**ELEC 425**
Machine Learning and Deep Learning
**Lab 1 - Week 2**

Welcome to ELEC 425 Lab Session where you will learn machine learning by practicing. For this lab, you are expected to form your own team with 2-3 students to work together on the lab questions. A team should not have more than 3 students. Working on your own is fine, but we encourage collaboration and discussion to solve the lab questions.

If your team can solve the lab questions without further help, you do not have to attend the lab session. Otherwise, we encourage you to attend the lab session in person, and TAs can help you there. If you go the lab session, you can either show your results to TAs to get your marks (3 marks) in the lab session, or you can choose to submit your solutions later on onQ.

Every team has until **9pm Thursday September 16th** to submit the solutions on onQ. You can submit any time—before, during, or after the lab session (**3:30-5:30pm Tuesday September 14th**). Each team only needs to make one shared submission. If you have received your marks in the in-person lab, you do not have to submit your solutions on onQ.

In your submission, include a file named **students.txt**, which contains all students' names and student numbers of your team. Zip all your code, including the code we provided to you and that you modified, plus the students.txt file, to one zip file named as follows:

**Fisrtname-Lastname1_Firstname-Lastname2_Firstname-Lastname2.zip**

For example, if your team has Joe Trudeau, Frank Cook, and Peter Martin as the members, you should name the file as Joe-Trudeau_Frank-Cook_Peter-Martin.zip.

## 1 Introduction

Matlab is a popular mathematical software that owns reputation for fast matrix computation. It's a perfect tool for mathematician, statistician and data scientist to quickly develop and verify their algorithms and models. Matlab provides easy-to-use interpreter functions which encapsulate complex CPU optimized matrix computation libraries, and many of its linear algebra functions stay as state-of-art algorithms for decades, so you do not need to worry about parallel processing, memory management etc.

Machine Learning algorithms often consist of matrix computation. We consider Matlab as a good choice for machine learning beginners.

## 2 Getting Started in Matlab

Since we are using Matlab for most of our experiments and homework, this lab session will first ensure that everyone can successfully deploy the Matlab environment on your working machine. Please go through the following steps below to set up Matlab on lab computers or your own laptop.

1. Visit matlab website for queensu authorization:
   https://www.mathworks.com/academia/tah-portal/queens-university-40561113.html

2. It needs your NetID and password to download.
3. You need to register a MathWorks account, if you have not yet.
4. Follow the installer and complete your installation.
5. Start Matlab!

## 3 Tutorial on Basic Matlab Operations

### 3.1 Matrix Basics

Create a 2*2 matrix and try the following code. Note that index in matlab start from 1, not 0. The following lines show some matrix operations that you may often use in the course. Try them in Command Window.

```
A = [1, 2; 3, 4]  %%  ',' means same row, ';' change to a new row
A(2, 1)           %%  () means indexing(get elements with coordinates)
A(1, :), A(:, 1)  %%  Slice, get rows and cols
B = [-1; -2]
B'                %%  transpose
size(B), length(B) %%  get matrix size
C = [A B]         %%  concatenate two matrix in column
C = [A; B']       %%  concatenate in row
diag(A)           %%  diagonal part of A
triu(A), tril(A)   %%  triangle part of A, u for upper, l for lower
D = reshape(A, 1, 4)  %% reshape A to size [1, 4]
repmat(A, 1, 2)   %%  repeat matrix A 1 times in row, 2 times in col
repelem(B, 1, 2)  %%  repeat elements several times along the given axis
doc repelem       %% open documentation of function 'repelem'
```

### 3.2 Operation Basics

```
A = [1, 2; 3, 4]
B = [-1, -2; -3, -4]
c = [1, 2]'
A + B             %% matrix of the same size, add, minus is also the same
A * c             %% matrix dot product
A .* B            %% element wise product
A + c             %% matrix add with different shape, broadcast C
A .* c            %% element wise product with different shape, broadcast C
d = A * c
A \ d             %% famous matlab left division, solve A^-1 d
c.^2, sqrt(c)     %% element wise sqaure/ square root
sum(c), prod(c)   %% sum and product
mean(A, 2)        %% mean over particular dimension
max(A), min(A)    %% maximum and minimum, along certain dim also supported
```

```
maxk(A, 2)          %% top 2 largest element
find(c == 2)        %% find elements, return index(first? last? all?)
```

### 3.3 String and Symbol Basics

```
str = "Hello"       %% create a string
str == "Hi"         %% compare(be sure to use "", '' means char array)
str == "Hello"

func = @mean        %% function handle
                    %% (make it easy to pass function as parameter)
func([1, 2])        %% use the handle

                    %% The following is a print
fprintf("You gain %d pts reading tutorial, %.2f pts coding knn below", 1, 2)
disp("Hi")          %% display a message
```

### 3.4 Coding Basics

For-loop and if-else selection:

```
tic                                         %% timer start
for i = 1:3                                 %% for loop
    if (i == 1)                              %% if elseif else
        fprintf("This is the %d st loop\n", i)
    elseif (i == 2)
        fprintf("This is the %d nd loop\n", i)
    else
        fprintf("This is the %d rd loop\n", i)
    end                                      %% end if
end                                          %% end for
toc                                          %% timer stop
```

Let's do a while loop

```
a = 10
while( a < 20 )                              %% while loop
    fprintf('value of a: %d\n', a)           %% you can use break in loop
    a = a + 1;
end
```

### 3.5 Navigation and Drawing

```
clear                 %% clear all variables in the memory
clc                   %% clear the command window
doc plot              %% open documentation of function 'plot'
```

### 3.6 Changing the Working Directory

```
pwd                  %% identify the current directory
dir                  %% list files and directories in the current directory
cd newDir            %% change the current directory to the newDir
```

## 4 Hands-on Practice with K-Nearest-Neighbour Classifier

In this section you will get some hand-on experience on k-nearest neighbor classification. On onQ course page, go to Lab1 under Week2 and download *lab1_data_code.zip* to a directory of your computer; unzip it. Change your working directory to that directory by using the corresponding commands taught in the tutorial.

You will use *a1digits.mat* as your dataset in lab1 here and also in assignment 1. So be sure to know how to deal with it.

### 4.1 Run a Demo to Test your Matlab Installation

1. Open the script *knn_demo.m* by typing *edit knn_demo.m* or *open knn_demo.m.*
2. Press the run button on the editor window, or type the script name 'knn_demo' in command window. If your matlab works fine, you can see we got a 96.15% accuracy already on the test set. Note that accuracy is defined as the number of correctly classified data points divided by the number of total data points in the test set.
3. Read and understand the *knn_demo.m* script.

### 4.2 Training-validation-test Process

1. Open *knn_tvt_incomplete.m.*
2. The code has some parts missing, marked with "The code is incomplete; add one line of code here!". Right above these missing lines, there are several lines of comments telling you what is missing. Please read those comments and finish the missing code.
3. Once finishing all the lines, run the script.
4. Think about the purpose that training, validation, and test set are used for. (We have a very special case here: KNN's training is super simple—just remember all training cases. We use validation set to select K to avoid overfitting.

Lab 1 is a bit light; hope it is a good warm-up for you. After you finish the lab, try to spend some time reading the code and ensure you understand it! **Enjoy the lab questions!** (If you submit your solutions on onQ, you do not need to submit anything for the Tutorial part; you only need to submit your solutions for the KNN part.)