RECURRENT NEURAL NETWORK

Problem Description :

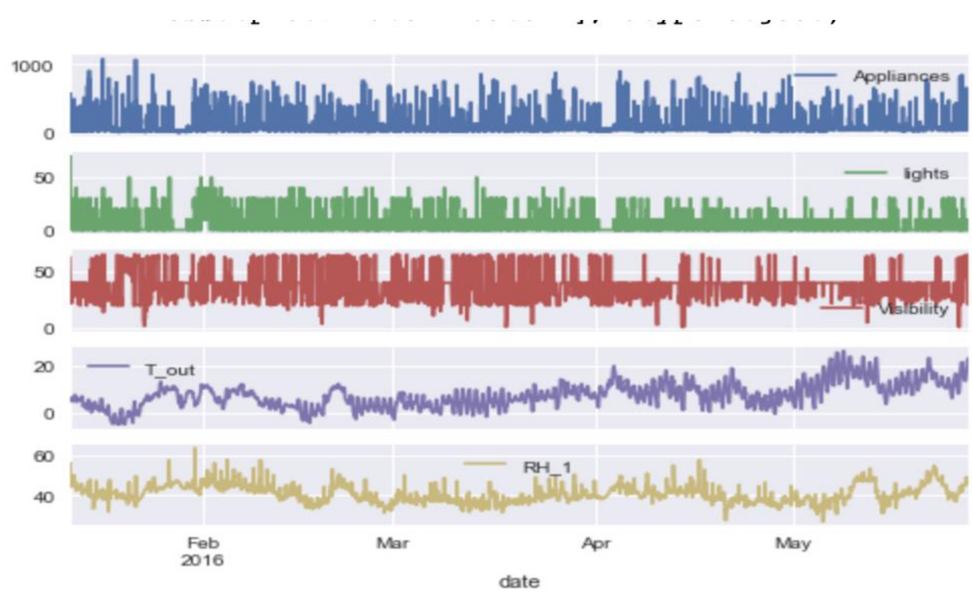
To demonstrate your ability to build effective temporal (time-lagged neural networks) and recurrent neural networks for time-series modeling in the presence of covariates

Dataset Used. :

The data set is at 10 min for about 4.5 months. The house temperature and humidity conditions were monitored with a ZigBee wireless sensor network. Each wireless node transmitted the temperature and humidity conditions around 3.3 min. Then, the wireless data was averaged for 10 minutes periods. The energy data was logged every 10 minutes with m-bus energy meters. Weather from the nearest airport weather station (Chievres Airport, Belgium) was downloaded from a public data set from Reliable Prognosis (rp5.ru), and merged together with the experimental data sets using the date and time column. Two random variables have been included in the data set for testing the regression models and to filter out non predictive attributes (parameters).

Glimpse of data is shown below in a subplot showing Appliances , lights, Visibility , Temperature(outside), Humidity over a period of 4.5 months.

Figure 1 : Dataset



Model Process :

To carry out the modeling process a RNN of type time lagged neural network is used. Recurrent Neural Networks like Long Short-Term Memory (LSTM) neural networks are able to almost seamlessly model problems with multiple input variables. This is a great benefit in time series forecasting, where classical linear methods can be difficult to adapt to multivariate or multiple input forecasting problems. It involves framing the dataset as a supervised learning problem and normalizing the input variables. Data is framed as the supervised learning problem predicting the

Appliances using Energy (in MW) current hour (t) given the weather conditions (temp, humidity) at the prior time step.

Normalization of data:

Before the data is fed to the model , as a part of pre – processing of data all features are normalized, then the dataset is transformed into a supervised learning problem. It is done so, as to remove the outliers in the dataset. The estimator standardizes features by removing the mean and scaling to unit variance such that the values are centered around 0.

Training and Testing data :

The total dataset contains 19750 entries collected over 4.5 months after every 10 mins averaging the recorded value for each attribute after every 3.3 secs. The training data constitutes 17500 entries of the total values and testing data has the remaining 11.8 % of the data.

The inputs of training and testing are reshaped into the 3D format expected by LSTMs. Further, the data is generated in 3d format using TimeseriesGenerator with parameter windows length =720, Batch size=32, and features = 28.

Model Training :

The data is fed to a LSTM model with no of epochs (as 3) , the dataset was large to reduce the time for training the model less no of iterations are used with learning rate of 0.0001. Mean absolute error , loss i.e. mean square error are being recorded after each epoch. The optimizer used is Adam as it perform good with sparse data and it combines the best properties of the AdaGrad and RMSProp algorithms to provide an optimization algorithm that can handle sparse gradients .

The result after training model are shown below:-

Figure 2 : Model Results

```
Epoch 1/3
/Users/akaash/opt/miniconda3/envs/tensorflow/lib/python3.8/site-packages/tensorflow/python/keras/engine/training.py:1
844: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future version. Please use `Model.fit
`, which supports generators.
  warnings.warn('`Model.fit_generator` is deprecated and '
523/523 - 681s - loss: 0.0096 - mae: 0.0569 - val_loss: 0.0082 - val_mae: 0.0390
Epoch 2/3
523/523 - 692s - loss: 0.0085 - mae: 0.0530 - val_loss: 0.0076 - val_mae: 0.0378
Epoch 3/3
523/523 - 696s - loss: 0.0074 - mae: 0.0476 - val_loss: 0.0061 - val_mae: 0.0336
```

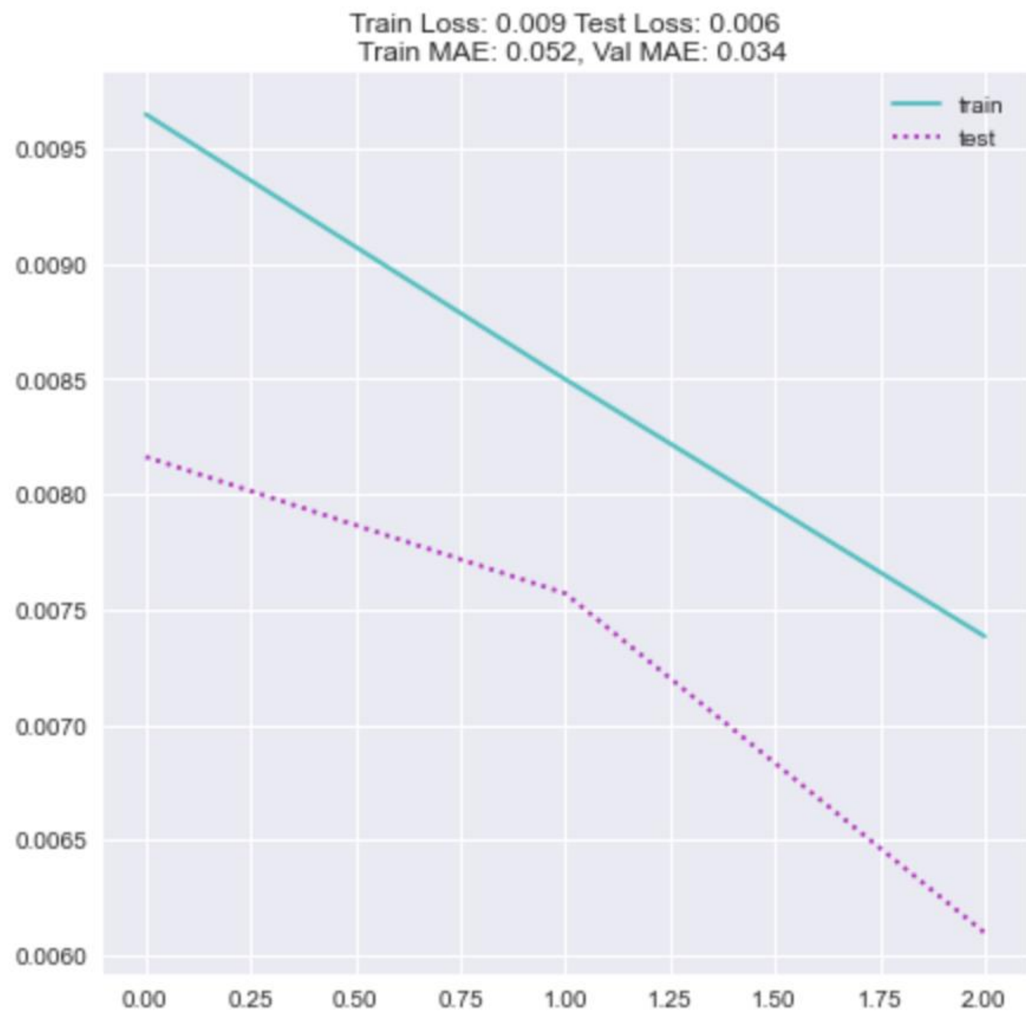
Early stopping :

To prevent model from getting over trained Early Stopping method is being implemented which monitored the loss value with error difference i.e. minimum delta value of 0.00001 and by restoring the best weight found in the whole process. It update the learner so as to make it better fit the training data with each iteration.

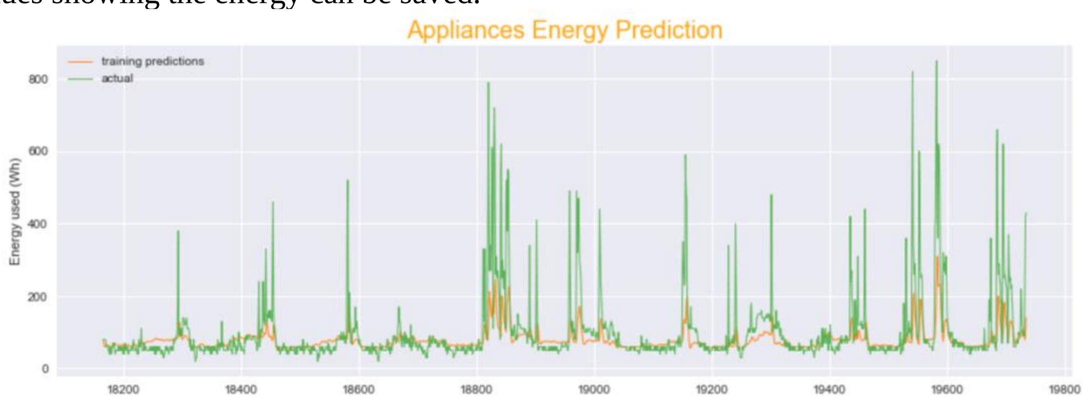
Results. :-

Below graph shows the result of training data plotting MSE and MAE values.

Figure 3 : Train loss and MAE



Next, the actual values and the result predicted by the model are shown below for appliance energy usage. As it is seen very clearly the spikes in predicted values are lower than the actual values showing the energy can be saved.



AutoCorrelation : the ACF and PACF is computed for Appliance Energy consumed attribute which is also our target attribute.

The horizontal axis of an autocorrelation plot shows the size of the lag between the elements of the time series. For example, the autocorrelation with lag 2 is the correlation between the time series elements and the corresponding elements that were observed two time periods earlier. Below graph shows the correlation after each Time-period. The correlation plot shows a decreasing trend signifying. the consumption of energy by appliance decreases over time.