Topic: Recurrent Neural Network (RNN)

Instructions:

1. Business Problem

1.1. Objective

1.2. Constraints (if any)

Using Python perform:

2. Data Pre-processing (if applicable)

2.1 Data cleaning, Feature Engineering etc.

3. Exploratory Data Analysis (EDA): (if applicable)

3.1. Summary

3.2. Univariate analysis

3.3. Bivariate analysis

4. Model Building

4.3 Using Python libraries perform the below tasks

5. Result Share the benefits/impact of the solution - how or in what way the business (client) gets benefit from the solution provided. (If applicable)

Note:

The assignment should be submitted in the following format:

- Python code
- Code Modularization should be maintained
- Documentation of the modules (elaborating on steps mentioned above).

1. **Here is the time series data [110, 125, 133, 146, 158, 172, 187, 196, 210].**

   Build RNN/LSTM model to predict the next 10 digits.

Solution:

1) As this is a univariate time series, LSTMs can be used to model univariate time series forecasting problems.

2) These are problems comprised of a single series of observations and the model is required to learn from the series of past observations to predict the next value in the sequence.

Data Preparation:

1) The LSTM model will learn a function that maps a sequence of past observations as input to an output observation. As such, the sequence of observations must be transformed into multiple examples from which the LSTM can learn.
2) We can divide the sequence into multiple input/output patterns called samples, Here, we took 3 time steps and is used as input and 1 time step is used as output for the one-step prediction that is being learned.

X			Y
110	125	133	146
125	133	146	158
133	146	158	172
146	158	172	187
158	172	187	196
172	187	196	210

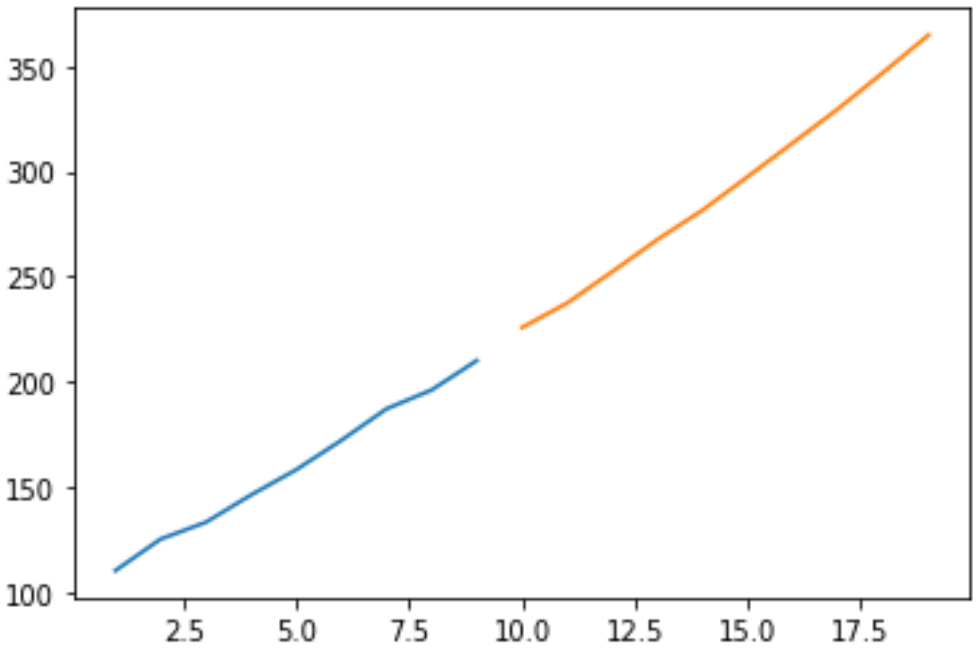
3) The *split sequence ()* function below implements this behavior and will split a given univariate sequence into multiple samples where each sample has a specified number of time steps and the output is a single time step.
4) The shape of the independent variables is (6,3) which means 6 is the input we have and using 3 timesteps we are predicting our dependent output 'Y'.

Model Building:

1) An LSTM layer requires a three-dimensional input and LSTMs by default will produce a two-dimensional output as an interpretation from the end of the sequence.
2) We can address this by having the LSTM output a value for each time step in the input data by setting the *return_sequences=True* argument on the layer. This allows us to have 3D output from hidden LSTM layer as input to the next.
3) The model is now define using the "relu" activation function with dense "1" and no of epochs given are 300.
4) Now the loop is created to demonstrate the prediction for next 10 days.

[225.69, 237.35, 252.32, 267.62, 281.64, 297.60, 313.70, 329.86, 347.22, 364.96]

Graphical Representation of output:



2. Write down the multiple applications of RNN.

Applications of Recurrent Neural Networks include:

- ✓ Machine Translation
- ✓ Robot control
- ✓ Time series prediction
- ✓ Speech recognition
- ✓ Speech synthesis
- ✓ Time series anomaly detection
- ✓ Rhythm learning
- ✓ Music composition
- ✓ Grammar learning
- ✓ Handwriting recognition
- ✓ Human action recognition
- ✓ Protein Homology Detection
- ✓ Predicting subcellular localization of proteins
- ✓ Several prediction tasks in the area of business process management
- ✓ Prediction in medical care pathway

3. How to do select the inputs for a LSTM/RNN models. Explain in the terms of timesteps, samples and feature.

The LSTM input layer is specified by the "*input_shape*" argument on the first hidden layer of the network is used to select the inputs.

- ✓ **Samples:** One sequence is one sample. A batch is comprised of one or more samples.
- ✓ **Time Steps:** One-time step is one point of observation in the sample.
- ✓ **Features:** One feature is one observation at a time step

4. What are the disadvantages of MLP when dealing with sequence data.

- ✓ The use of feedforward neural networks on sequence data raises two major problems:
- ✓ Input & outputs can have different lengths in different examples
- ✓ MLPs do not share features learned across different positions of the data sample.
- ✓ MLP include too many parameters because it is fully connected. Parameter number = width x depth x height. Each node is connected to another in a very dense web — resulting in redundancy and inefficiency.

Conclusion:

RNNs are a very powerful tool to deal with sequence data, they provide incredible memorizing capacities and are widely used in day-to-day life.

They also have many extensions which enable to address various types of data-driven problems, more particularly the ones discussing times series.