In [23]:

```
#importing the packages
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
%matplotlib inline
```

In [ ]:

```

```

In [24]:

```
#reading the datasets
companies=pd.read_csv("1000_Companies.csv")
x=companies.iloc[:, :-1].values
y=companies.iloc[:,4].values
```

In [25]:

```
#displaying the data
companies.head()
```

Out[25]:

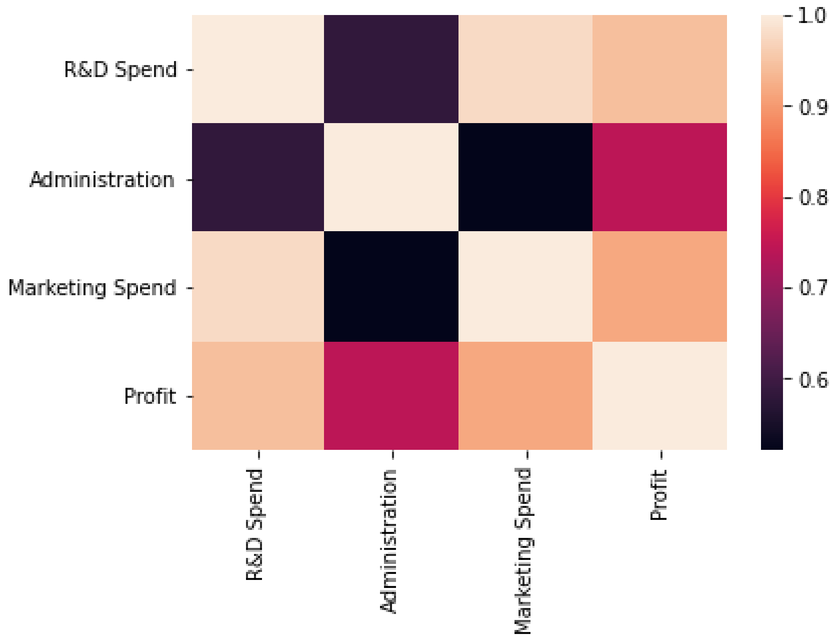
	R&D Spend	Administration	Marketing Spend	State	Profit
0	165349.20	136897.80	471784.10	New York	192261.83
1	162597.70	151377.59	443898.53	California	191792.06
2	153441.51	101145.55	407934.54	Florida	191050.39
3	144372.41	118671.85	383199.62	New York	182901.99
4	142107.34	91391.77	366168.42	Florida	166187.94

In [26]:

```
#display in visualise manner
sns.heatmap(companies.corr())
```

Out[26]:

```
<AxesSubplot:>
```



In [27]:

```
#Here we are coverting normal string data into integer to get the proper result
from sklearn.preprocessing import LabelEncoder, OneHotEncoder

labelencoder = LabelEncoder()
x[:, 3] = labelencoder.fit_transform(x[:, 3])
# transform = make_column_transformer((OneHotEncoder(), x[:, 3]), remainder = 'passthrough'

onehotencoder = OneHotEncoder()
enc_data = onehotencoder.fit_transform(x).toarray()
```

In [10]:

```
companies.
```

Out[10]:

	R&D Spend	Administration	Marketing Spend	State	Profit
0	165349.20	136897.800	471784.1000	New York	192261.83000
1	162597.70	151377.590	443898.5300	California	191792.06000
2	153441.51	101145.550	407934.5400	Florida	191050.39000
3	144372.41	118671.850	383199.6200	New York	182901.99000
4	142107.34	91391.770	366168.4200	Florida	166187.94000
...	...	...	...	...	...
995	54135.00	118451.999	173232.6695	California	95279.96251
996	134970.00	130390.080	329204.0228	California	164336.60550
997	100275.47	241926.310	227142.8200	California	413956.48000
998	128456.23	321652.140	281692.3200	California	333962.19000
999	161181.72	270939.860	295442.1700	New York	476485.43000

1000 rows × 5 columns

In [31]:

```
#avoid dummy values
x=x[:,1:]
```

In [52]:

```
#training the model using sklearn
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2  , random_state =
```

In [53]:

```
#Here we are training using LinearRegression
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(x_train, y_train)
```

Out[53]:

```
LinearRegression()
```

In [54]:

```
#to see the ouput after converting string into int
y_pred=regressor.predict(x_test)

y_pred
```

Out[54]:

```
array([ 90209.64901318,  88876.28325862,  95200.74973042, 174739.58503941,
        84013.74722284, 110572.81423949, 169438.58451208,  91855.1682484 ,
       163402.29005648,  54991.76617392,  67874.72481309, 150180.9861358 ,
       126512.60670878,  60430.86889281, 175991.46628657,  76097.46542299,
       118577.95366564, 163338.33779446, 165329.70854714, 180487.66382609,
       101238.40282718,  86180.18269683, 179933.75346041,  84689.06190888,
       105088.13051962, 101399.34780971,  40828.42858138,  58070.59626831,
        69777.85472814, 226802.72627013, 121052.52594777, 111658.45981284,
       101689.86321283, 137852.59801618,  64960.45224275, 108854.95146996,
       184345.94907957, 170595.15880734, 173725.92497117, 118014.06326024,
        97038.60706862, 164196.76013804, 107746.47058118,  52030.50278345,
       116882.08703837,  59741.11018494, 157857.27897357,  79750.07622713,
       159139.45816291, 131182.41311202, 183735.81760226, 173691.52102156,
        94077.16471212,  79055.37228415, 179193.73688257,  85539.18255677,
       142685.10466661, 169290.41285941,  84946.0123958 , 105424.79762652,
       141471.73344522,  53812.88851425, 141109.98209805, 138544.12449302,
        98525.35315256, 114019.84013539, 126396.98673988, 151216.5527697 ,
        60009.80841183, 173375.17181713, 124272.14963365, 167730.30512327,
        92206.01101456, 155162.10688204,  84847.65607993,  78602.24617494,
       120695.89889888,  93761.83359006, 138350.10811473, 142832.70064453,
       170307.33979742, 139388.94182688, 106463.63123443, 154603.07232232,
       139379.05136376, 110182.62230992,  70537.26059843,  88645.70361798,
       139438.45008128, 147641.05170022, 156869.72126349,  59643.77197986,
        94081.53927638, 113100.16181875,  57752.56877436, 107485.68247072,
       147038.4722533 , 151224.944042  , 166542.28652602, 118528.35593888,
       121037.42174722, 138899.82289638, 156489.59877727, 122209.67591778,
        87611.90462751, 105420.51243038,  95673.35467317, 176889.50642142,
       180322.17764231, 109848.65149198, 164002.92284422, 166498.47315072,
       156953.6334675 , 173362.67443317, 168836.26856357,  53550.90329145,
       175428.41512306, 104839.75018099,  83286.04901321, 138264.69665116,
       144028.53976029, 161047.62283516, 168853.97964945, 120731.98115931,
       158311.24418492, 110180.76505437, 168367.37799216,  61822.13404496,
       157923.48016043, 157075.12729573, 173097.51172293, 155283.77969052,
       103825.25107934,  86188.75284512, 140875.86692656, 164693.69969128,
       121446.91360112, 176415.2867655 , 101228.25635896,  83227.39996098,
       177095.75155581, 101943.93836181,  71395.02274908,  90700.7147071 ,
        62340.05164744,  69835.09399318,  74112.01200262, 175404.91965022,
        90666.22114537, 150611.55534056,  93568.74599705,  63961.30747995,
       171417.40945363,  61952.19803214, 168222.12034272, 165172.65956631,
       164711.23169268, 102856.15401618, 180023.62903163,  75018.26432529,
        91653.77885545, 135192.49017143,  65934.01337805,  72635.24588968,
        61710.70977498, 183156.8229963 , 175332.00553509, 157838.90773028,
       140678.85216661, 153637.42131765,  59701.85037545,  91559.88668045,
       151949.94585843, 167617.95209287,  73329.20016737, 116442.3868544 ,
        81262.17503945, 148960.9914253 , 116669.87853678, 129716.3764601 ,
       173618.51732983, 298075.79813027, 145439.85431614, 149760.67520877,
        87148.61949834,  71365.56394454,  71492.27147469,  69912.11428869,
       120444.25162176,  90141.01995445, 166079.00146558, 125143.15876482,
        68170.27493788, 143636.00582325, 118326.96654593, 164632.26473798,
       167869.68891337, 146162.33882386, 140714.93439153, 109120.29312493])
```

In [55]:

```
#to check the linearRegression coefficient
print(regressor.coef_)
```

```
[   1.04239904    0.35510511 -320.45531609]
```

In [56]:

```
#to check the linearRegression Intercept
print(regressor.intercept_)
```

```
-88623.76527343778
```

In [57]:

```
#we can see the output
from sklearn.metrics import r2_score
r2_score(y_test,y_pred)
```

Out[57]:

```
0.8985038788872521
```

In [ ]: