In [1]:

```
1
2 import numpy as np
3 import pandas as pd
4 import statsmodels.api as sm
5
6
```

In [2]:

```
1 data={"CIE":[90,95,98,92,97],
2       "SEE":[95,60,75,80,85]}
3 df=pd.DataFrame(data)
4 df
```

Out[2]:

	CIE	SEE
0	90	95
1	95	60
2	98	75
3	92	80
4	97	85

In [3]:

```
x=sm.add_constant(df['CIE'])
model=sm.OLS(df["SEE"],x).fit()
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    SEE   R-squared:                       0.227
Model:                            OLS   Adj. R-squared:                 -0.030
Method:                 Least Squares   F-statistic:                    0.8834
Date:                Tue, 07 Nov 2023   Prob (F-statistic):              0.417
Time:                        15:44:32   Log-Likelihood:                -18.694
No. Observations:                   5   AIC:                             41.39
Df Residuals:                       3   BIC:                             40.61
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const        252.3451    184.525      1.368      0.265    -334.895     839.585
CIE           -1.8363      1.954     -0.940      0.417      -8.054       4.381
==============================================================================
Omnibus:                        nan   Durbin-Watson:                   2.559
Prob(Omnibus):                  nan   Jarque-Bera (JB):                0.602
Skew:                        -0.761   Prob(JB):                        0.740
Kurtosis:                     2.244   Cond. No.                     2.97e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 2.97e+03. This might indicate that there are
strong multicollinearity or other numerical problems.

C:\Users\manju\anaconda3\Lib\site-packages\statsmodels\stats\stattools.py:74: ValueWarning: omni_normtest is not valid with less than 8 observations; 5 samples were given.
  warn("omni_normtest is not valid with less than 8 observations; %i "
```

In [4]:
```
import numpy as np
from sklearn.linear_model import LinearRegression
```

In [5]:
```
CIE=np.array([90,95,98,92,97]).reshape(-1,1)
SEE=np.array([95,60,75,80,85])
```

In [6]:
```
model = LinearRegression()
```

In [7]:
```
model.fit(CIE,SEE)
```

Out[7]:
```
LinearRegression()
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [8]:
```
print("intercept:",model.intercept_)
print("coefficient:",model.coef_[0])
```

```
intercept: 252.34513274336288
coefficient: -1.8362831858407083
```

In [10]:
```
import pandas as pd
data=pd.read_csv(r"C:\Users\manju\Desktop\HousingData.csv")
df=pd.DataFrame(data)
df
```

Out[10]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	I
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	
...	...	...	...	...	...	...	...	...	...	...	...	...	
501	0.06263	0.0	11.93	0.0	0.573	6.593	69.1	2.4786	1	273	21.0	391.99	
502	0.04527	0.0	11.93	0.0	0.573	6.120	76.7	2.2875	1	273	21.0	396.90	
503	0.06076	0.0	11.93	0.0	0.573	6.976	91.0	2.1675	1	273	21.0	396.90	
504	0.10959	0.0	11.93	0.0	0.573	6.794	89.3	2.3889	1	273	21.0	393.45	
505	0.04741	0.0	11.93	0.0	0.573	6.030	NaN	2.5050	1	273	21.0	396.90	

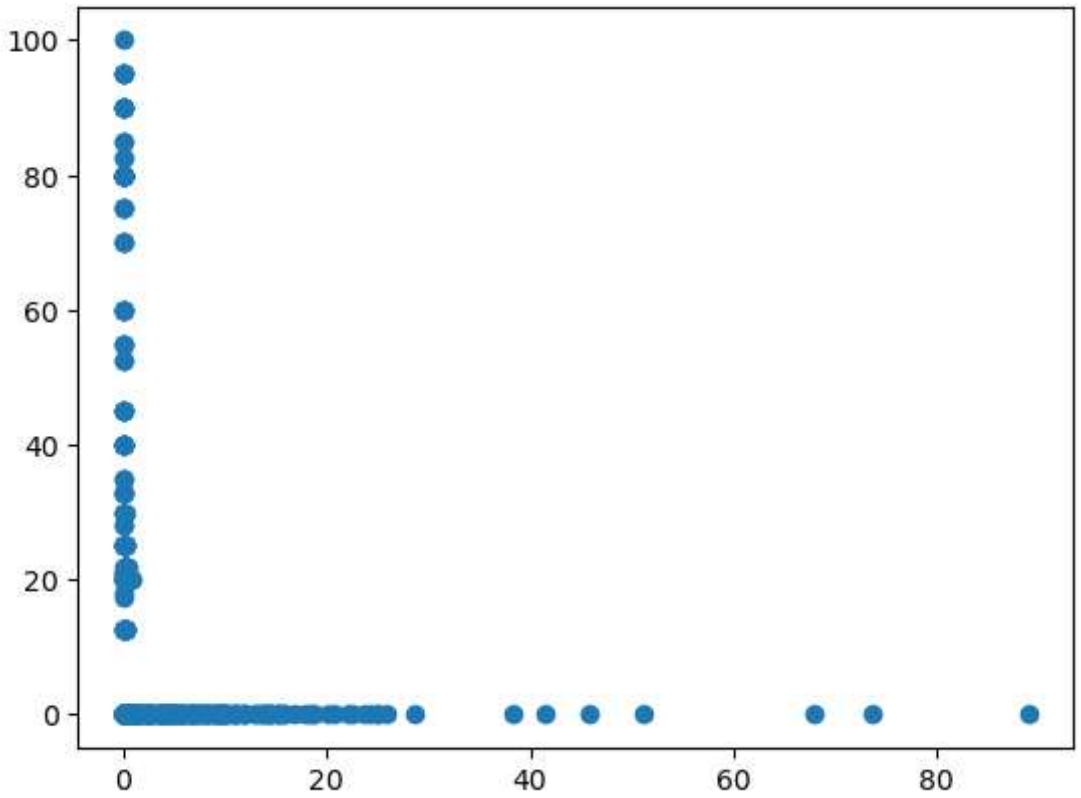
506 rows × 14 columns

In [11]:
```
import matplotlib.pyplot as plt
%matplotlib inline

```

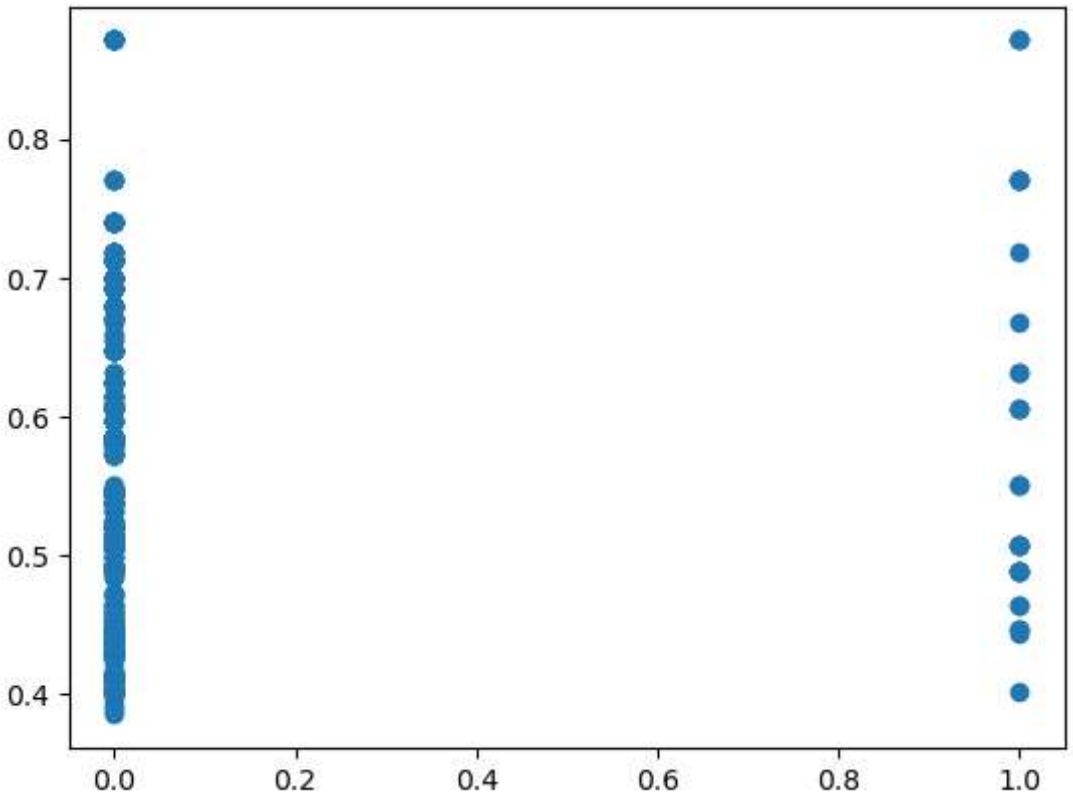
In [12]:
```
1 plt.scatter(df['CRIM'],df['ZN'])
```

Out[12]: <matplotlib.collections.PathCollection at 0x2c38f285110>



In [13]:
```
1 plt.scatter(df['CHAS'],df['NOX'])
```

Out[13]: <matplotlib.collections.PathCollection at 0x2c38f625410>



In [14]:
```
x = df[['CRIM','ZN']]
y = df['INDUS']
```

In [16]:
```
from sklearn.model_selection import train_test_split
```

In [24]:
```
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=0.2)
```

In [25]:
```
x_train
```

Out[25]:

	CRIM	ZN
306	0.07503	33.0
175	0.06664	0.0
427	37.66190	NaN
58	0.15445	25.0
17	0.78420	0.0
...	...	...
179	0.05780	0.0
501	0.06263	0.0
98	0.08187	0.0
258	0.66351	20.0
403	24.80170	0.0

404 rows × 2 columns

In [26]:
```
x_test
```

Out[26]:

	CRIM	ZN
274	0.05644	40.0
64	0.01951	17.5
53	NaN	21.0
337	0.03041	0.0
407	11.95110	0.0
...	...	...
184	0.08308	0.0
430	NaN	0.0
133	0.32982	NaN
68	0.13554	12.5
146	2.15505	NaN

102 rows × 2 columns

In [27]:
```
1 y_train
```

Out[27]:
```
306     2.18
175     4.05
427    18.10
58      5.13
17      8.14
       ...
179     2.46
501    11.93
98      2.89
258     3.97
403    18.10
Name: INDUS, Length: 404, dtype: float64
```

In [28]:
```
1 y_test
```

Out[28]:
```
274     6.41
64      1.38
53      5.64
337     5.19
407    18.10
       ...
184     2.46
430    18.10
133      NaN
68      6.07
146    19.58
Name: INDUS, Length: 102, dtype: float64
```

In [29]:
```
1 len(x_test)
```

Out[29]:
```
102
```

In [31]:
```
1 len(y_test)
```

Out[31]:
```
102
```

In [ ]:
```
1 len()
```