Logistic Regression & Support Vector Machines

Game Plan

- Basic classification algorithms
  - Nearest neighbor
  - Naïve Bayes
  - Perceptrons
    - Our first linear classifier

- **Today.** Advanced linear classifiers
  - Logistic Regression
  - Support Vector Machine

Recap

- Linear classifiers can be parameterized as:

$$f_\theta(\mathbf{x}) = \mathbf{1}\{\theta^\top \tilde{\mathbf{x}} > 0\}$$

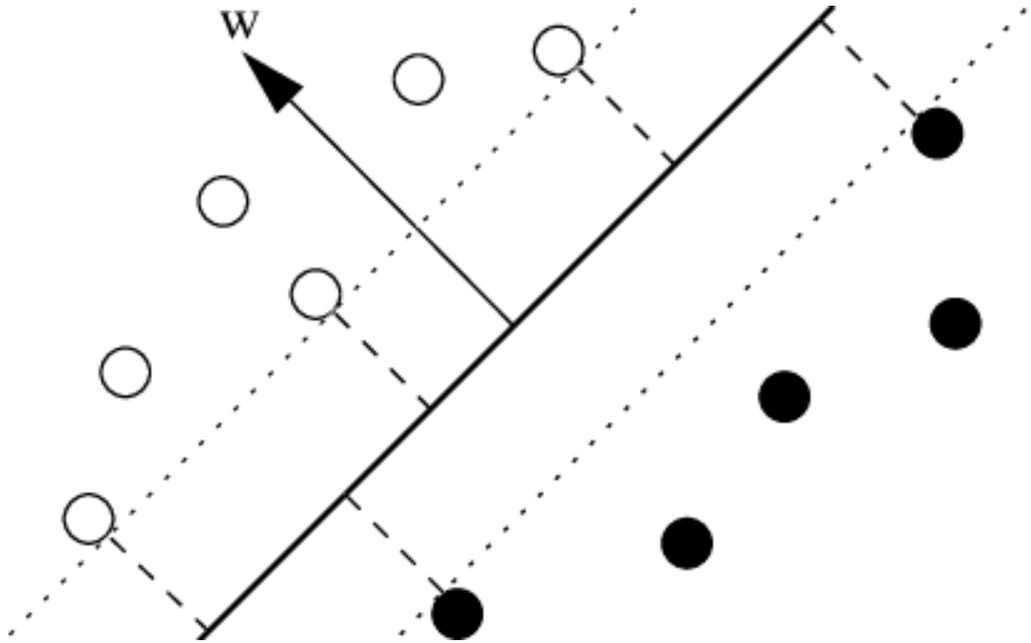
- **Question.** Which loss?
  - 0-1 loss is difficult to optimize

$$\ell(f_\theta(\mathbf{x}), y) = \mathbf{1}\{f_\theta(\mathbf{x}) \neq y\}$$

  - Perceptron uses the surrogate loss

$$\ell(f_\theta(\mathbf{x}), y) = (f_\theta(\mathbf{x}) - y) \cdot \theta^\top \tilde{\mathbf{x}}$$

    - $\theta^\top \tilde{\mathbf{x}}$ interpreted as “confidence”
    - Works, but somewhat arbitrary



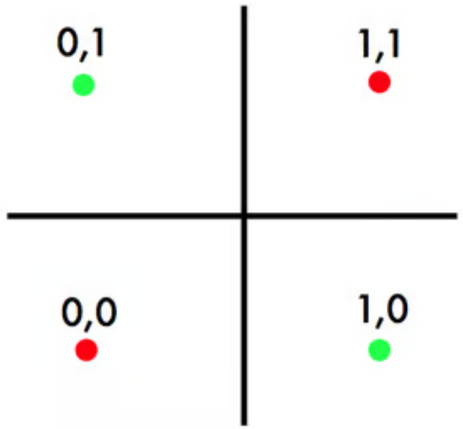
Recap

- **Limitations.** Perceptron works, but…

$$\ell(f_\theta(\mathbf{x}), y) = (f_\theta(\mathbf{x}) - y) \cdot \theta^\top \tilde{\mathbf{x}}$$

  - Calling $\theta^\top \tilde{\mathbf{x}}$ confidence is somewhat arbitrary
    - Any real probabilitic interpretation?

  - The iterative optimization algorithm may not converge
    - XOR example — we want imperfect but good solution



Logistic Regression

Logistic Regression

- Another popular method to train the linear classifier

$$f_\theta(\mathbf{x}) = \mathbf{1}\{\theta^\top \tilde{\mathbf{x}} \geq 0\}$$

- Logistic regression interprets $\theta^\top \tilde{\mathbf{x}}$ as a log-likelihood ratio of the model's internal probability estimate

$$\log\left(\frac{p(y=1\,|\,\mathbf{x})}{p(y=0\,|\,\mathbf{x})}\right) \approx \theta^\top \tilde{\mathbf{x}}$$

- **Brainteaser.** Why not interpret as $p(y=1\,|\,\mathbf{x})$?

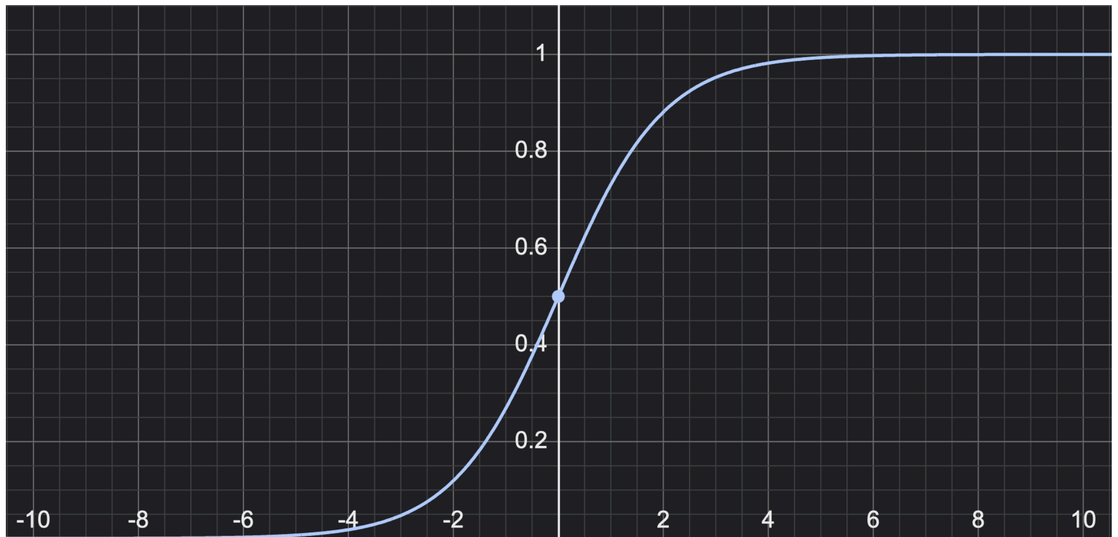
Logistic Regression

$$\log\left(\frac{p(y=1\,|\,\mathbf{x})}{p(y=0\,|\,\mathbf{x})}\right) \approx \theta^{\top}\tilde{\mathbf{x}}$$

- In other words, we are modeling the posterior distribution as

$$p(y=1\,|\,\mathbf{x}) = \frac{1}{1+\exp(-\theta^{\top}\tilde{\mathbf{x}})}$$

- The function $\sigma(t) = 1/1+\exp(-t)$ is the logistic function (a.k.a. sigmoid)



Training

- Given the data, maximize the log-likelihood

$$\max_{\theta} \frac{1}{n} \sum_{i=1}^{n} \log p(y_i \mid \mathbf{x}_i)$$

- Equivalently, minimize the NLL loss:

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \log \left(\frac{1}{p(y_i \mid \mathbf{x}_i)} \right)$$

Training

- Equivalently again, we are solving:

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f_\theta(\mathbf{x}_i))$$

where

- $f_\theta(\cdot)$ is the sigmoid of the prediction

$$f_\theta(\mathbf{x}) = \sigma(\theta^\top \tilde{\mathbf{x}})$$

- $\ell(\cdot, \cdot)$ is the cross-entropy

$$\ell(y, t) = \mathrm{CE}(\mathbf{1}_y, [t, 1-t]) = \log(t)^{-y} + \log(1-t)^{y-1}$$

Training

- More tediously, this can be written as

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} (-y_i)\log(\sigma(\theta^\top \tilde{\mathbf{x}}_i)) + (y_i - 1)\log(1 - \sigma(\theta^\top \tilde{\mathbf{x}}_i))$$

- Thankfully, this is **convex**!
  - Definition. A function $g(\theta)$ is called convex when it satisfies:

$$g(\lambda\theta_1 + (1-\lambda)\theta_2) \leq \lambda g(\theta_1) + (1-\lambda)g(\theta_2), \quad \forall \lambda \in [0,1]$$

(strictly convex, if holds with <)

  - Convex, regardless of the data

Training

- **Facts.** For convex functions:
  - If strictly convex, the solution is unique
  - Gradient descent provably converges to this unique solution

- The gradient descent iteration can be written as:

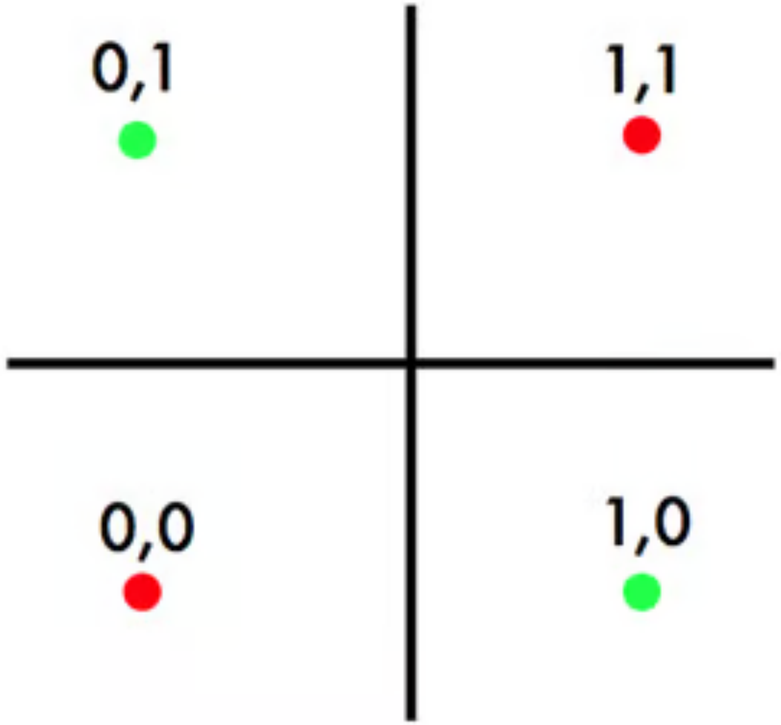
$$\theta^{(\text{new})} = \theta + \eta \cdot \frac{1}{n} \sum_{i=1}^{n} (y_i - \sigma(\theta^\top \tilde{\mathbf{x}}_i)) \tilde{\mathbf{x}}_i$$

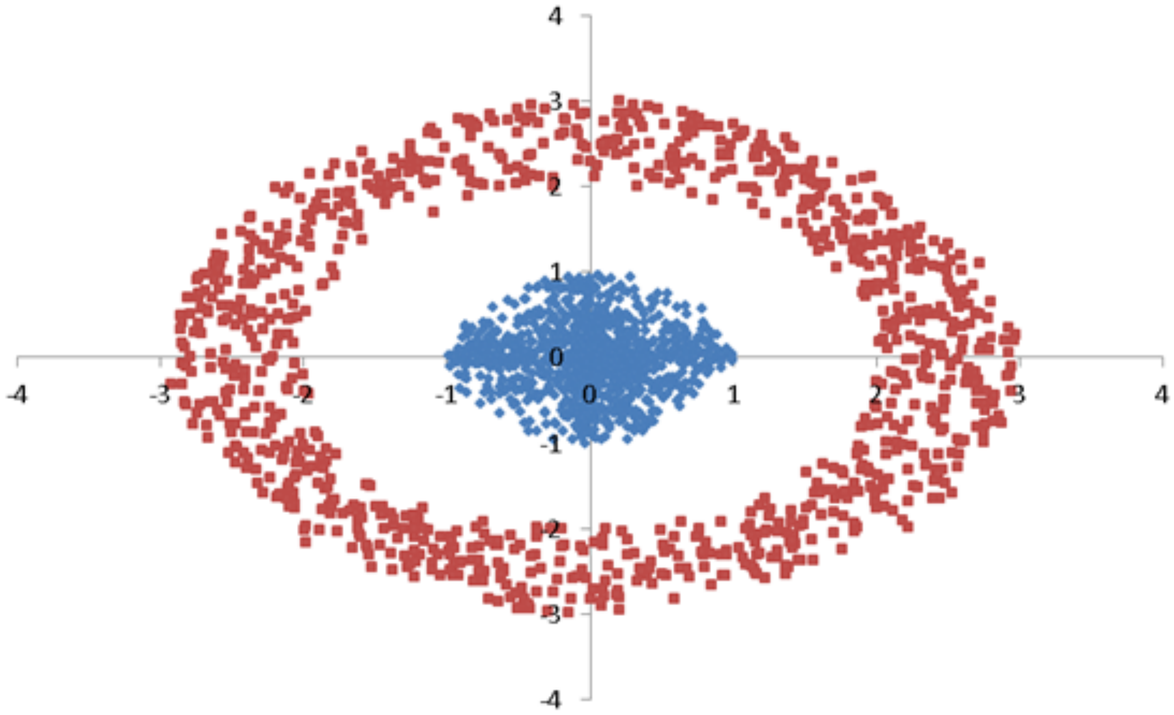
Properties

- **Computation.** Relatively easy
  - Training. Requires GD, but is convex
  - Inference. Easy — Dot product and apply threshold

Limitation

- It converges to the solution that minimizes training loss
  - But this minimum training loss is still high, for non-separable data





Limitation

- Does not admit an analytic solution
  - Even for linear models, requires running long GD

- Particularly weak against “far outliers”

$$\theta^{(\text{new})} = \theta + \eta \cdot \frac{1}{n} \sum_{i=1}^{n} (y_i - \sigma(\theta^\top \tilde{\mathbf{x}}_i)) \tilde{\mathbf{x}}_i$$

- Lacks much consideration on the generalizability

Support Vector Machines

Disclaimers

Peyman Milanfar
@docmilanfar

Of all the machine learning ideas I've been exposed to over the years, I think SVMs were by far the most boring; followed closely by PAC learning.

Disclaimers

Doc Xardoc @andrewb10687674 · 1h

I will always have a soft spot in my heart for SVM's because one of the first data science problems I worked on I struggled with for months and then ran it through an SVM and solved it within a half hour.

222

Disclaimers

Computer Science > Machine Learning

[Submitted on 31 Aug 2023 (v1), last revised 22 Feb 2024 (this version, v3)]

Transformers as Support Vector Machines

Davoud Ataee Tarzanagh, Yingcong Li, Christos Thrampoulidis, Samet Oymak

Since its inception in "Attention Is All You Need", transformer architecture has led to revolutionary advancements in NLP. The attention layer within the transformer admits a sequence of input tokens X and makes them interact through pairwise similarities computed as softmax$(\mathrm{X}\mathrm{Q}\mathrm{K}^\top\mathrm{X}^\top)$, where (K,Q) are the trainable key-query parameters. In this work, we establish a formal equivalence between the optimization geometry of self-attention and a hard-margin SVM problem that separates optimal input tokens from non-optimal tokens using linear constraints on the outer-products of token pairs. This formalism allows us to characterize the implicit bias of 1-layer transformers optimized with gradient descent: (1) Optimizing the attention layer with vanishing regularization, parameterized by (K,Q), converges in direction to an SVM solution minimizing the nuclear norm of the combined parameter $\mathrm{W} = \mathrm{K}\mathrm{Q}^\top$. Instead, directly parameterizing by W minimizes a Frobenius norm objective. We characterize this convergence, highlighting that it can occur toward locally-optimal directions rather than global ones. (2) Complementing this, we prove the local/global directional convergence of gradient descent under suitable geometric conditions. Importantly, we show that over-parameterization catalyzes global convergence by ensuring the feasibility of the SVM problem and by guaranteeing a benign optimization landscape devoid of stationary points. (3) While our theory applies primarily to linear prediction heads, we propose a more general SVM equivalence that predicts the implicit bias with nonlinear heads. Our findings are applicable to arbitrary datasets and their validity is verified via experiments. We also introduce several open problems and research directions. We believe these findings inspire the interpretation of transformers as a hierarchy of SVMs that separates and selects optimal tokens.

Comments: The proof of global convergence for gradient descent in the equal score setting has been fixed, referring to Theorem 2 of [TLZO23], and the experimental results have been extended

Subjects: **Machine Learning (cs.LG)**; Artificial Intelligence (cs.AI); Computation and Language (cs.CL); Optimization and Control (math.OC)

Cite as: arXiv:2308.16898 **[cs.LG]**

(or arXiv:2308.16898v3 **[cs.LG]** for this version)

https://doi.org/10.48550/arXiv.2308.16898

Submission history

From: Yingcong Li [view email]

[v1] Thu, 31 Aug 2023 17:57:50 UTC (1,671 KB)

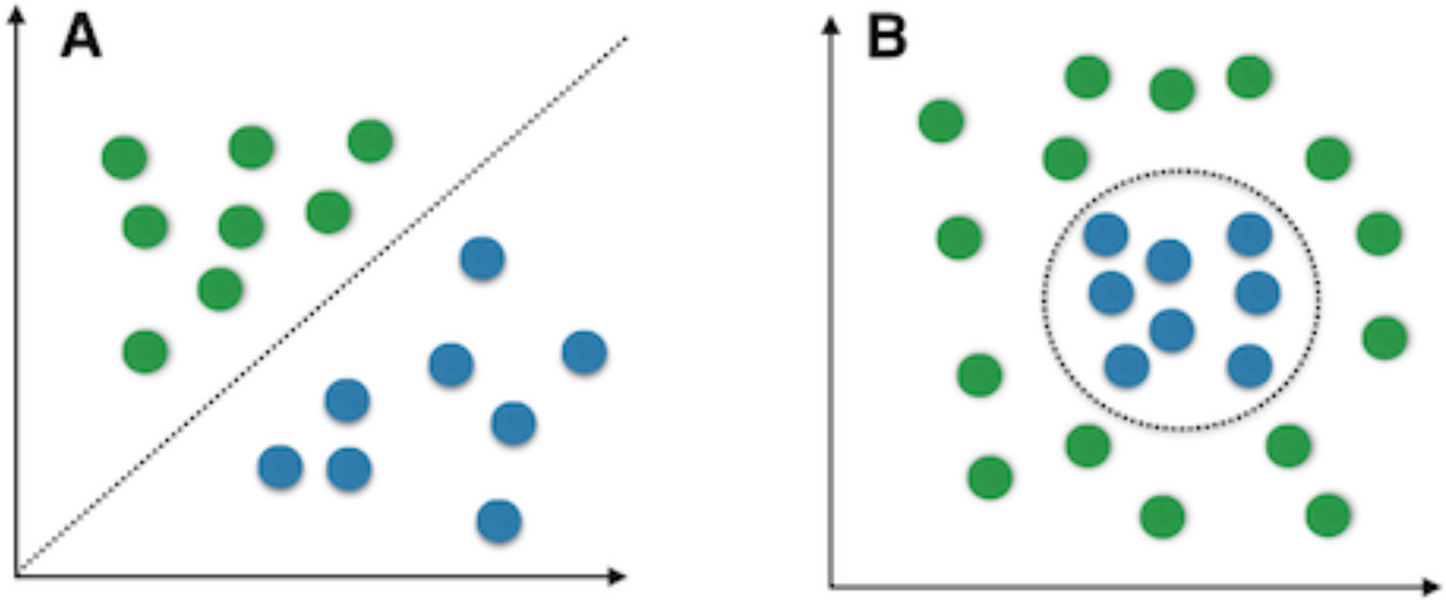
[v2] Thu, 7 Sep 2023 17:50:52 UTC (1,835 KB)

[v3] Thu, 22 Feb 2024 18:38:14 UTC (1,081 KB)

Philosophy

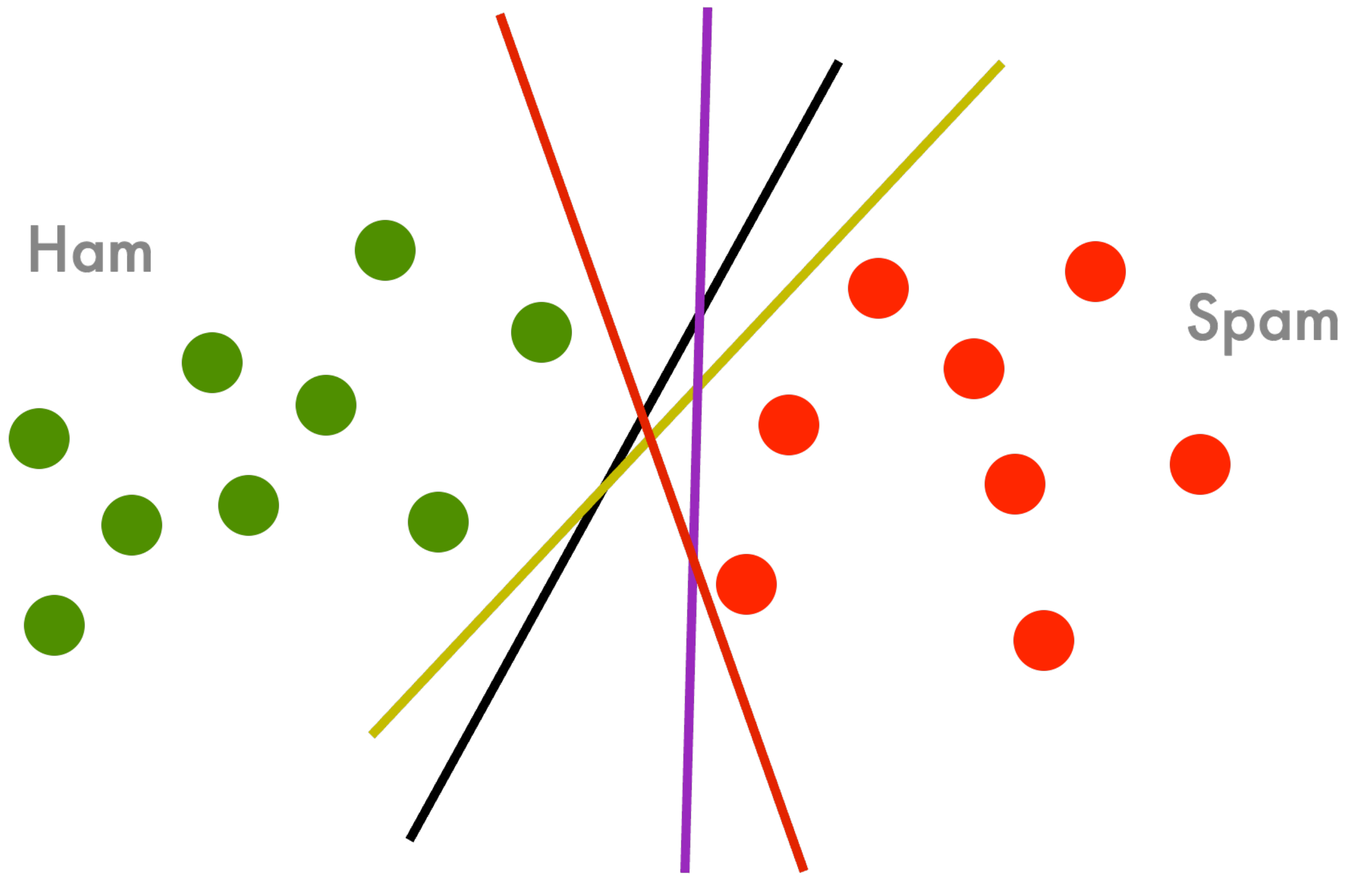
- Motivation comes from “what function generalizes better”

- Suppose that we have a **linearly separable** data
  - i.e., exists a linear classifier that perfectly classifies the training data



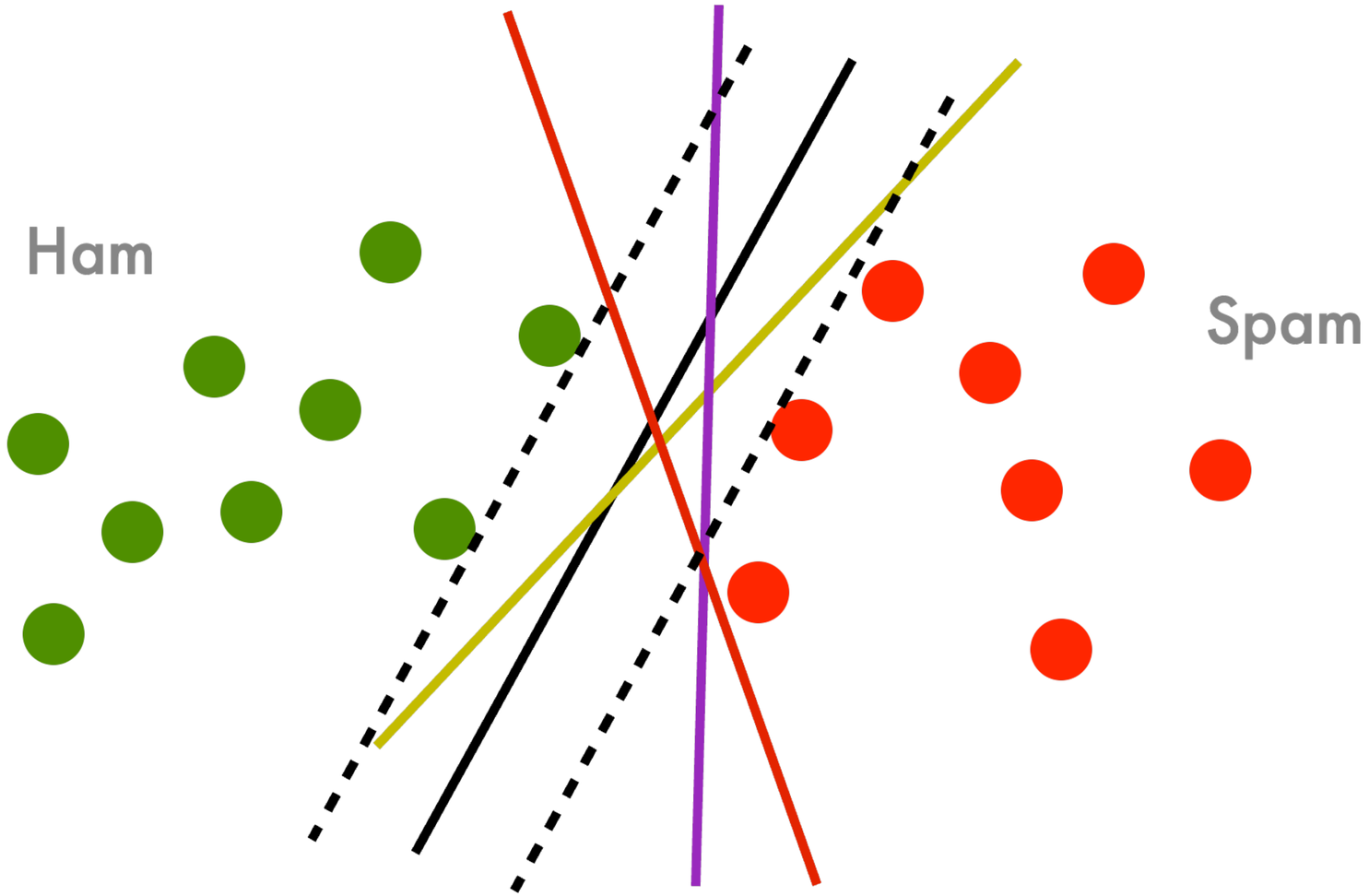
Philosophy

- Then, there can be **infinitely many** classifiers with perfect accuracy on the training data…
- **Question.** What is the best one?



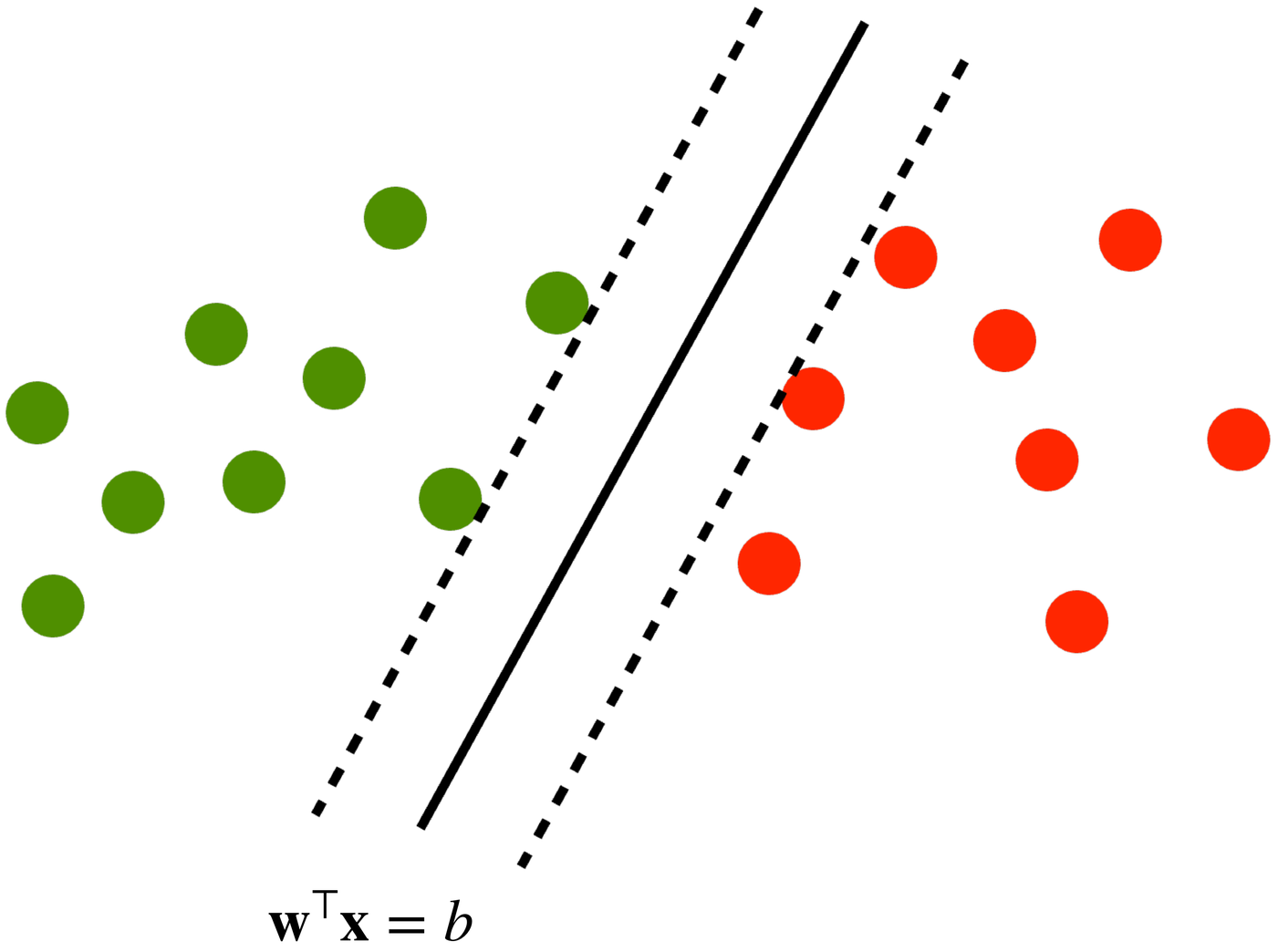
Philosophy

- **Idea.** We should choose the **maximum-margin** classifier
  - Reason. Robust to noise in the test data



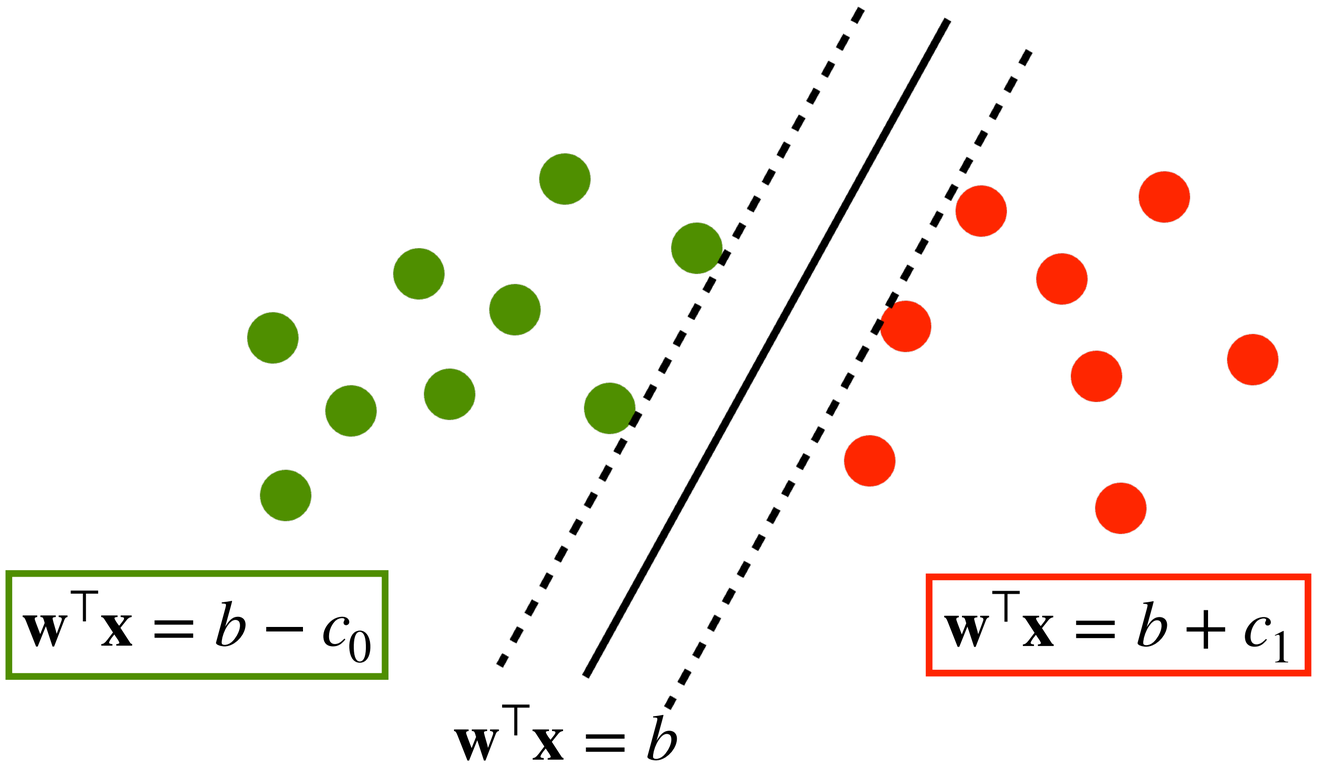
Margin

- **Question.** How do we formalize and quantify margin?



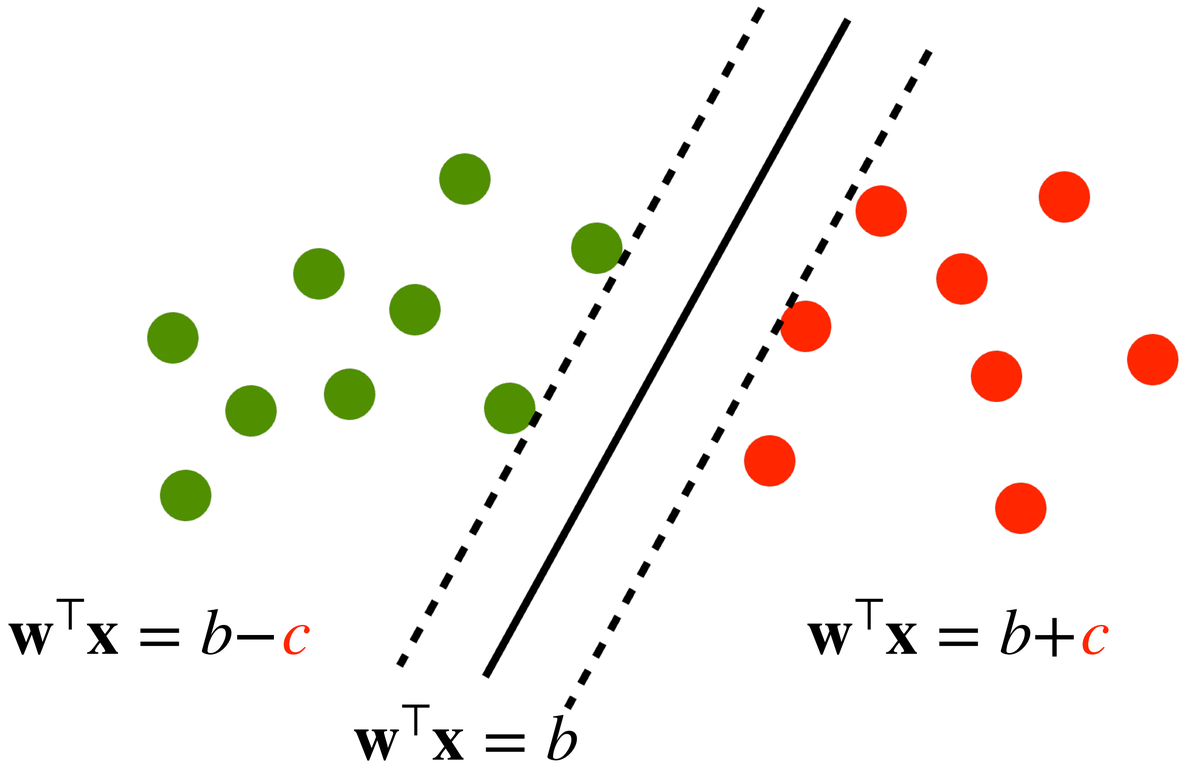
Margin

- **Answer.** Margin = Maximum shift the classifier can withstand



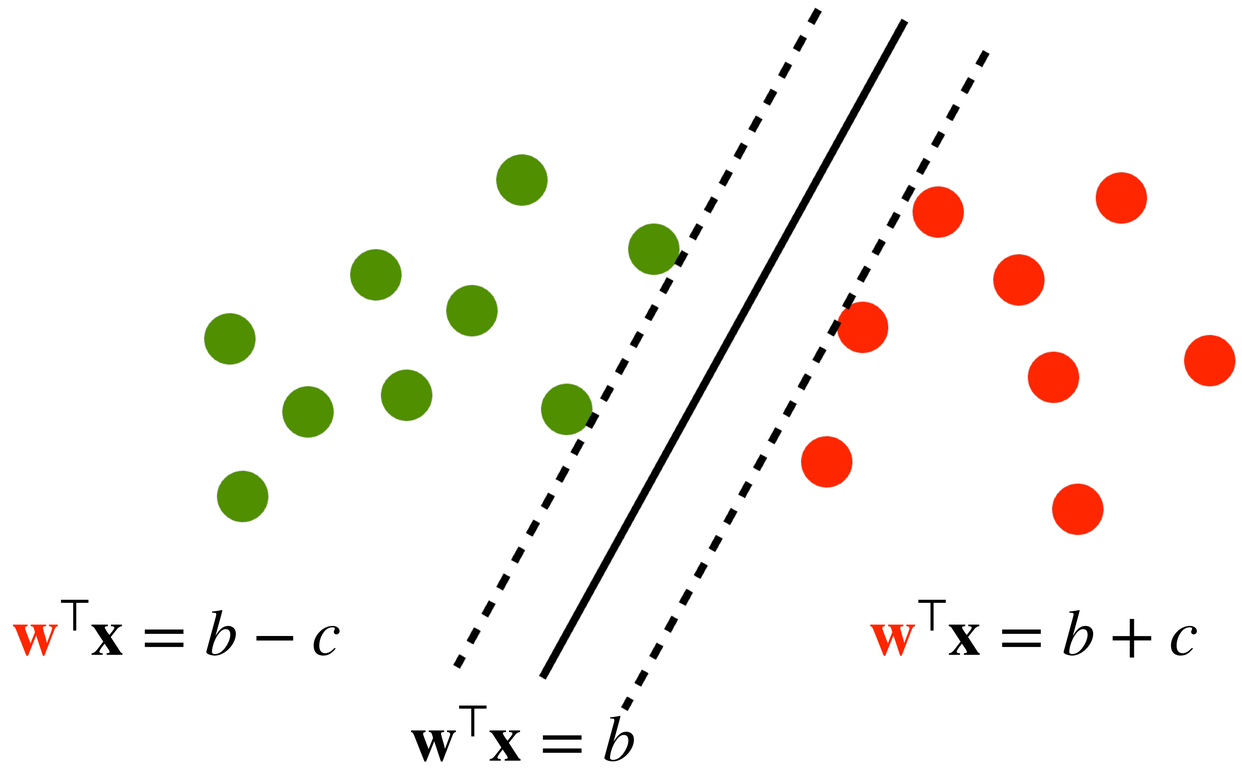
Margin

- **Answer.** Margin = Maximum shift the classifier can withstand
  - Take the midpoint



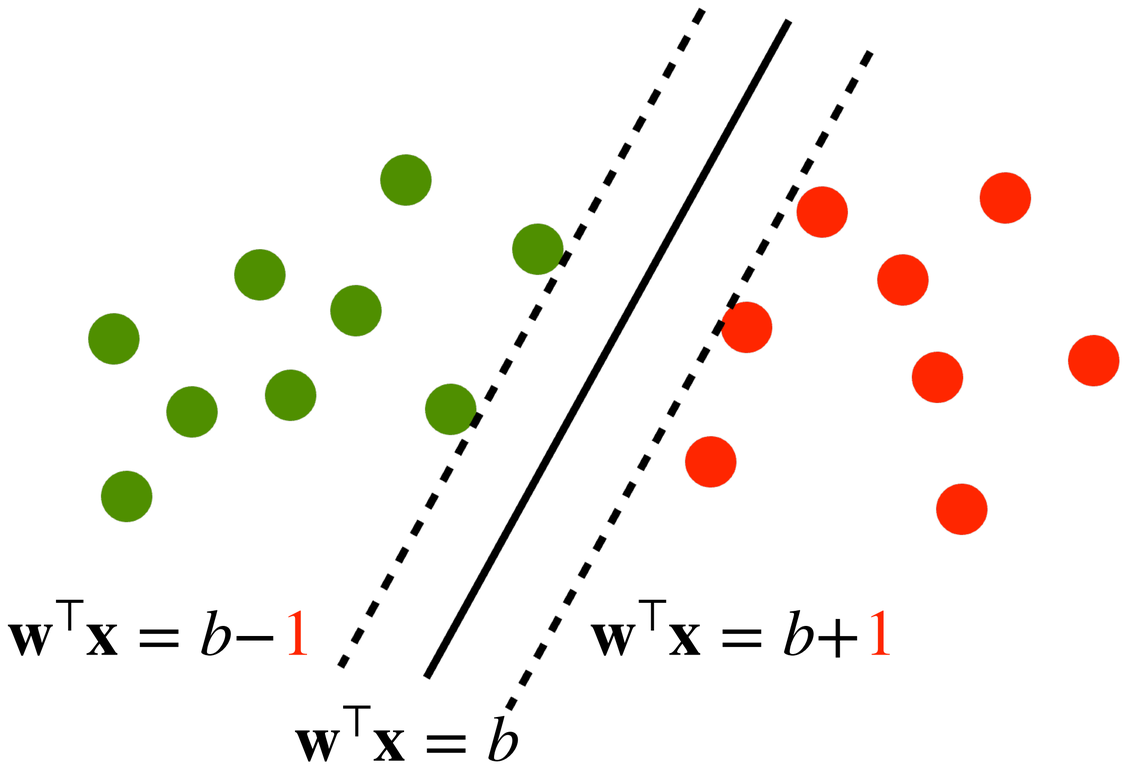
Margin

- **Answer.** Margin = Maximum shift the classifier can withstand
  - Take the midpoint
  - Normalize the size of $\mathbf{w}$ — Otherwise c can be arbitrarily large



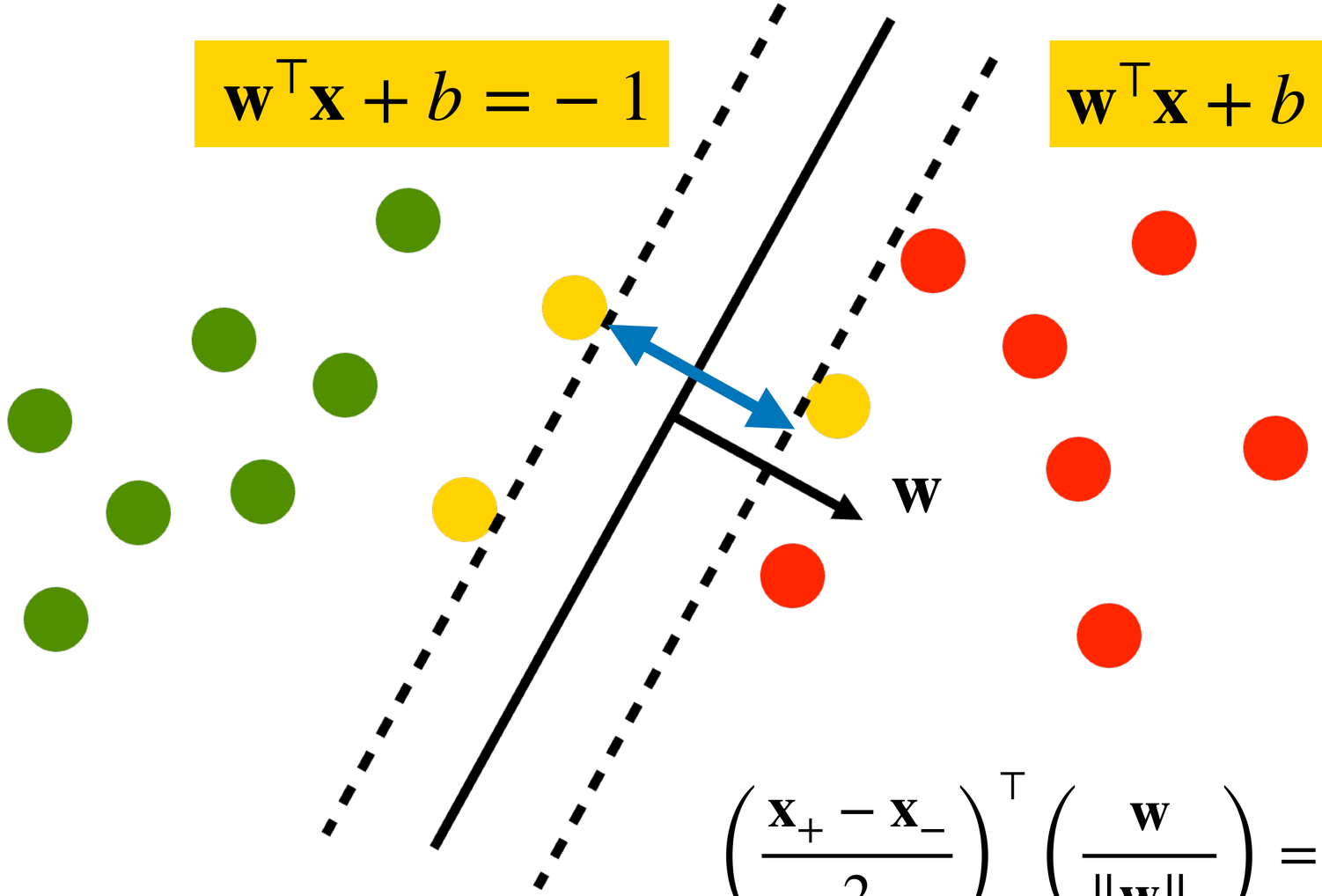
Margin

- **Answer.** Margin = Maximum shift the classifier can withstand
  - Take the midpoint
- Normalize the size of $\mathbf{w}$ – Otherwise c can be arbitrarily large
  - We can just fix $c = 1$, and look at the size of $1/\|\mathbf{w}\|_2$



Margin

- **Cleaner(?) Intuition.** Project it along the direction



$$\left(\frac{\mathbf{x}_+ - \mathbf{x}_-}{2}\right)^\top \left(\frac{\mathbf{w}}{\|\mathbf{w}\|_2}\right) = \frac{1}{\|\mathbf{w}\|}$$ ⇐ “Margin”

Maximum Margin Classifier

- SVM is designed to find a maximum-margin classifier
  - That is, we solve the constrained optimization problem:

$$\text{maximize}_{\mathbf{w},b} \frac{1}{\|\mathbf{w}\|_2} \qquad \text{subject to} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1$$

  - Maximize margin, subject to training accuracy = 100%

(We use $y_i \in \{-1, +1\}$, instead of the usual $\{0,1\}$)

Maximum Margin Classifier

- Rephrasing it slightly, we can make it a minimization problem

$$\ell^* = \min_{\mathbf{w},b} \frac{\|\mathbf{w}\|_2^2}{2} \qquad \text{subject to} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1$$

- **Question.** How do we solve this constrained optimization problem?

Solving the optimization: Dual

$$\ell^* = \min_{\mathbf{w},b} \frac{\|\mathbf{w}\|_2^2}{2} \quad \text{subject to} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1$$

- **Answer.** Consider the Lagrangian dual of the problem
  - Above is called the “primal” problem
  - Below is called the “dual”

(constraint-free)

How much you violated

$$\mathscr{L}(\mathbf{w}, b, \alpha) = \frac{\|\mathbf{w}\|^2}{2} + \sum_{i=1}^{n} \alpha_i \left(1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)\right)$$

Solving the optimization: Dual

$$\ell^* = \min_{\mathbf{w},b} \frac{\|\mathbf{w}\|_2^2}{2} \qquad \text{subject to} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1$$

- **Answer.** Consider the Lagrangian dual of the problem
  - Above is called the “primal” problem
  - Below is called the “dual”

$$\mathscr{L}(\mathbf{w}, b, \alpha) = \frac{\|\mathbf{w}\|^2}{2} + \sum_{i=1}^{n} \alpha_i \big(1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)\big)$$

- Then, interestingly, the following duality holds — Why?

$$\ell^* = \min_{\mathbf{w},b} \max_{\alpha \geq 0} \mathscr{L}(\mathbf{w}, b, \alpha)$$

Solving the optimization: Dual

$$\ell^* = \min_{\mathbf{w},b} \frac{\|\mathbf{w}\|^2}{2} \quad \text{subject to} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 \qquad \textbf{(primal)}$$

$$\ell^* = \min_{\mathbf{w},b} \max_{\alpha \geq 0} \frac{\|\mathbf{w}\|^2}{2} + \sum_{i=1}^{n} \alpha_i \left(1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)\right) \qquad \textbf{(dual)}$$

- **Intuition.** In dual, the adversary will choose α to maximize your loss
  - He/she will carefully look at the sign of $1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)$

Solving the optimization: Dual

$$\ell^* = \min_{\mathbf{w},b} \frac{\|\mathbf{w}\|^2}{2} \quad \text{subject to} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 \qquad \textbf{(primal)}$$

$$\ell^* = \min_{\mathbf{w},b} \max_{\alpha \geq 0} \frac{\|\mathbf{w}\|^2}{2} + \sum_{i=1}^{n} \alpha_i \left(1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)\right) \qquad \textbf{(dual)}$$

- **Intuition.** In dual, the adversary will choose α to maximize your loss
  - He/she will carefully look at the sign of $1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)$
    - If positive: Set $\alpha_i \to \infty$ P: Infeasible, D: ∞

Solving the optimization: Dual

$$\ell^* = \min_{\mathbf{w},b} \frac{\|\mathbf{w}\|^2}{2} \quad \text{subject to} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 \qquad \textbf{(primal)}$$

$$\ell^* = \min_{\mathbf{w},b} \max_{\alpha \geq 0} \frac{\|\mathbf{w}\|^2}{2} + \sum_{i=1}^{n} \alpha_i \left(1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)\right) \qquad \textbf{(dual)}$$

- Intuition. In dual, the adversary will choose α to maximize your loss
  - He/she will carefully look at the sign of $1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)$
    - If positive: Set $\alpha_i \to \infty$ P: Infeasible, D: ∞
    - If negative: Set $\alpha_i = 0$ P = D

Solving the optimization: Dual

$$\ell^* = \min_{\mathbf{w},b} \frac{\|\mathbf{w}\|^2}{2} \quad \text{subject to} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 \qquad \textbf{(primal)}$$

$$\ell^* = \min_{\mathbf{w},b} \max_{\alpha \geq 0} \frac{\|\mathbf{w}\|^2}{2} + \sum_{i=1}^{n} \alpha_i \left(1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)\right) \qquad \textbf{(dual)}$$

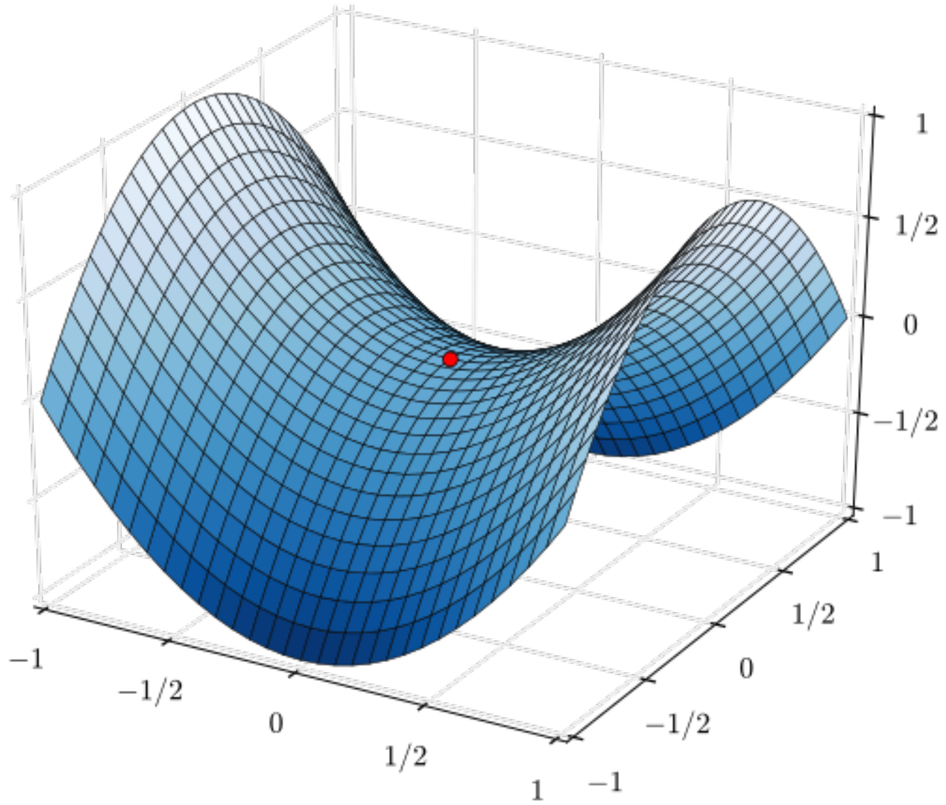
- Intuition. In dual, the adversary will choose α to maximize your loss
  - He/she will carefully look at the sign of $1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)$
    - If positive: Set $\alpha_i \to \infty$ P: Infeasible, D: ∞
    - If negative: Set $\alpha_i = 0$ P = D
    - If zero: Set α_i to be any value P = D (data on margin)

Solving the optimization: Dual

- As primal = dual, we can solve dual instead:

$$\ell^* = \min_{\mathbf{w},b} \max_{\alpha \geq 0} \quad \frac{\|\mathbf{w}\|^2}{2} + \sum_{i=1}^{n} \alpha_i \left(1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)\right)$$

- **Properties**
  - Constraint-free
  - Minimax problem
    - Saddle point finding



Solving the Dual

- To find the saddle point, find the **critical point**
  - The gradient can be written as:

$$\nabla_{\mathbf{w}} \mathcal{L} = \mathbf{w} - \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \qquad \nabla_b \mathcal{L} = -\sum_{i=1}^{n} \alpha_i y_i$$

  - Setting them equal to zero, we get

$$\mathbf{w}^* = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \qquad 0 = \sum_{i=1}^{n} \alpha_i y_i$$

Solving the Dual

- Plugging $\mathbf{w}^*$ back to Lagrangian, we get:

$$\mathscr{L} = -\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j \mathbf{x}_i^\top \mathbf{x}_j + \sum_{i=1}^{n}\alpha_i$$

- Summing up, the dual problem becomes:

$$\max_{\alpha}\left(-\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j \mathbf{x}_i^\top \mathbf{x}_j + \sum_{i=1}^{n}\alpha_i\right)$$
$$\text{subject to} \quad \sum_{i}\alpha_i y_i = 0, \;\; \alpha_i \geq 0$$

Solving the Dual

- Slightly rephrasing, this becomes a **quadratic program**
  - The search space is a convex polytope

$$\max_{\alpha} \left(-\frac{1}{2}\alpha^{\top}\mathbf{Z}\alpha + \mathbf{1}^{\top}\alpha \right) \qquad \text{subject to} \quad \alpha^{\top}\mathbf{y} = 0, \quad \alpha \succeq 0$$

- The solution is at either
  - Critical point
  - Extreme points

- Exists many automated solvers to get the optimal α^*

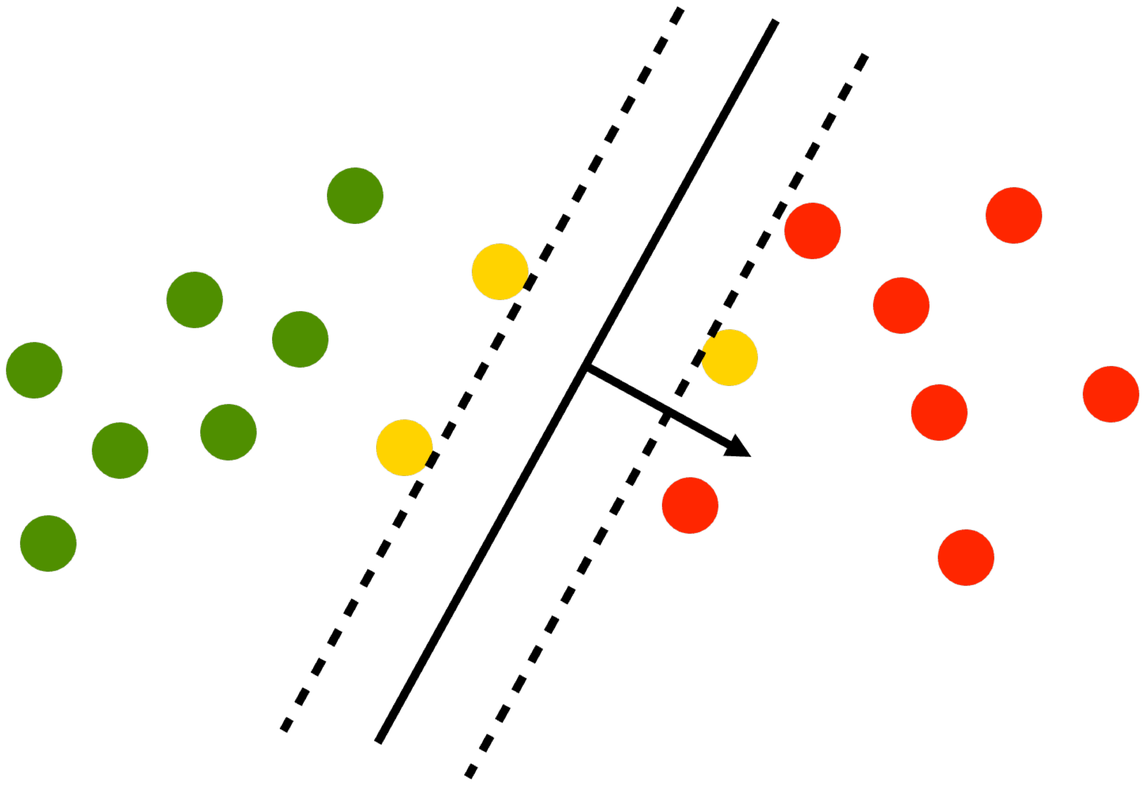
Solving the Bias

- Having computed α^*, our optimal weights become

$$\mathbf{w}^* = \sum_{i=1}^{n} \alpha_i^* y_i \mathbf{x}_i$$

Nonzero only for margin data (support vectors)

- **Question.** How about b^*?



Solving the Bias

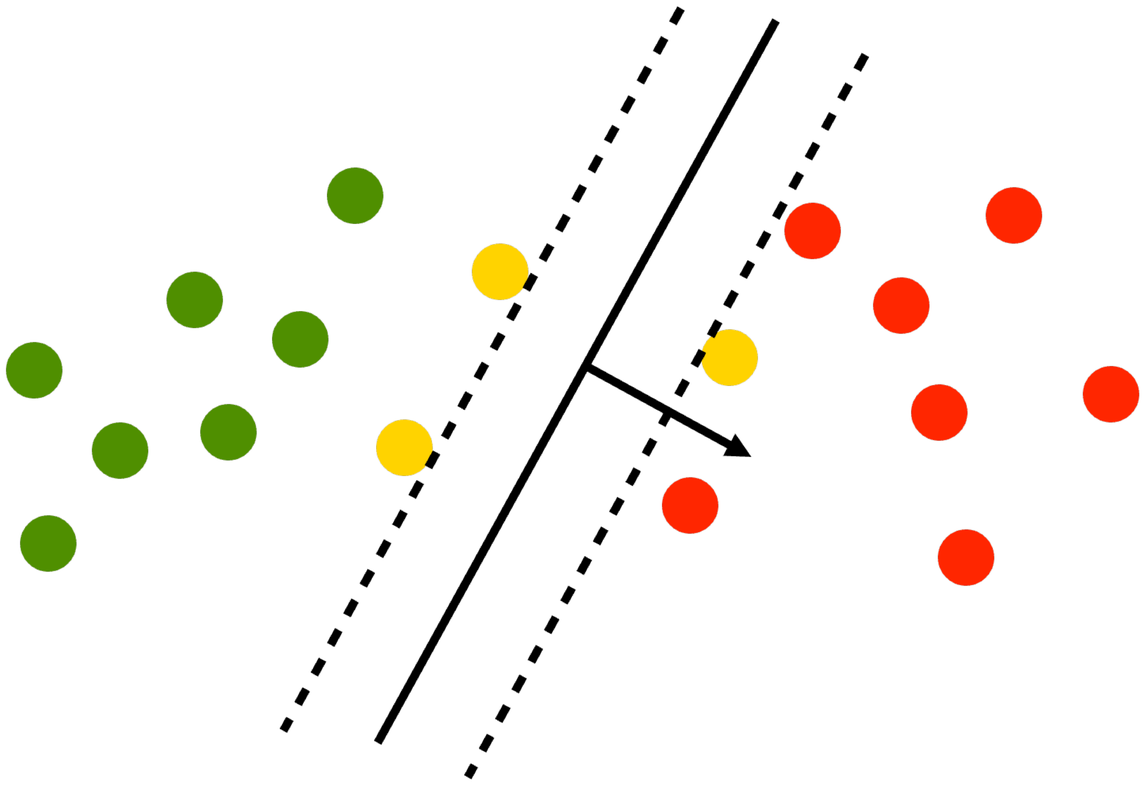
- Having computed α^*, our optimal weights become

$$\mathbf{w}^* = \sum_{i=1}^{n} \alpha_i^* y_i \mathbf{x}_i$$

Nonzero only for margin data (support vectors)

- Question. How about b^*?
- Answer. Plug in any support vector:

$$\mathbf{w}^{*\top}\mathbf{x} - b^* = \pm 1$$



Wrapping up

- Maximum margin principle leads to an analytical solution
  - We can put some extra condition to generalize better to the test data

- **Next class.**
  - Making this robust to outliers (soft SVM)
  - Handling non-linear data (kernel SVM)

</lecture 5>