Exp no: 3 **Develop a linear regression model for forecasting time series data**

Date: 18/2/25

Objectives:

The objective of developing a linear regression model for forecasting time series data is to predict future values based on historical patterns. In this case, the Netflix time series dataset is used to forecast metrics such as subscriber growth, revenue, or content consumption. Linear regression is a simple yet powerful statistical method that models the relationship between a dependent variable (target) and one or more independent variables (features). By analyzing trends and seasonality in the dataset, the model aims to provide actionable insights for decision-making, such as optimizing content strategies or improving user engagement.

Background Scope:

Time series forecasting is a critical tool in industries like entertainment, finance, and retail, where understanding future trends is essential for strategic planning. Netflix, as a leading streaming platform, generates vast amounts of time series data, including daily active users, watch time, and subscription rates. Linear regression can be applied to this dataset to identify relationships between variables, such as the impact of new content releases on subscriber growth or the effect of marketing campaigns on user engagement. While linear regression is a straightforward approach, it serves as a foundational model for more complex techniques like ARIMA or machine learning models. This project focuses on leveraging historical Netflix data to build a predictive model that can inform business strategies and enhance operational efficiency.

Step 1: Load the Dataset and libraries

1. **Load** the dataset from a CSV file into a Pandas DataFrame.

Step 1: import the necessary libraries

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

```
import plotly.express as px
```

Step 2: load the dataset and follow preprocessing steps

df=pd.read_csv("/content/NFLX.csv")

```
df['Close'].plot(figsize=(10, 7))
plt.title("Stock Price", fontsize=17)
plt.ylabel('Price', fontsize=14)
plt.xlabel('Time', fontsize=14)
plt.grid(which="major", color='k', linestyle='-.', linewidth=0.5)
plt.show()


df = df.drop(columns=['Date'])
```

target = "Close"

y = df[target]

X = df.drop(columns= target)

#Split the data into train and test sets

cutoff = int(len(X) * 0.8)

Step 3 : split the data into train and test sets

X_train, y_train = X.iloc[:cutoff], y.iloc[:cutoff]

X_test, y_test = X.iloc[cutoff:], y.iloc[cutoff:]

from sklearn.metrics import mean_absolute_error

y_pred_baseline = [y_train.mean()] * len(y_train)

mae_baseline = mean_absolute_error(y_train, y_pred_baseline)

print("Mean Close Prices:", round(y_train.mean(), 2))

print("Baseline MAE:", round(mae_baseline, 2))

Step 4: Train the model using linear regression

```
model = LinearRegression()

model.fit(X_train, y_train)

LinearRegression()
```

Step 5: Calculate error rate and accuracy rate

```
training_mae = mean_absolute_error(y_train, model.predict(X_train))

test_mae = mean_absolute_error(y_test, model.predict(X_test))

print("Training MAE:", round(training_mae, 2))

print("Test MAE:", round(test_mae, 2))


df_pred_test = pd.DataFrame(

        {

        "y_test": y_test,

        "y_pred": model.predict(X_test)

        }

)

df_pred_test.head()
```

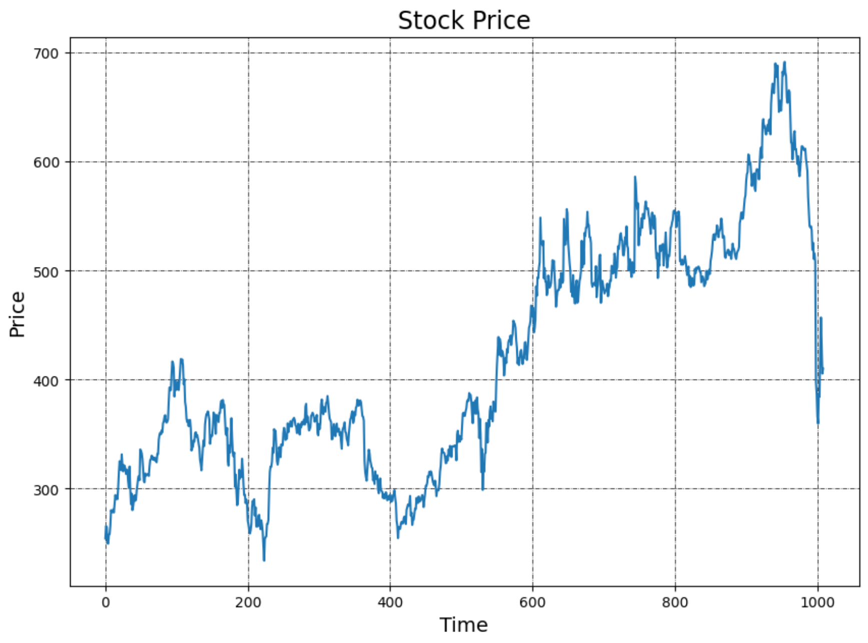
Step 6: Plot the graph

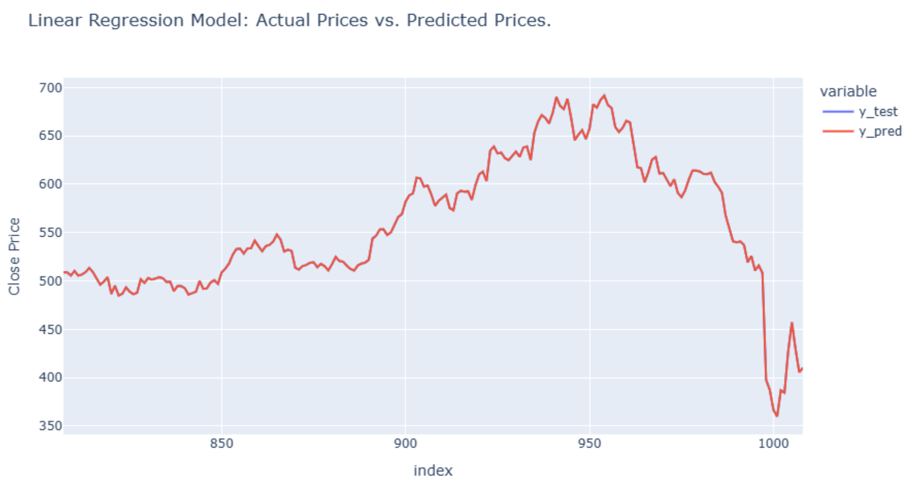
```
import plotly.express as px

fig = px.line(df_pred_test, labels= {"value": "Close Price"}, title = "Linear Regression Model: Actual Prices vs. Predicted Prices.")

fig.show()
```

Ouput:





Result: The program has been executed successfully using linear regression .