**Predict the Air Passengers**

Predict the Air Passengers in airline from the past Dataset (.csv file) and predict what will be the output/prediction in the upcoming following months

This code is designed to create a machine learning model that predicts future values in the "AirPassengers" dataset using a neural network implemented with Keras. Below is a detailed explanation of the code:

**Step-by-Step Explanation**

1. **Environment and Libraries Setup:**

python

Copy code

import os

os.environ['TF\_ENABLE\_ONEDNN\_OPTS'] = '0'

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.preprocessing import MinMaxScaler

from keras.\_tf\_keras.keras.models import Sequential

from keras.\_tf\_keras.keras.layers import Dense

from keras import metrics

* + The os module is used to set an environment variable.
  + Libraries like numpy, pandas, and matplotlib.pyplot are imported for numerical operations, data manipulation, and plotting, respectively.
  + MinMaxScaler from sklearn.preprocessing is imported for normalizing data.
  + Keras modules are imported for building the neural network.

1. **Loading the Dataset:**

python

Copy code

df = pd.read\_csv("AirPassengers.csv")

* + The AirPassengers.csv file is read into a DataFrame df.

1. **Preparing the Data:**

python

Copy code

L = len(df)

X = np.array([range(1, L)])

Y = np.array([df.iloc[:, 1]])

Y = Y[:, 0:L-1]

* + L is the length of the DataFrame.
  + X is an array containing the indices (time steps) of the dataset.
  + Y is an array containing the passenger numbers, adjusted to have one less element than X.

1. **Plotting the Data:**

python

Copy code

plt.figure(1)

plt.plot(X[0, :], Y[0, :])

plt.show(block=False)

* + The passenger numbers (Y) are plotted against the time steps (X).

1. **Creating Input Features:**

python

Copy code

X1 = Y[:, 0:L-4]

X2 = Y[:, 1:L-3]

X3 = Y[:, 2:L-2]

X = np.concatenate([X1, X2, X3], axis=0)

X = np.transpose(X)

Y = np.transpose(Y[:, 3:L-1])

* + The dataset is prepared with a sliding window approach to create lag features.
  + X1, X2, and X3 are shifted versions of Y.
  + X is a concatenated array of these shifted versions and transposed to the correct shape.
  + Y is adjusted accordingly.

1. **Data Normalization:**

python

Copy code

scaler = MinMaxScaler()

scaler.fit(X)

X = scaler.transform(X)

scaler1 = MinMaxScaler()

scaler1.fit(Y)

Y = scaler1.transform(Y)

* + The input features X and target values Y are normalized using MinMaxScaler to scale the values between 0 and 1.

1. **Building the Neural Network:**

python

Copy code

model = Sequential()

model.add(Dense(32, activation='relu', input\_dim=3))

model.add(Dense(32, activation='relu'))

model.add(Dense(1, activation='sigmoid'))

* + A Sequential neural network model is created.
  + The model has three layers: two hidden layers with 32 neurons each and ReLU activation, and an output layer with one neuron and sigmoid activation.

1. **Compiling the Model:**

python

Copy code

model.compile(loss="mean\_squared\_error", optimizer='rmsprop', metrics=[metrics.mean\_squared\_error])

* + The model is compiled with mean squared error as the loss function and RMSprop as the optimizer.

1. **Training the Model:**

python

Copy code

model.fit(X, Y, epochs=500, batch\_size=32, verbose=2)

* + The model is trained on the dataset for 500 epochs with a batch size of 32.

1. **Making Predictions:**

python

Copy code

predict = model.predict(X, verbose=1)

print(Y, predict)

* + The trained model makes predictions on the input data X.

1. **Plotting the Results:**

python

Copy code

plt.figure(2)

plt.scatter(Y, predict)

plt.show(block=False)

plt.figure(3)

Test = plt.scatter(X[:, 0], Y)

Predict = plt.scatter(X[:, 0], predict)

plt.legend([Predict, Test], ["Predict Data", "Real Data"])

plt.show()

* + The actual vs predicted values are plotted in a scatter plot.
  + Another plot compares the predicted data with the real data over time.

**Purpose of the Code**

The purpose of this code is to build and train a neural network to predict future values of the "AirPassengers" dataset. It uses past passenger numbers to predict future values, demonstrating a time series forecasting approach with a neural network. The normalization of data, creation of lag features, and training of a dense neural network are key steps in this process. The final plots allow for a visual comparison between the model's predictions and the actual values, providing insight into the model's performance.

**Explanation of all The Figures**

The figures generated by the code and explain what each of them represents:

**Figure 1: Initial Data Plot**

python

Copy code

plt.figure(1)

plt.plot(X[0, :], Y[0, :])

plt.show(block=False)

This figure plots the initial passenger data against the time steps.

* **X-axis**: Represents the time steps (months).
* **Y-axis**: Represents the number of passengers.
* **Purpose**: To visualize the original dataset and observe the trend and seasonality in the passenger numbers over time.

**Figure 2: Actual vs. Predicted Values Scatter Plot**

python

Copy code

plt.figure(2)

plt.scatter(Y, predict)

plt.show(block=False)

This figure plots the actual passenger values against the predicted passenger values in a scatter plot.

* **X-axis**: Represents the actual normalized passenger values.
* **Y-axis**: Represents the predicted normalized passenger values.
* **Purpose**: To show the correlation between the actual and predicted values. Ideally, if the predictions are accurate, the points should lie close to a line with a slope of 1 (45-degree line).

**Figure 3: Time Series Comparison Plot**

python

Copy code

plt.figure(3)

Test = plt.scatter(X[:, 0], Y)

Predict = plt.scatter(X[:, 0], predict)

plt.legend([Predict, Test], ["Predict Data", "Real Data"])

plt.show()

This figure plots the predicted data against the real data over time in a scatter plot.

* **X-axis**: Represents the time steps.
* **Y-axis**: Represents the normalized passenger values.
* **Test (Real Data)**: The actual passenger data points are plotted.
* **Predict (Predicted Data)**: The predicted passenger data points are plotted.
* **Purpose**: To visually compare how well the predicted data follows the trend of the real data over time. This plot helps to evaluate the model's performance in capturing the patterns and trends in the dataset.

**Summary of the Purpose of Figures**

1. **Figure 1**: Provides an initial view of the original dataset, highlighting trends and seasonality.
2. **Figure 2**: Demonstrates the accuracy of the model's predictions by comparing actual values to predicted values.
3. **Figure 3**: Offers a visual comparison of predicted and actual values over time, showing how well the model captures the temporal patterns in the data.

By analyzing these figures, we can assess the performance of the neural network model in predicting future values of the "AirPassengers" dataset.