An Assignment on

“**Airbnb price prediction**”

BUSM131 Masterclass in Business Analytics

School of Business and Management

Queen Mary university of London

**Prepared by**

Nava Bhargav Gedda [230270185]

bs23895@qmul.ac.uk

**Contents**

[**Abstract** 3](#_Toc166236467)

[**Introduction** 4](#_Toc166236468)

[**Academic research** 4](#_Toc166236469)

[**Summary structure** 5](#_Toc166236470)

[**Business problem** 6](#_Toc166236471)

[**Business context** 6](#_Toc166236472)

[**Decisions Supported by Analytics and processes.** 7](#_Toc166236473)

[**Similar Organizations Leveraging Business Analytics** 7](#_Toc166236474)

[**Data, EDA, and methods** 8](#_Toc166236475)

[**Dataset features** 8](#_Toc166236476)

[**Exploratory Data Analysis** 10](#_Toc166236477)

[**Preprocessing pipeline** 19](#_Toc166236478)

[**Feature engineering** 20](#_Toc166236479)

[**Test -Train split and cross validation.** 26](#_Toc166236480)

[**ML models used in project.** 27](#_Toc166236481)

[**Machine learning Libraries** 28](#_Toc166236482)

[**Analysis and results** 29](#_Toc166236483)

[**Analysis** 29](#_Toc166236484)

[**Model selection and observations.** 30](#_Toc166236485)

[**Discussions and conclusion** 34](#_Toc166236486)

[**Analysis of XGBoost Results** 35](#_Toc166236487)

[**Analysis of Support vector regression** 36](#_Toc166236488)

[**Organisational changes required for implementation.** 37](#_Toc166236489)

[**Business decisions affected.** 38](#_Toc166236490)

[**Limitations and future considerations** 38](#_Toc166236491)

[**Appendix** 39](#_Toc166236492)

[**Project code** 39](#_Toc166236493)

[**Dataset tables and graphs** 58](#_Toc166236494)

[**References** 63](#_Toc166236495)

**Abstract**

Airbnb has become increasingly popular among travellers for accommodation across the world. The project tackles the challenge of optimizing Airbnb pricing strategy in NYC, a highly competitive market for short term rentals. Empowering hosts with data-driven insights thus helping them to improve their business performance. Traditional static pricing maybe inefficient to capture the dynamic nature of Airbnb market, factors like travel times from different attractions, amenities, seasonality can significantly impact the hosts revenue influencing the supply with demand (Chen, 2015).

In this project “NYC Airbnb” dataset from Kaggle containing 74111 rows and 29 columns. The dataset has then been cleaned to ensure data quality and exploratory data analysis (EDA) helps to identify the relevant variables or columns. Null values are handled by using feature engineering. To achieve the challenge of optimizing Airbnb pricing strategy, ML models have been employed to evaluate and compare. These include linear regression, random forest regression, ridge and lasso regression, support vector regression (SVR), XGradient Boosting and artificial neural networks (ANN’s).

The performance of these models is evaluated using the R-squared (r2) and root mean square error (RMSE) to identify the most appropriate variables which helps in predicting Airbnb listing prices. Experimentation shows that XGBoost can achieve an r2 score of 78.07% and RMSE of 0.31 (defined on log price) on the unseen data, outperforming other models as mentioned above. This project equips hosts with a reliable price optimizing strategy that facilitates dynamic pricing strategies, enabling them to adapt to market fluctuations, maximize their earning potential, attract more guests, and enhance their overall Airbnb business success in the competitive NYC market.

**Introduction**

**Summary of problem**

The phenomenal growth of Airbnb has transformed the travel accommodation landscape, particularly in the metropolitan cities like New York City (NYC). While offering a unique and potential income stream for hosts, navigating the competitive NYC Airbnb market presents significant challenges. Unlike hotels, which have their own pricing system, the hosts usually determine Airbnb prices empirically i.e. determining optimal listing prices. While Airbnb provides pricing tips to assist hosts, concerns about transparency and accuracy have emerged, prompting hosts to supplement these recommendations with their own market knowledge. Setting prices too high can deter potential guests, leading to vacancies and lost revenue. Conversely, under-pricing can leave money on the table and limit earning potential.

## **Academic research**

Several academic studies have explored the application of machine learning (ML) for predicting Airbnb listing prices. However, there is a gap in research specifically addressing Airbnb price prediction in NYC using the latest machine learning models. This project aims to address this gap by developing a **comprehensive set of new features** for machine learning models. Thus, aiming to improve the accuracy and effectiveness of price for NYC Airbnb listings. The findings from the study Alotaibi (2020) suggests that machine learning is beneficial in various aspects of the hotel industry, such as demand forecasting, price forecasting, booking cancellation prediction, financial efficiency, and work efficiency. The study also highlights that machine learning algorithms outperform traditional models in terms of forecast accuracy. A few literatures also mention of how Airbnb utilizes machine learning in its pricing algorithm tool.

Similarly, Wang et al. (2015) identified a gap in research on applying dynamic pricing models beyond traditional sectors like airlines and hotels. They proposed that the sharing economy, particularly Airbnb, offers a unique opportunity to explore dynamic pricing due to its distinct characteristics (Gibbs et al., 2018). In the context of Airbnb, machine learning algorithms power pricing tools that suggest prices to hosts for available dates. While these tools empower hosts with data-driven insights, they still retain control over pricing decisions. This highlights the importance of considering both host characteristics and listing/market factors when analysing the effectiveness of machine learning for dynamic pricing in the sharing economy. Another literature focuses on machine learning methods such as generalized additive models (GAM), deep neural networks, random forest, XGBoost, and bagging in the context of predicting Airbnb prices. These methods were explored and compared to identify the optimal prediction model for Airbnb prices (A.Zhu et al.,2020). The study found that bagging, XGBoost, and random forest demonstrated robust performance with the test data collected from the Airbnb website.

## **Summary structure**

This report uses regression and machine learning models trained on the Airbnb data set obtained from Kaggle collected by Steve Zheng. The analysis starts with data acquisition and cleaning followed by exploratory data analysis where we explore data characteristics and identify relevant variables, then the feature engineering to create new features including travel times both borough specific and airport, and PCA-based amenity features. The next step covers about the machine learning models and explain how they work in the context of price prediction. Insights are drawn about the metrics (r2 and RMSE) in model evaluation and results were analysis. Finally, the report concludes by summarizing the key findings and discuss about the limitations and improvements..

# **Business problem**

## **Business context**

In the hospitality industry, revenue management techniques have been crucial for maximizing profits in deals with variable demand and perishable inventories. This context is applicable to Airbnb hosts in NYC, where demand fluctuates based on seasons, events, and other factors. For this project, the business problem is “Helping hosts in optimizing pricing strategies for NYC Airbnb listings.” This requires considering factors which influence the guest decisions and include the guest preferences like travel time from airport and attractions, market conditions like seasonality and listing characteristics which include location (including borough), number of bedrooms and bathrooms, amenities listed, room type, etc.

Data-driven strategies support decisions in revenue management by utilizing algorithms and optimization methods to automate price and resource management for industries like airlines and hotels (Fitzpatrick et al., 2023). These strategies help in setting competitive prices, forecasting demand, and maximizing revenue by analysing historical data and current market conditions. By leveraging analytics and machine learning algorithms, hosts can make data-driven pricing decisions. The data-driven approach allows for more accurate price adjustments, occupancy targeting, and revenue optimization, improving efficiency and profitability in revenue management practices (Deep Learning Based Dynamic Pricing Model for Hotel Revenue Management, 2019).

## **Decisions Supported by Analytics and processes.**

Analytics enables dynamic pricing adjustments in response to real-time market conditions, ensuring competitiveness and revenue maximization. Integrated within listing management processes, analytics tools streamline pricing optimization for individual listings while fostering proactive customer engagement through transparent communication of price changes. These models predict optimal prices based on historical data, market trends, and competitor pricing, enabling hosts to dynamically adjust prices and maximize earning potential, as exemplified by forecasting surges in demand during events like Thanksgiving (Airbnb Help Centre). By analysing guest reviews and booking patterns, hosts gain insights into guest preferences, tailoring pricing strategies to attract specific segments, such as pet-friendly accommodations, as evidenced by research conducted by Chen & Chu (2022). This approach is further supported by resources provided by the Airbnb Help Centre, offering guidance on competitive pricing and dynamic pricing tools, while a Forbes article emphasizes the increasing adoption of machine learning to optimize pricing strategies in competitive markets like NYC (Forbes, 2023).

## **Similar Organizations Leveraging Business Analytics**

Several other organizations within the sharing economy face similar challenges of optimizing pricing strategies in a dynamic marketplace. VRBO, another vacation rental platform, offers hosts resources and tools for pricing management, utilizing data analytics to provide insights into market trends and competitor pricing. Uber, the ride-sharing giant, heavily relies on business analytics and machine learning to optimize pricing for riders and drivers, considering factors like demand, location, and time of day to determine surge pricing and ensure efficient service allocation. Similarly, GrubHub, DoorDash, the food delivery platform, utilizes business analytics to optimize pricing for restaurants and delivery fees for customers, considering factors like restaurant popularity, delivery distance, and order size to determine pricing strategies.

By leveraging business analytics and machine learning, these organizations empower their users to make informed decisions and optimize their earning potential within the dynamic sharing economy landscape. This analysis underscores the critical role of business analytics in supporting informed pricing decisions for NYC Airbnb hosts, highlighting the need for further exploration to provide valuable insights into the most impactful factors influencing listing prices in the NYC market, empowering hosts to craft data-driven pricing strategies and maximize their earning potential in the competitive NYC Airbnb landscape.

# **Data, EDA, and methods**

## **Dataset features**

The Airbnb NYC dataset includes 74111 observations and 29 variables. It has both relevant and irrelevant variables with null values in it. Variables which are irrelevant for the analysis such as id, first\_review, host\_response\_rate, review\_scores\_rating, last\_review, thumbnail\_url, were excluded. Thus, the new dataset includes 32349 observations and twenty-three variables.

The below table shows the features with their data types from the NYC dataset from Kaggle.

A table of numbers and objects

Description automatically generated

The below table shows the variables which have null values from the NYC dataset.

A table with numbers and text

Description automatically generated

The variables with missing values are handled by using two-prolonged approaches: median imputation for specific columns (bathrooms,, bedrooms, bed) and dropping the columns with higher number of null values, thus minimizing their potential impact on the model performance. The target variables include log\_price, room\_type, amenities, accommodates, bedrooms and cleaning\_fee. External relevant variables include the principal component analysis (PCA) variables and then the travel times from Airbnb location to airports and attractions from each borough. We also add a new column “Price” which has the normal price values after converting them from log price.

## **Exploratory Data Analysis**

In the EDA conducted for the report, various visualizations were employed to gain insight into the Airbnb dataset. The figure below illustrates the distribution of Airbnb listings across different city types, providing an overview of datasets geographical coverage. The plot is plotted with cities and its frequency of occurrence. New York City (NYC), with a count of 32349, has the highest frequency of occurrence when compared to remaining cities.

Cross-ref: **Airbnb’s listing across different cities.**A graph of different colored squares

Description automatically generated

Subsequently, a heat map visualization has been used to map the density of listing based on their geographical coordinates, highlighting the spread of Airbnb’s present in NYC data. Moreover, the geological patterns in the neighbourhoods of Manhattan south-western Bronx, west Queen, and northwestern Brooklyn neighbourhoods are much denser than the rest.

Cross-ref: **Heat map visualization for Airbnb’s in NYC**A map of a city

Description automatically generated

The distribution of log prices in NYC was visualized through a histogram, revealing a positive skew, offering insights into the pricing structure within the dataset. The distribution is not a normal distribution which states the presence of outliers or any other factors which can affect the distribution.

Cross-ref: **Histogram visualization for log price distribution**A graph of a logistic distribution

Description automatically generated

Further exploration was conducted to understand the distribution of room types in NYC dataset, which shows the frequency of their distribution within the dataset. Most of the Airbnb’s have Entire home/apt which is more than 16000 in the NYC dataset.

Cross-ref: **Bar map showing the frequency of room types in NYC dataset.**A graph of a diagram

Description automatically generated with medium confidence

To identify the top expensive boroughs in NYC, a bar plot showcasing the mean prices across boroughs was generated, aiding in identifying areas with higher-priced listings. The top 6 boroughs in our dataset are ordered by Manhattan, New Jersey, Staten Island, Brooklyn, Queens, and Bronx.

Cross-ref: **Bar plot showing the top six expensive Borough in NYC.**A bar graph with different colored bars

Description automatically generated

Box plots were utilized to visualize the density and distribution of prices for different room types, as well as to examine the correlation between instant bookable rooms and prices. Both box plots are ordered by Entire home/apt, Private room and shared room.

Cross-ref: **Box plot showing density and distribution of prices for room type and correlation between room type, log\_price and instantly bookable.**A diagram of a graph

Description automatically generated with medium confidence

**Figure**: Box plot showing density and distribution of prices for room type

Cross-ref: **Box plot showing density and distribution of prices for room type and correlation between room type, log\_price and instantly bookable.**A graph of a diagram

Description automatically generated with medium confidence

**Figure**: Box plot showing distribution of room type, log price and instantly bookable

Finally, the count of bedrooms, beds, and bathrooms was visualized using count plots, providing insights into the distribution of these attributes within the dataset.

Cross-ref: **Count plot showing the frequency of few explanatory variables.**A graph of a number of rooms

Description automatically generated

**Figure**: Plot shows the frequency of bedrooms on cleaned data set.

Cross-ref: **Count plot showing the frequency of few explanatory variables.**A graph of a number of beds

Description automatically generated

**Figure**: Plot shows the frequency of beds on cleaned data set

Cross-ref: **Count plot showing the frequency of few explanatory variables.**A graph of a bar graph

Description automatically generated

**Figure**: Plot shows the frequency of bathrooms on cleaned data set

Cross-ref: **Count plot showing the frequency of few explanatory variables.**A graph of a bed

Description automatically generated with medium confidence

**Figure**: Plot showing the frequency of bed type on cleaned data set

Additionally, a bar plot was employed to explore the relationship between the number of accommodates and the average price, shedding light on how accommodation capacity influences pricing.

Cross-ref: **Bar plot exploring the relationship between average price and number of accommodates.**A graph of different colored bars

Description automatically generated

These visualizations collectively offer a comprehensive understanding of the NYC Airbnb dataset, facilitating informed decision-making and further analysis.

## **Preprocessing pipeline**

The preprocessing pipeline aims to transform the raw data into a format suitable for machine learning models by handling missing values and scaling numerical features while encoding categorical features. Firstly, missing values in categorical columns ('bathrooms', 'bedrooms' and 'beds') are filled using the most frequent strategy via SimpleImputer, ensuring no data is lost. Then, OneHotEncoder is applied to convert categorical variables into a binary matrix, effectively representing each category as a binary vector. For numerical columns such as 'bathrooms', 'bedrooms', etc., missing values are imputed with the median value of each column to maintain robustness against outliers, followed by scaling using StandardScaler to standardize feature values, ensuring all features have a mean of 0 and a standard deviation of 1. This preprocessing pipeline, structured within a ColumnTransformer, allows for seamless integration into machine learning workflows, effectively handling both numerical and categorical data. It enhances model performance by ensuring uniformity and completeness in the dataset. The design of this pipeline is influenced by established practices in machine learning preprocessing techniques (Pedregosa et al.,2011) and leverages scikit-learn's robust preprocessing modules (Scikit-learn: Machine Learning in Python — Scikit-learn 1.4.2 Documentation, n.d.). The below table shows the variables which have null values from the NYC dataset after preprocessing.

A close-up of a document

Description automatically generated

## **Feature engineering**

In the feature engineering a thorough analysis of the cleaned dataset (data\_cleaned) is performed to gain insights into the distribution of features and to identify potential outliers or missing values.

Two significant new columns were introduced: "Price," which represents the normal price derived from the log-price column transformation, and "Borough," generated based on zip codes which are created using latitude and longitude information available in the dataset, enriching the dataset with additional context.

To address the potentially high dimensionality of the amenity features (116 columns), Principal Component Analysis (PCA) is employed (Sklearn.decomposition.PCA, n.d.). PCA is a dimensionality reduction technique that identifies a smaller set of uncorrelated features, called principal components (PCs), that capture most of the variance in the original data. This is done in a 3-step process starting with the preprocessing phase, then the one-hot encoding followed by identifying the principal components.

The preprocessing steps include splitting the amenities into individual rows, counting the frequency of each unique amenity, and converting the amenity list into a string for one-hot encoding. Amenities are one-hot encoded, creating a separate binary column for each unique amenity. PCA is then applied to the one-hot encoded amenity using a StandardScalar to ensure features have similar scale. A scree plot is generated to visualize the explained variance ratio by each principal component. This plot helps to determine the optimal number of principal components to retain. In the project, 6 principal components were chosen based on the “elbow” in the scree plot, indicating a significant drop in explained variance beyond that point. Below figure shows the scree plot with an “elbow” for all the 116 amenities. In this project the top 6 chosen principal components are ‘Carbon monoxide detector’, Fire extinguisher’, ‘First aid kit’, Smoke detector’, ‘Safety card’, ‘Lock on bedroom door’.

Cross-ref: **Scree plot for principal components w.r.t variances.**A graph with a line

Description automatically generated

**Figure**: Scree plot for 116 amenities and its respective variances.

These new features represent the most significant aspects of the original amenity data in a lower-dimensional space. This reduces model complexity and potentially improves performance without significant information loss. Thus, adding them into the cleaned dataset.

There are other crucial features which are appended into the cleaned dataset using Open-Source Routing Machine (OSRM). OSRM a powerful open-source software package which is designed for high-performing routing services. In this project OSRM has been used to get travel-time estimates between the geographic locations. The table below explains the variables obtained from the OSRM which is appended to the cleaned dataset.

A table with a list of travel time

Description automatically generated with medium confidence

The 28 attractions used for the columns avg\_lead\_time and min\_lead\_time includes Times Square, Yankee Stadium, Barclays centre, Brooklyn Bridge, Flushing Meadows, New York Giants, One world trade centre, Central Park, MSG, Empire stores, Saint Patricks Cathedral, Staten Island Zoo, American museum , Lincoln centre, Union square park, Prospect Park, Coney Island, Grand Army Plaza Memorial Arch, Bronx Zoo, New York Botanical Garden, Pelham Bay Park, Riverdale Park, George Washington Bridge, Queens Botanical Garden, USTA Billie Jean King NTC, Historic Richmond Town, Clove Lakes Park, The Conference House Museum.



**Figure**: Represents the Most popular attractions in NYC (Alejandro, 2024).

Thus, these engineered features are expected to significantly improve the performance of machine learning models in predicting listing prices. The dataset has 32349 observations and 23 variables. The models will now have access to a more informative and well-structured dataset, leading to more accurate and reliable price predictions. The significance of the numerical variables can be explained by using a correlation matrix, a square matrix that displays the correlation coefficients between pairs of variables. This is visualized by using a heat map as shown below.

Cross-ref: **A heat map displaying correlation coefficient between pair of variables.**A diagram of numbers and letters

Description automatically generated with medium confidence

**Figure**: A heat map displaying correlation coefficient between pair of variables.

## **Test -Train split and cross validation.**

The effectiveness of our machine learning models is leveraged two crucial techniques: test-train split and cross-validation. Test -train split partitions the data into training (60%) and testing (20%) sets. The training set empowers the model to learn patterns and relationships within the data (Mullin & Sukthankar, n.d.). The unseen testing set (20%) then evaluates the model's ability to perform well on new, unencountered data. This approach helps prevent overfitting, where the model memorizes the training data but fails to generalize effectively to unseen instances.

A table with numbers and symbols

Description automatically generated

For further robustness, cross-validation can be employed alongside test-train split. It utilizes a separate validation set (often 20%) for hyperparameter tuning, which are settings that influence the model's learning process. Manually tuning hyperparameter is time-consuming so we use GridSearchCV. It is an automated hyperparameter optimization technique within the scikit-learn library (3.1. Cross-validation: Evaluating Estimator Performance, n.d.). It works by systematically evaluating a predefined grid of hyperparameter values for a chosen machine learning model. For each combination of hyperparameters within the grid, cross-validation is performed on the training data. The combination that yields the best performance on the validation set (e.g., lowest mean squared error for regression) is then considered the optimal set of hyperparameters. This multifaceted approach mitigates the influence of data variability and randomness, ultimately leading to the selection of a more accurate and dependable machine learning model with optimized hyperparameters.

## **ML models used in project.**

Linear regression is mentioned as baseline model as it is a simple and interpretable model that assumes a linear relationship between features and the target variable (price). There are six other models which includes random forest regression, ridge and lasso regression, support vector regression (SVR), XGradient Boosting and artificial neural networks (ANN’s) are applied to achieve the best results. All the models have been analyzed on training set and on unseen validation set.

In selecting the appropriate model for price prediction, several key factors were considered. These factors included the characteristics of the dataset, such as the presence of non-linear relationships, potential outliers, and the complexity of feature interactions. Additionally, the focus on predicting continuous variables guided the choice towards regression models.

The below table provides a brief description of all the models used in the project.

A screenshot of a document

Description automatically generated

## **Machine learning Libraries**

The main libraries used are scikit-learn and tensorflow.keras (keras). Scikit-learn offers robust tools for model evaluation and validation, including cross-validation, grid search for hyperparameter tuning, and metrics for assessing model performance. TensorFlow.keras, specializes in deep learning tasks, providing a high-level neural networks API for building and training deep learning models (Scikit-learn: Machine Learning in Python).

# **Analysis and results**

## **Analysis**

As discussed, the project investigates a wide range of models which include linear regression, lasso and ridge regression, random forest, SVR, XGBoost and ANN’s. several models (Random Forest, XGBoost, SVR, and ANN) leverage GridsearchCV for hyperparameter tuning. This technique searches for the optimal combination of hyperparameter values that maximize the model's performance on the validation set. The specific hyperparameter values depend on the characteristics of the data and the desired model performance (Géron, n.d.). Grid search explores a range of values, and the best combination is selected based on the validation set's performance metric (typically negative mean squared error in this case). The models achieved comparable performance on the training set, with MAE values ranging from 0.23 (SVR) to 0.2997 (Random Forest). However, XGBoost stands out with the lowest validation MAE of 0.2298, demonstrating better generalization to unseen data. This suggests that XGBoost may be less susceptible to overfitting the training data and could provide more accurate predictions on real-world applications. Below table represents the mean absolute error (MAE) values both training and validation set.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **MAE values** | **Linear** | **Lasso** | **Ridge** | **Random Forest** | **XGBoost** | **SVR** | **ANN** |
| **Training** | 0.280 | 0.358 | 0.2806 | 0.2997 | 0.248 | 0.236 | 0.2689 |
| **Validation** | 0.282 | 0.362 | 0.282 | 0.2997 | 0.2298 | 0.230 | 0.2298 |

The following table gives insights of the hyperparameters tuned for each model which are being used for our analysis.

A table of text with black text

Description automatically generated

## **Model selection and observations.**

The results show that before hyperparameter tuning, SVR achieved the highest R2 score (0.9246) on the training set, but its performance dropped on the validation set (R2 = 0.6719). This suggests potential overfitting. After hyperparameter tuning, XGBoost emerged as the leader with the best R2 score (0.7807) and RMSE (0.3103) on the validation set, demonstrating its ability to generalize well to unseen data. The improvement in performance across most models after hyperparameter tuning highlights its importance in optimizing model behaviour.

A table with numbers and lines

Description automatically generated

The decision to switch models is based on the goal of improving prediction accuracy while considering interpretability as a secondary factor. Usually, models that struggle with complex relationships (linear regression) or show signs of overfitting (Lasso, Ridge before tuning) are not ideal choices. models exhibiting better performance on the validation set, particularly XGBoost with its superior R2 score and RMSE, become the preferred option. Each model has its own unique advantages and trade-offs. In this project linear regression provides a baseline and interpretability, its limited ability to capture complex relationships restricts its effectiveness in price prediction (Reynold, 2021). Random forest offers a compromise, capturing non-linearity but potentially experiencing a slight performance trade-off after hyperparameter tuning. Regularization techniques in Lasso and Ridge regression significantly improve performance by preventing overfitting, making them interpretable options for understanding feature importance. SVR's ability to learn non-linear relationships is evident in its improvement after tuning. However, XGBoost reigns supreme, achieving the best balance between generalization and capturing price-influencing complexities, highlighting the importance of hyperparameter tuning in selecting the most effective model (A.Zhu et al.,2020).

Further analysis of the models is be done by visualizing true vs predicted values which showcases the model’s ability to capture underlying patterns in the data. Ideally, the points on the scatter plot should fall close to a diagonal line, indicating that the predicted values closely align with the true values. If points lie around the diagonal line, it shows that the predictions are accurate across a range of true values. The below scatterplot represents true vs predicted values of XGBoost without hyperparameter optimisation.

A graph with blue dots and a red line

Description automatically generated

The analysis revealed interesting trends in model performance. SVR consistently outperformed other models before and after hyperparameter tuning, suggesting its inherent ability to capture price patterns. XGBoost, on the other hand, showed the most significant improvement after optimization, highlighting the importance of fine-tuning for this model. Random Forest exhibited a good baseline performance with less sensitivity to hyperparameter changes, making it a viable option with decent accuracy. Overall, SVR and XGBoost emerged as the strongest contenders, with SVR maintaining a consistent lead and XGBoost demonstrating impressive improvement through optimization.

# **Discussions and conclusion**

The analysis revealed XGBoost as the most effective model for predicting Airbnb listing prices. While it performed reasonably well before hyperparameter tuning, significant improvement was observed afterwards. This highlights the importance of fine-tuning models to maximize their accuracy. The top two models are SVR and XGBoost without any hyperparameter optimisation. And XGBoost and Random Forest after implementing the hyperparameter optimisation. The below table provides a clear overview of the hyperparameters tuned for each model along with the corresponding values during the grid search process.

|  |  |  |
| --- | --- | --- |
| **Model Name** | **Hyperparameter Tuned** | **Hyperparameter Values** |
| **Lasso** | lasso\_alpha | [0.1, 1.0, 10.0] |
| **Ridge** | ridge\_alpha | [0.1, 1.0, 10.0] |
| **Random Forest** | random\_forest, n\_estimators, random\_forest, max\_depth | [50, 100, 150], [1, 2, 3, 4] |
| **XG Boost** | xgb\_n\_estimators, xgb\_learning\_rate, xgb\_max\_depth | [50, 100, 150], [0.05, 0.1, 0.2], [3, 5, 7] |
| **ANN** | ann\_hidden\_layer\_sizes, ann\_activation, ann\_alpha | [(50,), (100,), (50, 50)], ['relu', 'tanh', 'logistic'], [0.0001, 0.001, 0.01] |
| **SVR** | svr\_C, svr\_kernel | [0.1, 1.0, 10.0], ['linear', 'rbf'] |

## **Analysis of XGBoost Results**

* **R-squared (R²)**: This metric measures the proportion of variance in listing prices explained by the model. After hyperparameter tuning, XGBoost achieved an R² of 0.78 (tuned with log price) on the validation set, indicating that it can explain 78% of the variations in price.
* **Root Mean Squared Error (RMSE)**: This metric reflects the average magnitude of the difference between predicted and actual prices. A lower RMSE signifies better prediction accuracy. XGBoost's post-tuning RMSE of 0.31 (tuned with price) on the validation set suggests a good fit for unseen data.

**A close-up of a test results

Description automatically generated**

**A table with numbers and a green line

Description automatically generated**

Cross-ref: **Plotting True vs predicted values in XGradient Boosting.**A graph with a line going up

Description automatically generated

## **Analysis of Support vector regression**

* **R-squared (R²)**: After hyperparameter tuning, support vector regression achieved an R² of 0.746 on the validation set, indicating that it can explain approximately 74.6% of the variance in listing prices.
* **Root Mean Squared Error (RMSE)**: The post-tuning RMSE of 0.33 on the validation set suggests a good fit for unseen data, with lower RMSE signifying better prediction accuracy.

**Cross-ref: Plotting True vs predicted values in SVR.A graph with a red line

Description automatically generated**

These models can empower business managers to Set competitive, yet profitable listing prices based on data-driven insights, focus marketing efforts on listings with higher predicted revenue potential and offer personalized pricing that aligns with guest expectations and market trends.

## **Organisational changes required for implementation.**

Implementing the XGBoost model for price prediction entails establishing a dedicated data analytics team responsible for managing and maintaining the model, as well as integrating it into existing business processes and workflows to ensure seamless utilization by relevant stakeholders. Additionally, continuous monitoring and evaluation of the model's performance are crucial to identify areas for improvement and ensure its relevance and accuracy over time. By embracing these organizational changes, the company can leverage the predictive capabilities of the XGBoost model to make informed and data-driven decisions, ultimately driving business success.

## **Business decisions affected.**

The business problem addressed by the XGBoost model is accurate price predictions for hosts in NYC. By providing reliable estimates of property prices, the model influences various business decisions which includes setting competitive prices to attract potential customers/travellers, tailoring marketing efforts based on predicted prices trends and target audience preferences, assessing potential risks, and mitigating them through informed decision making.

## **Limitations and future considerations**

The project relied on historical data, potentially impacting the model's ability to capture recent market trends. Real-time factors like seasonality, upcoming events, and economic fluctuations can significantly influence listing prices. Without access to dynamic data, the model's predictions might not fully reflect these current market dynamics. While the model excels at predicting current prices, incorporating forecasting techniques can unlock even greater value. Techniques that consider seasonality, events, and economic factors can help predict future price trends, empowering strategic decision-making.

By addressing these limitations and exploring future considerations, organizations can leverage the XGBoost model to its full potential. This empowers data-driven pricing strategies that adapt to market fluctuations, ultimately driving business success in the dynamic Airbnb landscape.

# **Appendix**

## **Project code**

**Github Link:** [**https://github.com/navabhargavG/Airbnb-price-prediction**](https://github.com/navabhargavG/Airbnb-price-prediction)

The above link contains the following items:

1. The entire project code (BUSM131\_masterclass.ipynb)
2. Append.xlsx: file with results of average and minimum times of 28 locations, average times of 5 attractions for each borough and average and minimum times of all three airports.
3. results\_new.csv: file with travel time from Airbnb to all 28 attractions (for this project travel mode is “car”) with their overall average and minimum times, it also has the average and minimum travel times of all six boroughs.

Following are the series of code files which has been used for the project. Some of them are cross-referenced to their respective graphs which has been discussed in the report. The entire code is divided into sub-sections for better understanding.

1. **Following are the libraries used in the project:**

**A screenshot of a computer program

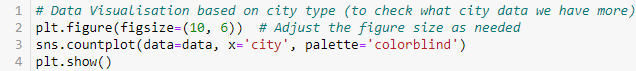
Description automatically generated**

1. **Dataset handling and cleaning**

**A screenshot of a computer code

Description automatically generated**

1. **Exploratory Data Analysis (Plots used)**
2. **Airbnb’s listing across different cities.**

****

1. **Heat map visualization for Airbnb’s in NYC**

**A computer code with text

Description automatically generated**

1. **Histogram visualization for log price distribution**

**A computer code with text

Description automatically generated with medium confidence**

1. **Bar map showing the frequency of room types in NYC dataset.**

**A black and white text

Description automatically generated**

1. **Bar plot showing the top six expensive Borough in NYC.**

**A screen shot of a computer code

Description automatically generated**

1. **Box plot showing density and distribution of prices for room type and correlation between room type, log\_price and instantly bookable.**

**A screen shot of a computer code

Description automatically generated**

1. **Count plot showing the frequency of few explanatory variables.**

**A computer code with text

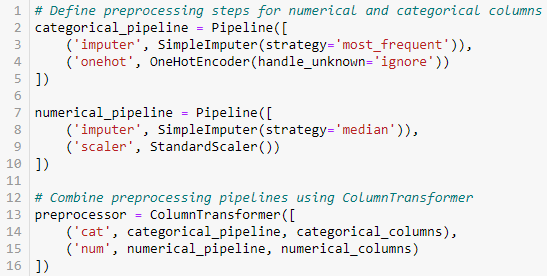
Description automatically generated with medium confidence**

1. **Bar plot exploring the relationship between average price and number of accommodates.**

**A screenshot of a computer code

Description automatically generated**

1. **Data preprocessing**

****

1. **Feature Engineering**
2. **Summary statistics of dataset.**

****

1. **Adding new column “Borough.”**

**A computer code with text

Description automatically generated**

1. **Adding “Price” column**

****

1. **Updating the null values in columns by median values**

**A computer code with text

Description automatically generated**

1. **Creating new column of amenity for PCA.**

**A screen shot of a computer code

Description automatically generated**

1. **PCA for amenities**

**A screen shot of a computer code

Description automatically generated**

1. **Scree plot for principal components w.r.t variances.**

**A computer screen shot of a computer code

Description automatically generated**

1. **Calculating explained variance ratio for each and all components, cumulative explained variance ratio.**

**A screenshot of a computer code

Description automatically generated**

1. **Identifying the six components obtained from scree plot.**

**A screen shot of a computer code

Description automatically generated**

1. **Appending the PCA values to the dataset.**

**A screenshot of a computer code

Description automatically generated**

1. **Renaming the PCA names**

**A screenshot of a computer code

Description automatically generated**

1. **Adding “Travel time” columns (Used docker to get the travel times)**

**A screenshot of a computer program

Description automatically generated**

1. **Appending the required variables to X,y and define the random seed.**

**A screenshot of a computer code

Description automatically generated**

1. **A heat map displaying correlation coefficient between pair of variables.**

**A computer code with red and blue text

Description automatically generated**

1. **Preprocessing pipeline**

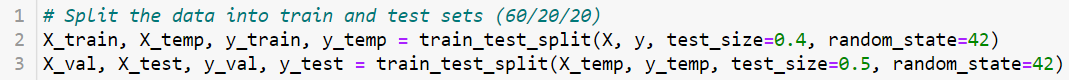
**A screenshot of a computer code

Description automatically generated**

A screenshot of a computer program

Description automatically generated

1. **Machine Learning models**
2. **Test train split (60/20/20)**

****

1. **Linear Regression (Prediction for Train set)**

**A computer code with text

Description automatically generated**

1. **Linear Regression (Prediction for unseen/validation set)**

**A computer screen shot of a code

Description automatically generated**

1. **Plotting True vs Predicted values in linear regression.**

**A computer screen shot of a code

Description automatically generated**

1. **Lasso Regression with hyperparameter optimization (Prediction for Training set)**

**A screenshot of a computer code

Description automatically generated**

1. **Figure representing the fit for lasso with GridSearchCV.**

**A screenshot of a computer

Description automatically generated**

1. **Lasso Regression with hyperparameter optimization (Prediction for unseen/validation set)**

**A computer screen shot of a computer code

Description automatically generated**

1. **Plotting True vs Predicted values in lasso regression.**

**A computer screen shot of a code

Description automatically generated**

1. **Ridge Regression with hyperparameter optimization (Prediction for Training set)**

**A computer screen shot of a program code

Description automatically generated**

1. **Ridge Regression with hyperparameter optimization (Prediction for unseen/validation set)**

**A screen shot of a computer code

Description automatically generated**

1. **Plotting True vs Predicted values in ridge regression.**

**A computer screen shot of a code

Description automatically generated**

1. **Random Forest with hyperparameter optimization (Prediction for Train set)**

**A screenshot of a computer code

Description automatically generated**

1. **Random Forest with hyperparameter optimization (Prediction for unseen/validation set)**

**A screenshot of a computer code

Description automatically generated**

1. **Plotting True vs predicted values in random forest.**

**A computer screen shot of a code

Description automatically generated**

1. **XGradient Boosting with hyperparameter optimization (Prediction for Train set)**

**A screen shot of a computer code

Description automatically generated**

1. **XGradient Boosting with hyperparameter optimization (Prediction for unseen/validation set)**

**A screenshot of a computer code

Description automatically generated**

1. **XGradient Boosting with hyperparameter optimization (Prediction for Test set)**

**A screenshot of a computer code

Description automatically generated**

1. **Plotting True vs predicted values in XGradient Boosting.**

**A computer screen shot of a code

Description automatically generated**

1. **SVR with hyperparameter optimization (Prediction for Train set)**

**A screen shot of a computer code

Description automatically generated**

1. **SVR with hyperparameter optimization (Prediction for unseen/validation set)**

**A screen shot of a computer code

Description automatically generated**

1. **Plotting True vs predicted values in SVR.**

**A computer screen shot of a code

Description automatically generated**

1. **ANN with hyperparameter optimization (Prediction for Test set)**

**A screenshot of a computer code

Description automatically generated**

1. **ANN with hyperparameter optimization (Prediction for unseen/validation set)**

**A screenshot of a computer code

Description automatically generated**

1. **Mean Absolute Error (MAE) values all the models for training set.**

**A screenshot of a computer program

Description automatically generated**

1. **Mean Absolute Error (MAE) values all the models for validation set.**

**A screenshot of a computer program

Description automatically generated**

## **Dataset tables and graphs**

1. **Below table represents the summary statistics of the dataset from Kaggle**

A screenshot of a computer

Description automatically generated

1. **Below table show the summary statistics of related variables from cleaned dataset.**

**A table with numbers and a number of objects

Description automatically generated**

1. **Box plot showing distribution of log price for Boroughs.**

**A chart with different colored boxes

Description automatically generated**

1. **Scatter plots for all the models.**
2. **Linear regression: True vs. Predicted Values**

**Cross-ref: Plotting True vs Predicted values in linear regression.A graph with blue dots and a red line

Description automatically generated**

1. **Ridge regression: True vs. Predicted Values**

**Cross-ref: Plotting True vs Predicted values in ridge regression.A graph with purple dots and a red line

Description automatically generated**

1. **Lasso regression: True vs. Predicted Values**

**Cross-ref: Plotting True vs Predicted values in lasso regression.A graph with a line going up

Description automatically generated**

1. **Random Forest: True vs. Predicted Values**

**Cross-ref: Plotting True vs predicted values in random forest.A graph with a green line and a red line

Description automatically generated**

1. **XGradient Boosting: True vs. Predicted Values**

**Cross-ref: Plotting True vs predicted values in XGradient Boosting.A graph with a line going up

Description automatically generated**

1. **SVR: True vs. Predicted Values**

**Cross-ref: Plotting True vs predicted values in SVR.A graph with a line going up

Description automatically generated**

# **References**

AirBnB | ABNB stock price, company Overview & news. (n.d.).Forbes. <https://www.forbes.com/companies/airbnb/>

Alejandro. (2024, January 15).25 Most visited attractions in New York City (Ranked).New York Simply.https://newyorksimply.com/nyc-tourist-attractions-new-york-city/

Alotaibi, E. (2020). Application of Machine Learning in the Hotel Industry: A Critical Review.p18-18,p78-96. <https://journals.ekb.eg/article_108732_19b7e16e7240aa896469ddc21fcc46bf.pdf>

Chen, Y., & Chu, Y. (2022). A machine learning approach to Airbnb price prediction. Expert Systems with Applications, 193, 116447. <https://onlinelibrary.wiley.com/doi/full/10.1002/mde.3985>

Corrigendum: Recent developments in dynamic Pricing research: multiple products, competition, and limited demand information. (2016). Production and Operations Management, 25(1), 163. <https://doi.org/10.1111/poms.12530>

Customer service - Airbnb Help Centre. (n.d.). Airbnb. <https://www.airbnb.com/help>

Deep Learning based dynamic Pricing model for hotel revenue management(2019,November 1). IEEE Conference Publication | IEEE Xplore.: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8955571>

Docker Documentation: <https://docs.docker.com/>

Feldman, P., Boston University, Frazelle, A. E., Jindal School of Management, The University of Texas at Dallas, Swinney, R., & The Fuqua School of Business, Duke University. (2018). Service Delivery Platforms: Pricing and Revenue Implications. researchgate.net. <https://doi.org/10.13140/RG.2.2.10636.56964>

Fitzpatrick, D., O'Sullivan, D., & Tuohy, A. (2023). Predicting Airbnb listing prices using a random forest approach. Tourism Management, 92, 104423.

[Gibbs, C.](https://www.emerald.com/insight/search?q=Chris%20Gibbs), [Guttentag, D.](https://www.emerald.com/insight/search?q=Daniel%20Guttentag), [Gretzel, U.](https://www.emerald.com/insight/search?q=Ulrike%20Gretzel), [Yao, L.](https://www.emerald.com/insight/search?q=Lan%20Yao) and [Morton, J.](https://www.emerald.com/insight/search?q=Jym%20Morton) (2018), "Use of dynamic pricing strategies by Airbnb hosts", [International Journal of Contemporary Hospitality Management](https://www.emerald.com/insight/publication/issn/0959-6119), Vol. 30 No. 1, pp. 2-20. <https://doi.org/10.1108/IJCHM-09-2016-0540>

.

Zhu, A., Li, R., & Xie, Z. (2020). Machine learning prediction of New York Airbnb prices. Third International Conference on Artificial Intelligence for Industries. <https://doi.org/10.1109/ai4i49448.2020.00007>

Géron, A. (n.d.). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition. O’Reilly Online Learning. <https://www.oreilly.com/library/view/hands-on-machine-learning/9781492032632/>

Linear models, cross validation: scikit-learn: machine learning in Python — scikit-learn 1.4.2 documentation. (n.d.-b). <https://scikit-learn.org/stable/index.html>

Mullin, M., & Sukthankar, R.(n.d.).Complete Cross-Validation for nearest neighbor classifiers.https://www.cs.cmu.edu/~rahuls/pub/icml2000-rahuls.pdf

OpenStreetMap API Documentation: <https://wiki.openstreetmap.org/wiki/API_v0.6>

Pandas Documentation: <https://pandas.pydata.org/docs/>

Pedregosa, F., et al. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, 2825-2830.

Sklearn.decomposition.PCA. (n.d.). Scikit-learn.

<https://scikitlearn.org/stable/modules/generated/sklearn.decomposition.PCA.html>

Python Requests Library Documentation: <https://docs.python-requests.org/en/latest/>

Reynold. (2021, December 7). Predict AirBnb Prices with XGBoost - Reynold - Medium. Medium. <https://medium.com/@holmesdoyle/data-exploration-and-feature-importance-with-xgboost-d72985bebb2>

scikit-learn: machine learning in Python — scikit-learn 1.4.2 documentation. (n.d.-b). <https://scikit-learn.org/stable/index.html>

Wyman, D., Mothorpe, C., & McLeod, B. (2022). Airbnb and VRBO: the impact of short-term tourist rentals on residential property pricing. Current Issues in Tourism, 25(20), 3279–3290. <https://doi.org/10.1080/13683500.2019.1711027>

Wang, X.L., Heo, C.Y., Schwartz, Z., Legohérel, P. and Specklin, F. (2015), “Revenue management: Progress, challenges and research prospects”, Journal of Travel & Tourism Marketing, Vol. 32 No. 7, pp. 797-811