

## CIS 472/572, Winter 2020

### Programming Homework 1: Nearest Neighbor

DUE DATE: Submit via Gradescope by Thursday, January 30th at 11:59pm.

## 1 Description

In this assignment, you will implement the  $k$  nearest neighbor algorithm. **Do not copy code from online or from other students. Students who are caught plagiarizing code or otherwise violating academic integrity will be given a failing grade in the class.** Your code must be your own. Undergraduates may complete the assignment in teams of 2. Graduates must complete the assignment alone.

Please start with the provided template code, `knn.py`, so that your code will work with the autograder. The template code will handle a lot of things for you, including processing command line arguments, reading in data files, and scoring your classifier. You will need to fill in code where you see the comment, “YOUR CODE HERE.”

The code you will write includes:

- `dist(x1,x2)` – return the Euclidean distance between instances `x1` and `x2`.
- `classify(train_x, train_y, k, x)` – predict the label of instance `x` using the  $k$  nearest neighbors from among training instances `train_x` with labels `train_y`.
- `normalize_data(train_x, test_x, rangenorm, varnorm, exnorm)` – rescale the instances in `train_x` and `test_x`, performing a feature range normalization (if `rangenorm` is true), feature variance normalization (if `varnorm` is true), and/or example magnitude normalization (if `exnorm` is true). Feature range normalization should rescale instances to range from -1 (minimum) to +1 (maximum), according to values in the training data. Feature variance normalization should rescale instances so they have a standard deviation of 1 in the training data. And example magnitude normalization should rescale each example to have a magnitude of 1 (under a Euclidean norm).

Submit your code to the Gradescope autograder and it will be scored for correctness automatically. However, if we spot obvious bugs in your code that the autograder missed, you could still lose points.

**NOTE:** This should be a pretty short assignment. I may add some written questions for you to answer based on experiments you run with your code, to help you get a feel for how  $k$  nearest neighbor works and what it’s doing.