

# Support Vector Machine - Kernel Functions

Mai Hieu Hien

20521305@gm.uit.edu.vn

Pham Thien Bao

20521107@gm.uit.edu.vn

Dang Thi Tuong Vy

20522176@gm.uit.edu.vn

December 2021

## 1 Introduction

Support Vector Machine (SVM) was first heard in 1992, introduced by Boser, Guyon, and Vapnik in COLT-92. It is a set of related supervised learning methods used for classification and regression. It belongs to a family of generalized linear classifiers. In other term, SVM is a classification and regression prediction tool that uses machine learning theory to maximize predictive accuracy while automatically avoiding over-fit to the data. Support Vector machines can be defined as systems that use hypothesis space of a linear function in a high dimensional feature space, trained with a learning algorithm from optimization theory that implements a learning bias derived from statistical learning theory.

SVM may not be very fast compared to some other classification methods, but owing to its ability to model complex nonlinear boundaries, it has high accuracy. Compared to newer algorithms like neural networks, they have two main advantages: higher speed and better performance with a limited number of samples (in the thousands). This makes the algorithm very suitable for text classification problems, where it's common to have access to a dataset of at most a couple of thousands of tagged samples. In real life, SVMs have been successfully used in three main areas: text categorization, image recognition, and bioinformatics (Cristianini & Shawe-Taylor, 2000). Specific examples include classifying news stories, handwritten digit recognition, and cancer tissue samples.

## 2 Background

### 2.1 Hyperplane

In  $\mathcal{R}^n$ , the  $(n - 1)$ -dimensional affine subspaces are called hyperplanes, and the corresponding parametric equation is:  $y = x_0 + \sum_{i=1}^{n-1} \lambda_i b_i$ , where  $b_1, \dots, b_{n-1}$  form a basis of an  $(n - 1)$ -dimensional subspace  $U$  of  $\mathbb{R}^n$  and  $\lambda_1, \lambda_2, \dots, \lambda_n \in \mathbb{R}$ .

This means that a hyperplane is defined by a support point  $x_0$  and  $(n - 1)$  linearly independent vectors  $b_1, \dots, b_{n-1}$  that span the direction space.

### 2.2 Constrained Optimization and Lagrange Multipliers

#### 2.2.1 Optimization

We considered the problem of solving for the minimum of a function:

$\min_x f(x)$ , where  $f : \mathbb{R}^D \rightarrow \mathbb{R}$  as a problem of optimization

#### 2.2.2 Lagrange multiplier

In this section, we have additional constraints, a real-valued functions  $g_i : \mathbb{R}^D \rightarrow \mathbb{R}$  for  $i = 1; \dots; m$ , we consider the constrained optimization problem:

$$\min_x f(x), \text{ subject to } g_i(x) \leq 0 \text{ for all } i = 1; \dots; m$$

The problem can be solved by introducing the Lagrange multipliers  $\lambda_i \geq 0$  corresponding to each inequality constraint respectively so that:

$$L(x, \lambda) = f(x) + \sum_{i=1}^m \lambda_i g_i = f(x) + \lambda^T g(x)$$

We have concatenated all constraints  $g_i(x)$  into a vector  $g(x)$ , and all the Lagrange multipliers into a vector  $\lambda \in \mathbb{R}^m$ .

#### 2.2.3 Primal problem

$$\min_x f(x), \text{ subject to } g_i(x) \leq 0 \text{ for all } i = 1; \dots; m$$

is known as the *primal problem*, corresponding to the primal variables  $x$ .

### 2.2.4 Lagrangian dual problem

The associated *Lagrangian dual problem* is given by:  $\max_{\lambda \in \mathbb{R}^m} D(\lambda)$ , subject to  $\lambda \geq 0$ , where

$\lambda$  are the dual variables and  $D(\lambda) = \min_{x \in \mathbb{R}^d} L(x, \lambda)$

### 2.3 Convex set

A set  $C$  is convex if the line segment between any two points in  $C$  lies in  $C$ , i.e., if for any  $x_1, x_2 \in C$  and any  $\theta$  with  $0 \leq \theta \leq 1$ , we have  $\theta x_1 + (1 - \theta)x_2 \in C$

### 2.4 Strong duality và Slater's constraint qualification

Strong duality: The optimal solution of the dual problem is the same as the optimal solution of the primal problem when the primal problem satisfies Slater's condition:

Minimize:  $f_0(x)$

Subject to:  $f_i(x) \leq 0; i = 1, \dots, m$ ;  $f_0(x), f_i(x)$  are

$Ax = b$

convex.

There exists an  $x^* \in \text{relint}(D)$  (relative interior of

the convex set  $D := \bigcap_{i=1}^m \text{dom}(f_i)$ ) such that

$f_i(x^*) \leq 0; i = 1, \dots, m$  and  $Ax^* = b$ . Such a point is sometimes called strictly feasible, since the inequality constraints hold with strict inequalities.

If constraint functions are affine, there will exist an  $x^*$ .

### 2.5 Convex hull

The convex hull of a set  $C$ , denoted  $\text{conv } C$ , is the set of all convex combinations of points in  $C$ :

$\text{conv } C = \{\theta_1 x_1 + \dots + \theta_k x_k \mid x_i \in C, \theta_i \geq 0, i = 1, \dots, k, \theta_1 + \dots + \theta_k = 1\}$

### 2.6 Quadratic Programming

Consider the case of a convex quadratic objective function, where the constraints are affine

$$\min_{x \in \mathbb{R}^d} \left( \frac{1}{2} x^T Q x + c^T x \right), \text{ subject to } Ax \leq b$$

where  $A \in \mathbb{R}^{m \times d}, b \in \mathbb{R}^m, c \in \mathbb{R}^d$ .  $Q \in \mathbb{R}^{d \times d}$  is square symmetric matrix

## 3 Separating Hyperplanes

SVM finds a hyperplane of the form:

$$f(x) = \langle w, x \rangle + b$$

- $w$  is the weight vector;  $b$  is the real number (bias).
- $\langle w, x \rangle$  denote the inner product of two vectors.

We define the hyperplane that separates the two classes in our binary classification problem as

$$\{x \in \mathbb{R}^D : f(x) = 0\}$$

The main idea behind many classification algorithms is to represent data and then partition this space, ideally in a way that examples with the same label (and no other examples) are in the same partition. In the case of binary classification, the space would be divided into two parts corresponding to the positive and negative classes, respectively. We consider a particularly convenient partition, which is to (linearly) split the space into two halves using a hyperplane.

For binary classification, the set of possible values that the label/output can attain is binary, and we denote them by  $\{+1, -1\}$ . Suppose, we have the training data set has the form  $\{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$  with  $n$  instances.

- $x_n$  is a feature vector  $\in \mathbb{R}^D$ .
- $y_n$  is the class label in  $\{-1, 1\}$ , in which '1' is the positive class and '-1' is the negative class.

When training the classifier, we want to ensure that the examples with positive labels are on the positive side of the hyperplane

$$\langle w, x_n \rangle + b \geq 0 \text{ when } y_n = +1 \quad (3.1)$$

$$\langle w, x_n \rangle + b < 0 \text{ when } y_n = -1 \quad (3.2)$$

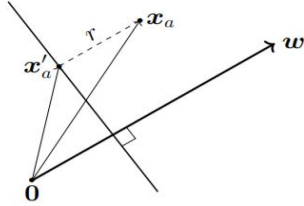
These two conditions (3.1) (3.2) are often presented in a single equation

$$y_n(\langle w, x_n \rangle + b) \geq 0$$

## 4 Primal Support Vector Machine

### 4.1 Concept of the Margin

Consider a hyperplane  $\langle w, x \rangle + b$ ,  $x_a$  to be on the positive side of the hyperplane, i.e.  $\langle w, x_a \rangle + b > 0$ . We would like to compute the distance  $r > 0$  of  $x_a$  from the hyperplane. We do so by considering the orthogonal projection of  $x_a$  onto the hyperplane, which we denote by  $x'_a$ . Since  $w$  is orthogonal to the hyperplane, we know that the distance  $r$  is just a scaling of this vector  $w$ .



If the length of  $w$  is known, then we can use this scaling factor  $r$  factor to work out the absolute distance between  $x_a$  and  $x'_a$ . For convenience, we choose to use a vector of unit length (its norm is 1) and obtain this by dividing  $w$  by its norm,  $\frac{w}{\|w\|}$ .

Using vector addition, we obtain

$$x_a = x'_a + r \frac{w}{\|w\|} \quad (4.1.1)$$

If we choose  $x_a$  to be the point closest to the hyperplane, this distance  $r$  is the margin.

Recall that we would like the positive examples to be further than  $r$  from the hyperplane, and the negative examples to be further than distance  $r$  (in the negative direction) from the hyperplane.

The objective:

$$\begin{aligned} & \max_{w, b, r} \quad r \\ & \text{subject to } \underbrace{y_n(\langle w, x_n \rangle + b)}_{\text{data fitting}} \geq r, \underbrace{\|w\|=1}_{\text{normalization}}, r > 0 \quad (4.1.3) \end{aligned}$$

which says that we want to maximize the margin  $r$  while ensuring that the data lies on the correct side of the hyperplane.

### 4.2 Traditional Derivation of the Margin

Instead of choosing that the parameter vector is normalized, we choose a scale for the data. We choose this scale such that the value of the predictor  $\langle w, x \rangle + b$  is 1 at the closest example.

The example  $x_a$  lies exactly on the margin, i.e.,  $\langle w, x_a \rangle + b = 1$ . Since  $x'_a$  is the orthogonal projection

of  $x_a$  onto the hyperplane, it must by definition lie on the hyperplane, i.e.,

$$\langle w, x'_a \rangle + b = 0 \quad (4.2.1)$$

By substituting (4.1.1) (4.1.3), we obtain

$$\langle w, x_a - r \frac{w}{\|w\|} \rangle + b = 0 \quad (4.2.2)$$

Exploiting the bilinearity of the inner product, we get

$$\langle w, x_a \rangle + b - r \frac{\langle w, w \rangle}{\|w\|} = 0 \quad (4.2.3)$$

Observe that the first term is 1 by our assumption of scale, i.e.,  $\langle w, x_a \rangle + b = 1$ . We know that  $\langle w, w \rangle = \|w\|^2$ . Hence, the second term reduces to  $r\|w\|$ . Using these simplifications, we obtain

$$r = \frac{1}{\|w\|} \quad (4.2.4)$$

Similar to the argument to obtain (4.1.2), we want the positive and negative examples to be at least 1 away from the hyperplane, which yields the condition

$$y_n(\langle w, x_n \rangle + b) \geq 1 \quad (4.2.5)$$

Combining the margin maximization with the fact that examples need to be on the correct side of the hyperplane (based on their labels) gives us

$$\max_{w, b} \frac{1}{\|w\|}$$

$$\text{subject to } y_n(\langle w, x_n \rangle + b) \geq 1 \text{ for all } n = 1, \dots, N \quad (4.2.6)$$

Or

$$\min_{w, b} \frac{1}{2} \|w\|^2$$

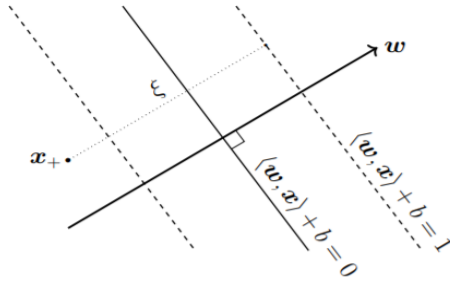
$$\text{subject to } y_n(\langle w, x_n \rangle + b) \geq 1 \text{ for all } n = 1, \dots, N \quad (4.2.7)$$

Equation is known as the **hard margin SVM**.

### 4.3 Soft Margin SVM: Geometric View

In the case where data is not linearly separable, we may wish to allow some examples to fall within the margin region, or even to be on the wrong side of the hyperplane. The model that allows for some classification errors is called the **soft margin SVM**.

The key geometric idea is to introduce a slack variable  $\xi_n$  corresponding to each example-label pair  $(x_n, y_n)$  that allows a particular example to be within the margin or even on the wrong side of the hyperplane.



We subtract the value of  $\xi_n$  from the margin, constraining  $\xi_n$  to be non-negative. To encourage correct classification of the samples, we add  $\xi_n$  to the objective:

$$\min_{w, b, \xi} \frac{1}{2} \|w\|^2 + C \sum_{n=1}^N \xi_n$$

$$\text{subject to } y_n(\langle w, x_n \rangle + b) \geq 1 - \xi_n, \xi_n \geq 0$$

The parameter  $C > 0$  trades off the size of the margin and the total amount of slack that we have. This parameter is called the regularization parameter, the margin term in the objective function is a regularization term. We have to choose the value of  $C$  such that a small deviation does not affect the classification too much to avoid overfitting.

#### 4.4 Soft Margin SVM: Loss Function View

Consider the error between the output of a predictor  $f(x_n)$  and the label  $y_n$ . The loss describes the error that is made on the training data. An equivalent way to derive is to use the hinge loss

$$l(t) = \max\{0, 1 - t\} \text{ where } t = y f(x) = y (\langle w, x \rangle + b) \quad (4.4.1)$$

If  $f(x)$  is on the correct side (based on the corresponding label  $y$ ) of the hyperplane, and further than distance 1, this means that  $t \geq 1$  and the hinge loss returns a value of zero. If  $f(x)$  is on the correct side but too close to the hyperplane ( $0 < t < 1$ ), the example  $x$  is within the margin, and the hinge loss returns a positive value. When the example is on the wrong side of the hyperplane ( $t < 0$ ), the hinge loss returns an even larger value, which increases linearly. In other words, we pay a penalty once we are closer than the margin to the hyperplane, even if the prediction is correct, and the penalty increases linearly. An alternative way to express the hinge loss is by considering it as two linear pieces.

$$l(t) = \begin{cases} 0 & \text{if } t \geq 1 \\ 1 - t & \text{if } t < 1 \end{cases} \quad (4.4.2)$$

The loss corresponding to the hard margin SVM (4.2.7) is defined as

$$l(t) = \begin{cases} 0 & \text{if } t \geq 1 \\ \infty & \text{if } t < 1 \end{cases} \quad (4.4.3)$$

For a given training set  $\{(x_1, y_1), \dots, (x_n, y_n)\}$ , we seek to minimize the total loss, while regularizing the objective with  $l_2$ -regularization. Using the hinge loss gives us the unconstrained optimization problem

$$\min_{w, b} \underbrace{\frac{1}{2} \|w\|^2}_{\text{regularizer}} + C \underbrace{\sum_{n=1}^N \max\{0, 1 - y_n(\langle w, x_n \rangle + b)\}}_{\text{error term}} \quad (4.4.4)$$

The first term in (4.4.4) is called the regularization term or the regularizer and the second term is called the loss term or the error term. Margin maximization can be interpreted as regularization regularization. By substituting this expression into (4.4.4) and rearranging one of the constraints, we obtain exactly the soft margin SVM.

## 5 Dual Support Vector Machine

### 5.1 Convex Duality via Lagrange Multipliers

Recall the primal soft margin SVM. We call the variables  $w, b, \xi$  corresponding to the primal SVM the primal variables. We use Lagrange multiplier  $\alpha_n \geq 0$  and  $\gamma_n \geq 0$  corresponding to the constraint that the examples are classified correctly and slack variables respectively. The Lagrangian is:

$$\mathcal{L}(w, b, \xi, \alpha, \gamma) = \frac{1}{2} \|w\|^2 + C \sum_{n=1}^N \xi_n - \sum_{n=1}^N \alpha_n (y_n (\langle w, x_n \rangle + b) - 1 + \xi_n) - \sum_{n=1}^N \gamma_n \xi_n$$

By differentiating the Lagrangian with respect to the three primal variables  $w$ ,  $b$ , and  $\xi$  respectively

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial w} &= w^T - \sum_{n=1}^N \alpha_n y_n x_n^T, (1) \quad w = \sum_{n=1}^N \alpha_n y_n x_n && \text{representer theorem} \\ \frac{\partial \mathcal{L}}{\partial b} &= -\sum_{n=1}^N \alpha_n y_n, (2) \quad \sum_{n=1}^N \alpha_n y_n = 0 && \text{that the optimal weight vector is an affine combination of} \\ \frac{\partial \mathcal{L}}{\partial \xi_n} &= C - \alpha_n - \gamma_n (3) \quad C - \alpha_n - \gamma_n = 0 \end{aligned}$$

$$\Rightarrow \mathcal{D}(\xi, \alpha, \gamma) = \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle - \sum_{i=1}^N \alpha_i y_i \left\langle \sum_{j=1}^N \alpha_j y_j x_j, x_i \right\rangle + C \sum_{i=1}^N \xi_i - b \sum_{i=1}^N \alpha_i y_i + \sum_{i=1}^N \alpha_i - \sum_{i=1}^N \alpha_i \xi_i - \sum_{i=1}^N \gamma_i \xi_i$$

$$\Leftrightarrow D(\xi, \alpha, \gamma) = \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle - \sum_{i=1}^N \alpha_i$$

We now obtain the dual optimization problem of the SVM. Recall from Lagrangian duality that we maximize the dual problem:

$$\begin{aligned} \max_{\alpha} \quad & \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle - \sum_{i=1}^N \alpha_i \\ \text{subject to:} \quad & \sum_{i=1}^N \alpha_i y_i = 0; \quad 0 \leq \alpha_i \leq C \quad \text{box constraint} \end{aligned}$$

Because  $C - \alpha_n - \gamma_n = 0 \Leftrightarrow C = \alpha_n + \gamma_n$  with  $\alpha_n \geq 0$  and  $\gamma_n \geq 0$   $C \geq \alpha_i$ . Without slack,  $\alpha_i \rightarrow \infty$  when constraints are violated (points misclassified).

## 5.2 Slater's condition and Karush Kuhn - Tucker conditions

We call  $p^*$  is the optimal solution of primal SVM and  $d^*$  is the optimal solution of dual SVM. Both of SVM' problem is similar when  $p^* = d^*$  (strong duality). However, we use Slater' conditions to prove it:

$$\min_{w, b, \xi} \frac{1}{2} \|w\|^2 + C \sum_{n=1}^N \xi_n$$

$$\text{subject to } y_n (w, x_n + b) \geq 1 - \xi_n; \quad \xi_n \geq 0$$

Since the inequality constraints of primal SVM are affine, and there always exists some solution that satisfy inequality constraints. So, the problem satisfies strong duality.

Karush–Kuhn–Tucker conditions:

$$1 - \xi_n - y_n (\langle w, x_n \rangle + b) \leq 0$$

$$+ \text{ Primal constraint: } -\xi_n \leq 0$$

$$\alpha_n \geq 0$$

$$+ \text{ Dual constraint: } \gamma_n \geq 0$$

+ Complementary slackness:

$$\alpha_n (y_n (\langle w, x_n \rangle + b) - 1 + \xi_n) = 0$$

$$\gamma_n \xi_n = 0$$

$$\text{Stationarity: } \exists w^*, b^*, \xi^* : \nabla L(w^*, b^*, \xi^*, \alpha, \gamma) = 0$$

If a convex optimization problem with differentiable objective and constraint functions satisfies Slater's condition, then the KKT

conditions provide necessary and sufficient conditions for optimality: Slater's condition implies that the optimal duality gap is zero and the

dual optimum is attained,  $w, b, \xi$  is optimal if and only if there are is optimal if and there are  $w, b, y_n, x_n, \xi, \alpha_n, \gamma_n$  satisfy the KKT conditions. The KKT conditions play an important role in optimization. In a few special cases it is possible to solve the KKT conditions (and therefore, the optimization problem) analytically.

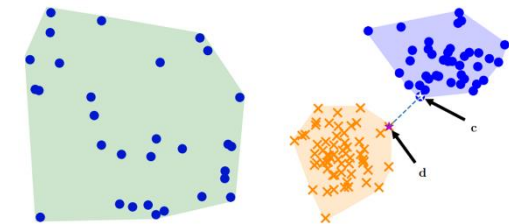
## 5.3 Explanation "Support Vector"

The examples  $x_n$  for which the corresponding parameters: if  $\alpha_n = 0$ , it does not contribute to

the solution  $w$  at all. if  $\alpha_n > 0$  It is called support vectors since they "support" the hyperplane.

## 5.4 Convex hull

Another approach to obtain the dual SVM is to consider an alternative geometric argument. We would like to build a convex set that contains all the examples such that it is the smallest possible set. This is called the convex hull. The convex hull of this area is the triangle formed by the edges corresponding to each pair of points. As we add more points, and the number of points becomes greater than the number of dimensions, some of the points will be inside the convex hull. the convex hull can be described as the set:



$$\text{conv}(X) = \left\{ \sum_{n=1}^N \alpha_n x_n \right\} \text{ with } \sum_{n=1}^N \alpha_n = 1 \text{ and } 0 \leq \alpha_n \leq 1$$

We pick a point  $c$ , which is in the convex hull of the set of positive examples and is closest to the negative class distribution. Similarly, we pick a point  $d$  in the convex hull of the set of negative examples and is closest to the positive class



distribution. We define a difference vector between  $\mathbf{d}$  and  $\mathbf{c}$ :  $\mathbf{w} := \mathbf{c} - \mathbf{d}$ .

The corresponding optimization problem:

$$\arg \min_w \|\mathbf{w}\| = \arg \min_w \frac{1}{2} \|\mathbf{w}\|^2$$

Since  $\mathbf{c}$  must be in the positive convex hull, it can be expressed as a convex combination of the positive examples:

$$\mathbf{c} = \sum_{n: y_n = +1} \alpha_n^+ \mathbf{x}_n; \alpha_n^+ \geq 0$$

We use the notation  $n: y_n = +1$  to indicate the set of indices  $n$  for which  $y_n = +1$ . Similarly, for the examples with negative labels, we obtain:

$$\mathbf{d} = \sum_{n: y_n = -1} \alpha_n^- \mathbf{x}_n; \alpha_n^- \geq 0$$

we obtain the objective:

$$\min_w \frac{1}{2} \|\mathbf{w}\|^2 = \frac{1}{2} \left\| \sum_{n: y_n = +1} \alpha_n^+ \mathbf{x}_n - \sum_{n: y_n = -1} \alpha_n^- \mathbf{x}_n \right\|^2$$

## Reduced convex hull

For inseparable problems, the convex hulls of the two sets will intersect. The difficult to classify points of one set will be in the convex hull of the other set. Therefore, we want the solution to be based on a lot of points, not just a few bad ones. Say we want the solution to depend on at least  $K$  points. This can be done by contracting or reducing the convex hull. The reduced convex hull is defined:

$$\text{conv}(X) = \left\{ \sum_{n=1}^N \alpha_n \mathbf{x}_n \right\} \text{ with } \sum_{n=1}^N \alpha_n = 1 \text{ and}$$

$$0 \leq \alpha_n \leq D; D < 1$$

A convex combination of the positive examples:

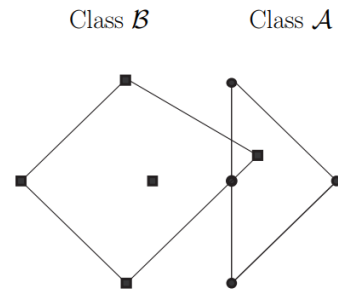
$$\mathbf{c}' = \left\{ \sum_{n=1}^N \alpha_n^+ \mathbf{x}_n \right\} \text{ with } \sum_{n=1}^N \alpha_n^+ = 1 \text{ and } 0 \leq \alpha_n^+ \leq D; D < 1$$

Similarly, we have a convex combination of the negative examples:

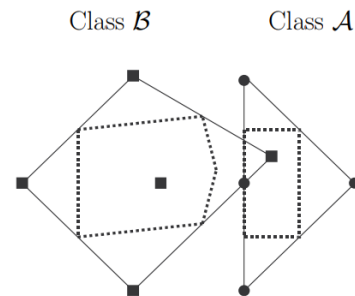
$$\mathbf{d}' = \left\{ \sum_{n=1}^N \alpha_n^- \mathbf{x}_n \right\} \text{ with } \sum_{n=1}^N \alpha_n^- = 1 \text{ and } 0 \leq \alpha_n^- \leq D; D < 1$$

We choose  $D = \frac{1}{K}$ ;  $K > 1$  reduced convex hull is

nonempty if  $K \leq m$  where  $m$  is the number of positive or negative examples. We reduce our feasible set away from the boundaries of the convex hulls so that no extreme point or noisy point can excessively influence the solution. we will need to choose  $K$  sufficiently large to ensure that the convex hulls do not intersect.



(The convex hulls of inseparable sets intersect)



(Convex hull and reduced convex hull with  $K = 2$ )

The problem of finding two closest points in the reduced convex hulls can be written as an optimization problem (RC-Hull):

$$\min_w \frac{1}{2} \|\mathbf{w}\|^2 = \frac{1}{2} \left\| \sum_{n: y_n = +1} \alpha_n^+ \mathbf{x}_n - \sum_{n: y_n = -1} \alpha_n^- \mathbf{x}_n \right\|^2$$

$$\text{subject to: } \sum_{n=1}^N \alpha_n^+ = \sum_{n=1}^N \alpha_n^- = 1 \text{ and } 0 \leq \alpha_i \leq D; D < 1$$

## 6. KERNELS

### 6.1 What is kernel?

Kernel is a function that returns the result of a dot product performed in another space.

Definition : Given a mapping function  $\phi : X \rightarrow V$ , we call the function  $K: X \rightarrow \mathbb{R}$  defined by

$K(x, x') = (\phi(x), \phi(x'))_{\mathcal{V}}$  where  $(\cdot)_{\mathcal{V}}$  denotes an inner product in  $V$ , a kernel function

### 6.2 The kernel trick

The kernel trick simply means replacing the dot product of two examples by a kernel function.

For example, we have a kernel  $K(x_i, x_j)$ , we can then rewrite the soft-margin dual problem:

$$\begin{aligned} & \text{maximize } \sum_{i=1}^m \alpha_i = \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j K(x_i, x_j) \\ & \text{subject to } 0 \leq \alpha_i \leq C \text{ for any } i=1, 2, \dots, m \\ & \sum_{i=1}^m \alpha_i y_i = 0 \end{aligned}$$

We have made a single change to the dual problem—we call it the kernel trick. Of course, we also need to change the hypothesis function to use the kernel function:

$$h(x_i) = \text{sign}(\sum_{j=1}^S \alpha_j y_j K(x_j, x_i) + b)$$

### 6.3 Properties of kernel functions

Symmetrical - because the dot product of two vectors is symmetric. In theory, the kernel function should satisfy the Mercer condition.

$$\sum_{n=1}^N \sum_{m=1}^N k(x_m, x_n) c_n c_m \geq 0,$$

$$\forall c_i \in \mathbb{R}, i = 1, 2, \dots, N$$

However, there are a few functions  $k()$  does not satisfy the Mercer condition but still gives acceptable results. These functions are still called kernels.

If a kernel function satisfies the condition Mercer, with  $c_n = y_n \lambda$

$$\begin{aligned} \lambda^T K \lambda &= \sum_{n=1}^N \sum_{m=1}^N k(x_m, x_n) y_n y_m \lambda_n \lambda_m \\ &\geq 0, \forall \lambda_n \end{aligned}$$

where  $K$  is a symmetric matrix whose element in the  $n$  row and  $m$  column is defined by

$$k_{nm} = k(x_m, x_n) y_n y_m$$

$\Rightarrow K$  is a positive semi-deterministic matrix.

### 6.4 Kernel types.

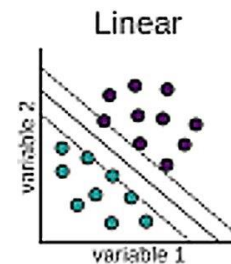
#### 6.4.1 Linear Kernel

This is the simplest kernel. In linear kernel, the kernel function takes the form of a linear function as follows :

$$K(x_1, x_2) = x_1^T \cdot x_2$$

Linear kernel is used when the data is linearly separable. It is mostly used when there are large number of features in a dataset. Linear kernel is often used for text classification purposes.

Training with a linear kernel is usually faster, because we only need to optimize the  $C$  regularization parameter. Linear kernel can be visualized with the following figure.

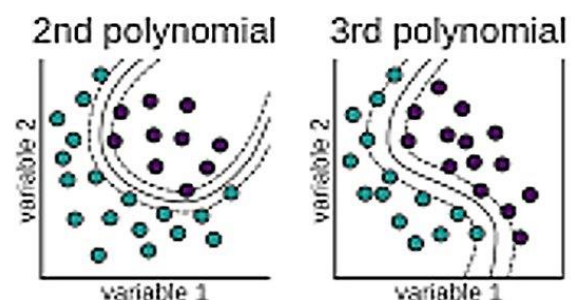


#### 6.4.2 Polynomial Kernel

We already saw the polynomial kernel earlier when we introduced kernels, but this time we will consider the more generic version of the kernel:

$$K(x_1, x_2) = (\gamma \cdot x_1^T \cdot x_2)^d, \gamma > 0$$

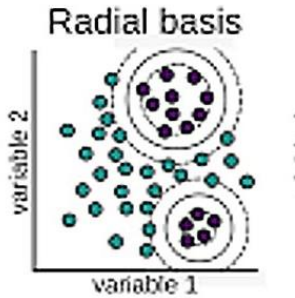
Polynomial kernel is very popular in Natural Language Processing



### 6.4.3 Radial Basis Function Kernel

Radial basis funkernel is a general purpose kernel. It is used when we have no prior knowledge ction about the data. The RBF kernel on two samples  $x$  and  $y$  is defined by the following equation.

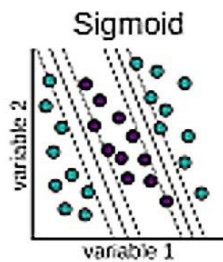
$$K(x_1, x_2) = \exp(\gamma \|x_1 - x_2\|_2^2), \gamma > 0$$



### 6.4.4 Sigmoid Kernel

Sigmoid kernel has its origin in neural networks. We can use it as the proxy for neural networks.

$$K(x, z) = \tanh(\gamma x^T z + r)$$



### 6.4.5 Which kernel should be used?

The recommended approach is to try a RBF kernel first, because it usually works well. However, it is good to try the other types of kernels if you have enough time to do so.

Building a custom kernel can also be a possibility, but it requires that you have a good mathematical understanding of the theory behind kernels. You can find more information on this subject in (Cristianini & Shawe-Taylor, 2000).

### 6.4.6 Summary

The kernel trick is one key component making Support Vector Machines powerful. Do not forget that there are many kernels, and try looking for kernels created to solve the kind of problems you are trying to solve. Using the

right kernel with the right dataset is one key element in your success or failure with SVMs.

## 7 Numerical Solution

Consider the loss function view of the SVM. This is a convex unconstrained optimization problem. Therefore, we apply a subgradient approach for solving it. However, the hinge loss is differentiable almost everywhere, except for one single point at the hinge  $t = 1$ . At this point, the gradient is a set of possible values that lie between 0 and  $-1$ . Therefore, the subgradient  $g$  of the hinge loss is given by

$$\begin{cases} -1 & t < 1 \\ [-1, 0] & t = 1 \\ 0 & t > 1 \end{cases}$$

Both the primal and the dual SVM result in a convex quadratic programming problem (constrained optimization). the primal SVM has optimization variables that have the size of the dimension  $D$  of the input examples. The dual SVM has optimization variables that have the size of the number  $N$  of example.

To express the primal SVM in the standard form for quadratic programming, let us assume that we use the dot product as the inner product. We rearrange the equation for the primal SVM, we use the phrase dot product to mean the inner product on Euclidean vector space. Such that the optimization variables are all on the right and the inequality of the constraint matches the standard form.

$$\begin{aligned} \min_{w, b, \xi} \quad & \frac{1}{2} \|w\|^2 + C \sum_{n=1}^N \xi_n \\ \text{subject to} \quad & -y_n x_n^T w - y_n b - \xi_n \leq 1 \\ & -\xi_n \leq 0 \end{aligned}$$

Then combine the variables  $w, b, x_n$  into a single vector, we get the following matrix form of soft margin:



$$\min_{w,b,\xi} \frac{1}{2} \begin{bmatrix} w \\ b \\ \xi \end{bmatrix}^T \begin{bmatrix} I_D & 0_{D,N+1} \\ 0_{N,D+1} & 0_{N+1,N+1} \end{bmatrix} \begin{bmatrix} w \\ b \\ \xi \end{bmatrix} + [0_{D,N+1} \quad C1_{N,1}]^T \begin{bmatrix} w \\ b \\ \xi \end{bmatrix}$$

$$\text{Subject to: } \begin{bmatrix} -YX & -y & -I_N \\ 0_{N,D+1} & -I_N \end{bmatrix} \begin{bmatrix} w \\ b \\ \xi \end{bmatrix} \leq \begin{bmatrix} -1_{N,1} \\ 0_{N,1} \end{bmatrix}$$

$I_m$  to represent the identity matrix of size  $m \times m$ ,  $0_{mn}$  to represent the matrix of zeros of size  $m \times n$ , and  $1_{mn}$  to represent the matrix of ones of size  $m \times n$ . In addition,  $y$  is the vector of labels  $[y_1, \dots, y_N]^T$ ,  $Y = \text{diag}(y)$  is an  $N$  by  $N$  matrix where the elements of the diagonal are from  $y$ , and  $X \in R^{N \times D}$  is the matrix obtained by concatenating all the examples.

To express the dual SVM in standard form, we first have to express the kernel matrix  $K$  such that each entry is  $K_{ij} = k(x_i, x_j)$ . For convenience of

notation we introduce a matrix with zeros everywhere except on the diagonal, where we store the labels, that is,  $Y = \text{diag}(y)$ . The dual SVM can be written as.

$$\min_{\alpha} \frac{1}{2} \alpha^T Y K Y \alpha - 1_{N,1}^T \alpha$$

$$\text{Subject to: } \begin{bmatrix} y^T \\ -y^T \\ -I_n \\ I_n \end{bmatrix} \alpha \leq \begin{bmatrix} 0_{N+2} \\ C1_{N,1} \end{bmatrix}$$

We introduced the standard forms of the constraints to be inequality constraints. We will express the dual SVM's equality constraint as two inequality constraints

$$Ax = b \text{ is replaced by } Ax \leq b \text{ and } Ax \geq b$$

Particular software implementations of convex optimization methods may provide the ability to express equality constraints.

## References:

- [1] Deisenroth, M., Faisal, A., & Ong, C. (2020). Mathematics for Machine Learning. Cambridge: Cambridge University Press. doi:10.1017/9781108679930
- [2] Boyd, S., & Vandenberghe, L. (2004). Convex Optimization. Cambridge: Cambridge University Press. doi:10.1017/CBO9780511804441
- [3] Tutorial on Support Vector Machine (SVM) - Vikramaditya Jakkula, School of EECS, Washington State University.
- [4] Support Vector Machines Succinctly by Alexandre Kowalczyk, 2017
- [5] Kaggle: SVM Classifier Tutorial by Prashant Banerjee
- [6] Kristin P. Bennett and Erin J. Brendensteiner. 2000. Duality and Geometry in SVM Classifiers. In Proceedings of the Seventeenth International Conference on Machine Learning (ICML '00). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 57–64.
- [7] Theodoridis, S. & Mavroforakis, Michael. (2007). Reduced Convex Hulls: A Geometric Approach to Support Vector Machines [Lecture Notes]. Signal Processing Magazine, IEEE. 24. 119 - 122. 10.1109/MSP.2007.361610.

