**Data Science Project Protocol**

*Author(s): Vladi Polonsky*

*May 2020*

Contents

[1 Introduction and Objectives 1](#_Toc40129814)

[1.1 General 1](#_Toc40129815)

[1.2 Audience and Document Scope 1](#_Toc40129816)

[1.3 References 1](#_Toc40129817)

[1.4 Terms and Definitions 1](#_Toc40129818)

[2 Methodology 1](#_Toc40129819)

[2.1 Data 1](#_Toc40129820)

[2.2 Models 1](#_Toc40129821)

[2.3 Deployment 1](#_Toc40129822)

[3 Results 1](#_Toc40129823)

[4 Conclusion 1](#_Toc40129824)

# Introduction and Objectives

## General

Airbnb is a home-sharing platform that allows home-owners and renters (‘hosts’) to put their properties (‘listings’) online, so that guests can pay to stay in them. Hosts are expected to set their own prices for their listings. Although Airbnb and other sites provide some general guidance, there are currently no free and accurate services which help hosts price their properties using a wide range of data points.

Paid third party pricing software is available, but generally you are required to put in your own expected average nightly price (‘base price’), and the algorithm will vary the daily price around that base price on each day depending on day of the week, seasonality, how far away the date is, and other factors.

Airbnb pricing is important to get right, particularly in big cities like San-Fransisco where there is lots of competition and even small differences in prices can make a big difference. It is also a difficult thing to do correctly — price too high and no one will book. Price too low and you’ll be missing out on a lot of potential income.

This project aims to solve this problem, by using machine learning to predict level of occupation of a property in San-Francisco given set of features describing propery, amenities in the property, neigborhood, price and etc.

## References

* <https://towardsdatascience.com/predicting-airbnb-prices-with-machine-learning-and-deep-learning-f46d44afb8a6>
* <https://towardsdatascience.com/going-dutch-how-i-used-data-science-and-machine-learning-to-find-an-apartment-in-amsterdam-part-def30d6799e4>
* <https://towardsdatascience.com/predicting-airbnb-prices-with-machine-learning-and-location-data-5c1e033d0a5a>
* <http://users.ece.northwestern.edu/~xws864/eecs349/doc/FinalReport.pdf>
* <https://medium.com/asif-anwar/utrecht-apartment-hunting-da9883131ff6>

# Methodology

## Data

The dataset used for this project comes from [Insideairbnb.com](https://insideairbnb.com/), an anti-Airbnb lobby group that scrapes Airbnb listings, reviews and calendar data from multiple cities around the world. The dataset

includes scrapping of San-Fransisco Airbnb listings starting 2015 and ending 2020 and includes information about 25,000 listings that were active during this period.

The objective of the project to predict property occupation level (Full, Medium, Low) in week granularity, therefore daily statistics were aggregated on week basis. Only weeks that have report of all the weekdays were left for processing. Insideairbnb scraps properties calendar periodically and have future calendar of the properies (but not the actual history).

To keep data trustworthy, we will use only on the first few weeks coming right after the scrapping date (e.g. if the scrapping date was 1 June 2018, we will use only the week of the June, assuming that most of the orders/cancellations for June were done before that and June calendar is more or less final).

## Models

* Dataset was split into 2 parts (train, dev and test) using 50/20/30 ratio.
* Dataset is pretty balanced no need to need to do any special balancing.
* Classification models was used to predict the Y (level of occupation)
* Cross Validation was used to find the best value of model parameters (mtry)
* To compare model performance LogLess and Accuracy measures were used

## Deployment

# Results

The final amount of data used (total, train, test, etc)

|  |  |
| --- | --- |
| **Train** | 181468 |
| **Dev** | 45367 |
| **Test** | 56709 |

The amount of outliers and the way of treating them

* Each variable was check for ourliers and treated accordingly (as described in Cleansing.ipynb)

The amount of missing values and the methods used for imputing them

* The percentage of entries with missing variables wasn;t high (5%), therefore they were deleted from the dataset

The distribution of the data (timeframes)

* Five years in week granularity

The methods used to transform the data and to generate new features

* Transformation done as part of the Cleansing (as described in ***Cleansing.ipynb***)
* Variables manipulation, dummy conversion (as described in ***Features Engineering.R***)

# Conclusion

* Dataset1

**Name Model MultiLogLoss Accuracy**

1        Decision Trees-tree  mod3    0.7757796     57.79032

2       Decision Trees-rpart  mod4    0.8345239     57.69730

3                RandomForest  mod5    0.5506116     76.10842

4  RandomForest (ranger)  mod6    0.5663723     77.05783

5                         XGBoost  mod7    0.7634453     66.11681

2                                SVM  mod9    0.6077998     75.23598

3                        Adaboost mod10    3.1645653     59.39615

4                                gbm mod11    0.6668967     68.80575

* Dataset2

**Name Model MultiLogLoss Accuracy**

1        Decision Trees-tree  mod3    0.7757796     58.69032

2       Decision Trees-rpart  mod4    0.8345239     58.59730

3                RandomForest  mod5    0.5406116     77.07842

4  RandomForest (ranger)  mod6    0.5263723     77.15783

5                         XGBoost  mod7    0.7434453     66.11681

2                                SVM  mod9    0.6087998     75.23598

3                        Adaboost mod10    3.1675653     59.29615

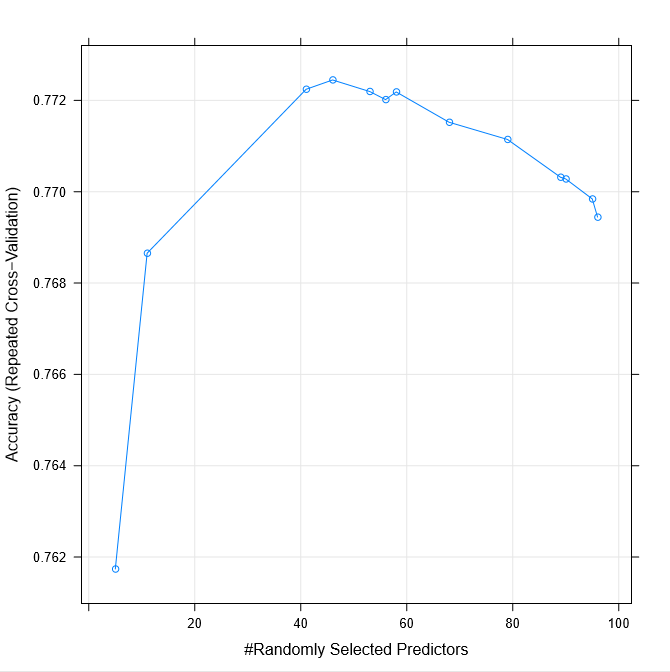
4                                gbm mod11    0.6663967     68.70575

* Final Result

A screenshot of a cell phone

Description automatically generated

* Result of Fine Tuning (run time 48 hours): mtry = 46



# Appendix A

Source files description (files are ordered according to the processing flow)

|  |  |
| --- | --- |
| File | Description |
| Airbnb - Data Retrieval Protocol.xlsx | Has 2 sheets:   * Protocol of the data * List of variables taking part in dataset 1 and dataset 2 |
| StoreInDB.R | * run through the calendars and listsing, extract the interval of 4 weeks since scraping day * remove variable with high missing percentage * upload to SQL Server |
| Research.sql | * Research of the data |
| FF.sql | * Data preparation for Flat File creation |
| FF Data Preparation.ipynb | * Flat File Preparation * Extract data from database * Add a Target variable * Clean invalid entries * Store as a .CSV file |
| Train\_test\_partition.ipynb | * Subsetting the dataset into Train, Dev and Test |
| before\_cleanising | * Output of exploreData (‘mechkar’ package) before the cleansing |
| Cleansing.ipynb | * Cleaning FF data (train, dev, test) from outliers and missing values * Running mechkar expoloreData function |
| after\_cleanising | * Output of exploreData (‘mechkar’ package) after the cleansing |
| Feature Engineering.R | Add new variables   * host\_about\_len – length of propery describtion * summary\_len – length of summary describtion * total\_amenities – number of amenities in the property * price\_per\_person – avg\_proce/accommodates   Convert categorical properties into dummies  Creating 2 datasets   * dataset 1 - set of variables of type *ordinal categories* represented as dummies * dataset 2 – same set of variables represented as numerics |
| Features Selection Dataset1.ipynb | * Selection most influential features from Dataset1 |
| Features Selection Dataset2.ipynb | * Selection most influential features from Dataset2 |
| Modeling.R | * Trains and test performance of different models: * Desicion trees * Desicion trees rpart * Random Forest * Random Forest Ranger * XGBoost * kNN * SVM * AdaBoost * Gbm |
| CV.R | * Model Fine Tuning via Cross Validation to find the best parameters (mtry) |
| FinalCheck.R | * Verify final model built with improved *mtry* parameter against both Dev and Test datasets |
| EDA.ipynb |  |
| LoadDataframes.R | * Loads train/dev/test dataset1 and dataset2 into memory (used by other R files) |