Implementing Multiple Linear Regression

Objective

- To predict the profit made by a startup on the basis of expenses incurred and the state where they operate

```
# Importing the libraries
import numpy as np
import pandas as pd
from numpy import math

from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error

import matplotlib.pyplot as plt
```

```
<ipython-input-1-f27a1d36e8b5>:4: DeprecationWarning: `np.math` is a deprecated alias for the standard
  from numpy import math
```

```
# Importing the dataset
dataset = pd.read_csv('50_Startups.csv')
```

```
len(dataset)
```

```
50
```

```
dataset.head()
```

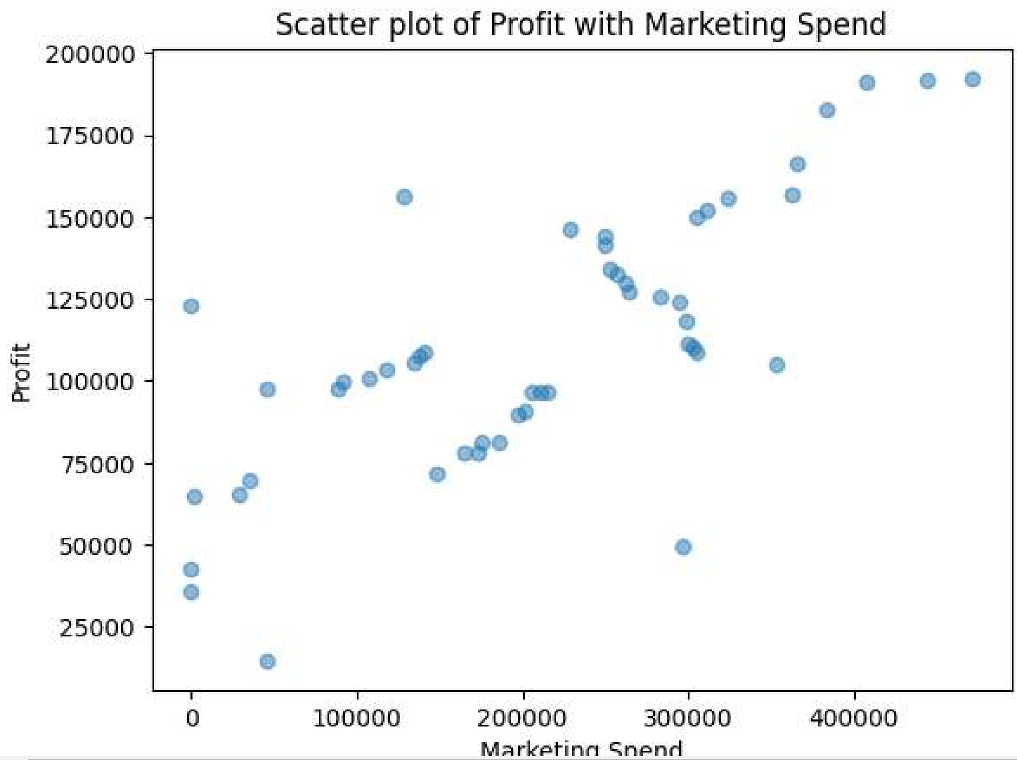
	R&D Spend	Administration	Marketing Spend	State	Profit
0	165349.20	136897.80	471784.10	New York	192261.83
1	162597.70	151377.59	443898.53	California	191792.06
2	153441.51	101145.55	407934.54	Florida	191050.39
3	144372.41	118671.85	383199.62	New York	182901.99
4	142107.34	91391.77	366168.42	Florida	166187.94

Next steps: Generate code with dataset View recommended plots New interactive sheet

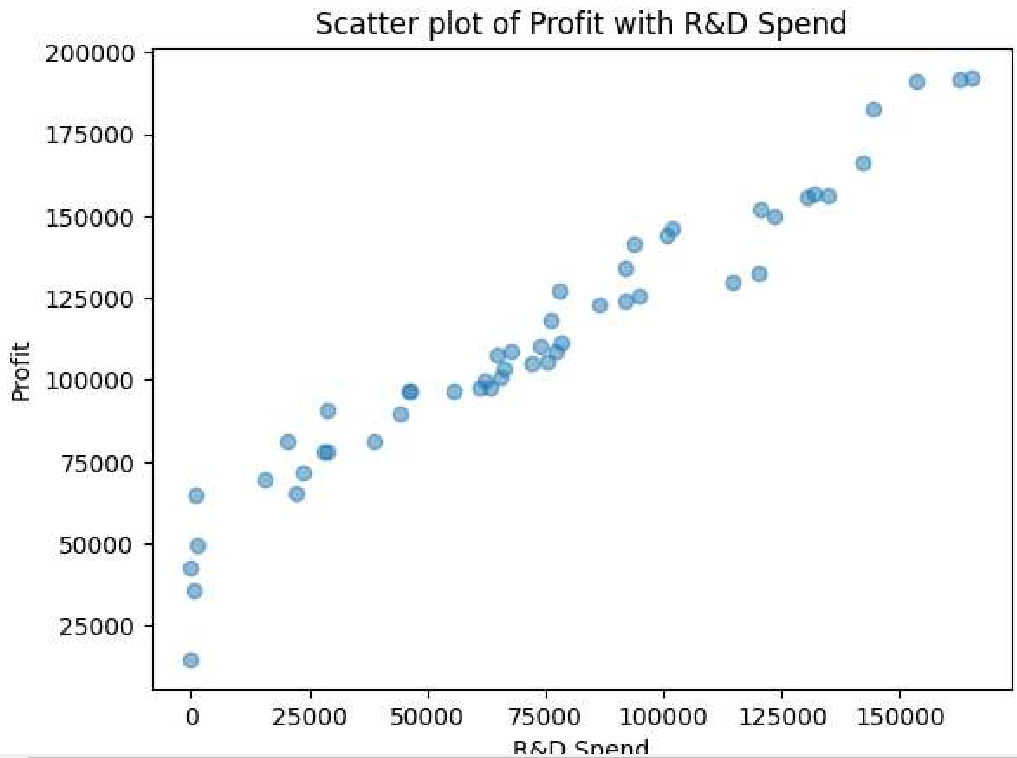
```
dataset.shape
```

```
(50, 5)
```

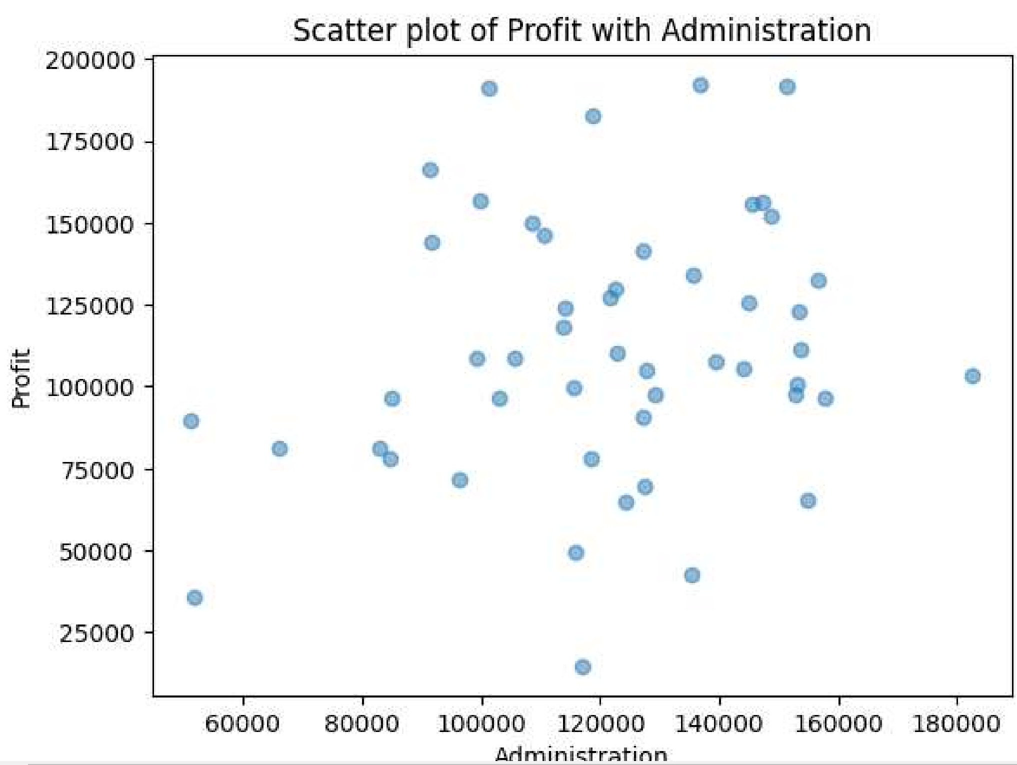
```
plt.scatter(dataset['Marketing Spend'], dataset['Profit'], alpha=0.5)
plt.title('Scatter plot of Profit with Marketing Spend')
plt.xlabel('Marketing Spend')
plt.ylabel('Profit')
plt.show()
```




```
plt.scatter(dataset['R&D Spend'], dataset['Profit'], alpha=0.5)
plt.title('Scatter plot of Profit with R&D Spend')
plt.xlabel('R&D Spend')
plt.ylabel('Profit')
plt.show()
```




```
plt.scatter(dataset['Administration'], dataset['Profit'], alpha=0.5)
plt.title('Scatter plot of Profit with Administration')
plt.xlabel('Administration')
plt.ylabel('Profit')
plt.show()
```



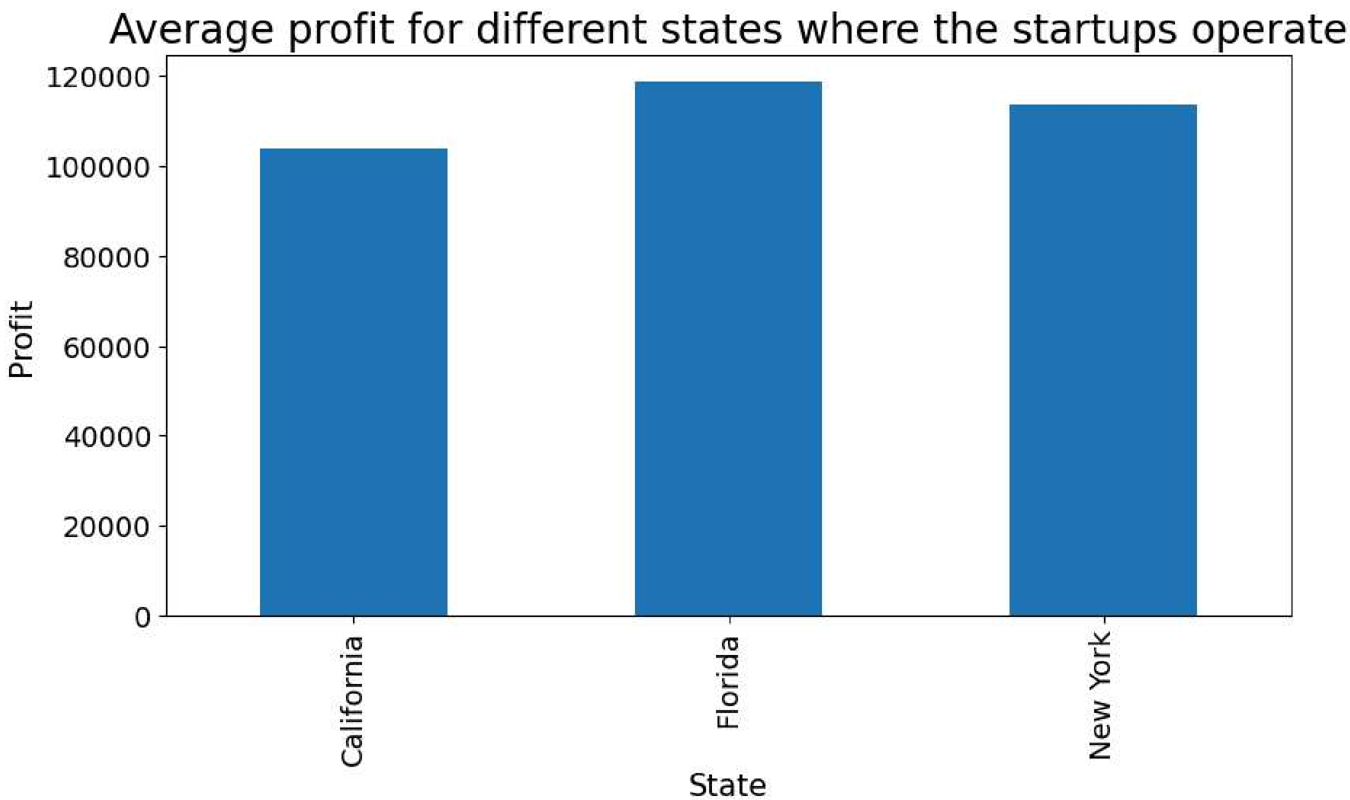

```
# Create the figure object
ax = dataset.groupby(['State'])['Profit'].mean().plot.bar(
```

```
    figsize = (10,5),
    fontsize = 14
)

# Set the title
ax.set_title("Average profit for different states where the startups operate", fontsize = 20)

# Set x and y-labels
ax.set_xlabel("State", fontsize = 15)
ax.set_ylabel("Profit", fontsize = 15)
```

```
Text(0, 0.5, 'Profit')
```




```
dataset.State.value_counts()
```

	count
State	
New York	17
California	17
Florida	16

```
# Create dummy variables for the catgeorical variable State
dataset['NewYork_State'] = np.where(dataset['State']=='New York', 1, 0)
dataset['California_State'] = np.where(dataset['State']=='California', 1, 0)
dataset['Florida_State'] = np.where(dataset['State']=='Florida', 1, 0)
```

```
# Drop the original column State from the dataframe
dataset.drop(columns=['State'],axis=1,inplace=True)
```

```
dataset.head()
```

	R&D Spend	Administration	Marketing Spend	Profit	NewYork_State	California_State	Florida_State
0	165349.20	136897.80	471784.10	192261.83	1	0	0
1	162597.70	151377.59	443898.53	191792.06	0	1	0
2	153441.51	101145.55	407934.54	191050.39	0	0	1
3	144372.41	118671.85	383199.62	182901.99	1	0	0

Next steps: Generate code with dataset | View recommended plots | New interactive sheet

```
dependent_variable = 'Profit'
```

```
# Create a list of independent variables
independent_variables = list(set(dataset.columns.tolist()) - {dependent_variable})
```

```
independent_variables
```

```
['Florida_State',
 'Marketing Spend',
 'Administration',
 'R&D Spend',
 'California_State',
 'NewYork_State']
```

```
# Create the data of independent variables
X = dataset[independent_variables].values

# Create the dependent variable data
y = dataset[dependent_variable].values
```

```
dataset[independent_variables]
```

0	0	471784.10	136897.80	165349.20	0	1
1	0	443898.53	151377.59	162597.70	1	0
2	1	407934.54	101145.55	153441.51	0	0
3	0	383199.62	118671.85	144372.41	0	1
4	1	366168.42	91391.77	142107.34	0	0
5	0	362861.36	99814.71	131876.90	0	1
6	0	127716.82	147198.87	134615.46	1	0
7	1	323876.68	145530.06	130298.13	0	0
8	0	311613.29	148718.95	120542.52	0	1
9	0	304981.62	108679.17	123334.88	1	0
10	1	229160.95	110594.11	101913.08	0	0
11	0	249744.55	91790.61	100671.96	1	0
12	1	249839.44	127320.38	93863.75	0	0
13	0	252664.93	135495.07	91992.39	1	0
14	1	256512.92	156547.42	119943.24	0	0
15	0	261776.23	122616.84	114523.61	0	1
16	0	264346.06	121597.55	78013.11	1	0
17	0	282574.31	145077.58	94657.16	0	1
18	1	294919.57	114175.79	91749.16	0	0
19	0	0.00	153514.11	86419.70	0	1
20	0	298664.47	113867.30	76253.86	1	0
21	0	299737.29	153773.43	78389.47	0	1
22	1	303319.26	122782.75	73994.56	0	0
23	1	304768.73	105751.03	67532.53	0	0
24	0	140574.81	99281.34	77044.01	0	1
25	0	137962.62	139553.16	64664.71	1	0
26	1	134050.07	144135.98	75328.87	0	0
27	0	353183.81	127864.55	72107.60	0	1
28	1	118148.20	182645.56	66051.52	0	0
29	0	107138.38	153032.06	65605.48	0	1
30	1	91131.24	115641.28	61994.48	0	0
31	0	88218.23	152701.92	61136.38	0	1
32	0	46085.25	129219.61	63408.86	1	0
33	1	214634.81	103057.49	55493.95	0	0

34	0	210797.67	157693.92	46426.07	1	0
35	0	205517.64	85047.44	46014.02	0	1
36	1	201126.82	127056.21	28663.76	0	0
37	0	197029.42	51283.14	44069.95	1	0
38	0	185265.10	65947.93	20229.59	0	1
39	0	174999.30	82982.09	38558.51	1	0
40	0	172795.67	118546.05	28754.33	1	0
41	1	164470.71	84710.77	27892.92	0	0
42	0	148001.11	96189.63	23640.93	1	0
43	0	35534.17	127382.30	15505.73	0	1
44	0	28334.72	154806.14	22177.74	1	0
45	0	1903.93	124153.04	1000.23	0	1
46	1	297114.46	115816.21	1315.46	0	0
47	0	0.00	135426.92	0.00	1	0
48	0	0.00	51743.15	542.05	0	1
49	0	45173.06	116983.80	0.00	1	0

```
# Splitting the dataset into the Training set and Test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 0)
```

```
X_train[0:10]
```

```
array([[1.0000000e+00, 2.1463481e+05, 1.0305749e+05, 5.5493950e+04,
        0.0000000e+00, 0.0000000e+00],
       [0.0000000e+00, 2.0551764e+05, 8.5047440e+04, 4.6014020e+04,
        0.0000000e+00, 1.0000000e+00],
       [1.0000000e+00, 1.3405007e+05, 1.4413598e+05, 7.5328870e+04,
        0.0000000e+00, 0.0000000e+00],
       [0.0000000e+00, 2.1079767e+05, 1.5769392e+05, 4.6426070e+04,
        1.0000000e+00, 0.0000000e+00],
       [1.0000000e+00, 2.9491957e+05, 1.1417579e+05, 9.1749160e+04,
        0.0000000e+00, 0.0000000e+00],
       [1.0000000e+00, 3.2387668e+05, 1.4553006e+05, 1.3029813e+05,
        0.0000000e+00, 0.0000000e+00],
       [1.0000000e+00, 2.5651292e+05, 1.5654742e+05, 1.1994324e+05,
        0.0000000e+00, 0.0000000e+00],
       [0.0000000e+00, 1.9039300e+03, 1.2415304e+05, 1.0002300e+03,
        0.0000000e+00, 1.0000000e+00],
       [0.0000000e+00, 0.0000000e+00, 5.1743150e+04, 5.4205000e+02,
        0.0000000e+00, 1.0000000e+00],
       [0.0000000e+00, 1.0713838e+05, 1.5303206e+05, 6.5605480e+04,
        0.0000000e+00, 1.0000000e+00]])
```

```
# Transforming data
scaler = MinMaxScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

```
X_train[0:10]
```

```
array([[1.        , 0.45494286, 0.48655174, 0.33561668, 0.        ,
         0.        ],
       [0.        , 0.43561799, 0.3173015 , 0.2782839 , 0.        ,
         1.        ],
       [1.        , 0.28413435, 0.87258866, 0.45557444, 0.        ,
         0.        ],
       [0.        , 0.44680961, 1.        , 0.2807759 , 1.        ,
         0.        ],
       [1.        , 0.62511553, 0.59103645, 0.55488118, 0.        ,
         0.        ],
       [1.        , 0.68649342, 0.88568959, 0.7880179 , 0.        ,
         0.        ],
       [1.        , 0.54370828, 0.98922572, 0.72539353, 0.        ,
         0.        ],
       [0.        , 0.0040356 , 0.6847981 , 0.0060492 , 0.        ,
         1.        ],
       [0.        , 0.        , 0.00432296, 0.00327821, 0.        ,
         1.        ],
       [0.        , 0.22709197, 0.95618996, 0.39676926, 0.        ,
         1.        ]])
```

```
# Fitting Multiple Linear Regression to the Training set
regressor = LinearRegression()
regressor.fit(X_train, y_train)
```

```
LinearRegression
LinearRegression()
```

```
regressor.intercept_
```

```
44153.95466784856
```

```
regressor.coef_
```

```
array([-8.72645791e+02,  1.72720281e+04,  3.49927567e+03,  1.27892182e+05,
        8.66383692e+01,  7.86007422e+02])
```

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

```
y_train
```

```
array([ 96778.92,  96479.51, 105733.54,  96712.8 , 124266.9 , 155752.6 ,
       132602.65,  64926.08,  35673.41, 101004.64, 129917.04,  99937.59,
        97427.84, 126992.93,  71498.49, 118474.03,  69758.98, 152211.77,
       134307.35, 107404.34, 156991.12, 125370.37,  78239.91,  14681.4 ,
       191792.06, 141585.52,  89949.14, 108552.04, 156122.51, 108733.99,
        90708.19, 111313.02, 122776.86, 149759.96,  81005.76,  49490.75,
       182901.99, 192261.83,  42559.73,  65200.33])
```

```
# Predicting the Test set results
y_pred = regressor.predict(X_test)
```

```
#Predicted profit on the test data
y_pred
```