WOA7015 Advanced Machine Learning
Universiti Malaya
2023/24 Sem1

Individual Assginment 2 (10%)

1 Instruction

In today's and last week's lecture, we learned that the core concept of machine learning is to use a function to map between input and output. The overall steps are as follows:

1. weights is randomly assigned
2. calculate the output
3. calculate the loss
4. calculate the gradient
5. based on the gradient, update the weight
6. repeat step1 to step5 again until loss is minimized

In this assignment, you are required to implement a logistic regression using toy datasets. In the dataset, the first two columns are features and the third column is the class. Your task is to implement a logistic regression from scratch without using the built-in functions from libraries.

Your code should have the following functions

1. obj = CalcObj(XTrain, YTrain, wHat)
2. grad = CalcGrad(XTrain, YTrain, wHat)
3. wHat = UpdateParams(weight, grad, lr)
4. hasConverged = CheckConvg(oldObj, newObj, tol)
5. [wHat, objVals] = GradientDescent(XTrain, YTrain)
6. [yHat, numErrors] = PredictLabels(XTest, YTest, wHat)

where

- XTrain is an n×p dimensional matrix that contains one training instance per row
- YTrain is an n×1 dimensional vector containing the class labels for each training instance

- wHat is a p+1×1 dimensional vector containing the regression parameter estimates $\hat{w}_0, \hat{w}_1, ..., \hat{w}_p$
- grad is a p+1×1 dimensional vector containing the value of the gradient of the objective function with respect to each parameter in wHat
- lr is the gradient descent step size that you should set to lr=0.01
- obj, oldObj and newObj are values of the objective function
- tol is the convergence tolerance, which you should set to tol=0.001
- objVals is a vector containing the objective value at each iteration of gradient descent
- XTest is an m×p dimensional matrix that contains one test instance per row
- yTest is an m×1 dimensional vector containing the true class labels for each test instance
- yHat is an m×1 dimensional vector containing your predicted class labels for each test instance
- numErrors is the number of misclassified examples,i.e.the differences between yHat and yTest

2 Your Tasks

1. Train your logistic regression using Train_toydata.txt with the relevant functions and test your model with Test_toydata.txt.
2. Train another logistic regression using built-in PyTorch function and test it using testing datasets.
3. Compare the testing accuracy between your own functions and the built-in PyTorch function.
4. Tips: Look up online resources on how to 'Build Logistic Regression from Scratch'

3 Submission

1. Your source code - ipython notebook.
2. A report consisting of a table that comparing the accuracy between your own function and built-in PyTorch function. Did you get the same results? If no, state why. Where could be go wrong?
3. Deadline - 3 Nov 2023, 2359

4 Evaluation

1. 6% - Each function implemented correctly get 1% - total 6 functions
2. 4% - All code run correctly. Results using your own functions and built-in functions match. If there is a mismatch, please specify the reason for the discrepancy. Evaluation will consider your comprehension of where the error might have occurred.