Multivariate Series prediction

Data Exploration

The Individual household electric power consumption Data Set is a time-series dataset that contains measurements of electric power consumption in one household with a one-minute sampling rate. The dataset spans over a period of almost four years, from December 2006 to November 2010, and includes 2,075,259 observations.

The dataset was collected by a single-phase energy meter, measuring the household's electric power consumption with a resolution of one minute. The meter measured the voltage and current in the house's main circuit, and from this, the active power, reactive power, and apparent power were calculated. In addition to the power measurements, the dataset also includes measurements of global active power, voltage, and current intensity.

The dataset includes 9 attributes, which are:

- Date: the date on which the measurement was taken (format dd/mm/yyyy)
- Time: the time at which the measurement was taken (format hh:mm:ss)
- Global_active_power: the household's total active power consumption in kilowatts (kW)
- Global_reactive_power: the household's total reactive power consumption in kilowatts (kW)
- Voltage: the average voltage (in volts) measured over one minute
- Global_intensity: the average current intensity (in amps) measured over one minute
- Sub_metering_1: the active power consumption (in kilowatts) in the kitchen area
- Sub_metering_2: the active power consumption (in kilowatts) in the laundry area
- Sub_metering_3: the active power consumption (in kilowatts) in the climate control system

Data processing

The preprocessing consists of the following steps:

- The 'read_csv' function is used to read the CSV file, and a number of arguments are supplied to it, including the CSV file's delimiter, the columns that should be parsed as dates, the way to handle missing values, and the data types of pertinent fields.
- The 'dropna()' function is used by the code to eliminate any rows with missing values after reading in the CSV file.
- The code then uses Pandas' to numeric function to change the data types of many columns to "float." To make sure the data is in the right format for analysis, this is done. Date and time features are merged and used as index

- A window size of 5 is chosen, such that each training example contains a batch of 5 datapoints that have 7 features which will be used to predict the next datapoint. Along with batch of 5, the training example will also contain the true label which the model tries to predict

Implementation of Multivariate series prediction

- Data Preparation: Gather the dataset and preprocess itas mentioned above. Sets for training, and testing are made using the dataset class to make sure that all variables have the same scale, it is also crucial to normalize the data.

- Define the LSTM Model: Build a model architecture that processes the data using LSTM layers. The number of time steps and features in the input data should be considered by the model. In order to improve the performance of the model, you can experiment with various hyperparameters.

- Train the LSTM Model: Use the training set to train the LSTM model. You can employ a variety of optimisation techniques, such as Adam or SGD, and a suitable loss function, such as mean square error (MSE).

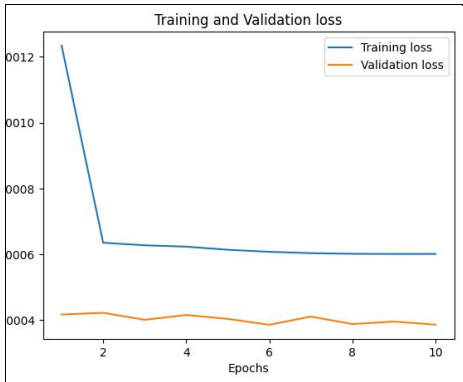
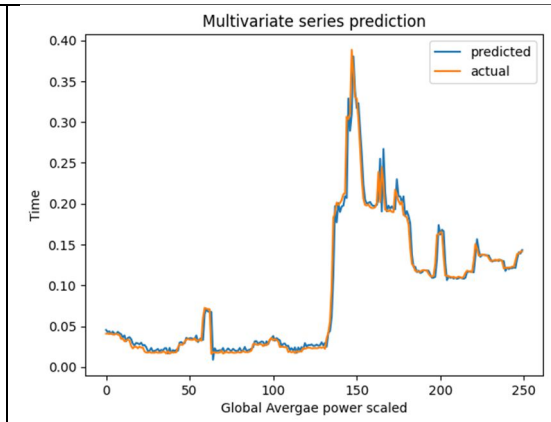
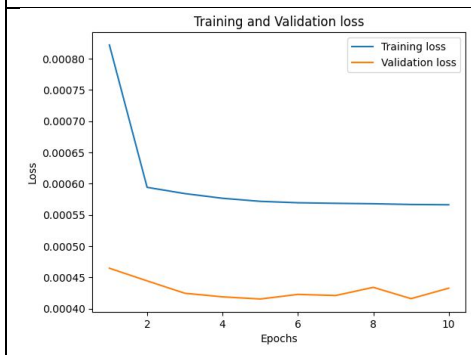
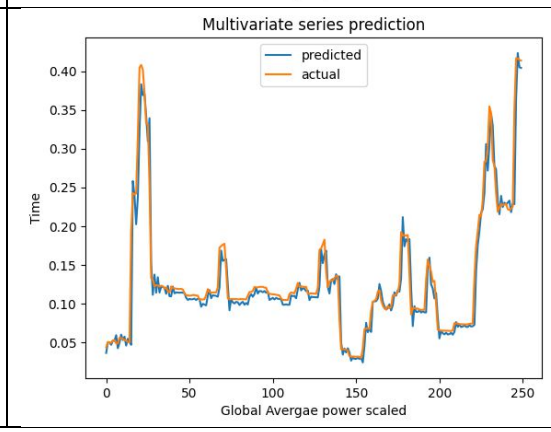
- Validate the LSTM Model: Use the validation set to evaluate the model's performance. To evaluate the model's correctness, you can keep an eye on various performance indicators including mean absolute error (MAE), root mean square error (RMSE), or coefficient of determination (R2).

- Test the LSTM Model: Use the testing set to evaluate the model's effectiveness. Use the same performance measures from the validation step to evaluate the model.

- Tune the LSTM Model: I altered the architecture or hyperparameters based on the model's performance to increase the model's accuracy.

Results and discussion

Loss Curves	Real global active power and predicted global active power for the testing days	MAE values	Dataset Split

		2.47404797872	80:20
		4.15742511107	70:30

Discussion

We can observe that we MAE value Is quite less for the model with 80:20 split as compared to the model with 70:30 split

Furthermore, the predicted and actual plot for 70:30 is a lot more shifted above as compared to 80:20