CSCI 5521: Spring'17
Introduction To Machine Learning

Homework 4

(Due Fri, April 28, 11:55 pm)

1. (30 points) Let $\mathcal{X} = \{\mathbf{x}^1, \ldots, \mathbf{x}^N\}$ with $\mathbf{x}^t \in \mathbb{R}^D, t = 1, \ldots, N$ be a given training set. Assume that the dataset is centered, i.e., $\sum_{t=1}^N \mathbf{x}^t = 0 \in \mathbb{R}^D$. We focus on performing linear dimensionality reduction on the dataset using PCA (principal component analysis). With PCA, for each $\mathbf{x}^t \in \mathbb{R}^D$, we get $\mathbf{z}^t = W^T\mathbf{x}^t$, where $\mathbf{z}^t \in \mathbb{R}^d, d < D$, is the low dimensional projection, and $W \in \mathbb{R}^{D\times d}$ is the PCA projection matrix. Let $\Sigma = \frac{1}{N}\sum_{t=1}^N \mathbf{x}^t(\mathbf{x}^t)^T$ be the sample covariance matrix. Further, let $\mathbf{v}^t = W\mathbf{z}^t$ so that $\mathbf{v}^t \in \mathbb{R}^D$.

   (a) (10 points) Professor HighLowHigh claims: $\mathbf{v}^t = \mathbf{x}^t$ for all $t = 1, \ldots, N$. Is the claim correct? Clearly explain and prove your answer with necessary (mathematical) details.

   (b) (20 points) Professor HighLowHigh also claims:

$$\sum_{t=1}^N \|\mathbf{x}^t\|_2^2 - \sum_{t=1}^N \|\mathbf{v}^t\|_2^2 = \sum_{t=1}^N \|\mathbf{x}^t - \mathbf{v}^t\|_2^2 \ ,$$

   where for a vector $\mathbf{a} = [a^1 \cdots a^k]$, $\|\mathbf{a}\|_2^2 = \sum_{t=1}^k (a^k)^2$. Is the claim correct? Clearly explain and prove your answer with necessary (mathematical) details.

2. (40 points) Let $\{(\mathbf{x}^1, \mathbf{r}^1), \ldots, (\mathbf{x}^N, \mathbf{r}^N)\}, \mathbf{x}^t \in \mathbb{R}^d, \mathbf{r}^t \in \mathbb{R}^k$ be a set of N training samples. We consider training a multilayer perceptron as shown in Figure 1. We consider a general setting where the transfer functions at each stage are denoted by g, i.e.,

$$z_h^t = g(a_h^t) = g\left(\sum_{j=1}^d w_{h,j}x_j^t + w_0\right) \qquad \text{and} \qquad y_i^t = g(a_i^t) = g\left(\sum_{h=1}^H v_{i,h}z_h^t + v_{i0}\right) \ ,$$

   where a_h^t, a_i^t respectively denote the input activation for hidden node h and output node i. Further, let $L(\cdot,\cdot)$ be the loss function, so that the learning focuses on minimizing:

$$E(W, V|\mathcal{Z}) = \sum_{t=1}^N \sum_{i=1}^k L(r_i^t, y_i^t) \ .$$

   (a) (15 points) Show that the stochastic gradient descent update for $v_{i,h}$ is of the form $v_{i,h}^{\text{new}} = v_{i,h}^{\text{old}} + \Delta v_{i,h}$, with the update

$$\Delta v_{i,h} = \eta\Delta_i^t z_h^t \ ,$$

   where $\Delta_i^t = -g'(a_i^t)\frac{\partial L(r_i^t, y_i^t)}{\partial y_i^t}$.

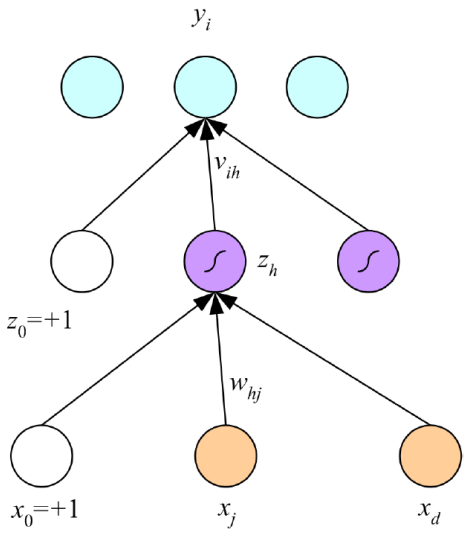

Figure 1: Two layer perceptron.

(b) (25 points) Show that the stochastic gradient descent update for $w_{h,j}$ is of the form $w_{h,j}^{\text{new}} = w_{h,j}^{\text{old}} + \Delta w_{h,j}$, with the update

$$\Delta w_{h,j} = \eta \Delta_h^t x_j^t \ ,$$

where $\Delta_h^t = g'(a_h^t) \sum_{i=1}^k \Delta_i^t v_{i,h}$.

Programming assignments:

The next problem involves programming. For Question 3, we will be using the 2-class classification datasets from `Boston50` and `Boston75`. In particular, we will develop code for Fisher's Linear Discriminant Analysis (LDA) by projecting the data into one dimension such that the classes are separated in the sense of LDA.

3. (30 points) We will develop code for 2-class Fisher's Linear Discriminant Analysis (LDA) and apply the classifier to `Boston50` and `Boston75`. Consider a 2-class classification dataset $\{(\mathbf{x}^1, r^1), \ldots, (\mathbf{x}^N, r^N)\}, \mathbf{x}^t \in \mathbb{R}^d, r^t \in -1, +1$. Building a LDA-based classifier needs two steps:

   (a) Projection step: Finding a linear projection $\mathbf{w} \in \mathbb{R}^d$ which optimizes the LDA objective and produces one dimensional version of the dataset, i.e., $\{(z^1, r^1), \ldots, (z^N, r^N)\}$ with $z^t = \mathbf{w}^T \mathbf{x}^t, t = 1, \ldots, N$.

   (b) Classification step: Build a suitable classifier for the one dimensional dataset. For the homework, we will focus on finding a suitable threshold z_0 such that our classification is based on:

$$y^t = \begin{cases} +1 \ , & \text{if } z^t \geq z_0 \ , \\ -1 \ , & \text{otherwise} \ . \end{cases}$$

   We will pick z_0 so that to maximize the training set accuracy.

We will develop code for `MyFLDA2` with corresponding `MyFLDA2.fit(X,y)` and `MyFLDA2.predict(X)` functions.

We will compare the performance of `MyFLDA2` with `LogisticRegression`[1] on two datasets: `Boston50` and `Boston75`. Using `my_cross_val` with 5-fold cross-validation, report the error rates in each fold as well as the mean and standard deviation of error rates across all folds for the two methods: `MyFLDA2` and `LogisticRegression`, applied to the two 2-class classification datasets: `Boston50` and `Boston75`.

You will have to submit (a) **code** and (b) **summary of results**:

(a) **Code**: You will have to submit code for `MyFLDA2()` as well as a wrapper code `q3()`.

For `MyFLDA2()`, you are encouraged to consult your code from HW2 and HW3 (or code for classifiers in `scikit-learn`). You need to make sure you have `__init__`, `fit`, and `predict` implemented in `MyFLDA2`. Your class will **NOT** inherit any base class in `sklearn`.

The wrapper code (main file) has no input and is used to prepare the datasets, and make calls to `my_cross_val`(`method`,X,$\mathbf{y}$,k) to generate the error rate results for each dataset and each method. The code for `my_cross_val`(`method`,X,$\mathbf{y}$,k) must be yours (e.g., code you wrote in HW1 with modifications as needed) and you cannot use `cross_val_score()` in `sklearn`. The results should be printed to terminal (not generating an additional file in the folder). Make sure the calls to `my_cross_val`(`method`,X,$\mathbf{y}$,k) are made in the following order and add a print to the terminal before each call to show which method and dataset is being used:

1. `MyFLDA2` with `Boston50`; 2. `MyFLDA2` with `Boston75`; 3. `LogisticRegression` with `Boston50`; 4. `LogisticRegression` with `Boston75`.

For the wrapper code, you need to make a `q3.py` file for it, and one should be able to run your code by calling "`python q3.py`" in command line window.

(b) **Summary of results**: For each dataset and each method, report the test set error rates for each of the $k = 5$ folds, the mean error rate over the k folds, and the standard deviation of the error rates over the k folds. Make a table to present the results for each method and each dataset (4 tables in total). Each column of the table represents a fold and add two columns at the end to show the overall mean error rate and standard deviation over the k folds.

Additional instructions: Code can only be written in Python (**not** IPython notebook); no other programming languages will be accepted. One should be able to execute all programs directly from command prompt (e.g., "`python q3.py`") without the need to run Python interactive shell first. Test your code yourself before submission and suppress any warning messages that may be printed. Your code must be run on a CSE lab machine (e.g., csel-kh1260-01.cselabs.umn.edu). Please make sure you specify the full Python version you are using as well as instructions on how to run your program in the README file (must be readable through a text editor such as Notepad). Information on the size of the datasets, including number of data points and dimensionality of features, as well as number of classes can be readily extracted from the datasets in `scikit-learn`. Each function must take the inputs in the order specified in the problem and display the output via the terminal or as specified.

[1] You should use `LogisticRegression` from `scikit-learn`, similar to HW1, HW2, and HW3.

For each part, you can submit additional files/functions (as needed) which will be used by the main file. Please put comments in your code so that one can follow the key parts and steps in your code.

Extra Credit Problem:

EC1 **(25 points)** Consider a two class classification problem setting with training data $\{(\mathbf{x}^t, y^t)\}_{t=1}^N$ where $y^t \in \{0, 1\}$ and $\mathbf{x}^t \in \mathbb{R}^d$. Consider a linear activation model $a^t = a(\mathbf{x}^t, \mathbf{w}) = \mathbf{w}^T\mathbf{x}^t$, and transfer function of the form

$$f_{\text{tlu}}(a) = \max(0, a) + \frac{1}{2}\min(0, a) \ . \tag{1}$$

The square loss on the entire dataset in terms of the activation function f_{tlu} is given by

$$L_{sq}^{(\text{tlu})}(\mathbf{w}) = \sum_{t=1}^{N} \left(y^t - f_{\text{tlu}}(\mathbf{w}^T\mathbf{x}^t)\right)^2 \ .$$

Is $L_{sq}^{(\text{tlu})}(\mathbf{w})$ a convex function of the parameter $\mathbf{w}$? Clearly explain and prove your answer with necessary (mathematical) details, including the definition of convexity you are using.

Follow the rules strictly. If we cannot run your code, you will not get any credit.

- **Things to submit**

  1. hw3.pdf: A document which contains the solution to Problems 1, 2, 3 (and extra credit problem) including the summary of results for 3 and 4. This document must be in PDF format (no word, photo, etc., is accepted). If you submit a scanned copy of a hand-written document, make sure the copy is clearly readable, otherwise no credit may be given.
  2. Python code for Problem 3 (must include the required `q3.py`).
  3. README.txt: README file that contains your name, student ID, email, instructions on how to run your code, the full Python version (e.g., Python 2.7) your are using, any assumptions you are making, and any other necessary details. The file must be readable by a text editor such as Notepad.
  4. Any other files, except the data, which are necessary for your code.