**K-Nearest Neighbor’s**

**Background Information**

In machine learning, **pattern recognition** is the assignment of a label to a given input value. In statistics, discriminant analysis was introduced for this same purpose. An example of pattern recognition is classification, which attempts to assign each input value to one of a given set of *classes* (for example, determine whether a given email is "spam" or "non-spam").

In pattern recognition, the ***k*-Nearest Neighbors algorithm** (or ***k*-NN** for short) is a non-parametric method used for classification and regression. In both cases, the input consists of the *k* closest training examples in the feature space. The output depends on whether *k*-NN is used for classification or regression:

* In *k-NN classification*, the output is a class membership. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its *k* nearest neighbors (*k* is a positive integer, typically small). If *k* = 1, then the object is simply assigned to the class of that single nearest neighbor.
* In *k-NN regression*, the output is the property value for the object. This value is the average of the values of its *k* nearest neighbors.

*K-NN* is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until classification.

Both for classification and regression, it can be useful to weight the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones. For example, a common weighting scheme consists in giving each neighbor a weight of 1/*d*, where *d* is the distance to the neighbor.

The training examples are vectors in a multidimensional feature space, each with a class label. The training phase of the algorithm consists only of storing the feature vectors and class labels of the training samples.

In the classification phase, k is a user-defined constant, and an unlabeled vector (a query or test point) is classified by assigning the label which is most frequent among the k training samples nearest to that query point.

A commonly used distance metric for continuous variables is Euclidean distance. For discrete variables, such as for text classification, another metric can be used, such as the **overlap metric** (or [Hamming distance](http://en.wikipedia.org/wiki/Hamming_distance)). Often, the classification accuracy of *k*-NN can be improved significantly if the distance metric is learned with specialized algorithms such as Large Margin Nearest Neighbor or Neighborhood components analysis.

A drawback of the basic "majority voting" classification occurs when the class distribution is skewed. That is, examples of a more frequent class tend to dominate the prediction of the new example, because they tend to be common among the *k* nearest neighbors due to their large number.[[3]](http://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm#cite_note-Coomans_Massart1982-3) One way to overcome this problem is to weight the classification, taking into account the distance from the test point to each of its *k* nearest neighbors.

**Algorithm**

Works based on minimum distance from the query instance to training samples to determine the k-Nearest Neighbors. After we gather the K-Nearest Neighbors, we take a simple majority of these Nearest Neighbors to be the prediction of the query instance.

Steps

1. Determine parameter k = Number of Nearest neighbors.
2. Calculate the distance between the query instance and all the training samples.
3. Sort the distances and determine the nearest neighbors based on the k-th minimum distance.
4. Use simple majority of the category of nearest Neighbors as the prediction value of the query instance.