# CSCI 5521: Spring'17
# Introduction To Machine Learning

# Homework 3

**(Due Sat, April 8, 12:00 pm (noon))**

1. (25 points) Let $\mathcal{Z} = \{(x_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ be a given training set. We consider the following regularized logistic regression objective function:

$$f(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n}\left\{-y_i\mathbf{w}^T\mathbf{x}_i + \log(1+\exp(\mathbf{w}^T\mathbf{x}_i))\right\} + \frac{\lambda}{2}\|\mathbf{w}\|^2 \ ,$$

   where $\lambda > 0$ is a constant. Let $\mathbf{w}^*$ be the global minimizer of the objective, and let $\|\mathbf{w}^*\|_2 \leq c$, for some constant $c > 0$.

   (a) (10 points) Clearly show and explain the steps of the projected gradient descent algorithm for optimizing the regularized logistic regression objective function. The steps should include an exact expression for the gradient.

   (b) (5 points) Is the objective function strongly convex? Clearly explain your answer using the definition of strong convexity.

   (c) (5 points) Is the objective function smooth? Clearly explain your answer using the definition of smoothness.

   (d) (5 points) Let $\mathbf{w}_T$ be the iterate after $T$ steps of the projected gradient descent algorithm. What is a bound on the difference $f(\mathbf{w}_T) - f(\mathbf{w}^*)$? Clearly explain all quantities in the bound.

2. (25 points) Let $\mathcal{X} = \{\mathbf{x}^1, \ldots, \mathbf{x}^N\}, \mathbf{x}^t \in \mathbb{R}^d$ be a set of $n$ samples drawn i.i.d. from a mixture of $k$ multivariate Gaussian distribution in $\mathbb{R}^d$. For component $G_i, i = 1, \ldots, k$, let $\pi_i, \mu_i, \Sigma_i$ respectively denote the prior probability, mean, and covariance of $G_i$. We will focus on the expectation maximization (EM) algorithm for learning the mixture model, in particular for estimating the parameters $\{(\pi, \mu_i, \Sigma_i), i = 1, \ldots, k\}$ as well the posterior probabilities $h_i^t = p(G_i|\mathbf{x}^t)$.

   (a) (10 points) In your own words, describe the EM algorithm for mixture of Gaussians, highlighting the two key steps (E- and M-), illustrating the methods used in the steps on a high level, and what information they need.

   (b) (10 points) Assuming the posterior probabilities $h_i^t$ are known, show the estimates of the component prior, mean, and covariance $\pi_i, \mu_i, \Sigma_i, i = 1, \ldots, N$ given by the M-step (you do not need to show how they are derived).

   (c) (5 points) Assuming the component prior, mean, and covariance $\pi_i, \mu_i, \Sigma_i, i = 1, \ldots, N$ are known, show how the posterior probabilities $h_i^t$ are computed in the E-step.

**Programming assignments:**

The next problem involve programming. For Question 3, we will be using the 2-class classification datasets from `Boston50`, `Boston75`, and for Question 4, we will be using the 10-class classification dataset from `Digits` which were used in Homework 1. For Q3, we will develop code for 2-class logistic regression with only one set of parameters $(\mathbf{w}, w_0)$. For Q4, we will develop code for $k$-class logistic regression with $k$ sets of parameters $(\mathbf{w}_i, w_{i0})$.

3. (25 points) We will develop code for 2-class logistic regression with one set of parameters $(\mathbf{w}, w_0)$. Assuming the two classes are $\{C_1, C_2\}$, the posterior probability of class $C_1$ is given by

$$\log P(C_1|\mathbf{x}) = \frac{\exp(\mathbf{w}^T\mathbf{x} + w_0)}{1 + \exp(\mathbf{w}^T\mathbf{x} + w_0)} ,$$

and $P(C_2|\mathbf{x}) = 1 - P(C_1|\mathbf{x})$.

We will develop code for `MyLogisticReg2` with corresponding `MyLogisticReg2.fit(X,y)` and `LogisticReg2.predict(X)` functions. Parameters for the model can be initialized following suggestions in the textbook.

We will compare the performance of `MyLogisticReg2` 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: `MyLogisticReg2` 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 `MyLogisticReg2()` as well as a wrapper code `q3()`.

For `MyLogisticReg2()`, you are encouraged to consult the code for `MultiGaussClassify()` from HW2 (or code for classifiers in `scikit-learn`). You need to make sure you have `__init__`, `fit`, and `predict` implemented in `MyLogisticReg2`. 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 made 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. `MyLogisticReg2` with `Boston50`; 2. `MyLogisticReg2` 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.

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

(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.

4. (25 points) We will develop code for c-class logistic regression with $c$ sets of parameters $\{(\mathbf{w}_i, w_{i0}), i = 1, \ldots, c\}$. Assuming the $c$ classes are $\{C_1, C_2, \ldots, C_c\}$, the posterior probability of class $C_i$ is given by

$$\log P(C_i|\mathbf{x}) = \frac{\exp(\mathbf{w}_i^T\mathbf{x} + w_{i0})}{\sum_{i'=1}^{c} \exp(\mathbf{w}_{i'}^T\mathbf{x} + w_{i'0})} \ .$$

We will develop code for `MyLogisticRegGen` with corresponding `MyLogisticRegGen.fit(X,y)` and `LogisticRegGen.predict(X)` functions. Parameters for the model can be initialized following suggestions in the textbook.

We will compare the performance of `MyLogisticRegGen` with `LogisticRegression`[2] on one dataset: `Digits`, for which the number of classes $c = 10$. 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: `MyLogisticRegGen` and `LogisticRegression`, applied to the 10-class classification dataset: `Digits`.

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

(a) **Code**: You will have to submit code for `MyLogisticRegGen()` as well as a wrapper code `q4()`.

For `MyLogisticRegGen()`, you are encouraged to consult the code for `MultiGaussClassify()` from HW2 (or code for classifiers in `scikit-learn`). You need to make sure you have `__init__`, `fit`, and `predict` implemented in `MyLogisticRegGen`. 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,`$X$`,y,`$k$`)` to generate the error rate results for each dataset and each method. The code for `my_cross_val(method,`$X$`,y,`$k$`)` must be yours (e.g., code you made 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,`$X$`,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. `MyLogisticRegGen` with `Digits`; 2. `LogisticRegression` with `Digits`.

*For the wrapper code, you need to create a `q4.py` file, and one should be able to run your code by calling "`python q4.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 (2 tables in total). Each column of the table represents a fold and add

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

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.

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.

**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 4 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 Problems 3 and 4 (must include the required `q3.py` and `q4.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.