ECE-GY 6143 / Spring 2020

Homework 2

Please scan and upload your assignments on or before February 20, 2020.

- You are encouraged to discuss ideas with each other; but
- you **must acknowledge** your collaborator, and
- you **must compose your own** writeup and/or code independently.
- We **strongly** encourage answers to theory questions in Latex, and answers to coding questions in Python (Jupyter notebooks).
- Please upload your solutions in the form of a single .pdf or .zip file on NYUClasses.
- Maximum score: 50 points.

---

1. **(10 points)** In class we *derived* the optimal linear predictor for scalar data, and *wrote down* the optimal lnear predictor for vector data (without proof). Here, let us *derive a proof* for the optimal linear predictor in the vector case. Suppose that $\{x_1, x_2, \dots, x_n\}$ denote training data samples, where each $x_i \in \mathbb{R}^d$. Suppose $\{y_1, y_2, \dots, y_n\}$ denote corresponding (scalar) labels.

   a. Show that the mean squared-error loss function for multivariate linear regression can be written in the following form:

   $$MSE(w) = \frac{1}{n} \sum_{i=1}^{n} \|y - Xw\|^2$$

   where X is an $n \times (d+1)$ matrix and where the first column of X is all-ones. What is the dimension of w and y? What do the coordinates of w represent?

   b. Theoretically prove that the optimal linear regression weights are given by:

   $$\hat{w} = (X^T X)^{-1} X^T y?$$

   What algebraic assumptions on X did you make in order to derive the closed form?

2. **(10 points)** In class, we argue that convexity together with smoothness is *a sufficient condition* for minimizing a loss function using gradient descent. Is convexity also a *necessary condition* for gradient descent to successfully train a model? Argue why or why not. You can intuitively explain your answer in words and/or draw a figure in support of your explanation. (Hint: What about $f(x) = x^2 + 3\sin^2 x$?)

3. **(20 points)** The goal of this problem is to implement multivariate linear regression from scratch using gradient descent and validate it.

   a. Implement a function for learning the parameters of a linear model for a given tranining data with user-specified learning rate η and number of epochs T. Note: you *cannot* use existing libraries such as sklearn; you need to write it out yourself.

   b. Validate your algorithm on the glucose dataset discussed in Lecture 2. Confirm that the model obtained by running your code gets similar R^2 values as the one produced by sklearn.

4. **(10 points)** In this lab, we will illustrate the use of multiple linear regression for calibrating robot control. The robot data for the lab is taken from TU Dortmund's Multiple Link Robot Arms Project. We will focus on predicting the current drawn into one of the joints as a function of the robot motion. Such models are essential in predicting the overall robot power consumption.

   a. Read in the data in the attached exp_train.csv file; check that the data that you read actually corresponds to the data in the .csv file. In Python, you can use the commands given at the end of this document.

   b. Create the training data:

      - Labels y: A vector of all the samples in the 'I2' column
      - Data X: A matrix of the data with the columns: ['q2','dq2','eps21', 'eps22', 'eps31', 'eps32','ddq2']

   c. Fit a linear model between X and y (using sklearn, or any other library of your choice). Report the MSE of your model.

   d. Using the linear model that you trained above, report the MSE on the test data contained in the attached exp_test.csv file.

5. **(optional)** How much time (in hours) did you spend working on this assignment?

```
import pandas as pd
names =[
        't',                                   # Time (secs)
        'q1', 'q2', 'q3',                      # Joint angle
        'dq1', 'dq2', 'dq3',                   # Joint velocity
        'I1', 'I2', 'I3',                      # Motor current (A)
        'eps21', 'eps22', 'eps31', 'eps32',    # Strain measurements
        'ddq1', 'ddq2', 'ddq3'                 # Joint accelerations
        ]
df = pd.read_csv('exp_train.csv', header=None,sep=',',
      names=names, index_col=0)
df.head(6)
```