UNIVERSITY of GUELPH

School of Engineering

ENGG*6500: Introduction to Machine Learning

Fall 2021

Assignment 2

Due: Friday November 12, 2021

Notes:

- The questions require thought, but do not require long answers so try to be concise.
- You can consult with your colleagues or check for solutions so long as what is finally submitted is your individual interpretation and not a copy paste.
- For problems that require programming, please include in your submission a copy of your code (with comments) and any figures that you were asked to plot.
- You should write your own code and not copy from any resource on the internet. If you consult any resource online before writing your own code, you must cite the source, add a comment to your code with a link to the source you got it from.
- All students must submit an electronic PDF version for the report and the related *.ipynb files.
- There will be a 5% penalty for submissions which are merely scanned versions of handwritten solutions.
- Submit the assignment pdf file and the related *.ipynb files in Assignment 2 dropbox of Courselink, before due date. (20% deduction per day for delayed submission).

Include a cover page indicating your name, title of work, course, instructor's name and date. Assignments will be judged on the basis of visual appearance, the grammatical correctness and quality of writing, and the visual appearance and readability of any graphics, as well as their contents. Please make sure that the text of your report is well-structured, using paragraphs, full sentences, and other features of a well-written presentation. Use itemized lists of points where appropriate. Text font size should be 12 point.

I. Perceptron (30 Marks)

For this part you will write code to implement perceptrons and the perceptron learning algorithm.

Perceptrons: You will train 10 perceptrons that will, as a group, learn to classify the handwritten digits in the MNIST dataset. See the class slides for details of the perceptron architecture and perceptron learning algorithm. Each perceptron will have 785 inputs and one output. Each perceptron target is one of the 10 digits.

Preprocessing: Scale each data value to be between 0 and 1 (i.e., divide each value by 255, which is the maximum value in the original data). This will help keep the weights from getting too large. Randomly shuffle the order of the training data.

Training: Train the perceptrons with three different learning rates: η = 0:01; 0:1; and 1:0:

For each learning rate:

1. Choose small random initial weights, $\omega_i \in [-0.05, 0.05]$. Recall that the bias unit is always set to 1, and the bias weight is treated like any other weight.
2. Repeat for 50 epochs: cycle through the training data, changing the weights (according to the perceptron learning rule) after processing each training example x^k by each perceptron as follows:

   (a) For each perceptron, compute $\omega\ x^k\ y^k$ and t^k at each output unit

   Where:

$$t^k = \begin{cases} 1\ if\ output\ unit\ is\ correct \\ 0\ otherwise \end{cases}$$

$$y^k = \begin{cases} 1\ if\ \omega\ x^k > 0 \\ 0\ otherwise \end{cases}$$

   Update all weights in each perceptron:

$$\omega_i \leftarrow \omega_i + \eta\,(t^k - y^k)\,x_i^k$$

   (Note that this means that for some output units $(t^k - y^k)$ could be zero, and thus the weights to that output unit would not be updated. That's okay!)

   (b) After each epoch (one cycle through training data), compute accuracy on training and test set (for plot), without changing weights. The output with the highest value of $\omega\ x^k$ is the prediction for this training example. For example, if the output corresponding to the digit '7' has the highest value, the prediction of the group of perceptrons is '7'. This

prediction is used for computing accuracy. Accuracy on a set of examples (e.g., training or test) is the fraction of correct classifications (prediction = true class) on that set.

The report should include the following:

1. For each learning rate:

   (a) Plot of accuracy (fraction of correct classifications) on the training and test set at each epoch (including epoch 0), along with comments as to whether you are seeing either oscillations or overfitting.

   (b) Confusion matrix on the test set, after training has been completed.

   (c) Short discussion of confusion matrix: which digits are classified most accurately, and which digits tend to be confused with one another?

2. Short discussion comparing results of different learning rates. Did you see any difference in the results when using different learning rates?

II. Regression (30 Marks)

A city is planning to improve the performance of parking lots. The city provides the occupancy of each parking lot for around three and a half years. Write a Python program to obtain the regression equations that represent the parking lot occupancy. Advise the city based on the regression equations regarding the enlargement of the parking lots. Rank the parking lots necessity for enlargement from the most to the least urgent. Assume that the parking fees of all the parking lots are equal. The file Parking_lots_dataset.csv can be downloaded from CourseLink.

III. K-Means Clustering (40 Marks)

In this programming assignment you will implement a “K-Means Clustering” heuristic, test your program on real datasets under various settings and interpret your results.

The code should be properly commented, and you will be required to submit the source code. You should write your own code and not copy from any resource on the internet. If you consult any resource online before writing your own code, you must cite the source, add a comment to your code with a link to the source you got it from.

In addition to the source code, you will submit a detailed report that describes the simulations and explains the results using figures and mathematical interpretations. Your report must address all the questions asked in this assignment. The report should be presented in a complete and professional manner, with appropriate labels on items and comments which demonstrate understanding of the results.

Your grade on the programming assignment will be based on the correctness of the code as well as the quality of the report. Since our evaluation of your code is based on its results as presented in your report, correct code is very important.

Dataset: The used dataset is the well-known MNIST (Mixed National Institute of Standards) database of handwritten digits which contains grayscale images of size 28 × 28 which can be represented as vectors of size 784. The dataset consists of N = 60000 images that you will cluster using K-means.
http://yann.lecun.com/exdb/mnist/

Implementing K-means: The K-means algorithm is a method to automatically cluster similar data examples together. Concretely, you are given a training set $X(1), \ldots, X(m)$ (where $X(i) \in \mathbb{R}^n$), and want to group the data into a few cohesive "clusters". The intuition behind K-means is an iterative procedure that starts by guessing the initial centroids, and then refines this guess by repeatedly assigning examples to their closest centroids and then recomputing the centroids based on the assignments.

1. Write a program K-means to implement K-means clustering. The function accepts two inputs: a list of N data vectors to be clustered, and the number of clusters, K. The program outputs three quantities: a list of N, group assignment indices $(c_1, c_2, \ldots, c_N)$ indicating the association of each data vector to a specific group, a list of K group representative vectors $(\mu_1, \ldots, \mu_k)$ and the value of J^{clust} after each iteration of the algorithm until convergence or termination.
Initialization and Termination: Initialize k-means by randomly assigning the data points to K group, i.e., randomly selecting each c_i to be an integer between 1 and K. Alternatively, you can

also randomly assign K data points as the K group representatives. Explicitly mention your initialization technique in the report. You can follow any of the termination rules suggested in the reading associated with the assignment but mention in your report what criterion you used.

2. Let K = 20. Implement k-means on the MNIST dataset, starting with a random assignment of the vectors to clusters, and repeating the experiment P = 30 times. Save the output of the algorithm for each of the 30 runs. Consider the two runs that produce the maximum and minimum value of J^{clust} upon convergence. For each of these two runs:

(a) Plot the value of J^{clust} as a function of the number of iterations. Comment.

(b) Visualize the K group representatives as images. Interpret the group representatives (if which digits, they represent, or if they represent something else) and compare them for these two runs. Discuss your results.

(c) Find the 10 nearest data points to each of the K group representatives. Manually list what digits they represent (by eye estimation). In your opinion, how many of these 10 images match the corresponding group representative and how many of them are misclassified? Generate a table and interpret your results. Plot a few of the correctly classified as well as misclassified data points as images, alongside the group representatives (also represented as images).

3. Repeat part 2 for K = 10 and P = 20. Compare with the previous results.

4. Repeat part 2 for K = 5 and P = 10. Compare with the previous results.