ADVANCED MACHINE LEARNING

TIME SERIES DATA SUMMARY REPORT

I started the code by downloading a dataset called "jena\_climate\_2009\_2016.csv.zip" and extracting its contents. The next step was to Parses the data getting it ready for analysis. This dataset include climate information with features, including temperature. it extracts both the header and the actual data records.Then I calculated the standard deviation of the data which is crucial for normalizing the data.

To better understand temperature variations a time series plot is generated to visualize how temperatures change over time.

Based on predetermined split percentages it determines the number of samples for training, validation and testing datasets.

To prepare and standardize the data I normalized it by subtracting the value and dividing by the deviation.

A dummy dataset is created to showcase how to use the timeseries\_dataset\_from\_array function

Datasets are established for training, validation and testing purposes. Batch sizes and other relevant parameters are defined as follows

num\_train\_samples: 210225

num\_val\_samples: 105112

num\_test\_samples: 105114

The code. Displays information about sample shapes within the training dataset, for inspection.

A sensible baseline model is implemented to calculate Mean Absolute Error (MAE) values for both validation and test datasets. The values obtained are as follows

After that various machine learning models are explored. It starts with a connected model where its performance is assessed.Then a 1D convolutional model is evaluated in terms of its effectiveness.

Lastly a straightforward Long Short Term Memory (LSTM) model is introduced to forecast time series data. The result obtained through this model is as follows

The provided code offers an implementation of a Recurrent Neural Network (RNN) using NumPy. Its purpose is to understand how RNNs work internally.

The code showcases types of RNN layers in Keras. These layers can handle sequences with varying lengths return the output step or even provide the complete output sequence. The result is as follows

Furthermore the code delves into RNN models. It explores techniques, like dropout to overcome overfitting and the stacking of layers.

Next the code proceeds to train and evaluate a stacked Gated Recurrent Unit (GRU) model with dropout regularization applied.

The following table shows the values obtained in the analysis of models:

|  |  |
| --- | --- |
| MODELS | TEST MAE |
| NAVIE METHOD | 2.62 |
| DENSLY CONNECTED NETWORK MODEL | 2.63 |
| 1D CONVOLUTIONAL MODEL | 3.21 |
| SIMPLE LSTM MODEL | 3.17 |
| RNN MODEL | 9.93 |
| STACKING RNN MODEL | 9.91 |
| DROUPOUT LSTM MODEL | 2.59 |
| SIMPLE LSTM MODEL WITH 32 UNITS | 2.63 |
| STACKED LSTM WITH 64 UNITS | 2.57 |
| STACKED LSTM WITH 8 UNITS | 2.78 |
| 1D CONVOLUTIONAL WITH RNN | 3.7 |
|  |  |
|  |  |

In summary this code provides an exploration of time series data analysis and forecasting through machine learning and deep learning models. It covers aspects such as data preprocessing, normalization, model development and evaluation.