UNIVERSITÀ
DEGLI STUDI
DI MILANO

LA STATALE

Department od Economics, Management and Quantitative Methods

Università degli Studi di Milano

Data science and economics

Statistical Learning Project

Analysis and prediction models for New York City Airbnb

Report

Author:

Andrea Ierardi

Academic year 2019-2020

Contents

List of Figures

1

Abstract

The aim of the project is to analyse, develop prediction models and define data clusters from the "New York City Airbnb Open Data" from a Kaggle competition.
In particular, one part is focused on the development of predictive models to forecast house prices using these Supervised Learning technics:

- Linear Regression
- Decision Tree
- Random Forest
- Ranger Random Forest
- Neural Newtworks

For each of these, a comparison between the Mean Square Error and between all R^2 measure of all methods has been made to highlight which have the best performance. Also, for training all the models, a partition of the dataset in three parts has been applied: entire dataset, filtered by neighboorhood group and filtered for neighboorhood group and room type. In this way, it is possible to check the perfomance giving less or more features in input.

The second part is focused on the cluster and data reduction technics using these Unsupervised Learning technics:

- K-means Algorithm
- Clustering for mixed-type data
- Principal Component Analysis

2

Problem Definition and Algorithm

2.1 Two main Goals

2.1.1 Develop predictive models for price

The first objective is the forecast of the prices given some input information. This could be usefull for a lot of scenarios. For example from a AirBnB customer point of view, he/she would like to get the list of houses more in alignment with his/her preference choice; or for a host point of view, where given the position and other information he/she could get information about the possible per day price of his property in New York City.

2.1.2 Define clusters and groups

The second objective is the definition of group or partition between the houses with different characteristics. For a user point of view could be usefull to have information about avaiable houses similar to those booked in the past. This could be also usefull after the booking for a suggestion analysis having the information about last booked houses in NewYork or houses in similar cities around the world.

2.2 Algorithms

2.2.1 Linear Regression

Linear regression is a linear approach to modeling the relationship between a dependent variable and one or more independent variables.

2.2.2 Decison Trees

In decision analysis, a decision tree can be used to visually and explicitly represent decisions and decision making.

2.2.3 Random Forest

Random forests are an ensemble learning method for classification, regression and other tasks that operate by constructing a multitude of decision trees.

2.2.4 Ranger Random Forest

Ranger is a fast implementation of random forests or recursive partitioning, particularly suited for high dimensional data.

2.2.5 Neural Networks

Neural networks are a set of algorithms, modeled loosely after the human brain, that are designed to recognize patterns.

2.2.6 K-means

K-means is a method of vector quantization, originally from signal processing, that aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean.

2.2.7 Principal Component Analysis

Principal Component Analysis (PCA) is mostly used as a tool in exploratory data analysis and for making predictive models.

3

Experimental Evaluation

3.1 Methodology

3.1.1 Data Inspection

The dataset is part of a Kaggle competition, called the New York City Airbnb Open Data. It contains 48.000 rows per 16 columns. The dataset is structured with these columns:

- **id**
- **name**: name of the listing
- **host_id**
- **host_name**
- **neighbourhood_group**: location
- **neighbourhood**: area
- **latitude**: coordinates
- **longitude**: coordinates
- **room_type**: space type
- **price**: in dollars
- **minimum_nights**: amount of nights minimum
- **number_of_reviews**: number of reviews
- **last_review**: latest review
- **reviews_per_month**: number of reviews per month
- **calculated_host_listings_count**: amount of listing per host
- **availability_365**: number of days when listing is available for booking

It has been select only 5 of these variables: price, latitude, longitude, neighbourhood_group and room_type. The reason is that it is reasonable to select them to predict the prices and to obtain different clusters. The position and the neighbourhood is important since if a property is positioned near the city center will have a higher price with respect to those situated in the outskirts; also the type of room, since an entire apartment will cost more than a single room.

From the Figure 1 is possible to see the distribution of the price. The minimum price is 0 and the maximum is 10000$. It is not possible to rent a house for free, so it is possible to filter the price with a price higher than 15$. It is possible that a luxury house cost a lot per day, but these value can not be consider in the model, instead they are outliers and for this reason it is convenient to filter the price again and take those that have a value lowert than 500$. The reason is that the third quantile has a price of 175$ which is a far from 10000$ (Figure 3). It has also been checked the null and missing value in the dataset and has not been found apart from the reviews_per_month column. It is not a problem, since this feature has been not taken in account to train the models.

```
     price
Min.   :    0.0
1st Qu.:   69.0
Median :  106.0
Mean   :  152.7
3rd Qu.:  175.0
Max.   :10000.0
```

Figure 1: Price summary

From Figure 2, it is possible to see the distribution of all houses in New York City and the price. This picture is not really informative since it can be noticed that the prices for the most part are in the range 0-500$ and only a low number of istances have a price greater than 500$. Deleting the outliers, the Figure 3 is more informative than the one before.

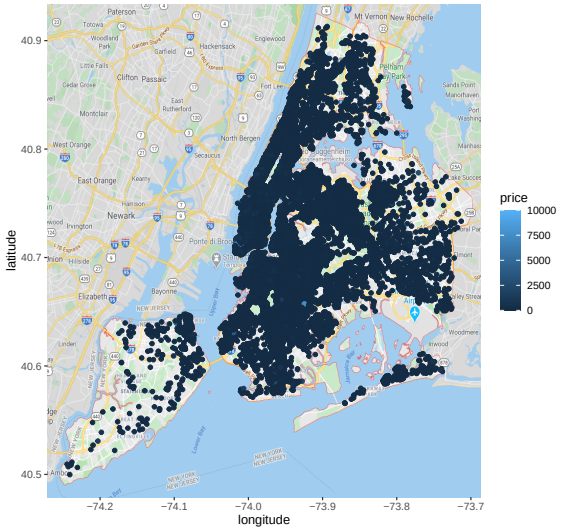

Figure 2: Distribution of all houses in NY colored by prices

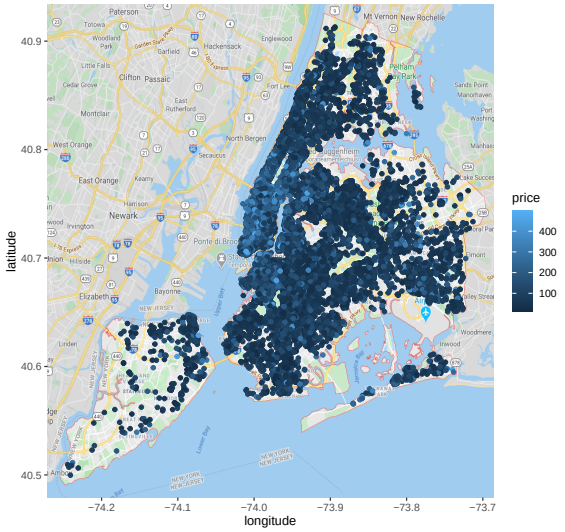

Figure 3: Distribution of all houses in NY of price between 15$ and 500$ per day

3.1.2 Data Cleaning and Pre-processing

The dataset has approximately 48.000 rows for each column, so an important part of the work is related to the pre-processing due to the large amount of data. For this reason, some variable has been rescaled to let the models to learn faster and better and to perform a better prediction. Latitude and longitude have not been rescaled for forecast price model, while for clustering methods to have a consistent distance calculation. Categorical variables have also been rescaled assigning a numerical value to each category, resulting as unordered factors.

The selected categorical variables are: neighbourhood_group and room type.
The selected numerical variables are: latitude, longitude and price.

Considering that every neighbourhood group has his characteristic, also taking in account the room type that could impact largerly the price. It is possible to define of different subset of the original set and try different methods of forecast. This is due to the fact that a customer should choose which of the different neighbourhood and room type is interested in. Models have been run to different scenarios (Figure 4): users interested in all New York City houses and all type of room, users interested only in a single neighbourhood and users interested in a single neighbourhood and a single room type.
Including different scenario could be potentially computationally expensive. In reality, the training proceeded without any problem. [1] Computantional problem may emerge for the case of hyperparametrisation tuning, even using multiprocessing and multithreading technics. For semplicity, in this project no model tuning have been made.
Also for clustering have been filter every scenario, but not for the case of Hierarchical Clustering because it need to generate a readable dendogram using the mean of all neighbourhood or room type price, Principal Component Analysis since avaiable data are really small for some specific neighbourhood like Staten Island and specific room type.

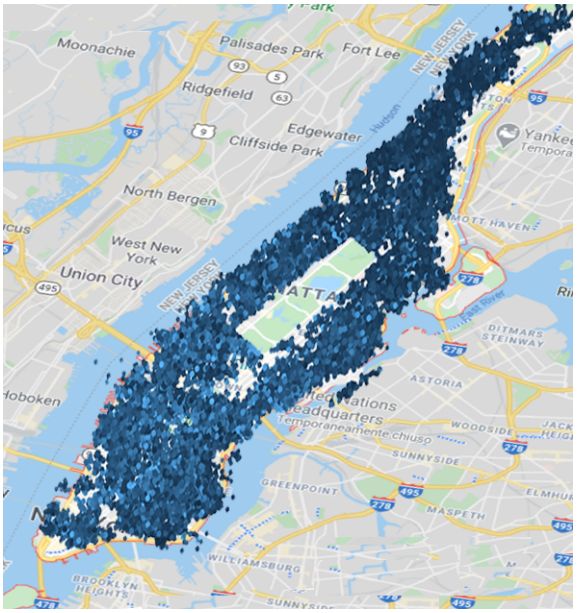

Figure 4: Approximated distribution map of all houses in Manhattan

[1] The models have been trained on a machine with quad-core 3.5 GHz processor, 8 Gb RAM and 4 Gb dedicated GPU.

3.2 Results

Since the number of models trained are very high, the study result will be presented for different macro groups that depend on the different subset of the dataset: entire dataset, filtering by neighborhood and filtering by neighborhood and room type. This is due to the fact that neighborhood in general have similar result (like Manhattan has similar shape of Brooklyn and Staten Island has a similar shape of Bronx).

===== DA RIVEDERE ===

Subset	Linear.Regression	Decision.Tree	Random.Forest	Ranger.Random.Forest	Neural.Networks
Brooklyn	0.4351730	0.4395665	0.4175302	0.4141696	0.5295534
Manhattan	0.7936143	0.7964149	0.7483505	0.7474648	0.8801335
Queens	0.3721530	0.3699898	0.3514677	0.3525291	0.3807765
Staten Island	0.3467308	0.3471879	0.3415798	0.3492562	0.6710544
Bronx	0.2741482	0.2744778	0.2774524	0.2699379	0.3388045
Brooklyn/Private room	0.1858138	0.1806116	0.1936970	0.1925123	0.2464772
Brooklyn/Entire home/apt	0.7262980	0.7179003	0.7702742	0.7709408	0.7863187
Brooklyn/Shared room	0.1157458	0.1320524	0.0881610	0.0855227	0.1180578
Manhattan/Private room	0.4445611	0.3753113	0.3778408	0.3774914	0.5059889
Manhattan/Entire home/apt	0.9361941	0.9585893	0.9728082	0.9773220	1.0688116
Manhattan/Shared room	0.5616574	0.5452982	0.5875275	0.5895540	1.1310954
Queens/Private room	0.1616290	0.1551588	0.1516938	0.1525349	0.2684766
Queens/Entire home/apt	0.6846303	0.6591288	0.6550463	0.6523420	0.6980945
Queens/Shared room	0.0709897	0.2989787	0.2102031	0.2043754	0.0560464
Staten Island/Private room	0.1241196	0.1802672	0.1355774	0.1375752	0.1395840
Staten Island/Entire home/apt	0.5703345	0.7737648	0.7185084	0.7119659	1.2433252
Staten Island/Shared room	0.0518804	0.0960227	0.0921335	0.0865780	0.2093138
Bronx/Private room	0.2146868	0.2159434	0.2161088	0.2180725	0.5847212
Bronx/Entire home/apt	0.5627967	0.7622148	0.6682157	0.6587408	0.7178106
Bronx/Shared room	0.2071295	0.1939584	0.1759492	0.1750096	0.1756585
All	0.6036113	0.6005651	0.5672255	0.5436218	0.6334017

Figure 5: Summary of the MSE for all the models for each subset of the dataset

3.2.1 Linear Regression Results

Linear Regression on the entire dataset

For the Linear regression the model give different results based on the variables used.

Linear Regression selecting the Neighboorhood group

For semplicity, the tests have been only taken filtering for Manhattan data points, but changing the neighboorhood the results are similar. Results are acceptable (Figure 5), given a R^2 value of 0.4011 and a Mean Square Error, comparing prediction and test set, of 0.67.

Linear Regression selecting the Neighboorhood group and the type of room

As in the previous case the tests are for Manhattan and for Entire home/Aparment type of room.

The model output (Figure 6) a value of R^2 equals to 0.05953 which is low and a Mean Square Error, comparing prediction and test, of 1.23. The model (Figure 7) obtain a R^2 value of 0.4031 which is acceptable and a Mean Square Error, comparing prediction and test set, of 0.59.

```
 === Linear Regression selecting the Neighboorhood group ===
Neighboorhood group = Manhattan

Call:
lm(formula = price ~ latitude + longitude + room_type, data = train_filtered[-1])

Residuals:
    Min      1Q  Median      3Q     Max
-1.5205 -0.3504 -0.1194  0.1855  4.6287

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -598.32453   22.46859 -26.629  < 2e-16 ***
latitude       5.45751    0.22945  23.785  < 2e-16 ***
longitude     -5.07910    0.24902 -20.396  < 2e-16 ***
room_type2     0.97734    0.01238  78.951  < 2e-16 ***
room_type3    -0.15381    0.04391  -3.503 0.000462 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.6595 on 11882 degrees of freedom
Multiple R-squared:  0.4013,    Adjusted R-squared:  0.4011
F-statistic:  1991 on 4 and 11882 DF,  p-value: < 2.2e-16

MSE:  0.6730552
```

Figure 6: Linear Regression output filtering by Manhattan

```
=== Linear Regression selecting the Neighboorhood group and room_type ===
Neighboorhood group = Manhattan and room_type = Entire home/apt

Call:
lm(formula = price ~ latitude + longitude, data = train_filtered[-1])

Residuals:
    Min      1Q  Median      3Q     Max
-0.7397 -0.2368 -0.0868  0.1206  4.5392

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -387.2888    19.9701  -19.39   <2e-16 ***
latitude       3.0005     0.2045   14.68   <2e-16 ***
longitude     -3.5771     0.2186  -16.36   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.4114 on 6065 degrees of freedom
Multiple R-squared:  0.05984,   Adjusted R-squared:  0.05953
F-statistic:   193 on 2 and 6065 DF,  p-value: < 2.2e-16

MSE:  1.230804
```

Figure 7: Linear Regression output filtering by Manhattan and Entire home/Apartment

```
=== Linear Regression without filters ===

Call:
lm(formula = price ~ latitude + longitude + room_type + neighbourhood_group,
    data = train)

Residuals:
    Min      1Q  Median      3Q     Max
-2.2092 -0.4469 -0.1364  0.2398  4.5600

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)             -2.019e+02  1.405e+01 -14.370  < 2e-16 ***
latitude                -1.634e+00  1.383e-01 -11.814  < 2e-16 ***
longitude               -3.620e+00  1.578e-01 -22.935  < 2e-16 ***
room_type2               1.012e+00  9.515e-03 106.379  < 2e-16 ***
room_type3              -2.637e-01  3.034e-02  -8.692  < 2e-16 ***
neighbourhood_group2     5.067e-01  1.601e-02  31.644  < 2e-16 ***
neighbourhood_group3     2.623e-01  1.964e-02  13.354  < 2e-16 ***
neighbourhood_group4    -9.063e-01  5.844e-02 -15.508  < 2e-16 ***
neighbourhood_group5     3.093e-01  3.853e-02   8.029 1.02e-15 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.7732 on 28566 degrees of freedom
Multiple R-squared:  0.4033,    Adjusted R-squared:  0.4031
F-statistic:  2413 on 8 and 28566 DF,  p-value: < 2.2e-16

MSE:  0.5932014
```

Figure 8: Linear Regression output without filters

Linear Regression	MSE	R^2
Entire dataset	0.59	.40
Filter: Mahnattan	0.76	0.355
Filter: Manhattan & Apt	0.44	0.12

3.2.2 Decison Trees Results

For the Decision tree the model give different results based on the variables used.

Decison tree without filters

Decison tree selecting the Neighboorhood group

Decison tree selecting the Neighboorhood group and room type

Decision Tree	MSE	R^2
Entire dataset	0.59	..
Filter: Mahnattan	0.77	..
Filter: Manhattan & Apt	0.40	..

3.2.3 Random Forest Results

There were no problem running Random Forest regression for the price, givin the default parameters. For the parameter tuning there were no possibilities for the entire dataset, due to

the large number of data points. For the filtered dataset, instead, were possible to tune the mtry, number of maximum nodes and number of trees.

Random Forest without filters
There were no possibility to run a forecast of the price

Random Forest selecting the Neighboorhood group

Random Forest selecting the Neighboorhood group and room type Using the tuning of the parameter the results are slightly better. The model start with a 23% explained variance to a value of 25%.

Random Forest	MSE	$\%Var$
Entire dataset	0.56	43.9
Filter: Mahnattan	0.70	38.9
Filter: Manhattan & Apt	0.38	21.6

3.2.4 Ranger Random Forest Results

Ranger Random Forest is known to be computationally light with respect to the classic Random Forest. In fact, for the tuning part there were not problem in running it for the entire dataset.

Ranger without filters
Ranger outputs for the entire dataset are consistent. We have a R^2 of 0.47 and OOB error of .

Ranger selecting the Neighboorhood group
The dataset filtered by Neighboorhood ouputs a value of 0.4 R^2 and a OOB error of.

Ranger selecting the Neighboorhood group and room type The dataset filtered by neighboorhood and room type gives as result a R^2 of 0.25.

Ranger RF	MSE	%*Var*
Entire dataset	0.53	46.6
Filter: Mahnattan	0.71	38.8
Filter: Manhattan & Apt	0.38	21.7

3.2.5 Neural Network Results

Neural Networks MSE	Loss	
Entire dataset	0.61	0.53
Filter: Mahnattan	0.84	0.64
Filter: Manhattan & Apt	0.48	0.44

3.2.6 K-means Results

3.2.7 Principal Component Analysis Results

3.3 Discussion

3.3.1 Linear Regression Discussion

Linear regression model gives interesting results for the non-filtered dataset and also for the filtered by neighbourhood group. All variable results to rejected by Null hyphotesis, so the model depends on all the selected variables. Latitude and longitude are correlated with the target, room type is strongly positive correlated with the price and neighbourhood group does not seem to have a great contribution in the prediction of the price.

3.3.2 Decision Tree Discussion

The performance with respect to the other models are not the best, but acceptable. The predicton results are not also very precised for the filtered neighboorhood and room type. Also, the plots of the predicted value are not so consistent since the values are divided in category which correspond to the leaves that are not so strong with respect to the other model predictions.

3.3.3 Random Forest Discussion

Random forest outputs consistent results and performs better than linear regression and decision tree. Parameters tuning does not give big improvement in performance and also are computationally expensive.

3.3.4 Ranger Random Forest Discussion

Results of Ranger are the best with respect to the previous models. Also the tuning part was fast and computationally cheaper than the classic Random Forest model but does not give great improvements in performance.

3.3.5 Neural Network Discussion

3.3.6 K-Means Discussion

3.3.7 Principal Component Analysis Discussion

4

Conclusion

From the result the method with the most higher accuracy is the Random Forest method... while the worst are ....
Moreover, Random Forest method is also the worst in term of computation time for the tuning part since it takes for a configuration with 4 core, more or less 1 hour to tune the parameters.

4.1 Linear Regression

4.2 Decision Tree

Decision tree are one of the most used model in the Machine Learning world since are very familiar to human users and can be easily plotted.

4.3 Random Forest

Random Forest is an ensemble method which use a combination of decision tree to get the prediction.

4.3.1 Ranger Random Forest

Ranger Random Forest is a computationally light model which results are very close the classical Random Forest.

5

Appendix

5.1 Footnote

You can create a footnote like this.[2]

[2] I created a footnote.

References

[1] Giusti, Santochi, *Tecnologia Meccanica e Studi di Fabbricazione*. Casa Editrice Ambrosiana, Seconda Edizione

[2] Mechteacher, *Knuckle Joint – Introduction, Parts and Applications*, http://mechteacher.com/knuckle-joint/

[3] Totalmateria, *G32NiCrMo8*, http://www.totalmateria.com

[4] Sandvik Coromant,*Catalogo generale 2018*, http://www.coromant.sandvik.com/it

[5] Norme UNI, Ente nazionale italiano di unificazione