CMPT459 DATA MINING

Milestone1

By

Duo Lu 301368672

Chen Zhao 301308092

Ziyi An 301371687

Github: https://github.com/Duo-Lu/CMPT459

February 9, 2020

1 Introduction

This is the report for the data mining project, phase one, which is majorly exploring data preprocess procedure, as well as some visualization of the raw data. The results for the first submissions are discussed in the following paragraphs, and the necessary code to produce those conclusions are within the Git repository, along with some extra experiments (involving visualization and feature extraction) which are, we think, beneficial to the project. Some technical details are also demonstrated in the README file.

2 Exploratory Data Analysis

This section is incrementally improved by the 2nd step as suggested.

1. The histograms of price before and after cleaning is as follows:

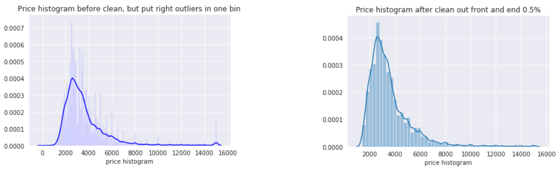


Figure 1:

For the values of latitude and longitude, intuitively, we rule out some points according to the geographical information of NYC, and then plot the histograms as follows. Those “outliers” will be discussed in the next section.

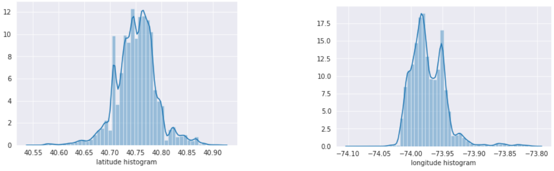
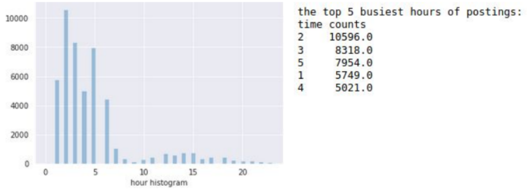


Figure 2:

2. The hour-wise listing trend is illustrated in histogram as well, and the top 5 are obtained by code (or eyeball).



```
the top 5 busiest hours of postings:
time counts
2    10596.0
3     8318.0
5     7954.0
1     5749.0
4     5021.0
```

Figure 3:

3. The proportion of target variable values (interest level) is represented in both pie chart and bar plot as follows:

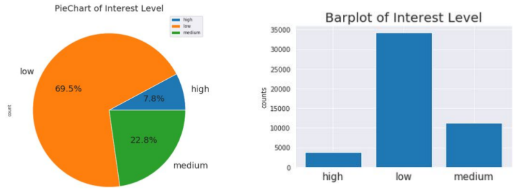
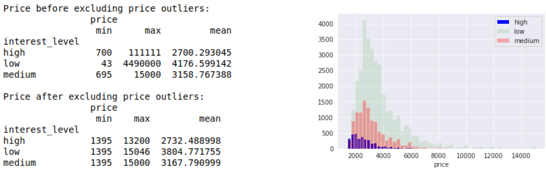


Figure 4:

*4. In addition to the required procedures, some other visualizations that might be helpful are produced, such as the statistical summaries and histograms of price in each interest level:



```
Price before excluding price outliers:
                price
                  min      max         mean
interest_level
high              700   111111  2700.293045
low                43  4490000  4176.599142
medium            695    15000  3158.767388

Price after excluding price outliers:
                price
                  min    max         mean
interest_level
high             1395  13200  2732.488998
low              1395  15046  3804.771755
medium           1395  15000  3167.790999
```

Figure 5:

We also use google map API to label each rental on a map of NYC according to the lat/longitudes. The following is the output of a 10% random sample from the original data.

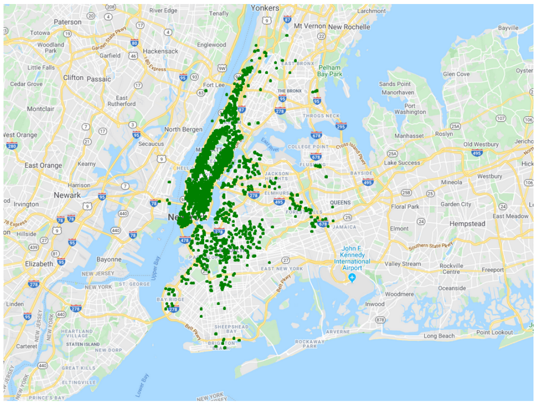
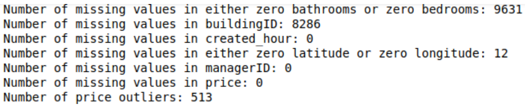


Figure 6: Note that the API key is not uploaded to the git repo

3 Dealing with missing values, outliers

1. A short summary of missing values in each variable is as such:



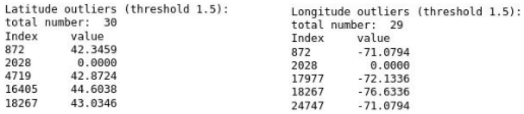
```
Number of missing values in either zero bathrooms or zero bedrooms: 9631
Number of missing values in buildingID: 8286
Number of missing values in created_hour: 0
Number of missing values in either zero latitude or zero longitude: 12
Number of missing values in managerID: 0
Number of missing values in price: 0
Number of price outliers: 513
```

Figure 7:

They are detected by values like 0, None, empty string, empty list, etc. Other missing values in text (ie, feature, address, description) would be further processed in the next section so that they could be decided to drop or not.

2.Variables that are capable of having outliers are Price, Latitude Longitude. They can either be defined manually by some bonding a percentile, geographical information (0.5% each end, NYC coordinates) or automatically by statistical measurements (Z-scores). The output of the latter approach is partially as follows:

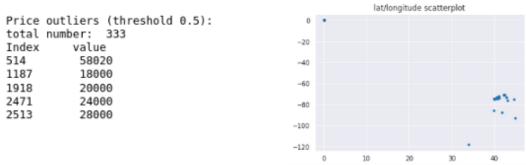
The boxplot of each variable is quite influenced by the remote outliers, and the price histogram of uncleaned data has been shown in the previous section, which indicates outliers. A scatter plot of latlongitude together might be a way of addressing the outliers as above.



```
Latitude outliers (threshold 1.5):
total number:  30
Index      value
872        42.3459
2028        0.0000
4719       42.8724
16405      44.6038
18267      43.0346
```

```
Longitude outliers (threshold 1.5):
total number:  29
Index      value
872        -71.0794
2028         0.0000
17977      -72.1336
18267      -76.6336
24747      -71.0794
```

Figure 8:



```
Price outliers (threshold 0.5):
total number:  333
Index      value
514         58020
1187        18000
1918        20000
2471        24000
2513        28000
```

Figure 9:

3.About the outliers and the missing values, if we are really rigorous to dealing rental information of NYC (only), empirically, some coordinates are considered to be noises and should be removed, while the missing values could be computed according to the address information. (We were trying this idea until realizing that there might be a payment of the Google API involved.) The outliers of prices are considered to be deleted as, empirically, some values are quite unreasonable to be a rental price (, a selling price rather). Therefore, they should be detected (according to some threshold subjective to define) and removed. Other variables either have no missing value, like managerID and created hour, or should not be defined as missing, like zeros in bath/bedroom . Anyhow, most records would not be removed except maybe the price outliers as they intuitively would influence the trained classifier, and they are difficult to be inferred by other information.

4 Feature Extraction from Images and Text

1.The image feature extraction is troublesome since the total data amount is too massive to be either downloaded as a whole or individually loaded as the program goes. Primarily, we have easily computed the number images (URLs) in each record as a feature. Moreover, we have tried to extract the logo information from each image as one of the extra code file has shown. The major problem encountered was the running time (using CSIL computers) that we hope to resolve later.

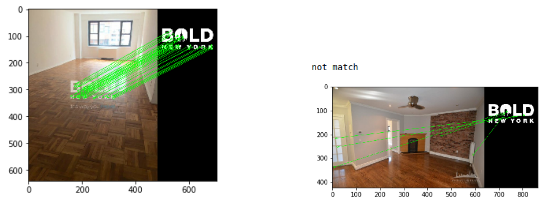


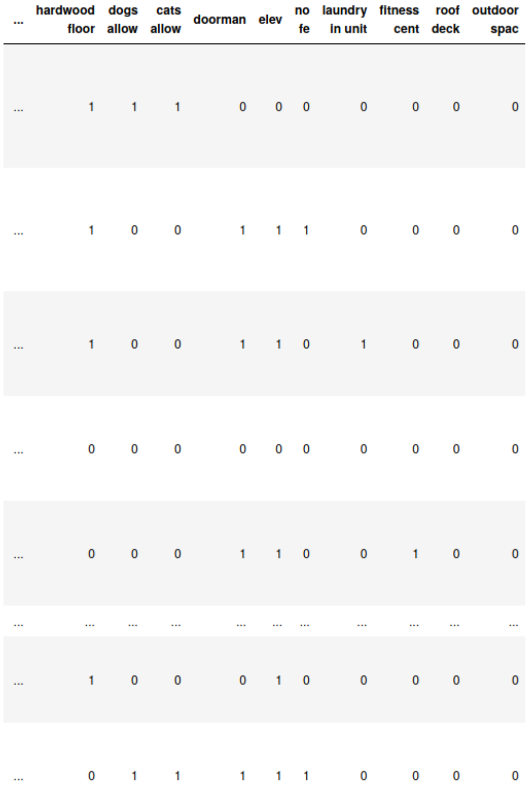
Figure 10:

2. The columns ‘features’ and ’description’ are what we mainly focused text information extraction. We build pipelines to formally conduct the work for both features and description.
For ’Features’ (Figure 11):

- cleaning the text list, dropping punctuation, spaces, and other irrelevant symbols;
- Stemming each feature, as a word (using a build-in PorterStemmer);
- Analyzing and selecting the top words with high frequency as extracted features;
- concluding each extracted feature into individual columns, as binary values, to indicate which specific feature is present to each rental record. (ie. dogs allowed, doorman, and laundry-in-unit, etc). Currently, every feature has been covered in a new column, which might be further discussed whether to keep or not.

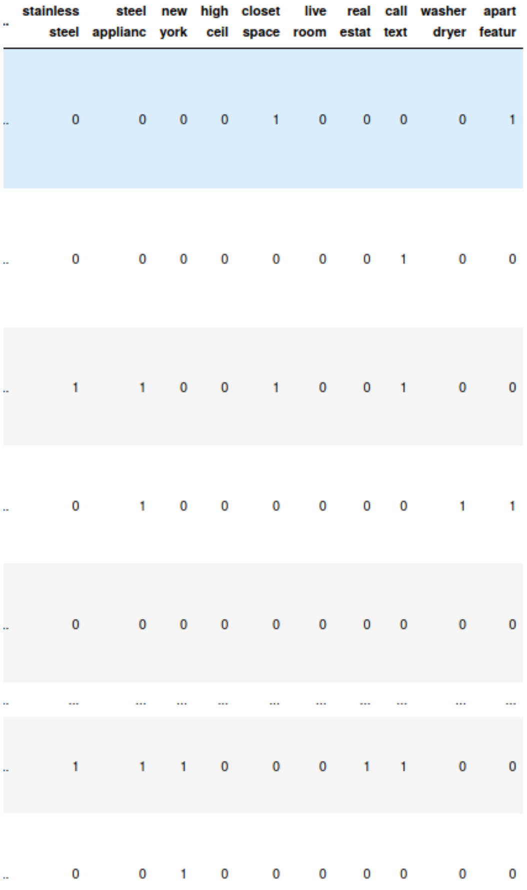
For ’Description’ (Figure 12):

- removing the stop words, cleaning the text list, dropping punctuation, spaces and other irrelevant symbols;
- Stemming each description, as a word (using a build-in PorterStemmer);
- Analyzing and selecting the top words in **B-gram** with high frequency as extracted features;
- concluding each extracted feature into individual columns, as binary values, to indicate which specific feature is present to each rental record. (ie. stainless steel’, ’steel applianc’, ’new york’ and etc). Currently, every feature has been covered in a new column, which might be further discussed whether to keep or not.



...	hardwood floor	dogs allow	cats allow	doorman	elev	no fe	laundry in unit	fitness cent	roof deck	outdoor spac
...	1	1	1	0	0	0	0	0	0	0
...	1	0	0	1	1	1	0	0	0	0
...	1	0	0	1	1	0	1	0	0	0
...	0	0	0	0	0	0	0	0	0	0
...	0	0	0	1	1	0	0	1	0	0
...	...	...	...	...	...	...	...	...	...	...
...	1	0	0	0	1	0	0	0	0	0
...	0	1	1	1	1	1	0	0	0	0

Figure 11:



..	stainless steel	steel applianc	new york	high ceil	closet space	live room	real estat	call text	washer dryer	apart featur
..	0	0	0	0	1	0	0	0	0	1
..	0	0	0	0	0	0	0	1	0	0
..	1	1	0	0	1	0	0	1	0	0
..	0	1	0	0	0	0	0	0	1	1
..	0	0	0	0	0	0	0	0	0	0
..	...	...	...	...	...	...	...	...	...	...
..	1	1	1	0	0	0	1	1	0	0
..	0	0	1	0	0	0	0	0	0	0

Figure 12: