

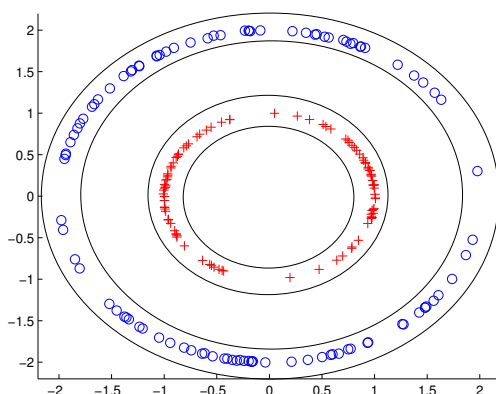
## Homework 2

Lecturer: Inderjit Dhillon

Date Due: Mar 10, 2023

**Keywords:** *Classification, Neural Networks, PyTorch*

1. Suppose we have 2-dimensional data from two classes as shown in the figure below. The ‘o’ class lies in the tube  $\{\mathbf{x} | 3.9 \leq \|\mathbf{x}\|_2^2 \leq 4.1\}$  while the ‘+’ class lies in the tube  $\{\mathbf{x} | 0.9 \leq \|\mathbf{x}\|_2^2 \leq 1.1\}$ .



- (a) Can the classes be separated by a linear surface? Write down the equation of a quadratic surface that completely separates the two classes.
- (b) Recall the probability density function of a multivariate Gaussian with mean  $\boldsymbol{\mu}$  and covariance  $\Sigma$ :

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{p/2} |\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^T \Sigma^{-1}(\mathbf{x} - \boldsymbol{\mu})\right).$$

In class, we saw that by modeling the data in the classes as multivariate Gaussians with different covariance matrices, and then applying Bayes decision theory, we can obtain a quadratic decision surface. Consider modeling the two classes using two multivariate Gaussians  $\Sigma_1 = \sigma_1^2 I$  and  $\Sigma_2 = \sigma_2^2 I$ , and means at the origin. Write down the equation of the resulting decision surface. Can this strategy give good classification accuracy on the above data set?

## 2. CIFAR-10 Classification

For this problem, you will work on the CIFAR-10 dataset. The CIFAR-10 dataset is a widely used benchmark dataset for image classification tasks in the field of computer vision. It consists of 60,000 (50,000 training and 10,000 test) 32x32 color images in 10 classes, with 6,000 (5,000 for training and 1,000 for test) images per class. The 10 classes are: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck.

- (a) Develop a 2-layer Multi-Layer Perceptron (MLP) based neural network using **PyTorch** to classify the CIFAR-10 dataset. The network should have 100 neurons in the first (input) layer with sigmoid non-linearity and 10 neurons in the output layer (one for each class). The loss function to be used is softmax cross-entropy. Use stochastic gradient descent with mini-batch size = 32.

Steps to solve the problem:

- Load the CIFAR-10 dataset into PyTorch using the available dataset loader.
- Create a two-layer MLP neural network in PyTorch with 100 neurons in the input layer and 10 neurons in the output layer.
- Write the forward propagation equations for the network.
- Define the loss function (softmax cross-entropy).
- Train the neural network for 10 epochs using the training dataset with the defined loss function and stochastic gradient descent optimizer.
- Experiment with different learning rates and report learning rate vs validation set accuracy for learning rates in  $\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 10^0\}$ . Also report the optimal learning rate and validation accuracy.
- Evaluate and report the performance of the trained model using the test dataset accuracy.

(Optional) Experiment with the network architectures (e.g., more hidden layers, different activation functions) to improve accuracy.

- (b) Now implement the same neural network and training code from scratch by using only **numpy** (i.e. without relying on any deep learning libraries such as **PyTorch**). For this question, you'll be implementing the backpropagation algorithm and stochastic gradient descent weight updates for training the neural network. Feel free to use **PyTorch** for debugging your numpy implementation (for e.g. to check if you are getting correct gradients).

Steps to solve the problem:

- Preprocess the dataset by normalizing the pixel values between -1 and 1 and reshaping the images into 1D arrays.
- Initialize the weights and biases for the two-layer MLP neural network randomly.
- Implement the forward pass of the neural network by computing the output probabilities of each class for a given input.
- Implement the backward pass of the neural network using the backpropagation algorithm to compute the gradients of the weights and biases.
- Update the weights and biases using the computed gradients and a specified learning rate.
- Train the neural network using the training dataset by iterating over multiple epochs and updating the weights and biases for each batch of inputs.
- Experiment with different learning rates and report learning rate vs validation set accuracy for learning rates in  $\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 10^0\}$ . Also report the optimal learning rate and validation accuracy.
- Evaluate and report the performance of the trained model using the test dataset by computing accuracy.

- (c) Try changing the initializations of your neural network weight matrices to following options: i) random values in  $[-0.1, 0.1]$  ii) random values in  $[-10, 10]$  iii) all zero values. Report your final test accuracies using different initializations and explain your observations.