Separating Hyperplane

Related terms:

Support Vector Machine, Kernel Function, Optimization Problem, Classification (Machine Learning), Support Vector Machines, Classification, Hyperplanes, Support Vector

View all Topics

Implementations

Ian H. Witten, ... Mark A. Hall, in Data Mining (Third Edition), 2011

Maximum-Margin Hyperplane

Support vector machines address both problems. They are based on an algorithm that finds a special kind of linear model: the *maximum-margin hyperplane*. We already know what a hyperplane is—it's just another term for a linear model. To visualize a maximum-margin hyperplane, imagine a two-class dataset whose classes are linearly separable—that is, there is a hyperplane in instance space that classifies all training instances correctly. The maximum-margin hyperplane is the one that gives the greatest separation between the classes—it comes no closer to either than it has to. An example is shown in Figure 6.9, where the classes are represented by open and filled circles, respectively. Technically, the *convex hull* of a set of points is the tightest enclosing convex polygon: Its outline emerges when you connect every point of the set to every other point. Because we have supposed that the two classes are linearly separable, their convex hulls cannot overlap. Among all hyperplanes that separate the classes, the maximum-margin hyperplane is the one that is as far as possible from both convex hulls—it is the perpendicular bisector of the shortest line connecting the hulls (shown dashed in the figure).

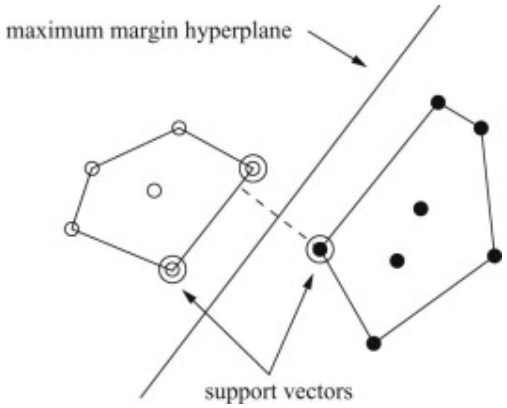

FIGURE 6.9. A maximum-margin hyperplane.

The instances that are closest to the maximum-margin hyperplane—the ones with the minimum distance to it—are called *support vectors*. There is always at least one support vector for each class, and often there are more. The important thing is that the set of support vectors uniquely defines the maximum-margin hyperplane for the learning problem. Given the support vectors for the two classes, we can easily construct the maximum-margin hyperplane. All other training instances are irrelevant—they can be deleted without changing the position and orientation of the hyperplane.

A hyperplane separating the two classes might be written asin the two-attribute case, where a_1 and a_2 are the attribute values and there are three weights w_i to be learned. However, the equation defining the maximum-margin hyperplane can be written in another form, in terms of the support vectors. Write the class value y of a training instance as either 1 (for *yes*, it is in this class) or –1 (for *no*, it is not). Then the maximum-margin hyperplane can be written as

Here, y_i is the class value of training instance **a(i)**, while b and α_i are numeric parameters that have to be determined by the learning algorithm. Note that **a(i)** and **a** are vectors. The vector **a** represents a test instance—just as the vector $[a_1, a_2]$ represented a test instance in the earlier formulation. The vectors **a(i)** are the support vectors, those circled in Figure 6.9; they are selected members of the training set. The term **a(i)** • **a** represents the dot product of the test instance with one of the support vectors: **a(i)** • **a** = $\sum_j a(i)_j a_j$. If you are not familiar with dot product notation, you should still be able to understand the gist of what follows: Just think of **a(i)** as the whole set of attribute values for the ith support vector. Finally, b and α_i are parameters that determine the hyperplane, just as the weights w_0, w_1, and w_2 are parameters that determine the hyperplane in the earlier formulation.

It turns out that finding the support vectors for the training instances and determining the parameters b and α_i belongs to a standard class of optimization problems known as *constrained quadratic optimization*. There are off-the-shelf software packages for solving these problems (see Fletcher, 1987, for a comprehensive and practical account of solution methods). However, the computational complexity can

be reduced, and learning accelerated, if special-purpose algorithms for training support vector machines are applied—but the details of these algorithms lie beyond the scope of this book (see Platt, 1998).

> Read full chapter

Precision medicine in digital pathology via image analysis and machine learning

Peter D. Caie BSc, MRes, PhD, ... Ognjen Arandjelović M.Eng. (Oxon), Ph.D. (Cantab), in Artificial Intelligence and Deep Learning in Pathology, 2021

Support vector–based methods

Support vector machines perform classification by constructing a series of class separating hyperplanes in a high-dimensional (potentially infinitely dimensional) space into which the original input data are mapped [26]. For comprehensive detail of this regression technique, the reader is referred to the original work by Vapnik [27]; herein we present a summary of the key ideas.

In the context of support vector machines, the seemingly intractable task of mapping data into a very high-dimensional space is achieved efficiently by performing the aforesaid mapping implicitly, rather than explicitly. This is done by employing the so-called *kernel trick* which ensures that dot products in the high-dimensional space are readily computed using the variables in the original space. Given labeled training data (input vectors and the associated labels) in the form $\{(x_1, y_1), \ldots, (x_n, y_n)\}$, a support vector machine aims to find a mapping which minimizes the number of misclassified training instances, in a regularized fashion. As mentioned earlier, an implicit mapping of input data $x \rightarrow \Phi(x)$ is performed by employing a Mercer-admissible kernel [28] $k(x_i, x_j)$ which allows for the dot products between mapped data to be computed in the input space: $\Phi(x_i) \cdot \Phi(x_j) = k(x_i, x_j)$. The classification vector in the transformed, high-dimensional space of the form

8.6

is sought by minimizing

8.7

subject to the constraints . The regularizing parameter λ penalizes prediction errors. Support vector–based approaches usually perform well even with relatively small

training datasets and have the advantage of well-understood mathematical behavior (which is an important consideration in the context of regularly compliance, among others).

Stepping back for a moment from the technical detail, intuitively what is happening here is that the algorithm is learning which class exemplars are the “most problematic” ones, i.e., which exemplars are nearest to the class boundaries and thus most likely to be misclassified. These are the support vectors that give the approach its name. Inspection of these is insightful. Firstly, a large number of support vectors (relative to the total amount of training data) should immediately raise eyebrows as it suggests overfitting. Secondly, by examining which exemplars end up as support vectors, an understanding of the nature of learning that took place can be gain as well as of the structure of the problem and data representation, which can lead to useful and novel clinical insight.

> Read full chapter

Statistical Learning

Igor Kononenko, Matjaž Kukar, in Machine Learning and Data Mining, 2007

10.5.1 Basic SVM principle

The basic idea of SVM methods is to place an optimal class separating hyperplane in the space of original (or transformed) attributes. If the learning examples are linearly separable, then in general there exist several possible separating hyperplanes (Figure 10.1).

Figure 10.1. For a linear two-class classification problem there exist several hyperplanes that exactly dichotomize between examples belonging to the first class (labeled with +) and examples belonging to the second class (labeled with o).

An optimal hyperplane is equally (and therefore most) distant from the nearest examples from both classes. Learning examples nearest to the optimal hyperplane are called *support vectors.* The distance between the hyperplane and its support vectors is called the *margin.* The optimal hyperplane is therefore selected so as to maximize the margin (Figure 10.2).

Figure 10.2. An optimal hyperplane for the classification problem from Figure 10.1. All three support vectors are emphasized.

Let n be the number of learning examples, and a the number of attributes. The set of learning examples is given with pairs $(\mathbf{t}_j, y_j)$, $j = 1 \ldots n$, where $y_j = 1$ if the corresponding learning example belongs to the first class, and $y_j = -1$, if the corresponding learning example belongs to the second class. We assume that all attributes are continuous, or they are treated as such, therefore $\mathbf{t}_j \in \Re^a$, $j = 1 \ldots n$. The hyperplane equation is given by

(10.47)

An optimal hyperplane should correctly classify all learning examples

(10.48)

as well as minimize its own complexity measure . By minimizing the magnitudes of coefficients we minimize the complexity of the solution. The stated problem of constrained optimization can be transformed to the functional maximization:

(10.49)

with constraints

(10.50)

and

(10.51)

This is a quadratic optimization problem that can be solved by using fast existing algorithms.

Let $\alpha_0 = (\alpha_{10}, \ldots, \alpha_{n0})$ be the solution of the above optimization problem. The only non-zero coefficients α_j are those that correspond to support vectors. A classification rule of the optimal hyperplane is thus defined with

(10.52)

The threshold value b_0 is given by

(10.53)

where $\mathbf{t}_+(1)$ and $\mathbf{t}_+(-1)$ respectively denote an arbitrary support vector from the first or the second class. The coefficient vector $\mathbf{w}_0$ of the optimal hyperplane is thus determined with its support vectors:

(10.54)

The solution complexity of the optimal hyperplane is determined with the number of support vectors. This number is rather small when compared with the number of learning examples – in practice, the ratio is usually between 3% and 5% – and therefore the solution complexity is also relatively small. Since the solution complexity depends solely on the number of support vectors, the result would be exactly the same if all non-support vectors were removed from the learning set. The optimization criterion also minimizes the magnitude of , thus minimizing the magnitude of coefficients in linear combinations of attributes.

> Read full chapter

Algorithms

Ian H. Witten, ... Mark A. Hall, in Data Mining (Third Edition), 2011

> Read full chapter

Algorithms

Ian H. Witten, ... Christopher J. Pal, in Data Mining (Fourth Edition), 2017

> Read full chapter

Classification

Jiawei Han, ... Jian Pei, in Data Mining (Third Edition), 2012

> Read full chapter

Phishing, SMishing, and Vishing

In Mobile Malware Attacks and Defense, 2009

> Read full chapter

Hybrid Approach for Classification of Electroencephalographic Signals Using Time–Frequency Images With Wavelets and Texture Features

N.J. Sairamya, ... M.S.P. Subathra, in Intelligent Data Analysis for Biomedical Applications, 2019

12.2.8 Support Vector Machine

SVM is a powerful classifier used for both linear and nonlinear classification by changing the kernel functions utilized [28]. In SVM, by using the kernel functions the data are mapped into a higher dimensional feature space, in which a hyperplane separating the classes are found. The general solution for finding the hyperplane with kernel functions is given as:

(12.15)

where $\{xi,yi\}$ is the training data with classes yi belonging to $\{-1,1\}$ and K is the kernel function. The standard SVM classifier with a multilayer perceptron function is first trained in this work and used for epileptic classification.

> Read full chapter

Artificial intelligence methods for predictive image-based grading of human cancers

Gerardo Fernandez, ... Michael Donovan, in Artificial Intelligence and Deep Learning in Pathology, 2021

SVM-based methods: Survival-SVM, SVCR, and SVRc

Support vector machine, introduced in the early 1990s [110], is a classification algorithm that learns to distinguish between binary labels of given data. SVM is a linear method, extended to encapsulate nonlinear problems through projecting the data to a higher dimensional space. This method has enjoyed particular success in its ease of handling high-dimensional data and the interpretability of its linear models. This method was also extended to solving regression problems, known as support vector regression. Essentially, the SVM algorithm is an optimization algorithm that works by maximizing the margin of a data set and finding the separating hyperplane that neatly divides the data. The margin is the smallest distance between a data point and the separating hyperplane. Implicitly, we have assumed that the data we are trying to classify and learn are perfectly separable by a hyperplane, but if this is not the case, we can still find an approximation by optimizing the distance to the separating hyperplane (pink box), Fig. 9.16, with error variables (blue box), subject to constraints (red box).

SVCR, a simple extension of SVMs for survival analysis that accounts for censoring [112], ignores constraints where the prediction time is later than the censor time (right censoring, second line in red box). In this model, errors in events and non-events and in early and late predictions are penalized identically, which creates a problem in that late prediction of an event is generally a far more serious error than early prediction, having consequences for patient treatment decisions.

SVRc, an effective and efficient solution to this problem, improves the handling of censored data by applying separate penalty factors [113]. The algorithm applies the most severe penalty to late prediction of events, followed by a lesser penalty to early prediction of nonevents, then early prediction of events, and finally a negligible penalty is applied to late prediction of nonevents (censored data). Optimal values for these parameters may be found using a hyper-parameter tuning algorithm to learn the values over multiple data sets. We have successfully used particle swarm optimization [114] to set up these parameters.

> Read full chapter

Advances and challenges in fMRI and DTI techniques

Ranjeet Ranjan Jha, ... Arnav Bhavsar, in Intelligent Data Security Solutions for e-Health Applications, 2020

5.1 Traditional classifiers

In the past few decades, several machine learning-based classifiers have been utilized for the detection of brain disorders. Among different traditional classifiers, SVM [33] is one of the most popular techniques. SVM is a sort of supervised learning classifier, which is based on the concept of finding the optimally separating hyperplane. SVM has several variants with different kernel functions, including the sigmoid kernel, Gaussian RBF kernel, and polynomial kernel, which makes SVM as a nonlinear classifier. As SVM takes features as input, the feature extractor along with several hyperparameters of SVM plays a pivotal role in overall classification/detection accuracy.

Linear discriminant analysis (LDA) could be a generalization of Fisher's linear discriminant and is based on the concept of looking for a linear combination of features that partition two groups. For reducing the overall computational costs and avoiding the overfitting problem, LDA projects data into lower-dimensional space. There exists several other traditional classifiers, namely decision tree, random forest, K-nearest neighbor, etc., which can also be used for performing mental disease classification.

> Read full chapter