Prediction of Rare Volatility Surface-Patch using Artificial Neural Network

Introduction :

Currently used Methods for predicting rare volatility surface patches are mostly regression based. These methods do not take the fact into account that, infrequently available patches are less reliable. On the other hand, neural networks can better model the underlying relationship between input and output data sets than regression based models and are easily implementable in today's fast computers. In regard to this problem, we have used a neural network that considers the frequency of data by varying the learning rate between observations.

Algorithmic Procedure :

1. Generate 1000 random surfaces (500 training, 250 validation and 250 test observation).
2. Divide the surface into 25 patches.
3. Make an array of vectors with 22 inputs (20 inputs from the 20 patches of a surface generated in step 1 and 2 inputs from parameters used to generate that surface) and 5 outputs (take this outputs from remaining 5 patches of the same surface).
4. Generate missing data location randomly from the 5 output columns of the **training data set** in step 3, while keeping records of the missing positions.
5. Fill the missing values with EM algorithm.
6. Train the neural network with the above data set with a **modified learning rate** using **incremental (online)** approach.
7. Simulate the above neural network with **testing data set** and find results.

Platform :

Artificial Neural Network (ANN) toolbox of Matlab.

Modified Learning Rate :

We have used feed forward neural network provided in the ANN toolbox, for this problem. The feed forward neural network uses a training function to train the neural network named as ***trainc,*** which implements an incremental algorithm where the observations are passed to the network sequentially in each epoch.

As the importance of every observation in our problem is not equal, we modify ***trainc*** function in the appropriate positions so that learning rate can be changed according to the importance of the observations. The importance of data is calculated in two different ways:

1. Frequency Norm: We calculate the weights of each observation with the following formula

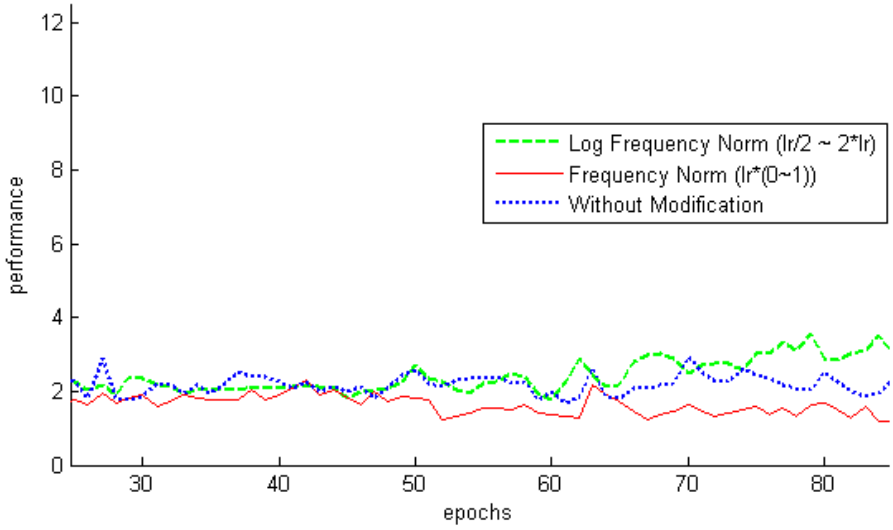
   Weight of an observation =

$$\frac{sum\ of\ frequencies\ of\ available\ output\ nodes\ in\ that\ observation}{sum\ of\ frequencies\ of\ all\ output\ node}$$

   Frequency of an output node is the number of observations, where a value for that output node is available (i.e., not missing)
   Then we normalize the weights between [0, 1] and use the following formula as learning rate =
   Learning rate of an observation = lr *normalized weight of that observation, where ***lr*** is some constant.

2. Log Frequency Norm: Here ***normalized weight of an observation*** (calculated in Frequency Norm Method) is mapped in ***[lr/2, 2*lr]*** using logarithmic scaling and use it as the learning rate of that observation.

Experimental Result :



3 different methods has been tested in our experiment – modified learning rate with frequency norm, modified learning rate with log frequency norm and ANN with constant learning rate. Each method is run for 10 times and 100 epochs. Average result of objective functions (performance) for each method is observed on the test set (Figure). Result shows that, the method with modified learning rate using frequency norm outperforms the other two methods consistently.