Assignment (2)

Artificial Intelligence and Machine Learning (24-787:B)

Due date: 01/10/2020 @ 11:59 pm EST
All the files must be put into one directory **andrewID-PHW2/**. Convert the jupyter notebook to a pdf file. Ensure that the submitted notebooks have been run and the cell outputs are visible - **Hint:** Restart and Run All option in the Kernel menu. Zip the directory **andrewID-PHW2/** and submit the zip on canvas. You can refer to Numpy documentation while working on this assignment. Any deviations from the submission structure shown below would attract penalty to the assignment score. Please use Piazza for any questions on the assignment.

```
andrewID-PHW2/
                q1.ipynb
                q1.pdf
                q2.ipynb
                q2.pdf
                q3.ipynb
                q3.pdf
                output-data/
```

Submission file structure

PROBLEM 1

Introduction to Numpy **[20 points]**

This question introduces numpy and some of the most powerful features it offers. Remember that array indexing in python starts from 0 unlike MATLAB which starts from 1.

♋Set the random seed to 24787 using numpy random seed and create a random integer array **a** of size (3,4,4) with values between 0 and 7, both included. To make visualization simple, you can think of the dimensions of a 3darray as (depth, rows, columns). Print **a** and its shape. Locate the locations of all the fours in the array **a** and print out the row and column indices (0 being the first row).

♋Create an array **b** of size (3,8,8) by repeating matrix **a** using the tile function. Example shown in the fig 1.

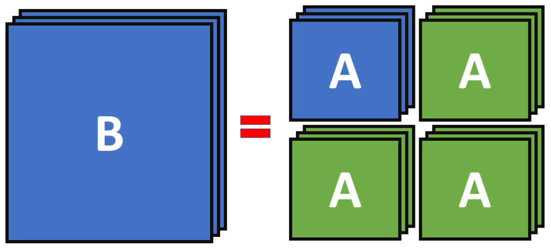

Fig. 1: Construct b matrix by repeating matrix a

♋Calculate and store the sum of matrix **b** along the depth in **c**. Use standard Numpy functions only, and observe the shape of **c**. Print **c** and its shape.

♋Set the seed back to 24787 and create two new arrays **a** and **b** of size (1000,1000). Create a function matmul which takes in two matrices as input and performs a matrix multiplication each time taking a dot product of a row from matrix 1 and the corresponding column from matrix 2 for each element in the product matrix. Run your function on the matrices created earlier and print the time taken to calculate the product. Repeat the same experiment, this time using the matrix product operator "@" and time the execution. Show the correctness of your implementation of the product function by comparing with the matrix obtained by the product operator. Also explain in two lines why the inbuilt implementation is faster than the function you wrote.

PROBLEM 2

Python Programming- Linear Regression using Gradient Descent **[40 points]**

The goal of this exercise is to find the weights associated with the linear regression algorithm using Gradient Descent that you found using Normal Equation in the previous homework. Write a program that performs Linear Regression using Gradient Descent. Find the b_0 and b_1 for the linear regression equation $y = b_0 + b_1 x$ that fits (x,y).
We suggest you to implement two functions: First function which computes the cost (loss) and gradients (using least squares loss) with the arguments (training data, b_0 and b_1), and the second function which updates the weights (using the gradient descent parameter update) with the arguments (training data, initial b_0, initial b_1 , learning rate, number of iterations). Ideally, second function calls the first function. Use the file **data.npy**. The first column corresponds to **x** and the second column contains **y** values.
You can use **numpy**, **matplotlib** libraries in this problem.

♋Find the b_0 and b_1. Plot (x,y) data and the fitted line in two different colors. (Hint= set learning rate to 0.0001).

♋Plot the cost (loss) vs. iterations for learning rates (0.0001, 0.0005, 0.001).

♋What in your opinion is the best learning rate and why? Use the next cell in the jupyter notebook to write down your answer.

♋What is the approximate upper bound of learning rate (i.e before divergence occurs)? Plot the cost function within the Jupyter cell to show your evidence of divergence occurring. Write down your conclusion in a new cell.

♋Use Mini-Batch Gradient Descent with batch size 20 and plot the cost per mini-batch. Choose an appropriate learning rate. What is your observation on this plot? Write it down in a new cell.

PROBLEM 3

Logistic regression **[40 points]**

The goal of this exercise is to find the weights associated with the logistic regression algorithm that classifies data into two classes (categories). Please use the jupyter notebook code template provided in conjunction with the following text to solve this question. The following convention should be used while answering the questions in the notebook: Input array X, Label array Y, Predicted array $\widehat{Y}$, Parameters θ, Number of Datapoints N, i^{th} element of A being a_i.
You can use **numpy**, **matplotlib** and **pandas** libraries in this problem.

♋Load up the csv files **class0-input.csv**, **class1-input.csv** and **labels.csv** from the directory **q3-data/** and store them in arrays **X** and **labels** respectively. Check the size of **labels**.

♋Visualize the data and implement the functions as directed in the jupyter notebook.

♋Calculate the weights by calling the functions you previously defined in the '**main()**' function.

♋Then calculate the accuracy of the predicted labels from weights calculated.

♋Visualize the misclassification as directed in the jupyter notebook.

♋Now use **sklearn**'s logistic regression function to fit a model and write down the accuracy.