New York City Airbnb Open Data Housing Price Prediction

SANJAY RAO PAWAR SHIVASHANKAR RAO

Submitted for the Degree of Master of Science in

Master of Science in Artificial Intelligence

Department of Computer Science

Royal Holloway University of London

Egham, Surrey TW20 0EX, UK

4 September 2023

Declaration

This report has been prepared on the basis of my own work. Where other published and unpublished source materials have been used, these have been acknowledged.

Word Count: 8947

Student Name: Sanjay Rao Pawar Shivashankar Rao

Date of Submission: 5th September 2023

Signature:

Abstract

The dissertation's main aim is to predict the price of Airbnb accommodations in New York city, United states of America (USA). Here We use two supervised regression machine learning techniques. They are ridge regression and gradient boosting regressor which predict numerical values as output which help us to analyse and develop the model for Airbnb housing price. Additionally, we use various metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared that give us both the model's performance and accuracy of predicting the output. By comparing both the models we can identify the predictive capabilities of ridge regression and gradient boosting and find the strengths and weaknesses in the context of Airbnb price prediction. Furthermore, we understand how well both can predict the housing prices and identify the important factors that affects the price of the house. In this entire project, we aim to understand the factors that influence the Airbnb pricing in new York city and provide useful information for both for the hosts and guests who want to book accommodations.

Table of Context

Section 1

Introduction

The main aim of this project is to predict the price of Airbnb accommodations in New York City, USA. We will use two supervised regression machine learning techniques, namely ridge regression and gradient boosting regressor. These techniques predict numerical values as output, which will help us to analyse and develop a model for Airbnb housing price. Secondly we will be using various metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE) and R-squared to evaluate performance and accuracy of the our models. By comparing both our models, we can find the capabilities of both our models (ridge regression and gradient boosting) and their strengths and weaknesses in predicting Airbnb prices. Thirdly we will understand how well both our models can predict the values and identify the important values that affect the price of the house in the dataset

Airbnb is a widely used by mobile users for people to rent the homes to the people who travel and stay in. The price depends the location, size, additional facility and duration.

In our project, we're going to use a bunch of data about Airbnb listings in New York City to teach two computer models how to predict prices. The first model is like a math equation called "ridge regression," and the second one is a bit more complex and is called a "gradient boosting regressor." After our models are readily trained, we find good they scores like MSE, RMSE, and R-squared. Lastly, we'll compare our two models and find out which things affect the price of Airbnb rentals in New York City.

We think this project will really help people who use Airbnb. Hosts will know better how to set prices, and guests will have a clearer idea of what to expect when booking. Plus, we'll learn a lot about what makes prices go up or down for Airbnb places.

Section 2

Project Plan

1. The aim of this project is to apply multiple machine learning techniques to predict housing prices using the Airbnb housing dataset.

2. Timeline:

2A. Week 1-2 (June 1st - June 15th): Dataset exploration, data preprocessing, and proof of concept implementation.

1.During these two weeks, the focus will be on understanding the Boston housing dataset, exploring its features, and analysing its structure.

2.Data preprocessing tasks will be performed, which may include handling missing values, data cleaning, and transforming variables if necessary.

3.A proof of concept implementation will be carried out, where initial machine learning models may be trained and evaluated on the dataset to assess their performance.

2B. Week 3-4 (June 16th - July 30th): Implement and compare multiple machine learning techniques.

1.In this period, various machine learning techniques will be implemented to predict housing prices using the Boston housing dataset.

2. Different algorithms, such as linear regression, decision trees, random forests, and support vector machines, may be applied and evaluated.

3.The performance of each technique will be assessed based on relevant evaluation metrics, such as mean squared error or accuracy.

3B. Week 5-6 (July 1st - July 15th): Implement online mode prediction and analyze the effect of time on housing prices.

1. During these two weeks, an online mode prediction system will be implemented, which can provide real-time predictions for housing prices based on the trained models.

2.The focus will also be on analysing the effect of time or temporal factors on housing prices. This may involve investigating seasonality, trends, or other time-related patterns in the dataset.

4B. Week 7-8 (July 16th - July 30th): Compile results and visualize findings

1. The result that is obtained from the previous step will be analysed.

2. Visualizations, such as graphs, may be created help us to find and provide meaningful insights into the performance of different machine learning techniques.

3. The visualization of the results can help us understand the relationships between variables and identifying any notable patterns or trends.

5B. Week 9-11 (August 1st - August 15th): Write the final report

1. During these time I document the final report for the project.

2. Our report will include a detailed description of the project, the methodology that we used , the results obtained, and the conclusions that we found.

3. The report includes the limitations and ethical concerns of our project

Section 3

Literature review

1) The study employs a hybrid regression model that amalgamates various regression techniques to offer a more accurate and reliable prediction of housing prices. By fusing multiple methodologies, the authors aim to overcome the limitations of using a single regression model, thereby enhancing the predictive accuracy and reliability of the results. This work serves as a significant contribution to the field of real estate economics, offering a nuanced and technologically advanced method for price prediction that could be invaluable for both investors and policymakers.

2) This paper[2] deals with the the intricate world of real estate economics with a particular focus on cities like Bengaluru. Utilizing an array of regression techniques, the study aims to predict housing prices with a level of accuracy that could be groundbreaking for the industry. Unlike traditional approaches, this research employs machine learning algorithms to enhance the predictive power of their models. The authors meticulously evaluate the performance of various regression methods, thereby offering a comprehensive understanding of which techniques are most effective for this specific market.

3) The author [3] explains the role of machine learning in the realm of smart buildings. Specifically, the study zeroes in on the forecasting of indoor temperatures, a critical aspect for energy management and occupant comfort. The authors employ a range of machine learning algorithms to assess their efficacy in predicting indoor climate conditions. By juxtaposing various algorithms, the paper provides an in-depth comparative analysis that serves as a guide for choosing the most effective predictive model. This work is not only academically rigorous but also holds immense practical value for architects, engineers, and smart building managers who aim to optimize energy usage while maintaining optimal indoor conditions.

4) The XGBoost [4] is explained detailly explained in this paper, The authors meticulously detail the mathematical underpinnings of the algorithm, demonstrating its superiority in handling large datasets and complex models. By incorporating regularization techniques and optimizing computational resources, XGBoost stands as a robust tool for predictive modeling across various domains. This work has had a profound impact on both academic research and practical applications, offering a robust and scalable solution for complex predictive tasks.

7) The author[7] explains the complexities of identifying anomalies or outliers that can significantly skew the accuracy of effort estimation models. By introducing a method specifically designed to detect these outliers, the authors aim to enhance the reliability and precision of software effort estimations. This is particularly crucial for project managers and developers who rely on these models for budgeting, resource allocation,

and project timelines. The research is both academically rigorous and practically relevant, offering a new lens through which the software industry can approach the challenge of effort estimation.

10) The authors [10] delve into the intricacies of ridge regression techniques, a staple in statistical analysis. Through a meticulous examination of its mathematical underpinnings and real-world implementations, they present an intricate comprehension that challenges prevailing assumptions and encourages a re-examination of established norms. This research functions as a dual-purpose resource, offering both a note of caution and a navigational aid for statisticians, data scientists, and researchers.

Section 3

Research Methodology

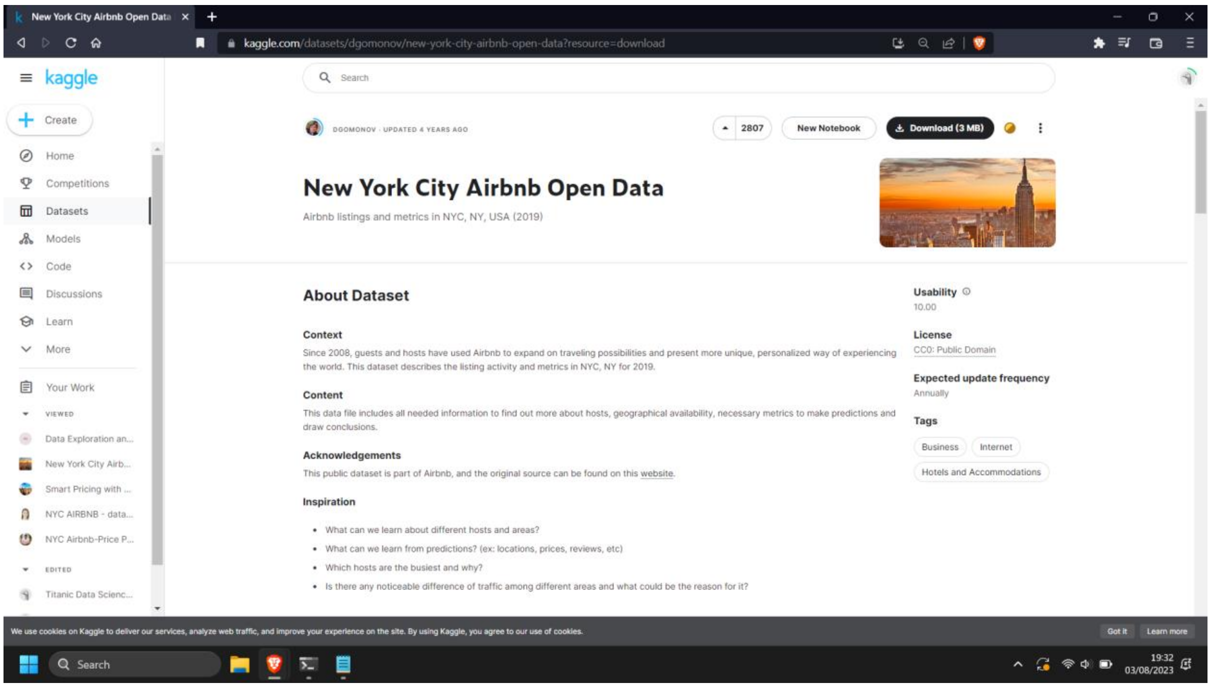
3.1 Data Source and Preprocessing

We highlighted the significance of Airbnb's accommodation on travel and the importance of understanding pricing in New York City. Our use of machine learning and available data that we have given us valuable insights that benefit the Airbnb community for improvement. As we move into the methodology and analysis, we uncover new knowledge and make meaningful discoveries.

1. Data Collection - We download the open dataset from Kaggle which is a popular online platform for data science and machine learning datasets and competitions, which contains all the information on accommodation listings in New York City. The dataset can be downloaded in csv (comma separated values) format.

Below is the link and the website image from where I downloaded the dataset from?

Link-https://www.kaggle.com/datasets/dgomonov/new-york-city-airbnb-open-data?resource=download



New York City Airbnb Open Data

Airbnb listings and metrics in NYC, NY, USA (2019)

About Dataset

Context

Since 2008, guests and hosts have used Airbnb to expand on traveling possibilities and present more unique, personalized way of experiencing the world. This dataset describes the listing activity and metrics in NYC, NY for 2019.

Content

This data file includes all needed information to find out more about hosts, geographical availability, necessary metrics to make predictions and draw conclusions.

Acknowledgements

This public dataset is part of Airbnb, and the original source can be found on this website.

Inspiration

- What can we learn about different hosts and areas?
- What can we learn from predictions? (ex: locations, prices, reviews, etc)
- Which hosts are the busiest and why?
- Is there any noticeable difference of traffic among different areas and what could be the reason for it?

Usability
10.00

License
CC0: Public Domain

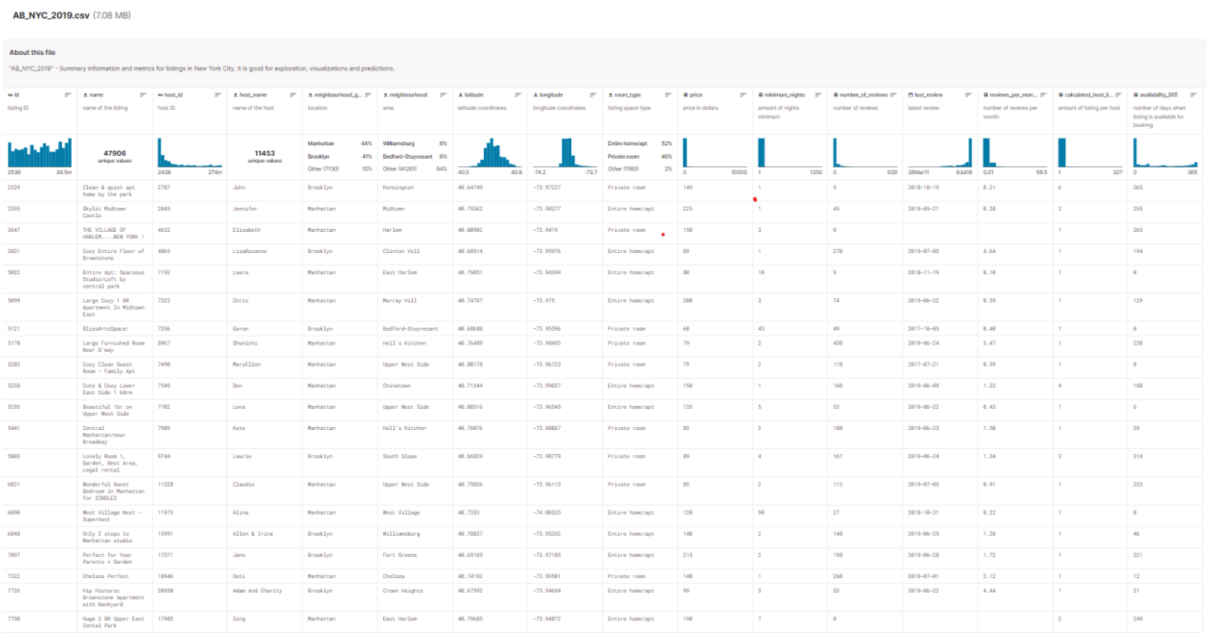
Expected update frequency
Annually

Tags
Business · Internet · Hotels and Accommodations

We use cookies on Kaggle to deliver our services, analyze web traffic, and improve your experience on the site. By using Kaggle, you agree to our use of cookies.

2. Data Description - For the New York City Airbnb Open Data Housing Price Prediction project, we have collected a comprehensive dataset from the Airbnb Open Data project. This dataset contains valuable information about different short term accommodation listings in New York City. It contains details like property features, prices, minimum nights required for booking, room type, availability etc throughout the particular year. We will use this data to explore and predict housing prices using machine learning techniques.

Below is the Kaggle airbnb dataset description image that shows all the column values in the dataset and its distribution.


AB_NYC_2019.csv (7.08 MB)

About this file

"AB_NYC_2019" - Summary information and metrics for listings in New York City. It is good for exploration, visualizations and predictions.

id	name	host_id	host_name	neighbourhood_g...	neighbourhood	latitude	longitude	room_type	price	minimum_nights	number_of_reviews	last_review	reviews_per_mon...	calculated_host_li...	availability_365
listing ID	name of the listing	host ID	name of the host	location	area	latitude coordinates	longitude coordinates	listing space type	price in dollars	amount of nights minimum	number of reviews	latest review	number of reviews per month	amount of listing per host	number of days when listing is available for booking
2539 – 36.5m	47906 unique values	2438 – 274m	11453 unique values	Manhattan 44% Brooklyn 41% Other (7130) 15%	Williamsburg 8% Bedford-Stuyvesant 8% Other (41261) 84%	40.5 – 40.9	-74.2 – -73.7	Entire home/apt 52% Private room 46% Other (1160) 2%	0 – 10000	1 – 1250	0 – 629	28Mar11 – 8Jul19	0.01 – 58.5	1 – 327	0 – 365
2539	Clean & quiet apt home by the park	2787	John	Brooklyn	Kensington	40.64749	-73.97237	Private room	149	1	9	2018-10-19	0.21	6	365
2595	Skylit Midtown Castle	2845	Jennifer	Manhattan	Midtown	40.75362	-73.98377	Entire home/apt	225	1	45	2019-05-21	0.38	2	355
3647	THE VILLAGE OF HARLEM....NEW YORK !	4632	Elisabeth	Manhattan	Harlem	40.80902	-73.9419	Private room	150	3	0			1	365
3831	Cozy Entire Floor of Brownstone	4869	LisaRoxanne	Brooklyn	Clinton Hill	40.68514	-73.95976	Entire home/apt	89	1	270	2019-07-05	4.64	1	194
5022	Entire Apt: Spacious Studio/Loft by central park	7192	Laura	Manhattan	East Harlem	40.79851	-73.94399	Entire home/apt	80	10	9	2018-11-19	0.10	1	0
5099	Large Cozy 1 BR Apartment In Midtown East	7322	Chris	Manhattan	Murray Hill	40.74767	-73.975	Entire home/apt	200	3	74	2019-06-22	0.59	1	129
5121	BlissArtsSpace!	7356	Garon	Brooklyn	Bedford-Stuyvesant	40.68688	-73.95596	Private room	60	45	49	2017-10-05	0.40	1	0
5178	Large Furnished Room Near B'way	8967	Shunichi	Manhattan	Hell's Kitchen	40.76489	-73.98493	Private room	79	2	430	2019-06-24	3.47	1	220
5203	Cozy Clean Guest Room - Family Apt	7490	MaryEllen	Manhattan	Upper West Side	40.80178	-73.96723	Private room	79	2	118	2017-07-21	0.99	1	0
5238	Cute & Cozy Lower East Side 1 bdrm	7549	Ben	Manhattan	Chinatown	40.71344	-73.99037	Entire home/apt	150	1	160	2019-06-09	1.33	4	188
5295	Beautiful 1br on Upper West Side	7702	Lena	Manhattan	Upper West Side	40.80316	-73.96545	Entire home/apt	135	5	53	2019-06-22	0.43	1	6
5441	Central Manhattan/near Broadway	7989	Kate	Manhattan	Hell's Kitchen	40.76076	-73.98867	Private room	85	2	188	2019-06-23	1.50	1	39
5803	Lovely Room 1, Garden, Best Area, Legal rental	9744	Laurie	Brooklyn	South Slope	40.66829	-73.98779	Private room	89	4	167	2019-06-24	1.34	3	314
6021	Wonderful Guest Bedroom in Manhattan for SINGLES	11528	Claudio	Manhattan	Upper West Side	40.79826	-73.96113	Private room	85	2	113	2019-07-05	0.91	1	333
6090	West Village Nest - Superhost	11975	Alina	Manhattan	West Village	40.7353	-74.00525	Entire home/apt	120	90	27	2018-10-31	0.22	1	0
6848	Only 2 stops to Manhattan studio	15991	Allen & Irina	Brooklyn	Williamsburg	40.70837	-73.95352	Entire home/apt	140	2	148	2019-06-29	1.20	1	46
7097	Perfect for Your Parents + Garden	17571	Jane	Brooklyn	Fort Greene	40.69169	-73.97185	Entire home/apt	215	2	198	2019-06-28	1.72	1	321
7322	Chelsea Perfect	18946	Doti	Manhattan	Chelsea	40.74192	-73.99501	Private room	140	1	260	2019-07-01	2.12	1	12
7726	Hip Historic Brownstone Apartment with Backyard	20950	Adam And Charity	Brooklyn	Crown Heights	40.67592	-73.94694	Entire home/apt	99	3	53	2019-06-22	4.44	1	21
7750	Huge 2 BR Upper East Cental Park	17985	Sing	Manhattan	East Harlem	40.79685	-73.94872	Entire home/apt	190	7	0			2	249

3. Data Cleaning:

Data Cleaning is a critical step in our project. It is the process of identifying and correcting errors and removing the inconsistencies, and inaccuracies in a dataset to ensure it is complete and free from irregular values and ready for our analysis.

We first address the duplicates and missing data by either imputing values or removing the columns or rows. We do it to prevent bias in our analysis. By handling missing data, we avoid errors and obtain accurate insights about the dataset.

In our dataset if a column has more null values or nan, we simply remove the entire column if it is not that important for our analysis or we replace the missing values with mean median or mode.

Later, we perform data validation where we check to identify and manage any inconsistent or inaccurate data points and eliminate the outliers in the dataset, which are extreme values that might affect our result analysis as few algorithms are less sensitive to outliers.

We also ensure that the data is properly formatted and standardized. We use Pandas factorize method to encode categorical variables in the Data Frame. Categorical variables are variables that have distinct groups, and they need to be converted into numerical format before using them in the machine learning algorithms, as most algorithms work only on numerical inputs.

3.2 Background and Context: Airbnb and Its Success Factors

Image credit -https://en.wikipedia.org/wiki/Airbnb

Airbnb stands for "Air Bed and Breakfast." which is a San Francisco-based company which provides an online marketplace which is one of the popular websites that provide services that enable people to rent or lease short term lettings that includes vacation rentals or hotel rooms. Founded in 2008 by Brian Chesky, Joe Gebbia, and Nathan Blecharczyk. Airbnb has revolutionized the hospitality industry by providing a popular alternative for the traditional hotels. It offers unique accommodation options on the website that has gained massive popularity over the past years. Airbnb has become a well know application over the years for both hosts to book rooms in major cities like London etc.

Below are the key factors for the success of Airbnb that have contributed to its success:

1. Peer-to-peer (P2P) strategy: Airbnb created a platform that makes it easier to get both the customers and hotels together with excess space they could rent out for a few days (hosts). This P2P approach offered a wide range of unique and personalized accommodation options beyond the traditional hotels that we had during the past .

2. Local insights: Staying with Airbnb hosts often provides the travellers with the opportunity to interact with locals and get to know about the tourist spots and get travel tips from them.

3. Reviews: The Airbnb platform's reviewing system allows guests and hosts to leave feedback about their experiences with each other in the application. This helps other users to make these decisions of booking the room based on the reviews that are put up.

4. Friendly App Interface: Airbnb App's interface is very user-friendly, and it has a mobile app making it easier to book rooms and hotels over the phone. On the website it clearly shows the accommodation's image, price, location, type of accommodation, overall rating for the place from the previous guests who stayed there and the duration it's available from.

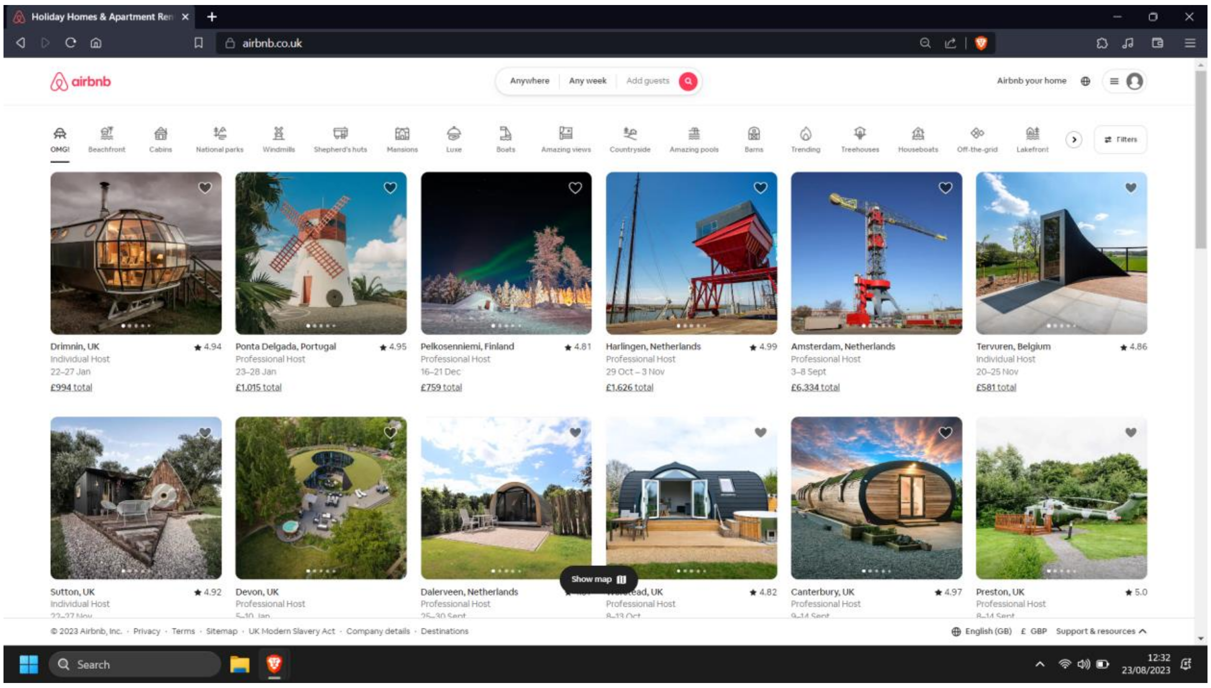


Fig 3.1 Snapshot of Airbnb

5. Cheap and Cost Savings: Airbnb often provides more affordable options compared to the traditional hotels. It is beneficial for the people who prefer budget accommodations and who want to book rooms cheaply.

6. Global Expansion: Airbnb is continually expanding its business and services to various countries and cities around the world. Make it globally available and easier to book accommodation with just one app globally.

3.3 Machine Learning Fundamentals

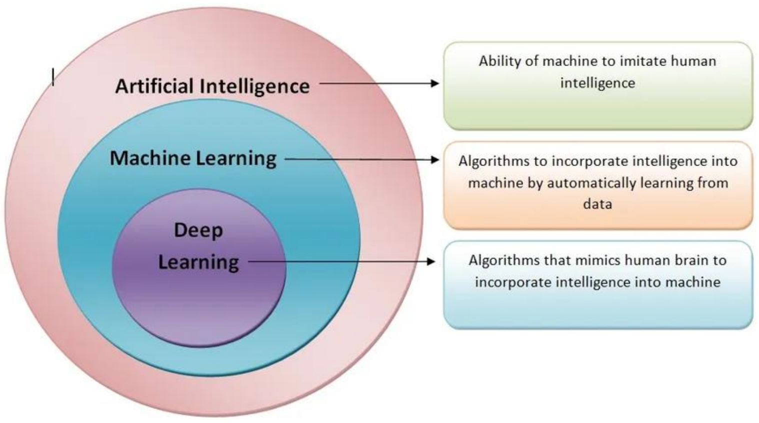

Artificial Intelligence
Machine Learning
Deep Learning
Ability of machine to imitate human intelligence
Algorithms to incorporate intelligence into machine by automatically learning from data
Algorithms that mimics human brain to incorporate intelligence into machine

Photo credit - Analytics Vidhya

https://www.analyticsvidhya.com

Fig 3.2 Machine Learning

Machine Learning is a subset of Artificial intelligence. The term "machine learning" was coined by Arthur Samuel in 1959. He defined it as the ability of computers to learn without being explicitly programmed. It means the ability of a computer or a machine to learn from the data that we provide and improve its performance on tasks without manually programming and coding. We use pattern identification, make predictions, and help us make decisions based on the data that is given to the computer. Machine learning is used in various applications: image and speech recognition, financial analysis, recommendation systems, fraud detection and self-driving cars. It will soon revolutionize industries by automating various complex tasks, optimizing processes.

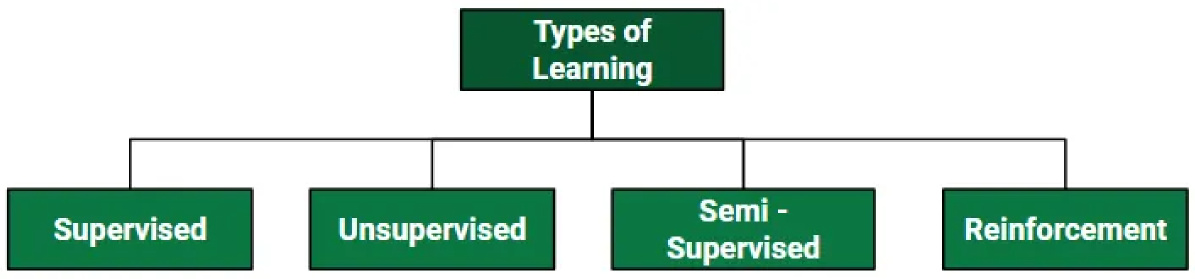


photo credit - greeksforgeeks.org

https://www.geeksforgeeks.org

Fig 3.3 Type of learning

Below are the types of machine learning

1) **Supervised learning** - This type of learning is trained on labeled data , where the input data is linked with the correct output value. Our Algorithm learns to map inputs to its output, making it easier for the algorithm to predict or classifications on new and unseen data.

There are 2 types in the supervised learning.

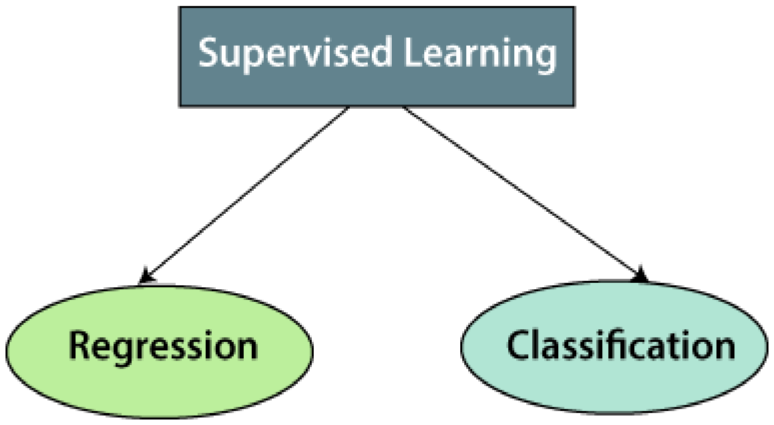


Photo Credit - Javatpoint

https://www.javatpoint.com/supervised-machine-learning

Fig 3.4 Supervised Learning

Regression - In this type we predict the numerical values (1,34.6,2313)

and the algorithm learns to find the relationship between input and output variables.

Example - We can find the price of home using the other details that we have like no off rooms, area, size etc.

Classification - In this type we predict the predefined classes (‘yes’ and ‘no’)

Example - we can use classification to find if our mail is spam or not spam.

2) **Unsupervised Learning** - In Unsupervised learning we work with unlabelled data, where our algorithm's work is to find patterns in the data and find relationships or correlation within our given data.

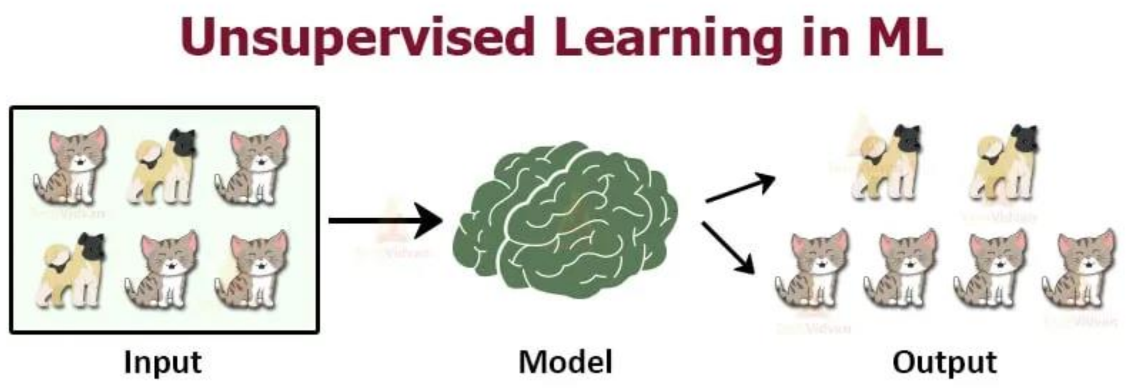


Photo Credit - https://techvidvan.com/tutorials/

Fig 3.5 Unsupervised Learning

There are two types in unsupervised learning,

i) Clustering - Here we are grouping similar data points together,

ii) Association - Here we find the relationships or patterns in our large datasets of transactions or events.

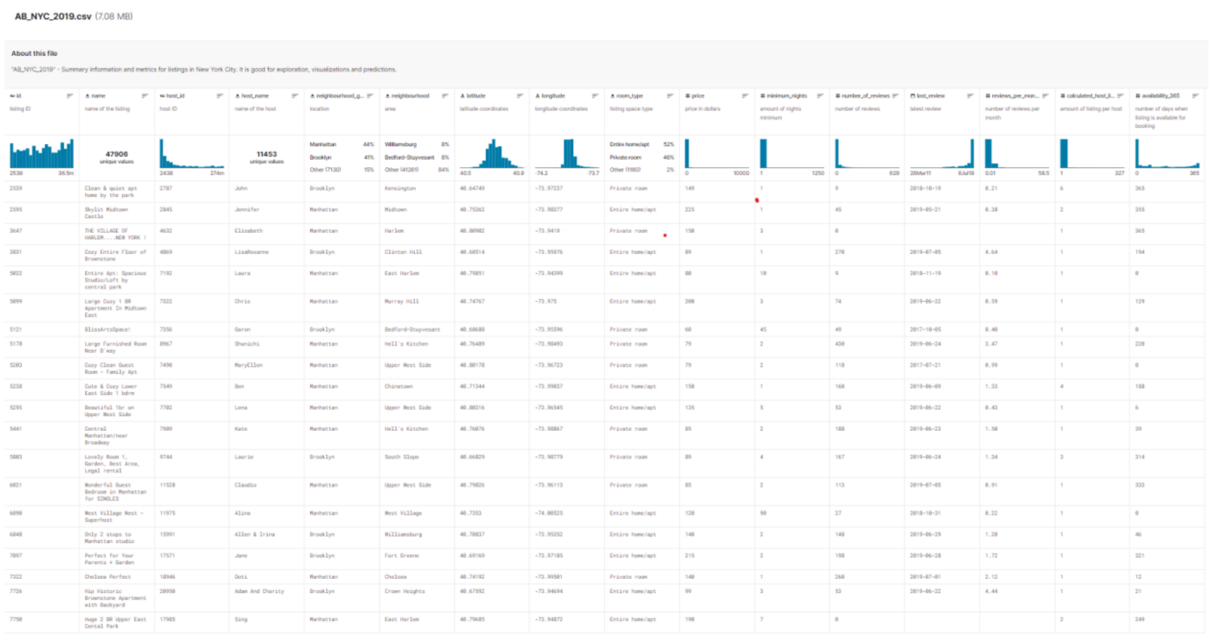
3) **Semi-Supervised Learning** - Semi-supervised learning works with the combination of both supervised and unsupervised learning. It uses a little bit of labeled data and a larger amount of unlabelled data to improve models learning and the accuracy of our model . This approach is useful when we are having a large amount of labelled data to work with.

4) **Reinforcement Learning** - In Reinforcement learning we train agents to interact with a given environment in order to learn the best possible action to take in different situations. Each time the agent gets feedback in the form of rewards or points based on the action it taking, and its goal is to learn maximize the points that it gets.

3.4 Data Description

For the New York City Airbnb Open Data Housing Price Prediction project, we have collected a comprehensive dataset from the Airbnb Open Data project. This dataset contains valuable information about different short term accommodation listings in New York City. It contains details like property features, prices, minimum nights required for booking, room type, availability etc throughout the particular year. We will use this data to explore and predict housing prices using machine learning techniques.

Below is the Kaggle airbnb dataset description image that shows all the column values in the dataset and its distribution.


AB_NYC_2019.csv (7.08 MB)

About this file

"AB_NYC_2019" - Summary information and metrics for listings in New York City. It is good for exploration, visualizations and predictions.

id	name	host_id	host_name	neighbourhood_g...	neighbourhood	latitude	longitude	room_type	price	minimum_nights	number_of_reviews	last_review	reviews_per_mon...	calculated_host_li...	availability_365
listing ID	name of the listing	host ID	name of the host	location	area	latitude coordinates	longitude coordinates	listing space type	price in dollars	amount of nights minimum	number of reviews	latest review	number of reviews per month	amount of listing per host	number of days when listing is available for booking
2539 – 36.5m	47906 unique values	2438 – 274m	11453 unique values	Manhattan 44% Brooklyn 41% Other (7130) 15%	Williamsburg 8% Bedford-Stuyvesant 8% Other (41261) 84%	40.5 – 40.9	-74.2 – -73.7	Entire home/apt 52% Private room 46% Other (1160) 2%	0 – 10000	1 – 1250	0 – 629	28Mar11 – 8Jul19	0.01 – 58.5	1 – 327	0 – 365
2539	Clean & quiet apt home by the park	2787	John	Brooklyn	Kensington	40.64749	-73.97237	Private room	149	1	9	2018-10-19	0.21	6	365
2595	Skylit Midtown Castle	2845	Jennifer	Manhattan	Midtown	40.75362	-73.98377	Entire home/apt	225	1	45	2019-05-21	0.38	2	355
3647	THE VILLAGE OF HARLEM....NEW YORK !	4632	Elisabeth	Manhattan	Harlem	40.80902	-73.9419	Private room	150	3	0			1	365
3831	Cozy Entire Floor of Brownstone	4869	LisaRoxanne	Brooklyn	Clinton Hill	40.68514	-73.95976	Entire home/apt	89	1	270	2019-07-05	4.64	1	194
5022	Entire Apt: Spacious Studio/Loft by central park	7192	Laura	Manhattan	East Harlem	40.79851	-73.94399	Entire home/apt	80	10	9	2018-11-19	0.10	1	0
5099	Large Cozy 1 BR Apartment In Midtown East	7322	Chris	Manhattan	Murray Hill	40.74767	-73.975	Entire home/apt	200	3	74	2019-06-22	0.59	1	129
5121	BlissArtsSpace!	7356	Garon	Brooklyn	Bedford-Stuyvesant	40.68688	-73.95596	Private room	60	45	49	2017-10-05	0.40	1	0
5178	Large Furnished Room Near B'way	8967	Shunichi	Manhattan	Hell's Kitchen	40.76489	-73.98493	Private room	79	2	430	2019-06-24	3.47	1	220
5203	Cozy Clean Guest Room - Family Apt	7490	MaryEllen	Manhattan	Upper West Side	40.80178	-73.96723	Private room	79	2	118	2017-07-21	0.99	1	0
5238	Cute & Cozy Lower East Side 1 bdrm	7549	Ben	Manhattan	Chinatown	40.71344	-73.99037	Entire home/apt	150	1	160	2019-06-09	1.33	4	188
5295	Beautiful 1br on Upper West Side	7702	Lena	Manhattan	Upper West Side	40.80316	-73.96545	Entire home/apt	135	5	53	2019-06-22	0.43	1	6
5441	Central Manhattan/near Broadway	7989	Kate	Manhattan	Hell's Kitchen	40.76076	-73.98867	Private room	85	2	188	2019-06-23	1.50	1	39
5803	Lovely Room 1, Garden, Best Area, Legal rental	9744	Laurie	Brooklyn	South Slope	40.66829	-73.98779	Private room	89	4	167	2019-06-24	1.34	3	314
6021	Wonderful Guest Bedroom in Manhattan for SINGLES	11528	Claudio	Manhattan	Upper West Side	40.79826	-73.96113	Private room	85	2	113	2019-07-05	0.91	1	333
6090	West Village Nest - Superhost	11975	Alina	Manhattan	West Village	40.7353	-74.00525	Entire home/apt	120	90	27	2018-10-31	0.22	1	0
6848	Only 2 stops to Manhattan studio	15991	Allen & Irina	Brooklyn	Williamsburg	40.70837	-73.95352	Entire home/apt	140	2	148	2019-06-29	1.20	1	46
7097	Perfect for Your Parents + Garden	17571	Jane	Brooklyn	Fort Greene	40.69169	-73.97185	Entire home/apt	215	2	198	2019-06-28	1.72	1	321
7322	Chelsea Perfect	18946	Doti	Manhattan	Chelsea	40.74192	-73.99501	Private room	140	1	260	2019-07-01	2.12	1	12
7726	Hip Historic Brownstone Apartment with Backyard	20950	Adam And Charity	Brooklyn	Crown Heights	40.67592	-73.94694	Entire home/apt	99	3	53	2019-06-22	4.44	1	21
7750	Huge 2 BR Upper East Cental Park	17985	Sing	Manhattan	East Harlem	40.79685	-73.94872	Entire home/apt	190	7	0			2	249

3.5 Data Cleaning:

Data Cleaning is a critical step in our project. It is the process of identifying and correcting errors and removing the inconsistencies, and inaccuracies in a dataset to ensure it is complete and free from irregular values and ready for our analysis.

We first address the duplicates and missing data by either imputing values or removing the columns or rows. We do it to prevent bias in our analysis. By handling missing data, we avoid errors and obtain accurate insights about the dataset.

In our dataset if a column has more null values or nan, we simply remove the entire column if it is not that important for our analysis or we replace the missing values with mean median or mode.

Later, we perform data validation where we check to identify and manage any inconsistent or inaccurate data points and eliminate the outliers in the dataset, which are extreme values that might affect our result analysis as few algorithms are less sensitive to outliers.

We also ensure that the data is properly formatted and standardized. We use Pandas factorize method to encode categorical variables in the Data Frame. Categorical variables are variables that have distinct groups, and they need to be converted into numerical format before using them in the machine learning algorithms, as most algorithms work only on numerical inputs.

3.6 Methodology

1) **Exploratory Data Analysis (EDA)** - Here we examine the dataset to gain more insight of the data's structure and pattern. The main objective of EDA is to extract insights and identify all the possible relationships or trends within the data. It is the foundation for analyses, modeling, and interpretation. It is used in various fields like research, business, and academics.

2) **Feature selection** - Feature selection is another crucial step to construct the ML predictive models. It involves taking the most important features from the original dataset. The main goal of feature selection is to improve our ML model performance and prevent overfitting in our model.

3) **Model selection** - Model selection is choosing the algorithm that works best for the task. When we are working on a predictive modelling project, there are various ML algorithms that are available to choose from. Then we compare different algorithms and pick the ML algorithm that has high accuracy.

4) **Model Training** - Split our Airbnb dataset into training data on which the model "learns" the patterns, relationships, and underlying structure present in the our train data and the test data that is used for evaluating our model's performance on new dataset it has not seen before. Then Fine tune the hyperparameters to optimize the models performance and prevent overfitting.

5) **Model Evaluation** - Evaluate the trained models on the validation set using evaluation metrics like

6) **Model Interpretation** - Interpret our model's predictions and coefficients to understand the impact of each feature on housing prices.

7) Model Output - We take the output of the model and visualize it graphically.

Section 4

Exploratory Data Analysis Output

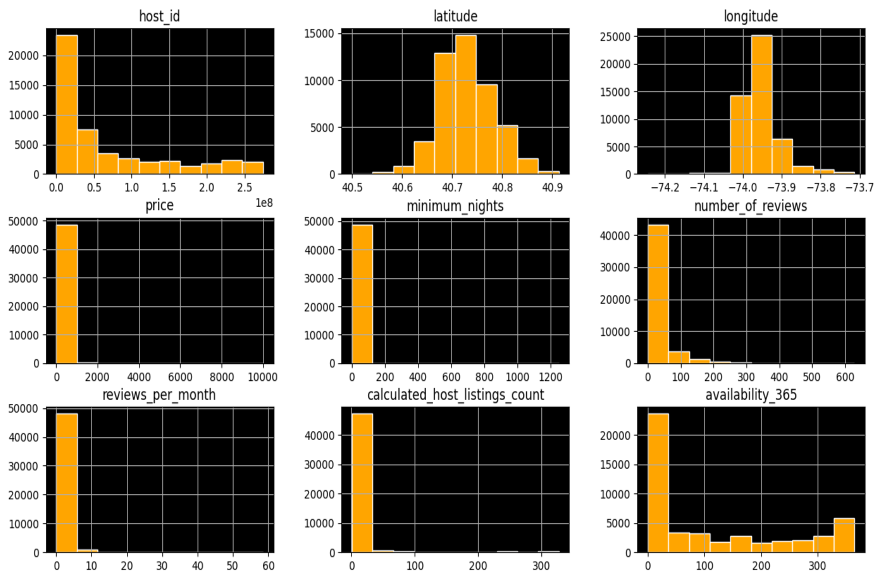


Fig 3.6 histogram plot

In the above graph, we customize the appearance of a histogram plot using the Matplotlib library in Python. We set the background colour of the plot to black by modifying the `rcParams` dictionary of Matplotlib and plot the histogram of the 'data' variable with the bars coloured in orange and the edges in white. We use the 'fig size' parameter which is used to set the size of the plot to 15 inches in width and 8 inches in height. Finally we display the histogram plot with the specified customizations where a histogram with a black background, orange bars, and white edges is created which makes it easier for us to interpret and understand the underlying data distribution.

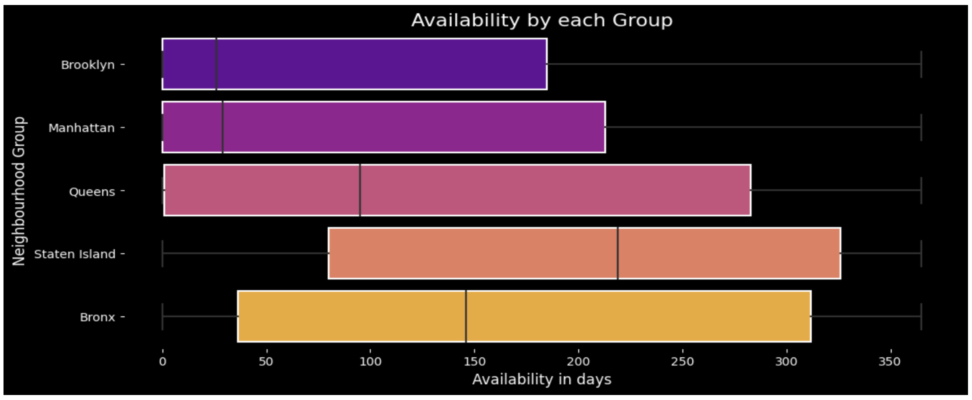


Fig 3.9 Bar Plot 2

The above graph shows a box plot that shows how the availability of rooms varies throughout the year in different neighbourhood groups. By converting our 'availability_365' column to numeric values, the code allows for an accurate plotting of our data. The plot is created with a palette of vibrant colours that makes our graph more clear and beautiful against the black background. Each box in the plot represents a neighbourhood group, showing the range of availability from least to most throughout the year. The end result is a visually clear and informative chart that gives us quick info about the room availability trends in different neighbourhoods

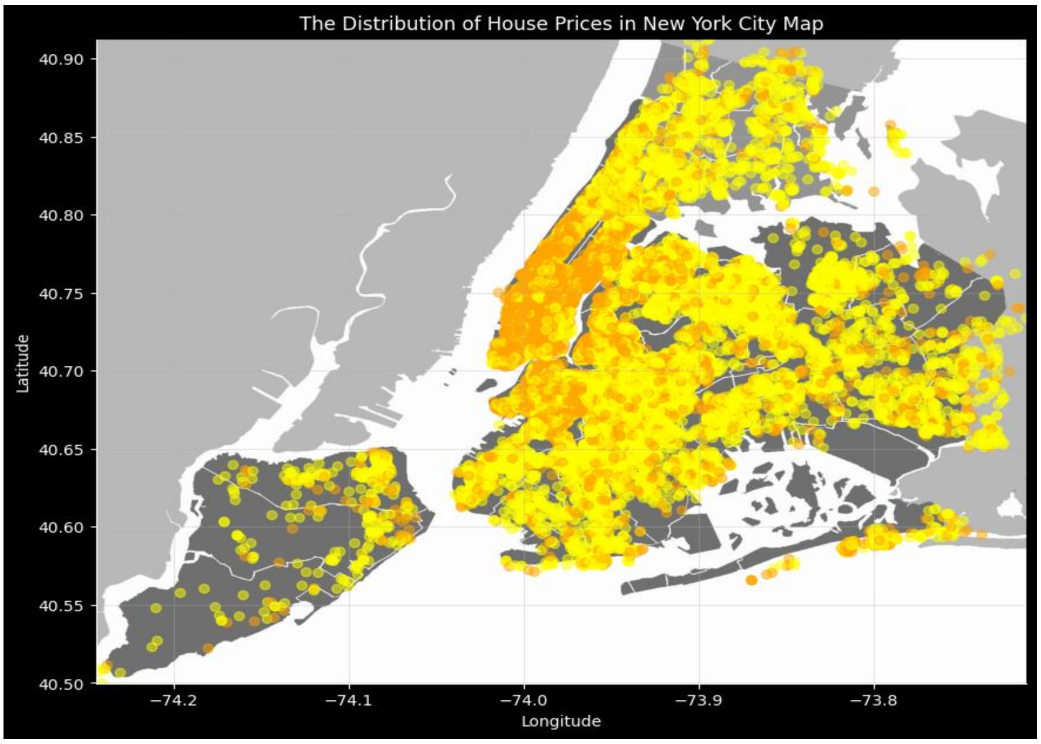


Fig 3.7 Scatter Plot On NYC map

This above image plots points on a loaded map of New York City that shows where homes are priced above and below the average. The average price of homes is calculated from a list of prices. Then, for each home if the price is below the average, we colour yellow and if it's above average, we colour orange. Using their longitude and latitude information we position it. The map itself is set against a black background, and gridlines are added for easy reference. The plot also includes a title and labels for the x and y axes, all in white. The end result is a color-coded map that gives you a quick visual understanding of home prices across the city.

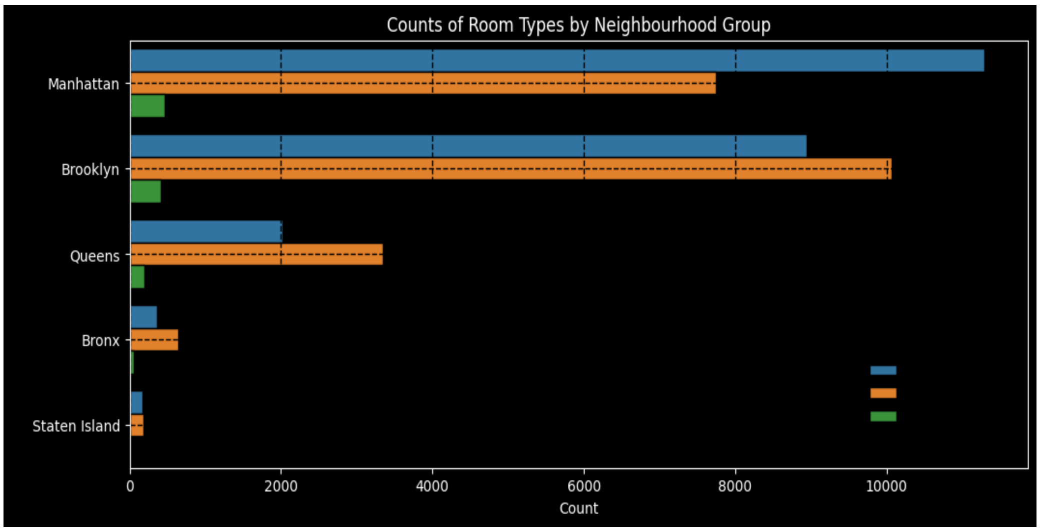


Fig 3.8 Bar Plot 1

The above graph we use the package called seaborn and display an engaging bar chart that shows us the number of different types of rooms available in each neighbourhood group. The data is first sorted and arranged by neighbourhood group and room type, and then we take the count of each type of room in each neighbourhood. We apply the chart with a black background and white text . Each bar's colour represents a different type of room, making it easy for us to distinguish between them. A legend is also provided for our reference. The result is a visually appealing and informative chart that tells us at a glance how many of each type of room are available in each neighbourhood group.

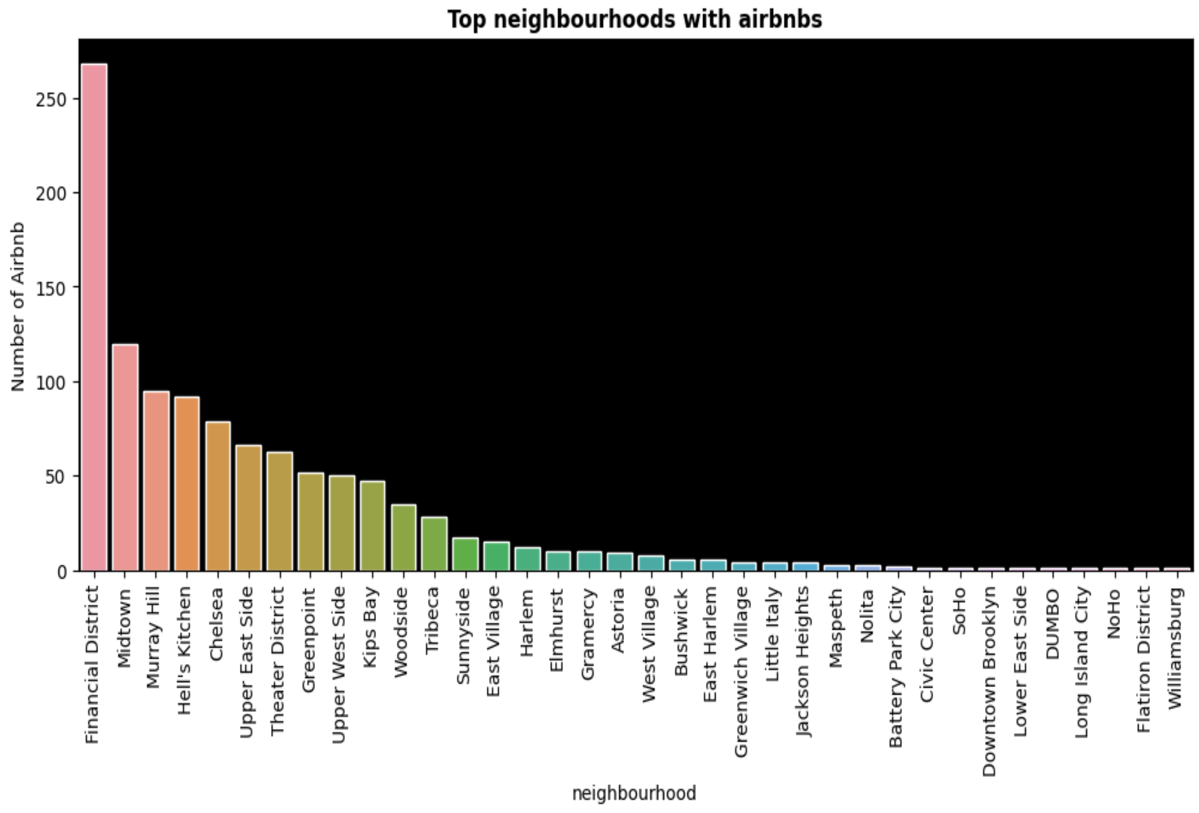


Fig 3.10 Bar Plot 3

The above graph shows us the top 10 hosts with the most listings in our dataset. Once we identify these super-hosts, we create a bar chart showing which neighbourhoods their properties are located in. Each bar in the chart gives us each neighbourhood name and the height of the bar which tells us how many properties from these top hosts are in that neighbourhood from the data . The neighbourhoods are arranged in descending order, so that we can easily see which one has the most properties. The chart is given a clear look with black text and with a black background, and the bars are outlined in white for more look . The result is handy for us as the visuals clearly guide us to where these top hosts' properties are concentrated.

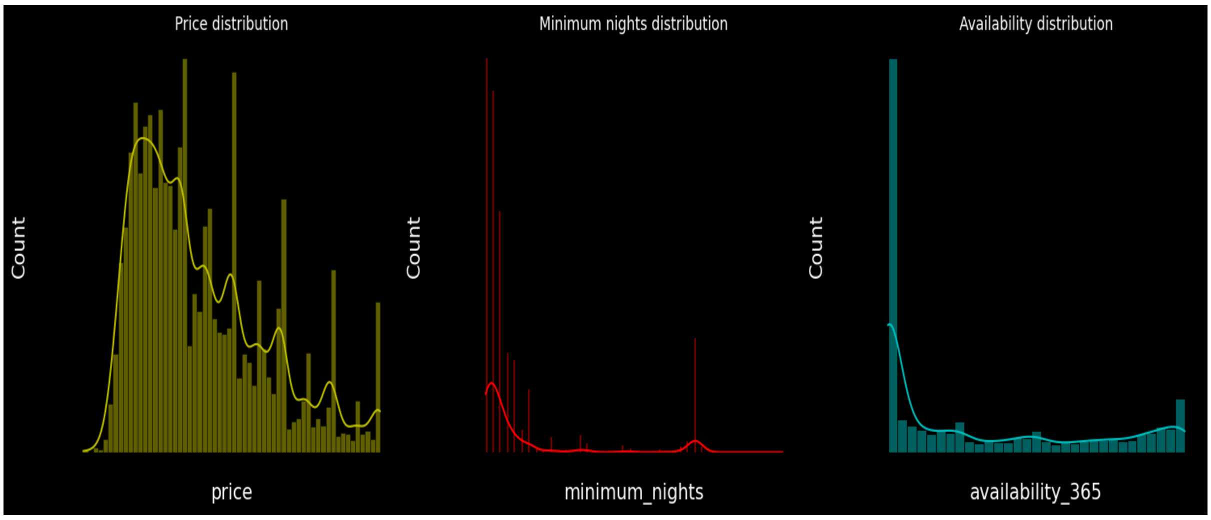


Fig 3.11 Histograms using seaborn library

Using the seaborn library in Python to show the above graph shows 3 graphs of Airbnb properties in the form of 3 different colours histograms. The first one that's in yellow, tells us how the prices are distributed. The second one that's in red, tells us the minimum number of nights guests are required to stay are spread out. The third one , shows us in cyan colour, tells the distribution of availability throughout the year for all properties. These 3 graphs are kept side by side for easy comparison for us , each with their title in white and a black background. The result is a clear and visual narrative that gives clarity to us.

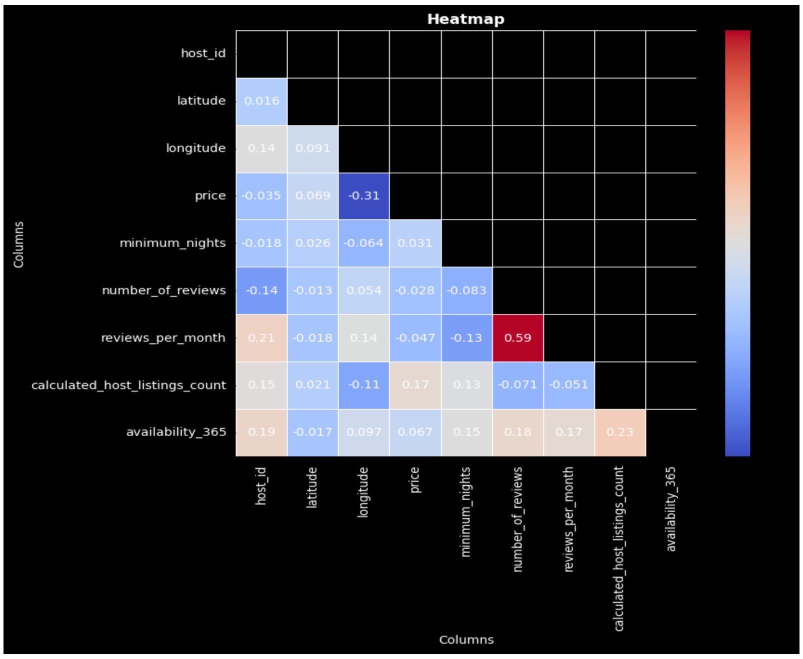


Fig 3.12 Heatmap

The above Heatmap visualizes the correlation matrices. This allows us to identify the relationships between each variable in our dataset. The chart gives us a clear look with white text and with a black background. The code uses Seaborn's heatmap function to generate a heatmap, with the mask parameter as we set the annot to True and we annotate the cells with their values. It also adjusts us the linewidths between the cells to 0.5.

Section 5

ML MODELS

5.1 Ridge Regression

Ridge Regression is one of the regularization technique used in linear regression to prevent overfitting and improve our models' prediction and give better prediction

In linear regression the main aim is to find the best fit line and find the relationship between input and output variables. But in our Airbnb dataset we have many features and they are highly correlated . so using linear regression would cause overfitting. To prevent these we use ridge regression as an alternative to solve the issue.

In Ridge Regression we add regularization terms to the linear regression equation.

The ridge regression formula is - **min RSS + α * $||\beta||$^2**

Where

RSS = residual sum of squares which is the sum of squared differences between the predicted and actual values

β = weights of the coefficients of the independent variables

α = a regularization parameter that controls the strength of the penalty term

Below is the step-by-step description of the working of our Ridge Regression

Step 1 - Data Preparation: Organize our Airbnb dataset with input values as all columns except the price value column and price value column as our target values.

Step 2 - Feature Scaling: Normalize our Airbnb data's input features to ensure they are in the same scales so that it makes our algorithm easier to learn.

Step 3 - Adding Regularization Term: Here we just add penalty term based on the L2 norm of the coefficients to the linear regression equation

Step 4 - Tuning Hyperparameter: Choose an appropriate regularization parameter that's the alpha value as we need it to balance model complexity.

Step 5 - Model Training: Then we Optimize the coefficients by minimizing the sum of squared errors and the regularization term.

Step 6 - Coefficient Shrinkage: The penalty term pushes the coefficients towards zero thereby reducing their impact on the model.

Step 7 - Predictions: We make use of our trained model to make predictions on new or unseen data in our dataset.

Step 8 - Model Evaluation: To check the model's performance we use various evaluation metrics like Mean Squared Error (MSE), Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R-square.

Step 9 - Choosing Alpha value: We use a list of alpha value [0.01,0.1,1.0,10.0,100] and at the end we choose that is optimal to us

Step 10 - Balancing Bias and Variance: We then adjust the regularization strength to achieve a trade-off between bias and variance.

Step 11 - Interpretability: We then interpret our model by examining the magnitude and sign of coefficients.

Step 12 - Prediction on New Data: We apply our trained model to new instances to predict the sales price (target values).

Below are the 3 major advantages of using Ridge Regression:

1) Regularization: It prevents overfitting by adding a regularization term.

2)Multicollinearity Handling: It reduces the impact of multicollinearity, making it suitable for datasets with more correlated features .

3)Bias-Variance Trade-off: Ridge Regression ensure it keeps a balance between bias and variance, improving generalization to new data.

Below are the 3 major disadvantages of using Ridge Regression:

1) Bias in Coefficients: The penalty term in the model might bias coefficients away from their actual values.

2) Loss of Interpretability: The Coefficients might become difficult to interpret due to the L2 regularization.

3) Inefficient for Large Features: when you have large feature spaces, Ridge Regression might not perform well compared to other algorithms like Lasso Regression.

5.2 XGBoost (eXtreme Gradient Boosting)

XGBoost, also known as Extreme Gradient Boosting, is a widely used machine learning approach to numerical problem or classification problem prediction. It was developed by Tianqi Chen. XGBoost is a type of ensemble learning approach in which multiple independent multiple models are integrated to generate the most efficient and precise model. This approach is similar to assembling a team of advanced models that work together to achieve good results.

XGBoost is designed to be more efficient, scalable, and highly effective in handling complex datasets when compared to other models.

Below is the XGBoost regression works, in a simple step-by-step process.

Step 1 - In XGBoost we start by selecting a constant value for all instances in the dataset. This initial value can be the mean of the target variable that is the sales price in our dataset for our regression problem. and this is called the base model of our algorithm.

Step 2 - We specify the number of iterations for the given loop to run.

Step 3 - For each iteration, we calculate the residuals that are the differences between the observed and predicted values of the sales price column (target variable).

Step 4 - We built to predict these residuals, instead of the actual target variable. That model is called a weak learner. The term "weak" refers to the fact that the model is only slightly better than our random guessing.

Step 5 - Predictions are updated by adding the predictions from the new weak learner to the existing predictions. The contribution of our new model to the final prediction is controlled by a parameter called shrinkage.

Step 6 -Once the predictions are updated our algorithm calculates a loss function like Mean Squared Error or mean absolute error for regression. We then apply a regularization term, which penalizes our model complexity, so that we can prevent overfitting.

Step 7 - we then use gradient descent to minimize the objective function and improve the overall performance of the model. This is done by changing the predictions based on the negative gradient of the loss function, therefore called “gradient boosting".

Step 8 - Steps 3 to 7 are repeated until we get the model's performance on a validation set stops improving.

Step 9 - Finally, our model’s output for a given instance is the sum of the predictions from all the models.

We us many parameters in our Xgboost algorithm

1) max depth - How deep our tree can grow so that we can avoid learning too much detail and prevent overfitting

2) min child weight - The minimum weight needed for us to create a new node in the tree, so that we do not have to create nodes.

3) eta - A step size for our model's learning, to make sure it's not rushing into conclusions.

4) subsample - The portion of our data is randomly picked for training, so that we can add variety to what the model learns.

5) colsample_bytree- The portion of our features randomly picked for creating each tree, again for variety.

6) gamma - How much benefit a split must provide to happen so that we prevent overfitting

7) reg lambda- A penalty term to keep the model's complexity in check, it controls how the model balances weights.

8) reg alpha - It is Another penalty term, mainly to force less important feature's weights towards zero.

9) silent - Whether or not to print updates while the model is learning, just for your peace of mind.

10) objective - It tell us what our model should aim to do

11) eval metric - It tells us how to measure our model's performance.

5.3 Model Metrics

Mean Squared Error (MSE) - It is a metric used to check the performance of our regression model. It calculates the average squared difference between the predicted values and the actual values in our Airbnb dataset. The lower our Mses core is, the better our model's predictions match the correct values.

MSE = Σ (predicted value - actual value) ^2 / N

where:

Σ - It represents the sum over all data points in our Airbnb dataset.

N - is the total number of data points in our Airbnb dataset.

Mean Absolute Error (MAE) - It is a metric which Similar to Mean Squared Error (MSE) but here it measures the accuracy of our model's predictions, but here instead of squaring the errors we take the absolute value of the differences between the predicted values and the actual values.

MAE = Σ |predicted value - actual value| / N

where:

Σ - It represents the sum over all data points in our Airbnb dataset.

N - total number of data points in our Airbnb dataset.

Root Mean Squared Error (RMSE) - It is a metric used to check the performance of our regression models.

RMSE = √ (Σ (predicted value - actual value)^2 / N)

where:

Σ - It represents the sum over all data points in our Airbnb dataset.

N - It is the total number of data points in our Airbnb dataset.

R-squared (R^2) -R^2 tells us how well our model explains the variability in the data. R^2 ranges from 0 to 1. A higher R^2 indicates that a larger proportion of the variation in the dependent variable can be explained by our independent variables.

R^2 =1- SSres/SStot

SSres - It sum of squared residuals

SStot - Its total sum of squares.

5.4 Model Output

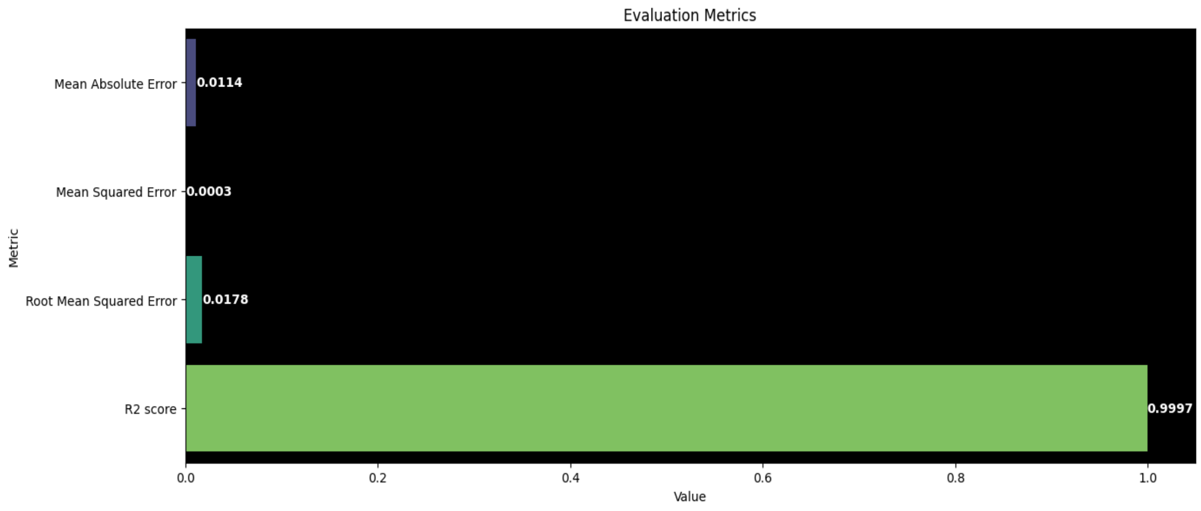


Fig 3.13 Evaluation Metrics

The length of each bar in the above graph shows the values of each metric, and the values are also printed on each bar. You would ideally want the bars for MAE, MSE, and RMSE to be as short (indicating lower error), while the bar for the R2 score should more (good performance)

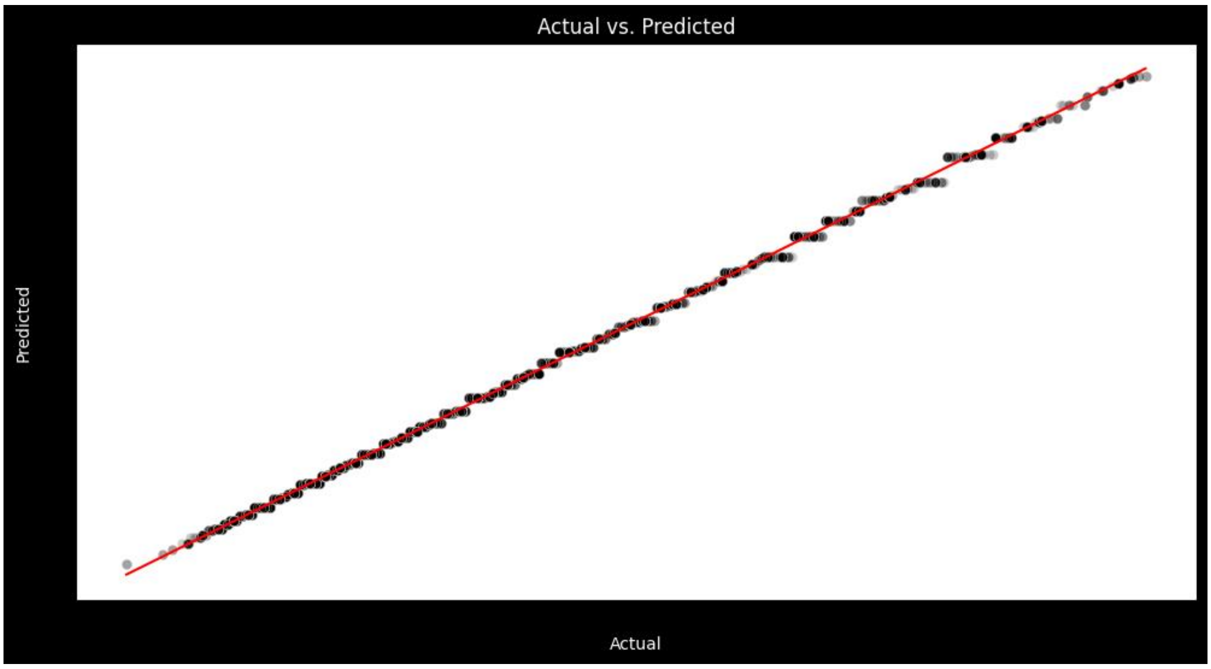


Fig 3.14 scatter plot for actual vs predicted

The scatter plot shows us the comparison of the predicted values from a model to the actual values. This type of plot is known as an "Actual vs. Predicted" plot.

5.5 Model Comparison

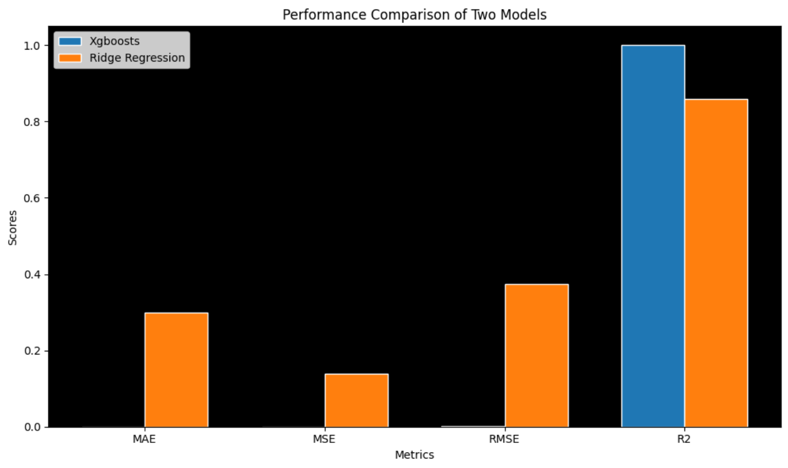


Fig 3.15 bar plot for XGBoost and Ridge Regression

The above graph is used to visualize and compare the performance of our two different supervised machine learning models (XGBoost and Ridge Regression) using a bar chart. with 2 different colours (orange for Ridge Regression and blue for boost) To build the above graph we use the Matplotlib library to create the visualization. It represents Four evaluation metrics, they are Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared (R2).

We can clearly see that our scores our xgboost regression model is performing better than ridge regression as r2 score are more and less error in xgboost

Section 6 Limitations

When we work with the project to build a robust Machin learning model for our Airbnb housing prices in New York City using the two regression model (Ridge Regression and Gradient Boosting Regressor algorithms) For an clear understanding of the our model results, it is important to accept the limitations of the our study that is conducted.

i) Data Amount and Time Restrictions

Our dataset is limited to New York City only and cannot be used to check for other dataset. This is a major limitation that restricts the generalizability of our model. Additionally, our data is not clear about which season or events or any market trend that the data was recorded So the trained model cannot be used for forecasting future prices and these is the major limitation of this entire project.

ii) Data Splitting methods

If we are simply splitting the dataset into train-test split, cross-validation. the model's predictive power can vary widely. In proper validation methods can lead our prediction to incorrect model's performance.

iii) Algorithm limitations

The two algorithms that we used in these project (Ridge Regression and Gradient Boosting) has its own drawbacks which is in Ridge Regression is a linear model, which might not be able to capture complex non-linear relationships between our features and on the other hand, Gradient Boosting Regressor is more Vulnerable to overfitting if not get carefully with the correct hyperparameters particularly when dealing with high-dimensional data or outliers.

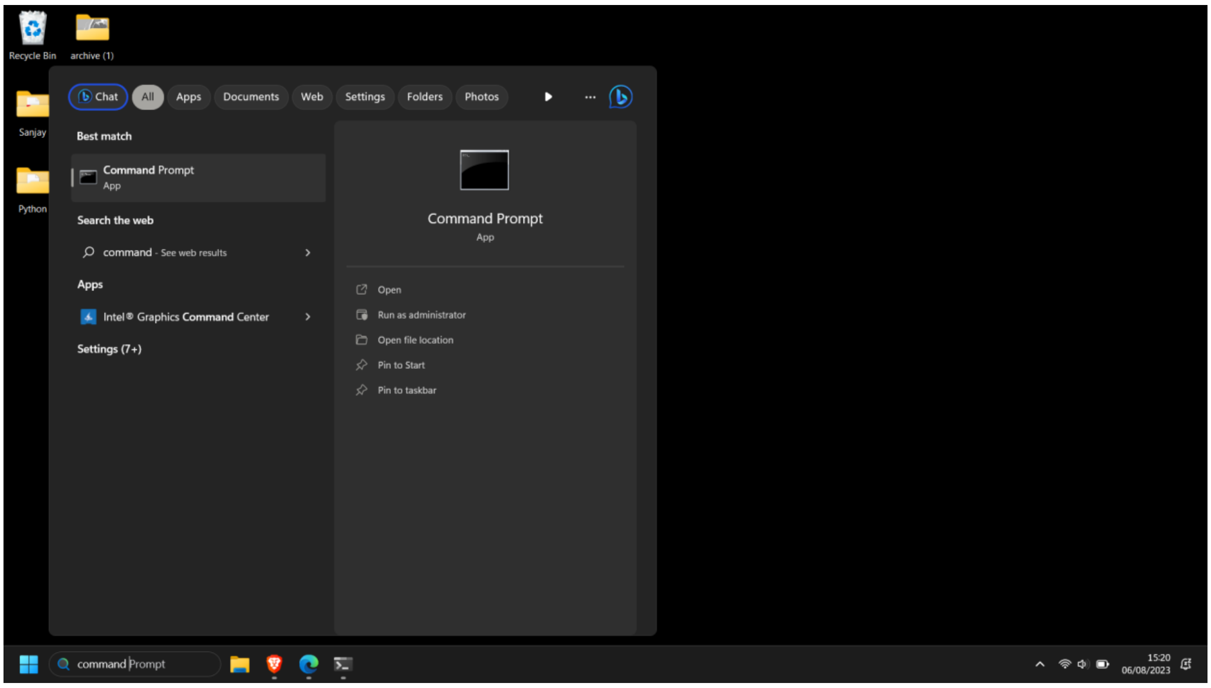
iv) Ethical issue

Our study does not consider the potential negative impacts or drawbacks of using machine learning pricing models for housing, such as discrimination or inequality. This is a complex issue that requires a more disciplinary approach, and it is beyond the scope of our study that’s done so far.

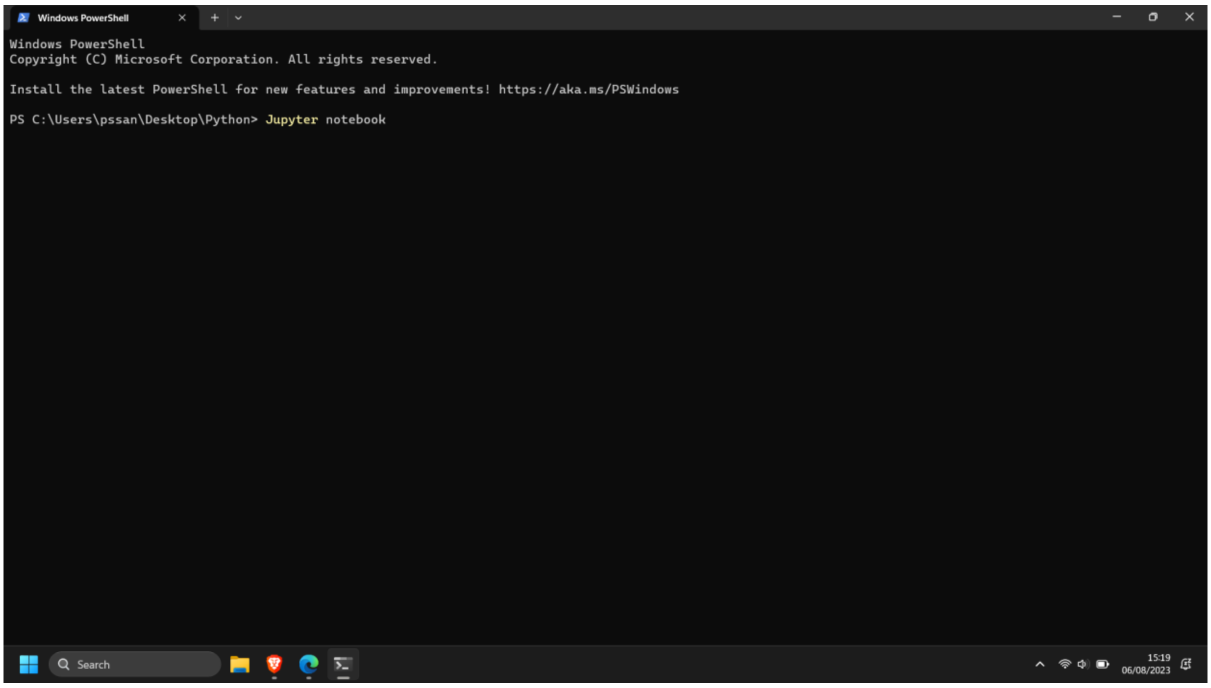
Section 7 Opening Work Environment

Steps To Open Jupyter Notebook

Step 1 - Type in Command prompt in Search bar in windows home Page and click enter.

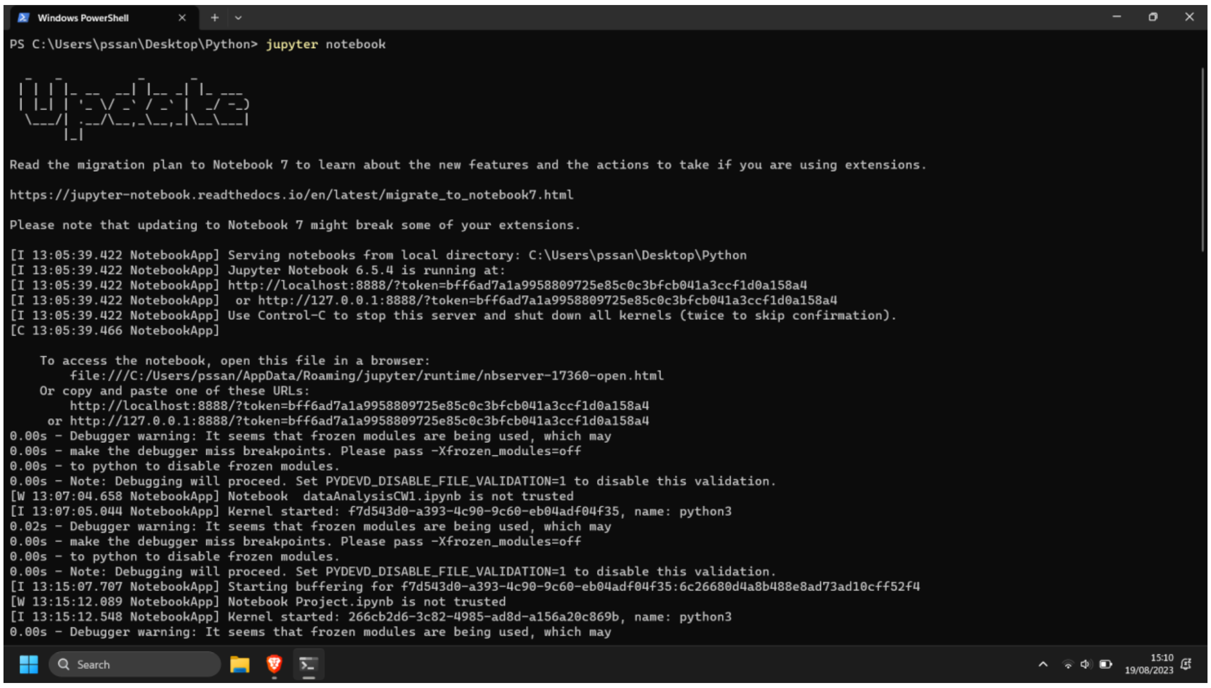


Step 2 -Type Jupter notebook in the command prompt



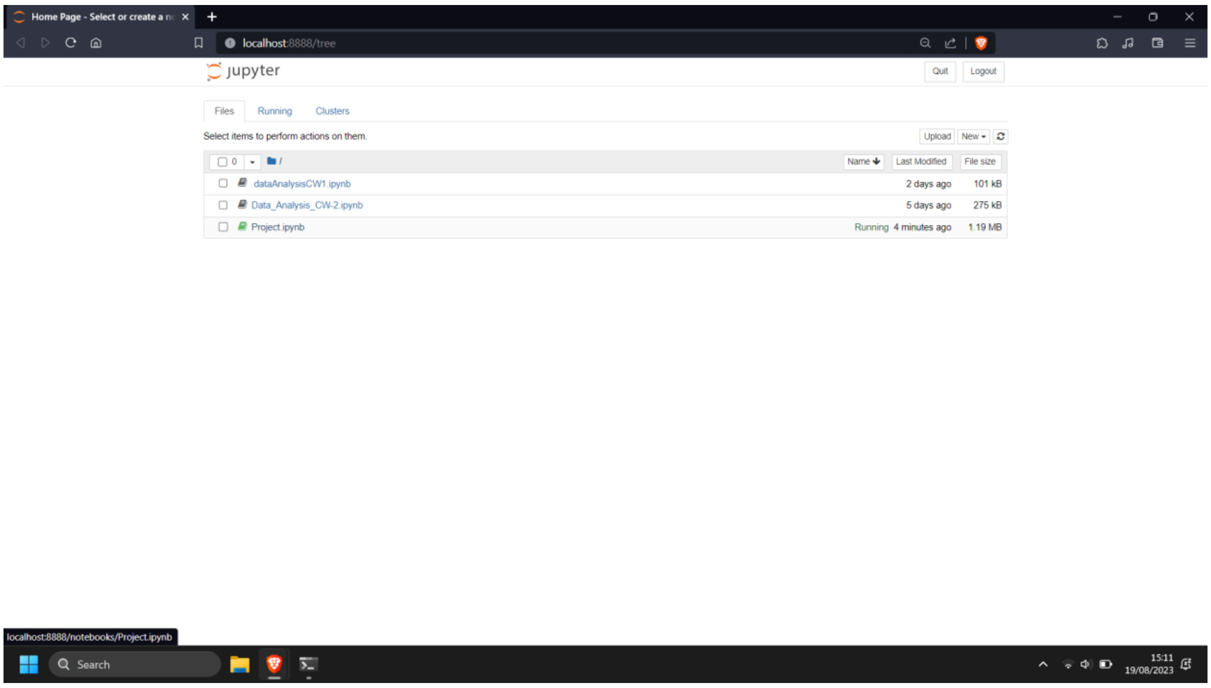
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\pssan\Desktop\Python> Jupyter notebook
```

Step 3 - Wait for the jupyter notebook to load and open.



```
PS C:\Users\pssan\Desktop\Python> jupyter notebook

  _   _          _      _
 | | | |_ __  __| |__ _| |_ ___
 | |_| | '_ \/ _` / _` |  _/ -_)
  \___/| .__/\__,_\__,_|\__\___|
       |_|

Read the migration plan to Notebook 7 to learn about the new features and the actions to take if you are using extensions.

https://jupyter-notebook.readthedocs.io/en/latest/migrate_to_notebook7.html

Please note that updating to Notebook 7 might break some of your extensions.

[I 13:05:39.422 NotebookApp] Serving notebooks from local directory: C:\Users\pssan\Desktop\Python
[I 13:05:39.422 NotebookApp] Jupyter Notebook 6.5.4 is running at:
[I 13:05:39.422 NotebookApp] http://localhost:8888/?token=bff6ad7a1a9958809725e85c0c3bfcb041a3ccf1d0a158a4
[I 13:05:39.422 NotebookApp]  or http://127.0.0.1:8888/?token=bff6ad7a1a9958809725e85c0c3bfcb041a3ccf1d0a158a4
[I 13:05:39.422 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 13:05:39.466 NotebookApp]

    To access the notebook, open this file in a browser:
        file:///C:/Users/pssan/AppData/Roaming/jupyter/runtime/nbserver-17360-open.html
    Or copy and paste one of these URLs:
        http://localhost:8888/?token=bff6ad7a1a9958809725e85c0c3bfcb041a3ccf1d0a158a4
     or http://127.0.0.1:8888/?token=bff6ad7a1a9958809725e85c0c3bfcb041a3ccf1d0a158a4
0.00s - Debugger warning: It seems that frozen modules are being used, which may
0.00s - make the debugger miss breakpoints. Please pass -Xfrozen_modules=off
0.00s - to python to disable frozen modules.
0.00s - Note: Debugging will proceed. Set PYDEVD_DISABLE_FILE_VALIDATION=1 to disable this validation.
[W 13:07:04.658 NotebookApp] Notebook  dataAnalysisCW1.ipynb is not trusted
[I 13:07:05.044 NotebookApp] Kernel started: f7d543d0-a393-4c90-9c60-eb04adf04f35, name: python3
0.02s - Debugger warning: It seems that frozen modules are being used, which may
0.00s - make the debugger miss breakpoints. Please pass -Xfrozen_modules=off
0.00s - to python to disable frozen modules.
0.00s - Note: Debugging will proceed. Set PYDEVD_DISABLE_FILE_VALIDATION=1 to disable this validation.
[I 13:15:07.707 NotebookApp] Starting buffering for f7d543d0-a393-4c90-9c60-eb04adf04f35:6c26680d4a8b488e8ad73ad10cff52f4
[W 13:15:12.089 NotebookApp] Notebook Project.ipynb is not trusted
[I 13:15:12.548 NotebookApp] Kernel started: 266cb2d6-3c82-4985-ad8d-a156a20c869b, name: python3
0.00s - Debugger warning: It seems that frozen modules are being used, which may
```

Step 4 - Choose the file you want to work with

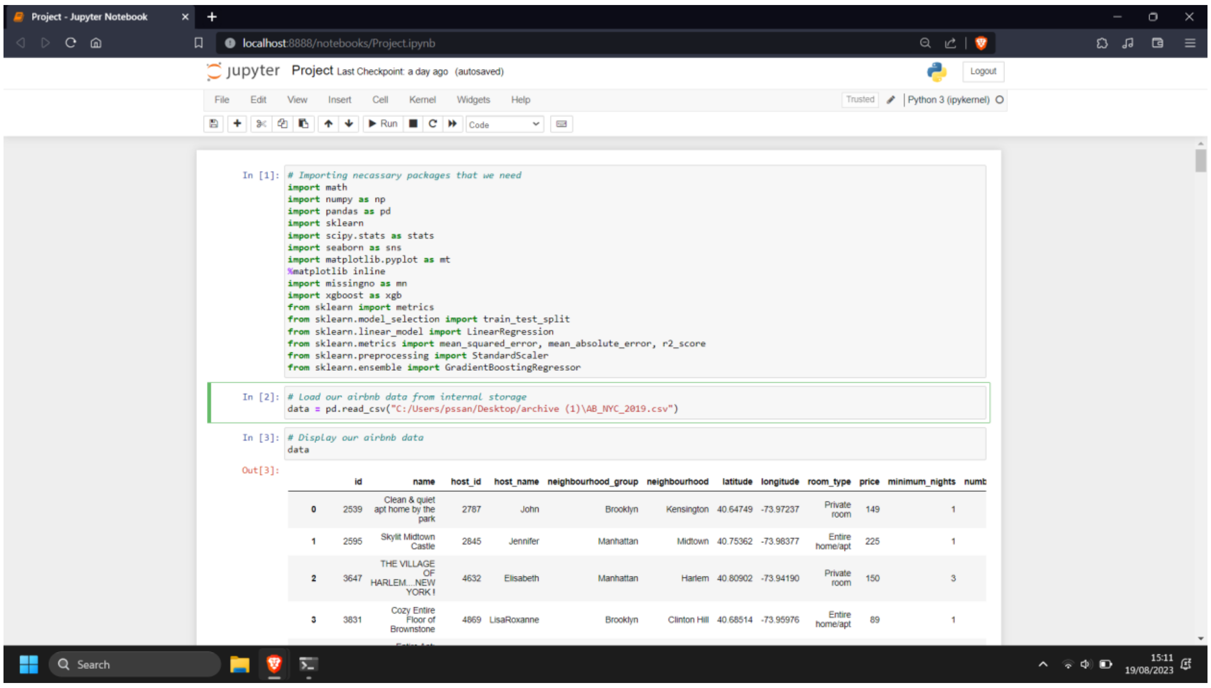


Select items to perform actions on them.

Name		Last Modified	File size
dataAnalysisCW1.ipynb		2 days ago	101 kB
Data_Analysis_CW-2.ipynb		5 days ago	275 kB
Project.ipynb	Running	4 minutes ago	1.19 MB

localhost:8888/notebooks/Project.ipynb

Step 5 - Now we can start coding.



```
In [1]: # Importing necassary packages that we need
        import math
        import numpy as np
        import pandas as pd
        import sklearn
        import scipy.stats as stats
        import seaborn as sns
        import matplotlib.pyplot as mt
        %matplotlib inline
        import missingno as mn
        import xgboost as xgb
        from sklearn import metrics
        from sklearn.model_selection import train_test_split
        from sklearn.linear_model import LinearRegression
        from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
        from sklearn.preprocessing import StandardScaler
        from sklearn.ensemble import GradientBoostingRegressor

In [2]: # Load our airbnb data from internal storage
        data = pd.read_csv("C:/Users/pssan/Desktop/archive (1)\AB_NYC_2019.csv")

In [3]: # Display our airbnb data
        data

Out[3]:
```

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price	minimum_nights	numb
0	2539	Clean & quiet apt home by the park	2787	John	Brooklyn	Kensington	40.64749	-73.97237	Private room	149	1	
1	2595	Skylit Midtown Castle	2845	Jennifer	Manhattan	Midtown	40.75362	-73.98377	Entire home/apt	225	1	
2	3647	THE VILLAGE OF HARLEM....NEW YORK !	4632	Elisabeth	Manhattan	Harlem	40.80902	-73.94190	Private room	150	3	
3	3831	Cozy Entire Floor of Brownstone	4869	LisaRoxanne	Brooklyn	Clinton Hill	40.68514	-73.95976	Entire home/apt	89	1	

Section 8 Ethical Considerations

While we conduct the machine learning projects, several ethical considerations need to take into consideration:

1) Data Privacy: We must ensure that the privacy and security of Airbnb users' data is important. It's important to handle the data with relevant privacy laws and regulations (e.g., GDPR) and take necessary steps to protect sensitive information.

2) Issues with Bias: We should avoid biases in our Airbnb dataset and the resulting model as these biased data can lead to incorrect predictions and wrong decisions leading to various problems.

3) Visibility: The process to build a project that contains data collection, feature selection, and model building should be transparent. The Users should know how their data is used and how predictions are made.

4) Responsibility: As a researcher, we are accountable for the impact of our work. If the model is used to make important decisions, it's important to frequently monitor the model's performance.

5) Interpretability: Ensuring that the model's judgements can be understood helps improve trust, particularly for models that have an impact on people's lives. Users can better comprehend and accept a model's results if it can explain its predictions.

Acknowledgment

I want to thank Professor Vladimir Vovk - Department of Computer Science for the guide and provide feedback throughout the entire project. Additionally I thank Airbnb for making the data publicly available and thereby contributing to the community of data science and machine learning researchers.

References

1. Introduction to Machine Learning with Python (2016)
   - Authors: Andreas Müller and Sarah Guido
   - Publisher: O'Reilly Media
   - Description: This book provides a hands-on introduction to machine learning using Python. It covers fundamental concepts, popular algorithms, and showcases practical examples.

2. An Introduction to Statistical Learning (2013)
   - Authors: Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani
   - Publisher: Springer
   - Description: This book offers an accessible overview of statistical learning, emphasizing both theory and practical applications. It includes real-world examples from various fields such as finance, marketing, and biology.

3. Python for Data Analysis (2012)
   -Author: Wes McKinney
   - Publisher: O'Reilly Media
   - Description: The book focuses on using Python for data analysis. It covers how to manipulate, clean, and analyze data using Python libraries like pandas.

4. The Hundred-Page Machine Learning Book (2019)
   - Author: Andriy Burkov
   - Publisher: Andriy Burkov
   - Description: This book offers a concise overview of key machine learning concepts and algorithms. It's designed for quick understanding and serves as a useful reference.

5. Pattern Recognition and Machine Learning (2006)
   - Author: Christopher M. Bishop
   - Publisher: Springer
   - Description: The book covers both pattern recognition and machine learning. It provides a comprehensive introduction to these fields and delves into the mathematical foundations and principles.

2. Research Papers:

- Sifei Lu, Zengxiang Li, Zheng Qin, Xulei Yang, Rick Siow Mong Goh (2017) - A Hybrid Regression Technique for House Prices Prediction -predicting house prices in cities like Bengaluru by employing various regression techniques

- Manasa J,Narahari N S,Radha Gupta (2020) - Machine Learning based Predicting House Prices using Regression Techniques -This study focuses on predicting house prices in cities like Bengaluru using various regression techniques

- Sadi Alawadi, David Mera, Manuel Fernández-Delgado, Fahed Alkhabbas, Carl Magnus Olsson & Paul Davidsson (2019) - Comparison of machine learning algorithms for forecasting indoor temperature in smart buildings

- Tianqi Chen, Tong He - Xgboost: eXtreme Gradient Boosting (2001) - XGBoost algorithms are efficient in both linear models and tree-based learning. It's Used for various tasks like regression, classification, and ranking.

- Iqbal H. Sarker -Machine Learning: Algorithms, Real-World Applications and Research (2021) - the author's paper serves as a technical reference for real-world scenarios and applications in the current landscape of AI and ML.

- Tom M. Mitchell - Machine Learning (1997) Machine learning is focused on creating computer programs that can enhance their performance through experience.

- Petr Silhavy, Radek Silhavy & Zdenka Prokopova (2019) Outliners Detection Method for Software Effort Estimation Models – In study study we study the into using outlier detection methods to fit in-house datasets, with the goal of improving the accuracy of estimation models.

- Tom Dietterich (1995) - Overfitting and Under computing in Machine Learning. It speaks about the major problem that exists in machine learning supervised learning that is, learning from labelled training data.

- Adeola Ogunleye and Qing-Guo Wang (2020) - XGBoost Model for Chronic Kidney Disease Diagnosis – The functioning of Xgboost in a particular dataset

- Gary Smith &Frank Campbell (2012) - A Critique of Some Ridge Regression Methods. This model estimates aim to counteract potentially excessive least squares estimates due to multicollinearity by shrinking them towards zero

- Yahya, W.B. and Olaitan, J.B. (2014) 'A note on ridge regression modelling techniques'

3. Websites:

- Airbnb. (2022). About Airbnb - [https://www.airbnb.com/about]

- javatpoint - Python Statement - [https://www.javatpoint.com/python-tutorial]

- w3schools - Python Functions - [https://www.w3schools.com/python/default.as]

- Matworks - Supervised learning - [https://se.mathworks.com/discovery/supervised-learning.html]

- Inside Airbnb. (2022). Data and Tools - [http://insideairbnb.com/get-the-data.html]

- Scikit-learn Documentation. (2022). Scikit-learn User Guide - [https://scikit-learn.org/stable/user_guide.html]

- Towards Data Science. (2022). A Comprehensive Guide to Different Regression Techniques - [https://towardsdatascience.com/a-comprehensive-guide-to-different-regression-techniques-41e009e102a7]

- Analytics Vidhya. (2022). Complete Guide to Parameter Tuning in Gradient Boosting (GBM) in Python - https://www.analyticsvidhya.com/blog/2016/02/complete-guide-parameter-tuning-gradient-boosting-gbm-python/]

- Towards Data Science. (2022). Feature Engineering Techniques in Machine Learning with Python - [https://towardsdatascience.com/feature-engineering-techniques-in-machine-learning-with-python-ef6f36a6ade8]

Appendix

Step 1: Import Libraries
Import all the necessary Python libraries like Pandas, NumPy, XGBoost, Matplotlib.

Step 2: Load Data
Load the Airbnb dataset from a CSV file into a Pandas DataFrame.

Step 3: Initial Data Inspection
- Display the data and its shape.
- Check for duplicate and missing values.

Step 4: Data Cleaning
- Remove unneeded columns.
- Handle missing values and outliers.

Step 5: Exploratory Data Analysis (EDA)
1. Visualize the geographical distribution of house prices.
2. Display the counts of room types by neighbourhood group.
3. Show the availability of listings by each group.
4. Display the top neighbourhoods with Airbnb.
5. Visualize the distribution of price, minimum nights, and availability.
6. Generate a heatmap to display correlations between variables.

Step 6: Data Transformation
1. Encode categorical variables.
2. Normalize numerical variables.
3. Split the data into training and testing sets.

Step 7: Model Building and Training
XGBoost Model
1. Implement XGBoost with cross-validation.
2. Evaluate the model using RMSE, MSE, MAE, and \\(R^2 \\) score.
3. Visualize the predicted vs actual values.
Custom Ridge Regression Model
1. Perform cross-validation to find the best lambda value.
2. Train the model.
3. Evaluate the model using RMSE, MSE, MAE, and \\(R^2 \\) score.

Step 8: Model Comparison
- Compare the performance metrics of the XGBoost and Custom Ridge Regression models using a bar chart.

```
# Importing necessary packages that we need
import math
import numpy as np
import pandas as pd
import sklearn
import scipy.stats as stats
import seaborn as sns
import matplotlib.pyplot as mt
%matplotlib inline
import missingno as mn
import xgboost as xgb
from sklearn import metrics
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import GradientBoostingRegressor
# Load our airbnb data from internal storage
data = pd.read_csv("C:/Users/pssan/Desktop/archive (1)\AB_NYC_2019.csv")
# Display our airbnb data
data
# Display missing values in the entire dataframe
mn.matrix(data)
mt.show()
# Display the total no of rows and columns
nrow, ncolumn = data.shape
print("The dataset has {} rows and {} columns.".format(nrow, ncolumn))
# Display the total no of elements
total = nrow * ncolumn
```

```
print("The total number of elements in the dataset is {}.".format(total))
# Display the total no of duplicates
duplicatescount = data.duplicated().sum()
print("The dataset contains {} duplicate entries.".format(duplicatescount))
# Display the total no of missing values
misssingcount = data.isnull().sum().sum()
print("The dataset has {} missing values.".format(misssingcount))
# Display the total no of null values in each columns
I = data.isnull().sum()
print("The number of null values in each columns",I)
# Drop all the unrequired columns
data.drop(columns=['id','name','host_name','last_review'], axis=1, inplace=True)
# Replace the missing values in our dataset
data.fillna({'reviews_per_month':0}, inplace=True)
# Display the total no of null values in each columns
print("The number of null values in the dataset is =",data.isnull().sum().sum())
# Display all the remain columns
data.columns
# Display the overall stats of our dataset
data.info()
print("\n")
data.describe()
#Remove outliers using quartiles
Q1 = data['price'].quantile(0.25)
Q3 = data['price'].quantile(0.75)
IQR = Q3 - Q1
LB = Q1 - 1.5 * IQR
UB = Q3 + 1.5 * IQR
df = data[(data['price'] >= LB) & (data['price'] <= UB)]
```

```
print('The shape of the datset before removing outliners',data.shape)
print('The shape of the datset after removing outliners',df.shape)
# Visulize the Distribution of House Prices in New York City Map in our dataset
map_image_path = 'C:/Users/pssan/Desktop/archive (1)/New_York_City.jpg'
pricevalues = sum(df['price']) / len(df['price'])
circlecolors = ['yellow' if price < pricevalues else 'orange' for price in df['price']]
LO = df['longitude']
LA = df['latitude']
value = 0.5
k='white'
fig, g = mt.subplots(figsize=(15, 8), facecolor='black')
nycmap_image = mt.imread(map_image_path)
g.imshow(nycmap_image, extent=[LO.min(), LO.max(), LA.min(), LA.max()])
g.grid(True, linestyle='-', linewidth=value, alpha=value)
g.scatter(LO, LA, c=circlecolors, alpha=value)
g.set_title('The Distribution of House Prices in New York City Map', color=k)
g.set_xlabel('Longitude', color=k)
g.set_ylabel('Latitude', color=k)
g.tick_params(colors=k)
g.xaxis.label.set_color(k)
g.yaxis.label.set_color(k)
g.spines['bottom'].set_color(k)
g.spines['left'].set_color(k)
mt.show()
# Visulize the Counts of Room Types by Neighbourhood Group in our dataset
w='black'
v=1
mt.figure(figsize=(12, 5),facecolor=w)
```

```
organiseddata = df.groupby(['neighbourhood_group', 'room_type']).size().reset_index(name='count')

arrangeddata = organiseddata.sort_values(by='count', ascending=False)

a = sns.barplot(data=arrangeddata, y='neighbourhood_group', x='count', hue='room_type', orient='h',edgecolor=w)

l=a.legend(loc='center right', bbox_to_anchor=(1.00, 0.2), title='Room Type')

l.get_frame().set_linewidth(v+0.5)

l.get_frame().set_edgecolor('white')

a.set_title('Counts of Room Types by Neighbourhood Group', color='white')

a.set_xlabel('Count', color='white')

a.set_ylabel('Neighbourhood Group', color=w)

a.tick_params(colors='white')

a.grid(True, color=w, linestyle='--', linewidth=v)

for sp in a.spines.values():

   sp.set_color('white')

mt.show()

# Visulize the availability of each Group in our dataset

df['availability_365'] = pd.to_numeric(df['availability_365'])

mt.figure(figsize=(12, 5), facecolor='black')

fs=12

o='white'

sns.boxplot(data=df, y='neighbourhood_group', x='availability_365', palette='plasma', orient='h', boxprops={'edgecolor': o})

mt.title('Availability by each Group', color='white', fontsize=fs+4)

mt.xlabel('Availability in days', color=o, fontsize=fs)

mt.ylabel('Neighbourhood Group', color=o, fontsize=fs)

mt.gca().set_facecolor('black')

mt.gca().patch.set_edgecolor(o)

mt.tick_params(colors=o)

mt.show()
```

```
# Visualize the top neighbourhoods with airbnb in our dataset

a = df.loc[df['host_id'].isin(df.host_id.value_counts().head(10).index)]

mt.figure(figsize=(12, 5))

sns.countplot(data=a,                                                    x='neighbourhood',
order=a['neighbourhood'].value_counts().index)

c='black'

ax = mt.gca()

for p in ax.patches:

   p.set_edgecolor('white')

   p.set_linewidth(1)

ax.set_facecolor(c)

mt.xticks(rotation=90)

mt.title("Top neighbourhoods with airbnbs", weight='bold', color=c)

mt.ylabel('Number of Airbnb', color=c)

mt.show()

# Visulize the availability of 365 in our dataset

f, axes = mt.subplots(1, 3, figsize=(15, 5), facecolor='black')

sns.histplot(data=df[df['price']<=df['price'].quantile(0.99)],      x="price",      kde=True,
color="y", ax=axes[0])

sns.histplot(data=df[df['minimum_nights']<=df['minimum_nights'].quantile(0.99)],
x="minimum_nights", kde=True, color="r", ax=axes[1])

sns.histplot(data=df, x="availability_365", kde=True, color="c", ax=axes[2])

for ax in axes:

   ax.set_xlabel(ax.get_xlabel(), color='white',fontsize=15)

   ax.set_ylabel(ax.get_ylabel(), color='white',fontsize=15)

   ax.title.set_color('white')

axes[0].set_title('Price distribution')

axes[1].set_title('Minimum nights distribution')

axes[2].set_title('Availability distribution')

mt.tight_layout()
mt.show()
# Display the heatmap
i = df.select_dtypes(include=[np.number])
inf = i.corr()
c='white'
m = np.triu(np.ones_like(inf, dtype=bool))
mt.figure(figsize=(8, 7), facecolor='black')
q = sns.heatmap(inf, mask=m, annot=True, cmap='coolwarm', linewidths=0.5,)
q.set_facecolor('black')
for t in q.texts:
    t.set_color(c)
q.set_xticklabels(q.get_xticklabels(), color=c)
q.set_yticklabels(q.get_yticklabels(), color=c)
mt.title('Heatmap', color=c,weight='bold')
mt.xlabel('Columns', color=c)
mt.ylabel('Columns', color=c)
mt.show()
# Display our airbnb data
df
# Encode categorical variables using pandas' factorize method
df['neighbourhood_group'] = pd.factorize(df['neighbourhood_group'])[0]
df['neighbourhood'] = pd.factorize(df['neighbourhood'])[0]
df['room_type'] = pd.factorize(df['room_type'])[0]
# Display our airbnb data
df
# We select the numerical columns that we  want to scale
numerical_columns = ['neighbourhood_group','neighbourhood','price',
'minimum_nights', 'number_of_reviews', 'reviews_per_month',
                    'calculated_host_listings_count',
'availability_365','latitude','longitude','room_type']
for column in numerical_columns:
    mean = df[column].mean()
    std = df[column].std()
    df[column] = (df[column]-mean)/std
# Display our airbnb data
df
# Split our dataset into data and target
data = df.drop('longitude', axis=1)
target = df['price']
# Split our dataset into xtrain, xtest, ytrain and  ytest
xtrain, xtest, ytrain, ytest = train_test_split(data,target, test_size =0.2,random_state = 95)
print("The shape of xtrain is =", xtrain.shape)
print("The shape of xtest is =", xtest.shape)
print("The shape of ytrain is =", ytrain.shape)
print("The shape of ytest is", ytest.shape)
# Implement Xgboost with cross validation
dtrain = xgb.DMatrix(data, label=target)
dtest = xgb.DMatrix(xtest, label=ytest)
p = {
    'max_depth': 3,
    'min_child_weight': 1,
    'eta': 0.3,
    'subsample': 1,
    'colsample_bytree': 1,
    'gamma': 0,
    'reg_lambda': 1,
    'reg_alpha': 0,
    'silent': 1,
    'objective': 'reg:squarederror',
    'eval_metric': 'rmse',
}
cv_results = xgb.cv(
    params=p,
    dtrain=dtrain,
    num_boost_round=1000,
    nfold=5,
    metrics='rmse',
    early_stopping_rounds=50,
    verbose_eval=True
)
num_rounds=1000
ans = xgb.train(p, dtrain, num_rounds, [(dtrain,'train'),
(dtest,'test')],early_stopping_rounds=50, evals_result={})

preds = ans.predict(dtest)

print("\nCross-validation RMSE:", cv_results["test-rmse-mean"].tail(1).values[0])
# Print all metrics scores
print('Mean Absolute Error:', metrics.mean_absolute_error(ytest, preds))
print('Mean Squared Error:', metrics.mean_squared_error(ytest, preds))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(ytest, preds)))
print('R2 score:', r2_score(ytest, preds))
# Visulaize the graph for Actual values vs Predicted values
mt.figure(figsize=(12, 6), facecolor='black')
sns.scatterplot(x=ytest, y=preds, alpha=0.2, color='black')
mt.title('Actual vs. Predicted', color='white')
mt.xlabel('Actual', color='white')
mt.ylabel('Predicted', color='white')
mt.plot([min(ytest), max(ytest)], [min(ytest), max(ytest)], color='red')
mt.show()
# List to all alpha values
lambda_values_list = [0.01,0.1,1.0,10.0,100]
# Converting DataFrame to a list of lists and list
data_Xtrain = xtrain.values.tolist()
data_ytrain = ytrain.values.tolist()
data_Xtest = xtest.values.tolist()
data_ytest = ytest.values.tolist()
# My Custom Ridge Regression
class MyCustomRidgeRegression:
  def __init__(self, lambda_=1.0):
    self.lambda_ = lambda_
    self.weights = None
  def fit(self, data_X, data_y):
    m, n = len(data_X), len(data_X[0])
    X_bias = [[1] + x_row for x_row in data_X]
    regularization_matrix = [[0] * (n + 1) for _ in range(n + 1)]
    for i in range(n + 1):
      regularization_matrix[i][i] = self.lambda_ if i != 0 else 0
    X_bias_transpose_X_bias = [[0] * (n + 1) for _ in range(n + 1)]
    for i in range(n + 1):
      for j in range(n + 1):
        for k in range(m):
          X_bias_transpose_X_bias[i][j] += X_bias[k][i] * X_bias[k][j]
    inverse_matrix = [[0] * (n + 1) for _ in range(n + 1)]
    for i in range(n + 1):
            for j in range(n + 1):
                if i == j:
                    inverse_matrix[i][j] =
1/(X_bias_transpose_X_bias[i][j]+regularization_matrix[i][j])
                else:
                    inverse_matrix[i][j] = -
X_bias_transpose_X_bias[i][j]/(X_bias_transpose_X_bias[i][i]*
X_bias_transpose_X_bias[j][j])
        X_bias_transpose_y = [0] * (n + 1)
        for i in range(n + 1):
            for k in range(m):
                X_bias_transpose_y[i] += X_bias[k][i] * data_y[k]
        self.weights = [0] * (n + 1)
        for i in range(n + 1):
            for j in range(n + 1):
                self.weights[i] += inverse_matrix[i][j] * X_bias_transpose_y[j]

    def predict(self, data_X):
        X_bias = [[1] + x_row for x_row in data_X]
        predictions = [0] * len(data_X)
        for i in range(len(data_X)):
            for j in range(len(self.weights)):
                predictions[i] += X_bias[i][j] * self.weights[j]
        return predictions

    def cross_validate(self, data_X, data_y, lambdas, num_folds=5):
        best_lambda = None
        best_mse = float('inf')
        fold_size = len(data_X) // num_folds
        for lambda_ in lambdas:
```

```
        total_mse = 0.0
        for fold in range(num_folds):
            valid_start = fold * fold_size
            valid_end = (fold + 1) * fold_size
            valid_X = data_X[valid_start:valid_end]
            valid_y = data_y[valid_start:valid_end]
            train_X = data_X[:valid_start] + data_X[valid_end:]
            train_y = data_y[:valid_start] + data_y[valid_end:]
            self.lambda_ = lambda_
            self.fit(train_X, train_y)
            y_pred = self.predict(valid_X)
            mse = sum([(y_pred[i] - valid_y[i]) ** 2 for i in range(len(valid_y))]) / len(valid_y)
            total_mse += mse
        average_mse = total_mse / num_folds
        if average_mse < best_mse:
            best_mse = average_mse
            best_lambda = lambda_
    return best_lambda, best_mse

def calculate_metrics(self, y_true, y_pred):
    n = len(y_true)
    mse = sum([(y_pred[i] - y_true[i]) ** 2 for i in range(n)]) / n
    rmse = math.sqrt(mse)
    mae = sum([abs(y_pred[i] - y_true[i]) for i in range(n)]) / n
    mean_y = sum(y_true) / n
    ss_total = sum([(y - mean_y) ** 2 for y in y_true])
    ss_residual = sum([(y_true[i] - y_pred[i]) ** 2 for i in range(n)])
    r2 = 1 - (ss_residual / ss_total)
    return mse, rmse, mae, r2
```

```
main = MyCustomRidgeRegression()

best_lambda, _ = main.cross_validate(data_Xtrain, data_ytrain, lambda_values_list)

main.lambda_ = best_lambda

main.fit(data_Xtrain, data_ytrain)

y_pred = main.predict(data_Xtest)

rrmse, rrrmse, rrmae, rrr2 = main.calculate_metrics(data_ytest, y_pred)

print('Mean Absolute Error:',rrmae )

print('Mean Squared Error:',rrmse )

print('Root Mean Squared Error:',rrrmse)

print('R2 score:',rrr2)

metrics = ['MAE', 'MSE', 'RMSE', 'R2']

scores_model1 = [xgbmae, xgbmqe, xgbrmse, xgbr2]

scores_model2 = [rrmae, rrmse, rrrmse, rrr2]

fig, ax = mt.subplots(figsize=(10, 6))

bar_width = 0.35

index = range(len(metrics))

bar1 = ax.bar(index, scores_model1, bar_width, label='Xgboosts')

bar2 = ax.bar([i + bar_width for i in index], scores_model2, bar_width, label='Ridge
Regression')

for p in ax.patches:

   p.set_edgecolor('white')

   p.set_linewidth(1)

ax.set_facecolor('black')

ax.set_xlabel('Metrics')

ax.set_ylabel('Scores')

ax.set_title('Performance Comparison of Two Models')

ax.set_xticks([i + bar_width / 2 for i in index])

ax.set_xticklabels(metrics)

ax.legend()
```

```
mt.tight_layout()

mt.show()
```