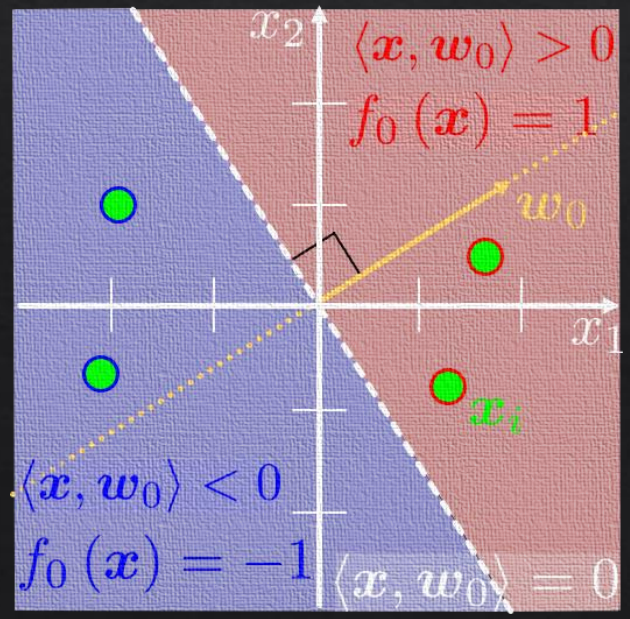
Supervised Learning – Discriminative Classification

Machine Learning Methods – Lecture 3



fixelalgorithms.gitlab.io

June 2021

Or Yair

Introduction and Notations – Binary Classification

- Consider a training set:

  $$\mathcal{D}_{\text{train}} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}$$

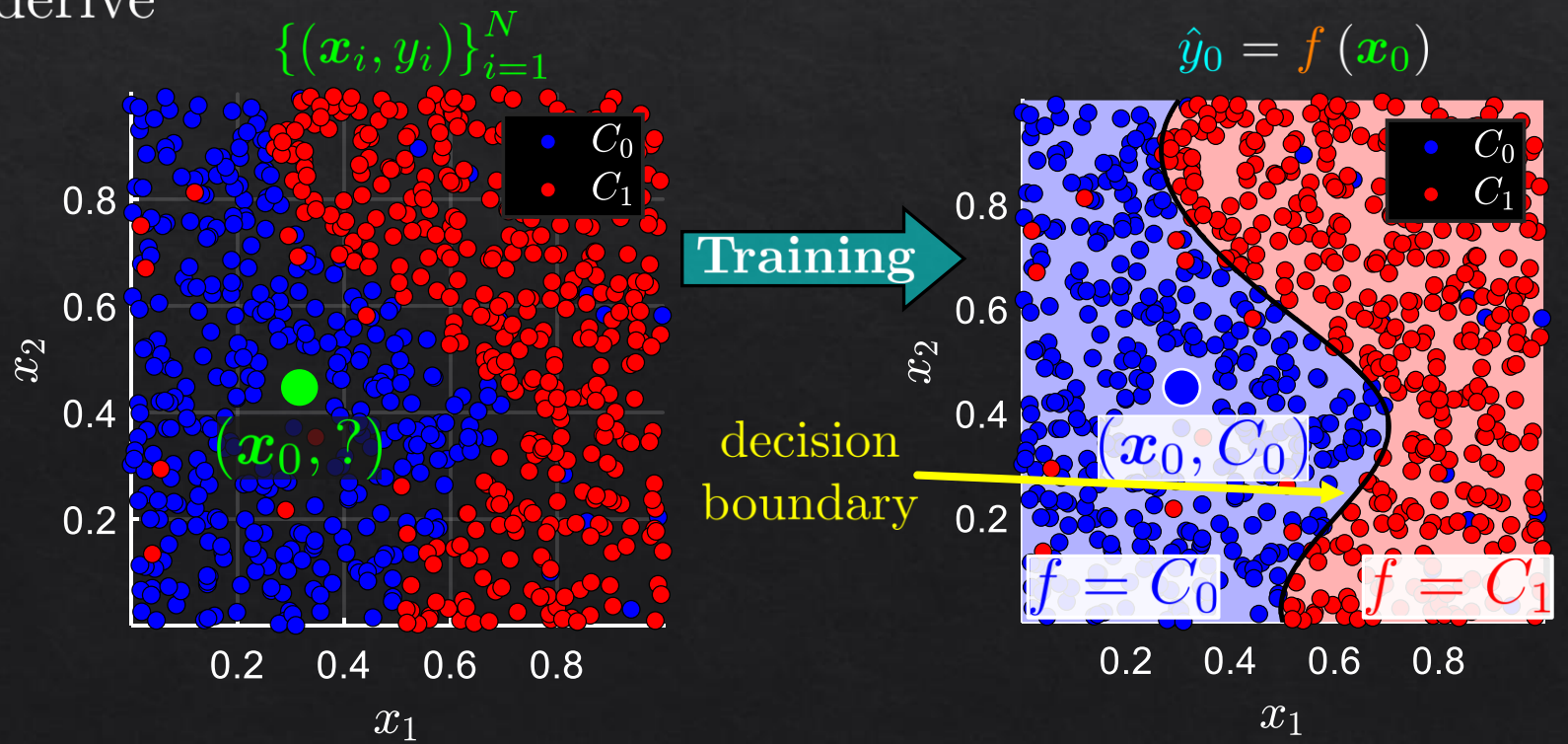
  where:
  - $\boldsymbol{x}_i \in \mathcal{X} \subseteq \mathbb{R}^d$ is the feature vector (a.k.a. sample or observation).
  - $y_i \in \mathcal{Y} = \{C_0, C_1\}$ is the class of $\boldsymbol{x}_i$ (a.k.a. category, label, response).
- Let $(\boldsymbol{x}_0, y_0)$ be a new pair, where $\boldsymbol{x}_0$ is known and y_0 is unknown.
- Using $\mathcal{D}_{\text{train}}$ our goal is to derive a classifier (a mapping):

  $$f : \mathcal{X} \to \mathcal{Y}$$

  such that: $y_0 = f(\boldsymbol{x}_0)$
- A set of new* pairs is known as a test set:

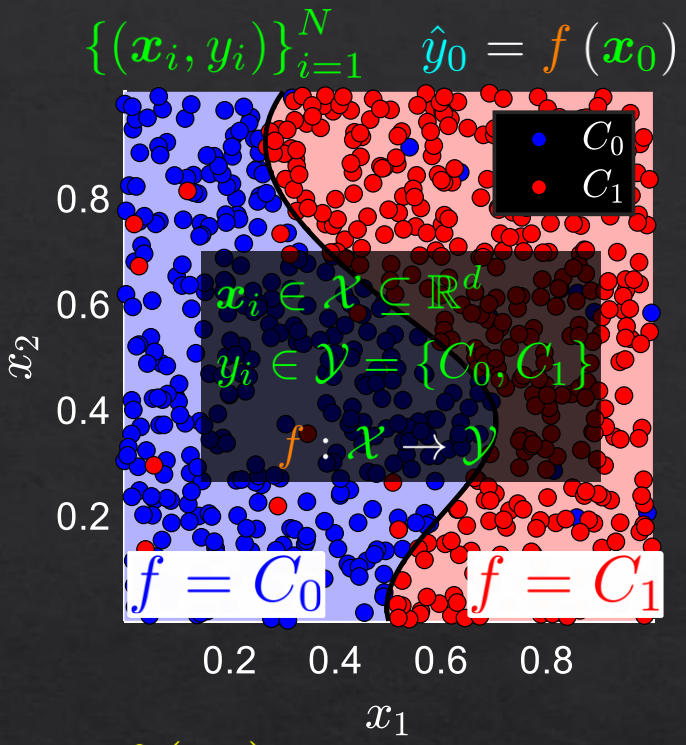
  $$\mathcal{D}_{\text{test}} = \{(\tilde{\boldsymbol{x}}_i, \tilde{y}_i)\}_{i=1}^{M}$$

(*) new – unseen during training.



Hamming Loss

- A loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$ satisfies:
  1. $\hat{y}_i = y_i \implies \ell(\hat{y}_i, y_i) = 0$
  2. $\ell(\hat{y}_i, y_i) \geq 0$, for all $\hat{y}_i, y_i \in \mathcal{Y}$
- Hamming $0-1$ loss: $\ell(\hat{y}_i, y_i) = \mathbb{I}\{\hat{y}_i \neq y_i\} = \begin{cases} 0 & \hat{y}_i = y_i \\ 1 & \hat{y}_i \neq y_i \end{cases}$



- The train error of a classifier f w.r.t. $\mathcal{D}_{\text{train}} = \{(x_i, y_i)\}_{i=1}^N$: $\hat{y}_i = f(x_i)$

$$L_{\text{train}}(f) := \frac{1}{N} \sum_{i=1}^{N} \ell(\hat{y}_i, y_i) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}\{\hat{y}_i \neq y_i\}$$

$1 - L(f)$ is the classifier accuracy

$L(f) = 0 \implies$ no errors

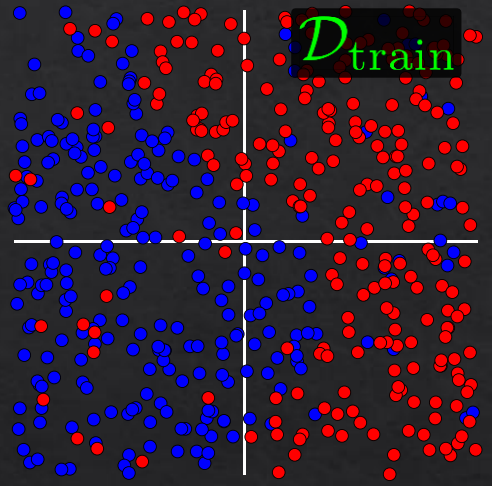
- The goal is to minimize $L_{\text{test}}(f)$ w.r.t $\mathcal{D}_{\text{test}}$.
- Since $\mathcal{D}_{\text{test}}$ is unavailable, we **usually** minimize:

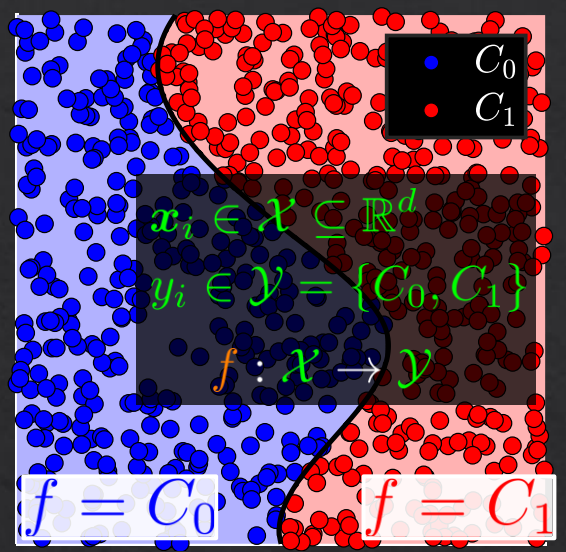
$$f^\star = \arg\min_{f \in ?} L_{\text{train}}(f) = \arg\min_{f \in ?} \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}\{f(x_i) \neq y_i\}$$

Classifier Models

- Generally, it is not feasible to search the optimal function in the space of all functions.
- A common practice is to fix a set of functions $\mathcal{H}$ (a model) and restrict the search to that set:

$$\arg\min_{f \in \mathcal{H}} \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}\{f(\boldsymbol{x}_i) = y_i\}$$

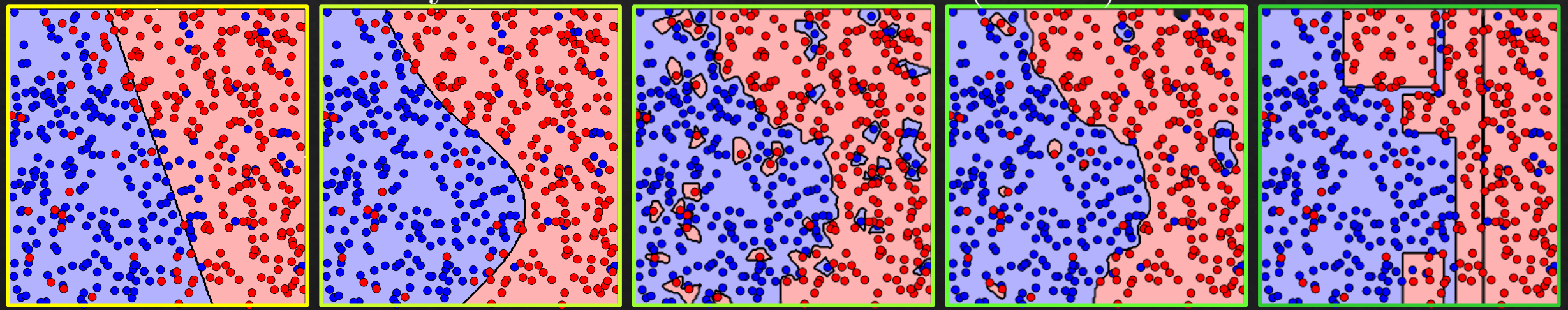




$$\arg\min_{f \in ?} \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}\{f(\boldsymbol{x}_i) \neq y_i\}$$

Some models for example:

Linear | Polynomial | K–NN | (Kernel) SVM | Decision tree



Linear Classifiers

- We model a linear classifier $f \in \mathcal{H}_{\text{linear}}$ using two parameters:

$$f(\boldsymbol{x}) = \text{sign}\left(\boldsymbol{w}^T \boldsymbol{x} - b\right), \quad \boldsymbol{w} \in \mathbb{R}^d, b \in \mathbb{R}$$

- $\boldsymbol{w} \in \mathbb{R}^d$ and $b \in \mathbb{R}$ are the model parameters.

The role of $\boldsymbol{w}$

- Consider the linear classifier f_0: with $b = 0$

$$f_0(\boldsymbol{x}) = \text{sign}\left(\boldsymbol{w}_0^T \boldsymbol{x}\right) = \text{sign}\left(\langle \boldsymbol{x}, \boldsymbol{w}_0 \rangle\right)$$

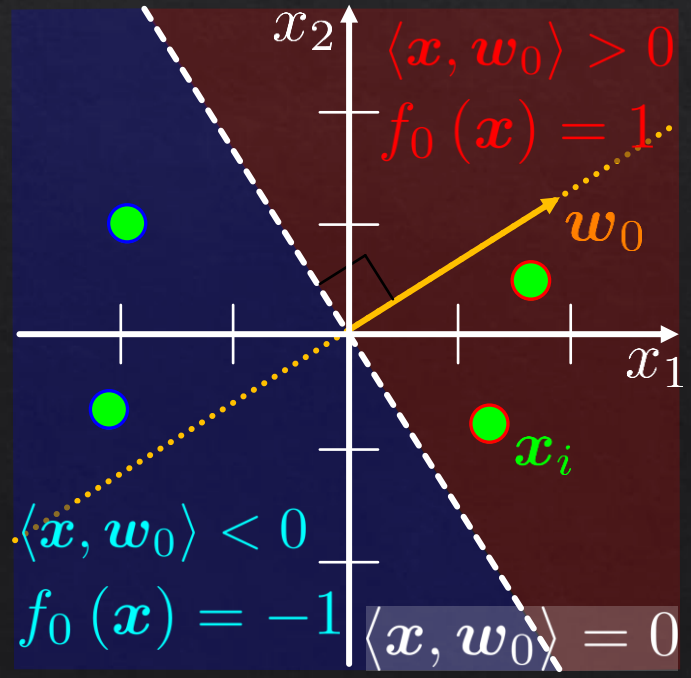
- The decision boundary is given by:

$$f_0(\boldsymbol{x}) = 0$$

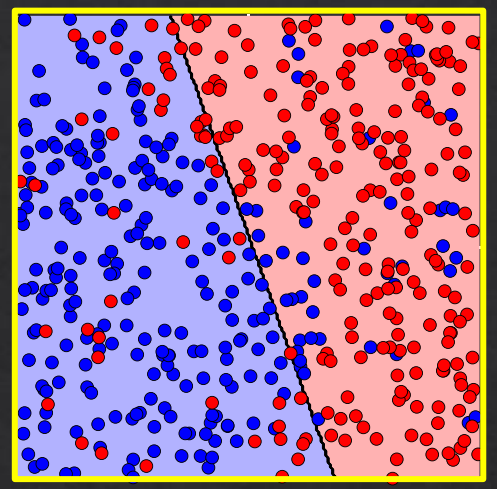
$$\text{sign}\left(\langle \boldsymbol{x}, \boldsymbol{w}_0 \rangle\right) = 0$$

$$\langle \boldsymbol{x}, \boldsymbol{w}_0 \rangle = 0$$

what is the role of b?



Linear



$$\arg\min_{f \in \mathcal{H}} \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}\{f(\boldsymbol{x}_i) = y_i\}$$

$$\text{sign}(\alpha) = \begin{cases} -1 & \alpha < 0 \\ 0 & \alpha = 0 \\ 1 & \alpha > 0 \end{cases}$$

$$\boldsymbol{w}^T \boldsymbol{x} = \langle \boldsymbol{x}, \boldsymbol{w} \rangle$$

Linear Classifiers

The role of b

- Consider the classifier f_1:

$$f_1(\boldsymbol{x}) = \text{sign}\left(\boldsymbol{w}_1^T \boldsymbol{x} - b_1\right), \quad \|\boldsymbol{w}_1\|_2 = 1$$

for simplicity

- The decision boundary is given by:

$$f_1(\boldsymbol{x}) = 0$$

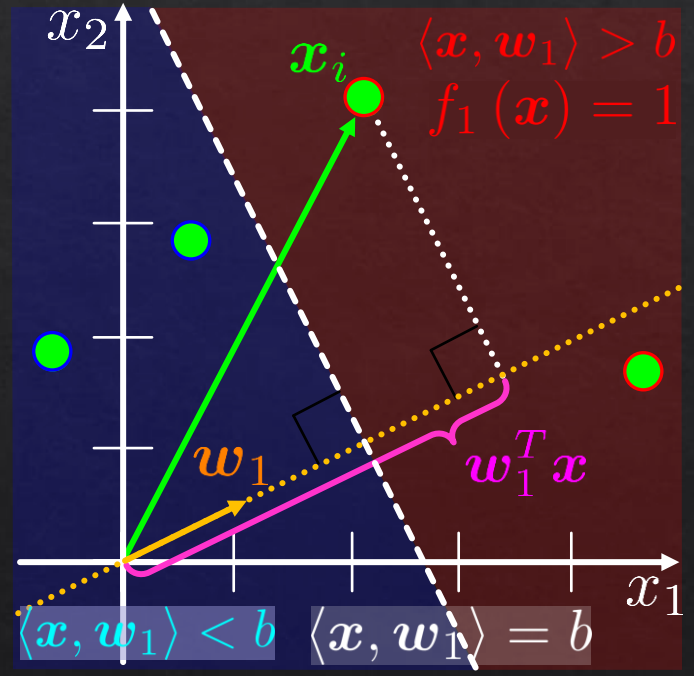
$$\text{sign}\left(\boldsymbol{w}_1^T \boldsymbol{x} - b_1\right) = 0$$

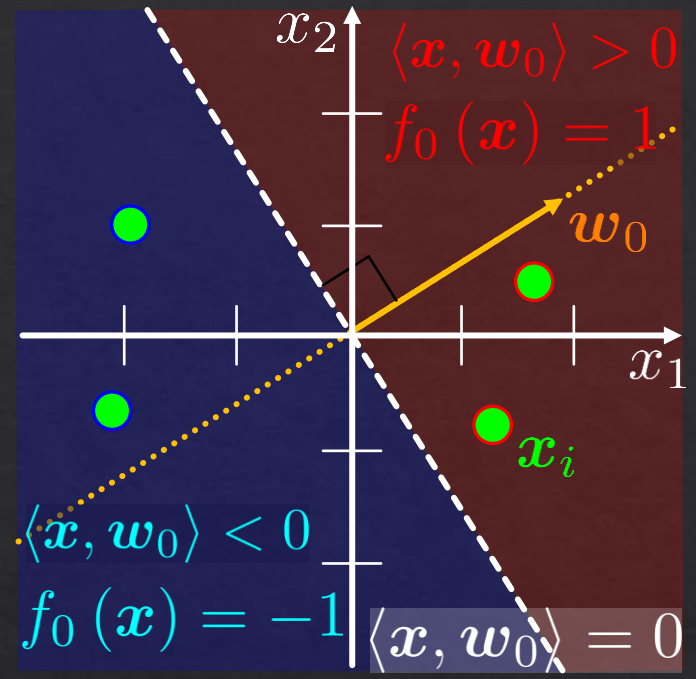
$$\boldsymbol{w}_1^T \boldsymbol{x} = b$$

- We search for $\boldsymbol{w}$ and b which minimize the train error:

$$\mathcal{D}_{\text{train}} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$$

$$f(\boldsymbol{x}) = \text{sign}\left(\boldsymbol{w}^T \boldsymbol{x} - b\right)$$





$$\arg\min_{f \in \mathcal{H}} \frac{1}{N} \sum_{i=1}^N \mathbb{I}\{f(\boldsymbol{x}_i) = y_i\}$$

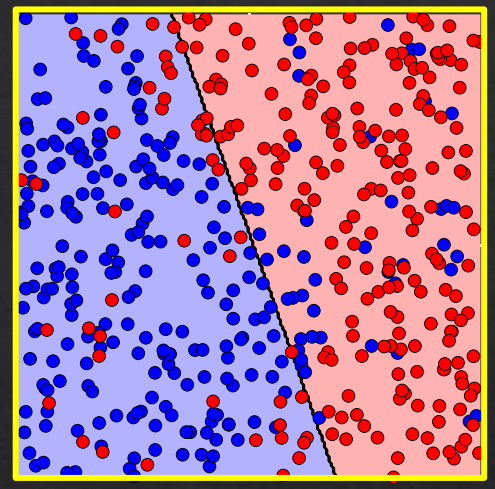
$$f(\boldsymbol{x}) = \text{sign}\left(\boldsymbol{w}^T \boldsymbol{x} - b\right)$$

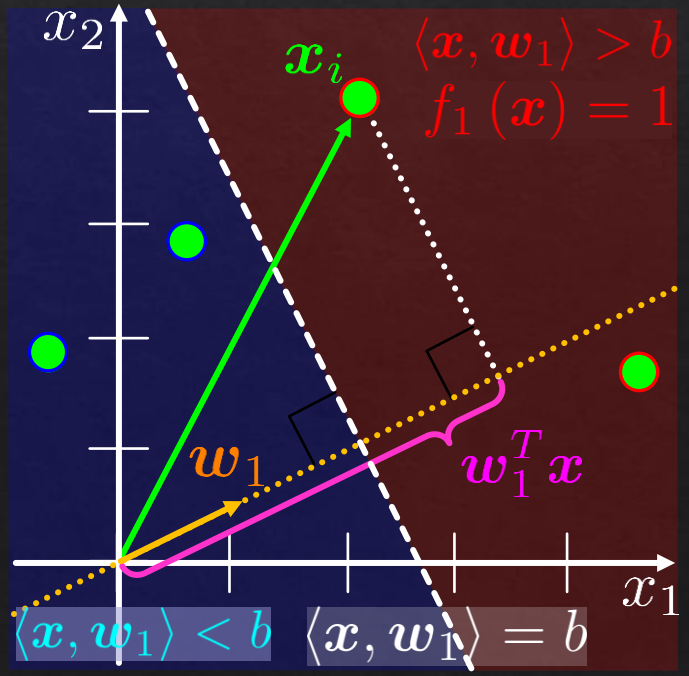
$$\boldsymbol{w}^\star, b^\star = \arg\min_{\boldsymbol{w}, b} L_{\text{train}}(f) = \arg\min_{\boldsymbol{w}, b} \frac{1}{N} \sum_{i=1}^N \mathbb{I}\{f(\boldsymbol{x}_i) \neq y_i\}$$

Linear Classifier – Example

Linear Classifier

$$f(\boldsymbol{x}) = \text{sign}\left(\boldsymbol{w}^T \boldsymbol{x} - b\right)$$





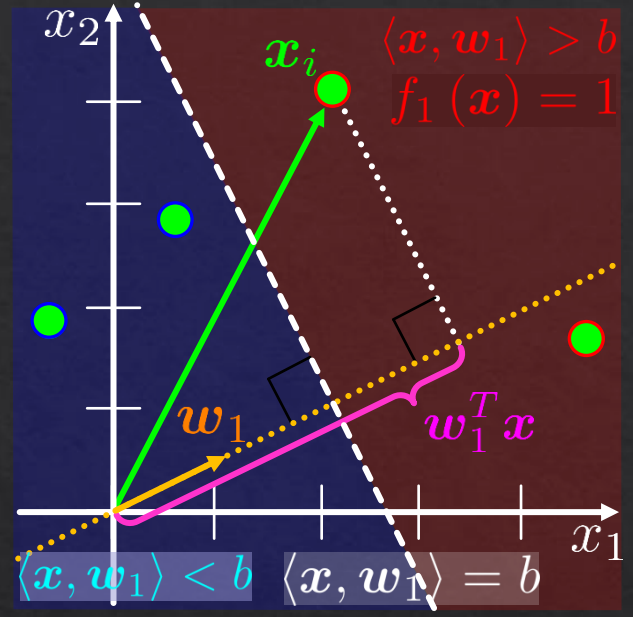
Linear Classifiers – Reparametrization

- Consider a training pair $(\boldsymbol{x}_i, y_i)$.
- The Hamming loss is given by:

$$\ell_i(f) = \mathbb{I}\{f(\boldsymbol{x}_i) \neq y_i\}$$

$$= \mathbb{I}\{\text{sign}(\boldsymbol{w}^T \boldsymbol{x}_i - b) \neq y_i\}$$

$$\implies \ell_i(f) = \mathbb{I}\{\text{sign}(\tilde{\boldsymbol{w}}^T \tilde{\boldsymbol{x}}_i) \neq y_i\}$$



$$\arg\min_{\boldsymbol{w},b} \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}\{f(\boldsymbol{x}_i) \neq y_i\}$$

$$f(\boldsymbol{x}) = \text{sign}(\boldsymbol{w}^T \boldsymbol{x} - b)$$

- We denote:

$$\tilde{\boldsymbol{x}}_i := \begin{bmatrix} -1 \\ \hline \\ \boldsymbol{x}_i \\ \\ \hline \end{bmatrix} \in \mathbb{R}^{d+1}, \quad \tilde{\boldsymbol{w}} := \begin{bmatrix} b \\ \hline \\ \boldsymbol{w} \\ \\ \hline \end{bmatrix} \in \mathbb{R}^{d+1}$$

$$\implies \tilde{\boldsymbol{w}}^T \tilde{\boldsymbol{x}}_i = \boldsymbol{w}^T \boldsymbol{x}_i - b$$

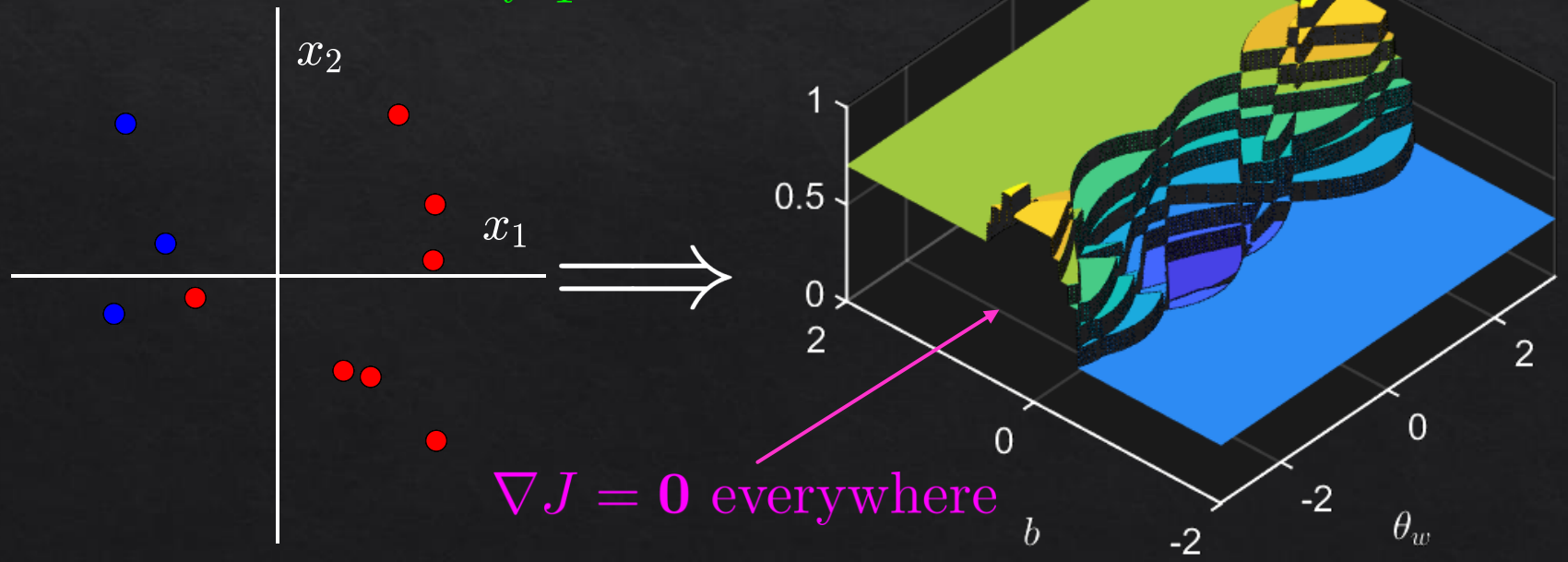
Linear Classifiers – Objective

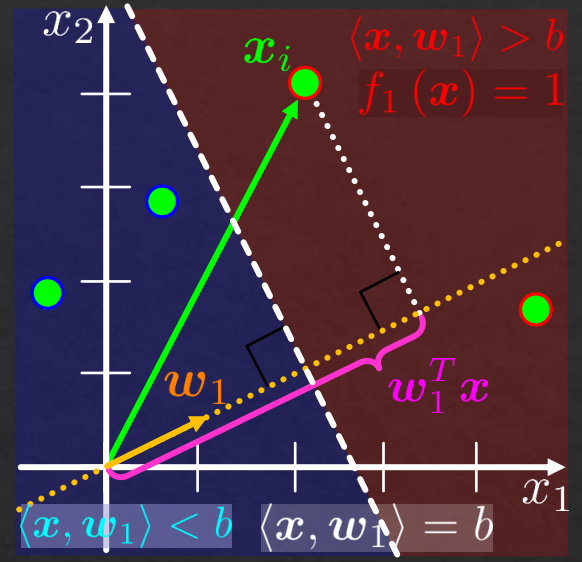
- The objective is given by:

$$\tilde{\boldsymbol{w}}^{\star} = \arg\min_{\tilde{\boldsymbol{w}}} \underbrace{\frac{1}{N}\sum_{i=1}^{N} \mathbb{I}\left\{\text{sign}\left(\tilde{\boldsymbol{w}}^{T}\tilde{\boldsymbol{x}}_i\right) \neq y_i\right\}}_{=:\, J\left(\tilde{\boldsymbol{w}}\right)}$$

- Note that the objective $J\left(\tilde{\boldsymbol{w}}\right)$ is not differentiable. For example:

$$\mathcal{D}_{\text{train}} = \left\{\boldsymbol{x}_i, y_i\right\}_{i=1}^{N}$$





$$\arg\min_{\boldsymbol{w},b} \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}\left\{f\left(\boldsymbol{x}_i\right) \neq y_i\right\}$$

$$f\left(\boldsymbol{x}\right) = \text{sign}\left(\boldsymbol{w}^{T}\boldsymbol{x} - b\right)$$

$$\tilde{\boldsymbol{x}}_i := \begin{bmatrix} -1 \\ \boldsymbol{x}_i \end{bmatrix} \in \mathbb{R}^{d+1}$$

$$\tilde{\boldsymbol{w}} := \begin{bmatrix} b \\ \boldsymbol{w} \end{bmatrix} \in \mathbb{R}^{d+1}$$

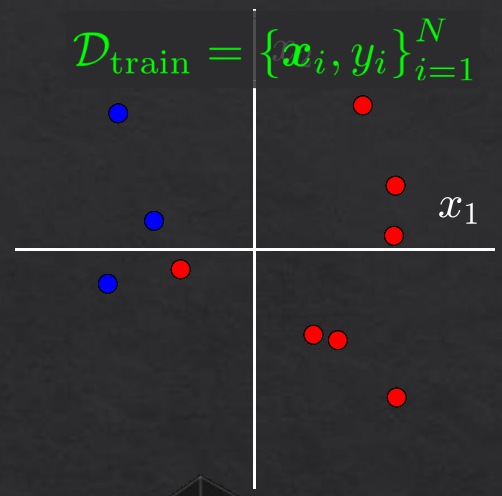
$$\implies \tilde{\boldsymbol{w}}^{T}\tilde{\boldsymbol{x}}_i = \boldsymbol{w}^{T}\boldsymbol{x}_i - b$$

Linear Classifiers – Objective Approximation

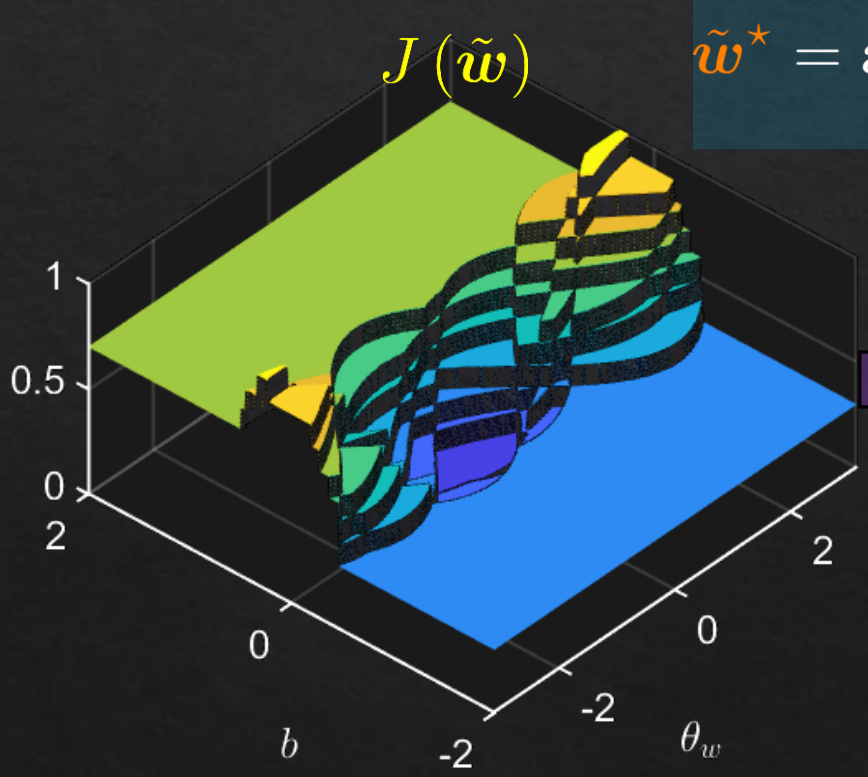
$$\tilde{\boldsymbol{w}}^{\star} = \arg\min_{\tilde{\boldsymbol{w}}} \underbrace{\frac{1}{N}\sum_{i=1}^{N} \mathbb{I}\left\{\text{sign}\left(\tilde{\boldsymbol{w}}^{T}\tilde{\boldsymbol{x}}_i\right) \neq y_i\right\}}_{=: J\left(\tilde{\boldsymbol{w}}\right)}$$

$$\tilde{\boldsymbol{w}}^{\star} = \arg\min_{\tilde{\boldsymbol{w}}} \overbrace{\frac{1}{N}\sum_{i=1}^{N} \left(\sigma\left(\tilde{\boldsymbol{w}}^{T}\tilde{\boldsymbol{x}}_i\right) - y_i\right)^2}^{=: \tilde{J}\left(\tilde{\boldsymbol{w}}\right)}$$

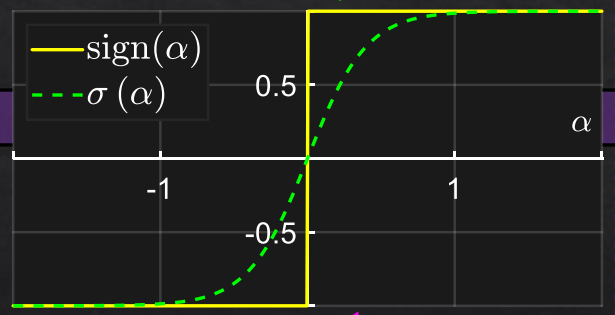
$\mathcal{D}_{\text{train}} = \{\boldsymbol{x}_i, y_i\}_{i=1}^{N}$



$J\left(\tilde{\boldsymbol{w}}\right)$



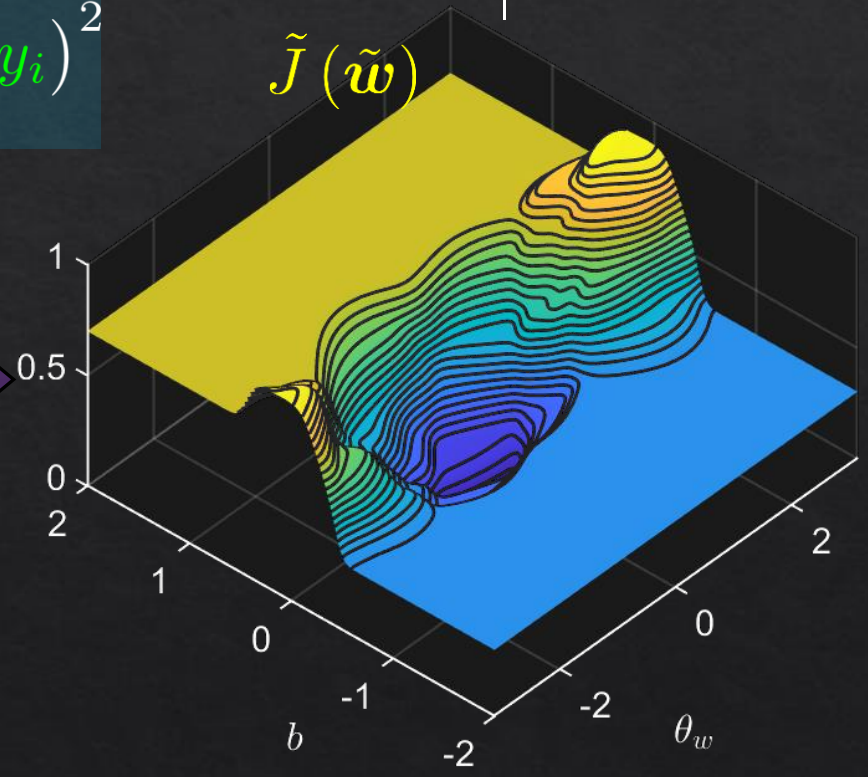
$$\sigma\left(\alpha\right) := 2\frac{e^{\alpha}}{1+e^{\alpha}} - 1$$



$$\mathbb{I}\left\{a \neq b\right\} = \frac{1}{4}\left(a-b\right)^2$$

$$a, b \in \{-1, 1\}$$

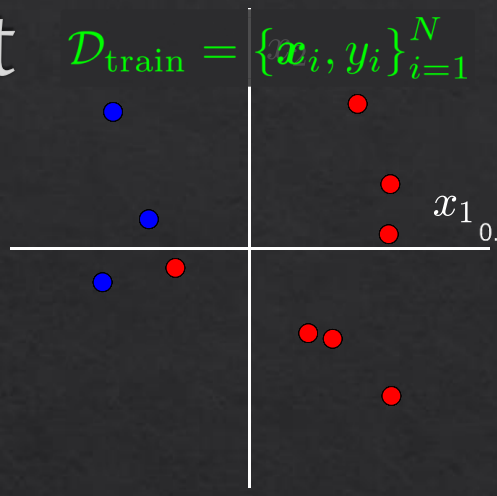
$\tilde{J}\left(\tilde{\boldsymbol{w}}\right)$

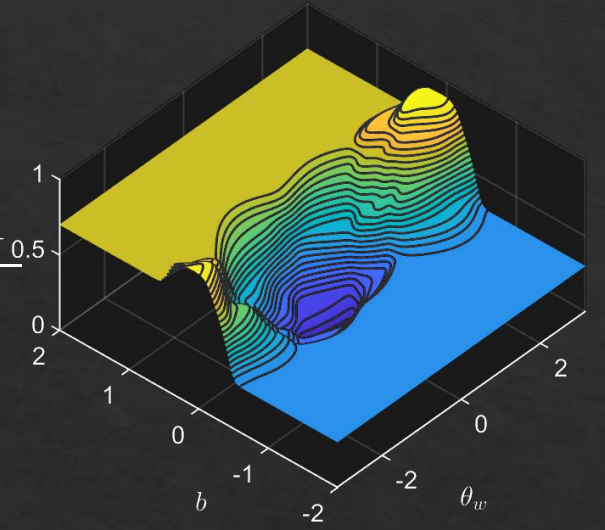


$\tilde{J}\left(\tilde{\boldsymbol{w}}\right)$ is a differentiable approximation of $J\left(\tilde{\boldsymbol{w}}\right)$

Sigmoid Approximation – Gradient

$$\mathcal{D}_{\text{train}} = \{\boldsymbol{x}_i, y_i\}_{i=1}^N$$





$$\tilde{\boldsymbol{w}}^\star = \arg\min_{\tilde{\boldsymbol{w}}} \frac{1}{N} \underbrace{\sum_{i=1}^N \left(\sigma\left(\tilde{\boldsymbol{w}}^T \tilde{\boldsymbol{x}}_i\right) - y_i\right)^2}_{=: \tilde{J}(\tilde{\boldsymbol{w}})}$$

$$\|\boldsymbol{v}\|_2^2 = \sum_i v_i^2$$

$$\tilde{J}(\tilde{\boldsymbol{w}}) = \left\| \begin{bmatrix} \sigma\left(\tilde{\boldsymbol{w}}^T \tilde{\boldsymbol{x}}_1\right) - y_1 \\ \vdots \\ \sigma\left(\tilde{\boldsymbol{w}}^T \tilde{\boldsymbol{x}}_N\right) - y_N \end{bmatrix} \right\|_2^2 = \left\| \sigma\left(\begin{bmatrix} \tilde{\boldsymbol{w}}^T \tilde{\boldsymbol{x}}_1 \\ \vdots \\ \tilde{\boldsymbol{w}}^T \tilde{\boldsymbol{x}}_N \end{bmatrix} \right) - \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \right\|_2^2 = \underbrace{\left\| \sigma\left(\begin{bmatrix} — & \tilde{\boldsymbol{x}}_1^T & — \\ & \vdots & \\ — & \tilde{\boldsymbol{x}}_N^T & — \end{bmatrix} \begin{bmatrix} | \\ \tilde{\boldsymbol{w}} \\ | \end{bmatrix} \right) - \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \right\|_2^2}_{\tilde{J}(\tilde{\boldsymbol{w}}) = \left\| \sigma\left(\tilde{\boldsymbol{X}} \tilde{\boldsymbol{w}}\right) - \boldsymbol{y} \right\|_2^2}$$

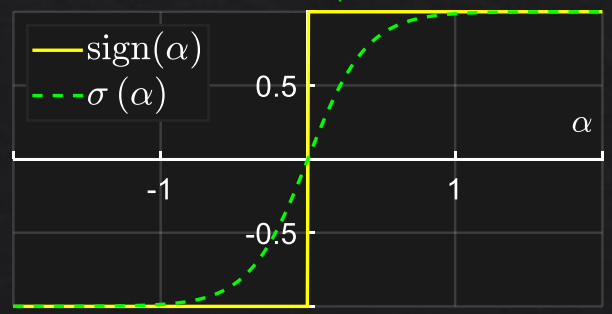
- 1D derivative $(w, x, y \in \mathbb{R})$:

$$\frac{\mathrm{d}}{\mathrm{d}w} \left(\sigma(wx) - y\right)^2 = 2x\sigma'(wx)\left(\sigma(wx) - y\right)$$

- Full gradient:

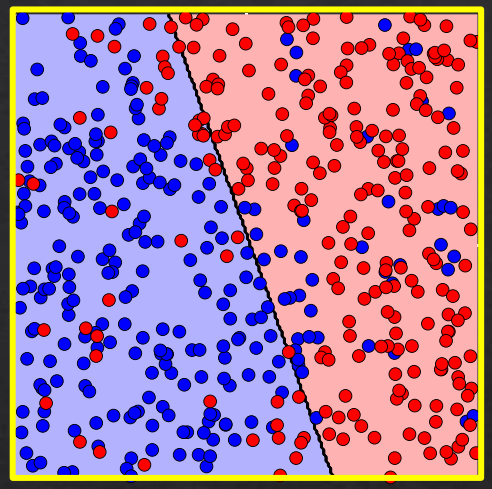
$$\nabla_{\tilde{\boldsymbol{w}}} J(\tilde{\boldsymbol{w}}) = 2\tilde{\boldsymbol{X}}^T \sigma'\left(\operatorname{diag}\left(\tilde{\boldsymbol{X}} \tilde{\boldsymbol{w}}\right)\right)\left(\sigma\left(\tilde{\boldsymbol{X}} \tilde{\boldsymbol{w}}\right) - \boldsymbol{y}\right)$$

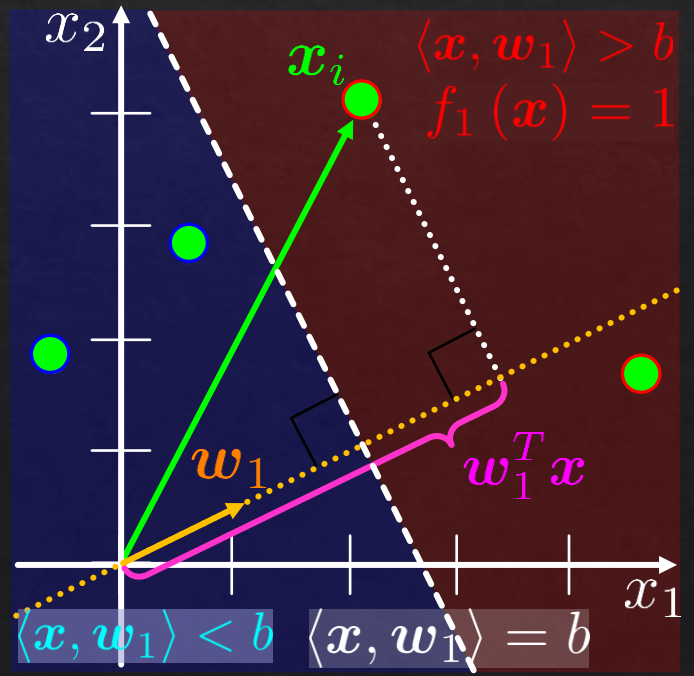
$$\sigma(\alpha) := 2\frac{e^\alpha}{1 + e^\alpha} - 1$$



Train Linear Classifier – Example

$$f(\boldsymbol{x}) = \operatorname{sign}\left(\boldsymbol{w}^T \boldsymbol{x} - b\right)$$





Train a Linear Classifier

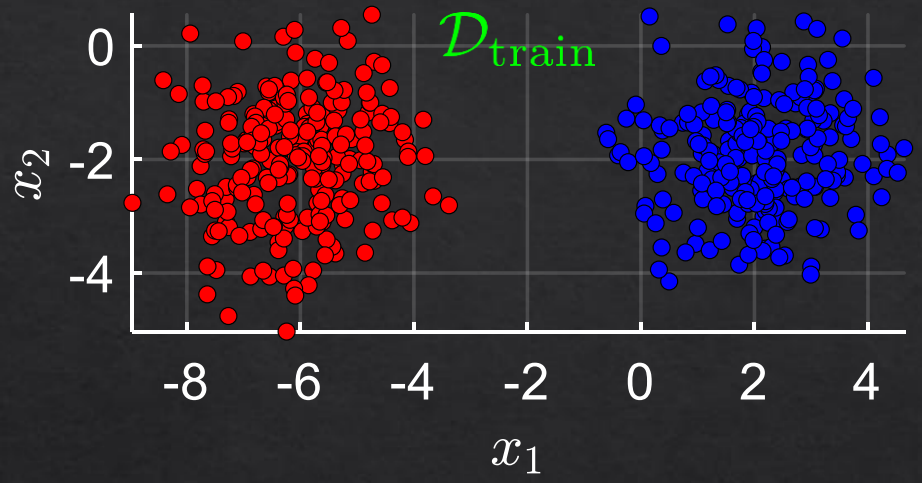
$$\nabla_{\tilde{\boldsymbol{w}}} J(\tilde{\boldsymbol{w}}) = 2\tilde{\boldsymbol{X}}^T \sigma'\left(\operatorname{diag}\left(\tilde{\boldsymbol{X}}\tilde{\boldsymbol{w}}\right)\right)\left(\sigma\left(\tilde{\boldsymbol{X}}\tilde{\boldsymbol{w}}\right) - \boldsymbol{y}\right)$$

Binary Classification Exercise

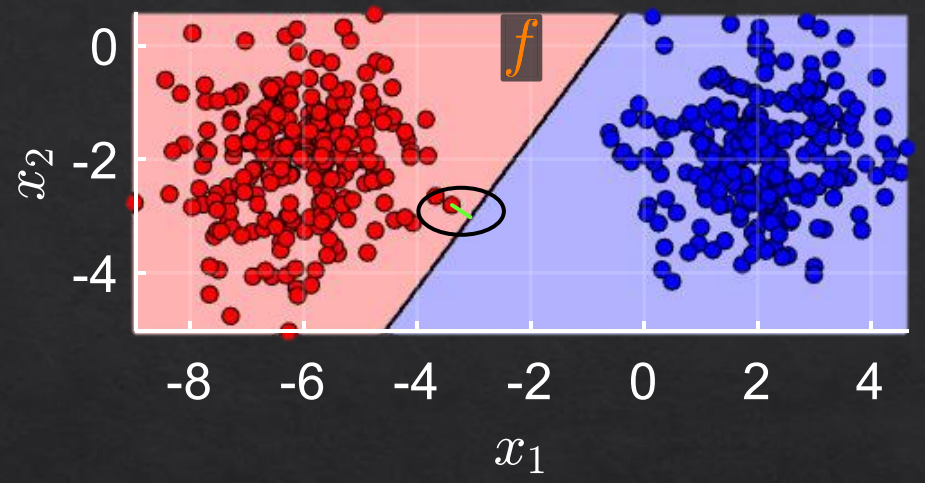
Binary Classification Exercise

Support Vector Machine – Introduction

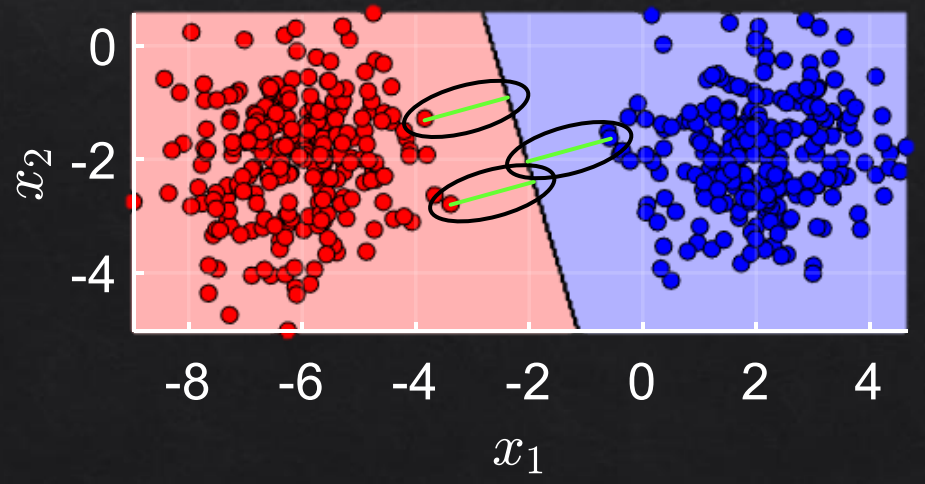
- Consider a linear seperable $\mathcal{D}_{\text{train}}$ and the following linear classifier f:



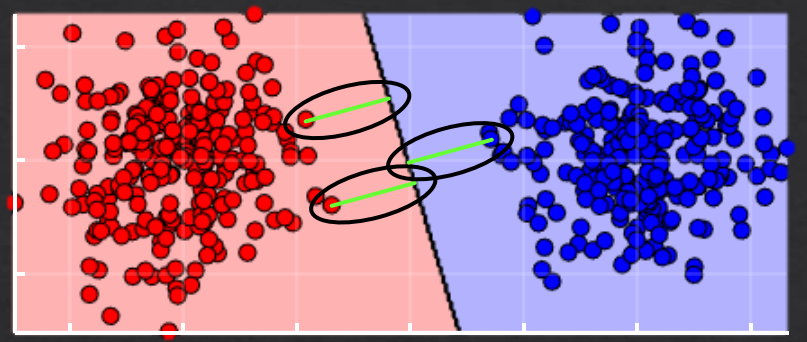
Training



- The accuracy of f is 100%.
  However, some points are close to the boundary. (small margins)
- The classifier f is sensitive to small perturbations.
- The SVM classifier seeks to maximize the margins.



Support Vector Machine – Margins



$$f(\boldsymbol{x}) = \text{sign}\left(\boldsymbol{w}^T\boldsymbol{x} - b\right)$$

- A perfect linear classifier satisfies:

$$\begin{cases} \boldsymbol{w}^T\boldsymbol{x}_i - b > 0 & y_i = 1 \\ \boldsymbol{w}^T\boldsymbol{x}_i - b < 0 & y_i = -1 \end{cases} \implies \underbrace{y_i\left(\boldsymbol{w}^T\boldsymbol{x}_i - b\right)}_{\text{the margin from the decision boundary}} > 0$$

- The SVM classifier requires:

$$y_i\left(\boldsymbol{w}^T\boldsymbol{x}_i - b\right) \geq 1, \quad \forall i$$

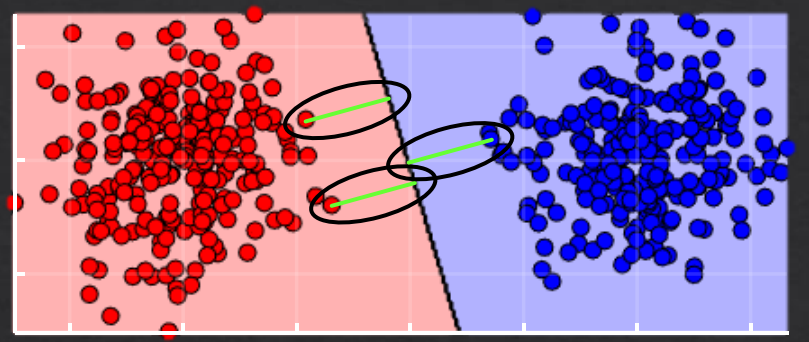
- The value 1 is arbitrary and could be replaced with any other positive value.
- This can always be satisfied by a perfect linear classifier:

$$\underbrace{y_i\left(\boldsymbol{w}^T\boldsymbol{x}_i - b\right) > 0}_{\text{a perfect classifier}} \implies y_i C\left(\boldsymbol{w}^T\boldsymbol{x}_i - b\right) \geq 1, \quad \text{some large } C$$

$$y_i\left(\underbrace{C\boldsymbol{w}}_{\text{new } \boldsymbol{w}}{}^T\boldsymbol{x}_i - \underbrace{Cb}_{\text{new } b}\right) \geq 1$$

Support Vector Machine – Gap

$$y_i \left(\boldsymbol{w}^T \boldsymbol{x}_i - b \right) \geq 1, \quad \forall i$$



$$f(\boldsymbol{x}) = \text{sign}\left(\boldsymbol{w}^T \boldsymbol{x} - b \right)$$

- Let $\boldsymbol{x}_+$ and $\boldsymbol{x}_-$ be exactly on the gutter, that is:

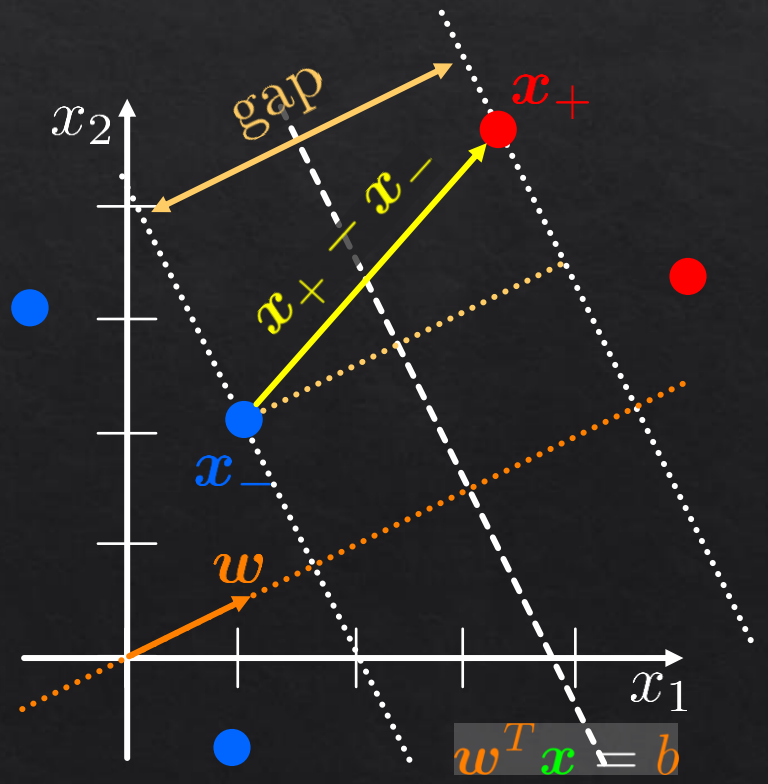
$$\begin{cases} \boldsymbol{w}^T \boldsymbol{x}_+ - b = 1 \\ \boldsymbol{w}^T \boldsymbol{x}_- - b = -1 \end{cases}$$

$$\text{gap} = \frac{\boldsymbol{w}^T}{\|\boldsymbol{w}\|} (\boldsymbol{x}_+ - \boldsymbol{x}_-)$$

$$= \frac{\boldsymbol{w}^T \boldsymbol{x}_+}{\|\boldsymbol{w}\|} - \frac{\boldsymbol{w}^T \boldsymbol{x}_-}{\|\boldsymbol{w}\|}$$

$$= \frac{1+b}{\|\boldsymbol{w}\|} - \frac{-1+b}{\|\boldsymbol{w}\|}$$

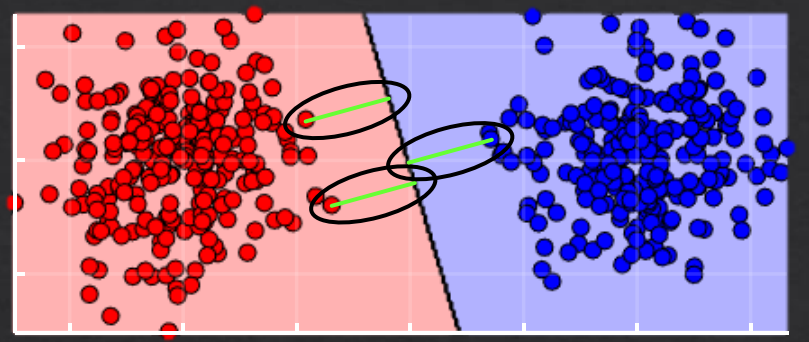
$$= \frac{2}{\|\boldsymbol{w}\|}$$



- The SVM classifier seeks a $\boldsymbol{w}$ that maximizes the gap.

Support Vector Machine – Objective

$$y_i \left(\boldsymbol{w}^T \boldsymbol{x}_i - b \right) \geq 1, \quad \forall i$$



$$f(\boldsymbol{x}) = \text{sign}\left(\boldsymbol{w}^T \boldsymbol{x} - b \right)$$

$$\text{gap} = \frac{2}{\|\boldsymbol{w}\|}$$

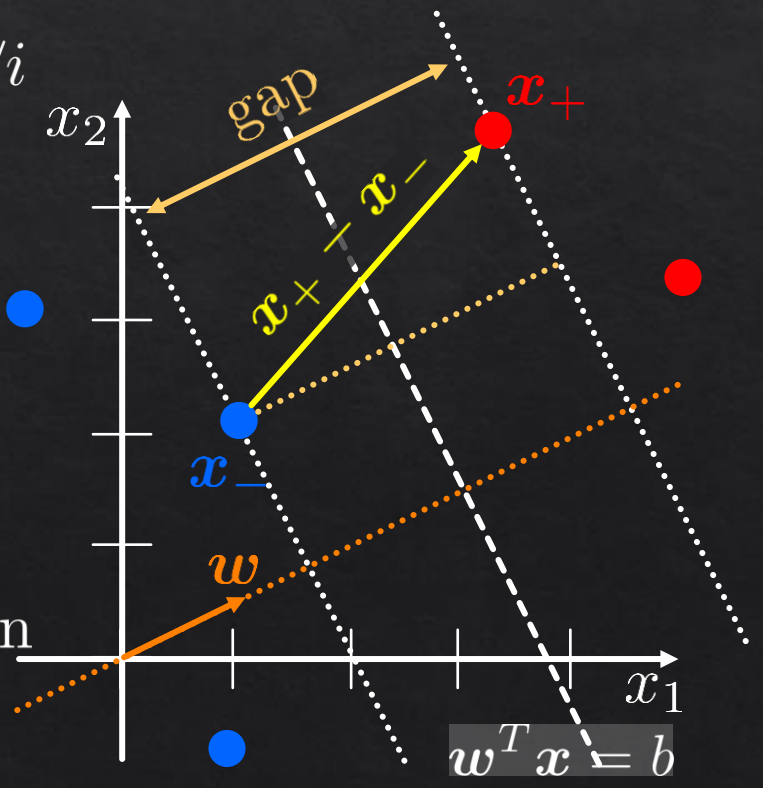
- SVM seeks to maximize the gap:

$$\begin{cases} \max\limits_{\boldsymbol{w},b} \frac{2}{\|\boldsymbol{w}\|} \\ \text{subject to} \\ y_i \left(\boldsymbol{w}^T \boldsymbol{x}_i - b \right) \geq 1 \quad \forall i \end{cases}$$

- Instead of maximize $\frac{1}{\|\boldsymbol{w}\|}$, we can minimize $\frac{1}{2} \|\boldsymbol{w}\|^2$:

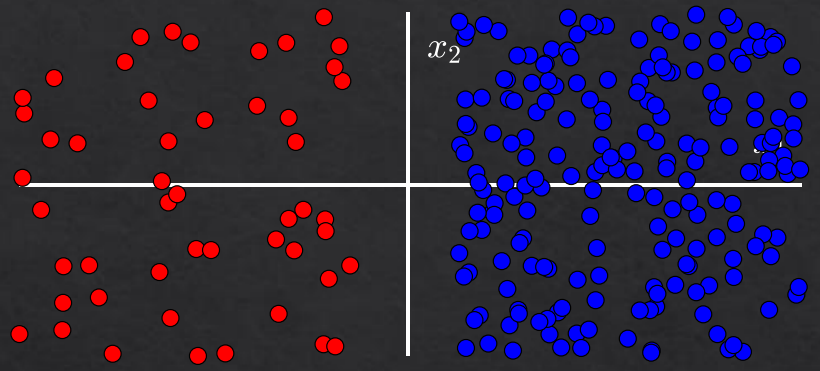
$$\begin{cases} \min\limits_{\boldsymbol{w},b} \frac{1}{2} \|\boldsymbol{w}\|^2 \\ \text{subject to} \\ y_i \left(\boldsymbol{w}^T \boldsymbol{x}_i - b \right) \geq 1 \quad \forall i \end{cases}$$

- There are efficient solvers for this constrained optimization (such as quadratic programming).

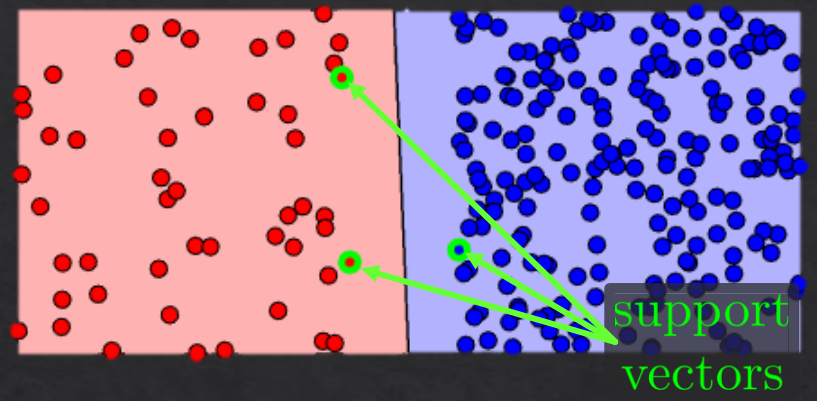


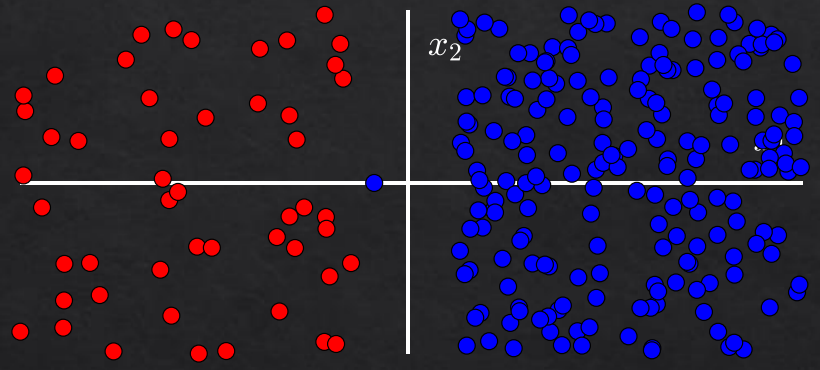
(Hard) Support Vector Machine – Examples

$$y_i \left(\boldsymbol{w}^T \boldsymbol{x}_i - b \right) \geq 1$$

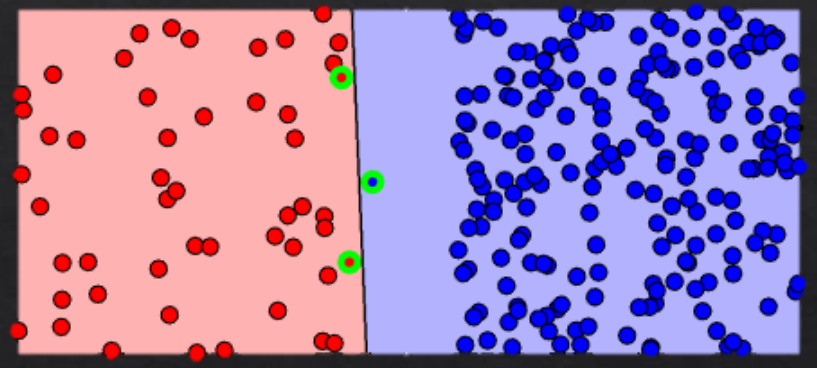


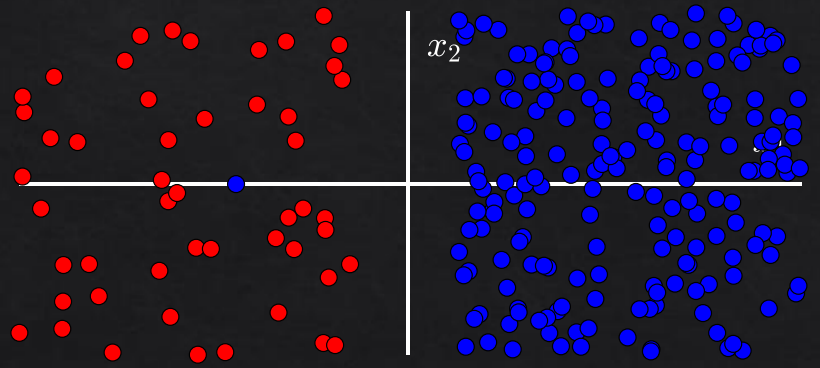
SVM





SVM



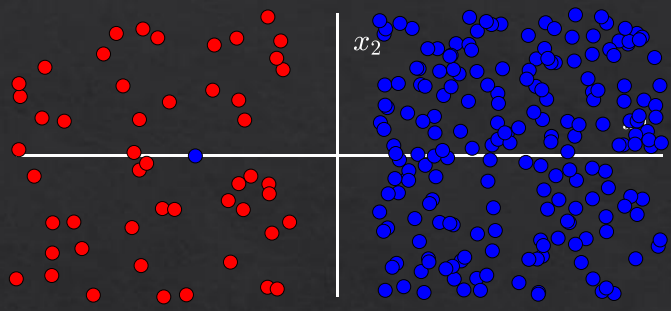


SVM

no (hard) solution

Soft Support Vector Machine

$$\begin{cases} \min\limits_{\boldsymbol{w},b} \frac{1}{2} \|\boldsymbol{w}\|^2 \\ \text{subject to} \\ y_i \left(\boldsymbol{w}^T \boldsymbol{x}_i - b\right) \geq 1 \quad \forall i \end{cases}$$



no (hard) solution

- In soft SVM,
  the (hard) constraints are relaxed using regularization:

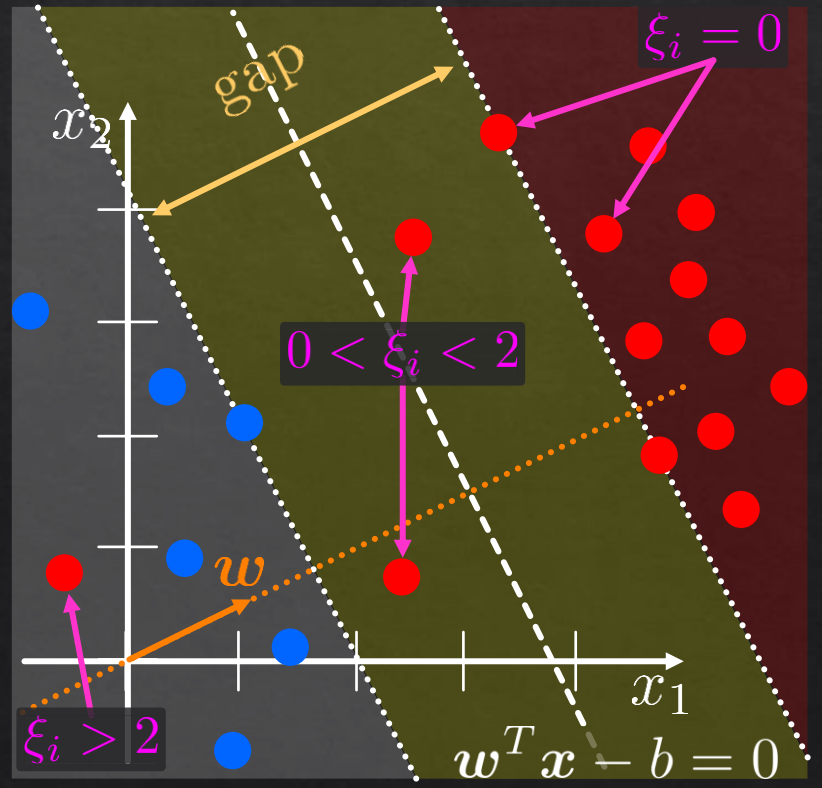
$$\xi_i := \max\left\{0, 1 - y_i \left(\boldsymbol{w}^T \boldsymbol{x}_i - b\right)\right\}$$

- ξ_i is the penalty cost for points
  on the wrong side of the margin

- The soft SVM is given by:

$$\min_{\boldsymbol{w},b} \frac{1}{2} \|\boldsymbol{w}\|^2 + C \sum_i \xi_i$$

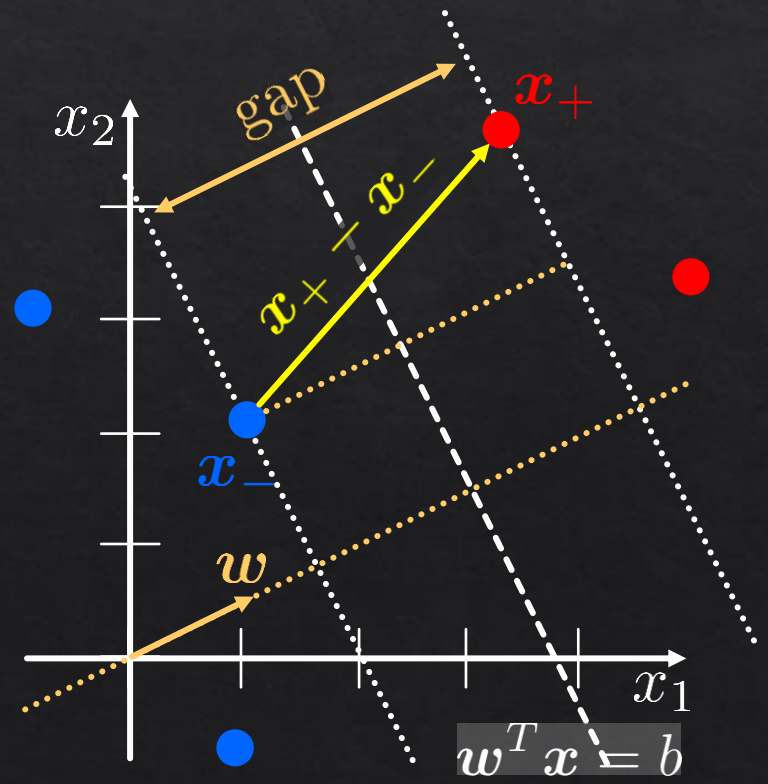
- $C > 0$ is the regularization factor.



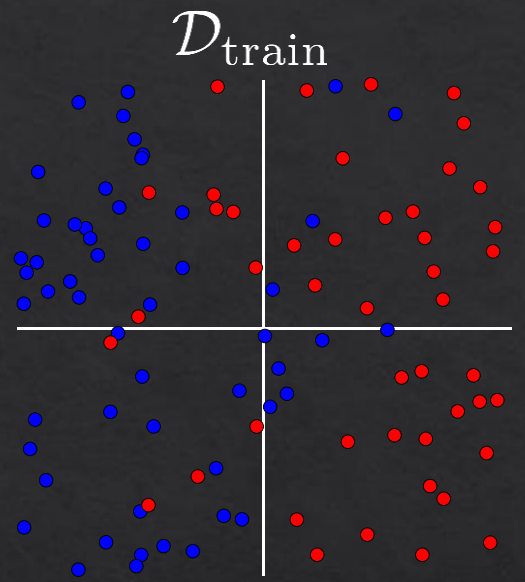
(Soft) SVM Example

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_i \xi_i$$

SVM Example



K-Nearest Neighbors



- Let $\mathcal{D}_{\text{train}} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}$.
- The 1–NN classifier is given by:

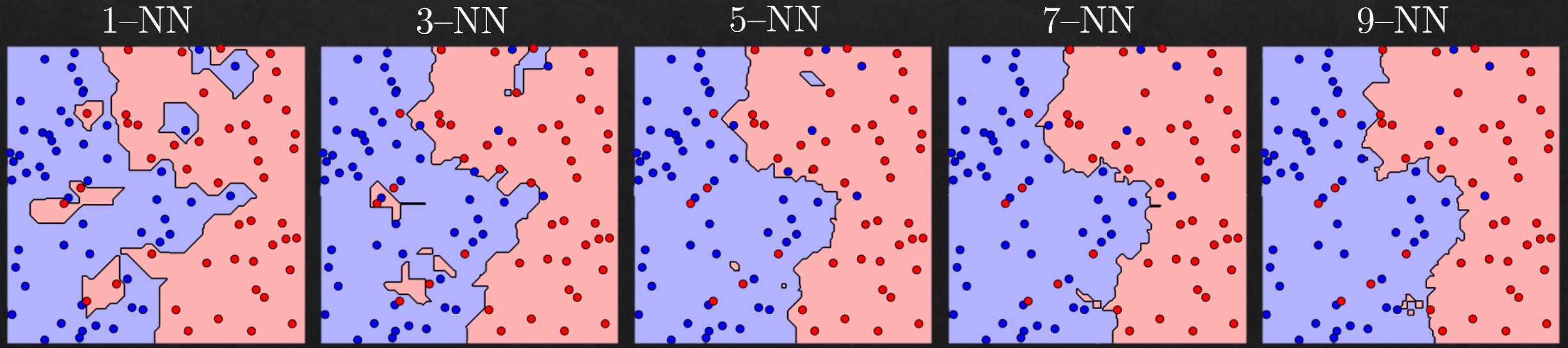
$$f_{\text{1–NN}}(\boldsymbol{x}) = y_{\hat{i}(\boldsymbol{x})}$$

where

$$\hat{i}(\boldsymbol{x}) = \arg\min_i d(\boldsymbol{x}, \boldsymbol{x}_i)$$

$\left(\begin{array}{c}\text{the model is}\\ \text{nonparametric}\end{array}\right)$

- For example, using the Euclidean metric $d(\boldsymbol{x}, \boldsymbol{x}_i) = \|\boldsymbol{x} - \boldsymbol{x}_i\|_2$:
- For $K > 1$, $\boldsymbol{x}$ will be labeled by the most common among its K NN.



K-Nearest Neighbors – Examples

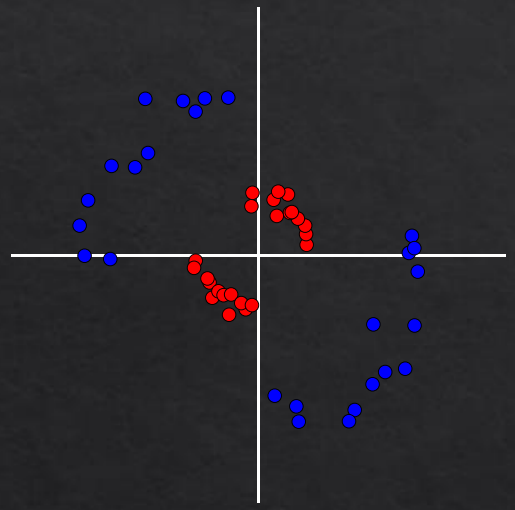
$$f_{1\text{–NN}}(\boldsymbol{x}) = y_{\hat{i}(\boldsymbol{x})}$$

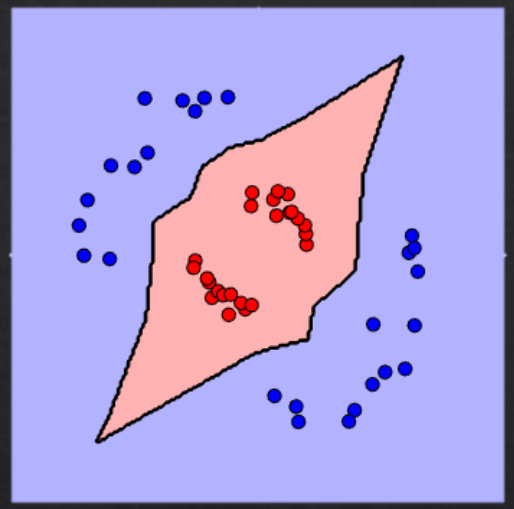
$$\hat{i}(\boldsymbol{x}) = \arg\min_i d(\boldsymbol{x}, \boldsymbol{x}_i)$$

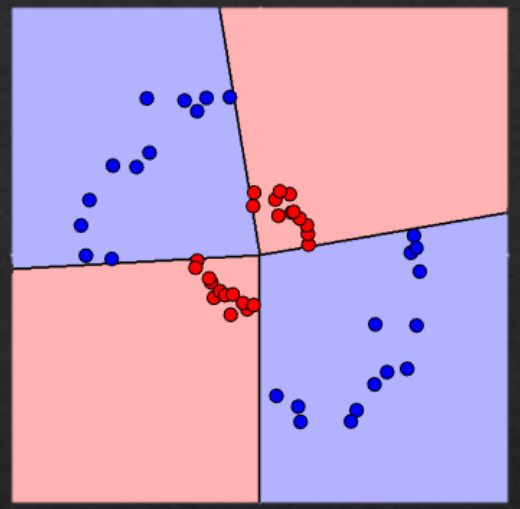
- 1–NN Euclidean and cosine distances:

$$d_{\text{Euclid}}(\boldsymbol{x}, \boldsymbol{x}_i) = \|\boldsymbol{x} - \boldsymbol{x}_i\|_2$$

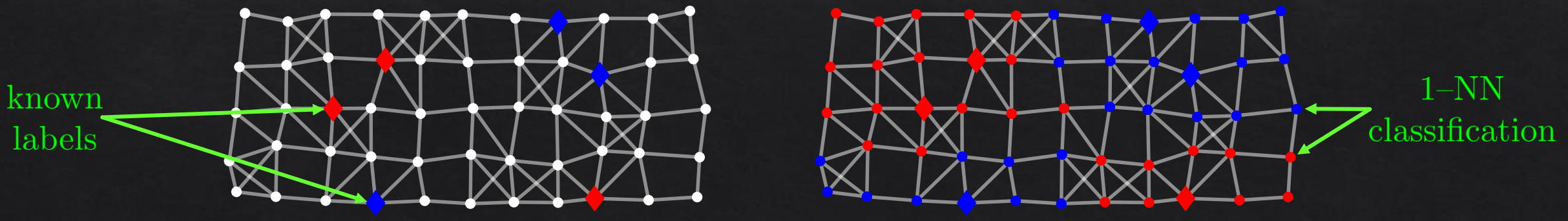
$$d_{\text{cosine}}(\boldsymbol{x}, \boldsymbol{x}_i) = 1 - \frac{\langle \boldsymbol{x}, \boldsymbol{x}_i \rangle}{\|\boldsymbol{x}\|_2 \|\boldsymbol{x}_i\|_2} = 1 - \cos(\theta)$$







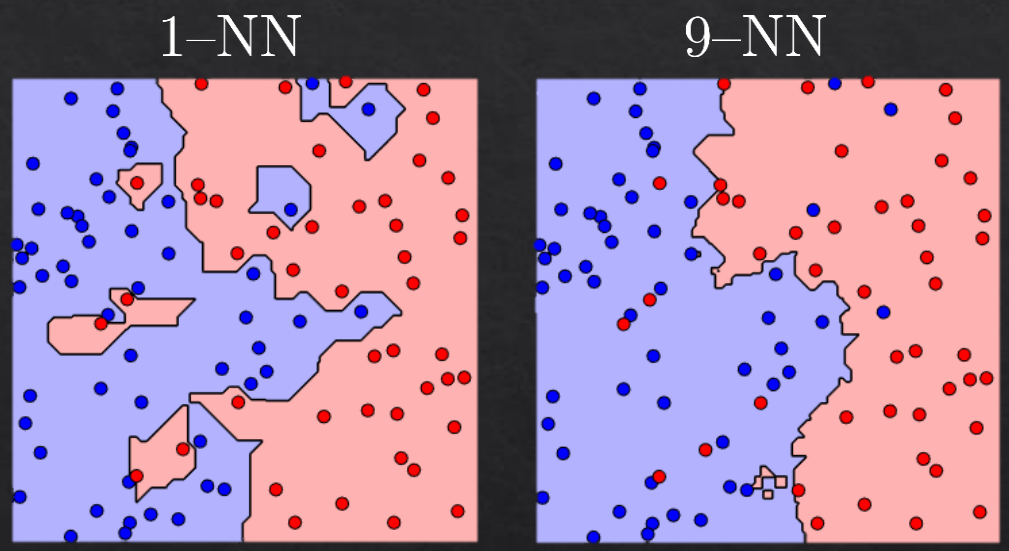
- 1–NN on graph (distance is shortest path):

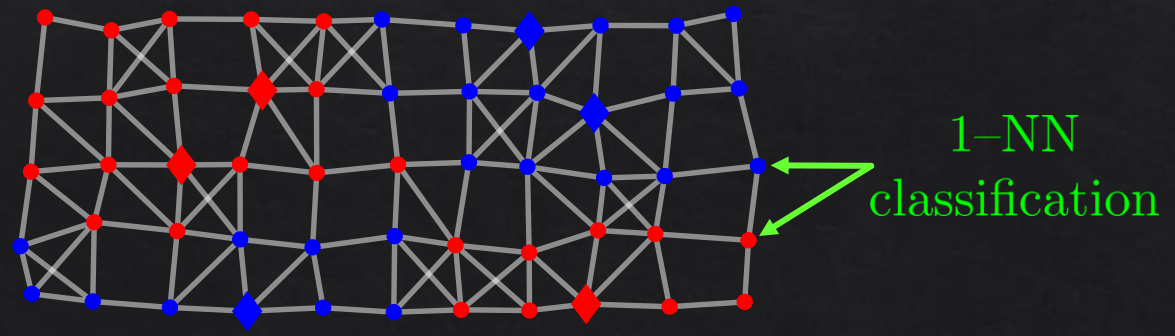


- What is the train accuracy of a 1–NN claisifer?

K-NN Example

K-NN Example





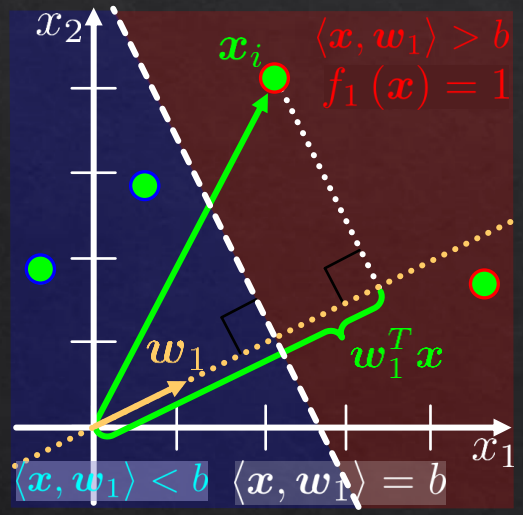
Questions

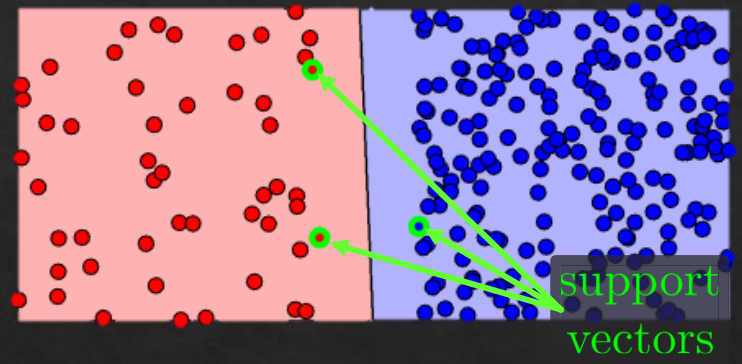
$$L_{\text{train}}(f) := \frac{1}{N}\sum_{i=1}^{N}\ell(\hat{y}_i, y_i) = \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}\{\hat{y}_i \neq y_i\}$$

$1 - L(f)$ is the classifier accuracy

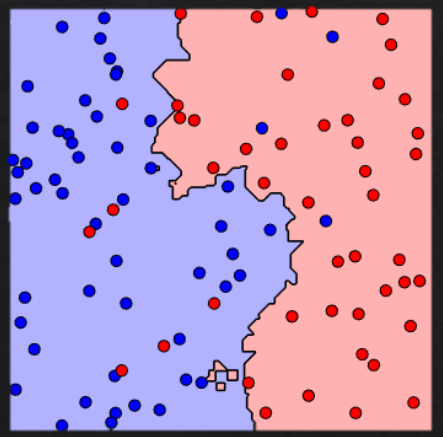
$L(f) = 0 \implies$ no errors

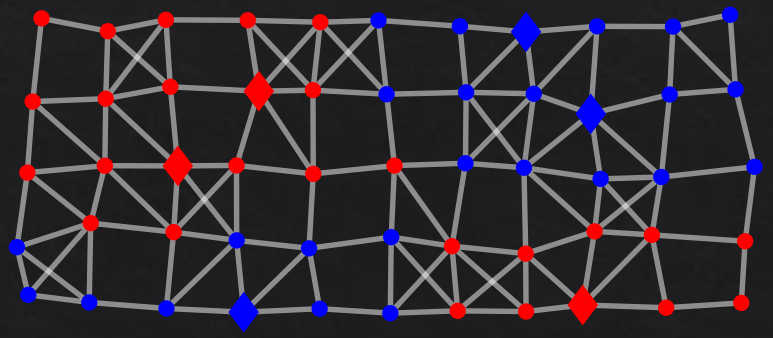
SVM:





K–NN





fixel your pixels

fixelalgorithms.gitlab.io