CIS831/CIS532 - Assignment 2

Note

You are required to use PyTorch to create and train the model in this assignment. Study the code examples for Logistic Regression in PyTorch, which are posted on Canvas. Those will be useful in writing your code and answering the questions in this assignment.

Logistic Regression Exercise

In this exercise, you will train and evaluate a logistic regression model. As you did for linear regression, you will analyze learning curves to gain insights into how the performance (in this case, measured using the accuracy metric) varies with the learning rate and the number of iterations used in the gradient descent algorithm.

The data you will use to train the logistic regression model in this assignment represents Android apps that are labeled as benign or malicious. The task is to train models that can classify apps as malicious apps (or malware) or benign, with the goal of preventing malware apps from entering the market. It is estimated that the ratio of malware to benign apps is close to 1:100. To keep things simpler, we provide train and test datasets with 1:1 malware to benign ratio (as opposed to the 1:100 ratio).

Dataset

These datasets are provided as csv files on Canvas: (`AndroidAppsTrainSmall.csv` and `AndroidAppsTestSmall.csv`). Each app in the train and test files is represented using 471 binary (0/1) features, which denote the presence/absence of various permissions, intent actions, discriminative APIs, obfuscation signatures, and native code signatures. For simplicity sake, you can assume that the names of the features are $f0, f2, \cdots, f470$. The last column of an instance/app ($f471$) corresponds to the class label y, which takes values 0 (benign) or 1 (malware).

Tasks/Questions

1. Load the training and test Android datasets. You can read the corresponding csv files as a DataFrame using `pandas.read_csv`.

2. Create a logistic regression model that optimizes the cost/loss given by the `binary cross-entropy loss`, or `BCELoss` (the cost function we used for Logistic Regression in Lecture 6). The model should have one layer, specifically the output layer with one sigmoid unit, and should take as input instances x representing the app features.

3. Use the `accuracy` as the performance metric (accuracy is defined as the number of correctly classified instances divided by the number of all instances).

4. Plot learning curves that show the variation of the loss and also variation of the accuracy metric with the number of iterations, for both training and test datasets, for the 2 values of the learning rate (a larger value and a smaller value).

5. Compare the accuracy on the training data with the accuracy on the test data. Did you see any major difference between the two learning rates that you used? How many iterations were needed for convergence?

What to submit

Please submit a Jupiter Notebook containing your code and answers to the questions above. If you use Google Colaboratory, please export your code as a notebooks and upload the files on Canvas, so that we have a timestamp for your submission (links to the Colaboratory notebooks, in addition the Jupyter Notebook files, are also be useful when grading).