Airbnb Price Prediction Model Using Ensemble Learning Methods

Xincheng Luo, Xinran Yao, Swapnil Pade, Sarvani Sambaraju, Sanjana Gupta

ESSEC & CentraleSupélec

Abstract. Airbnb has become a popular alternative to traditional hotels and has disrupted the hospitality industry. Through Airbnb, individuals can list their own properties as rental places. Taking New York City alone, there are around 40,000 listings. While traditional hotels have teams that carefully measure demand and supply to adjust pricing, as a host, it could be challenging to determine the optimal price for a listing. The variation in types of listings can also make it difficult for renters to get an accurate sense of fair pricing. In this project, we will use ensemble learning approaches to predict the price of Airbnb listings in New York City.

1 Introduction

With the help of Airbnb, people may make money by renting out their houses, flats, or other accommodations to tourists. Providing travellers with affordable and practical lodging options, Airbnb has grown to be a key player in the hospitality sector in major cities such as New York. The price of Airbnb listings, which can vary depending on a number of criteria such as location, seasonality, and availability, is one of the issues for both hosts and guests. Ensemble approaches can be used to forecast Airbnb rates in New York, helping owners improve their pricing tactics and assisting guests in finding the greatest offers.

These techniques combine several algorithms to provide predictions that are more reliable and precise, making them an excellent tool for this type of task. Our main goal is to apply all approaches taught in the course and practised in lab sessions (Decision Trees, Bagging, Random forests, Boosting, Gradient Boosted Trees, AdaBoost, Catboost etc.). The goal is to predict the target variable: price of the listing and compare performances of models using various metrics learned in class.

the dataset[1] for this project is obtained from Kaggle and contains the listings in New York City in 2019. The data set includes 15 features on listings, including:

- Name of the listing
- Neighborhood
- Price
- Review information
- Availability

The dataset contains around 47,000 listings.

2 Preprocessing & Exploratory Data Analysis

We first dealt with the missing data, there are 10052 null values in the 'reviews_per_month' and we filled them with column mean, at the same time, we deleted the rows without price information.

For the exploratory analysis, we divided the tasks into dealing with numerical features and dealing with categorical ones. For the numerical features, we firstly drew the distribution plots and checked the outliers among them. In this step, we originally tried to narrow the range of features such as 'calculated_host_listings count' and 'reviews_per_month'. Nevertheless, it made little improvement on the result during iterative experiments, so we left them out. Then we drew a correlation heatmap among all the numerical features and found a high correlation between the 'number_of_reviews' and 'reviews_per_month'.

For the categorical features, we found that Staten Island has the highest mean availability value around 220-250 days compared to others, Manhattan seems to have the most expensive prices, and entire roomsapts seems to deserve higher prices compared to private rooms and shared rooms by violin plots and box plots. To better check the correlation between categorical features and our dependent variable, we use ANOVA test and finally confirm that 'room_type', 'neighbourhood_group', 'host_id_count' have much to do with price. At last, we also tried to perform string operations on the categorical feature 'name' in order to better understand the impact of the description on the price. This process involved removing special characters and stopwords to create a clean final_name. However the trial is abandoned at the end due to a high accuracy but low recall.

3 Methodology

In our initial trial to build the predicting model,, we replace all the empty name fields with string, then get rid of all the punctuations, digits and special characters and then proceed to remove the stop words. We defined the parameters as follows: $price \geq 300$ was termed to be expensive and $price < 300$ was termed as affordable. We split the data into 80% training dataset and 20% test dataset. We used the TF-IDF metric to represent the weights of each feature. The predicting model was based solely on the textual descriptions of the listings by using the LGBM classifier and using the features 'target', 'price' and 'final_name'. We used accuracy and recall value as the evaluation metrics and got an accuracy score of 80%. The scores were not satisfying but taking into account that the predicting model is built solely on textual description of the listing, it seems like the words in AirBnb titles actually did matter.

We then move on to do predictive modelling using various ensemble learning algorithms. To do so we implement feature selection to get better targeted results. For the x-data we choose the following features:

'Neighbourhood_group', 'Neighbourhood', 'Latitude', 'Longitude', 'Number_of_reviews', 'Room_type', 'Minimum_nights', 'Reviews_per_month', 'calculated_host_listings_count', 'Availability_365', 'host_id_count'.

For the y-data we choose log10('price') as the selected variable. We then split our data into categorical and numerical columns and split them into test and train data with a 20% - 80% split ratio. For the categorical columns we chose One-hot-encoder as the pre-processor and for the numerical data we chose standard_scalar as the pre-processor.

4 Modelling

The first pre-trained model we trained on was the Lasso model, we set the $alpha = 0.001$. The Lasso model operates by including a penalty term in the linear regression's cost function, which pushes the model to pick just the most crucial features and reduce the coefficients of less crucial ones. The key benefit of the Lasso model is that it can deliver results that are easy to understand by emphasising the crucial aspects of the forecast. The Lasso model presupposes a linear relationship between the characteristics and the target variable, which may not always be the case in actual use.

The second pre-trained model we trained on was the Ridge model, we set the $alpha = 0.001$. The Ridge model employs the L2 regularisation penalty which motivates the model to minimise the squared sum of the coefficients. The Ridge model's key benefit is that it can increase the model's stability and generalisation. It is important to keep in mind that the Ridge model makes the assumption that all features are important for the prediction, which may not always be the case in actual use.

We then trained on a pre-trained Decision Tree Regressor with max $depth = 3$. The decision tree model divides the data into subsets recursively based on the most important feature until a stopping requirement is satisfied. After that, the subset to which the new data point belongs is averaged to provide the final prediction. The Decision Tree Regressor's key benefit is its ability to handle both continuous and categorical data as well as capture intricate nonlinear correlations between feature sets and the intended variable. Nonetheless, it is susceptible to overfitting, particularly if the tree grows too deep and intricate.

We also trained on a pre-trained GradientBoostRegressor. The GradientBoost Regressor builds decision trees onto the model iteratively, with each new tree being trained to fix the mistakes of the prior trees. By updating the predicted values of the target variable based on the mistakes of the prior trees, it minimises a loss function during the training phase. The anticipated values of all the trees are averaged to produce the final projection. The GradientBoostRegressor's key benefit is its ability to handle both continuous and categorical data as well as capture intricate nonlinear correlations between the features and the target variable. It can lessen the model's bias and variance, which will increase generalisation and accuracy. However, if the quantity or depth of the trees is excessive, it may be computationally expensive and subject to overfitting.

We then trained on XGBoostRegressor and RandomForestRegressor and did hyperparameter tuning using Grid Search CV and Random Search CV for both. Hyperparameters yielded better results. The XGBoost

Regressor builds decision trees onto the model iteratively, with each new tree being trained to fix the mistakes of the prior trees. The XGBoost Regressor's strong performance and scalability, which make it appropriate for big and complicated datasets, are its key advantages.

We then trained on RandomForestRegressor and hyperparameter tuned it using Randomized SearchCV which boosted our results. The Random Forest Regressor builds numerous decision trees using subsets of the features and data points that are chosen at random. Each decision tree is trained on a different subset of the data during the training phase, which adds diversity and lessens overfitting. The outcome is then determined by averaging the values predicted by each decision tree. The Random Forest Regressor's key advantage is that it can handle noisy or partial data thanks to its high level of accuracy and durability.

And finally we trained our model on an ensemble of XGBoost and RandomForest Regressors by creating a new dataset using the predictions of the base models and then training a meta learner on the new dataset and making predictions on the test set using the ensemble model.

5 Evaluation

We choose R2 score as our evaluation metric for all our models because the R2 score is a metric that indicates how well your model performed, as opposed to the loss in terms of how many wells it performed[2]. When compared with MAE and MSE, the R2 score seems to be better as it is context-independent whereas MAE and MSE depend on the context. R-squared score provides us with a baseline model against which to assess alternative models that none of the other measures do. In essence, R-squared score determines whether a regression line is superior to a mean line. Since R-squared score increases as the regression line goes closer to one. The model's performance also gets better. In the table below we present a comparison of the R2 scores of all our models:

Models	Hyperparameters	Mean r2 score
Gradient Boost	Default	59.9
XGBoost Regressor (tuned) & RandomSearchCV	learning_rate = 0.01906, max_depth = 8, n_estimators = 1062	63.7
Random Forest Regressor (tuned)	max_depth = 10 max_features = auto, n_estimators = 1147	61.6
Lasso Model	Default	52.2
Ridge Model	Default	54.8
Ensemble of XGBoost and Random Forest	Default	58
Decision Tree Regressor	Default	49.1

Table 1. Comparative study of R2 scores for all models

The model giving us the best R2 score of 63.7% is XGBoost. We tuned this model using hyperparameter tuning to boost our results. Another model performing very well and giving a R2 score of 61.6% is the Random Forest Regressor. We tuned this model using hyperparameter tuning to boost our results.

References

1. datasource: https://www.kaggle.com/datasets/dgomonov/new-york-city-airbnb-open-data.
2. Chicco D, Warrens MJ, Jurman G. The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation. PeerJ Comput Sci. 2021 Jul 5;7:e623. doi: 10.7717/peerj-cs.623. PMID: 34307865; PMCID: PMC8279135.
3. Jia Wu, Xiu-Yun Chen, Hao Zhang, Li-Dong Xiong, Hang Lei, Si-Hao Deng, Hyperparameter Optimization for Machine Learning Models Based on Bayesian Optimization, Journal of Electronic Science and Technology, Volume 17, Issue 1, 2019, Pages 26-40, ISSN 1674-862X, https://doi.org/10.11989/JEST.1674-862X.80904120.

Decision Tree Implementation on Iris Dataset

Xincheng Luo, Xinran Yao, Swapnil Pade, Sarvani Sambaraju, Sanjana Gupta

ESSEC & CentraleSupélec

1 Methodology

Decision tree is a popular supervised machine learning algorithm used for classification and regression tasks. It works by recursively splitting the input data into smaller subsets based on the features that best separate the data. The splitting process continues until a stopping criterion is met. The resulting tree structure can be used to make predictions on new data. The decision tree algorithm is widely used because of its simplicity, interpretability, and ability to handle both numerical and categorical data.

2 Modelling

We implemented the decision tree algorithm using Python and the Scikit-learn library. We used the Iris dataset, which is a classic dataset containing the measurements of the flowers of three different species of Iris. The dataset has four input features: sepal length, sepal width, petal length, and petal width. The target variable is the species of the Iris, which can take one of three values: setosa, versicolor, or virginica.

We split the data into training and testing sets, with a ratio of 80:20. We then trained the decision tree model on the training set using the Gini impurity criterion. The Gini impurity measures the degree of impurity of a set of labels, and the split that reduces the impurity the most is chosen at each node. We set the maximum depth of the tree to 100 to avoid overfitting. We evaluated the performance of the model on the testing set using accuracy as the evaluation metric.

3 Result

Our decision tree model achieved an accuracy of 33.33% on the testing set.

4 Conclusion

Decision trees are basically comparison sequences that can train to perform classification and regression tasks. We ran python scripts that trained a decision tree classifier, used our classifier to predict the class of several data samples, and computed the precision and recall metrics of the predictions on the training set and the test set.The accuracy of a model is improved when you prune the tree. Depending on the model, both pre-pruning and post-pruning may be needed to prevent overfitting, underfitting, and to fine-tune the model. When applying regression, labels in regression problems are continuous. This can also classify however labels are calculated dynamically using mean.

In conclusion, the decision tree algorithm is a powerful and interpretable machine learning algorithm that can be used for classification and regression tasks. The decision tree model can be used for a wide range of applications, from medical diagnosis to customer segmentation.

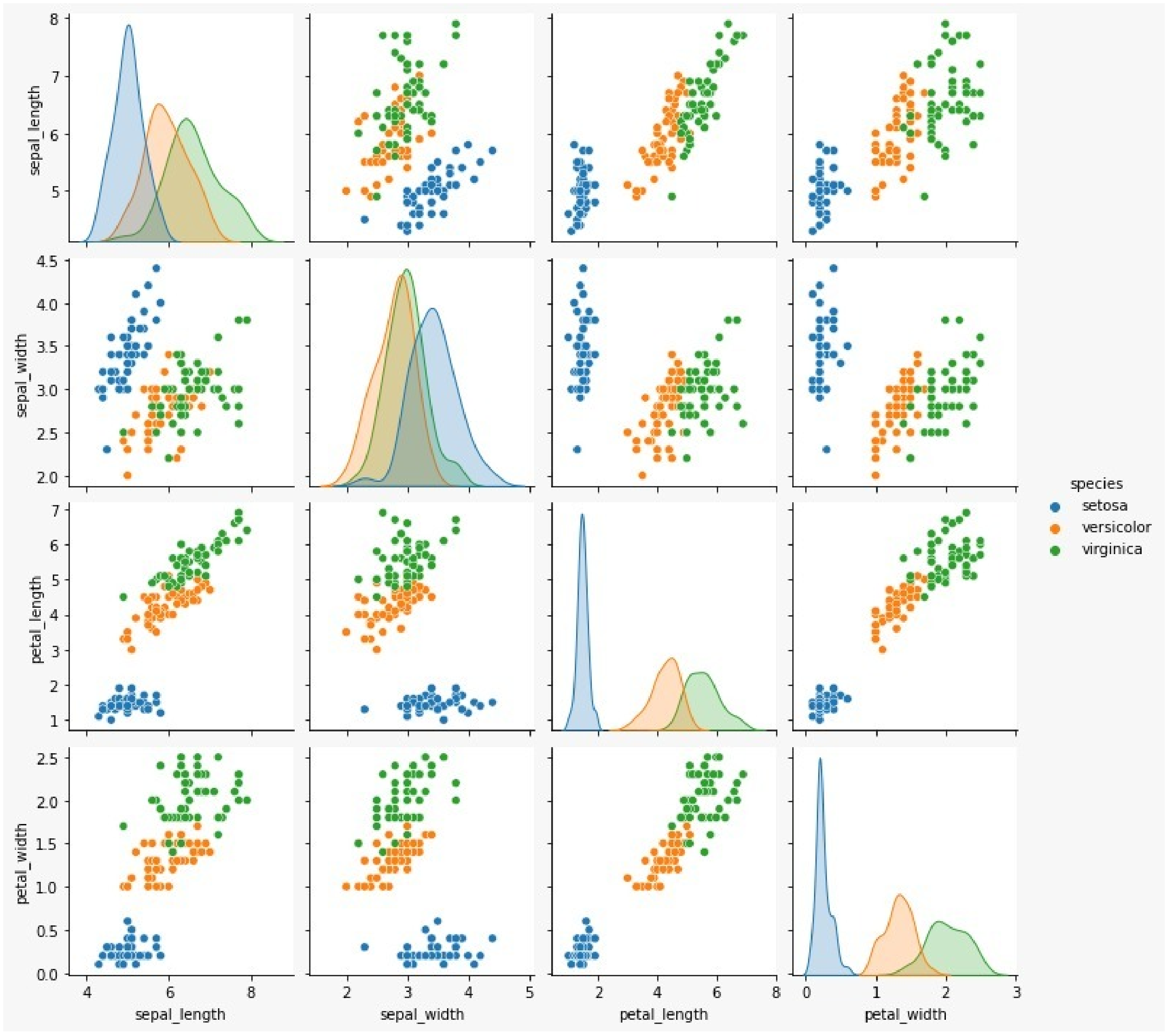

Fig. 1. Pairwise Relationships between features.