Import the required libraries we need for the lab.

```
import piplite
await piplite.install(['numpy'],['pandas'])
await piplite.install(['seaborn'])

import pandas as pd
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as pyplot
import scipy.stats
import statsmodels.api as sm
from statsmodels.formula.api import ols
```

Read the dataset in the csv file from the URL

```
from js import fetch
import io

URL = 'https://cf-courses-data.s3.us.cloud-object-
storage.appdomain.cloud/IBMDeveloperSkillsNetwork-ST0151EN-
SkillsNetwork/labs/boston_housing.csv'
resp = await fetch(URL)
boston_url = io.BytesIO((await resp.arrayBuffer()).to_py())

boston_df=pd.read_csv(boston_url)
```

Add your code below following the instructions given in the course to complete the peer graded assignment

```
import piplite
await piplite.install(['numpy'],['pandas'])
await piplite.install(['seaborn'])

import pandas as pd
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as pyplot
import scipy.stats
import statsmodels.api as sm
from statsmodels.formula.api import ols

import warnings
warnings.filterwarnings('ignore')

from js import fetch
import io

URL = 'https://cf-courses-data.s3.us.cloud-object-
storage.appdomain.cloud/IBMDeveloperSkillsNetwork-ST0151EN-
SkillsNetwork/labs/boston_housing.csv'
```

```
resp = await fetch(URL)
boston_url = io.BytesIO((await resp.arrayBuffer()).to_py())
```

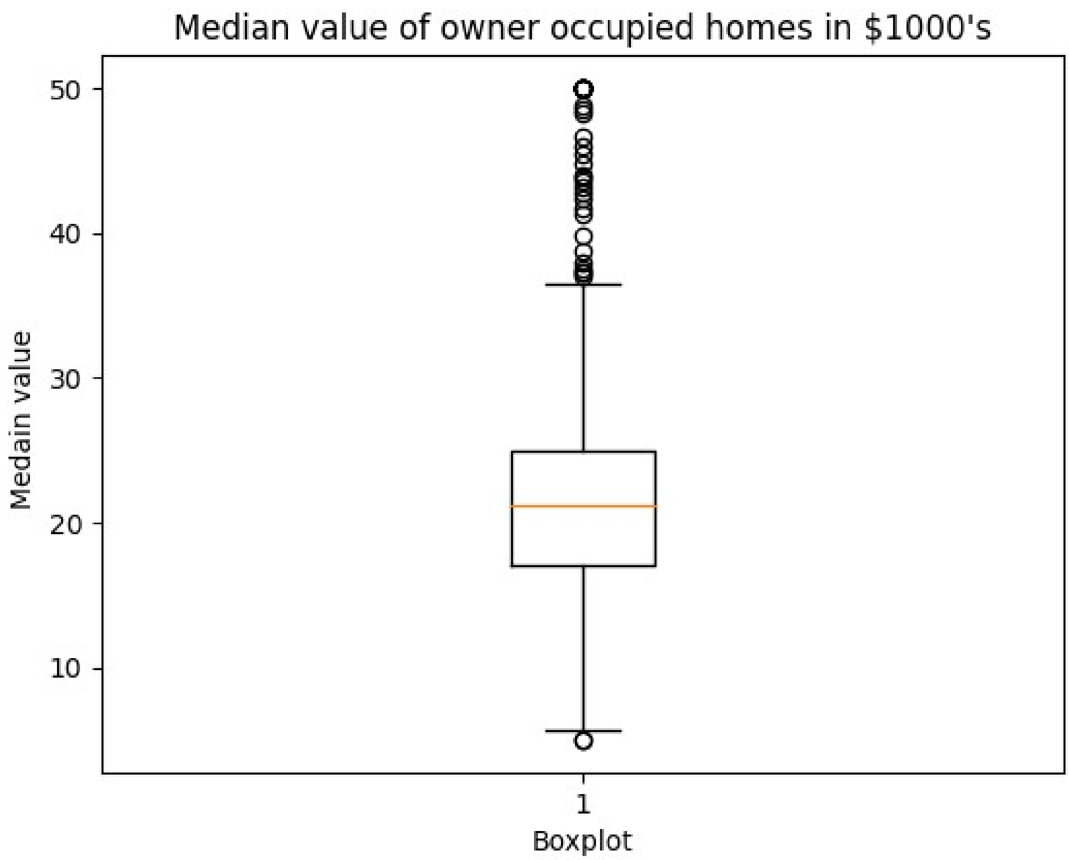
Data

```
import csv
boston_df=pd.read_csv(boston_url)
boston_df
```

```
     Unnamed: 0     CRIM    ZN  INDUS  CHAS    NOX     RM   AGE
DIS  RAD  \
0             0  0.00632  18.0   2.31   0.0  0.538  6.575  65.2
4.0900  1.0
1             1  0.02731   0.0   7.07   0.0  0.469  6.421  78.9
4.9671  2.0
2             2  0.02729   0.0   7.07   0.0  0.469  7.185  61.1
4.9671  2.0
3             3  0.03237   0.0   2.18   0.0  0.458  6.998  45.8
6.0622  3.0
4             4  0.06905   0.0   2.18   0.0  0.458  7.147  54.2
6.0622  3.0
..          ...      ...   ...    ...   ...    ...    ...   ...     ..
.  ...
501         501  0.06263   0.0  11.93   0.0  0.573  6.593  69.1
2.4786  1.0
502         502  0.04527   0.0  11.93   0.0  0.573  6.120  76.7
2.2875  1.0
503         503  0.06076   0.0  11.93   0.0  0.573  6.976  91.0
2.1675  1.0
504         504  0.10959   0.0  11.93   0.0  0.573  6.794  89.3
2.3889  1.0
505         505  0.04741   0.0  11.93   0.0  0.573  6.030  80.8
2.5050  1.0

       TAX  PTRATIO  LSTAT  MEDV
0    296.0     15.3   4.98  24.0
1    242.0     17.8   9.14  21.6
2    242.0     17.8   4.03  34.7
3    222.0     18.7   2.94  33.4
4    222.0     18.7   5.33  36.2
..     ...      ...    ...   ...
501  273.0     21.0   9.67  22.4
502  273.0     21.0   9.08  20.6
503  273.0     21.0   5.64  23.9
504  273.0     21.0   6.48  22.0
505  273.0     21.0   7.88  11.9

[506 rows x 14 columns]
```

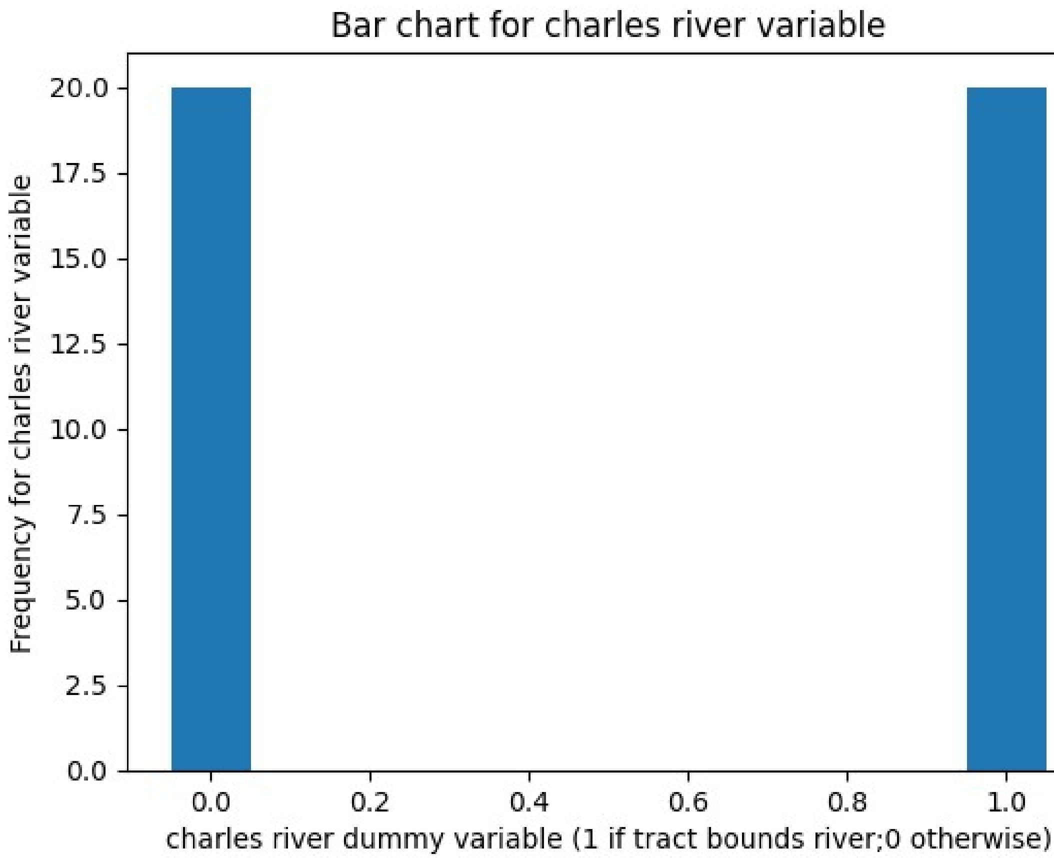
Boxplot for Median Value of owner-occupied home :

```
import matplotlib.pyplot as plt
plt.boxplot('MEDV',data=boston_df)
plt.xlabel("Boxplot")
plt.ylabel("Medain value ")
plt.title("Median value of owner occupied homes in $1000's")
plt.show()
```



Bar chart for the Charles river variable :

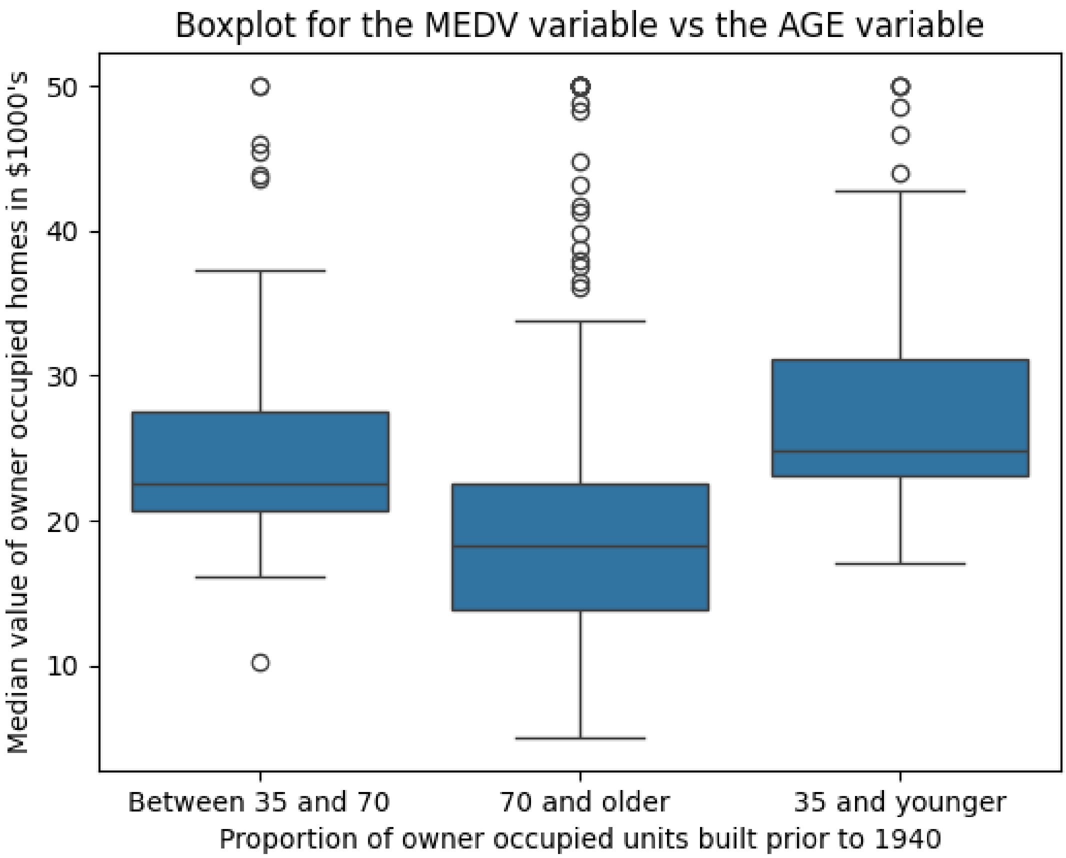
```
import matplotlib.pyplot as plt
plt.bar('CHAS', data=boston_df , height=20.0, width=0.1)
plt.xlabel("charles river dummy variable (1 if tract bounds river;0
otherwise)")
plt.ylabel("Frequency for charles river variable")
plt.title("Bar chart for charles river variable")
plt.show()
```



Boxplot for the MEDV variable vs the AGE variable :

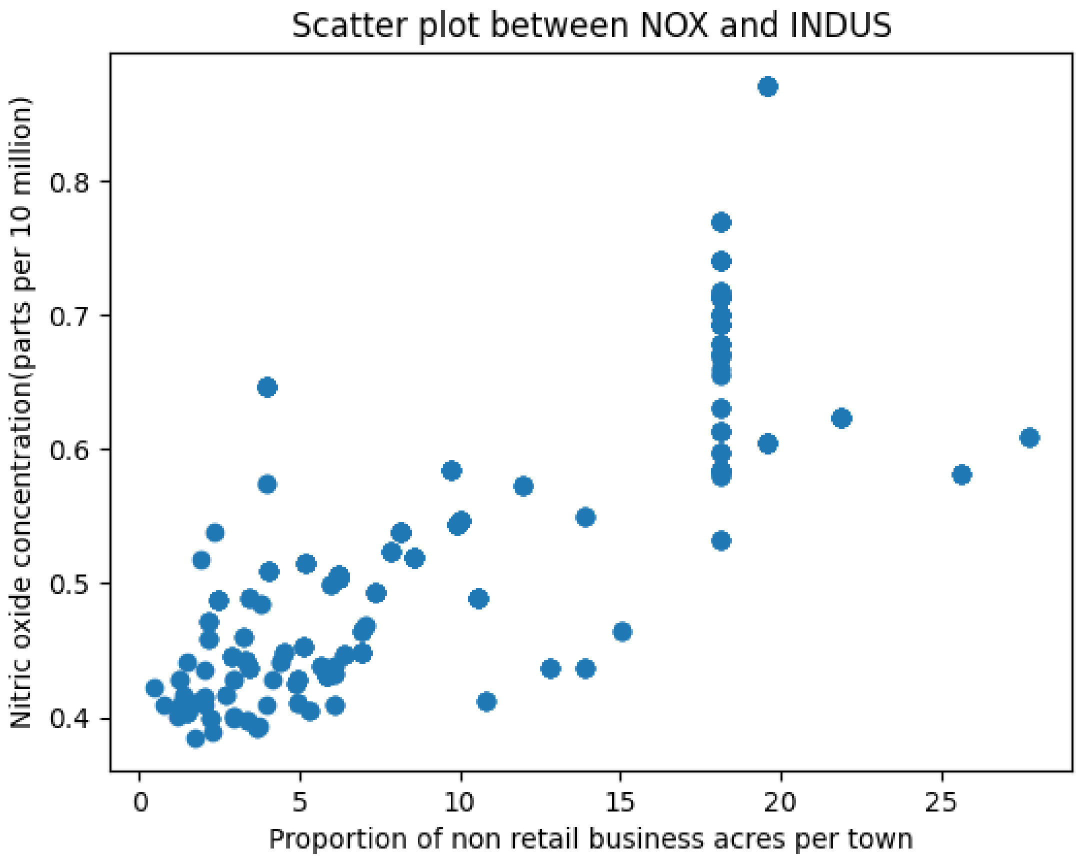
```
boston_df.loc[boston_df['AGE'] <= 35 , 'Age_Group' ] = "35 and
younger"
boston_df.loc[(boston_df['AGE'] > 35) & (boston_df['AGE'] < 70) ,
'Age_Group' ] = "Between 35 and 70"
boston_df.loc[(boston_df['AGE'] > 70 ) , 'Age_Group' ] = " 70 and
older"

boxplot=sns.boxplot(x = 'Age_Group' , y = 'MEDV' , data = boston_df)
boxplot.set(xlabel = "Proportion of owner occupied units built prior
to 1940" , ylabel = "Median value of owner occupied homes in
$1000's" , title = " Boxplot for the MEDV variable vs the AGE variable
" )
```

```
[Text(0.5, 0, 'Proportion of owner occupied units built prior to
1940'),
 Text(0, 0.5, "Median value of owner occupied homes in $1000's"),
 Text(0.5, 1.0, ' Boxplot for the MEDV variable vs the AGE variable
')]
```



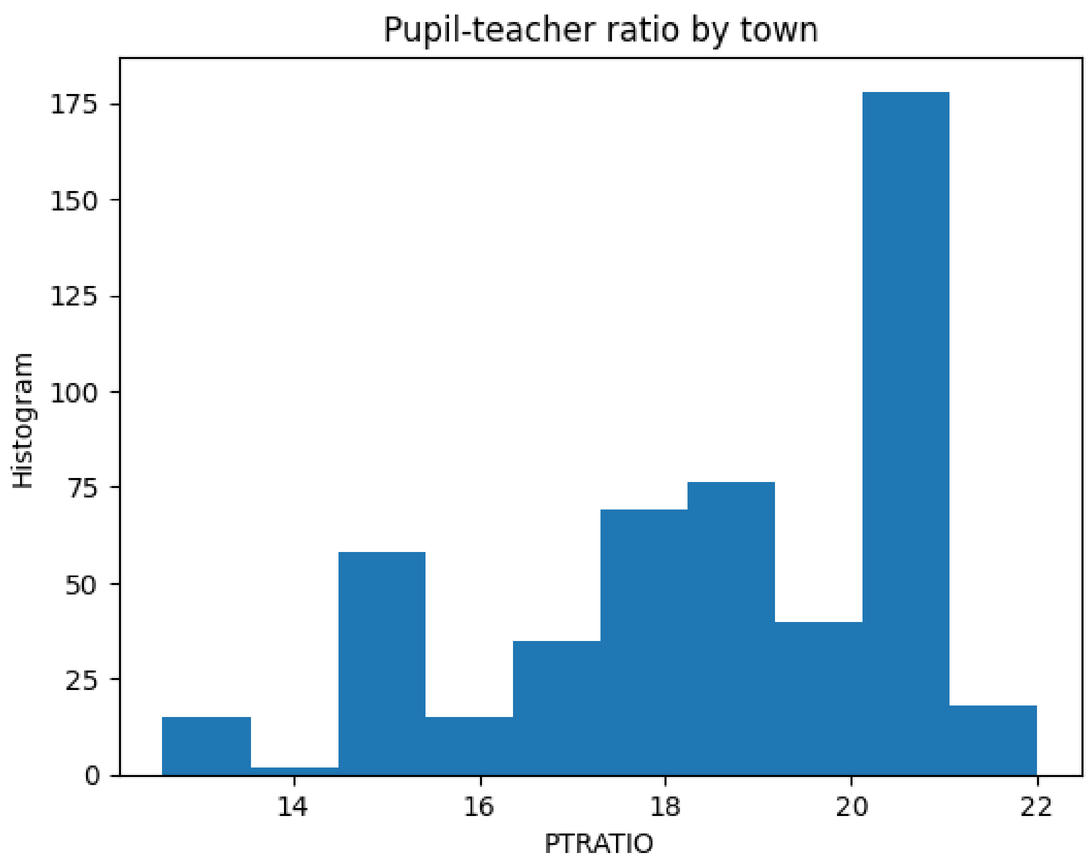
Scatter plot to show the relationship between NOX and INDUS :

```
import numpy as np
import matplotlib.pyplot as plt
plt.scatter(x='INDUS', y='NOX',data=boston_df)
plt.xlabel("Proportion of non retail business acres per town")
plt.ylabel("Nitric oxide concentration(parts per 10 million)")
plt.title("Scatter plot between NOX and INDUS")
plt.show()
```



Histogram for the pupil to teacher ratio variable :

```
import matplotlib.pyplot as plt
plt.hist('PTRATIO',data=boston_df)
plt.xlabel("PTRATIO")
plt.ylabel("Histogram")
plt.title("Pupil-teacher ratio by town")
plt.show()
```



TESTS

T-Test :

using the Boston housing data we need to check is there any significant difference in the median value of houses bounded by charles river or not

Hypothesis

- H0: m1=m2 (there is no difference in the median value of houses bounded by charles river)
- H1: m1!=m2 (there is difference in the median value of houses bounded by charles river)

we can use Levene's test in pyhton to check test significance

```
scipy.stats.levene(boston_df['MEDV'],boston_df['CHAS'],center='mean')
```

```
LeveneResult(statistic=532.6811164157666, pvalue=5.402535119732986e-95)
```

use the t test_ind from scipy_stats library

```
scipy.stats .ttest_ind(boston_df['MEDV'],boston_df['CHAS'])

TtestResult(statistic=54.9210289745203, pvalue=1.4651540072350996e-305, df=1010.0)
```

Conclusion : since the p_value is less than alpha value we reject the null hypothesis as there is a statistical difference in the median value of houses bounded by charles river

ANOVA:

Using the boston housing data ,is there a difference in the median value of houses for each proportion of owner occupied units built prior to 1940(AGE)

First we group the data into categories as the one way ANOVA cannot work with continuous variable

- 35 years and younger
- Between 35 and 70 years
- 70 years and older

```
boston_df.loc[(boston_df['AGE'] <= 35) , 'Age_Group' ] = "35 and younger"
boston_df.loc[(boston_df['AGE'] > 35) & (boston_df['AGE'] < 70) , 'Age_Group' ] = "Between 35 and 70"
boston_df.loc[(boston_df['AGE'] > 70) , 'Age_Group' ] = " 70 and older"
```

Hypothesis

- H0 : m1=m2=m3(" three population means are equal)
- H1 : atleast one of the means differ

Test for equality of variance

```
from scipy.stats import levene
scipy.stats.levene(boston_df[boston_df['Age_Group'] == "35 and younger"] ['MEDV'],
                   boston_df[boston_df['Age_Group'] == "Between 35 and 70"] ['MEDV'],
```

```
                   boston_df[boston_df['Age_Group'] == "70 and older"]
['MEDV'], center='mean')
```

```
LeveneResult(statistic=nan, pvalue=nan)
```

```
 thirtyfive_lower = boston_df[boston_df['Age_Group'] == "35 and 
younger"]['MEDV']
thirtyfive_seventy = boston_df[boston_df['Age_Group'] == "between 35 
and 70 years"]['MEDV']
seventy_older = boston_df[boston_df['Age_Group']=="70 and older"]
['MEDV']
```

Now run a one-way ANOVA

```
from scipy.stats import f_oneway
scipy.stats.f_oneway(thirtyfive_lower,thirtyfive_seventy,seventy_older
)
```

```
F_onewayResult(statistic=nan, pvalue=nan)
```

correlation:

using the boston housing data set we can conclude that there is no relationship between nitric oxide concentration and proportion of non retail business acres per town

Hypothesis

- H0: There is no relationship between nitric oxide concentration and proportion of non retail business acres per towm
- H1: there is a relationship between nitric oxide concentration and proportion of non retail business acres per town

```
scipy.stats.pearsonr(boston_df['INDUS'],boston_df['NOX'])
```

```
PearsonRResult(statistic=0.7636514469209192, 
pvalue=7.913361061210442e-98)
```

conclusion:

since the p-value is less than alpha value , we reject hte null hypothesis and conclude that there exists a relationship between nitric oxide concentration and the proportion of non retail business acres per town

Regression with T-test:

Using the boston housing dataset what is the impact of the additional weighted distance to the five Boston employment centres of the median value of owner occupied homes

Hypothesis

- H0 : B1 = 0(there is no impact of the additional weighed distance to the five boston employment centres of the median value of owner occupied homes)
- H1 :B1 is not equal to zero (there is an impact of additional weighed diatance to the five boston employment centres of the median value of owner occupied homes)

```
x=boston_df['DIS']
y=boston_df['MEDV']
x=sm.add_constant(x)
model=sm.OLS(y,x).fit()
predictions = model.predict(x)
model.summary()
```

```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results                            

==============================================================================
========
Dep. Variable:                   MEDV   R-squared:                       
0.062
Model:                            OLS   Adj. R-squared:                  
0.061
Method:                 Least Squares   F-statistic:                     
33.58
Date:                Mon, 03 Jun 2024   Prob (F-statistic):           
1.21e-08
Time:                        15:54:13   Log-Likelihood:                
-1823.9
No. Observations:                 506   AIC:                             
3652.
Df Residuals:                     504   BIC:                             
3660.
Df Model:                           1                                         

Covariance Type:            nonrobust                                         

==============================================================================
========
                 coef    std err          t      P>|t|      [0.025      
```

```
0.975]
------------------------------------------------------------------------------
--------
const         18.3901      0.817     22.499      0.000      16.784
19.996
DIS            1.0916      0.188      5.795      0.000       0.722
1.462
==============================================================================
========
Omnibus:                      139.779   Durbin-Watson:
0.570
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
305.104
Skew:                           1.466   Prob(JB):
5.59e-67
Kurtosis:                       5.424   Cond. No.
9.32
==============================================================================
========

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is
correctly specified.
"""
```