ADVANCED MACHINE LEARNING

TIME SERIES DATA SUMMARY REPORT

I downloaded and extracted the contents of a dataset named "jena_climate_2009_2016.csv.zip" before beginning the code. The data has to be parsed in order to be prepared for analysis. Temperature is one of the climate features included in this dataset. Both the header and the actual data records are extracted.Next, I determined the data's standard deviation, which is essential for data normalization.

A time series plot is created to show how temperatures vary over time in order to better comprehend temperature fluctuations. It establishes the number of samples for training, validation, and testing datasets based on preset split percentages.

I normalized the data by subtracting the value and dividing by the variance in order to prepare and standardize it, And a dummy dataset has been created to showcase how to use the timeseries_dataset_from_array function

Datasets are created for testing, validation, and training. The following defines batch sizes and other pertinent characteristics.
num_train_samples: 210225
num_val_samples: 105112
num_test_samples: 105114

This code. Shows the information about sample shapes within the training dataset, for inspection.

Mean Absolute Error (MAE) values are computed for both the test and validation datasets using a reasonable baseline model. The following are the values that were obtained:

Following that, several machine learning models are investigated. It begins with a connected model that is used to evaluate its performance. Next, the efficacy of a 1D convolutional model is assessed.

Finally, a simple Long Short Term Memory (LSTM) model for time series data forecasting is shown. The following is the outcome of this model.
This code provides a NumPy implementation of a Recurrent Neural Network (RNN). Its goal is to comprehend the internal operations of RNNs.

The code demonstrates the various RNN layer types in Keras. Sequences of different lengths can be handled by these layers, which can either return the output step or even the entire output sequence. The outcome is as follows:

Additionally, the code explores RNN models. It investigates methods such as dropout to get around overfitting and layer stacking.

The method then uses dropout regularization to train and assess a stacked Gated Recurrent Unit (GRU) model.

Below are the values obtained in the analysis of models:

Type of models	**Test MAE**
Basic Navie method	2.62
Densly connected network model	2.68
1D convolutional Model	3.03
Simple LSTM Model	2.57
RNN Model	9.95
Droupout LSTM Model	2.85
Simple LSTM Model with 32 units	2.59
Stacked LSTM with 64	2.54

In conclusion, this code explores the analysis and forecasting of time series data using deep learning and machine learning models. It addresses topics like model creation, evaluation, normalization, and data preprocessing.