Multi Linear Regression

------- 50_Startups Problem

In [1]:

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

from sklearn.preprocessing import LabelEncoder
```

In [2]:

```
startup_data = pd.read_csv('50_Startups.csv')
startup_data
```

Out[2]:

	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
21	78389.47	153773.43	299737.29	New York	111313.02
22	73994.56	122782.75	303319.26	Florida	110352.25
23	67532.53	105751.03	304768.73	Florida	108733.99
24	77044.01	99281.34	140574.81	New York	108552.04
25	64664.71	139553.16	137962.62	California	107404.34
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

	R&D Spend	Administration	Marketing Spend	State	Profit
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
49	0.00	116983.80	45173.06	California	14681.40

In [4]:

```
startup_data.head()
```

Out[4]:

	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

In [5]:

```
startup_data.isna().sum()
```

Out[5]:

```
R&D Spend          0
Administration     0
Marketing Spend    0
State              0
Profit             0
dtype: int64
```

In [6]:

```
1 startup_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
```

In [3]:

```
le = LabelEncoder()
startup_data['State'] = le.fit_transform(startup_data['State'])

startup_data
```

Out[3]:

	R&D Spend	Administration	Marketing Spend	State	Profit
0	165349.20	136897.80	471784.10	2	192261.83
1	162597.70	151377.59	443898.53	0	191792.06
2	153441.51	101145.55	407934.54	1	191050.39
3	144372.41	118671.85	383199.62	2	182901.99
4	142107.34	91391.77	366168.42	1	166187.94
5	131876.90	99814.71	362861.36	2	156991.12
6	134615.46	147198.87	127716.82	0	156122.51
7	130298.13	145530.06	323876.68	1	155752.60
8	120542.52	148718.95	311613.29	2	152211.77
9	123334.88	108679.17	304981.62	0	149759.96
10	101913.08	110594.11	229160.95	1	146121.95
11	100671.96	91790.61	249744.55	0	144259.40
12	93863.75	127320.38	249839.44	1	141585.52
13	91992.39	135495.07	252664.93	0	134307.35
14	119943.24	156547.42	256512.92	1	132602.65
15	114523.61	122616.84	261776.23	2	129917.04
16	78013.11	121597.55	264346.06	0	126992.93
17	94657.16	145077.58	282574.31	2	125370.37
18	91749.16	114175.79	294919.57	1	124266.90
19	86419.70	153514.11	0.00	2	122776.86
20	76253.86	113867.30	298664.47	0	118474.03
21	78389.47	153773.43	299737.29	2	111313.02
22	73994.56	122782.75	303319.26	1	110352.25
23	67532.53	105751.03	304768.73	1	108733.99
24	77044.01	99281.34	140574.81	2	108552.04
25	64664.71	139553.16	137962.62	0	107404.34
26	75328.87	144135.98	134050.07	1	105733.54
27	72107.60	127864.55	353183.81	2	105008.31
28	66051.52	182645.56	118148.20	1	103282.38
29	65605.48	153032.06	107138.38	2	101004.64
30	61994.48	115641.28	91131.24	1	99937.59
31	61136.38	152701.92	88218.23	2	97483.56
32	63408.86	129219.61	46085.25	0	97427.84

	R&D Spend	Administration	Marketing Spend	State	Profit
33	55493.95	103057.49	214634.81	1	96778.92
34	46426.07	157693.92	210797.67	0	96712.80
35	46014.02	85047.44	205517.64	2	96479.51
36	28663.76	127056.21	201126.82	1	90708.19
37	44069.95	51283.14	197029.42	0	89949.14
38	20229.59	65947.93	185265.10	2	81229.06
39	38558.51	82982.09	174999.30	0	81005.76
40	28754.33	118546.05	172795.67	0	78239.91
41	27892.92	84710.77	164470.71	1	77798.83
42	23640.93	96189.63	148001.11	0	71498.49
43	15505.73	127382.30	35534.17	2	69758.98
44	22177.74	154806.14	28334.72	0	65200.33
45	1000.23	124153.04	1903.93	2	64926.08
46	1315.46	115816.21	297114.46	1	49490.75
47	0.00	135426.92	0.00	0	42559.73
48	542.05	51743.15	0.00	2	35673.41
49	0.00	116983.80	45173.06	0	14681.40

In [4]:

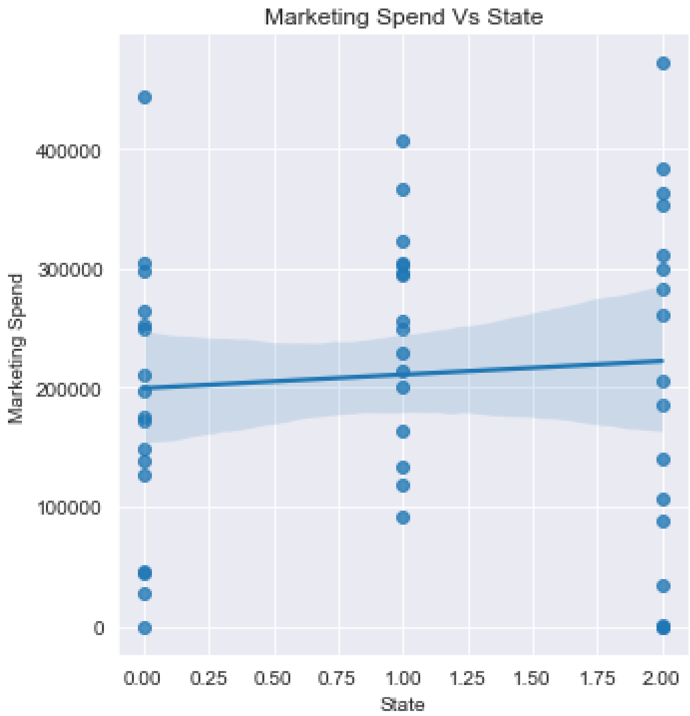
```
startup_data.info()
```

```
<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     int32  
 4   Profit           50 non-null     float64
dtypes: float64(4), int32(1)
memory usage: 1.9 KB
```

Linearity Check

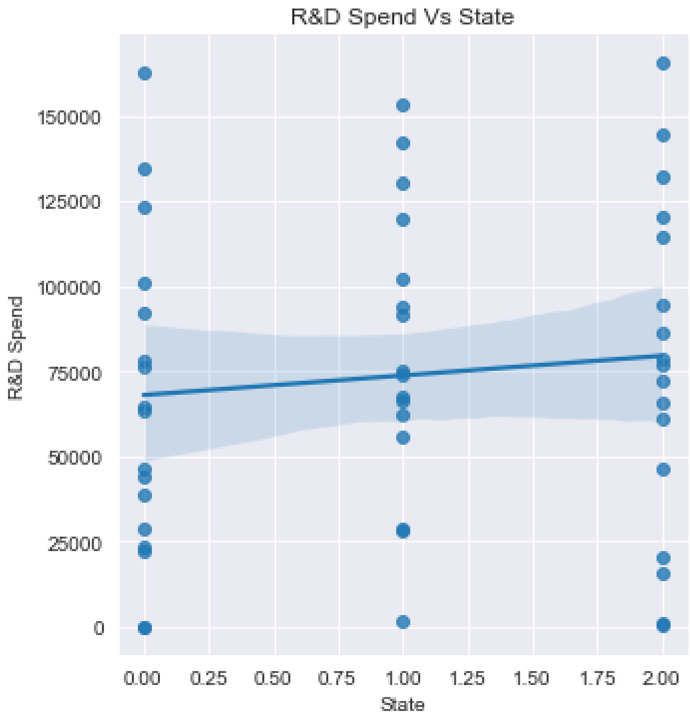
In [65]:

```
sns.lmplot(y='Marketing Spend',x='State',data=startup_data)
plt.title('Marketing Spend Vs State')
plt.show()
```



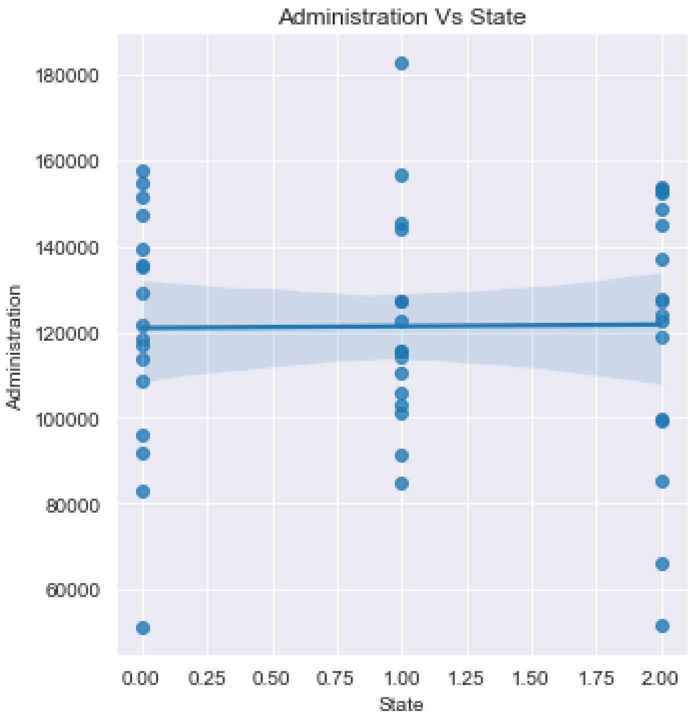
In [66]:

```
sns.lmplot(y='R&D Spend',x='State',data=startup_data)
plt.title('R&D Spend Vs State')
plt.show()
```



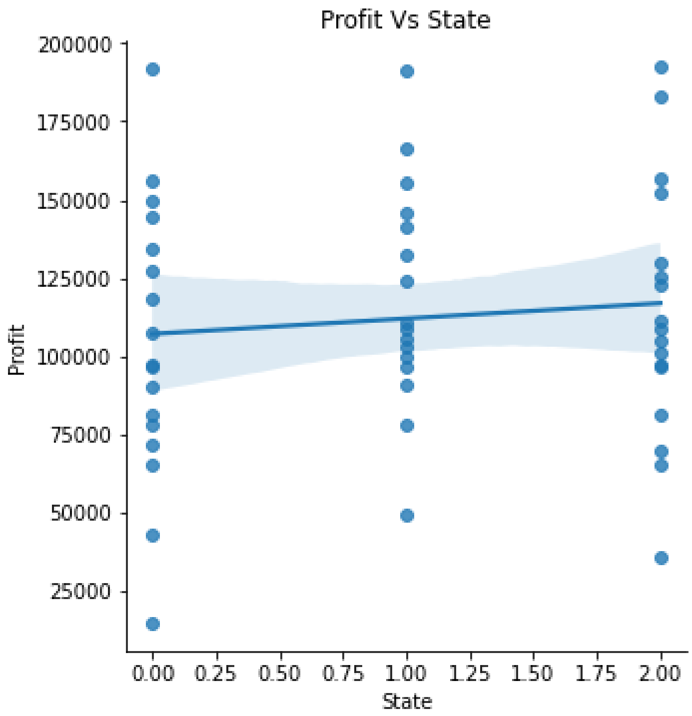
In [67]:

```
sns.lmplot(y='Administration',x='State',data=startup_data)
plt.title('Administration Vs State')
plt.show()
```



In [50]:

```
sns.lmplot(y='Profit',x='State',data=startup_data)
plt.title('Profit Vs State')
plt.show()
```

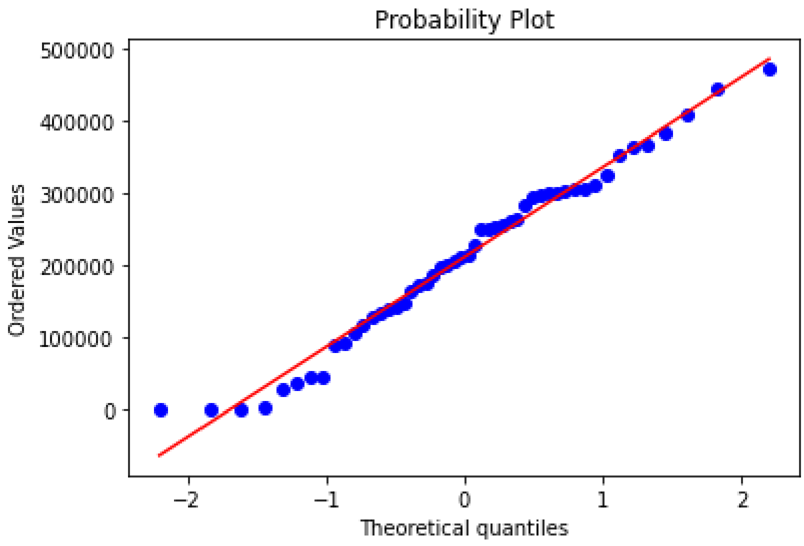


Linearity test failed

Normality Check

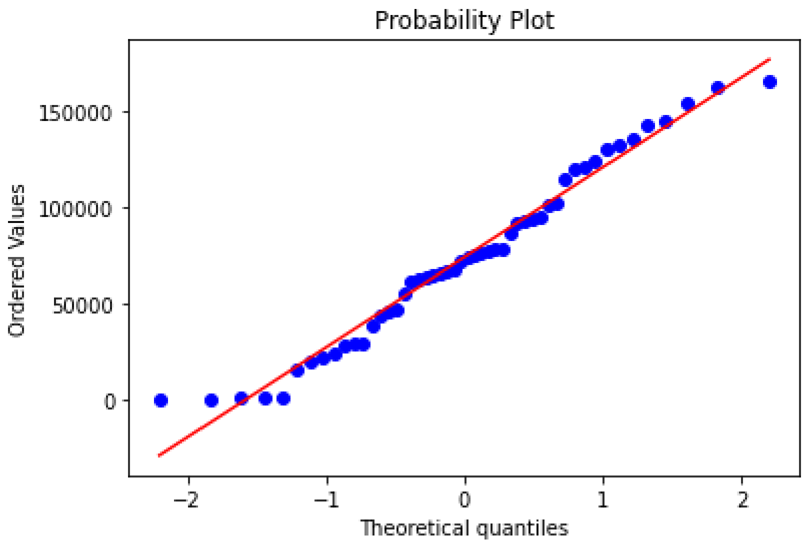
In [10]:

```
from scipy import stats
stats.probplot(x = startup_data['Marketing Spend'],dist='norm',plot=plt)
plt.show()
```



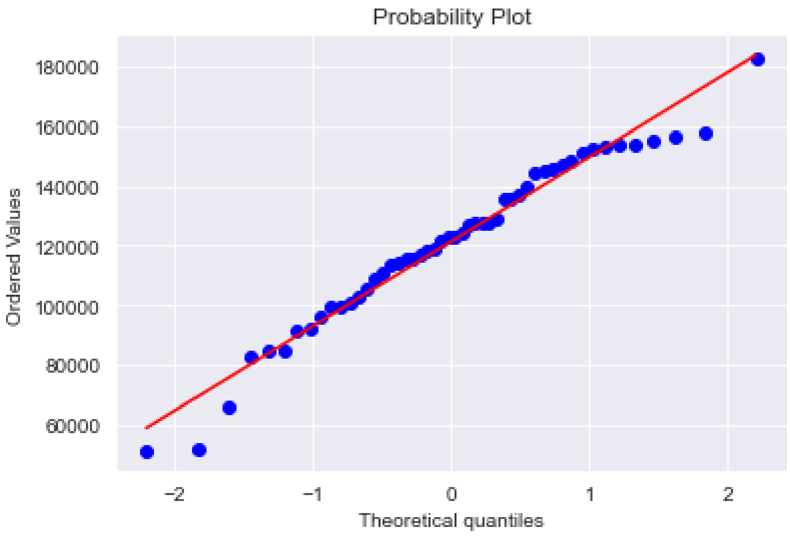
In [12]:

```
from scipy import stats
stats.probplot(x = startup_data['R&D Spend'],dist='norm',plot=plt)
plt.show()
```



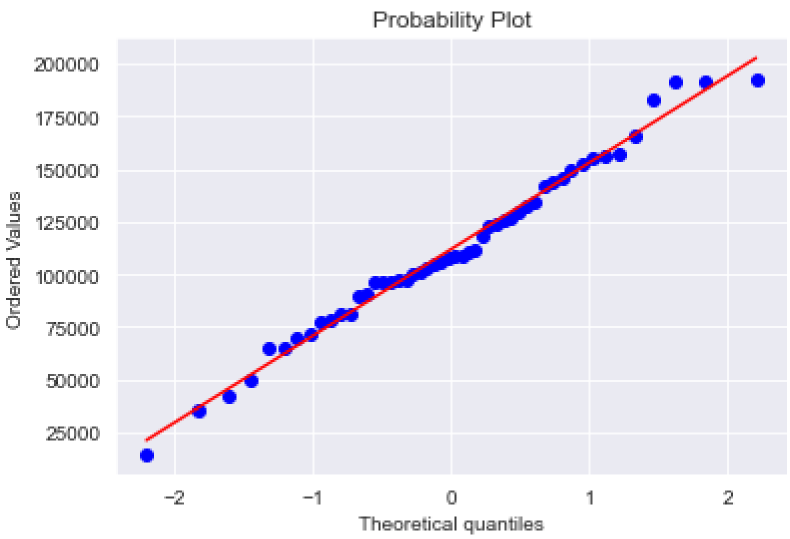
In [35]:

```
from scipy import stats
stats.probplot(x = startup_data['Administration'],dist='norm',plot=plt)
plt.show()
```



In [64]:

```
from scipy import stats
stats.probplot(x = startup_data['Profit'],dist='norm',plot=plt)
plt.show()
```



Normality test is passed

Multicolinearity Test

In [14]:

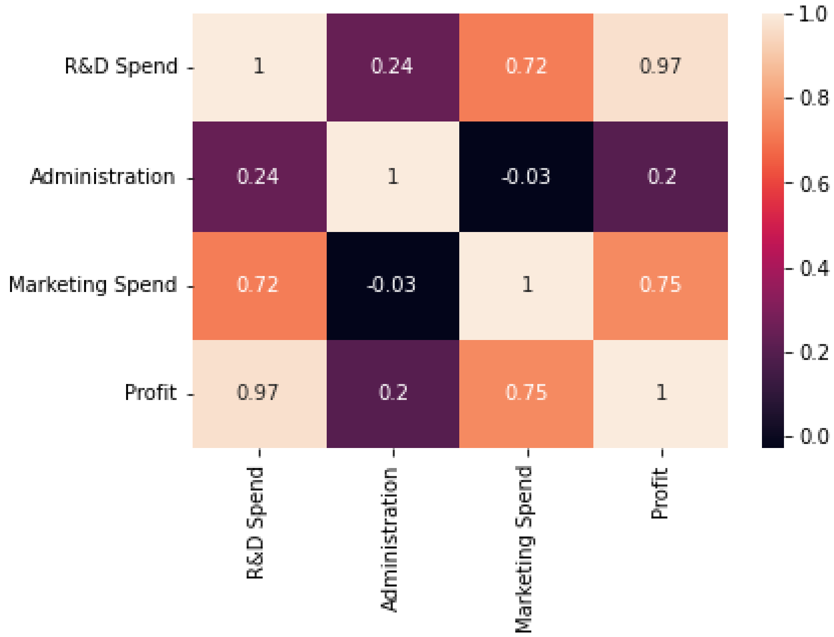
```
startup_corr = startup_data.corr().round(2)
startup_corr
```

Out[14]:

	R&D Spend	Administration	Marketing Spend	Profit
R&D Spend	1.00	0.24	0.72	0.97
Administration	0.24	1.00	-0.03	0.20
Marketing Spend	0.72	-0.03	1.00	0.75
Profit	0.97	0.20	0.75	1.00

In [15]:

```
sns.heatmap(startup_corr,annot=True)
plt.show()
```



There is multicoinearity in this data therefore Multicolinearity test is failed

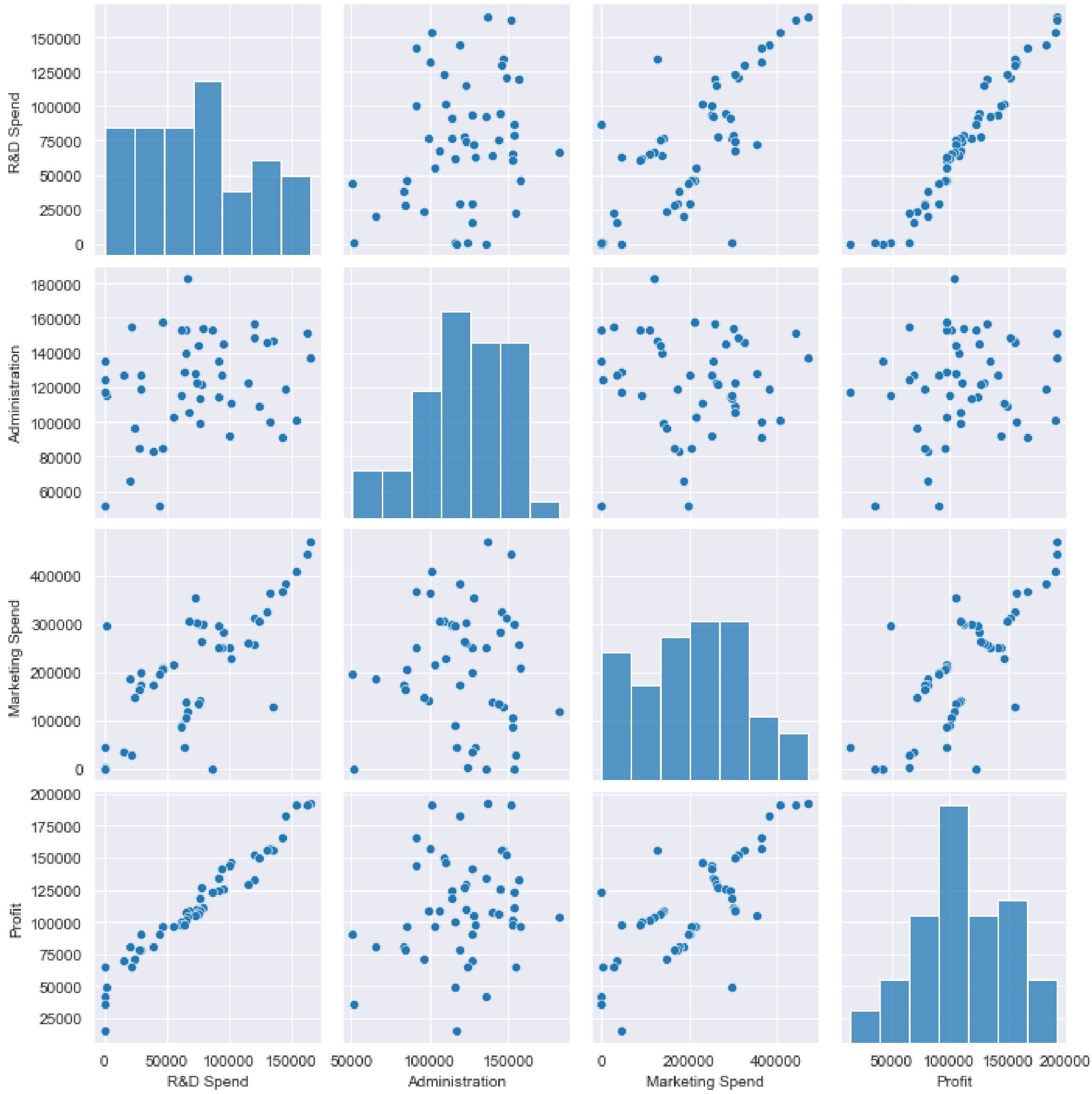
No auto regression in this data

In [16]:

```
sns.set_style(style='darkgrid')
sns.pairplot(startup_data)
```

Out[16]:

<seaborn.axisgrid.PairGrid at 0x1c0d5957ca0>



In [51]:

```
X = startup_data.drop('Profit',axis=1)
y = startup_data[['Profit']]
```

In [52]:

```
from sklearn.linear_model import LinearRegression
linear_model = LinearRegression()
linear_model.fit(X,y)
```

Out[52]:

```
LinearRegression()
```

In [53]:

```
linear_model.coef_
```

Out[53]:

```
array([[  0.80575968,  -0.02682585,   0.02722767, -22.32057723]])
```

In [54]:

```
linear_model.intercept_
```

Out[54]:

```
array([50142.50644348])
```

In [55]:

```
y_pred = linear_model.predict(X)
```

In [56]:

```
error = y - y_pred
error
```

Out[56]:

	Profit
0	-240.934416
1	2609.393955
2	8899.431581
3	9224.499382
4	-5954.860630
5	-6570.087958
6	-2016.402125
7	-4271.004155
8	490.611791
9	-5149.346740
10	10611.576482
11	8661.886997
12	12446.641369
13	6796.378735
14	-16947.693104
15	-16297.587589
16	10055.036102
17	-4800.428034
18	-4748.168968
19	7163.632009
20	1811.887956
21	-5983.963770
22	-4354.693173
23	-1262.466061
24	-4788.999732
25	5144.849591
26	-4866.912270
27	-9377.248176
28	1623.265402
29	-767.388601
30	485.636602
31	-181.152734
32	-1595.336763
33	-1135.453688

	Profit
34	7652.782939
35	5991.106571
36	15424.078702
37	307.906968
38	11555.779367
39	-2744.396769
40	3403.599361
41	2997.938430
42	857.718955
43	9616.848808
44	569.213149
45	17300.941187
46	-6672.246236
47	-3949.833956
48	-13473.163247
49	-33552.873495

Homoscedaticity Check

In [57]:

```
X.columns
```

Out[57]:

```
Index(['R&D Spend', 'Administration', 'Marketing Spend', 'State'], dtype='ob
ject')
```

In [58]:

```
from sklearn.preprocessing import MinMaxScaler
min_max_scaler = MinMaxScaler()
scaled_X = min_max_scaler.fit_transform(X)
scaled_X = pd.DataFrame(data=scaled_X,columns = X.columns)
scaled_X
```

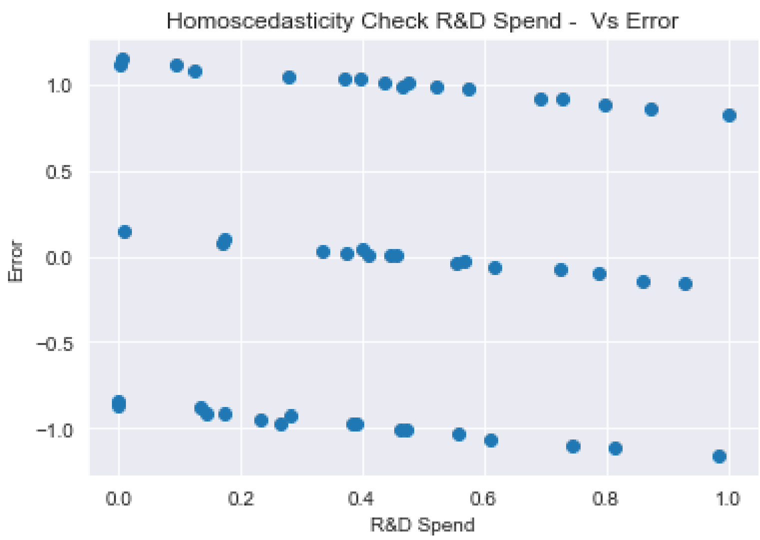
Out[58]:

	R&D Spend	Administration	Marketing Spend	State
0	1.000000	0.651744	1.000000	1.0
1	0.983359	0.761972	0.940893	0.0
2	0.927985	0.379579	0.864664	0.5
3	0.873136	0.512998	0.812235	1.0
4	0.859438	0.305328	0.776136	0.5
5	0.797566	0.369448	0.769126	1.0
6	0.814128	0.730161	0.270710	0.0
7	0.788018	0.717457	0.686493	0.5
8	0.729018	0.741733	0.660500	1.0
9	0.745906	0.436929	0.646443	0.0
10	0.616351	0.451506	0.485733	0.5
11	0.608845	0.308364	0.529362	0.0
12	0.567670	0.578836	0.529563	0.5
13	0.556352	0.641066	0.535552	0.0
14	0.725394	0.801327	0.543708	0.5
15	0.692617	0.543030	0.554864	1.0
16	0.471808	0.535270	0.560312	0.0
17	0.572468	0.714013	0.598948	1.0
18	0.554881	0.478772	0.625116	0.5
19	0.522650	0.778236	0.000000	1.0
20	0.461169	0.476424	0.633053	0.0
21	0.474084	0.780210	0.635327	1.0
22	0.447505	0.544293	0.642920	0.5
23	0.408424	0.414638	0.645992	0.5
24	0.465947	0.365388	0.297964	1.0
25	0.391080	0.671958	0.292427	0.0
26	0.455574	0.706845	0.284134	0.5
27	0.436093	0.582978	0.748613	1.0
28	0.399467	1.000000	0.250429	0.5
29	0.396769	0.774566	0.227092	1.0
30	0.374931	0.489928	0.193163	0.5
31	0.369741	0.772053	0.186989	1.0

	R&D Spend	Administration	Marketing Spend	State
32	0.383485	0.593294	0.097683	0.0
33	0.335617	0.394134	0.454943	0.5
34	0.280776	0.810055	0.446810	0.0
35	0.278284	0.257032	0.435618	1.0
36	0.173353	0.576825	0.426311	0.5
37	0.266527	0.000000	0.417626	0.0
38	0.122345	0.111636	0.392690	1.0
39	0.233194	0.241309	0.370931	0.0
40	0.173901	0.512041	0.366260	0.0
41	0.168691	0.254469	0.348614	0.5
42	0.142976	0.341852	0.313705	0.0
43	0.093776	0.579307	0.075319	1.0
44	0.134127	0.788072	0.060059	0.0
45	0.006049	0.554724	0.004036	1.0
46	0.007956	0.491260	0.629768	0.5
47	0.000000	0.640547	0.000000	0.0
48	0.003278	0.003502	0.000000	1.0
49	0.000000	0.500148	0.095749	0.0

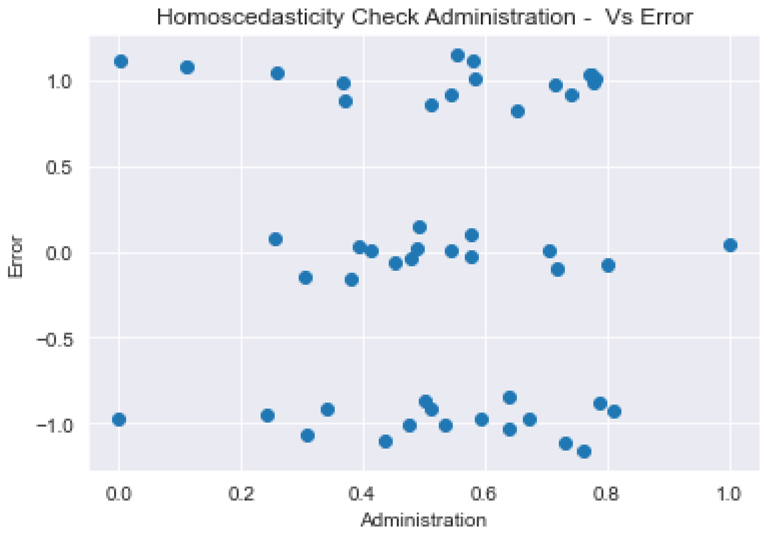
In [47]:

```
plt.scatter(x=scaled_X['R&D Spend'],y=error)
plt.title('Homoscedasticity Check R&D Spend -  Vs Error')
plt.xlabel('R&D Spend')
plt.ylabel('Error')
plt.show()
```



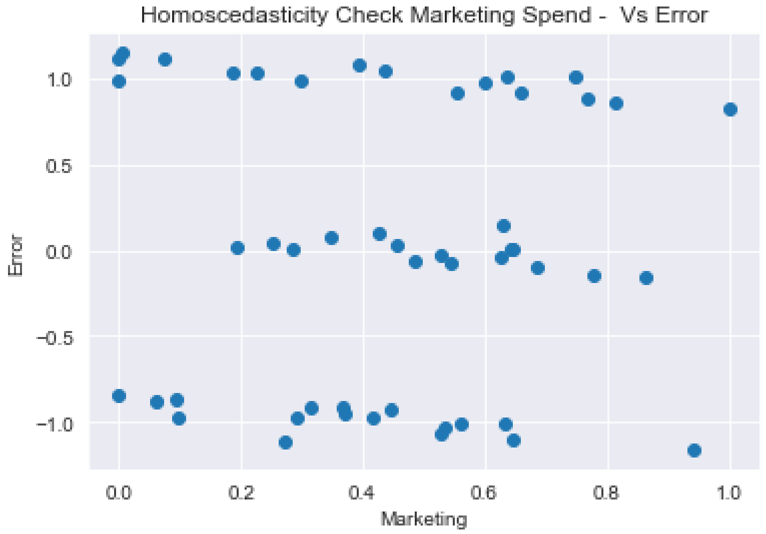
In [48]:

```
plt.scatter(x=scaled_X['Administration'],y=error)
plt.title('Homoscedasticity Check Administration -  Vs Error')
plt.xlabel('Administration')
plt.ylabel('Error')
plt.show()
```



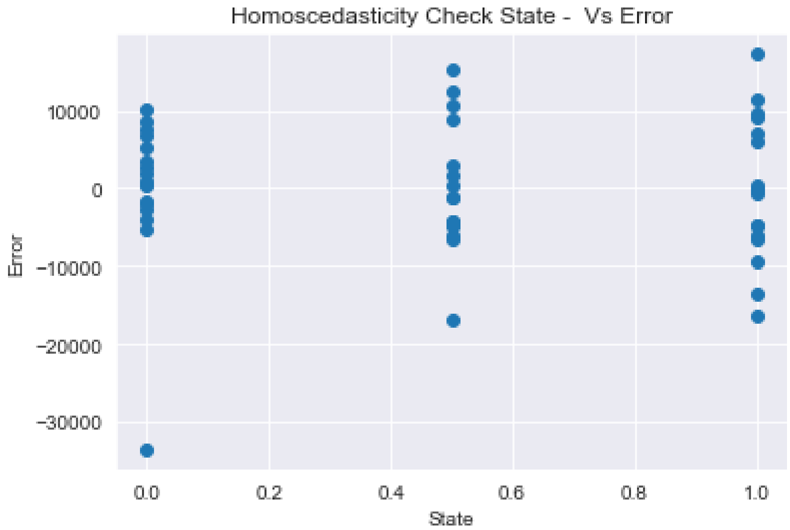
In [49]:

```
plt.scatter(x=scaled_X['Marketing Spend'],y=error)
plt.title('Homoscedasticity Check Marketing Spend -  Vs Error')
plt.xlabel('Marketing')
plt.ylabel('Error')
plt.show()
```



In [59]:

```
plt.scatter(x=scaled_X['State'],y=error)
plt.title('Homoscedasticity Check State -  Vs Error')
plt.xlabel('State')
plt.ylabel('Error')
plt.show()
```

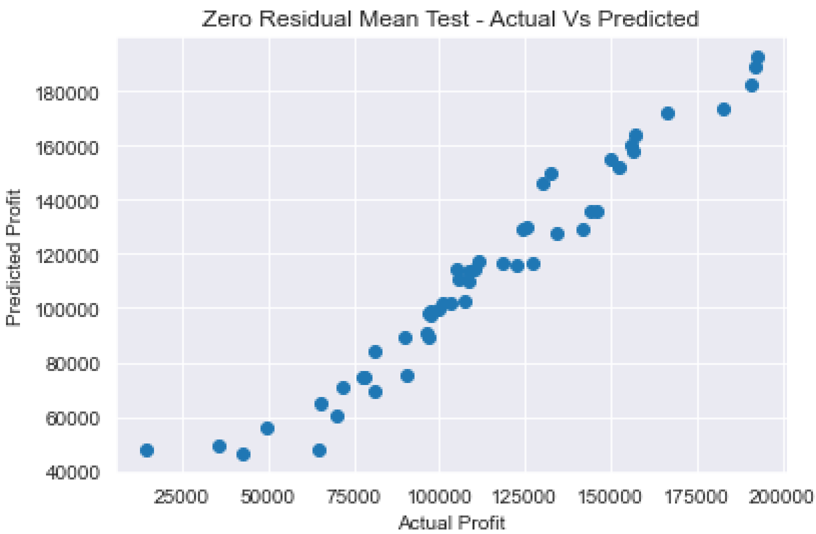


Homoscedasticity test is passed

Zero Residual Mean Test

In [69]:

```
plt.scatter(x=y,y=y_pred,)
plt.title('Zero Residual Mean Test - Actual Vs Predicted')
plt.xlabel('Actual Profit')
plt.ylabel('Predicted Profit')
plt.show()
```



Zero residual mean test is failed

Evaluation Metrics using Sk Learn

In [31]:

```
startup_data.head()
```

Out[31]:

	R&D Spend	Administration	Marketing Spend	State	Profit
0	165349.20	136897.80	471784.10	2	192261.83
1	162597.70	151377.59	443898.53	0	191792.06
2	153441.51	101145.55	407934.54	1	191050.39
3	144372.41	118671.85	383199.62	2	182901.99
4	142107.34	91391.77	366168.42	1	166187.94

In [9]:

```
X = startup_data.drop('Profit',axis = 1)
y = startup_data[['Profit']]
```

In [10]:

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.20,random_state=12, sh
```

In [11]:

```
X_train.shape,y_train.shape
```

Out[11]:

```
((40, 4), (40, 1))
```

In [12]:

```
X_test.shape,y_test.shape
```

Out[12]:

```
((10, 4), (10, 1))
```

In [13]:

```
from sklearn.linear_model import LinearRegression
lin_regres_model = LinearRegression()
lin_regres_model.fit(X_train,y_train)
```

Out[13]:

```
LinearRegression()
```

In [14]:

```
from sklearn.metrics import mean_squared_error,mean_absolute_error
```

In [15]:

```
y_pred_train = lin_regres_model.predict(X_train)
```

In [16]:

```
mean_squared_error(y_train,y_pred_train)
```

Out[16]:

```
82320640.28330517
```

In [17]:

```
mean_absolute_error(y_train,y_pred_train)
```

Out[17]:

```
6653.548562894053
```

In [18]:

```
y_pred_test = lin_regres_model.predict(X_test)
```

In [19]:

```
mean_squared_error(y_test,y_pred_test)
```

Out[19]:

```
69821055.63956103
```

In [20]:

```
mean_absolute_error(y_test,y_pred_test)
```

Out[20]:

```
6333.424549504579
```

Transformation using Log

In [24]:

```
import numpy as np
```

In [32]:

```
startup_data_2 = startup_data.copy()
startup_data_2
startup_data_2.head()
```

Out[32]:

	R&D Spend	Administration	Marketing Spend	State	Profit
0	165349.20	136897.80	471784.10	2	192261.83
1	162597.70	151377.59	443898.53	0	191792.06
2	153441.51	101145.55	407934.54	1	191050.39
3	144372.41	118671.85	383199.62	2	182901.99
4	142107.34	91391.77	366168.42	1	166187.94

In [28]:

```
import warnings
warnings.filterwarnings('ignore')
```

In [41]:

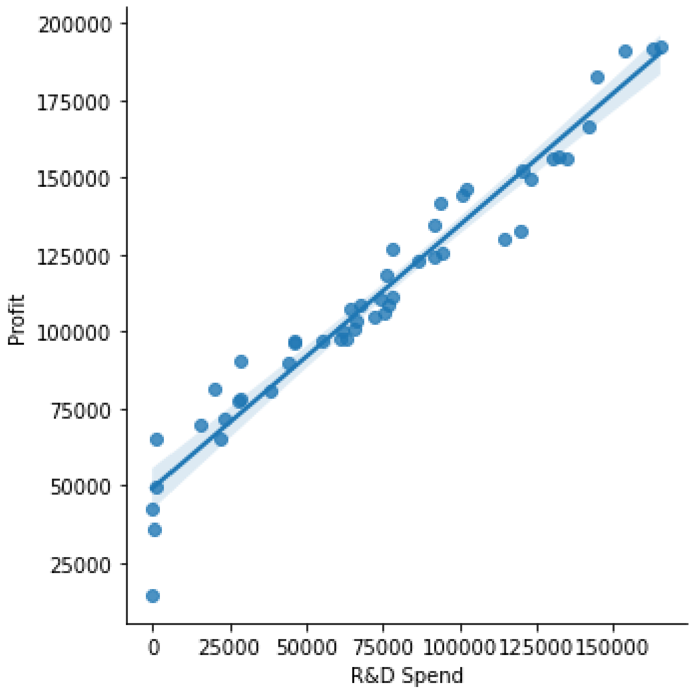
```
startup_data_2['log_R&D Spend']  = np.log(startup_data_2['R&D Spend'])
startup_data_2['log_Administration']  = np.log(startup_data_2['Administration'])
startup_data_2['log_Marketing Spend']  = np.log(startup_data_2['Marketing Spend'])
startup_data_2['log_State'] = np.log(startup_data_2['State'])
startup_data_2
startup_data_2.head()
```

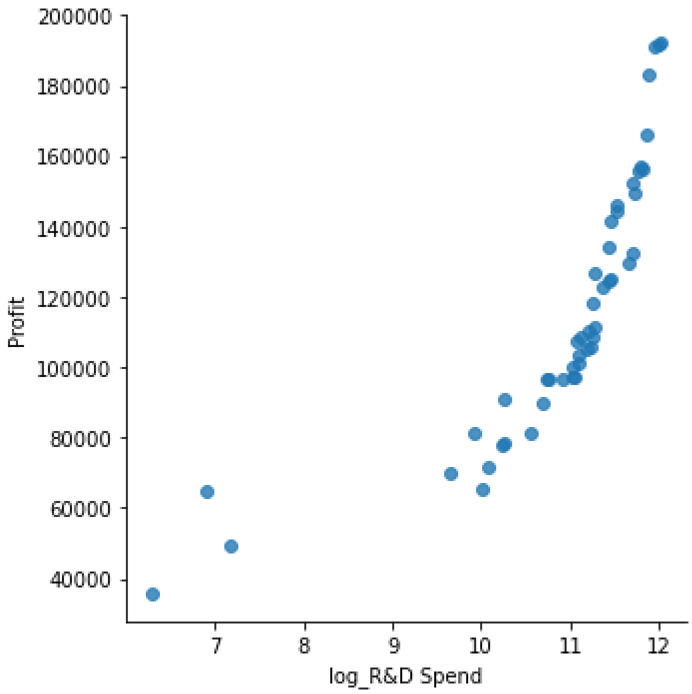
Out[41]:

	R&D Spend	Administration	Marketing Spend	State	Profit	log_R&D Spend	log_Administration	log_M
0	165349.20	136897.80	471784.10	2	192261.83	12.015815	11.826990	13
1	162597.70	151377.59	443898.53	0	191792.06	11.999034	11.927533	13
2	153441.51	101145.55	407934.54	1	191050.39	11.941075	11.524316	12
3	144372.41	118671.85	383199.62	2	182901.99	11.880151	11.684117	12
4	142107.34	91391.77	366168.42	1	166187.94	11.864338	11.422911	12

In [39]:

```
sns.lmplot(x='R&D Spend',y='Profit',data=startup_data_2)
sns.lmplot(x='log_R&D Spend',y='Profit',data=startup_data_2)
plt.show()
```





In [47]:

```
import statsmodels.formula.api as smf
```

In [48]:

```
model = smf.ols('Profit~log_R&D Spend', data = startup_data_2).fit()
print('AIC Value  : ',model.aic.round(2))
print('BIC Value  : ',model.bic.round(2))
print('R-square   : ',model.rsquared.round(4))
print('Adj.Rsquare: ',model.rsquared_adj.round(4))
```

```
Traceback (most recent call last):

  File "C:\ProgramData\Anaconda3\lib\site-packages\IPython\core\interactives
hell.py", line 3444, in run_code
    exec(code_obj, self.user_global_ns, self.user_ns)

  File "C:\Users\DELL\AppData\Local\Temp/ipykernel_2208/4027848182.py", line
1, in <module>
    model = smf.ols('Profit~log_R&D Spend', data = startup_data_2).fit()

  File "C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\base\model.p
y", line 169, in from_formula
    tmp = handle_formula_data(data, None, formula, depth=eval_env,

  File "C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\formula\formu
latools.py", line 63, in handle_formula_data
    result = dmatrices(formula, Y, depth, return_type='dataframe',

  File "C:\ProgramData\Anaconda3\lib\site-packages\patsy\highlevel.py", line
309, in dmatrices
    (lhs, rhs) = _do_highlevel_design(formula_like, data, eval_env,

  File "C:\ProgramData\Anaconda3\lib\site-packages\patsy\highlevel.py", line
164, in _do_highlevel_design
    design_infos = _try_incr_builders(formula_like, data_iter_maker, eval_en
v,

  File "C:\ProgramData\Anaconda3\lib\site-packages\patsy\highlevel.py", line
66, in _try_incr_builders
    return design_matrix_builders([formula_like.lhs_termlist,

  File "C:\ProgramData\Anaconda3\lib\site-packages\patsy\build.py", line 689
, in design_matrix_builders
    factor_states = _factors_memorize(all_factors, data_iter_maker, eval_en
v)

  File "C:\ProgramData\Anaconda3\lib\site-packages\patsy\build.py", line 354
, in _factors_memorize
    which_pass = factor.memorize_passes_needed(state, eval_env)

  File "C:\ProgramData\Anaconda3\lib\site-packages\patsy\eval.py", line 474,
in memorize_passes_needed
    subset_names = [name for name in ast_names(self.code)

  File "C:\ProgramData\Anaconda3\lib\site-packages\patsy\eval.py", line 474,
in <listcomp>
    subset_names = [name for name in ast_names(self.code)

  File "C:\ProgramData\Anaconda3\lib\site-packages\patsy\eval.py", line 105,
in ast_names
    for node in ast.walk(ast.parse(code)):

  File "C:\ProgramData\Anaconda3\lib\ast.py", line 50, in parse
```

```
    return compile(source, filename, mode, flags,

  File "<unknown>", line 1
    log_R & D Spend
              ^
SyntaxError: invalid syntax
```

In [ ]:

```
1
```