Module 4: K-Nearest-Neighbors Classifiers

Video: 680_mod4_vid1

Similarities: Nearest-Neighbor Classifiers

Two plants that look very much alike probably represent the same species; likewise, it is quite common that patients complaining of similar symptoms suffer from the same disease. In short, similar objects often belong to the same class—an observation that forms the basis of a popular approach to classification: when asked to determine the class of object x, find the training example most similar to it. Then label x with this example's class.

The module examines how to evaluate example-to-example similarities and presents concrete mechanisms that use these similarities for classification purposes.

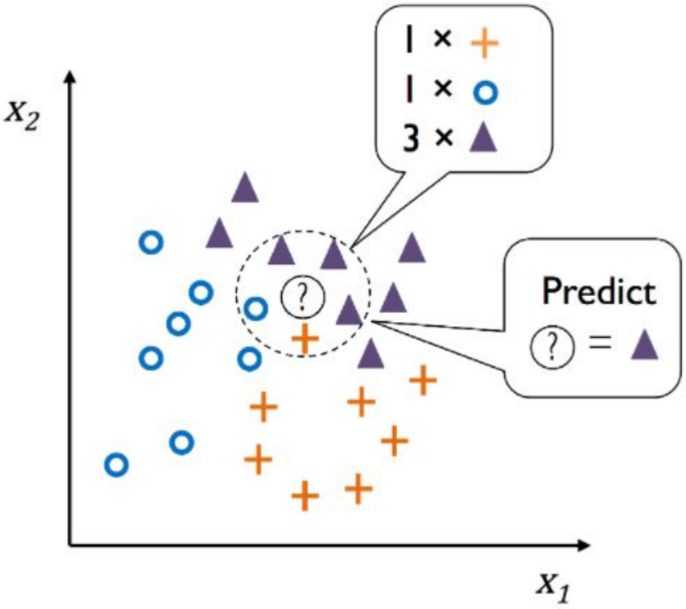
The K-Nearest-Neighbor Rule

How do we establish that an object is more similar to **x** than to **y**? Some may doubt that this is at all possible. Is a giraffe more similar to a horse than to a zebra? Questions of this kind raise suspicion because there are too many arbitrary and subjective factors involved in answering them.

Let's discuss the simplest version of the K-NN classifier. Suppose we have a mechanism to evaluate the similarly between attribute vectors. Let x denote the object whose class we want to determine.

1. Among the training examples, identify the k nearest neighbors of x (examples most similar to x).
2. Let ci be the class most frequently found among these k nearest neighbors.
3. Label x with ci.

These simple steps are depicted in the following figure.



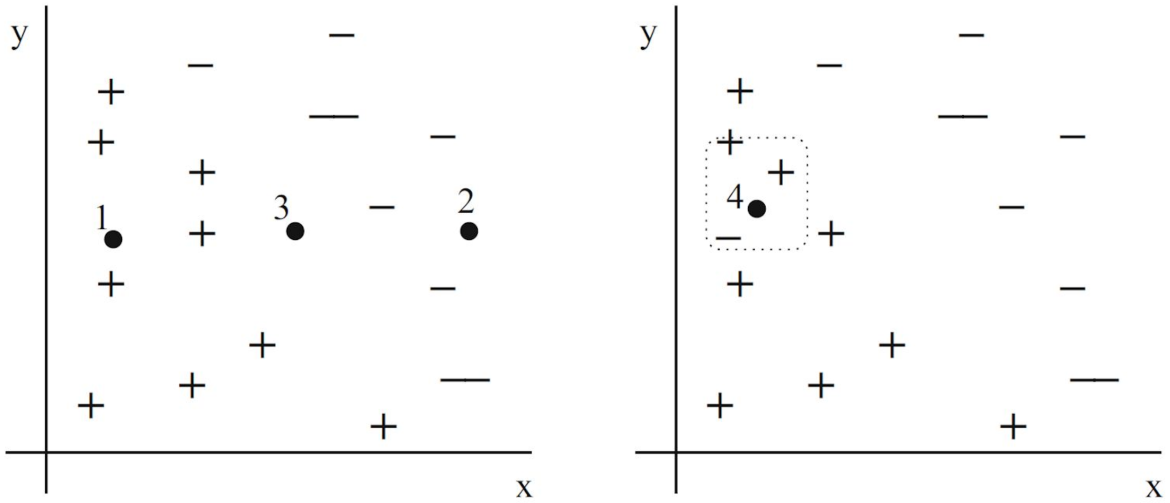
The Need for Multiple Neighbors

In noisy domains, the testimony of the nearest neighbor cannot be trusted. What if this single specimen's class label is incorrect? A more robust approach identifies not one, but several nearest neighbors, and then lets them vote. This is the essence of the so-called k-NN classifier, where k is the number of the voting neighbors, which is a user-specified parameter.

Note that, in a two-class domain, k should be an odd number so as to prevent ties. For instance, a 4-NN classifier might face a situation where the number of positive neighbors is the same as the number of negative neighbors. This will not happen to a 5-NN classifier.

As for domains that have more than two classes, using an odd number of nearest neighbors does not prevent ties. For instance, the 7-NN classifier can realize that three neighbors belong to class C1, three neighbors belong to class C2, and one neighbor belongs to class C3. The engineer designing the classifier then needs to define a mechanism to choose between C1 and C2.

The following figure shows several positive and negative training examples, and also some objects (the big black dots) whose classes the k-NN classifier is to determine.



The reader can see that objects 1 and 2 are surrounded by examples from the same class, and their classification is therefore straightforward. On the other hand, object 3 is located in the "no man's land" between the positive and negative regions, so that even a small amount of attribute noise can send it to either side. The classification of such borderline examples is unreliable.

In the right-hand part of the picture, object 4 finds itself deep in the positive region, but class noise has mislabeled its nearest neighbor in the training set as negative. Whereas the 1-NN classifier will go wrong, here, the 3-NN classifier will give the correct answer because the other two neighbors, which are positive, will outvote the single negative neighbor.

Measuring Similarity

The most common similarity measure is the Euclidean distance, defined as:

$$d(\mathbf{x},\mathbf{y}) = \sqrt{(x_1-y_1)^2+(x_2-y_2)^2+\ldots+(x_n-y_n)^2}$$

which is equivalent to:

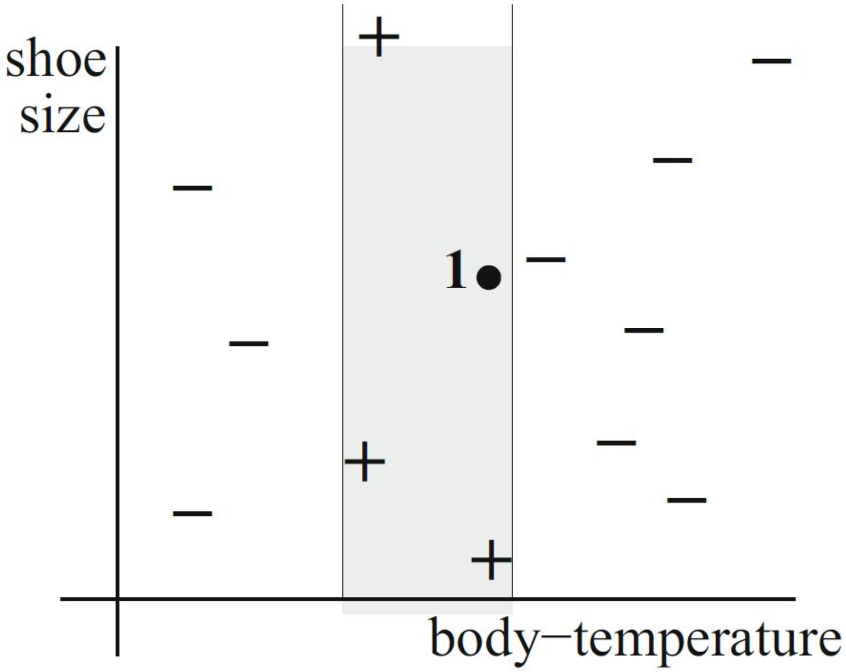
$$d(\mathbf{x},\mathbf{y}) = \sqrt{\sum_{i=1}^{n}(x_i-y_i)^2}$$

Irrelevant Attributes and Scaling Problems

It would be a mistake to think that all attributes are created equal. They are not. In the context of machine learning, some are irrelevant in the sense that their values have nothing to do with the given example's class. But they do affect the geometric distance between vectors.

Let’s consider an example. In the following figure, the training set examples are characterized by two numeric attributes: body-temperature (the horizontal axis) and shoe-size (the vertical axis). The black dot stands for the object that the k-NN classifier is expected to label as healthy (pos) or sick (neg) - that is, the black dot is the new instance that we want to make a prediction on.

It is intuitive that all positive examples find themselves in the shaded area delimited by two critical points along the “horizontal” attribute: temperatures exceeding the maximum indicate fever; those below the minimum, hypothermia. Now, the attribute measured along the vertical direction exhibits many positive and negative examples alike distributed along the entire range. This suggests that the *shoe size* attribute is unable to predict a person’s health. The object we want to classify is in the highlighted region, and common sense requires that it should be labeled as positive—despite the fact that its nearest neighbor happens to be negative.



Video: 680_mod4_vid2 and on do not have notes.