Type *Markdown* and LaTeX: α^2

In [9]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

In [10]:

```
df = pd.read_csv(r"E:\154\3_Fitness-1 - 3_Fitness-1.csv")
df
```

Out[10]:

	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

In [11]:

```
df.head()
```

Out[11]:

	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

Data cleaning and pre processing

In [12]:

```
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
```

In [13]:

```
df.describe()
```

Out[13]:

	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

In [14]:

```
df.columns
```

Out[14]:

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

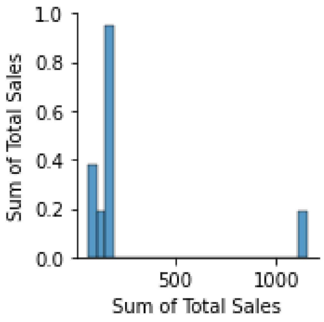
EDA and VISUALIZATION

In [15]:

```
sns.pairplot(df)
```

Out[15]:

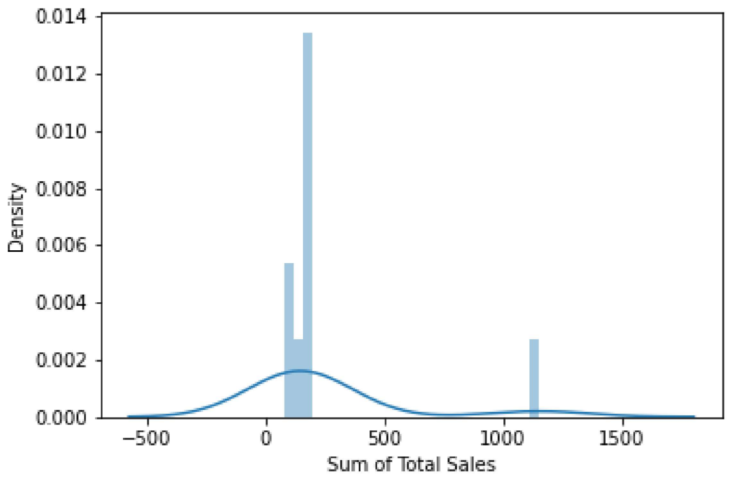
```
<seaborn.axisgrid.PairGrid at 0x201d40f67f0>
```



In [16]:
```
sns.distplot(df["Sum of Total Sales"])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarnin
g: `distplot` is a deprecated function and will be removed in a future version. Please
adapt your code to use either `displot` (a figure-level function with similar flexibili
ty) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

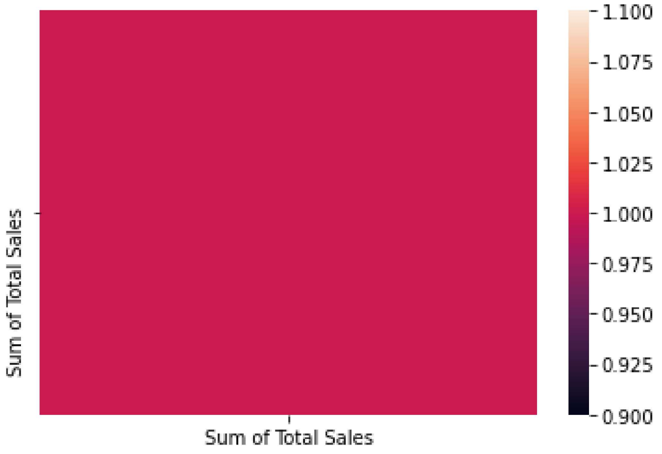
Out[16]:
```
<AxesSubplot:xlabel='Sum of Total Sales', ylabel='Density'>
```



In [17]:
```
df1 = df[['Row Labels', 'Sum of Jan', 'Sum of Feb', 'Sum of Mar',
       'Sum of Total Sales']]
```

In [18]:
```
sns.heatmap(df1.corr())
```

Out[18]:
```
<AxesSubplot:>
```



In [19]:
```
x = df1[['Sum of Total Sales','Sum of Total Sales']]
y = df1['Sum of Total Sales']
```

split the data into training and test data

In [20]:
```
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=0.3)
```

In [21]:
```
lr = LinearRegression()
lr.fit(x_train, y_train)
```

Out[21]:
```
LinearRegression()
```

In [22]:
```
lr.intercept_
```

Out[22]:
```
0.0
```

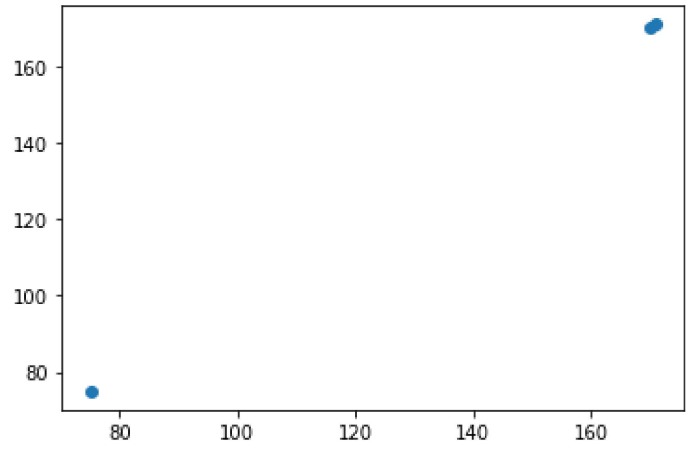
In [23]:
```
coeff = pd.DataFrame(lr.coef_, x.columns, columns =['Co-efficient'])
coeff
```

Out[23]:

	Co-efficient
Sum of Total Sales	0.5
Sum of Total Sales	0.5

In [24]:
```
prediction = lr.predict(x_test)
plt.scatter(y_test, prediction)
```

Out[24]:
```
<matplotlib.collections.PathCollection at 0x201d6d95ca0>
```



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

Out[25]:
```
1.0
```

Accuracy

In [26]:
```
lr.score(x_train,y_train)
```

Out[26]:
```
1.0
```

In [27]:
```
from sklearn.linear_model import Ridge,Lasso
```

In [28]:
```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[28]:
```
Ridge(alpha=10)
```

In [29]:
```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[29]:
```
Ridge(alpha=10)
```

In [30]:
```
rr.score(x_train,y_train)
```

Out[30]:
```
0.9999999999648033
```

ElasticNet

In [31]:
```
from sklearn.linear_model import ElasticNet
en = ElasticNet()
en.fit(x_train,y_train)
```

Out[31]:
```
ElasticNet()
```

In [32]:
```
print(en.coef_)
```

```
[9.99989321e-01 7.11916051e-06]
```

In [33]:
```
print(en.intercept_)
```

```
0.0011177320716342365
```

In [34]:
```
print(en.predict(x_test))
```

```
[171.00050903  75.00085076 170.00051259]
```

In [35]:
```
print(en.score(x_test,y_test))
```

```
0.9999999997951458
```

Evaluation Metrics

In [36]:
```
from sklearn import metrics
```

In [37]:
```
print("Mean Absolute Error",metrics.mean_absolute_error(y_test,prediction))
```

```
Mean Absolute Error 0.0
```

In [38]:
```
print("Mean Squared Error:",metrics.mean_squared_error(y_test,prediction))
```

```
Mean Squared Error: 0.0
```

In [39]:
```
print("Root Mean Absolute Error:",np.sqrt(metrics.mean_squared_error(y_test,prediction))
```

```
Root Mean Absolute Error: 0.0
```

In [ ]:
```

```

```
In [ ]:
```