Chapter 1:
Linear Classifiers & SVM

Classification is the problem of identifying to which of a set of categories a new observation belongs, on the basis of a training set of data elements whose category membership is known. In this Chapter we will consider a simplification of classification, namely binary classification. In binary classification the set of categories has cardinality 2. In the context of binary classification we will consider linear classifiers which are fairly simple but still work well surprisingly often.

1.1 Formal Setting

Let $\mathcal{X}$ (**input space**) and $\mathcal{Y}$ (**label space**) be some sets.

A given set of the form $M := \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\} \subseteq \mathcal{X} \times \mathcal{Y}$ is called **training data**, where $x_1, \ldots x_n$ are our **inputs** and $y_1, \ldots y_n$ are the corresponding **labels**.

Our goal is to write a function $f : \mathcal{X} \to \mathcal{Y}$ (**prediction function**), which correctly predicts the label of yet unseen data with the help of our training data.

We call the elements of $\mathcal{Y}$ **classes**. If there are finitely many classes, we also call f a **classifier**.

The algorithm used to find such an f, with the help of the training data is called **learning machine** or **learning algorithm**. This process is also called **training**.

Unless otherwise stated, our setting will be $\mathcal{X} = \mathbb{R}^d$, $\mathcal{Y} = \{-1, 1\}$ and
$M := \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\} \subseteq \mathcal{X} \times \mathcal{Y}$ as our training data. This is the binary classification setting.

1.2 Linear Classifier

Definition: Linear Classifier *A classifier of the form* $f(\mathbf{x}) = \operatorname{sign}(\mathbf{w}^T\mathbf{x} + b)$ *is called a* linear classifier*.*

Example: Nearest centroid classifier (NCC) The idea of NCC is the following: Given our training data, look at the two cluster
$A := \{x_i \, : (x_i, y_i) \in M, \, y_i = -1\}$, $B := \{x_i \, : (x_i, y_i) \in M, \, y_i = +1\}$.

These clusters partition our training data. To find the label of a new datapoint, we check whether the new datapoint is closer to the centroid of cluster A or the centroid of cluster B and give it the label -1 or $+1$ respectively.

Algorithm:

- **Training:** Compute the centroids $\mathbf{c}_{-1} = \dfrac{1}{|A|} \sum_{x \in A} \mathbf{x}$, $\mathbf{c}_{+1} = \dfrac{1}{|B|} \sum_{x \in B} \mathbf{x}$
- **Prediction:** Given a new datapoint x_k, set $y_k := \arg\min_{l \in \{-1,+1\}} \|\mathbf{c}_l - \mathbf{x}\|$

Claim: NCC is a linear classifier

Proof:
We want to write the condition whether a new datapoint is nearer to c_{-1} or to c_{+1} in the form sign($\mathbf{w}^T\mathbf{x} + b = 0$). As NCC classifies points based on which centroid is nearer, points that are equidistant to both centroids should be on the sought hyperplane.

So we want that $\mathbf{w}^T\mathbf{x} + b = 0 \iff \|\mathbf{x} - \mathbf{c}_{-1}\| = \|\mathbf{x} - \mathbf{c}_{+1}\|$. By expanding the right side we get:

$$
\begin{aligned}
& \|\mathbf{x} - \mathbf{c}_{-1}\| = \|\mathbf{x} - \mathbf{c}_{+1}\| \\
\iff & \|\mathbf{x} - \mathbf{c}_{-1}\|^2 = \|\mathbf{x} - \mathbf{c}_{+1}\|^2 \\
\iff & \sqrt{\sum_{i=1}^{d} (\mathbf{x}_i - \mathbf{c}_{-1i})^2}^{\,2} = \sqrt{\sum_{i=1}^{d} (\mathbf{x}_i - \mathbf{c}_{+1i})^2}^{\,2} \\
\iff & \sum_{i=1}^{d} \mathbf{x}_i^2 - \sum_{i=1}^{d} 2\mathbf{x}_i\mathbf{c}_{-1i} + \sum_{i=1}^{d} \mathbf{c}_{-1i}^2 = \sum_{i=1}^{d} \mathbf{x}_i^2 - \sum_{i=1}^{d} 2\mathbf{x}_i\mathbf{c}_{+1i} + \sum_{i=1}^{d} \mathbf{c}_{+1i}^2 \\
\iff & \|\mathbf{x}\|^2 - 2\mathbf{x}^T\mathbf{c}_{-1} + \|\mathbf{c}_{-1}\|^2 = \|\mathbf{x}\|^2 - 2\mathbf{x}^T\mathbf{c}_{+1} + \|\mathbf{c}_{+1}\|^2 \\
\iff & -2\mathbf{x}^T\mathbf{c}_{-1} + 2\mathbf{x}^T\mathbf{c}_{+1} + \|\mathbf{c}_{-1}\|^2 - \|\mathbf{c}_{+1}\|^2 = 0 \\
\iff & \underbrace{2\left(\mathbf{c}_{+1} - \mathbf{c}_{-1}\right)^T}_{=:\mathbf{w}} \mathbf{x} + \underbrace{\|\mathbf{c}_{-1}\|^2 - \|\mathbf{c}_{+1}\|^2}_{=:b} = 0. \quad \square
\end{aligned}
$$

This shows that NCC is a linear classifier.
Notice that, $\mathbf{w} := 2\left(\mathbf{c}_{-1} - \mathbf{c}_{+1}\right)^T$ and $b := \|\mathbf{c}_{+1}\|^2 - \|\mathbf{c}_{-1}\|^2$, also works but the two hyperplanes have swapped signed distance. To see which one matches with our setting, it suffices to plug in the centroids and see whether the labels match.

1.3 Support Vector Machine (SVM)

We seek the best hyperplane seperating our data, to this end consider figure 1 (left). Intuitively the best (fairest) hyperplane would be the one that separates the data with the largest margin, visualized in figure 1 (right). We can formalize this thought mathematically:

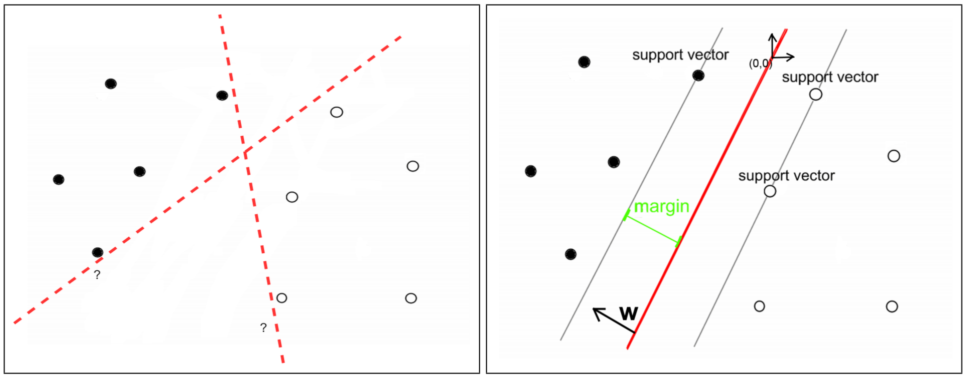

Figure 1: Visualization of linear classifiers. Datapoints with label -1 are black and $+1$ are white. Some differing classifiers (left). The hard-margin SVM H and its margin hyperplanes H_+ and H_- (right).

Formalization:
Assume our dataset is linearly separable. Let

$$H = \{\mathbf{x} \in \mathbb{R}^d | \mathbf{w}^T \mathbf{x} + b = 0\}$$

$\mathbf{w} \neq 0$ be a hyperplane. The margin γ is the size of the open space around the hyperplane that no points occupy. Its boundaries on either side can be described by

$$H_+ = \{\mathbf{x} \in \mathbb{R}^d | \mathbf{w}^T \mathbf{x} + b = \gamma\}$$

and

$$H_- = \{\mathbf{x} \in \mathbb{R}^d | \mathbf{w}^T \mathbf{x} + b = -\gamma\}.$$

We additionally require all points with label $+1$ to be on "above" H_+ and all points with label -1 to be "below" H_-. Our goal is to maximize γ by choosing appropriate $\mathbf{w}$ and b. Recall the signed distance from **Prop 0.2**

$$d(\mathbf{x}, H) = \frac{1}{\|\mathbf{w}\|} \left(\mathbf{w}^\top \mathbf{x} + b \right).$$

We wish to not just choose any hyperplane, we choose our hyperplane such that the classification is done correctly. Since we will use the signed distance to classify we need

$$d(\mathbf{x_i}, H) \geq 0 \text{ if } y_i = +1$$
$$d(\mathbf{x_i}, H) \leq 0 \text{ if } y_i = -1$$

We simplify these 2 constraints by the new constraint:

$$y_i \cdot d(\mathbf{x_i}, H) \geq 0. \tag{1}$$

Finally, the distance between any point and the hyperplane should be at least γ, giving the constraint:

$$y_i \cdot d(\mathbf{x_i}, H) \geq \gamma \iff y_i \left(\mathbf{w}^\top \mathbf{x}_i + b \right) \geq \|\mathbf{w}\| \gamma \tag{2}$$

Lastly, we make the observation that if (2) is satisfied (1) is trivially satisfied as $\gamma \geq 0$, because γ=0 is a trivial lowerbound of our maximization problem. Giving us our maximization problem:

$$\begin{aligned} \max_{\gamma, b \in \mathbb{R}, \mathbf{w} \in \mathbb{R}^d \setminus \{0\}} & \gamma \\ \text{s.t. } y_i \left(\mathbf{w}^\top \mathbf{x}_i + b \right) & \geq \|\mathbf{w}\| \gamma, \quad \forall i = 1, \ldots, n \end{aligned} \tag{3}$$

Definition: Hard Margin SVM *The mathematical program (3) is called the Hard Margin SVM.*
We state a few problems with the Hard Margin SVM. Firstly datapoints are not always linearly separable. For instance:

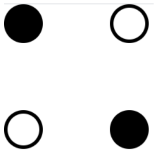

Figure 2: A set of data points that isn't linearly seperable

Also outlier points can potentially corrupt the SVM.

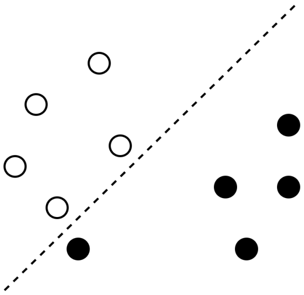

Figure 3: Outlier point is forcing us to choose a suboptimal hyperplane.

To fix this issue we look at (3). A first idea would be to allow some freedom. Not every datapoint needs to have γ distance to the hyperplane. If
$\xi_i \geq 0 \quad \forall i = 1, \ldots n$ then we can formalize this by:

$$y_i \left(\mathbf{w}^\top \mathbf{x}_i + b \right) \geq \|\mathbf{w}\| \gamma - \xi_i, \quad \forall i = 1, \ldots n$$

By choosing the ξ_i big enough our problem becomes unbounded. To not let this happen we need to somehow penalize our maximization function, every time we use ξ_i. One way could be:

$$\max_{\gamma, b \in \mathbb{R}, \mathbf{w} \in \mathbb{R}^d \setminus \{0\}, \xi_1, \ldots, \xi_n \geq 0} \quad \gamma - C \sum_{i=1}^{n} \xi_i$$

where C is a constant telling us how harsh to penalize. The higher C the more we penalize violations. Finally we get:

$$\begin{aligned} \max_{\gamma, b \in \mathbb{R}, \mathbf{w} \in \mathbb{R}^d \setminus \{0\}, \xi_1, \dots, \xi_n \geq 0} \quad & \gamma - C \sum_{i=1}^{n} \xi_i && \\ \text{s.t.} \quad & y_i \left(\mathbf{w}^\top \mathbf{x}_i + b \right) \geq \|\mathbf{w}\| \gamma - \xi_i, && \forall i = 1, \dots n \\ & \xi_i \geq 0 && \forall i = 1, \dots n \end{aligned} \tag{4}$$

Definition: Soft Margin SVM *The mathematical program (4) is called Soft Margin SVM.*

Denote by γ^*, $\mathbf{w}^*$ and $b*$ the optimal arguments of (4).

Definition: Support Vector *All vectors $\mathbf{x}_i$ with $y_i \cdot d\left(\mathbf{x}_i, H\left(\mathbf{w}^*, b^*\right)\right) \leq \gamma^*$ are called support vectors.*

Observation: We notice that (4) allows the existence of support vectors. Also notice that the value of (4) only depends on these support vectors. Removing all other datapoints would not have an effect on the outcome of (4) because if $\mathbf{x_i}$ is not a support vector, then $\xi_i = 0$.