AIRBNB ROOM CLASSIFICATION

Instructor: Prof. Ying Lin, Ph.D (Syracuse University)

Aniruddh Garge
Applied Data Science
Syracuse University
agarge@syr.edu

Jeet Ganatra
Applied Data Science
Syracuse University
jkganatr@syr.edu

Tanushree Shetty
Applied Data Science
Syracuse University
tshetty@syr.edu

ABSTRACT

Airbnb room classification system focuses on predicting the price-category each Airbnb room in NYC falls under based on the airbnb open data so that customers have a better experience selecting the type of rooms they want and fits their budget. We have implemented three algorithms (Random Forests, Support Vector Machines and Decision Trees) on the New York City airbnb open dataset. We concluded by comparing the accuracies of these three models to determine the most suitable method to accurately predict the price range of a room based on input parameters.

INTRODUCTION

Since 2008, guests and hosts have used Airbnb to expand on travelling possibilities and present more unique, personalized way of experiencing the world. Airbnb is an online marketplace for arranging or offering lodging, primarily homestays or tourism experiences. Today, Airbnb became one of a kind service that is used and recognized by the whole world. Data analysis on millions of listings provided through Airbnb is a crucial factor for the company. The millions of listings generate a lot of data - data that can be analyzed and used for security, business decisions, understanding of customers' and providers' (hosts) behavior and performance on the platform, guiding marketing initiatives, implementation of innovative additional services and much more.

Classification involves the predictive learning that differentiates a data item into one of several predefined classes. It constitutes examining the characteristics of an item and assigning to it a predefined class. Classification is divided into a two-step process. In the first step a model is built describing a predefined set of data classes and secondly, the model is used for classification. Airbnb room classification system would classify rooms on the following categories: Very Cheap, Cheap, Moderate, Expensive, Very Expensive.

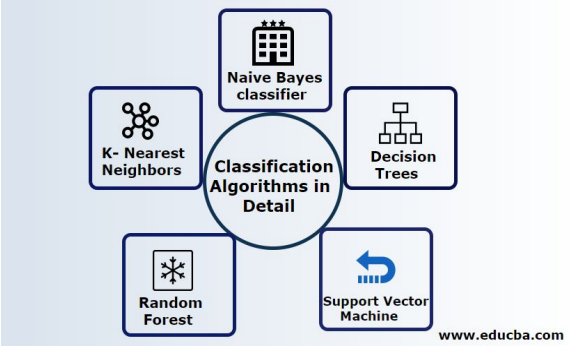

Figure 1: Types of Classification ML Algorithms

DATA DESCRIPTION

This dataset describes the listing activities and metrics in NYC, NY for 2019. The data has 48,895 entries and 16 total columns and is a mix of numerical and categorical values. The dataset has information on the borough, areas within the borough, location, type of room and reviews of Airbnb listings.

Id: listing ID
Name: name of the listing
Host_id: host ID
Host_name: name of the host
Neighbourhood_group: location
Neighbourhood: area
Latitude: latitude coordinates
Longitude: longitude coordinates
Room_type: listing space type
Price: 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

DATA PREPROCESSING

The project uses two types of preprocessing techniques like the hold-out method and discretization.

1. Hold-out method: Hold-out is when you split up your dataset into a 'train' and 'test' set. The training set is what the model is trained on, and the test set is used to see how well that model performs on unseen data. The project uses hold-out method by splitting 75% of the data for training and 25% of the data for testing.

2. Discretization: Discretization is the process of converting a continuous attribute into an ordinal attribute. A potentially infinite number of values are mapped into a small number of categories. It is commonly used in classification. The project uses discretization to convert the price attribute from a continuous attribute into a category of price ranges.

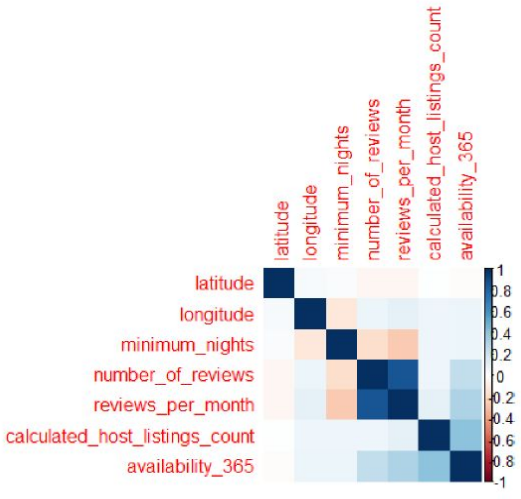

Figure 2: Output from Correlation Analysis

DATA FLOW

The original dataset consisted of 48, 895 records and 16 attributes. After data cleaning and exploratory analysis, the project was carried out with 47,840 records and 11 attributes which still contributed to approximately 99% of the data. This data was then split into the train and test set in the 3:1 ratio respectively. The train set was then used to train the Random Forests, Decision Trees and SVM models to predict the price range of a property listing based on input parameters.

MODEL DESCRIPTION

The following algorithms are implemented on the data to generate insights and create prediction models.

Decision Trees: Decision trees are one of the most popular algorithms used in machine learning, mostly for classification but also for regression problems. Our brain works like a decision tree every time we ask ourselves a question before making a decision. For example: is it cloudy outside? If yes, I will bring an umbrella.

When training a dataset to classify a variable, the idea of the Decision Tree is to divide the data into smaller datasets based on a certain feature value until the target variables all fall under one category. While the human brain decides to pick the “splitting feature” based on the experience (i.e. the cloudy sky), a computer splits the dataset based on the maximum information gain.

The algorithm defines nodes by calculating the Gini Index. To calculate Gini, we consider the probability of finding each class after a node, we sum the square of those values and we subtract this amount from 1. For this reason, when a subset is pure (i.e. there is only one class in it), Gini will be 0, because the probability of finding that class is 1 and in that case, we say we have reached a leaf, because there is no need to split anymore as we achieved our goal.

Depending on the splitting strategy that we choose, we will have different values of Gini for each subset, and depending on the Gini value after a node, we can define the Information Gain: this is defined as the difference between the

Gini of the parent, and the weighted average of the children's Gini. The decision tree will then consider all the possible splits and choose the one with the highest information gain.

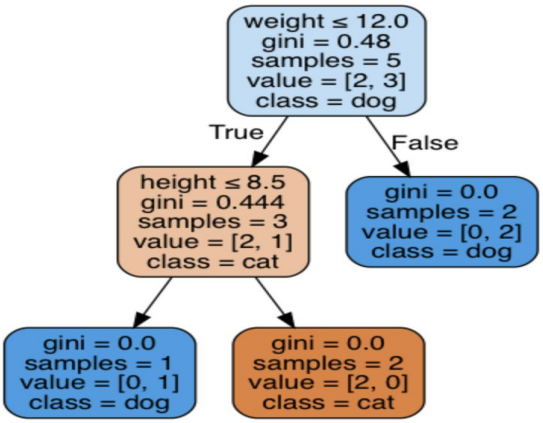

Figure 3: Structure of a Decision Tree Algorithm

All the three models are evaluated for the 'Accuracy' parameter. The model with the highest accuracy is then used to further predict price range using input parameters from the test data.

Support Vector Machines: The objective of the SVM algorithm is to find a hyperplane in an N-dimensional space(N — the number of features) that distinctly classifies the data points. To separate the two classes of data points, there are many possible hyperplanes that could be chosen. Our objective is to find a plane that has the maximum margin, i.e the maximum distance between data points of both classes. Maximizing the margin distance provides some reinforcement so that future data points can be classified with more confidence.

Hyperplanes are decision boundaries that help classify the data points. Data points falling on either side of the hyperplane can be attributed to different classes. Also, the dimension of the hyperplane depends upon the number of features. If the number of input features is 2, then the hyperplane is just a line. If the number of input features is 3, then the hyperplane becomes a two-dimensional plane. It becomes difficult to imagine when the number of features exceeds 3.

Support vectors are data points that are closer to the hyperplane and influence the position and orientation of the hyperplane. Using these support vectors, we maximize the margin of the classifier. Deleting the support vectors will change the position of the hyperplane. These are the points that help us build our SVM.

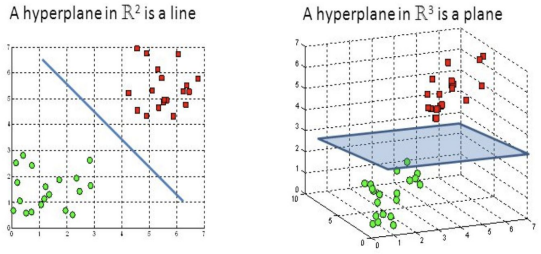

Figure 4: Hyperplanes in 2D and 3D feature space

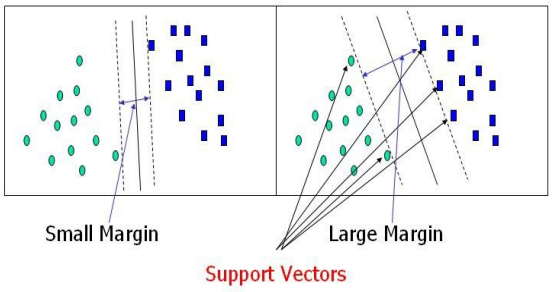

Figure 5: Small and Large Margin Support Vectors

Random Forest: Random Forest consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction.

The fundamental concept behind random forest is a simple but powerful one — the wisdom of crowds. In data science speak, the reason that the random forest model works so well is:
A large number of relatively uncorrelated models (trees) operating as a committee will outperform any of the individual constituent models.

The low correlation between models is the key. Just like how investments with low correlations (like stocks and bonds) come together to form a portfolio that is greater than the sum of its parts, uncorrelated models can produce ensemble predictions that are more accurate than any of the individual predictions. The reason for this wonderful effect is that the trees protect each other from their individual errors (as long as they don't constantly all err in the same direction). While some trees may be wrong, many other trees will be right, so as a group the trees are able to move in the correct direction. So the prerequisites for random forest to perform well are:

1. There needs to be some actual signal in our features so that models built using those features do better than random guessing.

2. The predictions (and therefore the errors) made by the individual trees need to have low correlations with each other.

Model	Features	Accuracy
Decision Trees	Neighborhood, Location, Room-type, Reviews, Availability	70%
Support Vector Machines	Neighborhood, Location, Room-type, Reviews, Availability	69.04%
Random Forest	Neighborhood, Location, Room-type, Reviews, Availability	60.84%

Table 1: Model Description and Evaluation

MODEL EVALUATION

Accuracy of Decision Trees: The model with the highest accuracy was the Decision Tree. After fine tuning the decision tree model, we got an accuracy of 70%. This model also took the least amount of time to run as compared to the other models. The hyperparameters were tuned such that the tuneLength was set to 10, minsplit was set to 9 and tuneGrid was used to improve the model performance.

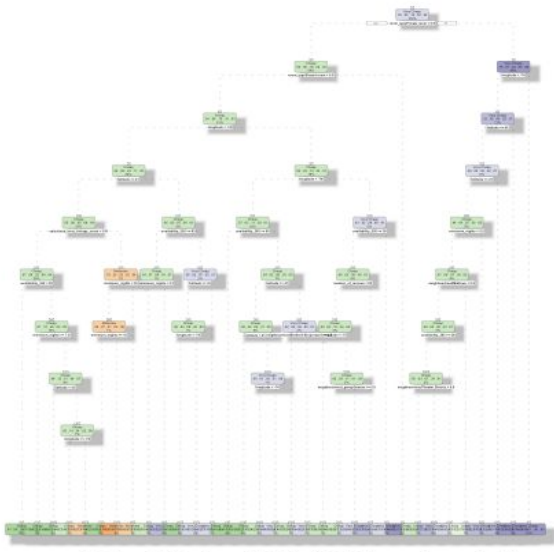

Figure 6: Output of Decision Tree Model

Accuracy of SVM: The model that took the longest to run was SVM. It had an accuracy of 69.04% which is very close to the Decision Tree model.

In order to improve the model performance, the hyperparameters were tuned by using TuneGrid in the sequence(1,10,3) and trainControl was used to control the computational nuances of the train function.

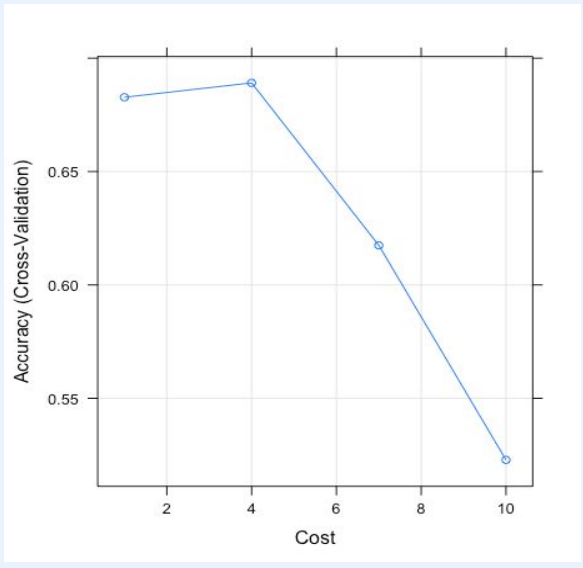

Figure 7: Plot of Cost vs Accuracy of SVM

Accuracy of Random Forest: The accuracy we got after implementing Random Forest on our training data was 60.84%.

The hyper-parameters used for this model were 'Accuracy' as the metric, 5-fold repeated cross-validation with 3 repeats for trainControl with grid search and tuneGrid. Finally, the tuneLength was limited to 15. Both 5-fold cross-validation and 3 repeats slows down the search process, but is intended to limit and reduce overfitting on the training set.

```
> varimp_rf <- varImp(model_rf)
> varimp_rf
rf variable importance

  only 20 most important variables shown (out of 231)

                                Overall
room_typePrivate room           100.000
longitude                        30.864
neighbourhood_groupManhattan     19.879
latitude                         12.800
neighbourhood_groupQueens         6.280
calculated_host_listings_count    5.970
neighbourhood_groupBrooklyn       5.616
minimum_nights                    4.985
room_typeShared room              4.711
neighbourhoodBushwick             4.350
neighbourhoodMidtown              3.644
neighbourhoodHell's Kitchen       2.901
neighbourhoodUpper East Side      2.734
neighbourhoodWest Village         2.107
neighbourhoodFinancial District   1.803
neighbourhoodChelsea              1.802
neighbourhoodEast Village         1.727
availability_365                  1.579
neighbourhoodBedford-Stuyvesant   1.467
neighbourhoodHarlem               1.365
```

Figure 8: Variable Importance with Random Forest

CONCLUSION

Our analysis on the Airbnb data using ML models like SVM, Decision Trees and Random Forest, predicts The price-category (Very cheap, Cheap, Moderate, Expensive, Very expensive) in which the listing will fall, based on the borough, neighborhood, location, reviews and availability. Overall, we

discovered a very good number of interesting relationships between features and explained each step of the process. From our test run, the insights we found are:

1. Manhattan and Brooklyn boroughs have the most expensive room listings in NYC
2. Almost all rooms in the Financial District, Manhattan have expensive rooms.

Future work can explore other model techniques and also tune the model for various hyper-parameters to attain better accuracy.

SHINY APP

Below is the link to our shiny web application:
https://tshetty.shinyapps.io/Shiny/

DATA SOURCE

https://www.kaggle.com/dgomonov/new-york-city-airbnb-open-data

http://insideairbnb.com/new-york-city/

ACKNOWLEDGEMENT

We would like to thank our Professor Ying Lin for his great teaching, support and guidance. We would also like to thank Vishwanath Hegde for his constant assistance during the entire semester.

REFERENCES

[1] Jeet Ganatra et al.2017, Mining Frequent Patterns Using Customer Experience. Int J Recent Sci Res. 8(3), pp. 15790-15795.

[2] Cox, M.–Slee, T. (2016): How Airbnb's data hid the facts in New York City
Retrieved from http://insideairbnb.com/reports/how-airbnbs-data-hid-the-facts-in-new-york-City.pdf

[3] Delgado-Medrano, H. M.–Lyon, K. (2016): Short Changing New York City – The impact of Airbnb on New York City's housing market BJH Advisors LLC.
Retrieved from http://www.sharebetter.org/wp-content/uploads/2016/06/NYCHousingReport_Final.pdf

[4] Savan Patel (2017). SVM (Support Vector Machine) - Theory.
Retrieved from https://medium.com/machine-learning-101/chapter-2-svm-support-vector-machine-theory-f0812effc72

[5] Eijaz Allibhai (2018). Hold-Out vs. Cross-Validation in Machine Learning.
Retrieved from https://medium.com/@eijaz/holdout-vs-cross-validation-in-machine-learning-7637112d3f8f

[6] Tarun Gupta (2019). Data Preprocessing in Data Mining and Machine Learning.
Retrieved from https://towardsdatascience.com/data-preprocessing-in-data-mining-machine-learning-79a9662e2eb

[7] Tony Yiu (2019). Understanding Random Forest.
Retrieved from https://towardsdatascience.com/understanding-random-forest-58381e0602d2

[8] Rohith Gandhi (2018). Support Vector Machine - Introduction to Machine Learning Algorithms.
Retrieved from https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47

[9] Emma Grimaldi (2018). Decision Tree - An algorithm that works like the human brain.
Retrieved from https://towardsdatascience.com/decision-tree-an-algorithm-that-works-like-the-human-brain-8bc0652f1fc6