CA683 DATA ANALYTICS AND DATA MINING

Group	Group H
Names	Priyank Saklani(18210645), Vidita(18210438), Ahmad Faizan(18210897), Gaurav Shivhare(18211206), Harshali Patil(18212797)
Program	MCM
Module Code	CA683
Assignment Title	Data Analytics and Data Mining
Submission date	22/4/2019
Module Coordinator	Dr. Andrew McCarren

We declare that this material, which we now submit for assessment, is entirely our own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of our work. We understand that plagiarism, collusion, and copying are grave and serious offences in the university and accept the penalties that would be imposed should we engage in plagiarism, collusion or copying. We have read and understood the Assignment Regulations set out in the module documentation. We have identified and included the source of all facts, ideas, opinions, and viewpoints of others in the assignment references. Direct quotations from books, journal articles, internet sources, module text, or any other source whatsoever are acknowledged and the source cited are identified in the assignment references. This assignment, or any part of it, has not been previously submitted by me or any other person for assessment on this or any other course of study. We have read and understood the referencing guidelines found recommended in the assignment guidelines.

Names: Group 5
Date: 22/4/2019

SEATTLE AIRBNB

INTRODUCTION

In the last decade a new 'Sharing Economy Model' has exploded and astounded even the most optimists. Increased ownership of smartphones worldwide and better connectivity makes it even easier to access goods and services on-demand. AirBnb and Uber have been at the forefront of this trend.

PROJECT MOTIVATION

AirBnb claims that 'Shared Homes' creates an extra income for existing homeowners and provides local experience to visiting tourists, creating a win-win situation. Any home owner with spare space can list the space and can host an Airbnb listing. Listing price is decided by home owner and in no way reflects the listing market in the neighbourhood. Many home owners are not capable enough or don't want to research the listing market resulting in lower earnings. This project aims to use data generated by AirBnb in Seattle, Washington area to find interesting patterns and insights. We are trying to get the optimum price for the already existing listings and suggest them to the hosts.

RELATED WORK

[1]Emily Tang and Kunal Sangani has done similar work for San Francisco Airbnb where they considered social factors like host response, number of reviews etc. are important factors for room booking than conventional factors. To select features they used Recursive feature elimination (RFE), for image analysis they used standard bag of words and for sentiment features they used TextBlob package. The conclusion suggested that there were correlation between many factors; hence we tried to find the correlation and eliminate the correlated columns.

[2]Praneeth Guggilla used review count as an indicator of occupancy. They directly scraped the site and created new variables like zip code, street name and neighbourhood using latitude and longitude information which helped them reduce the number of variables to be considered for analysis. They used logarithmic transformation to deal with skewed variables like price, security deposit, extra_person_fee, etc. They observed correlation in the dataset, used SAS Enterprise Miner for variable reduction. They built models for price and price with text topics (amenities) and the variance was 68.74% and 69.78% respectively.

NOTE: Both of the above models had high variance and thus were overfitting.

[3]This paper discusses that geographically weighted regression (GWR) model performs better than general linear model (GLM) in terms of accuracy and affected variable selection. In GLM model, the dependent variables (price) is estimated with a set of dependent variables (Reviews, Age, H-Distance, C-Distance, Ratings) globally and in GWR model, the dependent variable is predicted by a series of explanatory variable sin which the estimated parameters can vary spatially i.e. relationship between price and location of each Airbnb listing. GWR model was a better for investigating Air listing price determinants.

[4]This research had two hypotheses (1) Provide an optimal price to Airbnb hosts to charge. Analyze similar listings in past to recommend an optimal the host should charge? (2) In case of unavailability of chosen place, they wanted to recommend guests with a likelihood of a listing

being available. They were able to identify correlated features and eliminated the reductant feature. They applied RandomForestRegessor on both balanced and imbalanced data confirming that high frequency of some samples were effecting the predictions.

DATASET

The dataset was taken from 'Inside Airbnb' website – http://insideairbnb.com/get-the-data.html

'Inside AirBnb' is an independent and non-commercialized platform which publishes publicly available information on AirBnb listings. We extracted data for listings in Seattle city for a time period of Jan 2019 to Jan 2020. Following is the brief summary for dataset:

File Name	Dimensions (rows X columns)	Information
Listing.csv	8.4K X 106	Contains information about listings. For Eg: Number of Bed Rooms, Listing Summary, Location and IsSuperHost etc.
Calendar.csv	3.09M X 7	Contains information about 365 days (Jan 2019 to Jan 2020) of each listing(8.4K X 365). For Eg: Price, Availability etc.
Reviews.csv	380.48K X 6	Contains reviews by users for listings.(Only verified visitors are allowed to post reviews)

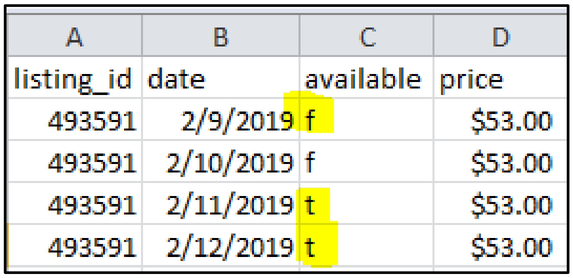
DATA CLEANING METHODS

- **Dropped rows having 'Availability' as 'True'**
  Availability column in Calendar.csv shows the availability of that particular listing on that particular day.

  True value implies that the listing is either not booked or is unavailable for any reason.

  False value implies that the listing is booked.

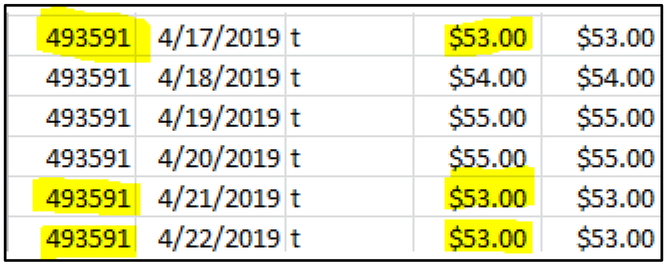
  Since we are interested in optimum price for an Airbnb listing we dropped all rows with Availability as True.



A	B	C	D
listing_id	date	available	price
493591	2/9/2019	f	$53.00
493591	2/10/2019	f	$53.00
493591	2/11/2019	t	$53.00
493591	2/12/2019	t	$53.00

Dropped the rows with 't'

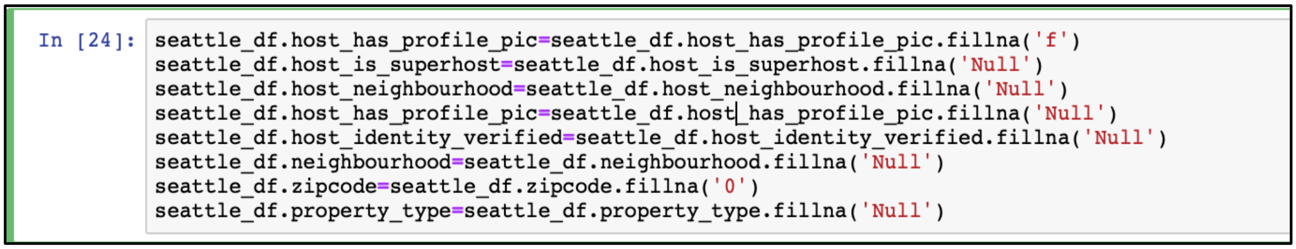
- **Dropped non-unique combinations of ‘Prices’ and ‘Listing_Id’s’**
  Since calendar.csv has 365 records (Jan 2019 to Jan 2020) for each Listing. Many rows were just duplicates (due to no variability in price of listing). Hence to remove duplication we created a ‘Key’ from combination of ‘Price’ and ‘Listing_Id’. We then dropped all the rows with duplicate ‘Key’.



493591	4/17/2019	t	$53.00	$53.00
493591	4/18/2019	t	$54.00	$54.00
493591	4/19/2019	t	$55.00	$55.00
493591	4/20/2019	t	$55.00	$55.00
493591	4/21/2019	t	$53.00	$53.00
493591	4/22/2019	t	$53.00	$53.00

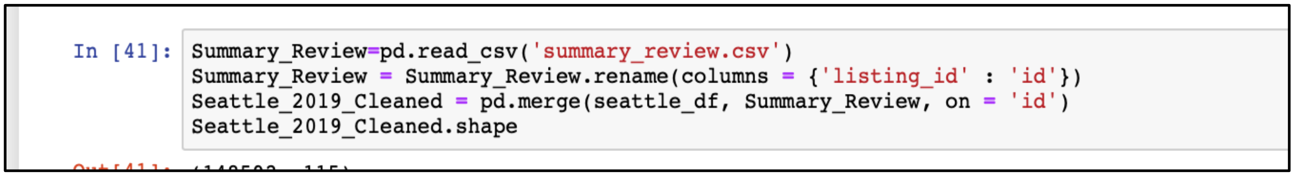
Removed Duplicate Rows

- **Handled Null Values (N/A) in Listing.csv**
  Replaced N/A values with “NULL” string to handle it more accurately.



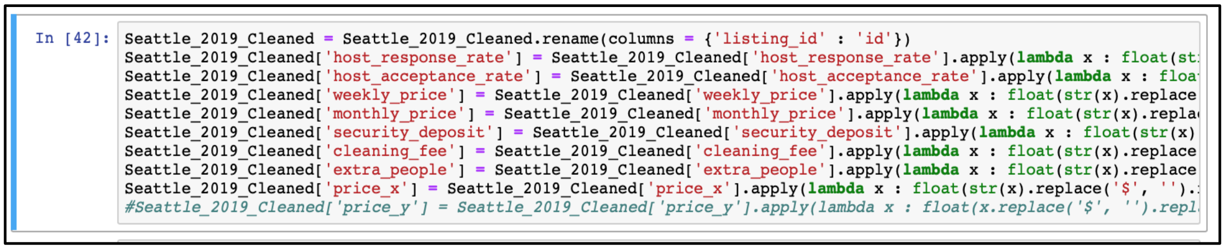
```
In [24]: seattle_df.host_has_profile_pic=seattle_df.host_has_profile_pic.fillna('f')
         seattle_df.host_is_superhost=seattle_df.host_is_superhost.fillna('Null')
         seattle_df.host_neighbourhood=seattle_df.host_neighbourhood.fillna('Null')
         seattle_df.host_has_profile_pic=seattle_df.host_has_profile_pic.fillna('Null')
         seattle_df.host_identity_verified=seattle_df.host_identity_verified.fillna('Null')
         seattle_df.neighbourhood=seattle_df.neighbourhood.fillna('Null')
         seattle_df.zipcode=seattle_df.zipcode.fillna('0')
         seattle_df.property_type=seattle_df.property_type.fillna('Null')
```

- **Merged Calendar.csv with Listing.**



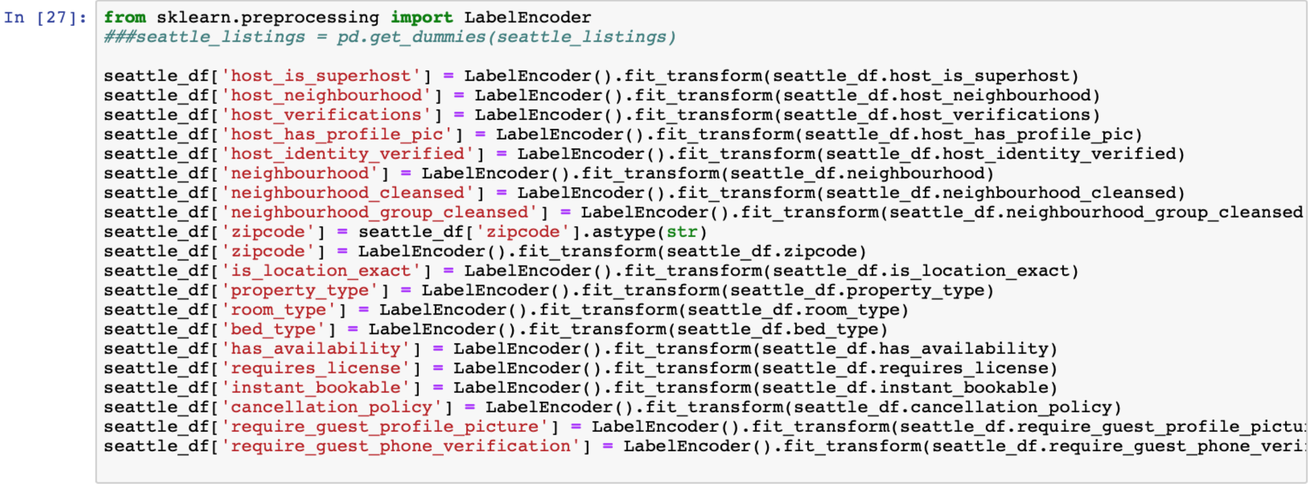
```
In [41]: Summary_Review=pd.read_csv('summary_review.csv')
         Summary_Review = Summary_Review.rename(columns = {'listing_id' : 'id'})
         Seattle_2019_Cleaned = pd.merge(seattle_df, Summary_Review, on = 'id')
         Seattle_2019_Cleaned.shape
```

- **Corrected the format type in the CSV**
  We corrected the format for many columns to the default type for the respective categories. Replaces unnecessary like “$” ,”%” etc after price details.



```
In [42]: Seattle_2019_Cleaned = Seattle_2019_Cleaned.rename(columns = {'listing_id' : 'id'})
         Seattle_2019_Cleaned['host_response_rate'] = Seattle_2019_Cleaned['host_response_rate'].apply(lambda x : float(st
         Seattle_2019_Cleaned['host_acceptance_rate'] = Seattle_2019_Cleaned['host_acceptance_rate'].apply(lambda x : floa
         Seattle_2019_Cleaned['weekly_price'] = Seattle_2019_Cleaned['weekly_price'].apply(lambda x : float(str(x).replace
         Seattle_2019_Cleaned['monthly_price'] = Seattle_2019_Cleaned['monthly_price'].apply(lambda x : float(str(x).repla
         Seattle_2019_Cleaned['security_deposit'] = Seattle_2019_Cleaned['security_deposit'].apply(lambda x : float(str(x)
         Seattle_2019_Cleaned['cleaning_fee'] = Seattle_2019_Cleaned['cleaning_fee'].apply(lambda x : float(str(x).replace
         Seattle_2019_Cleaned['extra_people'] = Seattle_2019_Cleaned['extra_people'].apply(lambda x : float(str(x).replace
         Seattle_2019_Cleaned['price_x'] = Seattle_2019_Cleaned['price_x'].apply(lambda x : float(str(x).replace('$', '').
         #Seattle_2019_Cleaned['price_y'] = Seattle_2019_Cleaned['price_y'].apply(lambda x : float(x.replace('$', '').repl
```

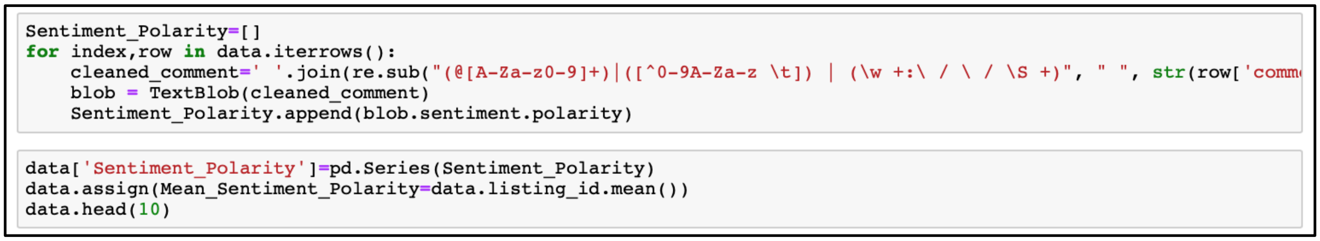
- **Encoded categorical value**
  We encoded the categorical features of our dataset so that it could be provided to ML algorithms to do a better job. For encoding we used the ‘Label Encoder’ feature of ‘Sklearn’ python library.



```
In [27]: from sklearn.preprocessing import LabelEncoder
         ###seattle_listings = pd.get_dummies(seattle_listings)

         seattle_df['host_is_superhost'] = LabelEncoder().fit_transform(seattle_df.host_is_superhost)
         seattle_df['host_neighbourhood'] = LabelEncoder().fit_transform(seattle_df.host_neighbourhood)
         seattle_df['host_verifications'] = LabelEncoder().fit_transform(seattle_df.host_verifications)
         seattle_df['host_has_profile_pic'] = LabelEncoder().fit_transform(seattle_df.host_has_profile_pic)
         seattle_df['host_identity_verified'] = LabelEncoder().fit_transform(seattle_df.host_identity_verified)
         seattle_df['neighbourhood'] = LabelEncoder().fit_transform(seattle_df.neighbourhood)
         seattle_df['neighbourhood_cleansed'] = LabelEncoder().fit_transform(seattle_df.neighbourhood_cleansed)
         seattle_df['neighbourhood_group_cleansed'] = LabelEncoder().fit_transform(seattle_df.neighbourhood_group_cleansed
         seattle_df['zipcode'] = seattle_df['zipcode'].astype(str)
         seattle_df['zipcode'] = LabelEncoder().fit_transform(seattle_df.zipcode)
         seattle_df['is_location_exact'] = LabelEncoder().fit_transform(seattle_df.is_location_exact)
         seattle_df['property_type'] = LabelEncoder().fit_transform(seattle_df.property_type)
         seattle_df['room_type'] = LabelEncoder().fit_transform(seattle_df.room_type)
         seattle_df['bed_type'] = LabelEncoder().fit_transform(seattle_df.bed_type)
         seattle_df['has_availability'] = LabelEncoder().fit_transform(seattle_df.has_availability)
         seattle_df['requires_license'] = LabelEncoder().fit_transform(seattle_df.requires_license)
         seattle_df['instant_bookable'] = LabelEncoder().fit_transform(seattle_df.instant_bookable)
         seattle_df['cancellation_policy'] = LabelEncoder().fit_transform(seattle_df.cancellation_policy)
         seattle_df['require_guest_profile_picture'] = LabelEncoder().fit_transform(seattle_df.require_guest_profile_pictu
         seattle_df['require_guest_phone_verification'] = LabelEncoder().fit_transform(seattle_df.require_guest_phone_veri
```

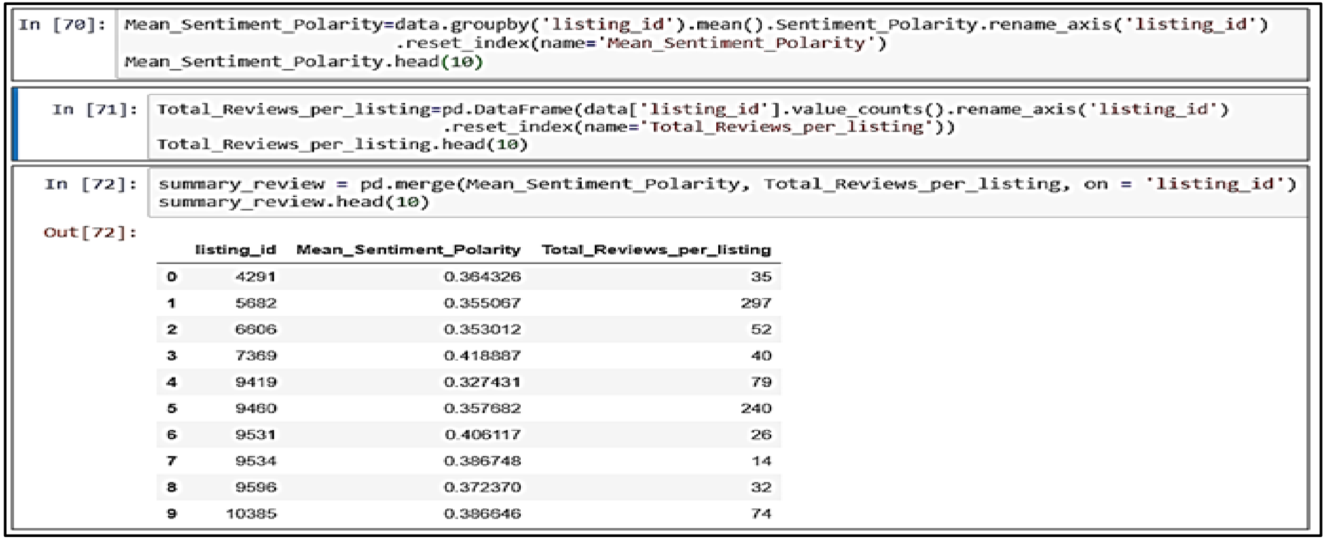
- **Sentiment score from reviews**
  We extracted the sentiment polarity of all the reviews from review.csv using ‘TextBlob’ library of python. ‘TextBlob’ library uses ‘Vader Sentiment Lexicon’ under the hood and returns the polarity value between -1 to 1.



```
Sentiment_Polarity=[]
for index,row in data.iterrows():
    cleaned_comment=' '.join(re.sub("(@[A-Za-z0-9]+)|([^0-9A-Za-z \t]) | (\w +:\ / \ / \S +)", " ", str(row['comm
    blob = TextBlob(cleaned_comment)
    Sentiment_Polarity.append(blob.sentiment.polarity)

data['Sentiment_Polarity']=pd.Series(Sentiment_Polarity)
data.assign(Mean_Sentiment_Polarity=data.listing_id.mean())
data.head(10)
```

- **Merged Sentiment Score with calendar.csv and listing.csv.**
  We extracted the average sentiment score from previous step for each listing and merged it with previous files to form a final cleaned CSV.



```
In [70]: Mean_Sentiment_Polarity=data.groupby('listing_id').mean().Sentiment_Polarity.rename_axis('listing_id')
                                .reset_index(name='Mean_Sentiment_Polarity')
         Mean_Sentiment_Polarity.head(10)

In [71]: Total_Reviews_per_listing=pd.DataFrame(data['listing_id'].value_counts().rename_axis('listing_id')
                                .reset_index(name='Total_Reviews_per_listing'))
         Total_Reviews_per_listing.head(10)

In [72]: summary_review = pd.merge(Mean_Sentiment_Polarity, Total_Reviews_per_listing, on = 'listing_id')
         summary_review.head(10)

Out[72]:
```

	listing_id	Mean_Sentiment_Polarity	Total_Reviews_per_listing
0	4291	0.364326	35
1	5682	0.355067	297
2	6606	0.353012	52
3	7369	0.418887	40
4	9419	0.327431	79
5	9460	0.357682	240
6	9531	0.406117	26
7	9534	0.386748	14
8	9596	0.372370	32
9	10385	0.386646	74

DIMENSIONALITY REDUCTION METHODS

After data cleaning we took all the values that we needed and created a new csv file. This resulted in 135 columns. On closer inspection we found that many features were highly correlated or were long text values with no immediate significance. So we tried to reduce dimensionality using below mentioned steps:

- Performed analyses for correlation among features and removed highly correlated redundant features to reduce the number of features.
- Dropped columns with long text values eg: Listing_Summary, Neighbourhood_Summary (Summaries written by hosts of listings). This type of information is already covered by other columns.
- Used SelectKbest method from sklearn.feature_selection to find the K best features. It takes features one by one and compares them using T-Test against Target (Price in this case) and returns a score for each feature. Score for each feature suggests 'If the feature is significantly similar to target or not'. [5]
- We also got a list of important features for predicting prices for Airbnb listings by Literature Survey (mentioned above) which matched closely with our findings from above steps.

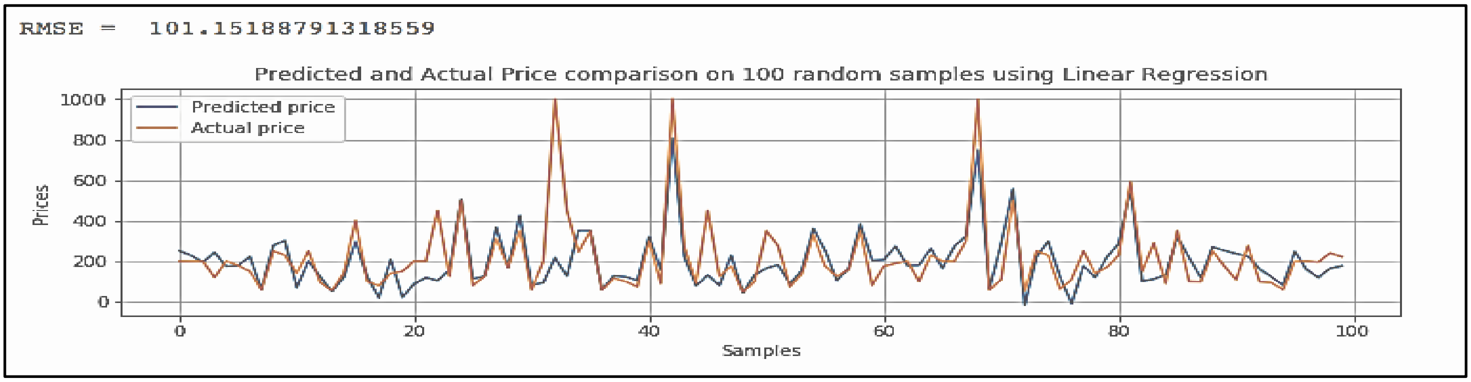
After all the reductions we were left with 35 features which were used for modelling. Dimensionality reduction helped in adverting overfitting and also sped up the performance.

RESULTS AND DISCUSSION

1. LINEAR REGRESSION

The first model we choose was linear regression to generate a benchmark score for our other ML Algorithms. Assumptions of linear regression were fulfilled as our cleaned dataset was free from correlations and multicollinearity.

We observed the RMSE score of 101.51 for selected 35 features.



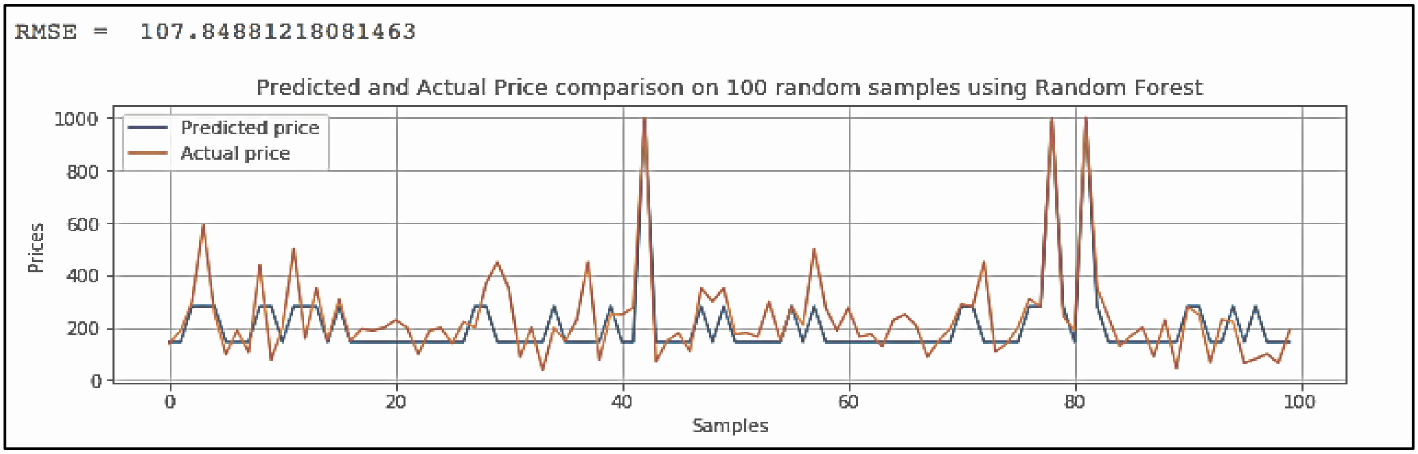
```
RMSE =  101.15188791318559
```

Predicted and Actual Price comparison on 100 random samples using Linear Regression
Predicted price
Actual price
Prices
1000
800
600
400
200
0
0
20
40
60
80
100
Samples

2. RANDOM FORREST

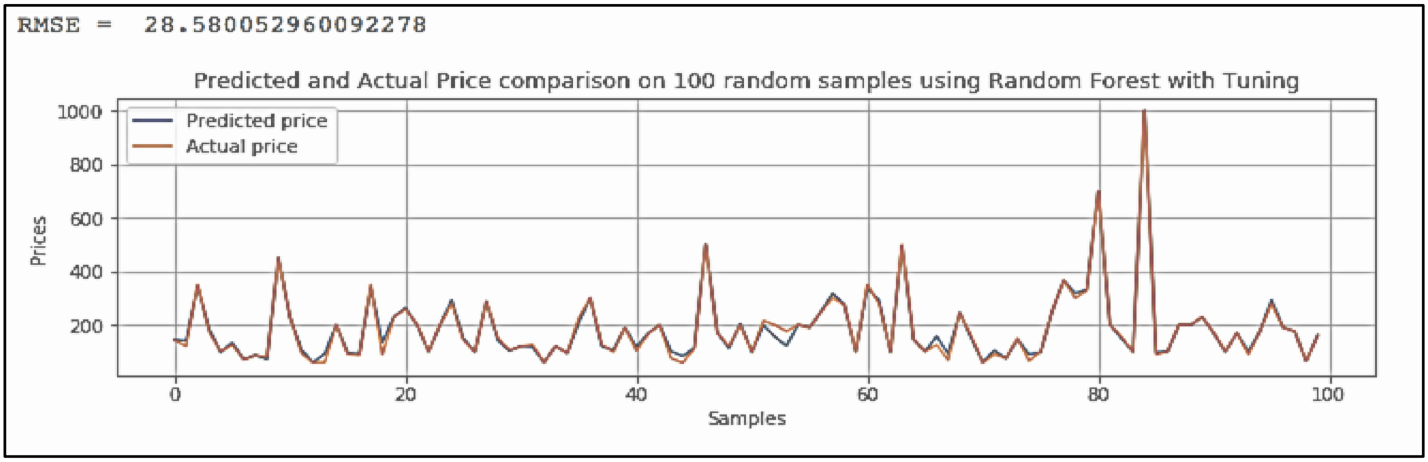
Since most of the dataset and features that we chose as part of feature selection were categorical data, we selected Random Forest as our next model. Random Forrest algorithm builds multiple decision trees and merge them together using ‘Bagging’ techniques. The general idea behind ‘Bagging’ is that a combination of many weak learners will eventually result in more accurate and stable predictions.

Random Forest gave us RMSE score of 107.8 and with fine tuning the score improved to 28.58.



```
RMSE =  107.84881218081463
```

Results before Fine-Tuning



```
RMSE =  28.580052960092278
```

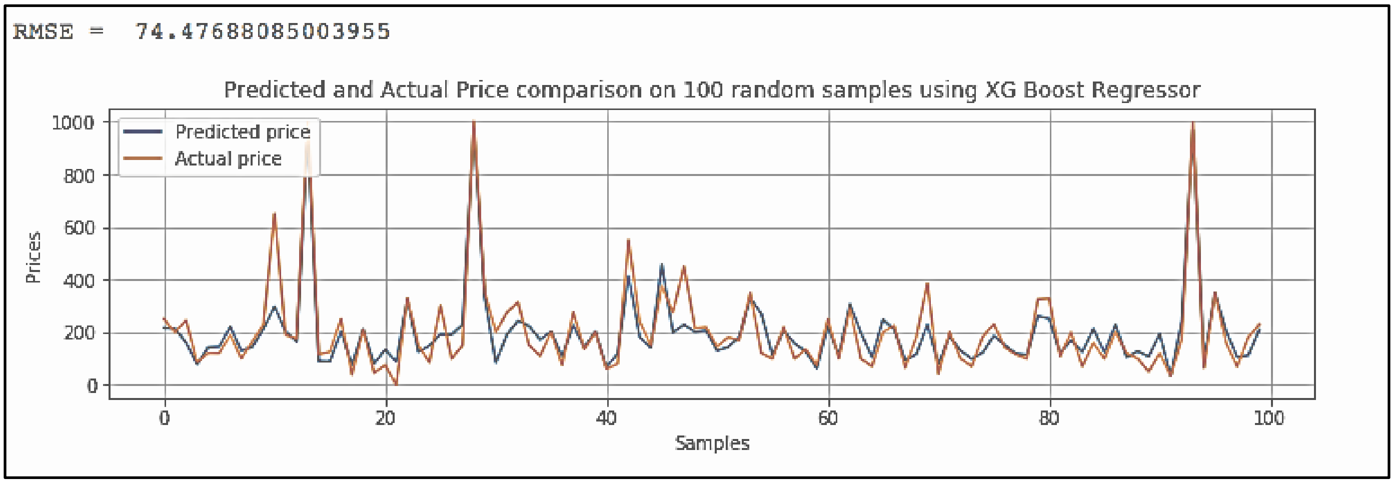
Results after Fine-Tuning

3. XG BOOST

[6]XGBoost stands for extreme gradient boosting and is an implementation of performance optimised gradient boosted decision trees. It is designed for speed and performance.

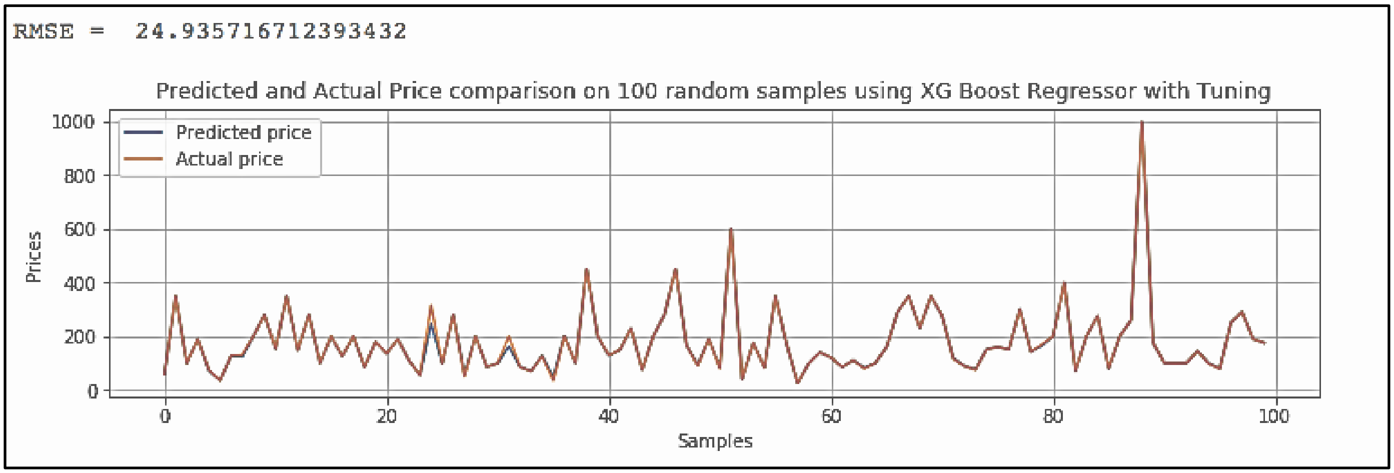
Gradient Boosting is an ensemble technique where new models are added to correct the errors made by existing models. This process is repeated until residuals become random with mean as zero. This results in better accurate models. However caution should be taken while attempting to reduce errors by this method or it can result in overfitting.

XGBoost for selected features gave us RMSE score of 74.47 and with tuning it improved to 24.93.



```
RMSE =  74.47688085003955
```

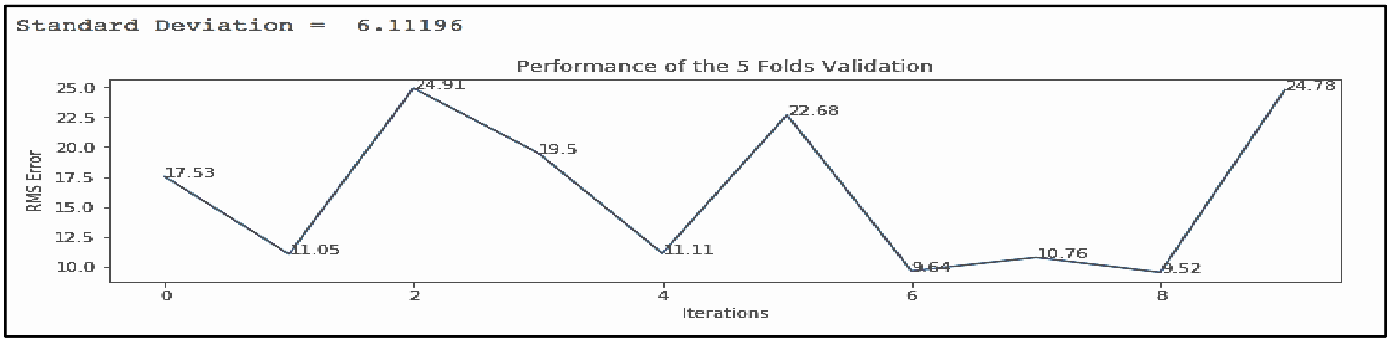
Results before Fine-Tuning



```
RMSE =  24.935716712393432
```

Results after Fine-Tuning

For further verification of independence of model accuracy from data and to test for overfitting we applied 5-fold validation technique on our last model.



```
Standard Deviation =  6.11196
```

As evident by low standard deviation, model is reasonably accurate and stable. Also there seems to be no evidence of overfitting.

CONCLUSION

Our main criteria for analyses was Root mean squared error (RMSE) score for the models which should be less than standard deviation of target variable. Following are the main highlights of our analyses.

- By merging files, we were able to generate new features like review score for each listing.
- Due to repeated nature of calendar data, we had to reject a lot of non-unique rows.
- We were able to reduce dimensionality with help of correlation and dropping text features.
- Our best model was XGBoost model with fine tuning, having RMSE value = 24 (<sd(target variable) = 161).

Some features were not used by us during this project. These were majorly text based features and would need natural language processing to convert into features. For future work, following features can be used.

- Amenities (Text)
- Street
- Description (Text)
- Reviews (Text)
- Interactions between owner and guest (Text)
- Ratings

RESOURCES

All the supporting documents, Python Codes, Raw files, Cleaned and Processed files and referenced papers are stored on Google drive and shared. To access please use following link: **https://drive.google.com/drive/folders/1xwBXnHJShopawsc0ijGSu0Vcfg5gGfBF**

Bibliography

[1] K. S. Emily Tang, "Neighborhood and Price Prediction for San Francisco Airbnb Listings," Stanford University, 2015.

[2] S. G. D. C. Praneeth Guggilla, "Price Recommendation Engine for Airbnb," Oklahoma State University, Oklahoma , 2017.

[3] R. J. C. C. L. D. H. L. Y. Zhihua Zhang, "Key Factors Affecting the Price of Airbnb Listings: A Geographically Weighted Approach," *Sustainability,* vol. 9, no. 9, p. 1635, September 2017.

[4] A. J. R. B. Paridhi Choudhary, "Unravelling Airbnb Predicting Price for New Listing," Carnegie Mellon University, 2018.

[5] P. Dhandre, "Hub.packtpub.com," 14 March 2018. [Online]. Available: https://hub.packtpub.com/selecting-statistical-based-features-in-machine-learning-application/.

[6] P. Grover, "Gradient Boosting from scratch," Medium, 9 Dec 2019. [Online]. Available: https://medium.com/mlreview/gradient-boosting-from-scratch-1e317ae4587d.