In [1]:

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

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

In [2]:

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

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

In [3]:

```
df.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 [4]:

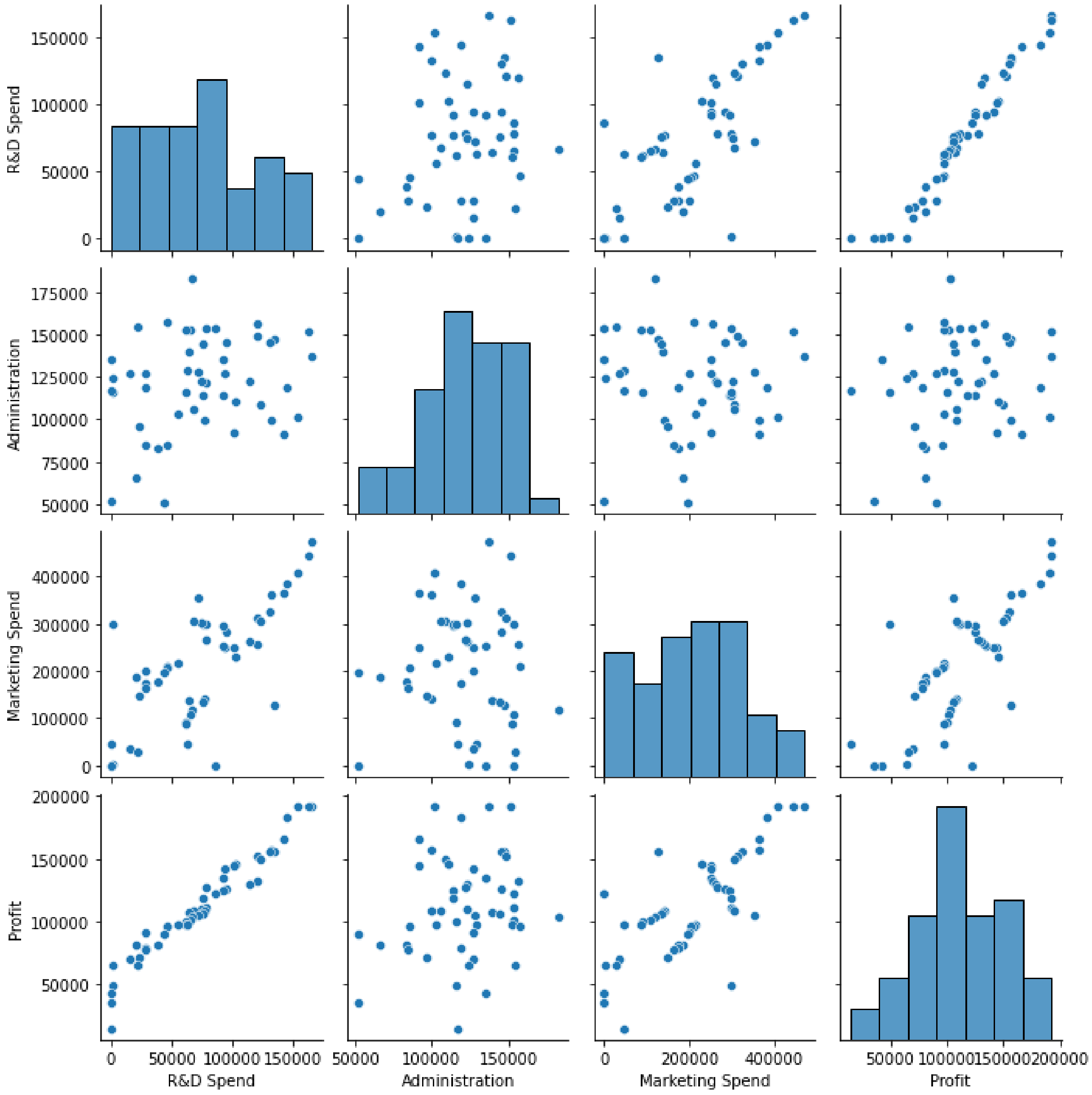
```
df.isna().sum()
```

Out[4]:

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

In [5]:
```
sns.pairplot(df)
```

Out[5]:
```
<seaborn.axisgrid.PairGrid at 0x6ab0e37f0>
```



In [6]:
```
x= df.iloc[:, :-1].values # [row:row, col:col]
y =df.iloc[:, -1].values
```

In [7]:
```
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
```

In [8]:
```
ct = ColumnTransformer(transformers=[("encoder", OneHotEncoder(), [3])], remainde

x = np.array(ct.fit_transform(x))
```

In [9]:
```
x
```

Out[9]:
```
array([[0.0, 0.0, 1.0, 165349.2, 136897.8, 471784.1],
       [1.0, 0.0, 0.0, 162597.7, 151377.59, 443898.53],
       [0.0, 1.0, 0.0, 153441.51, 101145.55, 407934.54],
       [0.0, 0.0, 1.0, 144372.41, 118671.85, 383199.62],
       [0.0, 1.0, 0.0, 142107.34, 91391.77, 366168.42],
       [0.0, 0.0, 1.0, 131876.9, 99814.71, 362861.36],
       [1.0, 0.0, 0.0, 134615.46, 147198.87, 127716.82],
       [0.0, 1.0, 0.0, 130298.13, 145530.06, 323876.68],
       [0.0, 0.0, 1.0, 120542.52, 148718.95, 311613.29],
       [1.0, 0.0, 0.0, 123334.88, 108679.17, 304981.62],
       [0.0, 1.0, 0.0, 101913.08, 110594.11, 229160.95],
       [1.0, 0.0, 0.0, 100671.96, 91790.61, 249744.55],
       [0.0, 1.0, 0.0, 93863.75, 127320.38, 249839.44],
       [1.0, 0.0, 0.0, 91992.39, 135495.07, 252664.93],
       [0.0, 1.0, 0.0, 119943.24, 156547.42, 256512.92],
       [0.0, 0.0, 1.0, 114523.61, 122616.84, 261776.23],
       [1.0, 0.0, 0.0, 78013.11, 121597.55, 264346.06],
       [0.0, 0.0, 1.0, 94657.16, 145077.58, 282574.31],
       [0.0, 1.0, 0.0, 91749.16, 114175.79, 294919.57],
       [0.0, 0.0, 1.0, 86419.7, 153514.11, 0.0],
       [1.0, 0.0, 0.0, 76253.86, 113867.3, 298664.47],
       [0.0, 0.0, 1.0, 78389.47, 153773.43, 299737.29],
       [0.0, 1.0, 0.0, 73994.56, 122782.75, 303319.26],
       [0.0, 1.0, 0.0, 67532.53, 105751.03, 304768.73],
       [0.0, 0.0, 1.0, 77044.01, 99281.34, 140574.81],
       [1.0, 0.0, 0.0, 64664.71, 139553.16, 137962.62],
       [0.0, 1.0, 0.0, 75328.87, 144135.98, 134050.07],
       [0.0, 0.0, 1.0, 72107.6, 127864.55, 353183.81],
       [0.0, 1.0, 0.0, 66051.52, 182645.56, 118148.2],
       [0.0, 0.0, 1.0, 65605.48, 153032.06, 107138.38],
       [0.0, 1.0, 0.0, 61994.48, 115641.28, 91131.24],
       [0.0, 0.0, 1.0, 61136.38, 152701.92, 88218.23],
       [1.0, 0.0, 0.0, 63408.86, 129219.61, 46085.25],
       [0.0, 1.0, 0.0, 55493.95, 103057.49, 214634.81],
       [1.0, 0.0, 0.0, 46426.07, 157693.92, 210797.67],
       [0.0, 0.0, 1.0, 46014.02, 85047.44, 205517.64],
       [0.0, 1.0, 0.0, 28663.76, 127056.21, 201126.82],
       [1.0, 0.0, 0.0, 44069.95, 51283.14, 197029.42],
       [0.0, 0.0, 1.0, 20229.59, 65947.93, 185265.1],
       [1.0, 0.0, 0.0, 38558.51, 82982.09, 174999.3],
       [1.0, 0.0, 0.0, 28754.33, 118546.05, 172795.67],
       [0.0, 1.0, 0.0, 27892.92, 84710.77, 164470.71],
       [1.0, 0.0, 0.0, 23640.93, 96189.63, 148001.11],
       [0.0, 0.0, 1.0, 15505.73, 127382.3, 35534.17],
       [1.0, 0.0, 0.0, 22177.74, 154806.14, 28334.72],
       [0.0, 0.0, 1.0, 1000.23, 124153.04, 1903.93],
       [0.0, 1.0, 0.0, 1315.46, 115816.21, 297114.46],
       [1.0, 0.0, 0.0, 0.0, 135426.92, 0.0],
       [0.0, 0.0, 1.0, 542.05, 51743.15, 0.0],
       [1.0, 0.0, 0.0, 0.0, 116983.8, 45173.06]], dtype=object)
```

In [10]:
```
from sklearn.model_selection import train_test_split
xtrain, xtest, ytrain, ytest= train_test_split(x,y, test_size=0.2, random_state=
```

In [11]:
```
from sklearn.linear_model import LinearRegression
linreg = LinearRegression()
linreg.fit(xtrain, ytrain)
ypred = linreg.predict(xtest)
```

In [12]:
```
linreg.coef_
```

Out[12]:
```
array([ 8.66383692e+01, -8.72645791e+02,  7.86007422e+02,  7.73467193e-01,
        3.28845975e-02,  3.66100259e-02])
```

In [13]:
```
linreg.intercept_
```

Out[13]:
```
42467.52924852117
```

In [14]:
```
dic = {"Acutal Y":ytest, "Predicted Y":ypred}
df =pd.DataFrame(dic)
df
```

Out[14]:

	Acutal Y	Predicted Y
0	103282.38	103015.201598
1	144259.40	132582.277608
2	146121.95	132447.738452
3	77798.83	71976.098513
4	191050.39	178537.482211
5	105008.31	116161.242302
6	81229.06	67851.692097
7	97483.56	98791.733747
8	110352.25	113969.435330
9	166187.94	167921.065696

In [15]:
```
from sklearn.metrics import r2_score
```

In [16]:
```
print(r2_score(ytest, ypred))
```

```
0.9347068473282567
```

In [17]:
```
from sklearn.metrics import mean_absolute_error, mean_squared_error
print(f"MAE -: {mean_absolute_error(ytest, ypred)}")
print(f"MSE -: {mean_squared_error(ytest, ypred)}")
print(f"RMSE -: {np.sqrt(mean_squared_error(ytest, ypred))}")
print(f"R-Squared -: {r2_score(ytest, ypred)}")
```

```
MAE -: 7514.293659643097
MSE -: 83502864.03255911
RMSE -: 9137.990152793946
R-Squared -: 0.9347068473282567
```

In [ ]: