1. ABSTRACT

Airbnb is an online marketplace for hosts to list their properties so that guests can rent to stay for lodging and tourism activities. Even though Airbnb provides general guidance currently there is no free service that will help owners to set the prices. Airbnb hosts are facing difficulty to set the optimal price for their listings. Our goal is to develop a machine learning model that will help hosts to set the prices for their properties using the pricing data for other listings in San Francisco.

2. DATA

The dataset for this project is downloaded from Inside Airbnb website. Two csv files are used in the analysis. *listings_detailed.csv* contains detailed information about Airbnb listings in San Francisco and it is scraped on 03-02-2021. The dataset contains 6883 rows with 73 attributes and loaded as df. *calendar.csv* contains the pricing information at the date of scraping for each listing for the next year. The dataset contains 2512295 rows with 7 attributes and loaded as calendar. The definition of each attributes that is used in analysis can be found in **Appendix** section. The main limitation of the dataset is it lacks the actual price that a guest pays for a listing. The pricing information is the list price and it could be different than the actual paid amount. In addition, the dataset does not contain other fees such as cleaning fee which could vary largely and can make a big difference in the final price paid by the guest.

2.1 Data Wrangling:

The first step of the data wrangling section is dropping of the attributes that has too many missing values (>50%) and that contain useless information (URL's for host, description for neighborhood or listing etc.) for the machine learning model. Following this initial step, each attribute is investigated one by one and necessary data wrangling steps are applied. These steps can be found in detail in the data wrangling notebook.

2.2 Exploratory Data Analysis:

Exploratory data analysis is critical for the any data science work since helps data scientist to look at the data before making any assumptions. For this purpose, I plotted the distribution of target and target's relation with various attributes.

```
count     6882.000000
mean       219.870096
std        603.523080
min         10.000000
2.5%        38.000000
25%         85.000000
50%        135.000000
75%        219.000000
97.5%      799.000000
max      25000.000000
Name: price, dtype: float64
```

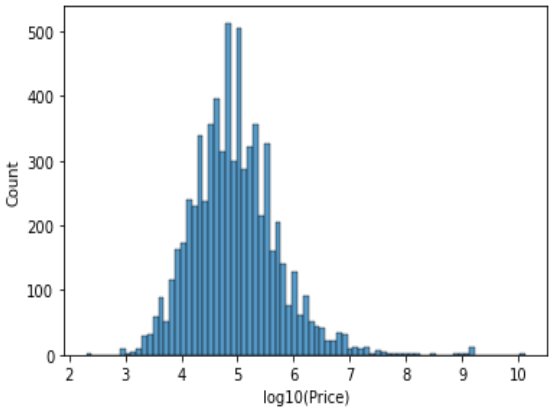

Figure 1: (a) Price distributions of Airbnb listings in San Francisco (b) Summary of Airbnb listings price.

Descriptive statistics in Figure 1a revealed price data has extremely low and high values which can cause problems for training of the machine learning model. Price data was log transformed for the visualization purposes (Figure 1b). These listings with very high or low price can be either a rare incidence or data entry error. Based on the initial performance of the model, these extreme values can be filtered.

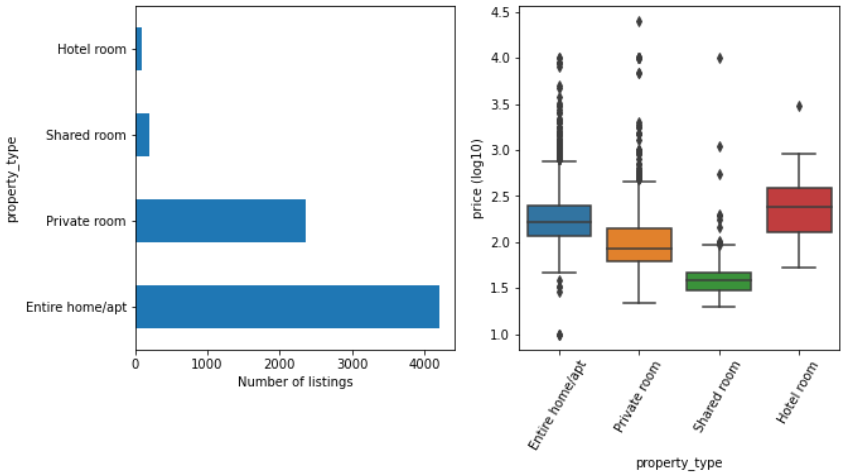

Figure 2: Number of listings vs property type (left), Price (log10) vs property type

Majority of the listings in San Francisco are 'entire home or apartments' and the median price of these listings are more than private or shared rooms.

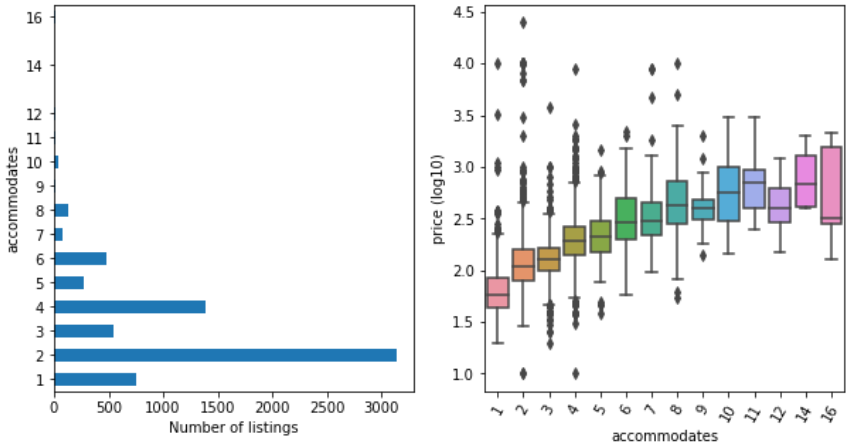

Figure 3: Number of listings vs accommodates (left), Price (log10) vs acommodates

The price of listings increases as the number of people it can accommodate. Majority of the listings can accommodate 3 or less guests.

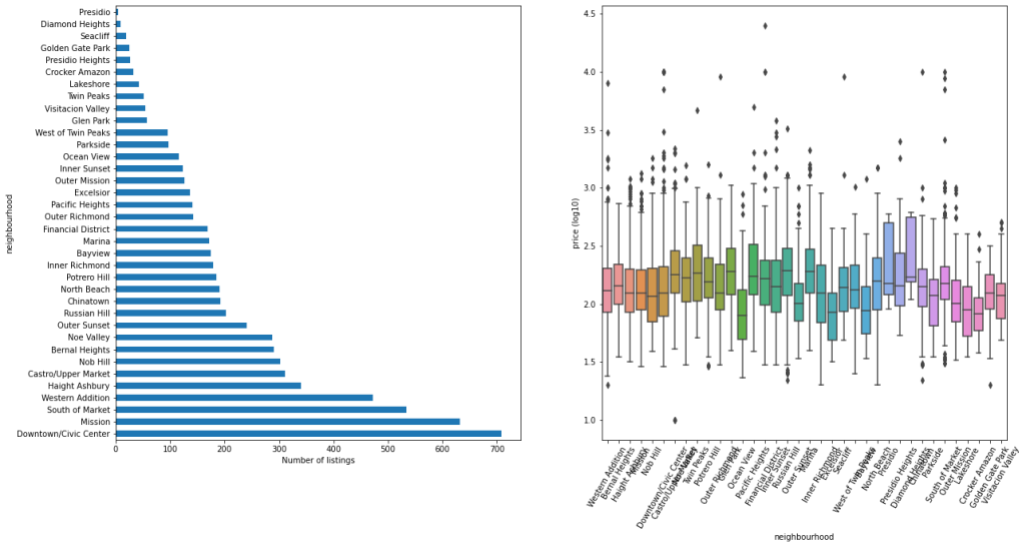

Figure 4: Number of listings vs neighbourhood (left), Price (log10) vs neighbourhood

Prices are similar among the neighborhoods in the San Francisco. Downtown/Civic Center has the most of the listings.

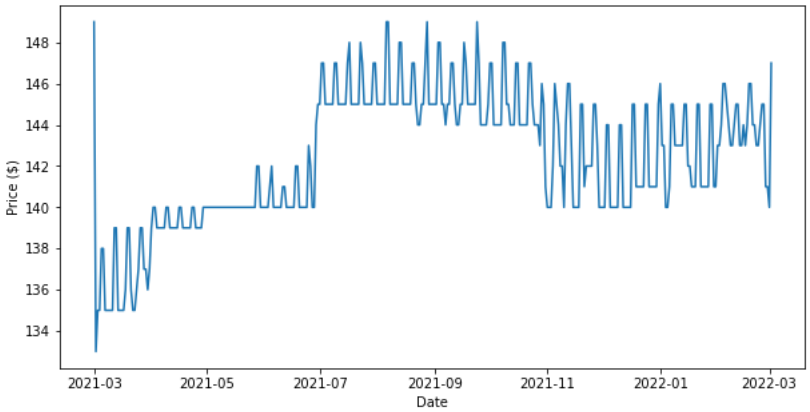

Figure 5: The median of listed prices between 03-02-2021 and 03-02-2021.

The listing prices have increased in summer months and show spikes that correspond to weekends.

2.3 Preprocessing of the Data:

Several attributes have been binned to create categorical variables. All of the categorical variables have been converted to dummy variables. Details can be found in data wrangling notebook. Tree-based models have been used as machine learning models. These models do not require standardization or normalization, however, since linear regression is used as base model both standardization and normalization are done to the data. MinMaxScaler is used as normalization method since it is applied to whole data frame and the data frame consists dummy variables with 0 and 1 entries. Details can be found in model notebook.

3. MODELING and RESULTS

Three models are used for price prediction:

1. Linear Regressor
2. Random Forest Regressor
3. XGBoost Regressor

Model	*Training Data*	*Test Data*
Linear Regressor	0.11	0.02
XGBoost Regressor	0.99	-0.02
Random Forest Regressor	0.92	0.11

Table 1: R-squared scores for 3 different models.

Initial results (Table 1) reveal that all three models have overfit to the training data and perform poorly on the test data. Looking deeper into model performance using 5-fold cross-validation revealed that some folds have really low R-squared values (negative values, data not shown) and others having reasonable R-squared values (0.5-0.7, data not shown). Plotting the Actual Price vs Predicted Price revealed a trend where high price listings tend to be predicted lower and low-price listings tend to be predicted higher (Figure 5). A new variable, % error, is created using actual and predicted price (% error = 100* (predicted price - actual price)/(actual price)). Actual price vs % error revealed that, model performs poorly on the low-price range for both training data (Figure 6) and test data (Figure 7).

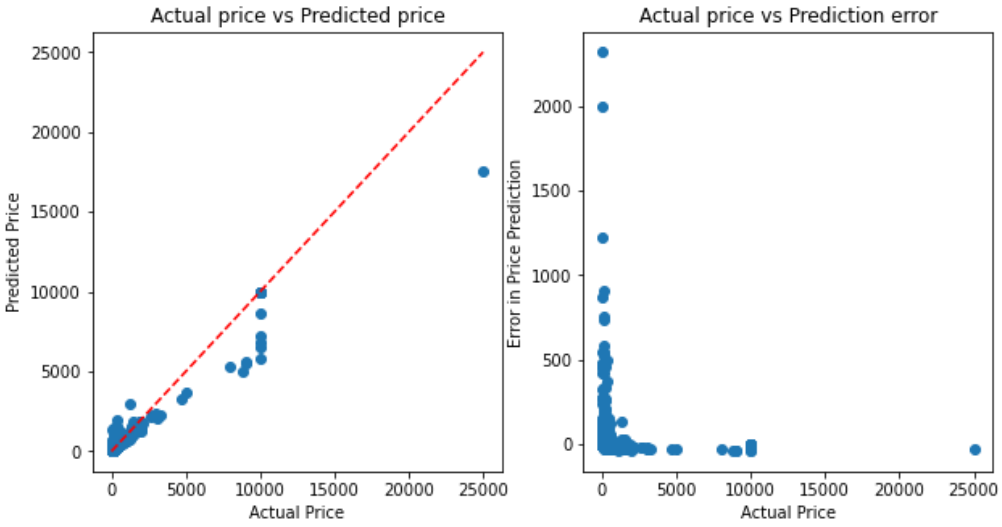

Figure 6: Actual price vs predicted price (left), Error in Price Prediction vs Actual Price (right) for the **training data**.

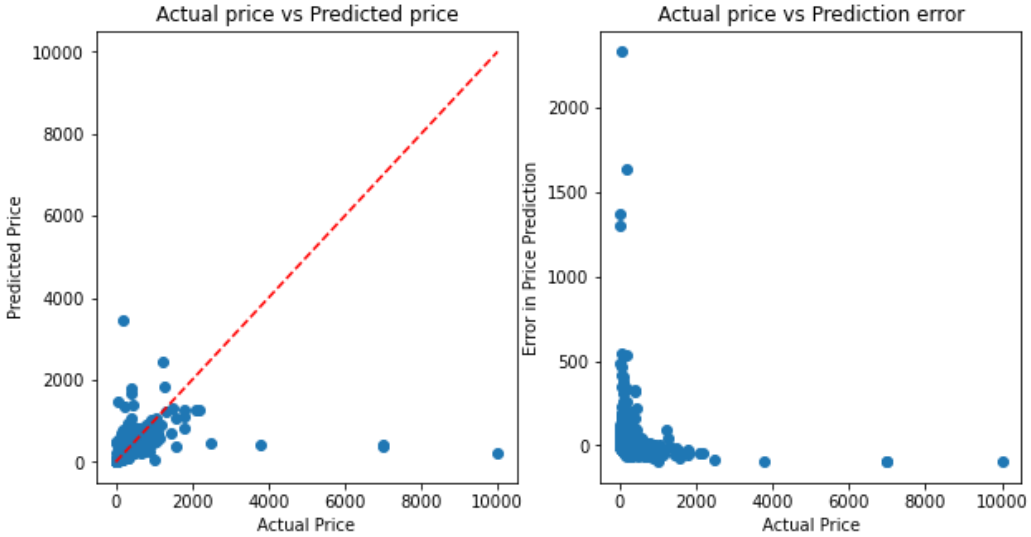

Figure 7: Actual price vs predicted price (left), Error in Price Prediction vs Actual Price (right) for the **test data**.

In the EDA section, descriptive statistics of the 'price' has revealed that some listings had very low price. To further investigate this, price_per_accommodate, variable is created. Descriptive statistics of this variable makes it clear that some listings have really low prices which seems as a data entry error.

```
count     6882.000000
mean        77.494464
std        289.329700
min          2.500000
2.5%        19.430357
25%         37.250000
50%         50.000000
75%         75.000000
97.5%      187.491667
max      12500.000000
Name: price_per_accommodate, dtype: float64
```

The outliers (lower 2.5% and upper 2.5%) are filtered out and models are trained with the filtered dataset.

Model	*Training Data*	*Test Data*
Linear Regressor	0.53	0.53
XGBoost Regressor	0.94	0.66
Random Forest Regressor	0.95	0.65

Table 2: R-squared scores for 3 different models following removal of outliers.

Removal of the outliers significantly improved the model performance. Hyperparameter tuning is performed for XGBoost Regressor using GridSearchCV. R-squared value for the training data is 0.85 and for the test data is 0.67.

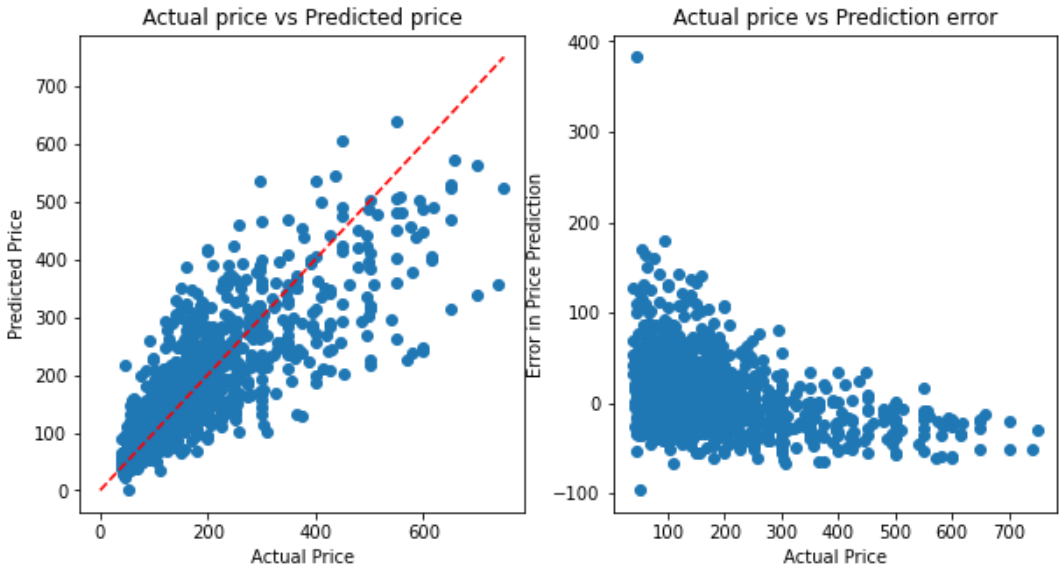

Figure 8: Actual price vs predicted price (left), Error in Price Prediction vs Actual Price (right) for the **test data** following removal of outliers and hyperparameter tuning.

Listings with low prices are predicted with large errors even after removal of outliers and hyperparameter tuning. One way to improve the model performance could be dividing the dataset based on price and train each dataset separately. % error distribution for various price ranges are plotted.

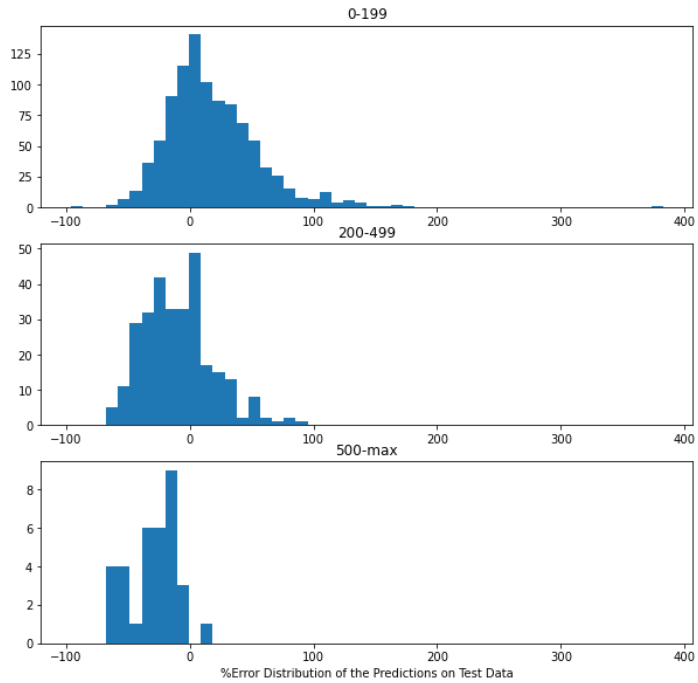

Figure 9: %Error distribution on price prediction with respect to price range ($0-199, $200-499, $500-max)

The distribution of %error reveal that, %error is mostly positive for prices lower than $200 and it is negative for prices higher than $200. The dataset is divided into two based on listing price and hyperparameter tuning was done for XGBoost Regressor model on both datasets.

Dataset	*Training Data*	*Test Data*
Price more than $200	0.78	0.29
Price less than $200	0.88	0.65
combined	0.85	0.67

Table 3: R-squared scores for XGBoost Regressor for 3 different models following hyperparameter tuning.

The likely reason why model has performed poorly on dataset with price more than $200 is the small size of the dataset. This problem could be overcome by using a larger dataset.

4. SUMMARY

The R-squared value of the XGBoost Regressor model for Airbnb price prediction is 0.66, meaning that the model can explain 66% of the variance of the Airbnb listing price. Two new variables are created to evaluate model performance:

% Error = 100 * (predicted price – actual price) / actual price

Absolute Error = predicted price - actual price

```
count    1305.000000
mean       10.567670
std        38.090377
min       -96.417501
25%       -13.935114
50%         4.046777
75%        30.970241
max       382.739766
Name: 0, dtype: float64
```

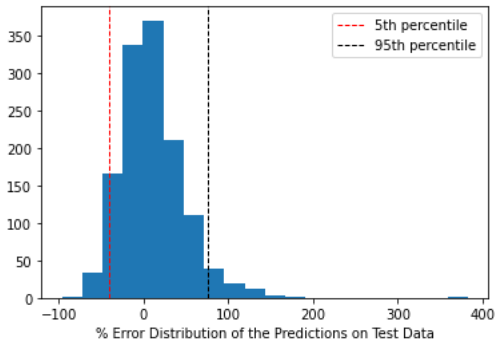

Figure 10: Descriptive statistics of the % Error (left), distribution of %Error (right)

```
count    1305.000000
mean        0.732301
std        69.274361
min      -384.110199
25%       -19.509892
50%         5.044113
75%        32.347427
max       240.521301
Name: 0, dtype: float64
```

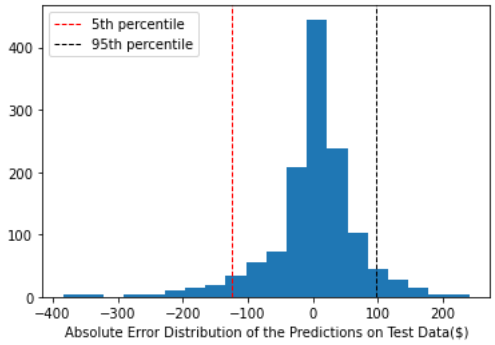

Figure 11: Descriptive statistics of the Absolute Error (left), distribution of Absolute Error (right)

90 % of the listings are within the range of [-40% to 75%] error in % Error distribution and [-$125 to $97] error in Absolute Error Distribution. % Error and Absolute Error distribution reveal that the model predicts the prices of the listings in low price range higher and vice versa.

Since the data used for the training of the model is not large (6882 listings) it can be impacted by the unreasonable pricing strategies of the hosts in the area. Some hosts can list their units with really high or low prices independent of the condition of the unit. This would make it hard for the model to explain the variance of the prices. This is a

limitation for our project. However, model performance can be improved by including location and how each listing is located to some hot spots or landmarks in the city.

5. APPENDIX

Definitions of the attributes of the listing data frame that are used in the model are listed below.

host_id: Id of the host
host_since: registration date of the host
host_response_time: Response time of the host
host_response_rate: Response rate of the host
host_acceptance_rate: Acceptance rate by the host
host_is_superhost: Is host a super host
host_listings_count: Number of listings that the host owns
host_total_listings_count: Number of listings that the host owns
host_verifications: Methods of verifications that host has done
host_has_profile_pic: Host has profile picture or not
host_identity_verified: Host has verified identity or not
neighbourhood: neighborhood information
neighbourhood_cleansed: neighborhood information
neighbourhood_group_cleansed: neighborhood information
latitude: latitude of the listing
longitude: longitude of the listing
property_type: sub-groups of room_type column
room_type: entire home, private room, shared room or hotel
accommodates: how many guests can a listing accommodate
bathrooms_text: number of bathrooms
bedrooms: number of bedrooms
beds: number of beds
amenities: amenities in the listing
price: price of the listing at the time of scraping
minimum_nights: minimum nights a listing available
maximum_nights: maximum nights a listing available
has_availability: listing is available or not at the time of the scraping
availability_30: availability in next 30 days
availability_60: availability in next 60 days
availability_90: availability in next 90 days
availability_365: availability in next 365 days
number_of_reviews
number_of_reviews_ltm: number of reviews lifetime
number_of_reviews_l30d: number of reviews last 30 days
first_review: date of first review
last_review: date of last review
review_scores_rating
review_scores_accuracy
review_scores_cleanliness
review_scores_checkin
review_scores_communication
review_scores_location
review_scores_value
instant_bookable: instant bookable or not
calculated_host_listings_count: Number of listings that the host owns

calculated_host_listings_count_entire_homes: Number of entire homes that the host owns
calculated_host_listings_count_private_rooms: Number of private rooms that the host owns
calculated_host_listings_count_shared_rooms: Number of shared rooms that the host owns
reviews_per_month

Definitions of the attributes of the calendar data frame that are used in the model are listed below.

listing_id: id of the listing
date: date of the pricing
available: listing available or not
price: price of the listing
adjusted_price: price of the listing
minimum_nights: minimum nights a listing available
maximum_nights: maximum nights a listing available