# EX: 5 DEVELOP A LINEAR REGRESSION MODEL

# DATE: FOR FORECASTING TIME SERIES DATA

**AIM:**

To develop a Linear Regression model for forecasting time series data and evaluate its performance using error metrics and residual analysis.

# ALGORITHM:

# Load Data – Import the dataset and convert the date column to datetime format.

# Preprocess Data – Convert dates into numerical values for regression.

# Split Data – Divide the dataset into training and testing sets.

# Train Model – Fit a Linear Regression model using the training data.

# Make Predictions – Predict values on the test set using the trained model.

# Evaluate Model – Calculate MAE, MSE, and RMSE for performance assessment.

# Visualize Results – Plot a residual plot to analyze prediction errors.

# PROGRAM:

**import pandas as pd**

**import numpy as np**

**import matplotlib.pyplot as plt**

**from sklearn.linear\_model import LinearRegression**

**from sklearn.model\_selection import train\_test\_split**

**from sklearn.metrics import mean\_squared\_error**

**# Load your Air Passenger dataset (update the file path accordingly)**

**df = pd.read\_csv('/content/airline-passengers.csv') # Replace with your dataset file path**

**# Convert 'Month' column to datetime type if it exists**

**df['Month'] = pd.to\_datetime(df['Month'], format='%Y-%m') # Adjust if the date format is different**

**# Visualize the original time series**

**plt.figure(figsize=(10, 6))**

**plt.plot(df['Month'], df['Passengers'])**

**plt.title('Air Passengers Time Series Data')**

**plt.xlabel('Date')**

**plt.ylabel('Number of Passengers')**

**plt.grid(True)**

**plt.show()**

**# Feature preparation for Linear Regression**

**df['Time'] = np.arange(len(df)) # Create a 'Time' column with sequential time steps**

**# Split the data into training and test sets**

**X = df[['Time']] # Feature: Time (index)**

**y = df['Passengers'] # Target: Passengers (number of passengers)**

**X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, shuffle=False)**

**# Train a Linear Regression model**

**model = LinearRegression()**

**model.fit(X\_train, y\_train)**

**# Make predictions**

**y\_pred = model.predict(X\_test)**

**# Visualize the predictions vs the actual values**

**plt.figure(figsize=(10, 6))**

**plt.plot(df['Month'][:len(X\_train)], y\_train, label='Training Data', color='blue')**

**plt.plot(df['Month'][len(X\_train):], y\_test, label='Actual Test Data', color='green')**

**plt.plot(df['Month'][len(X\_train):], y\_pred, label='Predictions', color='red', linestyle='--')**

**plt.title('Linear Regression Forecast for Air Passengers')**

**plt.xlabel('Date')**

**plt.ylabel('Number of Passengers')**

**plt.legend()**

**plt.grid(True)**

**plt.show()**

**# Evaluate the model**

**mse = mean\_squared\_error(y\_test, y\_pred)**

**rmse = np.sqrt(mse)**

**print(f'Mean Squared Error: {mse}')**

**print(f'Root Mean Squared Error: {rmse}')**

**# Residual Plot**

**residuals = y\_test - y\_pred # Calculate residuals (actual - predicted)**

**plt.figure(figsize=(10, 6))**

**plt.scatter(df['Month'][len(X\_train):], residuals, color='blue')**

**plt.axhline(y=0, color='red', linestyle='--')**

**plt.title('Residual Plot for Air Passengers Forecasting')**

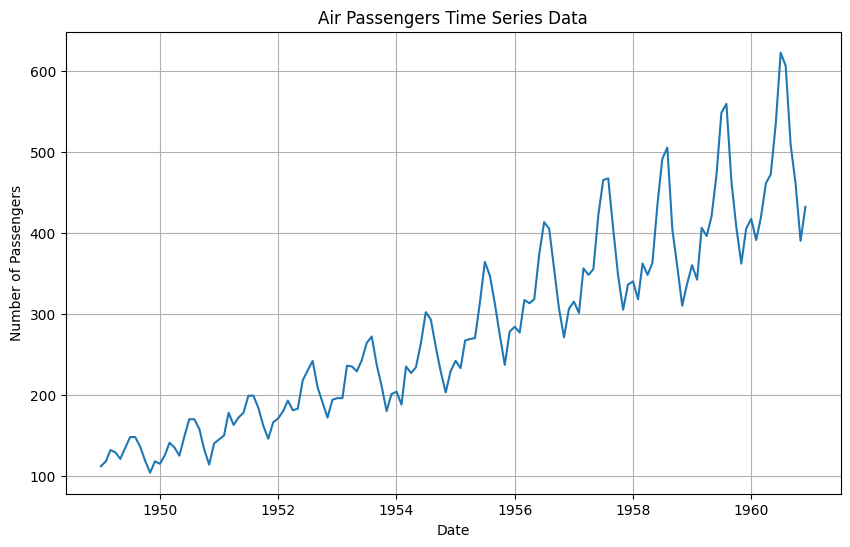
**plt.xlabel('Date')**

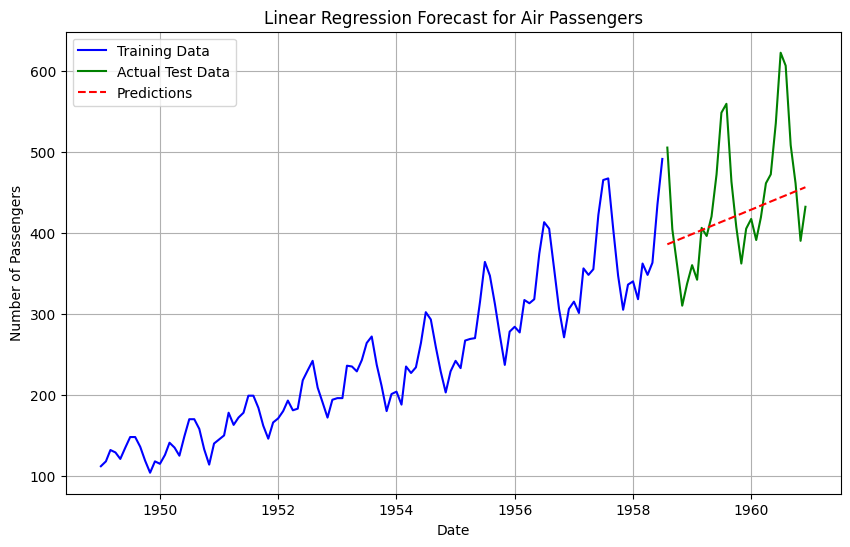
**plt.ylabel('Residuals (Actual - Predicted)')**

**plt.grid(True)**

**plt.show()**

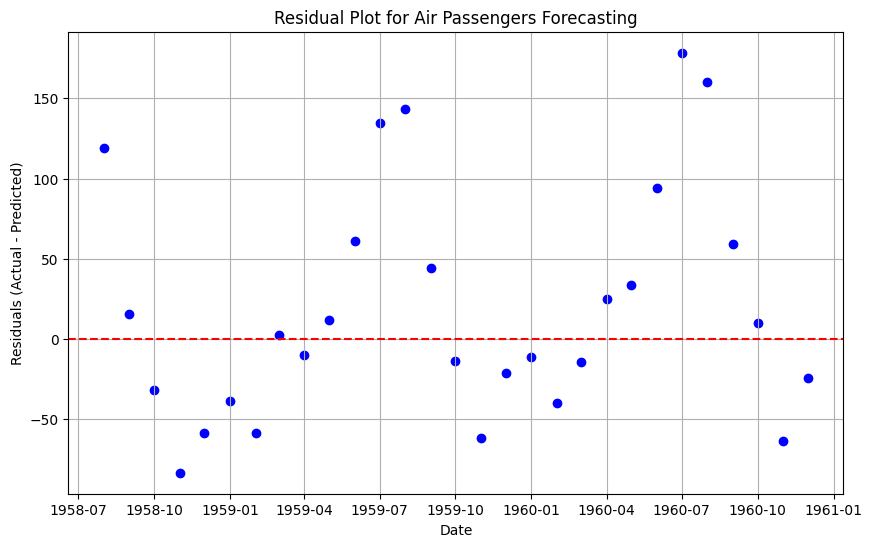
**OUTPUT:**

****

****

**Mean Squared Error: 5447.163612684265**

**Root Mean Squared Error: 73.8049023621349**

****

**RESULT:**

Thus, the Linear Regression model for forecasting Time Series data using Air passenger dataset was successfully developed and evaluated.