Machine Learning and Intelligent Systems

Support Vector Machines

Maria A. Zuluaga

Nov 17, 2023

EURECOM - Data Science Department

Table of contents

Recap: The Perceptron

The Perceptron

Data Assumptions:

- Binary classification : $y_i \in \{-1, 1\}$
- Data is linearly separable

Model Assumption:

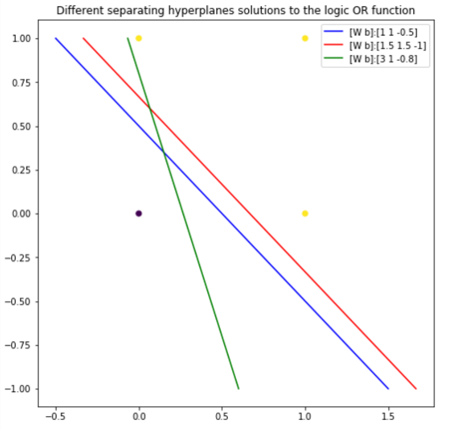
- The decision boundary is a hyperplane:

$$\mathcal{H} = \{\mathbf{x} : \hat{w}^T \mathbf{x} + b = 0\}$$

- $\mathbf{w}$: Weight vector that defines the hyperplane
- b: bias

Goal: Find a separating hyperplane by minimizing the number of errors, i.e. number of points in the "wrong" side of the decision boundary

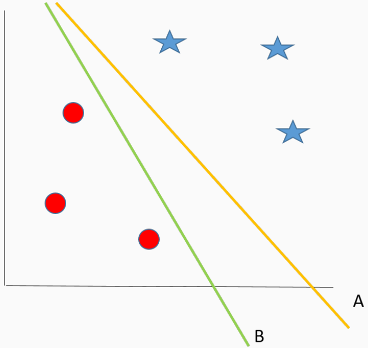
Initialization



Different initializations lead to different solutions

Choosing the Decision Boundary

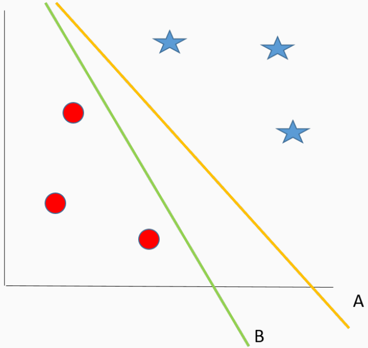
Training set



- Which of these to boundaries is a valid solution for the perceptron?

Choosing the Decision Boundary

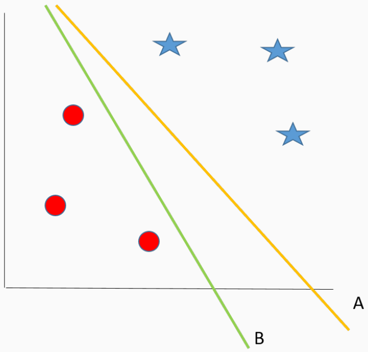
Training set



- Which of these to boundaries is a valid solution for the perceptron?
  - Answer: Both are valid as the perceptron loss is zero

Choosing the Decision Boundary

Training set

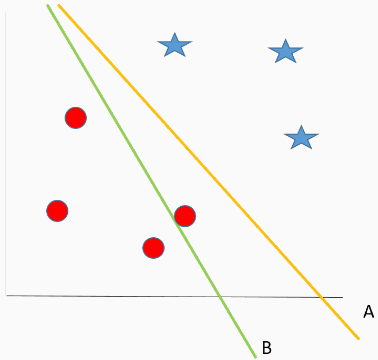


- Which of these to boundaries is a valid solution for the perceptron?
  - Answer: Both are valid as the perceptron loss is zero

- Which one is best according to you?

Choosing the Decision Boundary

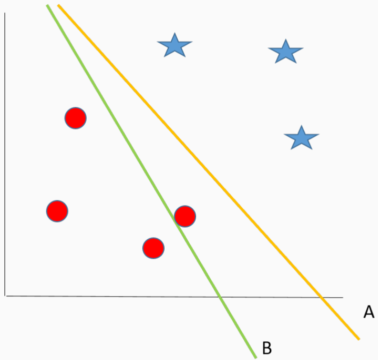
Training set and a test point



- Which of these to boundaries is a valid solution for the perceptron?
    - Answer: Both are valid as the perceptron loss is zero
- Which one is best according to you?

Choosing the Decision Boundary

Training set and a test point



- Which of these to boundaries is a valid solution for the perceptron?
  - Answer: Both are valid as the perceptron loss is zero
- Which one is best according to you?
  - Answer: Boundary A seems more "reliable"
- **Intuition:** The best boundary is the one that is as far as possible from both classes

Support Vector Machines and The Percetron

- A Support Vector Machine (SVM) makes predictions exactly like the perceptron
- SVMS, however, try to find a boundary that is as far as possible from both classes
- SVM also differs in the way in which it learns the parameters

Support Vector Machines and The Percetron

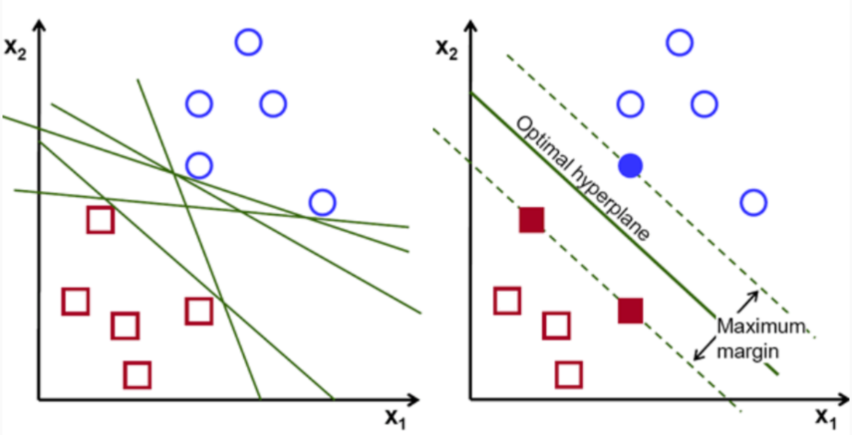
- A Support Vector Machine (SVM) makes predictions exactly like the perceptron
- SVMS, however, try to find a boundary that is as far as possible from both classes
- SVM also differs in the way in which it learns the parameters
- Reminder

$$E_p(\mathbf{w}) = -\sum_{i \in \mathcal{M}} \mathbf{w}^T \mathbf{x}_i y_i$$

The perceptron criterion

Maximum Margin Classifiers

Maximum Margin Classification



SVMs aim to find the boundary that maximizes the margin between the classes

Source: `https://towardsdatascience.com/svm-feature-selection-and-kernels-840781cc1a6c`

Assumptions and Definitions

Data Assumptions:

- Binary classification : $y_i \in \{-1, 1\}$
- Data is linearly separable

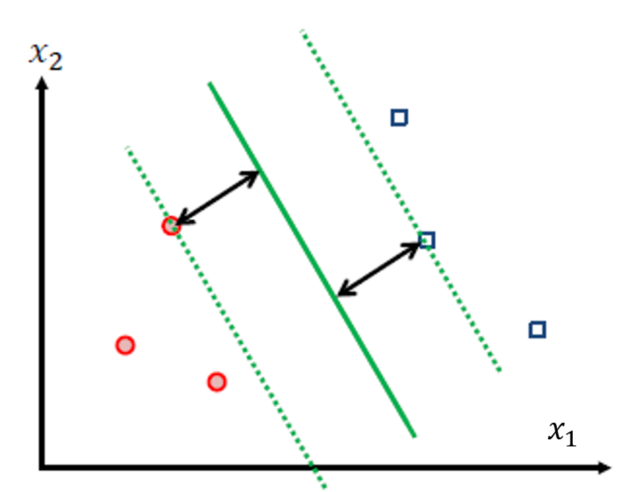
Goal:

- Maximize the margin width

Definitions:

- **Margin width:** Distance between the decision boundary and the nearest points on either class
- **Support vectors:** Points that define the location of the decision boundary

Support Vectors



Parenthesis: Linear Algebra of a Hyperplane

$\mathcal{H}$: Hyperplane or affine set defined by:

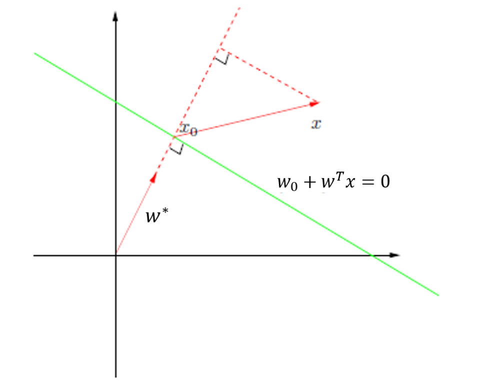
$$\mathcal{H} = \{\mathbf{x} : \mathbf{w}^T \mathbf{x} + w_0 = 0\}$$

Key properties:

1. For any $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{H}$

$$\mathbf{w}^T(\mathbf{x}_1 - \mathbf{x}_2) = 0$$

so, $\mathbf{w}^* = \dfrac{\mathbf{w}^T}{\|\mathbf{w}\|}$ is the vector normal to $\mathcal{H}$



Adapted from Fig 4.15 Hastie et al. ESL

Parenthesis: Linear Algebra of a Hyperplane

$\mathcal{H}$: Hyperplane or affine set defined by:

$$\mathcal{H} = \{\mathbf{x} : \mathbf{w}^T\mathbf{x} + w_0 = 0\}$$

Key properties:

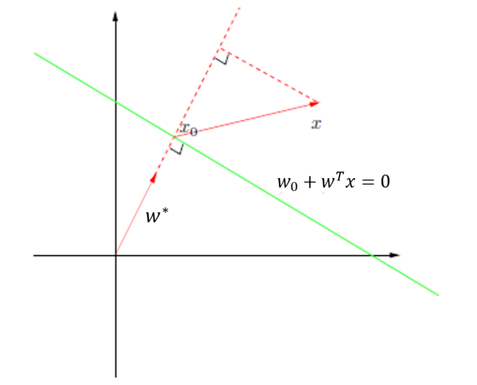
1. For any $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{H}$

$$\mathbf{w}^T(\mathbf{x}_1 - \mathbf{x}_2) = 0$$

so, $\mathbf{w}^* = \frac{\mathbf{w}^T}{\|\mathbf{w}\|}$ is the vector normal to $\mathcal{H}$

2. For any $\mathbf{x}_0 \in \mathcal{H}$,

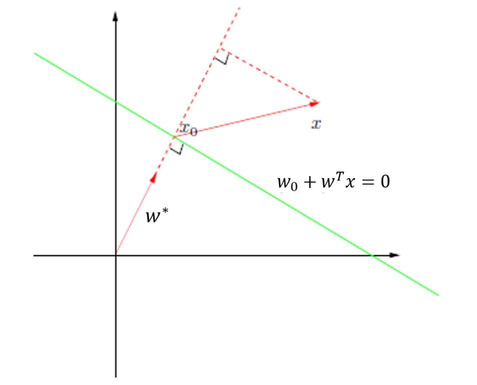
$$\mathbf{w}^T\mathbf{x}_0 = -w_0$$



Adapted from Fig 4.15 Hastie et al. ESL

Distance of any point $\mathbf{x}$ to the hyperplane

The **signed distance** of any point $\mathbf{x}$ to $\mathcal{H}$ is the projection of a vector $\mathbf{v}$ into the normal vector.



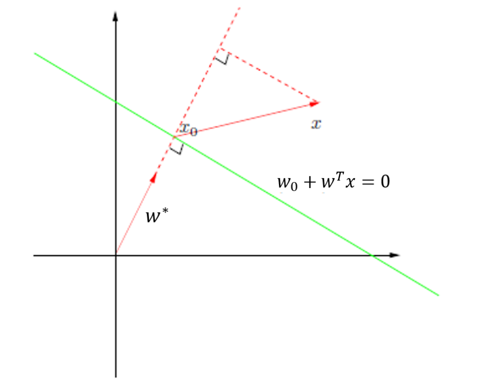
Adapted from Fig 4.15 Hastie et al. ESL

Distance of any point x to the hyperplane

The **signed distance** of any point **x** to $\mathcal{H}$ is the projection of a vector **v** into the normal vector.

We can obtain this projection using the dot product:

$$\mathbf{w}^* \cdot \mathbf{v} = \frac{\mathbf{w}^T}{\|\mathbf{w}\|}(\mathbf{x} - \mathbf{x}_0) \qquad \text{(key property 1)}$$



Adapted from Fig 4.15 Hastie et al. ESL

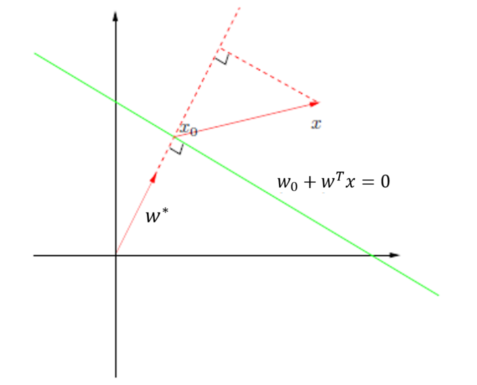
Distance of any point x to the hyperplane

The **signed distance** of any point **x** to $\mathcal{H}$ is the projection of a vector **v** into the normal vector.

We can obtain this projection using the dot product:

$$\mathbf{w}^* \cdot \mathbf{v} = \frac{\mathbf{w}^T}{\|\mathbf{w}\|}(\mathbf{x} - \mathbf{x}_0) \qquad \text{(key property 1)}$$

$$= \frac{1}{\|\mathbf{w}\|}(\mathbf{w}^T\mathbf{x} - \mathbf{w}^T\mathbf{x}_0) \quad \text{(key property 2)}$$



Adapted from Fig 4.15 Hastie et al. ESL

Distance of any point x to the hyperplane

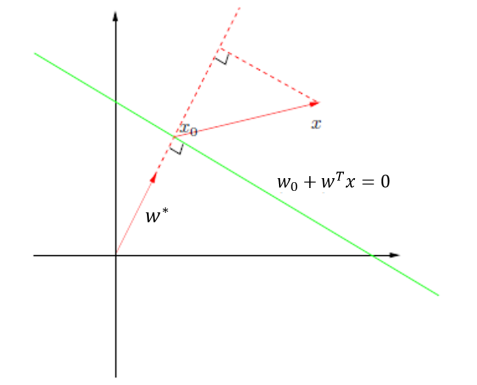
The **signed distance** of any point **x** to $\mathcal{H}$ is the projection of a vector **v** into the normal vector.

We can obtain this projection using the dot product:

$$\mathbf{w}^* \cdot \mathbf{v} = \frac{\mathbf{w}^T}{\|\mathbf{w}\|}(\mathbf{x} - \mathbf{x}_0) \qquad \text{(key property 1)}$$

$$= \frac{1}{\|\mathbf{w}\|}(\mathbf{w}^T\mathbf{x} - \mathbf{w}^T\mathbf{x}_0) \quad \text{(key property 2)}$$

$$= \frac{1}{\|\mathbf{w}\|}(\mathbf{w}^T\mathbf{x} + \mathbf{w}_0)$$



Adapted from Fig 4.15 Hastie et al. ESL

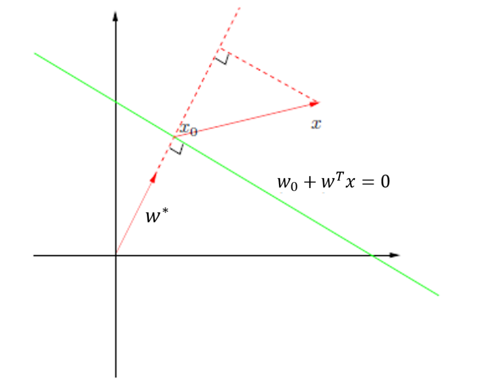
Distance of any point $\mathbf{x}$ to the hyperplane

The **signed distance** of any point $\mathbf{x}$ to $\mathcal{H}$ is the projection of a vector $\mathbf{v}$ into the normal vector.

We can obtain this projection using the dot product:

$$\begin{aligned}
\mathbf{w}^* \cdot \mathbf{v} &= \frac{\mathbf{w}^T}{\|\mathbf{w}\|}(\mathbf{x} - \mathbf{x}_0) \qquad \text{(key property 1)} \\
&= \frac{1}{\|\mathbf{w}\|}(\mathbf{w}^T\mathbf{x} - \mathbf{w}^T\mathbf{x}_0) \quad \text{(key property 2)} \\
&= \frac{1}{\|\mathbf{w}\|}(\mathbf{w}^T\mathbf{x} + \mathbf{w}_0)
\end{aligned}$$

Signed distance to $\mathcal{H}$: $\dfrac{1}{\|\mathbf{w}\|}(\mathbf{w}^T\mathbf{x} + \mathbf{w}_0)$

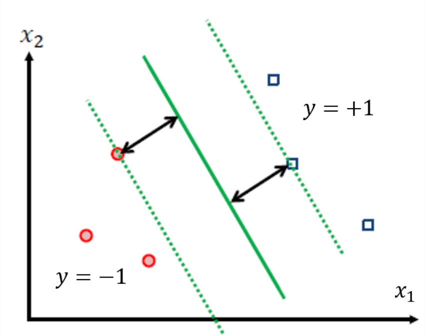


Adapted from Fig 4.15 Hastie et al. ESL

Back to Formalization

Data Assumptions:

- Training set $\mathcal{D}$ with $\mathbf{x} \in \mathbb{R}^D$, $y \in \{-1, 1\}$
- Data is linearly separable



Back to Formalization

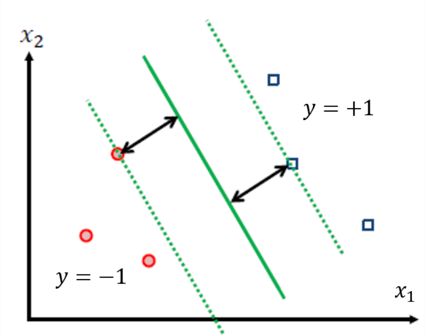
Data Assumptions:

- Training set $\mathcal{D}$ with $\mathbf{x} \in \mathbb{R}^D$, $y \in \{-1, 1\}$
- Data is linearly separable

Model Assumptions:

- The y_is encode the side of the perfect decision boundary each $\mathbf{x}_i$ is on
- Our model:

$$\hat{y} = h(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + w_0$$



Back to Formalization

Data Assumptions:

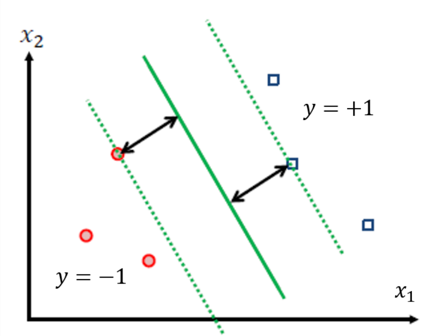
- Training set $\mathcal{D}$ with $\mathbf{x} \in \mathbb{R}^D$, $y \in \{-1, 1\}$
- Data is linearly separable

Model Assumptions:

- The y_is encode the side of the perfect decision boundary each $\mathbf{x}_i$ is on
- Our model:

$$\hat{y} = h(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0$$

Question: Have you seen this before?



Back to Formalization

Data Assumptions:

- Training set $\mathcal{D}$ with $\mathbf{x} \in \mathbb{R}^D$, $y \in \{-1, 1\}$
- Data is linearly separable

Model Assumptions:

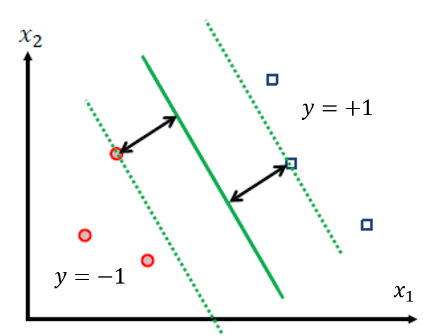
- The y_is encode the side of the perfect decision boundary each $\mathbf{x}_i$ is on
- Our model:

$$\hat{y} = h(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + w_0$$

Question: Have you seen this before?

- Correctly classified points satisfy:

$$y_i(\mathbf{w}^T \mathbf{x}_i + w_0) > 0$$



Back to Formalization

Data Assumptions:

- Training set $\mathcal{D}$ with $\mathbf{x} \in \mathbb{R}^D$, $y \in \{-1, 1\}$
- Data is linearly separable

Model Assumptions:

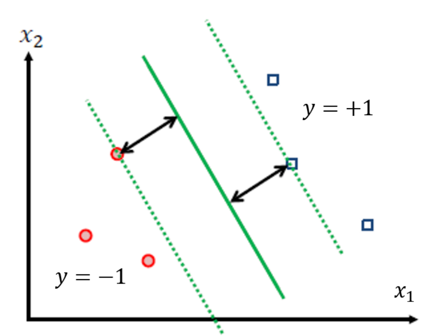
- The y_is encode the side of the perfect decision boundary each $\mathbf{x}_i$ is on
- Our model:

$$\hat{y} = h(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0$$

Question: Have you seen this before?

- Correctly classified points satisfy:

$$y_i(\mathbf{w}^T\mathbf{x}_i + w_0) > 0$$



Distance for the i-th training sample to the decision boundary:

$$d(\mathbf{x}_i, \mathcal{H}) = \frac{|\mathbf{w}^T\mathbf{x}_i + \mathbf{w}_0|}{\|\mathbf{w}\|}$$

The Learning Process: Maximum Margin

- **Margin width γ:** Distance from the decision boundary to the closest point

$$\gamma(\mathbf{w}, w_0) = \min_{\mathbf{x}_i \in \mathcal{D}} d(\mathbf{x}_i, \mathcal{H}) = \min_{\mathbf{x}_i \in \mathcal{D}} \frac{|\mathbf{w}^T \mathbf{x}_i + \mathbf{w}_0|}{\|\mathbf{w}\|}$$

- If the hyperplane is such that γ is maximized, it must be equidistant to the two classes

The Learning Process: Maximum Margin

- **Margin width γ:** Distance from the decision boundary to the closest point

$$\gamma(\mathbf{w}, w_0) = \min_{\mathbf{x}_i \in \mathcal{D}} d(\mathbf{x}_i, \mathcal{H}) = \min_{\mathbf{x}_i \in \mathcal{D}} \frac{|\mathbf{w}^T \mathbf{x}_i + \mathbf{w}_0|}{\|\mathbf{w}\|}$$

- If the hyperplane is such that γ is maximized, it must be equidistant to the two classes
- We want to find the margin as large as possible (maximization)

The Learning Process: Maximum Margin

- **Margin width γ:** Distance from the decision boundary to the closest point

$$\gamma(\mathbf{w}, w_0) = \min_{\mathbf{x}_i \in \mathcal{D}} d(\mathbf{x}_i, \mathcal{H}) = \min_{\mathbf{x}_i \in \mathcal{D}} \frac{|\mathbf{w}^T \mathbf{x}_i + \mathbf{w}_0|}{\|\mathbf{w}\|}$$

- If the hyperplane is such that γ is maximized, it must be equidistant to the two classes
- We want to find the margin as large as possible (maximization)
- SVM seeks to maximize, as a function of $\{\mathbf{w}, w_0\}$, the quantity:

$$\arg\max_{\mathbf{w}, w_0} \gamma(\mathbf{w}, w_0)$$

- **Question:** Do you see a problem?

The Learning Process: Constrained Optimization

- To guarantee that the data gets properly classified, we need to add a constraint to the optimization problem:

$$\begin{aligned} \underset{\mathbf{w}, w_0}{\arg\max} \quad & \gamma(\mathbf{w}, w_0) \\ \text{subject to} \quad & y_i(\mathbf{w}^T \mathbf{x}_i + w_0) > 0 \end{aligned}$$

The Learning Process: Constrained Optimization

- To guarantee that the data gets properly classified, we need to add a constraint to the optimization problem:

$$\begin{aligned} \underset{\mathbf{w}, w_0}{\arg\max} \quad & \gamma(\mathbf{w}, w_0) \\ \text{subject to} \quad & y_i(\mathbf{w}^T \mathbf{x}_i + w_0) > 0 \end{aligned}$$

- Let's replace γ:

$$\begin{aligned} \underset{\mathbf{w}, w_0}{\arg\max} \quad & \min_{\mathbf{x}_i \in \mathcal{D}} \frac{|\mathbf{w}^T \mathbf{x}_i + \mathbf{w}_0|}{\|\mathbf{w}\|} \\ \text{subject to} \quad & y_i(\mathbf{w}^T \mathbf{x}_i + w_0) > 0 \end{aligned}$$

The Learning Process: Constrained Optimization

- To guarantee that the data gets properly classified, we need to add a constraint to the optimization problem:

$$\begin{aligned} \underset{\mathbf{w}, w_0}{\arg\max} \quad & \gamma(\mathbf{w}, w_0) \\ \text{subject to} \quad & y_i(\mathbf{w}^T \mathbf{x}_i + w_0) > 0 \end{aligned}$$

- Let's replace γ:

$$\begin{aligned} \underset{\mathbf{w}, w_0}{\arg\max} \quad & \min_{\mathbf{x}_i \in \mathcal{D}} \frac{|\mathbf{w}^T \mathbf{x}_i + \mathbf{w}_0|}{\|\mathbf{w}\|} \\ \text{subject to} \quad & y_i(\mathbf{w}^T \mathbf{x}_i + w_0) > 0 \end{aligned}$$

- and reorganize a bit:

$$\begin{aligned} \underset{\mathbf{w}, w_0}{\arg\max} \quad & \frac{1}{\|\mathbf{w}\|} \min_{\mathbf{x}_i \in \mathcal{D}} |\mathbf{w}^T \mathbf{x}_i + \mathbf{w}_0| \\ \text{subject to} \quad & y_i(\mathbf{w}^T \mathbf{x}_i + w_0) > 0 \end{aligned}$$

Non-Unique Representation

- The hyperplane is scale invariant, i.e. $\mathcal{H} = \{\mathbf{x} : \mathbf{w}^T\mathbf{x} + w_0 = 0\}$ is equivalent to $\mathcal{H} = \{\mathbf{x} : \beta\mathbf{w}^T\mathbf{x} + \beta w_0 = 0\} \; \forall \beta \neq 0$

Non-Unique Representation

- The hyperplane is scale invariant, i.e. $\mathcal{H} = \{\mathbf{x} : \mathbf{w}^T\mathbf{x} + w_0 = 0\}$ is equivalent to $\mathcal{H} = \{\mathbf{x} : \beta\mathbf{w}^T\mathbf{x} + \beta w_0 = 0\}\ \forall\beta \neq 0$
- In words: The same boundary (the classifier) can be expressed in infinite ways
- We can use this property to simplify the optimization problem

Non-Unique Representation

- The hyperplane is scale invariant, i.e. $\mathcal{H} = \{\mathbf{x} : \mathbf{w}^T\mathbf{x} + w_0 = 0\}$ is equivalent to $\mathcal{H} = \{\mathbf{x} : \beta\mathbf{w}^T\mathbf{x} + \beta w_0 = 0\} \; \forall \beta \neq 0$
- In words: The same boundary (the classifier) can be expressed in infinite ways

- We can use this property to simplify the optimization problem
- **Idea:** The scale is not important, so we can fix it to an arbitrary value

$$\begin{aligned} \underset{\mathbf{w}, w_0}{\arg\max} \quad & \frac{1}{\|\mathbf{w}\|} \min_{\mathbf{x}_i \in \mathcal{D}} |\mathbf{w}^T\mathbf{x}_i + \mathbf{w}_0| \\ \text{subject to} \quad & \forall i \; y_i(\mathbf{w}^T\mathbf{x}_i + w_0) > 0 \end{aligned}$$

Non-Unique Representation

- The hyperplane is scale invariant, i.e. $\mathcal{H} = \{\mathbf{x} : \mathbf{w}^T\mathbf{x} + w_0 = 0\}$ is equivalent to $\mathcal{H} = \{\mathbf{x} : \beta\mathbf{w}^T\mathbf{x} + \beta w_0 = 0\}\ \forall\beta \neq 0$
- In words: The same boundary (the classifier) can be expressed in infinite ways

- We can use this property to simplify the optimization problem
- **Idea:** The scale is not important, so we can fix it to an arbitrary value

$$\begin{aligned} \arg\max_{\mathbf{w}, w_0} \quad & \frac{1}{\|\mathbf{w}\|} \min_{\mathbf{x}_i \in \mathcal{D}} |\mathbf{w}^T\mathbf{x}_i + \mathbf{w}_0| \\ \text{subject to} \quad & \forall i\ y_i(\mathbf{w}^T\mathbf{x}_i + w_0) > 0 \\ & \min_{\mathbf{x}_i \in \mathcal{D}} |\mathbf{w}^T\mathbf{x}_i + \mathbf{w}_0| = 1 \end{aligned}$$

The Learning Process: Back to the Objective Function

$$
\begin{aligned}
\underset{\mathbf{w}, w_0}{\arg\max} \quad & \frac{1}{\|\mathbf{w}\|} \cancel{\min_{\mathbf{x}_i \in \mathcal{D}} |\mathbf{w}^T \mathbf{x}_i + \mathbf{w}_0|} \\
\text{subject to} \quad & \forall i \; y_i(\mathbf{w}^T \mathbf{x}_i + w_0) > 0 \\
& \min_{\mathbf{x}_i \in \mathcal{D}} |\mathbf{w}^T \mathbf{x}_i + \mathbf{w}_0| = 1
\end{aligned}
$$

The Learning Process: Back to the Objective Function

$$\begin{aligned}
\arg\max_{\mathbf{w}, w_0} \quad & \frac{1}{\|\mathbf{w}\|} \cancel{\min_{\mathbf{x}_i \in \mathcal{D}} |\mathbf{w}^T \mathbf{x}_i + \mathbf{w}_0|} \\
\text{subject to} \quad & \forall i \; y_i(\mathbf{w}^T \mathbf{x}_i + w_0) > 0 \\
& \min_{\mathbf{x}_i \in \mathcal{D}} |\mathbf{w}^T \mathbf{x}_i + \mathbf{w}_0| = 1
\end{aligned}$$

As we prefer to do minimization rather than maximization we can use the fact that maximizing $\|\mathbf{w}\|^{-1}$ is equivalent to minimizing $\|\mathbf{w}\|^2$

The Learning Process: Back to the Objective Function

$$
\begin{aligned}
\arg\max_{\mathbf{w}, w_0} \quad & \frac{1}{\|\mathbf{w}\|} \cancel{\min_{\mathbf{x}_i \in \mathcal{D}} |\mathbf{w}^T \mathbf{x}_i + \mathbf{w}_0|} \\
\text{subject to} \quad & \forall i \; y_i(\mathbf{w}^T \mathbf{x}_i + w_0) > 0 \\
& \min_{\mathbf{x}_i \in \mathcal{D}} |\mathbf{w}^T \mathbf{x}_i + \mathbf{w}_0| = 1
\end{aligned}
$$

As we prefer to do minimization rather than maximization we can use the fact that maximizing $\|\mathbf{w}\|^{-1}$ is equivalent to minimizing $\|\mathbf{w}\|^2$

$$
\begin{aligned}
\arg\min_{\mathbf{w}, w_0} \quad & \frac{1}{2} \|\mathbf{w}\|^2 \\
\text{subject to} \quad & \forall i \; y_i(\mathbf{w}^T \mathbf{x}_i + w_0) > 0 \\
& \min_{\mathbf{x}_i \in \mathcal{D}} |\mathbf{w}^T \mathbf{x}_i + \mathbf{w}_0| = 1
\end{aligned}
$$

The Learning Process: Optimization Problem

- We can further simplify the optimization problem by simplifying the constraints.

The Learning Process: Optimization Problem

- We can further simplify the optimization problem by simplifying the constraints.
- The two original constraints:

$$
\begin{aligned}
\underset{\mathbf{w}, w_0}{\arg\min} \quad & \frac{1}{2}\|\mathbf{w}\|^2 \\
\text{subject to} \quad & \forall i \; y_i(\mathbf{w}^T\mathbf{x}_i + w_0) > 0 \\
& \min_{\mathbf{x}_i \in \mathcal{D}} |\mathbf{w}^T\mathbf{x}_i + \mathbf{w}_0| = 1
\end{aligned}
$$

The Learning Process: Optimization Problem

- We can further simplify the optimization problem by simplifying the constraints.
- The two original constraints:

$$
\begin{aligned}
\underset{\mathbf{w}, w_0}{\arg\min} \quad & \frac{1}{2}\|\mathbf{w}\|^2 \\
\text{subject to} \quad & \forall i \;\; y_i(\mathbf{w}^T\mathbf{x}_i + w_0) > 0 \\
& \min_{\mathbf{x}_i \in \mathcal{D}} |\mathbf{w}^T\mathbf{x}_i + \mathbf{w}_0| = 1
\end{aligned}
$$

- are equivalent to:

$$
\begin{aligned}
\underset{\mathbf{w}, w_0}{\arg\min} \quad & \frac{1}{2}\|\mathbf{w}\|^2 \\
\text{subject to} \quad & \forall i \;\; y_i(\mathbf{w}^T\mathbf{x}_i + w_0) \geq 1
\end{aligned}
$$

- Exercise: Can you proof that these are equivalent?

The Learning Process: Optimization Problem

- We can further simplify the optimization problem by simplifying the constraints.
- The two original constraints:

$$\begin{aligned} \arg\min_{\mathbf{w}, w_0} \quad & \frac{1}{2}\|\mathbf{w}\|^2 \\ \text{subject to} \quad & \forall i \; y_i(\mathbf{w}^T\mathbf{x}_i + w_0) > 0 \\ & \min_{\mathbf{x}_i \in \mathcal{D}} |\mathbf{w}^T\mathbf{x}_i + \mathbf{w}_0| = 1 \end{aligned}$$

- are equivalent to:

$$\begin{aligned} \arg\min_{\mathbf{w}, w_0} \quad & \frac{1}{2}\|\mathbf{w}\|^2 \\ \text{subject to} \quad & \forall i \; y_i(\mathbf{w}^T\mathbf{x}_i + w_0) \geq 1 \end{aligned}$$

- Exercise: Can you proof that these are equivalent?

In words: Find the simplest hyperplane (smaller $|\mathbf{w}\|^2$) such that all inputs lie at least 1 unit away from the hyperplane on the correct side.

Quadratic Optimization Function

$$\begin{aligned} \underset{\mathbf{w}, w_0}{\arg\min} \quad & \frac{1}{2}\|\mathbf{w}\|^2 \\ \text{subject to} \quad & \forall i \; y_i(\mathbf{w}^T \mathbf{x}_i + w_0) \geq 1 \end{aligned}$$

This formulation is a quadratic optimization problem: The objective is quadratic and the constraints are linear.

An advantage of these type of problems is that they can be efficiently solved with quadratic program solvers. This made SVMs very popular for many years.

Another interesting property is that it has a unique solution whenever a hyperplane exists.

Support Vectors

Support Vectors

- We defined a support vector as a point defining the location of the decision boundary
- The support vectors would be those points in the training set strictly satisfying

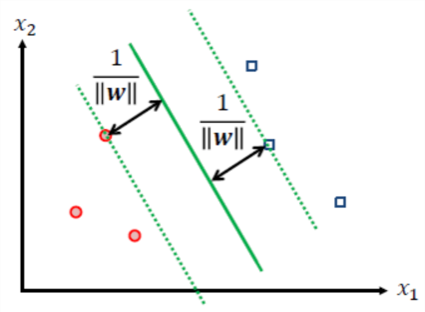
$$y_i(\mathbf{w}^T\mathbf{x}_i + w_0) = 1$$

Support Vectors

- We defined a support vector as a point defining the location of the decision boundary
- The support vectors would be those points in the training set strictly satisfying

$$y_i(\mathbf{w}^T\mathbf{x}_i + w_0) = 1$$

- They determine the shape of the hyperplane: If one is removed and the SVM is retrained, the resulting hyperplane will be different.
- The opposite occurs with non-support vectors
- Support vectors will become more important during the Kernels lecture



Hard SVM: No points allowed within the margin

Recap

Recap

- We have introduced support vector machines (SVMs) as maximum margin classifiers
- We derived the hard margin SVM objective
- We reviewed the concept of constrained optimization
- We introduced the concept of support vector

Key Concepts

- Hyperplane
- Maximum Margin Classifiers
- Support Vectors
- Constrained Optimization

References

Further Reading and Useful Material

Source	Notes
Support Vector Networks - Cortes and Vapnik	original publication (link)
Pattern Recognition and Machine Learning	Ch 7
The Elements of Statistical Learning	Sec. 4.5, Ch 12
Distance to a plane	Applet (link)