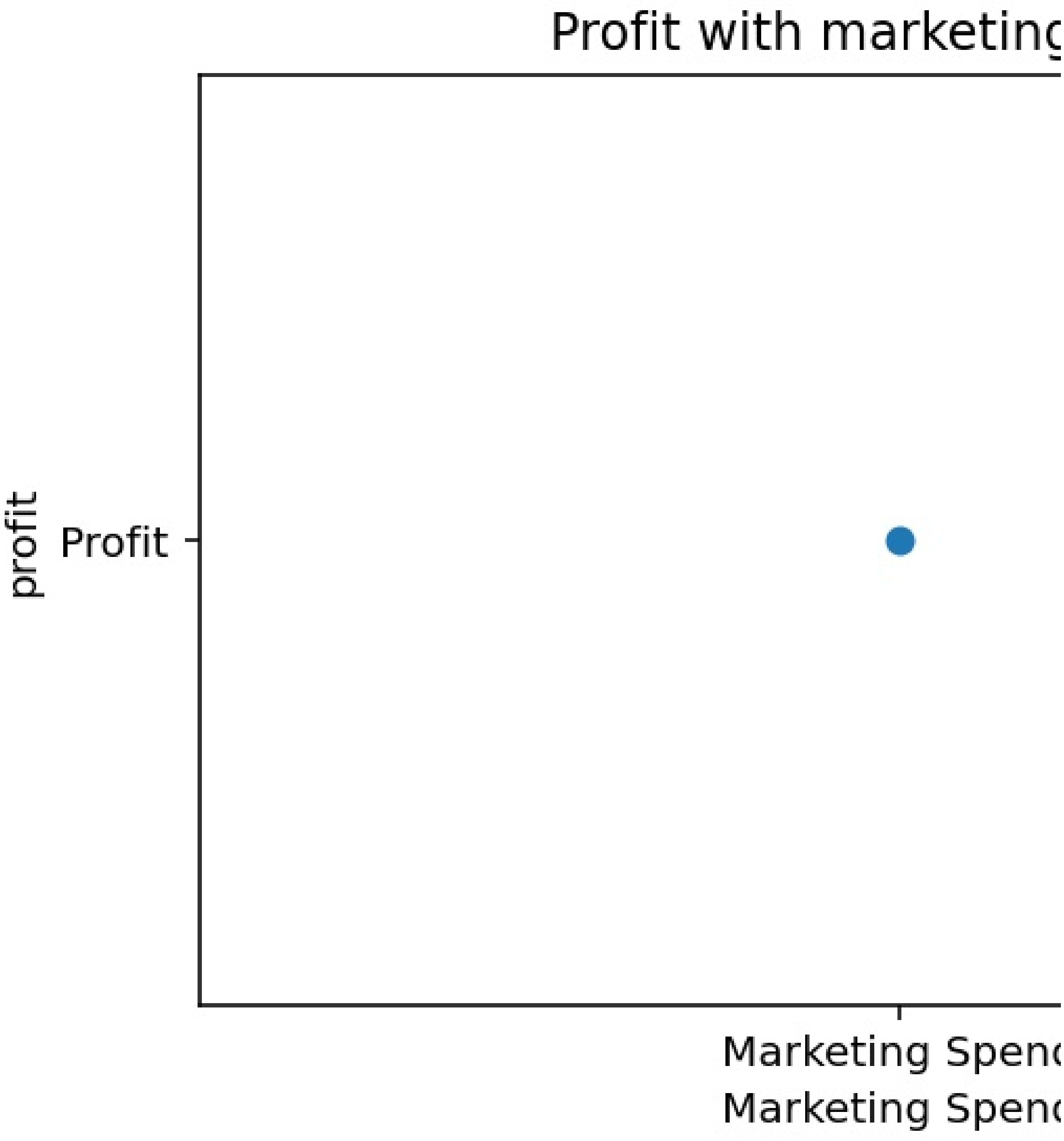
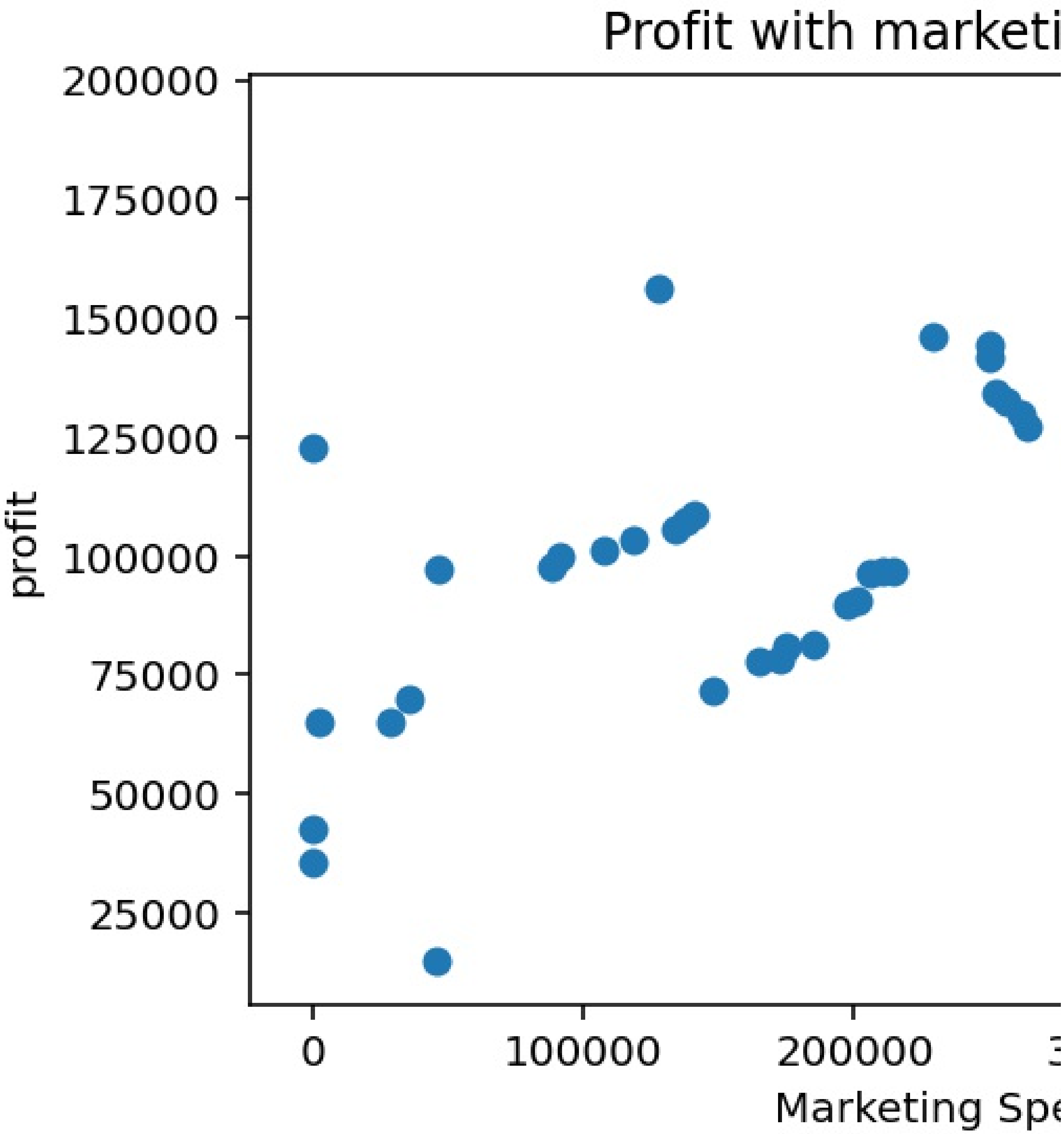
```
  File /opt/anaconda3/lib/python3.12/site-packages/pandas/io/parsers/
readers.py:1620 in __init__
    self._engine = self._make_engine(f, self.engine)

  File /opt/anaconda3/lib/python3.12/site-packages/pandas/io/parsers/
readers.py:1880 in _make_engine
    self.handles = get_handle(

  File /opt/anaconda3/lib/python3.12/site-packages/pandas/io/
common.py:873 in get_handle
    handle = open(

FileNotFoundError: [Errno 2] No such file or directory: '/Users/
babarhussain/MachineLearning/Multiple_Linear_Regression/MLR_insurnace/
insurance.csv 50_Startups'


In [36]: df = pd.read_csv("/Users/babarhussain/MachineLearning/
Multiple_Linear_Regression/MLRcompanies/50_Startups.csv")

In [37]: df.head()
Out[37]:
   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 [38]: # check null values

In [39]: df.isnull().sum()
Out[39]:
R&D Spend          0
Administration     0
Marketing Spend    0
State              0
Profit             0
dtype: int64

In [39]:

In [40]: df.dtypes
Out[40]:
R&D Spend          float64
Administration     float64
Marketing Spend    float64
State               object
Profit             float64
dtype: object

In [41]: # length of dataset

In [42]: len(df)
```

```
Out[42]: 50

In [43]: df.shape
Out[43]: (50, 5)

In [43]:

In [43]:

In [44]: plt.scatter(x='Marketing Spend', y ='Profit')
Out[44]: <matplotlib.collections.PathCollection at 0x13bb27d70>

In [45]: plt.title("Profit with marketing Spend")
Out[45]: Text(0.5, 1.0, 'Profit with marketing Spend')

In [46]: plt.xlabel("Marketing Spend")
Out[46]: Text(0.5, 22.359999999999992, 'Marketing Spend')

In [47]: plt.ylabel("profit")
Out[47]: Text(33.75, 0.5, 'profit')

In [48]: plt.show()
```



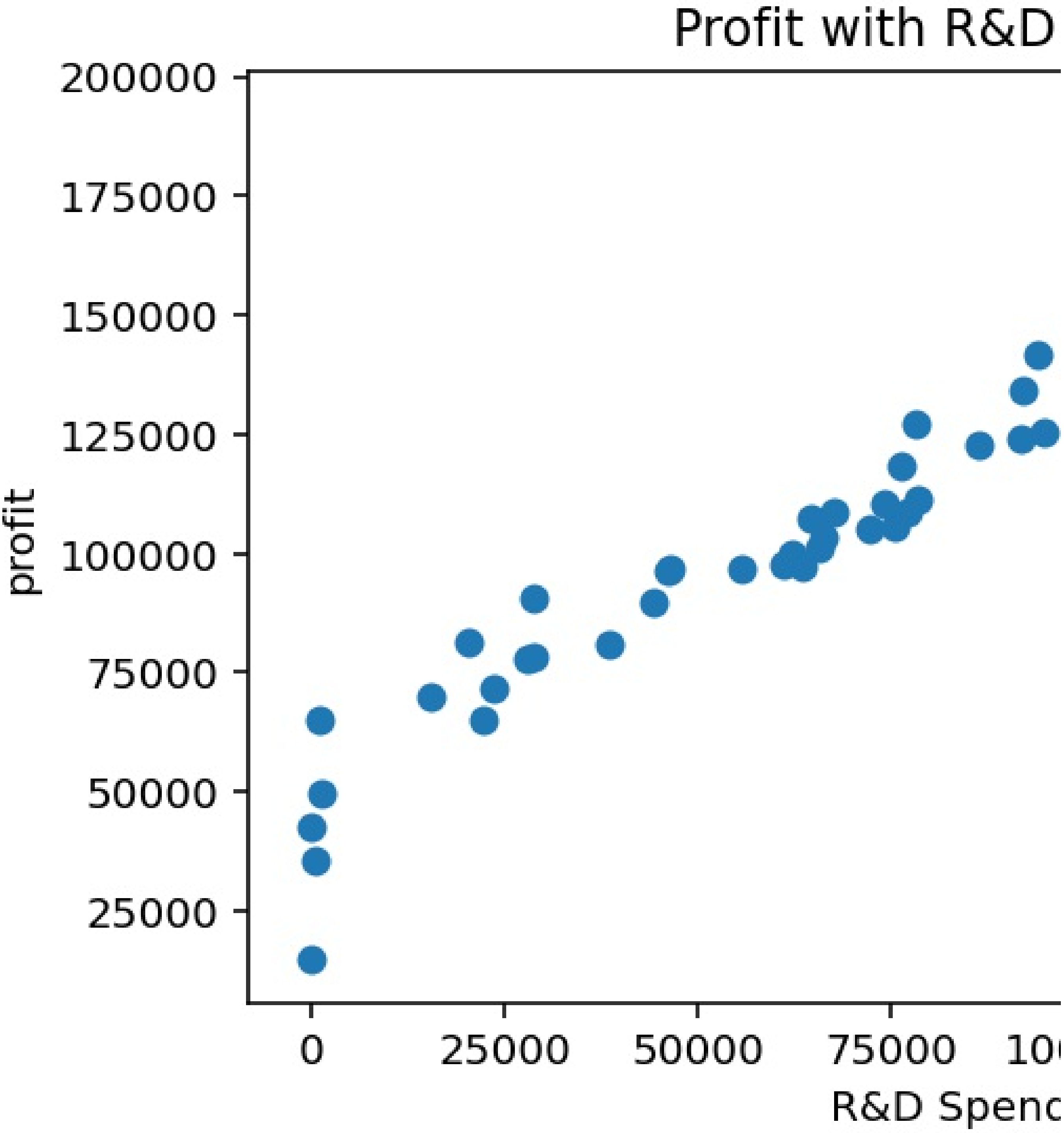
```
In [49]: plt.scatter(df['Marketing Spend'], df['Profit'])
Out[49]: <matplotlib.collections.PathCollection at 0x13c1030e0>

In [50]: plt.title("Profit with marketing Spend")
Out[50]: Text(0.5, 1.0, 'Profit with marketing Spend')

In [51]: plt.xlabel("Marketing Spend")
Out[51]: Text(0.5, 22.359999999999992, 'Marketing Spend')

In [52]: plt.ylabel("profit")
```

```
Out[52]: Text(9.5, 0.5, 'profit')

In [53]: plt.show()
```

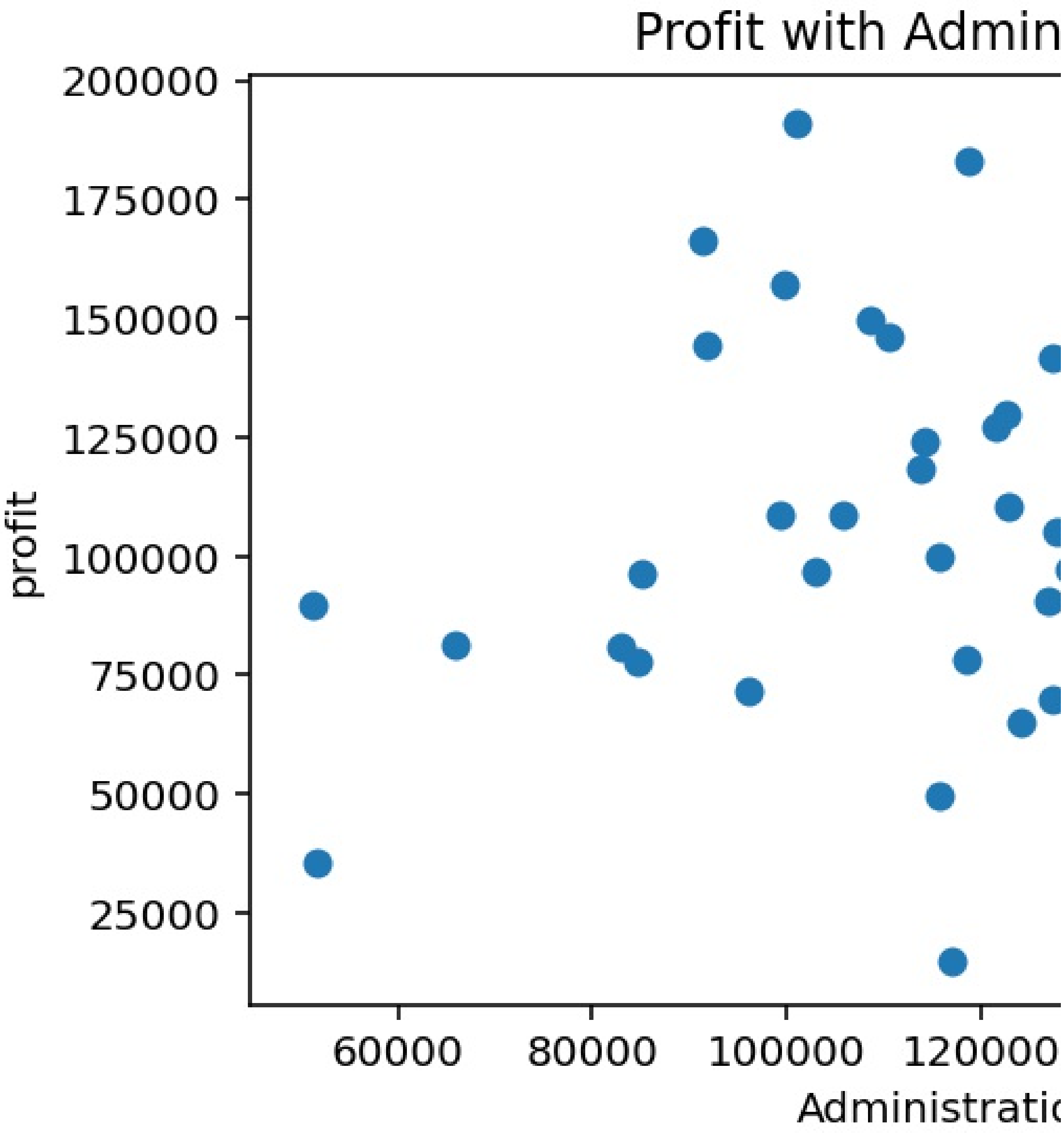


```
In [53]:

In [54]: plt.scatter(df['R&D Spend'], df['Profit'])
Out[54]: <matplotlib.collections.PathCollection at 0x13c1915b0>

In [55]: plt.title("Profit with R&D Spend")
```

```
Out[55]: Text(0.5, 1.0, 'Profit with R&D Spend')

In [56]: plt.xlabel("R&D Spend")
Out[56]: Text(0.5, 22.359999999999992, 'R&D Spend')

In [57]: plt.ylabel("profit")
Out[57]: Text(9.5, 0.5, 'profit')

In [58]: plt.show()
```



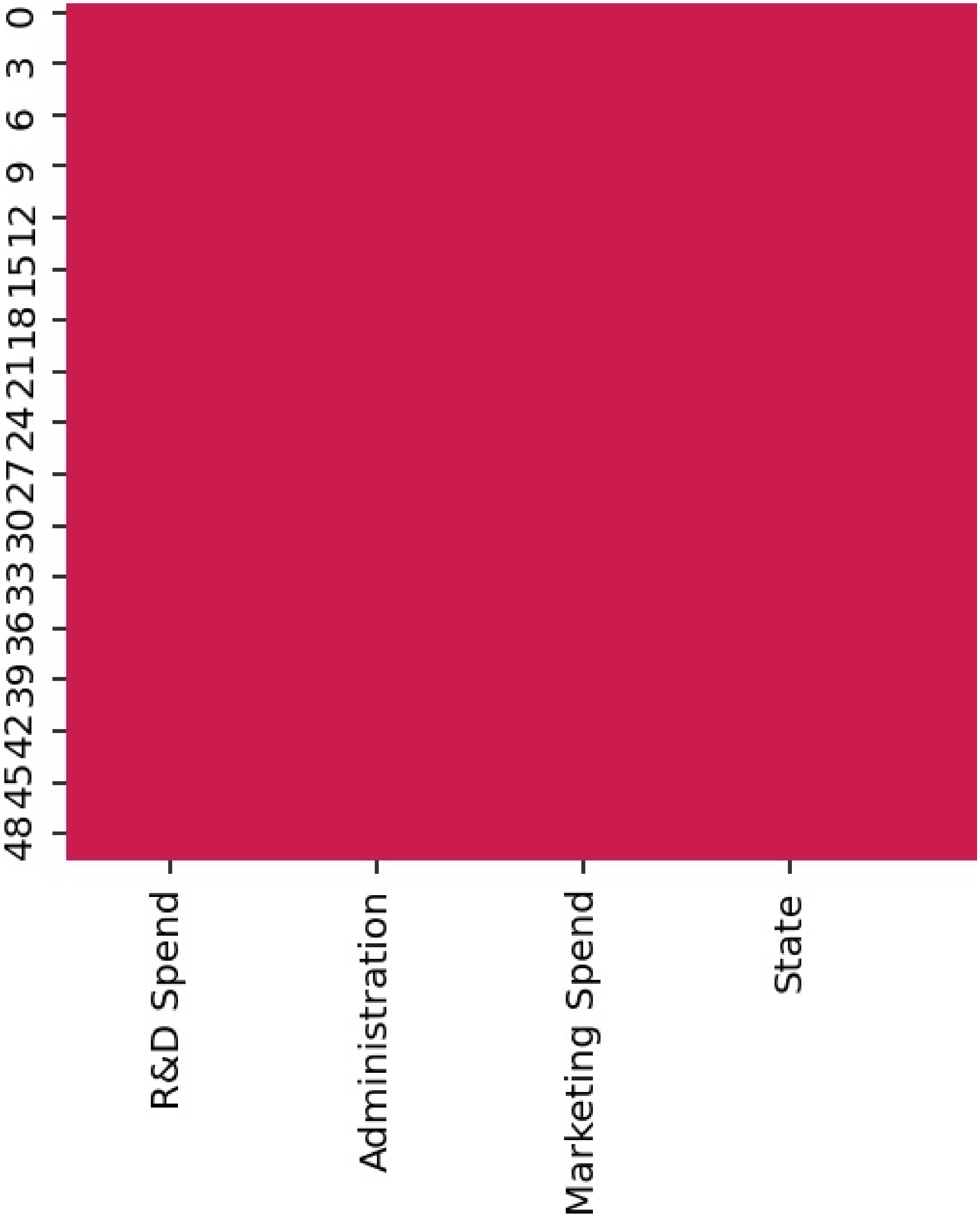
```
In [58]:

In [59]: plt.scatter(df['Administration'], df['Profit'])
Out[59]: <matplotlib.collections.PathCollection at 0x13c22c740>

In [60]: plt.title("Profit with Administration")
Out[60]: Text(0.5, 1.0, 'Profit with Administration')

In [61]: plt.xlabel("Administration")
Out[61]: Text(0.5, 22.359999999999992, 'Administration')

In [62]: plt.ylabel("profit")
Out[62]: Text(9.5, 0.5, 'profit')

In [63]: plt.show()
```



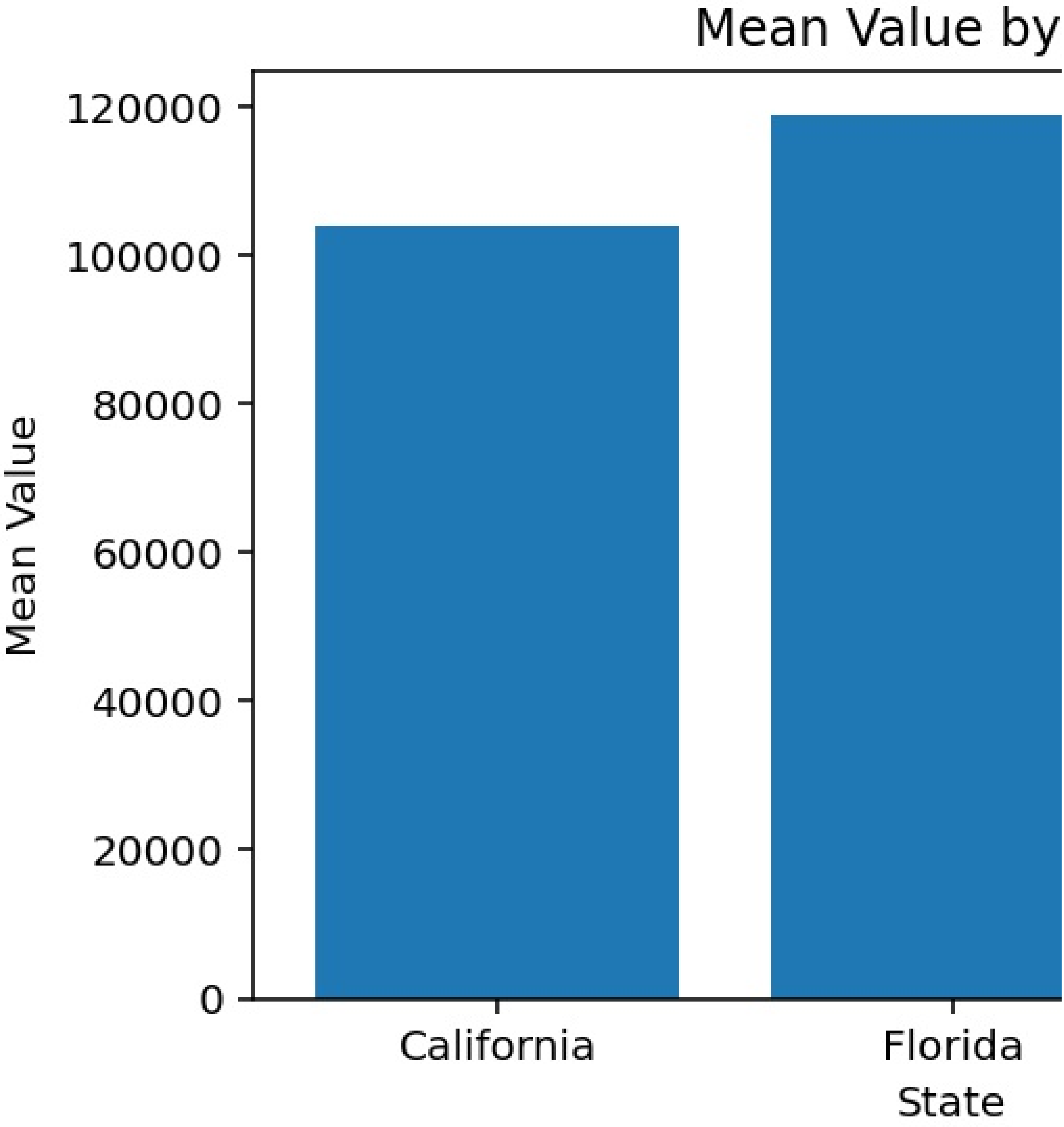
```
In [63]:

In [64]: ax = df.groupby([‘State’])
[‘profit’].mean().plot.bar(figuresize = (10,5), fontsize= 14)
Traceback (most recent call last):

  Cell In[64], line 1
    ax = df.groupby(['State'])['profit'].mean().plot.bar(figuresize =
(10,5), fontsize= 14)
```

```
  File /opt/anaconda3/lib/python3.12/site-packages/pandas/core/
groupby/generic.py:1951 in __getitem__
    return super().__getitem__(key)

  File /opt/anaconda3/lib/python3.12/site-packages/pandas/core/
base.py:244 in __getitem__
    raise KeyError(f"Column not found: {key}")

KeyError: 'Column not found: profit'


In [65]: ax = df.groupby(['State'])
['profit'].mean().plot.bar(figuresize = (10,5), fontsize= 14)
Traceback (most recent call last):

  Cell In[65], line 1
    ax = df.groupby(['State'])['profit'].mean().plot.bar(figuresize =
(10,5), fontsize= 14)

  File /opt/anaconda3/lib/python3.12/site-packages/pandas/core/
groupby/generic.py:1951 in __getitem__
    return super().__getitem__(key)

  File /opt/anaconda3/lib/python3.12/site-packages/pandas/core/
base.py:244 in __getitem__
    raise KeyError(f"Column not found: {key}")

KeyError: 'Column not found: profit'


In [66]: ax.set_title("Average profit")
Traceback (most recent call last):

  Cell In[66], line 1
    ax.set_title("Average profit")

NameError: name 'ax' is not defined


In [67]: ax.set_xlabel("State")
Traceback (most recent call last):

  Cell In[67], line 1
    ax.set_xlabel("State")

NameError: name 'ax' is not defined


In [68]: ax.set_ylabel("Profit")
Traceback (most recent call last):

  Cell In[68], line 1
    ax.set_ylabel("Profit")

NameError: name 'ax' is not defined
```

```
In [69]: ax.show()
Traceback (most recent call last):

  Cell In[69], line 1
    ax.show()

NameError: name 'ax' is not defined


In [70]: sns.heatmap(df.isnull())
Out[70]: <Axes: >

In [71]: plt.show()
```

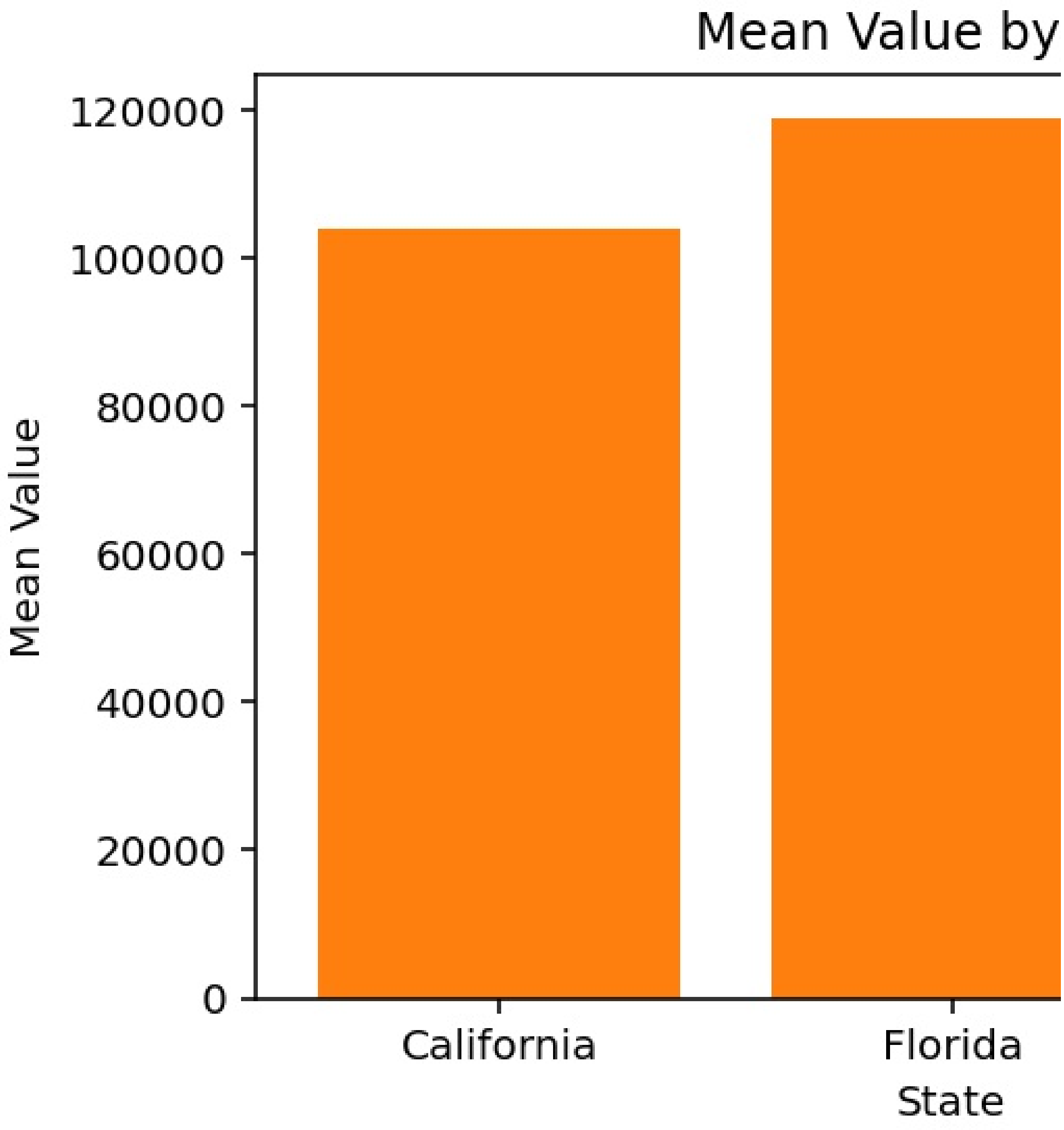


```
In [72]: mean_values = df.groupby('State')['profit'].mean()
Traceback (most recent call last):
  Cell In[72], line 1
    mean_values = df.groupby('State')['profit'].mean()

  File /opt/anaconda3/lib/python3.12/site-packages/pandas/core/
groupby/generic.py:1951 in __getitem__
    return super().__getitem__(key)

  File /opt/anaconda3/lib/python3.12/site-packages/pandas/core/
base.py:244 in __getitem__
    raise KeyError(f"Column not found: {key}")

KeyError: 'Column not found: profit'


In [73]: plt.bar(mean_values.index, mean_values)
Traceback (most recent call last):

  Cell In[73], line 1
    plt.bar(mean_values.index, mean_values)

NameError: name 'mean_values' is not defined


In [74]: mean_values = df.groupby('State')['Profit'].mean()

In [75]: plt.bar(mean_values.index, mean_values)
Out[75]: <BarContainer object of 3 artists>

In [76]: plt.xlabel('State')
Out[76]: Text(0.5, 22.359999999999992, 'State')

In [77]: plt.ylabel('Mean Value')
Out[77]: Text(9.5, 0.5, 'Mean Value')

In [78]: plt.title('Mean Value by State')
Out[78]: Text(0.5, 1.0, 'Mean Value by State')

In [79]: plt.show()
```



```
In [80]: mean_values = df.groupby('State')['Profit'].mean()

In [81]: plt.bar(mean_values.index, mean_values, Palette='Set2')
Traceback (most recent call last):

  Cell In[81], line 1
    plt.bar(mean_values.index, mean_values, Palette='Set2')

  File /opt/anaconda3/lib/python3.12/site-packages/matplotlib/
pyplot.py:2956 in bar
```

```
    return gca().bar(

  File /opt/anaconda3/lib/python3.12/site-packages/matplotlib/
__init__.py:1473 in inner
    return func(

  File /opt/anaconda3/lib/python3.12/site-packages/matplotlib/axes/
_axes.py:2591 in bar
    r._internal_update(kwargs)

  File /opt/anaconda3/lib/python3.12/site-packages/matplotlib/
artist.py:1216 in _internal_update
    return self._update_props(

  File /opt/anaconda3/lib/python3.12/site-packages/matplotlib/
artist.py:1190 in _update_props
    raise AttributeError(

AttributeError: Rectangle.set() got an unexpected keyword argument
'Palette'


In [82]: mean_values = df.groupby('State')['Profit'].mean()

In [83]: plt.bar(mean_values.index, mean_values)
Out[83]: <BarContainer object of 3 artists>

In [84]: plt.xlabel('State')
Out[84]: Text(0.5, 22.359999999999992, 'State')

In [85]: plt.ylabel('Mean Value')
Out[85]: Text(9.5, 0.5, 'Mean Value')

In [86]: plt.title('Mean Value by State')
Out[86]: Text(0.5, 1.0, 'Mean Value by State')

In [87]: plt.show()
```



```
In [88]: df["State"].value_counts()
Out[88]:
State
New York      17
California    17
Florida       16
Name: count, dtype: int64

In [89]: df['New York'] = np.where(df["State"]=='New York', 1,0)
In [90]: df['California'] = np.where(df["State"]=='California', 1,0)

In [91]: df['Florida'] = np.where(df["State"]=='Florida', 1,0)

In [92]: print(df)
    R&D Spend  Administration  Marketing Spend  ... New York
California  Florida
0   165349.20       136897.80        471784.10  ...        1
0        0
1   162597.70       151377.59        443898.53  ...        0
1        0
2   153441.51       101145.55        407934.54  ...        0
0        1
3   144372.41       118671.85        383199.62  ...        1
0        0
4   142107.34        91391.77        366168.42  ...        0
0        1
5   131876.90        99814.71        362861.36  ...        1
0        0
6   134615.46       147198.87        127716.82  ...        0
1        0
7   130298.13       145530.06        323876.68  ...        0
0        1
8   120542.52       148718.95        311613.29  ...        1
0        0
9   123334.88       108679.17        304981.62  ...        0
1        0
10  101913.08       110594.11        229160.95  ...        0
0        1
11  100671.96        91790.61        249744.55  ...        0
1        0
12   93863.75       127320.38        249839.44  ...        0
0        1
13   91992.39       135495.07        252664.93  ...        0
1        0
14  119943.24       156547.42        256512.92  ...        0
0        1
15  114523.61       122616.84        261776.23  ...        1
0        0
16   78013.11       121597.55        264346.06  ...        0
1        0
17   94657.16       145077.58        282574.31  ...        1
0        0
18   91749.16       114175.79        294919.57  ...        0
0        1
19   86419.70       153514.11             0.00  ...        1
0        0
20   76253.86       113867.30        298664.47  ...        0
1        0
21   78389.47       153773.43        299737.29  ...        1
0        0
22   73994.56       122782.75        303319.26  ...        0
0        1
23   67532.53       105751.03        304768.73  ...        0
0        1
```

```
24   77044.01        99281.34        140574.81  ...        1
0        0
25   64664.71       139553.16        137962.62  ...        0
1        0
26   75328.87       144135.98        134050.07  ...        0
0        1
27   72107.60       127864.55        353183.81  ...        1
0        0
28   66051.52       182645.56        118148.20  ...        0
0        1
29   65605.48       153032.06        107138.38  ...        1
0        0
30   61994.48       115641.28         91131.24  ...        0
0        1
31   61136.38       152701.92         88218.23  ...        1
0        0
32   63408.86       129219.61         46085.25  ...        0
1        0
33   55493.95       103057.49        214634.81  ...        0
0        1
34   46426.07       157693.92        210797.67  ...        0
1        0
35   46014.02        85047.44        205517.64  ...        1
0        0
36   28663.76       127056.21        201126.82  ...        0
0        1
37   44069.95        51283.14        197029.42  ...        0
1        0
38   20229.59        65947.93        185265.10  ...        1
0        0
39   38558.51        82982.09        174999.30  ...        0
1        0
40   28754.33       118546.05        172795.67  ...        0
1        0
41   27892.92        84710.77        164470.71  ...        0
0        1
42   23640.93        96189.63        148001.11  ...        0
1        0
43   15505.73       127382.30         35534.17  ...        1
0        0
44   22177.74       154806.14         28334.72  ...        0
1        0
45    1000.23       124153.04          1903.93  ...        1
0        0
46    1315.46       115816.21        297114.46  ...        0
0        1
47       0.00       135426.92             0.00  ...        0
1        0
48     542.05        51743.15             0.00  ...        1
0        0
49       0.00       116983.80         45173.06  ...        0
1        0

[50 rows x 8 columns]
```

```
In [93]: df.drop(columns=['State'], axis=1, inplace=True)

In [94]: print()


In [95]: print(df)
    R&D Spend  Administration  Marketing Spend  ...  New York
California  Florida
0   165349.20       136897.80        471784.10  ...         1
0        0
1   162597.70       151377.59        443898.53  ...         0
1        0
2   153441.51       101145.55        407934.54  ...         0
0        1
3   144372.41       118671.85        383199.62  ...         1
0        0
4   142107.34        91391.77        366168.42  ...         0
0        1
5   131876.90        99814.71        362861.36  ...         1
0        0
6   134615.46       147198.87        127716.82  ...         0
1        0
7   130298.13       145530.06        323876.68  ...         0
0        1
8   120542.52       148718.95        311613.29  ...         1
0        0
9   123334.88       108679.17        304981.62  ...         0
1        0
10  101913.08       110594.11        229160.95  ...         0
0        1
11  100671.96        91790.61        249744.55  ...         0
1        0
12   93863.75       127320.38        249839.44  ...         0
0        1
13   91992.39       135495.07        252664.93  ...         0
1        0
14  119943.24       156547.42        256512.92  ...         0
0        1
15  114523.61       122616.84        261776.23  ...         1
0        0
16   78013.11       121597.55        264346.06  ...         0
1        0
17   94657.16       145077.58        282574.31  ...         1
0        0
18   91749.16       114175.79        294919.57  ...         0
0        1
19   86419.70       153514.11             0.00  ...         1
0        0
20   76253.86       113867.30        298664.47  ...         0
1        0
21   78389.47       153773.43        299737.29  ...         1
0        0
22   73994.56       122782.75        303319.26  ...         0
0        1
23   67532.53       105751.03        304768.73  ...         0
```

```
0        1
24   77044.01        99281.34        140574.81  ...          1
0        0
25   64664.71       139553.16        137962.62  ...          0
1        0
26   75328.87       144135.98        134050.07  ...          0
0        1
27   72107.60       127864.55        353183.81  ...          1
0        0
28   66051.52       182645.56        118148.20  ...          0
0        1
29   65605.48       153032.06        107138.38  ...          1
0        0
30   61994.48       115641.28         91131.24  ...          0
0        1
31   61136.38       152701.92         88218.23  ...          1
0        0
32   63408.86       129219.61         46085.25  ...          0
1        0
33   55493.95       103057.49        214634.81  ...          0
0        1
34   46426.07       157693.92        210797.67  ...          0
1        0
35   46014.02        85047.44        205517.64  ...          1
0        0
36   28663.76       127056.21        201126.82  ...          0
0        1
37   44069.95        51283.14        197029.42  ...          0
1        0
38   20229.59        65947.93        185265.10  ...          1
0        0
39   38558.51        82982.09        174999.30  ...          0
1        0
40   28754.33       118546.05        172795.67  ...          0
1        0
41   27892.92        84710.77        164470.71  ...          0
0        1
42   23640.93        96189.63        148001.11  ...          0
1        0
43   15505.73       127382.30         35534.17  ...          1
0        0
44   22177.74       154806.14         28334.72  ...          0
1        0
45    1000.23       124153.04          1903.93  ...          1
0        0
46    1315.46       115816.21        297114.46  ...          0
0        1
47       0.00       135426.92             0.00  ...          0
1        0
48     542.05        51743.15             0.00  ...          1
0        0
49       0.00       116983.80         45173.06  ...          0
1        0

[50 rows x 7 columns]
```

```
In [96]: df1 = df.head()

In [97]: Dependent_Variable = 'Profit'

In [98]: Independent_Variable = df.columns.tolist()

In [99]: print(Independent_Variable)
['R&D Spend', 'Administration', 'Marketing Spend', 'Profit', 'New
York', 'California', 'Florida']

In [100]: Independent_Variable.remove(Dependent_Variable)

In [100]:

In [101]: print(Independent_Variable)
['R&D Spend', 'Administration', 'Marketing Spend', 'New York',
'California', 'Florida']

In [102]: X = df[Independent_Variable].values

In [103]: y = df[Dependent_Variable].values

In [104]: x_train,x_test,y_train,y_test = train_test_split(X,y,
test_size=0.2, random_state=0)

In [105]: scaler = MinMaxScaler()

In [106]: x_train = scaler.fit_transform(x_train)

In [107]: x_test = scaler.transform(x_test)

In [108]: model = LinearRegression()

In [109]: model.fit(x_train,y_train)
Out[109]: LinearRegression()

In [110]: # Predict the test set results

In [110]:

In [111]: y_pred = model.Predict(x_test)
Traceback (most recent call last):

  Cell In[111], line 1
    y_pred = model.Predict(x_test)

AttributeError: 'LinearRegression' object has no attribute 'Predict'


In [112]: y_pred = model.predict(x_test)

In [113]: import math *
  Cell In[113], line 1
    import math *
```

```
                ^
SyntaxError: invalid syntax


In [114]: import math

In [115]: # Predict the test set results

In [115]:

In [116]: y_pred = model.predict(x_test)

In [116]:

In [117]: # find the meansqured error

In [117]:

In [118]: math.sqrt(mean_squared_error(y_test, y_pred))
Out[118]: 9137.990152794939

In [118]:

In [118]:

In [119]: # Find the r2 score

In [119]:

In [120]: r2_score(y_test,y_pred)
Out[120]: 0.9347068473282426

In [121]:
```