Creado por:

Isabel Maniega

Ejercicio 3

In [1]:
```
# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
```

In [2]:
```
# Importing the dataset
dataset = pd.read_csv('50_Startups.csv')
dataset
```

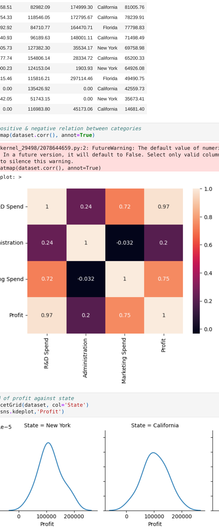
Out[2]:

	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
5	131876.90	99814.71	362861.36	New York	156991.12
6	134615.46	147198.87	127716.82	California	156122.51
7	130998.13	145530.06	323876.68	Florida	155752.60
8	120542.52	148718.95	311613.29	New York	152211.77
9	123334.88	108679.17	304781.62	California	149759.96
10	101913.08	110594.11	229160.95	Florida	146121.95
11	100671.96	91790.61	249744.55	California	144259.40
12	93863.75	127320.38	249839.44	Florida	141585.52
13	91992.39	135495.07	252664.93	California	134307.35
14	119943.24	156547.42	256512.92	Florida	132602.65
15	114523.61	122616.84	261776.23	New York	129917.04
16	78013.11	121597.55	264346.06	California	126992.93
17	94657.16	145077.58	282574.31	New York	125370.37
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	139953.94	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	214804.81	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
42	23640.93	96189.63	148001.11	California	71498.49
43	15505.73	127382.30	35534.17	New York	69758.98
44	22177.74	154806.14	28334.72	California	65200.33
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

In [3]:
```
#gives positive & negative relation between categories
sns.heatmap(dataset.corr(), annot=True)
```

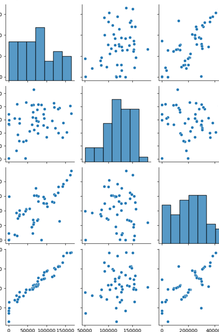
```
/tmp/ipykernel_29498/2078644659.py:2: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.
  sns.heatmap(dataset.corr(), annot=True)
```

Out[3]:
```
<AxesSubplot: >
```



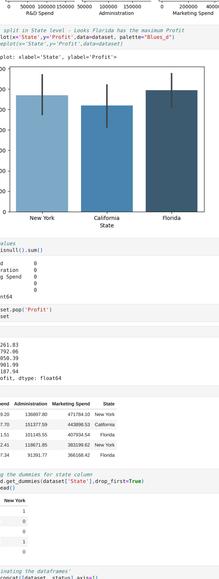
In [4]:
```
# spread of profit against state
gsns.FacetGrid(dataset, col='State')
gvg.map(sns.kdeplot,'Profit')
```

Out[4]:
```
<seaborn.axisgrid.FacetGrid at 0x7fbe327db820>
```



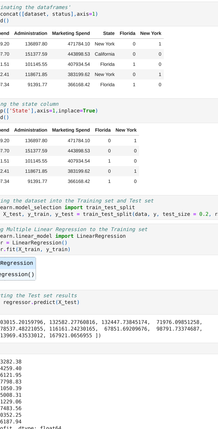
In [5]:
```
#Correlation  chart an different variables for comparision
# Profit Vs R & Spend is very linear and almost same for Marketing spend
# Profit spend vs Administration distribution is very scattered
sns.pairplot(dataset)
```

Out[5]:
```
<seaborn.axisgrid.PairGrid at 0x7fbe327db820>
```



In [6]:
```
# profit split in State level - Looks Florida has the maximum Profit
sns.barplot(x='State',y='Profit',data=dataset, palette="Blues_d")
#sns.lineplot(x='State',y='Profit',data=dataset)
```

Out[6]:
```
<AxesSubplot: xlabel='State', ylabel='Profit'>
```



In [7]:
```
# null values
dataset.isnull().sum()
```

Out[7]:
```
R&D Spend          0
Administration     0
Marketing Spend    0
State              0
Profit             0
dtype: int64
```

In [8]:
```
y = dataset.pop('Profit')
X = dataset
```

In [9]:
```
y.head()
```

Out[9]:
```
0    192261.83
1    191792.06
2    191050.39
3    182901.99
4    166187.94
Name: Profit, dtype: float64
```

In [10]:
```
X.head()
```

Out[10]:

	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 [11]:
```
# getting the dummies for state column
status=pd.get_dummies(dataset['State'],drop_first=True)
status.head()
```

Out[11]:

	Florida	New York
0	0	1
1	0	0
2	1	0
3	0	1
4	1	0

In [12]:
```
# concatinating the dataframes'
data=pd.concat([dataset, status],axis=1)
data.head()
```

Out[12]:

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

In [13]:
```
# dropping the state column
data.drop(['State'],axis=1,inplace=True)
data.head()
```

Out[13]:

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

In [14]:
```
# Splitting the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(data, y, test_size = 0.2, random_state = 0)
```

In [15]:
```
# Fitting Multiple Linear Regression to the Training set
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, y_train)
```

Out[15]:
```
▾ LinearRegression
LinearRegression()
```

In [16]:
```
# Predicting the Test set results
y_pred = regressor.predict(X_test)
y_pred
```

Out[16]:
```
array([103015.20159796, 132582.27760816, 132447.73845174,  71976.09851258,
       178537.48221055, 116161.24230165,  67851.69209676,  98791.73374687,
       114969.43533012, 167921.0656955 ])
```

In [17]:
```
y_test
```

Out[17]:
```
28    103282.38
11    144259.40
10    146121.95
41     77798.83
2     191050.39
27    105008.31
38     81229.06
31     97483.56
22    110352.25
4     166187.94
Name: Profit, dtype: float64
```

In [18]:
```
# predicting test results
from sklearn.metrics import r2_score
score=r2_score(y_test,y_pred)
score
```

Out[18]:
```
0.9347068473282423
```

In [19]:
```
testing_data_model_score = regressor.score(X_test, y_test)
print("Model Score/Performance on Testing data",testing_data_model_score)

training_data_model_score = regressor.score(X_train, y_train)
print("Model Score/Performance on Training data",training_data_model_score)
```

```
Model Score/Performance on Testing data 0.9347068473282423
Model Score/Performance on Training data 0.9501847627493607
```

Creado por:

Isabel Maniega