IIITB ML Project : Rental Listing Inquiries

Pushkar Kulkarni - MT2018088
Rishabh Jain - MT2018094
Suryansh Jain - MT2018123

December 7, 2018

Contents

1 Introduction

This report summarises the analysis and predictions done on Rental Hop Listings dataset as a part of Machine Learning course. The data used in this project was a part of Two Sigma Connect : Renthop Listing Enquiries competition on kaggle. The data provides information regarding various apartment listings in the New York region, their features, pricing, and the level of interest that these apartments generate in consumers.
In this project, we attempt to do some predictions from the dataset using various machine learning models.

2 Abstract

We use various machine learning models in order to predict interest level of consumers for apartments in the New York region from the available dataset. This project mainly involves cleaning of dataset, data processing, feature engineering, and finally fitting models over our training data. We have used Random forest classifier and XGBoost as our models.

Various combinations of features are experimented upon on these models to obtain better accuracy on test data.

This project report describes the above steps in detail with test models and metrics at the end.

3 About

3.1 Problem Statement

In this competition, we will predict how popular an apartment rental listing is based on the listing content like text description, photos, number of bedrooms, price, etc. The data comes from renthop.com, an apartment listing website. These apartments are located in New York City.

The target variable, interest_level, is defined by the number of inquiries a listing has in the duration that the listing was live on the site. So, the problem statement can be summarised as ”How much interest will a new rental listing on RentHop receive?”

3.2 Data description

train.json contains 39481 entries with 15 features including the target label. Data is as follows:

- bathrooms: number of bathrooms
- bedrooms: number of bathrooms
- building_id
- created
- description
- display_address
- features: a list of features about this apartment
- latitude
- listing_id
- longitude
- manager_id
- photos: a list of photo links. You are welcome to download the pictures yourselves from renthop site, but they are the same as imgs.zip.
- price: in USD

- street_address
- interest_level: this is the target variable. It has 3 categories: high, medium, low

Above list shows all the features available to us.

4 Exploratory Data Analysis

4.1 Distribution of target label

Following pie chart with color coding shows the distribution of interest_level over our training data:

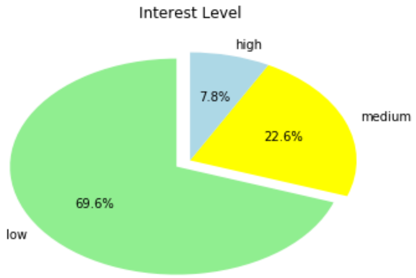

Figure 1: Distribution of interest_level across datapoints

4.2 Ditribution of created dates

Using to_datetime method of the pandas library, 'created' feature can be converted to date time format and date can be extracted into a separate column. Distribution of datapoints over date is shown as below:

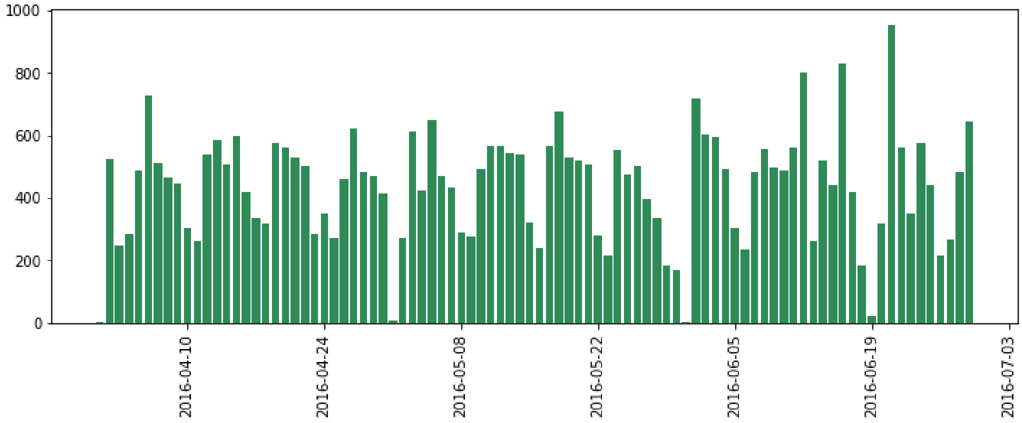

Figure 2: Distibution of created date across datapoints

We see that all dates are within a small range of 6 months, thus they may not be a major detrimental factor for interest generation.

4.3 Distribution of price

Checking the distribution of price over all listed apartments using a scatter plot:

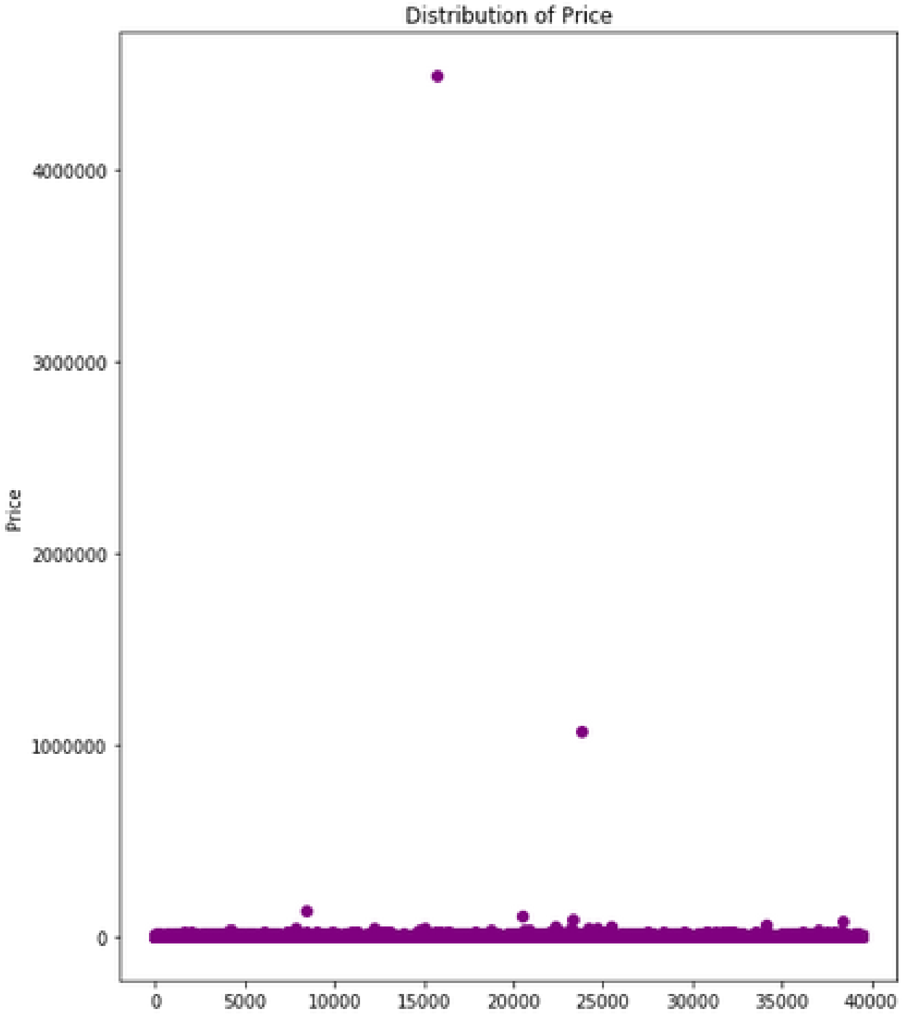

Figure 3: Price distribution with outliers

We can see that the distribution has some outliers, so we remove them by removing all points that lie above the upper limit of 99th percentile of datapoints. The result is as shown:

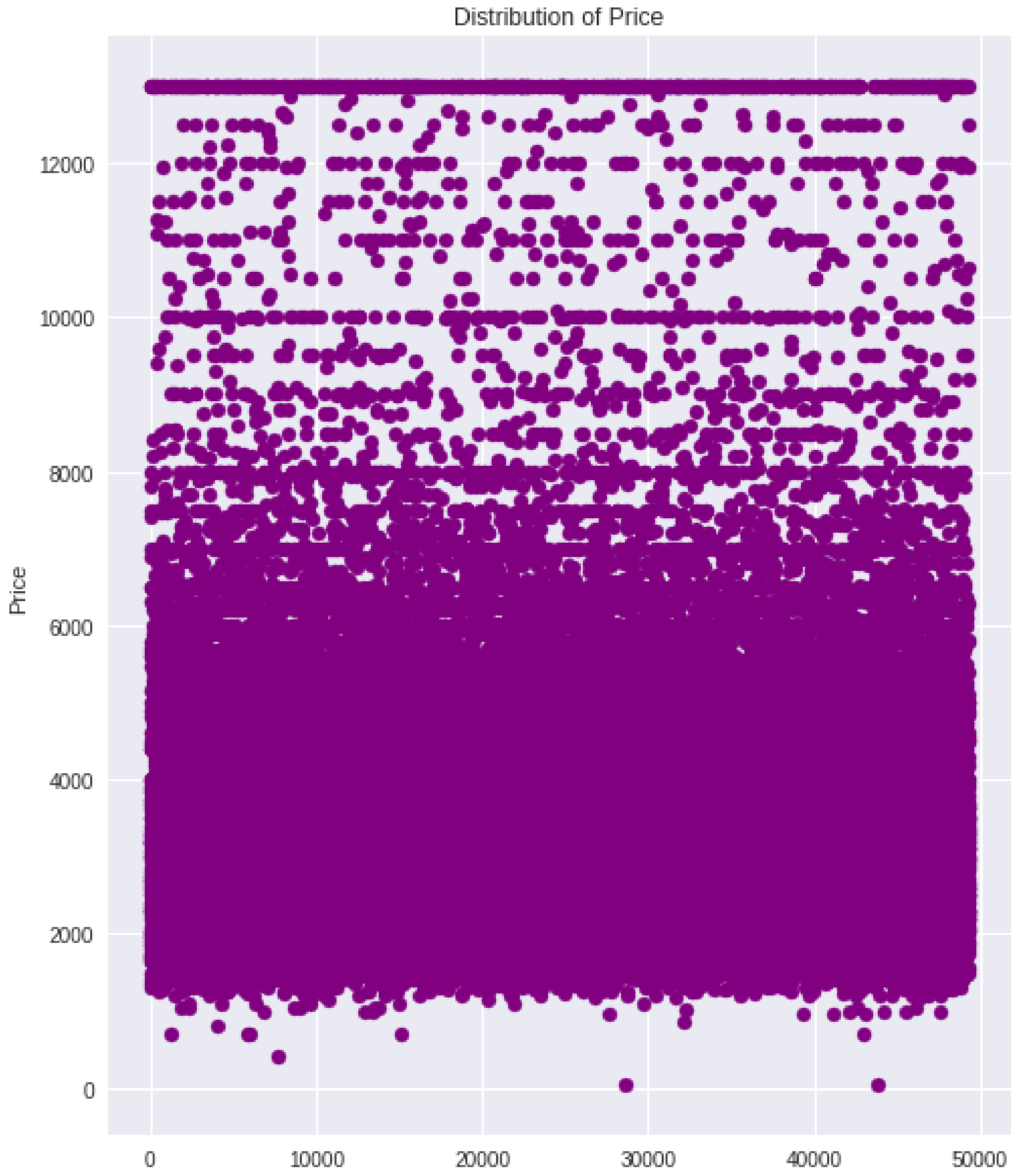

Figure 4: Price distribution without outliers

4.4 Plotting price vs interest level

Creating stripplot using seaborn library and plotting a distribution visualizing price of apartments with respect to the given interest levels so as to get an estimate of the distribution of price over the interest levels and see if the price data is skewed towards a certain interest level or vice versa. Following is the visualization:

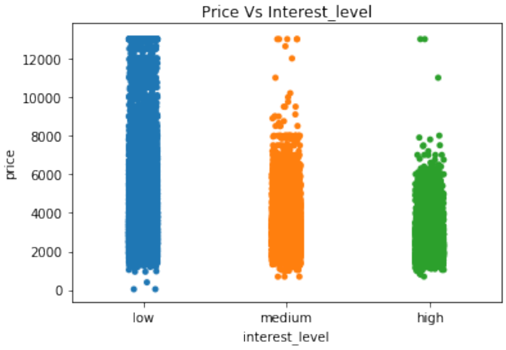

Figure 5: Price vs Interest Level

We see that majority of apartments with high prices have lower interest levels and very few have medium and high levels, which makes sense.

4.5 Analysing high profile managers

Some managers have more listings as compared to others so they have more influence over customers and their interest levels. We look at managers having over 20 listings with high interest levels, as high levels are low and overall listings more than 80. Figure 5 shows the result.

We see that there are 8 such high profile managers and they are responsible for a lot of listings with high level interest.

4.6 Analysing high profile buildings

Similar to managers, we can also classify buildings as high profile where a certain building has more apartments with listings corresponding to high interest level. We see that there are 5 such buildings with specified characteristics in Figure 6.

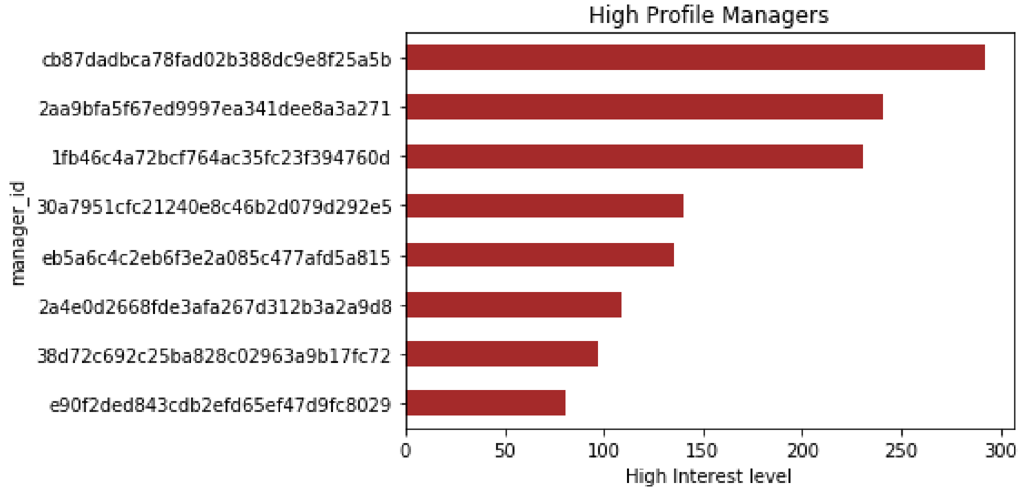

Figure 6: High profile managers

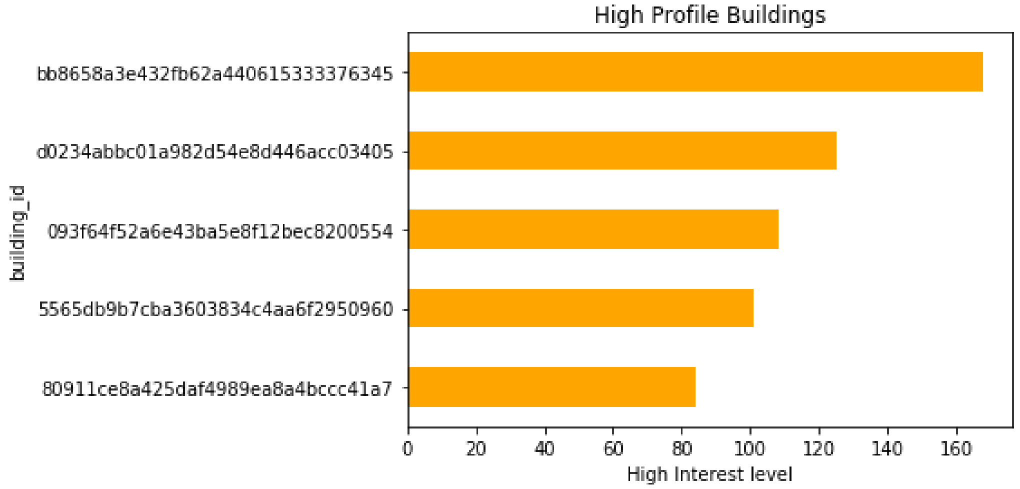

Figure 7: High profile buildings

4.7 Plotting location vs interest level

We try and cluster locations and assign each cluster a specific label. Further we check to see if a particular cluster of locations generates high or low interest levels as compared to others. We use stripplot from seaborn library. Figure 8 shows the result.
We see some locations generate low level interests but they are absent in medium and high columns implying that these locations are of more priority to customers.

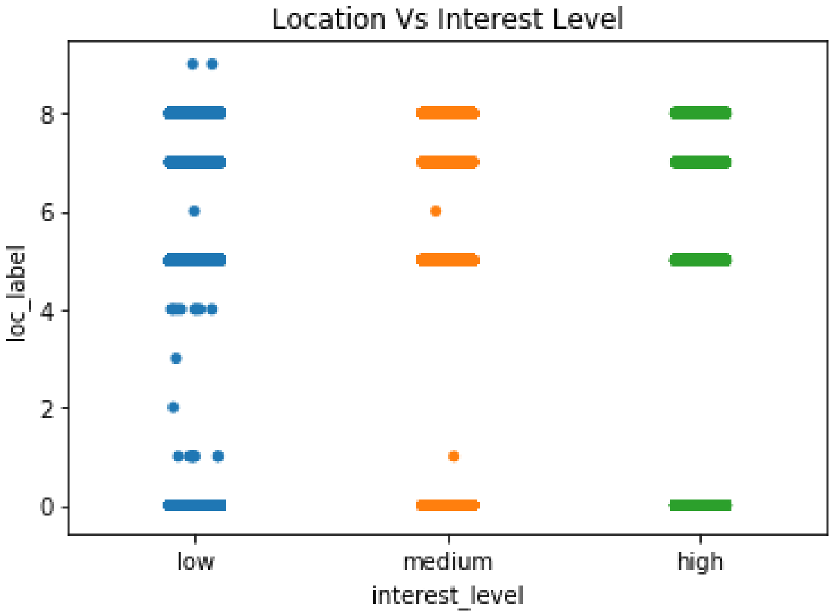

Figure 8: Location vs Interest Level

5 Data preprocessing

This section deals with cleaning and processing of data.

5.1 Missing values

Data did not consist of any NaN or missing values. So we take a deeper look into features and see if we can do enhancements or replacements to make the data more informative and useful.

5.2 Price

This has been discussed in previous section. We found a few outliers upon visualizing the price on a scatterplot. So, we removed them by replacing all points that lie above the upper limit of 99th percentile of datapoints.

5.3 Replacing keywords in features

Replaced redundant features like 24\7, 24— hour, 24hr with 24 and ft_doorman, 24_doorman, 24_hr_doorman with doorman. Similarly, this is done for all other features so as to make them more homogenous.

5.4 Label encoding

We label encode the categorical features so as to use them in our models. Features that are label encoded are namely display_address, manager_id, building_id, street_address.

Next we create new features from already available raw features for our models.

6 Feature engineering

This section deals with creating new features for the purpose of making them more utilizable and informative.

6.1 Date and time

Using pandas to_datetime function we transform the created feature to datetime format. Then we split the column and create new features with different parts of the date namely created_year, created_month, created_day, created_hour.
Then we create hr_sin, hr_cos, mnth_sin, mnth_cos from created_hour and created_month in radians using numpy. This is done to scale the time.
This new feature will help the model to know if apartments created earlier or during a specific period attracted more attention from the customers.

6.2 Location label

Location of an apartment is given to us in the form of latitude and longitude. Using k means clustering, we cluster these datapoints and create a label out of these as previously explained in the EDA section.
Thus each of our locations is now clustered into a label which is represented by a new feature loc_label.
This is done as we know that few locations attract more interest than others.

6.3 Numerical features

Features named features is given in form of a list, description is a text feature, and we also have photos for a apartment. We assume that the listings having more details attract more consumer interest as flow of more information gives them better perspective of the apartment and hence create more interest.
Taking this into account we create features with length of the mentioned features and create new columns namely num_photos, num_description_words, and num_features.

6.4 Room features

The cost of an apartment is surely not independent of the area of the apartment. Since, we do not have the area of an apartment but are given number of bedrooms and bathrooms that in it. To take this point into consideration

we create features that consist price per bedroom namely price_t and another that takes in price per bedroom + bathroom namely price_per_room.
This makes two features that consider the price of an apartment based on its size and the number of rooms.

6.5 Manager level

Some managers might have more influence and more listings over the data which directly relates to their skill of affecting the interest level of a customer. We categorise manager levels as low, medium, and high according to the number of listings they have in a building. We do so by first defining a building_level which consists of the managers corresponding to apartment listings. Then for each listing we check the interest level and add it to the corresponding manager level column. Hence, we get three new features namely manager_level_low, manager_level_medium, and manager_level_high.

Now we move on to using these newly created features and some raw features to create and test various models to solve our problem.

7 Models and test metrics

This deals with creating and using various sklearn models and our features to try and build a model for predicting consumer interest.

7.1 Random forest classifier

We use the random forest classifier with numerical features described in section 6.3 along with the following raw features bathrooms, bedrooms and price.
Using this we get the following accuracy scores,

Cross validation with 0.6 train_test_split : 0.738
Public score : 0.732

Since we did not get much positive results using random forest classifier, we move on to XGBoost.

7.2 XGBoost

We use features that we previously used in previous model with latitude and longitude raw without clustering them into labels and check the accuracy.
We get the following accuracy scores,
Cross validation : 0.57322
Public score : 0.57512

Now we use clustering for locations as described in section 6.2 and run the model again.
We get the following accuracy scores,
Cross validation : 0.57619
Public score : 0.57673

Next we included price per room as described in section 6.4 and run our model again.
We get the following accuracy scores,
Cross validation : 0.55603
Public score : 0.55987

Next we replaced common strings in the features list as described in section 5.3.
We get the following accuracy scores,

Cross validation : 0.54375
Public score : 0.55805

We use manager level as described in section 6.5.
We get the following accuracy scores,
Cross validation : 0.56734
Public score : 0.59259

Since our accuracy dropped in our last trial, we lowered our learning rate till we got a satisfiable result.
By reducing learning rate we got scores as,
Cross validation : 0.54338
Public score : 0.54480
This worked well with the manager levels.

Next we use the date and time features as described in section 6.1 and price features as described in section 6.4.
Using this our accuracy score,
Cross validation : 0.53400
Public score : 0.53931

For our final submission, we have blended our last two best results to obtain an accuracy of 0.53908 on the public leaderboard.

8 Summary

We observed that the problem is more focused on feature engineering and feature selection than on model selection. Most of the time we attempted creating and using new features on the XGBoost model. We conclude that choosing the set of features as critical as the model that we are using. We learned about the benefits of boosted trees and their applications in machine learning. However, complicated ensemble learning approaches may not be practically applicable in many scenarios as it tends to overfit the training data as in the case of random forest classifier.

For enhancing accuracy even more, we need to experiment with features like photos using image processing which may give a more realistic approximation of the consumer interest for a particular apartment.

9 References

- Random forest classifier - sklearn

  ```
  https://scikit-learn.org/stable/modules/generated
  /sklearn.ensemble.RandomForestClassifier.html
  ```

- Python XGBoost

  ```
  https://xgboost.readthedocs.io/en/latest/python
  /python_intro.html
  ```

- Seaborn for EDA

  ```
  https://seaborn.pydata.org/
  ```

- Parameter tuning blog on Analytics Vidhya

  ```
  https://www.analyticsvidhya.com/blog/2016/03
  /complete-guide-parameter-tuning-xgboost-with-codes-python/
  ```