1 Multi way linear classifier

In order to compare our MLP implementation for multi way classification, we have implemented 3 linear classifiers. Each one of them does a linear prediction :

1.1 Linear regression with squared error

For this part, each datapoint belongs to one of 5 class. Thus, the label attributed for each datapoint is no more a scalar but a binary 5-dimensinal vector. For example if x belongs to the class number 2, its label will be $\widetilde{t_i} = [0,1,0,0,0]$. Let consider the classification error on the training set to be :

$$E_2(W) = \sum_{i=1}^{N} \left\| y_i - \tilde{t}_i \right\|^2$$

The linear classifier computes the 5-dimensional output : $y_i = W[x_i; 1]$ (we had a constant coordinate in each datapoints in order to incorporate the bias in the matrix W). Then the class is determined by $argmax_j y_i(j)$. In the case of the squared error above, the course shows that the matrix W that optimizes the regression on the training set is given by :

$W_{optimal} = (\Phi^T \Phi)^{-1} \Phi T$ where $\Phi$is a matrix where each column is a datapoint, and T is the the matrix that regroup the corresponding labels.

By computing this classifier, we obtain 1228 misclassified elements on the test set (over a total of 5400 datapoints), which means an accuracy of 77%.

1.2 Squared error with Tichonov regularizer

A classic improvement of linear regression is to constrain the weights using a tichonov regularizer. The error is now :

$$E_2(W) = \frac{1}{2} \sum_{i=1}^{N} \left\| y_i - \tilde{t}_i \right\|^2 + \frac{\nu}{2} \left\| W \right\|^2$$

In this case, for a fixed ν, the course gives us the optimal weights for the regression on the training set (using the same notation introduced above) :

$$W_{tichonov} = (\Phi^T \Phi + \nu I)^{-1} \Phi^T T$$

In order to find the best regularizer ν, we compute a 10-folds cross validation on the training set, for different values of ν. We have selected the regularizer that minimizes the risk, which is computed as follow : after splitting the training set in 10 equals parts, $risk = \frac{1}{10} \sum_{m=1}^{10} R^{(-m)}(W^{(-m)})$, where $R^{(-m)}$is the error E_2computed only of the m^{th}part of the classifier trained with the 9 other parts of the training set.

Figure 1 shows the evolution of the risk with ν. Since minimum is obtained for $\nu \approx 400$, we have take this value to build the regularized classifier. This one

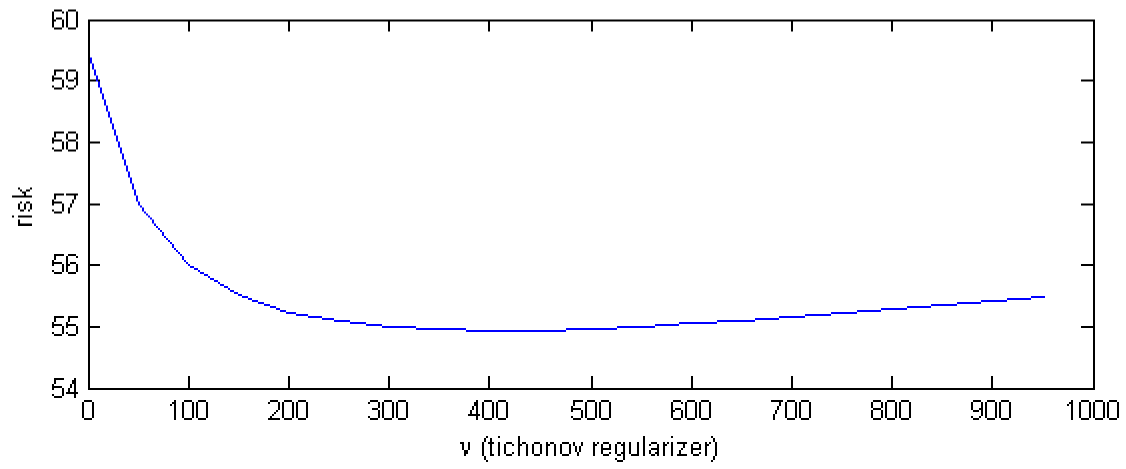

Figure 1: Evolution of the risk (defined in section 1.2) with the tichonov regularizer.

makes a good result on the test set, with only 817 misclassified elements (85% accuracy).

1.3 Logistic regression

Another improvement is to consider a logistic error :

$$E_{log}(W) = \frac{1}{2}\sum_{i=1}^{N} \text{lsexp}(y(x_i)) - \tilde{t_i^T} y(x_i)$$

There is no analytic solution of this kind of error. We are forced to implement a gradient descent. The course p. 145 gives the value of the gradient :

$$\nabla_{y_i} E_i = \sigma(y_i) - \tilde{t_i} = [\frac{e^{y_i(k)}}{\sum_k e^{(k)}}]_{k=1...5}$$

For early stopping, we have simply taken a validation set of 1/3 of the whole training set. Early stopping is triggered whenever the error on the validation does not decrease significantly ($error(epoch) > 0.95 * error(epoch - 1)$) The learning rate is 0.005. After launching the training of this classifier 10 times, we observe an average missclassification of 1009 elements on the test set (with a standard deviation of 55 elements).