Group B Deep Learning

Assignment No: 4

Title of the Assignment: Use the google stock price dataset and design a time series analysis and prediction system using RNN

Objective of the Assignment: Students should be able to google stock dataset using RNN

Prerequisite:

1. Basic of programming language
2. Concept of RNN

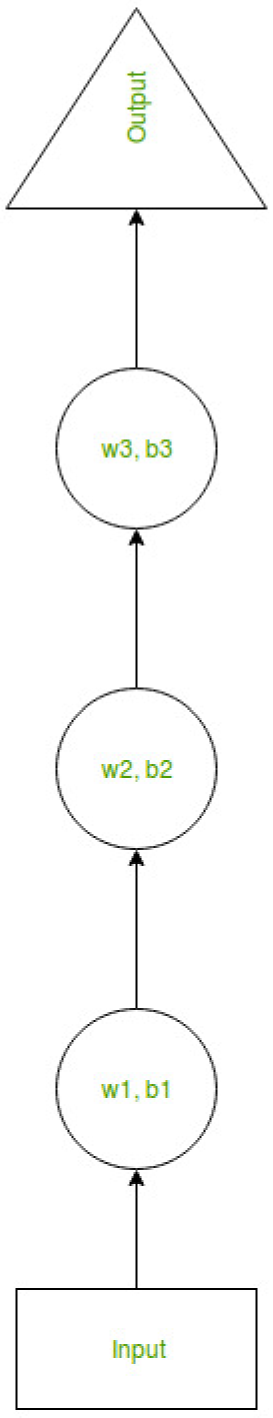
Contents for Theory:

1. What is RNN
2. How RNN works
3. Code with output

1)Recurrent Neural Network(RNN) is a type of Neural Network where the **output from the previous step are fed as input to the current step**. In traditional neural networks, all the inputs and outputs are independent of each other, but in cases like when it is required to predict the next word of a sentence, the previous words are required and hence there is a need to remember the previous words. Thus RNN came into existence, which solved this issue with the help of a Hidden Layer. The main and most important feature of RNN is **Hidden state**, which remembers some information about a sequence.

2)How RNN works

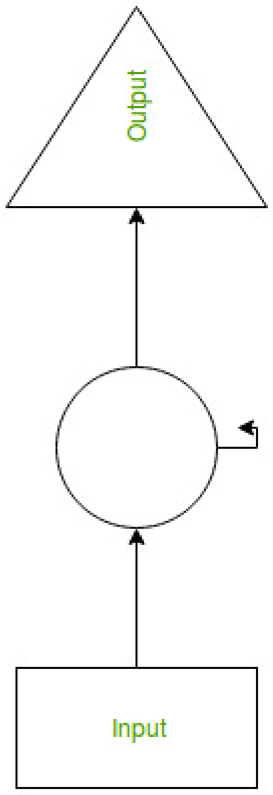
Example: Suppose there is a deeper network with one input layer, three hidden layers, and one output layer. Then like other neural networks, each hidden layer will have its own set of weights and biases, let's say, for hidden layer 1 the weights and biases are (w1 , b1), (w2, b2) for the second hidden layer, and (w3, b3) for the third hidden layer. This means that each of these layers is independent of the other, i.e. they do not memorize the previous outputs.



Now the RNN will do the following:

- RNN converts the independent activations into dependent activations by providing the same weights and biases to all the layers, thus reducing the complexity of increasing parameters and memorizing each previous output by giving each output as input to the next hidden layer.

- Hence these three layers can be joined together such that the weights and bias of all the hidden layers are the same, in a single recurrent layer.



The formula for calculating current state: $$h_t = f(h_{t-1}, x_t)$$ where:

h_t -> current state

h_{t-1} -> previous state

x_t -> input state

Formula for applying Activation function(tanh): $$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$ where:

w_{hh} -> weight at recurrent neuron

w_{xh} -> weight at input neuron

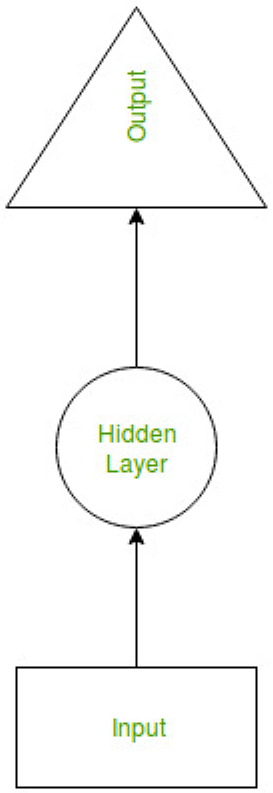
The formula for calculating output: $y_t = W_{hy}h_t$

Y_t -> output

W_{hy} -> weight at output layer

Training through RNN

1. A single-time step of the input is provided to the network.
2. Then calculate its current state using a set of current input and the previous state.
3. The current ht becomes ht-1 for the next time step.
4. One can go as many time steps according to the problem and join the information from all the previous states.
5. Once all the time steps are completed the final current state is used to calculate the output.
6. The output is then compared to the actual output i.e the target output and the error is generated.
7. The error is then back-propagated to the network to update the weights and hence the network (RNN) is trained.



RNN have a **"memory"** which remembers all information about what has been calculated. It uses the same parameters for each input as it performs the same task on all the inputs or hidden layers to produce the output. This reduces the complexity of parameters, unlike other neural networks.

Conclusion- In this way we can Predict Google stock price dataset and design a time series.