Cognitive Algorithms Assignment 3

Linear Classification and Brain-Computer Interfacing

Due on Tuesday, May 30, 2017 , 10 am via ISIS

In this assignment you will compare the three linear classification algorithms that you encountered in the lecture - the Perceptron, the Nearest Centroid Classifier (NCC) and the Linear Discriminant Analysis (LDA). This comparision is done on a toy data set and on two different real data sets - the USPS data from the last assignment, and a Brain-Computer-Interface (BCI) data set.
Download the python template `assignment3.py` and the data set `bcidata.mat` from the ISIS web site. Your task will be to implement LDA and use the provided code to analyse the data.
The BCI data set consists of preprocessed EEG data $X \in \mathbb{R}^{5\times 62\times 5322}$ and stimulus labels $Y \in \mathbb{R}^{2\times 5322}$ during a copy-spelling paradigm with a P300 speller. The data matrix X contains 5 selected time windows of EEG activity at 62 electrodes after a visual stimulus was presented on the screen in front of the subject. If the first row of Y is 1, the stimulus was a target stimulus, if the second row of Y is 1, the stimulus was a non-target stimulus. The goal is to predict if the simulus was a target or not given the EEG.

1. (**18 points**) Implement a linear discriminant analysis (LDA) classifier by completing the function stub `train_lda`, that is, find a vector $\mathbf{w}$ such that

   $$\mathbf{w} = \underset{\mathbf{w}}{\operatorname{argmax}} \frac{\mathbf{w}^\top S_B \mathbf{w}}{\mathbf{w}^\top S_W \mathbf{w}}$$

   where S_B denotes the 'between-class scatter' and S_W denotes the 'within-class scatter'

   $$\begin{aligned} S_B &= (\mathbf{w}_+ - \mathbf{w}_-)(\mathbf{w}_+ - \mathbf{w}_-)^\top \\ S_W &= \frac{1}{N_+}\sum_{i=1}^{N_+}(\mathbf{x}_{+i} - \mathbf{w}_+)(\mathbf{x}_{+i} - \mathbf{w}_+)^\top + \frac{1}{N_-}\sum_{j=1}^{N_-}(\mathbf{x}_{-j} - \mathbf{w}_-)(\mathbf{x}_{-j} - \mathbf{w}_-)^\top \end{aligned}$$

   and $\mathbf{w}_+$, $\mathbf{w}_-$ denote the respective class means.

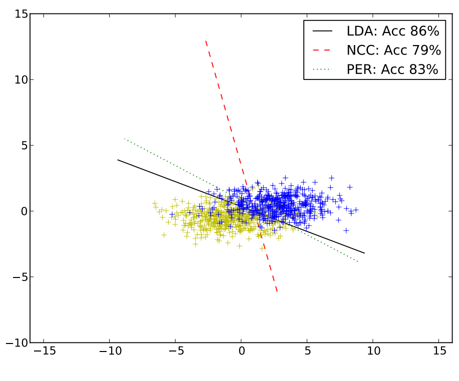
2. (**6 points**) Test your LDA implementation with the provided function `compare_classifiers_toy`. It generates a 2D toy data set and plots the resulting separting hyperplanes for the three linear classification methods as in Figure 1(a). Answer the following short questions:

   a) Run the function several times - what do you notice for the Perceptron as compared to NCC or LDA? In one sentence, explain the behaviour of the perceptron.

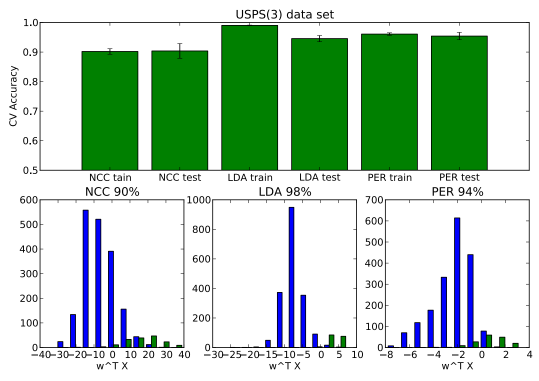
   b) Have a look in the code how the toy data is generated - is LDA optimal for this type of data?

   c) How would you have to change the data generation such that NCC and LDA yield the same result?

3. (**3 points**) Examine the function `crossvalidate`. Briefly explain in your own words what this function does. When we want to compare the performance of different classifiers, which values should we look at - the train or the test accuracies?



(a)



(b)

Figure 1: (a) Classification boundaries of LDA, NCC and Perceptron on a 2D toy data set. (b) Classification result for NCC, LDA and Percpetron on the USPS data set for digit 3. The top row shows classification accuracies for training and test blocks obtained through 5-fold cross-validation. The bottom row shows histograms of classifer outputs.

4. (**3 points)** Call `compare_classifiers` for a digit of your choice of the USPS data set, as well as for the BCI data. It plots the histogram of classifier outputs and the classification accuracies for the NCC, the LDA and the perceptron as in Figure 1(b). Which algorithm (Nearest Centroid Classifier, Linear Discriminant Analysis or Perceptron) would you prefer for which task?

Please hand in your completed `assignment3.py` via ISIS. Please write your name and your Matrikel Number as the first line of the code. Also hand in a pdf file that contains your name, the answers to the questions, the plots generated in Question 4, as well as your code of the function `train_lda`.