Pattern Recognition and Machine Learning
(Winter 2022)
Assignment 7: Neural Networks

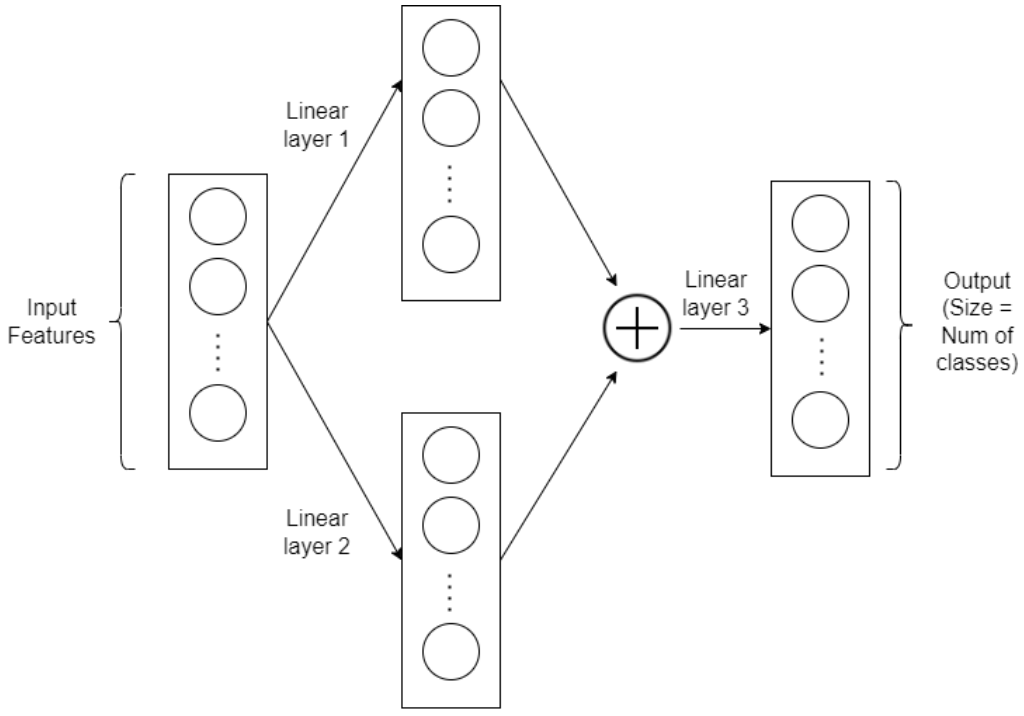
Deadline: **March 20th, 2022 23:59**

Guidelines for submission

1. Perform all tasks in a single colab file.
2. Create a report regarding the steps followed while performing the given tasks. The report should not include excessive unscaled preprocessing plots.
3. Try to modularize the code for readability wherever possible
4. Submit colab file [.ipynb] and report [.pdf] on the classroom (without zipping)
5. Submit the [.py] file on the floated form for the lab
6. Plagiarism will not be tolerated

Question 1. [20]

In this exercise, you need to predict the life of *Abalone* - a kind of shellfish, based on a number of characteristics (sex, length, diameter, height, weights in different forms, etc.). Model it as a classification problem to predict the class (based on the number of rings). The dataset is available here. You need to use the PyTorch library to create a neural network with the following specification, split the data and find out the accuracy on the test set after training:



Decide on the hidden layer size on your own. Use sigmoid activation for the output layer and tanh for the hidden layers. The '+' Symbol represents the addition of the outputs of the 2 branches.

Note: You can refer to the Colab file shared on the classroom (link) for sample code.
(It is a shared file so refrain from making changes in it.)

Question 2. [80]

You have been given a dataset here. It consists of different characteristics of dry beans (consider only: *area*, *perimeter*, *axes lengths*, *eccentricity*, *roundness, aspect ratio, and convex area - 7 features*). You need to perform classification into different varieties (*Cali*, *Bombay*, *Barbunya*, etc.). For this classification, you need to use a multi-layer perceptron.

a. Preprocess & visualize the data. Create train, val, and test splits but take into consideration the class distribution (Hint: Look up stratified splits). ~ **[5]**
b. Implement a multi-layer perceptron from scratch. This would include the following ~**[40]**
    i. Write activation functions.
    ii. Forward propagate the input.
    iii. Backward propagate the error.
    iv. Train the network using stochastic gradient descent.
    v. Predict the output for a given test sample and compute the accuracy.
c. Now experiment with different activation functions (at least 3 & to be written from scratch) and comment (in the report) on how the accuracy varies. Create plots to support your arguments. ~**[10]**
d. Experiment with different weight initialization: Random, Zero & Constant. Create plots to support your arguments. ~**[10]**
e. Change the number of hidden nodes and comment upon the training and accuracy. Create plots to support your arguments.~**[10]**
f. Add a provision to save and load weights in the MLP. ~**[5]**

Note: The report should contain detailed explanations and analysis for your observations in parts c,d,e. Just reiterating the code will not fetch you any marks.

Guidelines for the report

1. The report should be to the point. Justify the space you use!
2. Explanations for each task should be included in the report. You should know the '*why*' behind whatever you do.
