CS 559 Machine Learning

Lecture 5: Support Vector Machines

Ping Wang
Department of Computer Science
Stevens Institute of Technology

Today’s Lecture

- Vector Algebra, Formulation, Margin
- SVM for Linear Separable Case
- Non-separable Case, Penalties
- Non-linearity, Kernels

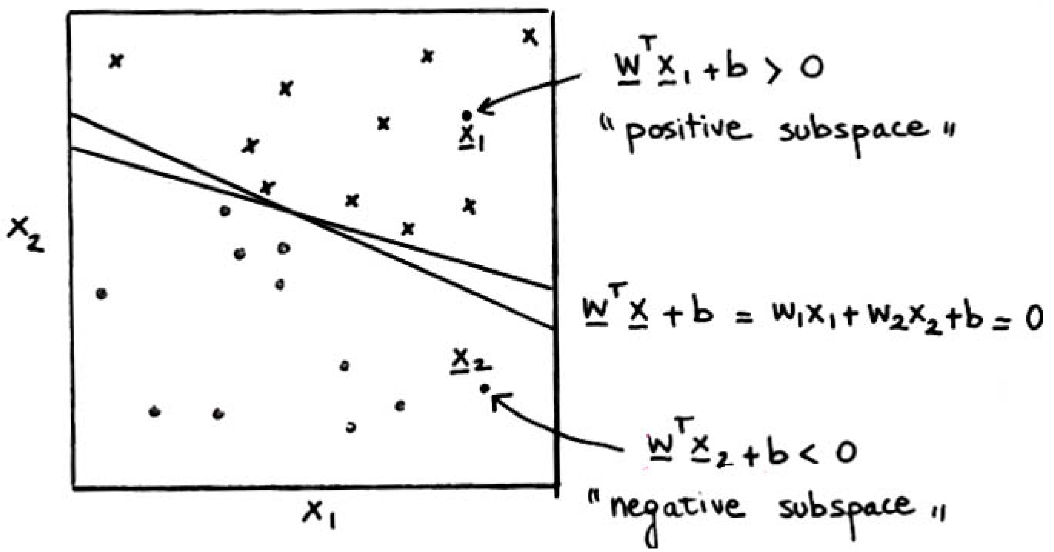
Linear Classifier

- Linear classifiers construct linear decision boundaries (hyperplanes) that try to separate the data into different classes as well as possible.
- Classification rule of the Perceptron algorithm:

$$Input: x \in \mathbb{R}^d$$

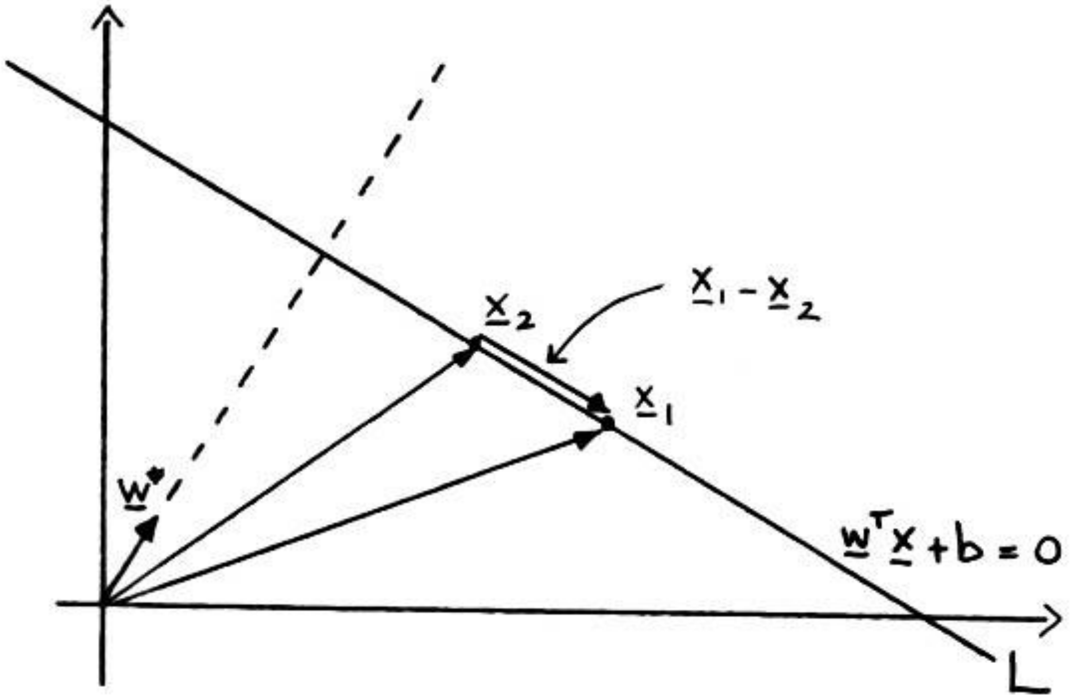
$$Output: sign(w^T x + b)$$

- The classifier computes a linear combination of the input features and return the sign.



Some Vector Algebra

- Hyperplane L: $f(x) = w^T x + b = 0$, when $x \in \mathbb{R}^d$, $f(x)$ is a linear boundary.



Some Vector Algebra: Property 1

- Consider any two points $x_1\ and\ x_2$, lying on hyperplane L:

$$\begin{matrix} w^T x_1 + b = 0 \\ w^T x_2 + b = 0 \end{matrix} \rightarrow w^T(x_1 - x_2) = 0$$

- Since $w^T(x_1 - x_2) = w \cdot (x_1 - x_2) = 0$, the two vectors w and $x_1 - x_2$ are orthogonal vectors.

$$w^* = \frac{w}{\|w\|}$$

 is the vector normal to the surface of L.

- Note 1: All vectors here are column vectors.
- Note 2: Dot product (inner product) of two vectors $a \cdot b = a^T b = \|a\| \times \|b\| \times \cos\alpha$, where α is the angle between a and b.

Some Vector Algebra: Property 2

- For any point x_0 on L:

$$w^T x_0 + b = 0$$

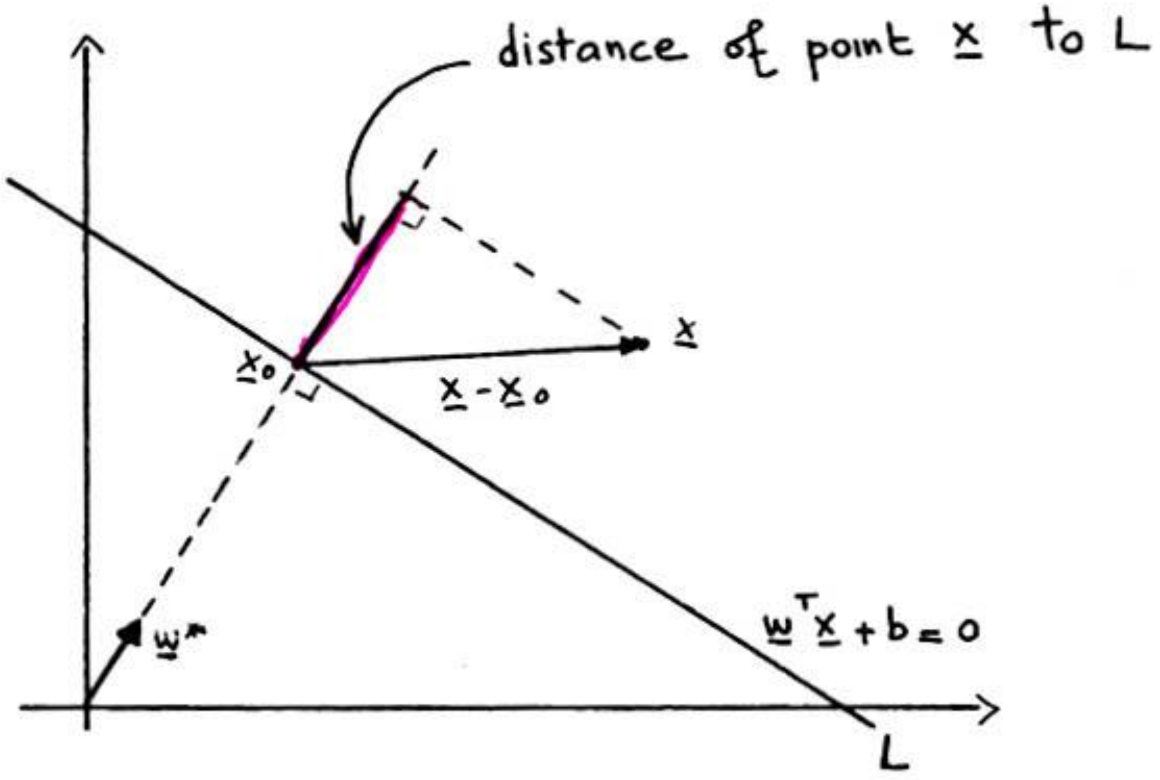
Thus:

$$w^T x_0 = -b$$

Some Vector Algebra: Property 3

- The signed distance of any point x to L is:

$$
\begin{aligned}
&(w^*)^T(x - x_0)\\
&= \frac{w^T}{\|w\|}(x - x_0)\\
&= \frac{1}{\|w\|}(w^T x - w^T x_0)\\
&= \frac{1}{\|w\|}(w^T x + b)\\
&= \frac{f(x)}{\|w\|}
\end{aligned}
$$



Perceptron Algorithm as Gradient Descent

Objective: Find a separating hyperplane that correctly classifies all input patterns.

- There are two types of error:

$$y_i = 1 \; and \; w^T x_i + b < 0$$
$$y_i = -1 \; and \; w^T x_i + b > 0$$

- Thus, for all misclassified points:

$$y_i(w^T x_i + b) < 0$$

- To reduce the number of misclassified points, the goal is to minimize:

$$D(w, b) = -\sum_{i \in M} y_i(w^T x_i + b)$$

where M indexes the set of misclassified points.

Perceptron Algorithm as Gradient Descent

Objective: Find a separating hyperplane that correctly classifies all input patterns.

- To minimize $D(w, b)$ we can perform gradient descent over the surface represented by $D(w, b)$ in parameter space. We iteratively move along the opposite direction of the gradient till a minimum is reached.
- The gradient is given by:

$$\frac{\partial D(w,b)}{\partial w} = -\sum_{i \in M} y_i x_i \, ; \quad \frac{\partial D(w,b)}{\partial b} = -\sum_{i \in M} y_i$$

- Update rule for parameters:

$$w' = w + \sum_{i \in M} y_i x_i \, ; \quad b' = b + \sum_{i \in M} y_i$$

Perceptron Algorithm as Gradient Descent

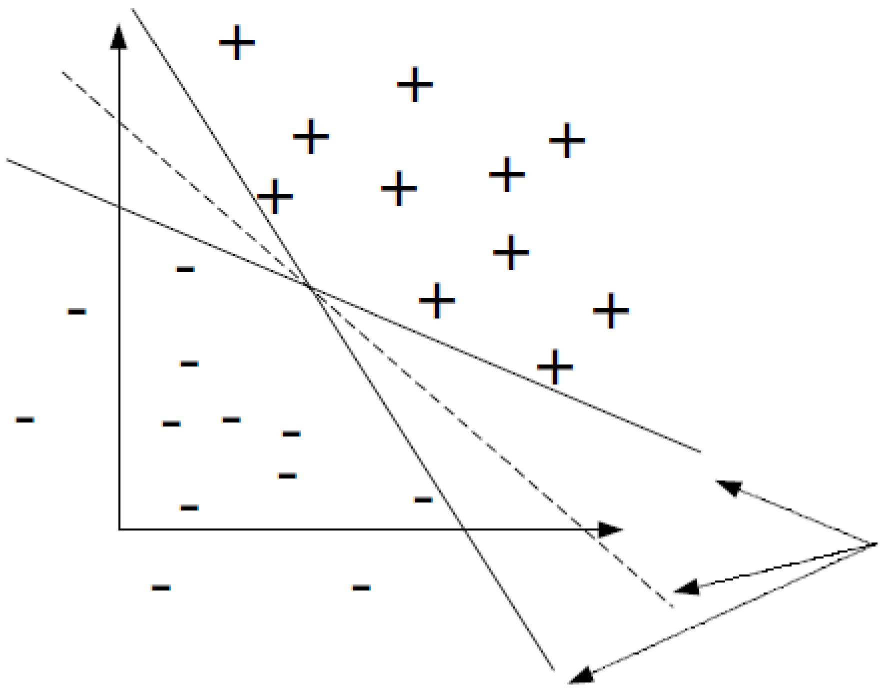
- In practice, Stochastic Gradient Descent (SGD) is used.
- Instead of computing the sum of the gradient contributions of each x_i, followed by a step in the negative gradient direction
- An iteration is taken after each observation is visited.
- The resulting update rule for parameters is:

$$w' = w + y_i x_i$$

$$b' = b + y_i$$

Limitations of Perceptron Algorithm

- When the data are linearly separable, there are many solutions, and which one is found depends on the starting values of the parameters.
- There can be an infinite number of hyperplanes that achieve 100% accuracy on training data.
- Which hyperplane is the optimal with respect to the accuracy on test data?

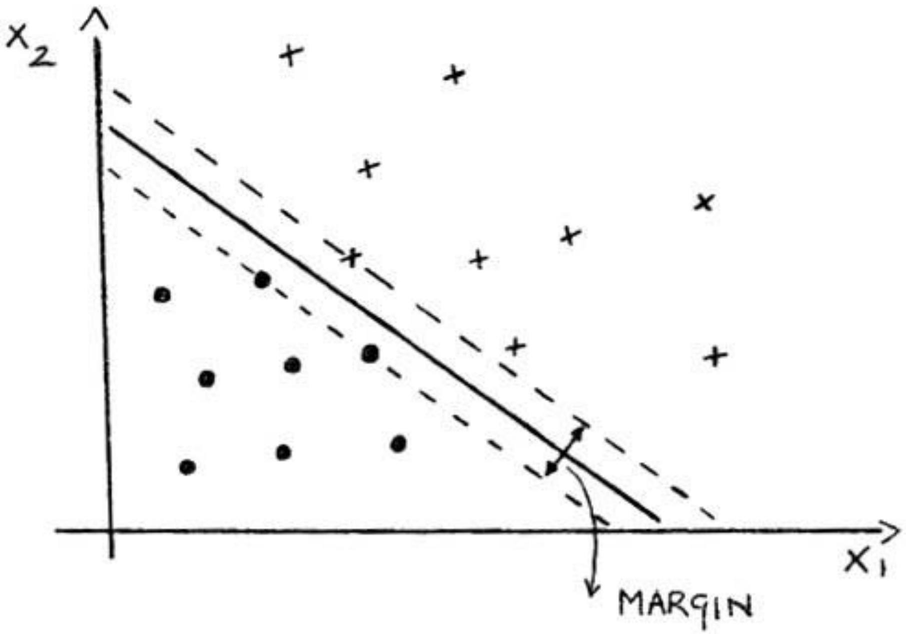


Three possible separations

Largest Margin Hyperplanes

Goal: Find the hyperplane that separates the two classes and maximizes the distance to the closest points from either class.

- Such distance is called margin.
- The added constraint:
  - Provide a unique solution to the separating hyperplane problem.
  - Maximizing the margin between the two classes on the training data gives better classification performance on test data.



Training Data

- For two classes:

$$(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)$$

$$x_i \in R^d$$

$$y_i \in \{-1, +1\}$$

- We need to formalize the largest margin criterion.

Formulation

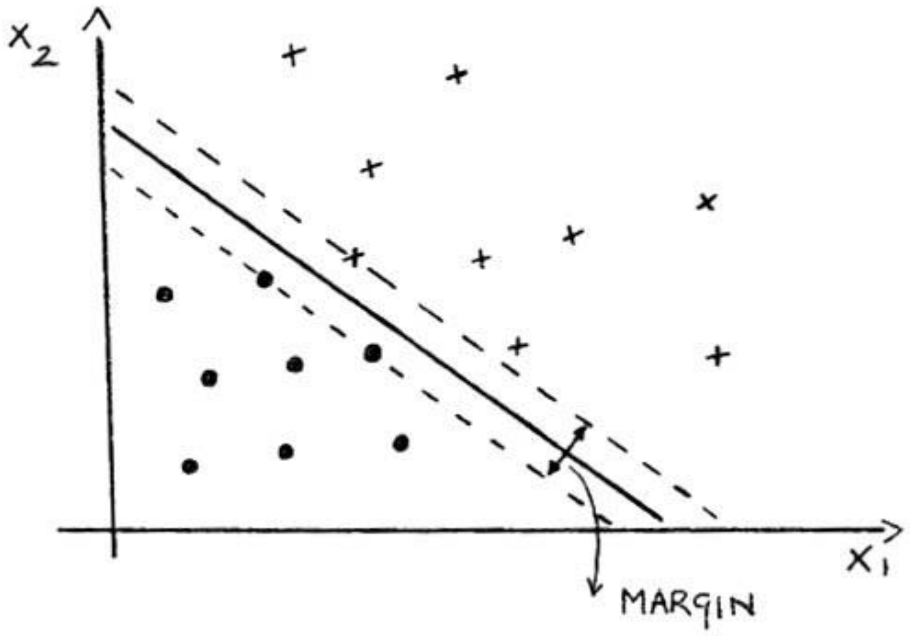
- Consider the following optimization problem:

$$\max_{w,b} 2C$$
$$subject\ to\quad \frac{1}{\|w\|} y_i(w^T x_i + b) \geq C, i = 1, \ldots, N$$

- Remember Property 3: The signed distance of any point x to L is:

$$\frac{1}{\|w\|}(w^T x_i + b)$$

- Thus, the set of conditions above ensure that all the training data are at least at distance C from the decision boundary.



Formulation

- The optimization problem:

$$\max_{w,b} 2C \quad subject\ to \ \ \frac{1}{\|w\|} y_i(w^T x_i + b) \geq C, i = 1, \dots, N$$

- We seek the largest C and associated parameters.
- We can rewrite the above conditions as:

$$y_i(w^T x_i + b) \geq C\|w\|$$

- Since $w^T x + b = 0$ and $c(w^T x + b) = 0$ define the same plane, we can arbitrarily normalize $\|w\| = \frac{1}{C}$.

Formulation

- The optimization problem:

$$\max_{w,b} 2C$$
$$subject\ to\ \ \frac{1}{\|w\|} y_i(w^T x_i + b) \geq C, i = 1, \ldots, N$$

- By normalizing $\|w\| = \frac{1}{C}$, the original maximization problem is equivalent to:

$$\min_{w,b} \frac{1}{2}\|w\|^2$$
$$subject\ to\ \ y_i(w^T x_i + b) \geq 1, i = 1, \ldots, N$$

- The constraints define an empty margin around the linear decision boundary of thickness $\frac{2}{\|w\|}$. We choose w, b to maximize its thickness.
- This is a convex quadratic optimization problem subject to linear constraints and there is a unique minimum.

Lagrange Multipliers

- We introduce the Lagrange multipliers $\alpha_i \geq 0, i = 1, \ldots, N$,
- One for each of the inequality constraints.
- Recall the rule:
  - For constraints of the form $C_i \geq 0$, the constraint equations are multiplied by Lagrange multipliers and subtracted from the objective function, to form the Lagrangian.

More details about Lagrange Multipliers: http://www.personal.psu.edu/sxj937/Notes/Lagrange_Multipliers.pdf

Lagrange Multipliers

- Lagrange multipliers allow us to take the constraints within the function to be minimized.
- Two reasons for doing this:
  - The constraints will be replaced by constraints on the Lagrange multipliers themselves, which are easier to handle.
  - In the new formulation of the problem, the training data will only appear in the form of dot products between vectors. This is a crucial property which will allow us to generalize the procedure to the nonlinear case.

Lagrange Multipliers: Primal Form

- We then obtain the Lagrangian: (also called **primal form**):

$$L_p = \frac{1}{2}||w||^2 - \sum_{i=1}^{N} \alpha_i [y_i(w^T x_i + b) - 1]$$

- We now minimize L_p with respect to w and b:

$$\min_{w,b} \max_{\alpha_i \geq 0} L_p$$

- This indicates that this is the **primal form** of the optimization problem.
- We will actually solve the primal optimization problem by solving the dual of the original problem, since they provide the same solution.

Optional reading for Lagrange duality: http://cs229.stanford.edu/summer2020/cs229-notes3.pdf. Section 6 in page 17.

Dual Form

- The solution to the dual form provides a lower bound to the solution of the primal form.
- What is the dual form?

$$\max_{\alpha_i \geq 0} \min_{w,b} L_p$$

- Setting the derivatives to zero gives:

$$\frac{\partial L_p}{\partial w} = w - \sum_{i=1}^{N} \alpha_i y_i x_i = 0 \qquad \Rightarrow w = \sum_{i=1}^{N} \alpha_i y_i x_i \qquad (1)$$

$$\frac{\partial L_p}{\partial b} = -\sum_{i=1}^{N} \alpha_i y_i = 0 \qquad \Rightarrow \sum_{i=1}^{N} \alpha_i y_i = 0 \qquad (2)$$

Dual Form

- Substituting Eq. (1) and (2) in L_p gives:

$$L_D = \frac{1}{2}\left(\sum_{i=1}^{N} \alpha_i y_i x_i\right)\left(\sum_{k=1}^{N} \alpha_k y_k x_k\right) - \sum_{i=1}^{N} \alpha_i \left[y_i \left(x_i^T \left(\sum\nolimits_{k=1}^{N} \alpha_k y_k x_k\right) + b\right) - 1\right]$$

$$= \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k x_i^T x_k - \sum_{i=1}^{N}\sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k x_i^T x_k - b\sum_{i=1}^{N} \alpha_i y_i + \sum_{i=1}^{N} \alpha_i$$

$$= \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k x_i^T x_k$$

Subject to $\alpha_i \geq 0$

The Lagrangian Dual Form

- The solution is obtained by maximizing L_D with respect to the α_i.
- The solution must satisfy the conditions:

$$w = \sum_{i=1}^{N} \alpha_i y_i x_i$$

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

$$\alpha_i \geq 0$$

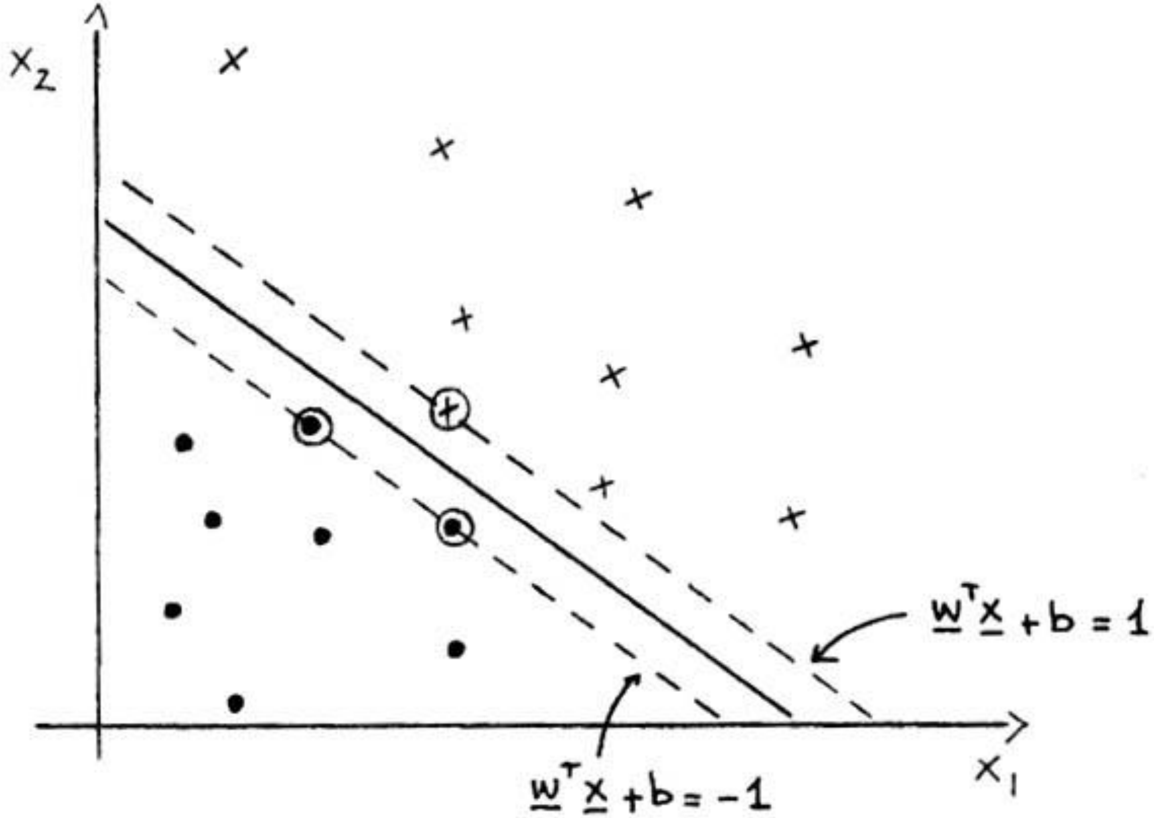
$$\alpha_i [y_i (w^T x_i + b) - 1] = 0 \quad \forall i = 1, \dots, N$$

Dual Form

$$\alpha_i[y_i(w^T x_i + b) - 1] = 0$$
$$\forall i = 1, \dots, N$$

- If $\alpha_i > 0$, then $y_i(w^T x_i + b) = 1$, that is x_i is on the boundary of the margin.

- If $y_i(w^T x_i + b) > 1$, x_i is not on the boundary of the margin, and $\alpha_i = 0$.



Dual Form

- The solution vector w is: $w = \sum_{i=1}^{N} \alpha_i y_i x_i$. Thus: The solution is defined as a linear combination of those x_i for which $\alpha_i > 0$.
- Such x_i are the points on the boundary of the margin. They are called SUPPORT VECTORS. We have three support vectors in the above example.
- To obtain the value of b: solve $\alpha_i[y_i(w^T x_i + b) - 1] = 0$ for any of the support vectors.
- The largest margin hyperplane gives a function: $f(x) = w^T x + b$ for classifying new observations $\hat{y} = sign(f(x))$.

Observations

- The support vectors are the critical elements of the training set. They lie closest to the decision boundary.
- Only the support vectors affect the solution - If all other training points were removed (or moved around, but so as not to cross the margin), and training was repeated, the same separating hyperplane would be found.
- However, the identification of the support vectors requires the use of all the training data.
- Although none of the training observations fall within the margin (by construction), this will not necessarily be the case of test data. (The intuition is that a large margin on the training data indicates a good separation of the two classes and therefore a good separation on the test data as well).

Summary so far

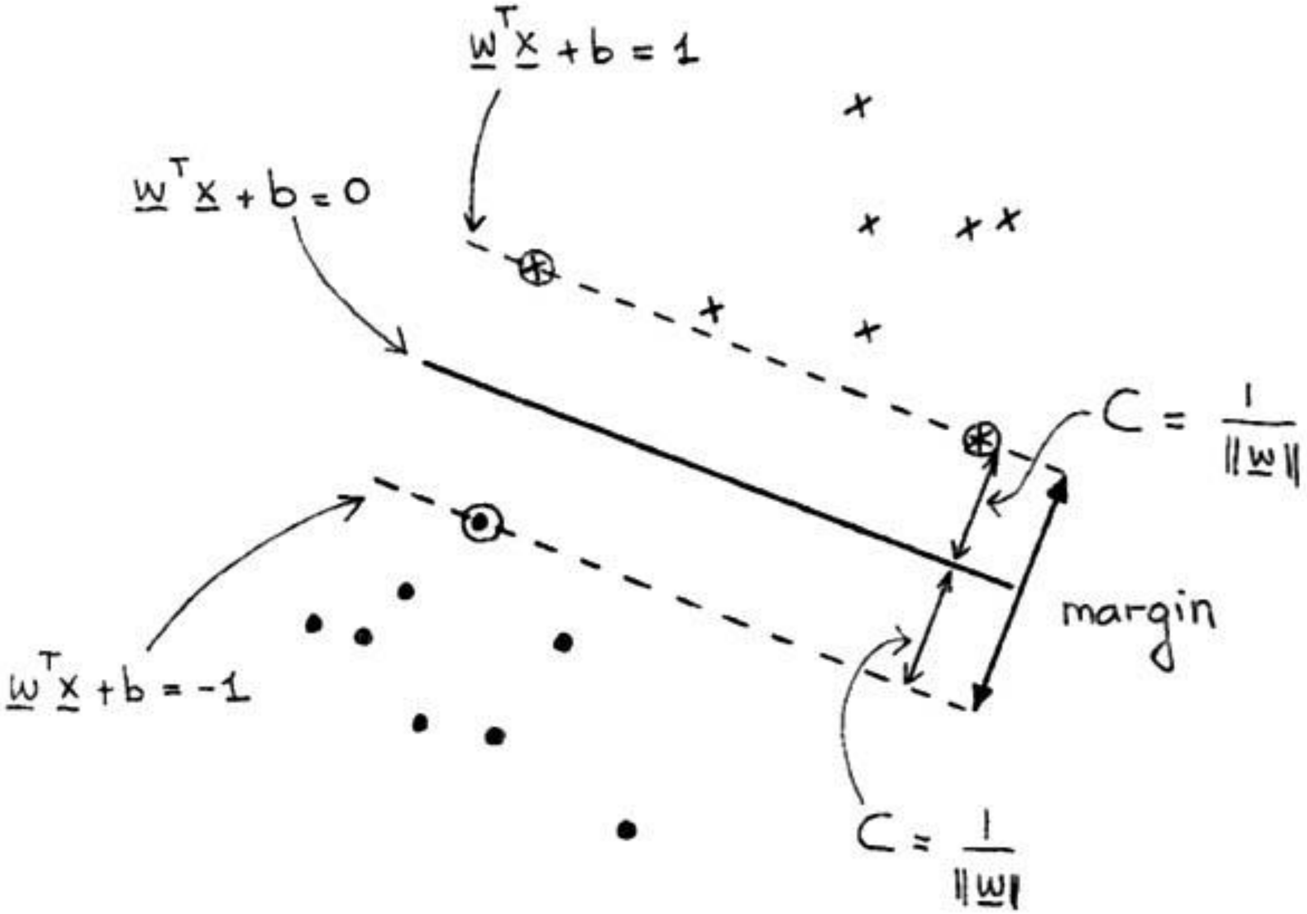
- Training data: $(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N), x_i \in R^d, y_i \in \{-1, +1\}$
- When the two classes are linearly separable, we can find a function $f(x) = w^T x + b$ with $y_i f(x_i) > 0 \ \forall i$.
- In particular, we can find the hyperplane that creates the largest margin between the training points of two classes.
- The optimization problem captures this concept

$$\max_{w,b} 2C \quad subject\ to \ \ \frac{1}{\|w\|} y_i(w^T x_i + b) \geq C, i = 1, \dots, N$$

- It can be more conveniently rewritten as below where $C = \frac{1}{\|w\|}$

$$\max_{w,b} \frac{1}{2} \|w\|^2 \ \ subject\ to \ \ y_i(w^T x_i + b) \geq 1, i = 1, \dots, N$$

Geometric Perspective



The Non-separable Case

The Non-separable Case

- Suppose now the classes overlap. We can still maximize C, but allow for some points to be on the wrong side of the margin.
- We need to modify the constraints we had for the separable case:
$\frac{1}{\|w\|} y_i(w^T x_i + b) \geq C, i = 1, \dots, N$
- To achieve this goal, we define N slack variables:

$$\xi_1, \xi_2, \dots, \xi_N$$

- Then a natural way to modify the constraints above is:

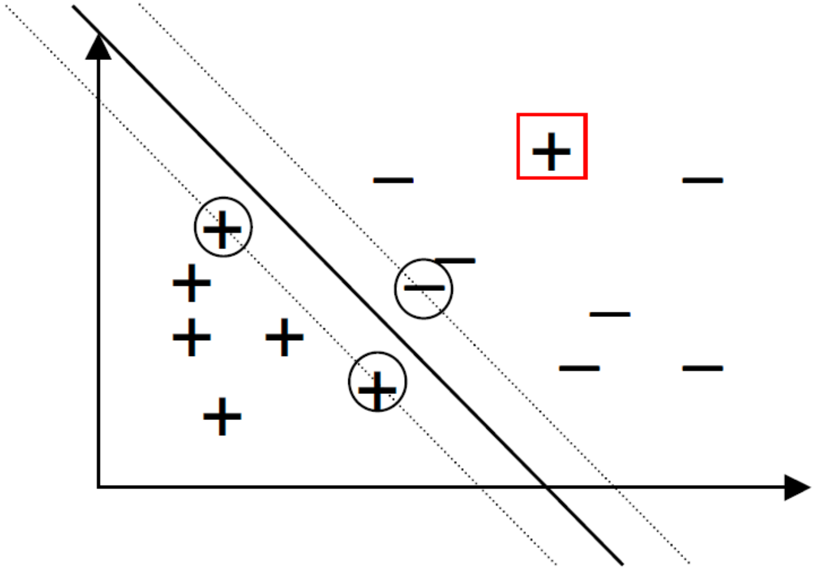
$$\frac{1}{\|w\|} y_i(w^T x_i + b) \geq C(1 - \xi_i), i = 1, \dots, N$$

with $\xi_i \geq 0, \forall i, \ \sum_{i=1}^{N} \xi_i \leq Constant$

The Non-separable Case

$$\frac{1}{\|w\|} y_i(w^T x_i + b) \geq C(1 - \xi_i), i = 1, \ldots, N$$

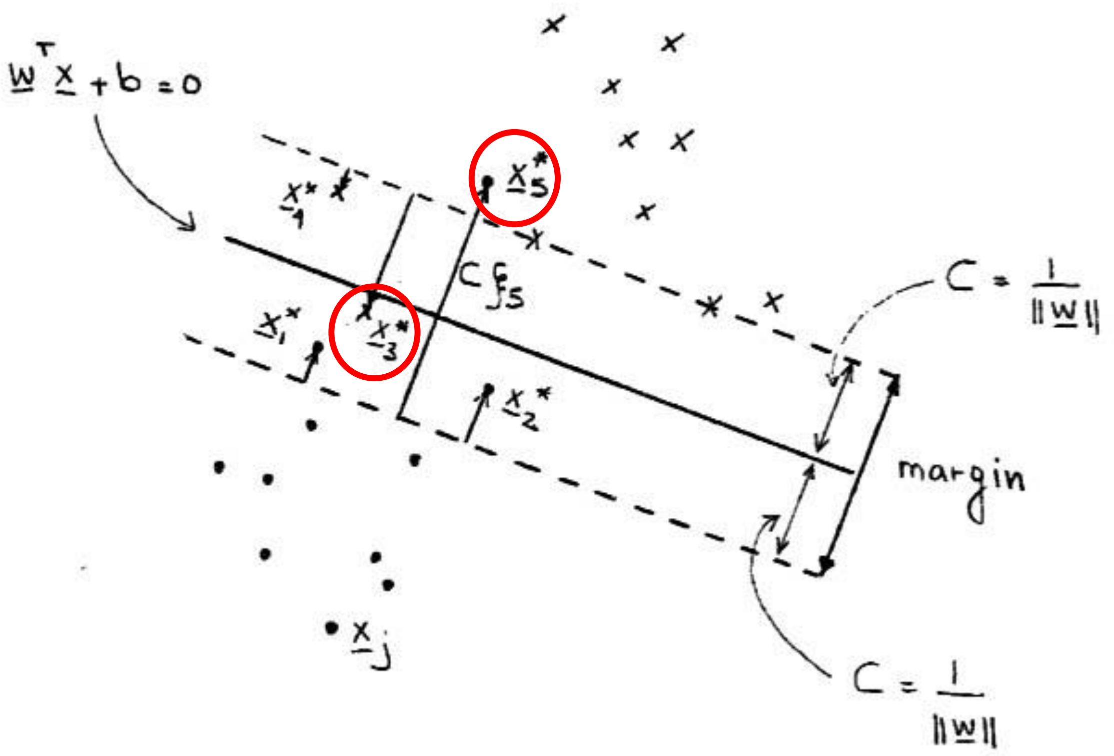
with $\xi_i \geq 0, \ \forall i, \ \sum_{i=1}^{N} \xi_i \leq Constant$



- Idea of the formulation: ξ_i is the proportional amount by which the prediction $f(x_i)$ is on the wrong side of the margin.
- Note: $C(1 - \xi_i) = C - C\xi_i$

Slack Variables

- The points $(x_1^*, x_2^*, x_3^*, x_4^*, x_5^*)$ are on the wrong side of their margin.
- Point x_i^* is on the wrong side of its margin by an amount $C\xi_i$
- Point x_j^* on the correct side have $\xi_j = 0$
- Misclassification occurs when $\xi_i > 1 \Rightarrow C(1 - \xi_i) < 0$, e.g., points x_3^* and x_5^* are misclassified by the given boundary.



A geometric perspective

Slack Variables

- The condition $\sum_{i=1}^{N} \xi_i \leq Constant$ bounds the sum $\sum_{i=1}^{N} \xi_i$.
- Thus, it bounds the total proportional amount by which predictions fall on the wrong side of their margin.
- Since misclassification occur when $\xi_i > 1$ (in this case $y_i f(x_i) < 0$, bounding $\sum_{i=1}^{N} \xi_i < k$, bounds the total number of training misclassifications at k.
- So, for the non-separable case, we have the optimization problem:

$$\max_{w,b} 2C \quad subject\ to \ \frac{1}{\|w\|} y_i(w^T x_i + b) \geq C(1 - \xi_i), i = 1, \dots, N$$

with $\xi_i \geq 0,\ \forall i,\ \sum_{i=1}^{N} \xi_i \leq Constant$

Slack Variables

- Similar to the separable case, we define $C = \frac{1}{\|w\|}$ and rewrite the above maximization problem in the equivalent form:

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

$$subject\ to\ \ y_i(w^T x_i + b) \geq 1 - \xi_i, i = 1, \dots, N$$

with $\xi_i \geq 0,\ \forall i,\ \sum_{i=1}^{N} \xi_i \leq Constant$

- We have obtained a quadratic optimization problem with linear constraints. We will solve it using Lagrange multipliers.

Lagrange Multipliers for Slack Variables

- First, one more step: we have seen that the condition $\sum_{i=1}^{N} \xi_i \leq Constant$, bounds the number of training misclassifications.
- We can incorporate this condition into the objective function by adding an extra cost for errors:

$$\min_{w,b} \frac{\|w\|^2}{2} + \gamma \sum_{i=1}^{N} \xi_i$$

$$subject\ to\ \ y_i(w^T x_i + b) \geq 1 - \xi_i, i = 1, \dots, N$$

with $\xi_i \geq 0, \forall i$

here, γ is a parameter to be chosen by the user. A larger γ corresponds to assigning a higher penalty to errors.

Lagrange Multipliers for Slack Variables

$$\min_{w,b} \frac{\|w\|^2}{2} + \gamma \sum_{i=1}^{N} \xi_i$$

$$subject\ to\ \ y_i(w^T x_i + b) \geq 1 - \xi_i, i = 1, \dots, N$$

with $\xi_i \geq 0, \forall i$

- Introducing the Lagrange multipliers α_i and μ_i (one for each constraint), gives the following Lagrange (primal) function:

$$L_p = \frac{1}{2}\|w\|^2 + \gamma \sum_{i=1}^{N} \xi_i - \sum_{i=1}^{N} \alpha_i [y_i(w^T x_i + b) - (1 - \xi_i)] - \sum_{i=1}^{N} \mu_i \xi_i$$

- Our objective is:

$$\min_{w,b,\xi_i} L_p$$

Lagrange Multipliers for Slack Variables

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

$$subject\ to\ \ y_i(w^T x_i + b) \geq 1 - \xi_i, i = 1, \ldots, N$$

$$\xi_i \geq 0,\ \forall i$$

$$\sum_{i=1}^{N} \xi_i \leq Constant$$

$$\Downarrow$$

$$L_p = \frac{1}{2}\|w\|^2 + \gamma \sum_{i=1}^{N} \xi_i - \sum_{i=1}^{N} \alpha_i [y_i(w^T x_i + b) - (1 - \xi_i)] - \sum_{i=1}^{N} \mu_i \xi_i$$

Lagrange Multipliers Solution

- Setting the respective derivatives to zero gives:

$$\frac{\partial L_p}{\partial w} = w - \sum_{i=1}^{N} \alpha_i y_i x_i = 0 \qquad \Rightarrow \mathrm{w} = \sum_{i=1}^{N} \alpha_i y_i x_i \qquad (3)$$

$$\frac{\partial L_p}{\partial b} = -\sum_{i=1}^{N} \alpha_i y_i = 0 \qquad \Rightarrow \sum_{i=1}^{N} \alpha_i y_i = 0 \qquad (4)$$

$$\frac{\partial L_p}{\partial \xi_i} = \gamma - \alpha_i - \mu_i, \forall i \qquad \Rightarrow \alpha_i = \gamma - \mu_i, \forall i \qquad (5)$$

along with the positivity constraints $\alpha_i, \mu_i, \xi_i \geq 0, \forall i$

- Substituting Eq. (3), (4), (5) in L_p, we obtain the so called dual objective function:

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j x_i^T x_j$$

where L_D gives a lower bound on the objective function $\frac{1}{2}\|w\|^2 + \gamma \sum_{i=1}^{N} \xi_i$

Deriving the Dual Form

$$\frac{1}{2}\Big(\sum_{i=1}^{N}\alpha_i y_i \mathrm{x}_i\Big)^\top\Big(\sum_{j=1}^{N}\alpha_j y_j \mathrm{x}_j\Big) + \gamma\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i y_i \mathrm{x}_i^\top \mathrm{w}^\top - \sum_{i=1}^{N}\alpha_i y_i b + \sum_{i}\alpha_i(1-\xi_i) - \sum_{i=1}^{N}\mu_i\xi_i$$

$$= \frac{1}{2}\sum_i\sum_j \alpha_i\alpha_j y_i y_j \mathrm{x}_i \mathrm{x}_j - \sum_i\sum_j \alpha_i\alpha_j y_i y_j \mathrm{x}_i \mathrm{x}_j + \gamma\sum_{i=1}^{N}\xi_i - b\underbrace{\sum_{i=1}^{N}\alpha_i y_i}_{=0} + \sum_i \alpha_i(1-\xi_i) - \sum_{i=1}^{N}\mu_i\xi_i$$

$$= -\frac{1}{2}\sum_i\sum_j \alpha_i\alpha_j y_i y_j \mathrm{x}_i \mathrm{x}_j + \gamma\sum_{i=1}^{N}\xi_i + \sum_{i=1}^{N}\alpha_i - \sum_i^N \alpha_i\xi_i - \sum_i^N \mu_i\xi_i$$

$$= \sum_i^N \alpha_i - \frac{1}{2}\sum_i\sum_j \alpha_i\alpha_j y_i y_j \mathrm{x}_i \mathrm{x}_j + \underbrace{\sum_i \overbrace{(\gamma-\mu_i)}^{=\alpha_i}\xi_i - \sum_i \alpha_i\xi_i}_{=0}$$

$$= \sum_i \alpha_i - \frac{1}{2}\sum_i\sum_j \alpha_i\alpha_j y_i y_j \mathrm{x}_i \mathrm{x}_j$$

Lagrange Multipliers Solution

- Thus: the solution is obtained by maximizing L_D w.r.t the α_i, subject to:

$$\sum_{i=1}^{N} \alpha_i y_i = 0, \qquad 0 \leq \alpha_i \leq \gamma$$

- The solution must satisfy the conditions:
  - $w = \sum_{i=1}^{N} \alpha_i y_i x_i$ (6)
  - $\sum_{i=1}^{N} \alpha_i y_i = 0$ (7)
  - $\alpha_i = \gamma - \mu_i, \forall i$ (8)
  - $\alpha_i [y_i (w^T x_i + b) - (1 - \xi_i)] = 0, \forall i$ (9)
  - $\mu_i \xi_i = 0, \forall i$ (10)
  - $y_i (w^T x_i + b) - (1 - \xi_i) \geq 0, \forall i$ (11)

Lagrange Multipliers Solution

- From (6), the solution is $w = \sum_{i=1}^{N} \alpha_i y_i x_i$.
- From (9), $\alpha_i > 0$ when constraint (11) is exactly met.
- The points (x_i) with $\alpha_i > 0$ are the SUPPORT VECTORS. Two types:
  - Those for which $\xi_i = 0$: they lie on the edge of the margin. From (8) and (10): $0 < \alpha_i < \gamma$
  - Those for which $\xi_i > 0$: they have $\alpha_i = \gamma$ and they lie on the wrong side of their margin.
- To estimate b, we can use (9) with any of the support vectors with $\xi_i = 0$.

- $w = \sum_{i=1}^{N} \alpha_i y_i x_i$ (6)
- $\sum_{i=1}^{N} \alpha_i y_i = 0$ (7)
- $\alpha_i = \gamma - \mu_i, \forall i$ (8)
- $\alpha_i[y_i(w^T x_i + b) - (1 - \xi_i)] = 0, \forall i$ (9)
- $\mu_i \xi_i = 0, \forall i$ (10)
- $y_i(w^T x_i + b) - (1 - \xi_i) \geq 0, \forall i$ (11)

Lagrange Multipliers Solution

- Once we have w and b, the decision function can be written as:
$$\hat{y} = sign\big(f(x)\big) = sign(w^T x + b)$$
- The tuning parameter of this procedure is γ. Its optimal value can be estimated via cross validation.

Recast as Unconstrained Optimization Problem

- A constrained optimization problem over w and ξ

$$\min_{w,b} \frac{\|w\|^2}{2} + \gamma \sum_{i=1}^{N} \xi_i$$

Subject to: $y_i(w^T x_i + b) \geq 1 - \xi_i, i = 1, \dots, N$

- The constraint can be written more concisely as: $y_i f(x_i) \geq 1 - \xi_i$ together with $\xi_i \geq 0$ is equivalent to

$$\xi_i = \max(0,1 - y_i f(x_i))$$

- Hence the learning problem is equivalent to the unconstrained optimization problem over w:

$$\min_{w,b} \frac{\|w\|^2}{2} + \gamma \sum_{i=1}^{N} \max(0,1 - y_i f(x_i))$$

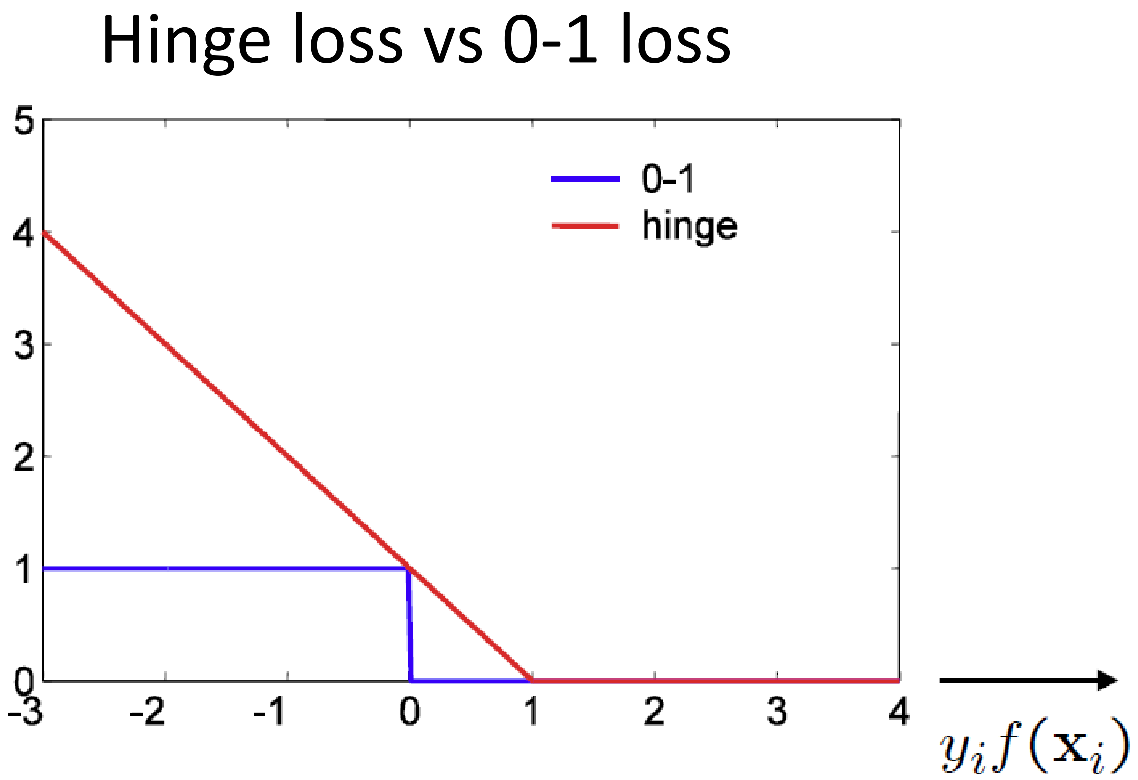
Loss Function

We can consider it as the minimization of a regularized error function.

$$\min_{w,b} \frac{\|w\|^2}{2} + \gamma \sum_{i=1}^{N} \underbrace{\max(0, 1 - y_i f(x_i))}_{\text{Hinge Loss}}$$

- $y_i f(x_i) > 1$: points outside margin. No contribution to loss.
- $y_i f(x_i) = 1$: points on margin. No contribution to loss (hard margin case)
- $y_i f(x_i) < 1$: points violates margin constraints. Contribute to loss.

Hinge Loss



- A convex approximation to the 0-1 loss.
- It is an upper bound on the 0-1 loss.
- Not differentiable, we need to compute the sub-gradient.

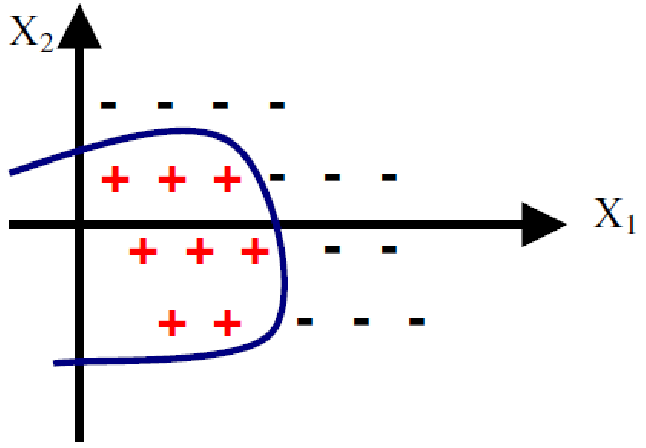
Optimization

- Solving the Quadratic Programming Problems
- Platt's sequential minimal optimization (SMO) algorithm
- Stochastic sub-gradient descent algorithms.

More details about optimization: http://cs229.stanford.edu/summer2020/cs229-notes3.pdf

Non-linear SVM

- Problem: SVM represented with a linear function have very limited representational power, and could not be very useful in practical classification problems.
- How can the above methods be generalized to the case where the decision function is non-linear?
- Good news: With a slight modification, SVM could solve highly nonlinear classification problems!!
- Assumption: Suppose that data set D is nonlinearly separable in the original attribute space. The attribute space can be transformed into a new attribute space where D is linearly separable!



Non-linear SVM

- It turns out that the generalization to a nonlinear boundary can be accomplished in a straightforward way using a simple mathematical trick!
- One major observation on the dual objective function:

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j x_i^T x_j$$

Dot product

$$= \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j < x_i, x_j >$$

- The only way in which the data appear in the training problem is in the form of dot products.

Non-linear SVM

- How about the solution function?
- From $w = \sum_{i=1}^{N} \alpha_i y_i x_i$, the solution function can be written as:

$$
\begin{aligned}
f(x) &= w^T x + b \\
&= \sum_{i}^{N_s} \alpha_i y_i x_i^T x + b \\
&= \sum_{i}^{N_s} \alpha_i y_i < x_i, x > +b
\end{aligned}
$$

where N_s is the number of support vectors.

- In the solution function, the data also appear in the form of dot products where the (x_i)s are the support vectors.

Non-linear SVM

- Now, suppose we first map the data to some high dimension Euclidean space using a mapping Φ (usually $h > d$):
$$\Phi: \mathbb{R}^d \to \mathbb{R}^h$$
- The idea is to enlarge the input space to achieve better training class separation.
- In general, linear boundaries in the enlarged space translate to nonliear boundaries in the original space (true for any nonlinear mapping).

Mapping

- Then, we compute the largest margin hyperplane in the new space $\mathbb{R}^h$.
- Of course, the training algorithm would only depend on the data through dot products in $\mathbb{R}^h$, i.e., $< \Phi(x_i), \Phi(x_j) >$, where $\Phi(x_i) \in \mathbb{R}^h$.
- Suppose we have a function (called **kernel function**) K that computes such dot products in the transformed space:

$$K(x_i, x_j) =< \Phi(x_i), \Phi(x_j) >$$

- Then: all we need in the training algorithm is K, and we would never need to explicitly even know what Φ is.
- Resulting procedure: replace $< \Phi(x_i), \Phi(x_j) >$ with $K(x_i, x_j)$ everywhere in the training algorithm.

Mapping

- The algorithm constructs a linear support vector machine in $\mathbb{R}^h$.
- It achieves the objective in roughly the same amount of time it would take to train on the original data.
- How can we use such a machine? In test phase, given the test points x:

$$f(x) = \sum_{i}^{N_s} \alpha_i y_i < x_i, x > +b \ = \sum_{i}^{N_s} \alpha_i y_i K(x_i, x) + b$$

  where x_i are the support vectors and N_s is the number of support vectors.
- **Kernel trick**: with the kernel function K, we can work with vectors in input space, without even knowing the mapping function Φ.

Example: Kernel Functions

Example: an allowed kernel for which we can construct the mapping Φ:

- Training data are vectors in $\mathbb{R}^2$.
- Suppose we choose $K(x_i, x_j) = (< x_i, x_j >)^2$. We can find a mapping $\Phi: \mathbb{R}^2 \rightarrow \mathbb{R}^h$, such that $(< x_i, x_j >)^2 =< \Phi(x_i), \Phi(x_j) >$
- One such mapping is: $\Phi: \mathbb{R}^2 \rightarrow \mathbb{R}^3$defined as

$$\Phi(x) = \begin{pmatrix} x_1^2 \\ \sqrt{2}x_1x_2 \\ x_2^2 \end{pmatrix}$$

where $x = \binom{x_1}{x_2}$

Example: Kernel Functions

- We can verify that this is indeed the case:

$$K(x, y) = (< x, y >)^2 = (x_1 y_1 + x_2 y_2)^2$$
$$= x_1^2 y_1^2 + x_2^2 y_2^2 + 2 x_1 y_1 x_2 y_2$$

$$< \Phi(x), \Phi(y) >= \Phi(x)^{\mathrm{T}} \Phi(y)$$
$$= \left(x_1^2, \sqrt{2} x_1 x_2, x_2^2\right) \begin{pmatrix} y_1^2 \\ \sqrt{2} y_1 y_2 \\ y_2^2 \end{pmatrix}$$
$$= x_1^2 y_1^2 + x_2^2 y_2^2 + 2 x_1 y_1 x_2 y_2$$

- Note: in general, the mapping Φ and the space $\mathbb{R}^h$ are not unique for a given kernel.

Example: Kernel Functions

- You can verify the following two mappings Φ also satisfy $K(x, y) =< \Phi(x), \Phi(y) >$ for kernel given above.
- Example 1: Φ: $\mathbb{R}^2 \rightarrow \mathbb{R}^3$

$$\Phi(x) = \frac{1}{\sqrt{2}} \begin{pmatrix} x_1^2 - x_2^2 \\ 2x_1x_2 \\ x_1^2 + x_2^2 \end{pmatrix}$$

- Example 2: Φ: $\mathbb{R}^2 \rightarrow \mathbb{R}^4$

$$\Phi(x) = \frac{1}{\sqrt{2}} \begin{pmatrix} x_1^2 \\ x_1x_2 \\ x_1x_2 \\ x_2^2 \end{pmatrix}$$

Properties of a Valid Kernel

- To ensure that a mapping Φ and an expansion $K(x, y) =< \Phi(x), \Phi(y) >$ exist , the mathematical properties that a function K have been studied and are well known as **Mercer theorem**.
- Two popular choices for K are:
  - d^{th} degree polynomial: $K(x, y) = (1 +< x, y >)^d$
  - Radial Basis Function (RBF) Kernel: $K(x, y) = e^{\frac{-\left\|x_i - x_j\right\|^2}{2\sigma^2}}$
- The best choice of a kernel for a given problem is still a research issue; (e.g., Latent Semantic Kernel for document classification)

More details about Mercer theorem: http://cs229.stanford.edu/summer2020/cs229-notes3.pdf

Importance of Parameter γ for Non-linear SVMs

- In general, perfect separation of training data can be achieved in the enlarged space $\mathbb{R}^h$.
- Such perfect separation may lead to the overfitting of the data. As a consequence, the classier will generalize poorly.
- A proper setting of γ allows to avoid overfitting.

Importance of Parameter γ for Non-linear SVMs

- Let's look at the objective function minimized by an SVM:

  $$\frac{1}{2}\|w\|^2 + \gamma \sum_i^N \xi_i$$

  where γ is the penalty factor for errors.
- Large γ $\rightarrow$ discourage any positive ξ_i $\rightarrow$ tendency to overfit the data $\rightarrow$ highly complicated decision boundary in input space.
- Small γ $\rightarrow$ encourage a small value of $\|w\|$ $\rightarrow$larger margin $\rightarrow$ more data on the wrong side of their margin $\rightarrow$ smoother decision boundary in input space.
- In practice: we need to tune γ so to achieve best test error performance.

Speed and Size for Training and Testing

- Training: the evaluation of the dual objective function requires the computation of all dot products $< x_i, x_j > \Rightarrow$ time complexity $O(N^2 d)$ where N is the number of training data and d is the dimension.
- Testing: need to evaluate: $f(x) = \sum_i^{N_s} \alpha_i y_i x_i^T x + b$
- $\Rightarrow$ Time complexity $O(M \cdot N_s)$ where M is the number of operation required to evaluate the kernel.
- $\Rightarrow$ Time complexity $O(d \cdot N_s)$ for RBF kernel, $M = O(d)$.

Readings

1. http://www.personal.psu.edu/sxj937/Notes/Lagrange_Multipliers.pdf
2. http://cs229.stanford.edu/summer2020/cs229-notes3.pdf

Summary of Today's Lecture

- Vector algebra, Formulation, Margin
- SVM for Linear Separable Case
- Non-separable Case, Penalties
- Non-linearity, Kernels