Analyzing Seattle Airbnb Dataset for Price Prediction

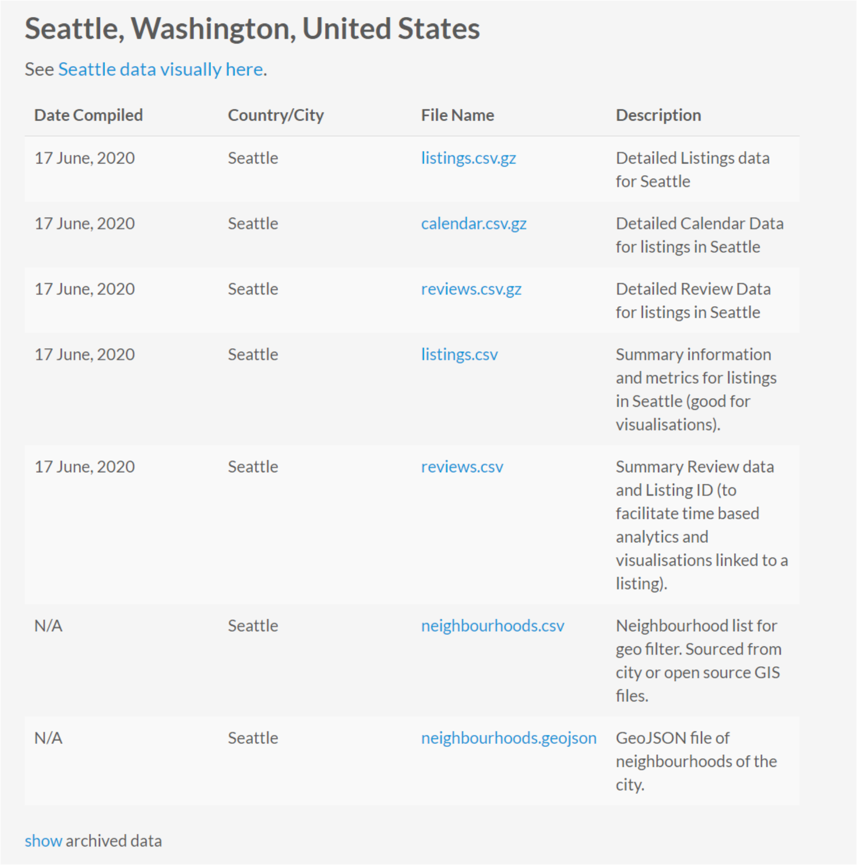
Contents

Introduction - Background

Airbnb, Inc. is an American vacation rental online marketplace company based in San Francisco, California, United States. It offers arrangement for lodging, primarily homestays, or tourism experiences. via online marketplace. Airbnb published in mar-2019 that the Airbnb hosts share more than six million listings around the world and it is expanding in emerging markets, growing around the world. As part of the Airbnb inside initiative, they publish a dataset describing listing activity of homestays for various cities. This dataset offers interesting opportunity to understand the booking behavior, price changes and various trends in different areas of the city. For the purpose of this project, we analyzed the dataset describing listing activity of homestays in Seattle to understand various trends such as finding busiest times of the year and how prices spike, general trend of Airbnb listing and Airbnb visitors etc. We also applied machine learning on the dataset to predict rental pricing or the factors which are seen affecting listing price etc.

Dataset:

The data behind the Inside Airbnb site is sourced from publicly available information from the Airbnb site. The page features various cities including Seattle. The data was also seen published on Kaggle website 2 years ago.



Seattle, Washington, United States

See Seattle data visually here.

Date Compiled	Country/City	File Name	Description
17 June, 2020	Seattle	listings.csv.gz	Detailed Listings data for Seattle
17 June, 2020	Seattle	calendar.csv.gz	Detailed Calendar Data for listings in Seattle
17 June, 2020	Seattle	reviews.csv.gz	Detailed Review Data for listings in Seattle
17 June, 2020	Seattle	listings.csv	Summary information and metrics for listings in Seattle (good for visualisations).
17 June, 2020	Seattle	reviews.csv	Summary Review data and Listing ID (to facilitate time based analytics and visualisations linked to a listing).
N/A	Seattle	neighbourhoods.csv	Neighbourhood list for geo filter. Sourced from city or open source GIS files.
N/A	Seattle	neighbourhoods.geojson	GeoJSON file of neighbourhoods of the city.

show archived data

Fig 1: Inside Airbnb dataset for Seattle, Washington, United States

From the dataset, we have scoped to 3 files – listings.csv, calendar.csv, reviews.csv

The following Airbnb activity is included in this Seattle dataset:

- **Listing.csv:** Listings, including full descriptions and average review score.
- **Reviews.csv:** Reviews, including unique id for each reviewer and detailed comments.
- **Calendar.csv:** Calendar, including listing id and the price and availability for that day.

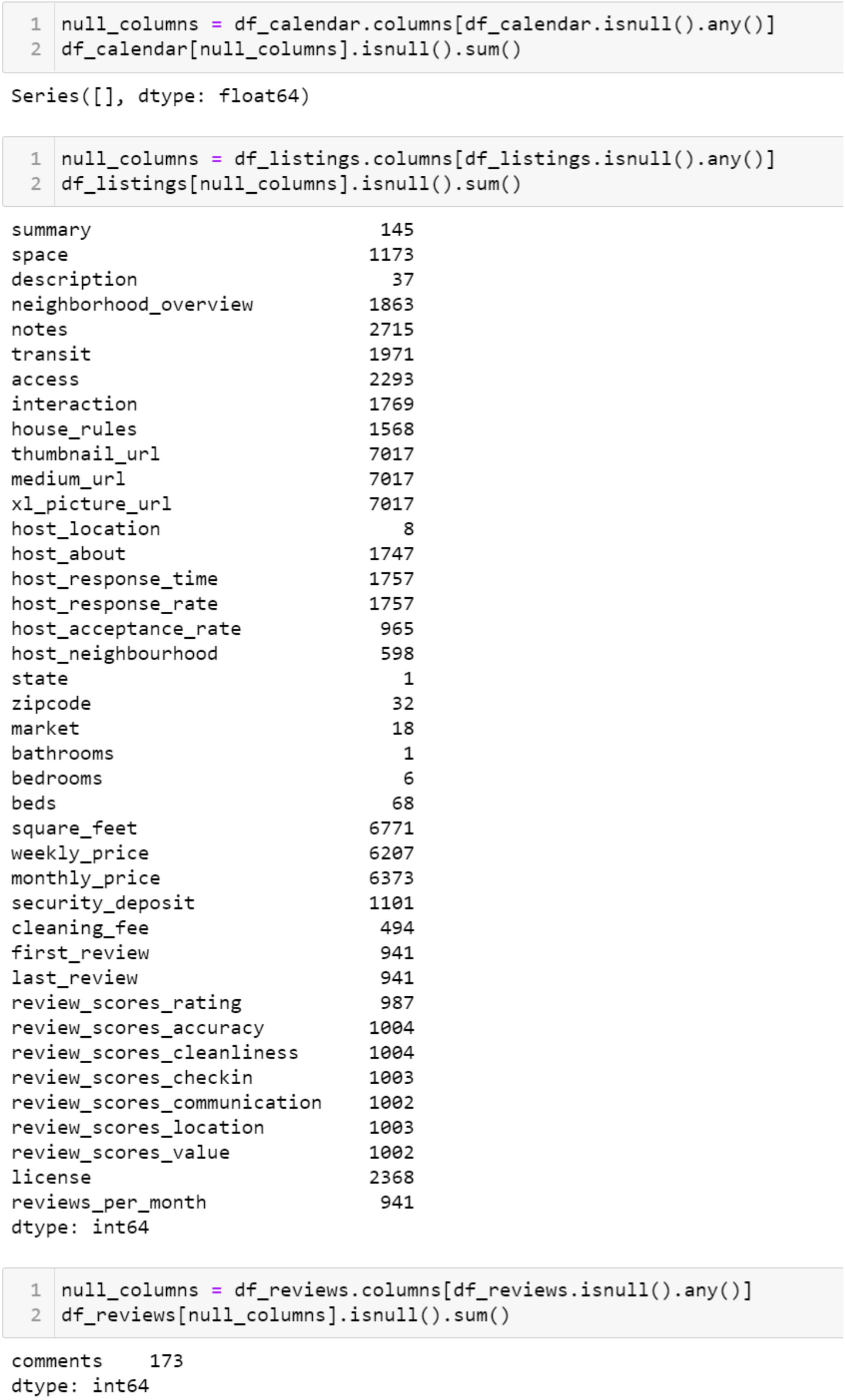
Data Preparation and Cleaning

We will download zipped version of these files directly from Jupyter notebook. Jupyter notebook is an open source web application that allows you to create and share documents that contain live code, visualization and narrative text. We will do all the steps associated with the dataset in the interactive jupyter notebook. We will use read_csv method of Pandas library to directly read from the downloaded zip links as show in fig 1.

```
df_calendar = pd.read_csv('./data/calendar.csv.gz')
df_listings = pd.read_csv('./data/listings.csv.gz')
df_reviews = pd.read_csv('./data/reviews.csv.gz')
```

Fig 2: Read dataset from zipped version

First, we will look at all 3 datasets if there are any empty values. For Calendar datasets, there were no empty values observed, however for listings and reviews multiple columns were seen with null values. Some of the columns in each dataset are expected to have empty columns and may not be needed to replace for now.



```
null_columns = df_calendar.columns[df_calendar.isnull().any()]
df_calendar[null_columns].isnull().sum()
```

```
Series([], dtype: float64)
```

```
null_columns = df_listings.columns[df_listings.isnull().any()]
df_listings[null_columns].isnull().sum()
```

```
summary                         145
space                          1173
description                      37
neighborhood_overview          1863
notes                          2715
transit                        1971
access                         2293
interaction                    1769
house_rules                    1568
thumbnail_url                  7017
medium_url                     7017
xl_picture_url                 7017
host_location                     8
host_about                     1747
host_response_time             1757
host_response_rate             1757
host_acceptance_rate            965
host_neighbourhood              598
state                             1
zipcode                          32
market                           18
bathrooms                         1
bedrooms                          6
beds                             68
square_feet                    6771
weekly_price                   6207
monthly_price                  6373
security_deposit               1101
cleaning_fee                    494
first_review                    941
last_review                     941
review_scores_rating            987
review_scores_accuracy         1004
review_scores_cleanliness      1004
review_scores_checkin          1003
review_scores_communication    1002
review_scores_location         1003
review_scores_value            1002
license                        2368
reviews_per_month               941
dtype: int64
```

```
null_columns = df_reviews.columns[df_reviews.isnull().any()]
df_reviews[null_columns].isnull().sum()
```

```
comments    173
dtype: int64
```

Fig 3: Check null values in dataset

As we look into this dataset further, we see additional opportunities to clean the dataset further especially price related columns in each dataset. Such as

- Price columns in the dataset have $ sign appended.
- Price values in dataset had , which need to be removed.
- Some of the values are Nan which need to be replaced with 0.

In addition to above, some of the columns such as price columns, date columns in dataset will need to be converted to correct datatypes to prepare for data analysis.

Data Wrangling

In order to prepare the dataset for analysis and visualization, we performed some steps. The available column is calendar dataset has t, where as host_is_superhost column in listing dataset has t value. We will replace these values with 1 and 0 for further analysis. We can use lambda functions to achieve this.

We will also create a new dataset showing availability (no of days available) per listing_id.

This dataset will be joined with listing dataset to populate additional columns available showing no of days available per listing id.

Data Visualization

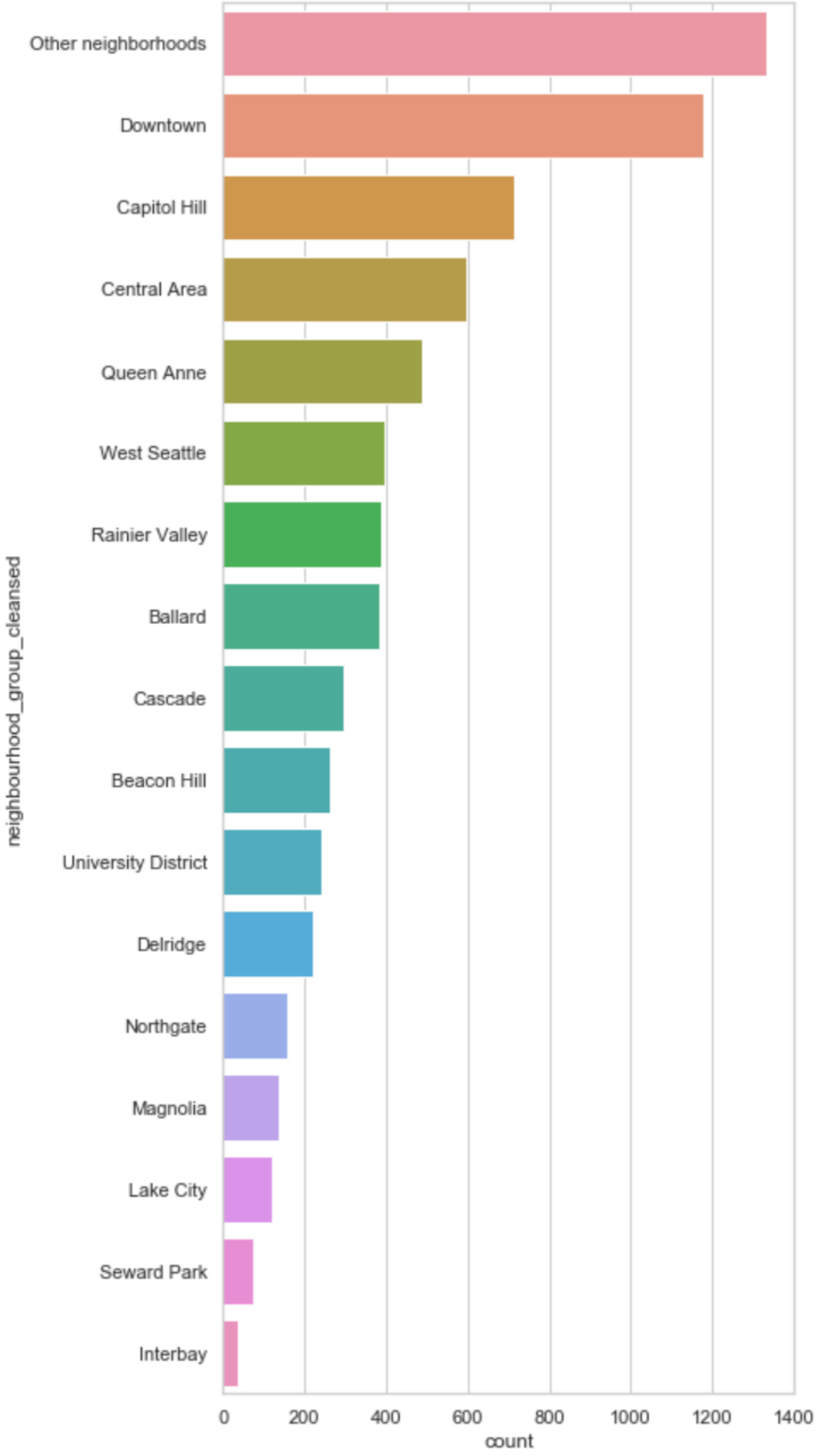


Fig 5: Number of Properties per Neighborhood

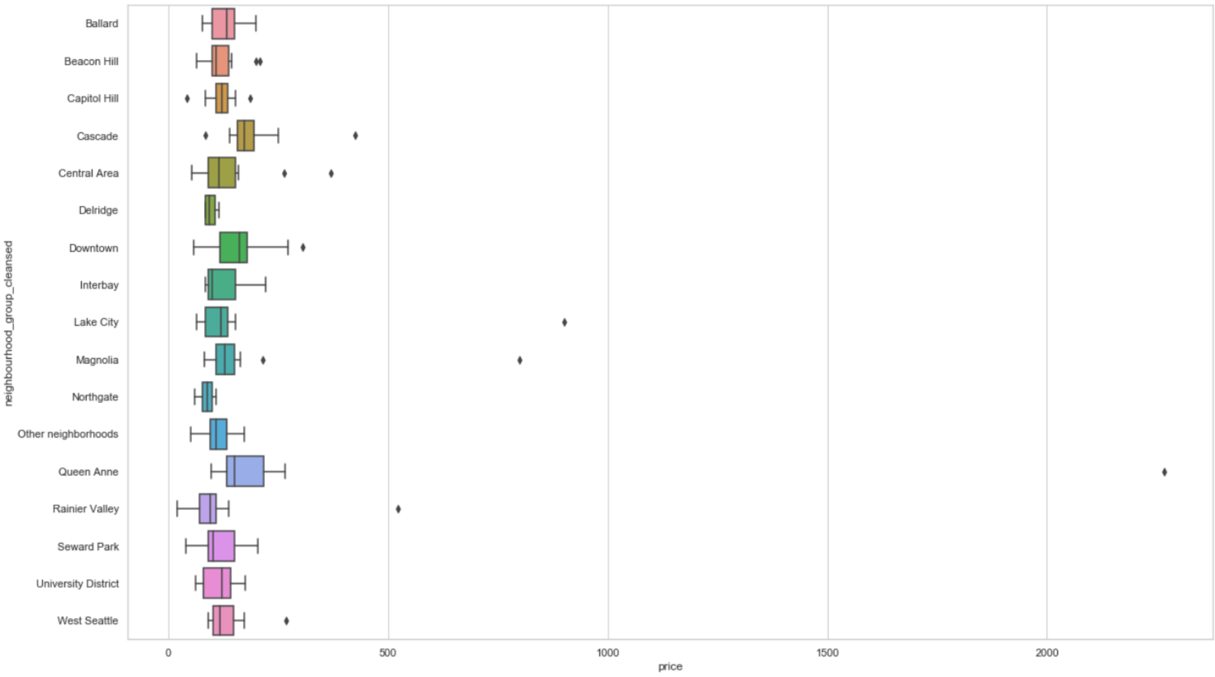


Fig 6: Price distribution per Neighborhood

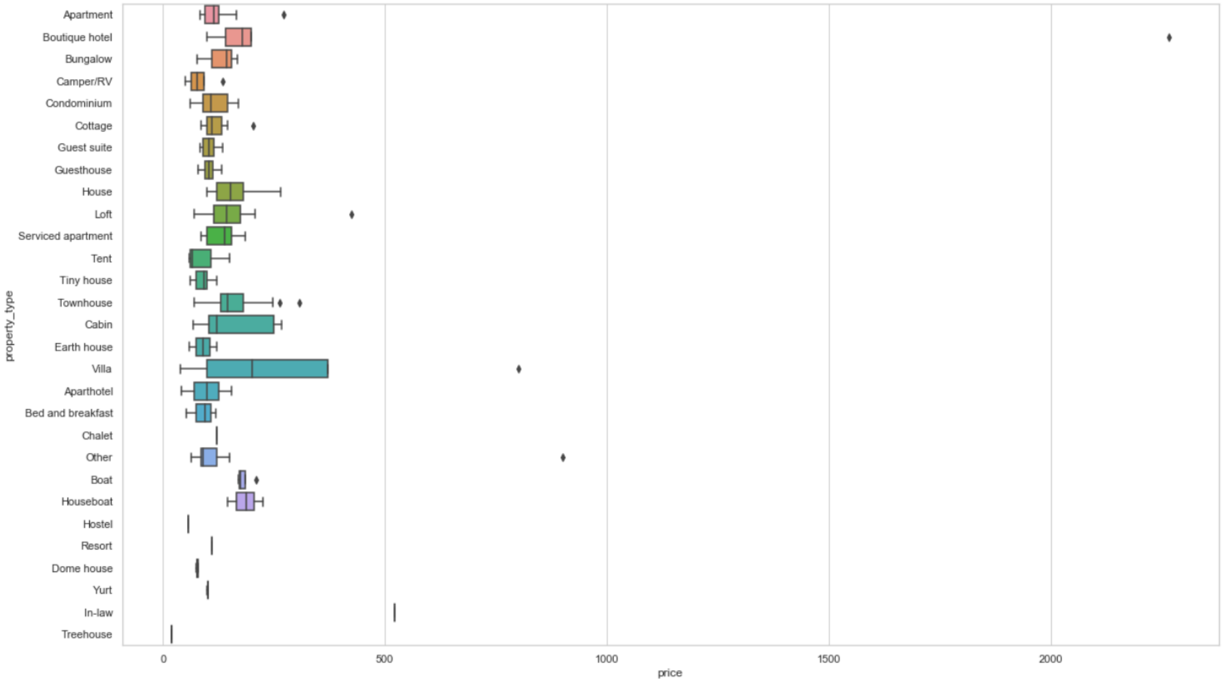


Fig 7: Price distribution Per Property Type

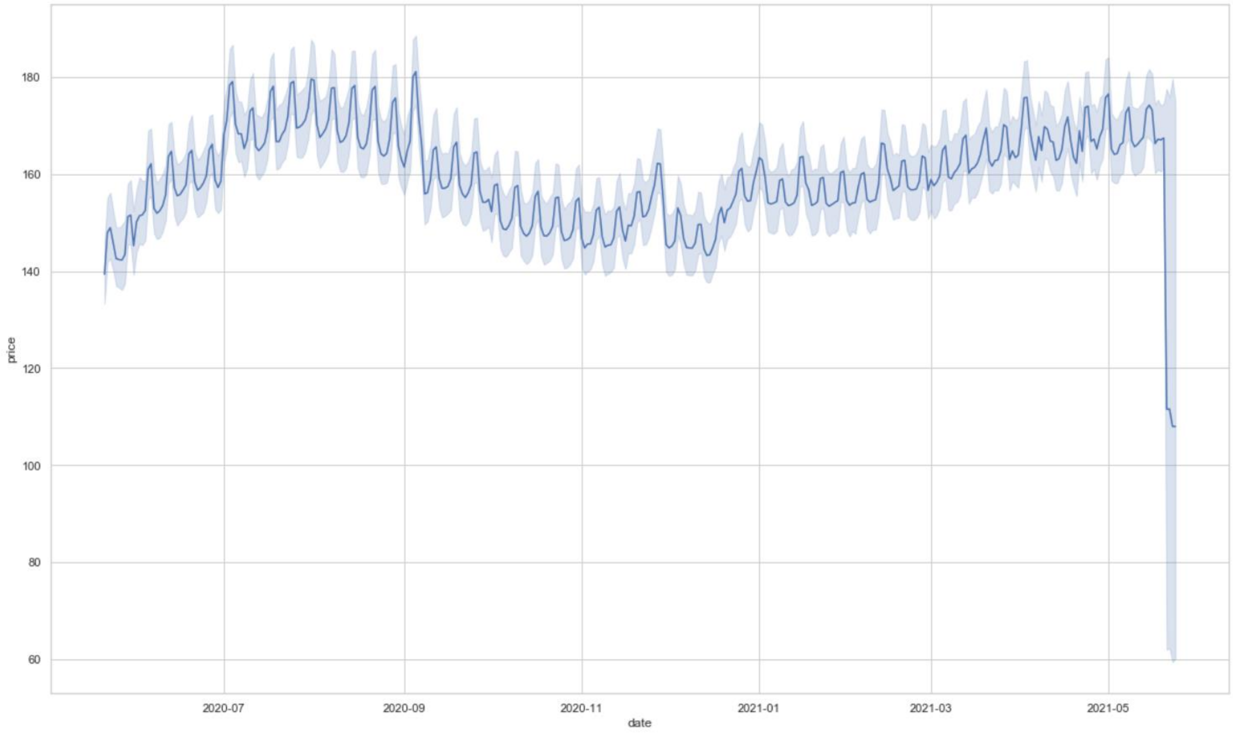


Fig 8: Price Trend per Calendar Date.

Data Analysis

Primary goal of analyzing this dataset was to determine if there are any strong correlation between pairs of independent variable of the dataset or between an independent and dependent variable (price).

Let’s segregate price related columns and analyze its distribution via boxplot.

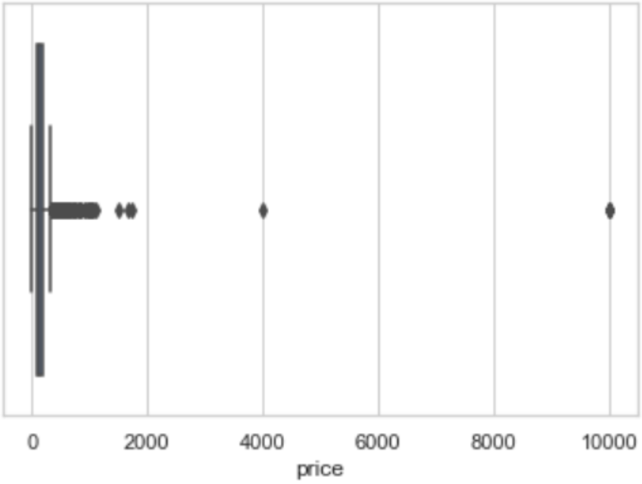
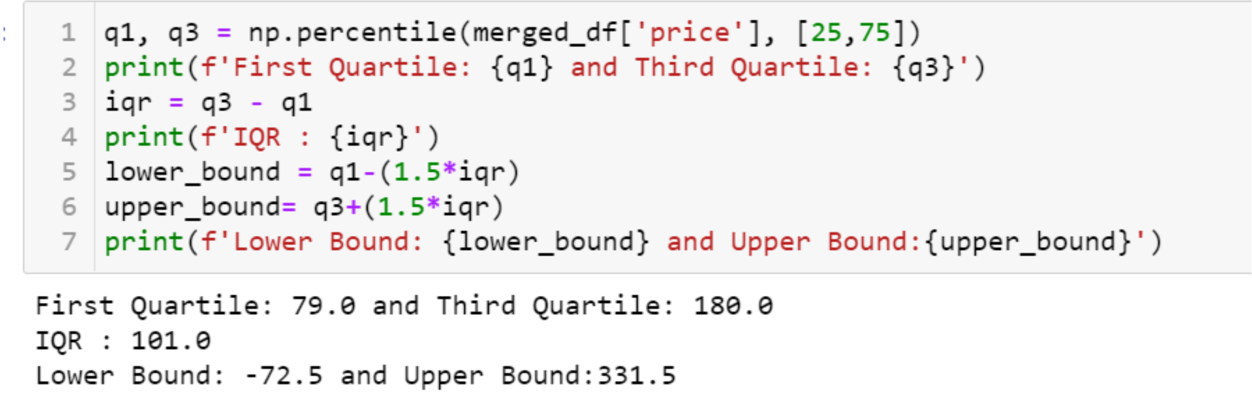


Fig 9: Box plot of price distribution.

As you can see, we have few outliers due to which the box plot is heavily right skewed.

Let's identify these outliers and remove them. To do this, we will find first and third quartile ranges of the dataset among with Inter Quartil Range (IQR). We can then calculate lower and upper bound of the dataset which can be 2 times of IQR.

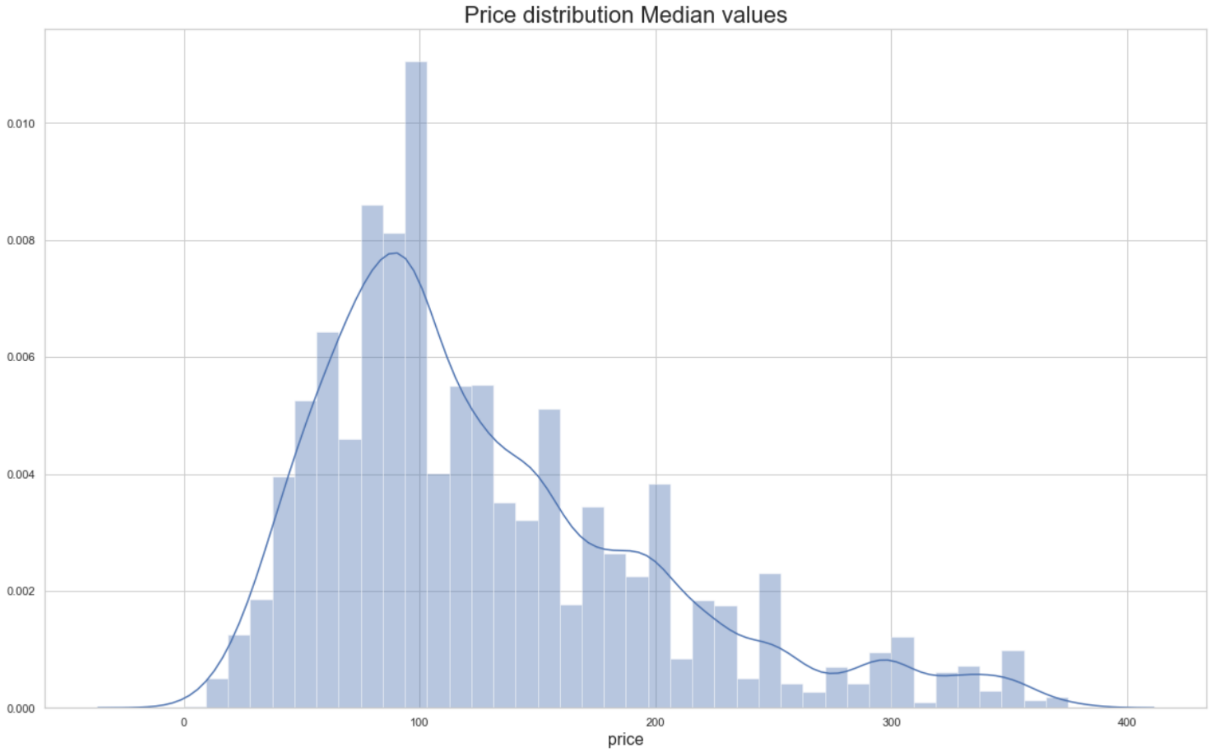


```python
q1, q3 = np.percentile(merged_df['price'], [25,75])
print(f'First Quartile: {q1} and Third Quartile: {q3}')
iqr = q3 - q1
print(f'IQR : {iqr}')
lower_bound = q1-(1.5*iqr)
upper_bound= q3+(1.5*iqr)
print(f'Lower Bound: {lower_bound} and Upper Bound:{upper_bound}')
```

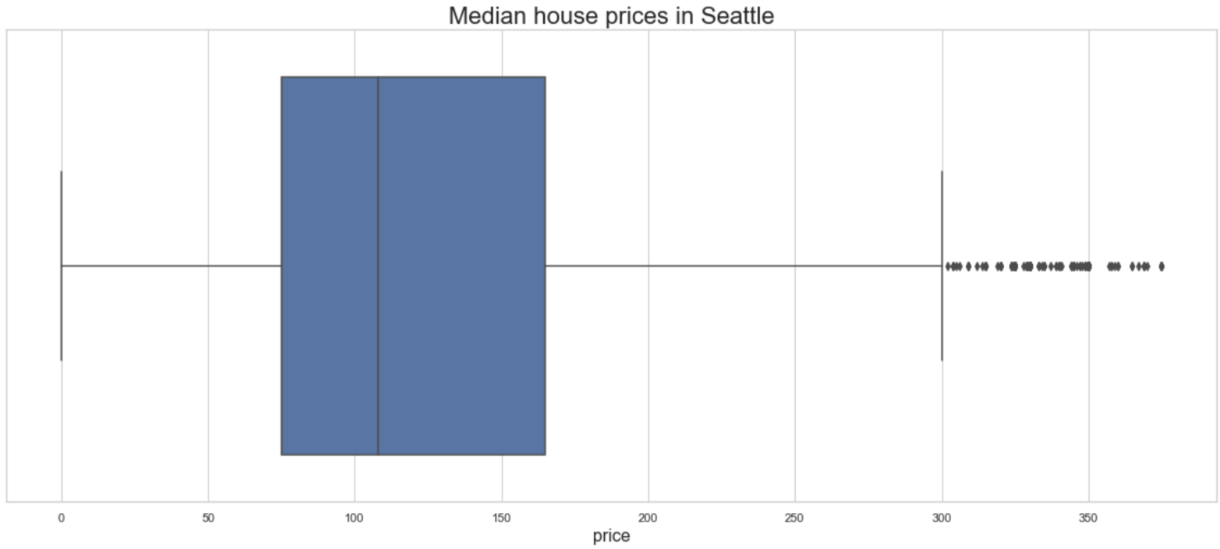
```
First Quartile: 79.0 and Third Quartile: 180.0
IQR : 101.0
Lower Bound: -72.5 and Upper Bound:331.5
```

Fig 10: Calculate IQR and lower and upper bound ranges.

Anything above upper bounds can be removed from the original dataset and remaining can be used to analyze and visualize.



Once we removed outliers, we will again plot boxplot to visualize median house prices in Seattle.



We will compute Empirical cumulative distribution function (ECDF) on the price column of the dataset and generate the plot.

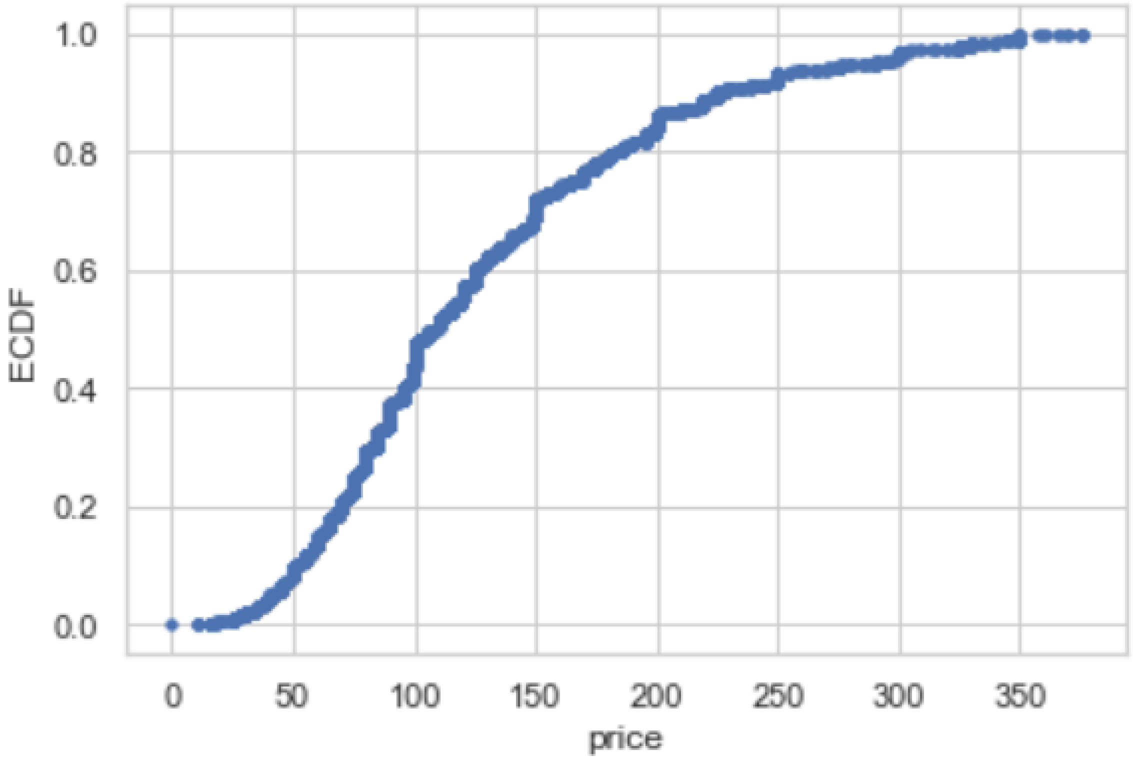


Fig 11: ECDF plot for Price column

The above plot shows 90% of the listing prices are under 250$.

Now let’s plot the dataset with outliers removed to understand price distribution across various property types.

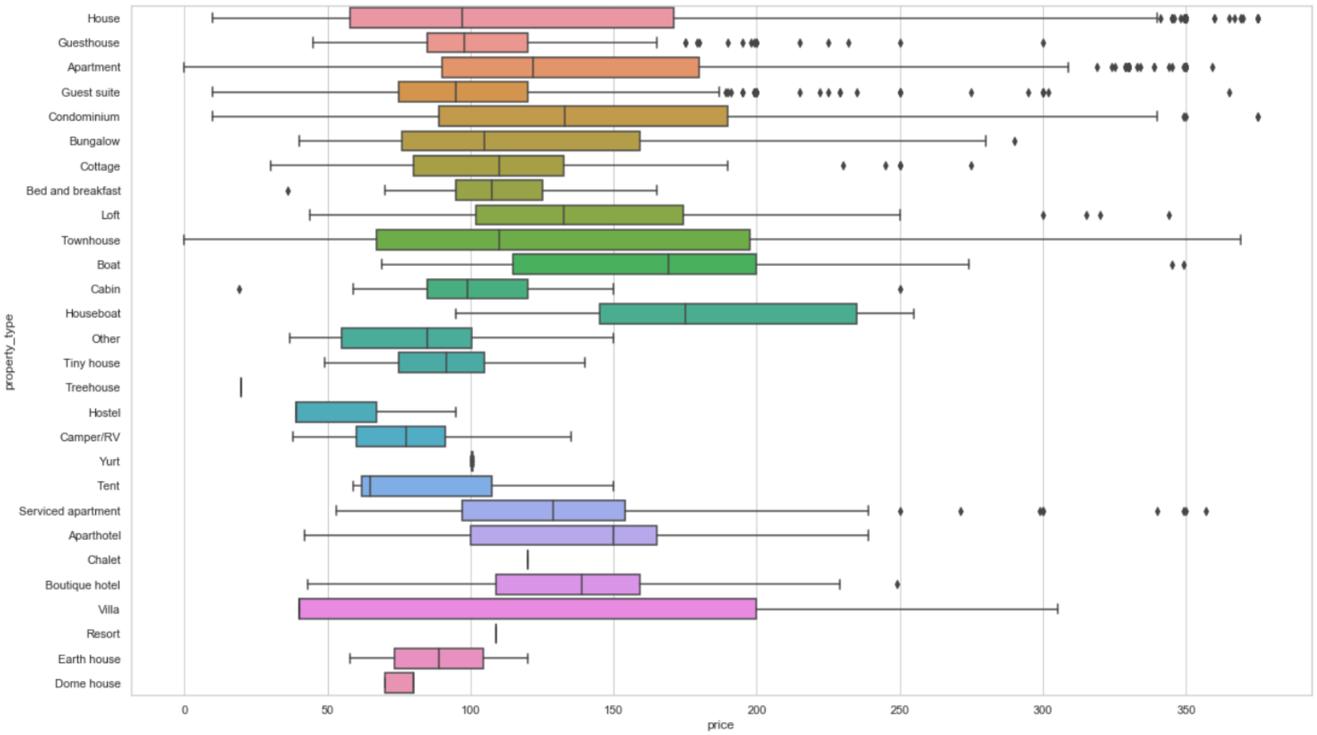


Fig 12: Price distribution per Property type

We will now compare price against various features to understand strong correlation.

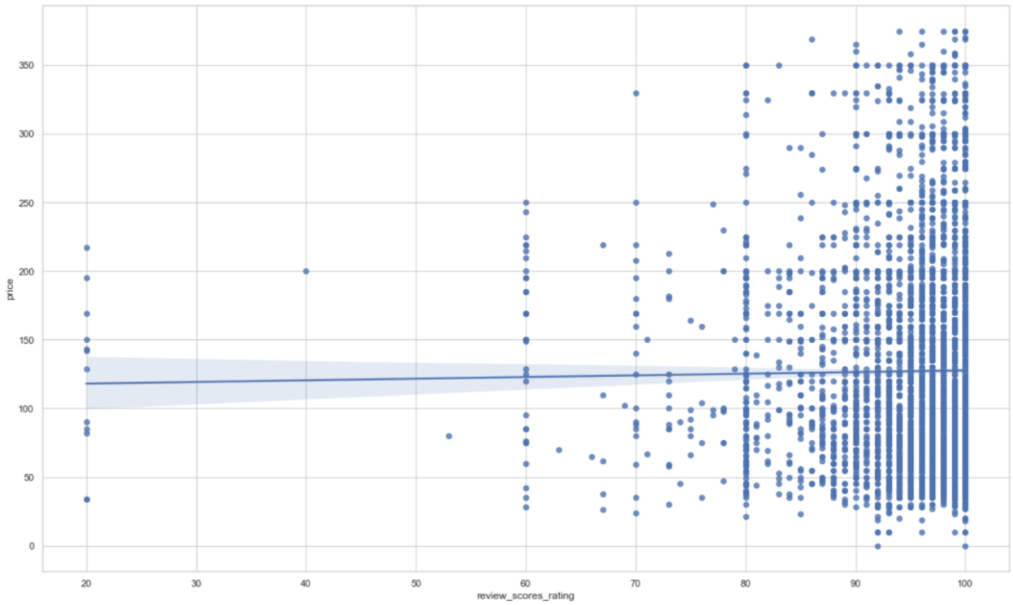


Fig 13: Review Score Rating vs Price

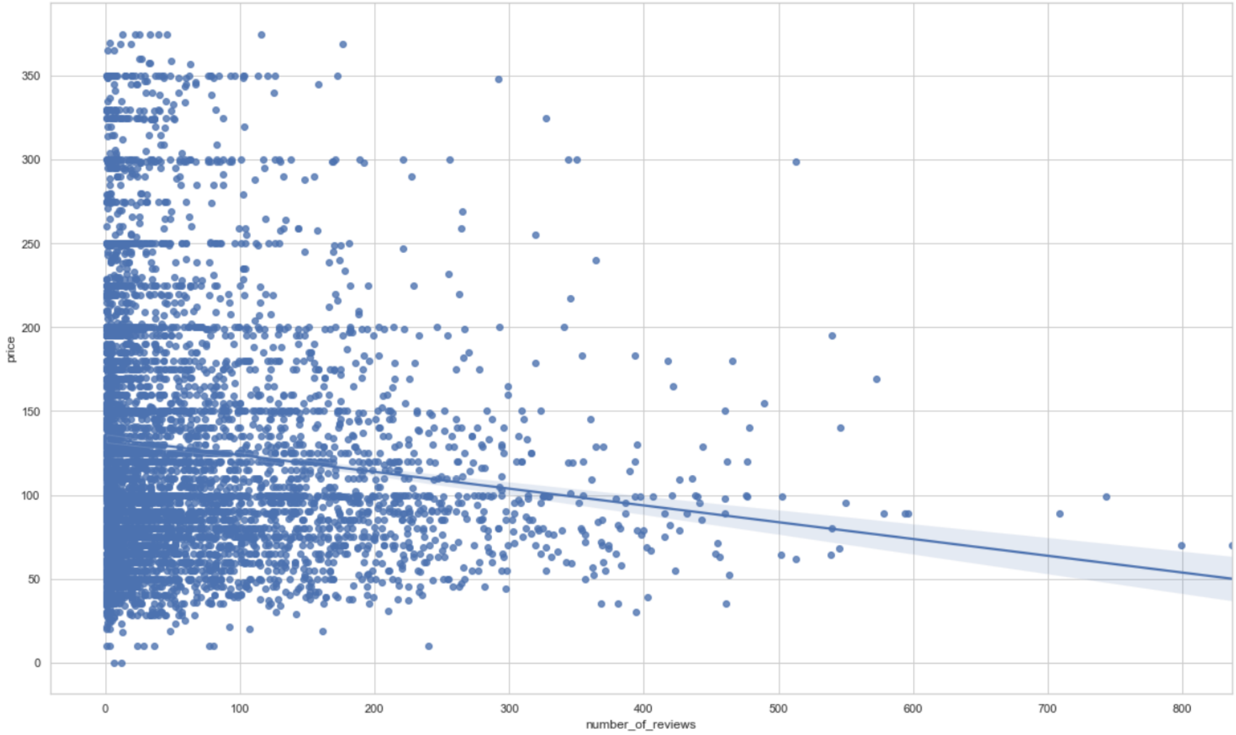


Fig 14: Number of Review vs Price

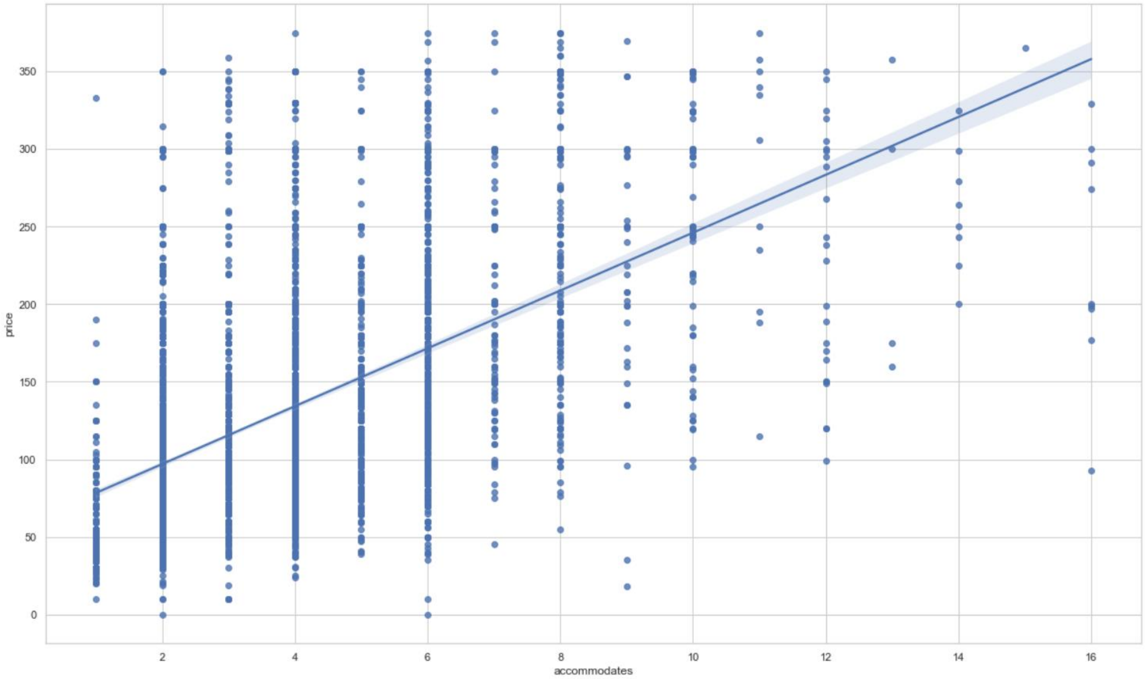


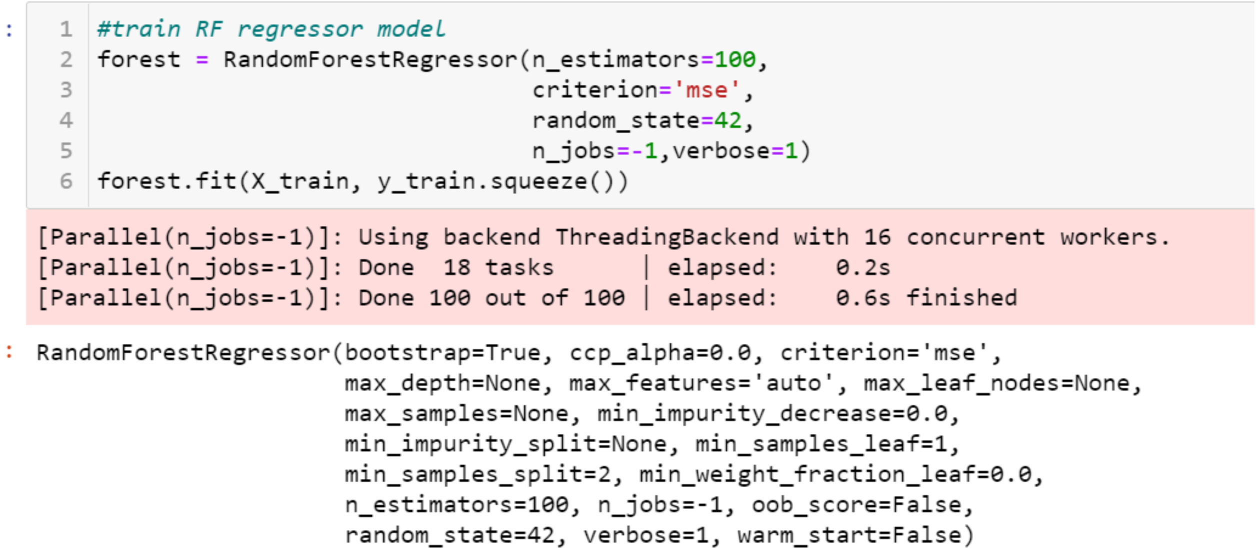
Fig 15: Accommodates vs Price

From all of the above charts, it is clearly evident that there is strong correlation in last chart which is between no of accommodates against the price. This means listing price directly affected by no of accommodates allowed in the property. We did not see any strong correlation between either no of reviews or review scores to the price.

Data Modelling

From the original dataset, we have selected specific columns to reduce dimensionality as well drop NaN rows. We will then perform one-hot encoding for the categorical columns. Finally we will check if there are any null values in the dataset.

We will split the dataset into 70% training and 30 % testing. We will use random forest regressor to fit the model. A random forest is a meta estimator that fits a number of classifying decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.

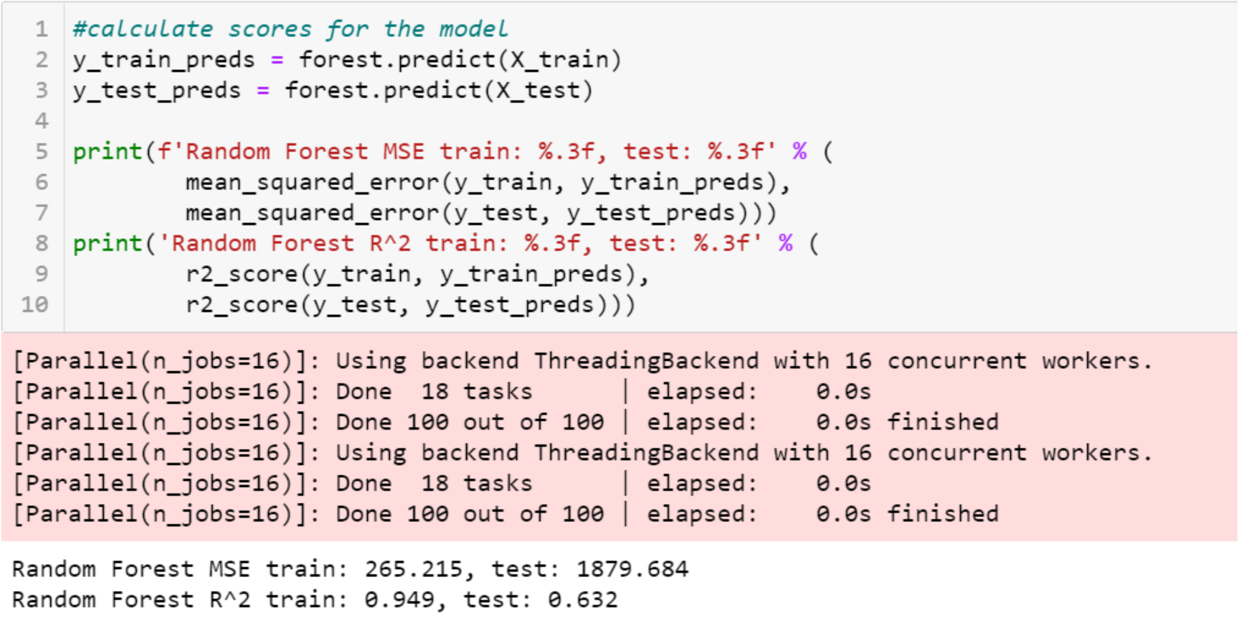


```
#train RF regressor model
forest = RandomForestRegressor(n_estimators=100,
                               criterion='mse',
                               random_state=42,
                               n_jobs=-1,verbose=1)
forest.fit(X_train, y_train.squeeze())
```

```
[Parallel(n_jobs=-1)]: Using backend ThreadingBackend with 16 concurrent workers.
[Parallel(n_jobs=-1)]: Done  18 tasks      | elapsed:    0.2s
[Parallel(n_jobs=-1)]: Done 100 out of 100 | elapsed:    0.6s finished
```

```
RandomForestRegressor(bootstrap=True, ccp_alpha=0.0, criterion='mse',
                      max_depth=None, max_features='auto', max_leaf_nodes=None,
                      max_samples=None, min_impurity_decrease=0.0,
                      min_impurity_split=None, min_samples_leaf=1,
                      min_samples_split=2, min_weight_fraction_leaf=0.0,
                      n_estimators=100, n_jobs=-1, oob_score=False,
                      random_state=42, verbose=1, warm_start=False)
```

Fig 16: RandomForestRegressor to train and Fit the model



```
#calculate scores for the model
y_train_preds = forest.predict(X_train)
y_test_preds = forest.predict(X_test)

print(f'Random Forest MSE train: %.3f, test: %.3f' % (
        mean_squared_error(y_train, y_train_preds),
        mean_squared_error(y_test, y_test_preds)))
print('Random Forest R^2 train: %.3f, test: %.3f' % (
        r2_score(y_train, y_train_preds),
        r2_score(y_test, y_test_preds)))
```

```
[Parallel(n_jobs=16)]: Using backend ThreadingBackend with 16 concurrent workers.
[Parallel(n_jobs=16)]: Done  18 tasks      | elapsed:    0.0s
[Parallel(n_jobs=16)]: Done 100 out of 100 | elapsed:    0.0s finished
[Parallel(n_jobs=16)]: Using backend ThreadingBackend with 16 concurrent workers.
[Parallel(n_jobs=16)]: Done  18 tasks      | elapsed:    0.0s
[Parallel(n_jobs=16)]: Done 100 out of 100 | elapsed:    0.0s finished
```

```
Random Forest MSE train: 265.215, test: 1879.684
Random Forest R^2 train: 0.949, test: 0.632
```

Fig 17: Scores for the model

Lastly, we will get feature importance from the model and plot it to see which features are directly affecting the price. From the below chart, we can see no of bedroom has positive correlation with the price. This is expected as no of accommodations or bedroom increases, price will also increase.

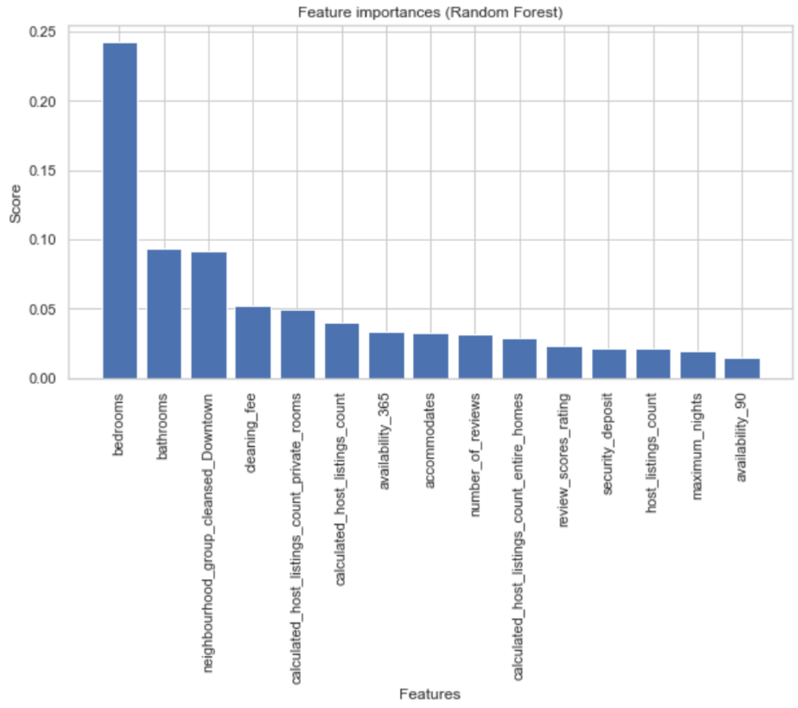


Fig 18: Feature Importance Matrix for the Model

Conclusion and Final Thoughts

We started the Airbnb data analysis project by acquiring Seattle listing dataset published by Airbnb site. After acquiring dataset, we performed various data cleaning and wrangling steps to be prepare for data analysis, visualization, and modelling steps. After visualizing price distribution of the clean dataset, we identified outliers in price columns. The outliers were then removed by calculating Inter Quartile Range (IQR) and upper bounds which is 2 times IQR. We visualized the dataset again after removing outliers. We also plotted various attributes of dataset against price column to understand correlation. We found a strong correlation between no of accommodates. To apply machine learning, we selected specific columns, removed nan rows, performed one hot encoding for categorical columns, split the datasets and applied random forest regressor to train and fit the model. From the feature importance matrix, we could see bedrooms as the most important feature affecting the price. Given more time for the project, we could have applied various machine learning algorithms to predict the price and select model with the most accuracy rate.

Project Code References

- Project GitHub Repository : https://github.com/kirti-chaudhari/SpringBoard_DataScience_Career
- Jupyter Notebook: https://github.com/kirti-chaudhari/SpringBoard_DataScience_Career/blob/master/CapstoneProject/AirbnbProject.ipynb