Objective : How Sales is impacted by TV ad, Radio Ad , Newspaper AD?

In [1]:
```
# import the library
import pandas as pd # handle the data set
import numpy as np # numerical python
import matplotlib.pyplot as plt # data visualisation library
```

Read and understand data

In [4]:
```
# read the csv file
df = pd.read_csv('Advertising Budget and Sales.csv',index_col=0)
# display first five rows
df.head()
```

Out[4]:

	TV Ad Budget ($)	Radio Ad Budget ($)	Newspaper Ad Budget ($)	Sales ($)
1	230.1	37.8	69.2	22.1
2	44.5	39.3	45.1	10.4
3	17.2	45.9	69.3	9.3
4	151.5	41.3	58.5	18.5
5	180.8	10.8	58.4	12.9

In [5]:
```
# change the columns names
df.columns = ['TV','Radio','Newspaper','Sales']
df.head()
```

Out[5]:

	TV	Radio	Newspaper	Sales
1	230.1	37.8	69.2	22.1
2	44.5	39.3	45.1	10.4
3	17.2	45.9	69.3	9.3
4	151.5	41.3	58.5	18.5
5	180.8	10.8	58.4	12.9

In [6]:
```
# check number of rows and columns
df.shape # 200 rows and 4columns
```

Out[6]:
```
(200, 4)
```

In [7]:
```
# check null values
df.isna().sum() # this has no null values
```

Out[7]:
```
TV           0
Radio        0
Newspaper    0
Sales        0
dtype: int64
```

In [8]:
```
# check the data type of columns
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 200 entries, 1 to 200
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   TV         200 non-null    float64
 1   Radio      200 non-null    float64
 2   Newspaper  200 non-null    float64
 3   Sales      200 non-null    float64
dtypes: float64(4)
memory usage: 7.8 KB
```

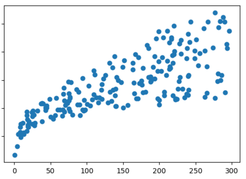
In [9]:
```
# check the statistical summary
df.describe()
```

Out[9]:

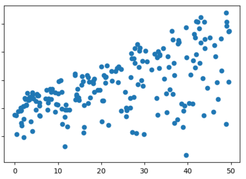
	TV	Radio	Newspaper	Sales
count	200.000000	200.000000	200.000000	200.000000
mean	147.042500	23.264000	30.554000	14.022500
std	85.854236	14.846809	21.778621	5.217457
min	0.700000	0.000000	0.300000	1.600000
25%	74.375000	9.975000	12.750000	10.375000
50%	149.750000	22.900000	25.750000	12.900000
75%	218.825000	36.525000	45.100000	17.400000
max	296.400000	49.600000	114.000000	27.000000

Visualise the data

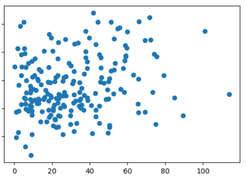
In [13]:
```
## Find the relation between variable we use scatter plot
# relation between TV and sales
plt.figure(figsize=(6,4))
plt.scatter(x=df.TV, y=df.Sales)
plt.show()
```



In [14]:
```
## Find the relation between variable we use scatter plot
# relation between Radio and sales
plt.figure(figsize=(6,4))
plt.scatter(x=df.Radio, y=df.Sales)
plt.show()
```



In [15]:
```
## Find the relation between variable we use scatter plot
# relation between Newspaper and sales
plt.figure(figsize=(6,4))
plt.scatter(x=df.Newspaper, y=df.Sales)
plt.show()
```

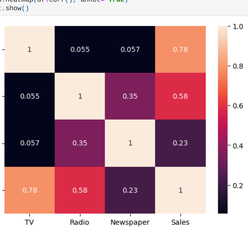


In [16]:
```
# Correlation
# value lies between -1 and 1
# 1 perfect postive corelation , as y increase x also increases
# -1 perfect negative correlation , as y increase  x will decrease
# any value between greater then 0.5 (positive or negative) indicates high correlation
# corr is the function
df.corr()
```

Out[16]:

	TV	Radio	Newspaper	Sales
TV	1.000000	0.054809	0.056648	0.782224
Radio	0.054809	1.000000	0.354104	0.576223
Newspaper	0.056648	0.354104	1.000000	0.228299
Sales	0.782224	0.576223	0.228299	1.000000

In [18]:
```
#visualise the correlation using heat map
import seaborn as sns # seaborn is data visualisation library
sns.heatmap(df.corr(), annot= True)
plt.show()
```



Perform Simple linear regression

- y = mx +c (equation for a=straight line)
- y will take sales , x will take as TV
- sales = m(TV) + c
- Find best fit line by minimising the error.

Split X and y

In [44]:
```
# Define X and y
X= df[['TV']]
y=df['Sales']
```

In [45]:
```
X.shape
```

Out[45]:
```
(200, 1)
```

In [46]:
```
X.head()
```

Out[46]:

	TV
1	230.1
2	44.5
3	17.2
4	151.5
5	180.8

In [47]:
```
y.head()
```

Out[47]:
```
1    22.1
2    10.4
3     9.3
4    18.5
5    12.9
Name: Sales, dtype: float64
```

Split Train and test

- Train data will be used for training the model
- Test will be used for testing

In [48]:
```
# import the library for train-test split
# test = 20% , train = 80%
from sklearn.model_selection import train_test_split
X_train_lm,X_test_lm, y_train_lm, y_test_lm = train_test_split(X,y , test_size=0.2 , random_state=21)
```

In [49]:
```
# check the shape
X_train_lm.shape
```

Out[49]:
```
(160, 1)
```

Build the linear model

In [50]:
```
# import the library for linear model from sklearn
from sklearn.linear_model import LinearRegression
lm = LinearRegression()
```

In [52]:
```
# Train the model using fit method
lm.fit(X_train_lm,y_train_lm)
```

Out[52]:
```
LinearRegression()
```

In [ ]: