Kernel Machines

Theory And Practice

Marco Del Vecchio
marco@delvecchiomarco.com

Warwick Manufacturing Group
University of Warwick

26/07/2017

Outline I

Outline I

Outline II

Training Data

Definition (Training Data)

Let the training data be denoted as

$$\{(\boldsymbol{x}^{(n)}, y^{(n)}) \in \mathbb{R}^D \times \mathbb{R} \mid n = 1, \ldots, N\},$$

in the case where the response variable is a scalar, and

$$\{(\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)}) \in \mathbb{R}^D \times \mathbb{R}^K \mid n = 1, \ldots, N\}$$

when it is a multidimensional vector, where N denotes the total number of training examples.

Loss Functions

Definition (Loss Function)

A loss function $L(\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{w})$ is a single, overall measure of loss incurred in taking any of the available decisions or actions. In particular, in this context, we define a loss function to be a mapping that quantifies how unhappy we would be if we used $\boldsymbol{w}$ to make a prediction on $\boldsymbol{X}$ when the correct output is $\boldsymbol{y}$.

Generalisation And Overfitting I

Definition (Overfitting)

A model, is said to overfit to the data if it does not generalise to out-of-sample cases although it fits the training data very well. More specifically, a model which explains the random error or noise in the data instead of underlying relationship is said to be overfitting.

Generalisation And Overfitting II

Ideally, we would like to choose a model which performs best (i.e. minimises the loss) on new unseen data.
That is, we would like a model that can generalises well beyond the data used during training. However, by the very nature of the problem, unseen data is not available.

Generalisation And Overfitting III

One solution to this is to train and validate the model on two different datasets.

Definition (K-Fold Cross Validation)

In k-fold cv, the original sam- ple is randomly partitioned into k equal sized subsamples. Of the k subsamples, a single subsample is retained as the validation data for testing the model, and the remaining $k1$ subsamples are used as training data. The cross-validation process is then repeated k times (the folds), with each of the k subsamples used exactly once as the validation data. The k results from the folds are then averaged to produce a single measure of performance.

Outline I

Outline II

An Introduction I

Many linear parametric models can be re-cast into an equivalent "dual representation" in which the predictions are based on linear combinations of a *kernel function* evaluated at the training data points.
As we shall see, for models which are based on a fixed non-linear feature space mapping $\boldsymbol{\phi}(\boldsymbol{x})$, the kernel function is given by

$$k(\boldsymbol{x}^{(n)}, \boldsymbol{x}^{(m)}) = \boldsymbol{\phi}(\boldsymbol{x}^{(n)})^T \boldsymbol{\phi}(\boldsymbol{x}^{(m)})$$

An Introduction II

The kernel concept was introduced into the field of pattern recognition by Aizerman et al. (1964) however, it was neglected for many years until Boser et al. (1992) popularised it by giving rise to the technique of *large margin classifiers* which lead to *support vector machines* introduced by Vapnik (1995).

An Introduction III

The concept of a kernel formulated as an inner product in a feature space allows us to build interesting extensions of many algorithms by making use of the kernel trick, also known as *kernel substitution*.
In this case, the kernel can be written in form of a feature map $\boldsymbol{\phi} : \mathcal{X} \to \mathcal{V}$ which satisfies

$$k(\boldsymbol{x}^{(n)}, \boldsymbol{x}^{(m)}) = \langle \boldsymbol{\phi}(\boldsymbol{x}^{(n)}), \boldsymbol{\phi}(\boldsymbol{x}^{(m)}) \rangle_{\mathcal{V}}$$

for $\boldsymbol{x}^{(n)}$ and $\boldsymbol{x}^{(m)} \in \mathcal{X}$, where $\langle \cdot, \cdot \rangle_{\mathcal{V}}$ is a proper inner product.

An Introduction IV

Viewing kernel functions in this way gives rise to a powerful result

Result

An implicitly defined function ϕ exists whenever the space $\mathcal{X}$ can be equipped with a suitable measure ensuring the function k satisfies Mercer's condition.
Hence, an explicit representation for ϕ is not necessary, as long as $\mathcal{V}$ is an inner product space.

An Introduction V

The general idea is that, if we have an algorithm formulated in such a way that the input vector $\boldsymbol{x}$ enters only in the form of scalar products, then we can replace that scalar product a kernel function.

An Introduction VI

Question: What if the algorithm to which we want to apply the kernel trick does not posses the above stated requirement?

Dual Representations I

Many linear models for regression and classification can be reformulated in terms of a dual representation in which the kernel function arises naturally.

Dual Representations II

Definition ((Lagrangian) Dual Problem)

Given a Lagrangian $\mathcal{L}(\boldsymbol{x}, \{\boldsymbol{\lambda}_s\}_{s=1}^M)$ where $\boldsymbol{x}$ is the input vector and $\{\boldsymbol{\lambda}_s\}_{s=1}^M$ are the non-negative Lagrangian multipliers, the dual problem can be derived by:

1. Solving for some primal variable values that minimize the Lagrangian
2. Writing the solution in terms of *primal variables* as functions of the Lagrange multipliers called *dual variables*
3. Formulate the dual problem to maximize the objective function with respect to the dual variables under the derived constraints on the dual variables (including non-negativity).

Dual Representations III

Example (Dual Representation of Ridge Regression with Basis Functions)

Consider the linear regression model $h(\boldsymbol{x}; \boldsymbol{w}) = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x})$ whose parameters are determined by minimizing a regularized sum-of-squares error function given by

$$L(\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{w}, \lambda) = \frac{1}{2} \sum_{n=1}^{N} (\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}^{(n)}) - y^{(n)})^2 + \frac{\lambda}{2} \boldsymbol{w}^T \boldsymbol{w}$$

where $\lambda \geq 0$ gives the regularisation strength, and $\boldsymbol{\phi} : \mathbb{R}^{D+1} \to \mathbb{R}^{M+1}$.

Dual Representations IV

Example (Dual Representation of Ridge Regression with Basis Functions (Cont.))

Our objective is then so solve

$$\arg\min_{\boldsymbol{w},\lambda\geq 0} L(\boldsymbol{X},\boldsymbol{y},\boldsymbol{w},\lambda).$$

Taking derivatives w.r.t. $\boldsymbol{w}$, we find that at the optimum, $\boldsymbol{w}$ satisfies

$$\boldsymbol{w} = -\frac{1}{\lambda}\sum_{n=1}^{N}(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}^{(n)}) - y^{(n)})\boldsymbol{\phi}(\boldsymbol{x}^{(n)}) = \sum_{n=1}^{N} a_n\boldsymbol{\phi}(\boldsymbol{x}^{(n)}) = \boldsymbol{\Phi}^T\boldsymbol{a}$$

Dual Representations V

Example (Dual Representation of Ridge Regression with Basis Functions (Cont.))

where $\boldsymbol{\Phi}$ is a $N \times (M+1)$ matrix whose n^{th} row is given by $\boldsymbol{\phi}(\boldsymbol{x}^{(n)})^T$. Hence the vector $\boldsymbol{a} = (a_1, \ldots, a_N)^T$ is defined as

$$a_n = -\frac{1}{\lambda}(\boldsymbol{w}^T \boldsymbol{x}^{(n)} - y^{(n)})$$

Now, we can give rise to the dual formulation of this optimisation problem by writing the loss function in terms of $\boldsymbol{a}$ by substituting $\boldsymbol{w}$ with the expression derived previously.

Dual Representations VI

Example (Dual Representation of Ridge Regression with Basis Functions (Cont.))

Doing so gives rise to

$$L(\boldsymbol{a}) = \frac{1}{2}\boldsymbol{a}^T\boldsymbol{\Phi}\boldsymbol{\Phi}^T\boldsymbol{\Phi}\boldsymbol{\Phi}^T\boldsymbol{a} - \boldsymbol{a}^T\boldsymbol{\Phi}\boldsymbol{\Phi}^T\boldsymbol{y} + \frac{1}{2}\boldsymbol{y}^T\boldsymbol{y} + \frac{\lambda}{2}\boldsymbol{a}^T\boldsymbol{\Phi}\boldsymbol{\Phi}^T\boldsymbol{a}$$

where $\boldsymbol{y} = (y_1, \ldots, y_N)^T$. Now, define the Gram matrix $\boldsymbol{K}$ as $\boldsymbol{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^T$, which is an $N \times N$ symmetric matrix with elements

$$K_{nm} = \boldsymbol{\phi}(\boldsymbol{x}^{(n)})^T\boldsymbol{\phi}(\boldsymbol{x}^{(m)}) = k(\boldsymbol{x}^{(n)}, \boldsymbol{x}^{(m)})$$

Dual Representations VII

Example (Dual Representation of Ridge Regression with Basis Functions (Cont.))

Now, writing the regularised sum-of-squares in terms of K yields

$$L(\boldsymbol{a}) = \frac{1}{2}\boldsymbol{a}^T\boldsymbol{K}\boldsymbol{K}^T\boldsymbol{a} - \boldsymbol{a}^T\boldsymbol{K}\boldsymbol{y} + \frac{1}{2}\boldsymbol{y}^T\boldsymbol{y} + \frac{\lambda}{2}\boldsymbol{a}^T\boldsymbol{K}\boldsymbol{a}$$

Thus we see that the dual formulation allows the solution to the least-squares problem to be expressed entirely in terms of the kernel function $k(\boldsymbol{x}^{(n)}, \boldsymbol{x}^{(m)})$.

Dual Representations VIII

Example (Dual Representation of Ridge Regression with Basis Functions (Cont.))

To conclude, setting the partial derivative of $L(\boldsymbol{a})$ w.r.t. $\boldsymbol{a}$ to zero we yields the following solution

$$\boldsymbol{a} = (\boldsymbol{K} + \lambda \boldsymbol{I}_N)^{-1} \boldsymbol{y}.$$

If we then substitute this back into the linear regression model, we obtain the following

$$h(\boldsymbol{x}; \boldsymbol{w}) = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}) = \boldsymbol{a}^T \boldsymbol{\Phi} \boldsymbol{\phi}(\boldsymbol{x}) = \boldsymbol{k}(\boldsymbol{x})^T (\boldsymbol{K} + \lambda \boldsymbol{I}_N)^{-1} \boldsymbol{y}$$

where $\boldsymbol{k}(\boldsymbol{x}) = (k_1(\boldsymbol{x}), \ldots, k_N(\boldsymbol{x}))$ with elements $k_n(\boldsymbol{x}) = k(\boldsymbol{x}^{(n)}, \boldsymbol{x})$

Constructing Kernels I

In order to exploit the kernel trick, we need to be able to construct valid kernel functions. Generally, there are two approaches:

1. Choose a feature space mapping $\boldsymbol{\phi}(\boldsymbol{x})$ and then use this to find the corresponding kernel function.
2. Construct the kernel function directly.

Constructing Kernels II

Example (From Feature Map to Kernel)

Suppose that we want to find the kernel which corresponds to the following feature mapping $\boldsymbol{\phi}(\boldsymbol{x}) = (1, x_1, x_2, x_1 x_2)^T$ then, the kernel corresponding to this feature mapping is

$$\begin{aligned} k(\boldsymbol{x}, \boldsymbol{x}') &= \boldsymbol{\phi}(\boldsymbol{x})^T \boldsymbol{\phi}(\boldsymbol{x}') \\ &= (1, x_1, x_2, x_1 x_2)^T (1, x_1', x_2', x_1' x_2') \\ &= 1 + x_1' + x_2' + x_1' x_2' \\ &+ x_1 + x_1 x_1' + x_1 x_2' + x_1 x_1' x_2' \\ &+ x_2 + x_2 x_1' + x_2 x_2' + x_2 x_1' x_2' \\ &+ x_1 x_2 + x_1 x_2 x_1' + x_1 x_2 x_2' + x_1 x_2 x_1' x_2'. \end{aligned}$$

Constructing Kernels III

Example (From Kernel to Feature Map)

Let $\boldsymbol{x} \in \mathbb{R}^2$ and suppose that we want to know what is the feature map associated with the following kernel, and if this is indeed a valid one:

$$\begin{aligned} k(\boldsymbol{x}, \boldsymbol{x}') &= (\boldsymbol{x}^T \boldsymbol{x}')^2 = (x_1 x_1', x_2 x_2')^2 \\ &= (x_1^2 x_1'^2 + 2 x_1 x_1' x_2 x_2' + x_2 x_2'^2) \\ &= (x_1, \sqrt{2} x_1 x_2, x_2^2)^T (x_1', \sqrt{2} x_1' x_2', x_2'^2) \\ &= \boldsymbol{\phi}(\boldsymbol{x}) \boldsymbol{\phi}(\boldsymbol{x}')^T. \end{aligned}$$

Constructing Kernels IV

Example (From Kernel to Feature Map (Cont.))

We see that the feature mapping takes the form

$$\boldsymbol{\phi}(\boldsymbol{x}) = (x_1, \sqrt{2}x_1x_2, x_2^2)^T.$$

and therefore comprises all possible second order terms, with a specific weighting between them. Furthermore, we can see that the kernel can be written in terms of an inner product in s space defined by $\boldsymbol{\phi}(\boldsymbol{x})$.

Constructing Kernels V

More generally, however, we need a simple way to test whether a function constitutes a valid kernel without having to construct the function $\boldsymbol{\phi}(\boldsymbol{x})$ explicitly.

Constructing Kernels VI

Result

A necessary and sufficient condition for a function $k(\boldsymbol{x}^{(n)}, \boldsymbol{x}^{(m)})$ *to be a valid kernel (Shawe Taylor and Cristianini, 2004) is that the Gram matrix* $\boldsymbol{K}$*, whose elements are given by* $k(\boldsymbol{x}^{(n)}, \boldsymbol{x}^{(m)})$*, should be positive semi-definite for all possible choices of the set* $\{\boldsymbol{x}^{(n)}, \boldsymbol{x}^{(m)}\}$

Constructing Kernels VII

Therefore, we can use a kernel $k(\cdot, \cdot)$ without having to do any calculation in the underlying feature space !

Constructing Kernels VIII

Result (Gaussian Kernel)

Consider the Gaussian kernel, which belongs to the class of Radial Basis Function (RBF) kernels

$$k(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\gamma||\boldsymbol{x} - \boldsymbol{x}'||^2\right)$$

then, the underlying space is infinite dimensional.

Constructing Kernels IX

Proof.

Take the case where $\boldsymbol{x} \in \mathbb{R}$ and $\gamma = 1$ then,

$$
\begin{aligned}
k(x, x') &= \exp\left(-(x - x')^2 \right) \\
&= \exp(x^2) \exp(x'^2) \exp(2xx') \\
&= \exp(x^2) \exp(x'^2) \sum_{k=0}^{\infty} \frac{2^k (x)^k (x')^k}{k!} \\
&= \exp(x^2) \sum_{k=0}^{\infty} \frac{2^{\frac{k}{2}} (x)^k}{\sqrt{k!}} \exp(x'^2) \sum_{k=0}^{\infty} \frac{2^{\frac{k}{2}} (x')^k}{\sqrt{k!}}
\end{aligned}
$$

Constructing Kernels X

One powerful technique for constructing new kernels is to build them out of simpler kernels as building blocks. This can be done using the following properties.

Let $c > 0$ be a constant, $f(\cdot)$ any function, $q(\cdot)$ a polynomial with non-negative coefficients, $\boldsymbol{\phi} : \mathbb{R}^D \to \mathbb{R}^M$, $k_3(\cdot, \cdot)$ a valid kernel in $\mathbb{R}^M$, and $\boldsymbol{A}$ a positive semidefinite matrix.

Constructing Kernels XI

Given valid kernels $k_1(\boldsymbol{x}, \boldsymbol{x}')$ and $k_2(\boldsymbol{x}\boldsymbol{x}')$ the following are also valid kernels

- $k(\boldsymbol{x}, \boldsymbol{x}') = ck_1(\boldsymbol{x}, \boldsymbol{x}')$
- $k(\boldsymbol{x}, \boldsymbol{x}') = f(\boldsymbol{x})k_1(\boldsymbol{x}, \boldsymbol{x}')f(\boldsymbol{x}')$
- $k(\boldsymbol{x}, \boldsymbol{x}') = q(k_1(\boldsymbol{x}, \boldsymbol{x}'))$
- $k(\boldsymbol{x}, \boldsymbol{x}') = \exp(k_1(\boldsymbol{x}, \boldsymbol{x}'))$
- $k(\boldsymbol{x}, \boldsymbol{x}') = k_1(\boldsymbol{x}, \boldsymbol{x}') + k_2(\boldsymbol{x}, \boldsymbol{x}')$
- $k(\boldsymbol{x}, \boldsymbol{x}') = k_1(\boldsymbol{x}, \boldsymbol{x}')k_2(\boldsymbol{x}, \boldsymbol{x}')$
- $k(\boldsymbol{x}, \boldsymbol{x}') = k_3(\boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{x}'))$
- $k(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{x}^T \boldsymbol{A}\boldsymbol{x}'$

Outline I

Outline II

An Overview I

Support Vector Machines (SVMs) belong to a class of non probabilistic classification models called maximum margin classifiers.

Definition (Margin)

We define the margin as the smallest distance between the decision boundary and any of the samples.

An Overview II

An important property of support vector machines is that the determination of the model parameters corresponds to a convex optimization problem, and so any local solution is also a global optimum.

Hard Margin SVMs For Classification I

Consider the class of linear models for binary classification problems specified by

$$h(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}) + b$$

where $\boldsymbol{\phi} : \mathbb{R}^D \to \mathbb{R}^M$ denotes a fixed feature-space transformation, and $b \in \mathbb{R}$ denotes the bias term.

Hard Margin SVMs For Classification II

Consider a data set given by

$$\{(\boldsymbol{x}^{(n)}, y^{(n)}) \in \mathbb{R}^D \times \{-1, +1\} \mid n = 1, \ldots, N\},$$

and *assume* that is it linearly separable in the feature space specified by $\boldsymbol{\phi} : \mathbb{R}^D \to \mathbb{R}^M$.

Hard Margin SVMs For Classification III

by definition of linear separability, there exist at least one choice of the parameters $\boldsymbol{w}$ and b such that

$$h(\boldsymbol{x}^{(n)}) = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}^{(n)}) + b = \begin{cases} > 0 & \text{if} \quad y^{(n)} = +1 \\ < 0 & \text{if} \quad y^{(n)} = -1 \end{cases}$$

So that

$$h(\boldsymbol{x}^{(n)}) y^{(n)} > 0$$

for all training data points.

Hard Margin SVMs For Classification IV

> In support vector machines the decision boundary is chosen to be the one for which the margin is maximized.

The maximum margin solution can be motivated using computational learning theory, also known as statistical learning theory.

Hard Margin SVMs For Classification V

the perpendicular distance of a point $\boldsymbol{x}$ from a hyperplane defined by $h(\boldsymbol{x}) = 0$ where $h(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}) + b$ is given by

$$\frac{|h(\boldsymbol{x})|}{||\boldsymbol{w}||_2}.$$

Furthermore, we are only interested in solutions for which all data points are correctly classified, so that $h(\boldsymbol{x}^{(n)})y^{(n)} > 0$

Hard Margin SVMs For Classification VI

Thus the distance of an observation $\boldsymbol{x}^{(n)}$ to the decision surface under perfect classification is given by

$$\frac{|h(\boldsymbol{x}^{(n)})y^{(n)}|}{||\boldsymbol{w}||_2} = \frac{|(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}^{(n)}) + b)y^{(n)}|}{||\boldsymbol{w}||_2}$$

Hard Margin SVMs For Classification VII

The margin, is the perpendicular distance to the closest point $\boldsymbol{x}^{(n)}$ from decision boundary. Thus, the maximum margin solution is found by solving

$$\arg\max_{\boldsymbol{w},b} \left\{ \frac{1}{||\boldsymbol{w}||_2} \min_n \{(\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}^{(n)}) + b) y^{(n)}\} \right\}.$$

Hard Margin SVMs For Classification VIII

Direct solutions to this optimisation problem would be very complex, and so we shall impose that

$$(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}^{(n)}) + b)y^{(n)} = 1$$

for the point that is closest to the surface. So that, all data points will satisfy the constraints

$$(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}^{(n)}) + b)y^{(n)} \geq 1, \quad \forall\, n = 1, 2, \ldots, N.$$

Hard Margin SVMs For Classification IX

So, the optimisation problem that needs to be solved is given by

$$\begin{aligned} &\arg\max_{\boldsymbol{w},b} \frac{1}{||\boldsymbol{w}||_2} \\ &\quad \text{s.t. } (\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}^{(n)}) + b)y^{(n)} \geq 1, \quad \forall\, n = 1, 2, \ldots, N. \end{aligned}$$

which is equivalent to

$$\begin{aligned} &\arg\min_{\boldsymbol{w},b} \frac{1}{2}||\boldsymbol{w}||_2^2 \\ &\quad \text{s.t. } (\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}^{(n)}) + b)y^{(n)} \geq 1, \quad \forall\, n = 1, 2, \ldots, N. \end{aligned}$$

Hard Margin SVMs For Classification X

The above optimisation problem is an example of a *quadratic programming problem* in which we are minimizing a quadratic function subject to a set of linear inequality constraints.

Hard Margin SVMs For Classification XI

In order to solve this constrained optimization problem, we introduce Lagrange multipliers $\{\alpha_n \geq 0\}_{n=1}^{N}$, giving the Lagrange function

$$\mathcal{L}(\boldsymbol{w}, \boldsymbol{\alpha}, b) = \frac{1}{2}||\boldsymbol{w}||_2^2 - \sum_{n=1}^{N} \alpha_n((\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}^{(n)}) + b)y^{(n)} - 1)$$

where $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_N)^T$. Note the minus sign is in front of the Lagrangian multipliers because we are minimising w.r.t. $\boldsymbol{w}$ and b, but we are maximising with respect to α since we want as the classifications to be correct, that is we want

$$(\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}^{(n)}) + b)y^{(n)} \geq 1.$$

Hard Margin SVMs For Classification XII

Setting the derivatives of $\mathcal{L}(\boldsymbol{w}, \boldsymbol{\alpha}, b)$ with respect to $\boldsymbol{w}$ and b equal to zero, we obtain the following two conditions

$$\frac{\partial \mathcal{L}(\boldsymbol{w}, \boldsymbol{\alpha}, b)}{\partial \boldsymbol{w}} = 0 \implies \boldsymbol{w} = \sum_{n=1}^{N} \alpha_n \boldsymbol{\phi}(\boldsymbol{x}^{(n)}) y^{(n)}$$

$$\frac{\partial \mathcal{L}(\boldsymbol{w}, \boldsymbol{\alpha}, b)}{\partial b} = 0 \implies 0 = \sum_{n=1}^{N} \alpha_n y^{(n)}.$$

Hard Margin SVMs For Classification XIII

Using the conditions above to get rid of the dependencies on $\boldsymbol{w}$ and b from $\mathcal{L}(\boldsymbol{w}, \boldsymbol{\alpha}, b)$ leads to the *dual formulation* of the optimisation problem:

$$\begin{aligned}
&\arg\max_{\boldsymbol{\alpha}} \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \alpha_n \alpha_m \boldsymbol{\phi}(\boldsymbol{x}^{(n)})^T \boldsymbol{\phi}(\boldsymbol{x}^{(m)}) y^{(n)} y^{(m)} \\
&\quad \text{s.t. } \alpha_n \geq 0, \quad \forall\, n = 1, 2, \ldots, N, \\
&\qquad \sum_{n=1}^{N} \alpha_n y^{(n)} = 0.
\end{aligned}$$

Hard Margin SVMs For Classification XIV

We note that the data enters the function to optimise only in the form of an inner product in the feature space specified by $\boldsymbol{\phi} : \mathbb{R}^D \to \mathbb{R}^M$. Hence, we can apply the *kernel trick* by replacing $\boldsymbol{\phi}(\boldsymbol{x}^{(n)})^T \boldsymbol{\phi}(\boldsymbol{x}^{(m)})$ with

$$k(\boldsymbol{x}^{(n)}, \boldsymbol{x}^{(m)}).$$

Hard Margin SVMs For Classification XV

In order to classify new data points using the trained model, we evaluate the sign of $h(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}) + b$ which can be expressed in terms of the kernel function and the Lagrange multipliers by substituting for

$$\boldsymbol{w} = \sum_{n=1}^{N} \alpha_n \boldsymbol{\phi}(\boldsymbol{x}^{(n)}) y^{(n)}$$

giving

$$h(\boldsymbol{x}) = \sum_{n=1}^{N} \alpha_n k(\boldsymbol{x}, \boldsymbol{x}^{(n)}) y^{(n)} + b.$$

Hard Margin SVMs For Classification XVI

Result (Hard Margin SVM)

Consider an Hard Margin SVM where the decision boundary is modelled as

$$h(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}) + b,$$

where $\boldsymbol{\phi} : \mathbb{R}^D \to \mathbb{R}^M$ *denotes a fixed feature-space transformation, and* $b \in \mathbb{R}$ *denotes the bias term. Let the training data be of the form*

$$\{(\boldsymbol{x}^{(n)}, y^{(n)}) \in \mathbb{R}^D \times \{-1, +1\} \mid n = 1, \ldots, N\}.$$

Hard Margin SVMs For Classification XVII

Result (Hard Margin SVM (Cont.))

Assume that the data is linearly separable in the feature space specified by $\boldsymbol{\phi}$, and impose that

$$(\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}^{(n)}) + b) y^{(n)} \geq 1, \quad \forall\, n = 1, 2, \ldots, N.$$

Hard Margin SVMs For Classification XVIII

Result (Hard Margin SVM (Cont.))

Then, the set of parameters $\boldsymbol{w}$ *and* b *which achieve perfect classification can be found by solving the dual problem given by*

$$\begin{aligned} \arg\max_{\boldsymbol{\alpha}} & \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \alpha_n \alpha_m k(\boldsymbol{x}^{(n)}, \boldsymbol{x}^{(m)}) y^{(n)} y^{(m)} \\ s.t. \ & \alpha_n \geq 0, \quad \forall \ n = 1, 2, \ldots, N, \\ & \sum_{n=1}^{N} \alpha_n y^{(n)} = 0. \end{aligned} \tag{3.1}$$

where $k(\boldsymbol{x}^{(n)}, \boldsymbol{x}^{(m)})$ *is a valid kernel.*

Hard Margin SVMs For Classification XIX

Result (Hard Margin SVM (Cont.))

Finally, we can make predictions by computing

$$h(\boldsymbol{x}) = \sum_{n=1}^{N} \alpha_n k(\boldsymbol{x}, \boldsymbol{x}^{(n)}) y^{(n)} + b.$$

Soft Margin SVMs For Classification I

So far, we have assumed that the training data points are linearly separable in the feature space given by $\boldsymbol{\phi}$.
In practice, however, the training data might not be linearly separable neither in the input nor in the feature space.

Soft Margin SVMs For Classification II

Question: How can we deal with data that is not linearly separable neither in the input nor in the feature space?

Soft Margin SVMs For Classification III

Answer: Modify the support vector machine so as to allow some of the training points to be misclassified.

Soft Margin SVMs For Classification IV

To do so we introduce the slack variables $\xi_n \geq 0$ to allow for misclassification errors to occur. That is, we allow non linearly separable dataset to be dealt with.
These are defined as

- $\xi_n = 0$, if $\boldsymbol{x}^{(n)}$ is on or inside the correct margin boundary
- $\xi_n = |y^{(n)} - h(\boldsymbol{x}^{(n)})|$ if $\boldsymbol{x}^{(n)}$ is *not* on or inside the correct margin boundary

Note: ξ_n can be greater than 0 even thought the n^{th} observation is on the correct side of the decision boundary if it is inside the margin.

Soft Margin SVMs For Classification V

The exact classification constraints are then replaced with

$$h(\boldsymbol{x}^{(n)})y^{(n)} \geq 1 - \xi_n, \quad n = 1, 2, \ldots, N.$$

in which the slack variables are constrained to satisfy $\xi_n \geq 0$.

Soft Margin SVMs For Classification VI

Our goal is now to maximize the margin while softly penalizing points that lie on the wrong side of the margin boundary. We therefore minimize

$$\arg\min_{\boldsymbol{w},b} ||\boldsymbol{w}||_2^2 + C\sum_{n=1}^{N}\xi_n$$
$$\text{s.t.}(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}^{(n)}) + b)y^{(n)} \geq 1 - \xi_n, \quad \xi_n \geq 0 \quad \forall\, n = 1, 2, \ldots, N.$$

where the parameter $C > 0$ controls the trade-off between the slack variable penalty and the margin.

Soft Margin SVMs For Classification VII

The corresponding Lagrangian is given by

$$\begin{aligned}\mathcal{L}(\boldsymbol{w}, \boldsymbol{\alpha}, b) =& ||\boldsymbol{w}||_2^2 \\ &+ C \sum_{n=1}^{N} \xi_n \\ &- \sum_{n=1}^{N} \alpha_n ((\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}^{(n)}) + b) y^{(n)} - 1 + \xi_n) \\ &- \sum_{n=1}^{N} \mu_n \xi_n,\end{aligned}$$

where $\{\mu_n \geq 0\}_{n=1}^{N}$ are the Lagrange multipliers associated with $\{\xi_n \geq 0\}_{n=1}^{N}$.

Soft Margin SVMs For Classification VIII

By setting to zero the partial derivatives w.r.t $\boldsymbol{w}$, $\{\xi_n\}_{n=1}^N$, and b we get the following three conditions:

$$\frac{\partial L(\boldsymbol{w}, \boldsymbol{\alpha}, b)}{\partial \boldsymbol{w}} = 0 \implies \boldsymbol{w} = \sum_{n=1}^{N} \alpha_n \boldsymbol{x}^{(n)} y^{(n)}$$

$$\frac{\partial L(\boldsymbol{w}, \boldsymbol{\alpha}, b)}{\partial b} = 0 \implies 0 = \sum_{n=1}^{N} \alpha_n y^{(n)}.$$

$$\frac{\partial L(\boldsymbol{w}, \boldsymbol{\alpha}, b)}{\partial \xi_n} = 0 \implies \alpha_n = C - \mu_n \quad \forall\, n = 1, 2, \ldots, N.$$

Soft Margin SVMs For Classification IX

Using these conditions to get rid of the dependencies on $\boldsymbol{w}$ and b from $\mathcal{L}(\boldsymbol{w}, \boldsymbol{\alpha}, b)$ leads to the *dual formulation* of the optimisation problem:

$$\begin{aligned}
&\arg\max_{\boldsymbol{\alpha}} \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \alpha_n \alpha_m \boldsymbol{\phi}(\boldsymbol{x}^{(n)})^T \boldsymbol{\phi}(\boldsymbol{x}^{(m)}) y^{(n)} y^{(m)} \\
&\quad \text{s.t. } 0 \leq \alpha_n \leq C, \\
&\qquad \sum_{n=1}^{N} \alpha_n y^{(n)} = 0 \\
&\qquad \forall\ n = 1, 2, \ldots, N.
\end{aligned}$$

Soft Margin SVMs For Classification X

In order to classify new data points using the trained model, we evaluate the sign of $h(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}) + b$ which can be expressed in terms of the kernel function and the Lagrange multipliers by substituting for

$$\boldsymbol{w} = \sum_{n=1}^{N} \alpha_n \boldsymbol{\phi}(\boldsymbol{x}^{(n)}) y^{(n)}$$

giving, as for the hard margin solution,

$$h(\boldsymbol{x}) = \sum_{n=1}^{N} \alpha_n k(\boldsymbol{x}, \boldsymbol{x}^{(n)}) y^{(n)} + b.$$

Soft Margin SVMs For Classification XI

Result (Soft Margin SVM)

Consider a Soft Margin SVM where the decision boundary is modelled as

$$h(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}) + b,$$

where $\boldsymbol{\phi} : \mathbb{R}^D \to \mathbb{R}^M$ *denotes a fixed feature-space transformation, and* $b \in \mathbb{R}$ *denotes the bias term. Let the training data be of the form*

$$\{(\boldsymbol{x}^{(n)}, y^{(n)}) \in \mathbb{R}^D \times \{-1, +1\} \mid n = 1, \ldots, N\},$$

Soft Margin SVMs For Classification XII

Result (Soft Margin SVM (Cont.))

Impose that

$$(\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}^{(n)}) + b) y^{(n)} \geq 1 - \xi_n, \quad \forall\ n = 1, 2, \ldots, N.$$

where $\xi_n \geq 0$.

Soft Margin SVMs For Classification XIII

Result (Soft Margin SVM (Cont.))

Then, the set of parameters $\boldsymbol{w}$ *and* b *can be found by solving the dual problem given by*

$$\begin{aligned} \arg\max_{\boldsymbol{\alpha}} & \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \alpha_n \alpha_m k(\boldsymbol{x}^{(n)}, \boldsymbol{x}^{(m)}) y^{(n)} y^{(m)} \\ & s.t.\ 0 \leq \alpha_n \leq C, \\ & \sum_{n=1}^{N} \alpha_n y^{(n)} = 0 \\ & \forall\ n = 1, 2, \ldots, N. \end{aligned} \tag{3.2}$$

where $k(\boldsymbol{x}^{(n)}, \boldsymbol{x}^{(m)})$ *is a valid kernel.*

Soft Margin SVMs For Classification XIV

Result (Soft Margin SVM (Cont.))

Finally, we can make predictions by computing

$$h(\boldsymbol{x}) = \sum_{n=1}^{N} \alpha_n k(\boldsymbol{x}, \boldsymbol{x}^{(n)}) y^{(n)} + b.$$

A Note on Computations I

We first note that the objective function given by 3.1 or 3.2 is quadratic and so any local optimum will also be a global optimum since the constraints define a convex region as a consequence of being liner.

A Note on Computations II

Direct solution of the quadratic programming problem using traditional techniques is often infeasible due to the demanding computation and memory requirements.
So, a wide range of more practical approaches have been proposed:

- Protected conjugate gradients (Burges, 1998).
- Decomposition methods (Osuna et al., 1996).
- Sequential minimal optimization (Platt, 1999).

Support Vector Machines: A Note on Probabilistic Output I

As we have seen, support vector machines do not provide probabilistic outputs but instead make *hard* classification decisions for new input vectors.
However, it is possible to construct conditional probabilities of the form

$$\mathbb{P}(y = 1|\boldsymbol{x}) = 1 - \mathbb{P}(y = -1|\boldsymbol{x})$$

by fitting a logistic sigmoid to the outputs of a previously trained SVM Platt (2000).

Support Vector Machines: A Note on Probabilistic Output II

Specifically, the conditional probability is assumed to be of the form

$$\mathbb{P}(y = 1|\boldsymbol{x}; \hat{\boldsymbol{w}}) = \sigma(\hat{w}_0 + \hat{w}_1 h(\boldsymbol{x}))$$

where the parameters $\hat{\boldsymbol{w}} = (\hat{w}_0, \hat{w}_1) \in \mathbb{R}^2$ are found by minimising the negative log-likelihood of the training data:

$$-\log \mathbb{P}(\boldsymbol{y}|\hat{\boldsymbol{w}}, \boldsymbol{X}) = -\sum_{n=1}^{N} \Big(y^{(n)} \log \mathbb{P}(y^{(n)} = 1|\hat{\boldsymbol{w}}, \boldsymbol{x}^{(n)}) + (1 - y^{(n)}) \log(1 - \mathbb{P}(y^{(n)} = 1|\hat{\boldsymbol{w}}, \boldsymbol{x}^{(n)})) \Big)$$

Support Vector Machines: A Note on Probabilistic Output III

Note: it is important to fit the logistic model using data that has not been used to train the underlying SVM in order to avoid severe overfitting.

Support Vector Machines: A Note on Multiclass Problems I

The support vector machine is fundamentally a two-class classifier. In practice, however, we often have to tackle problems involving $K > 2$ classes. Various methods have therefore been proposed for combining multiple two-class SVMs in order to build a multiclass classifier.

Support Vector Machines: A Note on Multiclass Problems II

- *one-versus-the-rest*: train K separate SVMs, in which the k^{th} model $h_k(\boldsymbol{x})$ is trained using the data from class $\mathcal{C}_k$ as the positive examples and the data from the remaining $K-1$ classes as the negative examples.
- *one-versus-one*: train $K(K-1)/2$ different 2-class SVMs on all possible pairs of classes, and then to classify test points according to which class has the highest number of votes.

Outline I

Outline II

Support Vector Machines For Regression: An Overview

We now extend support vector machines to regression problems while at the same time preserving the property of sparseness.
In regularised linear regression, the objective is to minimise the regularised error function given by

$$\frac{1}{2}\sum_{n=1}^{N}(h(\boldsymbol{x}^{(n)}) - y^{(n)})^2 + \frac{\lambda}{2}||\boldsymbol{w}||_2^2$$

where $\lambda \geq 0$ determines the regularisation strength.
Support vector regression replaces the sum-of-squared error function with an ϵ- *insensitive error function* in order to recover sparse solutions.

Derivation I

Let us take a special case of an ϵ- *insensitive error function* having a linear cost associated with errors outside the insensitive region

$$E_\epsilon(h(\boldsymbol{x};\boldsymbol{w}) - y) = \begin{cases} 0, & \text{if } |h(\boldsymbol{x};\boldsymbol{w}) - y| < \epsilon \\ |h(\boldsymbol{x};\boldsymbol{w}) - y| - \epsilon, & \text{otherwise} \end{cases}$$

This error function gives zero error if the absolute difference between the predicted value $h(\boldsymbol{x};\boldsymbol{w})$ and the target y is less than ϵ where $\epsilon > 0$.

Derivation II

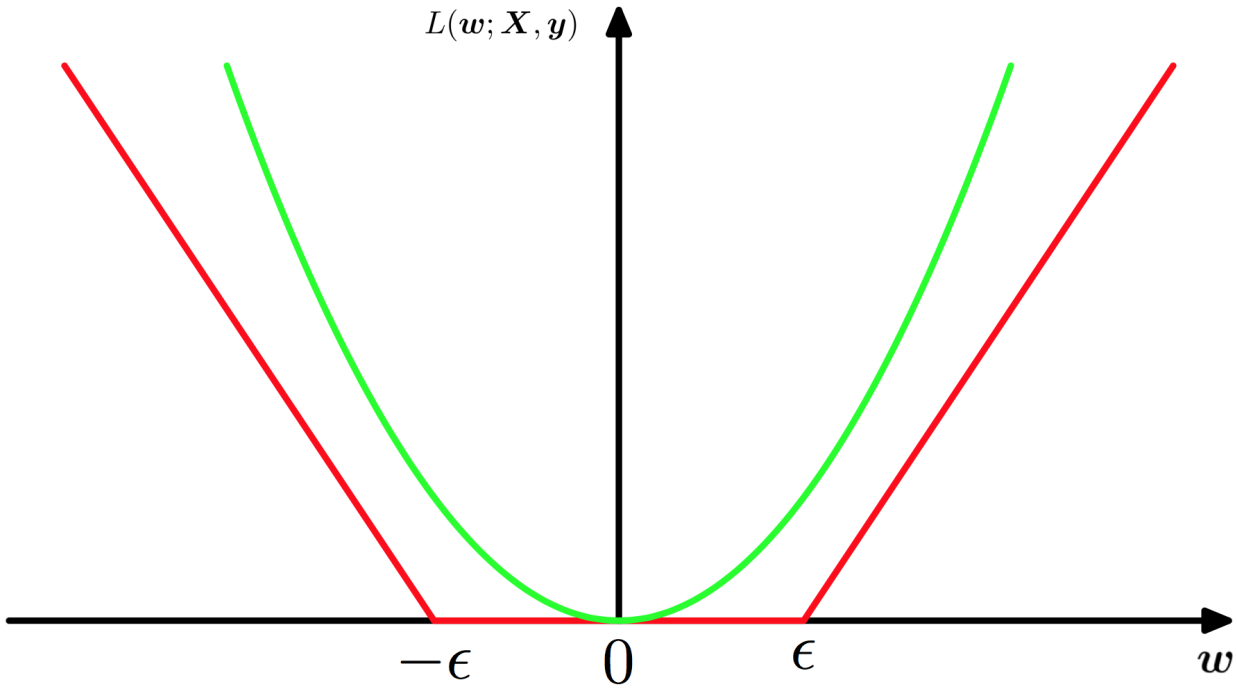

Figure: Plot of an ϵ- *insensitive error function* (red), and the quadratic error function (green) for comparison with $\boldsymbol{w} \in \mathbb{R}$.

Derivation III

Given the model family expressed as

$$h(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}) + b$$

where $\boldsymbol{\phi} : \mathbb{R}^D \rightarrow \mathbb{R}^M$ denotes a fixed feature-space transformation,
we therefore want to find the weights $\boldsymbol{w} \in \mathbb{R}^D$ and b which minimize the regularized error function given by

$$C \sum_{n=1}^{N} E_\epsilon(h(\boldsymbol{x}^{(n)}) - y^{(n)}) + \frac{1}{2}||\boldsymbol{w}||_2^2$$

Derivation IV

As before, we re-express the optimization problem by introducing slack variables. However, unlike before, we need to introduce two kind of slack variables.
For each observation $\boldsymbol{x}^{(n)}$ we need two slack variables $\xi_n \geq 0$ and $\hat{\xi}_n \geq 0$ where

- $\xi_n \geq 0$ corresponds to a point for which

$$y^{(n)} > h(\boldsymbol{x}^{(n)}) + \epsilon$$

- $\hat{\xi}_n \geq 0$ corresponds to a point for which

$$y^{(n)} < h(\boldsymbol{x}^{(n)}) - \epsilon$$

Derivation V

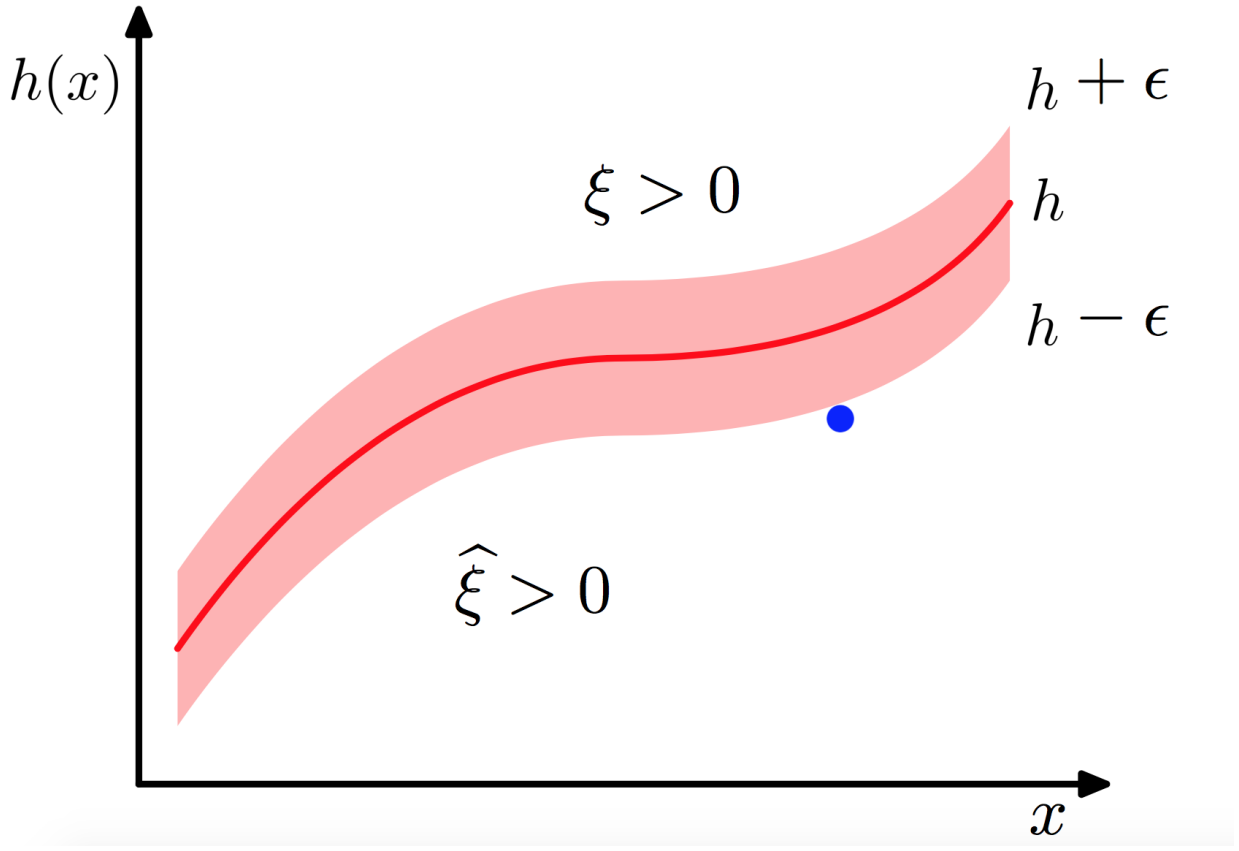

Figure: Illustration of SVM regression, showing the regression curve together with the ϵ-insensitive bound with examples of slack variables.

Derivation VI

The condition for a target point to lie inside the ϵ-tube is that

$$h(\boldsymbol{x}^{(n)}) - \epsilon \geq y^{(n)} \leq h(\boldsymbol{x}^{(n)}) + \epsilon.$$

Introducing the slack variables allows points to lie outside the tube provided the slack variables are non-zero, and the corresponding conditions are

$$\begin{aligned} y^{(n)} &\leq h(\boldsymbol{x}^{(n)}) + \epsilon + \xi_n, \\ y^{(n)} &\geq h(\boldsymbol{x}^{(n)}) - \epsilon - \hat{\xi}_n. \end{aligned}$$

Derivation VII

The error function for support vector regression can then be written as

$$C \sum_{n=1}^{N} (\xi_n + \hat{\xi}_n) - y^{(n)}) + \frac{1}{2} ||\boldsymbol{w}||_2^2.$$

Derivation VIII

From this, we can write the optimisation problem as

$$\begin{aligned}
\arg\max_{\boldsymbol{w}} \; & C\sum_{n=1}^{N}(\xi_n + \hat{\xi}_n) - y^{(n)}) + \frac{1}{2}||\boldsymbol{w}||_2^2 \\
\text{s.t.} \; & \xi_n \geq 0, \\
& \hat{\xi}_n \geq 0, \\
& y^{(n)} \leq h(\boldsymbol{x}^{(n)}) + \epsilon + \xi_n, \\
& y^{(n)} \geq h(\boldsymbol{x}^{(n)}) - \epsilon - \hat{\xi}_n, \\
& \forall \; n = 1, 2, \ldots, N.
\end{aligned}$$

Derivation IX

In order to solve this constrained optimization problem, we introduce Lagrange multipliers $\{\alpha_n \geq 0\}_{n=1}^N$, $\{\hat{\alpha}_n \geq 0\}_{n=1}^N$, $\{\mu_n \geq 0\}_{n=1}^N$, and $\{\hat{\mu}_n \geq 0\}_{n=1}^N$ giving the Lagrange function

Derivation X

$$\begin{aligned}\mathcal{L} =& C\sum_{n=1}^{N}(\xi_n + \hat{\xi}_n) + \frac{1}{2}||\boldsymbol{w}||_2^2 \\ & - \sum_{n=1}^{N}(\mu_n\xi_n + \hat{\mu}_n\hat{\xi}_n) \\ & - \sum_{n=1}^{N}\alpha_n(\epsilon + \xi_n + h(\boldsymbol{x}) - y^{(n)}) \\ & - \sum_{n=1}^{N}\hat{\alpha}_n(\epsilon + \hat{\xi}_n - h(\boldsymbol{x}) + y^{(n)}).\end{aligned}$$

Derivation XI

By setting to zero the partial derivatives w.r.t $\boldsymbol{w}$, $\{\xi_n\}_{n=1}^N$,$\{\hat{\xi}_n\}_{n=1}^N$, and b we get the following four conditions:

$$\frac{\partial L(\boldsymbol{w}, \boldsymbol{\alpha}, b)}{\partial \boldsymbol{w}} = 0 \implies \boldsymbol{w} = \sum_{n=1}^{N} (\alpha_n - \hat{\alpha}_n) \boldsymbol{\phi}(\boldsymbol{x}^{(n)})$$

$$\frac{\partial L(\boldsymbol{w}, \boldsymbol{\alpha}, b)}{\partial b} = 0 \implies 0 = \sum_{n=1}^{N} (\alpha_n - \hat{\alpha}_n)$$

$$\frac{\partial L(\boldsymbol{w}, \boldsymbol{\alpha}, b)}{\partial \xi_n} = 0 \implies \alpha_n = C - \mu_n \quad \forall\ n = 1, 2, \ldots, N$$

$$\frac{\partial L(\boldsymbol{w}, \boldsymbol{\alpha}, b)}{\partial \hat{\xi}_n} = 0 \implies \hat{\alpha}_n = C - \hat{\mu}_n \quad \forall\ n = 1, 2, \ldots, N.$$

Derivation XII

Using these conditions to get rid of the dependencies on $\boldsymbol{w}$ and b from $\mathcal{L}$ leads to the *dual formulation* of the optimisation problem:

$$\begin{aligned}
\underset{\boldsymbol{\alpha},\hat{\boldsymbol{\alpha}}}{\arg\min} \; & -\frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}(\alpha_n-\hat{\alpha}_n)(\alpha_m-\hat{\alpha}_m)k(\boldsymbol{x}_n,\boldsymbol{x}_m) \\
\text{s.t. } & 0 \leq \alpha_n \leq C, \\
& 0 \leq \hat{\alpha}_n \leq C, \\
& \sum_{n=1}^{N}(\alpha_n-\hat{\alpha}_n) = 0 \\
& \forall\; n = 1,2,\ldots,N.
\end{aligned}$$

where we have introduced the kernel $k(\boldsymbol{x}_n,\boldsymbol{x}_m) = \boldsymbol{\phi}(\boldsymbol{x}^{(n)})^T\boldsymbol{\phi}(\boldsymbol{x}^{(m)})$.

Derivation XIII

In order to classify new data points using the trained model, we evaluate the sign of $h(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}) + b$ which can be expressed in terms of the kernel function and the Lagrange multipliers by substituting for

$$\boldsymbol{w} = \sum_{n=1}^{N} (\alpha_n - \hat{\alpha}_n) \boldsymbol{\phi}(\boldsymbol{x}^{(n)})$$

giving

$$h(\boldsymbol{x}) = \sum_{n=1}^{N} (\alpha_n - \hat{\alpha}_n) k(\boldsymbol{x}, \boldsymbol{x}_n) + b.$$

Derivation XIV

Result (Support Vector Regression)

Consider a support vector regression model where specified as

$$h(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}) + b,$$

where $\boldsymbol{\phi} : \mathbb{R}^D \to \mathbb{R}^M$ *denotes a fixed feature-space transformation, and* $b \in \mathbb{R}$ *denotes the bias term. Let the training data be of the form*

$$\{(\boldsymbol{x}^{(n)}, y^{(n)}) \in \mathbb{R}^D \times \mathbb{R} \mid n = 1, \ldots, N\}.$$

Derivation XV

Result (Hard Margin SVM (Cont.))

Impose that

$$y^{(n)} \leq h(\boldsymbol{x}^{(n)}) + \epsilon + \xi_n,$$
$$y^{(n)} \geq h(\boldsymbol{x}^{(n)}) - \epsilon - \hat{\xi}_n.$$

where $\{\xi_n \geq 0\}_{n=1}^{N}$, $\{\hat{\xi}_n \geq 0\}_{n=1}^{N}$, *and* $\epsilon > 0$

Derivation XVI

Result (Hard Margin SVM (Cont.))

Then, the set of parameters $\boldsymbol{w}$ *and* b *can be found by solving the dual problem given by*

$$\begin{aligned} \underset{\boldsymbol{\alpha},\hat{\boldsymbol{\alpha}}}{\arg\min} \; & -\frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}(\alpha_n-\hat{\alpha}_n)(\alpha_m-\hat{\alpha}_m)k(\boldsymbol{x}_n,\boldsymbol{x}_m) \\ s.t. \; & 0 \leq \alpha_n \leq C, \\ & 0 \leq \hat{\alpha}_n \leq C, \\ & \sum_{n=1}^{N}(\alpha_n-\hat{\alpha}_n)=0 \\ & \forall \; n=1,2,\ldots,N. \end{aligned} \tag{4.1}$$

Derivation XVII

Result (Hard Margin SVM (Cont.))

where $k(\boldsymbol{x}^{(n)}, \boldsymbol{x}^{(m)})$ *is a valid kernel.*
Finally, we can make predictions by computing

$$h(\boldsymbol{x}) = \sum_{n=1}^{N} (\alpha_n - \hat{\alpha}_n) k(\boldsymbol{x}, \boldsymbol{x}_n) + b.$$

Some of the most common choices of kernels are

- Linear kernel: $k(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{x}^T \boldsymbol{x}'$
- Radial Basis Function (RBF) kernel: $k(\boldsymbol{x}, \boldsymbol{x}') = \exp(-\gamma ||\boldsymbol{x} - \boldsymbol{x}'||_2^2)$
- Polynomial kernel: $k(\boldsymbol{x}, \boldsymbol{x}') = (c + \boldsymbol{x}^T \boldsymbol{x}')^\beta$
- Sigmoid kernel: $k(\boldsymbol{x}, \boldsymbol{x}') = \sigma(c + \boldsymbol{x}^T \boldsymbol{x}')$

For Further Reading I


Shawe-Taylor, J. and N. Cristianini (2004)
Kernel Methods for Pattern Analysis.
Cambridge University Press.


Aizerman, M. A., E. M. Braverman, and L. I. Rozonoer (1964)
The probability problem of pattern recognition learning and the method of potential functions
Automation and Remote Control, 25:1175–1190.


Boser, B. E., I. M. Guyon, and V. N. Vapnik (1992)
A training algorithm for optimal margin classi- fiers. In D. Haussler (Ed.)
Proceedings Fifth Annual Workshop on Computational Learning Theory (COLT), 144–152.

For Further Reading II


Cortes, C. and V. N. Vapnik (1995)
Support vector networks
Machine Learning, 20: 273-297.