Machine Learning - 2301CS621

635 | Vishal Baraiya | 23010101014

Lab - 4

Simple Linear Regression

Step 1. Import the necessary libraries

In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Step 2. Import the dataset

In [2]:

```
df = pd.read_csv("50_Startups.csv")
df.head(5)
```

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

Step 3 . Check the State Column

In [3]:
```
df["State"].value_counts()
```

Out[3]:
```
State
New York      17
California    17
Florida       16
Name: count, dtype: int64
```

Step 4 . Splitting dataset in to input and output

In [4]:
```
x = df.iloc[::,:4:]
y = df.iloc[::,4::]
x.head(5)
```

Out[4]:

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

In [5]:
```
y.head(5)
```

Out[5]:

	Profit
0	192261.83
1	191792.06
2	191050.39
3	182901.99
4	166187.94

Step 5 . Convert state Column into Numeric Column

Step 5.1 . Perform Transformation

In [6]:
```
x1 = pd.get_dummies(x,columns=["State"],drop_first=True)
x1.head(5)
```

Out[6]:

	R&D Spend	Administration	Marketing Spend	State_Florida	State_New York
0	165349.20	136897.80	471784.10	False	True
1	162597.70	151377.59	443898.53	False	False
2	153441.51	101145.55	407934.54	True	False
3	144372.41	118671.85	383199.62	False	True
4	142107.34	91391.77	366168.42	True	False

Step 6 . Dummy variable trap

In [7]:

```
# Already Performed using | drop_first =True
```

Step 7 Splitting dataset in to Train and Test

In [8]:

```
from sklearn.model_selection import train_test_split

x_train,x_test,y_train,y_test = train_test_split(x1,y,test_size=0.2,random_state=42
```

In [9]:

```
x_train
```

Out[9]:

	R&D Spend	Administration	Marketing Spend	State_Florida	State_New York
12	93863.75	127320.38	249839.44	True	False
4	142107.34	91391.77	366168.42	True	False
37	44069.95	51283.14	197029.42	False	False
8	120542.52	148718.95	311613.29	False	True
3	144372.41	118671.85	383199.62	False	True
6	134615.46	147198.87	127716.82	False	False
41	27892.92	84710.77	164470.71	True	False
46	1315.46	115816.21	297114.46	True	False
47	0.00	135426.92	0.00	False	False
15	114523.61	122616.84	261776.23	False	True
9	123334.88	108679.17	304981.62	False	False
16	78013.11	121597.55	264346.06	False	False
24	77044.01	99281.34	140574.81	False	True
34	46426.07	157693.92	210797.67	False	False
31	61136.38	152701.92	88218.23	False	True
0	165349.20	136897.80	471784.10	False	True
44	22177.74	154806.14	28334.72	False	False
27	72107.60	127864.55	353183.81	False	True
33	55493.95	103057.49	214634.81	True	False
5	131876.90	99814.71	362861.36	False	True
29	65605.48	153032.06	107138.38	False	True
11	100671.96	91790.61	249744.55	False	False
36	28663.76	127056.21	201126.82	True	False
1	162597.70	151377.59	443898.53	False	False
21	78389.47	153773.43	299737.29	False	True
2	153441.51	101145.55	407934.54	True	False
43	15505.73	127382.30	35534.17	False	True
35	46014.02	85047.44	205517.64	False	True
23	67532.53	105751.03	304768.73	True	False
40	28754.33	118546.05	172795.67	False	False

	R&D Spend	Administration	Marketing Spend	State_Florida	State_New York
10	101913.08	110594.11	229160.95	True	False
22	73994.56	122782.75	303319.26	True	False
18	91749.16	114175.79	294919.57	True	False
49	0.00	116983.80	45173.06	False	False
20	76253.86	113867.30	298664.47	False	False
7	130298.13	145530.06	323876.68	True	False
42	23640.93	96189.63	148001.11	False	False
14	119943.24	156547.42	256512.92	True	False
28	66051.52	182645.56	118148.20	True	False
38	20229.59	65947.93	185265.10	False	True

In [10]:

```
x_test
```

Out[10]:

	R&D Spend	Administration	Marketing Spend	State_Florida	State_New York
13	91992.39	135495.07	252664.93	False	False
39	38558.51	82982.09	174999.30	False	False
30	61994.48	115641.28	91131.24	True	False
45	1000.23	124153.04	1903.93	False	True
17	94657.16	145077.58	282574.31	False	True
48	542.05	51743.15	0.00	False	True
26	75328.87	144135.98	134050.07	True	False
25	64664.71	139553.16	137962.62	False	False
32	63408.86	129219.61	46085.25	False	False
19	86419.70	153514.11	0.00	False	True

In [11]:

```
y_train
```

Out[11]:

	Profit
12	141585.52
4	166187.94
37	89949.14
8	152211.77
3	182901.99
6	156122.51
41	77798.83
46	49490.75
47	42559.73
15	129917.04
9	149759.96
16	126992.93
24	108552.04
34	96712.80
31	97483.56
0	192261.83
44	65200.33
27	105008.31
33	96778.92
5	156991.12
29	101004.64
11	144259.40
36	90708.19
1	191792.06
21	111313.02
2	191050.39
43	69758.98
35	96479.51
23	108733.99
40	78239.91

	Profit
10	146121.95
22	110352.25
18	124266.90
49	14681.40
20	118474.03
7	155752.60
42	71498.49
14	132602.65
28	103282.38
38	81229.06

In [12]:

```
y_test
```

Out[12]:

	Profit
13	134307.35
39	81005.76
30	99937.59
45	64926.08
17	125370.37
48	35673.41
26	105733.54
25	107404.34
32	97427.84
19	122776.86

Step 8 Import LinearRegression model from linear_model family

In [13]:

```
from sklearn.linear_model import LinearRegression
model = LinearRegression()
```

Step 9 Fit the data

In [14]:
```
model.fit(x_train,y_train)
```

Out[14]:
```
▾ LinearRegression
▶ Parameters
```

Step 10 Predict the data

In [15]:
```
y_predict = model.predict(x_test)
y_predict
```

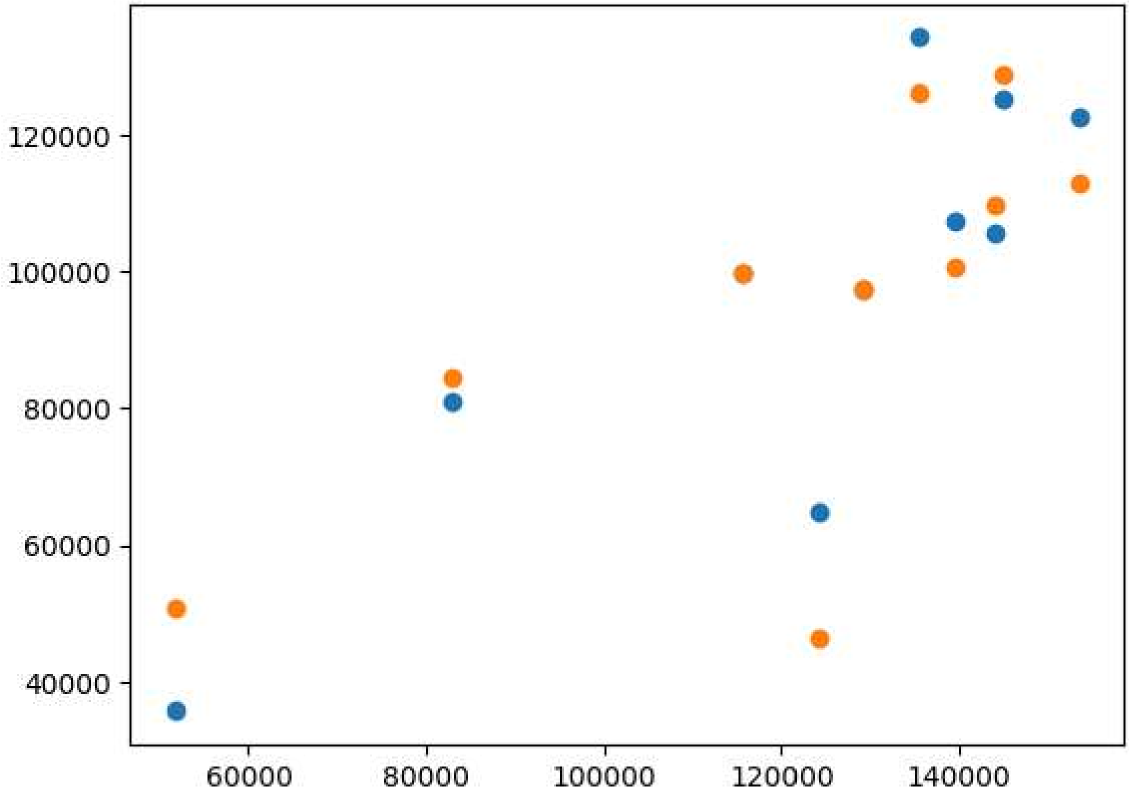
Out[15]:
```
array([[126362.87908255],
       [ 84608.45383634],
       [ 99677.49425147],
       [ 46357.46068582],
       [128750.48288504],
       [ 50912.4174188 ],
       [109741.35032702],
       [100643.24281647],
       [ 97599.27574594],
       [113097.42524432]])
```

Step 11 Display Result

In [16]:
```
# y_test and y_predict
```

In [17]:
```
plt.scatter(x_test['Administration'],y_test)
plt.scatter(x_test['Administration'],y_predict)
```

Out[17]:
```
<matplotlib.collections.PathCollection at 0x258fff6bb10>
```



```
In [ ]:
```

RSS

```
In [18]: import numpy as np
```

```
In [19]: print(np.sum( (y_test.values - y_predict) ** 2))

820103630.443011
```

```
In [20]: len(y_test)

Out[20]: 10
```

```
In [21]: from sklearn.metrics import mean_squared_error
```

```
In [22]: mean_squared_error(y_test.values,y_predict)*len(y_predict)

Out[22]: 820103630.443011
```

```
In [23]: len(y_predict)

Out[23]: 10
```

R Square

In [24]:
```
from sklearn.metrics import r2_score
```

In [25]:
```
r2_score(y_test,y_predict)
```

Out[25]:
```
0.8987266414328636
```

In [ ]:
```
```

Now use Polynomial Regression on Position_Salaries dataset

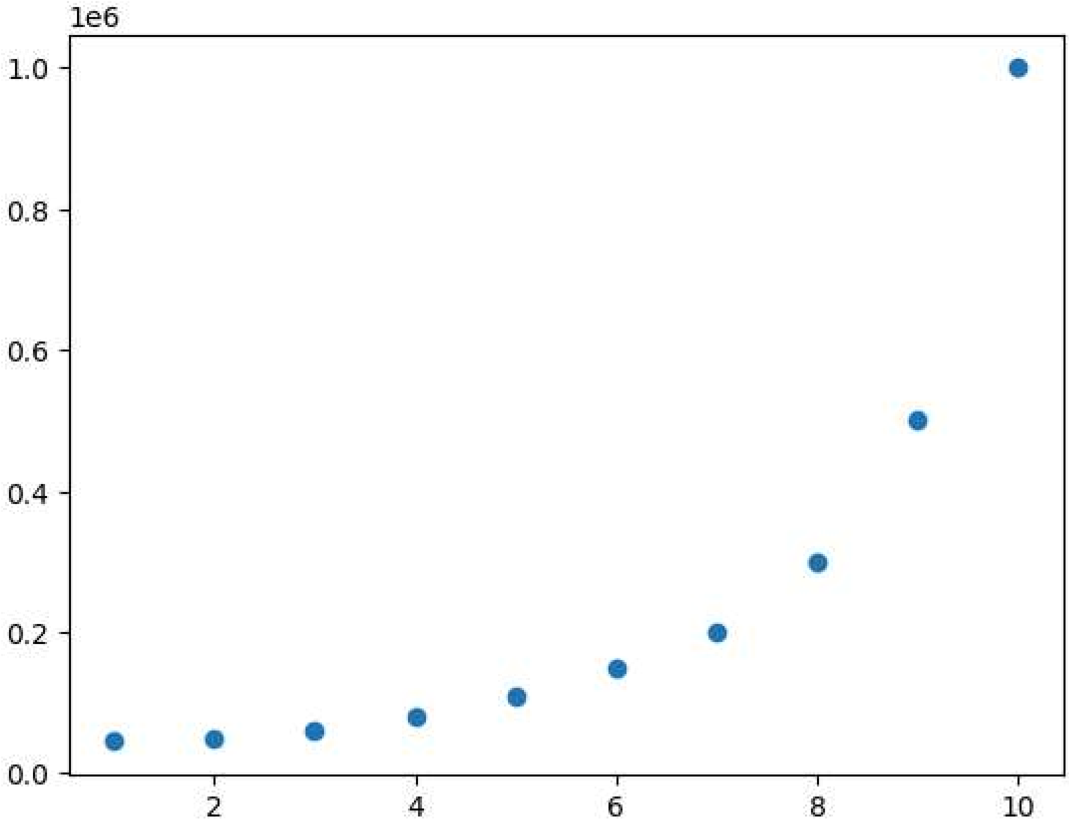
In [26]:
```
ps_df = pd.read_csv("Position_Salaries.csv")
ps_df
```

Out[26]:

	Position	Level	Salary
0	Business Analyst	1	45000
1	Junior Consultant	2	50000
2	Senior Consultant	3	60000
3	Manager	4	80000
4	Country Manager	5	110000
5	Region Manager	6	150000
6	Partner	7	200000
7	Senior Partner	8	300000
8	C-level	9	500000
9	CEO	10	1000000

In [27]:
```
plt.scatter(ps_df['Level'],ps_df['Salary'])
```

Out[27]:
```
<matplotlib.collections.PathCollection at 0x2588d15ad50>
```



In [28]:

```
x = ps_df.iloc[::,1:2:]
y = ps_df.iloc[::,2::]
x
```

Out[28]:

	Level
0	1
1	2
2	3
3	4
4	5
5	6
6	7
7	8
8	9
9	10

In [29]:

```
y
```

Out[29]:

	Salary
0	45000
1	50000
2	60000
3	80000
4	110000
5	150000
6	200000
7	300000
8	500000
9	1000000

In [30]:

```
from sklearn.model_selection import train_test_split

x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3,random_state=42)
```

In [31]:

```
from sklearn.preprocessing import PolynomialFeatures
poly = PolynomialFeatures(degree=2)
```

In [32]:

```
x1 = poly.fit_transform(x_train)
x1
```

Out[32]:

```
array([[  1.,   1.,   1.],
       [  1.,   8.,  64.],
       [  1.,   3.,   9.],
       [  1.,  10., 100.],
       [  1.,   5.,  25.],
       [  1.,   4.,  16.],
       [  1.,   7.,  49.]])
```

In [33]:

```
# poly.fit(x1,y_train)
```

In [34]:

```
model1 = LinearRegression()
model1.fit(x1,y_train)
```

Out[34]:

```
▾ LinearRegression  i ?

▶ Parameters
```

In [36]:

```
y_poly_predict = model1.predict(poly.fit_transform(x_test))
```

In [37]:

```
y_poly_predict
```

Out[37]:
```
array([[652544.72066783],
       [ 37834.14365654],
       [141632.41904413]])
```

In [38]:
```
y_test
```

Out[38]:

	Salary
8	500000
1	50000
5	150000

In [ ]:
```

```