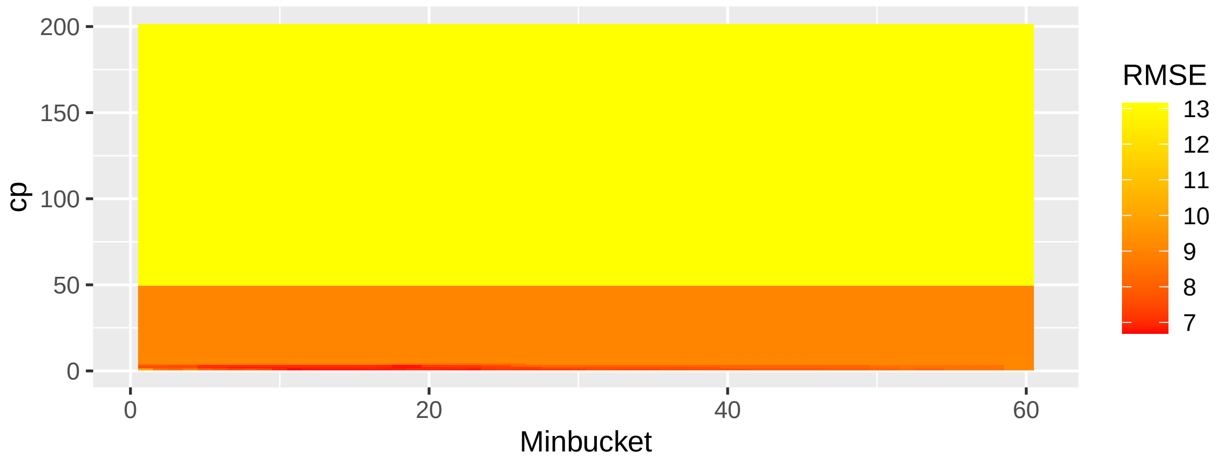
Here is the lowest RMSE on the **test_set**:

Method	RMSE
Regression Trees without transaction date	6.7

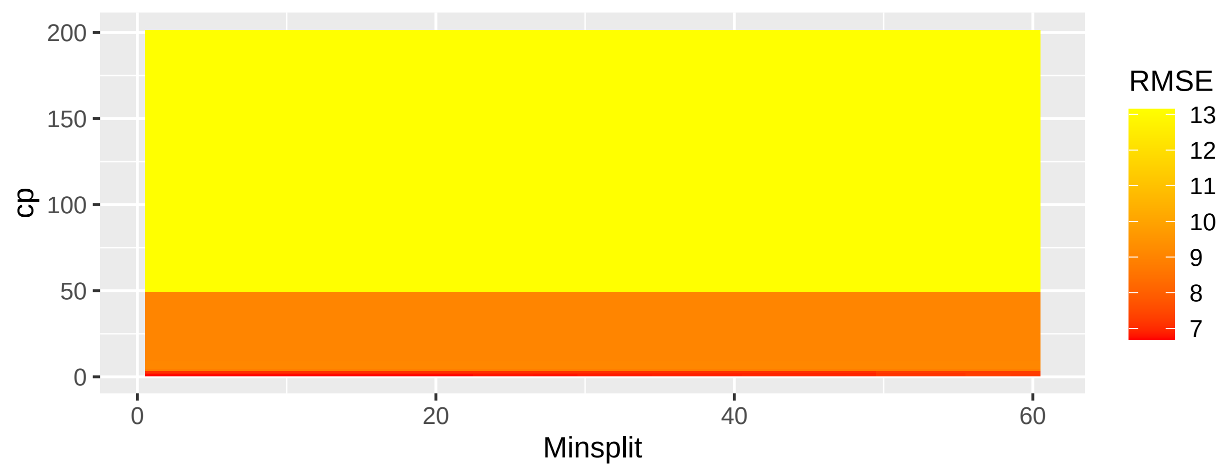
We get a much better result when not considering the transaction date.

Let's explore the plots of the RMSE on the **test_set** when not considering the transaction date and when:
- setting one parameter as the one resulting in the best RMSE
- changing the two other parameters

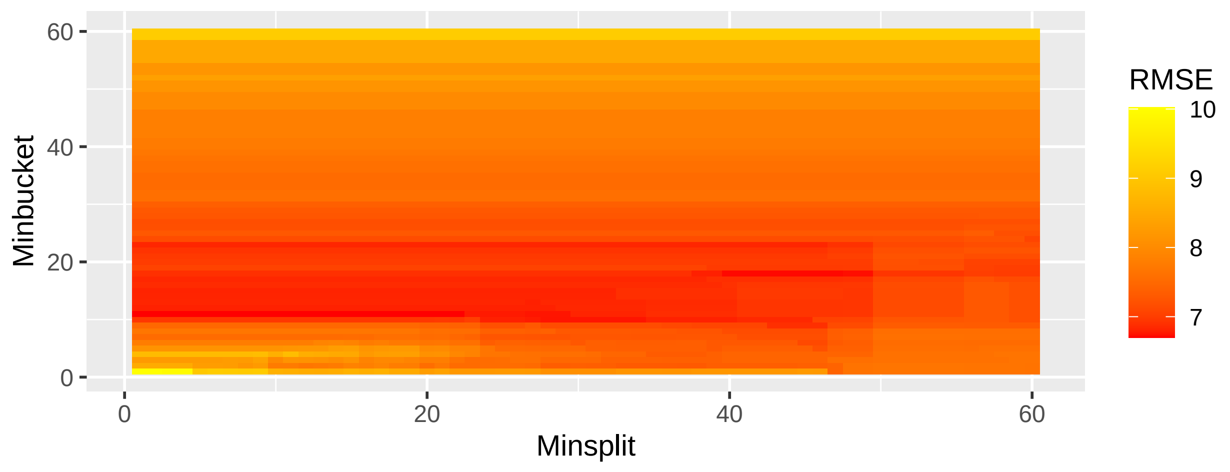
Plot for the best Minsplit:



Plot for the best Minbucket:



Plot for the best cp:



When looking at the three previous plots, we observe that cp seems to have the strongest influence on the RMSE.

4.4 Random Forest

The idea of the random forest technique is to randomly generate predictions using regression trees and to average these predictions. This is supposed to get a better stability in the results.

We train our data with the Rborist model from the caret package.
The advantage of this model compared to the rf one is that we can set as tuning parameter the minimum node size (*minNode*).
We evaluate the RMSE with *minNode* from 1 to 300 by 1 (this is very large and it does not seem useful to try higher values as our **train_set** only has 329 observations).

Here is the best RMSE on the **test_set** :

Method	RMSE
Random Forest (Rborist)	6.3

which is obtained for the following *minNode* value:

```
## [1] 290
```

As in the linear regression model, we now try the Rborist model but without considering the transaction date.

Here is the best RMSE on the **test_set**:

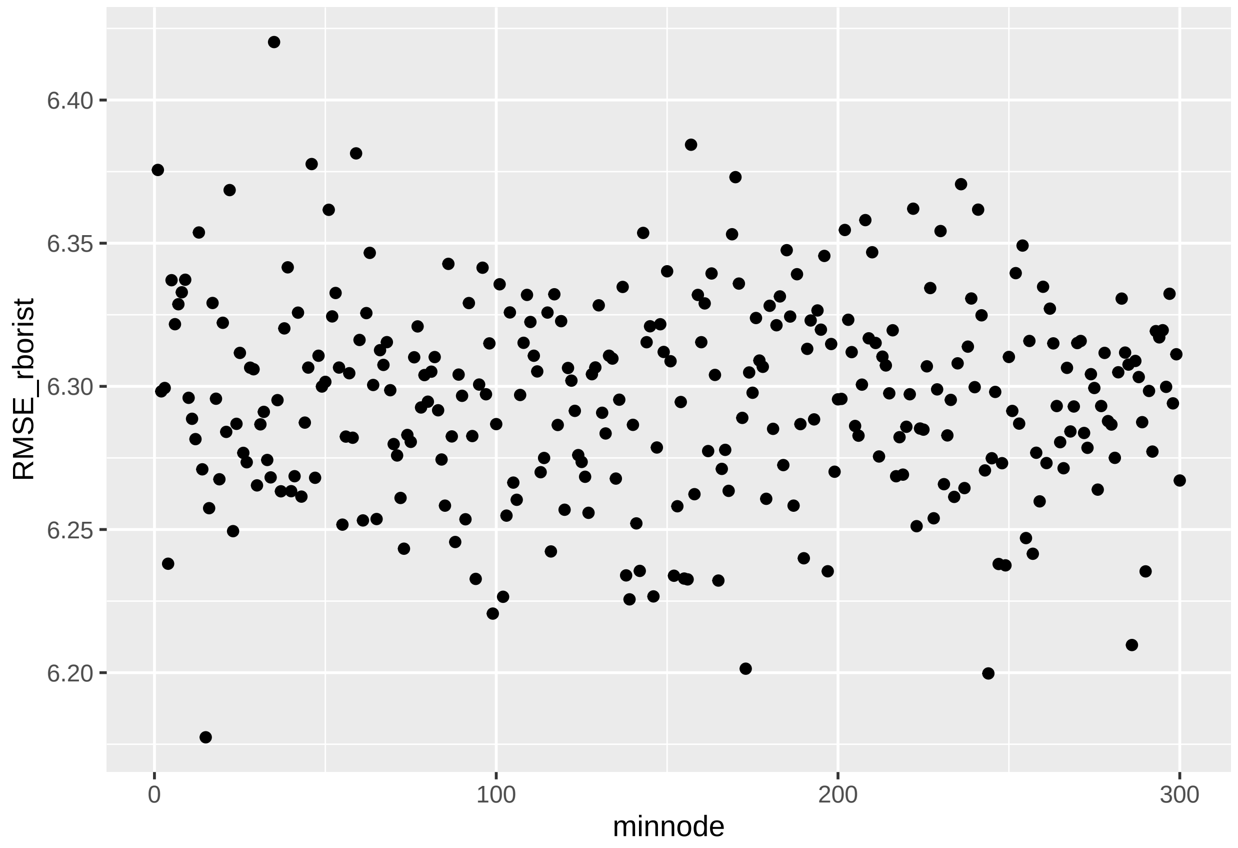
Method	RMSE
Random Forest (Rborist) without transaction date	6.2

which is obtained for the following *minNode* value:

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

We get a better result when not considering the transaction date.

Here is a plot of RMSE on the **test_set** regarding the *minNode* values when not considering the transaction date:



We should note that the dispersion seems important but all the results stay in a reasonable range (RMSE stays more or less between 6.2 and 6.4).

5 Results

Here is a summary of the models we used and the resulting RMSE on the **test_set**:

Method	RMSE on the test_set
Multivariate linear regression (lm)	8.1
Multivariate linear regression (lm) without transaction date	7.9
k-nearest neighbours (knn)	8.4
k-nearest neighbours (knn) without transaction date	8.4
Regression Trees	8
Regression Trees without transaction date	6.7
Random Forest (Rborist)	6.3
Random Forest (Rborist) without transaction date	6.2

- Each time we do not consider the transaction date, we get a roughly equal (knn) or better (all but knn) result. This is an important insight as one may naturally think that considering more predictors would have led to a more accurate model, but this is not the case here
- The knn method does not lead to a better result than the lm one, we interpret it in the fact that lm considers all the observations whereas knn only considers the k nearest observations. The nearest can be in reality far away. If we had much more observations, with a smooth repartition, we might expect a better result with the knn method than with the lm one

- The Rborist method leads to a better result than the Regression Trees, which seems logical as it has the same principle but is more advanced
- The Rborist method without considering transaction date leads to the best result, we interpret it in the fact that it is the only method that we used that:
  - is not linear
  - considers all the observations for each prediction
  - smooths random variability
  - does not consider the variable "transaction date" which did not seem to have a strong influence on the Price (*Pearson* coefficient close to zero)

6 Conclusion

The Rborist random forest model without considering the transaction date reached the best RMSE on the **test_set**.

To go further we could:
- compute Monte Carlo simulations on these models to get more stable results
- remove a given quantile from the original data set (e.g. the observations with MRT>5000, Price>90, or the 10% geographically furthest from the rest…). This might lead to a lower accuracy when predicting in the area of the removed observations, but to a better accuracy on the area of most of the observations
- explore other random forest algorithms with different tuning parameters, such as *ranger*, *extraTrees*, or *rfRules*
- map the properties on a satellite-view map, as a human eye can understand things differently when looked on a map

7 Citations (original data provider)

Yeh, I. C., & Hsu, T. K. (2018). Building real estate valuation models with comparative approach through case-based reasoning. Applied Soft Computing, 65, 260-271.