APPLIED DATA SCIENCE GROUP-2

FUTURE SALES PREDICTION (PHASE -3)

DATASET:

Dataset: www.kaggle.com

Dataset name: future-sales-prediction

Dataset link: https://www.kaggle.com/datasets/chakradharmattapalli/future-sales-prediction

DETAILS ABOUT THE DATASET:

It consist of 4 columns and every column consist of 200 data's

TV

Radio

Newspaper

Sales

TV : **Advertising cost spent in dollars for advertising on TV**

Radio : **Advertising cost spent in dollars for advertising on Radio**

Newspaper : **Advertising cost spent in dollars for advertising on Newspaper**

Sales: **Number of units sold**

LIBRARIES:

```
import numpy as np
import pandas as pd
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
import statsmodels.api as sm
import plotly.express as px
from sklearn.preprocessing import StandardScaler
```

Importing Dataset:

```
data = pd.read_csv("F:\AIML DATASET\Sales.csv")
```

Print first 5 data's(head):

```
print(data.head())
```

```
      TV  Radio  Newspaper  Sales
0  230.1   37.8       69.2   22.1
1   44.5   39.3       45.1   10.4
2   17.2   45.9       69.3   12.0
3  151.5   41.3       58.5   16.5
4  180.8   10.8       58.4   17.9
```

Print last 5 data's(tail):

```
print(data.tail())
```

```
        TV  Radio  Newspaper  Sales
195   38.2    3.7       13.8    7.6
196   94.2    4.9        8.1   14.0
197  177.0    9.3        6.4   14.8
198  283.6   42.0       66.2   25.5
199  232.1    8.6        8.7   18.4
```

Checking for missing values:

```
print(data.isnull().sum())
```

```
TV           0
Radio        0
Newspaper    0
Sales        0
dtype: int64
```

Splitting the data into train and test:

```
xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size=0.2, random_state=42)
```

TRAIN- xtrain , ytrain

TEST- xtest , ytest

Once we have pre-processed the data into a format that's ready to be used by the model, we need to split up the data into train and test sets. This is because the machine learning algorithm will use the data in the training set to learn what it needs to know.

It will then make a prediction about the data in the test set, using what it has learned. We can then compare this prediction against the actual target variables in the test set in order to see how accurate the model is.

We will do the train/test split in proportions. The larger portion of the data split will be the train set and the smaller portion will be the test set.

This will help to ensure that we are using enough data to accurately train the model.

In general, we carry out the train-test split with an 80:20 ratio(also 70:30 ratio)

Feature Scaling:

Feature scaling is a method in which we scale the data into an accurate and scalable size for the purpose of increasing accuracy and reducing error. It basically prevents the large variance of data points to be used in the algorithm and allows us to achieve better results.

```
from sklearn.preprocessing import StandardScaler
```

Standard Scaler:

Python sklearn library offers us with StandardScaler() function to standardize the data values into a standard format. According to the syntax, we initially create an object of the StandardScaler() function. Further, we use fit_transform() along with the assigned object to transform the data and standardize it.