

**Data Visualization and Data Analysis Great Task**

**学 号 2017007809**

**姓 名 卫万灵**

**专 业 软件工程(信息与大数据工程方向)**

**授课老师 韩晓红**

**日 期 2019.6**

# The Deadline to Hand in the Task:

You must hand in this task on **July 4, 2019**, otherwise, your grades will decrease 10 points. The task must be printed out on paper.

# Dataset for the Task

The dataset used in the task is a house dataset, which contains 506 rows of instances and 14 features. Each row includes the information of house and its surrounding information of the house.

File of dataset：house\_data.csv

**Table 1 Dataset**

|  |  |  |
| --- | --- | --- |
| Number | Features | Description |
| 1 | Crime\_rate | Per capita Crime rate in cities |
| 2 | P\_RL | Proportion of Residential Land exceeding 25,000 sq |
| 3 | P\_NRL | Proportion of Non-Retail Land |
| 4 | CRSV | Charles River Spatial Variables ( if the boundary is a river, CRSV =1,or, CRSV =0) |
| 5 | NOC | Nitric Oxide Concentration |
| 6 | ANR | Average Number of Rooms in a house |
| 7 | P\_SUH | Proportion of previously built Self-Use Housing |
| 8 | WDCA | Weighted Distance to Central Areas |
| 9 | PIRH | Approaching Index of Radiative Highway |
| 10 | Tax\_rate | Tax rate of full-value property per $10,000 |
| 11 | P\_TS | Proportion of Teachers and Students in cities |
| 12 | P\_black | Proportion of black in cities |
| 13 | P\_under | Proportion of the underprivileged in the population |
| 14 | Style | Style of house |
| 15 | Price | Price of house |

# Problems Description:

In this task, you should first complete the following assigns and then create models to predict the price of a house based on the comprehensive information of house.

# 1. Data processing

## 1.1 Import Data from house\_data.csv

### 1.1.1 Import data from house\_data.csv

**Your answer（包括代码和运行结果）:**

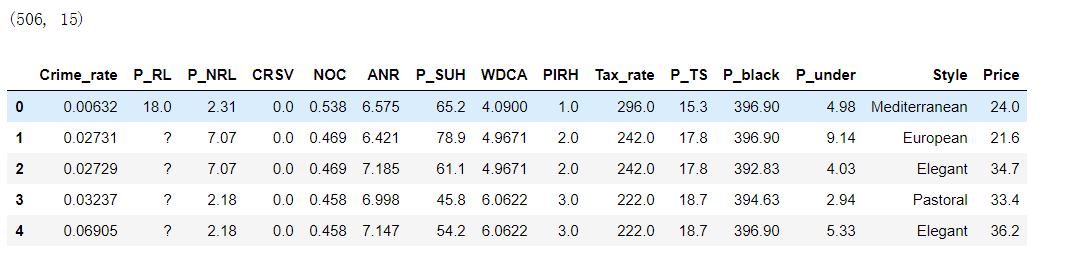
path=r'./house\_data.csv'

df=pd.read\_csv(path)

df.drop(['Unnamed: 0'],inplace=True,axis=1)

print(df.shape)

df.head()



## 1.2 Identify and handle missing values

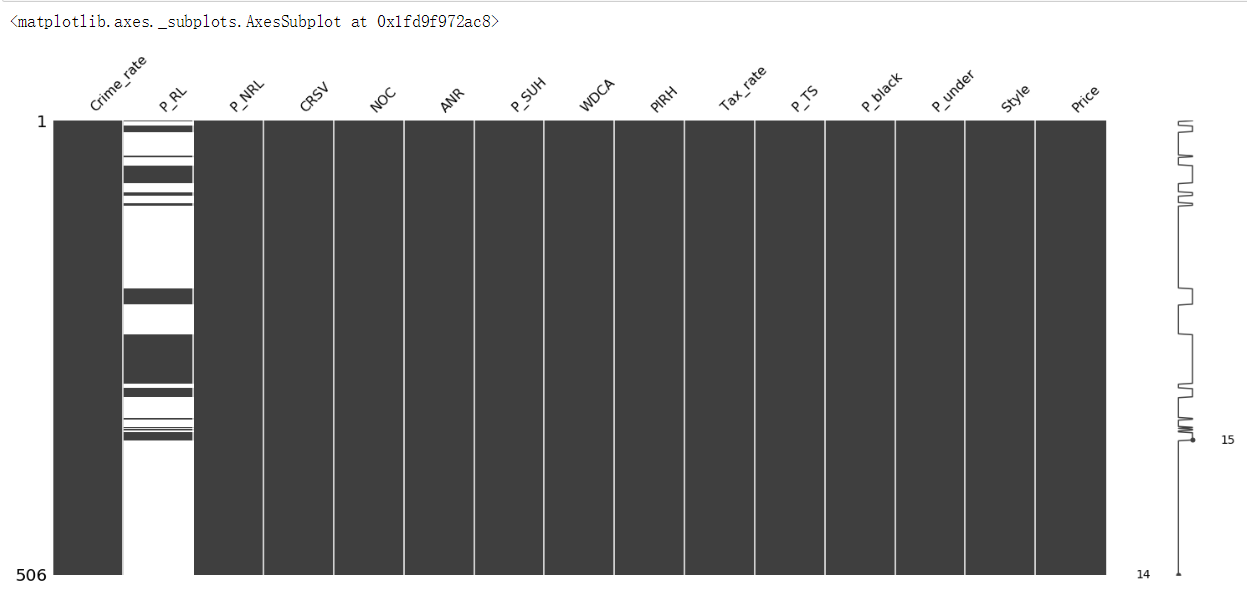
### 1.2.1 Deal with missing data and replace it by frequency

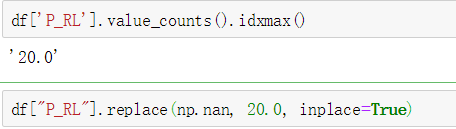
**Your answer（包括代码和运行结果）:**

import missingno as msno

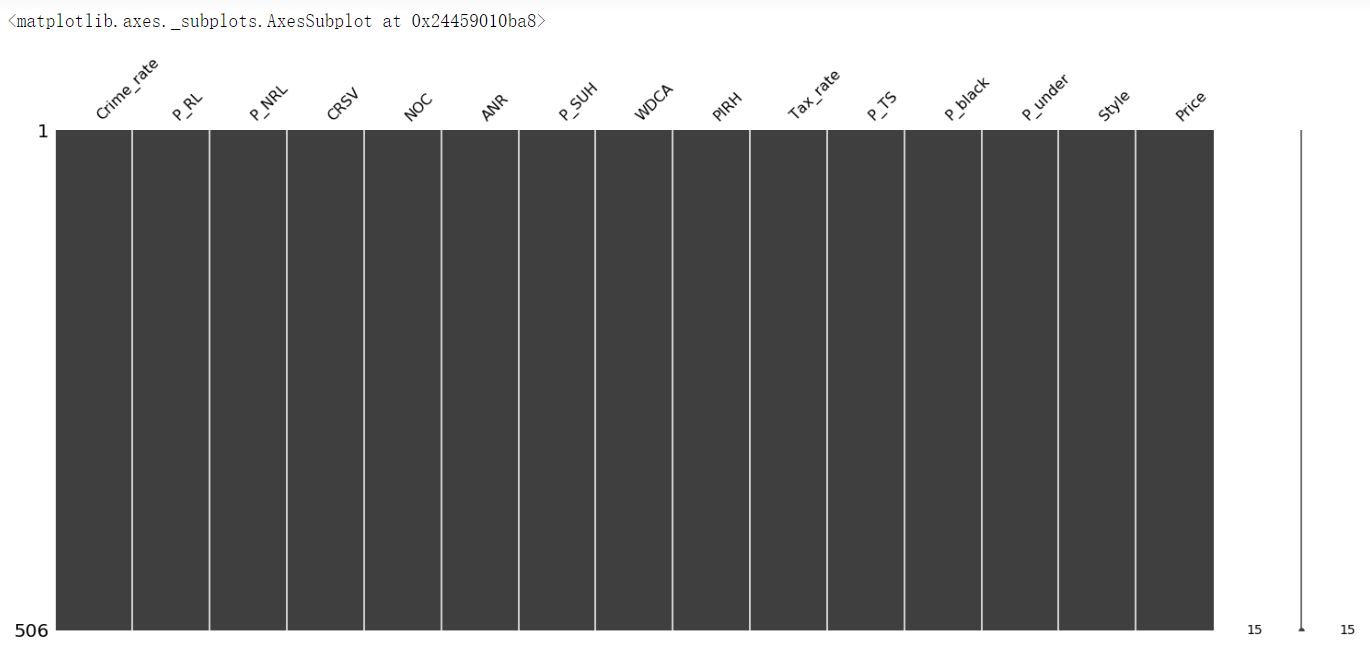
df.replace('?',np.nan,inplace=True)

msno.matrix(df)









### 1.2.2 Normalize 'Tax\_rate' and 'P\_black' variables so their value ranges from 0 to 1

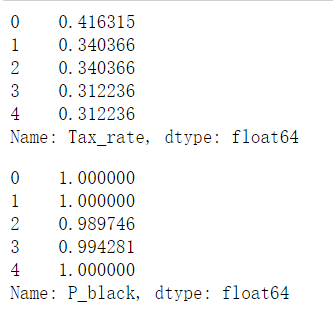
**Your answer（包括代码和运行结果）:**

df['Tax\_rate']=df['Tax\_rate']/df['Tax\_rate'].max()

df['P\_black']=df['P\_black']/df['P\_black'].max()

print(df['Tax\_rate'].head())

print(df['P\_black'].head())



# 2. Data analysis

## 2.1. Import Data from house\_data.csv

### 2.1.1 Import data from house\_data.csv

**Your answer（包括代码和运行结果）:**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import scipy.stats as stats

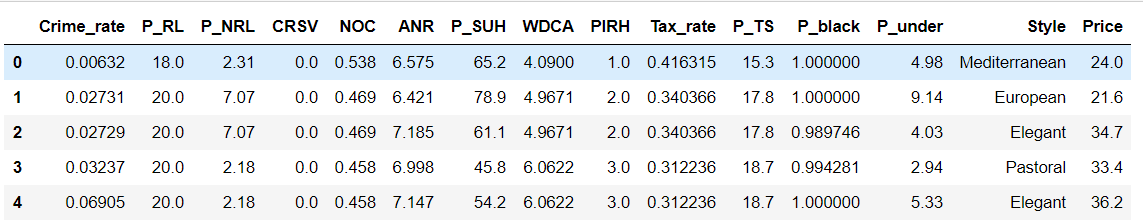
%matplotlib inline

path=r'./house\_to\_module2.csv'

df=pd.read\_csv(path)

df.drop(['Unnamed: 0'],inplace=True,axis=1)

df.head()

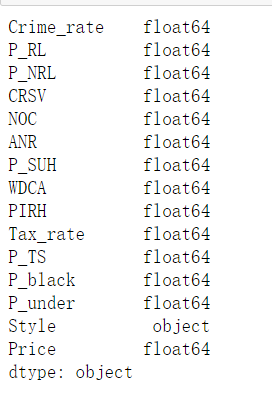


## 2.2 Analyzing Individual Feature Patterns using Visualization

### 2.2.1 List the data types for each column

**Your answer（包括代码和运行结果）:**

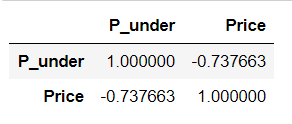
df.dtypes



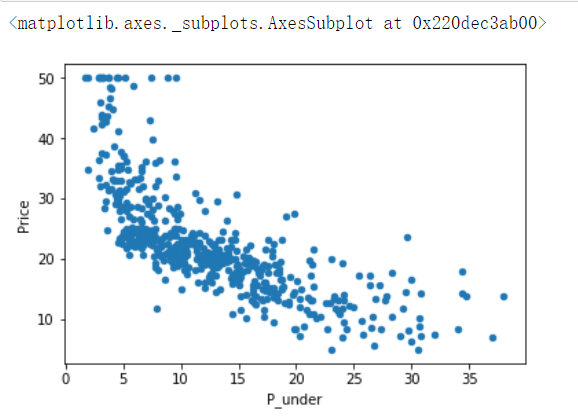
### 2.2.2 Find the correlation between " P\_under " and "Price", and  plots the scatterplot of " P\_under " and "Price".

**Your answer（包括代码和运行结果）:**

df[['P\_under','Price']].corr()



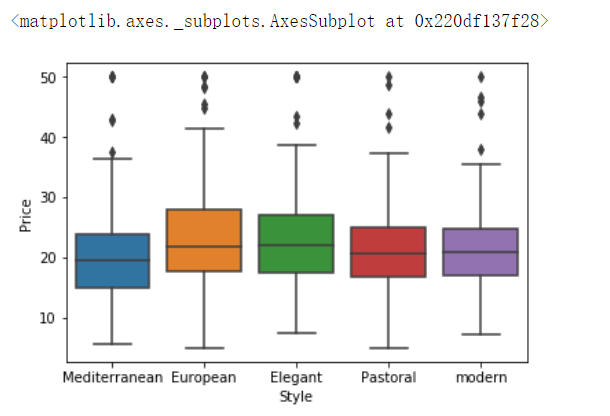
df.plot(kind='scatter',x='P\_under',y='Price')



### 2.2.3 Visualize relationship between "Style" (categorical variables) and "Price" by using boxplots.

**Your answer（包括代码和运行结果）:**

sns.boxplot(x="Style", y="Price", data=df) # pastoral 田园 elegant 优雅

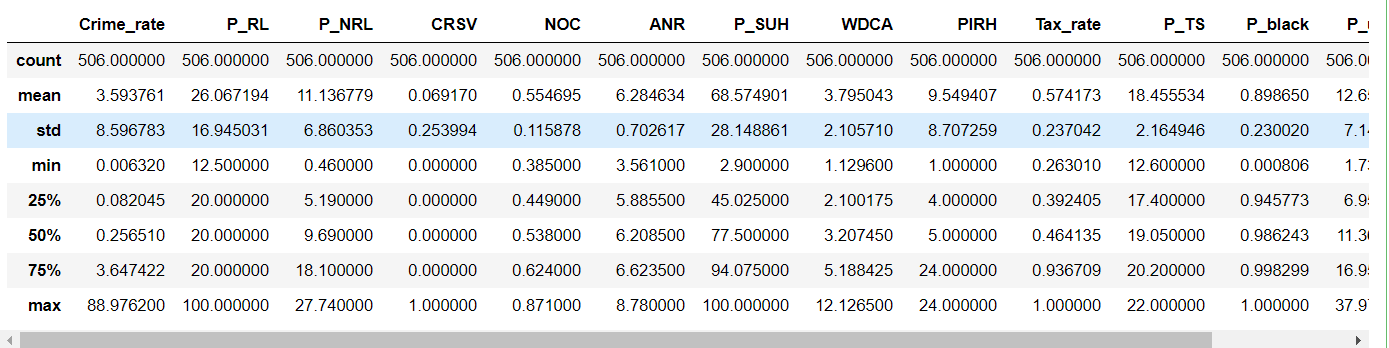


### 2.2.4 Use describe function compute basic statistics for all continuous variables, including

* the count of that variable
* the mean
* the standard deviation (std)
* the minimum value
* the IQR (Interquartile Range: 25%, 50% and 75%)
* the maximum value

**Your answer（包括代码和运行结果）:**

df.describe()



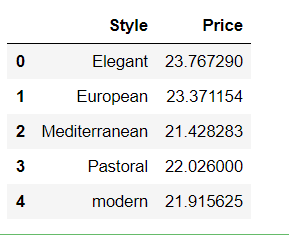
### 2.2.5 Use the "groupby" function to find the average "Price" of each car based on "Style"

**Your answer（包括代码和运行结果）:**

df\_group\_one = df[['Style','Price']]

df\_group\_one = df\_group\_one.groupby(['Style'],as\_index=False).mean()

df\_group\_one



### 2.2.6 Use a heat map to visualize the relationship between "Style" and "Price"

**Your answer（包括代码和运行结果）:**

grouped\_pivot=df\_group\_one.pivot(index='Style',columns=[1,1,1,1,1])

fig, ax = plt.subplots()

im = ax.pcolor(grouped\_pivot, cmap='RdBu')

#label names

row\_labels = grouped\_pivot.columns.levels[1]

col\_labels = grouped\_pivot.index

#move ticks and labels to the center

ax.set\_xticks(np.arange(grouped\_pivot.shape[1]) + 0.5, minor=False)

ax.set\_yticks(np.arange(grouped\_pivot.shape[0]) + 0.5, minor=False)

#insert labels

ax.set\_xticklabels(row\_labels, minor=False)

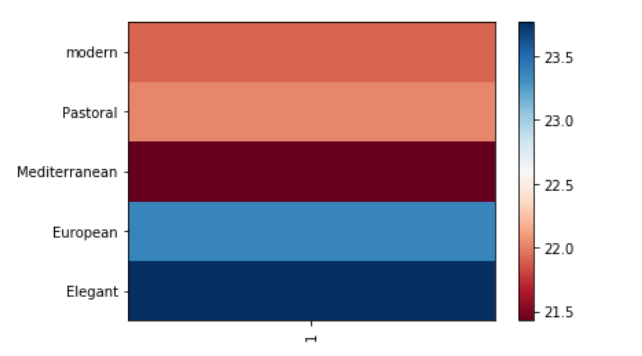
ax.set\_yticklabels(col\_labels, minor=False)

#rotate label if too long

plt.xticks(rotation=90)

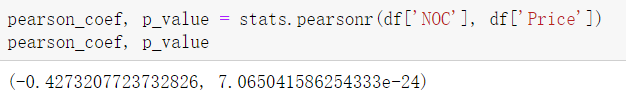
fig.colorbar(im)

plt.show()



### 2.2.7 Calculate the Pearson Correlation Coefficient and P-value of ' NOC ' and 'price':

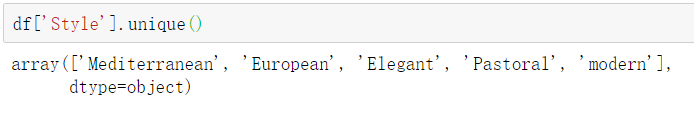
**Your answer（包括代码和运行结果）:**

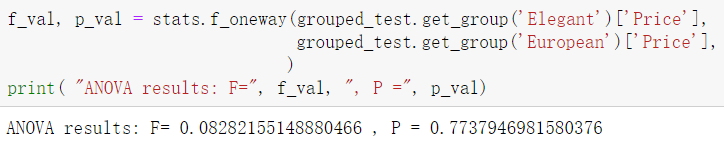


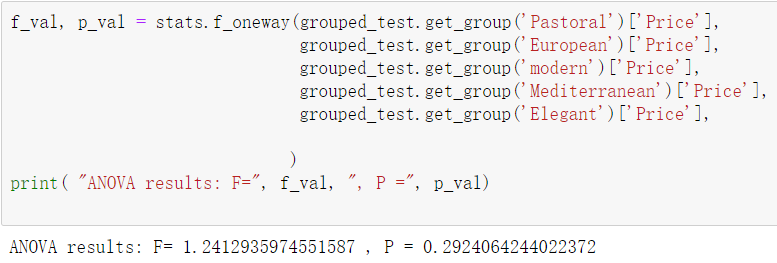
### 2.2.8 Use the function 'f\_oneway' in the module 'stats' to obtain the F-test score and P-value:

**Your answer（包括代码和运行结果）:**









### 2.2.9 Which variables are important to take into account when predicting the house price.

**Your answer（包括代码和运行结果）:**

col\_coef\_pvalue=[]

for col in df.columns.values.tolist()[0:13]:

pearson\_coef, p\_value = stats.pearsonr(df[col], df['Price'])

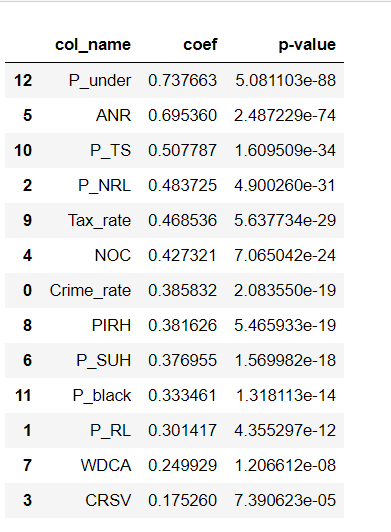
col\_coef\_pvalue.append([col,pearson\_coef,p\_value])

col\_coef\_pvalue=pd.DataFrame(col\_coef\_pvalue,columns=['col\_name','coef','p-value'])#coef降序

col\_coef\_pvalue['coef']=abs(col\_coef\_pvalue['coef'])

col\_coef\_pvalue.sort\_values(['coef'],ascending=False,inplace=True,axis=0)

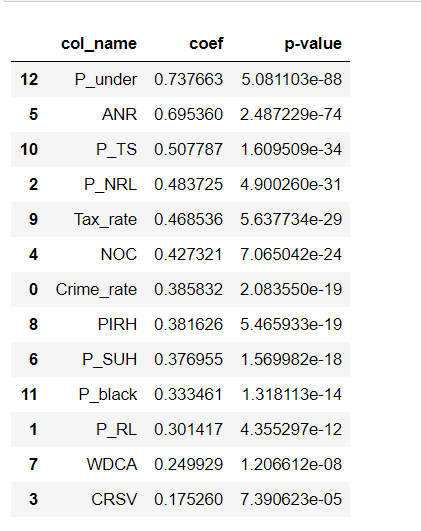
col\_coef\_pvalue



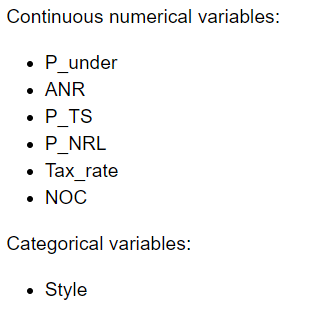
#p-value 升序

col\_coef\_pvalue.sort\_values(['p-value'],ascending=True,inplace=True,axis=0)

col\_coef\_pvalue



Having narrowed it down to the following variables:



## 2.3 Select Import Variables

### 2.3.1 Based on The Pearson Correlation Coefficient and P-value to determine which variables are important to take into account when predicting the house price. You should narrow it down to the following variables:

**Your answer（包括代码和运行结果）:**

① P\_under

② ANR

③ P\_TS

④ P\_NRL

⑤ Tax\_rate

⑥ NOC

# 3. Model Development

## 3.1 Linear Regression and Multiple Linear Regression

### 3.1.1 Use simple linear regression to create a Linear Regression Mode (LRM) with " Crime\_rate " as the predictor variable and the "Price" as the response variable. What is the equation of the predicted line? You can use x and yhat

**Your answer（包括代码和运行结果）:**

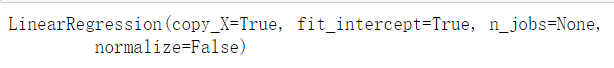
from sklearn.linear\_model import LinearRegression

lr=LinearRegression()

x=df[['Crime\_rate']];

y=df['Price']

lr.fit(x,y)



yhat=lr.predict(x)

print('slope=',lr.coef\_,' intercept=',lr.intercept\_)





### 3.1.2 Create a Multiple Linear Regression Model (MLRM)where the response variable is Price, and the predictor variable is 'WDCA' , 'PIRH', and 'Tax\_rate'. ?

**Your answer（包括代码和运行结果）:**

mlr=LinearRegression()

X=df[['WDCA','PIRH','Tax\_rate']]

Z=df['Price']

mlr.fit(X,Z)

print(mlr.intercept\_,mlr.coef\_)



## 3.2 Model Evaluation using Visualization

### 3.2.1 Use ****regression plots to**** visualize ' P\_under' as potential predictor variable of 'Price'

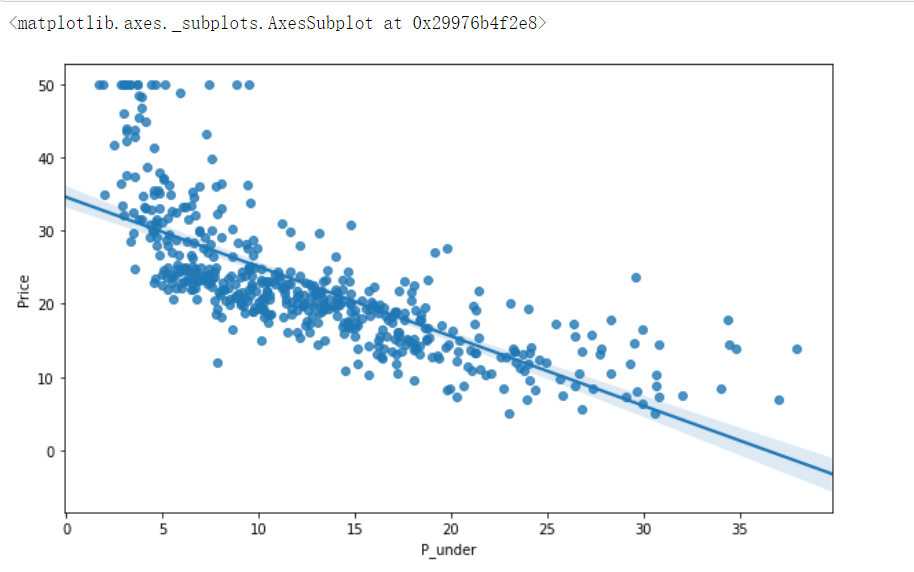
**Your answer（包括代码和运行结果）:**

width = 12

height = 10

plt.figure(figsize=(width, height))

sns.regplot(x="P\_under", y="Price", data=df)



### 3.2.2 Use residual plots to determine which linear model is appropriate for the data among the following models:

model1: 'PIRH' as potential predictor variable of 'Price'

model2: 'Tax\_rate' as potential predictor variable of 'Price'

model3: 'P\_TS' as potential predictor variable of 'Price'

model4: 'P\_black' as potential predictor variable of 'Price'

model5: 'P\_under' as potential predictor variable of 'Price'

**Your answer（包括代码和运行结果）:**

width = 12

height = 10

col5=['PIRH','Tax\_rate','P\_TS','P\_black','P\_under']

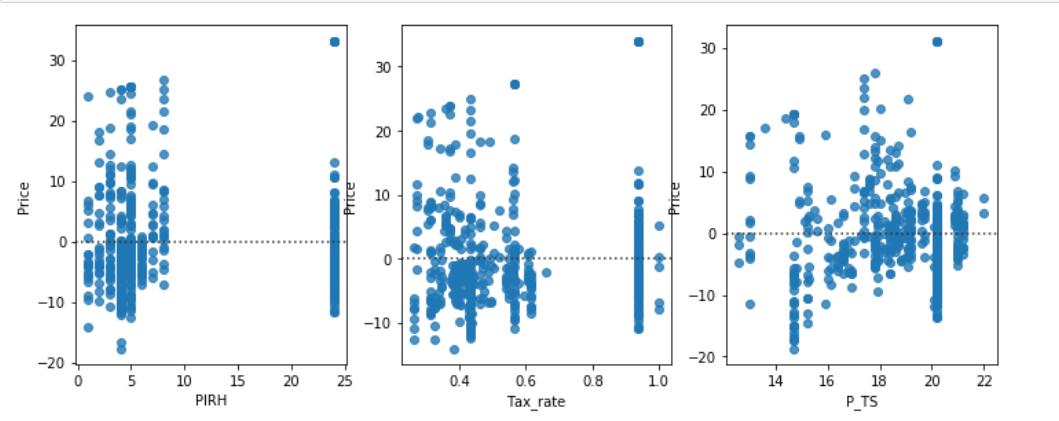
plt.figure(figsize=(width, height))

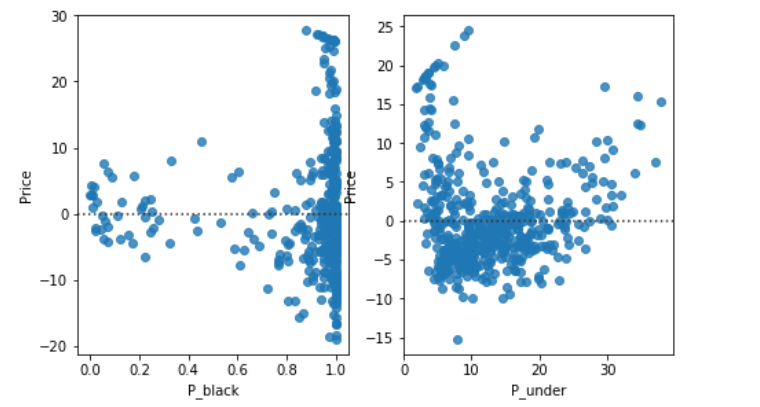
for i in range(5):

plt.subplot(2,3,i+1) #'PIRH', 'Tax\_rate' 'P\_TS' 'P\_black' 'P\_under'

sns.residplot(df[col5[i]], df['Price'])

plt.show()





**Conclusion:**

The appropriate linear model is model5

### 3.2.3 Use the distribution plot to look at the distribution of the fitted values of that result from the model MLRM and compare it to the distribution of the actual values.

**Your answer（包括代码和运行结果）:**

Yhat=mlr.predict(X)

plt.figure(figsize=(width, height))

ax1 = sns.distplot(df['Price'], hist=False, color="r", label="Actual Value")

sns.distplot(Yhat, hist=False, color="b", label="Fitted Values" , ax=ax1)

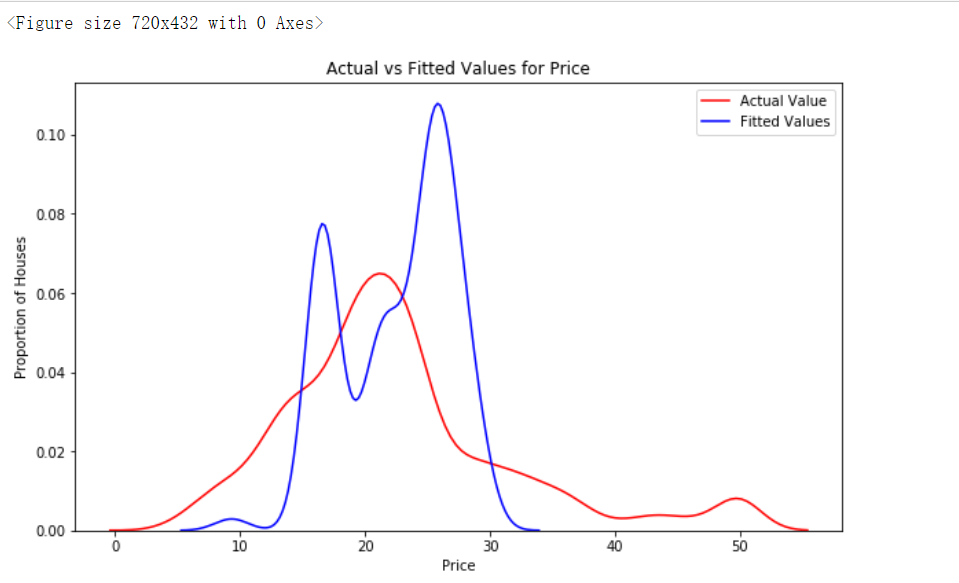
plt.title('Actual vs Fitted Values for Price')

plt.xlabel('Price')

plt.ylabel('Proportion of Houses')

plt.show()

plt.close()



## 3.3 Polynomial Regression and Pipelines

### 3.3.1 Use a polynomial of the 3rd order (cubic) to create a Polynomial Regression Model (PRM) through the function polyfit, then use the function poly1d to display the polynomial function, " Crime\_rate " as the predictor variable and the "Price" as the response variable.

**Your answer（包括代码和运行结果）:**

xp = df['Crime\_rate']

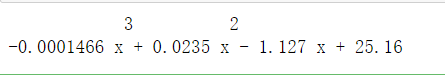
yp = df['Price']

f = np.polyfit(xp, yp, 3)

p = np.poly1d(f)

yphat=p(xp)

print(p)



### 3.3.2 Plot the above polynomial function(PRM)

**Your answer（包括代码和运行结果）:**

def PlotPolly(model, independent\_variable, dependent\_variabble, Name):

x\_new = np.linspace(independent\_variable.min(),independent\_variable.max() , 100)

y\_new = model(x\_new)

plt.plot(independent\_variable, dependent\_variabble, '.', x\_new, y\_new, '-')

plt.title('Polynomial Fit with Matplotlib for Price ~ Length')

ax = plt.gca()

ax.set\_facecolor((0.898, 0.898, 0.898))

fig = plt.gcf()

plt.xlabel(Name)

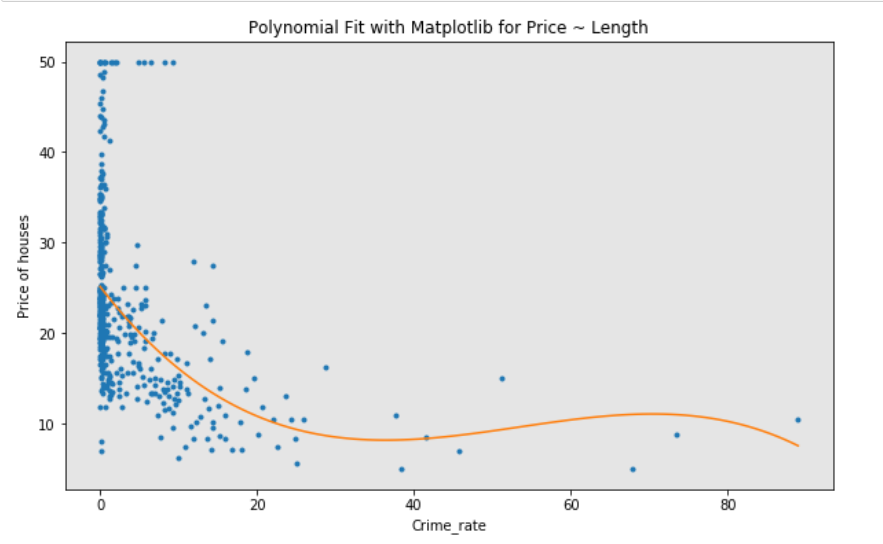
plt.ylabel('Price of houses')

plt.show()

plt.close()

plt.figure(figsize=(10,6))

PlotPolly(p, xp, yp, 'Crime\_rate')



## 3.4 Measures for In-Sample Evaluation

### ****3.4.1 Compute the R-squared and Mean Squared Error (MSE) of**** LRM ****model (****created ****in**** 3.1.1 ) MLRM model (created in 3.1.2 ), and PRM model ( created in 3.3.1)

**Your answer（包括代码和运行结果）:**

# 模型(属性,价格实际值,价格预测值) lr(x,y,yhat) mlr(X,Z,Yhat) f,p(xp,yp,yphat)

from sklearn.metrics import mean\_squared\_error

from sklearn.metrics import r2\_score

#计算score

score1=lr.score(x,y)

score2=mlr.score(X,Z)

score3=r2\_score(yp,yphat)

#计算MSE

mse1= mean\_squared\_error(y, yhat)

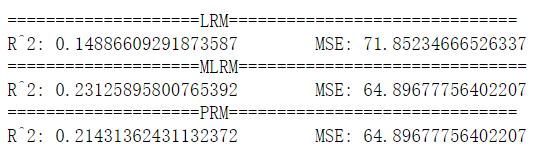
mse2 = mean\_squared\_error(Z, Yhat)

mse3 = mean\_squared\_error(yp, Yhat)

print('='\*20,'LRM','='\*30,'\nR^2: ',score1,'\tMSE: ',mse1,sep='')

print('='\*20,'MLRM','='\*30,'\nR^2: ',score2,'\tMSE: ',mse2,sep='')

print('='\*20,'PRM','='\*30,'\nR^2: ',score3,'\tMSE: ',mse3,sep='')



## 3.5 Prediction and Decision Making

### 3.5.1 Create all possible simple linear models and select a best simple linear model.(Hint: Based on the important variables you have selected in 2.3.1, so you should create all simple linear models. You can choose a best simple linear model through comparing their ****R-squared and Mean Squared Error)****

**Your answer（包括代码和运行结果）:**

attributes=['P\_under','ANR','P\_TS','P\_NRL','Tax\_rate','NOC']

#寻找一元线性拟合的一个最优属性值

lr=LinearRegression()

def slrfun(x,y,yhat):

score=lr.score(x,y)

mse= mean\_squared\_error(y, yhat)

return score,mse;

slr\_r2\_mse=[]

for i in range(len(attributes)):

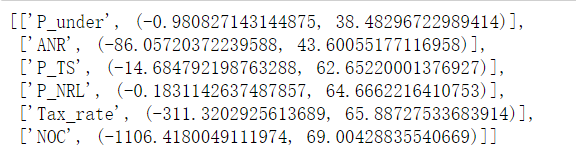
lr.fit(df[[attributes[i]]],df['Price'])

yhat=lr.predict(df[[attributes[i]]])

temp=slrfun(x,y,yhat)

slr\_r2\_mse.append([attributes[i],temp])

slr\_r2\_mse



**Conclusion:**

The best simple linear model is based on the feature "P\_under".

### 3.5.2 Create all possible multiple linear regression model and select a best multiple linear regression model. (Hint: Based on the important variables you have selected in 2.3.1, you can choose a best multiple linear regression model through comparing their R-squared and Mean Squared Error)

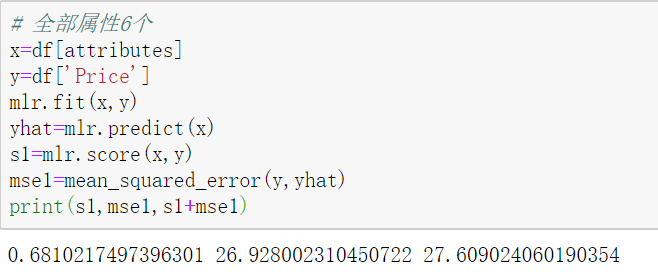
**Your answer（包括代码和运行结果）:**











**Conclusion:**

The best multiple linear model is based on the all features that are selected in 2.3.1

### 3.5.3 Create all possible polynomial regression model and select a best polynomial regression model. (Hint: Based on the important variables you have selected in 2.3.1, you can choose a best polynomial regression model through comparing their R-squared and Mean Squared Error)

**Your answer（包括代码和运行结果）:**

#寻找多项式拟合的一个最优属性值

def polyfun(x,y,yhat):

score=r2\_score(x,y)

mse= mean\_squared\_error(y, yhat)

return score,mse;

plr\_r2\_mse=[]

for i in range(len(attributes)):

attrs=df[attributes[i]]

f = np.polyfit(attrs, df['Price'], 3)

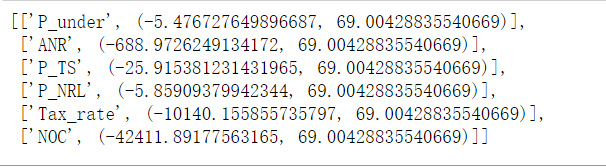
p = np.poly1d(f)

yphat=p(attrs)

temp=polyfun(attrs,df['Price'],yhat)

plr\_r2\_mse.append([attributes[i],temp])

plr\_r2\_mse



**Conclusion:**

The best polynomial regression model is based on the feature "P\_under".

# 4. Predict the price using Ridge Regression

## 4.1 predict the price of house using ridge regression

**Your answer（包括代码和运行结果）:**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.linear\_model import LinearRegression,Ridge

from sklearn.preprocessing import PolynomialFeatures

from sklearn.metrics import mean\_squared\_error

from sklearn.model\_selection import train\_test\_split

%matplotlib inline

attributes=['P\_under','ANR','P\_TS','P\_NRL','Tax\_rate','NOC']

path=r'./house\_to\_module2.csv'

df=pd.read\_csv(path)

df.drop(['Unnamed: 0'],inplace=True,axis=1)

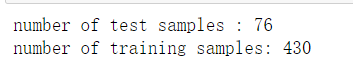
x\_data=df[attributes]

y\_data=df['Price']

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x\_data, y\_data, test\_size=0.15, random\_state=1)

print("number of test samples :", x\_test.shape[0])

print("number of training samples:",x\_train.shape[0])



attributes=['P\_under','ANR','P\_TS','P\_NRL','Tax\_rate','NOC']

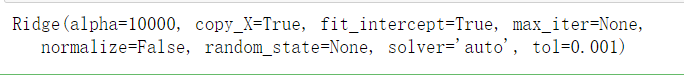
pr=PolynomialFeatures(degree=2)

x\_train\_pr=pr.fit\_transform(x\_train[attributes])

x\_test\_pr=pr.fit\_transform(x\_test[attributes])

RigeModel=Ridge(alpha=10000)

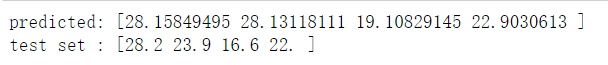
RigeModel.fit(x\_train\_pr, y\_train)



yhat = RigeModel.predict(x\_test\_pr)

print('predicted:', yhat[0:4])

print('test set :', y\_test[0:4].values)



score=RigeModel.score(x\_test\_pr,y\_test)

mse=mean\_squared\_error(y\_test,yhat)

print('score: ',score,'MSE: ',mse)



plt.figure(figsize=(10,6))

plt.plot(range(0,len(y\_test)), y\_test,label='Actual values')

plt.plot(range(0,len(y\_test)), yhat,label='Predicted values')

plt.ylabel('Price')

plt.xlabel('range(0,len(y\_test))')

plt.legend()

