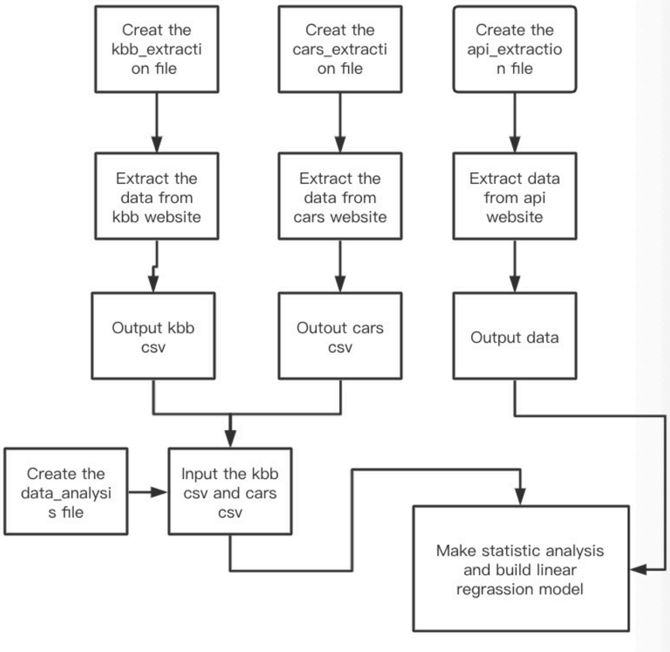
Name: Di Jin    Student number: 6475336873

# Final Project

## Section 1：

I want to get 1800 cars' price, mileage, year and other information from three well-known websites selling used cars in the United States. This project studies the influence of vehicle mileage on vehicle price and the influence of vehicle year on vehicle price, This project also studies the comprehensive influence of mileage and year on vehicle price. The linear regression algorithm is used to build the prediction model. I study the features of the most popular cars on the website, the color distribution of vehicles, and count the most and least colors on the website. This information can provide reference value for customers when they buy a car.

## Section 2:

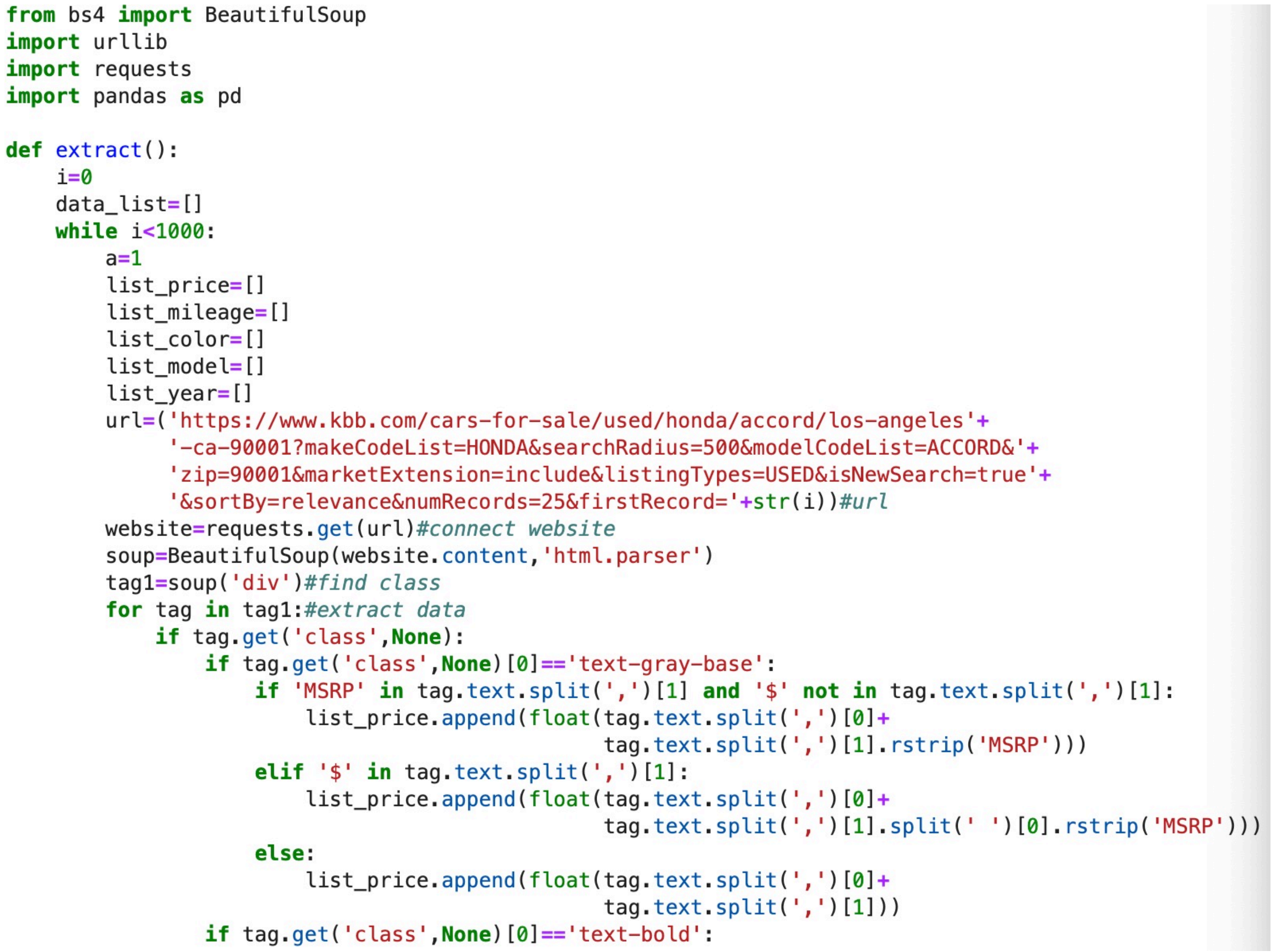

Creat the kbb_extraction file
Creat the cars_extraction file
Create the api_extraction file
Extract the data from kbb website
Extract the data from cars website
Extract data from api website
Output kbb csv
Outout cars csv
Output data
Create the data_analysis file
Input the kbb csv and cars csv
Make statistic analysis and build linear regrassion model

My code is divided into four files. the first file's name is 'kbb_extraction'. The main function of this file is to extract the data of 1000 Honda used cars. The output is a CSV file and the CSV file will be stored in the same place with the 'data_analysis' file. You can successfully import the data in the 'data_analysis' file. The second file's name is 'cars_extraction'. The main function of this file is to extract the data of 800 BMW used cars. The output of a CSV file and the CSV file will be stored in the same place with the 'data_analysis' file. You can successfully import the data in the 'data_analysis' file. The third file's name is 'API_extraction'. It is to use the API to get the price of the new Honda series. The fourth file's name is ‘data_analysis’. The main function of this file is to import the two CSV files and make statistical analysis and establish a linear regression model.

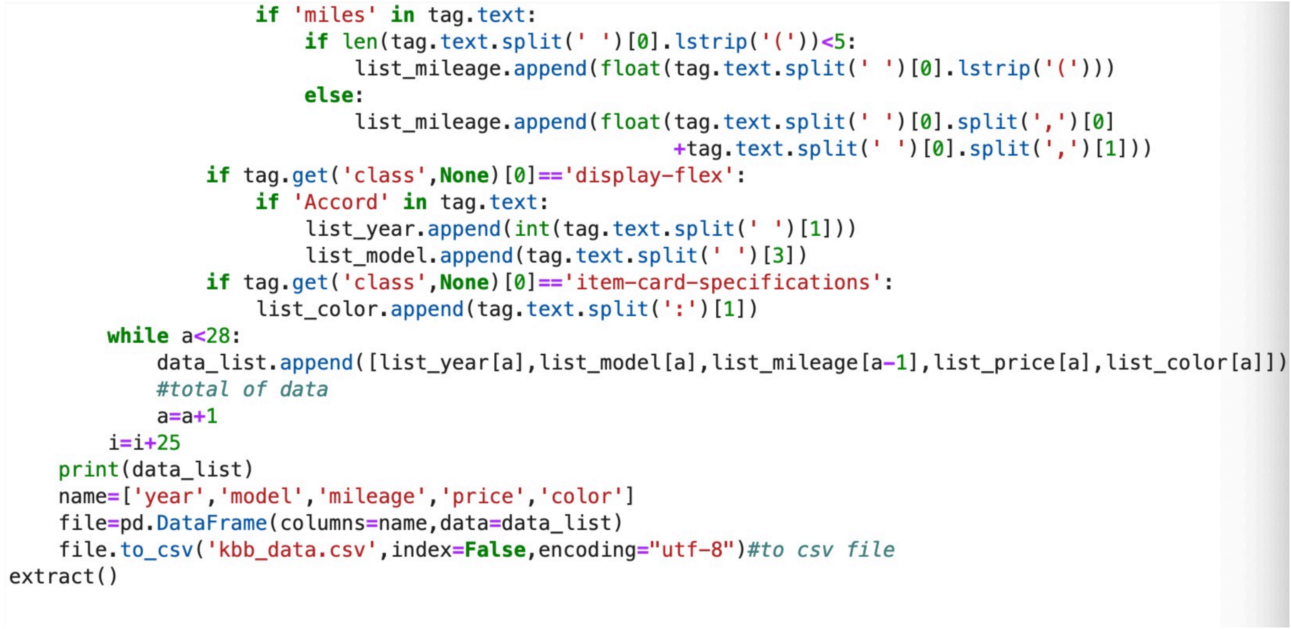
**Section 3:**

a) kbb_extraction.py

For the kbb website, there are only 25 vehicles on each page, so I design a loop to process the URL. I change the page number displayed in the URL to the variable i. The variable i will increase every time in the loop, so the data of each page can be automatically extracted. When extracting the data of each page, I create five empty lists (list_price, list _mile, list_color, list_model, list_year) to store the data in order. And then, I connect to the website with the method of requests.get. I use the method of beautiful soup to get the HTML of this website and find that the data is in the class' div '. Each type of data has a different class name, so I extract these data by looking up different class names and save these data in the empty lists I create. I process for the formal of each type of data, and I change the digital data to int or float. Finally, I used a loop to combine each data in the five lists in order and extract it into a new list. The storage type is a large list containing many small lists. Finally, I used PD. Dataframe to convert these data into CSV file.



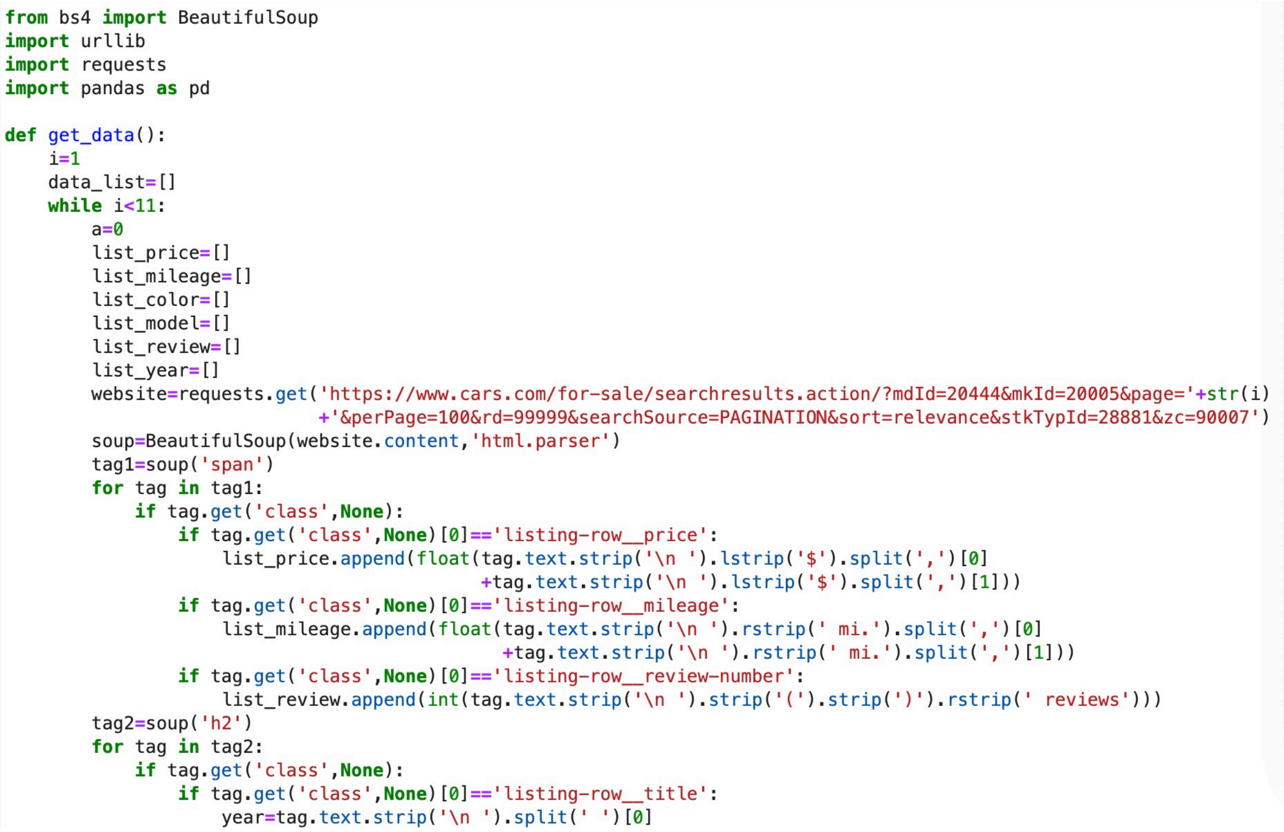
```python
from bs4 import BeautifulSoup
import urllib
import requests
import pandas as pd

def extract():
    i=0
    data_list=[]
    while i<1000:
        a=1
        list_price=[]
        list_mileage=[]
        list_color=[]
        list_model=[]
        list_year=[]
        url=('https://www.kbb.com/cars-for-sale/used/honda/accord/los-angeles'+
             '-ca-90001?makeCodeList=HONDA&searchRadius=500&modelCodeList=ACCORD&'+
             'zip=90001&marketExtension=include&listingTypes=USED&isNewSearch=true'+
             '&sortBy=relevance&numRecords=25&firstRecord='+str(i))#url
        website=requests.get(url)#connect website
        soup=BeautifulSoup(website.content,'html.parser')
        tag1=soup('div')#find class
        for tag in tag1:#extract data
            if tag.get('class',None):
                if tag.get('class',None)[0]=='text-gray-base':
                    if 'MSRP' in tag.text.split(',')[1] and '$' not in tag.text.split(',')[1]:
                        list_price.append(float(tag.text.split(',')[0]+
                                                tag.text.split(',')[1].rstrip('MSRP')))
                    elif '$' in tag.text.split(',')[1]:
                        list_price.append(float(tag.text.split(',')[0]+
                                                tag.text.split(',')[1].split(' ')[0].rstrip('MSRP')))
                    else:
                        list_price.append(float(tag.text.split(',')[0]+
                                                tag.text.split(',')[1]))
                if tag.get('class',None)[0]=='text-bold':
```



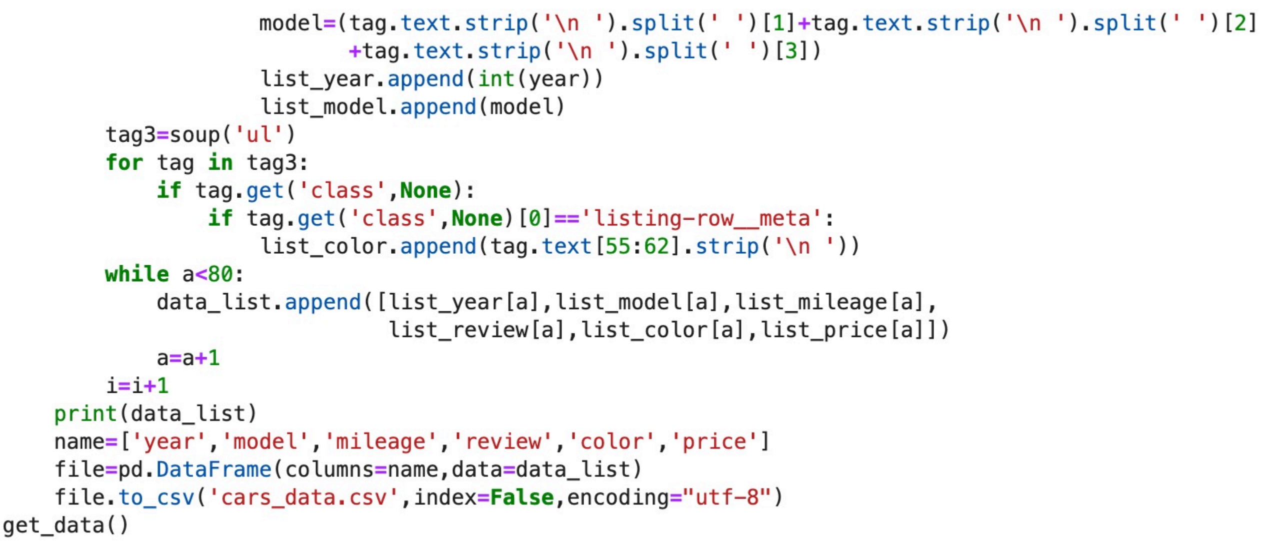
```python
                    if 'miles' in tag.text:
                        if len(tag.text.split(' ')[0].lstrip('('))<5:
                            list_mileage.append(float(tag.text.split(' ')[0].lstrip('(')))
                        else:
                            list_mileage.append(float(tag.text.split(' ')[0].split(',')[0]
                                                     +tag.text.split(' ')[0].split(',')[1]))
                if tag.get('class',None)[0]=='display-flex':
                    if 'Accord' in tag.text:
                        list_year.append(int(tag.text.split(' ')[1]))
                        list_model.append(tag.text.split(' ')[3])
                if tag.get('class',None)[0]=='item-card-specifications':
                    list_color.append(tag.text.split(':')[1])
        while a<28:
            data_list.append([list_year[a],list_model[a],list_mileage[a-1],list_price[a],list_color[a]])
            #total of data
            a=a+1
        i=i+25
    print(data_list)
    name=['year','model','mileage','price','color']
    file=pd.DataFrame(columns=name,data=data_list)
    file.to_csv('kbb_data.csv',index=False,encoding="utf-8")#to csv file
extract()
```

b) cars_extraction.py

For cars website, there are 100 cars on each page, so I take the same method as above to extract the data of 800 cars.

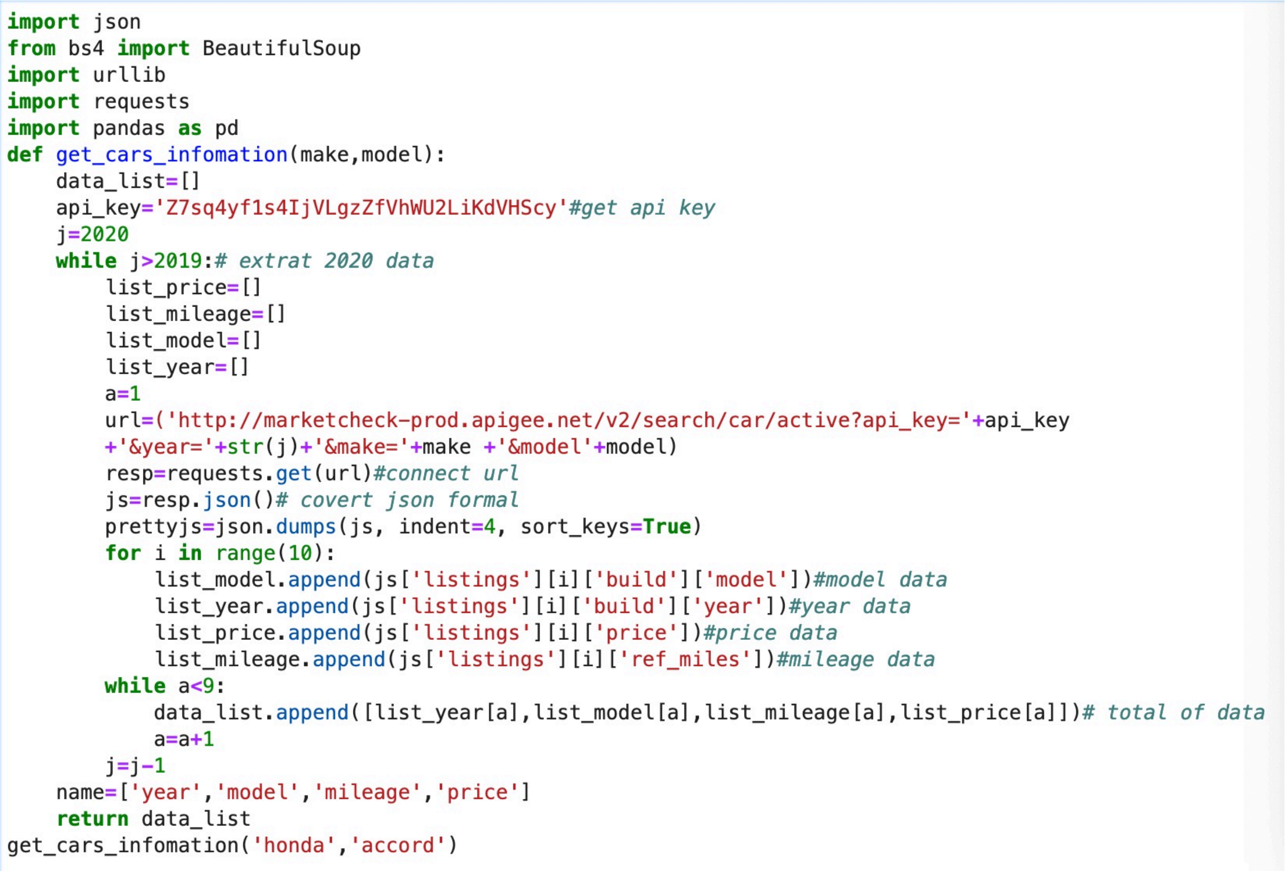


```
from bs4 import BeautifulSoup
import urllib
import requests
import pandas as pd

def get_data():
    i=1
    data_list=[]
    while i<11:
        a=0
        list_price=[]
        list_mileage=[]
        list_color=[]
        list_model=[]
        list_review=[]
        list_year=[]
        website=requests.get('https://www.cars.com/for-sale/searchresults.action/?mdId=20444&mkId=20005&page='+str(i)
                             +'&perPage=100&rd=99999&searchSource=PAGINATION&sort=relevance&stkTypId=28881&zc=90007')
        soup=BeautifulSoup(website.content,'html.parser')
        tag1=soup('span')
        for tag in tag1:
            if tag.get('class',None):
                if tag.get('class',None)[0]=='listing-row__price':
                    list_price.append(float(tag.text.strip('\n ').lstrip('$').split(',')[0]
                                            +tag.text.strip('\n ').lstrip('$').split(',')[1]))
                if tag.get('class',None)[0]=='listing-row__mileage':
                    list_mileage.append(float(tag.text.strip('\n ').rstrip(' mi.').split(',')[0]
                                              +tag.text.strip('\n ').rstrip(' mi.').split(',')[1]))
                if tag.get('class',None)[0]=='listing-row__review-number':
                    list_review.append(int(tag.text.strip('\n ').strip('(').strip(')').rstrip(' reviews')))
        tag2=soup('h2')
        for tag in tag2:
            if tag.get('class',None):
                if tag.get('class',None)[0]=='listing-row__title':
                    year=tag.text.strip('\n ').split(' ')[0]
```



```
                    model=(tag.text.strip('\n ').split(' ')[1]+tag.text.strip('\n ').split(' ')[2]
                           +tag.text.strip('\n ').split(' ')[3])
                    list_year.append(int(year))
                    list_model.append(model)
        tag3=soup('ul')
        for tag in tag3:
            if tag.get('class',None):
                if tag.get('class',None)[0]=='listing-row__meta':
                    list_color.append(tag.text[55:62].strip('\n '))
        while a<80:
            data_list.append([list_year[a],list_model[a],list_mileage[a],
                              list_review[a],list_color[a],list_price[a]])
            a=a+1
        i=i+1
    print(data_list)
    name=['year','model','mileage','review','color','price']
    file=pd.DataFrame(columns=name,data=data_list)
    file.to_csv('cars_data.csv',index=False,encoding="utf-8")
get_data()
```

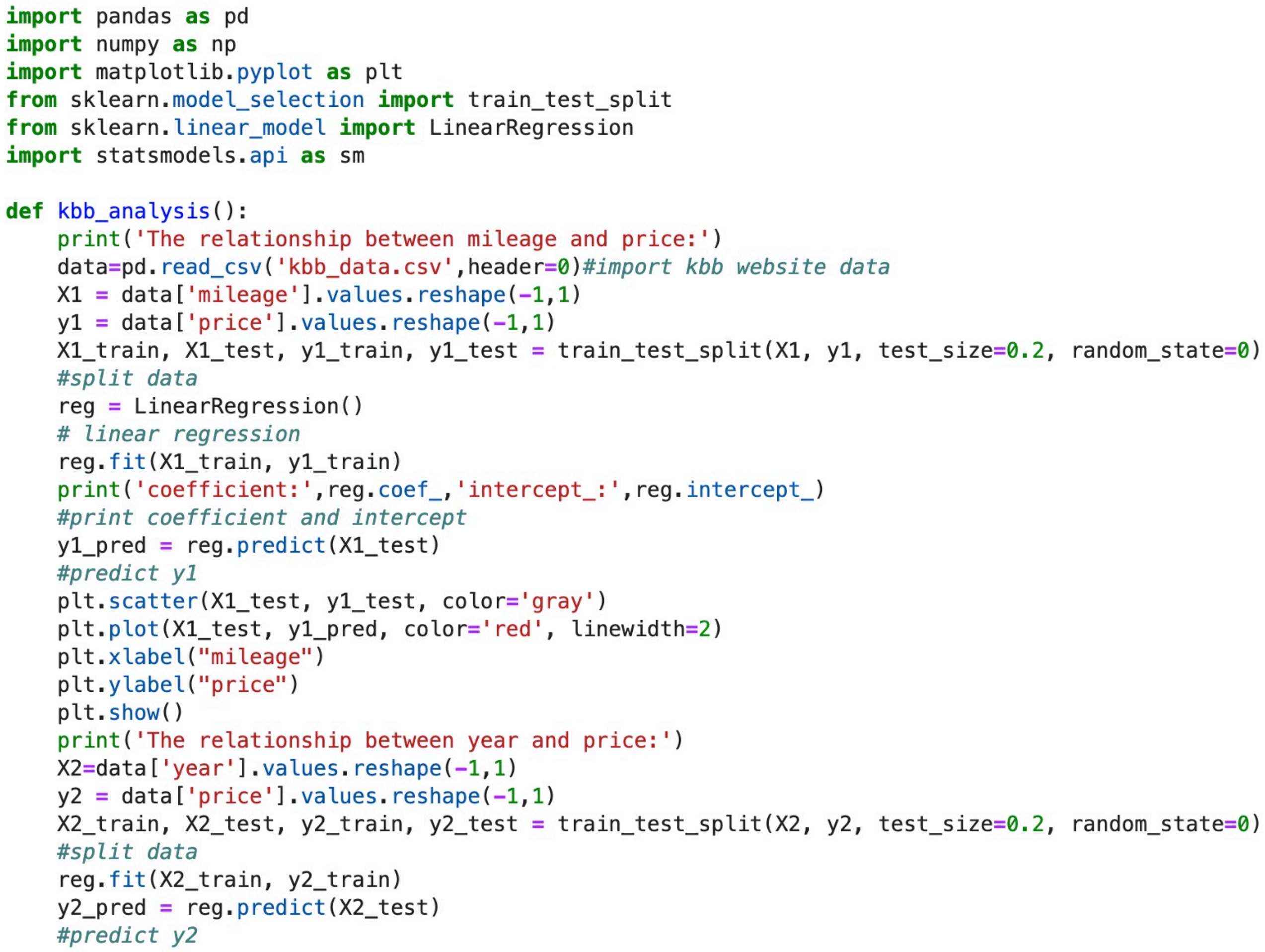
c) api_extraction.py



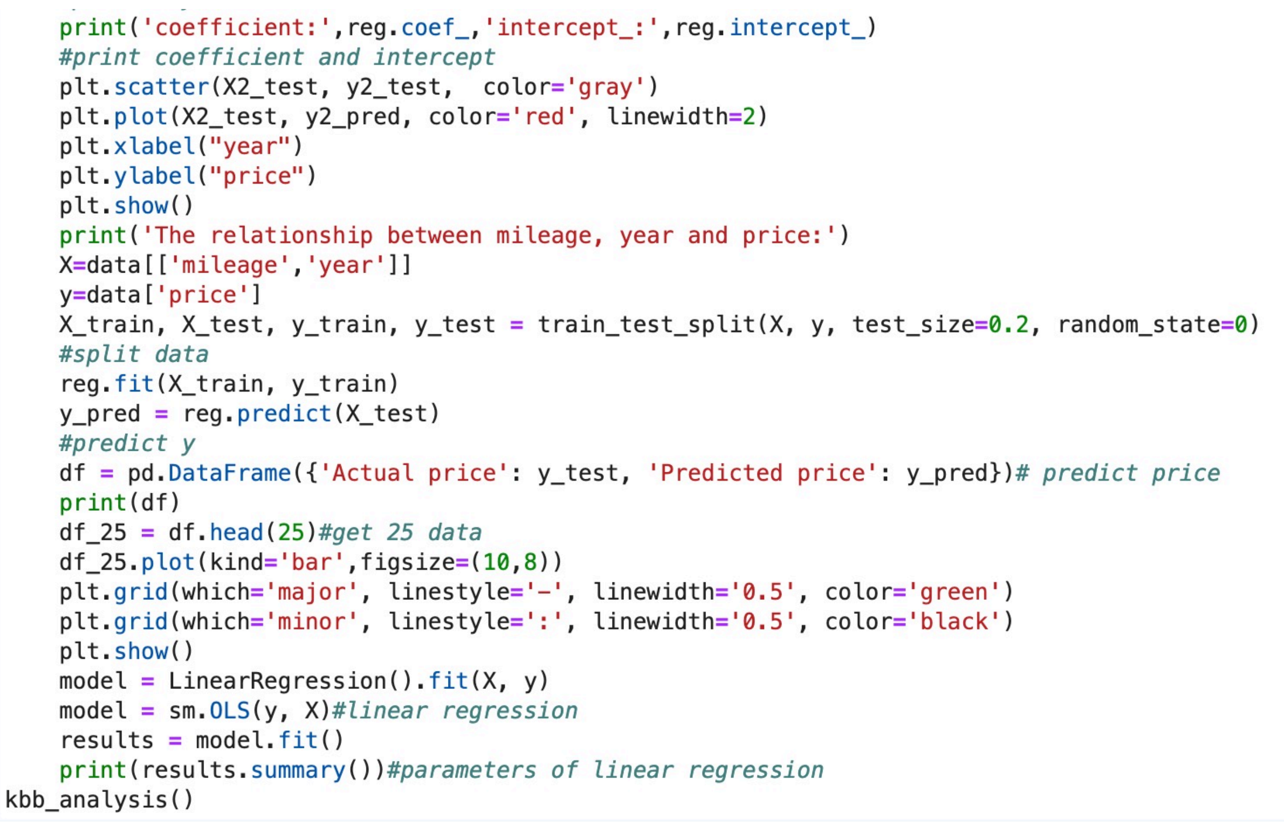
```
import json
from bs4 import BeautifulSoup
import urllib
import requests
import pandas as pd
def get_cars_infomation(make,model):
    data_list=[]
    api_key='Z7sq4yf1s4IjVLgzZfVhWU2LiKdVHScy'#get api key
    j=2020
    while j>2019:# extrat 2020 data
        list_price=[]
        list_mileage=[]
        list_model=[]
        list_year=[]
        a=1
        url=('http://marketcheck-prod.apigee.net/v2/search/car/active?api_key='+api_key
        +'&year='+str(j)+'&make='+make +'&model'+model)
        resp=requests.get(url)#connect url
        js=resp.json()# covert json formal
        prettyjs=json.dumps(js, indent=4, sort_keys=True)
        for i in range(10):
            list_model.append(js['listings'][i]['build']['model'])#model data
            list_year.append(js['listings'][i]['build']['year'])#year data
            list_price.append(js['listings'][i]['price'])#price data
            list_mileage.append(js['listings'][i]['ref_miles'])#mileage data
        while a<9:
            data_list.append([list_year[a],list_model[a],list_mileage[a],list_price[a]])# total of data
            a=a+1
        j=j-1
    name=['year','model','mileage','price']
    return data_list
get_cars_infomation('honda','accord')
```

Firstly, I apply for a free API key from the API website. Then I build four empty lists to store the data( list_price, list_mileage, list_model, list_year). I use the method of requests.get to connect with websites. And then I get the html and convert it into JSON format to help my data search. According to the different positions of each data, I take them out by using the method of the dictionary and store them in four lists. Finally, I used a loop to combine each data in the four lists in order and extract it into a new list. The storage type is a large list with many small lists.

d) data_analysis.py



```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
import statsmodels.api as sm

def kbb_analysis():
    print('The relationship between mileage and price:')
    data=pd.read_csv('kbb_data.csv',header=0)#import kbb website data
    X1 = data['mileage'].values.reshape(-1,1)
    y1 = data['price'].values.reshape(-1,1)
    X1_train, X1_test, y1_train, y1_test = train_test_split(X1, y1, test_size=0.2, random_state=0)
    #split data
    reg = LinearRegression()
    # linear regression
    reg.fit(X1_train, y1_train)
    print('coefficient:',reg.coef_,'intercept_:',reg.intercept_)
    #print coefficient and intercept
    y1_pred = reg.predict(X1_test)
    #predict y1
    plt.scatter(X1_test, y1_test, color='gray')
    plt.plot(X1_test, y1_pred, color='red', linewidth=2)
    plt.xlabel("mileage")
    plt.ylabel("price")
    plt.show()
    print('The relationship between year and price:')
    X2=data['year'].values.reshape(-1,1)
    y2 = data['price'].values.reshape(-1,1)
    X2_train, X2_test, y2_train, y2_test = train_test_split(X2, y2, test_size=0.2, random_state=0)
    #split data
    reg.fit(X2_train, y2_train)
    y2_pred = reg.predict(X2_test)
    #predict y2
```



```
    print('coefficient:',reg.coef_,'intercept_:',reg.intercept_)
    #print coefficient and intercept
    plt.scatter(X2_test, y2_test,  color='gray')
    plt.plot(X2_test, y2_pred, color='red', linewidth=2)
    plt.xlabel("year")
    plt.ylabel("price")
    plt.show()
    print('The relationship between mileage, year and price:')
    X=data[['mileage','year']]
    y=data['price']
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
    #split data
    reg.fit(X_train, y_train)
    y_pred = reg.predict(X_test)
    #predict y
    df = pd.DataFrame({'Actual price': y_test, 'Predicted price': y_pred})# predict price
    print(df)
    df_25 = df.head(25)#get 25 data
    df_25.plot(kind='bar',figsize=(10,8))
    plt.grid(which='major', linestyle='-', linewidth='0.5', color='green')
    plt.grid(which='minor', linestyle=':', linewidth='0.5', color='black')
    plt.show()
    model = LinearRegression().fit(X, y)
    model = sm.OLS(y, X)#linear regression
    results = model.fit()
    print(results.summary())#parameters of linear regression
kbb_analysis()
```

Firstly, the data in CSV format is imported into python by read_CSV method. The relationship between vehicle mileage and price should be studied. So the mileage data is given to X1 and the price data is given to y1. I use the method of train_test_split to divide the data into training data and test data. And I use the package of linear regression of sk-learn to get the function slope, intercept and drawing. Secondly, I study the relationship between vehicle year and price. So I give the mileage data to X2 and the price data to y2. I use the method of train_test_split to divide the data into training data and test data. I use the package of the linear regression of sk-learn to get the function slope, intercept and drawing. Thirdly,I study the comprehensive relationship of vehicle year, mileage and price. So I give the mileage and year data to X and the price data to y. I use the method of train_test_split to divide the data into training data and test data. I use sklearn's linear regression package so, I can use the predict function to generate the forecast price and draw a picture. The parameters form of linear regression are generated by the package of OLS.

**Section4:**

In the data_analysis file, I use two packages, linear regression package from sk-learn, which we don't learn in class. It can perform linear regression on the data. The other one is statsmodels.api, which is used to get all the parameters of linear regression.

**Section 5:**

step 1: cd Desktop/final_6475336873

step 2: pip install -r requirements.txt

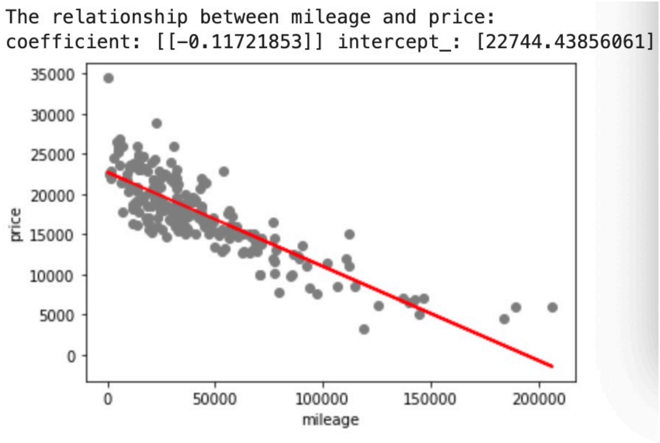
step 3: cd code

step 4: input python3 kbb_extraction.py , python3 cars_extraction.py, python3 api_extraction.py , python3 data_analysis.py

**Section 6:**

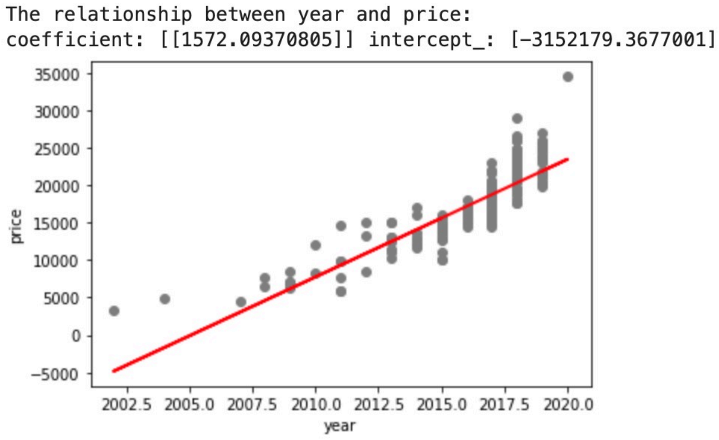
For Honda Accord:

From this graph, it can be seen that the mileage is inversely proportional to the price through univariate linear regression.



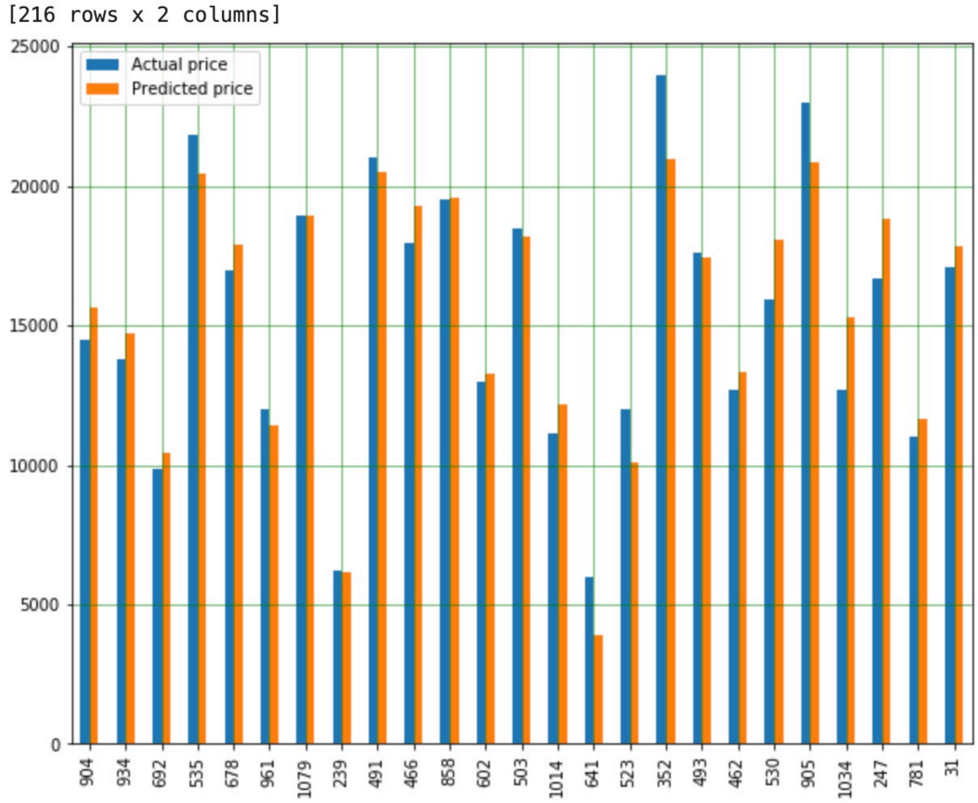
```
The relationship between mileage and price:
coefficient: [[-0.11721853]] intercept_: [22744.43856061]
```

From this graph, we can see that the year is directly proportional to the price through univariate linear regression.



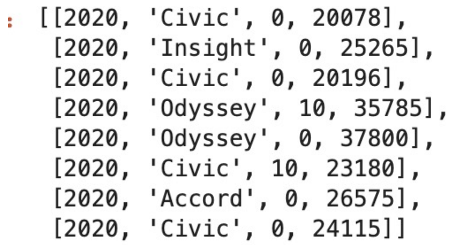
```
The relationship between year and price:
coefficient: [[1572.09370805]] intercept_: [-3152179.3677001]
```

Through multiple linear regression, 25 second-hand cars are randomly selected from the sample to predict their prices. From this figure, it can be seen that the price predicted by using the model is probably accurate with little error.



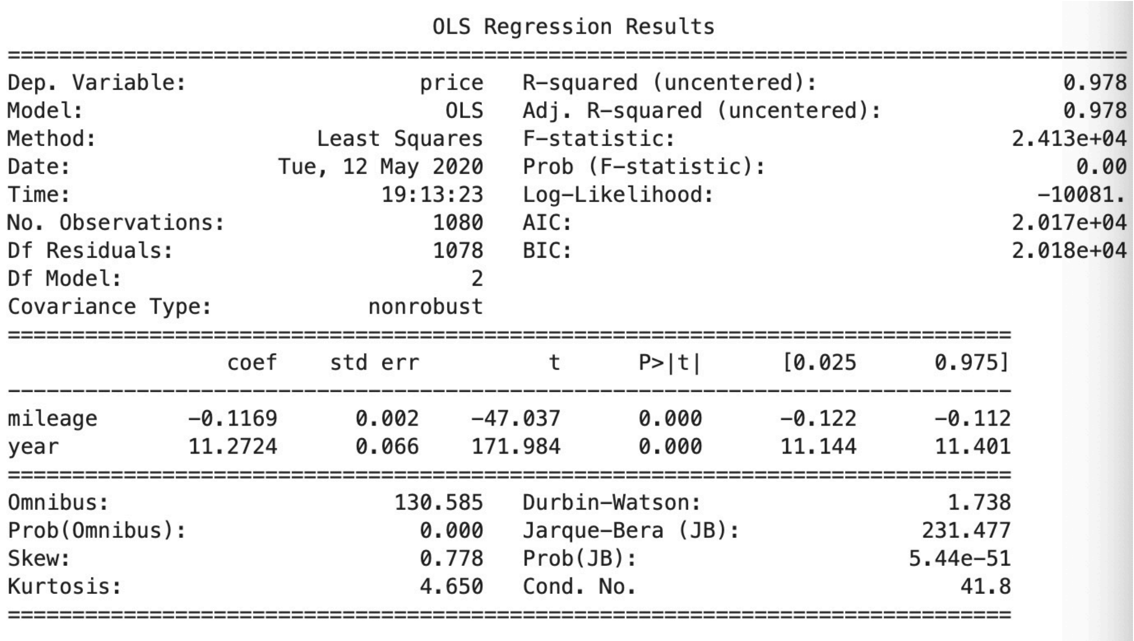
```
[216 rows x 2 columns]
```

I use api to search the new Honda Accord’ price is 26575. Therefore, no cars exceed 25000 in the picture.



```
: [[2020, 'Civic', 0, 20078],
   [2020, 'Insight', 0, 25265],
   [2020, 'Civic', 0, 20196],
   [2020, 'Odyssey', 10, 35785],
   [2020, 'Odyssey', 0, 37800],
   [2020, 'Civic', 10, 23180],
   [2020, 'Accord', 0, 26575],
   [2020, 'Civic', 0, 24115]]
```

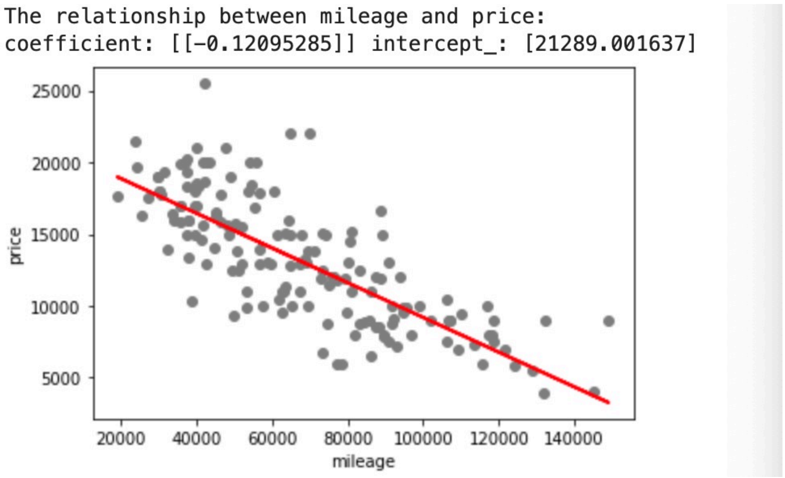
Through the parameter analysis of multiple linear regression, the square of R is 0.978, which shows that the model fits well.



```
                                 OLS Regression Results
=======================================================================================
Dep. Variable:                  price   R-squared (uncentered):                   0.978
Model:                            OLS   Adj. R-squared (uncentered):              0.978
Method:                 Least Squares   F-statistic:                          2.413e+04
Date:                Tue, 12 May 2020   Prob (F-statistic):                        0.00
Time:                        19:13:23   Log-Likelihood:                         -10081.
No. Observations:                1080   AIC:                                  2.017e+04
Df Residuals:                    1078   BIC:                                  2.018e+04
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
mileage       -0.1169      0.002    -47.037      0.000      -0.122      -0.112
year          11.2724      0.066    171.984      0.000      11.144      11.401
==============================================================================
Omnibus:                      130.585   Durbin-Watson:                   1.738
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              231.477
Skew:                           0.778   Prob(JB):                     5.44e-51
Kurtosis:                       4.650   Cond. No.                         41.8
==============================================================================
```

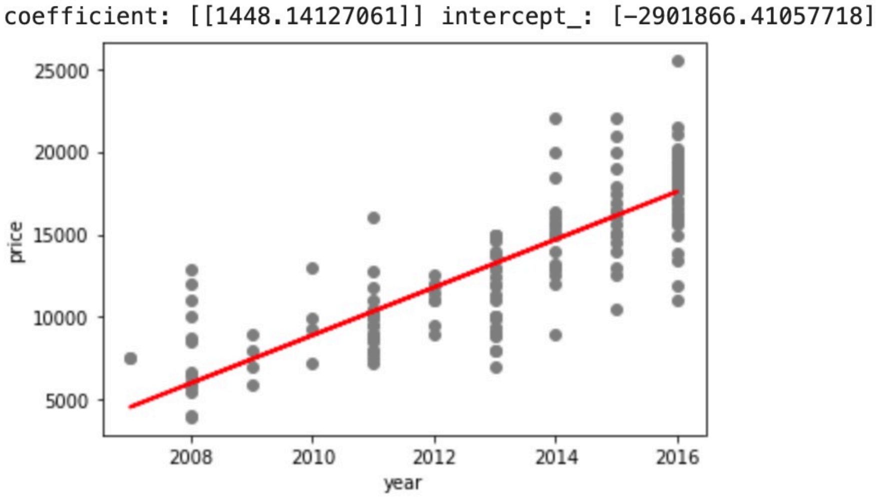
For BMW328i:

From this graph, we can see that the mileage is inversely proportional to the price.



```
The relationship between mileage and price:
coefficient: [[-0.12095285]] intercept_: [21289.001637]
```

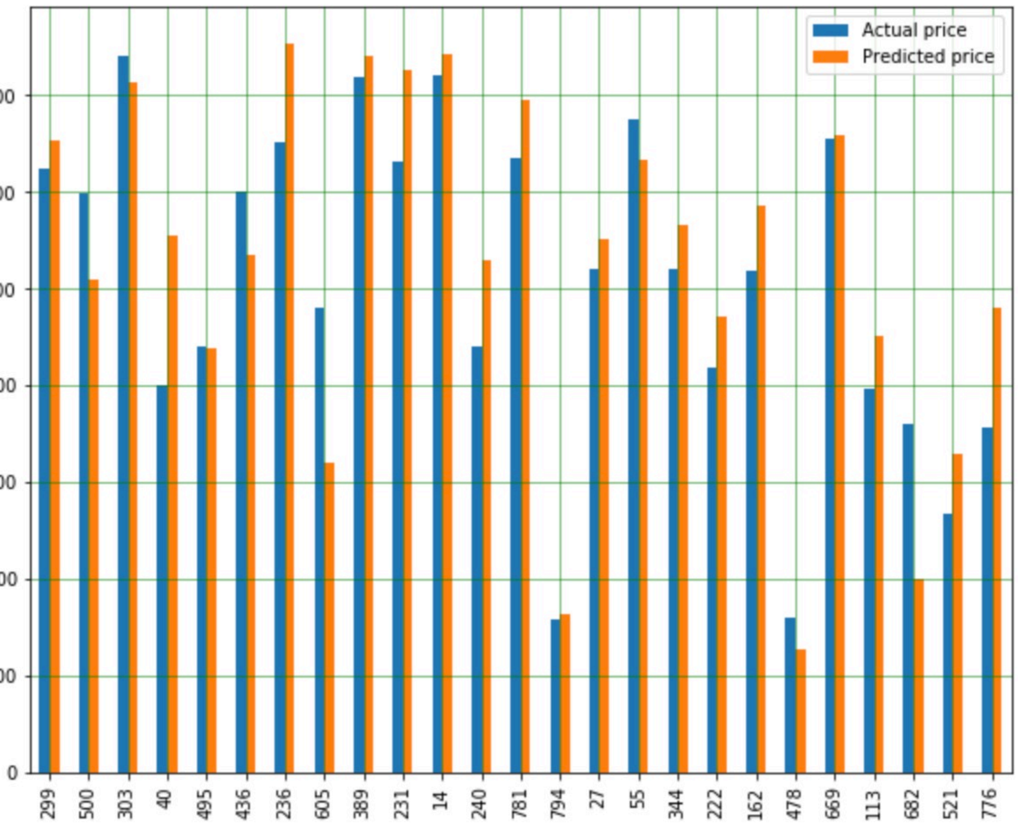
From this graph, we can see that the year is directly proportional to the price through univariate linear regression.



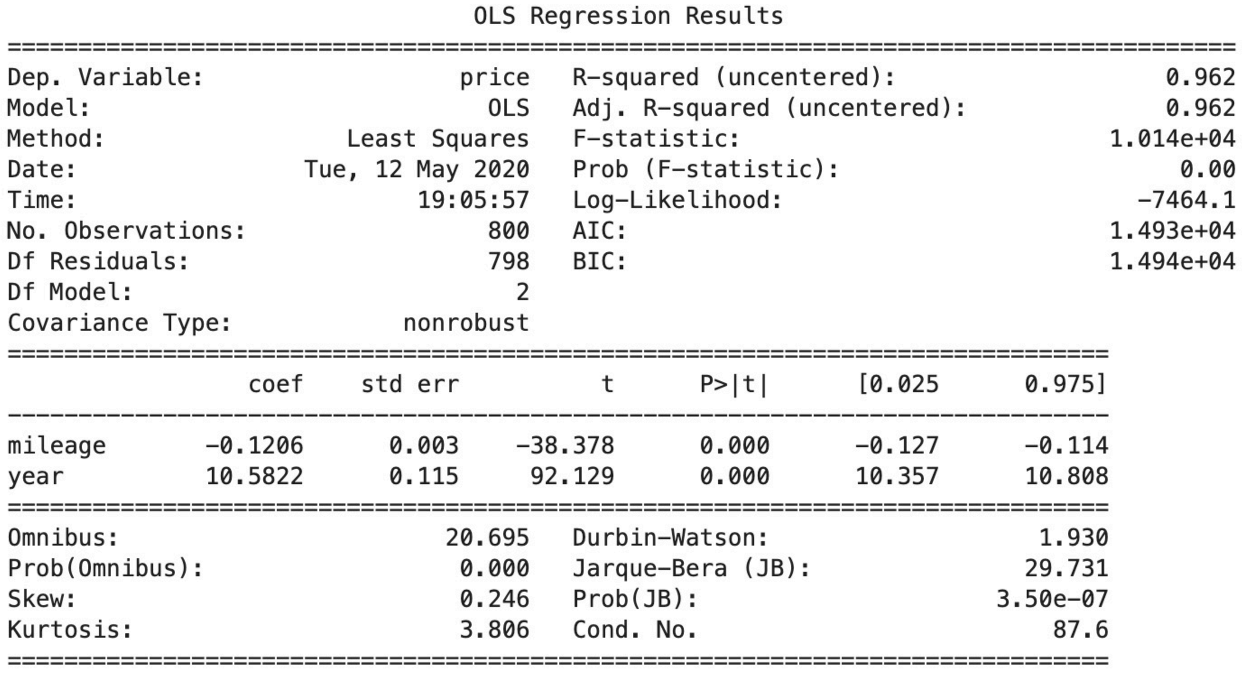
```
coefficient: [[1448.14127061]] intercept_: [-2901866.41057718]
```

Through multiple linear regression, 25 second-hand cars are randomly selected from the sample to predict their prices. From this figure, it can be seen that the price predicted by using the model is probably accurate with little error.

```
[160 rows x 2 columns]
```

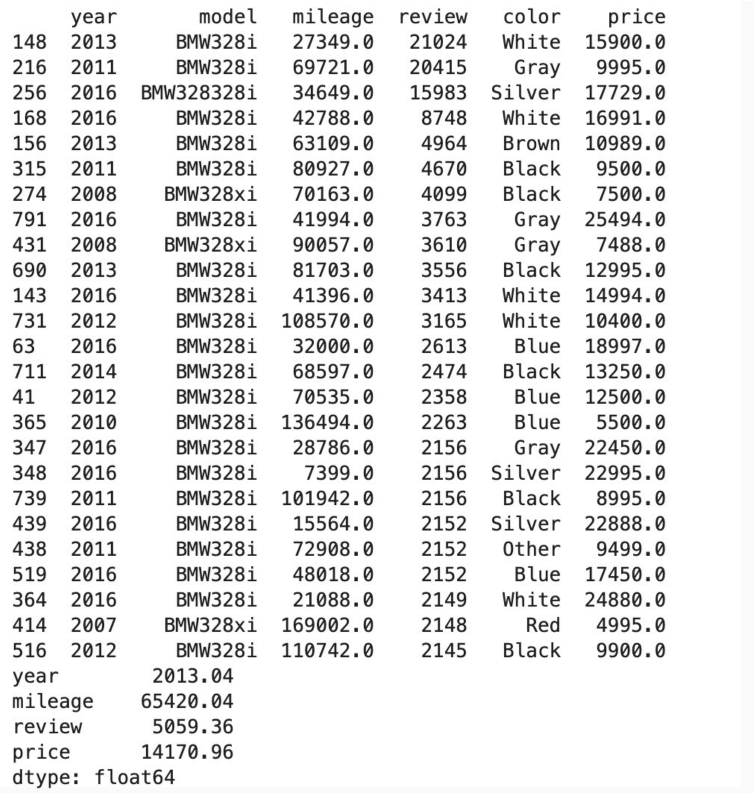


Through the parameter analysis of multiple linear regression, the square of R is 0.962, which shows that the model fits well.



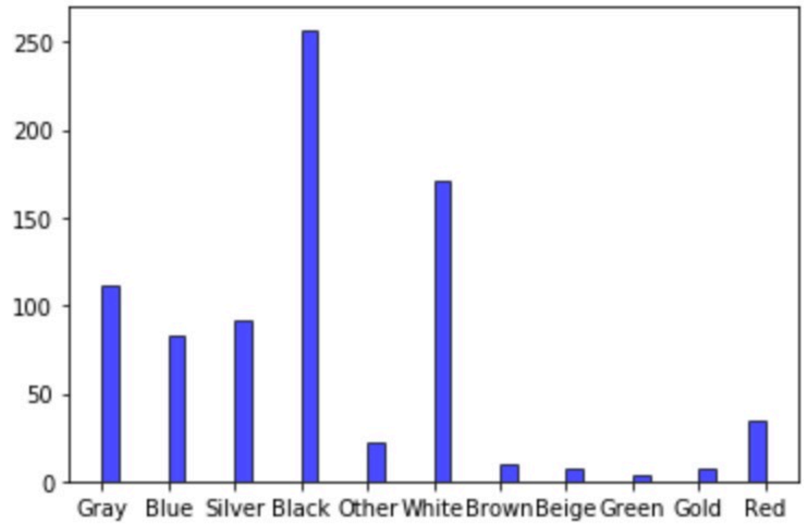
```
                                 OLS Regression Results
=======================================================================================
Dep. Variable:                  price   R-squared (uncentered):                   0.962
Model:                            OLS   Adj. R-squared (uncentered):              0.962
Method:                 Least Squares   F-statistic:                          1.014e+04
Date:                Tue, 12 May 2020   Prob (F-statistic):                        0.00
Time:                        19:05:57   Log-Likelihood:                         -7464.1
No. Observations:                 800   AIC:                                  1.493e+04
Df Residuals:                     798   BIC:                                  1.494e+04
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
mileage       -0.1206      0.003    -38.378      0.000      -0.127      -0.114
year          10.5822      0.115     92.129      0.000      10.357      10.808
==============================================================================
Omnibus:                       20.695   Durbin-Watson:                   1.930
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               29.731
Skew:                           0.246   Prob(JB):                     3.50e-07
Kurtosis:                       3.806   Cond. No.                         87.6
==============================================================================
```

By selecting the 25 cars with the most of reviews and calculating their means, I find that BMW 328 car with the year of 2013, the mileage of 65420 and the price of 14170 is very popular on the website.



```
     year       model   mileage  review   color    price
148  2013     BMW328i   27349.0   21024   White  15900.0
216  2011     BMW328i   69721.0   20415    Gray   9995.0
256  2016  BMW328328i   34649.0   15983  Silver  17729.0
168  2016     BMW328i   42788.0    8748   White  16991.0
156  2013     BMW328i   63109.0    4964   Brown  10989.0
315  2011     BMW328i   80927.0    4670   Black   9500.0
274  2008    BMW328xi   70163.0    4099   Black   7500.0
791  2016     BMW328i   41994.0    3763    Gray  25494.0
431  2008    BMW328xi   90057.0    3610    Gray   7488.0
690  2013     BMW328i   81703.0    3556   Black  12995.0
143  2016     BMW328i   41396.0    3413   White  14994.0
731  2012     BMW328i  108570.0    3165   White  10400.0
63   2016     BMW328i   32000.0    2613    Blue  18997.0
711  2014     BMW328i   68597.0    2474   Black  13250.0
41   2012     BMW328i   70535.0    2358    Blue  12500.0
365  2010     BMW328i  136494.0    2263    Blue   5500.0
347  2016     BMW328i   28786.0    2156    Gray  22450.0
348  2016     BMW328i    7399.0    2156  Silver  22995.0
739  2011     BMW328i  101942.0    2156   Black   8995.0
439  2016     BMW328i   15564.0    2152  Silver  22888.0
438  2011     BMW328i   72908.0    2152   Other   9499.0
519  2016     BMW328i   48018.0    2152    Blue  17450.0
364  2016     BMW328i   21088.0    2149   White  24880.0
414  2007    BMW328xi  169002.0    2148     Red   4995.0
516  2012     BMW328i  110742.0    2145   Black   9900.0
year        2013.04
mileage    65420.04
review      5059.36
price      14170.96
dtype: float64
```

The most popular colors of BMW 328i series on the website are black, white and gray. It is recommended not to buy green, gold and brown. They are very rare and hard to sell.


Gray
Blue
Silver
Black
Other
White
Brown
Beige
Green
Gold
Red