Knowledge Discovery in Data:
Support Vector Machines

Classification using Support Vector Machines (SVMs)

Support Vector Machines (SVMs) are a mathematical model for *binary* classification, i.e., for classifying all elements d from some dataset D into two categories: $C = \{c_1, c_2\}$.

Linear Separability

Consider a dataset $D = \{d_1, \ldots, d_M\}$, $d_i \in \mathcal{R}^n$. Let $C = \{c_1, c_2\}$. Assume, for a moment, that the class function $class(d) : D \rightarrow C$ is defined for all points $d \in D$. We write $D = D_1 \cup D_2$, $D_1 = \{d \in D | class(d) = c_1\}$, $D_2 = \{d \in D | class(d) = c_2\}$.

Linear Separability. The dataset D is called **linearly separable** iff, there exists an $n-1$-dimensional *hyperplane* H, such that all points in set D_1 are located on *one side* of H while all points in set D_2 are located on the other side of H.

Example. Consider the dataset shown on Figure 1. The figure shows multiple hyperplanes (lines) separating the two sets of points.

Separating Hyperplanes. If D is linearly separable, then **any** hyperplane H that separates D_1 from D_2 can be used as a the *classifier*.

Consider the three lines denoted a1, a2 and a2 on Figure 1. Each line has its own $y = kx + b$ equation, and each can be used as the classifier for dataset D.

Question. Given a linearly separable dataset D, what is the *"best"* hyperplane that linearly separates it?

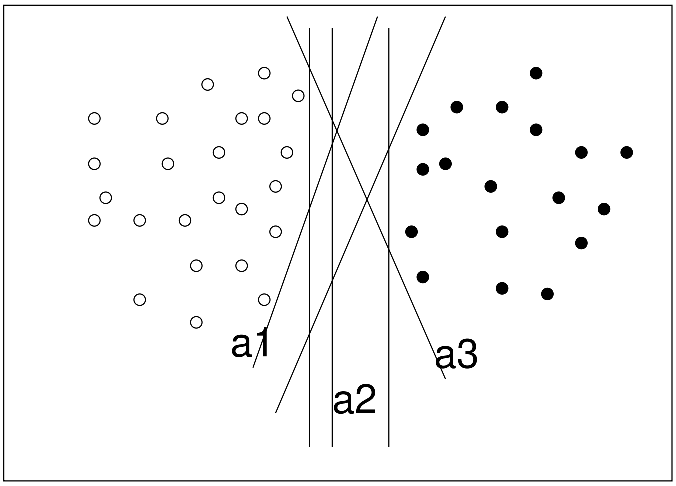

Figure 1: A Linearly separable dataset.

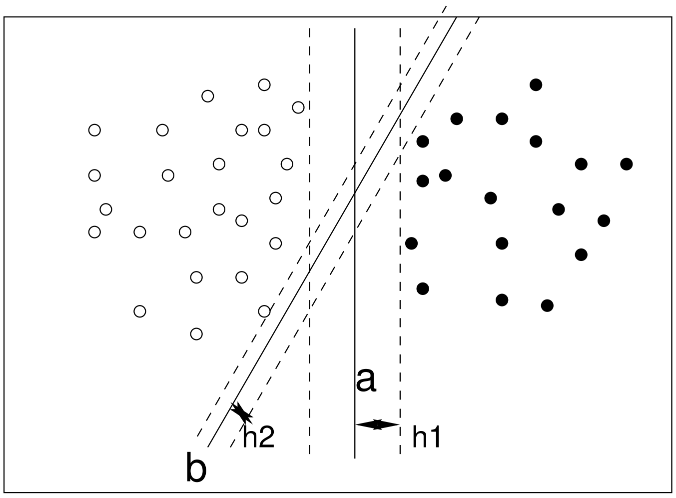

Figure 2: Hyperplane margins.

Classification Margin. Given a hyperplane H, its **margin** (w.r.t., classification) is the maximum number s such that $H+s$ and $H-s$ (i.e., hyperplanes constructed by moving H by $+s$ and $-s$ in the space) do not misclassify any $d \in D$.

The margin of a hyperplane is essentially a **margin of error** that the hyperplane affords us as the *classifier* for D.

Example. Consider Figure 2. It shows two hyperplanes, a and b that linearly separate dataset D. For each hyperplane, its margin (h1 and h2 respectively) is shown. From the example it is clear that h1 $>$ h2.

Maximum Margin Hyperplanes

Goal. Given a (linearly separable) dataset D, find the linearly separating hyperplane with the maximum margin, a.k.a. the **maximum margin hyperplane**.

This hyperplane will be used as **the** *classifier model* for D.

Why maximum margin hyperplanes. Large margins mean fewer misclassifications of unseen data.

Support Vector Machines: Linearly Separable Case

Support Vector Machines is the modeling technique for finding **maximum margin hyperplanes** given datasets (or training sets).

We distinguish two cases: linearly separable dataset D and non-linearly separable dataset D.

Linearly separable case is considered first.

Linear Decision Boundary: the Math

Let training set $T = \{d_1, \ldots d_N\}$. Each training example d_i is denoted as

$$d_i = (x_i, y_i),$$

where $x_i = (x_{i1}, \ldots x_{in})$ is the data point, and $y_i \in \{-1, 1\}$ is the class label.

The linear decision boudary for T can be written as follows:

$$w \cdot x + b = 0.$$

Here, w and b are parameters of the boundary (classifier) model. Our goal is to find the values for w and b that maximize the margin.

If $d^* = (x^*, y^*) \in T$ lies on the decision boundary, then

$$w \cdot x^* + b = 0.$$

If $d' = (x', y') \in T$ lies *above* the decision boundary, then for some $k > 0$,

$$w \cdot x' + b = k.$$

Similarly, if $d'' = (x'', y'') \in T$ lies *below* the decision bondary, then for some $l > 0$,

$$w \cdot x' + b = -l.$$

We assume that for all points $d = (x, y)$, if $w \cdot x + b > 0$, then $y = +1$ and if $w \cdot x + b < 0$ then $y = -1$.

For any unclassified point $x^\star$, we use the linear decision boundary as the classifier as follows:

$$class(x^\star) = y^\star = sign(w \cdot x^\star + b).$$

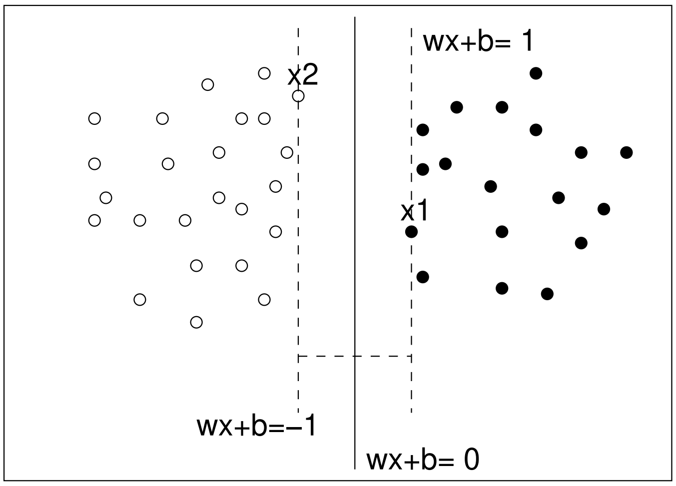

Figure 3: Linear boundaries and thier margins.

Margin of a Linear Boundary

Consider a linear boundary $w \cdot x + b = 0$ for a training set T. The margin of this boundary is determined by two points: the point on the *positive side* of this boundary that is closest to it, and the point on the *negative side* of this boundary that is closest to it.

Using some scaling magic, we can construct two boundaries that are parallel to $w \cdot x + b = 0$, pass through the points described above, and have the following equations:

$$b_{i1} : w \cdot x + b = 1$$

$$b_{i2} : w \cdot x + b = -1$$

See Figure 3 for illustation.

The margin of the hyperplane $w \cdot x + b = 0$ is the distance between b_{i1} and b_{i2}. Let $x1$ be a point from T located on b_{i1} and x_2 be a point from T located on b_{i2}.

Substituting into the equations for b_{i1} and b_{i2} we get:

$$w \cdot (x_1 - x_2) = 2$$

$$||w|| \times d = 2$$

$$d = \frac{2}{||w||}.$$

Here, d is the margin of the hyperplane $w \cdot x + b = 0$.

Learning a Linear SVN model

To find the hyperplane with the best margin, we will estimate the values of w and b from training set data. Our goal is to find w and b such that:

$$w \cdot x_i + b \geq 1 \text{ if } y_i = 1.$$
$$w \cdot x_i + b \leq -1 \text{ if } y_i = -1.$$

From here, we generalize:

$$y_i \cdot (w \cdot x_i + b) \geq 1; i = 1, 2 \ldots N.$$

Given the conditions above, we want the margin to be maximized. This is equivalent to minimizing the following objective function:

$$f(w) = \frac{||w||^2}{2}.$$

Linear Support Vector Machine: Separable Case. A linear support vector machine classification model for a training set $T = \{(x_i, y_i)\}$ is the solution to the following optimization problem:

$$\min_w \frac{||w||^2}{2}$$

subject to

$$y_i \cdot (w \cdot x_i + b) \geq 1; i = 1, 2 \ldots N.$$

Recall that $w = (w_1, \ldots, w_n)$ and $||w|| = \sqrt{w_1^2 + w_2^2 + \ldots + w_n^2}$.

Objective function: **quadratic**.

Constraints: **linear**.

This type of optimization problem is known as **convex optimization** problem. It is solved using the **Lagrange multiplier** method.

Replace the objective function with the following **Lagrangian**:

$$L_P = \frac{1}{2}||w||^2 - \sum_{i=1}^{N} \lambda_i \left(y_i (w \cdot x_i + b) - 1 \right).$$

Here, $\lambda_1, \ldots \lambda_N$ are function parameters called Lagrange multipliers.

The function L_P is now can be **unconditionally** optimized for w and b. (the only conditions are that $\lambda_1 \geq 0$).

To optimize L_P we differentiate is for w and b. The partial derivatives are[1]:

$$\frac{\partial L_P}{\partial w} = 0 \Longrightarrow w = \sum_{i=i}^{N} \lambda_i \cdot y_i \cdot x_i.$$

$$\frac{\partial L_P}{\partial b} = 0 \Longrightarrow \sum_{i=1}^{N} \lambda_i \cdot y_i = 0.$$

[1]Notice that we chose our original objective function to be $||w||^2/2$ to make the partial derivatives simpler.

Because we do not know λ_i, we still cannot solve these. However, we can transform the original $y_i(w \cdot x_i + b) \geq 1$ inequalities into:

$$\lambda_i \cdot (y_i(w \cdot x + b) - 1) = 0.$$
$$\lambda_i \geq 0.$$

This **forces** many Largrange multipliers λ_i to become **0**, because if $y_i(w \cdot x + b) - 1 \neq 0$, then $\lambda_i = 0$.

Support vectors. All vectors x_i, such that $\lambda_i \neq 0$. Points x_i must lie on one of the two marginal hyperplanes b_{i1} or b_{i2}.

Dual problem. We still have a lot of parameters, so solving the original optimization problem is hard.

Instead, we can solve a dual problem: Maximize:

$$L_D = \sum_{i=1}^{N} \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i \lambda_j y_i y_j x_i \cdot x_j.$$

subject to

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

This optimization problem (obtained, but eliminating w and b from the original Lagrangian via the substitutions obtained from partial derivatives)

The latter problem is solved numerically using *quadratic programming* techniques.