Melbourne Housing Price Prediction

Problem Statement:

In this project, we will be trying to develop a model which can accurately estimate and predict the price of the residential houses in Melbourne, Australia. The real estate business is of a dynamic nature and a vital aspect of any economy. Home buyers need to make an informed decision before buying any property based on the neighborhood and market reach. Accurately predicting the price of the property is crucial for all the stakeholders involved, namely buyers and sellers. In recent times, the real estate industry increasingly relies on data-driven insights to make a decision. As a result, the need for robust predictive estimation model has increased to harness the abundant data available to generate precise predictions. This dataset provided us with an opportunity to leverage Machine Learning techniques to build a predictive model that can estimate property price considering various factors. With this project, we will be able to discover patterns and relationships within the data, meaning, we will be able to identify which features strongly influence the house prices, which suburbs would be the best to buy, and which area is the expensive. This also helps the real estate agents to provide recommendations to their clients. Our project aims to contribute to the domain by improving accuracy, offering data-backed insights, and ultimately ease stakeholders to make informed decisions in the real estate market.

End Users: Buyers, Sellers, and Real Estate Agents

Data Source:

This dataset has been taken from the Kaggle website, which is a snapshot of Melbourne Housing Dataset.

The link of the dataset: Melbourne Housing Snapshot (kaggle.com)

The dataset comprises of 21 features, which includes features like Address, Suburb, Types of Real Estate, Rooms, Real Estate Agent, Price, Date of Sale, Distance from CBD, etc.

The dataset contains 13580 rows before performing any cleaning or preprocessing.

Data Cleaning/Preprocessing:

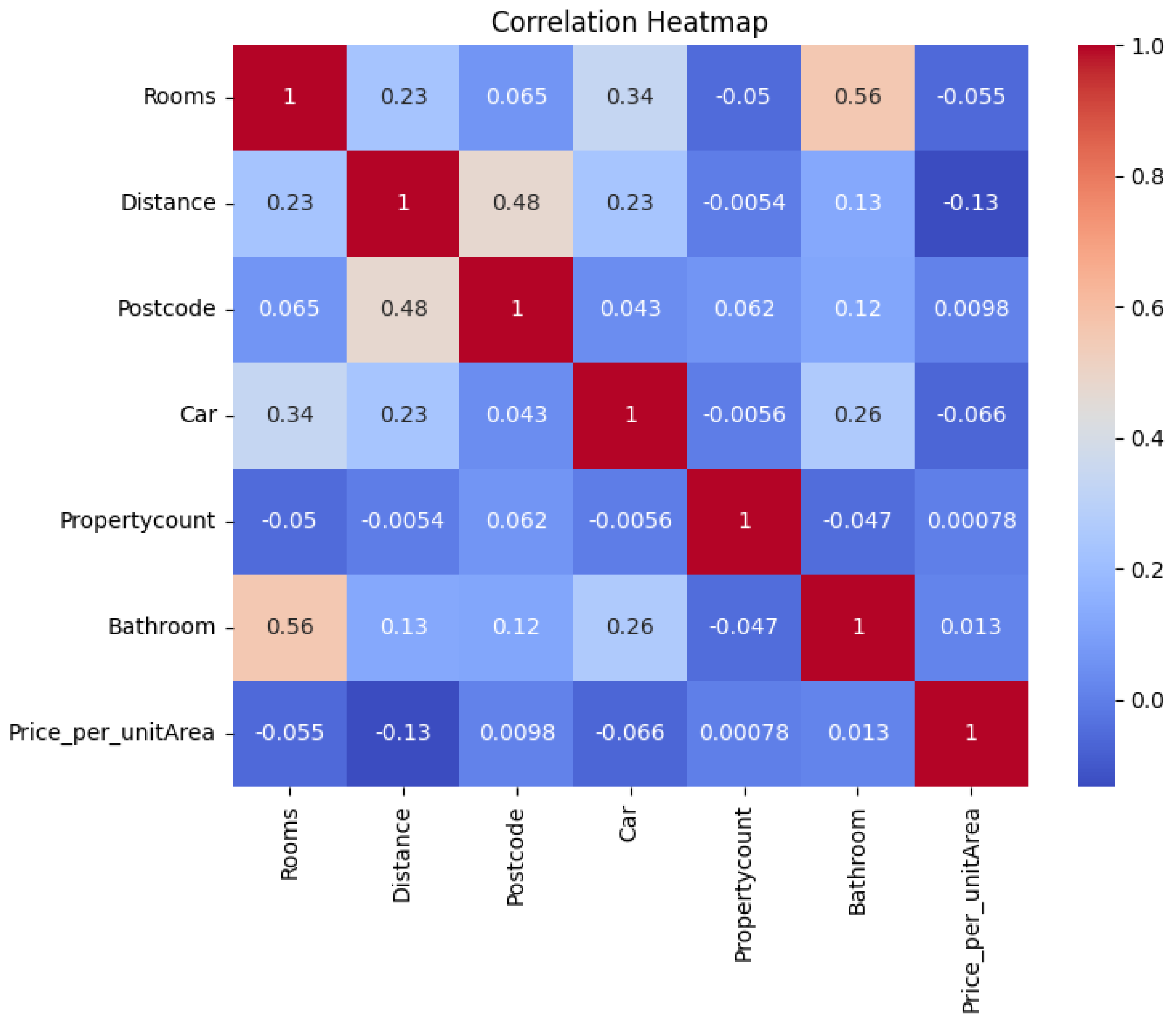
Note: The data cleaning steps from 7 until 12 were performed after the EDA steps, because we wanted to visualize and analyze the data first and then understand where to perform the data cleaning and preprocessing.

1. Firstly, we would like to understand the data. So, we generated the summary statistics for statistical analysis of the dataset.
2. Next, we tried to find if the dataset contains any NULL values. We found that the columns '*Car*', '*BuildingArea*', '*YearBuilt*', and '*CouncilArea*' contained null values.

    - We dropped the columns ‘YearBuilt’, ‘BuildingArea’, and ‘CouncilArea’ due to the presence of large number of missing values.
    - For the column ‘Car’, we replaced the null values by the values which occur the most often for the column.
3. There were few features which were of Object data type. So, we converted those to Categorical type.
4. The Date column converted to Datetime type from Object datatype.
5. Next, we tried to detect if there were any outliers in the dataset. For this, first we computed the interquartile range and got the lower and upper bound. The features containing outliers were found if the feature’s data was less than lower bound or greater than the upper bound.
6. In the next step, we removed the outliers from the features one-by-one.
    - For the ‘Price’ and ‘Landsize’ features, we kept the values which were only between the lower and upper bound.
7. Next, we encoded the categorical data features {‘Suburb’, ‘Address’, ‘Type’, ‘Method’, ‘SellerG’, ‘CoucilArea’, and ‘Regionname’} with one hot encoding.
8. We then aimed to remove the duplicate columns. We noticed that the columns ‘Rooms’ and ‘Bedroom2’ were duplicate. So, we removed the column ‘Bedroom2’.
9. Additional feature engineering: There were few columns which were non-essential in the dataset. So, we dropped the columns *‘Latitude’* and *‘Longitude’* from the dataset as these features are irrelevant for house price prediction.
10. Dimensionality reduction of dataset: We aim to reduce the dimensionality of the dataset by merging multiple features into a single feature. We replaced the columns ‘Price’ and ‘Landsize’ with a new feature ‘Price_per_unit_area’. We will be predicting the price based on per unit area.
11. Normalize the dataset: We normalized the dataset for the features {*'Rooms','Distance','Postcode','Car','Propertycount', 'Price_per_unitArea' and ‘Bathroom’}* so that all the values appear to be in the range for a feature. We used min-max normalization over the data.
12. Next, we formatted the Date column and split it into *‘Year’, ‘Month’*, and *‘Day’*. Then we performed one hot encoding on *‘Year’* and *‘Month’* columns and normalized the *‘Day’* column.

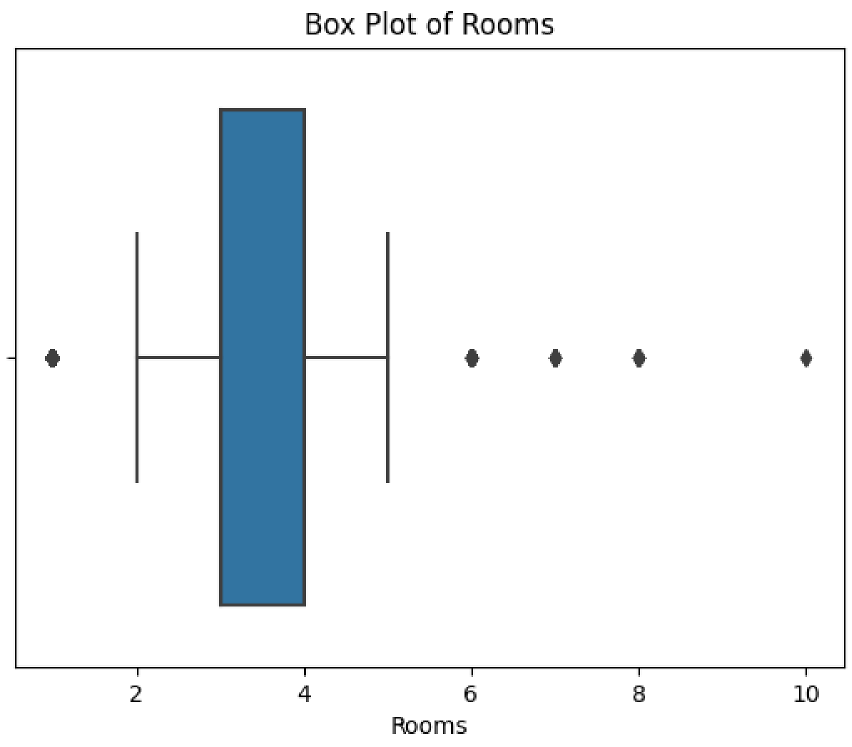
Exploratory Data Analysis:

1. First, we created a Correlation Matrix between the features which were continuous to understand the relationship between them and then created a heatmap for the same.

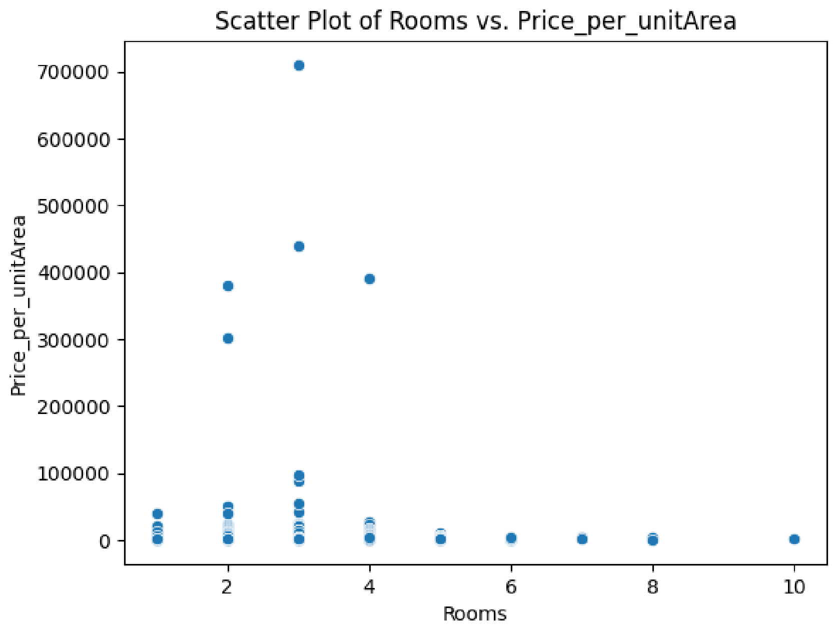


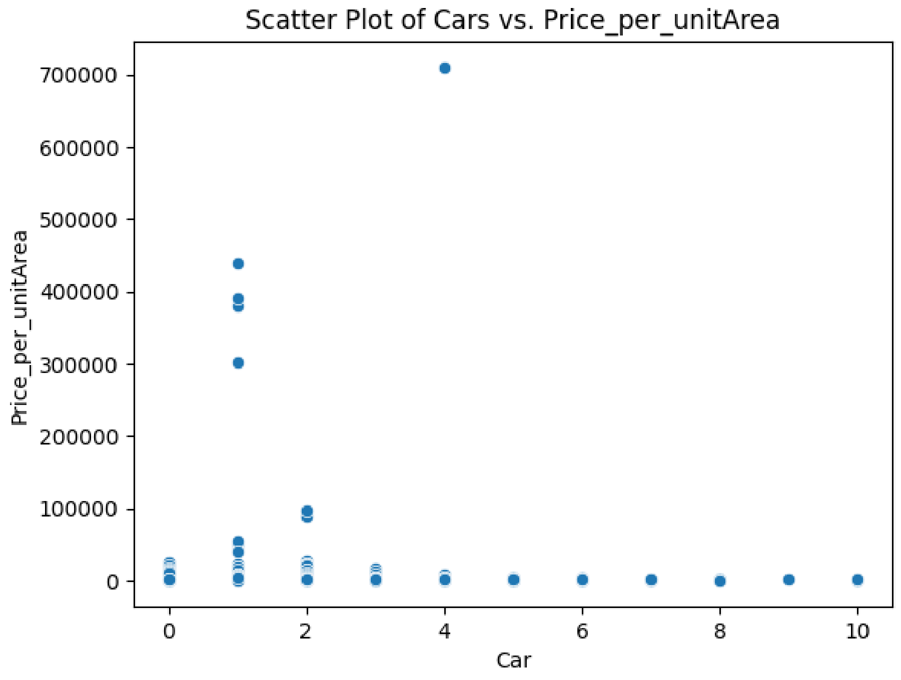
	Rooms	Distance	Postcode	Car	Propertycount	Bathroom	Price_per_unitArea
Rooms	1	0.23	0.065	0.34	-0.05	0.56	-0.055
Distance	0.23	1	0.48	0.23	-0.0054	0.13	-0.13
Postcode	0.065	0.48	1	0.043	0.062	0.12	0.0098
Car	0.34	0.23	0.043	1	-0.0056	0.26	-0.066
Propertycount	-0.05	-0.0054	0.062	-0.0056	1	-0.047	0.00078
Bathroom	0.56	0.13	0.12	0.26	-0.047	1	0.013
Price_per_unitArea	-0.055	-0.13	0.0098	-0.066	0.00078	0.013	1

2. Initially we did not remove the outliers from the 'Rooms' feature while performing data cleaning. So, we plotted the Box Plot for 'Rooms' column.

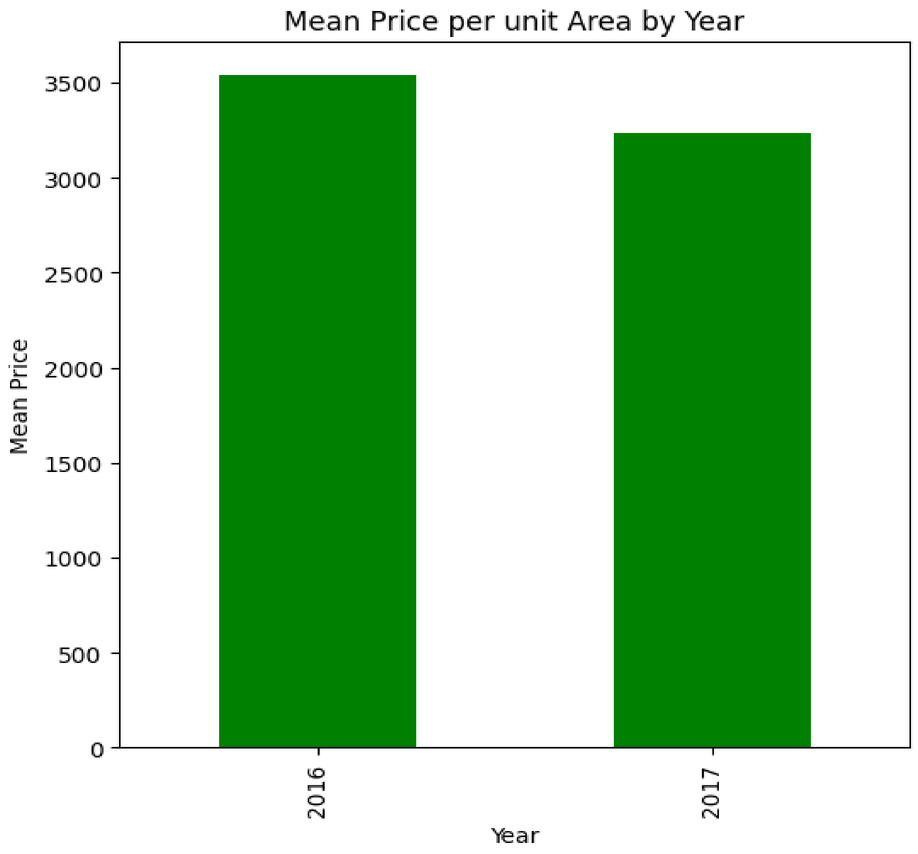


3. Next, we wanted to visualize to check what is the trend when the number of rooms increase in a house. Does the price per unit area increase? Based on the assumption that the price per unit might increase, we plotted the scatter plot for the 'Rooms' vs 'Price_per_unitArea'. We found that with more number of rooms, the house price will not increase. We also plotted the scatter plot between the 'Car' feature and the Price_per_unitArea. Our conclusion was the same that solely based on a greater number of cars, the price does not increase.

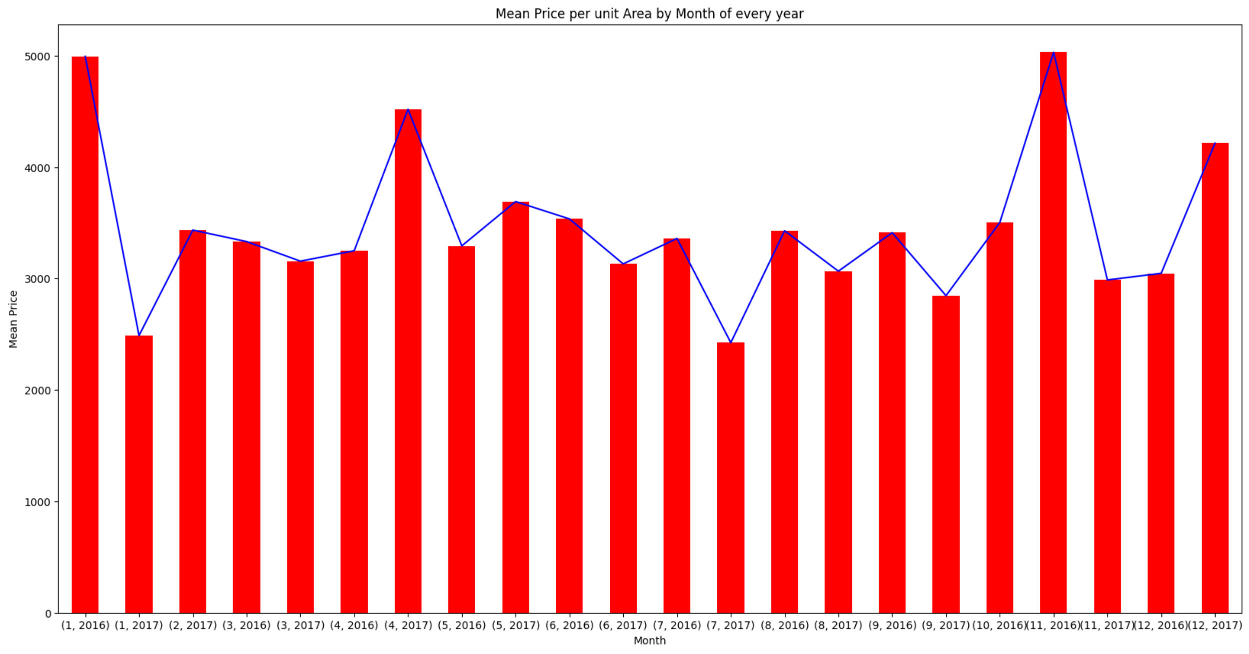




4. We then visualized the trend of housing price based on year. For this, we calculated the average price per unit area by year and plotted the bar graph. We observed that the average price per unit in the year '2016' was greater than in '2017'.

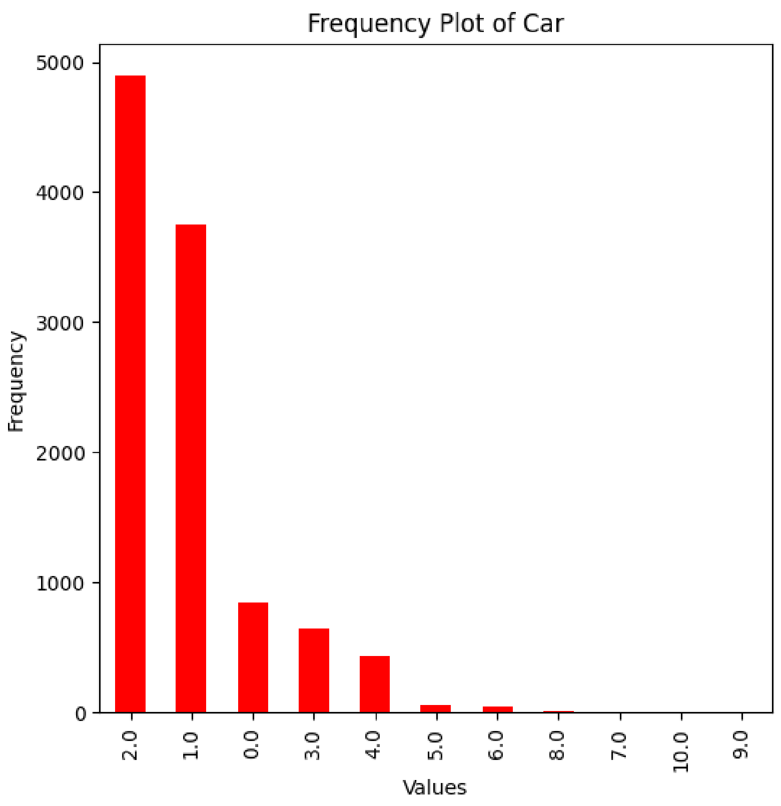


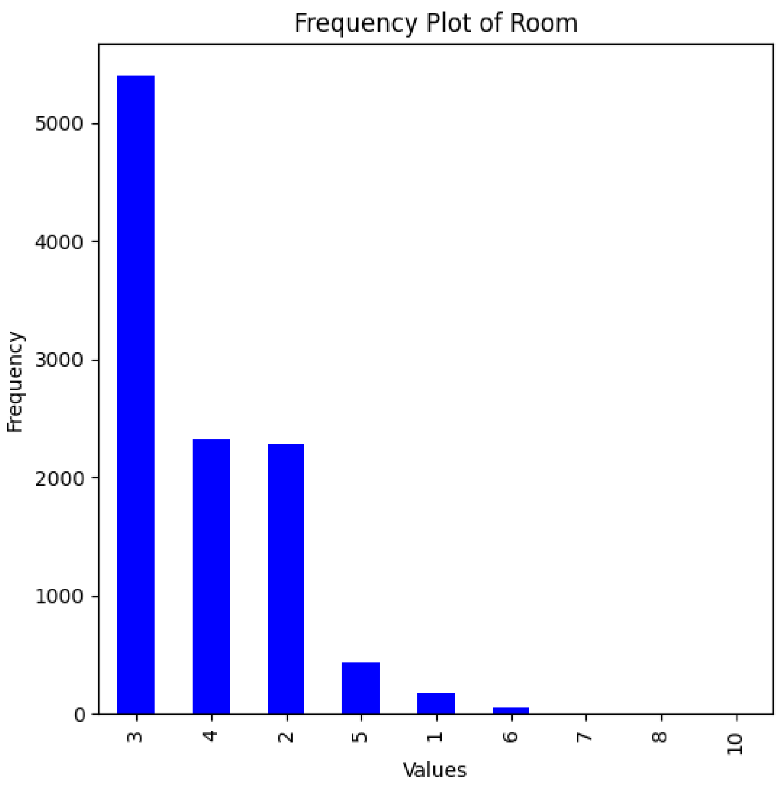
5. Next, we visualized the monthly trend of the house price throughout year and plotted bar graph for the same. We observed that the average price per unit area was highest in January 2016 and November 2016.



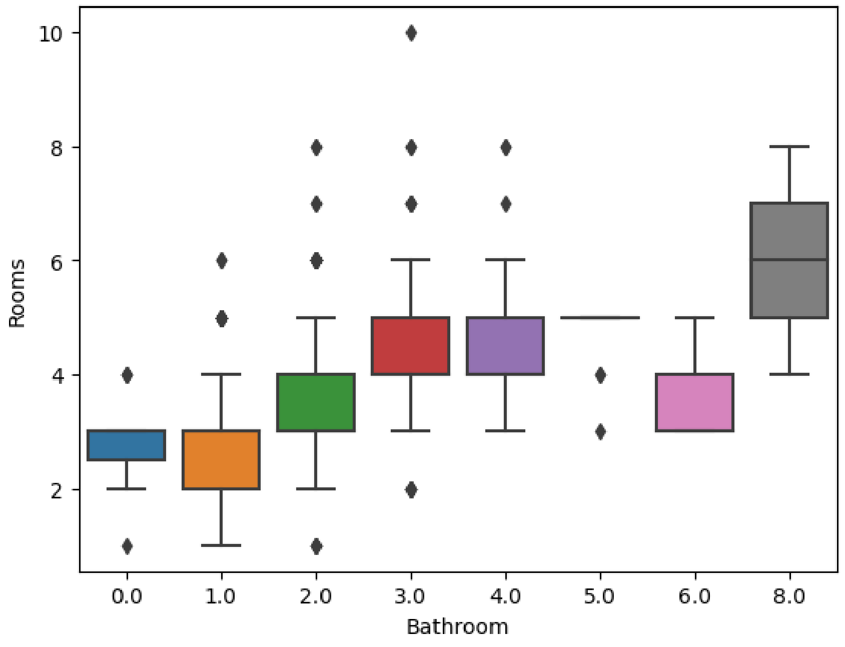
6. Univariate Analysis: In the data cleaning for ‘Car’ feature, we used the mode to fill in the missing values. So, we checked for the skewness of ‘Car’ and for certain other features as well. The skewness value for Car, Rooms and Bathroom features are: 1.24, 0.52, 1.18 respectively. Hence, they are all right skewed.

   We plotted a frequency plot of ‘Car’ column and found that most of the houses have space for 2 cars. We also plotted frequency plot for the ‘Rooms’ feature and observed that most of the houses have 3 rooms.

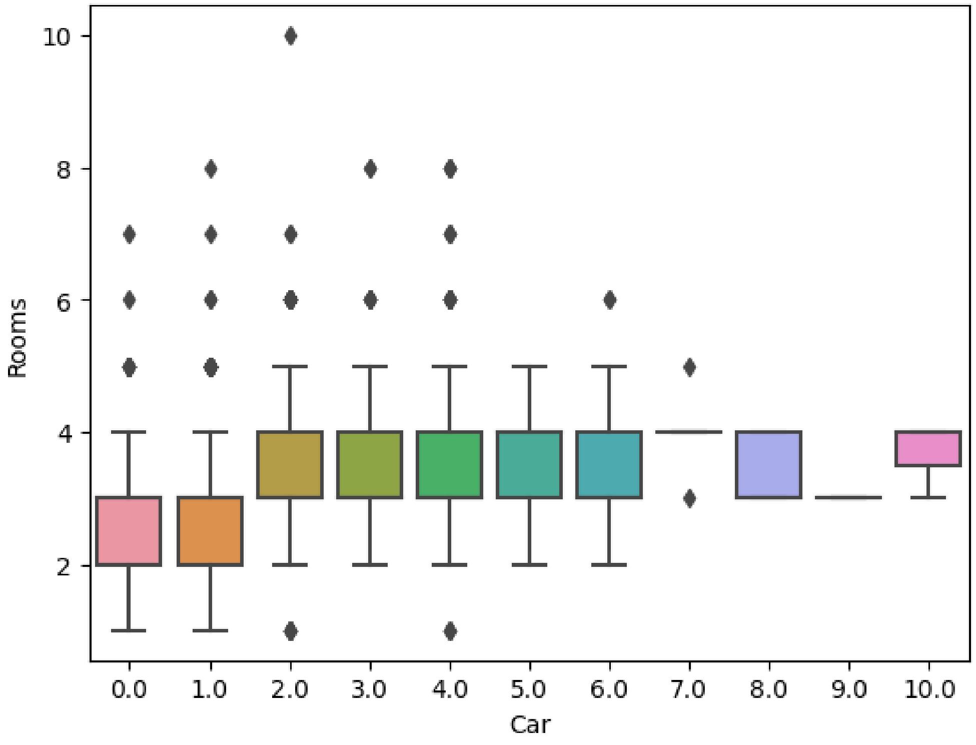

Frequency Plot of Car
Frequency
Values
0
1000
2000
3000
4000
5000
2.0
1.0
0.0
3.0
4.0
5.0
6.0
8.0
7.0
10.0
9.0


Frequency Plot of Room
Frequency
Values
0
1000
2000
3000
4000
5000
3
4
2
5
1
6
7
8
10

7. Bivariate Analysis: After thoroughly understanding the correlation matrix, we observed that the features 'Rooms' and 'Bathrooms' and 'Rooms' and 'Car' were highly correlated. We plotted box plot for those 2 correlated features.

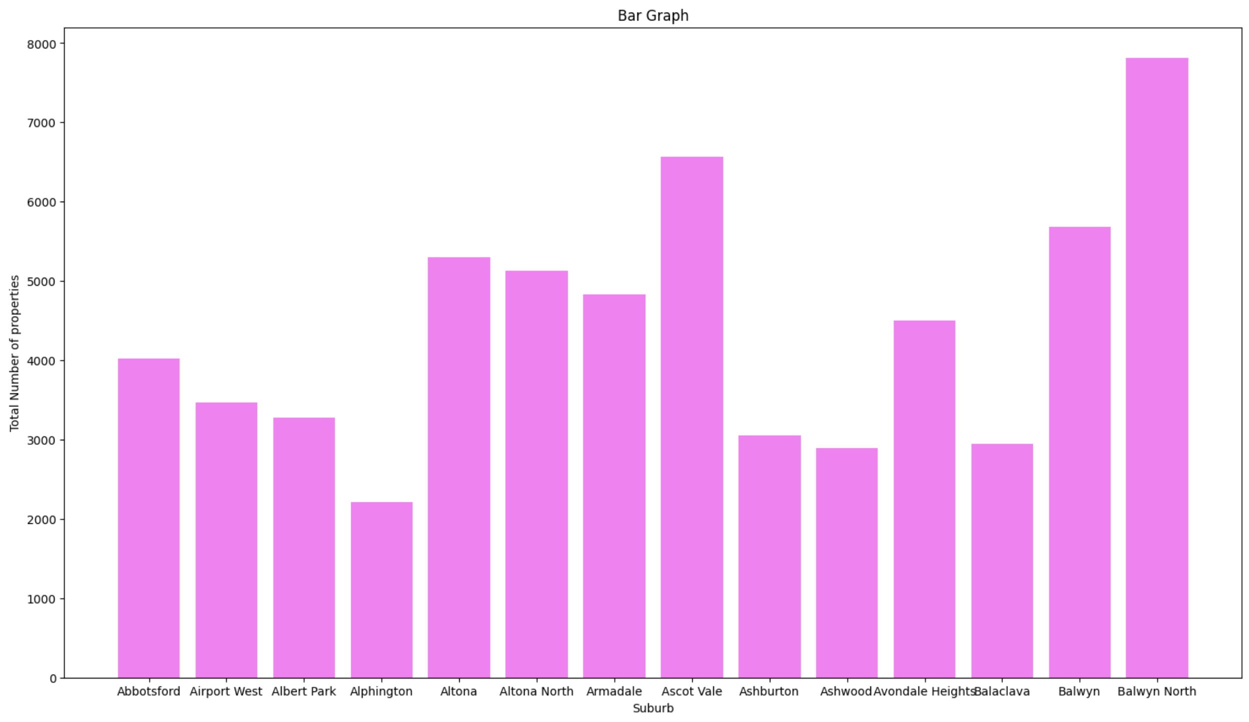


From the above plot, we notice that most of the data points follow consistent relationship between the two features but there are few points which show abnormality.

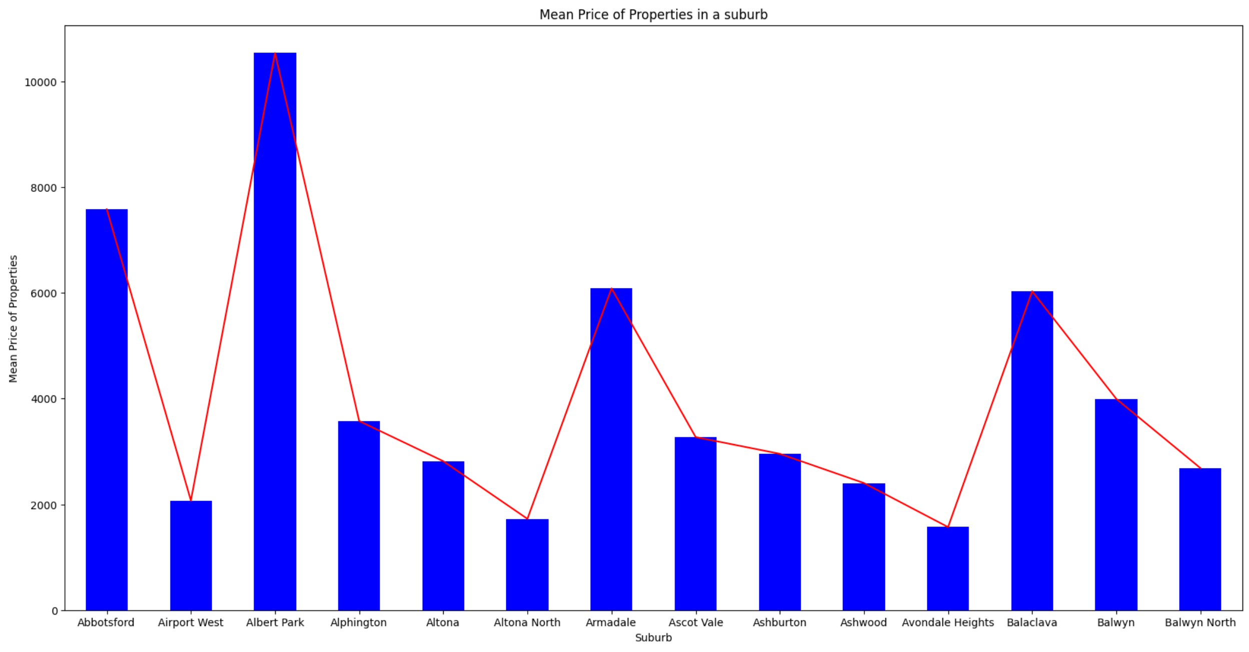


From the above plot, we notice that there are more points which show abnormality between these two features compared to the above plot between 'Rooms' and 'Bathrooms'.

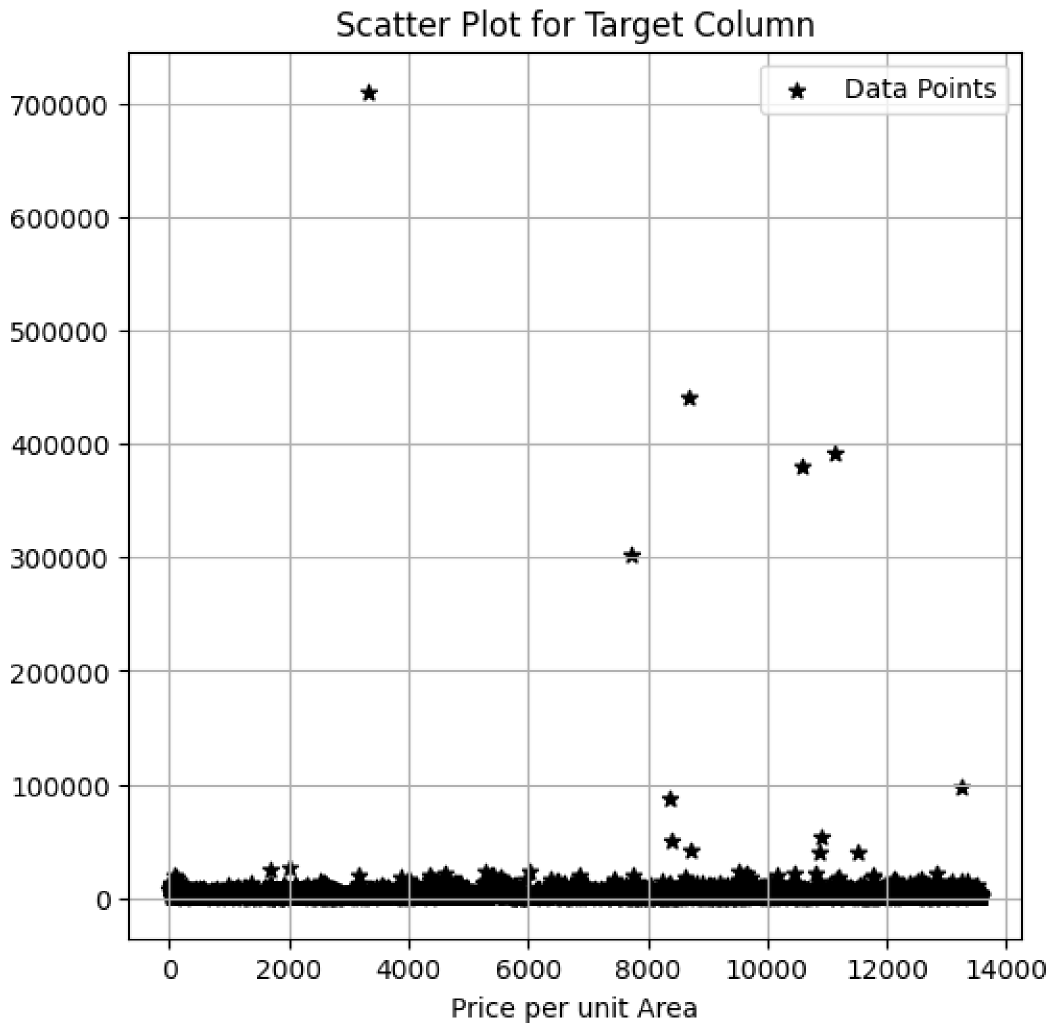
8. We then sought to analyze the number of properties in a Suburb. For this, we selected a subset of the dataframe. We extracted the unique suburb and property counts and plotted them on a bar graph. We observed that the Suburb 'Balwyn North' had the highest total number of properties.



9. Next, we analyzed the mean price of properties in a suburb. We calculated the mean price of properties for each unique suburb. We observed that the Suburb 'Albert Park' had the highest mean price.



10. We next visualized the target feature, which is 'Price_per_unitArea'. We plotted the scatter plot of the data point with Price per unit area on the x-axis.



This plot helps us to understand the distribution of our target feature when it is not normalized.

Significance of performing EDA Operation:

- After performing EDA, it helped us to better understand the dataset.
- We got to know the correlation between different features of the data.
- With the scatter plots, we realized in which features we need to remove outliers.
- With the EDA steps, we understood the importance of scaling the data.
- From the correlation matrix, we found the relevant features which will be useful to perform further steps in this project and thus ignoring the less significant features.
- After performing EDA, we have reiterated the cleaning steps to get the most out of the given dataset.

Analysis of the Algorithms Implemented:

We have applied the following regression algorithms to predict the housing price per unit area based on the input features.

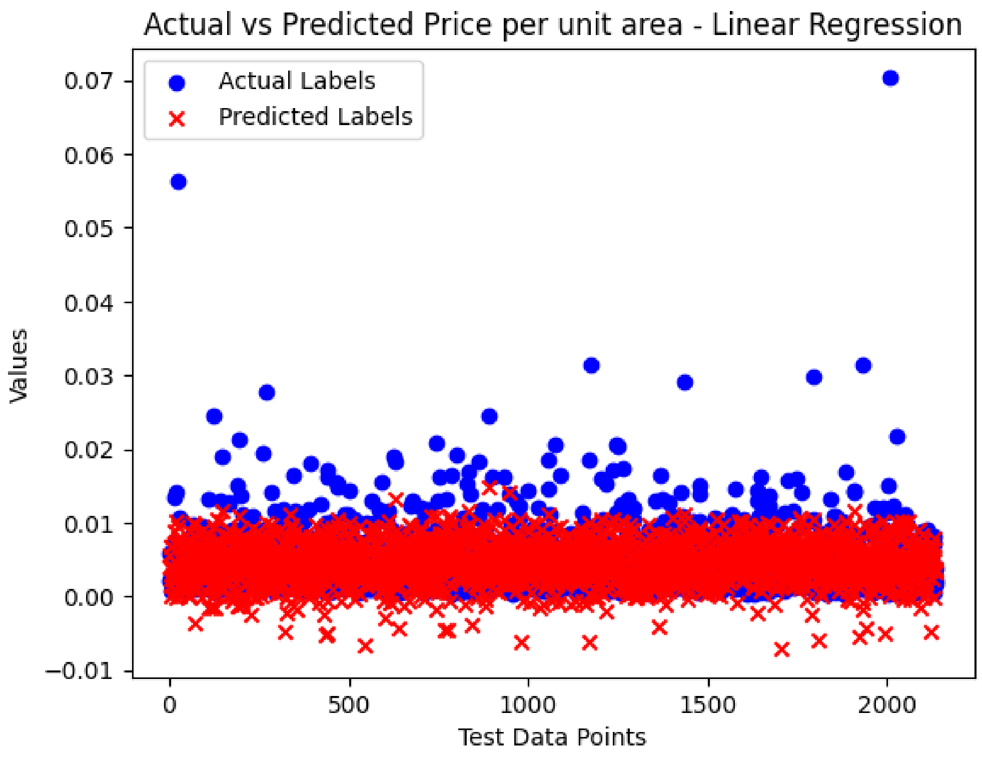
1. **Linear Regression:**

Linear Regression is a simple algorithm which assumes a linear relationship between the features and the output variable.

Why Linear Regression? It is the simplest model to perform regression task. Hence, we started with this algorithm to create baseline performance.

We trained our model using the training data with this algorithm and then utilized the trained model to make predictions on the test dataset. We then calculated the mean squared error by comparing the predicted values with the target values to evaluate the model's performance. The mean squared error came out to be 1.0804823014909359e-05, which indicates that the model's predictions are close to the target value. We calculated the regression score (R-score) which measures the goodness of fit. The R-score of this model was 0.34463503618834024. It suggests that this model explains only 34.46% of the variance in the target variable. After evaluating the model's performance, we concluded that the relationship between the features and target is not linear.

Visualization:



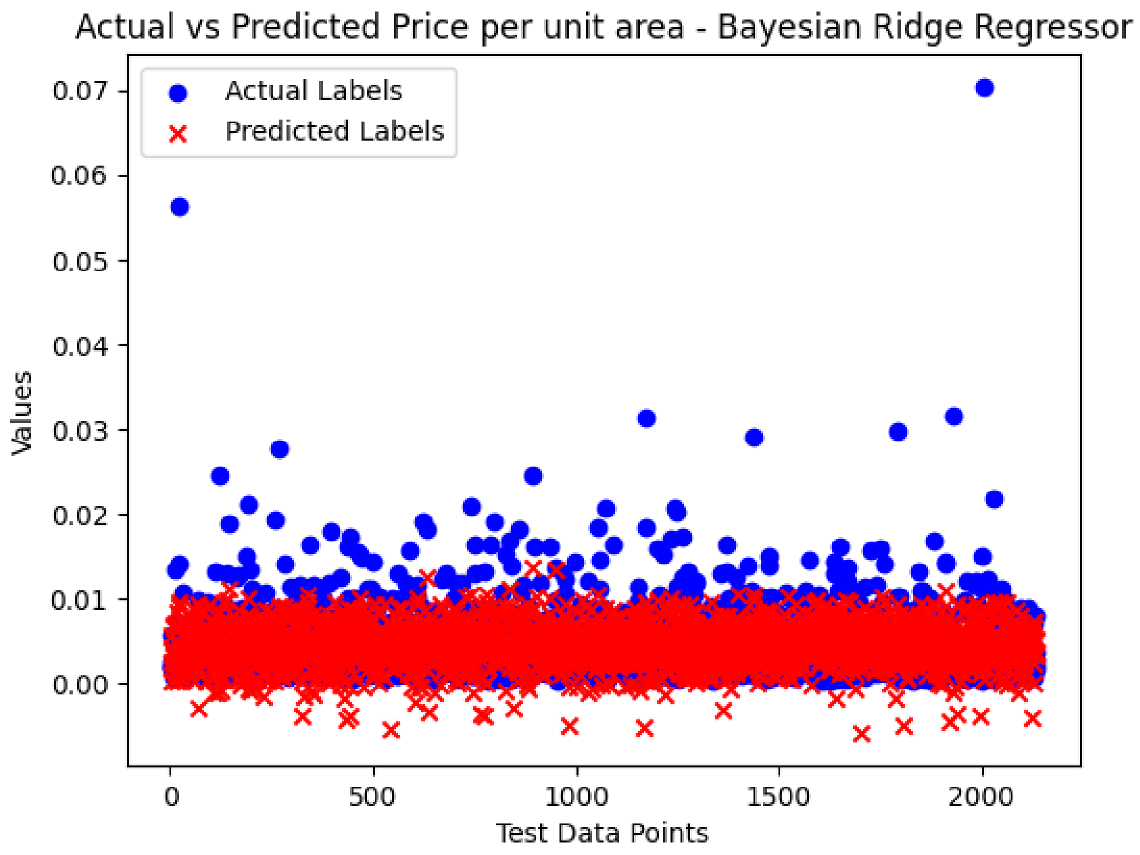
2. **Bayesian Ridge Regression:**

Bayesian Ridge regression algorithm is like Ridge regression which also includes a regularization term. The only difference in this algorithm is that it automatically adapts the regularization strength based on the complexity of the data. It provides a probabilistic framework that can estimate the uncertainty associated with the model predictions.

Why Bayesian Ridge Regression? Our dataset encompasses complex relationship amongst different features. So, Bayesian ridge regression seemed to be a good choice for our algorithm.

We trained our model using the training data and then predictions were made on the test dataset. To evaluate the model's performance, we calculated the mean squared error between the predicted value and the target value in the test set. The MSE value was 1.0781221516400757e-05, which is very low and approximately same as of linear regression model and ridge regression model. This also suggested that the model's predictions are close to the target values. The R-score value was 0.3454552519313868, which means that model can explain 34.54% of the variance in the target variable.

Visualization:



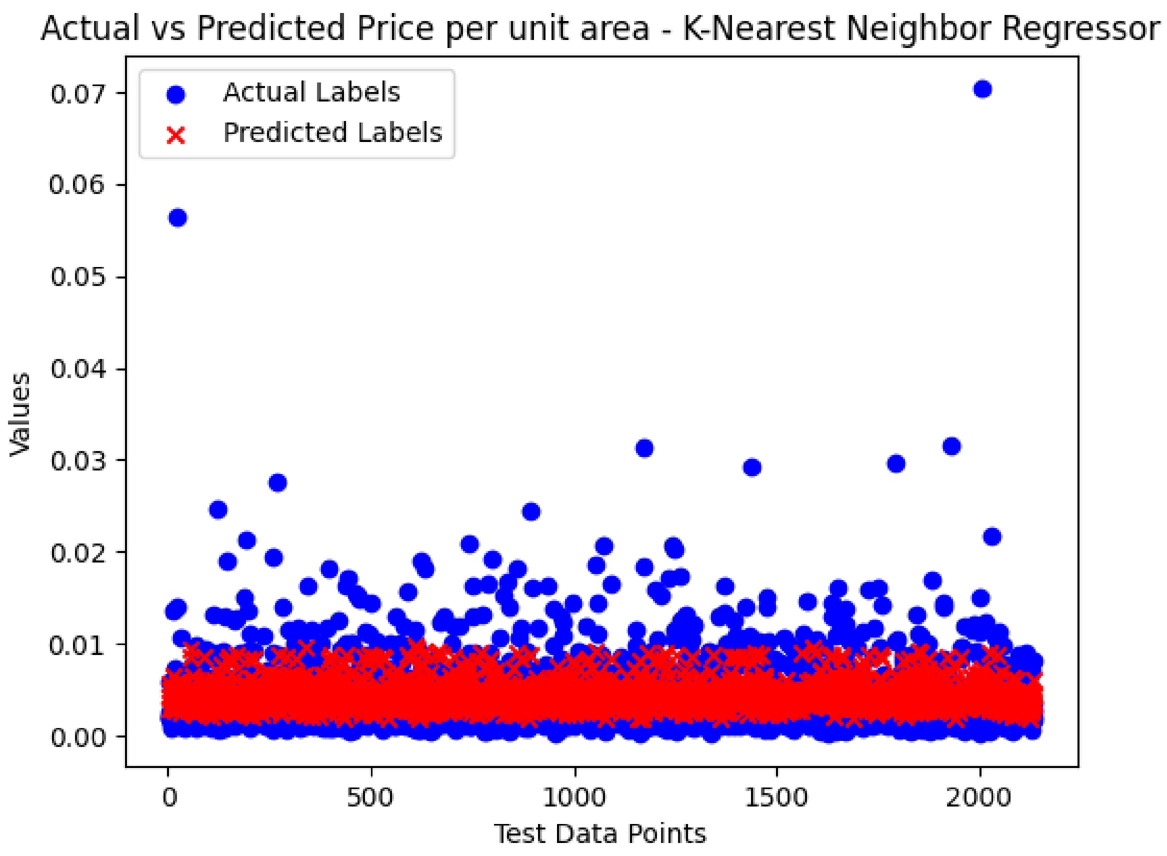
3. **K-Nearest Neighbors Regressor:**

K-Nearest Neighbors regression is a non-parametric, instance-based learning algorithm that predicts the target variable based on the majority vote of its k-nearest neighbors.

Why KNN Regressor? It is a simple algorithm, less sensitive to outliers and suitable for datasets with non-linear relationships.

The KNN regressor model is created and we set the number of neighbors (K) parameter to 223, which means that the model makes prediction using the majority vote considering the 223 nearest neighbors. The number of neighbors set to 223 has been determined after experimenting with different values of neighbors to obtain optimal value and hyper parameter tuning. The MSE for this algorithm is: 1.0764696923766554e-05 and R-score value: 0.3464584856842817, which means the model can explain 34.64% of the variance of the target variable.

Visualization:



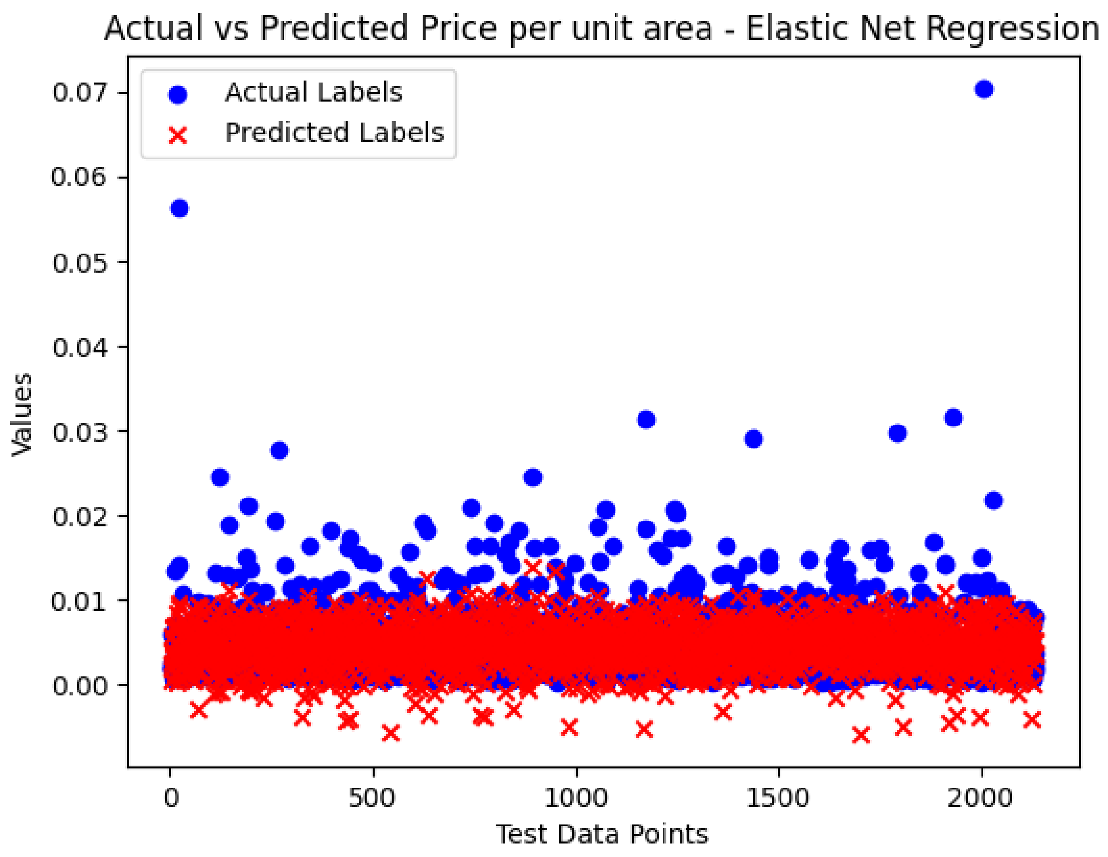
4. **Elastic Net:**

Elastic Net regression is a linear regression model that combines both L1 and L2 regularization.

Why ElasticNet? It works fairly well with datasets having correlated features. The features used in our dataset have varying levels of correlation, as was found out by performing EDA. As a result, we chose this algorithm for our dataset.

The hyperparameters used for this algorithm include- alpha: 0.001, l1_ratio: 0.001. Low l1_ratio suggests that the L2 regularization dominates over L1 regularization. The MSE value is 1.0799987847445448e-05 and the Regression score is 0.3443159187484879. This means that model can explain 34.43% of the variance of the target variable. We have experimented with a number of different alpha and l1_ratio parameters. A comparatively lower value of alpha works well for our dataset, suggesting that lower effect of regularization boosts the model's accuracy. Similarly, the low value of l1_ratio suggests that L2 regularization is more important in our case than L1 regularization, to boost accuracy.

Visualization:



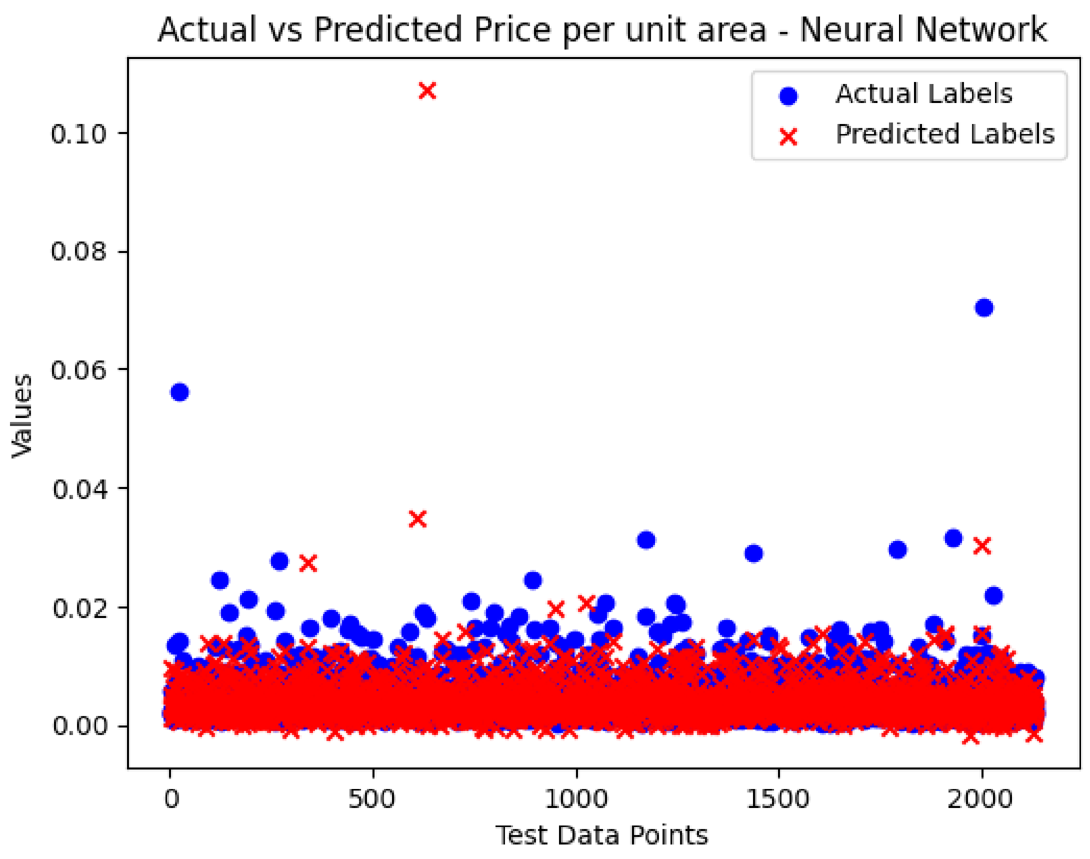
5. **Neural Network:**

Neural Network regression is an algorithm where Artificial Neural Network is used to model and predict continuous numerical values. It uses neurons to simulate the working of a human brain.

Why Neural Network? They can automatically learn complex patterns from data and can handle relationships between input variables and the target variables which are non-linear. Hence, it was an obvious choice for our dataset.

The basic neural network structure consists of input layer, hidden layer, and output layer with appropriate number of neurons per layer. We have used Adam optimizer, with ReLU activation function. The number of epochs for training: 50 with batch size of 32. After 50 epochs, we have got MSE: 8.924178501040513e-06 and R-score: 0.45819922540345226. A comparatively higher R-score from neural networks suggests that the features in our data have complex relationship and hence cannot be described effectively by linear regression models.

Visualization:



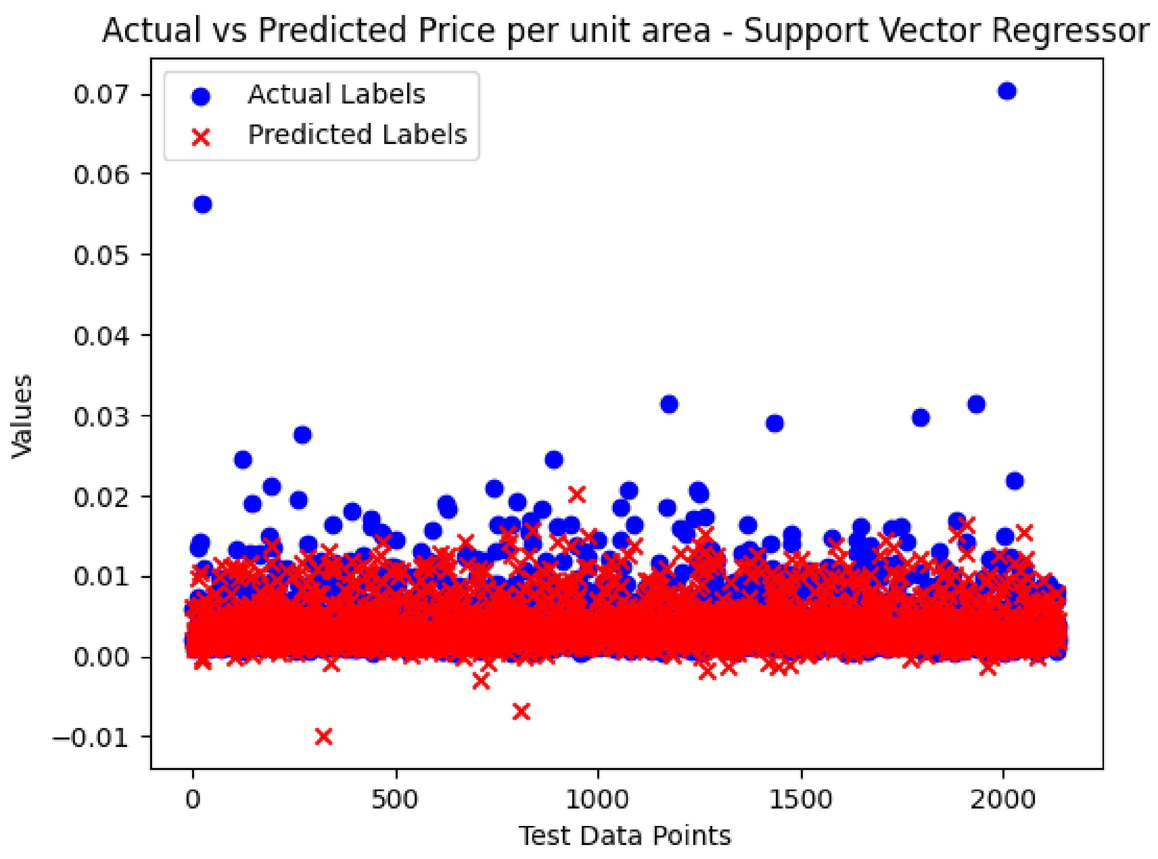
6. **<u>Support Vector Regression:</u>**

Support Vector Regression is an ML algorithm, belonging to the Support Vector Machine (SVM) class. Unlike SVM, SVR has been modified for regression task, where we try to find optimal hyperplane (with maximum margin) in higher dimension to represent the relationship between the attributes and the output variable.

Why SVR? SVR works well when the features have complex non-linear relationship amongst each other. This in fact, suits well with the nature of our dataset and was further validated by the comparatively underwhelming performances by traditional linear regression models. Hence, this model was well suited for our task.

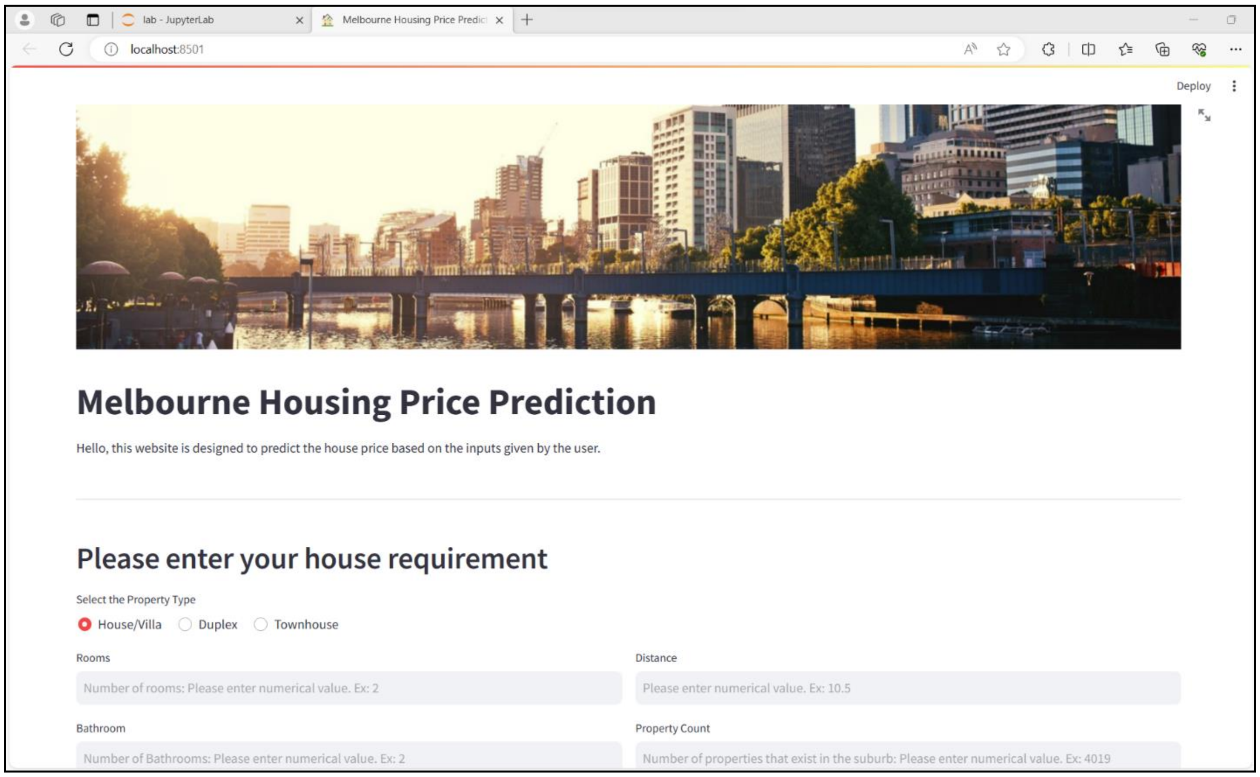
And as expected, SVR, outperformed all other algorithms previously tested. The objective function involves a loss term and a regularization term. The values of hyperparameters are: C=1 and epsilon = 0.001. These hyperparameters were tuned to get maximum possible accuracy. Also, the chosen kernel function is polynomial. The MSE: 8.813359119317844e-06 and the R-score is: 0.4649272426490343. Hence, this model is the best performing model of the lot. This means that model can explain 46.49% of the variance of the target variable.

Visualization:



Description of the product:

In this project, we will be trying to develop a model which can accurately estimate and predict the price of the residential houses in Melbourne, Australia. The real estate business is of a dynamic nature and a vital aspect of any economy. Home buyers need to make an informed decision before buying any property based on the neighborhood and market reach. Accurately predicting the price of the property is crucial for all the stakeholders involved, namely buyers and sellers. Our product consists of a website using which users can input attributes about the house/property and our model working in the backend can take the input from the user, feed them into the model and fetch back the predicted price to the user. Additionally, our product gives additional recommendations to users regarding other properties lying within the range of users' input of different attributes regarding the property.



Melbourne Housing Price Prediction

Hello, this website is designed to predict the house price based on the inputs given by the user.

Please enter your house requirement

Select the Property Type

[✓ House/Villa] Duplex Townhouse

Rooms

Number of rooms: Please enter numerical value. Ex: 2

Distance

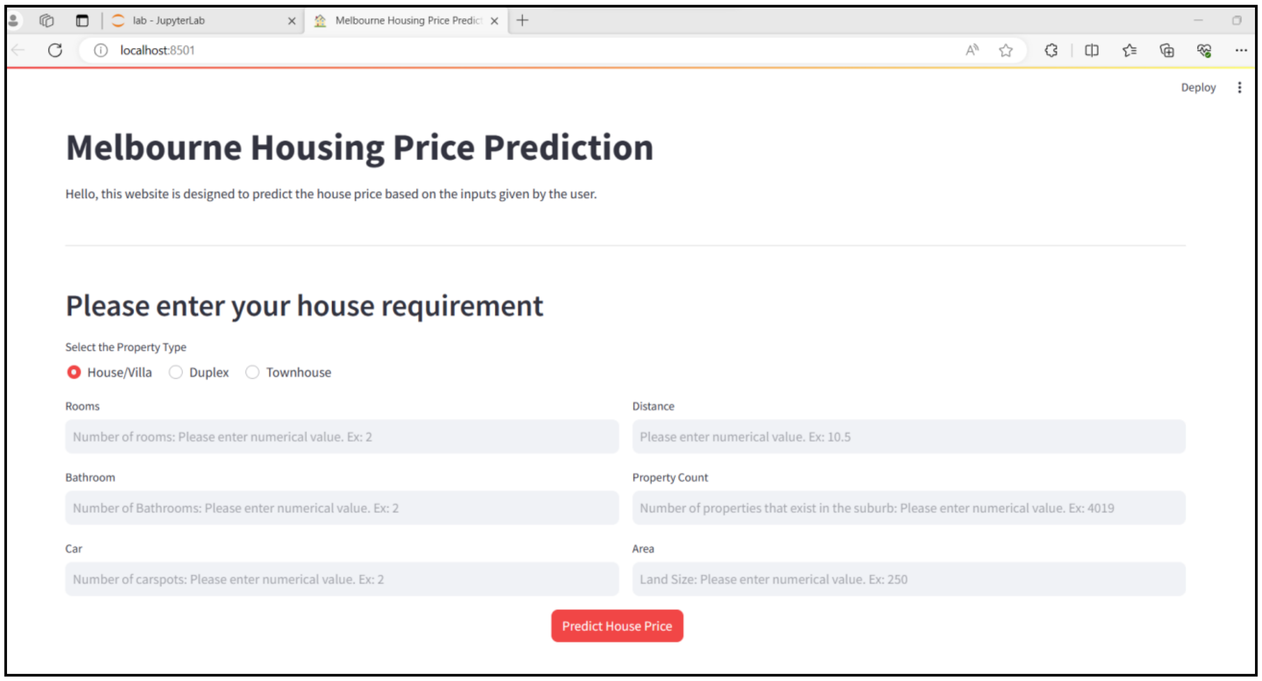
Please enter numerical value. Ex: 10.5

Bathroom

Number of Bathrooms: Please enter numerical value. Ex: 2

Property Count

Number of properties that exist in the suburb: Please enter numerical value. Ex: 4019



Melbourne Housing Price Prediction

Hello, this website is designed to predict the house price based on the inputs given by the user.

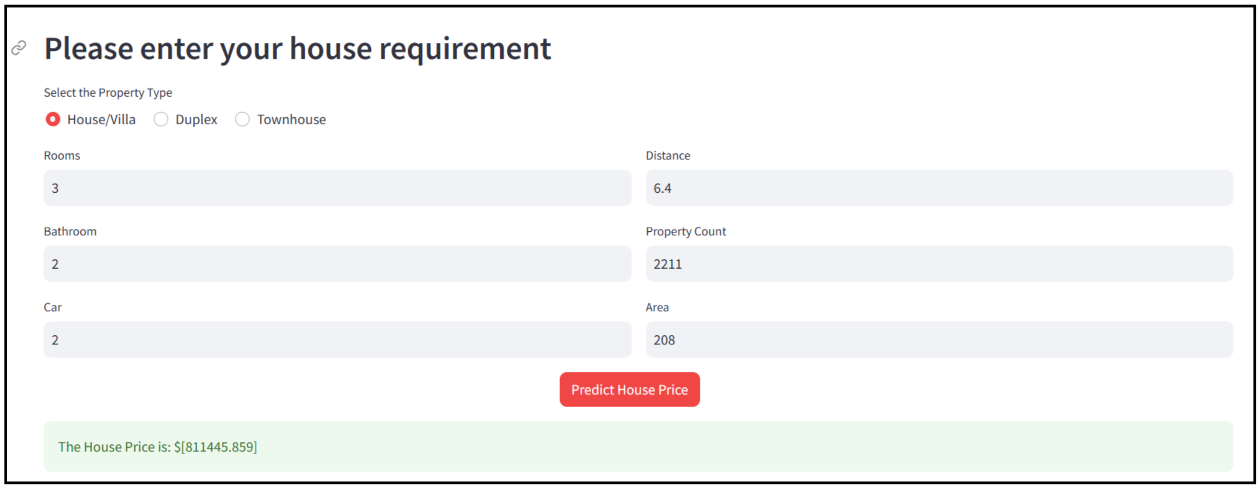
Please enter your house requirement

Select the Property Type

[✓ House/Villa] Duplex Townhouse

Rooms	Distance
Number of rooms: Please enter numerical value. Ex: 2	Please enter numerical value. Ex: 10.5
Bathroom	Property Count
Number of Bathrooms: Please enter numerical value. Ex: 2	Number of properties that exist in the suburb: Please enter numerical value. Ex: 4019
Car	Area
Number of carspots: Please enter numerical value. Ex: 2	Land Size: Please enter numerical value. Ex: 250

Property Type: House



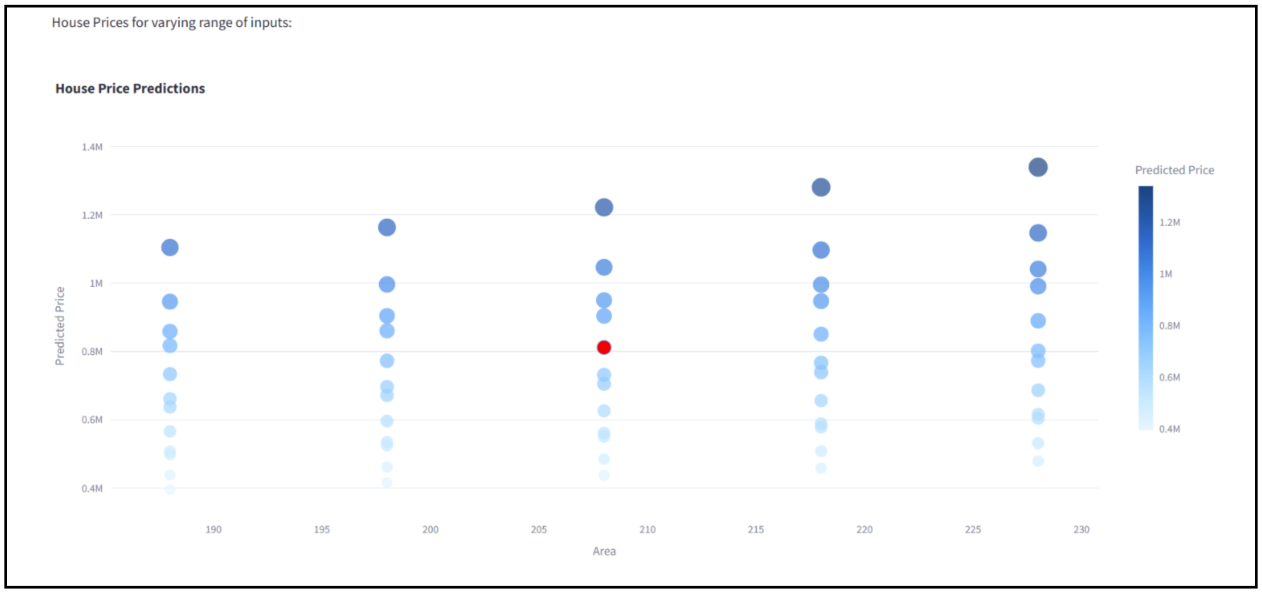
Please enter your house requirement

Select the Property Type

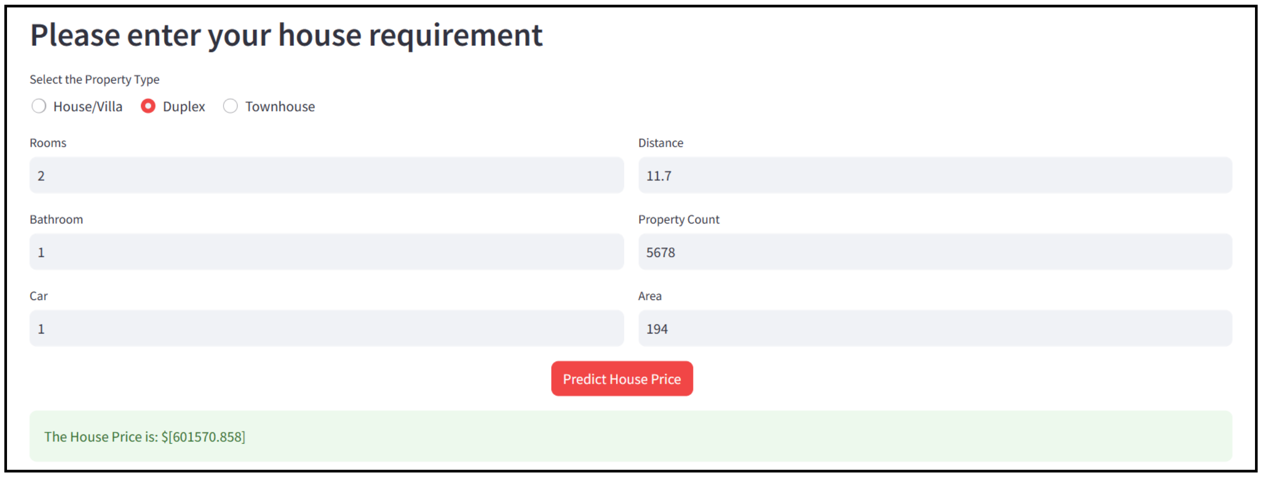
[✓ House/Villa] Duplex Townhouse

Rooms	Distance
3	6.4
Bathroom	Property Count
2	2211
Car	Area
2	208

The House Price is: $[811445.859]



Property Type: Duplex



Please enter your house requirement

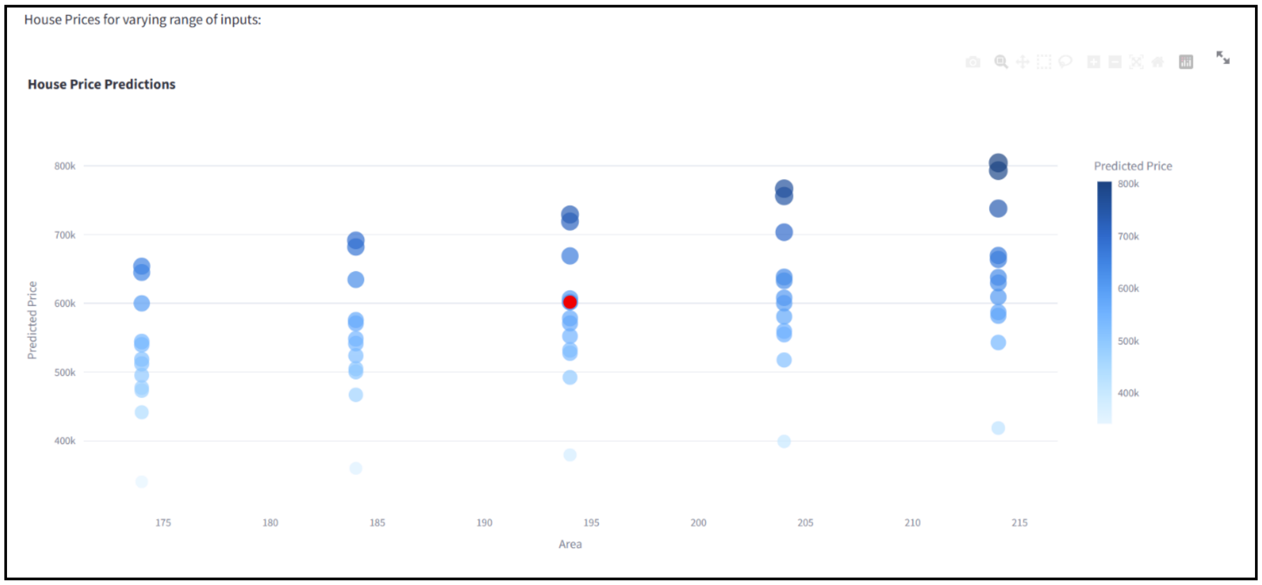
Select the Property Type

House/Villa [✓ Duplex] Townhouse

Rooms	Distance
2	11.7
Bathroom	Property Count
1	5678
Car	Area
1	194

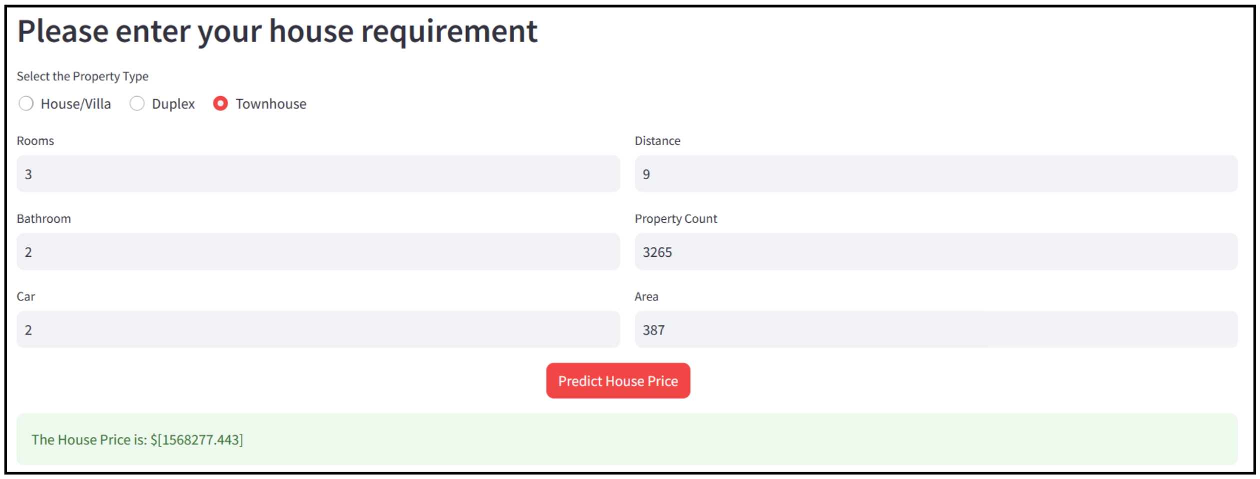
Predict House Price

The House Price is: $[601570.858]


House Prices for varying range of inputs:

House Price Predictions

Property Type: Townhouse

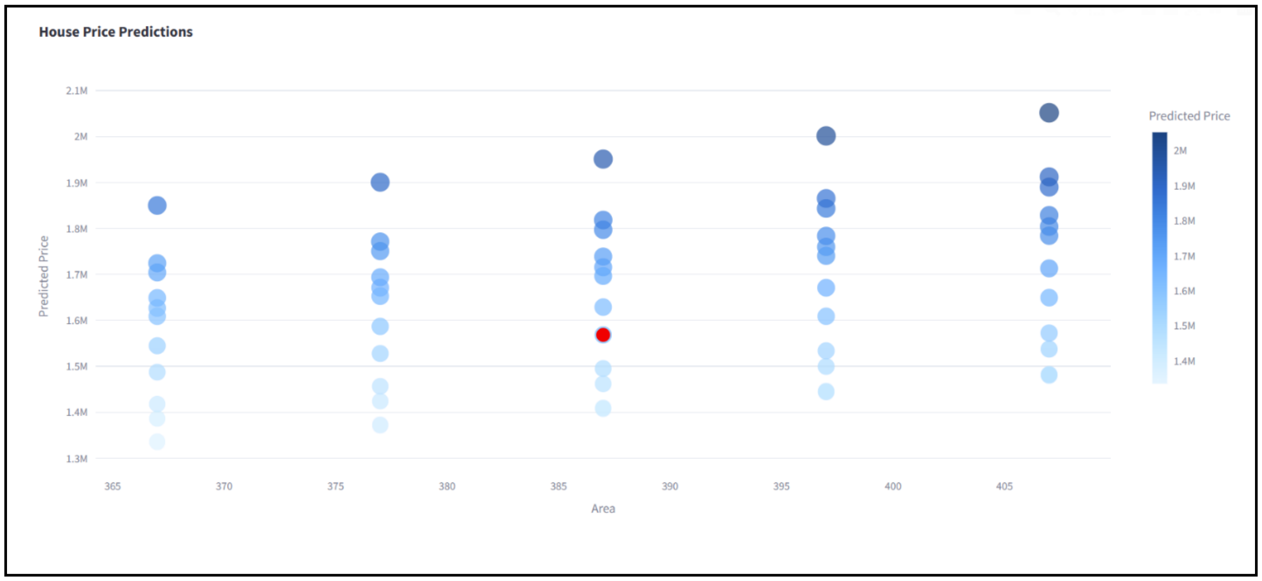

Please enter your house requirement

Select the Property Type

House/Villa Duplex [✓ Townhouse]

Rooms	Distance
3	9
Bathroom	Property Count
2	3265
Car	Area
2	387

The House Price is: $[1568277.443]



Note: The red point in the interactive plot signifies the input entered by the user.

In the interface, the user can provide their house requirements and based on it the house price will be predicted. We ask the user to select the type of property, number of rooms, bathrooms, car, distance, propertycount, and the land size. Once the user clicks on the predict button, the model will predict the house price at the backend and show the output on the UI.

Model used for our product:

In phase 2, we have experimented with a number of different Machine Learning and Deep Learning based regression algorithms for house price prediction given different features / attributes as input. Out of all the models we have tested, the Support Vector Regression (SVR) model gave the best result with an R-score of approximately 0.465. The Mean Squared Error recorded for this model was: 8.813359119317844e-06. As a result, we have implemented the SVR model in our final product in phase 3. This model is implemented in the backend of our product for predicting the house price for a certain set of feature values input by the user. The predicted price is then displayed to the user in the form of the front-end of our product.

For getting the best performance from SVR, we have done some hyperparameter tuning. The objective function of this model involves a loss term and a regularization term. Hyperparameter tuning is required to balance these loss functions. The values of hyperparameters are: C=1 and epsilon = 0.001. These hyperparameters were tuned after several iterations of experimentation with the model parameters.

Recommendations provided to the users:

Our product is developed to give the users an estimate of the price of a house in the Melbourne City Area, based on the values of the features input by the users. The front-end or the client-side of the web page of our product consists of an interactive and modern User Interface (UI), that is designed to suit even the non-technical clients. The UI consists of the following fields that the user needs to fill:

a) Rooms: Enter a numerical value (like 2, 3 etc.) signifying the number of rooms in the property.

b) Distance: Numerical value signifying the distance of the property from the city center, especially around the Suburb.

c) Bathroom: Numerical value signifying number of bathrooms in the property.

d) Car: Numerical value signifying space for number of cars in the property.

e) Property Count: Numerical value signifying number of properties in the surrounding area of the client's property.

f) Area: Numerical value signifying the area of the property.

g) Property Type: Radio button option in the UI, consisting of the following options: "House/Villa", "Duplex", "Townhouse".

After the user has filled in the above fields, the data collected from the user (from the front end) is fed into the SVR model running in the backend. The model in-turn returns the predicted house price, and it is displayed to the client on the client side.

Additionally, our product provides additional recommendations to the users based on their choice. This feature works dynamically when the user feeds in the data, based on that our product recommends other properties that fall in similar value range for the users' choice for the fields: Distance, Area, and Number of rooms. These recommendations for the users are presented in the form of interactive plot visualizations, making it convenient for the users to make sense of the recommended properties and their prices for different combinations of the above-mentioned features. All these features in our product will help the users to make informed decisions.

Future Avenues of Work:

Our product helps to predict house prices depending on the features of the property and its neighborhood. This can help the common people to get an estimate of the valuation of the property they are looking for just by a few clicks of the button. This can have a massive impact in the real estate business. Oftentimes it is seen, common people are being mis-informed and misguided by the agents and brokers during purchasing of the property. Our product can at least help the users make more informed choices. Currently, our product works on the “Melbourne Housing Dataset” and so it is effective in Melbourne and its suburbs. As a future work, we can extend our product for every major city in the world. But it will require reliable data collection and pre-processing.

Secondly, our product currently gives an estimate of the price of the property along with some additional recommendations to the users based on their set of choices. As a future work, we can extend our product by incorporating many other recommendations that may provide more insights to the users. This step will, however, require significant domain knowledge. So, we may need to consult real-estate agents, property dealers and also property managers regarding the additional relevant features that can be incorporated to help the users even more.

References:

[1] Dataset: https://www.kaggle.com/datasets/dansbecker/melbourne-housing-snapshot/data

[2]. C. O’Neill and R. Schutt. Doing Data Science., O’Reilly. 2013.

[3]. Data Science from Scratch book

[4] https://www.itl.nist.gov/div898/handbook/eda/section1/eda11.htm

[5] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html

[6] https://scikit-learn.org/stable/modules/linear_model.html#bayesian-ridge-regression

[7] https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsRegressor.html#sklearn-neighbors-kneighborsregressor

[8] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.ElasticNet.html#sklearn.linear_model.ElasticNet

[9] https://www.tensorflow.org/tutorials/quickstart/beginner

[10] https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html

[11] https://docs.streamlit.io/library/api-reference/widgets

[12] https://docs.streamlit.io/library/api-reference/charts

[13] https://plotly.com/python/line-and-scatter/