Content-based retrieval of visual information

Oerlemans, A.A.J.

Chapter 3

Machine Learning

This chapter gives an overview of the automated learning techniques that were used in this thesis. First, a short introduction to machine learning for classification is given and then all methods used are described in detail.

3.1 Introduction

Searching for images in a database requires some form of similarity measure to determine if an image is a match to the query. The result of the search is then a list of images, ranked by similarity to the query image. The similarity can be calculated in many ways, for example by using the low level features and the feature similarity measures that were mentioned in the previous chapter. However, classification methods based on high level semantics are also commonly used. A classifier would then be an algorithm that can answer a question like 'Does this image contain grass?'

First, we introduce some mathematical notation for describing the datasets that we have used in the machine learning tasks. We denote a dataset by D, one data point is represented by x, the number of dimensions of a data point is denoted by n, the number of data points in the set by m and the classification of a data point as c. Note that the features and the resulting feature vectors we have described in the previous chapter, can be concatenated to form one large vector that forms one data point in the dataset.

A binary classification problem can be seen as a mapping between data point and two possible outputs. We define these outputs as -1 and 1. The formal definition of a dataset with binary labels can then be given as:

$$D = \{(x_i, c_i) | x_i \in \mathbb{R}^n, c_i \in \{-1, 1\}\}_{i=1}^{m} \tag{3.1}$$

3.1.1 A sample binary classification problem

This section shows an example of a toy binary classification problem. In this example, the objective is to determine if a car is a sports car. Given a few properties of a car, we would like to automatically determine if the car is a sports car.

First, let us take a look at a the example in table 3.1.

Car	Weight	Engine displ.	Supercharger	Sports car?
Audi TT	1290	1.8	yes	yes
DAF Truck	4000	8.0	no	no
Ford Focus	1200	2.0	no	no
Ferrari	1500	4.0	no	yes

Table 3.1: A sample dataset for binary classification.

This short list of examples can be used to train a binary classifier. After training, the classifier would then hopefully be able to classify new samples based on weight, engine displacement and the presence of a supercharger. If we would present a new sample to the classifier, for example $(900, 1.4, no)$, then the classifier would probably output that this is not a sports car.

The following sections demonstrate a few classification techniques that were used in this thesis.

3.2 *k*-nearest neighbor

The k-nearest neighbor classification algorithm is an algorithm that needs to keep all training data within reach when classifying new examples. First, the algorithm needs a distance function for objects that need to be classified. This function is then used to calculate the distance between the new sample and all training samples.

The simplest form of nearest-neighbor classification is to find the closest matching training sample and to classify the new sample with the same classification that this closest training sample has. However, a more robust version of this is to use a few of the closest training samples, to see if they all have the same classification.

The k in k-nearest-neighbor classification stands for the number of close training samples that are used in determining which label to assign to the new sample. In case of a 3-nearest-neighbor classification, the three closest training samples are selected and the label that has the highest occurrence is selected as the label for the new sample. Figure 3.1 illustrates this.

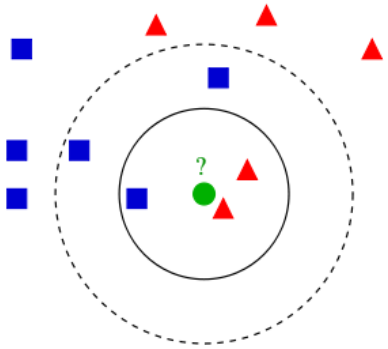

Figure 3.1: Example of the k-nearest neighbor classification. Based on the three nearest neighbors, the input will be classified as the type represented by the triangles.

3.3 Artifical neural networks

An artificial neural network is a biologically inspired method for learning mathematical functions. Neural networks have many more applications than binary classification, but they are well suited for them. Several recommended surveys of neural network research are [13] [18] [98].

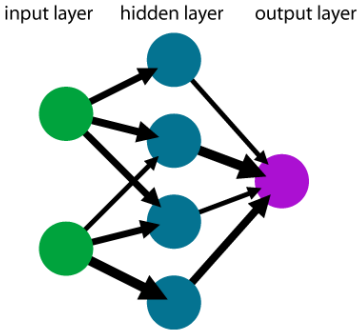

Figure 3.2: An example of a simple three layer neural network with seven artificial neurons. The thickness of an arrow represents the weight of the connection.

Neural networks are based on a simple computational element, which is used in a network-like structure to perform complex computations. This simple element is called an artificial neuron, a simplified model of the main component of the human brain, the neuron.

An artificial neuron can have several weighted inputs and the sum of these inputs is fed into an activation function to determine the output of the neuron.

$$output = f\left(\sum_{i=1}^{n} x_i w_i\right) \tag{3.2}$$

where x_i is input value i and w_i is the weight for input i. Inputs usually have a value between -1 and 1. The activation function is a function that outputs a value between -1 and 1, based on the input value. There are several options for this activation function, but a sigmoid is a common choice.

A combination of several artificial neurons that are connected to each other, results in a neural network. Some neurons process the inputs and other neurons process the outputs of these input neurons. These neurons are usually organized in layers and in each successive layer, the number of neurons decreases.

For our binary classification problem, a neural network that ends in just one neuron can be used. The network learns to classify samples by applying a learning algorithm such as the back-propagation algorithm to a set of training samples. If a well-suited network size is chosen, the network will generalize the training samples and it can then be used to classify new samples.

3.4 Support vector machines

Support vector machines, or SVMs in short, is a technique that was developed by Vladimir Vapnik [90]. The basic idea is that the input data are handled as vectors in a vector space and that a hyperplane is determined that best separates the positively labeled input vectors from the negative input vectors. The hyperplane is said to have maximum margin, as it forms the best possible separation of the two classes and has maximum distance to the closest vectors of each class.

To find this maximum margin hyperplane, two other hyperplanes are used that are placed at the boundaries of both classes. By maximizing the distance between these two hyperplanes, the resulting maximum margin hyperplane can be determined.

Non-linear classification is accomplished by transforming the vector space with a given kernel function and trying to find the hyperplane in this new vector space. If the kernel function is not linear, the resulting hyperplane in the transformed space is in fact a representation of a non-linear shape in the original vector space.

A good tutorial can be found in [4]. We have used a library by Joachims [34] in our experiments.

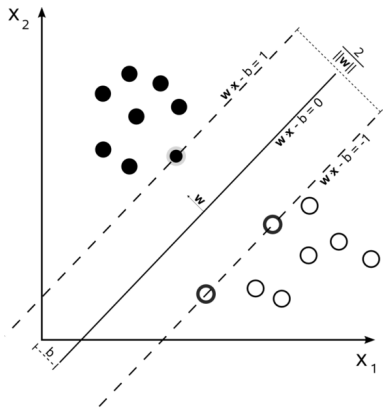

Figure 3.3: An example of the maximum margin hyperplane that was found after training the support vector machine. Vectors on the two margins are called the support vectors.