SC4001 Tutorial 3 **Neuron Layers**

1. Train a softmax layer of neurons to perform the following classification, given the inputs $\boldsymbol{x} = (x_1, x_2)$ and target class labels d:

(x_1, x_2)	(0 4)	(-1 3)	(2 3)	(-2 2)	(0 2)	(1 2)	(-1 2)	(-3 1)	(-1 1)
d	A	A	A	B	B	A	B	B	B

(x_1, x_2)	(2 1)	(4 1)	(-2 0)	(1 0)	(3 0)	(-3 -1)	(-2 -1)	(2 -1)	(4 -1)
d	C	C	B	C	C	B	B	C	C

(a) Show one iteration of gradient descent learning at a learning factor 0.05.
Initialize the weights to $\begin{pmatrix} 0.88 & 0.08 & -0.34 \\ 0.68 & -0.39 & -0.19 \end{pmatrix}$ and biases to zero
(b) Find the weights and biases at the convergence of learning
(c) Indicate the probabilities that the network predicts the classes of trained patterns.
(d) Plot the decision boundaries separating the three classes.

2. Read the Linnerud dataset from sklearn.datasets package by using

```
from sklearn.datasets import load_linnerud
```

The Linnerud dataset is a multi-output regression dataset consisting of three exercise (data) and three physiological variables (targets) collected from twenty middle-aged men in a fitness club:
physiological - Weight, Waist and Pulse
exercise - Chins, Situps and Jumps.

Divide the dataset into train and test partitions at 0.75:0.25 ratio and train a perceptron layer to predict physiological data from exercise variables by implementing with
a) direct gradients
b) 'autograd' functions available in pytorch

Remember to Gaussian normalize inputs and scale the output data. You can use preprocessing API in sklearn:

```
from sklearn import preprocessing
```

Draw learning curves and find mean square error and R^2 values of prediction. Which physiological variable can be better predicted by exercise data?

Compare the time-taken for a weight update and the number of epochs required for convergence for (a) and (b).

3. Use mini-batch gradient decent learning to train a softmax layer to classify Iris dataset (https://archive.ics.uci.edu/ml/datasets/Iris). The dataset contains 150 data points. Use 90 data points for training the classifier and the remaining 60 data points for testing. Plot cross-entropies and classification accuracies against epochs for both train and test data. Set learning rate = 0.1, batch size = 16, and number of epochs = 1000.

   You can use the following python commands to load Iris data:

   from **sklearn** import **datasets**
   iris = datasets.load_iris()

   Repeat the classification with batch sizes = 2, 4, 8, 16, 24, 32, and 64, and compare the accuracies and the times taken for an epoch at different batch sizes.