Dear readers,

We'll get right to it: This week we ask you to help Wikipedia. To protect our independence, we'll never run ads. We're sustained by donations averaging about $15. Only a tiny portion of our readers give. Now is the time we ask. If everyone reading this right now gave $3, our fundraiser would be done within an hour. That's right, the price of a cup of coffee is all we need. If Wikipedia is useful to you, please take one minute to keep it online and growing. We're a small non-profit with costs of a top site: servers, staff and programs. We serve millions of readers, but run on a fraction of what other top sites spend. We believe knowledge is a foundation. A foundation for human potential, for freedom, for opportunity. We believe everyone should have access to knowledge—for free, without restriction, without limitation. Please help us end the fundraiser and improve Wikipedia. *Thank you.*

Please help Wikipedia

● ONE-TIME ○ MONTHLY*

Select an amount ($)

3 5 10 20

30 50 100 Other

CREDIT CARD

PAYPAL

AMAZON

Problems donating? | Other w ays to give | Frequently asked questions | By donating, you are agreeing to our donor privacy policy. The Wikimedia Foundation is a nonprofit, tax-exempt organization. *If you make a recurring donation, you w ill be debited by the Wikimedia Foundation until you notify us to stop. We'll send you an email receipt for each payment, w hich w ill include a link to easy cancellation instructions.

Not logged in Talk Contributions Create account Log in

Article Talk Read Edit View history Search

WIKIPEDIA
The Free Encyclopedia

Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikipedia store

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page

Tools
What links here
Related changes
Upload file
Special pages
Permanent link
Page information
Wikidata item
Cite this page

Print/export
Create a book
Download as PDF
Printable version

Languages
العربية
Català
Čeština
Deutsch
Español
Euskara
فارسی
Français
한국어
Italiano
עברית
Nederlands
日本語
Polski
Português
Русский
Slovenščina
Suomi
Svenska

Support vector machine

From Wikipedia, the free encyclopedia

Not to be confused with Secure Virtual Machine.

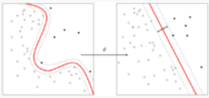
In machine learning, **support vector machines** (**SVMs**, also **support vector networks**[1]) are supervised learning models with associated learning algorithms that analyze data and recognize patterns, used for classification and regression analysis. Given a set of training examples, each marked for belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples into one category or the other, making it a non-probabilistic binary linear classifier. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on.

In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.

When data is not labeled, a supervised learning is not possible, and an unsupervised learning is required, that would find natural clustering of the data to groups, and map new data to these formed groups. The clustering algorithm which provides an improvement to the support vector machines is called **support vector clustering** (**SVC**[2]) is highly used in industrial applications either when data is not labeled or when only some data is labeled as a preprocessing for a classification pass; the clustering method was published.[2]

Contents [hide]

Machine learning and data mining



Problems
Classification · Clustering · Regression · Anomaly detection · Association rules · Reinforcement learning · Structured prediction · Feature engineering · Feature learning · Online learning · Semi-supervised learning · Unsupervised learning · Learning to rank · Grammar induction ·

Supervised learning
(**classification** • **regression**)
Decision trees · Ensembles (Bagging, Boosting, Random forest) · *k*-NN · Linear regression · Naive Bayes · Neural networks · Logistic regression · Perceptron · Relevance vector machine (RVM) · **Support vector machine (SVM)** ·

Clustering
BIRCH · Hierarchical · *k*-means · Expectation-maximization (EM) · DBSCAN · OPTICS · Mean-shift ·

Dimensionality reduction
Factor analysis · CCA · ICA · LDA · NMF · PCA · t-SNE ·

Structured prediction
Graphical models (Bayes net, CRF, HMM) ·

Anomaly detection
k-NN · Local outlier factor ·

Neural nets
Autoencoder · Deep learning · Multilayer perceptron · RNN · Restricted Boltzmann machine · SOM · Convolutional neural network ·

Theory
Bias-variance dilemma · Computational learning theory · Empirical risk minimization · PAC learning · Statistical learning · VC theory ·

Conferences and Journals
NIPS · ICML · JMLR · KDD · ICDM · SDM ·

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

Machine learning portal
V·T·E·

Definition [edit]

More formally, a support vector machine constructs a hyperplane or set of hyperplanes in a high- or infinite-dimensional space, which can be used for classification, regression, or other tasks. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training-data point of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier.

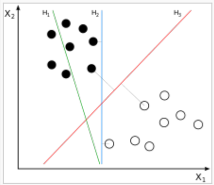
Whereas the original problem may be stated in a finite dimensional space, it often happens that the sets to discriminate are not linearly separable in that space. For this reason, it was proposed that the original finite-dimensional space be mapped into a much higher-dimensional space, presumably making the separation easier in that space. To keep the computational load reasonable, the mappings used by SVM schemes are designed to ensure that dot products may be computed easily in terms of the variables in the original space, by defining them in terms of a kernel function $k(x, y)$ selected to suit the problem.[3] The hyperplanes in the higher-dimensional space are defined as the set of points whose dot product with a vector in that space is constant. The vectors defining the hyperplanes can be chosen to be linear combinations with parameters α_i of images of feature vectors x_i that occur in the data base. With this choice of a hyperplane, the points x in the feature space that are mapped into the hyperplane are defined by the relation: $\textstyle\sum_i \alpha_i k(x_i, x) = \mathrm{constant}.$ Note that if $k(x, y)$ becomes small as y grows further away from x, each term in the sum measures the degree of closeness of the test point x to the corresponding data base point x_i. In this way, the sum of kernels above can be used to measure the relative nearness of each test point to the data points originating in one or the other of the sets to be discriminated. Note the fact that the set of points x mapped into any hy a result, allowing much more complex discrimination between sets which are not convex at all in the original space.

History [edit]

The original SVM algorithm was invented by Vladimir N. Vapnik and Alexey Ya. Chervonenkis in 1963. In 1992, Bernhard E. Boser, Isabelle M. Guyon and Vladimir N. Vapnik suggested a way to create nonlinear classifiers by applying the kernel trick to maximum-margin hyperplanes.[4] The current standard incarnation (soft margin) was proposed by Corinna Cortes and Vapnik in 1993 and published in 1995.[1]

Motivation [edit]

Classifying data is a common task in machine learning. Suppose some given data points each belong to one of two classes, and the goal is to decide which class a *new* data point will be in. In the case of support vector machines, a data point is viewed as a p-dimensional vector (a list of p numbers), and we want to know whether we can separate such points with a $(p-1)$-dimensional hyperplane. This is called a linear classifier. There are many hyperplanes that might classify the data. One reasonable choice as the best hyperplane is the one that represents the largest separation, or margin, between the two classes. So we choose the hyperplane so that the distance from it to the nearest data point on each side is maximized. If such a hyperplane exists, it is known as the *maximum-margin hyperplane* and the linear classifier it defines is known as a *maximum margin classifier*; or equivalently, the *perceptron of optimal stability*.[citation needed]



H_1 does not separate the classes. H_2 does, but only with a small margin. H_3 separates them with the maximum margin.

Linear SVM [edit]

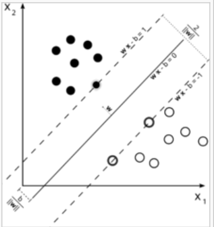
Given some training data $\mathcal{D}$, a set of n points of the form

$$\mathcal{D} = \left\{ (\mathbf{x}_i, y_i)\mid\mathbf{x}_i \in \mathbb{R}^p,\, y_i \in \{-1,1\}\right\}_{i=1}^n$$

where the y_i is either 1 or −1, indicating the class to which the point $\mathbf{x}_i$ belongs. Each $\mathbf{x}_i$ is a p-dimensional real vector. We want to find the maximum-margin hyperplane that divides the points having $y_i=1$ from those having $y_i=-1$. Any hyperplane can be written as the set of points $\mathbf{x}$ satisfying

$$\mathbf{w}\cdot\mathbf{x} - b=0,$$

where $\cdot$ denotes the dot product and $\mathbf{w}$ the (not necessarily normalized) normal vector to the hyperplane. The parameter $\tfrac{b}{\|\mathbf{w}\|}$ determines the offset of the hyperplane from the origin along the normal vector $\mathbf{w}$.



Maximum-margin hyperplane and margins for an SVM trained with samples from two classes. Samples on the margin are called the support vectors.

If the training data are linearly separable, we can select two hyperplanes in a way that they separate the data and there are no points between them, and then try to maximize their distance. The region bounded by them is called "the margin". These hyperplanes can be described by the equations

$$\mathbf{w}\cdot\mathbf{x} - b=1$$

and

$$\mathbf{w}\cdot\mathbf{x} - b=-1.$$

Geometrically, the distance between these two hyperplanes is $\tfrac{2}{\|\mathbf{w}\|}$, so to maximize the distance between the planes we want to minimize $\|\mathbf{w}\|$. As we also have to prevent data points from falling into the margin, we add the following constraint: for each i either

$$\mathbf{w}\cdot\mathbf{x}_i - b \ge 1\qquad\text{ for }\mathbf{x}_i$$ of the first class

or

$$\mathbf{w}\cdot\mathbf{x}_i - b \le -1\qquad\text{ for }\mathbf{x}_i$$ of the second.

This can be rewritten as:

$$y_i(\mathbf{w}\cdot\mathbf{x}_i - b) \ge 1, \quad \text{ for all } 1 \le i \le n.\qquad\qquad(1)$$

We can put this together to get the optimization problem:

Minimize (in $\mathbf{w}, b$)

$$\|\mathbf{w}\|$$

subject to (for any $i = 1, \dots, n$)

$$y_i(\mathbf{w}\cdot\mathbf{x_i} - b) \ge 1.$$

Primal form [edit]

The optimization problem presented in the preceding section is difficult to solve because it depends on $\|\mathbf{w}\|$, the norm of $\mathbf{w}$, which involves a square root. Fortunately it is possible to alter the minimization objective $\|\mathbf{w}\|$ by substituting it with $\tfrac{1}{2}\|\mathbf{w}\|^2$ (the factor of $\tfrac{1}{2}$ being used for mathematical convenience) without changing the solution (the minimum of the original and the modified equation have the same $\mathbf{w}$ and b). This is a quadratic programming optimization problem. More clearly:

$$\underset{(\mathbf{w},b)}{\operatorname{arg\,min}}\frac{1}{2}\|\mathbf{w}\|^2$$

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

subject to (for any $i = 1, \dots, n$)

$$y_i(\mathbf{w}\cdot\mathbf{x_i} - b) \ge 1.$$

By introducing KKT multipliers $\boldsymbol{\alpha}$, the previous constrained problem can be expressed as

$$\arg\min_{\mathbf{w},b}\max_{\boldsymbol{\alpha}\ge 0}\left\{\frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{n}\alpha_i[y_i(\mathbf{w}\cdot\mathbf{x_i} - b) - 1]\right\}$$

that is we look for a saddle point. In doing so all the points which can be separated as $y_i(\mathbf{w}\cdot\mathbf{x_i} - b) - 1 > 0$ do not matter since we must set the corresponding α_i to zero.

This problem can now be solved by standard quadratic programming techniques and programs. The "stationary" Karush–Kuhn–Tucker condition implies that the solution can be expressed as a linear combination of the training vectors

$$\mathbf{w} = \sum_{i=1}^{n}\alpha_i y_i \mathbf{x_i}.$$

Only a few α_i will be greater than zero. The corresponding $\mathbf{x_i}$ are exactly the *support vectors*, which lie on the margin and satisfy $y_i(\mathbf{w}\cdot\mathbf{x_i} - b) = 1$. From this one can derive that the support vectors also satisfy

$$\mathbf{w}\cdot\mathbf{x_i} - b = \frac{1}{y_i} = y_i \iff b = \mathbf{w}\cdot\mathbf{x_i} - y_i$$

which allows one to define the offset b. This estimate of b, the centerpoint of the division, depends only on the single pair y_i and x_i. We may get a more robust estimate of the center by averaging over all of the N_{SV} support vectors, if we believe the population mean is a good estimate of the midpoint, so in practice, b is often computed as:

$$b = \frac{1}{N_{SV}}\sum_{i=1}^{N_{SV}}(\mathbf{w}\cdot\mathbf{x_i} - y_i)$$

Dual form [edit]

Writing the classification rule in its unconstrained dual form reveals that the *maximum-margin hyperplane* and therefore the classification task is only a function of the *support vectors*, the subset of the training data that lie on the margin.

Using the fact that $\|\mathbf{w}\|^2 = \mathbf{w}^T\cdot\mathbf{w}$ and substituting $\mathbf{w} = \sum_{i=1}^{n}\alpha_i y_i \mathbf{x_i}$, one can show that the dual of the SVM reduces to the following optimization problem:

Maximize (in α_i)

$$\tilde{L}(\alpha) = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j \mathbf{x}_i^T\mathbf{x}_j = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j)$$

subject to (for any $i = 1, \dots, n$)

$$\alpha_i \ge 0,$$

and to the constraint from the minimization in b

$$\sum_{i=1}^{n}\alpha_i y_i = 0.$$

Here the kernel is defined by $k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i\cdot\mathbf{x}_j$.

W can be computed thanks to the α terms:

$$\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i.$$

Biased and unbiased hyperplanes [edit]

For simplicity reasons, sometimes it is required that the hyperplane pass through the origin of the coordinate system. Such hyperplanes are called *unbiased*, whereas general hyperplanes not necessarily passing through the origin are called *biased*. An unbiased hyperplane can be enforced by setting $b = 0$ in the primal optimization problem. The corresponding dual is identical to the dual given above without the equality constraint

$$\sum_{i=1}^{n}\alpha_i y_i = 0$$

Soft margin [edit]

In 1995, Corinna Cortes and Vladimir N. Vapnik suggested a modified maximum margin idea that allows for mislabeled examples.[1] If there exists no hyperplane that can split the "yes" and "no" examples, the *Soft Margin* method will choose a hyperplane that splits the examples as cleanly as possible, while still maximizing the distance to the nearest cleanly split examples. The method introduces non-negative slack variables, ξ_i, which measure the degree of misclassification of the data x_i

$$y_i(\mathbf{w}\cdot\mathbf{x_i} - b) \ge 1 - \xi_i \quad 1 \le i \le n. \qquad (2)$$

The objective function is then increased by a function which penalizes non-zero ξ_i, and the optimization becomes a trade off between a large margin and a small error penalty. If the penalty function is linear, the optimization problem becomes:

$$\arg\min_{\mathbf{w},\xi,b}\left\{\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i\right\}$$

subject to (for any $i = 1, \dots n$)

$$y_i(\mathbf{w}\cdot\mathbf{x_i} - b) \ge 1 - \xi_i, \qquad \xi_i \ge 0$$

Using the hinge function notation like that in MARS, this optimization problem can be rewritten as $\sum_i [1 - y_i(w\cdot x_i + b)]_+ + \lambda\|w\|^2$, wherein let

$$[1 - y_i(w\cdot x_i + b)]_+ = [\xi_i]_+ = \xi_i, \quad \lambda = 1/2C.$$

This constraint in (2) along with the objective of minimizing $\|\mathbf{w}\|$ can be solved using Lagrange multipliers as done above. One then has to solve the following problem:

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

$$\underset{\mathbf{w},\xi,b}{\arg\min}\ \max_{\boldsymbol{\alpha},\boldsymbol{\beta}} \left\{ \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\alpha_i[y_i(\mathbf{w}\cdot\mathbf{x_i} - b) - 1 + \xi_i] - \sum_{i=1}^{n}\beta_i\xi_i \right\}$$

with $\alpha_i, \beta_i \geq 0$.

Dual form [edit]

Maximize (in α_i)

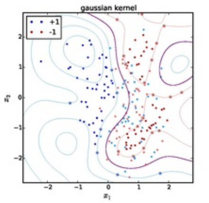
$$\tilde{L}(\alpha) = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j)$$

subject to (for any $i = 1, \dots, n$)

$$0 \leq \alpha_i \leq C,$$

and

$$\sum_{i=1}^{n}\alpha_i y_i = 0.$$

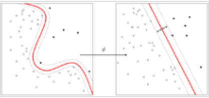


An example for a result of soft-margin SVM

The key advantage of a linear penalty function is that the slack variables vanish from the dual problem, with the constant *C* appearing only as an additional constraint on the Lagrange multipliers. For the above formulation and its huge impact in practice, Cortes and Vapnik received the 2008 ACM Paris Kanellakis Award.[5] Nonlinear penalty functions have been used, particularly to reduce the effect of outliers on the classifier, but unless care is taken the problem becomes non-convex, and thus it is considerably more difficult to find a global solution.

Nonlinear classification [edit]

The original optimal hyperplane algorithm proposed by Vapnik in 1963 was a linear classifier. However, in 1992, Bernhard E. Boser, Isabelle M. Guyon and Vladimir N. Vapnik suggested a way to create nonlinear classifiers by applying the kernel trick (originally proposed by Aizerman et al.[6]) to maximum-margin hyperplanes.[4] The resulting algorithm is formally similar, except that every dot product is replaced by a nonlinear kernel function. This allows the algorithm to fit the maximum-margin hyperplane in a transformed feature space. The transformation may be nonlinear and the transformed space high dimensional; thus though the classifier is a hyperplane in the high-dimensional feature space, it may be nonlinear in the original input space.



Kernel machine

If the kernel used is a Gaussian radial basis function, the corresponding feature space is a Hilbert space of infinite dimension. Maximum margin classifiers are well regularized, and previously it was widely believed that the infinite dimensions do not spoil the results. However, it has been shown that higher dimensions do increase the generalization error, although the amount is bounded.[7]

Some common kernels include:

- Polynomial (homogeneous): $k(\mathbf{x_i}, \mathbf{x_j}) = (\mathbf{x_i}\cdot\mathbf{x_j})^d$
- Polynomial (inhomogeneous): $k(\mathbf{x_i}, \mathbf{x_j}) = (\mathbf{x_i}\cdot\mathbf{x_j} + 1)^d$
- Gaussian radial basis function: $k(\mathbf{x_i}, \mathbf{x_j}) = \exp(-\gamma\|\mathbf{x_i} - \mathbf{x_j}\|^2)$, for $\gamma > 0$. Sometimes parametrized using $\gamma = 1/2\sigma^2$
- Hyperbolic tangent: $k(\mathbf{x_i}, \mathbf{x_j}) = \tanh(\kappa\mathbf{x_i}\cdot\mathbf{x_j} + c)$, for some (not every) $\kappa > 0$ and $c < 0$

The kernel is related to the transform $\varphi(\mathbf{x_i})$ by the equation $k(\mathbf{x_i}, \mathbf{x_j}) = \varphi(\mathbf{x_i})\cdot\varphi(\mathbf{x_j})$. The value **w** is also in the transformed space, with $\mathbf{w} = \sum_i \alpha_i y_i \varphi(\mathbf{x}_i)$. Dot products with **w** for classification can again be computed by the kernel trick, i.e. $\mathbf{w}\cdot\varphi(\mathbf{x}) = \sum_i \alpha_i y_i k(\mathbf{x}_i, \mathbf{x})$. However, there does not in general exist a value **w'** such that $\mathbf{w}\cdot\varphi(\mathbf{x}) = k(\mathbf{w}', \mathbf{x})$.

Properties [edit]

SVMs belong to a family of generalized linear classifiers and can be interpreted as an extension of the perceptron. They can also be considered a special case of Tikhonov regularization. A special property is that they simultaneously minimize the empirical *classification error* and maximize the *geometric margin*; hence they are also known as **maximum margin classifiers**.

A comparison of the SVM to other classifiers has been made by Meyer, Leisch and Hornik.[8]

Parameter selection [edit]

The effectiveness of SVM depends on the selection of kernel, the kernel's parameters, and soft margin parameter C. A common choice is a Gaussian kernel, which has a single parameter γ. The best combination of *C* and γ is often selected by a grid search with exponentially growing sequences of *C* and γ, for example, $C \in \{2^{-5}, 2^{-3}, \dots, 2^{13}, 2^{15}\}$; $\gamma \in \{2^{-15}, 2^{-13}, \dots, 2^{1}, 2^{3}\}$. Typically, each combination of parameter choices is checked using cross validation, and the parameters with best cross-validation accuracy are picked. Alternatively, recent work in Bayesian optimization can be used to select *C* and γ , often requiring the evaluation of far fewer parameter combinations than grid search. The final model, which is used for testing and for classifying new data, is then trained on the whole training set using the selected parameters.[9]

Issues [edit]

Potential drawbacks of the SVM are the following three aspects:

- Requires full labeling of input data
- Uncalibrated class membership probabilities
- The SVM is only directly applicable for two-class tasks. Therefore, algorithms that reduce the multi-class task to several binary problems have to be applied; see the multi-class SVM section.
- Parameters of a solved model are difficult to interpret.

Extensions [edit]

Support Vector Clustering (SVC) [edit]

SVC is a similar method that also builds on kernel functions but is appropriate for unsupervised learning and data-mining. It is considered a fundamental method in Data-Science.

Multiclass SVM [edit]

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

Multiclass SVM aims to assign labels to instances by using support vector machines, where the labels are drawn from a finite set of several elements.

The dominant approach for doing so is to reduce the single multiclass problem into multiple binary classification problems.[10] Common methods for such reduction include:[10] [11]

- Building binary classifiers which distinguish between (i) one of the labels and the rest (*one-versus-all*) or (ii) between every pair of classes (*one-versus-one*). Classification of new instances for the one-versus-all case is done by a winner-takes-all strategy, in which the classifier with the highest output function assigns the class (it is important that the output functions be calibrated to produce comparable scores). For the one-versus-one approach, classification is done by a max-wins voting strategy, in which every classifier assigns the instance to one of the two classes, then the vote for the assigned class is increased by one vote, and finally the class with the most votes determines the instance classification.
- Directed acyclic graph SVM (DAGSVM)[12]
- Error-correcting output codes[13]

Crammer and Singer proposed a multiclass SVM method which casts the multiclass classification problem into a single optimization problem, rather than decomposing it into multiple binary classification problems.[14] See also Lee, Lin and Wahba.[15][16]

Transductive support vector machines [edit]

Transductive support vector machines extend SVMs in that they could also treat partially labeled data in semi-supervised learning by following the principles of transduction. Here, in addition to the training set $\mathcal{D}$, the learner is also given a set

$$\mathcal{D}^\star = \{\mathbf{x}_i^\star | \mathbf{x}_i^\star \in \mathbb{R}^p\}_{i=1}^k$$

of test examples to be classified. Formally, a transductive support vector machine is defined by the following primal optimization problem:[17]

Minimize (in $\mathbf{w}, b, \mathbf{y}^\star$)

$$\frac{1}{2}\|\mathbf{w}\|^2$$

subject to (for any $i = 1, \dots, n$ and any $j = 1, \dots, k$)

$$y_i(\mathbf{w} \cdot \mathbf{x_i} - b) \geq 1,$$
$$y_j^\star(\mathbf{w} \cdot \mathbf{x_j^\star} - b) \geq 1,$$

and

$$y_j^\star \in \{-1, 1\}.$$

Transductive support vector machines were introduced by Vladimir N. Vapnik in 1998.

Structured SVM [edit]

SVMs have been generalized to structured SVMs, where the label space is structured and of possibly infinite size.

Regression [edit]

A version of SVM for regression was proposed in 1996 by Vladimir N. Vapnik, Harris Drucker, Christopher J. C. Burges, Linda Kaufman and Alexander J. Smola.[18] This method is called support vector regression (SVR). The model produced by support vector classification (as described above) depends only on a subset of the training data, because the cost function for building the model does not care about training points that lie beyond the margin. Analogously, the model produced by SVR depends only on a subset of the training data, because the cost function for building the model ignores any training data close to the model prediction. Another SVM version known as least squares support vector machine (LS-SVM) has been proposed by Suykens and Vandewalle.[19]

Training the original SVR means solving[20]

$$\text{minimize } \frac{1}{2}\|w\|^2$$
$$\text{subject to } \begin{cases} y_i - \langle w, x_i \rangle - b \leq \epsilon \\ \langle w, x_i \rangle + b - y_i \leq \epsilon \end{cases}$$

where x_i is a training sample with target value y_i. The inner product plus intercept $\langle w, x_i \rangle + b$ is the prediction for that sample, and ϵ is a free parameter that serves as a threshold: all predictions have to be within an ϵ range of the true predictions. Slack variables are usually added into the above to allow for errors and to allow approximation in the case the above problem is infeasible.

Interpreting SVM models [edit]

The SVM algorithm has been widely applied in the biological and other sciences. Permutation tests based on SVM weights have been suggested as a mechanism for interpretation of SVM models.[21][22] Support vector machine weights have also been used to interpret SVM models in the past.[23] Posthoc interpretation of support vector machine models in order to identify features used by the model to make predictions is a relatively new area of research with special significance in the biological sciences.

Implementation [edit]

The parameters of the maximum-margin hyperplane are derived by solving the optimization. There exist several specialized algorithms for quickly solving the QP problem that arises from SVMs, mostly relying on heuristics for breaking the problem down into smaller, more-manageable chunks.

Another approach is to use an interior point method that uses Newton-like iterations to find a solution of the Karush–Kuhn–Tucker conditions of the primal and dual problems.[24] Instead of solving a sequence of broken down problems, this approach directly solves the problem altogether. To avoid solving a linear system involving the large kernel matrix, a low rank approximation to the matrix is often used in the kernel trick.

Another common method is Platt's Sequential Minimal Optimization (SMO) algorithm, which breaks the problem down into 2-dimensional sub-problems that are solved analytically, eliminating the need for a numerical optimization algorithm and matrix storage. This algorithm is conceptually simple, easy to implement, generally faster, and has better scaling properties for difficult SVM problems.[25]

The special case of linear support vector machines can be solved more efficiently by the same kind of algorithms used to optimize its close cousin, logistic regression; this class of algorithms includes sub-gradient descent (e.g., PEGASOS[26]) and coordinate descent (e.g., LIBLINEAR[27]). LIBLINEAR has some attractive training time properties. Each convergence iteration takes time linear in the time taken to read the train data and the iterations also have a Q-Linear Convergence property, making the algorithm extremely fast.

The general kernel SVMs can also be solved more efficiently using sub-gradient descent (e.g. P-packSVM[28]), especially when parallelization is allowed.

Kernel SVMs are available in many machine learning toolkits, including LIBSVM, MATLAB, SAS, SVMlight, kernlab, scikit-learn, Shogun, Weka, Shark, JKernelMachines, OpenCV

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

and others.

Applications [edit]

SVMs can be used to solve various real world problems:

- SVMs are helpful in text and hypertext categorization as their application can significantly reduce the need for labeled training instances in both the standard inductive and transductive settings.
- Classification of images can also be performed using SVMs. Experimental results show that SVMs achieve significantly higher search accuracy than traditional query refinement schemes after just three to four rounds of relevance feedback.[citation needed]
- SVMs are also useful in medical science to classify proteins with up to 90% of the compounds classified correctly.
- Hand-written characters can be recognized using SVM.

See also [edit]

- In situ adaptive tabulation
- Kernel machines
- Fisher kernel
- Platt scaling
- Polynomial kernel
- Predictive analytics
- Regularization perspectives on support vector machines
- Relevance vector machine, a probabilistic sparse kernel model identical in functional form to SVM
- Sequential minimal optimization
- Space mapping
- Winnow (algorithm)

References [edit]

1. ^ a b c Cortes, C.; Vapnik, V. (1995). "Support-vector networks". *Machine Learning* **20** (3): 273. doi:10.1007/BF00994018.
2. ^ a b Ben-Hur, Asa, Horn, David, Siegelmann, Hava, and Vapnik, Vladimir; “Support vector clustering” (2001) Journal of Machine Learning Research, 2: 125-137.
3. ^ *Press, William H.; Teukolsky, Saul A.; Vetterling, William T.; Flannery, B. P. (2007). "Section 16.5. Support Vector Machines". *Numerical Recipes: The Art of Scientific Computing* (3rd ed.). New York: Cambridge University Press. ISBN 978-0-521-88068-8.
4. ^ a b Boser, B. E.; Guyon, I. M.; Vapnik, V. N. (1992). "A training algorithm for optimal margin classifiers". *Proceedings of the fifth annual workshop on Computational learning theory - COLT '92*. p. 144. doi:10.1145/130385.130401. ISBN 089791497X.
5. ^ ACM Website, Press release of March 17th 2009. http://www.acm.org/press-room/news-releases/awards-08-groupa
6. ^ Aizerman, Mark A.; Braverman, Emmanuel M.; and Rozonoer, Lev I. (1964). "Theoretical foundations of the potential function method in pattern recognition learning". *Automation and Remote Control* **25**: 821–837.
7. ^ Jin, Chi; Wang, Liwei (2012). *Dimensionality dependent PAC-Bayes margin bound*. Advances in Neural Information Processing Systems.
8. ^ Meyer, D.; Leisch, F.; Hornik, K. (2003). "The support vector machine under test". *Neurocomputing* **55**: 169. doi:10.1016/S0925-2312(03)00431-4.
9. ^ Hsu, Chih-Wei; Chang, Chih-Chung; and Lin, Chih-Jen (2003). *A Practical Guide to Support Vector Classification* (PDF) (Technical report). Department of Computer Science and Information Engineering, National Taiwan University.
10. ^ a b Duan, K. B.; Keerthi, S. S. (2005). "Which Is the Best Multiclass SVM Method? An Empirical Study". *Multiple Classifier Systems*. LNCS **3541**. pp. 278–285. doi:10.1007/11494683_28. ISBN 978-3-540-26306-7.
11. ^ Hsu, Chih-Wei; and Lin, Chih-Jen (2002). "A Comparison of Methods for Multiclass Support Vector Machines". *IEEE Transactions on Neural Networks*.
12. ^ Platt, John; Cristianini, N.; and Shawe-Taylor, J. (2000). "Large margin DAGs for multiclass classification". In Solla, Sara A.; Leen, Todd K.; and Müller, Klaus-Robert; eds. *Advances in Neural Information Processing Systems* (PDF). MIT Press. pp. 547–553.
13. ^ Dietterich, Thomas G.; and Bakiri, Ghulum; Bakiri (1995). "Solving Multiclass Learning Problems via Error-Correcting Output Codes" (PDF). *Journal of Artificial Intelligence Research, Vol. 2* **2**: 263–286. arXiv:cs/9501101. Bibcode:1995cs........1101D.
14. ^ Crammer, Koby; and Singer, Yoram (2001). "On the Algorithmic Implementation of Multiclass Kernel-based Vector Machines" (PDF). *J. of Machine Learning Research* **2**: 265–292.
15. ^ Lee, Y.; Lin, Y.; and Wahba, G. (2001). "Multicategory Support Vector Machines" (PDF). *Computing Science and Statistics 33*.
16. ^ Lee, Y.; Lin, Y.; Wahba, G. (2004). "Multicategory Support Vector Machines". *Journal of the American Statistical Association* **99** (465): 67. doi:10.1198/016214504000000098.
17. ^ Joachims, Thorsten; "Transductive Inference for Text Classification using Support Vector Machines", Proceedings of the 1999 International Conference on Machine Learning (ICML 1999), pp. 200-209.
18. ^ Drucker, Harris; Burges, Christopher J. C.; Kaufman, Linda; Smola, Alexander J.; and Vapnik, Vladimir N. (1997); "Support Vector Regression Machines", in *Advances in Neural Information Processing Systems 9, NIPS 1996*, 155–161, MIT Press.
19. ^ Suykens, Johan A. K.; Vandewalle, Joos P. L.; *Least squares support vector machine classifiers*, Neural Processing Letters, vol. 9, no. 3, Jun. 1999, pp. 293–300.
20. ^ Smola, Alex J.; Schölkopf, Bernhard (2004). "A tutorial on support vector regression" (PDF). *Statistics and Computing* **14** (3): 199–222.
21. ^ Bilwaj Gaonkar, Christos Davatzikos Analytic estimation of statistical significance maps for support vector machine based multi-variate image analysis and classification
22. ^ R. Cuingnet, C. Rosso, M. Chupin, S. Lehéricy, D. Dormont, H. Benali, Y. Samson and O. Colliot, Spatial regularization of SVM for the detection of diffusion alterations associated with stroke outcome, Medical Image Analysis, 2011, 15 (5): 729-737
23. ^ Statnikov, A., Hardin, D., & Aliferis, C. (2006). Using SVM weight-based methods to identify causally relevant and non-causally relevant variables. sign, 1, 4.
24. ^ Ferris, M. C.; Munson, T. S. (2002). "Interior-Point Methods for Massive Support Vector Machines". *SIAM Journal on Optimization* **13** (3): 783. doi:10.1137/S1052623400374379.
25. ^ John C. Platt (1998). *Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines* (PDF). NIPS.
26. ^ Shai Shalev-Shwartz; Yoram Singer; Nathan Srebro (2007). *Pegasos: Primal Estimated sub-GrAdient SOlver for SVM* (PDF). ICML.
27. ^ R.-E. Fan; K.-W. Chang; C.-J. Hsieh; X.-R. Wang; C.-J. Lin (2008). "LIBLINEAR: A library for large linear classification". *Journal of Machine Learning Research* **9**: 1871–1874.
28. ^ Zeyuan Allen Zhu; et al. (2009). *P-packSVM: Parallel Primal grAdient desCent Kernel SVM* (PDF). ICDM.

External links [edit]

- www.support-vector.net The key book about the method, "An Introduction to Support Vector Machines" with online software
- Burges, Christopher J. C.; A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery 2:121–167, 1998
- www.kernel-machines.org *(general information and collection of research papers)*
- www.support-vector-machines.org *(Literature, Review, Software, Links related to Support Vector Machines — Academic Site)*
- videolectures.net *(SVM-related video lectures)*
- Karatzoglou, Alexandros et al.; Support Vector Machines in R, Journal of Statistical Software April 2006, Volume 15, Issue 9.
- libsvm LIBSVM is a popular library of SVM learners

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

- liblinear liblinear is a library for large linear classification including some SVMs
- Shark Shark is a C++ machine learning library implementing various types of SVMs
- dlib dlib is a C++ library for working with kernel methods and SVMs
- SVM light is a collection of software tools for learning and classification using SVM.
- SVMJS live demo is a GUI demo for Javascript implementation of SVMs
- Gesture Recognition Toolkit contains an easy to use wrapper for libsvm

Bibliography [edit]

- Theodoridis, Sergios; and Koutroumbas, Konstantinos; "Pattern Recognition", 4th Edition, Academic Press, 2009, ISBN 978-1-59749-272-0
- Cristianini, Nello; and Shawe-Taylor, John; *An Introduction to Support Vector Machines and other kernel-based learning methods*, Cambridge University Press, 2000. ISBN 0-521-78019-5 *([1] SVM Book)*
- Huang, Te-Ming; Kecman, Vojislav; and Kopriva, Ivica (2006); *Kernel Based Algorithms for Mining Huge Data Sets*, in *Supervised, Semi-supervised, and Unsupervised Learning*, Springer-Verlag, Berlin, Heidelberg, 260 pp. 96 illus., Hardcover, ISBN 3-540-31681-7 [2]
- Kecman, Vojislav; *Learning and Soft Computing — Support Vector Machines, Neural Networks, Fuzzy Logic Systems*, The MIT Press, Cambridge, MA, 2001.[3]
- Schölkopf, Bernhard; and Smola, Alexander J.; *Learning with Kernels*, MIT Press, Cambridge, MA, 2002. ISBN 0-262-19475-9
- Schölkopf, Bernhard; Burges, Christopher J. C.; and Smola, Alexander J. (editors); *Advances in Kernel Methods: Support Vector Learning*, MIT Press, Cambridge, MA, 1999. ISBN 0-262-19416-3. [4]
- Shawe-Taylor, John; and Cristianini, Nello; *Kernel Methods for Pattern Analysis*, Cambridge University Press, 2004. ISBN 0-521-81397-2 *([5] Kernel Methods Book)*
- Steinwart, Ingo; and Christmann, Andreas; *Support Vector Machines*, Springer-Verlag, New York, 2008. ISBN 978-0-387-77241-7 *([6] SVM Book)*
- Tan, Peter Jing; and Dowe, David L. (2004); *MML Inference of Oblique Decision Trees*, Lecture Notes in Artificial Intelligence (LNAI) 3339, Springer-Verlag, pp. 1082-1088. (This paper uses minimum message length (MML) and actually incorporates probabilistic support vector machines in the leaves of decision trees.)
- Vapnik, Vladimir N.; *The Nature of Statistical Learning Theory*, Springer-Verlag, 1995. ISBN 0-387-98780-0
- Vapnik, Vladimir N.; and Kotz, Samuel; *Estimation of Dependences Based on Empirical Data*, Springer, 2006. ISBN 0-387-30865-2, 510 pages [this is a reprint of Vapnik's early book describing philosophy behind SVM approach. The 2006 Appendix describes recent development].
- Fradkin, Dmitriy; and Muchnik, Ilya; *Support Vector Machines for Classification* in Abello, J.; and Carmode, G. (Eds); *Discrete Methods in Epidemiology*, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, volume 70, pp. 13–20, 2006. [7]. Succinctly describes theoretical ideas behind SVM.
- Bennett, Kristin P.; and Campbell, Colin; *Support Vector Machines: Hype or Hallelujah?*, SIGKDD Explorations, 2, 2, 2000, 1–13. [8]. Excellent introduction to SVMs with helpful figures.
- Ivanciuc, Ovidiu; *Applications of Support Vector Machines in Chemistry*, in *Reviews in Computational Chemistry*, Volume 23, 2007, pp. 291–400. Reprint available: [9]
- Catanzaro, Bryan; Sundaram, Narayanan; and Keutzer, Kurt; *Fast Support Vector Machine Training and Classification on Graphics Processors*, in *International Conference on Machine Learning*, 2008 [10]
- Campbell, Colin; and Ying, Yiming; *Learning with Support Vector Machines*, 2011, Morgan and Claypool. ISBN 978-1-60845-616-1. *[11]*
- Ben-Hur, Asa, Horn, David, Siegelmann, Hava, and Vapnik, Vladimir; “Support vector clustering” (2001) Journal of Machine Learning Research, 2: 125-137.

Authority control	GND: 4505517-8 ·

Categories: Support vector machines | Classification algorithms | Statistical classification

This page w as last modified on 17 November 2015, at 06:10.

Text is available under the Creative Commons Attribution-ShareAlike License; additional terms may apply. By using this site, you agree to the Terms of Use and Privacy Policy. Wikipedia® is a registered trademark of the Wikimedia Foundation, Inc., a non-profit organization.

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com