# CMPUT 366, Winter 2021
# Assignment #3

Due: Monday, March 29, 2021, 11:59pm
Total points: 68

For this assignment use the following consultation model:

1. you can discuss assignment questions and exchange ideas with other *current* CMPUT 366 students;
2. you must list all members of the discussion in your solution;
3. you may **not** share/exchange/discuss written material and/or code;
4. you must write up your solutions individually;
5. you must fully understand and be able to explain your solution in any amount of detail as requested by the instructor and/or the TAs.

Anything that you use in your work and that is not your own creation must be properly cited by listing the original source. Failing to cite others' work is plagiarism and will be dealt with as an academic offence.

---

**First name:** Weixi

**Last name:** Cheng

**CCID:** weixi@ualberta.ca

**Collaborators:** _____

1. **(Neural Networks)** The MNIST dataset is a set of images of handwritten digits labelled by their actual digit. We will operate on two versions of this dataset: one with the images shifted 2 pixels to the upper-left, and one with the images shifted 2 pixels to the bottom-right.

   This question requires the use of TensorFlow. You may need to install Tensorflow using the following command:

   ```
   pip3 install tensorflow
   ```

   (a) **[10 points]** Implement a fully-connected feed-forward neural network for classifying MNIST images according to the digit that they represent by editing the `mlp2` function in the provided `cnn.py` file.

   The network should have two hidden layers: one with 128 rectified linear ('relu') units, and one with 64 rectified linear units. The output should be a fully-connected layer of 10 units with the softmax activation. The `cnn.py` file contains an example implementation of a network with a single hidden layer in the `mlp1` function that you may template from. The `main` function will test your program for you; you may add any tests that you like. It will also create a file called `examples.png` that contains examples images from the two test sets.

   The function should train the network using the training features `train_x` and labels `train_y`, and then evaluate the accuracy of the trained network on two different test sets: `test1_x`, `test1_y`, and `test2_x`, `test2_y`. This is demonstrated in `mlp1`.

   We will run your code by importing the `cnn` module and calling `mlp2`, so it is important that your code follow these naming conventions.

   Submit all of your code for this question and question 1 (including provided boilerplate files) in a single zip file.

   (b) **[30 points]** Implement a convolutional neural network for classifying MNIST images by editing the `cnn` function in the provided `cnn.py` file.

   The network should have the following architecture:

   - A layer of 32 convolutional units with a kernel size of $5 \times 5$ and a stride of $1, 1$.
   - A max-pooling layer with a pool size of $2 \times 2$ and a stride of $2, 2$.
   - A layer of 64 convolutional units with a kernel size of $5 \times 5$ and the default stride.
   - A max-pooling layer with a pool size of $2 \times 2$ and the default stride.
   - A `Flatten` layer (to reshape the image from a 2D matrix into a single long vector)
   - A layer of 512 fully-connected relu units
   - A layer of 10 fully-connected softmax units (the output layer)

   Submit all of your code for this question and question 1 (including provided boilerplate files) in a single zip file.

   (c) **[2 points]** What was the accuracy of your trained 2-hidden-layer feedforward network on the two test sets?

   ```
   Evaluating MLP2 on test set 1
   313/313 [==============================] - 0s 563us/step - loss: 0.0830 - accuracy: 0.9755
   Evaluating MLP2 on test set 2
   313/313 [==============================] - 0s 570us/step - loss: 2.3324 - accuracy: 0.5668
   ```

   (d) **[2 points]** What was the accuracy of your trained convolutional neural network on the two test sets?

   ```
   Evaluating CNN on test set 1
   313/313 [==============================] - 2s 5ms/step - loss: 0.0465 - accuracy: 0.9880
   Evaluating CNN on test set 2
   313/313 [==============================] - 1s 4ms/step - loss: 1.1133 - accuracy: 0.8257
   ```

   (e) **[10 points]** Did one of your implementations perform substantially better on one of the test sets than the other implementation did? If so, why? If not, why not?

convolutional neural network implementations perform substantially better on one of the test sets than the 2-hidden-layer feedforward network implementations did.

Because CNNs are effective in reducing the number of weights which are need to be computing.

And CNNs can be efficient learning via : ① Sparse interactions ② Parameter sharing ③ Equivariant representations

CNNs also introduce two new operations : ① Convolutions ② Pooling

2. **(Bayesian Learning)**

Suppose that you have three models $(\theta_1, \theta_2, \theta_3)$ of changes in the price of a single stock. You know that one of these models is the true model. Each model gives the probability that tomorrow's price will be higher than today's price $(y_{t+1})$, based on whether today's price was higher than the price the day before $(y_t)$. So you can make money on average by buying the stock when $p(y_{t+1} \mid y_t, \theta^*) > .5$ and selling when $p(y_{t+1} \mid y_t, \theta^*) < .5$, where $\theta^*$ is the true model.

Your prior belief is that $\theta_1$ and $\theta_2$ are equally likely to be true, and $\theta_3$ is three times more likely than $\theta_1$ to be true.

You have a dataset $D$ of past observations, and you have computed that

$$\Pr(\theta \mid D) = \frac{\Pr(D \mid \theta)\Pr(\theta)}{\Pr(D)}$$

$$p(D \mid \theta_1) = .00084$$
$$p(D \mid \theta_2) = .00105$$
$$p(D \mid \theta_3) = .00007.$$

$$\begin{cases} \theta_3 = 3\theta_1 = 3\theta_2 \\ \theta_1 + \theta_2 + \theta_3 = 1 \end{cases}$$

3. **[6 points]** What are the posterior probabilities of each model being the true model?

From the question we have $\theta_1 = \theta_2$, $\theta_3 = 3\theta_1 \Rightarrow \theta_3 = 3\theta_1 = 3\theta_2$. $\Rightarrow \begin{cases} \theta_1 = 0.2 \\ \theta_2 = 0.2 \\ \theta_3 = 0.6 \end{cases}$

$$P(D) = P(D \mid \theta_1)P(\theta_1) + P(D \mid \theta_2)P(\theta_2) + P(D \mid \theta_3)P(\theta_3)$$
$$= 0.00084 \times 0.2 + 0.00105 \times 0.2 + 0.00007 \times 0.6 = 0.00042$$

① $P(\theta_1 \mid D) = \frac{P(D \mid \theta_1)P(\theta_1)}{P(D)} = 0.4$

② $P(\theta_2 \mid D) = \frac{P(D \mid \theta_2)P(\theta_2)}{P(D)} = 0.5$

③ $P(\theta_3 \mid D) = \frac{P(D \mid \theta_3)P(\theta_3)}{P(D)} = 0.1$

4. **[4 points]** Now suppose that you have run each model, and they make the following predictions:

$$\begin{cases} p(y_{t+1} \mid y_t, \theta_1) = .75 \\ p(y_{t+1} \mid y_t, \theta_2) = .4 \\ p(y_{t+1} \mid y_t, \theta_3) = .6. \end{cases}$$

What is the maximum a posterior estimate for $p(y_{t+1} \mid y_t)$? Based on the MAP estimate, would you be better off buying or selling?

Since $P(\theta_2 \mid D) = 0.5 > P(\theta_1 \mid D) = 0.4 > P(\theta_3 \mid D) = 0.1$

The maximum a posterior estimate for $P(y_{t+1} \mid y_t)$ is $P(\theta_2 \mid D) = 0.5$.

Since $P(y_{t+1} \mid y_t, \theta_2) = 0.4 < P(\theta_2 \mid D) = 0.5$

Based on the MAP estimate, it would be better off selling.

5. **[4 points]**

What is the estimate according to the posterior predictive distribution for $p(y_{t+1} \mid y_t)$? (I.e., using model averaging.) Based on the PPD, would you be better off buying or selling?

$$P(Y \mid D) = \sum_\theta P(Y \mid \theta)P(\theta \mid D)$$
$$= P(y_{t+1} \mid y_t, \theta_1)P(\theta_1 \mid D) + P(y_{t+1} \mid y_t, \theta_2)P(\theta_2 \mid D) + P(y_{t+1} \mid y_t, \theta_3)P(\theta_3 \mid D)$$
$$= 0.75 \times 0.4 + 0.4 \times 0.5 + 0.6 \times 0.1$$
$$= 0.56 > 0.5$$

∴ Based on the PPD, it would be better off buying.

## Submission

The assignment you downloaded from eClass is a single ZIP archive which includes this document as a PDF *and* its LaTeX source as well as Python files needed for the coding questions. You are to unzip the archive into an empty directory, work on the problems and then zip the directory into a new single ZIP archive for submission.

Each assignment is to be submitted electronically via eClass by the due date. **Your submission must be a single ZIP file containing**:

1. a single PDF file with your answers;
2. file(s) with your Python code.

To generate the PDF file with your answers you can do any of the following:

- insert your answers into the provided LaTeX source file between `\begin{answer}` and `\end{answer}`. Then run the source through LaTeX to produce a PDF file;
- print out the provided PDF file and *legibly* write your answers in the blank spaces under each question. Make sure you write as legibly as possible for we cannot give you any points if we cannot read your hand-writing. Then scan the pages and include the scan in your ZIP submission to be uploaded on eClass;
- use your favourite text processor and type up your answers there. Make sure you number your answers in the same way as the questions are numbered in this assignment.