Support Vector Machines

EECE454 Intro. to Machine Learning Systems

Fall 2024

Today

- **Last class.** Simple classifiers
    - Nearest neighbors
    - Naïve Bayes
    - Perceptrons
- **Today.** More linear models
    - Logistic Regression
    - Support Vector Machines

Recap: Perceptrons

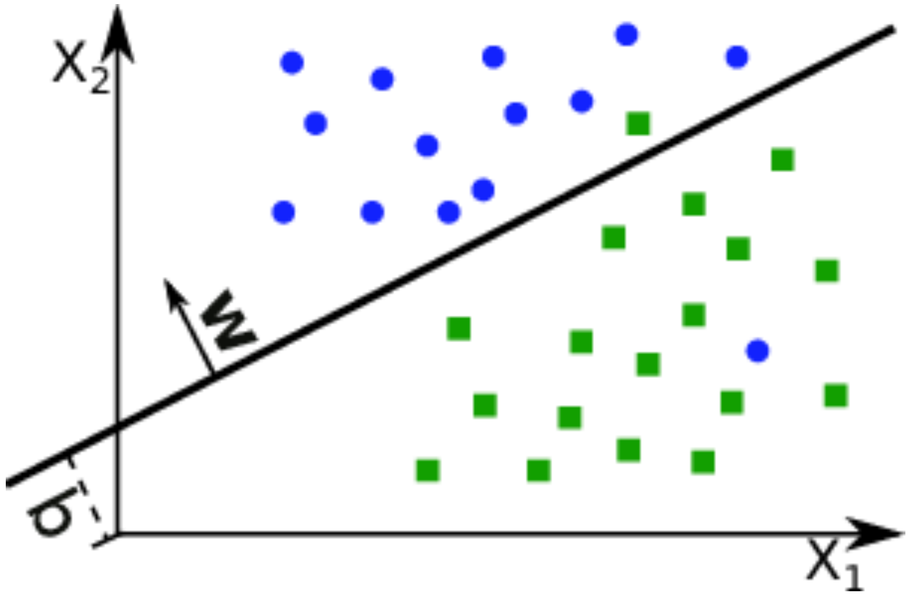
- **Classifier.** Linear model + indicator function

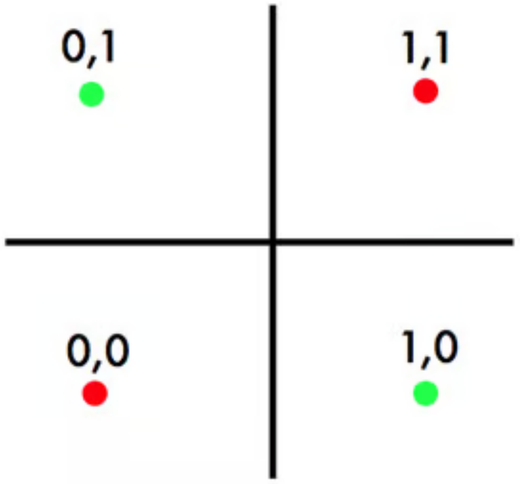
$$f_\theta(\mathbf{x}) = \mathbf{1}\left[\theta^\top \tilde{\mathbf{x}} > 0\right]$$

- **Problem.** Gradient is discrete
  - Solution. Use a surrogate loss function

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

  - Gave birth to some specialized algorithm
- **Limitation.** Cannot express complicated function (e.g., XOR)





Logistic Regression

Logistic regression

- Solve the classification, just like linear regression
  - **Idea.** Do not predict the label directly, but predict the log likelihood ratio (note the direction)

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

  - **Question.** Why don't we simply predict $p(y=1\,|\,\mathbf{x})$?

Logistic regression

- Solve the classification, just like linear regression
  - **Idea.** Do not predict the label directly, but predict the log likelihood ratio (note the direction)

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

    - **Question.** Why don't we simply predict $p(y=1\,|\,\mathbf{x})$?
      - Answer. To utilize the full range; $p(y=1\,|\,\mathbf{x}) \in [0,1]$, but $\theta^{\mathsf{T}}\tilde{\mathbf{x}} \in (-\infty, +\infty)$

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

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

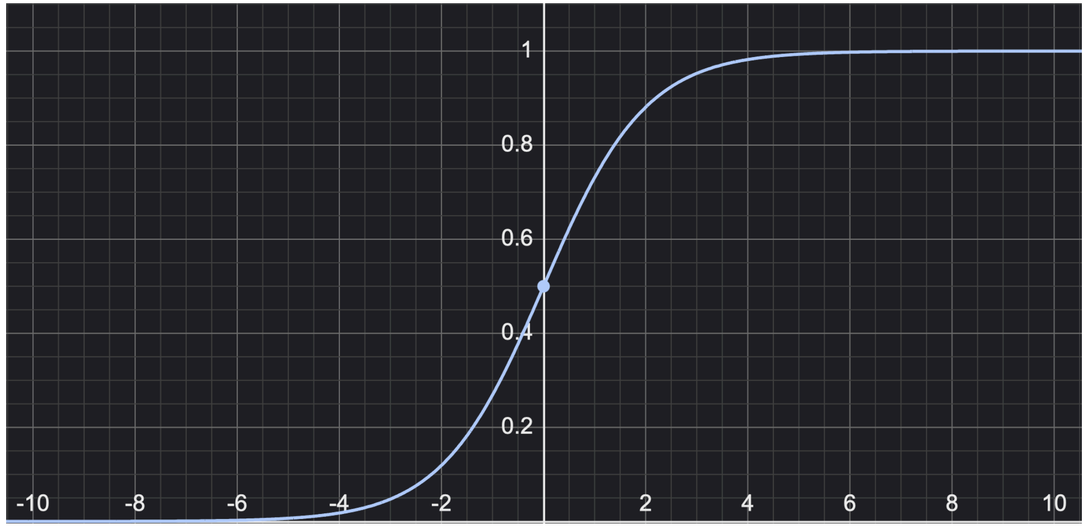
Logistic regression

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

- 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 called the logistic function



Training

- **Training.** Given the data $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, we 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, solve an ERM with:
  - Hypothesis space $\{f_\theta(\mathbf{x}) = \sigma(\theta^\top \tilde{\mathbf{x}})\}$
  - Loss is the cross-entropy $\ell(y, t) = \mathrm{CE}(\mathbf{1}_y, [t, 1-t]) = \log(t)^{-y} + \log(1-t)^{y-1}$

Training

- Equivalently again, solve an ERM with:
  - Hypothesis space $\{f_\theta(\mathbf{x}) = \sigma(\theta^\top \tilde{\mathbf{x}})\}$
  - Loss is the cross-entropy $\ell(y, t) = \mathrm{CE}(\mathbf{1}_y, [t, 1-t]) = \log(t)^{-y} + \log(1-t)^{y-1}$
- More tediously, minimize

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

  - Convex, but no general closed-form solution —> use gradient descent

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

Remarks

- **Computation.** Relatively easy
    - Training. Requires solving GD, but is convex
    - Testing. Dot product, and apply some threshold

Remarks

- **Computation.** Relatively easy
  - Training. Requires solving GD, but is convex
  - Testing. Dot product, and apply some threshold

- **Limitations.** Again, limited expressive power
  - But will give you a working classifier anyways

Support Vector Machine

Some 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.

Some 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

Some 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$(\mathrm{XQK}^\top\mathrm{X}^\top)$, 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 $\mathrm{W} = \mathrm{KQ}^\top$. 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]

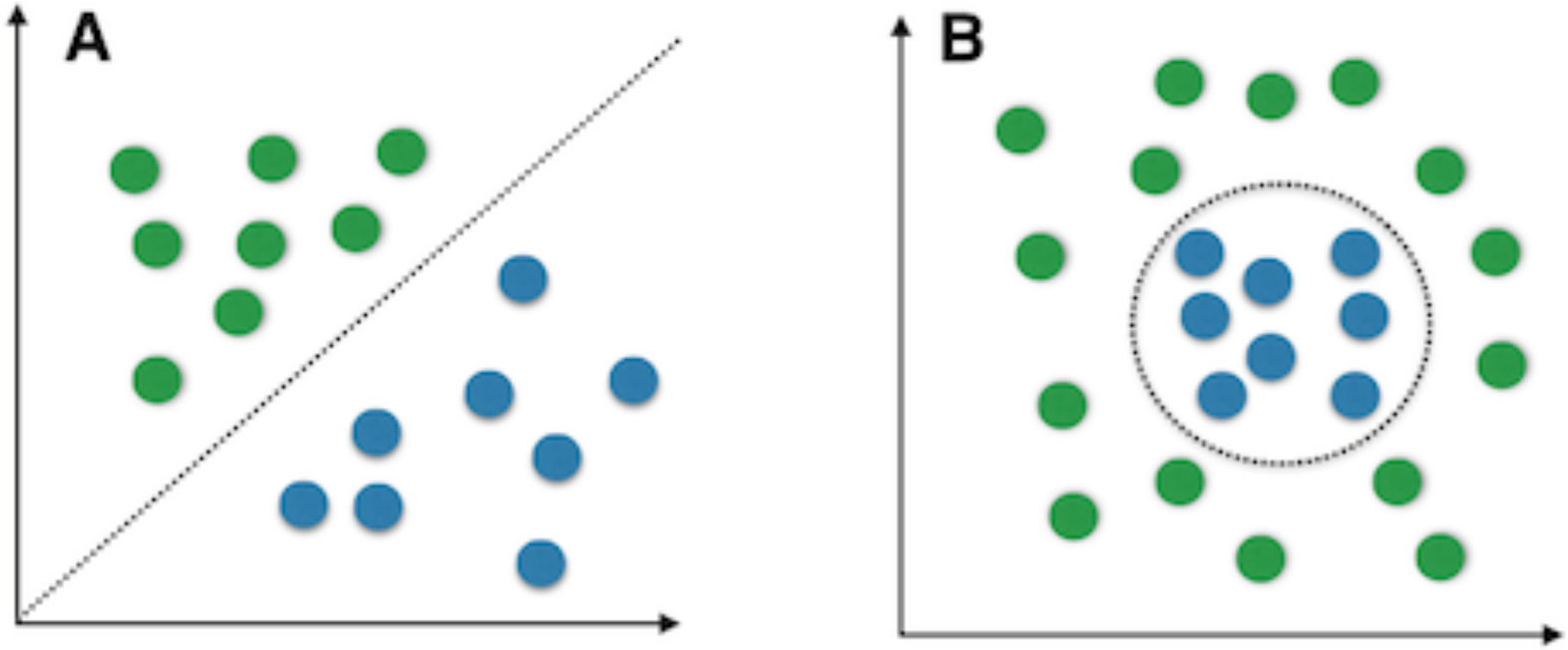
[v1] Thu, 31 Aug 2023 17:57:50 UTC (1,671 KB)

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

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

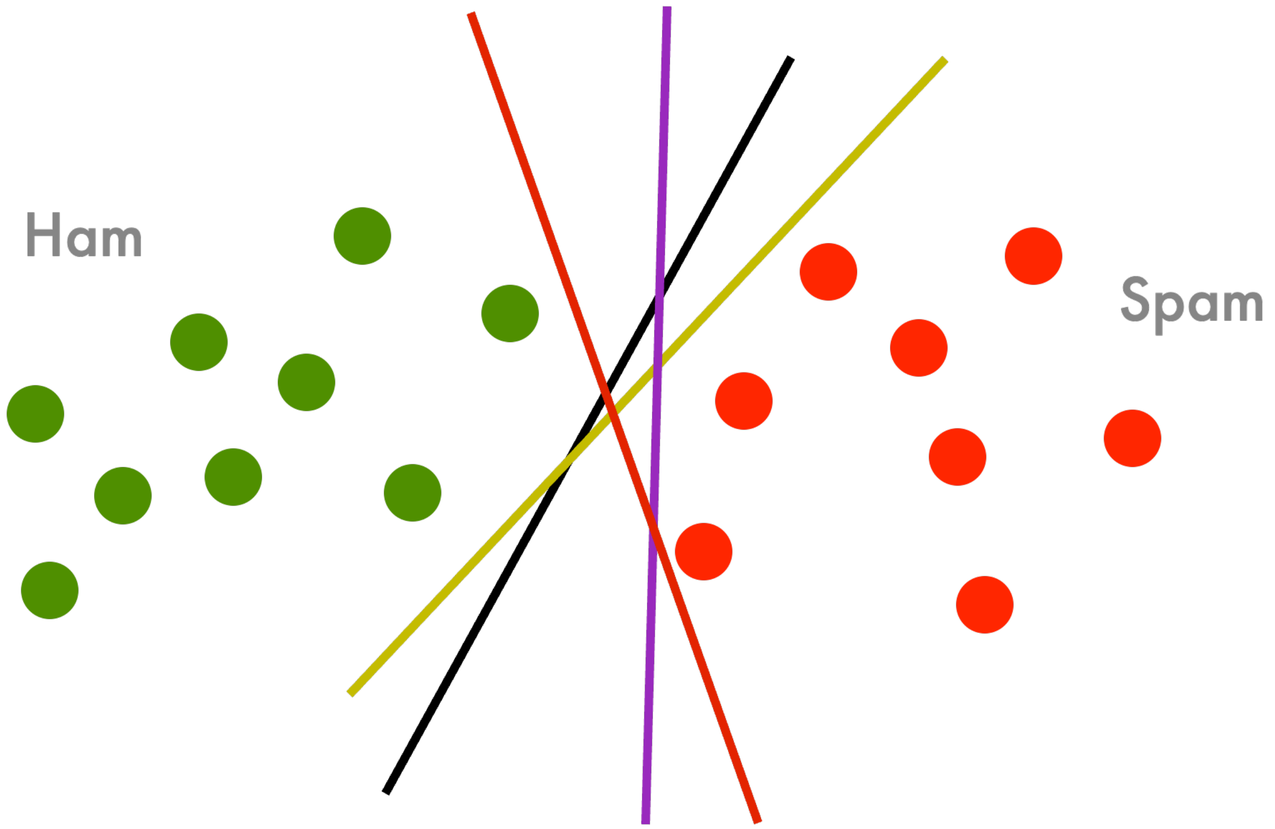
Core philosophy

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



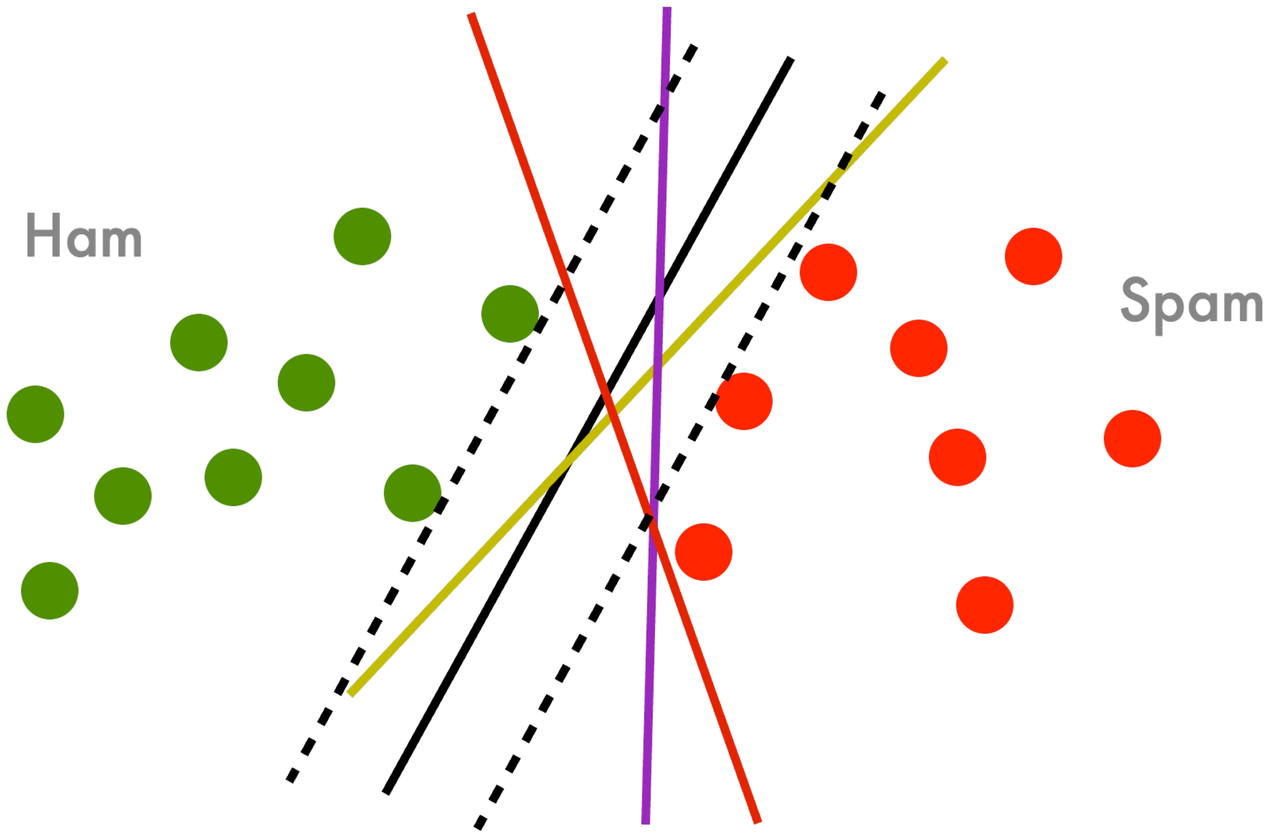
Core philosophy

- Suppose that we have a **linearly separable** data
  - i.e., exists a linear classifier that perfectly classifies the training data.
- Then there could be <u>many correct</u> classifiers…
- **Question.** How should we choose one?



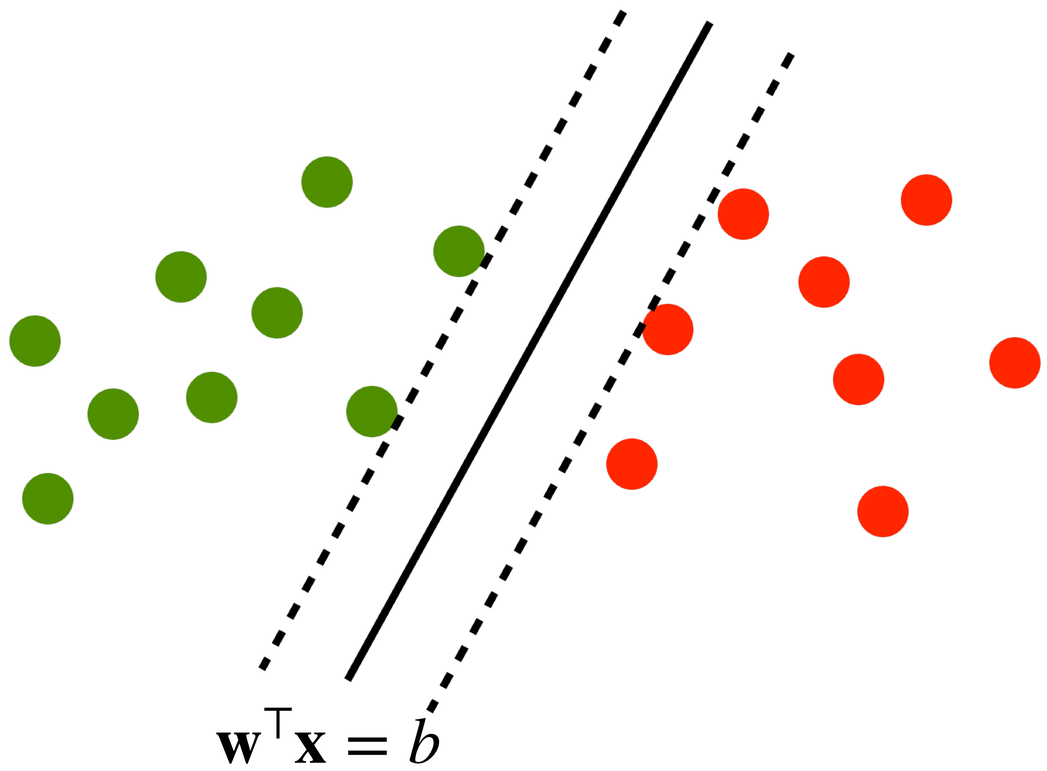
Core philosophy

- Suppose that we have a **linearly separable** data
  - i.e., exists a linear classifier that perfectly classifies the training data.
- Then there could be many correct classifiers...
- **Question.** How should we choose one?
  - Idea. We should choose the maximum-margin classifier!
  - Reason. Robust to noise in test data (there could be any noise in test data)



Large Margin Classifiers

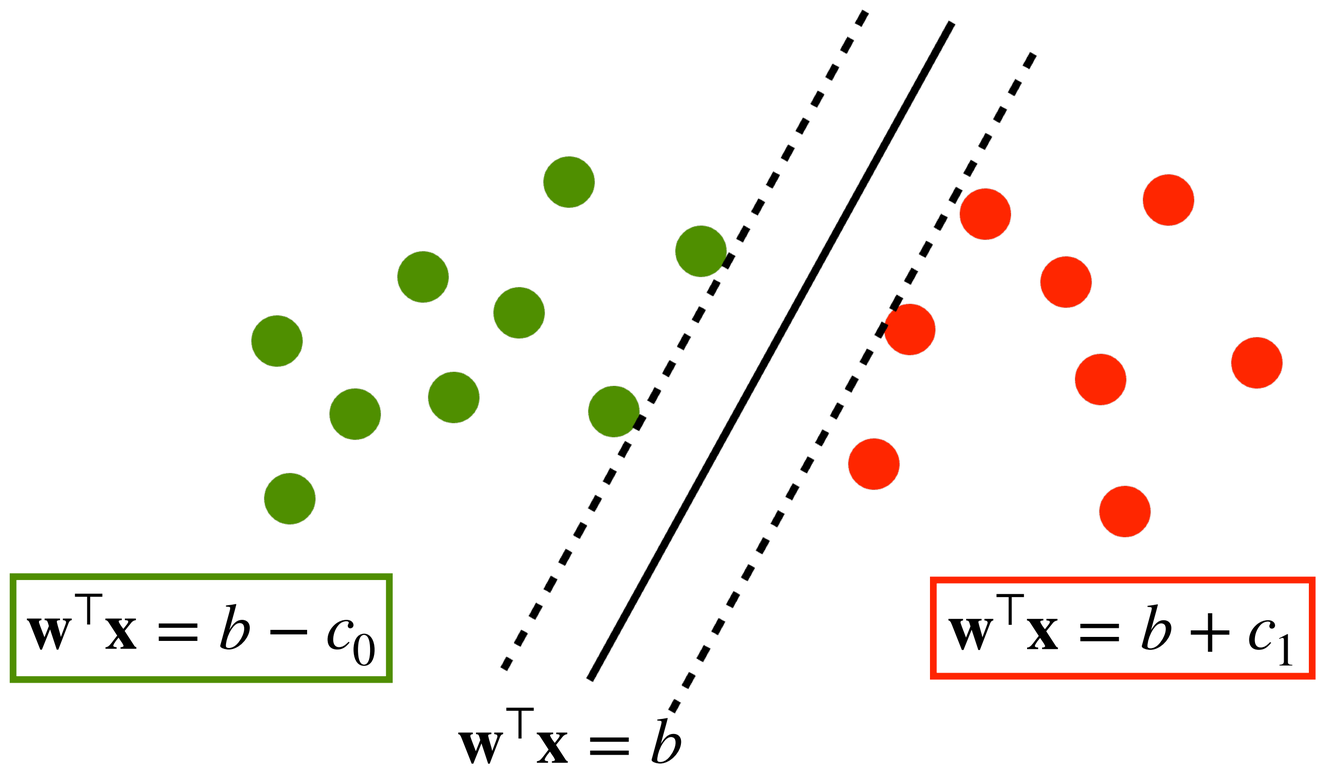
- **Question.** How do we formalize the concept of margin?



Large Margin Classifiers

- **Question.** How do we formalize the concept of margin?

- Idea. Maximum shift that the classifier can withstand

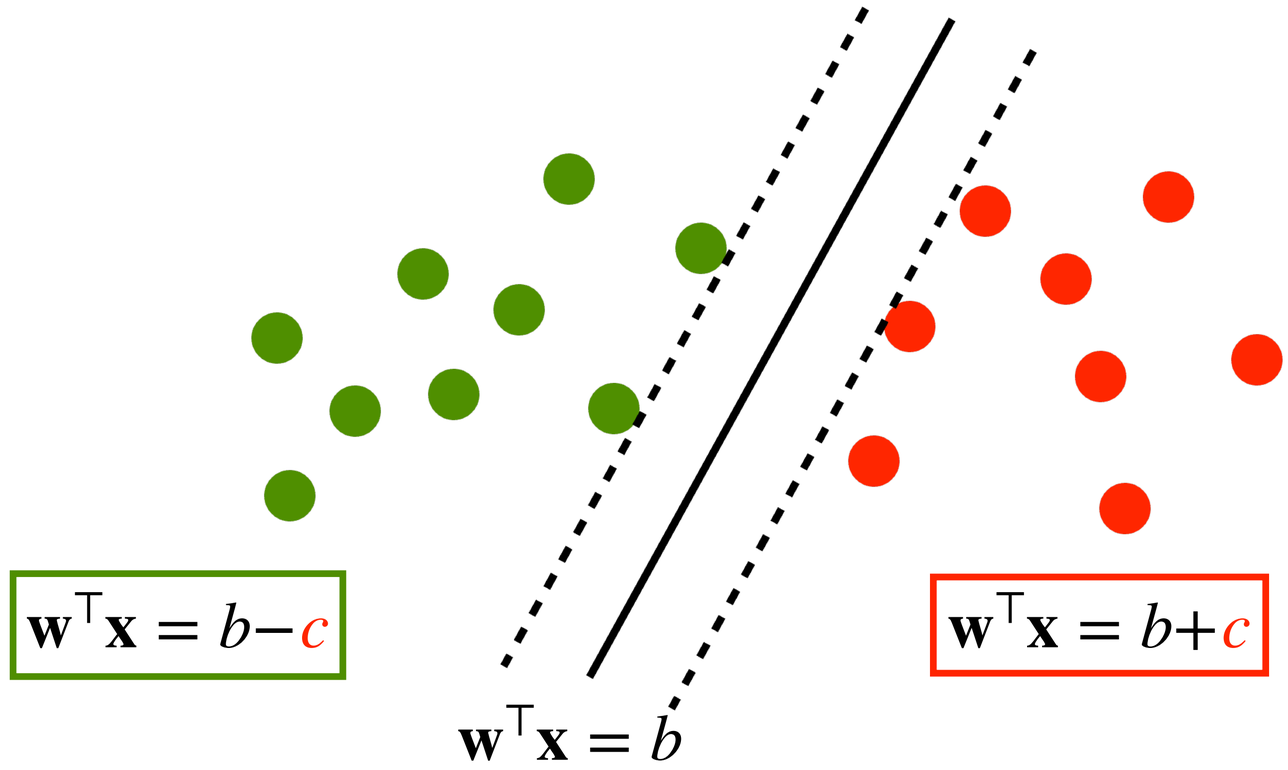


Large Margin Classifiers

- **Question.** How do we formalize the concept of margin?

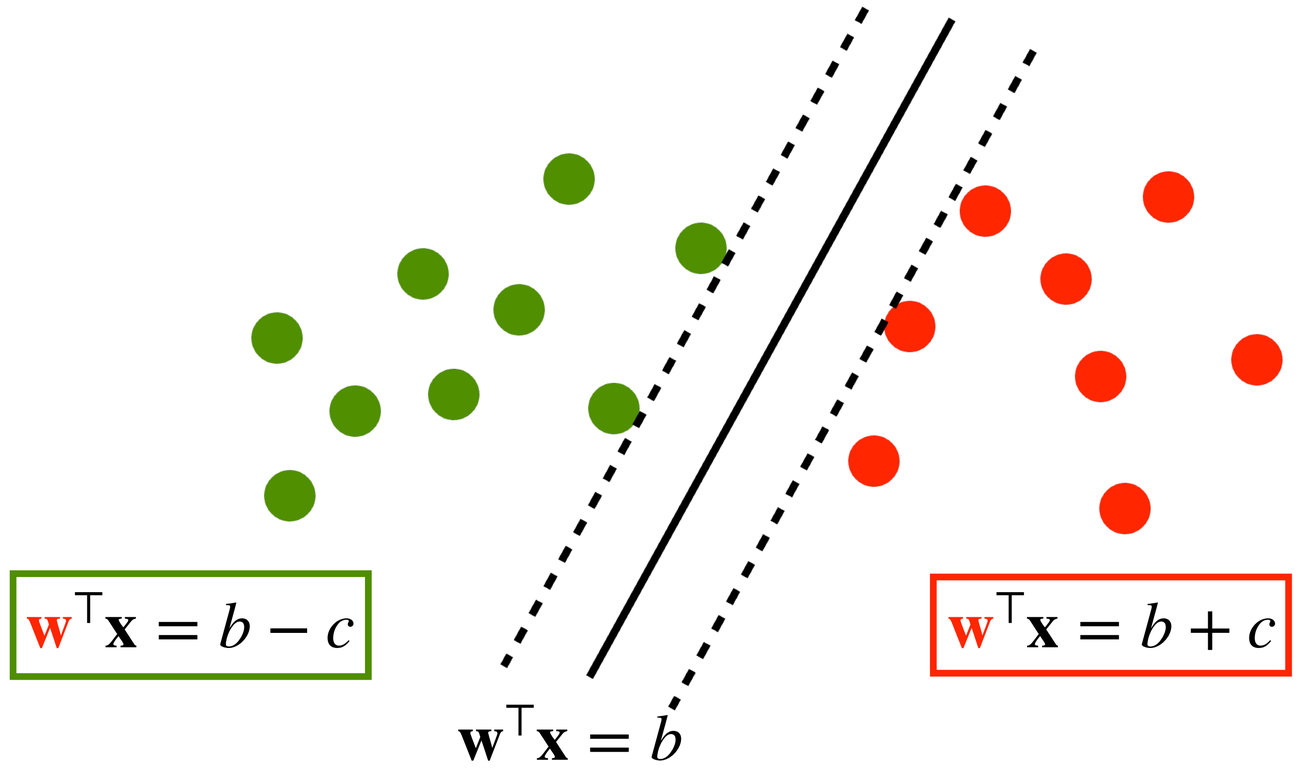
- Idea. Maximum shift that the classifier can withstand

  - We should take the midpoint:



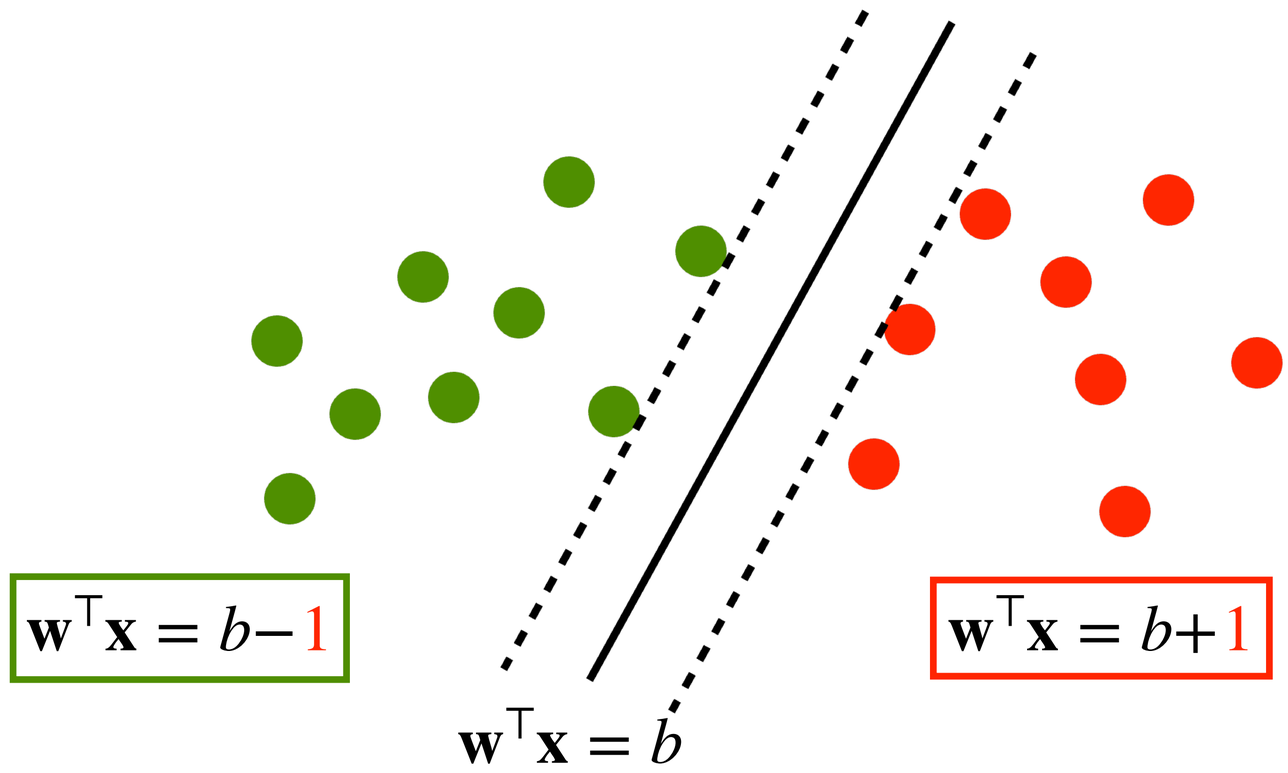
Large Margin Classifiers

- **Question.** How do we formalize the concept of margin?
- Idea. Maximum shift that the classifier can withstand
  - We should take the midpoint
    - We should normalize the size of $\mathbf{w}$
      - Otherwise the size of c can be arbitrarily large



Large Margin Classifiers

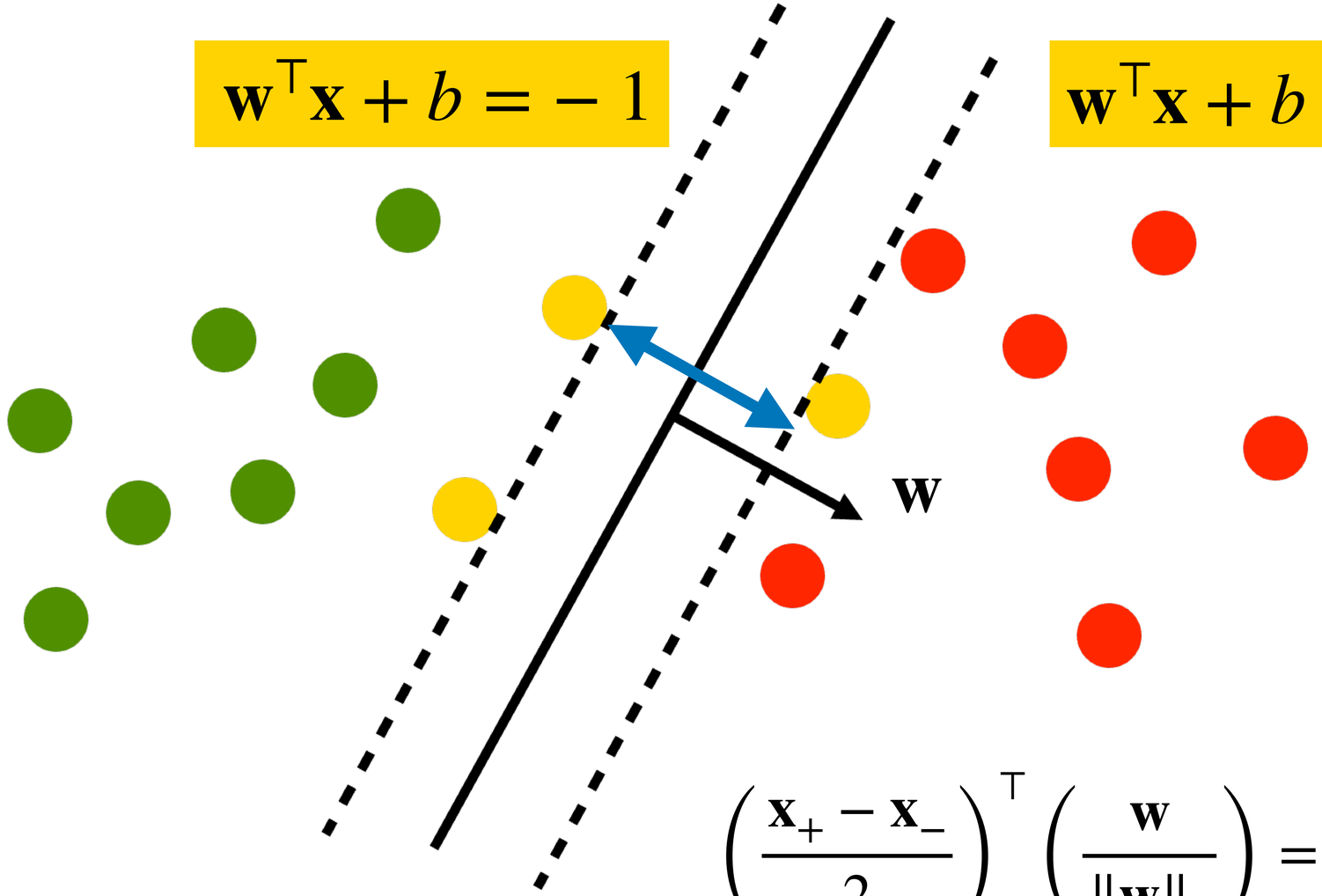
- **Question.** How do we formalize the concept of margin?
- Idea. Maximum shift that the classifier can withstand
  - We should take the midpoint
  - We should normalize the size of $\mathbf{w}$
    - Otherwise the size of c can be arbitrarily large



- Other way around, we can just fix $c = 1$, and look at the size of $1/\|\mathbf{w}\|_2$

Large Margin Classifiers

- **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"

Max Margin Classifiers

- **SVM.** Designed as a maximum-margin classifier
  - We solve the constrained optimization problem

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

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

Max Margin Classifiers

- **SVM.** Designed as a maximum-margin classifier
  - We solve the constrained optimization problem

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

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

- **Question.** How do we solve the constrained optimization?
  - Remark. As convention, we will slightly rephrase as a minimization problem

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

Solving the Optimization: Dual

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

- **Solution.** Consider the Lagrangian dual of the problem (the original problem is called "primal")

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

How much you violated

Solving the Optimization: Dual

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

- **Solution.** Consider the Lagrangian dual of the problem (the original problem is called "primal")

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

- Then, interestingly, we know that the following duality holds:

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

  - Question. Why?

Solving the Optimization: Dual

- **Primal.** $\ell^* = \min_{\mathbf{w},b} \frac{\|\mathbf{w}\|^2}{2}$ subject to $y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1$
- **Dual.** $\ell^* = \min_{\mathbf{w},b} \max_{\alpha \geq 0} \frac{\|\mathbf{w}\|^2}{2} + \sum_{i=1}^{n} \alpha_i \big(1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)\big)$
  - Interpretation. The adversary will gauge the quantity $1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)$

Solving the Optimization: Dual

- Primal. $\ell^* = \min_{\mathbf{w},b} \frac{\|\mathbf{w}\|^2}{2}$ subject to $y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1$
- Dual. $\ell^* = \min_{\mathbf{w},b} \max_{\alpha \geq 0} \frac{\|\mathbf{w}\|^2}{2} + \sum_{i=1}^{n} \alpha_i \big(1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)\big)$
  - Interpretation. The adversary will gauge the quantity $1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)$
    - If > 0: $\cdots$ $\alpha_i \to \infty$

infeasible for primal, ∞ for dual

Solving the Optimization: Dual

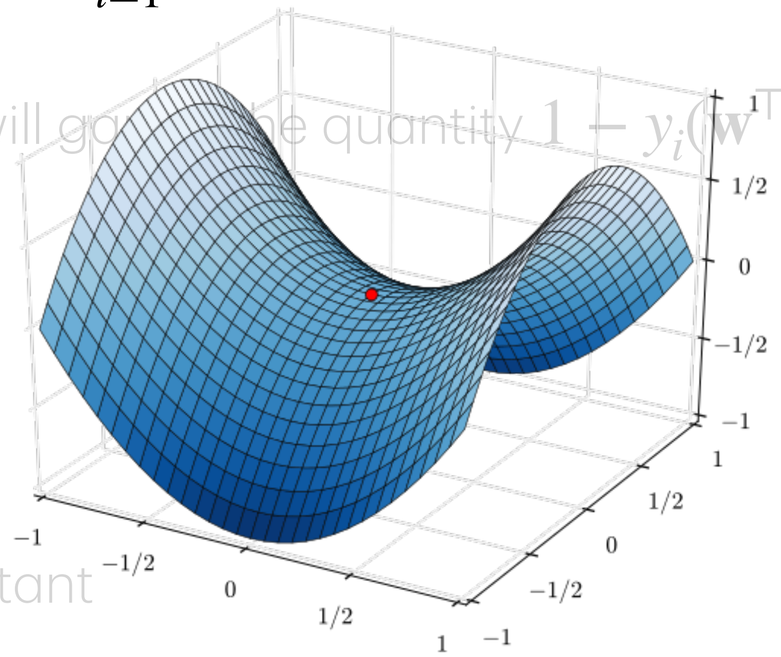
- **Primal.** $\ell^* = \min_{\mathbf{w},b} \frac{\|\mathbf{w}\|^2}{2}$ subject to $y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1$
- **Dual.** $\ell^* = \min_{\mathbf{w},b} \max_{\alpha \geq 0} \frac{\|\mathbf{w}\|^2}{2} + \sum_{i=1}^{n} \alpha_i \big(1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)\big)$
  - Interpretation. The adversary will gauge the quantity $1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)$
    - If > 0: $\cdots$ $\alpha_i \to \infty$ infeasible for primal, ∞ for dual
    - If < 0: $\cdots$ $\alpha_i = 0$ primal = dual

Solving the Optimization: Dual

- **Primal.** $\ell^* = \min_{\mathbf{w},b} \frac{\|\mathbf{w}\|^2}{2}$ subject to $y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1$
- **Dual.** $\ell^* = \min_{\mathbf{w},b} \max_{\alpha \geq 0} \frac{\|\mathbf{w}\|^2}{2} + \sum_{i=1}^{n} \alpha_i \big(1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)\big)$
  - Interpretation. The adversary will gauge the quantity $1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)$
    - If > 0: ⋯ $\alpha_i \to \infty$ infeasible for primal, ∞ for dual
    - If < 0: ⋯ $\alpha_i = 0$ primal = dual
    - If $= 0$ ⋯ any constant primal = dual (margin points!)

Solving the Optimization: Dual

- **Primal.** $\ell^* = \min_{\mathbf{w},b} \frac{\|\mathbf{w}\|^2}{2}$ subject to $y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1$
- **Dual.** $\ell^* = \min_{\mathbf{w},b} \max_{\alpha \geq 0} \frac{\|\mathbf{w}\|^2}{2} + \sum_{i=1}^{n} \alpha_i \big(1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)\big)$
  - Interpretation. The adversary will go for the quantity $1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)$
    - If > 0: $\cdots$ $\alpha_i \to \infty$ infeasible for primal, ∞ for dual
    - If < 0: $\cdots$ $\alpha_i = 0$ primal = dual
    - If $= 0$ $\cdots$ any constant primal = dual (margin points!)



- Thus, what remains is solving this minimax problem, finding the **saddle point**

Saddle point

- Of course, we look for the **critical point;** writing down the gradient, we get

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

Saddle point

- Of course, we look for the **critical point;** writing down the gradient, we get

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

- Setting them equal to zero, we get

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

Saddle point

- Of course, we look for the **critical point;** writing down the gradient, we get

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

- Setting them equal to zero, we get

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

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

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

Saddle point as a quadratic program

- Summing up, our optimization problem becomes

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

Saddle point as a quadratic program

- Summing up, our optimization problem becomes

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

- Slightly rephrased as a **quadratic program** over a convex polytope

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

  - The solution @ critical point or the extreme points
  - Use quadratic program solvers to get the optimal α^*

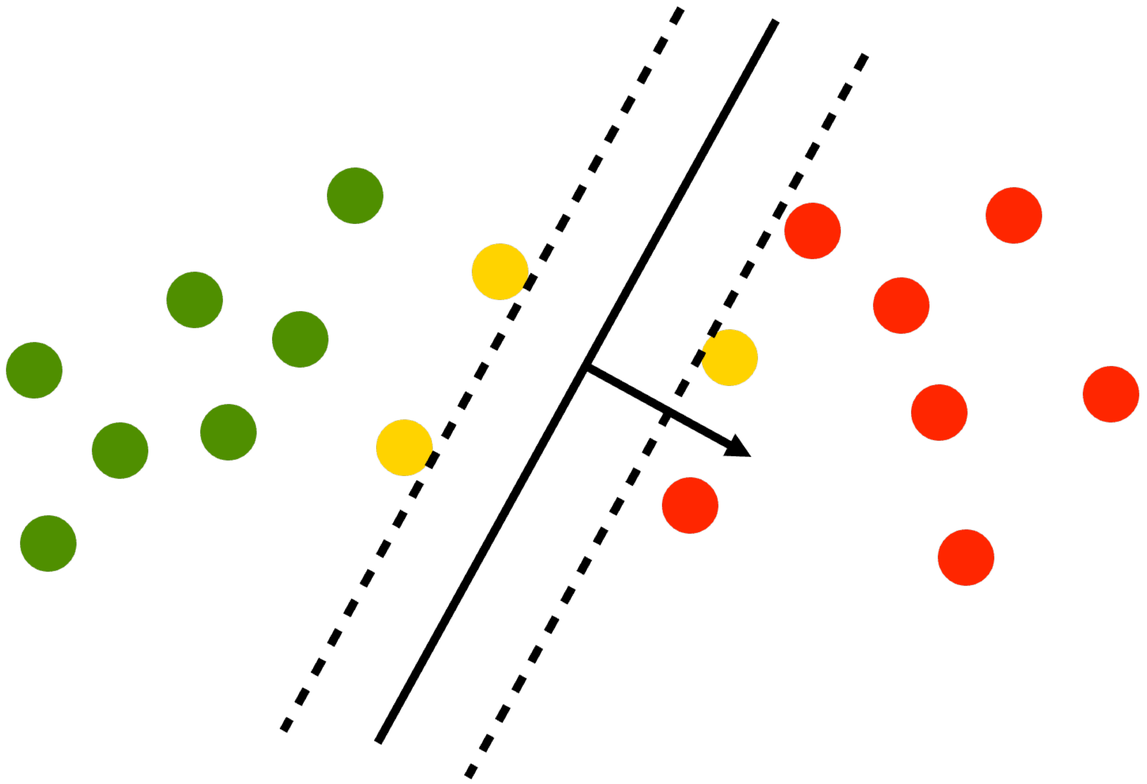
Solving for Bias

- Having computed the α^*, our optimal weights become:

Nonzero only for margin data (support vectors)

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

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



Solving for Bias

- Having computed the α^*, our optimal weights become:

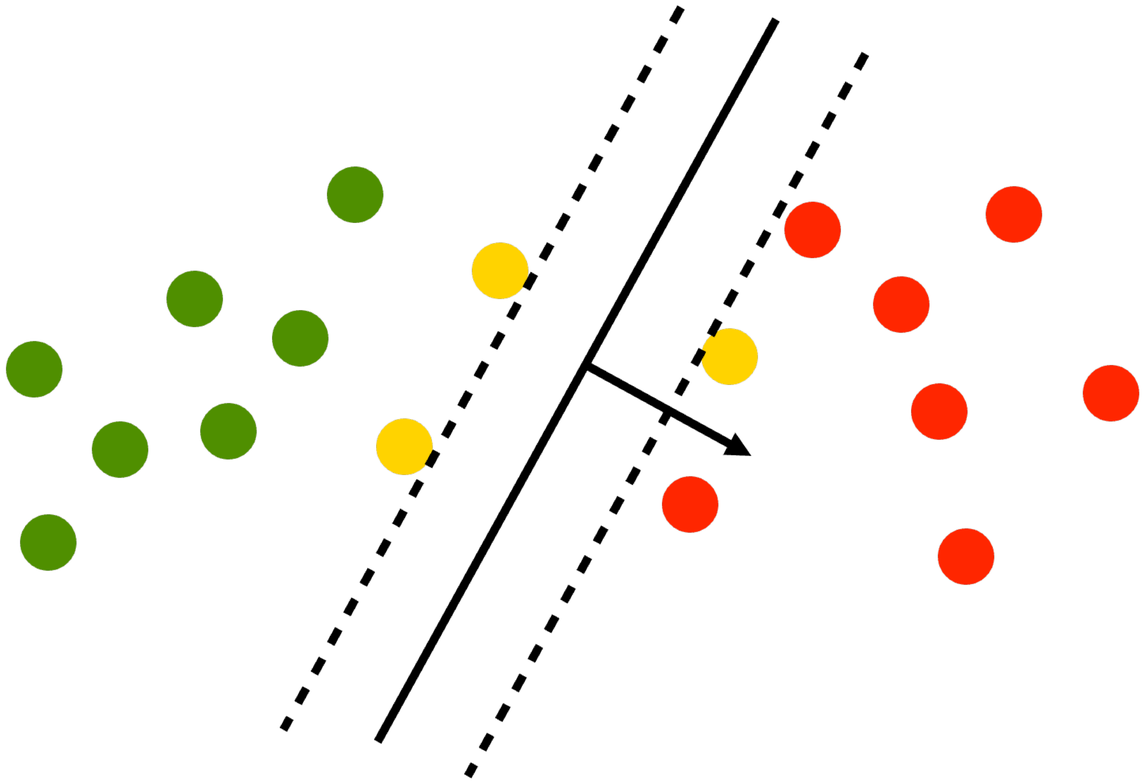
$$\mathbf{w}^* = \sum_{i=1}^{n} \alpha_i^* y_i \mathbf{x}_i$$

Nonzero only for margin data (support vectors)

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

  - Answer. Plug in any support vector to

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



Wrapping up

- Looked at a clever way to use linear classifier to solve classification
    - Idea. Maximize the margin
        - Putting in some extra condition to generalize better to test data (similar to what we did in k-NN, when deciding the hyperparameter)

Wrapping up

- Looked at a clever way to use linear classifier to solve classification
  - Idea. Maximize the margin
    - Putting in some extra condition to generalize better to test data (similar to what we did in k-NN, when deciding the hyperparameter)

- **Limitation.**
  - Messed up when outlier exists (yet) —> Soft SVM
  - More generally, cannot handle nonlinear data —> Kernel SVM

Cheers