1、 Overview

In business, data mining techniques could be very useful tools. One of the applications would be in the real estate business. When the renting contract is over, it is important to find the next renter as quick as possible while still making the largest benefit. Thus the Renthop, a website offering renting information in New York, held a competition in predicting in how a room or a house might highly attract the potential customers.

2、 Statement

2.1、 Evaluating metrics

The goal of this project is to develop an algorithm that predicts the interest level for the samples as accurate as possible. The metric we used to evaluate the result is multiple class log loss. The calculation is given by

$$logloss = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij}\log(p_{ij}),$$

[1]

This is a common metric for multiple class tasks. It should be noticed that when the predicted probability is extreme, this metric will give a severe penalization. For example, giving the probability 0.1 for a sample on class 0, which is actually belonging to 0 by the label, will give a penalty factor as –log(0.1)=2.3 while giving an extreme value 0.01 for the sample will give a penalty factor as –log(0.01)=4.6, although according to the given probability they both got a very low probability of belonging to class 0.

2.2、 Dataset

The dataset of this competition is provided by Renthop at Kaggle[2].

The given raw features in the dataset includes numerical features like room numbers and the renting price, high cardinality categorical features like the street address of the room and some unstructured features like the pictures of the room and the descriptions for the room. This will be discussed in detail in the exploratory data analysis and feature engineering part.

2.3、 Strategy and Benchmark

This is a typical data-mining project. The main methodology in this project is

1) Construct and some good features by exploratory data analysis or single model validation
2) Setup feature set for different basic models, develop the basic models, stack the basic models

as described in Abhishek Thakur's blog[3].

This model is compared to an open kernel shared by SRK at Kaggle [4].

3、 Analysis

The dataset has the following columns:

Column Name	Feature Type	Column Meaning
bathrooms	Numerical	Bathrooms of the house
bedrooms	Numerical	Bedrooms
building_id	High cardinality categorical(HCC)	Building id for the room to rent
created	Temporal	Created time for the record
description	Text	The description on the website
display_address	High cardinality categorical(HCC)	Where the record is displayed n street
features	Text/List of categorical features	Features of the house/room
latitude	Numerical/Spatial	
longitude	Numerical/Spatial	
listing_id	Id	Id for the record
manager_id	High cardinality categorical(HCC)	Manager for the record
photos	List	A list of photo url for the record
price	Numerical	
street_address	High cardinality categorical(HCC)	
interest_level	Label	The interest level of the potential customer.

The size of the dataset is:

Train-set: 49352

Test-set: 74659

And the label distribution in the train set :

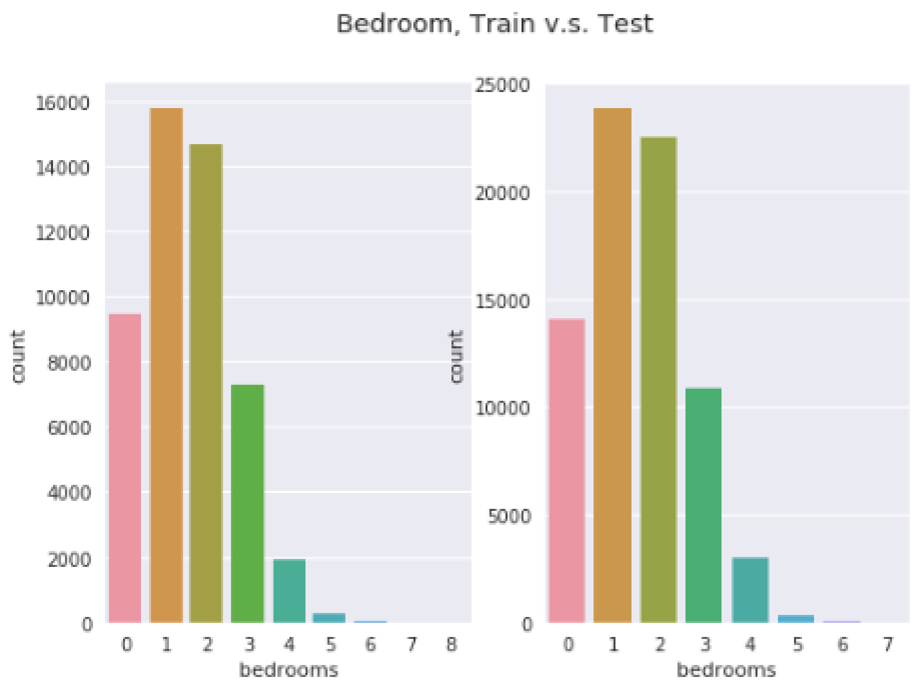
Low	Medium	High
0.694683092884	0.227528772897	0.0777881342195

Now let's take a look at the features.

3.1、 Numerical Features

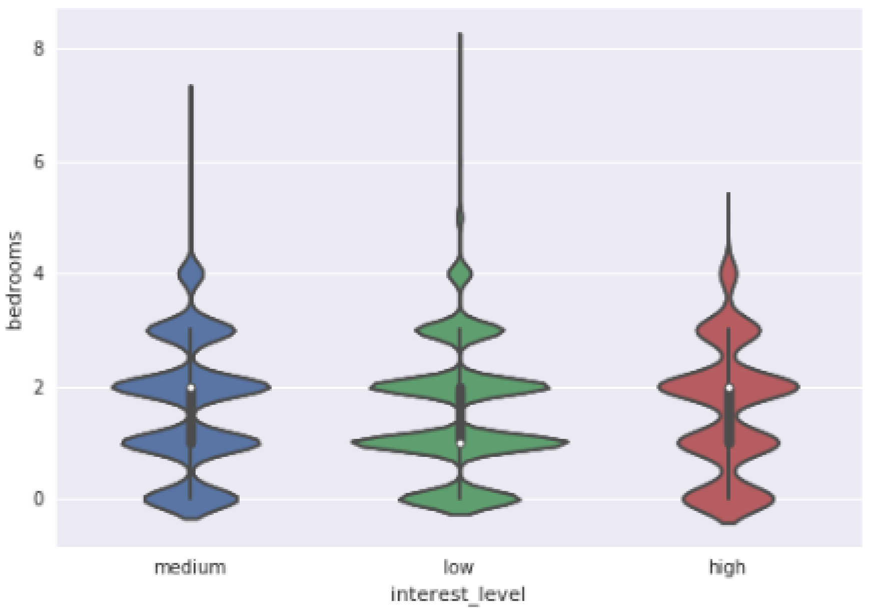
Firstly let's take a look at the numerical features.

It is clear that in the train set and test set the distribution of the bathrooms are the same.



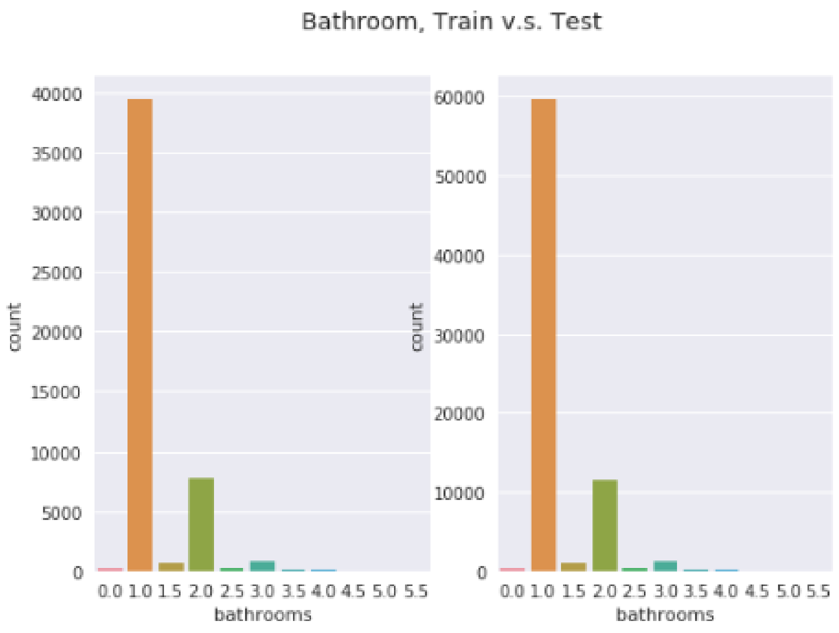
3-1 Bedroom number distribution in train and test

And take a look at the distribution difference for the labels in train set:



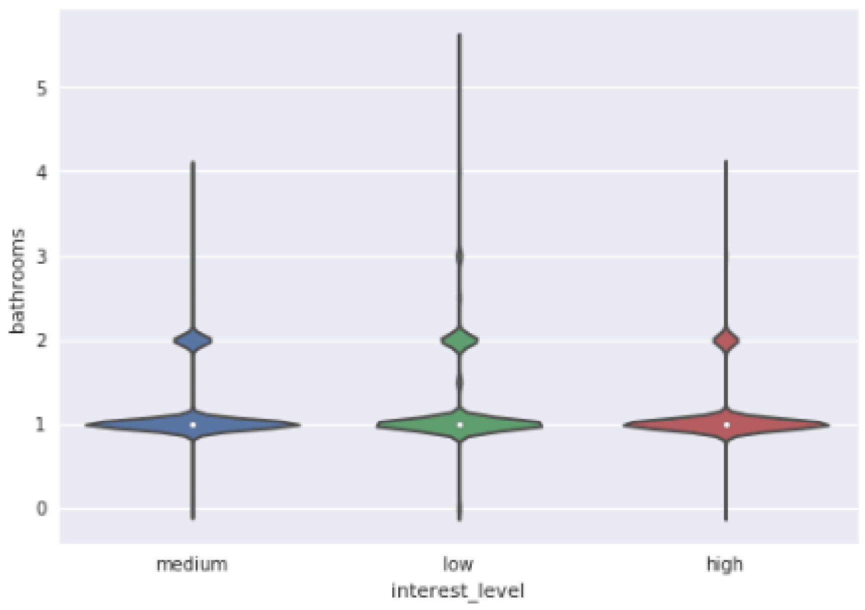
3-2 Bedroom number distribution on each label in train

And take a look at the bathrooms:



3-3 Bathroom number distribution in train and test

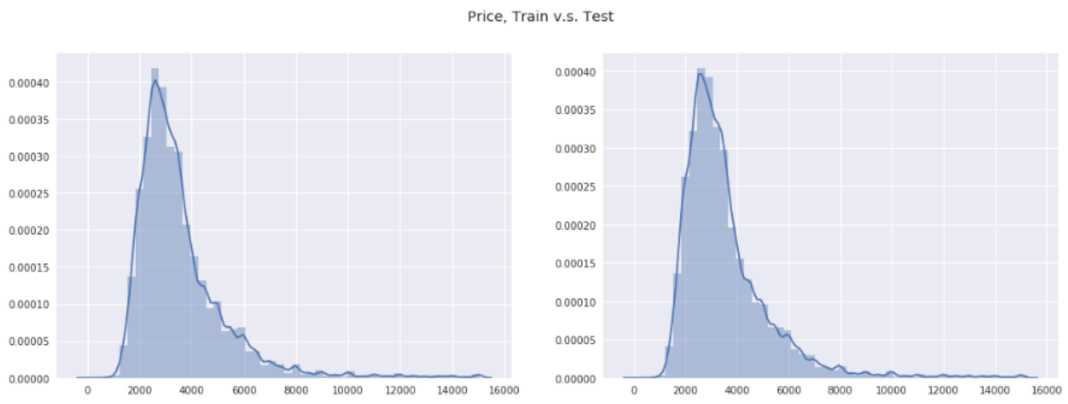
This is a filtered version where samples with bathrooms> 6 are filtered. It is clear from this plot that most suites or houses got one or two bathrooms. And below is the distribution in different labels.



3-4 Bathroom number distribution on each label in train

Price, which I think is a very import part when considering whether to rent a house, is visualized below.

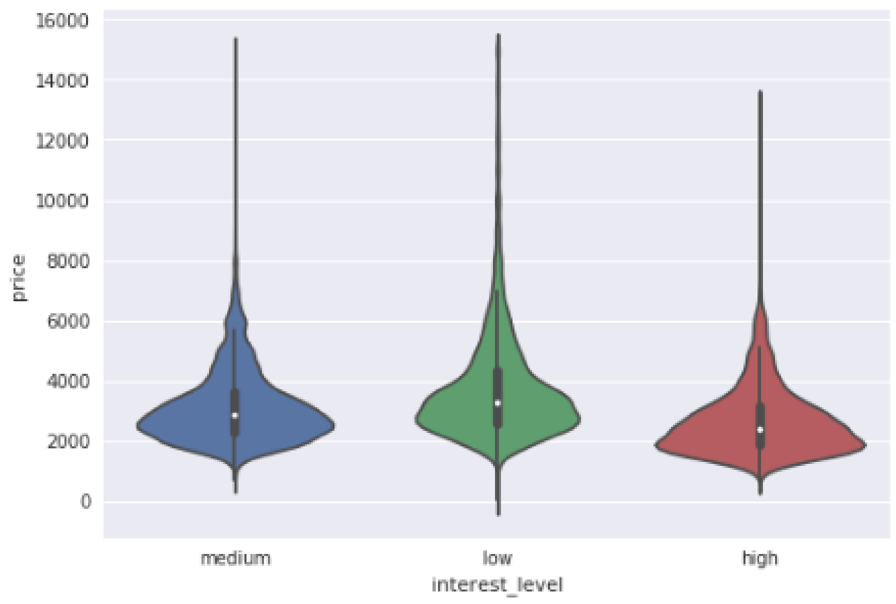
Firstly we take a look at the distribution of price in the training set and the test set.



3-5 Price distribution in train and test, cut by 99% percentile

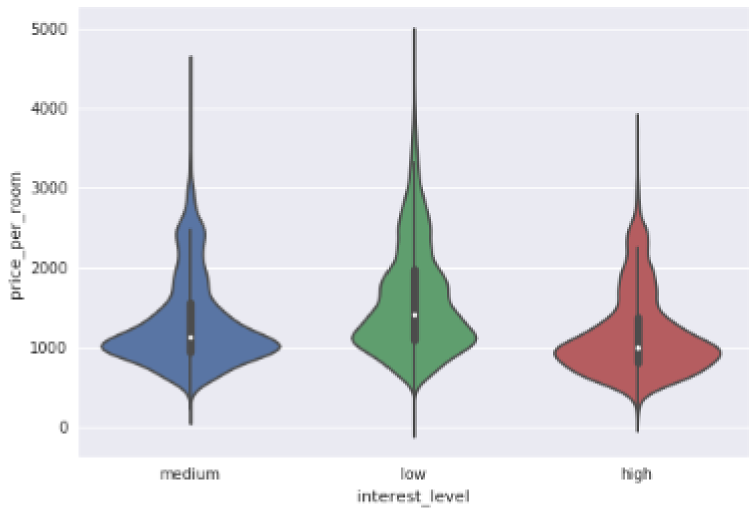
Still the train set and the test set are sharing similar distributions. From the plot it seems that the price is in a log-normal distribution. Thus in the feature engineering part we might apply box-cox transform for it if we are going to use it in the linear-based models, including logistic regression. Also we might take advantage of this when using it for the neural networks.

And the distributions on different labels are shown below. The plot seems to be corresponding to intuition that the low price ones would have a higher probability to be more popular in the potential customers.



3-4 Price distribution on each label in train

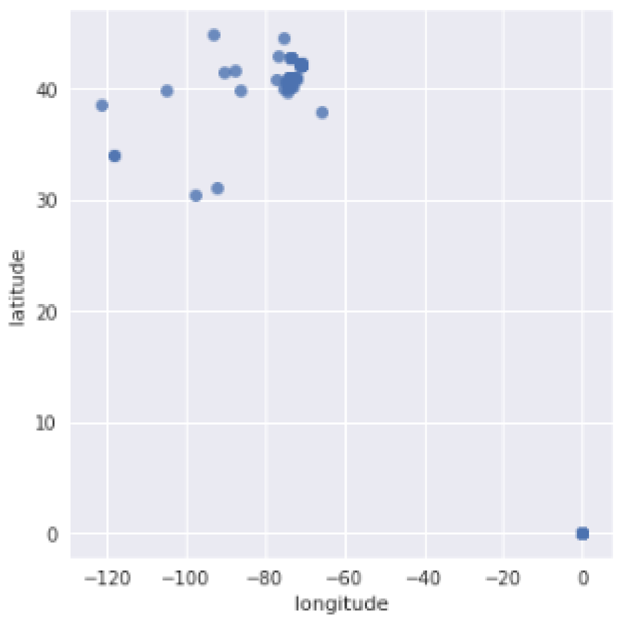
Above we just compared the total price distribution in different labels. However it might be also useful to see the average price per room. Thus this is also plot below as a reference.



3-4 Price/room distribution on each label in train

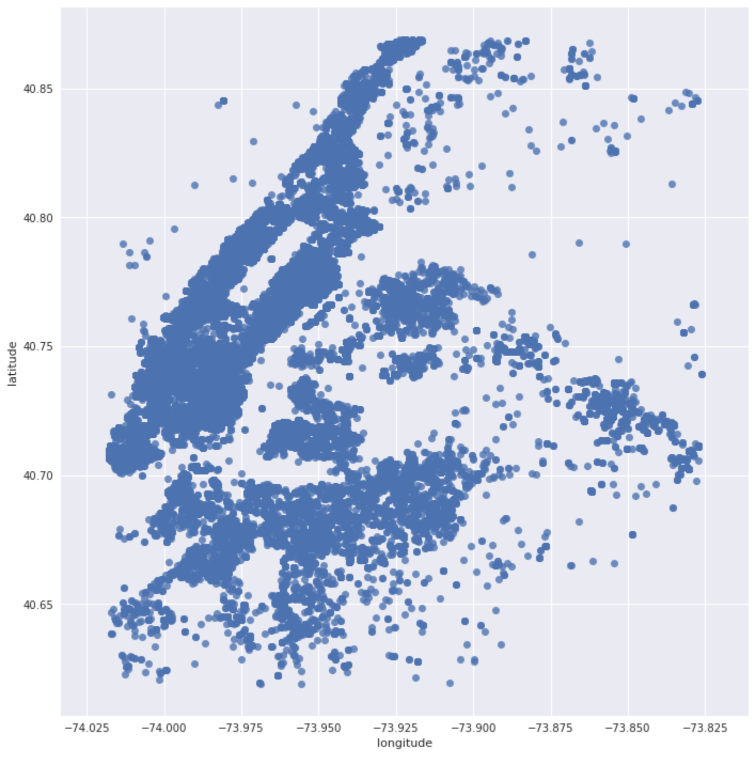
3.2、 Spatial Features

Now let's take a look at the spatial features, longitude and latitude. Firstly let's take a look at the distribution of the samples.



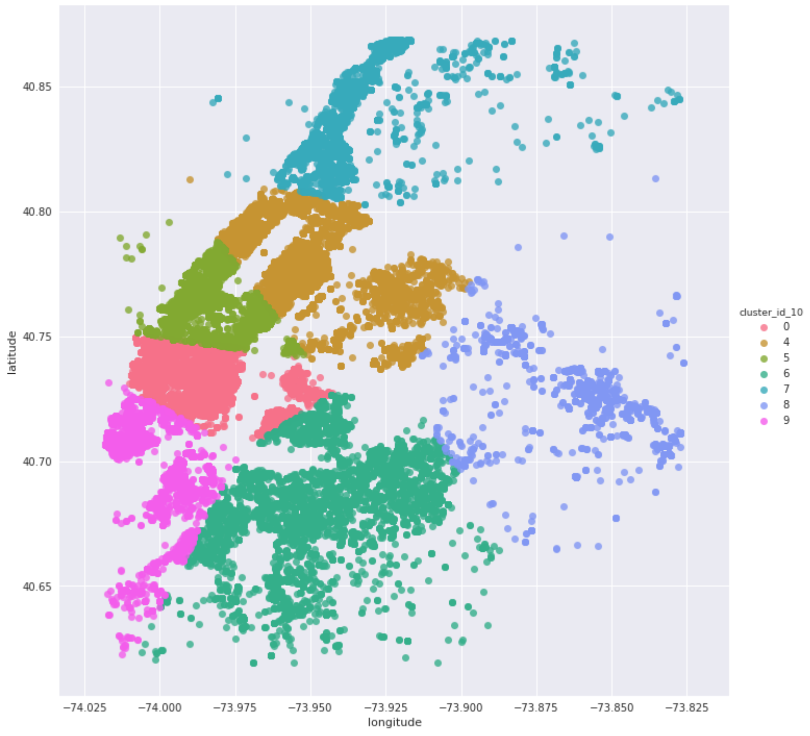
3-4 Spatial distribution of the samples on each in train and test

However from the Renthop’s website[5] it is clear that the Renthop is a website mainly dealing with renting business in New York (Coordinates: 40° 42′ 46″ N , 74° 00′ 21″ W)[6]. Thus we got some outliers in these two columns. After these outliers are cleared, we got a spatial distribution for all the samples plotted below.

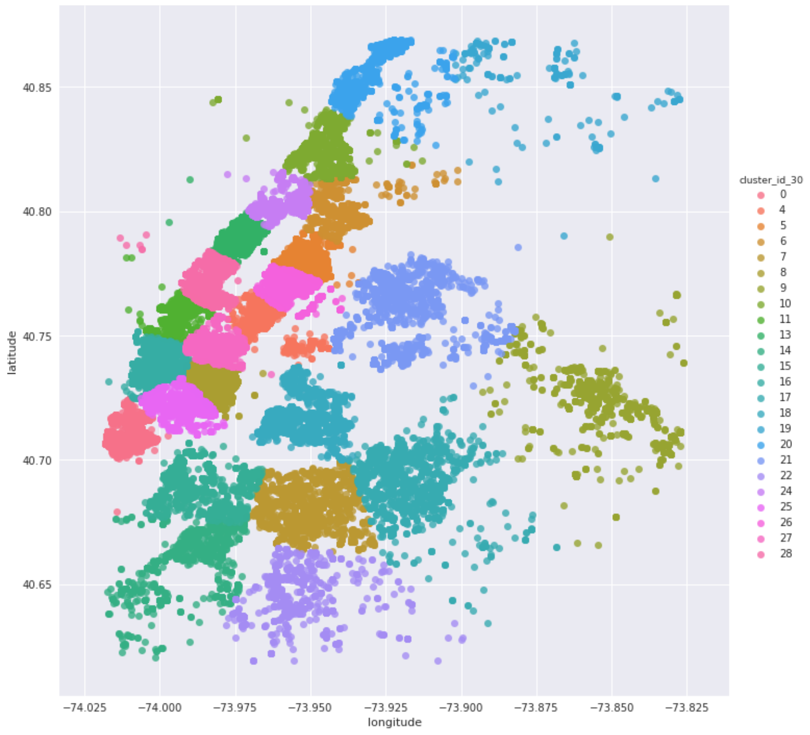


3-4 Spatial distribution of the samples on each in train and test, outliers filtered

Some clustering is done on the spatial features in order to acquire the district information and the results of clustering are plot below.



3-5 Spatial clustering the samples into 10 clusters



3-6 Spatial clustering the samples into 30 clusters

3.3、 Categorical features

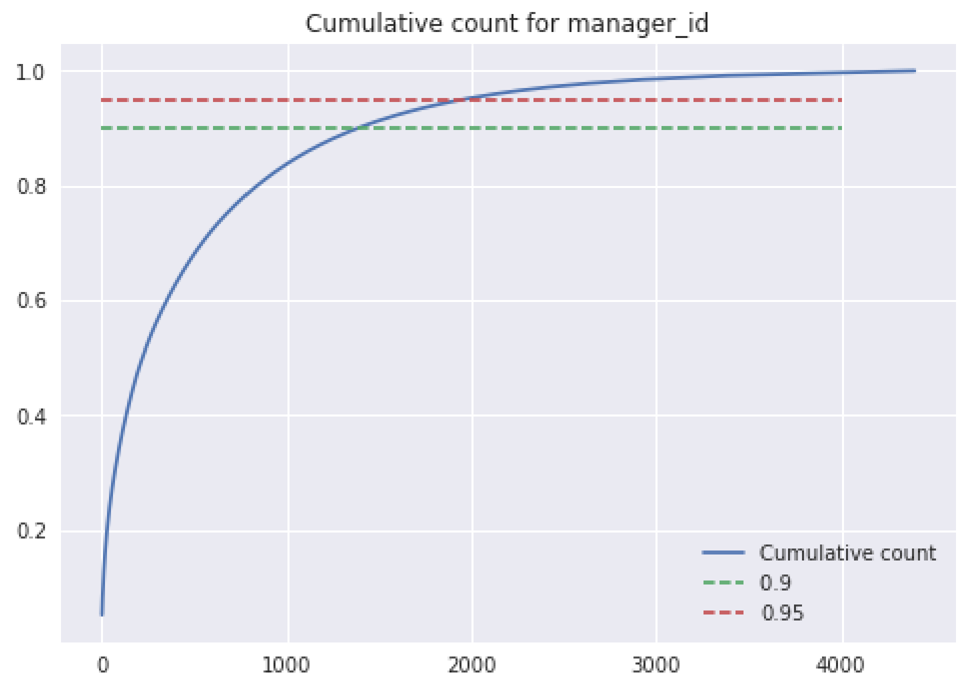
Now let's take a look at the manager features. Below is a table showing the unique value numbers in train and test.

	Train Unique	Test Unique	Total Unique
manager_id	3481	3851	4399
building_id	7585	9321	11635
street_address	15358	19561	25766
display_address	8826	11670	16068

While the size for train set is 49352 and the size for the test size is 74695.

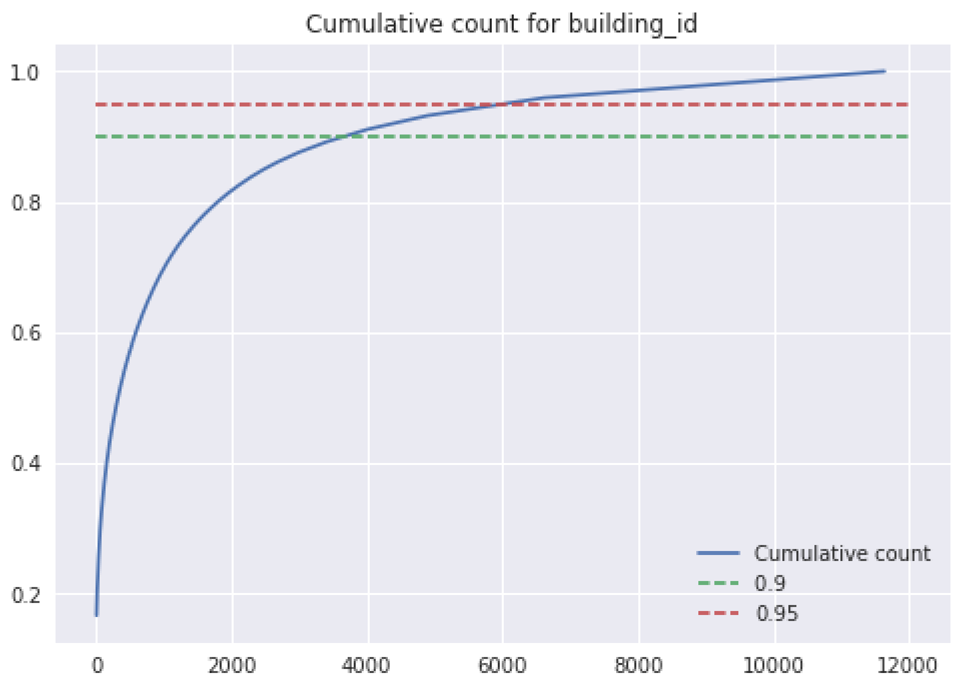
From above it is clear that these categorical features all got a very large cardinality. Although most of the values in test set had been in train set, there are still some values remained unseen. Thus it is also important to process these unseen values.

And now let's take a look at the distribution of each value. Below is a cumulative count plot for the manager_id feature.



3-7 Cumulative counting for manager_id

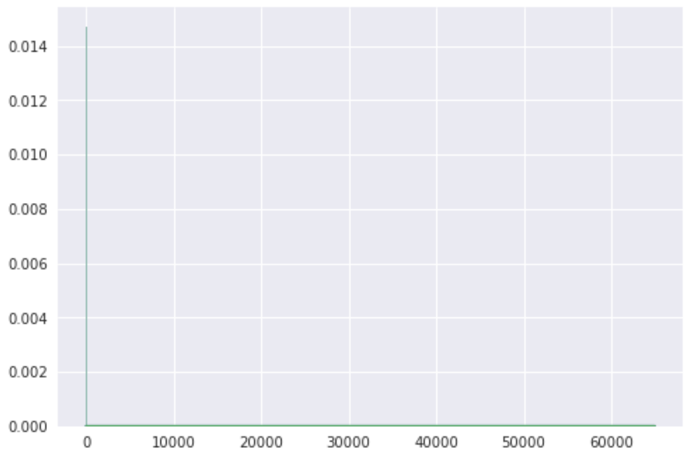
From the above plot it is clear that most of the samples are uploaded by several mangers, say 0.8 of the samples are uploaded by 700+ of the managers though there are 4399 unique managers overall. Likewise a similar analysis is also done on the building_id and other categorical features, and some of the plots are listed below.



3-8 Cumulative counting for building_id

3.4、 House features

Now let's take a look at the house features which are stored in the 'features' column in the dataset. Below is the a distribution of the appearance frequency of features



3-9 Cumulative counting for building_id

From the plot above it can be told that most of the features would only appeared several times. Below is a table recording the least appearance frequency and feature numbers, while in the dataset there are totally 2897 unique features.

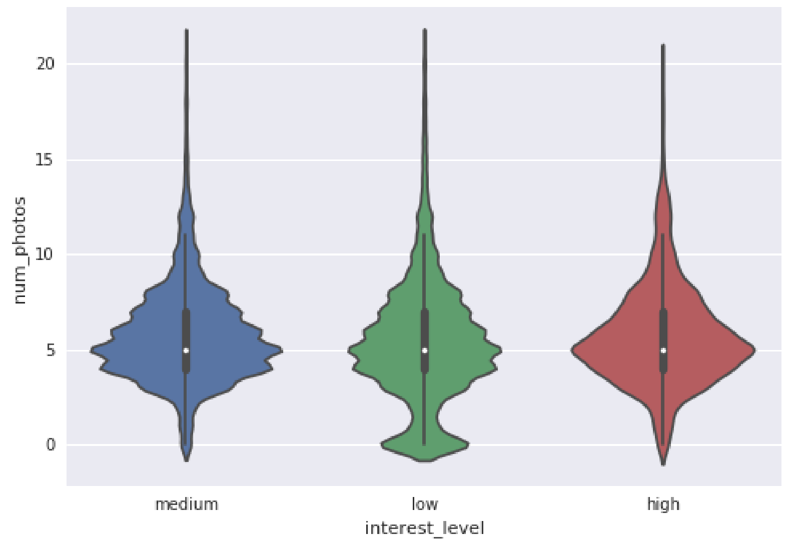
Least feature appearance number	Least feature appearance frequency	Features
12	1e-4	266
124	1e-3	98
1240	1e-2	37
12401	0.1	14

Thus features which appeared rarely would be filtered later before used.

4、 Feature Engineering

There are several sets of features generated in this competition, mainly seven sets.

The first set are some counts from the unstructured feature sets, including the number of the pictures for the house, number of the features for the house and the length of the description for the house. The distribution for the number of the pictures for the house on each label is plotted below.

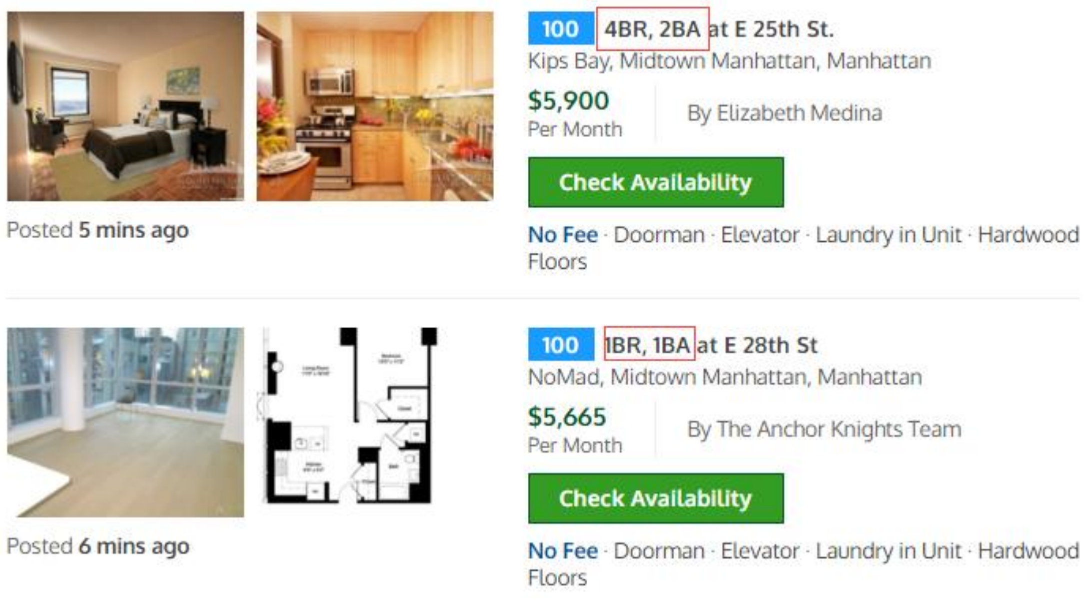


3-9 Cumulative counting for building_id

It can be clearly told from the plot that if the number of a sample is zero, then it is likely to be least attractive to the potential customers.

The second set of the constructed features are some price related features, including price/room, price/bedroom. These features give out some more general information about the price. Now the price of houses with difference room numbers can be compared through this feature.

The third set of the constructed features are the features related to the house type.


100 4BR, 2BA at E 25th St.
Kips Bay, Midtown Manhattan, Manhattan
$5,900
Per Month
By Elizabeth Medina
Check Availability
Posted 5 mins ago
No Fee · Doorman · Elevator · Laundry in Unit · Hardwood Floors

100 1BR, 1BA at E 28th St
NoMad, Midtown Manhattan, Manhattan
$5,665
Per Month
By The Anchor Knights Team
Check Availability
Posted 6 mins ago
No Fee · Doorman · Elevator · Laundry in Unit · Hardwood Floors

3-10 Samples on RentHop

By searching on Renthop I found that house type is significant factor for the samples. From intuition it is clear that when people search for their house, they would choose several acceptable house types at first. Thus this set of features is generated.

The fourth set of the constructed feature is generated from the 'features' column in the original data set. As analyzed, features rarely appeared would be filtered and the remaining ones are mapped to one-hot coding features to be used for the classifying models.

The fifth set of the constructed features are some manager performance features. This is inspired by this article [7] and this kernel [8]. From this I found that manager related indexes are very important for renting the house.

The sixth set of constructed features is the spatial clustered information for the samples. This is already discussed in 3.2.

And the final set of the constructed features are some statistical describing indexes for the above raw features and constructed features. This is inspired by this kernel [9].

Below is a table for the refinement of sequentially adding the features, using gradient boosting tree model (using Microsoft's Light GBM [10]).

	5-fold Cross Validation
Raw Features (including 'features' mapped and counts for unstructured)	0.539493314516
Adding price related constructed features	0.53726793624
Adding house type features	0.537537580672
Adding manager performance related features	0.521420261046
Adding spatial clustering features	0.521191168819
Adding some statistical Features	0.517062219379

3-11 Performance of adding the features sequentially

5、 Base Models

The base models I used in this competition are:

(1) K Nearest Neighbors

(2) Gradient Boosting Tree

(implemented by Xgboost[11] and LightGBM)

(3) Extra Trees

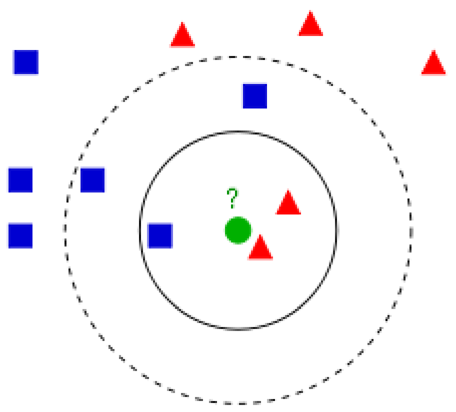
(4) Random Forest

(5) Logistic Regression

(6) Neural Network

5.1、 K Nearest Neighbor

K Nearest Neighbor is an algorithm that simply decides the samples' label or value by its nearest K neighbor's labels or values. This might be explained more clearly by the picture below.



5-1 An explaining picture for K Nearest Neighbor algorithm[12]

5.2、 Gradient Boosting Tree

In Gradient Boosting method, new functions are trained in order to fit the residual, say the difference between the old prediction and the given labels. As the residual equals the negative gradient, the method was called gradient boosting. All the functions are combined together for the final prediction. [13]

In multiple class classification problems, there are n functions for n classes to be predicted (n>2), each function computing the score for the class it represents, while the final probability is given by

$$P_i(x) = \frac{e^{F_i(x)}}{\sum_{c=1}^{n} e^{F_c(x)}} \text{ [14]}$$

for the ith class in the total n classes.

In each step, the residual was calculated by KL-divergence for the distribution difference, and the new scoring function for each class is built by fitting the residual.

5.3、 Extra Trees

Extra Trees algorithm, or Extremely Randomized Trees, fit multiple trees then take the average/voting of their result. Each tree is fitted on the whole training set by following steps:

1) If meets the ending condition then return
2) Otherwise randomly choose K features (K< M where M is the total number of features for the data set)
3) Randomly choose a splitting criteria for each selected feature
4) Choose the feature splitting criteria that best splits the dataset
5) Keep on splitting until meeting the exit condition, normally this is defined by the minimum number of samples

5.4、 Random Forest

Random Forest is also a tree based ensemble model. The difference of it and the Extra Trees model is that in Random Forest algorithm, each tree was developed by randomly chosen part of the training samples (bootstrap method), and the splitting criteria on each feature was optimized. Normally in

Random Forest algorithm CART algorithm (Classification and Regression Tree) is used for building the trees.

5.5、 Logistic Regression

Logistic Regression got a similar hypothesis with linear regression, while a sigmoid function was used to transfer the value between (-∞, ∞) to (-1, 1). The hypothesis of the Logistic Regression could be written as

$$p = S(\sum_{i=0}^{d} w_i x_i)$$

Where S() is the sigmoid function, which could be written as

$$S(t) = \frac{1}{1+e^{-t}}$$

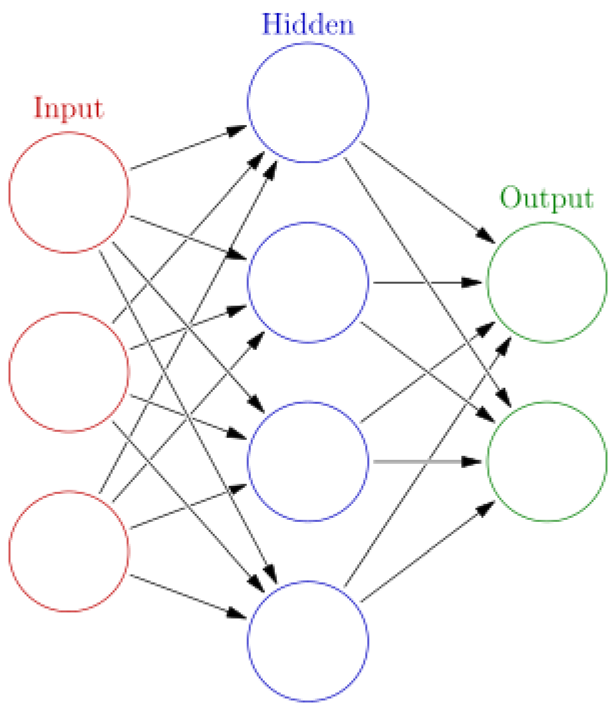
The final hypothesis can be acquired by applying gradient descend algorithm.

For multiple-class scheme one-vs-rest method is applied.

5.6、 Neural Network

Neural Network is a structure with multiple elements, called neurons, in different layers. In each neuron there is a step of linear combination of the input from the previous layer, and a step of transforming using tanh function or sigmoid function.

A typical Neural Network can be drawn like this:



5-1 An example of typical Neural Network [15]

5.7、 Base Model implementation

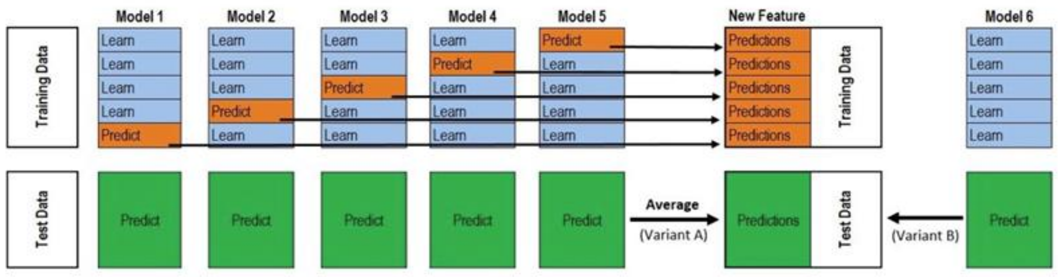
The implementation information of the base models are described below:

Model	Implemented Notebook	Main package used for model
Neural Network	BaseModel-ANN.ipynb	Keras
Extra Trees	BaseModel-extraTrees&randomForests.ipynb	Sckit-learn
Random Forest	BaseModel-extraTrees&randomForests.ipynb	Scikit-learn
Gradient Boosting	BaseModel-GradientBoosting.ipynb	Xgboost, LigthGbm
K Nearest Neighbors	BaseModel-knn.ipynb	Scikit-learn
Logistic Regression	BaseModel-LogisticRegression.ipynb	Scikit-learn

6、 Stacking

Stacking is a method of combining multiple models. By using stacking, the output of the meta-model (often the second layer model) would perform better for its smoothing nature and ability to highlight each base model where it performs best and discredit each base model where it performs poorly [16].

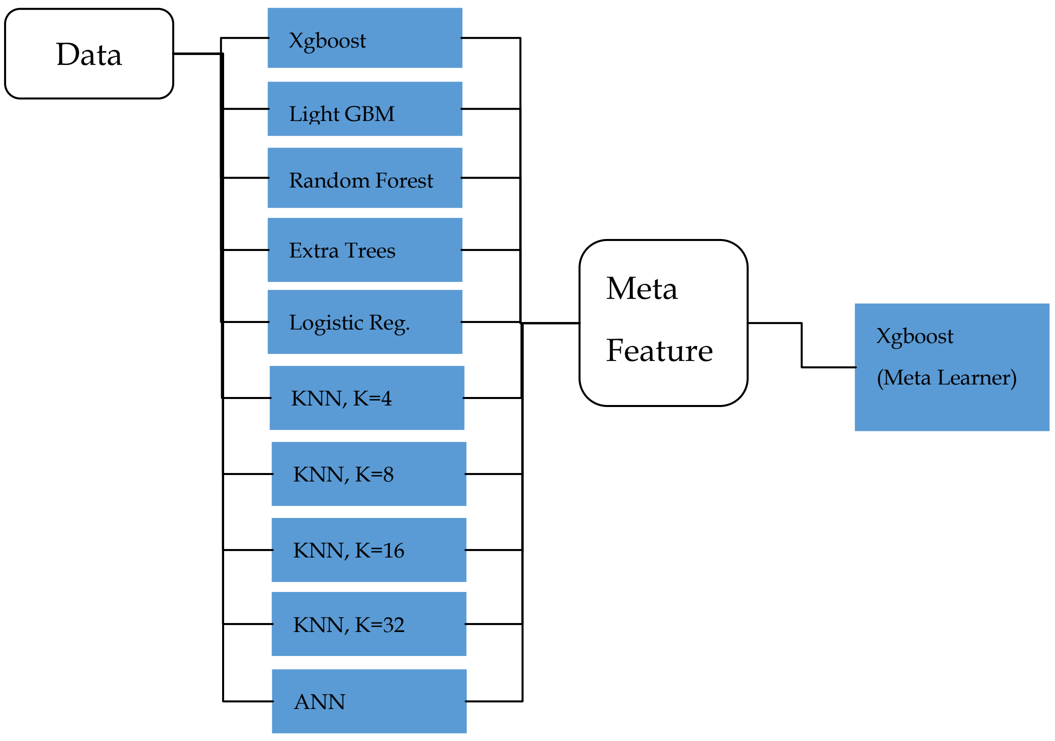
When doing stacking it is important to train the base-model and produce the meta-features by different set of data. However when implementing the stacking model, we often hope to utilize the training data as much as possible. Thus a cross-validation style method was introduced: say splitting the data into k-fold and using k-1 part to train the base model and the remaining part to generate the meta-features, doing it for k times, as shown in the picture below:



6-1 A picture explaining how data sets are used in stacking [17]

The final model of my case is a two layer structure, using KNN, Gradient Boosting Tree, Extra Trees, Random Forest, Logistic Regression and Neural Network as base models.

-



6-2 A picture of the overall structure of the model

7、 Results

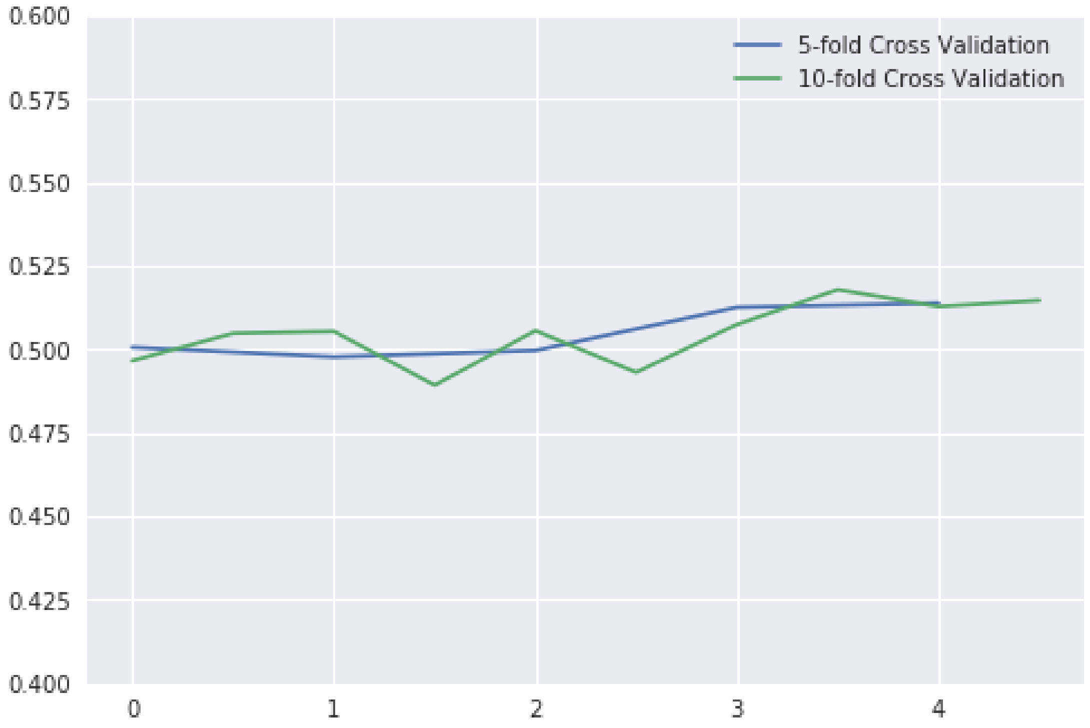
The 5-fold cross-validation result for the meta-learner is:

Turns	1	2	3	4	5	mean	std
Result	0.5006	0.4977	0.4996	0.5126	0.5138	0.5049	0.0069

The 10-fold cross-validation for the meta-learner is:

Turns	1	2	3	4	5	mean	std
Result	0.4966	0.5048	0.5054	0.4893	0.5056	0.5047	0.0089
Turns	6	7	8	9	10		
Result	0.4931	0.5074	0.5178	0.5128	0.5146		

The cross-validation result could be visualized as below:



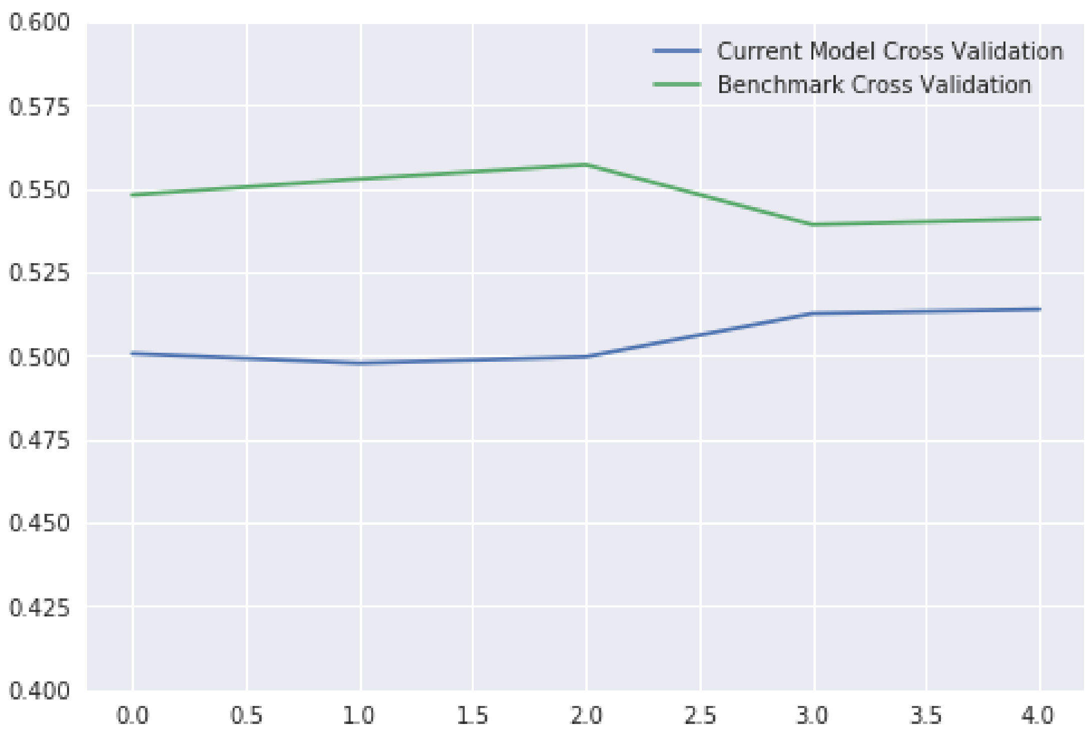
From the plot above and the low standard-variance for both 5-fold CV and 10-fold CV, it is confirmed that the performance of this model is quite robust.

Compared to the benchmark [7], the result is listed as below:

Model	5-fold Cross-validation	Public Board	Private Board
Benchmark	0.5476	0.5513	0.5521
Own model	0.5049	0.5102	0.5114

As the result by validation and out-of-fold test sets (Public Board test sets and Private Board test sets) is quite close, it is confirmed that the result can be trusted.

The comparison of 5-fold validation of the current model and the benchmark model is visualized as below:



From this plot it is clear that current model is better than the benchmark model steadily.

8、 Reflections and Improvements

8.1、 Methods and skill that had been learned

By using the method above I was able to get into the top 5% and got a silver medal. This is not a bad result. In this competition I had mainly learned about feature engineering and stacking.

In feature engineering, these methods had been applied:

- Transfer time into different length of period
- Use counts as feature
- Construct new feature by feature interactions
- Clustering sample by the spatial features
- Statistical features that describing the categorical features
- Target probability encoding for categorical features(must be applied using out-of-fold implementation)

And I had also learned about how to play a typical data science competition in general.

1） Do some exploratory data analysis to see how data should be cleaned, what feature can be extracted and the overall state of the dataset
2） Try the generated feature sets by a single model, normally gradient boosting tree method is used at this step for its fastness and enough complexity
3） Construct multiple models (and feature set if possible, though I didn't but the champion of this competition did this), and combine them together by stacking. In stacking only one or a few strong models are required. It is alright to have other weak models to be included.

8.2、 Discovery of the important factors that making a house renting case popular

Below is a top 15 feature importance generated by the Gradient Boosting model, implemented by LightGBM:

Rank	Feature	Type	Feature Gain
1	price	Original Feature	111105.81484564491
2	manager_id_perf	Constructed for manager performance	109068.75583491054
3	m30perf_f	Constructed for manager performance	79844.741289742698
4	m30perf	Constructed for manager performance	58040.872572038752
5	price_per_room	Constructed by price	56669.404150545808
6	time_stamp	'Leak' feature from[18]	55622.17248256685
7	price_per_bed	Constructed by price	41634.145451034034
8	longitude	Original Feature	23683.972601481179
9	building_id	Original Feature	22332.627464597979
10	m7perf	Constructed for manager performance	20987.8018460518

11	latitude	Original Feature	20371.339321306681
12	with_no_fee	House features	14989.634100167459
13	m14perf	Constructed for manager performance	14582.150959051782
14	building0	Constructed for identifying whether the building is correctly inserted	14323.601198303197
15	num_photos	Number of the photos	14169.227611361337

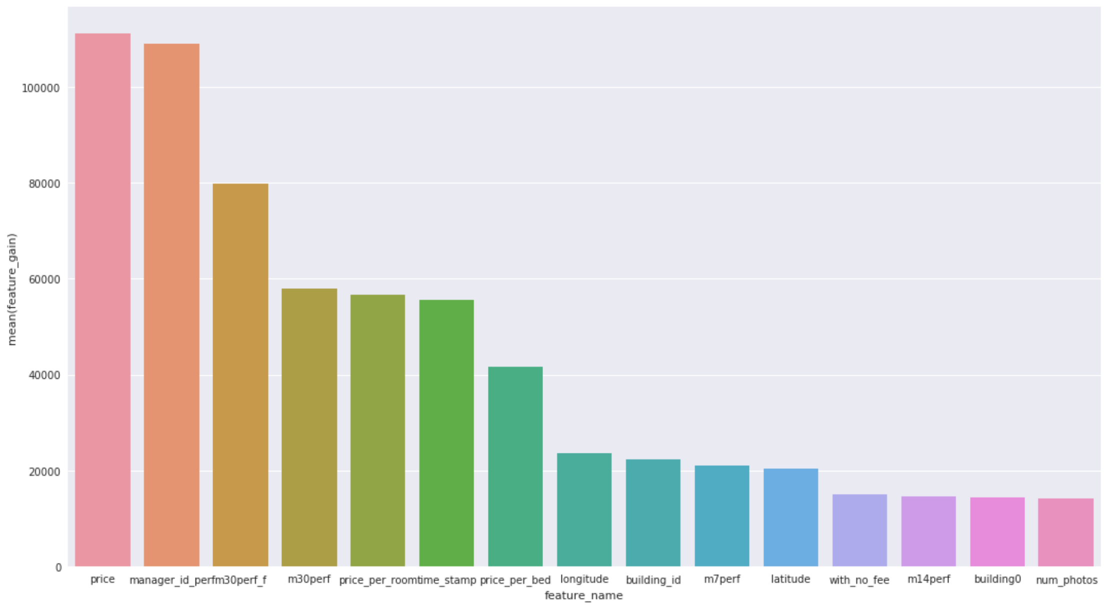
One third of the top features are related to manager performance! This told us that manager is still a key factor in the real estate market, at least in renting part. However the most significant feature, I think, is price, as 3 of the top 10 are related to price.

And the location of the building is also an important factor, as both longitude and latitude are both in the top 15. If we add up their gain to be the importance of location, then we might say the importance of location is 7th.

The building id, and the number of photos reflects that whether this case is maintained well on the web-site and they also have a large influence on the result.

Among all the features for the house itself, whether it has any fee is the most important one.

The top 15 feature importance could be visualized as below:



8.3、 Improvements

Although the competition is over, there are still some skills that might be helpful to improve the result, as listed below:

1) The density around the sample could be extracted as a feature, for where there are many samples, it tend to be a hot area for renting and thus the interest from potential customers might have a trend to be ‘high’.
2) Instead of the representing the house type by (bedrooms, bathrooms) and doing categorical statistics on that, using bedroom number as a categorical feature for generating statistics features might be a better choice, as in real case most of us think less about the bathroom numbers.
3) As there is a sequential relationship for the labels (‘high’,’medium’,’low’), the base model could also be set to predicting values by regression, as long as these labels are converted to numbers in a sequence (‘high’->3,’medium’-> 2,’low’->1). By doing this we could generate more meta-features for the meta-predictor and hence increase the stacking performance.

9、 Reference

[1]https://www.kaggle.com/c/two-sigma-connect-rental-listing-inquiries#evaluation

[2]https://www.kaggle.com/c/two-sigma-connect-rental-listing-inquiries/data

[3]http://blog.kaggle.com/2016/07/21/approaching-almost-any-machine-learning-problem-abhishek-thakur/

[4]https://www.kaggle.com/sudalairajkumar/xgb-starter-in-python/notebook

[5] https://www.renthop.com/

[6] https://en.wikipedia.org/wiki/New_York_City

[7]https://www.fastcompany.com/3001334/want-disrupt-industry-try-actually-working-it-first#full

[8]https://www.kaggle.com/den3b81/improve-perfomances-using-manager-features/notebook

[9]https://www.kaggle.com/guoday/cv-statistics-better-parameters-and-explaination

[10] https://github.com/Microsoft/LightGBM

[11] https://github.com/dmlc/xgboost

[12] https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

[13] https://en.wikipedia.org/wiki/Gradient_boosting

[14] Cheng Li, A Gentle Introduction to Gradient Boosting

[15] https://en.wikipedia.org/wiki/Artificial_neural_network

[16]http://blog.kaggle.com/2016/12/27/a-kagglers-guide-to-model-stacking-in-practice/

[17]Raddar, Tips & tricks to win data competitions

[18]https://www.kaggle.com/c/two-sigma-connect-rental-listing-inquiries/discussion/31870