Business Problem - Identify relation between total advertising spend and sales?

our next advertising campaign will have a total spend of $ 2,00,000 how many units do we expect to sell as a result of this?

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

In [2]:
```
df = pd.read_csv("Advertising.csv")
df
```

Out[2]:

	TV	radio	newspaper	sales
0	230100	37800	69200	22100
1	44500	39300	45100	10400
2	17200	45900	69300	9300
3	151500	41300	58500	18500
4	180800	10800	58400	12900
...	...	...	...	...
195	38200	3700	13800	7600
196	94200	4900	8100	9700
197	177000	9300	6400	12800
198	283600	42000	66200	25500
199	232100	8600	8700	13400

200 rows × 4 columns

In [3]:
```
df.shape
```

Out[3]:
```
(200, 4)
```

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   TV         200 non-null    int64
 1   radio      200 non-null    int64
 2   newspaper  200 non-null    int64
 3   sales      200 non-null    int64
dtypes: int64(4)
memory usage: 6.4 KB
```

Data Preprocessing

In [5]:
```
# By combining all the features, we get "total_spend"
```

In [6]:
```
df["total_spend"] = df["TV"] + df["radio"] + df["newspaper"]
df.head()
```

Out[6]:

	TV	radio	newspaper	sales	total_spend
0	230100	37800	69200	22100	337100
1	44500	39300	45100	10400	128900
2	17200	45900	69300	9300	132400
3	151500	41300	58500	18500	251300
4	180800	10800	58400	12900	250000

In [7]:
```
# here, we are going to drop remaingng three columns
```

In [8]:
```
df.drop(columns = ["TV", "radio", "newspaper"], inplace = True)
```

In [9]:
```
df
```

Out[9]:

	sales	total_spend
0	22100	337100
1	10400	128900
2	9300	132400
3	18500	251300
4	12900	250000
...	...	...
195	7600	55700
196	9700	107200
197	12800	192700
198	25500	391800
199	13400	249400

200 rows × 2 columns

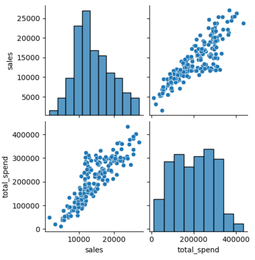
In [10]:
```
# Exploratory Data Analysis
```

In [11]:
```
df.describe()
```

Out[11]:

	sales	total_spend
count	200.000000	200.000000
mean	14022.500000	200860.500000
std	5217.456566	92985.180587
min	1600.000000	11700.000000
25%	10375.000000	123550.000000
50%	12900.000000	207350.000000
75%	17400.000000	281125.000000
max	27000.000000	433600.000000

In [12]:
```
sns.pairplot(df)
plt.show()
```



In [13]:
```
df.corr()
```

Out[13]:

	sales	total_spend
sales	1.000000	0.867712
total_spend	0.867712	1.000000

creating x and y

In [14]:
```
x = df[["total_spend"]]
y = df["sales"]
```

In [15]:
```
# Train Test Split
```

In [16]:
```
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 = 9)
```

Modelling

In [17]:
```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr.fit(x_train, y_train)

print("Inetercept:", lr.intercept_)
print("Coefficient:", lr.coef_)
```

```
Inetercept: 4064.8025773895315
Coefficient: [0.04927216]
```

Prediction

In [18]:
```
ypred_test = lr.predict(x_test)
```

Evaluation

In [19]:
```
print("Test R2:", lr.score(x_test, y_test))
```

```
Test R2: 0.7451800140806997
```

Model selection

In [20]:
```
ypred_train = lr.predict(x_train)
```

In [21]:
```
print("Train R2:", lr.score(x_train, y_train))
```

```
Train R2: 0.751955367544253
```

cross validation score

In [22]:
```
from sklearn.model_selection import cross_val_score
s = cross_val_score(lr, x, y, cv = 5)
print("Cross validation score:", s.mean())
```

```
Cross validation score: 0.7433783178555419
```

From modelling all in a single cell upto cross validation score.

In [23]:
```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr.fit(x_train, y_train)

print("Inetercept:", lr.intercept_)
print("Coefficient:", lr.coef_)

ypred_test = lr.predict(x_test)
ypred_train = lr.predict(x_train)

print("Test R2:", lr.score(x_test, y_test))
print("Train R2:", lr.score(x_train, y_train))

from sklearn.model_selection import cross_val_score
s = cross_val_score(lr, x, y, cv = 5)
print("Cross validation score:", s.mean())
```

```
Inetercept: 4064.8025773895315
Coefficient: [0.04927216]
Test R2: 0.7451800140806997
Train R2: 0.751955367544253
Cross validation score: 0.7433783178555419
```

a 1 unit increase in total spend is associated with an increase of 0.049 units in sales. This basically means that for every $ 1000 dollars spend on ads, we could expect 49 more units sold.

For a total spend of 2,00,000 on ads, how many units could we expect to be sold?

In [24]:
```
# use the model to make prediction on a new value
```

In [25]:
```
lr.predict([[200000]])
```

```
C:\Users\siddh\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarning: X does not have valid feature names, but LinearRegression was fitted with feature names
  warnings.warn(
```

Out[25]:
```
array([13919.23443373])
```

Save a model

In [26]:
```
from joblib import dump
dump(lr, "sales_lr.joblib")
```

Out[26]:
```
['sales_lr.joblib']
```

Load a model

In [27]:
```
from joblib import load
loaded_lr = load("sales_lr.joblib")
loaded_lr.predict([[200000]])
```

```
C:\Users\siddh\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarning: X does not have valid feature names, but LinearRegression was fitted with feature names
  warnings.warn(
```

Out[27]:
```
array([13919.23443373])
```

In [ ]:
```

```