CS 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.

2. Requirements

2.1 Tasks

(1) [10 points] Design and implement a Perceptron binary classifier.
(2) [10 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.

  (a) [3 points] Your program should have places to set 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.
  (b) [2 points] It should have proper error checking functions (e.g., make sure the classifier name is a valid one).
  (c) [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.

  (a) [2 points] For each classifier, you should report the accuracy of the prediction, where accuracy is the percentage of the correctly classified instances. The accuracy needs to be reported for each iteration (or epoch). Note that you do not need to separate the dataset to training and testing. The accuracy can be reported for different iterations.
  (b) [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.
  (c) [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). Given the nature of the classifiers, all the attribute values need to be numerical. If your second dataset contains categorical/discrete values, you need to properly convert them to numerical value. For example, if one column is car size with values small, medium, and large, you can convert the values to be 1 (for small), 2 (for medium), and 3 (for large). When you submit the dataset, you need to provide the original dataset (which contains categorical values) and the processed dataset (which contains only numerical values).
  (d) [5 points] Properly analyze the classifiers behavior. For example, how do your classifiers converge? what is the effect of feature scaling to your classifiers?

(e) [3 points] You can include 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 detailed instructions to run your program.

(7) [20 points] 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 (iris.data) and its description (iris.names.txt) need to be downloaded from the iris_data folder in Canvas.

2.2 Other requirements

- You are NOT allowed to use the classification functions provided in scikit-learn library.
- Your Python code should be written for Python version 3.10 or higher.
- Please write proper comments in your code to help the instructor and teaching assistants to understand it.
- Please properly organize your Python code (e.g., create proper classes, modules). You can put your code to Jupyter Notebook or a .py file.

3. Submission instructions

Put all your files (Python code, readme file, report, etc.) to a zip file named **hw2_<YourName>.zip** and upload it to Canvas.

4. Grading criteria

- **ZERO point** will be given if your code does not work. Please do not submit code that you did not test and make sure it works.
- The score allocation has been put beside the questions.
- FIVE points will be deducted if files are not submitted in the required format.
- If the total points are more than 100. Your grades will be scaled to the range of [0,100].
- Please make sure that you test your code thoroughly by considering all possible test cases. Your code may be tested using more datasets.