Math6373 Homework 3

An application of MLP AutoEncoders

1 Prediction Task :

Select 50 major companies $Comp_1$ ... $Comp_{50}$ on the US stockmarket

We consider $Comp_{50}$ as a "target" company

On each day "t" we want *to predict* the future stock price of our target on day (t+1) given the past evolution of our 50 stocks observed up to time "t"

2 Data Set

Denote TIMPER the time period 2014-2015-2016-2017

Let t=1, 2, ... N be the days on which the US stock exchange was open during TIMPER

you will have N roughly in the range 1005- 1010

For day "t", let Sj(t) be the stock price of company $Comp_j$ (at closing time)

On each day "t" , we want to predict the unknown target stock price $S_{50}(t+1)$ given the evolution of our 50 stock prices over the last 5 days up to day "t" , with day "t" included.

Download the 50 time series S1 ... S50 (which have the same length N)

2 PreProcessing of time series

Replace *isolated* missing values Sj(t) by the mean of two actual values closest to time t

If there are too many missing values in Sj , discard the stock $Comp_j$ and download another stock

Compute the moving mean of each series avSj(t)= [Sj(1) + ... + Sj(t)] /t

Normalize each series Sj(t) by Yj(t) = Sj(t) / avSj(t)

We now only use the time series Y1 ... Y50

3 Create Training and Test sets for an MLP predictor :

On each day t ≥ 10

the *recent past* of the series Yj will be the 1x5 line vector [Yj(t-4) Yj(t-3) Yj(t-2) Yj(t-1) Yj(t)]

the *recent past* of our 50 time series Y1 Y50 will be recorded by a 50x5 matrix Mt

each row "j" of Mt will be the line vector [Yj(t-4) Yj(t-3) Yj(t-2) Yj(t-1) Yj(t)]

The 50x5 = 250 coefficients of Mt can be "flattened" into a long 1x250 line vector Xt

for 5 ≤ t ≤ N-1 the vectors Xt will be the original *input vector*s

the MLP predictor to be trained later on (see question 5 below) will have the Xt as inputs , and a *single* output neuron with state Zt.

Zt will be the MLP prediction of the *target* TARGt = $S_{\mathbf{50}}$(t+1), which is not known at time t.

Now construct a data set of (N-5) "cases" for the prediction task :

$\text{Case}_{\mathbf{10}}$ $\text{Case}_{\mathbf{11}}$ $\text{Case}_{\mathbf{12}}$ ... $\text{Case}_{\mathbf{N-1}}$, indexed by t= 5 6 7 .... N-1

each $\text{Case}_{\mathbf{t}}$ is described by 250 features = 250 cordinates of vector Xt

for Case t , the TRUE output to be predicted at time t is the yet unknown TARGt = $S_{\mathbf{50}}$(t+1)

Define the data set of (N-5) cases for prediction learning by

PredCases = { all pairs (Xt, TARGt) with t= 5 6 7 .... N-1 }

Randomly Split the set *PredCases* with proportions (90%,10%)

90% for the training set *PredTRAIN*, and 10 % for the test set *PredTEST*

4 AutoEncoder to compress the input vectors Xt

Consider an MLP Auto Encoder with architecture

INPUT ==> HiddenLayer H ==> OUTPUT

where dim(INPUT) = dim(OUTPUT) = 250

dim(H) = h < 250

Ideally for each input Xt we want the autoencoder to compute an output Yt very close to Xt.

First step will be to compute a reasonable value for h , as follows

4.1 Compute a plausible dimension h for H

Implement PCA on the set of all input vectors Xt , with t= 10 ,11, ..., N

Plot the 250 eigenvalues $\lambda_1 > \lambda_2 > ... > \lambda_{250}$ associated to the 250 principal components

Plot the ratios $R(k) = (\lambda_1 + ... + \lambda_k) / (\lambda_1 + ... + \lambda_{250})$

Determine the number **h** of principal components which preserves 90% of the variance

Fix *dim(H)= h*.

4.2 AutoEncoder Training

Define the training and tet sets for the MLP auto encoder by

AutoTrain = all (Xt, Xt) where Xt is in *PredTRAIN*

AutoTest = all (Xt, Xt) where Xt is in *TestTRAIN*

Use the RELU response function and the Loss function "Mean Squared Error"

Use Stochastic Gradient Descent, Batch Learning, and Early Stopping based on comparing MSE on AutoTrain and AutoTest

Plot MSE(AutoTrain) and MSE(AutoTest) versus the number of batches

4.4 Compute Compressed Inputs

For each Xt in PredTRAIN or PredTEST compute the vector Ht (of dimension h) which gathers the states of all the neurons in the hidden layer H of the trained AutoEncoder

Ht will be called a *compressed input vector* (dimension h)

The Ht generated by Xt in PredTRAIN define a new training set *NewTrain*

NewTrain = all (Ht,TARGt) such that Xt is in PredTRAIN

NewTest = all (Ht,TARGt) such that Xt is in PredTEST

5 MLP predictor (deep learning method)

Deep Learning methodology suggests to construct an MLP predictor *(MLPpred)*

which will have the Ht as inputs , and a 3 layers architecture :

input H ==> hidden layer K ==> Output , with size(H) =h and size(Output) = 1

On day t, the *MLPpred* input is the compressed vector Ht computed by auto encoding of Xt,

and the MLP output is Zt , which should be close TARGt = $S_{50}(t+1)$

The training and test sets of MLPpred will be NewTrain and NewTest

5.1 Selection of size(K)

Let *numcases* = number of cases in NewTrain

Let k = size(hidden layer K). Compute *numwth* = {number of weights & thresholds} in *MLPpred*.

This should give a linear formula of the type *numwth* = u k +v with explicit values for u and v

Select for k the largest integer such that *numwth* < *numcases*

Intuitive justification for this choice of k ?

5.2 Training of MLPpred

Implement an automatic training of MLPpred, with the same options used above :

RELU response, Loss = "MSE", Stochastic Gradient Descent, Batch Learning,Early Stopping

Plot MSE(NewTrain) and MSE(NewTest) versus the number of batches

5.3 Evaluation of Results

Comment on the comparison MSE(NewTrain) versus MSE(NewTest)

Plot on the same graph the true values TARGt and the predicted values Zt

Comments on the graph?

Compute the Mean Relative Errors of Prediction MREP on NewTrain , using the formula

MREP(NewTrain) = average (| Zt - TARG t | / TARGt) over all cases in NewTrain

Compute similarly MREP(NewTest). Comments ?