BI-CAPSTONE
Airbnb Recommendation System

SUBMITTED BY

AKANKSHA SHUKLA, MIHIR MIRAJKAR, PIYUSH CHOUDHARY

INTRODUCTION

When renting out an house, it is often a conundrum to decide upon a competitive price would seem enticing on the market.

To help with this, we have developed a model for owners which recommends the price range in which the house should be rented out on based on aspects like amenities the house has, location of the house and reviews score of house.

This system is built upon concepts like Sentiment Analysis, Time Series Forecasting and Supervised/Unsupervised Learning. Using these we were able to predict the price range of any given house which a an accuracy of 40%.

Part 1: Time Series Forecasting

- We cleaned calendar dataset i.e. removed rows where prices were null and removed '$' from price so as to get clean dataset.
- Then calculated the average price on daily basis from calendar dataset using python and saved it in file daily_price.csv
- Loading this file in R, we find that this time series data is non-stationary. So as to make it stationary, first order differencing is performed. Using ARIMA(3,0,2), we fit the model.
- We use Ljung-Box test which gives higher p-value and so we fail to reject the null hypothesis and conclude that it is good fit. ‘Forecast’ package enables us to find the forecasting of this price of houses.

Part 1: Analysing data to find week days having minimum/maximum cost

- Apart from time series data forecasting, we try to answer one business question
- Question: Which days in a week will have higher cost/lower cost of houses? We can provide this information to users which will help them plan decisively.
- So as to achieve this, we check prices for each listing based on weekdays from **calendar dataset**. Taking base as sunday, we create dataframe consisting results for each day of week for each listing_id. We see that averge value shows that the price is highest on friday and Saturday

Part 1: Analyzing data to find week days having maximum cost

- This gives result as :

'Wed': 0.9930306441057060 7,
'Fri': 1.029393190209331,
'Mon': 0.9954617738791223 8,
'Sat': 1.031146576455956
'Sun': 1.0,
'Tue': 0.9924598046738485 3,
'Thu': 0.9987980713824251 6,

- As we can see from the result, Sunday has value 1.0, Friday and Saturday has approximate 130% of Sunday's cost. Thus, relative price we can estimate as % increase/decrease for other days.

Part 2: Sentiment Analysis

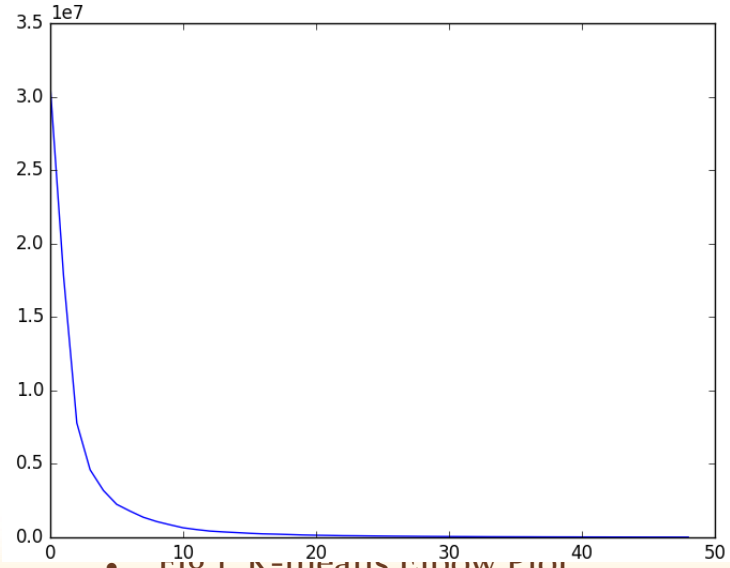
- From the available reviews of each listing, we attempted to generate scores for each listing.
- Each reviews were filtered for stop words to make a more robust sentiment analysis.
- Using 2 lists gathered from [1] and [2], we constructed potential positive, negative lists. These lists were inclusive of commonly committed spelling mistakes in reviews.
- We generated scores from these reviews by calculating the frequency of positive and negative words. The scores were normalized to the range 0 to 2, with 0 being completely negative and 2 being completely positive.

Part 3: Forecasting new Prices

- Using the given data, we extract 365 prices (for each day) for each listing.
- Performed some data cleansing like replacing '$' symbol and hadling commas, etc. Used python's 'locale.atof' for most of this.
- For handling the missing data, listings that don't have base price given for a particular day, the most recent price available for that listing is used to populate those entries.
- Using auto.arima on the the time series data (365 prices of each day for each listing), we attempt to forecast the price of each listing for the next 30 days.
- To list the new price, the mean of the next 30 days' fore-casted prices is taken.

Part 4: Clustering and Training a Prediction model

- Using the new prices (mean of the predicted prices), we attempt clustering the prices of the listing into 10 clusters.
- We used k-means and decided the number of clusters based on plotting the SSE on various centers (i.e elbow plot Fig 1)



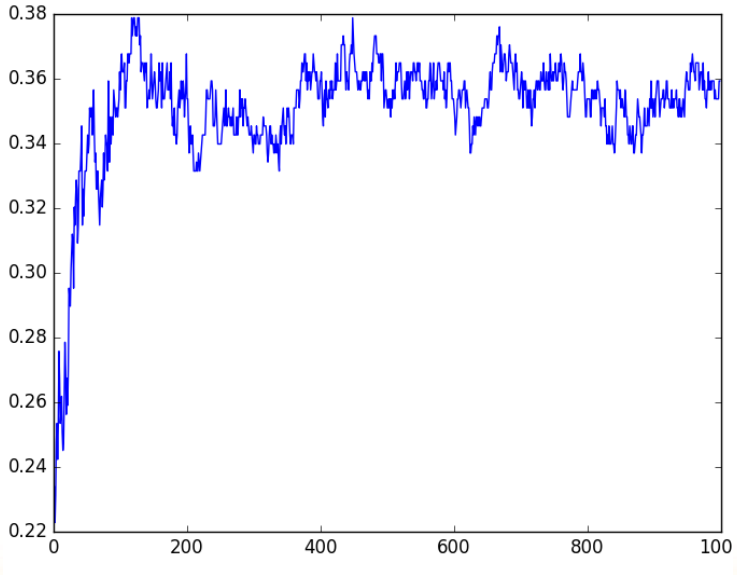
- Fig 1: K-means Elbow Plot

Part 4: Clustering and Training a Prediction model (contd…)

- Then for each listing, we assign labels as the mean of the cluster it is a part of. This way we could compress labels from being unique for each listings into groups for different ranges of prices.
- Based on these labels, and using features like amenities (like pool, TV, Refrigerator,Washer/Dryer, etc), location of the listing and the review score we next, train the model.
- Based on this set of features and labels, we attempted to form a prediction model using multiple classifiers (SVM's, Random Forest, Decision Trees and KNNs)

Part 4: Clustering and Training a Prediction model (contd…)

- We found the best result using KNN's using 117 neighbors. The decision to select this number of neighbors was made after plotting a graph of neighbor vs accuracy. The graph peaked at 117.



Part 4: Training Model (contd…)

- After tuning the model, using a combination of parameters we achieved an accuracy of ~40%.
- Though accuracy seems low, it didn't have a huge impact on our objective. This is because the ground truth was also set by us, so an approximation was sufficient to get decent enough results.
- Using this classification model, we predicted the price label of a newly put up listing. This prediction is then matched with the cluster, and the highest and lowest price values in that cluster is then used the price range which is presented to the owner of the new listing.

Part 5: The Interface

- To use the system, the following steps are followed
- The amenities are displayed, from which the user selects the amenities that are present in the house that is being put up for the rent.
- Next, the location of the place is required as an input. The location is entered in terms of coordinates. In the future using some map api we can simply implement a drag-and-drop feature.
- The model then predicts, the optimal price, the range of price that should be considered when deciding the price, and the price respective to the day of the week.

Git Hub Portfolio

https://github.ncsu.edu/mmmirajk/Airbnb-Recommendation-System

References:

[1] Minqing Hu and Bing Liu. "Mining and Summarizing Customer Reviews."

[2] Bing Liu, Minqing Hu and Junsheng Cheng. "Opinion Observer: Analyzing and Comparing Opinions on the Web."

[3] G. Peter Zhang "Time series forecasting using a hybrid ARIMA and neural network model"

[4] Alexander Pak, Patrick Paroubek "Twitter as a Corpus for Sentiment Analysis and Opinion Mining"

[5] Zhenyun Deng et al "Efficient kNN classification algorithm for big data"