Algorithm 1: Nearest Neighbor Algorithm

1 **Input**:
2 $\mathcal{S} = \{(x_i, t_i) \mid x_i \in \mathbb{R}^m, t_i \in \mathbb{N}, i \in \{1, 2, ..., n\}\}$ – the set of n training samples and labels;
3 $Z = \{z_i \mid z_i \in \mathbb{R}^n, i \in \{1, 2, ..., l\}\}$ – the set of l test samples;
4 k – the number of neighbors;
5 Δ – a distance measure.
6 **Initialization**:
7 $Y \leftarrow \emptyset$;
8 **Computation**:
9 **for** $z_i \in Z$ **do**
10 $\quad \mathcal{N} \leftarrow$ the nearest k neighbors to z_i from $\mathcal{S}$ according to Δ;
11 $\quad y \leftarrow$ the majority label obtained through a voting scheme on $\mathcal{N}$;
12 $\quad Y \leftarrow Y \cup \{y\}$;
13 **Output**:
14 $Y = \{y_i \mid y_i \in \mathbb{N}, i \in \{1, 2, ..., l\}\}$ – the set of predicted labels for the test samples in Z.

based on the notions of similarity and nearest neighbors. An overview of kernel methods is given in Section 2.4. The chapter ends with Section 2.5, which gives an overview of clustering methods based on similarity.

2.2 Nearest Neighbor Approach

Since the k-nearest neighbors algorithm (k-NN) was introduced in [Fix & Hodges, 1951], it has been studied by many researchers and it is still an active topic in machine learning. The k-nearest neighbors algorithm is one of the simplest of all the machine learning algorithms, proving that simple models are always attractive for researchers. The nearest neighbor model is described in Algorithm 1.

The k-nearest neighbors classification rule employed in step 11 of Algorithm 1 works as follows: an object is assigned to the most common class of its k nearest neighbors, where k is a positive integer value. If $k = 1$, then the object is simply assigned to the class of its nearest neighbor. When $k > 1$, the decision is based on a majority vote. It is convenient to let k be odd, to avoid voting ties. However, if voting ties do occur, the object can be assigned to the class of its 1-nearest

neighbor, or one of the tied classes can be randomly chosen to be the class assigned to the object. The output of Algorithm 1 is a set of labels associated to the test samples.

The example about handwritten digit recognition presented in Figure 2.1 gives some insights of how the k-NN model works in practice. In this example, digits are represented in a two-dimensional feature space. When a new sample x comes in, the algorithm selects the nearest 3 neighbors and assigns the majority class to x. In Figure 2.1, the majority label among the nearest 3 neighbors of x is 4. Thus, label 4 is assigned to x. This model can be referred to as a 3-NN model. To better understand how the decision of the k-NN model is taken in general, it is worth considering a 1-NN model. For this model, the decision at every point is to assign the label of the closest data point. This process generates a Voronoi partition of the training samples, as seen in Figure 2.2. Each training data point corresponds to a Voronoi cell. When a new data point comes in, it is assigned to the class associated to the Voronoi cell that the respective data point falls in.

The k-NN algorithm is a non-parametric method for classification. In other words, no parameters have to be learned. In fact, the k-NN model does not require training at all. The decision of the classifier is only based on the nearest k neighbors of an object with respect to a similarity or distance function. The euclidean distance measure is a very common choice, but other similarity measures can also be used instead. Actually, the performance of the k-NN classifier depends on the strength and the discriminatory power of the distance measure used. It is worth mentioning that a good choice of the distance metric can help to achieve invariance with respect to a certain family of transformations. For example, a distance metric that is invariant to scale, rotation, luminosity and contrast changes is a suitable choice for computer vision tasks. Researchers continue to study and develop new similarity or dissimilarity measures for a broad variety of applications in different domains. But, when it comes to testing the similarity measure in machine learning tasks, the method of choice is the k-NN model, because it deeply reflects the strength of the similarity measure. Good examples of this fact are the Tangent distance [Simard et al., 1996] and the shape matching distance [Belongie et al., 2002], which are both used for handwritten digit recognition. For the same reason, the k-NN model is used to assess the

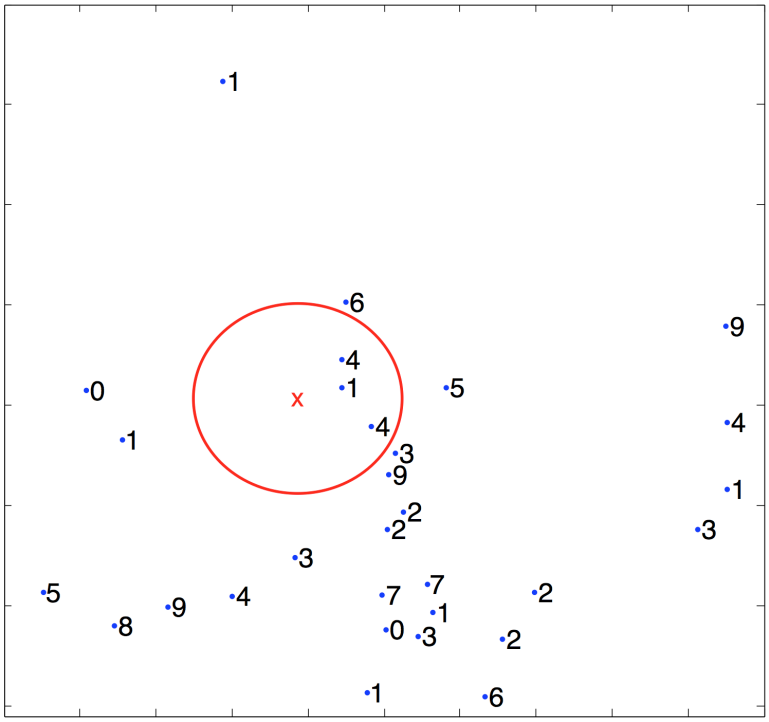

Figure 2.1: A 3-NN model for handwritten digit recognition. For visual interpretation, digits are represented in a two-dimensional feature space. The figure shows 30 digits sampled from the popular MNIST data set. When the new digit x needs to be recognized, the 3-NN model selects the nearest 3 neighbors and assigns label 4 based on a majority vote.

performance of the new dissimilarity measure for images presented in Chapter 4 of this work.

It is interesting to mention that the k-NN model is one of the first classifiers for which an upper bound of its error rate has been demonstrated. More precisely, a theoretical result demonstrated in [Cover & Hart, 1967] states that the nearest neighbor rule is asymptotically at most twice as bad as the Bayes rule. Furthermore, if k is allowed to grow with n such that $k/n \to 0$, the nearest neighbor rule is universally consistent. More consistency results and other theoretical aspects of the k-NN model are discussed in [Devroye et al., 1996].

The k-NN model defers all the computations to the test phase. This rep-

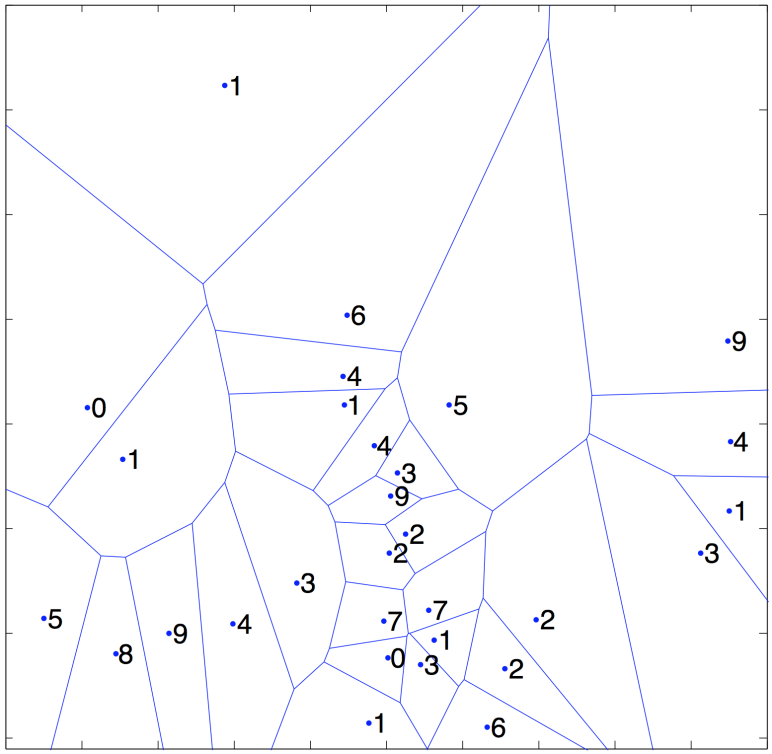

Figure 2.2: A 1-NN model for handwritten digit recognition. The figure shows 30 digits sampled from the popular MNIST data set. The decision boundary of the 1-NN model generates a Voronoi partition of the digits.

resents a great disadvantage when the computational time is taken into consideration. Searching for the k nearest neighbors among n training samples may take time proportional to $O(n \cdot k \cdot d)$ using a naive approach, where d represents the computational cost of the distance function. Different approaches based on multidimensional search trees that partition the space and guide the search have been proposed to reduce the time complexity [Dasarathy, 1991]. Other fast k-NN approaches are proposed in [Faragó et al., 1993] and [Zhang & Srihari, 2004].