PROJECT 2 - NEAREST NEIGHBORS CLASSIFICATION AND REGRESSION

DAVID ATLAS

Abstract. This paper will introduce the K Nearest-Neighbors (KNN) algorithm for supervised learning. It will also introduce two variants - Edited KNN and Condensed KNN - that provide more efficient predictions (and potentially better performance). In several experiments on real world classification problems, the algorithms perform comparably. On regression problems, KNN does not perform particularly well compared to an OLS baseline.

1. Problem Statement & Hypothesis

A non-parametric density estimation technique, K-Nearest Neighbors, is adapted for supervised learning, and applied to real world classification and regression problems. Additionally, two variants are introduced that seek to make prediction more efficient by reducing the set of points included in calculated the nearest neighbors. These variants are only valid for classifications problems, and so are not used on regression problems.

Because these algorithms rely on density estimation, it is expected that they will not perform well in high dimensional spaces. Additionally, the algorithms cannot detect interactions between features, and so would not be expected to perform well if there are interactions. The density estimation technique (explained in more detail below) treats all features equally, and so performance may be poor if many unnecessary features are included in the model, as the important features may be drowned out by the distances of the unimportant features.

Out of the algorithms in this paper, the standard KNN should outperform in situations where the data is not too noisy, and the classes are reasonably seperated. An Edited KNN model might perform better on noisy data, as points that lie in the space of a different class are deleted from the prediction set. Additionally, if there are irrelevant features included in the algorithm, an Edited KNN may perform well, as it may drop points in a space that are far from the true class center only along irrelevant attribute dimensions.

A Condensed KNN might perform poorly on noisy data, as points far from their class center will be added back into the prediction set. However, it might do better in a situation where classes occupy multiple spaces.

2. Description of Algorithms

K Nearest Neighbors. The K Nearest-Neighbors algorithm relies on the general hypothesis that points that are near each other in feature space are also near each other in hypothesis space. As such, for a given instance, the prediction class is chosen via a majority voting scheme within the k training points that are nearest to the instance.

The K Nearest-Neighbors algorithm is a lazy algorithm, in that it does not do anything on fitting, but does all the work at prediction time. It calculates the distance between the instance to label, x_0, and the training instances, $X = (x_1, \cdots, x_n)$. The training points with the k shortest distances are then selected, and the class with the most elements of the k is chosen for x_0. Ties are broken randomly.

There are two choices to make regarding the algorithm. One is the choice of k, or the number of neighbors that participate in the majority vote. This value can be chosen via a holdout validation set. The other choice is that of the distance metric to be used in finding the "nearest" neighbors. This is commonly the 2-norm (Euclidean distance), but other p-norms can be chosen. This likely depends on the real-world meaning of the data.

One other important note on nearest-neighbor schemes is the importance of standardizing the data before running the algorithm. Because the distance between data points is used, if one feature is of a larger scale, it can easily drown out others, and if it is of a smaller scale, it can easily be drowned out by others.

Edited KNN. The motivation behind Edited KNN is twofold. Consider that with the standard KNN, at prediction time, given n training points, n distance calculations must be made in order to make a single prediction. If the set of training points can be edited such that there are fewer points, but the predictions are roughly the same, prediction becomes computationally easier. Additionally, if there are many points in the training set that are not near their class centers, prediction quality may suffer, as an instance near them may be misclassified.

Edited KNNs[1] are very similar to the standard KNN. At fitting time, each of the training points x_i are iteratively classified using all of the other data points, X_{-i}. If the prediction $\hat{y}_i \neq y_i$, then x_i is dropped from the training set. In [1], all points are considered for removal once. Alternatively, this process can be repeated until no further changes are made, or until performance on a validation set starts to decrease.

Alternatively, the editing procedure can remove points that are correctly classified, with the hypothesis that they are likely not needed for correct classification going forward.

Note that at prediction time, the class of the single nearest neighbor is used.

Condensed KNN. The motivation for the Condensed KNN is similar to that of the Edited KNN. It is desirable to have fewer points in the training set, and to retain only those points that are needed to avoid an increase in error.

The condensing algorithm begins with an empty set $Z = \emptyset$. The first data point considered is added to Z. Then, for each of the remaining data points x_i, the distances between x_i and all points in Z are calculated, and the nearest data point z_* is found. If the class of x_i is not equal to the class of z_*, x_i is added to Z. This process is repeated until no changes are made.

Note that at prediction time, the class of the single nearest neighbor is used.

3. Description of Experimental Approach

The 3 algorithms were applied to 2 real life classification problems:

(1) The Ecoli dataset for classifying the localization site of protein.
(2) The Image Segmentation dataset for classifying the part of the image of a given pixel.

Standard KNN was applied to 2 additional regression problems.

(1) The CPU Performance dataset for predicting the performance of a given CPU (regression problem).
(2) The Forest Fires dataset in which the area burned is predicted (regression problem).

Discrete inputs were one-hot encoded. A 5-Fold cross validation scheme was used to estimate the expected out-of-sample error. For classification problems, a stratefied cross-validation approach was used to accurately represent the various classes. Within the cross-validation scheme, k was tuned by choosing

$$\arg\max_{1 \leq k \ 5} f(k),$$

where $f(k)$ was accuracy for classification problems, and negative MSE for regression problems. k was chosen by running 5-fold CV over the first training set of the overall 5-fold CV, and taking the mean of $f(k)$ across the folds.

Each dataset was standardized dividing by the column-wise standard deviation. This ensures that the scale of each feature is the same (a distance of 1 indicates a 1 standard deviation difference).

The KNN structure naturally handles multi-class classification problems, and regression problems were solved by taking the mean across the target values of the k neighbors.

4. Experimental Results

References

[1] Dennis L. Wilson. Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems, Man, and Cybernetics*, pages 408–421, 1972.