Introduction to Machine Learning, Programming Assignment 3

Comparing K-Nearest Neighbor Classifiers

Julie P. Garcia

Johns Hopkins University

Abstract

In this paper, we evaluate several methods for k-Nearest Neighbors as a nonparametric classification and regressor algorithm. We will analyze the value of condensing a training set with a reducing algorithm like Condensed k-Nearest Neighbors against the use of k-Nearest Neighbors on the entire dataset. K-Nearest neighbors often performs well as a classifier, however, when datasets get large, the algorithm’s space complexity increases as it is required to store O(N), where N is the size of the training set. For this reason, we try to reduce the size of the dataset by eliminating examples that do not add value to the percent of accuracy of the classifier.

Hypothesis

Here we try to prove that for larger datasets, Condensed k-Nearest Neighbors performs as well as k-Nearest Neighbors in accuracy for classification and regression problems, while reducing the time and space complexity of the algorithm.

Algorithms Implemented

The algorithms implemented were K-Nearest Neighbors (kNN) and the variation Condensed K-Nearest Neighbors (CNN). Because the kNN is a nonparametric algorithm in which the entire training data set is stored in memory, rather than a derived model from the dataset, space and time complexity of the algorithm can be high.[3] To reduce this many variations on kNN can be used. We chose to test kNN vs CNN.

K-Nearest Neighbors

The k-Nearest Neighbors algorithm can be used as a nonparametric classifier and works by searching through a dataset for examples that are similar. The k-most similar instances, measured by a distance algorithm, are selected to group or classify the data. It is a lazy algorithm, which means the entire dataset is stored in memory, and only until examples are fed into the algorithm is any calculation done. A model is not predefined from the dataset before testing is done. In the simplest version of k-Nearest Neighbors, all of the training data is used as the model. The steps to kNN are as follows:

1. Split data into a test set and four training sets
2. Loop through each test example:

   a. Find all distances between each example in the training set and the test instance, we used Euclidean distance
   b. Get the k nearest neighbors by sorting the distances ascending and taking k values off the top
   c. Use the classes of the k-nearest neighbors selected in (b) to vote for the class of test set and set the test class prediction to that (ties were chose arbitrarily).
3. Test the class predictions for accuracy against the actual classes

Condensed K-Nearest Neighbors

With condensed nearest neighbor, we try to find a subset of the training set for which the error is lower or equal to that of the entire training set. This reduces computation time and space and ideally will give us a better result by eliminating noise and/or training examples that do not lend themselves to learning. This is a greedy algorithm that takes the following steps:

1. Initialize Z to the empty set.
2. Add the first point to Z
3. Loop through each point in the training set
   a. Find a point within Z minimum to each point in the training set
   b. If the classes are not the same, add that point to Z
   c. Check the error rate, if there is no change stop

Experimental Approach

All algorithms were run on four datasets, E.coli, Image Segmentation, Computer Hardware and Forest Fires. For this a stratified 5-fold cross-validation method was used. Each dataset was divided into five randomly-selected equal datasets. The datasets are stratified to ensure equal representation of classes across each set. Five experiments were run, each time using a different partition (20%) as the test set and the remaining 80% as the training set. For classification problems, averages were calculated over the five experiments. Euclidean distance was used as measure of distance between points.

K was tuned by starting with 1 and increasing until the score started to significantly decrease, then tuned backwards. For the Ecoli dataset, k = 2 was the optimal value for kNN, while

Results

Because it uses the entire training data set as the model, kNN is computationally expensive. The advantage of kNN over CNN was that it proved to be more accurate, however, as the size of the dataset increased, the time to complete and memory usage also grew. Using CNN greatly decreased the space complexity, because the training set was often reduced greatly from the original set. The time complexity increased as k increased for each dataset.

Tuning of k was done by starting with one and increasing until a significant decline in performance was seen. Then selecting the best performing value of k for each dataset. For the ecoli dataset the value of k = 3 was chosen, with 81% accuracy, for kNN and k = 10 for CNN, with 54% accuracy. For the image segmentation dataset the value of k = 3 was chosen, with 77% accuracy, for kNN and k = 7 for CNN, with 40% accuracy. For the computer hardware dataset the value of k = 7 was chosen for kNN. For the forest fires dataset the value of k = 8 was chosen for kNN.

Summary

In summary, kNN can be used as an accurate classifier alone, however, with very large datasets, some method of reducing the dataset will be necessary when time and space are factors. kNN introduces very little inductive bias, as it does not make many assumptions about the data, only that distance measure can be used as a measure of similarity in two examples. The distance measure selected matters. CNN performed better on larger datasets with larger values of k, while kNN performed better on smaller datasets and small values of k.

References

1. Alpaydin, E. (2014). *Introduction to machine learning*. Cambridge, MA: The MIT Press.
2. Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.
3. Makkar, T., Kumar, Y., Dubey, A. K., Rocha, A., & Goyal, A. (2017). Analogizing time complexity of KNN and CNN in recognizing handwritten digits. *2017 Fourth International Conference on Image Information Processing (ICIIP)*. doi:10.1109/iciip.2017.8313707