ECE-GY 6143, Spring 2020

Midterm Exam

- Total duration: 90 minutes.
- You **can** use one page as a cheat sheet.
- You **cannot** consult your notes, textbook, your neighbor, or Google.
- Maximum points: 60.

---

1. **(5 points)** Please write down the time at the *start* and *end* of your exam. The difference should not exceed 90 minutes. Please also include your *signature* on the first page; by doing so, you are affirming the NYU Tandon School of Engineering student code of conduct.

2. **(10 points)** This is a slight variant of a homework problem. Let $\{x_1, x_2, \ldots, x_n\}$ be a set of points in d-dimensional space, and let $\{\beta_1, \beta_2, \ldots, \beta_n\}$ be positive numbers that represent weights. Suppose we wish to produce a single point estimate $\mu \in \mathbb{R}^d$ that minimizes the *weighted* squared-error:

$$\beta_1 \|x_1 - \mu\|_2^2 + \beta_2 \|x_2 - \mu\|_2^2 + \ldots + \beta_2 \|x_n - \mu\|_2^2$$

Find a closed form expression for μ and prove that your answer is correct.

3. **(10 points)** Assuming a classification application with n data points in d dimensions where $n = 10d = 10 * 10^8$, *rank* the following algorithms in decreasing order of (i) training times; (ii) testing times. Include a few sentences justifying your reasoning.

   a. Perceptron.

   b. Nearest neighbors.

   c. Kernel perceptron with a polynomial kernel of order 4.

   d. Kernel perceptron with gaussian kernel.

   e. Linear support vector machines.

4. **(15 points)** The following represents python code for an algorithm that attempts to perform linear regression. (a) Identify the algorithm. (b) Explain why this algorithm may not converge as implemented below, and identify the line in the algorithm that makes this happen. (c) Suggest a way to fix this algorithm.

```
def optim_alg(init, steps, grad):
  xs = [init]
  for step in steps:
    xs.append(xs[-1] - step * grad(xs[-1]))
  return xs

def linear_reg_grad(X,y,w):
  row_id = numpy.random.randint(X.shape[1])
  x = X[row_id,:]
  return x.T.dot(x.dot(w)-y)

input_to_optim_alg = lambda w: linear_reg_grad(X,y,w)
learning_rates = [0.001]*300
ws = optim_alg(w0,learning_rates,input_to_optim_alg)
```

5. **(10 points)** To combat the COVID-19 pandemic, an enterprising NYU Tandon graduate student decides to build a logistic regression model to predict the conditional likelihood of a person being one of three states – susceptible, infected, cured – based on forehead temperature measurements over the last 30 days. Fortunately, such measurements for a population of 10,000 persons is available.

   a. Identify the parameters of the problem (number of samples n, data dimension d, number of classes k.)

   b. If X and y denote the arrays that encode the training data points and labels, what are the sizes of X and y?

   c. Starting from the definition of conditional likelihood, derive the loss function used to train the model.

6. **(10 points)** Recall that the kernel trick is basically used to compute inner products between data points that are implicitly transformed into some higher dimensional space via a mapping ϕ:

$$K(x, z) = \langle \phi(x), \phi(z) \rangle.$$

Let's assume that data points are 2-dimensional. So if $K(x, y)$ = (1 + xˆT z)ˆ2$ is the quadratic kernel, write down the explicit form of the mapping ϕ (i.e., write down what $\phi(x)$ and $\phi(z)$ are for this kernel).