COMP 3105 Introduction to Machine Learning
Assignment 4

Instructor: Junfeng Wen (junfeng.wen [AT] carleton.ca)

Fall 2023
School of Computer Science
Carleton University

Deadline: 11:59 pm, Sunday, Nov. 26, 2023

Instruction: **Submit the following three files** to Brightspace for marking

- A Python file `A4codes.py` that includes all your implementations of the required functions
- A pip requirements file named `requirements.txt` that specifies the running environment including a list of Python libraries/packages and their versions required to run your codes.
- A PDF file `A4report.pdf` that includes all your answers to the written questions. It should also specify your team members (names and student IDs). Please clearly specify question/sub-question numbers in your submitted PDF report so TAs can see which question you are answering.

Do not submit a compressed file, or it may result in a mark deduction. We recommend trying your code using Colab or Anaconda/Virtualenv before submission.

Rubrics: This assignment is worth 15% of the final grade. Your codes and report will be evaluated based on their scientific qualities including but not limited to: Are the implementations correct? Is the analysis rigorous and thorough? Are the codes easily understandable (with comments)? Is the report well-organized and clear?

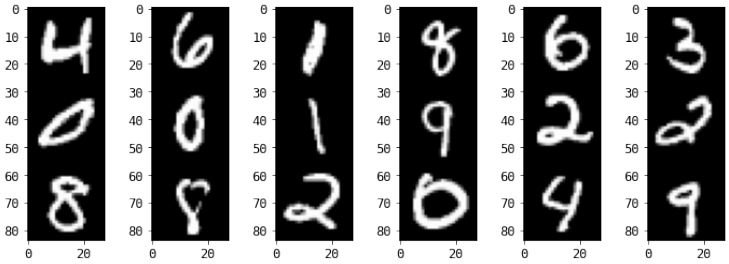
Policies:

- You can finish this assignment in groups of two. All members of a group will receive the same mark when the workload is shared.
- You may consult others (classmates/TAs/LLMs) about general ideas but don't share codes/answers. Please specify in the PDF file any individuals or programs (e.g., ChatGPT) you consult for the assignment. If you use large language models (LLMs), clearly show us how you use it. Any group found to cheat or violating this policy will receive a score of 0 for this assignment.
- Remember that you have **three** excused days *throughout the term* (rounded up to the nearest day), after which no late submission will be accepted.
- Specifically for this assignment, you can use **any** Python libraries you want, as long as the program can complete within a reasonable time and space limit. **However, you must not use any additional/external data sources**.

Question 0: The Task

The goal of this assignment is to mimic a real-world machine learning scenario where you are given a limited dataset to learn a classifier so that it can generalize to unseen test data.

The dataset consists of hand-written digit images taken from the MNIST dataset, and the task is to retrieve the digit label *at a specific location*. Fig. 1 provides several example images. Each image has three sub-images: the top image serves as a pointer to the middle or bottom images. If the top image is a digit in $\{0, 1, 2, 3, 4\}$, then the label of the whole image is the digit in the middle image (e.g., the first example in Fig. 1 with the label "0"), otherwise, the top image is a digit in $\{5, 6, 7, 8, 9\}$ and the label of the whole image is the digit in the bottom image (e.g., the second example in Fig. 1 with the label "8").



Label 0 Label 8 Label 1 Label 0 Label 4 Label 2

Figure 1: Sample images

You can use any algorithm (even those that have not been taught in this course). You can even try ensemble methods (combining multiple models) or apply any pre-processing steps before training. You may also try different regularizations or controlling model complexities.

The constraints are

1. The implementation must be your own, although you can use **any** Python library, including ML libraries such as scikit-learn, PyTorch and TensorFlow. It is highly recommended to test your code in Google Colab or Anaconda environment. Marks will be deducted if we cannot run your code.
2. Your program can run within a reasonable time and space limit. You should aim for a running time that is within 30 minutes when training in Google Colab and this is a soft constraint.
3. **You must not use any additional/external data sources (including the original MNIST dataset).**

Note: If you want to try out different deep learning models, Colab offers free GPU access that may speed up your method (Runtime → Change runtime type)

In the zip file, we provided a training dataset (`A4train.csv`) and a validation dataset (`A4val.csv`). The first column of each CSV file includes the labels $\mathbf{y}$ and the rest of the columns are features X. They will be similar to the new training and test datasets used for evaluating your method (see Question 3 for more detail). You should use the training dataset for training and the validation dataset to check the generalization capability of your model (as this will be how we call your functions).

Be creative :)

Question 1 (5%) Learning and Classifying

Implement a Python function

```
model = learn(X, y)
```

that takes an $n \times d$ input matrix `X` and an $n \times 1$ label vector `y` (where each entry is an integer between 0 and 9, inclusive, representing the image labels), and returns your `model` of any type, as long as it can be used to classify new data (see next).

Implement a Python function

```
yhat = classify(Xtest, model)
```

that takes an $m \times d$ input matrix `Xtest` and a `model` learned by your algorithm, and returns an $m \times 1$ prediction vector `yhat`.

The matrices and vectors are represented as NumPy arrays.

The functions must be able to handle arbitrary $n > 0$ and $m > 0$. Note that in this assignment, $d = 28 \times 28 \times 3 = 2352$ for a vectorized image. You can use the following function to visualize one image:

```
def plotImg(x):
    img = x.reshape((84, 28))
    plt.imshow(img, cmap='gray')
    plt.show()
    return
```

Question 2 (5%) Explanation

In the PDF file, explain your learning algorithm and more importantly, why it is designed in this way. The report should be at least 2 pages, but no longer than 8 pages in length.

Your report will be evaluated mainly on how well you justify your design choices. In general, each modelling step should be justified by reasons and/or empirical evidence. For example, if you tried several different algorithms, how did you decide to submit one algorithm over the others? The same applies to other modelling strategies such as optimization schemes, hyper-parameter selection, pre-processing and/or post-processing steps. One possible supporting evidence might be tables of accuracies with different algorithm configurations. Note that these are only guidelines and you do not have to follow them strictly. Feel free to discuss any findings that you feel are interesting or relevant. We will also take into account the overall quality of your report.

Question 3 (5%) Evaluation

For this question, you don't need to submit/write an answer. We will apply your learning method to a **different** training set and evaluate the performance of the learned model on a hold-out test dataset. Your method will be evaluated based on

- Classification accuracy (main criterion)
- Time-complexity: the overall run-time of your algorithm
- Space-complexity: whether your algorithm can finish within a reasonable memory limit

Hint: In order to improve the accuracy, you may want to take a look at the misclassified examples in the validation dataset and think about why they are misclassified.