**C S 487/519 Applied Machine Learning I**

# Single-layer Linear Neural Networks

1. **Objective**

In this *individual* homework, you are required to get familiar with several single-layer linear neural networks.

# Requirements

## Tasks

* + 1. (20 points) Design and implement a **Perceptron** binary classifier.
    2. (20 points) Design and implement an **Adaline** binary classifier.
    3. (20 points) Design and implement a Stochastic Gradient Descent (**SGD**) binary classifier.
    4. (20 points) Write program to test the different classifiers.
       1. (1 point) Your program should be called **main.py**.
       2. (2 points) It should have two arguments: (i) classifier name, which can be ‘perceptron’, ‘adaline’, and ‘sgd’, and (ii) data file (including path information). You can have other arguments depending on the design of your program.
       3. (2 points) It should have proper error checking functions (e.g., make sure the classifier name is a valid one).
       4. (15 points) It calls the different classifiers to train models, make predictions, and report prediction errors.
    5. (18 points) Write a report **report.pdf** to analyze the predictive power and the running time of different classifiers.
       1. (2 points) For each classifier, you should report the accuracy of the prediction, where accuracy is the percentage of the correctly classified instances. Note that you do not need to separate the dataset to training and testing. The accuracy can be reported for different iterations.
       2. (3 points) For each classifier, please report the errors or costs in each iteration and plot figures for the errors/costs for all the iterations.
       3. (5 points) Each classifier needs to be tested using two datasets: (1) Iris (by treating one class as positive class and the other two classes as negative class) and (2) another dataset. You need to find your second dataset from UCI machine learning repository. This dataset needs to be bigger than the Iris dataset (more samples and more features).
       4. (5 points) Properly analyze the classifiers’ behavior. For example, how do your classifiers converge? what is the effect of feature scaling to your classifiers.
       5. (3 points) Analysis on any other aspects that are not mentioned above and that you think important. For example, the effect of different learning rates on model convergence.
    6. (2 points) Write a readme file **readme.txt** with the commands to run your code. An **example** of your readme file is as follows. If you need more information to tell your TA how to grade your homework, please write them in the readme file.

python main.py perceptron iris.data

python main.py adaline iris.data

python main.py sgd iris.data

python main.py perceptron <dataset2>

python main.py adaline <dataset2>

python main.py sgd <dataset2>

* + 1. (20 points) (only for CS 519 students) Implement a multiclass classifier using One-vs-Rest strategy and the SGD binary classifier. Properly test the classifier using Iris data set and another dataset with more than two class labels. Your second dataset should be from UCI machine learning repository. Include a proper analysis for this multiclass classifier in the report. Include the commands to run this classifier in the readme file.

References:

UCI machine learning repository (<https://archive.ics.uci.edu/ml/index.php>).

The Iris dataset can be downloaded from [here](https://www.cs.nmsu.edu/~hcao/teaching/cs487519/data/iris.data) and the description of the data can be downloaded from [here](https://www.cs.nmsu.edu/~hcao/teaching/cs487519/data/iris.names.txt).

## Other requirements

* You are **NOT** allowed to use functions provided in scikit-learn libray.
* Your Python code should be written for Python version 3.5.2 or higher.
  + Please properly organize your Python code (e.g., create proper classes, modules). Each required task had better be implemented in a separate python file and imported into the main script. For example, to implement the perceptron classifier, you can create the script **myperceptron.py**, then use “import myperceptron” in the main.py file to test your implementation.

# Submission instructions

Compress your python code and readme file to a zip file and upload it to Canvas.

# Grading criteria

* + 1. The score allocation has already been put beside the questions.
    2. Please make sure that you test your code thoroughly by considering all possible test cases. Your code may be tested using more datasets.
    3. 5 points will be deducted if submitted files (including file types, file names, etc.) do not follow the instructions.
    4. If the total points are more than 100. Your grades will be scaled to the range of [0,100].