hgtlceuh4

July 28, 2023

[1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

[2]:
```
df=pd.read_csv("/content/3_Fitness-1.csv")
df
```

[2]:
```
    Row Labels Sum of Jan Sum of Feb Sum of Mar  Sum of Total Sales
0            A      5.62%      7.73%      6.16%                  75
1            B      4.21%     17.27%     19.21%                 160
2            C      9.83%     11.60%      5.17%                 101
3            D      2.81%     21.91%      7.88%                 127
4            E     25.28%     10.57%     11.82%                 179
5            F      8.15%     16.24%     18.47%                 167
6            G     18.54%      8.76%     17.49%                 171
7            H     25.56%      5.93%     13.79%                 170
8  Grand Total    100.00%    100.00%    100.00%                1150
```

[3]:
```
df.head()
```

[3]:
```
  Row Labels Sum of Jan Sum of Feb Sum of Mar  Sum of Total Sales
0          A      5.62%      7.73%      6.16%                  75
1          B      4.21%     17.27%     19.21%                 160
2          C      9.83%     11.60%      5.17%                 101
3          D      2.81%     21.91%      7.88%                 127
4          E     25.28%     10.57%     11.82%                 179
```

1 DATA CLEANING AND DATA PREPROCESSING

[4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9 entries, 0 to 8
Data columns (total 5 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
```

```
 0   Row Labels          9 non-null      object
 1   Sum of Jan          9 non-null      object
 2   Sum of Feb          9 non-null      object
 3   Sum of Mar          9 non-null      object
 4   Sum of Total Sales  9 non-null      int64
dtypes: int64(1), object(4)
memory usage: 488.0+ bytes
```

[5]:
```
df.describe()
```

[5]:
```
       Sum of Total Sales
count            9.000000
mean           255.555556
std            337.332963
min             75.000000
25%            127.000000
50%            167.000000
75%            171.000000
max           1150.000000
```

[6]:
```
df.columns
```

[6]:
```
Index(['Row Labels', 'Sum of Jan', 'Sum of Feb', 'Sum of Mar',
       'Sum of Total Sales'],
      dtype='object')
```

[7]:
```
df1=df.dropna(axis=1)
df1
```

[7]:
```
    Row Labels Sum of Jan Sum of Feb Sum of Mar  Sum of Total Sales
0            A      5.62%      7.73%      6.16%                  75
1            B      4.21%     17.27%     19.21%                 160
2            C      9.83%     11.60%      5.17%                 101
3            D      2.81%     21.91%      7.88%                 127
4            E     25.28%     10.57%     11.82%                 179
5            F      8.15%     16.24%     18.47%                 167
6            G     18.54%      8.76%     17.49%                 171
7            H     25.56%      5.93%     13.79%                 170
8  Grand Total    100.00%    100.00%    100.00%                1150
```

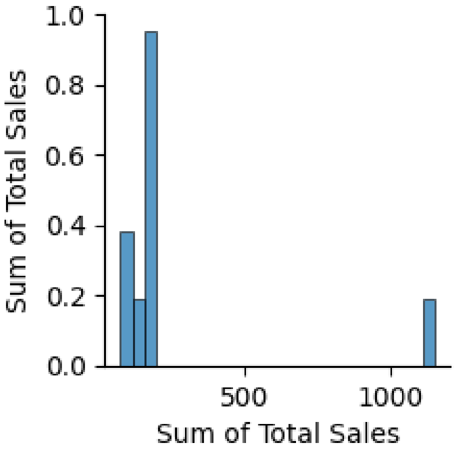
[8]:
```
df1.columns
```

[8]:
```
Index(['Row Labels', 'Sum of Jan', 'Sum of Feb', 'Sum of Mar',
       'Sum of Total Sales'],
      dtype='object')
```

2 EDA AND VISUALIZATION

[9]:
```
sns.pairplot(df1)
```

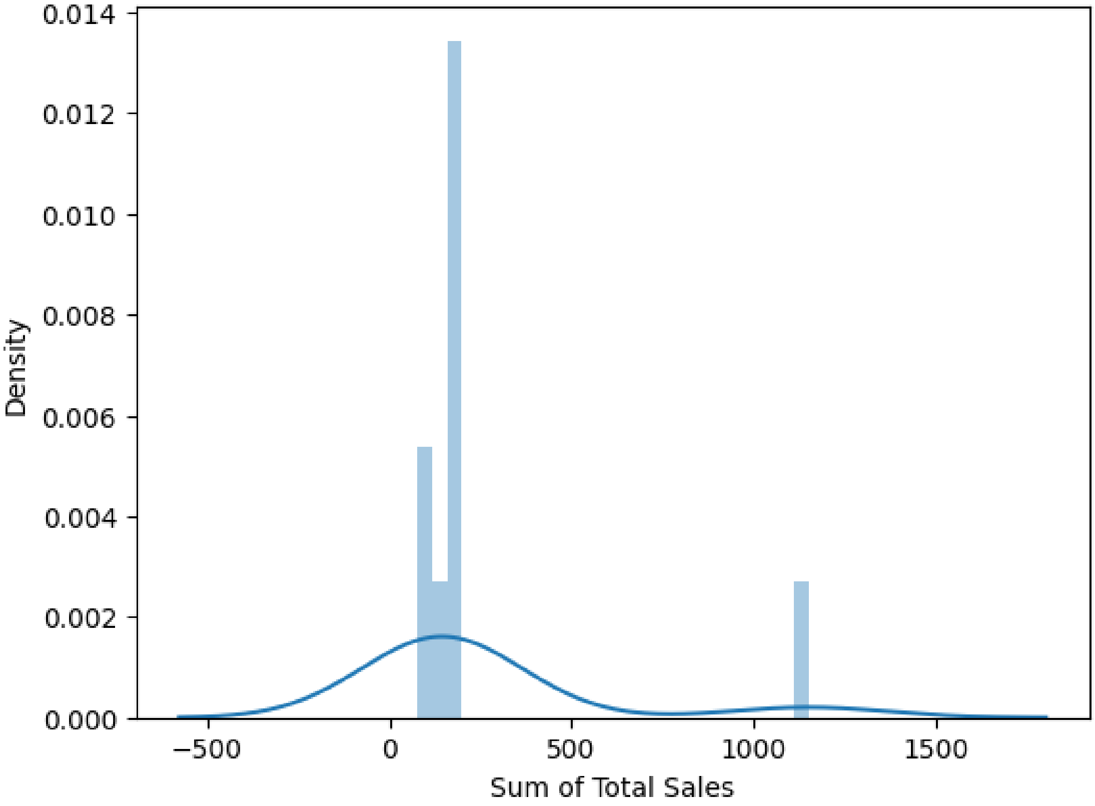
[9]:
```
<seaborn.axisgrid.PairGrid at 0x7ca8325d7520>
```



[10]:
```
sns.distplot(df1['Sum of Total Sales'])
```

```
<ipython-input-10-269cd82fce18>:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df1['Sum of Total Sales'])
```

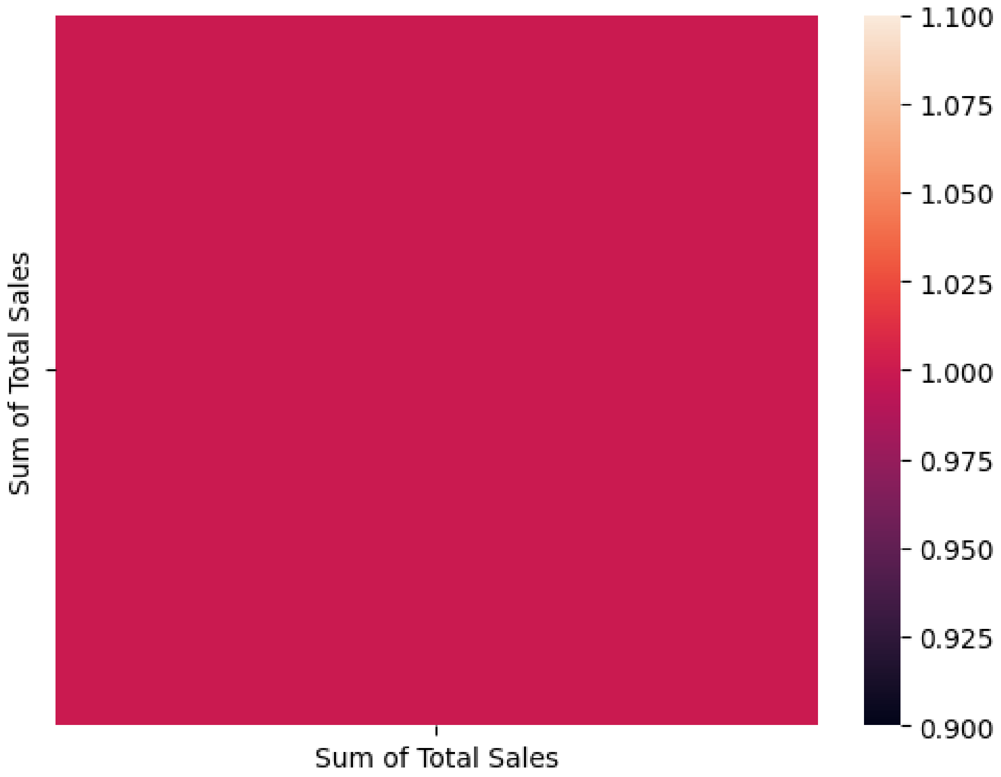
[10]:
```
<Axes: xlabel='Sum of Total Sales', ylabel='Density'>
```



[11]:
```
sns.heatmap(df1.corr())
```

```
<ipython-input-11-3ed1a1a51dc0>:1: 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(df1.corr())
```

[11]:
```
<Axes: >
```



3 TO TRAIN THE MODEL AND MODEL BULDING

[12]:
```
x=df[['Sum of Total Sales','Sum of Total Sales' ]]
y=df['Sum of Total Sales']
```

[13]:
```
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)
```

[14]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
```

[14]:
```
LinearRegression()
```

[15]:
```
lr.intercept_
```

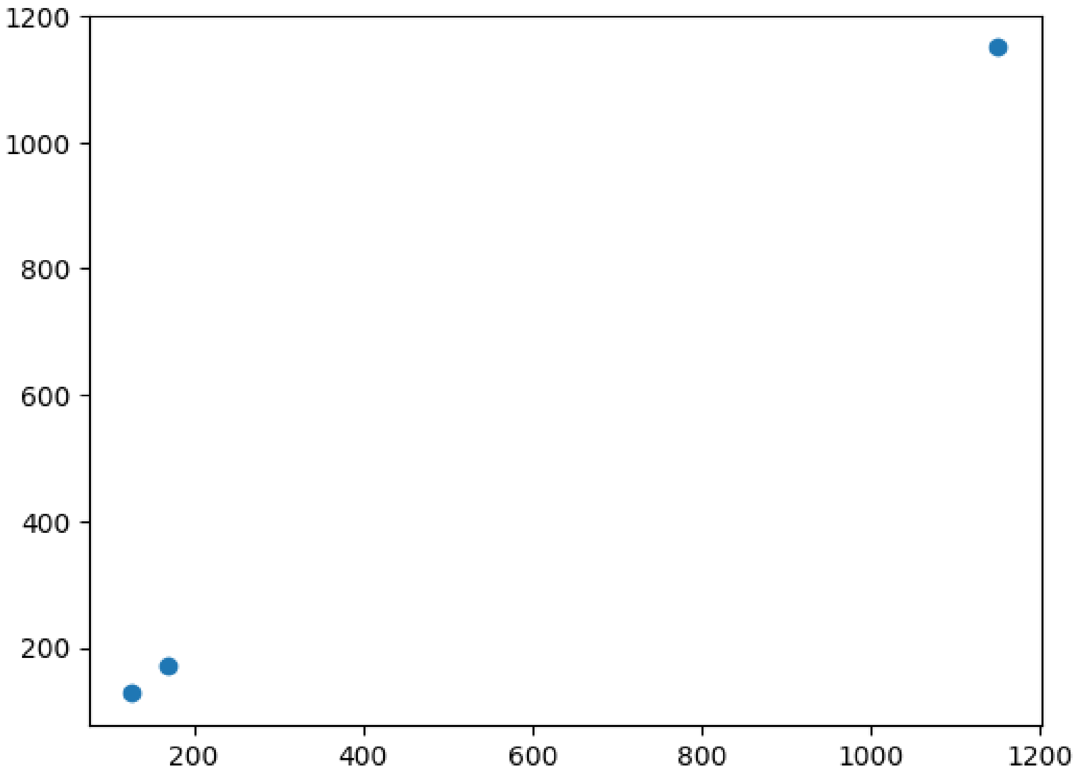
[15]:
```
2.842170943040401e-14
```

[16]:
```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

```
[16]:                        Co-efficient
      Sum of Total Sales              0.5
      Sum of Total Sales              0.5
```

```
[17]: prediction =lr.predict(x_test)
      plt.scatter(y_test,prediction)
```

```
[17]: <matplotlib.collections.PathCollection at 0x7ca82d40b580>
```



4 ACCURACY

```
[18]: lr.score(x_test,y_test)
```

```
[18]: 1.0
```

```
[19]: lr.score(x_train,y_train)
```

```
[19]: 1.0
```

```
[20]: from sklearn.linear_model import Ridge,Lasso
      rr=Ridge(alpha=10)
```

```
rr.fit(x_train,y_train)
```

```
[20]: Ridge(alpha=10)
```

```
[21]: rr.score(x_test,y_test)
```

```
[21]: 0.9999995642714175
```

```
[22]: rr.score(x_train,y_train)
```

```
[22]: 0.9999997130409505
```

```
[23]: la=Lasso(alpha=10)
      la.fit(x_train,y_train)
```

```
[23]: Lasso(alpha=10)
```

```
[24]: la.score(x_train,y_train)
```

```
[24]: 0.9999586335899656
```

```
[25]: la.score(x_test,y_test)
```

```
[25]: 0.9999371878069674
```