MTH 4320/5320 Homework 1

Deadline: Wednesday, August 26

Problems

1. Assume the mean of $x_1, ..., x_n \in \mathbb{R}$ is μ and the standard deviation is σ. Define $z_i = \frac{x_i - \mu}{\sigma}$ for each i. Prove these $z_1, ..., z_n$ have mean 0 and variance 1. [5 points]

2. For $x = (x_1, ..., x_m) \in \mathbb{R}^m$, prove that $\|x\|_\infty = \lim_{p \to \infty} \|x\|_p = \max_i\{|x_i|\}$. [5 points]

3. Write an implementation of k-nearest neighbors algorithm with two hyperparameters: k and p for the L^p norm (be sure that it can accept all $p > 0$ including $p = \infty$).

   Follow the scikit-learn structure where the classifier is an object from a class (in the programming sense) with fit and predict functions. [10 points]

4. Use your k-nearest neighbor classifier with L^p norm to classify the CIFAR-10 dataset. Randomly split it into training and test sets. Find the accuracy, precision, and recall for each class on the test set.[1] [5 points]

5. Tune the hyperparameters to get the best fit you can. Test at least ten different options for (k, p) and at least one normalization method.

   Randomly split the dataset into 60%/20%/20% training/validation/testing sets. When tuning hyperparameters, test on the validation set. At the end, use the test set.[2] [5 points]

Bounty: Whoever achieves the highest classification accuracy gets +5 points.

Instructions

- Preferably, create an 1 notebook file. Or, create a PDF and submit code files separately.
- You document should be similar to my notes for Week 1 in GitHub, containing three parts:
  1. Typed work or pictures of your written work for problems 1-2
  2. Text explanations for problems 3-5
  3. Well-commented code for problems 3-5
- Submit your notebook or PDF + code files (+ any other files, if needed) in GitHub Classroom (details coming soon).
  - Any language I can easily run is acceptable, but I highly recommend Python due to the built-in functions and compatibility with code from my notes (so you can use it).
  - If you do not know Python, learning the basics is almost certainly worth the time investment for this class (and beyond). You do not need too much for the class.

[1] Use the full dataset if you have the computational resources to do it. If not, use at least 1000 images.

[2] Use **random_state = 1** in **train_test_split** or set the random seed to 1 in any language before splitting data.