CS 4850/6850: Introduction to Machine Learning **Fall 2018**

Topic 4: Support Vector Machines

Instructor: Daniel L. Pimentel-Alarcón

4.1 Introduction

Support vector machines (SVMs) are considered one of the best"out of the box" classifiers. Similar to logistic regression and random forests, their main purpose is prediction/classification, and are used for similar applications. However, SVMs are a more geometric approach. The main idea is to think of each sample as a point in a high-dimensional space, and classify it according to the region where it is located with respect to a *boundary* (see Figure 4.1). The challenge is to find such separating/classifying boundary.

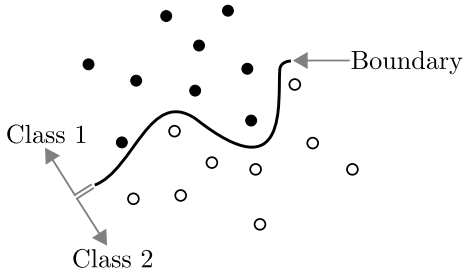


Figure 4.1: An SVM classifies each point according to the region where it is located with respect to a boundary.

4.2 Hyperplanes

Hyperplanes lie at the heart of SVMs. Intuitively, a hyperplane is the generalization of a line and a plane (in 2 and 3 dimensions) to higher dimensions. In words, hyperplanes are subspaces almost as big as the whole space. More formally, a hyperplane $\mathbb{H} \subset \mathbb{R}^{\mathrm{d}}$ is a linear subspace of dimension $\mathrm{d}-1$ (see Figure 4.2 to build some intuition).

Recall from linear algebra that the dimensions of a subspace $\mathbb{U} \subset \mathbb{R}^{\mathrm{d}}$ and its orthogonal complement $\mathbb{U}^{\perp} \subset \mathbb{R}^{\mathrm{d}}$ add to the ambient dimension, i.e.,

$$\mathrm{d} \;=\; \dim(\mathbb{U}) + \dim(\mathbb{U}^{\perp}),$$

which means that $\dim(\mathbb{H}^{\perp}) = 1$, i.e., $\mathbb{H}^{\perp}$ is a line. As such, it is characterized by a single vector. Let $\boldsymbol{\beta} \in \mathbb{R}^{\mathrm{d}}$ be the vector spanning $\mathbb{H}^{\perp}$. Also recall that $\mathbb{U}$ is the collection of all points orthogonal to $\mathbb{U}^{\perp}$. Consequently, we can characterize $\mathbb{H}$ as the collection of all points in $\mathbb{R}^{\mathrm{d}}$ that are orthogonal to $\boldsymbol{\beta}$, i.e.,

$$\Big\{\, \mathbf{x} \in \mathbb{R}^{\mathrm{d}} \;:\; \langle \mathbf{x}, \boldsymbol{\beta} \rangle = 0 \,\Big\},$$

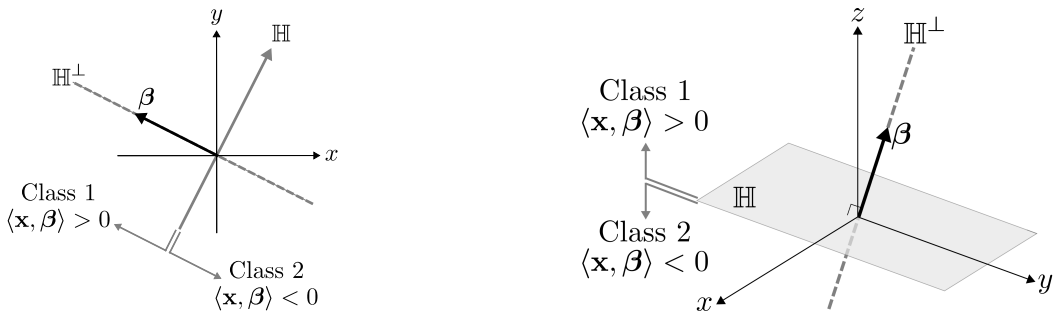
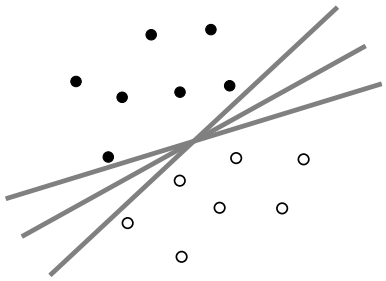


Figure 4.2: Hyperplanes are subspaces of dimension equal to the whole space minus 1. **Left:** A line in $\mathbb{R}^2$ is a hyperplane. **Right:** A plane in $\mathbb{R}^3$ is a hyperplane. Hyperplanes divide the whole space in two halves, providing a natural classifying border. **Question:** Is a line in $\mathbb{R}^3$ a hyperplane? Each hyperplane $\mathbb{H}$ is characterized by the vector $\boldsymbol{\beta}$ spanning its orthogonal complement $\mathbb{H}^\perp$.

where $\langle \mathbf{x}, \boldsymbol{\beta} \rangle = \boldsymbol{\beta}^\mathsf{T}\mathbf{x}$ denotes the inner product (see Figure 4.2 to build some intuition). Hyperplanes are powerful classification tools because they divide the space in two halves: the points $\mathbf{x}$ *above* the hyperplane, meaning $\langle \mathbf{x}, \boldsymbol{\beta} \rangle > 0$, and the points $\mathbf{x}$ *below* the hyperplane, meaning $\langle \mathbf{x}, \boldsymbol{\beta} \rangle < 0$ (see Figure 4.2).

4.3 Maximal Margin Classifier

SVMs happen to be a generalization of the maximal margin classifier (MMC), which uses a *hyperplane* as the separating/classifying border. Notice, however, that given some data, it is possible that there are infinitely many hyperplanes that could separate it, for example:

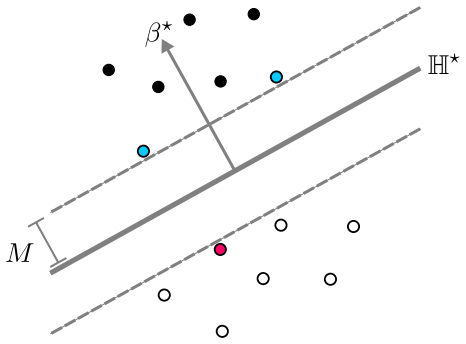


As the name suggests, the MMC finds the hyperplane that separates data with the highest possible margin M. More precisely, given data points $\mathbf{x}_1, \ldots, \mathbf{x}_\mathrm{n} \in \mathbb{R}^\mathrm{d}$, with labels/classes $\mathrm{y}_1, \ldots, \mathrm{y}_\mathrm{n} \in \{-1, 1\}$, the MMC aims to find:

$$\boldsymbol{\beta}^\star := \underset{\substack{\boldsymbol{\beta} \in \mathbb{R}^\mathrm{d}: \\ \|\boldsymbol{\beta}\| = 1}}{\arg\max}\ M \qquad \text{subject to} \qquad \mathrm{y}_\mathrm{i} \langle \mathbf{x}_\mathrm{i}, \boldsymbol{\beta} \rangle \geq M \quad \forall\ \mathrm{i} = 1, \ldots, \mathrm{n}. \tag{4.1}$$

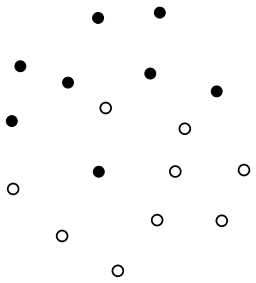
The intuition behind (4.1) is quite simple. First, M denotes the margin that we want to maximize. The first constraint $\|\boldsymbol{\beta}\| = 1$ simply guarantees that we do not cheat by making $\boldsymbol{\beta}$ too large. The second constraint $\mathrm{y}_\mathrm{i} \langle \mathbf{x}_\mathrm{i}, \boldsymbol{\beta} \rangle \geq M$ ensures that each training point is on the right side of the hyperplane (because it requires that the sign of the *true* label y_i and the *classification* $\langle \mathbf{x}_\mathrm{i}, \boldsymbol{\beta} \rangle$ are equal) and also ensures that the perpendicular distance between $\mathbf{x}_\mathrm{i}$ and the hyperplane (given by $\mathrm{y}_\mathrm{i} \langle \mathbf{x}_\mathrm{i}, \boldsymbol{\beta} \rangle$) is larger than the margin M.

All vectors that are exactly within M distance of the hyperplane are called *support vectors*:



4.4 Support Vector Classifier

It is pretty clear that a hyperplane may not classify two classes perfectly:

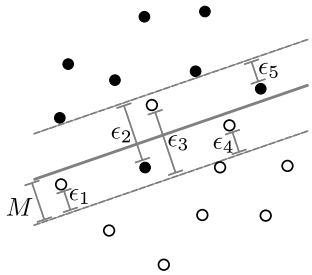


The support vector classifier (SVC), also known as the soft margin classifier, arises as an idea to address this problem. They key idea is to allow some wiggle room by letting some points to be misclassified, using the following optimization instead of (4.1)

$$\boldsymbol{\beta}^\star := \underset{\substack{\boldsymbol{\beta}\in\mathbb{R}^{\mathrm{d}},\ \boldsymbol{\epsilon}\in\mathbb{R}^{\mathrm{d}}:\\ \|\boldsymbol{\beta}\|=1,\ \|\boldsymbol{\epsilon}\|\leq E}}{\arg\max} \quad M \qquad \text{subject to} \qquad \begin{cases} \mathrm{y_i}\langle \mathbf{x}_\mathrm{i}, \boldsymbol{\beta}\rangle \geq M(1-\epsilon_\mathrm{i}) \\ \epsilon_\mathrm{i} \geq 0 \end{cases} \qquad \forall\ \mathrm{i} = 1,\ldots,\mathrm{n}. \tag{4.2}$$

There are a few differences between (4.1) and (4.2). First, we are now also maximizing over $\boldsymbol{\epsilon}$, which models the wiggle room that will be allowed in our sample. The constraint $\|\boldsymbol{\epsilon}\| \leq E$ guarantees that the overall wiggle room does not exceed the overall allowed error E, which is a tuning parameter. The constraint $\mathrm{y_i}\langle \mathbf{x}_\mathrm{i}, \boldsymbol{\beta}\rangle \geq M(1-\epsilon_\mathrm{i})$ ensures that the orthogonal distance from $\mathrm{x_i}$ and the hyperplane is larger than M, or only *a little* smaller; how *little* is determined by ϵ_i, which should be greater than zero (do you understand why?).

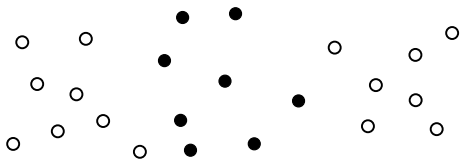
This type of approach is often called a *relaxation*, and will allow some points to be inside the margin, and even on the wrong side of the hyperplane, as in the following figure:



,

4.5 Nonlinear Boundaries

It is pretty clear that not all datasets can be divided by a linear boundary:



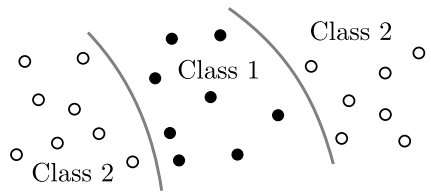
Hence, instead of classifying each point using the linear function $\langle \mathbf{x}, \boldsymbol{\beta} \rangle = \beta_1 \mathrm{x}_1 + \beta_2 \mathrm{x}_2 + \cdots + \beta_\mathrm{d} \mathrm{x}_\mathrm{d}$, we can use other non-linear functions, for example quadric (degree-2) polynomials:

$$\begin{aligned} f_2(\mathbf{x}) \;:=\; & \beta_{11}\mathrm{x}_1^2 + \beta_{12}\mathrm{x}_1\mathrm{x}_2 + \beta_{13}\mathrm{x}_1\mathrm{x}_3 + \cdots + \beta_{1\mathrm{d}}\mathrm{x}_1\mathrm{x}_\mathrm{d} && (4.3)\\ + \;& \beta_{22}\mathrm{x}_2^2 + \beta_{23}\mathrm{x}_2\mathrm{x}_3 + \cdots + \beta_{2\mathrm{d}}\mathrm{x}_2\mathrm{x}_\mathrm{d} && (4.4)\\ + \;& \vdots && (4.5)\\ + \;& \beta_{\mathrm{dd}}\mathrm{x}_d^2. && (4.6) \end{aligned}$$

This suggests extending the SVC formulation in (4.2) to:

$$\boldsymbol{\beta}^\star \;:=\; \underset{\substack{\boldsymbol{\beta}\in\mathbb{R}^\mathrm{d},\ \boldsymbol{\epsilon}\in\mathbb{R}^\mathrm{d};\\ \|\boldsymbol{\beta}\|=1,\ \|\boldsymbol{\epsilon}\|\le E}}{\arg\max} \quad M \qquad \text{subject to} \qquad \begin{cases} \mathrm{y_i} f_2(\mathbf{x}_\mathrm{i}) \ge M(1-\epsilon_\mathrm{i}) \\ \epsilon_\mathrm{i} \ge 0 \end{cases} \qquad \forall\ \mathrm{i} = 1, \ldots, \mathrm{n}. \tag{4.7}$$

The only difference with (4.2) is that now we use $f_2(\mathbf{x}_\mathrm{i})$ instead of $\langle \mathbf{x}_\mathrm{i}, \boldsymbol{\beta} \rangle$. Intuitively, this is guaranteeing that the true label $\mathrm{y_i}$ and the prediction $f_2(\mathbf{x}_\mathrm{i})$ agree by more than $(1 - \epsilon_\mathrm{i})$ of the margin M. This will produce a non-linear boundary:



Letting

$$\boldsymbol{\beta}\otimes\boldsymbol{\beta} \;:=\; \begin{bmatrix}\beta_1\boldsymbol{\beta}\\ \beta_2\boldsymbol{\beta}\\ \vdots\\ \beta_{\mathrm{d}}\boldsymbol{\beta}\end{bmatrix} \;=\; \left[\begin{array}{c}\beta_1\beta_1\\ \beta_1\beta_2\\ \vdots\\ \beta_1\beta_{\mathrm{d}}\\ \hline \beta_2\beta_1\\ \beta_2\beta_2\\ \beta_2\beta_3\\ \vdots\\ \beta_2\beta_{\mathrm{d}}\\ \hline \vdots\\ \hline \beta_{\mathrm{d}}\boldsymbol{\beta}_1\\ \beta_{\mathrm{d}}\boldsymbol{\beta}_2\\ \vdots\\ \beta_{\mathrm{d}}\boldsymbol{\beta}_{\mathrm{d}}\end{array}\right],$$

and similarly for $\mathbf{x}\otimes\mathbf{x}$, we can rewrite (4.7) as

$$\boldsymbol{\beta}^\star \;:=\; \underset{\substack{\boldsymbol{\beta}\in\mathbb{R}^{\mathrm{d}},\ \boldsymbol{\epsilon}\in\mathbb{R}^{\mathrm{d}};\\ \|\boldsymbol{\beta}\|=1,\ \|\boldsymbol{\epsilon}\|\le E}}{\arg\max}\ M \qquad \text{subject to} \qquad \begin{cases}\mathrm{y_i}\langle\mathbf{x_i}\otimes\mathbf{x_i},\boldsymbol{\beta}\otimes\boldsymbol{\beta}\rangle \ge M(1-\epsilon_{\mathrm{i}})\\ \epsilon_{\mathrm{i}}\ge 0\end{cases} \qquad \forall\ \mathrm{i}=1,\dots,\mathrm{n}, \quad (4.8)$$

which is linear in the *lifted* vectors $\mathbf{x}\otimes\mathbf{x}$ and $\boldsymbol{\beta}\otimes\boldsymbol{\beta}$. Consequently, by simply multiplying our data by itself, we can construct non-linear boundaries with the same linear techniques as before. The downside is that the number of parameters increases polynomially. For instance, instead of the d parameters (corresponding to the entries in $\boldsymbol{\beta}$) we now have $\binom{\mathrm{d}}{2} = \mathrm{d}(\mathrm{d}-1)/2$ parameters, corresponding to the entries $\boldsymbol{\beta}\otimes\boldsymbol{\beta}$ (Notice that $\boldsymbol{\beta}\otimes\boldsymbol{\beta}\in\mathbb{R}^{\mathrm{d}^2}$, but some entries are duplicated; in total there are $\binom{\mathrm{d}}{2}$ distinct entries).

4.6 Kernels and Support Vector Machines

Kernels are functions that quantify the similarity between two vectors. For example, if you remember from your linear algebra 101 class, the inner product is a kernel that measures angle as follows:

$$\angle(\mathbf{x},\boldsymbol{\beta}) \;=\; \frac{\langle\mathbf{x},\boldsymbol{\beta}\rangle}{\|\mathbf{x}\|\|\boldsymbol{\beta}\|}$$

There are many other kernels (functions that measure similarity between vectors), for example:

$$K(\mathbf{x},\boldsymbol{\beta}) \;=\; \langle\mathbf{x}\otimes\mathbf{x},\boldsymbol{\beta}\otimes\boldsymbol{\beta}\rangle,$$

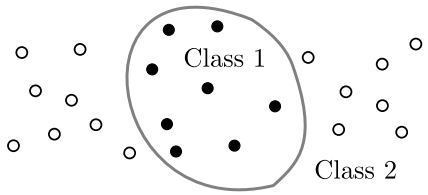
which is precisely the function that we used in (4.8), corresponding to the quadric polynomial in (4.3). The more similar $\mathbf{x}$ and $\boldsymbol{\beta}$ are, the larger $K(\mathbf{x},\boldsymbol{\beta})$ will be. We can extend this idea to obtain a polynomial kernel of degree r:

$$K(\mathbf{x},\boldsymbol{\beta}) \;=\; (1+\langle\mathbf{x},\boldsymbol{\beta}\rangle)^{\mathrm{r}} \;=\; \left(1+\sum_{\mathrm{i}=1}^{\mathrm{d}}\mathrm{x_i}\beta_{\mathrm{i}}\right)^{\mathrm{r}}.$$

Kernels are not necessarily only polynomial functions. For example, another popular choice is the *radial* kernel:

$$K(\mathbf{x}, \boldsymbol{\beta}) \;=\; e^{-\gamma \|\mathbf{x}-\boldsymbol{\beta}\|^2} \;=\; \exp\left(-\gamma \sum_{\mathrm{i}=1}^{\mathrm{d}} (\mathrm{x_i} - \beta_\mathrm{i})^2\right),$$

which will produce a boundary like the following:



Depending on the dataset at hand, one kernel may be better than another. The support vector machine (SVM) is simply a generalization of the SVC that uses a kernel function:

$$\boldsymbol{\beta}^\star \;:=\; \underset{\substack{\boldsymbol{\beta}\in\mathbb{R}^\mathrm{d},\ \boldsymbol{\epsilon}\in\mathbb{R}^\mathrm{d};\\ \|\boldsymbol{\beta}\|=1,\ \|\boldsymbol{\epsilon}\|\le E}}{\arg\max} \quad M \qquad \text{subject to} \qquad \begin{cases} \mathrm{y_i} K(\mathbf{x_i}, \boldsymbol{\beta}) \ge M(1-\epsilon_\mathrm{i}) \\ \epsilon_\mathrm{i} \ge 0 \end{cases} \qquad \forall\ \mathrm{i} = 1, \ldots, \mathrm{n}. \tag{4.9}$$