# CS534 — Implementation Assignment 1 — Due April 18th noon, 2014

## General instruction.

1. You are encouraged to consider Matlab for your implementation. However, other languages are also accepted.
2. You can work solo or in team of 2 people. Each team will only need to submit one copy of the source code and report.
3. Your source code and report will be submitted through the TEACH site

`https://secure.engr.oregonstate.edu:8000/teach.php?type=want_auth`

Please clearly indicate your team members' information.
4. You will be graded based on your code as well as the report. In particular, **the clarity and quality of the report will be worth 10 pts**. So please write your report in clear and concise manner. Clearly label your figures, legends, and tables if any.

## Linear regression with $L_2$ regularziation (40 pts)

For the first part of the assignment, you need to implement linear regression with $L_2$ (quadratic) regularization, which learns an weight vector $\mathbf{w}$ that optimize the following regularized SSE objective:

$$\sum_{i=1}^{N}(y_i - \mathbf{w}^{\mathbf{x}})^2 + \lambda|\mathbf{w}|^2$$

To optimize this objective, you can either use the gradient descent algorithm or directly use the close-form analytical solution. Both solutions, if implemented correctly, will be accepted.

Data: you are provided with a training set and a test set, both in csv format. Each row of the data file contains the data for one example. The first column contains the dummy feature taking the constant value of 1 for all examples and the last column stores the target $y$ values for each example. Here is what you need to do and report:

1. Experiments (20 pts): You will test the effect of the regularizer on your linear regressor. It is often the case that we don't really what the right $\lambda$ value should be and we will need to consider a range of different $\lambda$ values. For this project, consider at least the following values for $\lambda$: $0, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 100$. Feel free to explore the choices of $\lambda$ in a more finer search resolution. Report the SSE on the training and testing for each value of $lambda$. What trend do you observe from the training SSE as we change $\lambda$ value? What tread do you observe from the test SSE? Provide an explanation for the observed behaviors.

2. Explorations (20 pts): The provided data set has 45 dimensions (including the first dummy feature). Only a few of them are actually useful. Design your own strategy to identify what set of features are relevant to the target variable. As a starting point, given proper $\lambda$ regularization, the weights of the irrelevant features will likely to be very small. What will happen if we remove these features?

## Perceptron and voted perceptron (50 pts)

For the second part, you will code the *batch perceptron* and *voted perceptron* algorithms. You are provided with two data sets for this task. You will test your batch perceptron algorithm on the first data set (twogaussian), which contains two linearly separable gaussian classes. Each row of the data corresponds to one instance and the first column stores the target class variable. Note that for this data set the dummy feature of constant 1 is not included and need to be added by you in your implementation. Your report needs to include:

1. (10pts) A plot of the classification error on the training set as a function of the number of training epoches. Note that one epoch of training goes through the full training set exactly once.
2. (5pts) A scatter plot of the training data using different colors for different classes. On this scatter plot, please plot the final linear decision boundary learned by your perceptron algorithm (please also provide the weights output by your perceptron algorithm).

You will test your voted perceptron implementation on the second data set iris-twoclass. This is based on a commonly used bench-mark data set iris. In particular, we extracted two classes, and two input features from the original problem. Similarly, the first column stores the target class variable, and the dummy feature of constant 1 is not included. You need to provide the following in your report:

1. (15 pts) A plot of the classification error on the training set as a function of the number of training epoches (up to 100 epoches). Note that one epoch of training goes through the full training set exactly once. Voted perceptron is sensitive to the order in which the training examples are presented to the learning algorithm. For this, you will test two different variants. In the first variant, the training examples are always presented in the same order as given in the data file. For the second variant, each training epoch, the order is randomly shuffled. Produce the plots for both variants (for the second variant, you might want to take a few random runs and average them to get a more smooth curve) and comment on their difference and provide an explanation for the observation.

2. (10 pts) Visualize the final decision boundary produced by your voted perceptron algorithm (trained for 100 epoches). To produce this decision boundary, one strategy is to sample a large number of $\mathbf{x}$ points on a fine grid in the input space, and compute their predicted output. You can then visualize the decision boundary by plotting these points using different colors based on their predicted classes.

3. (10 pts) For voted perceptron, there is a simple modification people often use in order to avoid storing all intermediate weight vectors. That is to compute

$$\mathbf{w}_{avg} = \sum_{n=0}^{n=N} c_n \mathbf{w}_n$$

   and use the following decision rule:

$$h(\mathbf{x}) = sgn\{\mathbf{w}_{avg} \cdot \mathbf{x}\}$$

   Please compute $\mathbf{w}_{avg}$ based on the weights that are learned by your voted perceptron algorithm (100 epoches), and plot the linear decision boundary it produces on a scatter plot of the training data. Is this decision boundary identical to the boundary visualized in 2 (you could overlay the two figures to answer this question)? Provide an explanation for your observation.

**Clarifications for learning rate.** For batch perceptron, please use $\lambda = 1$, which is referred to as the fixed increment perceptron. For voted perceptron, we do not use a learning rate.