**Math6535 spring2018 Deep learning and data mining Robert Azencott**

**HW2**

*1 Describe your Data set and the associated classification task*

the **number of cases N** in the data set must be at least 5,000

- **Describe in words**  the automatic classification task your HW2 will study :

- Outline the characteristics of each descriptor D1 D2 …Dp available for all case; the number p must remain **fixed** for all cases ;

- For each descriptor Di indicate if its values are real numbers and in which range they lie, or if Di takes only a finite number of values ; If Di has values described by names or acronyms, recode these values by numbers (if you use an unbiased binary coding for these names, note that the corresponding descriptor is replaced by a finite set of new binary descriptors)

- Outline the pragmatic meaning of each descriptor

- present in words the distinct classes C1 C2 .. Cr involved in the classification task; number r of classes? Number Nj of cases in each class Cm?

*2 Select a training set, a test set, and an MLP architecture*

- Give the size of each class within the training set and within the test set ; give the proportions of cases in each class for the test set and for the training set; these two proportions should be similar for each class

- select an MLParchitecture with three layers L0 H L2 of dimensions p0, h, p2 ;

- to determine p0 indicate how you will precisely encode the value of each descriptor on specific “units” of layer L0

- fix p2= number “r” of classes, and on L2 assign one output node Um to each class Cm; indicate precisely how the output nodes will encode the true output OUTk prescribed for each input INPk in the database ;

- explain precisely how you will transform the vector output OUTk generated by the MLP for any given input INPk , in order to decide into which class Cm with m = m(k) , the trained MLP classifies INPk

*3. Select 3 tentative sizes h for the hidden layer*

To estimate a "small" plausible value h= "s" < p0 , apply PCA analysis to your set of N input vectors to generate p0 eigenvalues ordered in decreasing order, and compute the smallest number "s" of eigenvalues preserving 95% of the total sum of eigenvalues

To estimate an intermediate plausible value h= S > s,

- apply PCA analysis to the set of Nj input vectors corresponding to the class Cj, to generate and Nj eigenvalues in decreasing order, and compute the smallest number "Sj" of eigenvalues preserving 90% of the total sum of eigenvalues;

- then define h= "S" = S1 + ... + Sk

To define another larger value of h, take h= "SL = 2S

*3. Implement learning by gradient descent*

- Select explicitly the loss function (also called error function) defined by mean squared error MSE on the training set ; select explicitly the sizes of the test set and of the training set ; make sure that the freqency of each class Cj in the test set is roughly equal to its frequency in the training set; for classes with very small frequency use redundant copies of these cases to artificially inflate the class size

- Select one existing open source software tool ST you are choosing to implement MLP learning rule (for instance in Matlab, or in R, or in Tensorflow)

- for this specific MLP software tool ST explain and list clearly what are the options offered by ST for MLP learning, namely for initialization of the weights, for batch learning , for the successive gradient descent steps sizes, for stopping the learning, for intermediary outputs to monitor learning quality

- select a batch size B and a time dependent gain rate  **ε(n) = constant / n** for the gradient descent step size ; select an option for stopping the learning; apply a random initialization of weights and thresholds

For each one of the 3 preceding MLP architectures , use the software tool ST to perform automatic learning, which will naturally implement the following steps

- at STEP(n-1) select a new batch BATn containing B cases, and apply the learning rule to update the last vector of weights W(n-1) into a new vector Wn . We include formally the list of thresholds in the vector of weights

- compute the Root Mean Squared Error RMSEn = sqrt(MSEn) by running the MLP parametrized by Wn on the current Batch BATn, and compute the vector Gn = gradient of MSEn at step n by the formula Gn = ( 1/ ε(n) ) [ W(n+1)- W(n) ]

- compute and plot the curve n🡺 RMSEn

- compute and plot the curve n🡺 || W(n+1)- W(n)||

- compute and plot the curve n🡺 ||G\_n|| = ( 1/ ε(n) ) || W(n+1)- W(n)||

- comment on these three curves

*4 Evaluation of terminal MLP\**

- After learning is stopped , call W\* the terminal set of weights; the trained MLP\* is now parametrized by W\*;

compute the RMSE\* of MLP\* on the whole training set and on the whole test set; compare these performances, taking account of estimation errors on frequencies ;

-after learning is stopped compute as above the terminal gradient G\* of MSE (W\*); ; compute the histogram of the absolute values of the coordinates of G\*; can you consider that G\* is approximately zero?

- compute the confusion matrix Conf of MLP\* defined by

Conf(m,n) = percentage (within class Cm) of cases which were truly of class Cm and which were classified into class Cn by MLP\*

- Note that there are two confusion matrices, one for the training set and one for the test set. Compare these two matrices

- Compare the percentages of overall correct classifications obtained on the traing set and on the test set. Take account of estimation errors on these two percentages

*5 Impact of various learning options*

Evaluate experimentally the impact on final performance of various factors such as changes in

batch size ,

initialization,

gradient descent step size ,

dimension h of the hidden layer

*6 PCA analysis of hidden layer*

- For each one of the three hidden layer sizes h= s, S, SL and after learning is completed , generate and store the configuration Hk of the hidden layer H computed by the MLP\* when the input is INPk . The set of all Hk defines a “cloud” of vectors in R**h**

- Apply a PCA analysis to this cloud ; evaluate the h eigenvalues of the corresponding covariance matrix Covar of the cloud and **plot** these eigenvalues in decreasing order; indicate the cutoff dimension corresponding to 90% of the PCA “energy” ; project all the vectors Hk in the R**3** space spanned by the first three eigenvectors of the matrix Covar; Use different colors to display the projected classes