Prepare a prediction model for profit of 50_startups data. Do transformations for getting better predictions of profit and make a table containing R^2 value for each prepared model.

R&D Spend -- Research and develop spend in the past few years
Administration -- spend on administration in the past few years
Marketing Spend -- spend on Marketing in the past few years
State -- states from which data is collected
Profit -- profit of each state in the past few years

—--------------Import Important Libraries—-----------------------

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.formula.api as smf
from statsmodels.graphics.regressionplots import influence_plot
import statsmodels.api as sm
```

—-------------Read the Dataset—--

```
data = pd.read_csv ('Downloads/50_Startups.csv')
data
```

	R&D Spend	Administration	Marketing Spend	State	Profit
0	165349.20	136897.80	471784.10	New York	192261.83
1	162597.70	151377.59	443898.53	California	191792.06
2	153441.51	101145.55	407934.54	Florida	191050.39
3	144372.41	118671.85	383199.62	New York	182901.99
4	142107.34	91391.77	366168.42	Florida	166187.94
5	131876.90	99814.71	362861.36	New York	156991.12
6	134615.46	147198.87	127716.82	California	156122.51
7	130298.13	145530.06	323876.68	Florida	155752.60
8	120542.52	148718.95	311613.29	New York	152211.77
9	123334.88	108679.17	304981.62	California	149759.96
10	101913.08	110594.11	229160.95	Florida	146121.95
11	100671.96	91790.61	249744.55	California	144259.40
12	93863.75	127320.38	249839.44	Florida	141585.52
13	91992.39	135495.07	252664.93	California	134307.35
14	119943.24	156547.42	256512.92	Florida	132602.65
15	114523.61	122616.84	261776.23	New York	129917.04
16	78013.11	121597.55	264346.06	California	126992.93
17	94657.16	145077.58	282574.31	New York	125370.37
18	91749.16	114175.79	294919.57	Florida	124266.90
19	86419.70	153514.11	0.00	New York	122776.86
20	76253.86	113867.30	298664.47	California	118474.03

—----------------Provide information about the dataset—-----------------
data.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50 entries, 0 to 49
Data columns (total 5 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   R&D Spend        50 non-null     float64
 1   Administration   50 non-null     float64
 2   Marketing Spend  50 non-null     float64
 3   State            50 non-null     object
 4   Profit           50 non-null     float64
dtypes: float64(4), object(1)
memory usage: 2.1+ KB
```

—-----------------Rename the Column names—------------------------------
rename = data.rename ({'R&D Spend':'RD','Administration':'Admin','Marketing Spend':'Market','State':'State','Profit':'Profit'},axis=1)
rename

	RD	Admin	Market	State	Profit
0	165349.20	136897.80	471784.10	New York	192261.83
1	162597.70	151377.59	443898.53	California	191792.06
2	153441.51	101145.55	407934.54	Florida	191050.39
3	144372.41	118671.85	383199.62	New York	182901.99
4	142107.34	91391.77	366168.42	Florida	166187.94
5	131876.90	99814.71	362861.36	New York	156991.12
6	134615.46	147198.87	127716.82	California	156122.51
7	130298.13	145530.06	323876.68	Florida	155752.60
8	120542.52	148718.95	311613.29	New York	152211.77
9	123334.88	108679.17	304981.62	California	149759.96
10	101913.08	110594.11	229160.95	Florida	146121.95
11	100671.96	91790.61	249744.55	California	144259.40
12	93863.75	127320.38	249839.44	Florida	141585.52
13	91992.39	135495.07	252664.93	California	134307.35
14	119943.24	156547.42	256512.92	Florida	132602.65
15	114523.61	122616.84	261776.23	New York	129917.04
16	78013.11	121597.55	264346.06	California	126992.93
17	94657.16	145077.58	282574.31	New York	125370.37
18	91749.16	114175.79	294919.57	Florida	124266.90
19	86419.70	153514.11	0.00	New York	122776.86
20	76253.86	113867.30	298664.47	California	118474.03

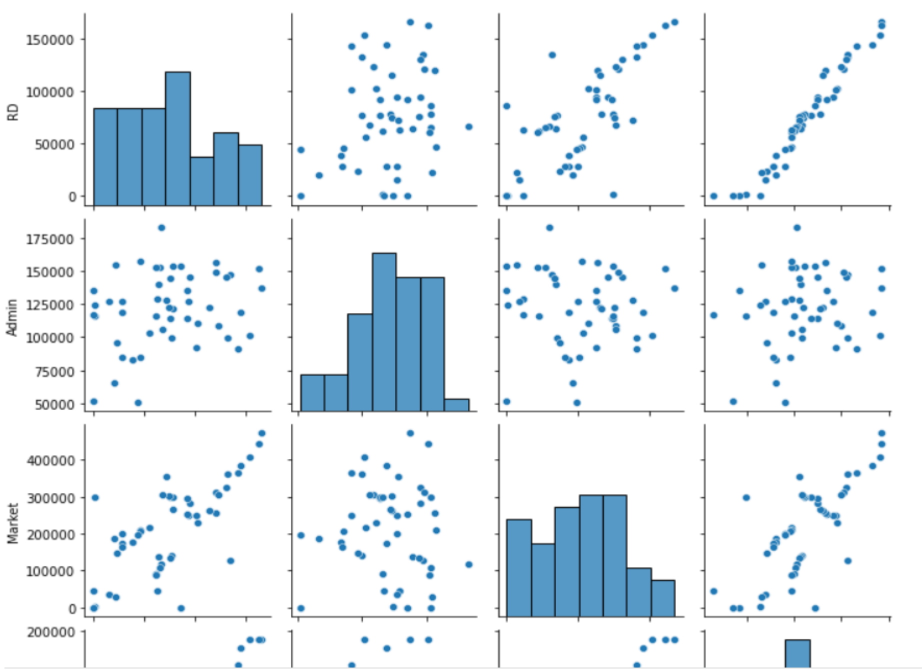
—--------------------Data Description—--------------------------------
rename.describe()

	RD	Admin	Market	Profit
count	50.000000	50.000000	50.000000	50.000000
mean	73721.615600	121344.639600	211025.097800	112012.639200
std	45902.256482	28017.802755	122290.310726	40306.180338
min	0.000000	51283.140000	0.000000	14681.400000
25%	39936.370000	103730.875000	129300.132500	90138.902500
50%	73051.080000	122699.795000	212716.240000	107978.190000
75%	101602.800000	144842.180000	299469.085000	139765.977500
max	165349.200000	182645.560000	471784.100000	192261.830000

—-----------------Correlation Analysis—------------------------------
rename.corr()

	RD	Admin	Market	Profit
RD	1.000000	0.241955	0.724248	0.972900
Admin	0.241955	1.000000	-0.032154	0.200717
Market	0.724248	-0.032154	1.000000	0.747766
Profit	0.972900	0.200717	0.747766	1.000000

—-------------------Pairplot—--------------------------------------
sns.pairplot(rename)



—-----------------Model Building—----------------------------------
import statsmodels.formula.api as smf
model = smf.ols ('Profit~Admin+Market+State+RD', data = rename).fit()

—--------------------Calculate parameters—----------------------------------
model.params

```
Intercept              50125.343832
State[T.Florida]         198.788793
State[T.New York]        -41.887019
Admin                     -0.027004
Market                     0.026980
RD                         0.806023
dtype: float64
```

—-----------------Find T Values and P Values—------------------------
print (model.tvalues, '\n', model.pvalues)

```
Intercept             7.280560
State[T.Florida]      0.058970
State[T.New York]    -0.012864
Admin                -0.517012
Market                1.573889
RD                   17.368580
dtype: float64
 Intercept            4.444178e-09
State[T.Florida]     9.532429e-01
State[T.New York]    9.897941e-01
Admin                6.077373e-01
Market               1.226769e-01
RD                   2.578772e-21
dtype: float64
```

—------------------Calculate Rcsquared values—-------------------
print (model.rsquared, model.rsquared_adj)

```
0.9507524843355148 0.945156175737278
```

—----------Calculate T Values and P values for [Profit,Admin]-------------------
mlv = smf.ols ('Profit~Admin', data = rename).fit()
print (mlv.tvalues, '\n', mlv.pvalues)

```
Intercept    3.040044
Admin        1.419493
dtype: float64
 Intercept    0.003824
Admin        0.162217
dtype: float64
```

—--------Calculate T values and P values for [Profit,Market]------------------
nlv = smf.ols ('Profit~Market', data = rename).fit()
print (nlv.tvalues, '\n', nlv.pvalues)

```
Intercept    7.808356
Market       7.802657
dtype: float64
 Intercept    4.294735e-10
Market       4.381073e-10
dtype: float64
```

—-------Calculate T values and P values for [Profit,Admin,Market]----------------
hlv = smf.ols ('Profit~Admin+Market', data = rename).fit()
print (hlv.tvalues, '\n', hlv.pvalues)

```
Intercept    1.142741
Admin        2.467779
Market       8.281039
dtype: float64
 Intercept    2.589341e-01
Admin        1.729198e-02
Market       9.727245e-11
dtype: float64
```

—------------Calculate R^2 value for [RD,Admin,Market]------------------
mllv = smf.ols ('RD~Admin+Market', data = rename).fit().rsquared
m = 1/(1-mllv)
m

```
2.4689030699947017
```

—-----------Calculate R^2 value for [Admin,RD,Market]----------------
nllv = smf.ols ('Admin~RD+Market', data = rename).fit().rsquared
n = 1/(1-nllv)
n

```
1.1750910070550458
```

—-------------Calculate R^2 value for [Market,RD,Admin]-------------------
hllv = smf.ols ('Market~RD+Admin', data = rename).fit().rsquared
h = 1/(1-hllv)
h

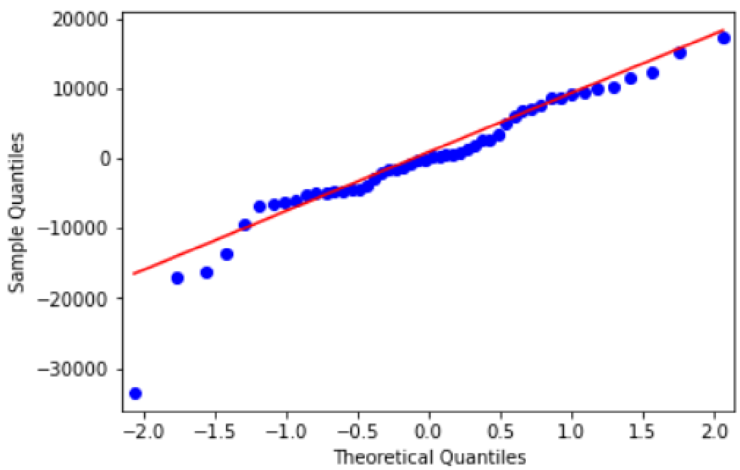
```
2.3267732905308773
```

—--------------Put the values in DataFrame—-------------------
list = {'Variables':['RD','Admin','Market'],'values':[m,n,h]}
simple = pd.DataFrame (list)
simple

	Variables	values
0	RD	2.468903
1	Admin	1.175091
2	Market	2.326773

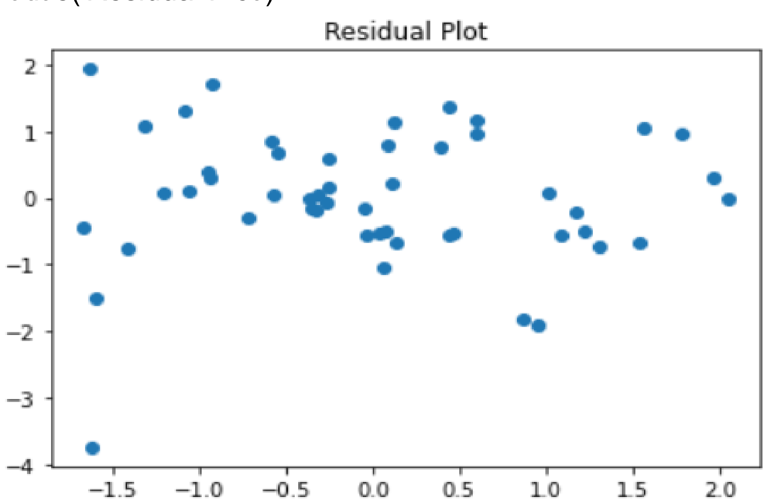
—----------------Plotting QQPLOT—------------------------------

```
import statsmodels.api as sm
qqplot = sm.qqplot (model.resid, line = 'q')
fig = plt.figure (figsize = (16,8))
```



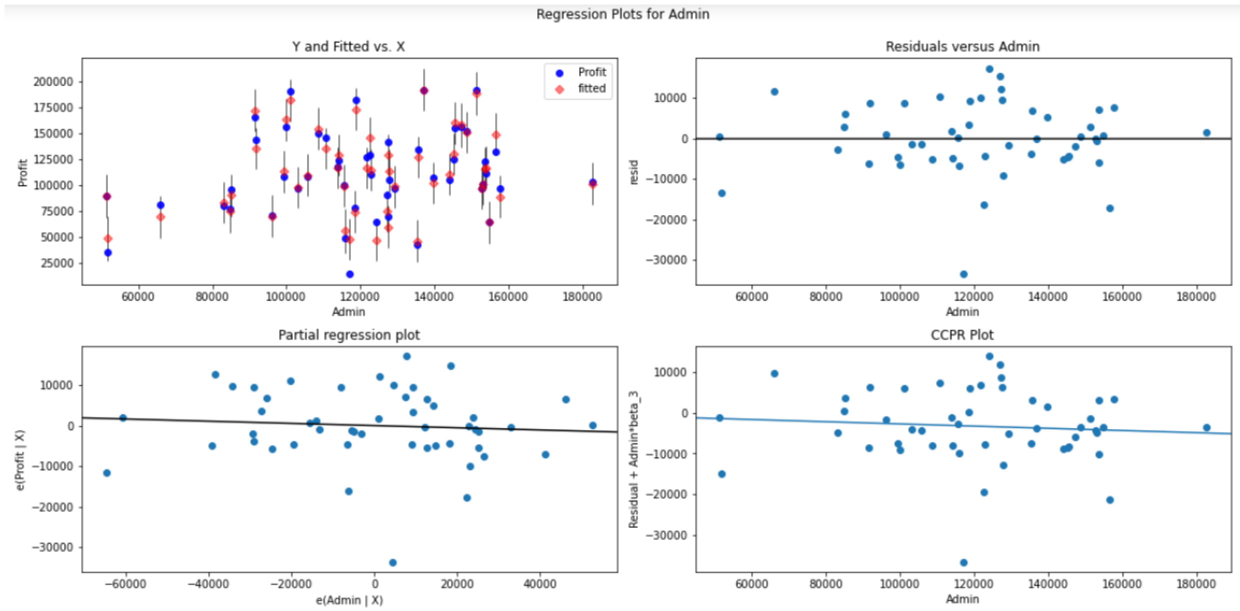
—-------------------Plot Residual Plot—-----------------------------

```
def standard_values(vals) : return (vals - vals.mean())/vals.std()
plt.scatter (standard_values(model.fittedvalues), standard_values (model.resid))
plt.title('Residual Plot')
```



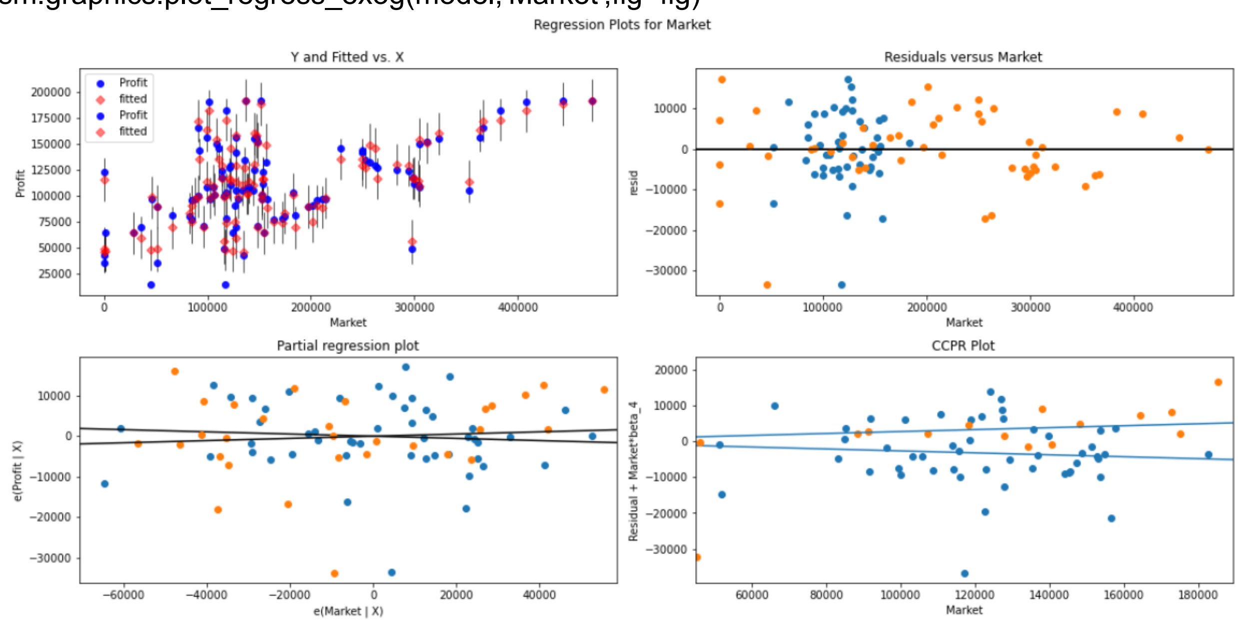
—------------------Regression Plots for [Admin]-----------------------------

```
import statsmodels.formula.api as smf
import statsmodels.api as sm
from statsmodels.graphics.regressionplots import influence_plot
sm.graphics.plot_regress_exog(model,'Admin',fig=fig)
```



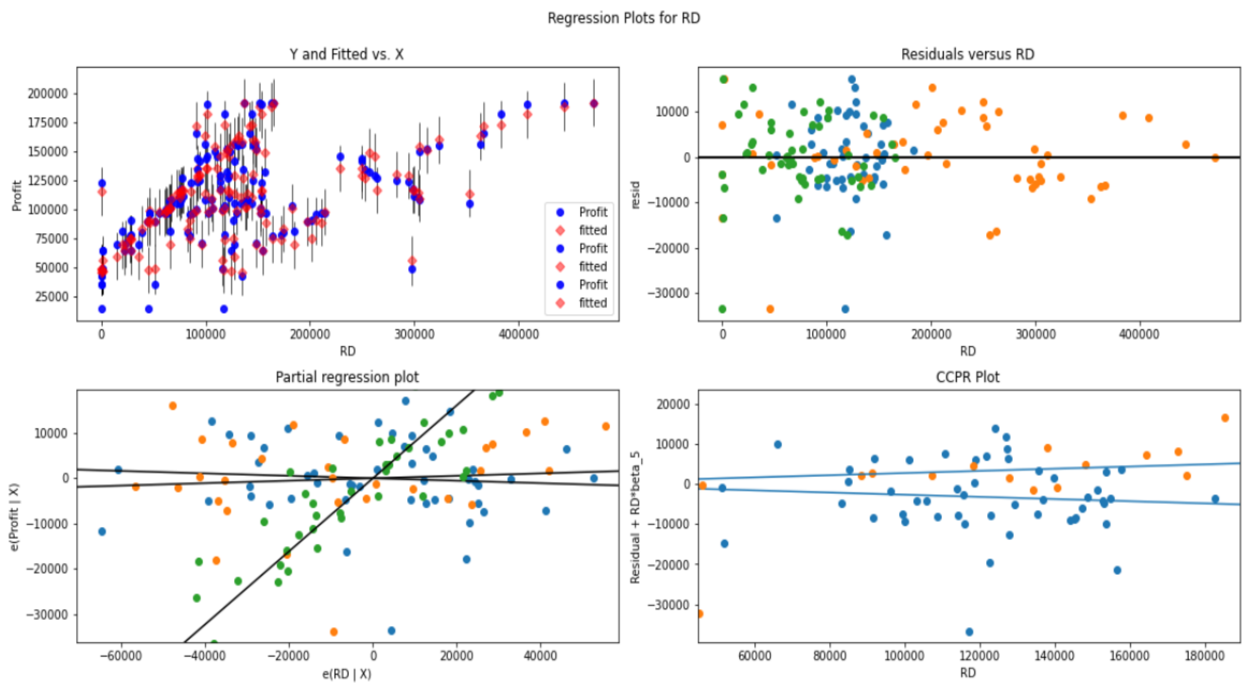
—------------------Regression Plots for [Market]-----------------------------

```
import statsmodels.api as sm
import statsmodels.formula.api as smf
from statsmodels.graphics.regressionplots import influence_plot
sm.graphics.plot_regress_exog(model,'Market',fig=fig)
```



—------------------Regression Plots for [RD]----------------------------

```
import statsmodels.api as sm
import statsmodels.formula.api as smf
from statsmodels.graphics.regressionplots import influence_plot
sm.graphics.plot_regress_exog(model,'RD',fig=fig)
```

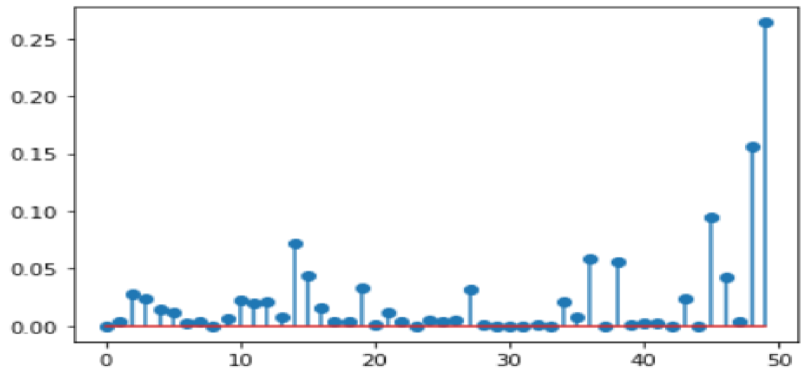


—-----------------Implement Cooks Distance—------------------

```
from statsmodels.graphics.regressionplots import influence_plot
model_influence = model.get_influence()
(c,_) = model_influence.cooks_distance
c
```

```
array([7.67941285e-06, 3.96002384e-03, 2.78948395e-02, 2.35705108e-02,
       1.40231490e-02, 1.17098970e-02, 2.49314176e-03, 4.16542624e-03,
       7.29467176e-05, 6.31415598e-03, 2.21391699e-02, 1.93512168e-02,
       2.13263552e-02, 7.40092001e-03, 7.20165958e-02, 4.34157410e-02,
       1.57591120e-02, 4.33058862e-03, 3.43997076e-03, 3.28909738e-02,
       7.03247647e-04, 1.17002661e-02, 3.52541534e-03, 3.68801928e-04,
       5.07030667e-03, 4.16365620e-03, 5.79414020e-03, 3.25030423e-02,
       1.07438091e-03, 1.14685871e-04, 2.67092819e-05, 4.26003187e-06,
       6.55180125e-04, 2.69550649e-04, 2.09894518e-02, 8.32171521e-03,
       5.92471519e-02, 7.19280439e-05, 5.58017593e-02, 1.60830329e-03,
       2.27122555e-03, 2.19513492e-03, 1.66164967e-04, 2.33988898e-02,
       1.16697070e-04, 9.43947846e-02, 4.23233340e-02, 4.48153392e-03,
       1.56376134e-01, 2.63959436e-01])
```

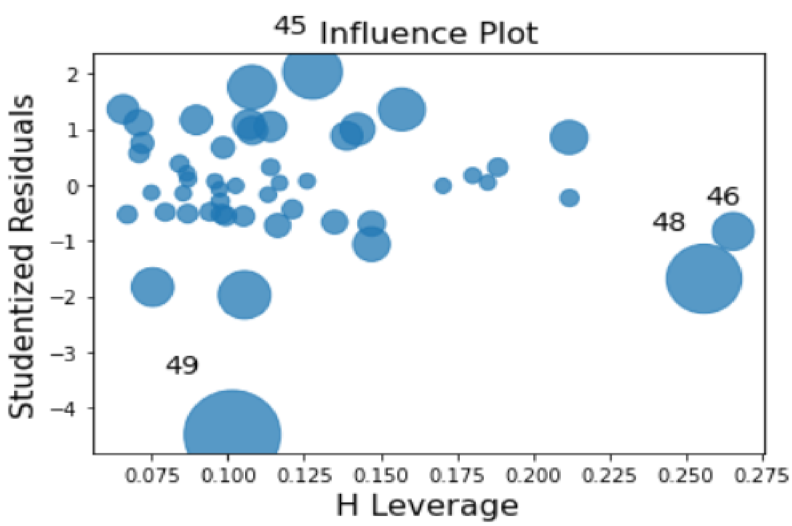
—-----------------Use Stem Plot for Cooks distance Visualization—-------------
plt.stem(np.arange(len(data)),np.round(c,5))
<StemContainer object of 3 artists>



—----------Index and values of Influencer—--------------------------
np.argmax(c), np.max(c)
(49, 0.2639594358711302)

In [ ]:

—-----------------Influence Plot—---------------------------
influence_plot(model)



—-------------Leverage Cutoff Value—---------------------------
k = data.shape[1]
n = data.shape[0]
leverage = (3*(k+1))/n
leverage

0.36

In [ ]:

—-----------------Look for 49th Row—----------------------------
data[data.index.isin([49])]

	R&D Spend	Administration	Marketing Spend	State	Profit
49	0.0	116983.8	45173.06	California	14681.4

—-----------Drop 49th Row—----------------------
drop = data.drop(data.index[[49]],axis=0).reset_index(drop=True)
drop

26	75328.87	144135.98	134050.07	Florida	105733.54
27	72107.60	127864.55	353183.81	New York	105008.31
28	66051.52	182645.56	118148.20	Florida	103282.38
29	65605.48	153032.06	107138.38	New York	101004.64
30	61994.48	115641.28	91131.24	Florida	99937.59
31	61136.38	152701.92	88218.23	New York	97483.56
32	63408.86	129219.61	46085.25	California	97427.84
33	55493.95	103057.49	214634.81	Florida	96778.92
34	46426.07	157693.92	210797.67	California	96712.80
35	46014.02	85047.44	205517.64	New York	96479.51
36	28663.76	127056.21	201126.82	Florida	90708.19
37	44069.95	51283.14	197029.42	California	89949.14
38	20229.59	65947.93	185265.10	New York	81229.06
39	38558.51	82982.09	174999.30	California	81005.76
40	28754.33	118546.05	172795.67	California	78239.91
41	27892.92	84710.77	164470.71	Florida	77798.83
42	23640.93	96189.63	148001.11	California	71498.49
43	15505.73	127382.30	35534.17	New York	69758.98
44	22177.74	154806.14	28334.72	California	65200.33
45	1000.23	124153.04	1903.93	New York	64926.08
46	1315.46	115816.21	297114.46	Florida	49490.75
47	0.00	135426.92	0.00	California	42559.73
48	542.05	51743.15	0.00	New York	35673.41

—----------------Model Detection—-------------------------------
```
while np.max(c)>0.5:
    model = smf.ols('Profit~RD+Admin+Market', data=data2).fit()
    (c,_) = model_influence.cooks_distance
    c
    np.argmax(c),np.max(c)
    data2=data2.drop(data2.index[[np.agrmax(c)]],axis=0).reset_index(drop=True)
    data2
else:
    final_model=smf.ols("Profit~RD+Admin+Market",data=data2).fit()
    final_model.rsquared , final_model.aic
    print("Thus model accuracy is improved to",final_model.rsquared)
```

Thus model accuracy is improved to 0.9613162435129847

In [ ]:

—-------------------Final Results—----------------------------

```
list = {'Names':['model','final_model'],'Values':[model.rsquared_adj,final_model.rsquared]}
simple=pd.DataFrame(list)
simple
```

	Names	Values
0	model	0.958737
1	final_model	0.961316