```
[1] # Import libraries and load the dataset
    import numpy as np
    import pandas as pd
    df=pd.read_csv("https://raw.githubusercontent.com/arib168/data/main/50_Startups.csv")
    df
```

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
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

[2]
```
#Print the first 5 rows of the dataset
df.head()
```

	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

[3]
```
#Print the last 5 rows of the dataset
df.tail()
```

	R&D Spend	Administration	Marketing Spend	State	Profit
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

[4]
```
# observing the shape of the dataset
df.shape
```

```
(50, 5)
```

[5]
```
# checking missing values
df.isna().sum()
```

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

[6]
```
# splitting input and output
x=df.iloc[:,:-1]
x
```

18	91749.16	114175.79	294919.57	Florida
19	86419.70	153514.11	0.00	New York
20	76253.86	113867.30	298664.47	California
21	78389.47	153773.43	299737.29	New York
22	73994.56	122782.75	303319.26	Florida
23	67532.53	105751.03	304768.73	Florida
24	77044.01	99281.34	140574.81	New York
25	64664.71	139553.16	137962.62	California
26	75328.87	144135.98	134050.07	Florida
27	72107.60	127864.55	353183.81	New York
28	66051.52	182645.56	118148.20	Florida
29	65605.48	153032.06	107138.38	New York
30	61994.48	115641.28	91131.24	Florida
31	61136.38	152701.92	88218.23	New York
32	63408.86	129219.61	46085.25	California
33	55493.95	103057.49	214634.81	Florida
34	46426.07	157693.92	210797.67	California
35	46014.02	85047.44	205517.64	New York
36	28663.76	127056.21	201126.82	Florida
37	44069.95	51283.14	197029.42	California
38	20229.59	65947.93	185265.10	New York
39	38558.51	82982.09	174999.30	California
40	28754.33	118546.05	172795.67	California
41	27892.92	84710.77	164470.71	Florida
42	23640.93	96189.63	148001.11	California

[6]

48	542.05	51743.15	0.00	New York
49	0.00	116983.80	45173.06	California

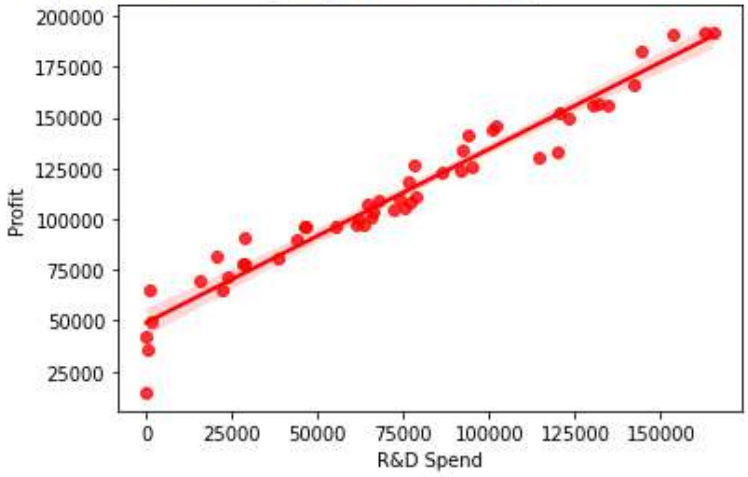
```
y=df.iloc[:,-1]
y
```

```
0     192261.83
1     191792.06
2     191050.39
3     182901.99
4     166187.94
5     156991.12
6     156122.51
7     155752.60
8     152211.77
9     149759.96
10    146121.95
11    144259.40
12    141585.52
13    134307.35
14    132602.65
15    129917.04
16    126992.93
17    125370.37
18    124266.90
19    122776.86
20    118474.03
21    111313.02
22    110352.25
23    108733.99
24    108552.04
25    107404.34
26    105733.54
27    105008.31
28    103282.38
29    101004.64
30     99937.59
31     97483.56
32     97427.84
33     96778.92
34     96712.80
35     96479.51
36     90708.19
37     89949.14
38     81229.06
39     81005.76
```

```
45     64926.08
46     49490.75
47     42559.73
48     35673.41
49     14681.40
Name: Profit, dtype: float64
```

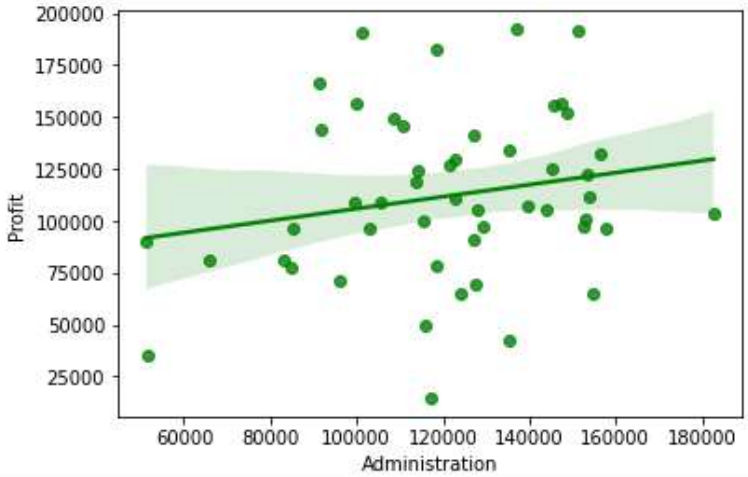
[8]
```
#Show graphically the influence of each factors using seaborn
import seaborn as sns
sns.regplot(x=df['R&D Spend'],y=y,color='red')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6fa8142e50>
```



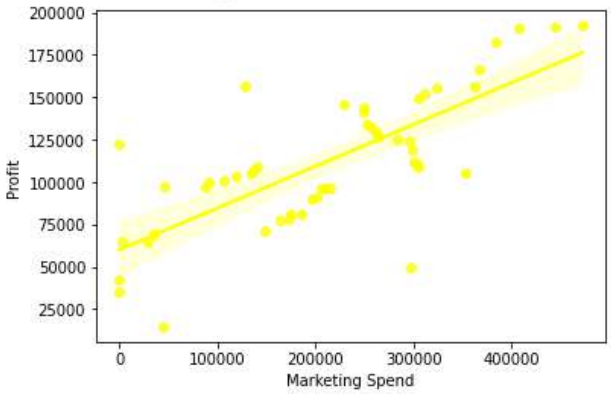
[9]
```
sns.regplot(x=df['Administration'],y=y,color='green')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6fa7bdf100>
```



```
[10] sns.regplot(x=df['Marketing Spend'],y=y,color='yellow')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6fa7bd6af0>
```




```
[11] df['State'].value_counts()
```

```
New York      17
California    17
Florida       16
Name: State, dtype: int64
```

```
[12] df.dtypes
```

```
R&D Spend          float64
Administration     float64
Marketing Spend    float64
State               object
Profit             float64
dtype: object
```

```
[13] # One hot encoding
     from sklearn.compose import make_column_transformer
     from sklearn.preprocessing import OneHotEncoder
     col_trans=make_column_transformer((OneHotEncoder(handle_unknown='ignore'),['State']),remainder='passthrough')
     x=col_trans.fit_transform(x)
     x
```

```
[1.0000000e+00, 0.0000000e+00, 0.0000000e+00, 6.3408860e+04,
 1.2921961e+05, 4.6085250e+04],
[0.0000000e+00, 1.0000000e+00, 0.0000000e+00, 5.5493950e+04,
 1.0305749e+05, 2.1463481e+05],
[1.0000000e+00, 0.0000000e+00, 0.0000000e+00, 4.6426070e+04,
 1.5769392e+05, 2.1079767e+05],
```

[13]

```
#split data into training data and testing data, testing data= 30%, training data=70%
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.30,random_state=42)
x_train
```

```
array([[1.0000000e+00, 0.0000000e+00, 0.0000000e+00, 1.3461546e+05,
        1.4719887e+05, 1.2771682e+05],
       [0.0000000e+00, 1.0000000e+00, 0.0000000e+00, 2.7892920e+04,
        8.4710770e+04, 1.6447071e+05],
       [0.0000000e+00, 1.0000000e+00, 0.0000000e+00, 1.3154600e+03,
        1.1581621e+05, 2.9711446e+05],
       [1.0000000e+00, 0.0000000e+00, 0.0000000e+00, 0.0000000e+00,
        1.3542692e+05, 0.0000000e+00],
       [0.0000000e+00, 0.0000000e+00, 1.0000000e+00, 1.1452361e+05,
        1.2261684e+05, 2.6177623e+05],
       [1.0000000e+00, 0.0000000e+00, 0.0000000e+00, 1.2333488e+05,
        1.0867917e+05, 3.0498162e+05],
       [1.0000000e+00, 0.0000000e+00, 0.0000000e+00, 7.8013110e+04,
        1.2159755e+05, 2.6434606e+05],
       [0.0000000e+00, 0.0000000e+00, 1.0000000e+00, 7.7044010e+04,
        9.9281340e+04, 1.4057481e+05],
       [1.0000000e+00, 0.0000000e+00, 0.0000000e+00, 4.6426070e+04,
        1.5769392e+05, 2.1079767e+05],
       [0.0000000e+00, 0.0000000e+00, 1.0000000e+00, 6.1136380e+04,
        1.5270192e+05, 8.8218230e+04],
       [0.0000000e+00, 0.0000000e+00, 1.0000000e+00, 1.6534920e+05,
        1.3689780e+05, 4.7178410e+05],
       [1.0000000e+00, 0.0000000e+00, 0.0000000e+00, 2.2177740e+04,
        1.5480614e+05, 2.8334720e+04],
       [0.0000000e+00, 0.0000000e+00, 1.0000000e+00, 7.2107600e+04,
        1.2786455e+05, 3.5318381e+05],
       [0.0000000e+00, 1.0000000e+00, 0.0000000e+00, 5.5493950e+04,
        1.0305749e+05, 2.1463481e+05],
       [0.0000000e+00, 0.0000000e+00, 1.0000000e+00, 1.3187690e+05,
        9.9814710e+04, 3.6286136e+05],
       [0.0000000e+00, 0.0000000e+00, 1.0000000e+00, 6.5605480e+04,
        1.5303206e+05, 1.0713838e+05],
       [1.0000000e+00, 0.0000000e+00, 0.0000000e+00, 1.0067196e+05,
        9.1790610e+04, 2.4974455e+05],
       [0.0000000e+00, 1.0000000e+00, 0.0000000e+00, 2.8663760e+04,
```

[15]
```
# Now we create a model using LinearRegression and predict the output
from sklearn.linear_model import LinearRegression
model=LinearRegression()
model.fit(x_train,y_train)
y_pred=model.predict(x_test)
y_pred
```

```
array([126187.39411505,  85788.82259512,  99777.02815177,  45706.12238326,
       127062.20722772,  51891.83884457, 109114.62977494, 100600.61123701,
        97953.99874714, 111730.57706807, 128818.49200668, 174195.35772633,
        93736.28538439, 148381.04097161, 172313.8713939 ])
```

[16]
```
# Comparing actual values with predicted values
df1=pd.DataFrame({'Actual_value':y_test,'Predicted_value':y_pred})
df1
```

	Actual_value	Predicted_value
13	134307.35	126187.394115
39	81005.76	85788.822595
30	99937.59	99777.028152
45	64926.08	45706.122383
17	125370.37	127062.207228
48	35673.41	51891.838845
26	105733.54	109114.629775
25	107404.34	100600.611237
32	97427.84	97953.998747
19	122776.86	111730.577068
12	141585.52	128818.492007
4	166187.94	174195.357726
37	89949.14	93736.285384
8	152211.77	148381.040972
3	182901.99	172313.871394

```
[17] #Slope and intercept
     print("intercept",model.intercept_)
     print("Slope is",model.coef_)
```

```
intercept 57153.61206241345
Slope is [ 2.59028652e+02  7.17099427e+02 -9.76128080e+02  8.04937292e-01
 -9.12577104e-02  2.80672826e-02]
```

```
[18] list(zip(x,model.coef_))
```

```
[(array([0.000000e+00, 0.000000e+00, 1.000000e+00, 1.653492e+05,
         1.368978e+05, 4.717841e+05]), 259.0286523053593),
 (array([1.0000000e+00, 0.0000000e+00, 0.0000000e+00, 1.6259770e+05,
         1.5137759e+05, 4.4389853e+05]), 717.0994272258821),
 (array([0.0000000e+00, 1.0000000e+00, 0.0000000e+00, 1.5344151e+05,
         1.0114555e+05, 4.0793454e+05]), -976.1280795289858),
 (array([0.0000000e+00, 0.0000000e+00, 1.0000000e+00, 1.4437241e+05,
         1.1867185e+05, 3.8319962e+05]), 0.8049372918011102),
 (array([0.0000000e+00, 1.0000000e+00, 0.0000000e+00, 1.4210734e+05,
         9.1391770e+04, 3.6616842e+05]), -0.09125771038947761),
 (array([0.0000000e+00, 0.0000000e+00, 1.0000000e+00, 1.3187690e+05,
         9.9814710e+04, 3.6286136e+05]), 0.028067282565416463)]
```

```
[19] # Performance measurements
     from sklearn.metrics import mean_absolute_error
     print("Error is",mean_absolute_error(y_test,y_pred))
```

```
Error is 7395.4335315232565
```

```
     from sklearn.metrics import mean_absolute_percentage_error
     print("Percentage error is",mean_absolute_percentage_error(y_test,y_pred))
```

```
Percentage error is 0.08929865344171896
```

```
[21] from sklearn.metrics import mean_squared_error
     print("Mean squared error is",mean_squared_error(y_test,y_pred))
```

```
Mean squared error is 84826955.03534976
```

```
Error is 7395.4335315232565
```

```
[20] from sklearn.metrics import mean_absolute_percentage_error
     print("Percentage error is",mean_absolute_percentage_error(y_test,y_pred))
```

```
Percentage error is 0.08929865344171896
```

```
[21] from sklearn.metrics import mean_squared_error
     print("Mean squared error is",mean_squared_error(y_test,y_pred))
```

```
Mean squared error is 84826955.03534976
```

```
[22] from sklearn.metrics import mean_squared_error
     root_mean=np.sqrt(mean_squared_error(y_test,y_pred))
     print("Root mean squared error",root_mean)
```

```
Root mean squared error 9210.154995186007
```

```
from sklearn.metrics import r2_score
print("R2 score is",r2_score(y_test,y_pred))
```

```
R2 score is 0.9397108063355675
```