INFOSYS 722 - DATA MINING AND BIG DATA

UNIVERSITY OF AUCKLAND

DEPARTMENT OF INFORMATION SYSTEMS AND OPERATIONS MANAGEMENT

Iteration 4 - Analysis of House Price in Beijing

Authors:
Student - Lei ZHENG
UPI - lzhe770
ID - 5919649

Git Source:
GitHub Username - ezprogramming
GitHub Repository - https://github.com/ezprogramming/Infosys722_iteration4
Files to be reviewed - Iteration4.ipynb, house_price_beijing_raw.xlsx

Date: October 10, 2019

Contents

1 Situation understanding

[4]House price risk has received much attention in recent years. Many industrialized economies, including the United States, Britain, and Spain have witnessed it for a long-term House price rose sharply in the mid-2000s. The significant increase in the house price greatly impacts the housing affordability in these areas. People living these cities have to afford the high loan and interest to purchase a house or pay high rent to have a home to live. Otherwise, they may have to live on the street or even can not survive if they can not have a place to live. This danger has been mentioned in Article 11 of the UN Sustainable Development Goals, which called Sustainable Cities and Communities. The goal considers the worlds population is constantly increasing, so building modern, sustainable cities to accommodate everyone is a vital goal for the world. Also, for all of us to survive and prosper, the new and intelligent urban planning that creates safe, affordable, and resilient cities with green and culturally inspiring living conditions is necessary.

Moreover, with the house price rising sharply, more capital is attracted to get into the housing market to chase the substantial and facile capital gain, [4]and the perceived lower risk encourage loose loan standards in the mortgage market. t is the main reason which has greatly contributed to the US subprime mortgage crisis and the ensuing global financial crisis. The global financial crisis affects the growth of global economics and raising the unemployment rate. This also against the 8th goal of UN Sustainable Development Goals, the decent work, and economic growth, which should promote sustained, inclusive and sustainable economic growth, full and productive employment and decent work for all. Therefore, finding the relative relationship between house need and price, predict the house price trend in the future has become significant concerns in all over the world.

1.1 Identify the objectives of the situation

In the past 20 years, China's housing prices have risen rapidly. From 2003 to 2007, [7]the price increase rate reached up to 14% per year averagely. [7]some cities, such as Beijing have an annual growth rate of 22%. If we include rental income and Capital income, and then the return of housing capital exceeds the business sector, which is causing serious concern, because there may be a price bubble. This is a major concern for China policymakers because the bubble burst will seriously impact on the Chinese economic growth and unemployment rate. Therefore, it is important to determine if the housing bubble does exist, and predict the house price trend in China, then the policymaker can use the information to make the proper policy such as release more land to build the house in specific areas, or restrict the number of houses which individual can purchase. These policies can control the risk bring by high house price, then protect the economic growth and build up the modern, sustainable cities to accommodate

everyone.

Even so, predict the trend of the house price is not an easy task. Many scholars and institutes have considered a lot of ways to try to make prediction more precise in economics and philosophy. However, with the theory of data science developing, more and more institutes have found using data mining, and big data analysis may be a more feasible way to have more accurate results of prediction.

Beijing is the capital of the People's Republic of China, and it is the political and economic center of China. As the capital of the one of the fastest-growing economy in the last 30 years, studying and analyzing its house price have precise representativeness to the globe house price. In this report, we will use the dataset of 2011-2017 house price in Beijing to study.

Most related literature tests for house price bubbles by comparing the present value of houses with housing market prices. The main debate in the literature concerns how to calculate the current value. A popular method is to discount future cash flows (rental income), but this approach is not reliable. Future rental income in China is difficult to predict because rental income is affected by economic variables such as GDP, population density, etc., that continue to change over time. Furthermore, it is difficult to choose an appropriate discount rate for housing assets.

1.2 Assess the situation

- Personnel
  The traditional economic model often analyzes interest level, policy, inflation, and elevation when analyzing when and how much the house price will go up and down by economist. Therefore, they are very good at analyzing the impact of these characteristics in the housing market. However, their prediction model is more based on experience and no verification of relevant data. Thus, a data specialist may be needed for the project, because institutes need the results of this study to be the basis for further research in the future, and the current research is long-term effective.

- Data
  The dataset should be collected from the reliable, credible and authoritative dataset websites. Such as,

  - **Kaggle:** https://www.kaggle.com/datasets
  - **UCI Machine Learning:** https://archive.ics.uci.edu/ml/datasets.php
  - **NZ Government Data** Portal: https://catalogue.data.govt.nz/dataset
  - **KD Nuggets:** https://www.kdnuggets.com/datasets/index.html

- Requirements
  In this project, we do not have to worry about the security and legal restrictions such as issues from privacy policy on the project results because the dataset will be collected from public open-source data platform, and the result of the model is anonymous. It does not show any personal information on it.

- Assumptions
  The dataset may not contain all features which relevant to the response variable, but we assume our dataset is representative and qualified ,and there is no other factors can affect the research except the quality of the dataset.

- Constraints
  There is no apparent limitation for data access and use, because the dataset is collected from Kaggle which is an open-source data platform for data mining study and research. All datasets are provided by Kaggle is free to access and use. Also, there are no financial constraints covered in the project budget due to the fact that this is a course-based student project.

- Risks
  The features contained in the dataset may not representative enough. We should not ignore the features which are not included in the features. Also, the information contained in the dataset is enormous and complex, and the number of features recorded for each instance is also large. Therefore, it may cause the dataset very difficult to analyse.

- Contingency Plan
  For the risk we are going to deal with, our explanation, inference and prediction should only depend on the truth we have. therefore, we should clarify our result from research to the people and institutions who would like to use it. Also, we should make more effort on pre-processing of data which will make the data much easier to analyse, and to have the much clear result.

1.3 Determine data mining Objectives

1.3.1 Data mining goals

- **Prediction:**This research uses the datasets to build a efficient model to predict the house price in Beijing.

- **Explanation:** This research uses the datasets to find the relationship between provided features and house prices.

- **Desired Outcome:**We desire the prediction accuracy as higher as possible and a rank for important features.

1.3.2 Success Criteria

- **Prediction:**The prediction will be considered as success if the prediction accuracy is higher than 80%.
- **Explanation:** The explanation will be considered as success if we get a rank for the important features.

1.4 Produce a project plan

Phase	Time	Resource	Risks
Situation understanding	0.5 weeks	All analysis	Situation change
Data understanding	0.5 weeks	analysis	Data problems, Technology problem
Data preparation	0.5 weeks	Jupyter Notebook, Data mining consultant, some database analyst time	Data problems, Technology problem.
Modeling	0.5 weeks	Jupyter Notebook, Data mining consultant, some database analyst time	Technology problems, inability to find adequate model
Evaluation	0.5 weeks	All analysis	Economic change, inability to implement results
Deployment	0.5 weeks	Jupyter Notebook, Data mining consultant, some database analyst time	Economic change, inability to implement results.

Table 1: Project Plan

2 Data understanding

2.1 Collect initial data

The dataset is downloaded from **Kaggle Dataset:**
https://www.kaggle.com/ruiqurm/lianjia

The data is uploaded by **Qichen Qiu**.
His personal kaggle homepage is: https://www.kaggle.com/ruiqurm

Kaggle is an authoritative platform for data mining study and research where can collect and download suitable, verified datasets. Therefore, we can say the dataset is reliable and credible from Kaggle.

The downloading of the dataset from Kaggle is smooth and there is no issue during data collection. This dataset is fetching from Lianjia.com orginally. The dataset contains lots of attributes. Therefore, we need to exclude some attributes in future stages which are less critical for the price prediction before modeling.

2.2 Describe the data

- Amount of data:

  This dataset contains house price observations in Beijing. The dataset has a total of 318,820 records and 28 attributes.

- Format of data

  The format of the dataset is mostly numeric and those numbers are used to describe the house price and other attributes might affect it such as the number.

- Value types:
  - url: the link of dataset source
  - id: Property ID in the system
  - Lng: Longitude,coordinates, using the BD09 protocol.
  - Lat: Latitude, coordinates, using the BD09 protocol.
  - Cid: Community ID
  - tradeYear: year of sold
  - tradeMonth: month of sold
  - tradeYear: date of sold
  - DOM: active days on market
  - followers: the number of people follow the transaction

  - totalPrice: the total price
  - price: the average price by square
  - square: the square of house
  - livingRoom: the number of living room
  - drawingRoom: the number of drawing room
  - kitchen: the number of kitchen
  - bathRoom: the number of bathroom
  - floor: floor
  - buildingType: including tower(1) , bungalow(2)combination of plate and tower(3), plate(4)
  - constructionTime: the time of construction
  - renovationCondition: including other(1), rough(2),Simplicity(3), hardcover(4)
  - buildingStructure: including unknow(1), mixed(2), brick and wood(3), brick and concrete(4),steel(5) and steel-concrete composite (6)
  - ladderRatio: which is the proportion between number of residents on the same floor and number of elevator of ladder. It describes how many ladders a resident have on average.
  - elevator: have (1) or not have elevator(0)
  - fiveYearsProperty: if the owner have the property for less than 5 years
  - subway: if have subway nearby
  - district: district
  - communityAverage: Community average price

- coding schemes:

  Checking the detail of value type on above.

2.3 Explore the data

In the House Price in Beijing Dataset, there are many feature attributes, some of them are important factors to affect the target attributes, but also there are a few less critical features.

2.3.1 Import dataset

- Firstly, we should import the xlsx file with PySpark. However, there is no existing library in PySpark can support to import a xlsx file to a Spark data frame directly. Therefore, in Iteration 4, I will still use Pandas to process the data, but use PySpark to make the models after. Here we import the necessary Python library for this project to make sure the data can be processed smoothly.

```
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
%matplotlib inline
import pandas as pd
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import export_graphviz, DecisionTreeClassifier
from sklearn import metrics
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn import linear_model
from sklearn.neighbors import KNeighborsRegressor
from sklearn.preprocessing import PolynomialFeatures
from sklearn.model_selection import cross_val_score
import seaborn as sns
import folium
from folium.plugins import HeatMap
from mpl_toolkits.mplot3d import Axes3D
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
import statsmodels.formula.api as sm
import math
from sklearn.preprocessing import scale
```

Figure 1: The snapshot for the library

- Reading the data from data source. In this case, the data source is a xlsx file - house_price_beijing_raw.xlsx. After import the data, the data now save to the Jupyter Notebook as a DataFrame type named hp_df.

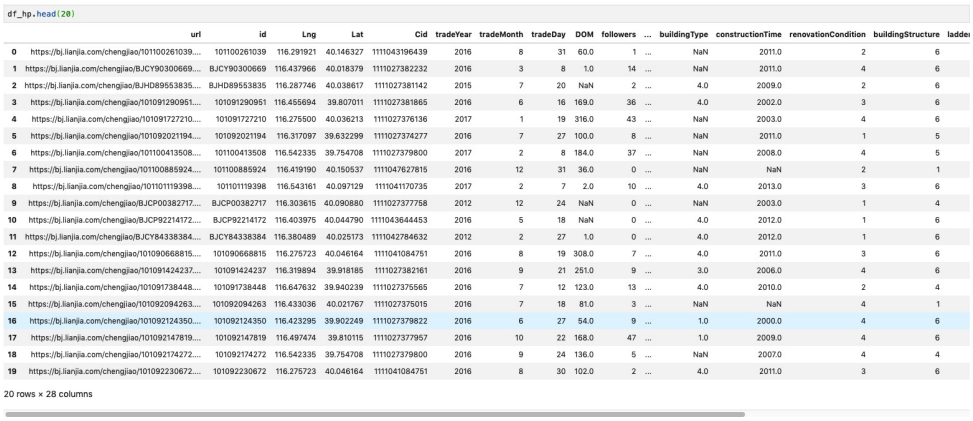
```
df_hp = pd.read_excel("house_price_beijing_raw.xlsx")
df_hp.describe().astype(np.int64)
```

	Lng	Lat	Cid	tradeYear	tradeMonth	tradeDay	DOM	followers	totalPrice	price	...	buildingType	constructionTime	renovationCondition	buildingStructure	ladderRatio	elevator	fiveYearsProperty	subway
count	318819	318819	318819	318819	318819	318819	160849	318819	318819	318819	...	316798	299536	318819	318819	318819	318819	318819	318815
mean	116	39	1129115149323	2014	6	16	28	16	349	43533	...	3	1999	2	4	63	0	0	0
std	0	0	2363565416585	1	3	8	50	34	230	21708	...	1	8	1	1	25069	0	0	0
min	116	39	1111027373683	2002	1	1	1	0	0	1	...	1	1906	1	1	0	0	0	0
25%	116	39	1111027376255	2013	3	9	1	0	205	28052	...	1	1994	1	2	0	0	0	0
50%	116	39	1111027378407	2015	7	17	6	5	294	38738	...	4	2001	3	6	0	1	1	1
75%	116	40	1111027380579	2016	10	24	37	18	425	53821	...	4	2006	4	6	0	1	1	1
max	116	40	1114619720585020	2018	12	31	1677	1143	18130	156250	...	4	2016	4	6	10009400	1	1	1

8 rows × 26 columns

Figure 2: The snapshot for the data import

- head() method give us a quick view for the dataset(as shown in Figure 2). Here we show the first 20 rows of instance in the dataset. Now, the data is ready to be explored.



```
df_hp.head(20)
```

	url	id	Lng	Lat	Cid	tradeYear	tradeMonth	tradeDay	DOM	followers	...	buildingType	constructionTime	renovationCondition	buildingStructure	ladde
0	https://bj.lianjia.com/chengjiao/101100261039....	101100261039	116.291921	40.146327	1111043196439	2016	8	31	60.0	1	...	NaN	2011.0	2	6	
1	https://bj.lianjia.com/chengjiao/BJCY90300669....	BJCY90300669	116.437966	40.018379	1111027382232	2016	3	8	1.0	14	...	NaN	2011.0	4	6	
2	https://bj.lianjia.com/chengjiao/BJHD89553835....	BJHD89553835	116.287746	40.038617	1111027381142	2015	7	20	NaN	2	...	4.0	2009.0	2	6	
3	https://bj.lianjia.com/chengjiao/101091290951....	101091290951	116.455694	39.807011	1111027381865	2016	6	16	169.0	36	...	4.0	2002.0	3	6	
4	https://bj.lianjia.com/chengjiao/101091727210....	101091727210	116.275500	40.036213	1111027376136	2017	1	19	316.0	43	...	NaN	2003.0	4	6	
5	https://bj.lianjia.com/chengjiao/101092021194....	101092021194	116.317097	39.632299	1111027374277	2016	7	27	100.0	8	...	NaN	2011.0	1	5	
6	https://bj.lianjia.com/chengjiao/101100413508....	101100413508	116.542335	39.754708	1111027379800	2017	2	8	184.0	37	...	NaN	2008.0	4	5	
7	https://bj.lianjia.com/chengjiao/101100885924....	101100885924	116.419190	40.150537	1111047627815	2016	12	31	36.0	0	...	NaN	NaN	2	1	
8	https://bj.lianjia.com/chengjiao/101101119398....	101101119398	116.543161	40.097129	1111041170735	2017	2	7	2.0	10	...	4.0	2013.0	3	6	
9	https://bj.lianjia.com/chengjiao/BJCP00382717....	BJCP00382717	116.303615	40.090880	1111027377758	2012	12	24	NaN	0	...	NaN	2003.0	1	4	
10	https://bj.lianjia.com/chengjiao/BJCP92214172....	BJCP92214172	116.403975	40.044790	1111043644453	2016	5	18	NaN	0	...	4.0	2012.0	1	6	
11	https://bj.lianjia.com/chengjiao/BJCY84338384....	BJCY84338384	116.380489	40.025173	1111042784632	2012	2	27	1.0	0	...	4.0	2012.0	1	6	
12	https://bj.lianjia.com/chengjiao/101090668815....	101090668815	116.275723	40.046164	1111041084751	2016	8	19	308.0	7	...	4.0	2011.0	3	6	
13	https://bj.lianjia.com/chengjiao/101091424237....	101091424237	116.319894	39.918185	1111027382161	2016	9	21	251.0	9	...	3.0	2006.0	4	6	
14	https://bj.lianjia.com/chengjiao/101091738448....	101091738448	116.647632	39.940239	1111027375565	2016	7	12	123.0	13	...	4.0	2010.0	2	4	
15	https://bj.lianjia.com/chengjiao/101092094263....	101092094263	116.433036	40.021767	1111027375015	2016	7	18	81.0	3	...	NaN	NaN	4	1	
16	https://bj.lianjia.com/chengjiao/101092124350....	101092124350	116.423295	39.902249	1111027379822	2016	6	27	54.0	9	...	1.0	2000.0	4	6	
17	https://bj.lianjia.com/chengjiao/101092147819....	101092147819	116.497474	39.810115	1111027377957	2016	10	22	168.0	47	...	1.0	2009.0	4	6	
18	https://bj.lianjia.com/chengjiao/101092174272....	101092174272	116.542335	39.754708	1111027379800	2016	9	24	136.0	5	...	NaN	2007.0	4	4	
19	https://bj.lianjia.com/chengjiao/101092230672....	101092230672	116.275723	40.046164	1111041084751	2016	8	30	102.0	2	...	4.0	2011.0	3	6	

20 rows × 28 columns

Figure 3: The snapshot for the data detail

2.3.2 Exploration

For exploration of the data, we run the info() method to DataFrame we get on previous step, and we get the result as shown in Figure 4. We can see that the 'id' and "url features are character values. The other features are numeric values. some of them are integer values, and the others are float values. We will deal with these different data type in the following chapters(Data Transformation). Moreover, there are some features having the miss values. The 'DOM' feature have the most of the missing values, nearly half of the records have the missing values on 'DOM'. We have to deal with this large missing values carefully in following stages(Data Perpetration) to prevent them to affect the performance of our model.

```
df_hp.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 318819 entries, 0 to 318818
Data columns (total 28 columns):
url                    318819 non-null object
id                     318819 non-null object
Lng                    318819 non-null float64
Lat                    318819 non-null float64
Cid                    318819 non-null int64
tradeYear              318819 non-null int64
tradeMonth             318819 non-null int64
tradeDay               318819 non-null int64
DOM                    160849 non-null float64
followers              318819 non-null int64
totalPrice             318819 non-null float64
price                  318819 non-null int64
square                 318819 non-null float64
livingRoom             318819 non-null int64
drawingRoom            318819 non-null int64
kitchen                318819 non-null int64
bathRoom               318819 non-null int64
floor                  318819 non-null int64
buildingType           316798 non-null float64
constructionTime       299536 non-null float64
renovationCondition    318819 non-null int64
buildingStructure      318819 non-null int64
ladderRatio            318819 non-null float64
elevator               318819 non-null int64
fiveYearsProperty      318819 non-null int64
subway                 318819 non-null int64
district               318819 non-null int64
communityAverage       318356 non-null float64
dtypes: float64(9), int64(17), object(2)
memory usage: 68.1+ MB
```

Figure 4: Data Overview

2.3.3 Visualization and Analysis

The figure below is the visualization of the data. The x-axis is all total price in the dataset, and y-axis is the number of houses in similar price area. This graph shows the distribution of the number of the total price. Through the diagram, we can see the total price of house which are less than 5 millions had been sold the most in the market, so we need to consider carefully when we process this part of the data.

```
#visualization 1
df_hp["totalPrice"].plot.hist(grid=True, bins=15, rwidth=0.9, color='darkblue')
plt.xlabel('totalPrice')
plt.ylabel('Frequency')
plt.title('Distribution of number of the total price')
```

```
Text(0.5, 1.0, 'Distribution of number of the total price')
```

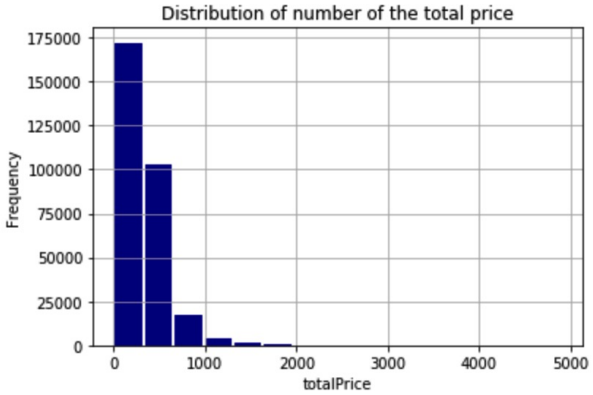


Figure 5: Distribution of the number of the total price

The Figure 6 shows the mean of house price grouped by years. As shown on the graph, the house price keep increasing since 2010, and it seems that there are some linear relationships between year and house price. Therefore, we should consider this relationship when we try to choose the model for the data.

```
#visualization 2
df_hp_year = df_hp.groupby(['tradeYear'])['totalPrice'].mean()
df_hp_year
df_hp_year.plot.bar()
plt.title('mean of house price over years')
plt.ylabel('mean of house price')
```

```
Text(0, 0.5, 'mean of house price')
```

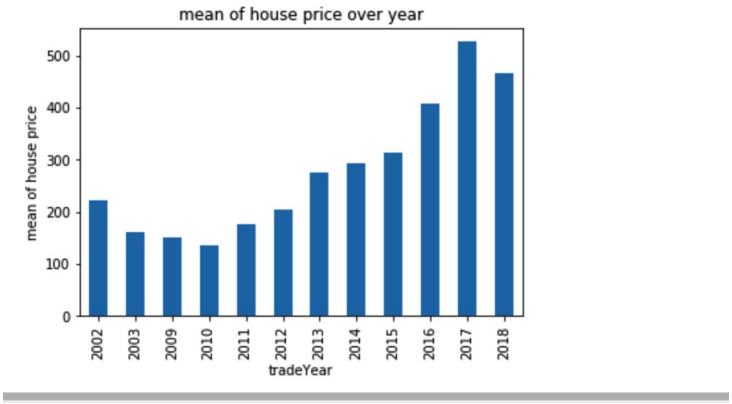


Figure 6: Mean of house price grouped by years

The figure 7 show the box plot of the house. As shown on the graph, there are some extreme high values in the total price features, we should process these extreme values carefully on the Data Preparation stage.

```
#visualization 3
df_hp.boxplot(column='totalPrice')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1a3cfa4f28>
```

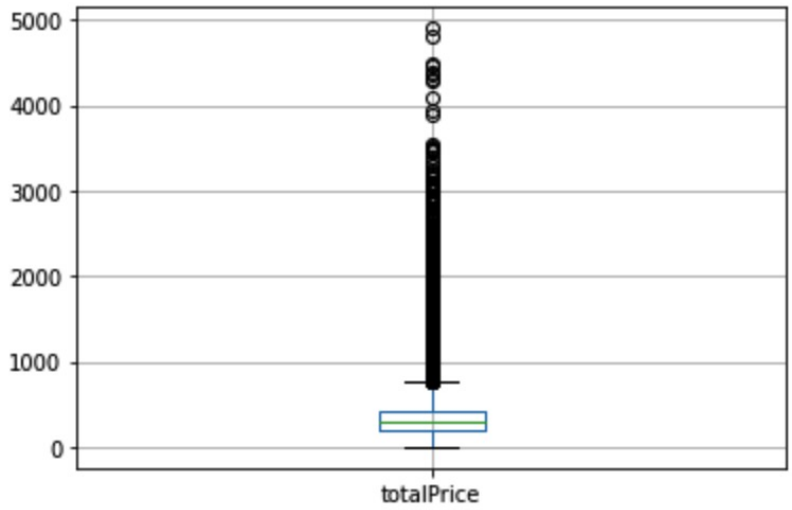


Figure 7: Box Plot of House Price

The figure below is the histograms of the data which shows the distribution of all variables on different features in the dataset. This graph gives us a rough idea what the data looks like, and how we should process these data on the following stages.

```
#visualization 4
df_hp.select_dtypes(include=['int64']).columns
df_hp.hist(figsize=(15,15))
plt.show()
```

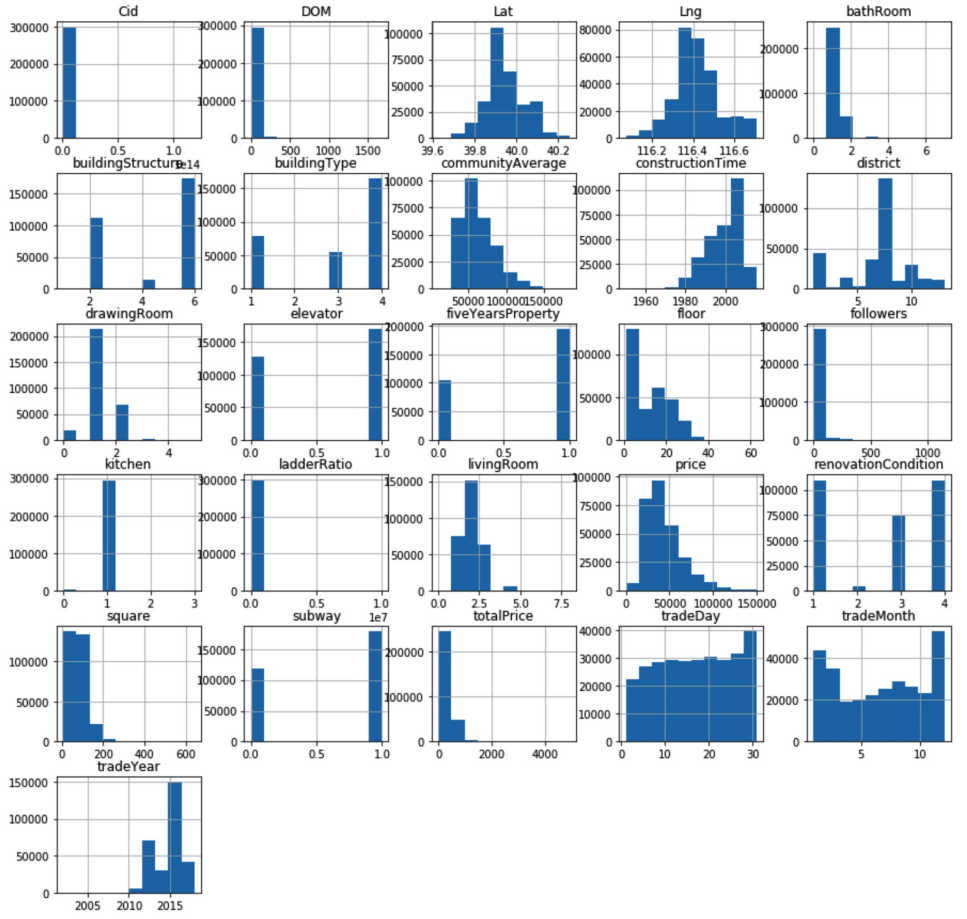


Figure 8: Data Visualization

Our purpose is to find out the relation among the features in the dataset and how they can affect the house price, then try to predict the trend of house price. The features mostly are numeric. Therefore, the regression model can be considered to apply in this project.

2.4 Verify the data quality

The data quality is considered as the most important feature in the entire data mining project, because the quality of the data can effect the model accuracy directly. The bad data quality can significantly reduce the performance of the model, even if we apply the a really good data mining algorithm which is fitted to the dataset well. Therefore, we have to carefully check the data quality before we move forward to model fitting stage.

In this step, I will look into the data quality through missing value, outliers and extreme values and measurement errors respectively.

2.4.1 Missing value

From Figure 9, as we can see from this result, the dataset used in the study has ethical integrity. Also, we can see the attribute 'DOM' have the most missing values in the dataset. There are also few missing of values in 'livingRoom', "drawingRoom", etc, but not so many.

```
In [16]: df_hp.isnull().sum()

Out[16]: url                        0
         id                         0
         Lng                        0
         Lat                        0
         Cid                        0
         tradeYear                  0
         tradeMonth                 0
         tradeDay                   0
         DOM                   157970
         followers                  0
         totalPrice                 0
         price                      0
         square                     0
         livingRoom                 0
         drawingRoom                0
         kitchen                    0
         bathRoom                   0
         floor                      0
         buildingType            2021
         constructionTime       19283
         renovationCondition        0
         buildingStructure          0
         ladderRatio                0
         elevator                   0
         fiveYearsProperty          0
         subway                     0
         district                   0
         communityAverage         463
         dtype: int64
```

Figure 9: Missing values

```
df_hp.isnull()
```

	url	id	Lng	Lat	Cid	tradeYear	tradeMonth	tradeDay	DOM	followers	...	buildingType	constructionTime	renovationCondition	buildingStructure	ladderRatio	elevator	fiveYearsProperty	subway	district	communityAverage
0	False	False	False	False	False	False	False	False	False	False	...	True	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False	...	True	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	True	False	...	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	...	True	False	False	False	False	False	False	False	False	False
5	False	False	False	False	False	False	False	False	False	False	...	True	False	False	False	False	False	False	False	False	False
6	False	False	False	False	False	False	False	False	False	False	...	True	False	False	False	False	False	False	False	False	False
7	False	False	False	False	False	False	False	False	False	False	...	True	True	False	False	False	False	False	False	False	False
8	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False
9	False	False	False	False	False	False	False	False	True	False	...	True	False	False	False	False	False	False	False	False	False
10	False	False	False	False	False	False	False	False	True	False	...	False	False	False	False	False	False	False	False	False	False
11	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False
12	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False
13	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False
14	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False
15	False	False	False	False	False	False	False	False	False	False	...	True	True	False	False	False	False	False	False	False	False
16	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False
17	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False
18	False	False	False	False	False	False	False	False	False	False	...	True	False	False	False	False	False	False	False	False	False
19	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False
20	False	False	False	False	False	False	False	False	False	False	...	True	False	False	False	False	False	False	False	False	False

Figure 10: Null data description

2.4.2 Outliers and extreme values

Outliers and extremes are not so bad in this datasets. In this subsection, I use the boxplot to find the outliers and extremes. Here I choose three most important attributs which are livingRoom, bathRoom and tradeYear to find out if they have Outliers and extreme values. Through the boxplot figures below, we can see the outliers and extreme values are not so many in these attributes, but still some of them. For example, in the livingRoom boxplot, the housing with six rooms have some extremely high price. Also, in Bathroom plot, there is an obvious outlier in four and five bathrooms. Moreover, there are some extreme values in tradeYear 2017, we can check it out through the tradeYear figure.

```
In [7]: sns.set_style('darkgrid')

        sns.boxplot(data = df_hp, x = 'tradeYear', y = 'totalPrice')

Out[7]: <matplotlib.axes._subplots.AxesSubplot at 0x1ab30485f28>
```

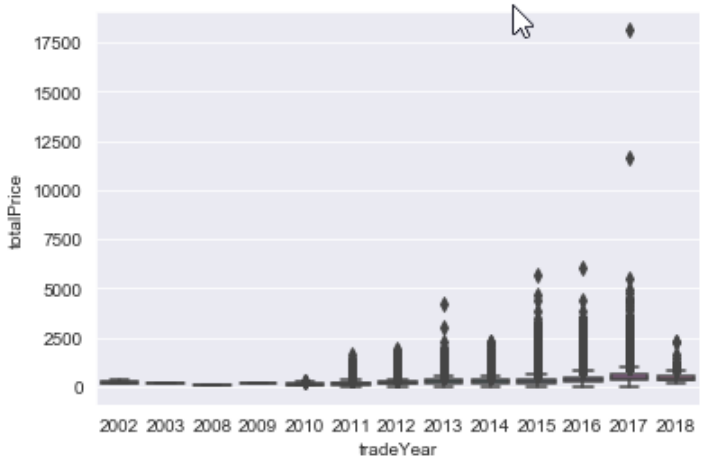


Figure 11: Boxplot of tradeYear and totalPrice

```
In [6]: sns.set_style('darkgrid')

        sns.boxplot(data = df_hp, x = 'bathRoom', y = 'totalPrice')

Out[6]: <matplotlib.axes._subplots.AxesSubplot at 0x1ab319659e8>
```

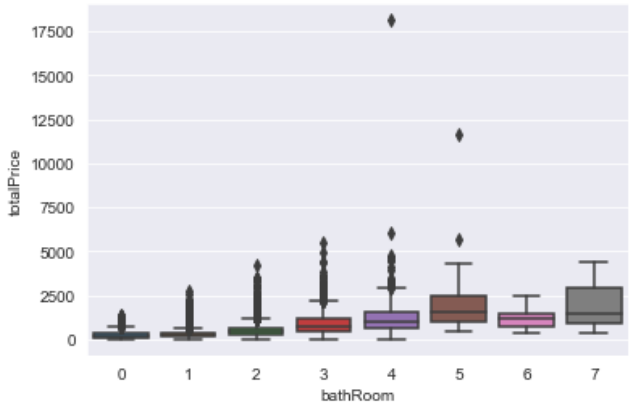


Figure 12: Boxplot of bathRoom and totalPrice

```
In [5]: sns.set_style('darkgrid')

        sns.boxplot(data = df_hp, x = 'livingRoom', y = 'totalPrice')

Out[5]: <matplotlib.axes._subplots.AxesSubplot at 0x1ab2df4cc88>
```

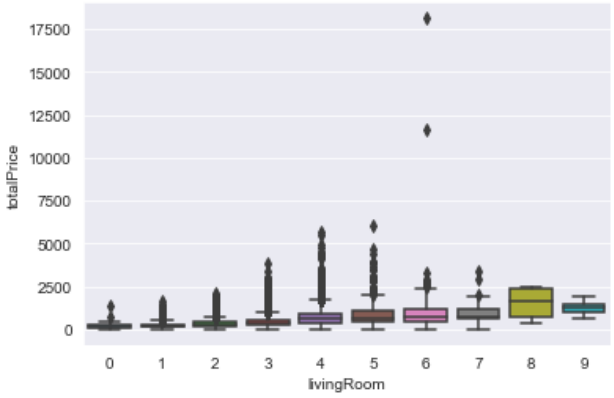


Figure 13: Boxplot of livingRoom and totalPrice

2.4.3 Measurement errors

From Figure 14, we can see that some attributes have extreme large max value which are much bigger than its mean value. This means these attributes may have outliers and extreme values. These outliers and extreme values may caused by Data errors or Measurement errors. We should take care of these instances when we building the model. Also, there are meaningless numbers and symbols in the some features, like 'bathRoom', 'buildingType', etc. We should handle this errors in Data Preparation chapter.

```
df_hp = pd.read_excel("house_price_beijing_raw.xlsx")
df_hp.describe().astype(np.int64)
```

	Lng	Lat	Cid	tradeYear	tradeMonth	tradeDay	DOM	followers	totalPrice	price	...	buildingType	constructionTime	renovationCondition	buildingStructure	ladderRatio	elevator	fiveYearsProperty	subway
count	318819	318819	318819	318819	318819	318819	160849	318819	318819	318819	...	316798	299536	318819	318819	318819	318819	318819	318815
mean	116	39	1129115149323	2014	6	16	28	16	349	43533	...	3	1999	2	4	63	0	0	0
std	0	0	2363565416585	1	3	8	50	34	230	21708	...	1	8	1	1	25069	0	0	0
min	116	39	1111027373683	2002	1	1	1	0	0	1	...	1	1906	1	1	0	0	0	0
25%	116	39	1111027376255	2013	3	9	1	0	205	28052	...	1	1994	1	2	0	0	0	0
50%	116	39	1111027378407	2015	7	17	6	5	294	38738	...	4	2001	3	6	0	1	1	1
75%	116	40	1111027380579	2016	10	24	37	18	425	53821	...	4	2006	4	6	0	1	1	1
max	116	40	1114619720585020	2018	12	31	1677	1143	18130	156250	...	4	2016	4	6	10009400	1	1	1

8 rows × 26 columns

Figure 14: Data description

3 Data preparation

In this chapter, I will deal with the problems which are found on last Data Understanding chapter, then give the solutions to process the dataset. These process improve the data quality overall, and the project can be benefited through these process, because the data quality normally may affect the model performance significantly.

3.1 Select the data

The original dataset contains 318,820 rows(instances) and 28 columns(features). However, as the analysis carrying on, some categories could be removed by the following conditions: explained by other existing variables or not used in the analysis. For example, it is easy to know that the features 'id' and 'url' are useless since it is unique identifier, through the Figure 15 on below . Also, The 'price' is the should be remove, because it is the average price by square, and can be calculated by 'total price' and 'square'. Therefore, we will remove these useless attributes during the Data Transformation stage.

Moreover, we should apply feature selection to select the useful features because we

```
df_hp.head(20)
```

	url	id	Lng	Lat	Cid	tradeYear	tradeMonth	tradeDay	DOM	followers	...	buildingType	constructionTime	renovationCondition	buildingStructure	ladde
0	https://bj.lianjia.com/chengjiao/101100261039....	101100261039	116.291921	40.146327	1111043196439	2016	8	31	60.0	1	...	NaN	2011.0	2	6	
1	https://bj.lianjia.com/chengjiao/BJCY90300669....	BJCY90300669	116.437966	40.018379	1111027382232	2016	3	8	1.0	14	...	NaN	2011.0	4	6	
2	https://bj.lianjia.com/chengjiao/BJHD89553835....	BJHD89553835	116.287746	40.038617	1111027381142	2015	7	20	NaN	2	...	4.0	2009.0	2	6	
3	https://bj.lianjia.com/chengjiao/101091290951....	101091290951	116.455694	39.807011	1111027381865	2016	6	16	169.0	36	...	4.0	2002.0	3	6	
4	https://bj.lianjia.com/chengjiao/101091727210....	101091727210	116.275500	40.036213	1111027376136	2017	1	19	316.0	43	...	NaN	2003.0	4	6	
5	https://bj.lianjia.com/chengjiao/101092021194....	101092021194	116.317097	39.632299	1111027374277	2016	7	27	100.0	8	...	NaN	2011.0	1	5	
6	https://bj.lianjia.com/chengjiao/101100413508....	101100413508	116.542335	39.754708	1111027379800	2017	2	8	184.0	37	...	NaN	2008.0	4	5	
7	https://bj.lianjia.com/chengjiao/101100885924....	101100885924	116.419190	40.150537	1111047627815	2016	12	31	36.0	0	...	NaN	NaN	2	1	
8	https://bj.lianjia.com/chengjiao/101101119398....	101101119398	116.543161	40.097129	1111041170735	2017	2	7	2.0	10	...	4.0	2013.0	3	6	
9	https://bj.lianjia.com/chengjiao/BJCP00382717....	BJCP00382717	116.303615	40.090880	1111027377758	2012	12	24	NaN	0	...	NaN	2003.0	1	4	
10	https://bj.lianjia.com/chengjiao/BJCP92214172....	BJCP92214172	116.403975	40.044790	1111043644453	2016	5	18	NaN	0	...	4.0	2012.0	1	6	
11	https://bj.lianjia.com/chengjiao/BJCY84338384....	BJCY84338384	116.380489	40.025173	1111042784632	2012	2	27	1.0	0	...	4.0	2012.0	1	6	
12	https://bj.lianjia.com/chengjiao/101090668815....	101090668815	116.275723	40.046164	1111041084751	2016	8	19	308.0	7	...	4.0	2011.0	3	6	
13	https://bj.lianjia.com/chengjiao/101091424237....	101091424237	116.319894	39.918185	1111027382161	2016	9	21	251.0	9	...	3.0	2006.0	4	6	
14	https://bj.lianjia.com/chengjiao/101091738448....	101091738448	116.647632	39.940239	1111027375565	2016	7	12	123.0	13	...	4.0	2010.0	2	4	
15	https://bj.lianjia.com/chengjiao/101092094263....	101092094263	116.433036	40.021767	1111027375015	2016	7	18	81.0	3	...	NaN	NaN	4	1	
16	https://bj.lianjia.com/chengjiao/101092124350....	101092124350	116.423295	39.902249	1111027379822	2016	6	27	54.0	9	...	1.0	2000.0	4	6	
17	https://bj.lianjia.com/chengjiao/101092147819....	101092147819	116.497474	39.810115	1111027377957	2016	10	22	168.0	47	...	1.0	2009.0	4	6	
18	https://bj.lianjia.com/chengjiao/101092174272....	101092174272	116.542335	39.754708	1111027379800	2016	9	24	136.0	5	...	NaN	2007.0	4	4	
19	https://bj.lianjia.com/chengjiao/101092230672....	101092230672	116.275723	40.046164	1111041084751	2016	8	30	102.0	2	...	4.0	2011.0	3	6	

20 rows × 28 columns

Figure 15: The snapshot for the data detail

have 28 features and not all of them are useful for predicting the response variable. Therefore, we should ignore these unimportant features during our mining process. For example, the 'kitchen' attribute is a categorical value. Through Figure 13, we can see the most value of kitchen is 1 which make kitchen not an important factor to the model due to redundancy of value in the attribute.

```
df_hp["kitchen"].describe()
```

```
count    318819.000000
mean          0.994338
std           0.106185
min           0.000000
25%           1.000000
50%           1.000000
75%           1.000000
max           4.000000
Name: kitchen, dtype: float64
```

Figure 16: The detail of kitchen attribute

For other attributes, there are some useful feature selection functions in Sklearn package to find which attributes are important. As shown in Figure in below, we used SelectKBest and f_classif libraray in Sklearn to summarise the feature importance for other attributes in this dataset, and we can find the most important attributes are tradeYear, square, communityAverage and livingRoom. However, we decide to save these attributes, and fit them into the model, because they may still affect the model performance on somehow, and the current computing power which is provided in this project is good enough to process all of these attributes and fit into model together.

In [10]:

```
Y = df_hp_sub['totalPrice']
X = df_hp_sub
from sklearn.feature_selection import f_classif
from sklearn.feature_selection import SelectKBest
bestfeatures = SelectKBest(score_func=f_classif , k=10)


fit = bestfeatures.fit(X,Y)
dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(X.columns)
#concat two dataframes for better visualization
featureScores = pd.concat([dfcolumns,dfscores],axis=1)
featureScores.columns = ['Specs','Score']  #naming the dataframe columns
print(featureScores.nlargest(20,'Score'))
```

```
                Specs      Score
8          totalPrice        inf
3           tradeYear  22.625726
9              square  17.874587
23    communityAverage  15.629885
10         livingRoom  12.155064
16  renovationCondition   8.516921
12           bathRoom   8.154877
11        drawingRoom   6.676742
7           followers   3.831078
19           elevator   3.592496
17   buildingStructure   3.184148
6                  DOM   3.175255
13               floor   3.049320
21              subway   2.272344
0                  Lng   1.941695
20   fiveYearsProperty   1.848927
2                  Cid   1.751696
1                  Lat   1.644736
15    constructionTime   1.567300
14        buildingType   1.552413
```

Figure 17: Feature Importance

3.2 Clean the data

In this section, we will take care the data issue which have been found in step 2.4 (Verify the data quality)

3.2.1 The missing values

As mentioned in step 2.4.1 (Missing value), we notice that there are over 50% of the total dataset have missing value on the attribute of 'DOM', and we can not just simply remove the missing value in it, because the result of the data mining may be affected by missing a large part of the records in the dataset. Therefore, we need to find a way to make the complement for DOM. Here we take the mean of the DOM to fill up all missing values in the attribute of DOM.

```
In [17]: df_hp["DOM"].fillna(df_hp["DOM"].mean(), inplace = True)
         df_hp.isnull().sum()

Out[17]: url                      0
         id                       0
         Lng                      0
         Lat                      0
         Cid                      0
         tradeYear                0
         tradeMonth               0
         tradeDay                 0
         DOM                      0
         followers                0
         totalPrice               0
         price                    0
         square                   0
         livingRoom               0
         drawingRoom              0
         kitchen                  0
         bathRoom                 0
         floor                    0
         buildingType          2021
         constructionTime     19283
         renovationCondition      0
         buildingStructure        0
         ladderRatio              0
         elevator                 0
         fiveYearsProperty        0
         subway                   0
         district                 0
         communityAverage       463
         dtype: int64
```

Figure 18: Fill the missing values by the mean in DOM 1

As shown in the figure 4, the 'drawingRoom', 'livingRoom', 'communityAverage', constructionTime', 'buildingType', 'bathRoom', 'elevator', 'subway', 'fiveYearsProperty' features are all have missing values, but most of them have less than 2% missing values. Therefore, we can use the dropna() method to remove these missing values. The result shown as below.

```
In [18]: df_hp.dropna(inplace=True)
         df_hp.isnull().sum()

Out[18]: url                   0
         id                    0
         Lng                   0
         Lat                   0
         Cid                   0
         tradeYear             0
         tradeMonth            0
         tradeDay              0
         DOM                   0
         followers             0
         totalPrice            0
         price                 0
         square                0
         livingRoom            0
         drawingRoom           0
         kitchen               0
         bathRoom              0
         floor                 0
         buildingType          0
         constructionTime      0
         renovationCondition   0
         buildingStructure     0
         ladderRatio           0
         elevator              0
         fiveYearsProperty     0
         subway                0
         district              0
         communityAverage      0
         dtype: int64
```

Figure 19: Drop the rest of the missing values

3.2.2 Outliers and extreme values

After we removed the missing values, there are still some outliers and extreme values in some of the features, such as 'price', 'DOM', etc, which may affect our model performance as well. Therefore, we will take care these outliers and extreme values in this step.
Firstly, let us draw some plots which can give us more detail about these features which contain outliers and extreme values. Here we choose some continues variables which

can find whether they have outliers and extreme values through the box plot. The plots are shown in Figure 20-25.

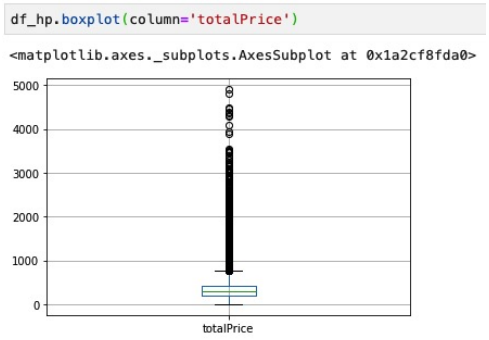


Figure 20: Box plot of total price

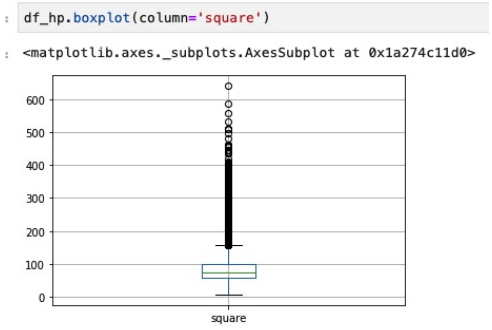


Figure 21: Box plot of square

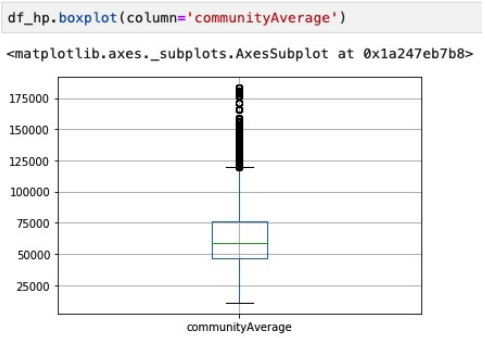


Figure 22: Box plot of community average price

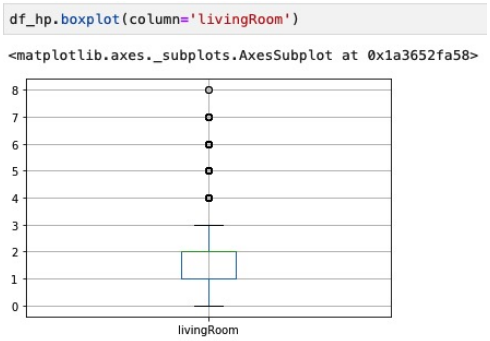


Figure 23: Box plot of living room

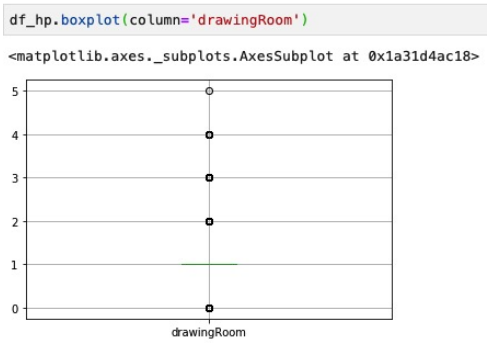


Figure 24: Box plot of drawing room

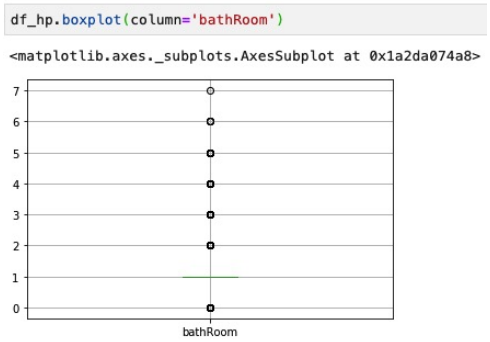


Figure 25: Box plot of bathroom

As shown on above figures, we can see that the total price and square attributes have some outliers and the extreme values. In this project, we use the standard deviation to check if thet are the outliers and the extreme values, if so remove them from the data. In detaild, we need to check every record in 'totalPrice' and 'square' which value is greater than three standard deviations of attribute distribution itself, then we remove them from datatset, because the dataset is large enough, and it can afford to lose few outliers and extreme values to make sure the model going to have a better accuracy.

Also, these outliers and extreme values are only a small partition in the data, removing them may not lose the critical points of data. Considering the error of observation, it is a good idea to discard them when we analyze the data. The Python code has shown below, we can see after remove the outliers and the extreme values, the dataset has 288938 records left.

```
df_hp.info()
df_hp = df_hp[np.abs(df_hp.totalPrice - df_hp.totalPrice.mean()) <= (3*df_hp.totalPrice.std())]
df_hp = df_hp[np.abs(df_hp.square - df_hp.square.mean()) <= (3*df_hp.square.std())]
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 297701 entries, 2 to 318818
Data columns (total 28 columns):
url                    297701 non-null object
id                     297701 non-null object
Lng                    297701 non-null float64
Lat                    297701 non-null float64
Cid                    297701 non-null int64
tradeYear              297701 non-null int64
tradeMonth             297701 non-null int64
tradeDay               297701 non-null int64
DOM                    297701 non-null float64
followers              297701 non-null int64
totalPrice             297701 non-null float64
price                  297701 non-null int64
square                 297701 non-null float64
livingRoom             297701 non-null int64
drawingRoom            297701 non-null int64
kitchen                297701 non-null int64
bathRoom               297701 non-null int64
floor                  297701 non-null int64
buildingType           297701 non-null float64
constructionTime       297701 non-null float64
renovationCondition    297701 non-null int64
buildingStructure      297701 non-null int64
ladderRatio            297701 non-null float64
elevator               297701 non-null int64
fiveYearsProperty      297701 non-null int64
subway                 297701 non-null int64
district               297701 non-null int64
communityAverage       297701 non-null float64
dtypes: float64(9), int64(17), object(2)
memory usage: 65.9+ MB
```

Figure 26: Before removing outliers and extreme values

```
df_hp.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 288938 entries, 2 to 318818
Data columns (total 28 columns):
url                    288938 non-null object
id                     288938 non-null object
Lng                    288938 non-null float64
Lat                    288938 non-null float64
Cid                    288938 non-null int64
tradeYear              288938 non-null int64
tradeMonth             288938 non-null int64
tradeDay               288938 non-null int64
DOM                    288938 non-null float64
followers              288938 non-null int64
totalPrice             288938 non-null float64
price                  288938 non-null int64
square                 288938 non-null float64
livingRoom             288938 non-null int64
drawingRoom            288938 non-null int64
kitchen                288938 non-null int64
bathRoom               288938 non-null int64
floor                  288938 non-null int64
buildingType           288938 non-null float64
constructionTime       288938 non-null float64
renovationCondition    288938 non-null int64
buildingStructure      288938 non-null int64
ladderRatio            288938 non-null float64
elevator               288938 non-null int64
fiveYearsProperty      288938 non-null int64
subway                 288938 non-null int64
district               288938 non-null int64
communityAverage       288938 non-null float64
dtypes: float64(9), int64(17), object(2)
memory usage: 63.9+ MB
```

Figure 27: After removing outliers and extreme values

3.3 Construct the data

Sometimes we need to construct the new data from existing dataset, which may explain the data better, and easy to fit the models. In this project, for showing the time plot of house price, it is necessary to merge the date. Therefore, here we create a new attribute to put tradeYear, tradeMonth, tradeDay fields together to become a continuous timestamp from 2002 to 2018.

```
hp0 = df_hp
hp0["tradeTime"] = hp0["tradeYear"].map(str)+'/' + hp0["tradeMonth"].map(str) + '/' +hp0["tradeDay"].map(str)
hp0['tradeTime']
```

```
2          2015/7/20
17        2016/10/22
64         2013/9/9
66        2014/10/19
119        2016/7/29
130        2016/9/27
143         2016/9/2
232        2017/3/11
236        2017/2/20
246        2017/3/14
251        2017/6/30
312         2015/5/1
325        2016/1/14
397        2014/2/17
428        2015/6/17
440        2016/2/27
460        2016/5/15
484         2016/3/6
489         2013/5/8
490        2013/8/25
501        2016/3/18
502         2016/6/9
531         2012/8/8
532         2012/8/8
536        2013/10/8
537        2013/10/8
545         2016/2/1
551         2016/5/7
552         2016/5/7
574        2016/8/10
```

Figure 28: Timestamp

3.4 Integrate various data sources

In some project, it may contain various data files from different data management system and data resource. For process and analyse these data, we have to integrate these data together.

In this research, we assume all instances are recorded in a two xlsx file. Therefore, we should integrate two data files together and become one file.

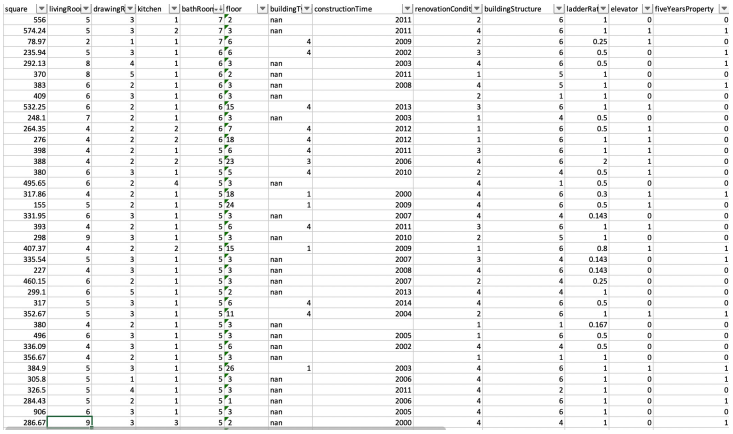
As shown in figure below, I upload these two files on Jupyter Notebook, then check the files and figure out the two files contain some attributes, but different instances. Therefore, here we can use append function to merge them. I assigned the merged dataset as final_set. The information on final_set is shown by info () function. After this process, we can get the complete data from the both xlsx files.

```
In [11]:  data1 = pd.read_excel("house_price_beijing_1.xlsx")
          data2 = pd.read_excel("house_price_beijing_2.xlsx")
          dataset = data1.append(data2)

          export_csv = dataset.to_csv (r'Newdata.csv', index = None, header=True)
          final_set = pd.read_csv("Newdata.csv")

In [13]:  final_set.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 318819 entries, 0 to 318818
Data columns (total 28 columns):
url                   318819 non-null object
id                    318819 non-null object
Lng                   318819 non-null float64
Lat                   318819 non-null float64
Cid                   318819 non-null int64
tradeYear             318819 non-null int64
tradeMonth            318819 non-null int64
tradeDay              318819 non-null int64
DOM                   160849 non-null float64
followers             318819 non-null int64
totalPrice            318819 non-null float64
price                 318819 non-null int64
square                318819 non-null float64
livingRoom            318819 non-null int64
drawingRoom           318819 non-null int64
kitchen               318819 non-null int64
bathRoom              318819 non-null int64
floor                 318819 non-null int64
buildingType          316798 non-null float64
constructionTime      299536 non-null float64
renovationCondition   318819 non-null int64
buildingStructure     318819 non-null int64
ladderRatio           318819 non-null float64
elevator              318819 non-null int64
fiveYearsProperty     318819 non-null int64
subway                318819 non-null int64
district              318819 non-null int64
communityAverage      318356 non-null float64
dtypes: float64(9), int64(17), object(2)
memory usage: 68.1+ MB
```

Figure 29: Integration 2 xlsx files

url	id	Lng	Lat	Cid	tradeYear	tradeMonth	tradeDay	DOM	followers	totalPrice	price
https://bj.lia	1.011E+11	116.291921	40.146327	1111043196439	2016	8	31	60	1	1450	26,080
https://bj.lia	BJCY9030066	116.437966	40.018379	1111027382232	2016	3	8	1	14	4350	75,753
https://bj.lia	BJHD895538	116.287746	40.038617	1111027381142	2015	7	20	nan	2	380	48,120
https://bj.lia	1.0109E+11	116.455694	39.807011	1111027381865	2016	6	16	169	36	400	16,954
https://bj.lia	1.0109E+11	116.2755	40.036213	1111027376136	2017	1	19	316	43	2429	83,148
https://bj.lia	1.0109E+11	116.317097	39.632299	1111027374277	2016	7	27	100	8	788	21,298
https://bj.lia	1.011E+11	116.542335	39.754708	1111027379800	2017	2	8	184	37	1570	40,993
https://bj.lia	1.011E+11	116.41919	40.150537	1111047627815	2016	12	31	36	0	1260	30,807
https://bj.lia	1.011E+11	116.543161	40.097129	1111041170735	2017	2	7	2	10	1500	28,183
https://bj.lia	BJCP003827	116.303615	40.09088	1111027377758	2012	12	24	nan	0	1140	45,950
https://bj.lia	BJCP922141	116.403975	40.04479	1111043644453	2016	5	18	nan	0	400	15,132
https://bj.lia	BJCY843383	116.380489	40.025173	1111042784632	2012	2	27	1	0	698.3	25,300
https://bj.lia	1.0109E+11	116.275723	40.046164	1111041084751	2016	8	19	308	7	2690	67,588
https://bj.lia	1.0109E+11	116.319894	39.918185	1111027382161	2016	9	21	251	9	2550	65,722
https://bj.lia	1.0109E+11	116.647632	39.940239	1111027375565	2016	7	12	123	13	928	24,422
https://bj.lia	1.0109E+11	116.433036	40.021767	1111027375015	2016	7	18	81	3	2400	48,422
https://bj.lia	1.0109E+11	116.423295	39.902249	1111027379822	2016	6	27	54	9	1550	48,764
https://bj.lia	1.0109E+11	116.497474	39.810115	1111027377957	2016	10	22	168	47	808	52,130
https://bj.lia	1.0109E+11	116.542335	39.754708	1111027379800	2016	9	24	136	5	1150	34,644
https://bj.lia	1.0109E+11	116.275723	40.046164	1111041084751	2016	8	30	102	2	2987	76,006
https://bj.lia	1.0109E+11	116.319699	39.62703	1111027378149	2016	6	26	36	1	600	20,135
https://bj.lia	1.0109E+11	116.383881	39.909637	1111027374196	2017	3	14	288	134	4304	105,654
https://bj.lia	1.011E+11	116.542335	39.754708	1111027379800	2016	7	22	32	3	870	25,929
https://bj.lia	1.011E+11	116.317277	40.062967	1111046327190	2016	7	10	18	1	1790	78,855
https://bj.lia	1.011E+11	116.437127	40.021589	1111027382584	2016	8	31	63	1	2940	63,893
https://bj.lia	1.011E+11	116.188253	40.233417	1111027375941	2016	9	18	75	5	720	24,073
https://bj.lia	1.011E+11	116.530942	40.062347	1111062904348	2016	10	31	94	8	1700	53,628
https://bj.lia	1.011E+11	116.440069	39.902423	1111027374222	2016	10	29	86	29	2850	80,813
https://bj.lia	1.011E+11	116.530942	40.062347	1111062904348	2016	10	10	66	6	2130	56,053
https://bj.lia	1.011E+11	116.433036	40.021767	1111027375015	2016	9	24	6	0	2728	55,000
https://bj.lia	1.011E+11	116.403693	40.116058	1111027380230	2016	10	6	18	10	600	17,853
https://bj.lia	1.011E+11	116.530942	40.062347	1111062904348	2017	3	7	150	12	2490	69,813
https://bj.lia	1.011E+11	116.321095	39.946957	1111027379026	2017	3	5	107	7	3950	102,625
https://bj.lia	1.011E+11	116.575746	39.911252	1111027379319	2016	12	31	26	4	1454	47,548
https://bj.lia	1.011E+11	116.317097	39.632299	1111027374277	2017	11	30	346	175	1260	38,592
https://bj.lia	1.011E+11	116.575746	39.911252	1111027379319	2017	3	27	41	8	1480	52,034
https://bj.lia	1.011E+11	116.447681	40.002373	1111027375841	2017	3	24	32	0	11600	128,036
https://bj.lia	1.011E+11	116.39771	40.089793	1111027373834	2017	7	19	140	45	1890	65,930

Figure 30: First file



square	livingRoom	drawingR	kitchen	bathRoom	floor	buildingTy	constructionTime	renovationCondit	buildingStructure	ladderRa	elevator	fiveYearsProperty
556	5	3	1	7	2	nan	2011	2	6	1	0	0
574.24	5	3	2	7	3	nan	2011	4	6	1	1	1
78.97	2	1	1	7	6	4	2009	2	6	0.25	1	0
235.94	5	3	1	6	6	4	2002	3	6	0.5	0	1
292.13	8	4	1	6	3	nan	2003	4	6	0.5	0	1
370	8	5	1	6	2	nan	2011	1	5	1	0	0
383	6	2	1	6	3	nan	2008	4	5	1	0	1
409	6	3	1	6	3	nan		2	1	1	0	0
532.25	6	2	1	6	15	4	2013	3	6	1	1	0
248.1	7	2	1	6	3	nan	2003	1	4	0.5	0	0
264.35	4	2	2	6	7	4	2012	1	6	0.5	1	0
276	4	2	2	6	18	4	2012	1	6	1	1	0
398	4	2	1	5	6	4	2011	3	6	1	1	0
388	4	2	2	5	23	3	2006	4	6	2	1	0
380	6	3	1	5	5	4	2010	2	4	0.5	1	0
495.65	6	2	4	5	3	nan		4	1	0.5	0	0
317.86	4	2	1	5	18	1	2000	4	6	0.3	1	1
155	5	2	1	5	24	1	2009	4	6	0.5	1	0
331.95	6	3	1	5	3	nan	2007	4	4	0.143	0	0
393	4	2	1	5	6	4	2011	3	6	1	1	0
298	9	3	1	5	3	nan	2010	2	5	1	0	0
407.37	4	2	2	5	15	1	2009	1	6	0.8	1	1
335.54	5	3	1	5	3	nan	2007	3	4	0.143	0	1
227	4	3	1	5	3	nan	2008	4	6	0.143	0	0
460.15	6	2	1	5	3	nan	2007	2	4	0.25	0	0
299.1	6	5	1	5	2	nan	2013	4	4	1	0	0
317	5	3	1	5	6	4	2014	4	6	0.5	0	0
352.67	5	3	1	5	11	4	2004	2	6	1	1	1
380	4	2	1	5	3	nan		1	1	0.167	0	0
496	6	3	1	5	3	nan	2005	1	6	0.5	0	0
336.09	4	3	1	5	6	nan	2002	4	4	0.5	0	0
356.67	4	2	1	5	3	nan		1	1	1	0	0
384.9	5	3	1	5	26	1	2003	4	6	1	1	1
305.8	5	1	1	5	3	nan	2006	4	6	1	0	1
326.5	5	4	1	5	3	nan	2011	4	2	1	0	0
284.43	5	2	1	5	1	nan	2006	4	6	1	0	1
906	6	3	1	5	3	nan	2005	4	6	1	0	0
286.67	9	3	3	5	2	nan	2000	4	4	1	0	1

Figure 31: Second file

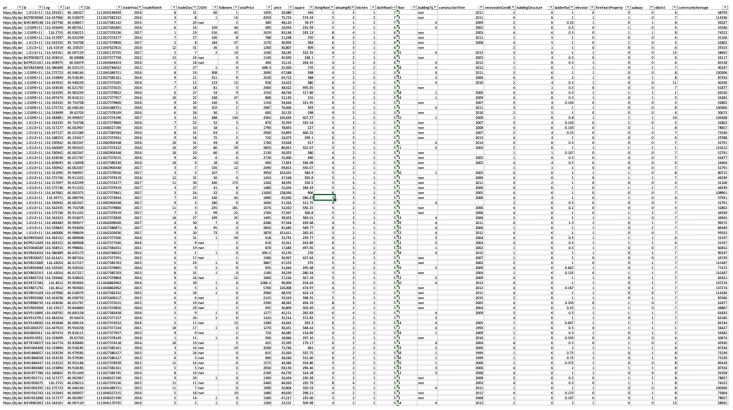


Figure 32: After integration

3.5 Format the data as required

Formatting data can make the type of data correct. We should check the format of house price and other features in the dataset in correct format. Ensuring the record ID and the target are right.

In this project, we are going to fit a model to predict the house price in Beijing. As we know, the house price prediction is a regression problem, therefore, we need to make sure the attributes contains the numeric values which can solve most regression problems. The formats of features shown in Figure 33 below. Through the figure, we can see that only the 'url' and 'id' features contain character values. However, these two attributes will be dropped on next chapter due to the fact that that are unique attribute, so they can not affect the model performance in this project.

```
df_hp.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 288938 entries, 2 to 318818
Data columns (total 28 columns):
url                    288938 non-null object
id                     288938 non-null object
Lng                    288938 non-null float64
Lat                    288938 non-null float64
Cid                    288938 non-null int64
tradeYear              288938 non-null int64
tradeMonth             288938 non-null int64
tradeDay               288938 non-null int64
DOM                    288938 non-null float64
followers              288938 non-null int64
totalPrice             288938 non-null float64
price                  288938 non-null int64
square                 288938 non-null float64
livingRoom             288938 non-null int64
drawingRoom            288938 non-null int64
kitchen                288938 non-null int64
bathRoom               288938 non-null int64
floor                  288938 non-null int64
buildingType           288938 non-null float64
constructionTime       288938 non-null float64
renovationCondition    288938 non-null int64
buildingStructure      288938 non-null int64
ladderRatio            288938 non-null float64
elevator               288938 non-null int64
fiveYearsProperty      288938 non-null int64
subway                 288938 non-null int64
district               288938 non-null int64
communityAverage       288938 non-null float64
dtypes: float64(9), int64(17), object(2)
memory usage: 63.9+ MB
```

Figure 33: Format the data

4 Data Transformation

In House Price in Beijing dataset, it is necessary to complete the data transformation to make the analysis much more straightforward. This part of the procedure consists of both reducing the data and projecting the data.

4.1 Reduce the data

The attributes of 'id' and 'url' should be removed from the dataset, because the data is all come from same url, which is lianjia.com. It is the irrelative factor for house price in this case. Also, it is easy to realise that record ID (id) is irrelative factor as well for the house price. Therefore, we also should remove id from dataset and prevent them to the following steps.

```
df_hp_sub = df_hp
df_hp_sub = df_hp_sub.drop(['id', 'url','kitchen'], axis = 1)
df_hp_sub.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 288938 entries, 2 to 318818
Data columns (total 26 columns):
Lng                    288938 non-null float64
Lat                    288938 non-null float64
Cid                    288938 non-null int64
tradeYear              288938 non-null int64
tradeMonth             288938 non-null int64
tradeDay               288938 non-null int64
DOM                    288938 non-null float64
followers              288938 non-null int64
totalPrice             288938 non-null float64
price                  288938 non-null int64
square                 288938 non-null float64
livingRoom             288938 non-null int64
drawingRoom            288938 non-null int64
bathRoom               288938 non-null int64
floor                  288938 non-null int64
buildingType           288938 non-null float64
constructionTime       288938 non-null float64
renovationCondition    288938 non-null int64
buildingStructure      288938 non-null int64
ladderRatio            288938 non-null float64
elevator               288938 non-null int64
fiveYearsProperty      288938 non-null int64
subway                 288938 non-null int64
district               288938 non-null int64
communityAverage       288938 non-null float64
tradeTime              288938 non-null object
dtypes: float64(9), int64(16), object(1)
memory usage: 59.5+ MB
```

Figure 34: Remove unimportant features

Moreover, the project used feature selection function in Sklearn to find the feature importance. As shown in Figure below, the most important attributes are tradeYear,

square, communityAverage, livingRoom, renovationCondition, bathRoom, and drawingRoom. There are several guesses about this result. First of all, because the larger building square can have a larger house, then it may have the higher price. Secondly, the property market often has a periodic attribute, so the trade year is also a importance factor. Moreover, the number of living room, bathroom and drawing room also can effect the square of the house, so they are definitely have high correlation with the house price. In addition, the renovationCondition shows the condition of the house, the better condition means the house may have higher price. Also, the communityAverage can reflect the location of house, the house in a better are normally has a higher price.

However, even if we know some attributes are not that important, I still decide to keep these attributes and will feed these attributes into the model creation due to two reasons. First, these attributes which are not very important may affect the model performance as well. second, we have enough computing power in this project. Therefore, we will keep these attributes, and try to make a better model.

In [10]:
```
Y = df_hp_sub['totalPrice']
X = df_hp_sub
from sklearn.feature_selection import f_classif
from sklearn.feature_selection import SelectKBest
bestfeatures = SelectKBest(score_func=f_classif , k=10)


fit = bestfeatures.fit(X,Y)
dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(X.columns)
#concat two dataframes for better visualization
featureScores = pd.concat([dfcolumns,dfscores],axis=1)
featureScores.columns = ['Specs','Score']  #naming the dataframe columns
print(featureScores.nlargest(20,'Score'))
```

```
                Specs      Score
8          totalPrice        inf
3           tradeYear  22.625726
9              square  17.874587
23    communityAverage  15.629885
10         livingRoom  12.155064
16  renovationCondition   8.516921
12           bathRoom   8.154877
11        drawingRoom   6.676742
7           followers   3.831078
19           elevator   3.592496
17   buildingStructure   3.184148
6                DOM   3.175255
13             floor   3.049320
21            subway   2.272344
0                Lng   1.941695
20   fiveYearsProperty   1.848927
2                Cid   1.751696
1                Lat   1.644736
15    constructionTime   1.567300
14        buildingType   1.552413
```

Figure 35: Feature Selection

4.2 Project the data

According to the time plot graph, the house price in Beijing may have a relationship with the season, so in this part aims to find a correlation between house price and the season. Firstly, it is vital to create a new attribute called season to show the price of sold house in each season, so we can find out the difference of price among each season. Here we set January to March as winter, April to June as spring, July to September as summer and October to December as Autumn.

```
def season(tradeMonth):
    if tradeMonth < 4:
        return 'winter'
    elif tradeMonth < 7:
        return 'spring'
    elif tradeMonth < 10:
        return 'summer'
    else:
        return 'autumn'
df_hp_sub.loc[:, 'season'] = df_hp_sub['tradeMonth'].apply(season)
df_hp_sub[['season','tradeMonth']]
```

	season	tradeMonth
2	summer	7
17	autumn	10
64	summer	9
66	autumn	10
119	summer	7
130	summer	9
143	summer	9
232	winter	3
236	winter	2
246	winter	3
251	spring	6
312	spring	5
325	winter	1

Figure 36: Create the new attribute as season

Then, we uses 'sort' method to find the 1000 highest house price to find which season has the most high price.

```
sort = df_hp_sub.sort_values(by='totalPrice', ascending=False).head(100)
sort.groupby(['season'])['totalPrice'].count().plot.bar()
plt.title('count of top 100 highest house price over season')
plt.xlabel('season')
plt.ylabel('count')
```

```
Text(0, 0.5, 'count')
```

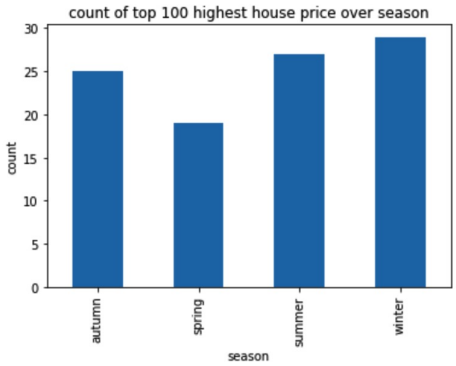


Figure 37: Highest sold prices group by season

As shown in figure, it is clearly to find out the winter have the most highest sold price in all seasons.

5 Data mining methods selection

5.1 Match and discuss the objectives of data mining to data mining methods

The objectives of data mining in this study should focus on finding which factors affect the house price in Beijing and predict the house price based on different features in the dataset. Also, the study may reveal the price trend of the house price in Beijing. Therefore, this project may be involved in a variety of data mining methods and try to find the most suitable model to make the prediction.

Therefore, The research should focus on the following data mining goals:

- Explanation: Finding out important features relevant to influence the house price. Explaining the relationship between features and the house price.
- Explanation: Describing how the features can affect the house price. Explaining the latitude of the features which can affect the response variables.
- Prediction: Predicting the trend of the house price. Predict the response variables by provided features

This part of the study focuses on finding which factors affect the house price in Beijing and predict the house price based on different attributes in the dataset. Also, the study may reveal the price trend of the house price in Beijing. Therefore, this project may be involved in a variety of data mining methods and try to find the most suitable model to make the prediction.

5.2 Select the appropriate data mining method based on discussion

There are three data mining methods based on general discussion, including classification, clustering, and regression. This study is going to analyze, which is the more suitable method for this dataset.

- **Supervised Learning - Classification**
  Classification is the most commonly used analysis method in data mining. [8]The classification model is suitable for systems that generate discrete responses.[8] There are several classification models that can be grouped under mathematically intensive models, hierarchical models, and layered models. Some of these models. Classification is not an ideal method for predicting house price index because the house price index is continuous value which makes the classified catalogue much difficult.

- **Unsupervised Learning - Clustering**
  [8]The clustering model focuses on identifying groups of similar records and labeling the records according to the group to which the records belong. [8]This is done without prior knowledge of the group and its characteristics. In fact, you may not even know the number of groups you are looking for. This is why the clustering model differs from other machine learning techniques, because there are no predefined output or target fields for model prediction. These models are often referred to as unsupervised learning models, because there are no external criteria to judge the classification performance of the model. These models do not have correct or incorrect answers. Their value depends on their ability to capture interesting groups in the data and provide a useful description of these groups.[8]

- **Supervised Learning - Regression**
  Regression analysis is the most common data analysis method. [8]Regression models are predictors and they are suitable for the systems that produce continuous responses. There are several regression models, which include standard regression, random forest regression, lasso regression, and elastic-net regression. This project is going to use regression model to analyze the important indicators for house price.[8]

This research aims to explore the relationship between features and house price in our dataset and predict the trend of the house price. Due to the fact that the house price is the continuous variable, the regression can be adopted as the data mining method in this study. The specific algorithms we would use will be discussed in the next section

6 Data mining algorithms selection

There are various modeling methods can be taken from machine learning, artificial intelligence, and statistics to fit our own model. Each of the method he methods allows us to derive the information from our data and create the predictive models. Moreover, each of method has certain strengths can be suited to deal with particular types of problems. In this part, we are going to figure out how to select proper algorithms to deal with house price dataset.

6.1 Conduct exploratory analysis and discuss

The data mining methods have different type of algorithms, such as decision tree, random forest, Bayes algorithms and so on. Each type of algorithm has its unique strengths and weakness.

The purpose of this study is to explore the main factors that influence trend of house price and to predict house price based on these factors. Therefore, the method is to use the value of one or more input fields to predict the value of one or more output (or target) fields, so the select of algorithm is mainly based on the "supervised" model, including the decision tree (C&R tree, QUEST, CHAID and C5.0 algorithms), regression (linear, logistic, generalized linear, and Cox regression algorithms), neural networks and Bayesian networks. For example, [9]Neural networks offer a number of advantages, including requiring less formal statistical training, ability to implicitly detect complex nonlinear relationships between dependent and independent variables, ability to detect all possible interactions between predictor variables, and the availability of multiple training algorithms. Disadvantages include its black box nature, greater computational burden, proneness to the overfitting, and the empirical nature of model development.

Therefore, the algorithm taking should be considered to choose appropriate algorithms by comparing the accuracy of target and complexity.

6.2 Select data mining algorithms based on discussion

This project try to find the relationship of house price and variables in the dataset and predict the house price in the future. The **Supervised Learning**, **Unsupervised Learning** and **Semis-supervised Learning** are different, and we can use them in different situations. We can use Supervised Learning when we have input variables and an output, and we use an algorithm to learn the mapping function from the input to the

output. However, in Unsupervised Learning, we can only have input data with no corresponding output variables.

- In this project, we are trying to find the relation of house price and variables in the dataset and predict the house price in the future. Due to the fact that the house price is continuous variable and most of data in the dataset are numeric, so using **regression analysis** is a good way to figure out the impacted factors of the house price and predict the trend of house price.
- To get a relatively high accuracy model, I have to try different regression models to find the best one to create our model. The algorithms in this project will use in the next step are **Simple Linear Regression**, **Multiple Regression**, and **Random Forest Regression**.

Definition and explanation of data mining algorithms we choose

- **Simple Linear Regression**
  [3]Simple Linear Regression estimates the coefficients of the linear equation, involving one independent variables, that best predict the value of the dependent variable. For example, you can try to predict a salesperson's total yearly sales (the dependent variable) from one of the independent variables such as age, education, and years of experience.

- **Multiple Linear Regression**
  [1]Multiple regression is a statistical method for studying the relationship between a single dependent variable and one or more independent variables. It is unquestionably the most widely used statistical technique in the social sciences. It is also widely used in the biological and physical sciences. [1]There are two major uses of multiple regression: prediction and causal analysis. In a prediction study, the goal is to develop a formula for making predictions about the dependent variable, based on the observed values of the independent variables.

- **Random Forest**
  [3]Random Forest is an advanced implementation of a bagging algorithm with a tree model as the base model. In random forests, each tree in the ensemble is built from a sample drawn with replacement (for example, a bootstrap sample) from the training set. When splitting a node during the construction of the tree, the split that is chosen is no longer the best split among all features. Instead, the split that is picked is the best split among a random subset of the features. Because of this randomness, the bias of the forest usually slightly increases (with respect to the bias of a single non-random tree) but, due to averaging, its variance also decreases, usually more than compensating for the increase in bias, hence yielding

an overall better model.

For regression, we use regression and feature selection algorithms to analyse. For summarization, we use distribution graph and time plot graph to show the information.

6.3 Select appropriate models and choose relevant parameters

Parameter is most important factor for the model. So this step will show that how to set up the models with the chosen algorithms properly which the method we have selected on above.

6.3.1 Simple Linear Regression Model Parameter Settings

For Linear Regression model, house price is target, and we try to find relationship between house price and a most important variable in the dataset. We using partitioned data to train and test data which can avoid overfitting and build model for each spilt using the method of enter. To be convenient, we include constant in equation.
The simple linear regression fit model in a linear relationship between a response and

```
In [10]: Y = df_hp_sub['totalPrice']
         X = df_hp_sub
         from sklearn.feature_selection import f_classif
         from sklearn.feature_selection import SelectKBest
         bestfeatures = SelectKBest(score_func=f_classif , k=10)


         fit = bestfeatures.fit(X,Y)
         dfscores = pd.DataFrame(fit.scores_)
         dfcolumns = pd.DataFrame(X.columns)
         #concat two dataframes for better visualization
         featureScores = pd.concat([dfcolumns,dfscores],axis=1)
         featureScores.columns = ['Specs','Score']  #naming the dataframe columns
         print(featureScores.nlargest(20,'Score'))
```

```
                  Specs      Score
8           totalPrice        inf
3            tradeYear  22.625726
9               square  17.874587
23    communityAverage  15.629885
10          livingRoom  12.155064
16  renovationCondition   8.516921
12             bathRoom   8.154877
11         drawingRoom   6.676742
7             followers   3.831078
19             elevator   3.592496
17   buildingStructure   3.184148
6                  DOM   3.175255
13               floor   3.049320
21              subway   2.272344
0                  Lng   1.941695
20   fiveYearsProperty   1.848927
2                  Cid   1.751696
1                  Lat   1.644736
15    constructionTime   1.567300
14        buildingType   1.552413
```

Figure 38: Feature Importance

just one explanatory variable. If we want to predict house prices, our response variable is totalPrice. However, for a simple linear model we only select one feature. Here we use feature selection function in sklearn to find the feature importance, and shown the features importance on above. Then, the most important attribute is tradeYear, which is a categorical attribute, so it is not a good predictor for the continuous response(totalPrice). Therefore, I choose the second important feature which is square (living area) to be the predictor in this simple linear regression model. Also, When we examine the correlation matrix, we find out that totaPrice has the highest correlation coefficient with square (living area). Thus, I should use square (living area) as the feature to fit into the model. However, if we want to examine the relationship between totalPrice and another feature, you may fit the feature you chosen into the model to get their correlation. The following graph shows the model parameter settings of simple linear regression.

```
In [8]: import findspark
        findspark.init('/home/ubuntu/spark-2.1.1-bin-hadoop2.7')
        import pyspark
        from pyspark.sql import SparkSession
        spark = SparkSession.builder.appName('simple_linear_regression').getOrCreate()
        from pyspark.ml.regression import LinearRegression

In [9]: df_hp_sub_spark1 = spark.createDataFrame(df_hp_sub)

In [29]: df_hp_sub_spark1.printSchema()

...

In [11]: from pyspark.ml.linalg import Vectors
         from pyspark.ml.feature import VectorAssembler

In [12]: assembler = VectorAssembler(
             inputCols=["square"],
             outputCol="features")

In [14]: output1 = assembler.transform(df_hp_sub_spark1)

In [16]: final_data1 = output1.select("features",'totalPrice')
         final_data1.show()

...

In [18]: train_data,test_data = final_data1.randomSplit([0.8,0.2])

In [19]: lr1 = LinearRegression(labelCol='totalPrice')

In [21]: lrModel1 = lr1.fit(train_data)
```

Figure 39: Simple Linear Regression Model Parameter Settings

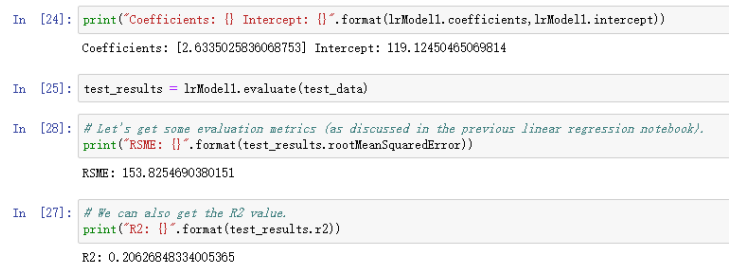


Figure 40: Simple Linear Regression Model Result

6.3.2 Multiple Linear Regression Model Parameter Settings

Sometimes, the simple linear regression may not have a good performance. In order to improve the performance of the model, we should fit more features into the model. Therefore, the more complex multiple regression should be created. In this project, we are going to try multiple regression on different features. the first time, we only feed selected features into the model, and the second time will feed all features to the model. Then, we can find out which one going to have a better performance.

Figure 41: Selected Features Multiple Linear Regression Model Parameter Settings

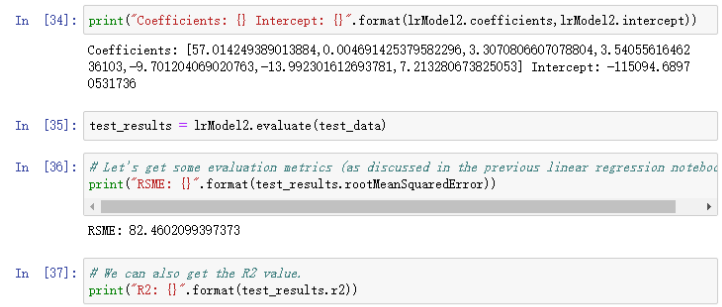


Figure 42: Selected Features Multiple Linear Regression Model Result

Figure 43: All Features Multiple Linear Regression Model Parameter Settings

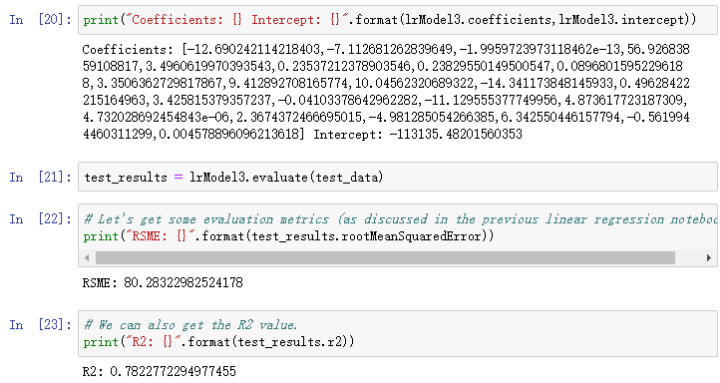


Figure 44: All Features Multiple Linear Regression Model Result

6.3.3 Random Forest Model Parameter Settings

The figure below shows the options of random forest. By using random forest algorithm, we use default setting from Spark due to the fact that my datasets is relative large which contains over 280,000 instance. It is quite hard to optimise the parameters. However, I customise one parameter which is the number of trees in the model which has been set to 200. Also, In the model, we partitioned data to train and test data which can avoid over fitting.

However, the relative parameters of the random forest regression have been listed on below for future study.

- numTrees - Number of trees in the forest. This parameter is usually the most important setting.
- maxDepth - Max number of levels in each decision tree.
- featureSubsetStrategy - Number of features to use as candidates for splitting at each tree node
- minInfoGain - Minimum information gain for a split to be considered at a tree node
- minInstancesPerNode - Minimum number of instances each child must have after split

Figure 45: Random Forest Model Setting

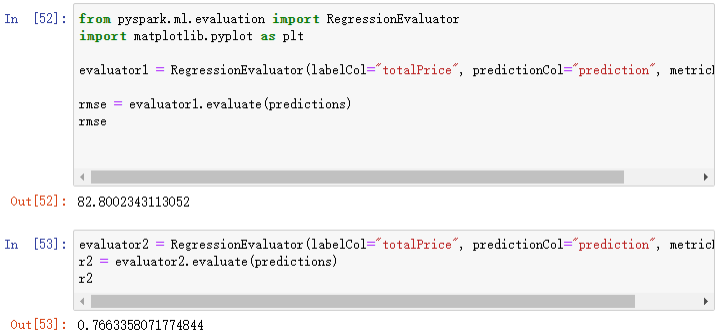


Figure 46: Random Forest Model Result

[2]For regression models other than the linear model, R-squared type goodness-of-fit summary statistics have been constructed for particular models using a variety of methods. We propose an R-squared measure of goodness of fit for the class of exponential family regression models, which includes logit, probit, Poisson, geometric, gamma, and exponential. [2]This R-squared is defined as the proportionate reduction in uncertainty, measured by Kullback-Leibler divergence, due to the inclusion of regressors. Under further conditions concerning the conditional mean function it can also be interpreted as the fraction of uncertainty explained by the fitted model.

Therefore, the model which has the closest value to 1 is the best to be our predictive model. In this project, we can choose multiple regression to fit a model, because it has the best result which R squared is 0.78. The result is shown on the above figure.

7 Data mining

7.1 Create and justify test designs

[6]In machine learning, there are several ways of data partitioning for experimentation. The most popular ways are typically referred to as training/test partitioning or cross-validation. [6]The training/test partitioning typically involves the partitioning of the data into a training set and a test set in a specific ratio, e.g., 70% of the data are used as the training set and 30% of the data are used as the test set. This data partitioning can be done randomly or in a fixed way. However, for a better model performance, we decide to use 80/20 to partition data into training and test set.

After a model has been processed by using the training set, we test the model by making predictions against the test set. [5]Because the data in the testing set already contains known values for the attribute that we want to predict, it is easy to determine whether the models guesses are correct.

To test the results of the algorithm, this project used randomSplit([0.8,0.2]) method to divide two subsets, the training and test datasets, which were 80% and 20%, respectively. However, in the following steps, this ratio may be changed based on the results. In addition, this project created an Evaluation variable to display the results.

7.2 Conduct data mining

- **Parameter setting**
  The parameter setting has been shown on previous chapter. Also, depends on we choose multiple regression to predict house prices, the goal of these algorithms is totalPrice. The input is the relative elements of the dataset. In addition, regression algorithms can help to find out which factors can impact the price.

- **Model setting**
  For each algorithms, operator can set different model setting for each algorithm. This study uses both of testing set and training set in all of algorithms to reduce the error.

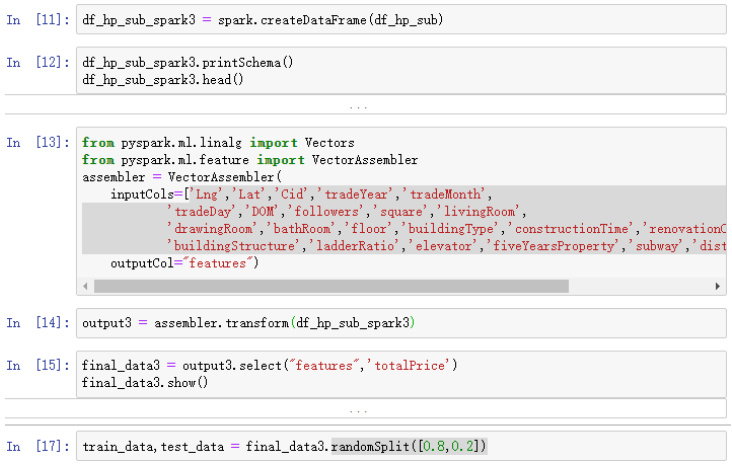


Figure 47: Training and Test Set

- **Run and Result**

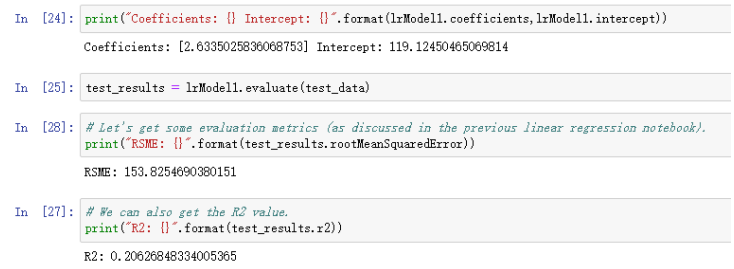


Figure 48: Simple Linear Regression Model Result

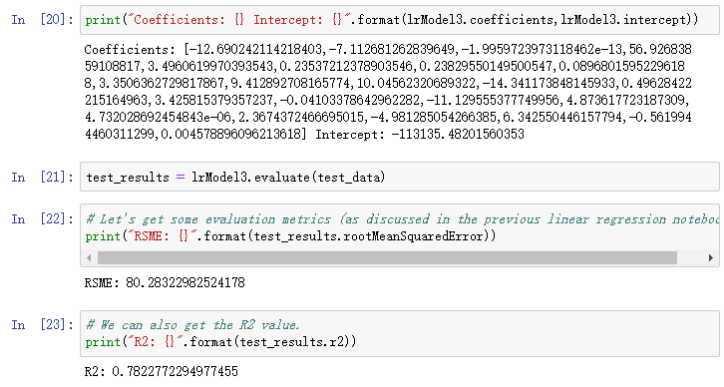


Figure 49: All Features Multiple Linear Regression Model Result

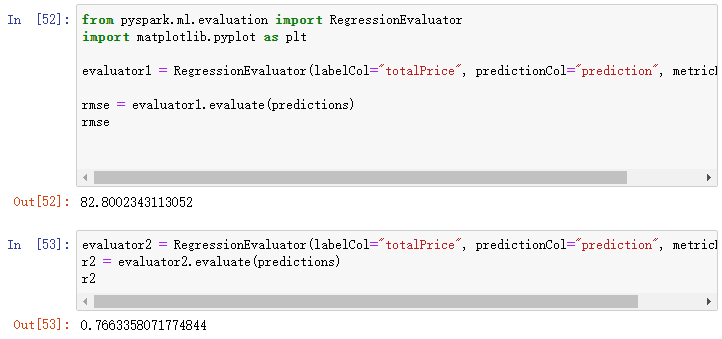


Figure 50: Random Forest Model Result

The result of Simple Linear Regression, Multiple Linear Regression and Random Forest Regression has been showed above. From the figure, we can see the results of different algorithms. The Multiple Linear Regression Model has the best result which R squared is the largest 0.78.

7.3 Search for patterns

Though the result of different models in above figure, we can set Simple Linear regression as the benchmark. Simple Linear regression has the lowest R squared in test set which is acceptable because we want to use the Simple Linear regression as benchmark and explain the correlation among the attributes in the dataset and house price, and predict the trend of house price. The simple linear regression get a really bad R squared in test set than that of multiple linear regression and random forest regression. Therefore, I will use the multiple linear regression explain the affection of house price on various variables, interpret the data mining pattern.

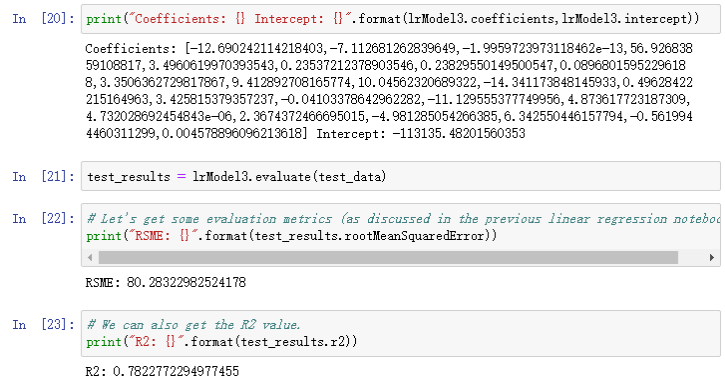


Figure 51: Multiple Linear Regression Model Result

8 Interpretation

8.1 Study and discuss the mined patterns

Since our data mining method is regression, I used three regression algorithms to the dataset: Simple Linear Regression, Regression, Multiple Regression, and Random Forest Regression to mine the pattern. Then, we can compare the pattern mined by these algorithms. As mentioned previously, the Multiple Regression has the highest accuracy. Therefore, we will mainly interpret it section 8.3. The Simple Linear Regression will also be interpreted because it is the benchmark and provide the explanation for relationships between house price and response variables.

The establishment of the model is a spiral ascending and continuous optimization process. At the end of each model, it is necessary to determine whether the results of the analysis are meaningful, whether the features are obvious or acceptable. If the results are not ideal, we need to try to adjust and optimize the model. For achieving this objective, it can be done by adjusting the input values, variables, coefficients of the mining algorithm, etc.

Through the above processing, a series of analysis results and patterns are obtained. They are multifaceted descriptions of the target problem. At this point, it needs to be verified and evaluated to obtain reasonable and complete decision information. The results of the model need to be verified by comparison, accuracy and support to determine the value of the model. At this stage, we need to introduce more layers and back-end users for testing and verification, and generate a final optimization model by comprehensively comparing multiple models.

8.2 Visualize the data, result, models, and patterns

The goal of data visualization and pattern discovery is to reveal relationships between the house features and the response variable, total price. The study in this part is try to identify which house features that can affect total price and could be potential predictors. According to the visualization, the following information can be collected from the data.

- Using box chart to illustrate basic information from data. From the following chart, it shows a normal distribution with a long tail, which is exactly what we would expect for data of this kind. The most expensive house cost $18,100 million and there are a total of 151,260 houses sold for more around $300 million.

```
#visualization 3
df_hp.boxplot(column='totalPrice')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1a3cfa4f28>
```

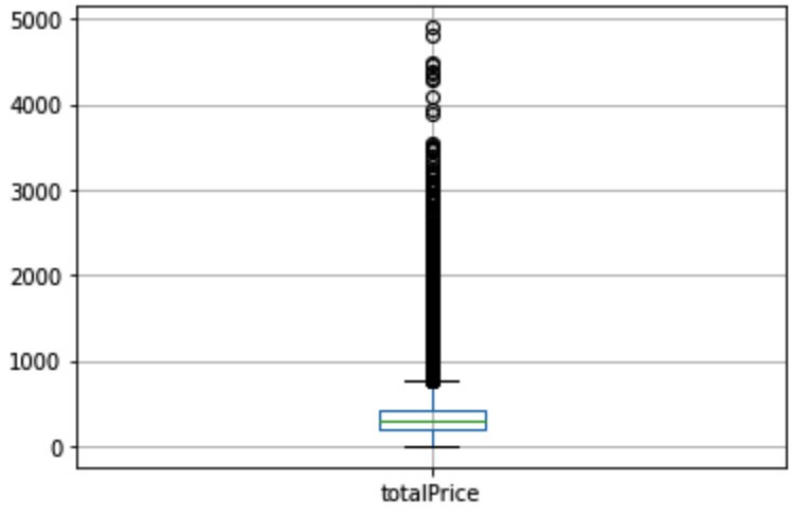


Figure 52: Box chart of total price

- In this dataset, we have latitude and longtitude information for the houses. By using lat and long columns, we can displayed the below heat map which is very useful for the people who does not know Seattle well.

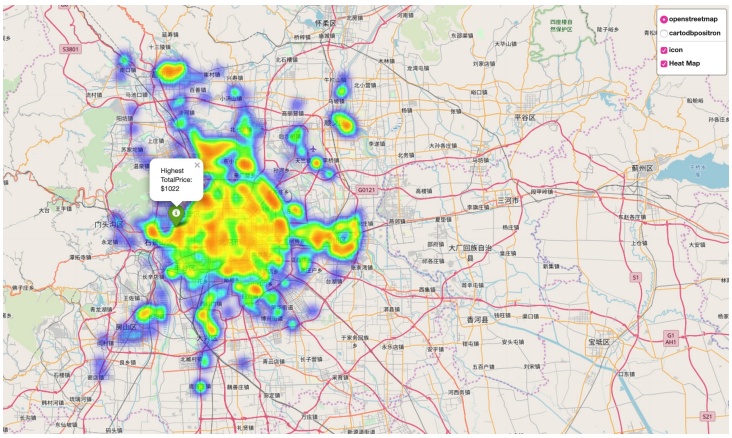


Figure 53: Heat map

- From the histogram matrix, we can infer that house price increases with increase in Square and communityAverage roughly.

Figure 54: Scatterplot Matrix

- Total Price Vs Renovation Level:
  As mentioned before, 2 represents the house has not been renovated and 3 represents the house has been renovated simply, ”4” represents house has been renovated very well. From the below column chart, it can be observed that the house price associated with better renovated houses is higher than the house price associated with non-renovated and simple renovated house.

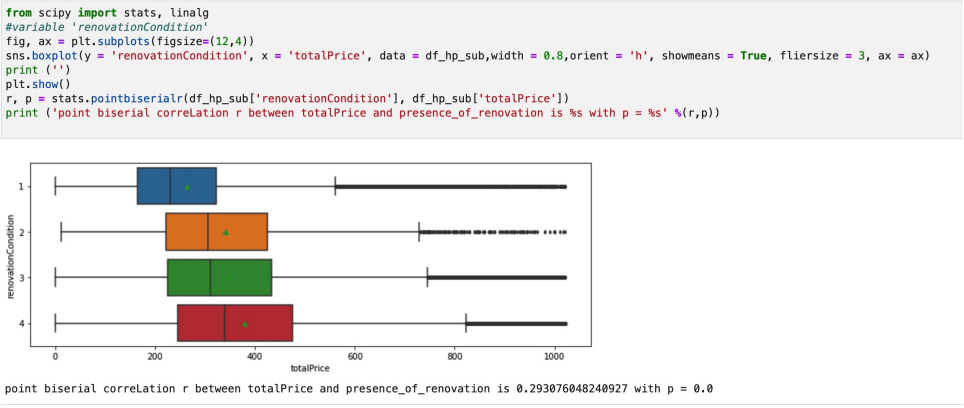


Figure 55: Total Price Vs Renovation Level

- Price Vs Square, Colored by Number of living room:
  The below scatterplot is indicative of house price increase with increase in square and the number of living rooms. Home prices below 500 million dollars have less than 2 living rooms most likely and less square in total of the house. Also, the house with more than 4 living rooms are the most expensive house in the market.

season

```
: # Use the 'hue' argument to provide a factor variable
sns.lmplot( x="totalPrice", y="square", data=df_hp_sub, fit_reg=False, hue='livingRoom', legend=False)

# Move the legend to an empty part of the plot
plt.legend(loc='lower right')
```

```
: <matplotlib.legend.Legend at 0x1a26dfbb70>
```

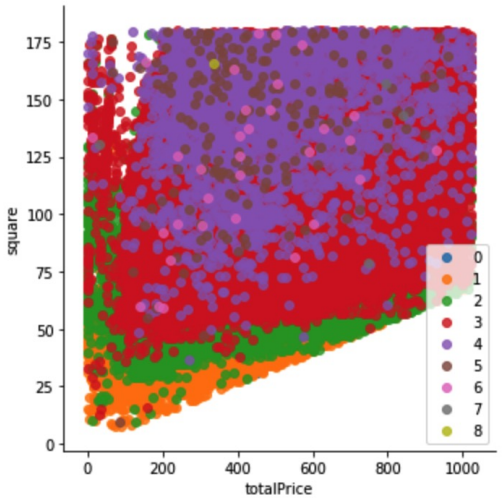


Figure 56: Total Price Vs Square with living room

- Mean (Price) Vs seasons:
  We create a histogram graph to show the house price in four seasons. It shows that the mean of house price is usually higher in winter and summer.

```
sort = df_hp_sub.sort_values(by='totalPrice', ascending=False)
sort.groupby(['season'])['totalPrice'].mean().plot.bar()
plt.title('Mean (Price) house price over season')
plt.xlabel('season')
plt.ylabel('price')
```

```
Text(0, 0.5, 'price')
```

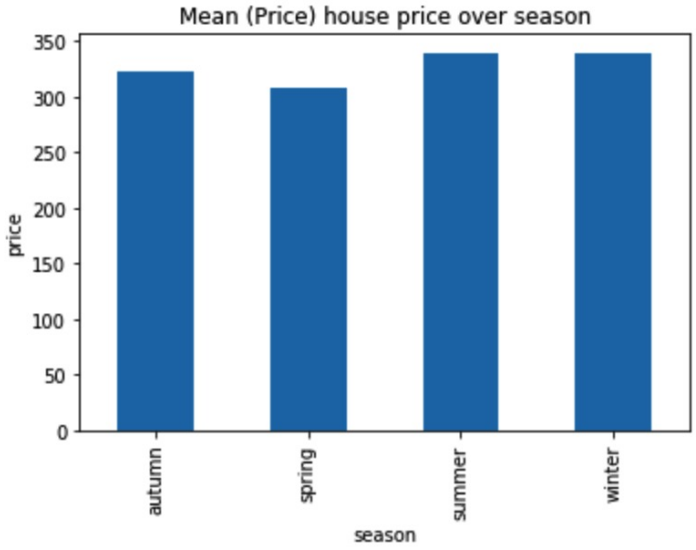


Figure 57: Season plot

- In order to valid the house price in winter is higher than that in other seasons, this study finds the worst 1000 data of house price in Beijing. From this graph, we can see that the highest 1000 data distributing in winter most likely.

```
sort = df_hp_sub.sort_values(by='totalPrice', ascending=False).head(100)
sort.groupby(['season'])['totalPrice'].count().plot.bar()
plt.title('count of top 100 highest house price over season')
plt.xlabel('season')
plt.ylabel('count')
```

```
Text(0, 0.5, 'count')
```

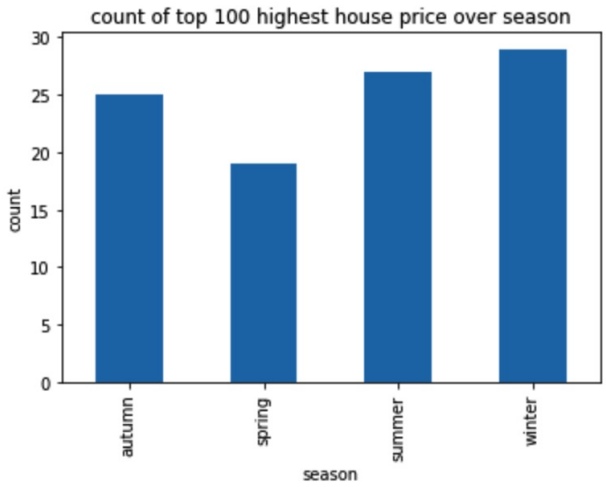


Figure 58: seasons

- Mean (Price) Vs subway:
  As mentioned before, 1 means the house has subway nearby and 0 means not. From the below column chart, it can be observed that the average house price with houses which had subway nearby is higher than the average house price associated with no subway nearby.

```
# Use the 'hue' argument to provide a factor variable
sns.lmplot( x="subway", y="totalPrice", data=df_hp_sub, fit_reg=False, legend=False)

# Move the legend to an empty part of the plot
plt.legend(loc='lower right')


sort = df_hp_sub.sort_values(by='totalPrice', ascending=False)
sort.groupby(['subway'])['totalPrice'].count().plot.bar()
plt.title('count of Mean (TotalPrice) house totalPrice over subway')
plt.xlabel('subway')
plt.ylabel('count')
```

```
No handles with labels found to put in legend.
Text(-2.450000000000003, 0.5, 'count')
```

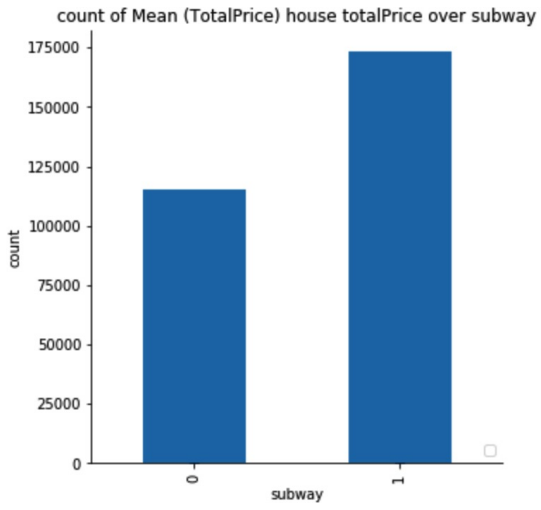


Figure 59: Subway

- Latitude Vs Longitude: We observe that the predicted price will increase with increase in latitude as the location moves from South to North and will decrease with increase in longitude as the location moves from West to East.

totalPrice

```
plt.scatter(df_hp_sub.Lat, df_hp_sub.totalPrice)
plt.title("Latitude va TotalPrice")
plt.xlabel('Latitude')
plt.ylabel('TotalPrice')
plt.show()
sns.despine
```

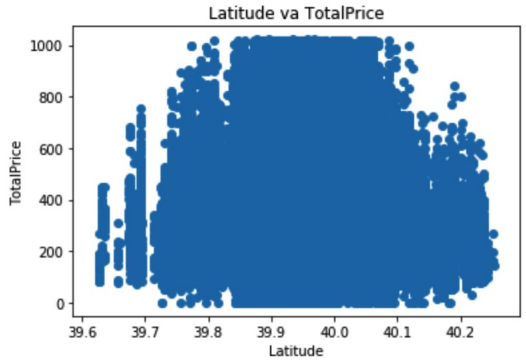


Figure 60: Latitude Vs Price

```
plt.scatter(df_hp_sub.Lng, df_hp_sub.totalPrice)
plt.title("Longtitude va Price")
plt.xlabel('Longtitude')
plt.ylabel('Price')
plt.show()
sns.despine
```

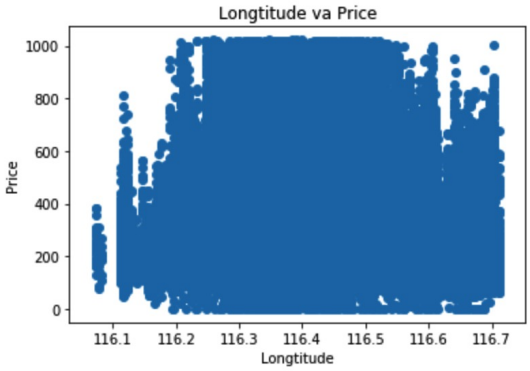


Figure 61: Longitude Vs Price

- The chart from regression algorithms shows the correlation of different variables with house price.

```
features = ['totalPrice','Lng','Lat','Cid','tradeYear','tradeMonth',
            'tradeDay','DOM','followers','square','livingRoom',
      'drawingRoom','bathRoom','floor','buildingType','constructionTime','renovationCondition',
            'buildingStructure','buildingType','ladderRatio','elevator','fiveYearsProperty','subway','district','communityAverage']

mask = np.zeros_like(df_hp_sub[features].corr(), dtype=np.bool)
mask[np.triu_indices_from(mask)] = True

f, ax = plt.subplots(figsize=(16, 12))
plt.title('Pearson CorreLation Matrix',fontsize=25)

sns.heatmap(df_hp_sub[features].corr(),linewidths=0.25,vmax=0.7,square=True,cmap="BuGn", #"BuGn_r" to reverse
            linecolor='w',annot=True,annot_kws={"size":8},mask=mask,cbar_kws={"shrink": .9});
```

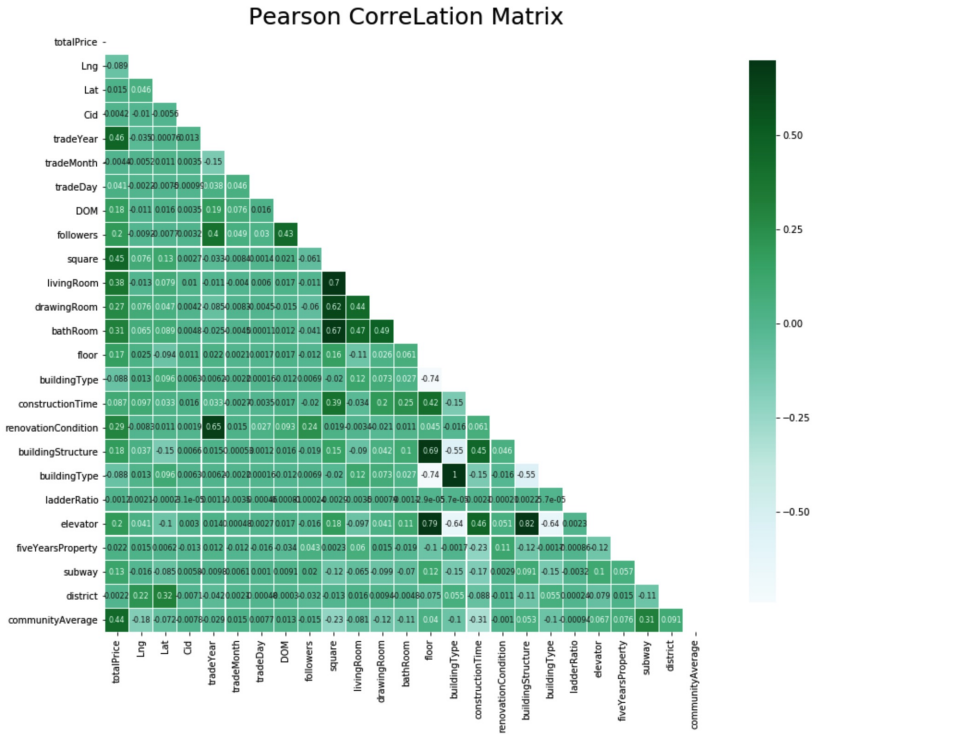


Figure 62: Correlation of different variables with house price

8.3 Interpret the results, models, and patterns

8.3.1 Interpret from Visualization Graphs

From above visualization graphs, we can get conclusions as follow:

- Price increases with increase in Square, the number of living room, and the number of drawing, DOM, tradeYear.
- Price increase with increase in ladder ratio, building structure.
- Renovated houses are likely to have a higher price compared to Non-Renovated houses.

- Houses with subway nearby are likely to have a higher price compared to houses with no subway nearby.
- Houses with more than four living rooms are likely to have a higher price compared to houses less than three living rooms.
- The house price in winter is higher than the other seasons.

8.3.2 Interpret from Pearson Correlation Matric

- square,communityAverage, and tradeYear are moderately correlated with price.
- There is strong correlation between elevator and floor.
- There is strong correlation between elevator and buildingStructure.
- There are strong correlation between the three room variables (livingRoom, drawingRoom, bathRoom) and square.

8.3.3 Interpret from Models

- **Simple Linear Regression coefficient:**
  When we look into the attributes of the dataset, square (living area) seems the most important feature. When we examine the correlation matrix, we may observe that total price has the highest correlation coefficient with square (living area), so this also supports my opinion in previous chapter. Thus, square (living area) should be used as feature. However, if we want to examine the relationship between price and another feature, we can change to use that feature. After fit the features into the model, we can see the coefficient of square and total price using the simple linear regression.

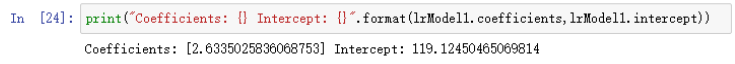


Figure 63: Simple Linear Regression coefficient

- **Multiple Linear Regression coefficient:**
  From the graph above, we can get the conclusion as followed:
  - Most house increase in Square, and the number of living room will promote the predicted price.

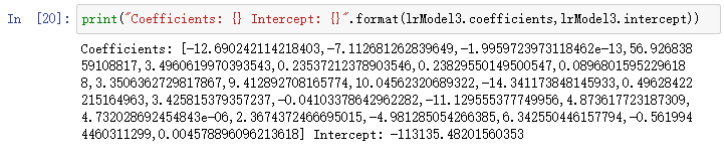


Figure 64: Multiple Linear Regression coefficient

- The predicted price will increase with increase in the number of drawing, square, ladder ration.
- The predicted price will increase with location changed. The house price will increase if the location close to the centre city area. we can see this trend from the Heat Map on above.
- Predicted price is high for houses with subway.
- Predicted price is high in the winter and Autumn because of seasonality impact.
- Predicted price increases with better renovation condition.

- **Random Forest Regression coefficient:**

- Random Forest Regression model can not show the exact coefficients like most of linear regressions, so we leave it over here.

8.3.4 Interpret from Patterns

After comparing various data mining algorithms, the performance of Multiple Linear Regression is better than others. Using Multiple Linear Regression can predict the house price in some extent. The accuracy of Multiple Linear Regression is 78%, which is higher than other algorithms.

8.4 Assess and evaluate results, models, and patterns

Regression analysis is a statistical analysis method to determine the quantitative relationship between two or more than two variables. In this study, regression algorithm finds the important factors for house price successfully.

- **Model Evaluation**
  After comparing various regression algorithms, the performance of multiple regression is better than other regression methods. Using the regression algorithms can predict the house price in some extent.

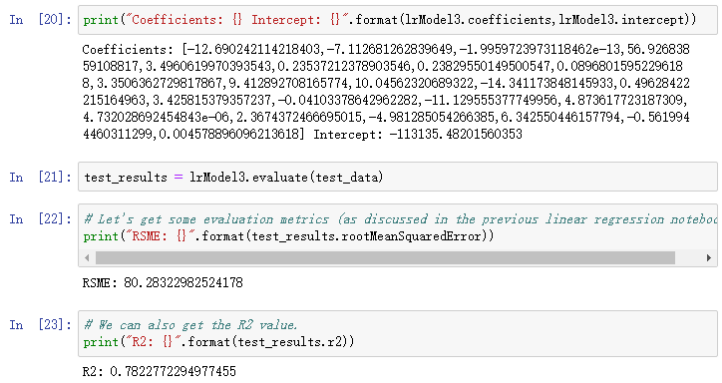


Figure 65: Results of Multiple Linear Regression

- **Results Evaluation**

  - Price increases with increase in Square, the number of living room, and the number of drawing, DOM, tradeYear.
  - Price increase with increase in ladder ratio, building structure.
  - Renovated houses are likely to have a higher price compared to Non-Renovated houses.
  - Houses with subway nearby are likely to have a higher price compared to houses with no subway nearby.
  - Houses with more than four living rooms are likely to have a higher price compared to houses less than three living rooms.

- **Pattern Evaluation**

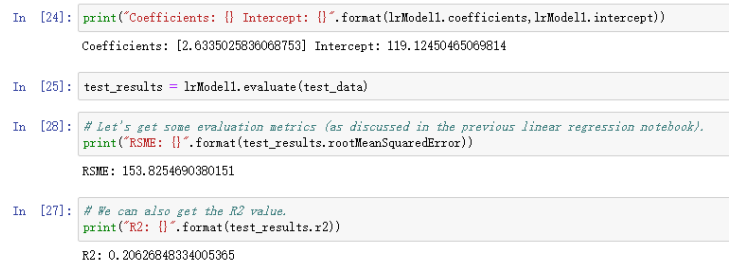


Figure 66: Simple Linear Regression Model Result

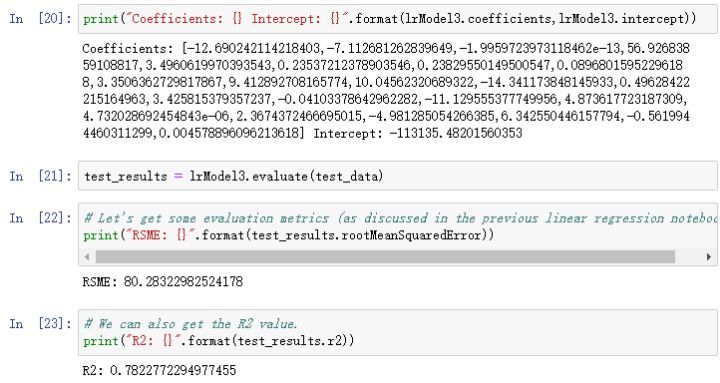


Figure 67: All Features Multiple Linear Regression Model Result

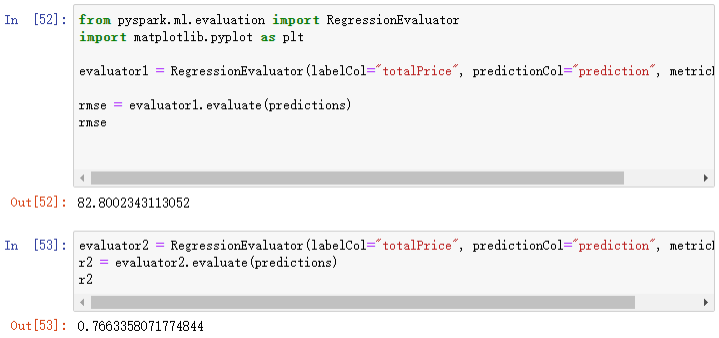


Figure 68: Random Forest Model Result

8.5 Iterate prior steps (1-7) as required

8.5.1 Iteration for Step 1 - Business Understanding

In step one, we understand the situation of house price in Beijing and set the objectives. We had a hypothesis we assume that it is possible to predict the sale price of a house from information about that house such as the size, number of bedrooms, condition etc. Meanwhile, we aim to understand the market and social background, and get the basic ability to comprehend the situation and the background of the house price.This helps us to find out the objective of this study. Our objective is to figure out the relationship between house price and attributes in the dataset and predict the trend of house price. The conclusion could be used for government build more affordable house to the poor people. Once the object is determined, the project plan with a time table will help us organize the work with the schedule.

8.5.2 Iteration for Step 2 - Data Understanding

In step two, this study researches the data initially. We describe, explore and analyse the data. To search for the available data online and collected the data from an open source platform named Kaggle. Then, we go to understand the content of the dataset, such as the attributes meaning, records quantity, etc. Moreover, we should verified the data quality and find that if there are missing values, outliers and extreme values which need be processed in the later steps.

8.5.3 Iteration for Step 3 - Data Preparation

In step three, data was prepared including selection, cleaning, construct and formation. In this study, we use different way to clean data, including discard both outliers and extreme values from some attributes, coerce the missing value in DOM attribute and fill the missing values with the mean of the the feature.

8.5.4 Iteration for Step 4 - Data Transformation

In step four, using different methods which provides in the Python and Pandas library to reduce and project the data. Projecting the dataset for the purpose of next steps.I found house price may have a regularity based on seasons. Also, we found house price may have a relationship with different seasons.

8.5.5 Iteration for Step 5 - Data Mining Methods Selection

In step five, after discussion the different methods, this study choose to use multiple regression methods to mining data due to the feature of the dataset.

8.5.6 Iteration for Step 6 - Data Mining Algorithms Selection

In step six, the study finds various suitable models and algorithms by discussion. In this step, we iterated different models and performed an exploratory analysis to see which algorithm is more suitable for my analysis. After discussion, we choose simple linear regression, multiple linear regression, and random forest regression in this research. Then Set the simple linear regression as the benchmark, because it is easy and simple to interpret the relation between house price and response variable which is one of the two mining goals(explanation). After, we use both multiple linear regression and random forest regression to fit the new models, because they may have the better prediction accuracy which is another mining goal (prediction). After choosing the algorithms, we set the algorithms parameters and ready to move forward to the next step.

8.5.7 Iteration for Step 7 - Data Mining

In step seven, this study finds various suitable models and algorithms by discussion. In this step, we subset the data as training data and test date as 80/20 from the dataset, and explain the reason of this subsection with supervised learning. Then, the project created the testing method for model, conducts data mining and searches correct patterns. In this step, we iterated at least three method to search a correct pattern after conducting data mining with different parameters, and data mining is conducted step by step. Finally, we got all the patterns for further interpretation of the output.we iterate the different parameters to show which one is better. As a result, we got the best parameters within certain range.

9 Actions

This project focuses on house price, finding the relationship between house price and correlative factors, and searching the trend of house price in Beijing. This study can help policymaker do the preparation based on analysis. However, its forecast the house price and the price trend of house based on historical data and specific attributes like particular location(community id), which has some limitation. Further research should focus on the cause of house price increase or reduce. After combining the effects from these more aspects, house price can be predicted more accurate.

Disclaimer

”I acknowledge that the submitted work is my own original work in accordance with the University of Auckland guidelines and policies on academic integrity and copyright. (See: https://www.auckland.ac.nz/en/students/forms-policies-and-guidelines/student-policies-and-guidelines/academic-integrity-copyright.html).

I also acknowledge that I have appropriate permission to use the data that I have utilised in this project. (For example, if the data belongs to an organisation and the data has not been published in the public domain then the data must be approved by the rights holder.) This includes permission to upload the data file to Canvas. The University of Auckland bears no responsibility for the student’s misuse of data.”

References

[1] Paul D Allison. *Multiple regression: A primer*. Pine Forge Press, 1999. pages 38

[2] A Colin Cameron and Frank AG Windmeijer. An r-squared measure of goodness of fit for some common nonlinear regression models. *Journal of econometrics*, 77(2):329–342, 1997. pages 44

[3] IBM Knowledge Center, 2019. pages 38

[4] Eloisa T Glindro, Tientip Subhanij, Jessica Szeto, Haibin Zhu, et al. Determinants of house prices in nine asia-pacific economies. *International Journal of Central Banking*, 7(3):163–204, 2011. pages 4

[5] Chih-Wei Hsu, Chih-Chung Chang, Chih-Jen Lin, et al. A practical guide to support vector classification. 2003. pages 45

[6] Han Liu and Mihaela Cocea. Semi-random partitioning of data into training and test sets in granular computing context. *Granular Computing*, 2(4):357–386, 2017. pages 45

[7] Yu Ren, Cong Xiong, and Yufei Yuan. House price bubbles in china. *China Economic Review*, 23(4):786–800, 2012. pages 4

[8] Shan Suthaharan. Machine learning models and algorithms for big data classification. *Integr. Ser. Inf. Syst*, 36:1–12, 2016. pages 35, 36

[9] Jack V Tu. Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes. *Journal of clinical epidemiology*, 49(11):1225–1231, 1996. pages 37