DS 2500: Intermediate Programming with Data
Profs. Rachlin and Park
Northeastern University

K-Nearest Neighbor classification: Finding the Best K

Problem Description:

In this assignment you will implement your own k-nearest neighbor learning system and explore its capabilities on a dataset of your own choosing.

Step 1. Choose a data set from the UCI Machine Learning Repository (https://archive.ics.uci.edu/ml/datasets.php). It should be a dataset with at least some numerical (integer or real-valued) attributes, having 100-2000 instances, and be aimed at the problem of classification. The classes (which you are trying to predict) should be categorical, not numeric. (It's ok if the classes are integers representing different categories, however.) Do not choose a dataset from Kaggle or one that we have already explored in class. Pick a less-widely-studied dataset that looks interesting to you! Warning: k-nearest neighbor learning is compute-intensive and not practical for very large datasets.

Step 2. Build a k-nearest neighbor classifier. You may not use the scikit-learn library for this assignment except where noted below. Instead, implement your own home-grown classification algorithms. You will learn so much more about machine learning by writing your own code! For this exercise, you will do *n-fold cross validation:* Instead of breaking the data into a test set and a training set, you will classify each example by finding the k nearest neighbors among all *other* instances in the dataset. You may use a standard Euclidean distance measure but design your classifier so that you can readily experiment with alternative distance metrics. Your final distance function should be clearly documented sufficient to reproduce your results.

Step 3. Hyper-parameter tuning. Test the performance of your classifier for k=1, 3, 5, … 31. Report the overall accuracy as well as the weighted average precision, recall and f1-score for each k-value (four values for each k). You may use the scikit learn **metrics** library to generate a classification report and extract these values. Create one or more visualizations to convey how the performance of your classifier is impacted by k.

What to submit:

Submit your code (.py files) and one or more visualizations showing how the *k* parameter impacts classifier performance. You do not need to submit your data.