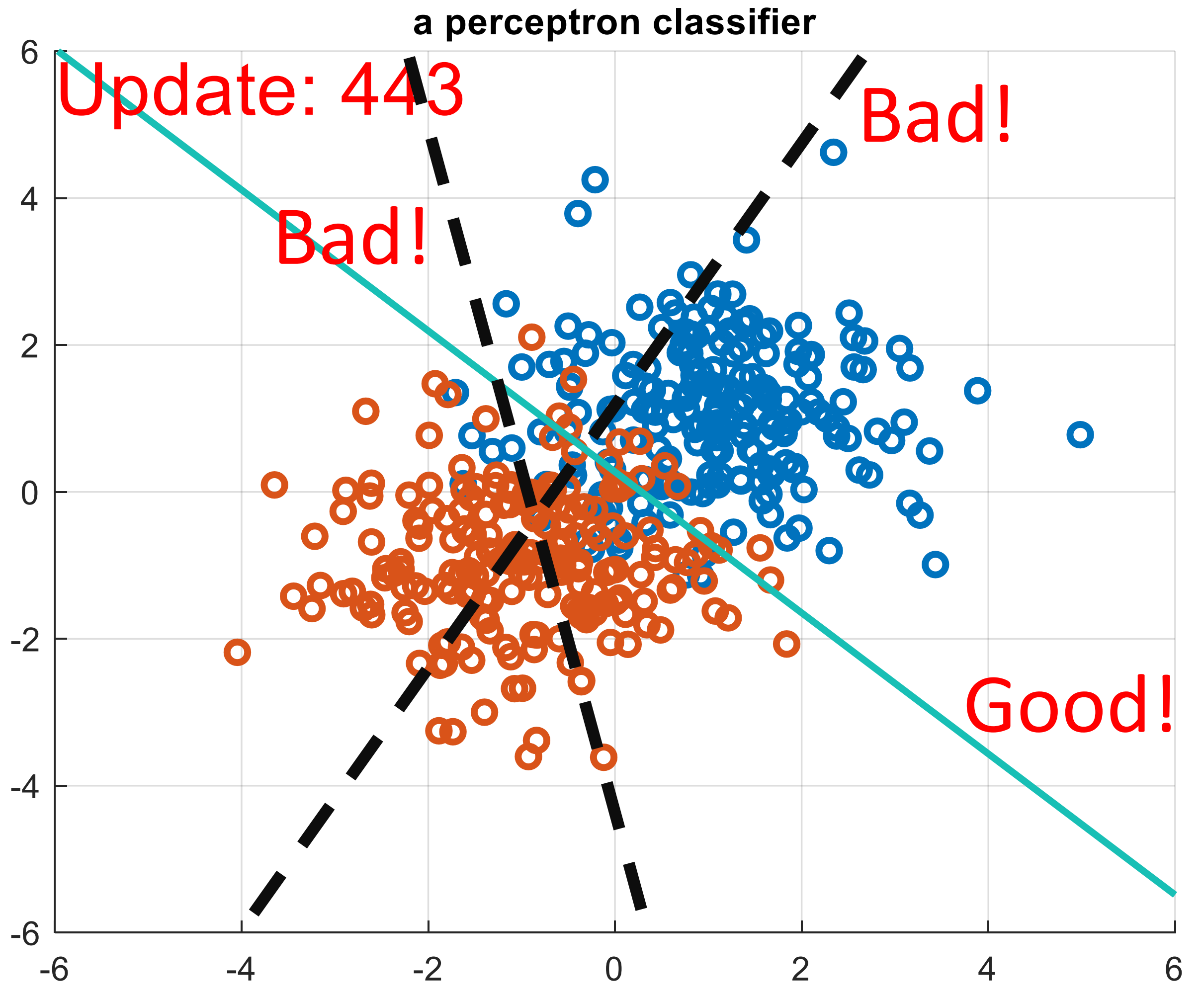
Support Vector Machines

Song Liu (song.liu@bristol.ac.uk)

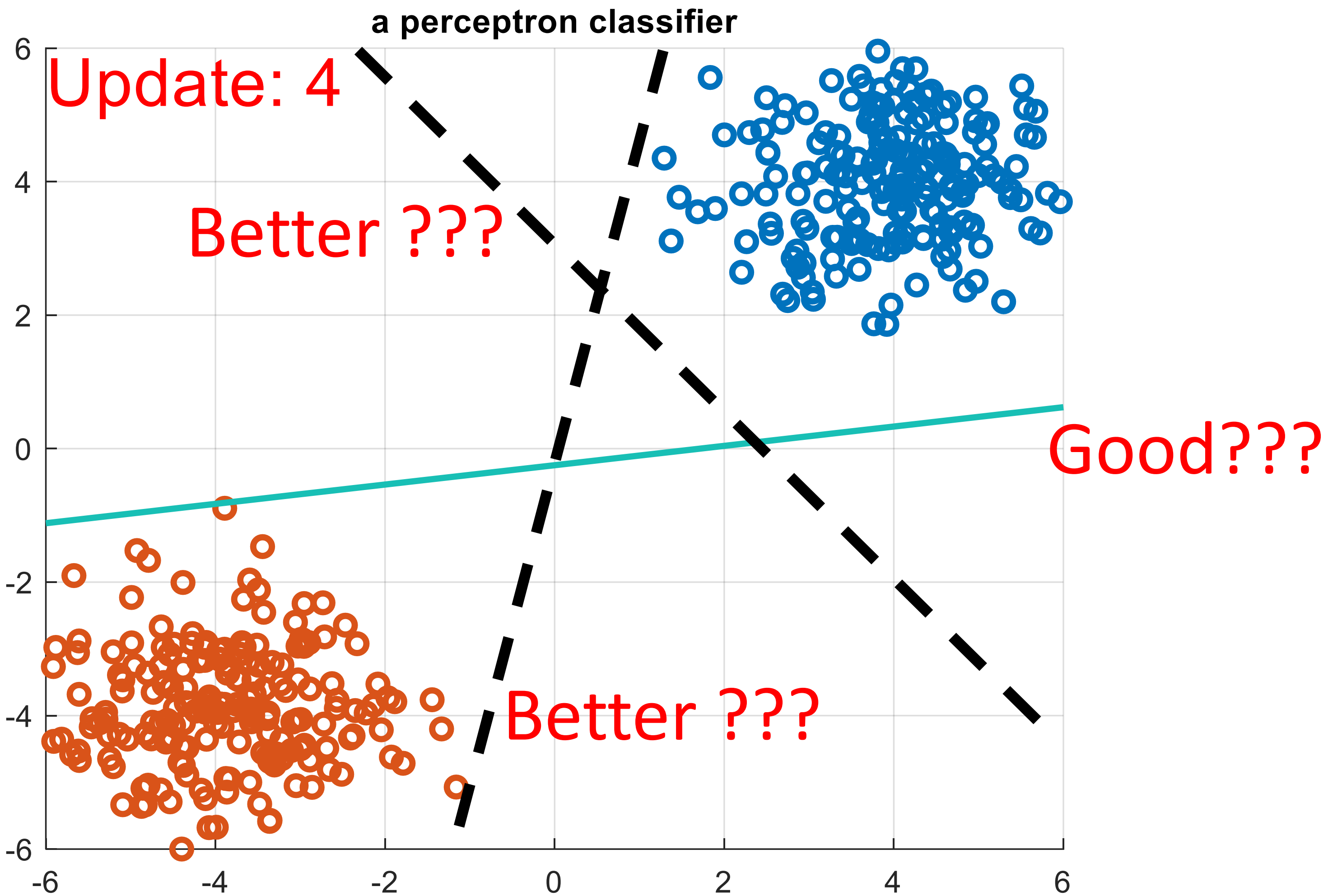
Outlines

- Problem Support Vector Machine (SVM) tries to solve.
- Objective of SVM
- **Dual objective** of SVM
- Limitations of SVM

Perceptron Classifier



Perceptron Classifier



More than one “good” solutions!!

What is
The “Optimal” Decision Boundary in binary classification?

Vladimir Vapnik and Alexey Chervonenkis

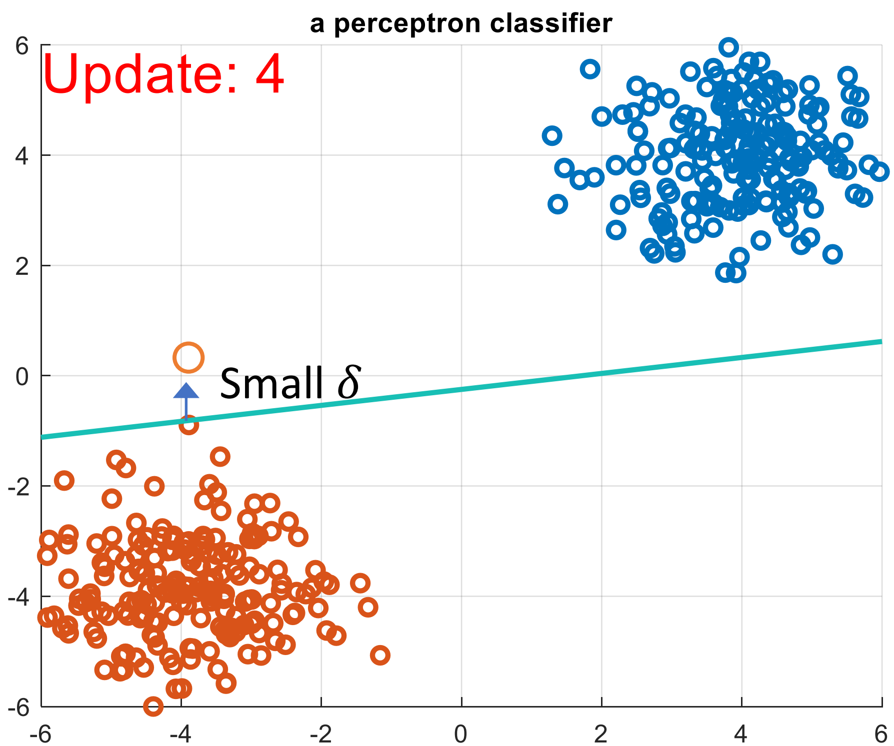
Vladimir Vapnik

Alexey Chervonenkis

Contributions: Statistical Learning Theory, Support Vector Machines

The Error Margin

- **Generalization Principle:** Optimal decision boundary should minimize the error on **unseen datasets rather than training data.**

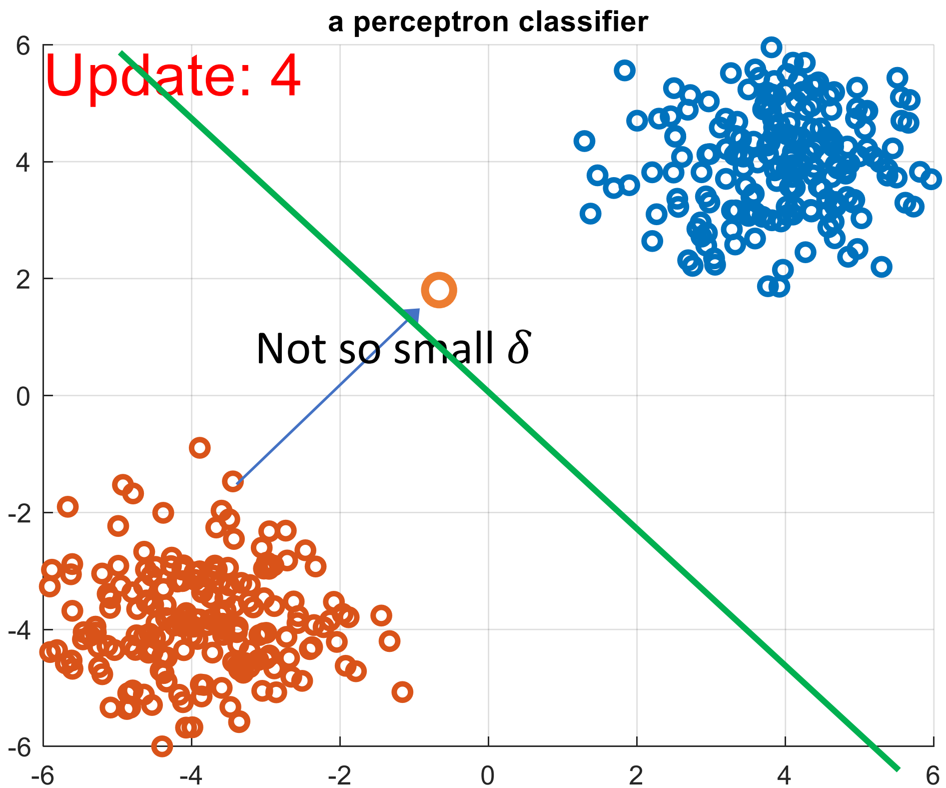


This is not a good decision boundary as a **small change** added to our data point would lead to **misclassification**.

Our decision boundary has a thin "**error margin**".

The Error Margin

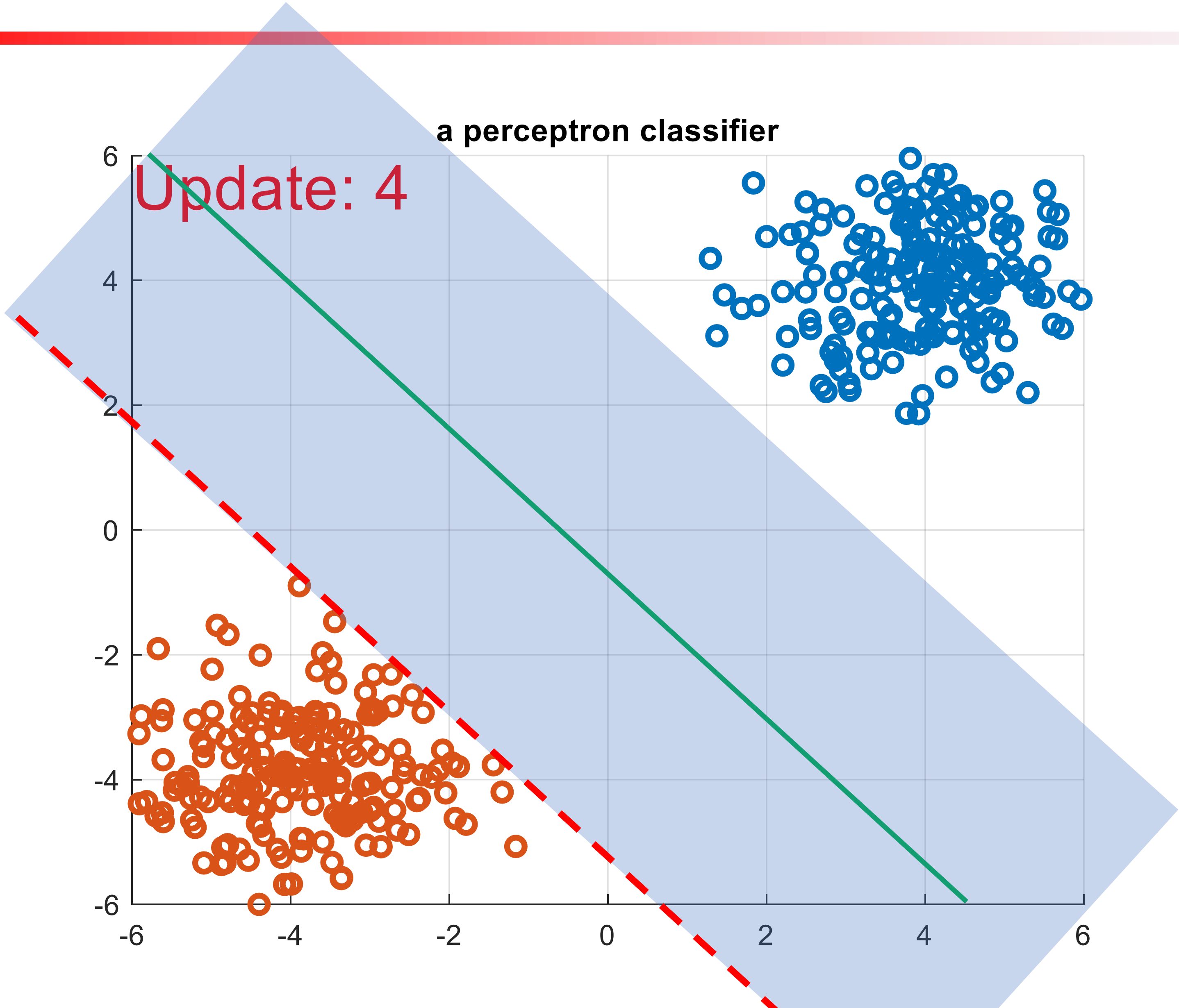
- **Thin margin** is bad for generalization as some random unseen data points may easily "drift" to the other side of the decision boundary.



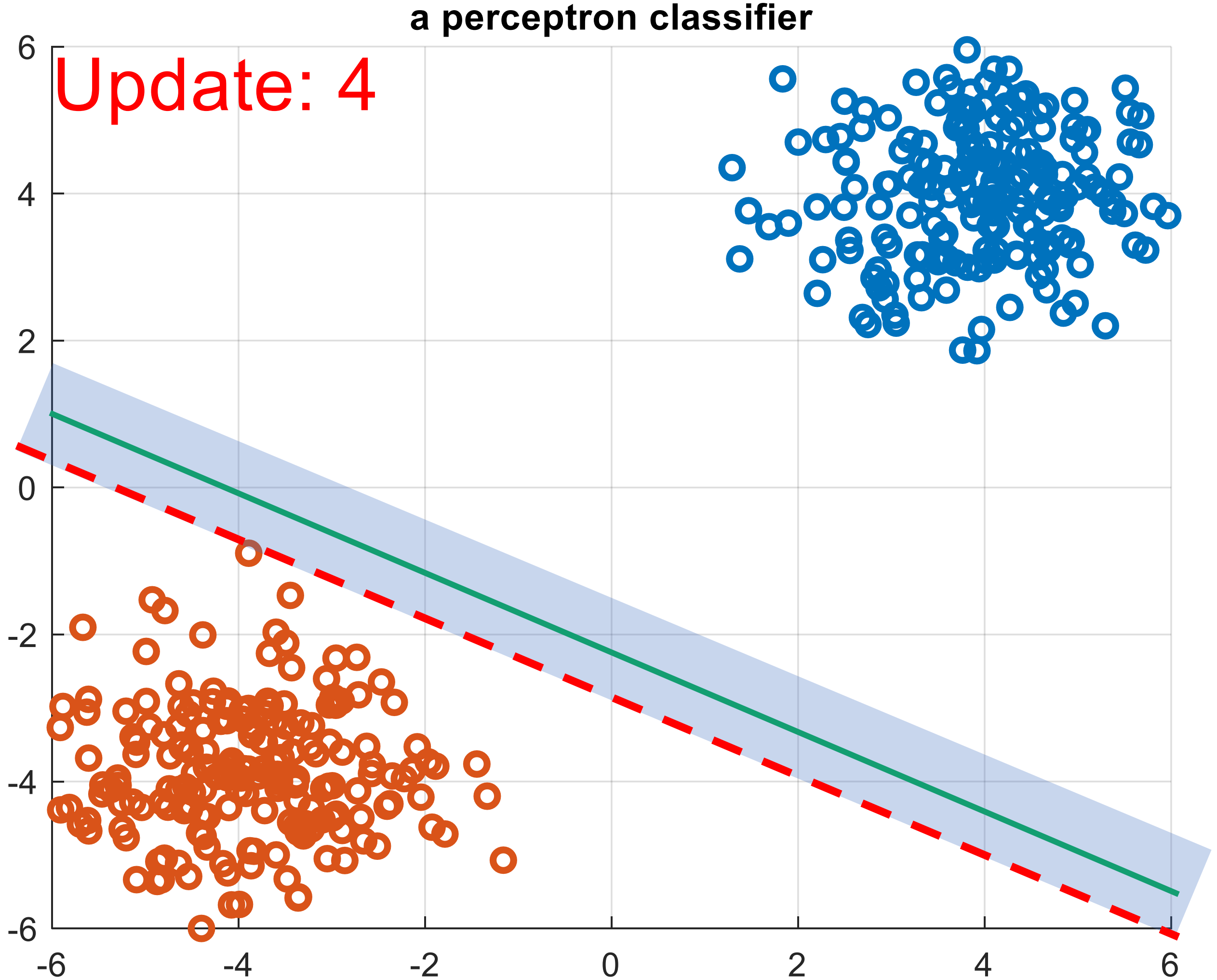
This is a good decision boundary as small perturbations of our data points unlikely lead to **misclassification**.

Our decision boundary has a thick "**error margin**".

Thick Error Margin



Thin Error Margin



What is the “Optimal” Decision Boundary?

- If decision boundary is characterized by $f(\boldsymbol{x};\boldsymbol{w}) = 0$, we have the following criteria:

- 1. $\forall i, y_i = +, f(\boldsymbol{x}_i;\boldsymbol{w}) \geq 0$
- 2. $\forall i, y_i = -, f(\boldsymbol{x}_i;\boldsymbol{w}) \leq 0$

- 3. The error margin of $f(\boldsymbol{x};\boldsymbol{w})$ should be as **THICK** as possible!

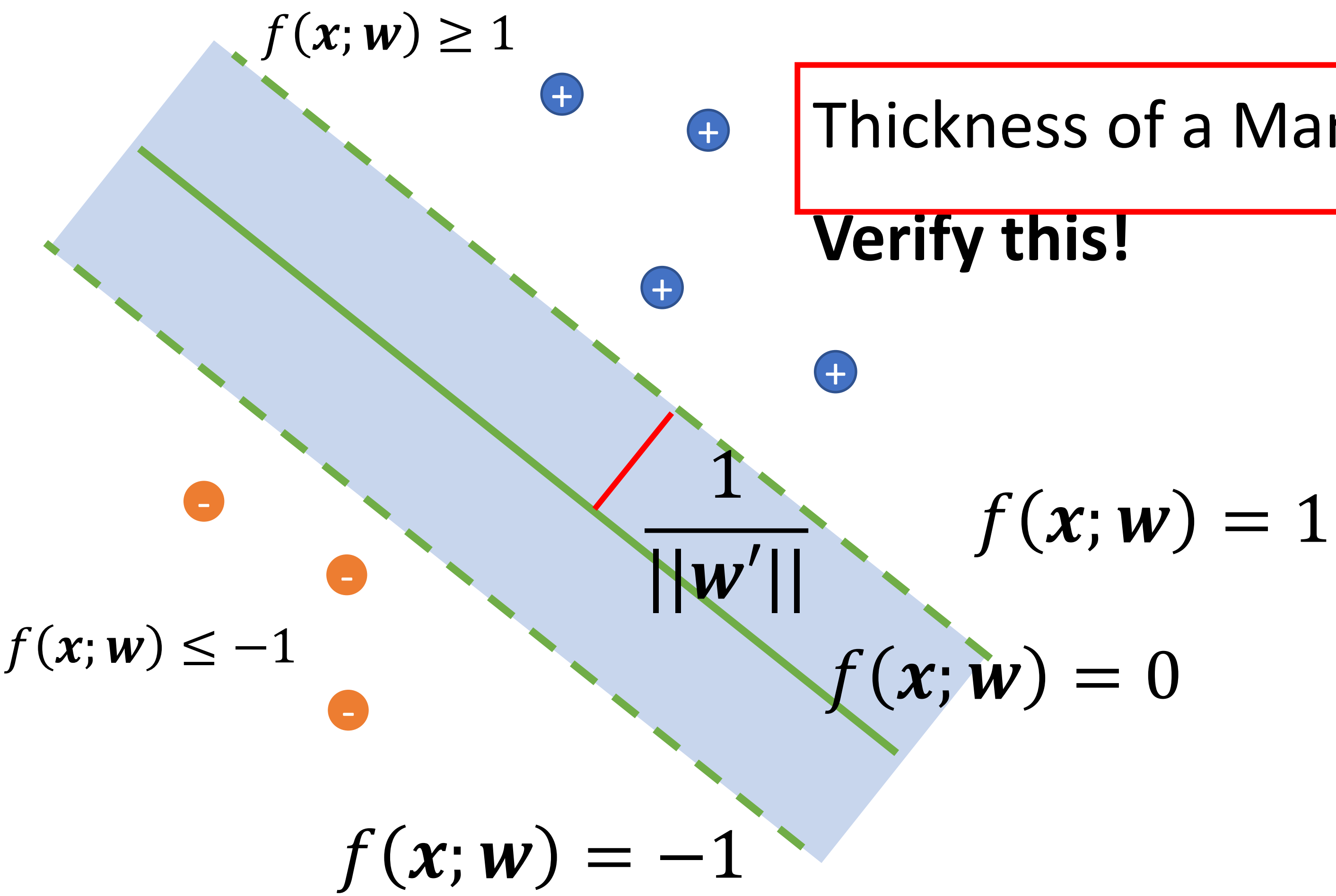
- How do you translate the above criteria into optimization?

Margin of Linear Model

- Suppose $f(\boldsymbol{x};\boldsymbol{w}) \coloneqq \langle \boldsymbol{w}', \boldsymbol{x} \rangle + w_0$

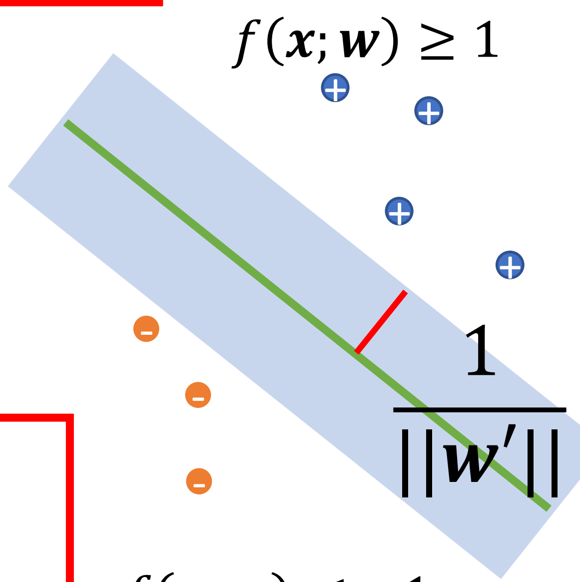
Thickness of a Margin: $\dfrac{1}{||\boldsymbol{w}'||}$!

Verify this!



Maximal Margin Classifier

- Maximize the Width of a Margin
- Keep datapoints on the correct side of the margin.



- $\Leftrightarrow$
- Maximize $\frac{1}{||\boldsymbol{w}'||}$
- and maintain $\forall_i, y_i = +, f(\boldsymbol{x}_i;\boldsymbol{w}) \geq 1,$
  $\forall_i, y_i = -, f(\boldsymbol{x}_i;\boldsymbol{w}) \leq -1$

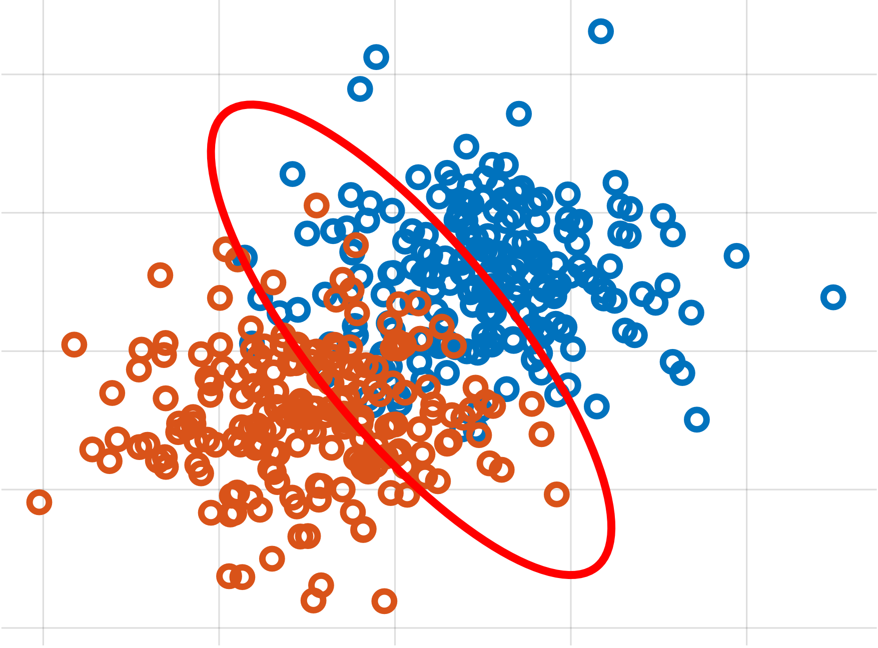
Maximal Margin Classifier

- Maximize $\frac{1}{||\boldsymbol{w'}||}$
- and maintain $\forall_i, y_i = +, f(\boldsymbol{x}_i; \boldsymbol{w}) \geq 1,$
  $\forall_i, y_i = -, f(\boldsymbol{x}_i; \boldsymbol{w}) \leq -1$
- $\Leftrightarrow$
- Minimize $||\boldsymbol{w}'||^2$
- Subject to $\forall_i, y_i f(\boldsymbol{x}_i; \boldsymbol{w}) \geq 1,$

- **This is a constrained minimization!**
- **Unlike LS and Logistic regression, which are both unconstrained minimizations.**

Soft-margin Classifiers

- In many cases, the dataset is not separable.



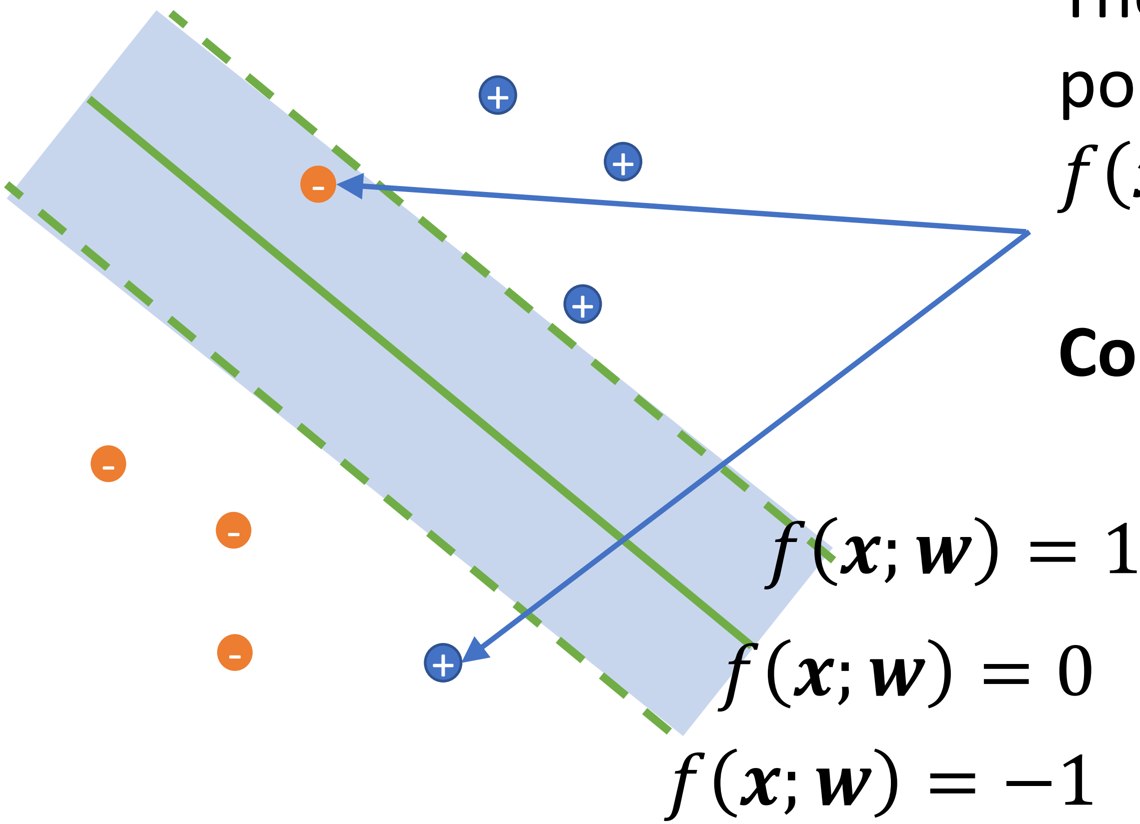
An error margin cannot be constructed due to the overlapping-ness of two classes!

Soft-margin Classifiers

- We allow our f make some errors!

These misclassified points will have $f(\boldsymbol{x};\boldsymbol{w})y \leq 1$!

Constraint not satisfied!



Soft-Margin Classifier

- Minimize $_{w,\epsilon}$ $||\boldsymbol{w}'||^2 + \sum_i \epsilon_i$
- Subject to $\forall_i, y_i f(\boldsymbol{x}_i; \boldsymbol{w}) + \epsilon_i \geq 1, \epsilon_i \geq 0$

- For each $\boldsymbol{x}_i$, we hope $y_i f(\boldsymbol{x}_i; \boldsymbol{w})$ can be at right side of the margin after some small positive “compensation” ϵ_i.

- At the same time, we want such “compensation” is as small as possible, i.e., the classifier makes as few mistakes as possible.

- The solution for ϵ is sparse. Why?

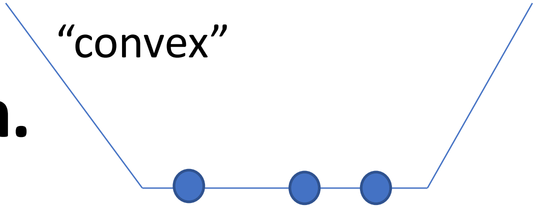
Soft-Margin Classifier

- Formally, the soft-margin classifier

- $\min_{\boldsymbol{w},\boldsymbol{\epsilon}} ||\boldsymbol{w}'||^2 + \sum_i \epsilon_i$
- Subject to $\forall_i, y_i(\langle \boldsymbol{w}', \boldsymbol{x}_i \rangle + w_0) + \epsilon_i \geq 1, \epsilon_i \geq 0$

- It turns out
- **Soft-Margin Classifier** is a **convex** minimization problem.

- **Every local minimum is a global minimum.**



The Lagrangian Dual

- Solving constrained problem can be rather complicated.
- **Lagrangian Dual**: a technique transforms constrained problem into a less constrained problem.

- For a constrained problem,
- $\min_{\theta} f(\theta)$ subject to $g_i(\theta) \leq 0, \forall i$
- We can construct a **Lagrangian** $l(\boldsymbol{\lambda})$:
- $l(\boldsymbol{\lambda}) \coloneqq \min_{\theta} f(\theta) + \sum_i \lambda_i \, g_i(\theta)$,
- $\lambda_i \geq 0$ are called **Lagrangian multipliers**.
- **PRML Appendix E.**

The Lagrangian Dual

- Under regularity conditions*, maximizing $l(\boldsymbol{\lambda})$ w.r.t. $\boldsymbol{\lambda}$ would allow us to recover the optimal solutions in the original constrained minimization problem.

To maximize $l(\boldsymbol{\lambda})$, do the following 4 steps:

- 1. Write down $l(\boldsymbol{\lambda})$ for soft-margin classifier:
- $l(\boldsymbol{\lambda}) :=$

$$\min_{\boldsymbol{w},\boldsymbol{\epsilon}} ||\boldsymbol{w}'||^2 + \sum_i \epsilon_i - \lambda_i[y_i(\langle \boldsymbol{w}', \boldsymbol{x}_i\rangle + w_0) + \epsilon_i - 1] - \lambda_i' \epsilon_i$$

- 2. Derive optimality condition w.r.t. $\boldsymbol{w}$ and $\boldsymbol{\epsilon}$:
- $\boldsymbol{w}' = \frac{\sum_{i=1} \lambda_i y_i \boldsymbol{x}_i}{2}, \sum_{i=1} \lambda_i y_i = 0, \lambda_i + \lambda_i' = 1,$

Verify this!

The Lagrangian Dual

- Using optimality conditions:
- $\boldsymbol{w}' = \frac{\sum_{i=1} \lambda_i y_i \boldsymbol{x}_i}{2}, \lambda_i + \lambda_i' = 1, \sum_{i=1} \lambda_i y_i = 0$

- 3. Rewrite $l(\lambda) = -\frac{\tilde{\boldsymbol{\lambda}}^\top \boldsymbol{X}^\top \boldsymbol{X} \tilde{\boldsymbol{\lambda}}}{4} + \langle \boldsymbol{\lambda}, \mathbf{1} \rangle,$ Verify it!

  $\boldsymbol{X} = [\boldsymbol{x}_1 \ldots \boldsymbol{x}_n] \in R^{d \times n}, \tilde{\boldsymbol{\lambda}} := [\lambda_1 \cdot y_1 \ldots \lambda_n \cdot y_n]$

- 4. Maximize $l(\lambda)$ w.r.t. $\boldsymbol{\lambda}$ under constraints:
  - $0 \leq \lambda_i \leq 1$
  - $\sum_{i=1} \lambda_i y_i = 0$

Needed to make sure the optimality of the **original problem**

Soft-margin Classifier (Dual)

- $\max_{\boldsymbol{\lambda}} - \frac{\tilde{\boldsymbol{\lambda}}^{\top} \boldsymbol{X}^{\top} \boldsymbol{X} \tilde{\boldsymbol{\lambda}}}{4} + \langle \boldsymbol{\lambda}, \mathbf{1} \rangle$
- Subject to
- $0 \leq \lambda_i \leq 1, \sum_{i=1} \lambda_i y_i = 0$

- Recover $\hat{\boldsymbol{w}}' := \frac{\sum_{i=1} \widehat{\lambda}_i y_i \boldsymbol{x}_i}{\mathbf{2}}$ using optimality condition.
- Put $\hat{\boldsymbol{w}}'$ back in the original problem and solve for $\hat{w}_0$.

- We now obtained $\widehat{\boldsymbol{w}}$ using Lagragian multipliers $\boldsymbol{\lambda}$.

Soft-margin Classifier (Dual)

- $\max_{\boldsymbol{\lambda}} -\frac{\tilde{\boldsymbol{\lambda}}^\top \boldsymbol{X}^\top \boldsymbol{X} \tilde{\boldsymbol{\lambda}}}{4} + \langle \boldsymbol{\lambda}, \mathbf{1} \rangle$
- Subject to $0 \leq \lambda_i \leq 1, \sum_{i=1} \lambda_i y_i = 0$

- Our input data $\{\boldsymbol{x}_i\}$ **only appear at $\boldsymbol{X}^\top \boldsymbol{X}$**
- **Let $\boldsymbol{K} = \boldsymbol{X}^\top \boldsymbol{X}$, then** $K^{(i,j)} = \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle$
- Instead of using the inner product, we can use kernel functions $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ to perform training and prediction.

- **Homework**: write down decision function $f(\boldsymbol{x}; \boldsymbol{w})$ using kernel function k, w_0 and dual variable $\boldsymbol{\lambda}$.

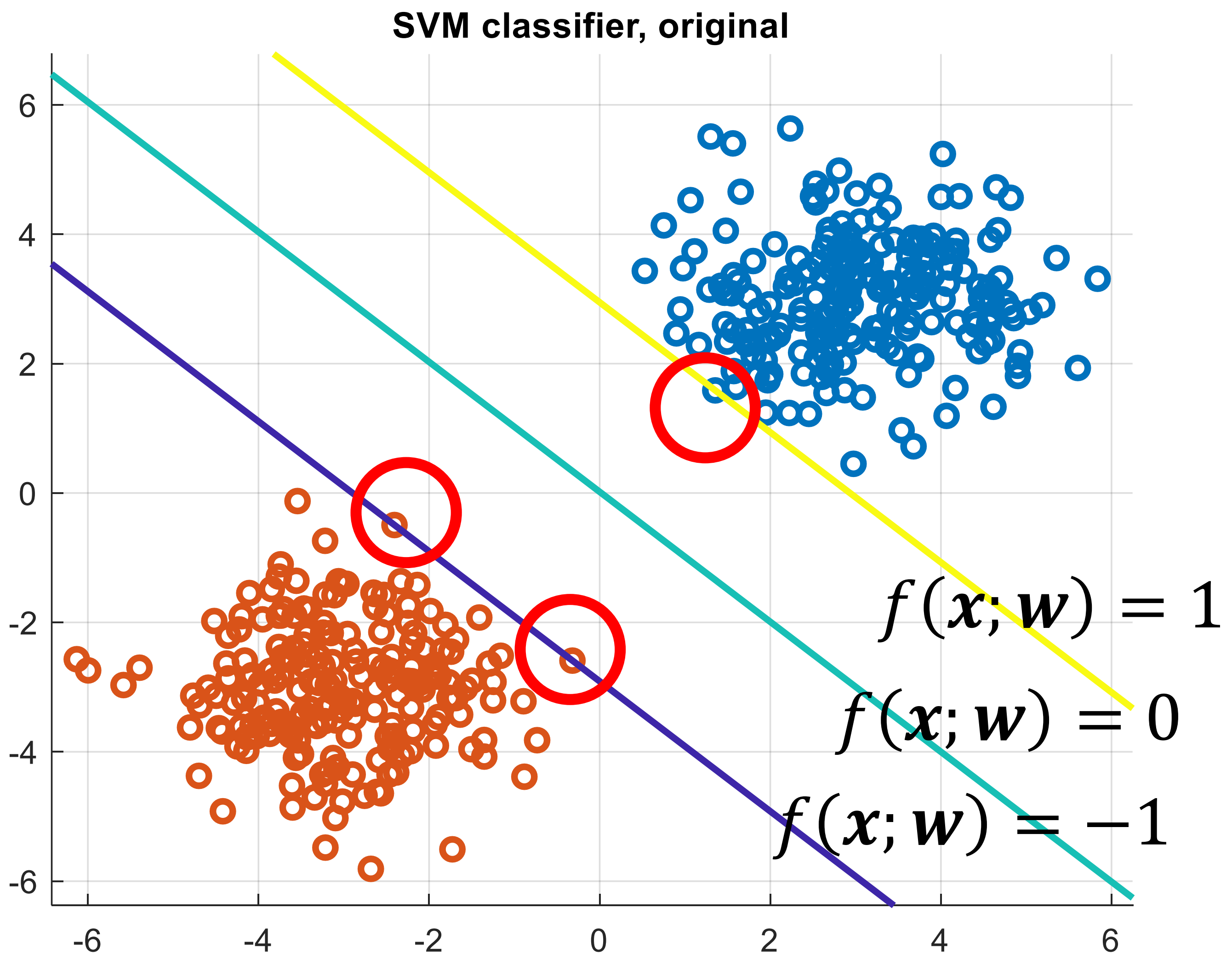
Original vs. Dual Problem

- $\min_{\boldsymbol{w},\epsilon} ||\boldsymbol{w}'||^2 + \sum_i \epsilon_i$
- Subject to $\forall_i$, $y_i(\langle \boldsymbol{w}', \boldsymbol{x}_i \rangle + w_0) + \epsilon_i \geq 1$, $\epsilon_i > 0$

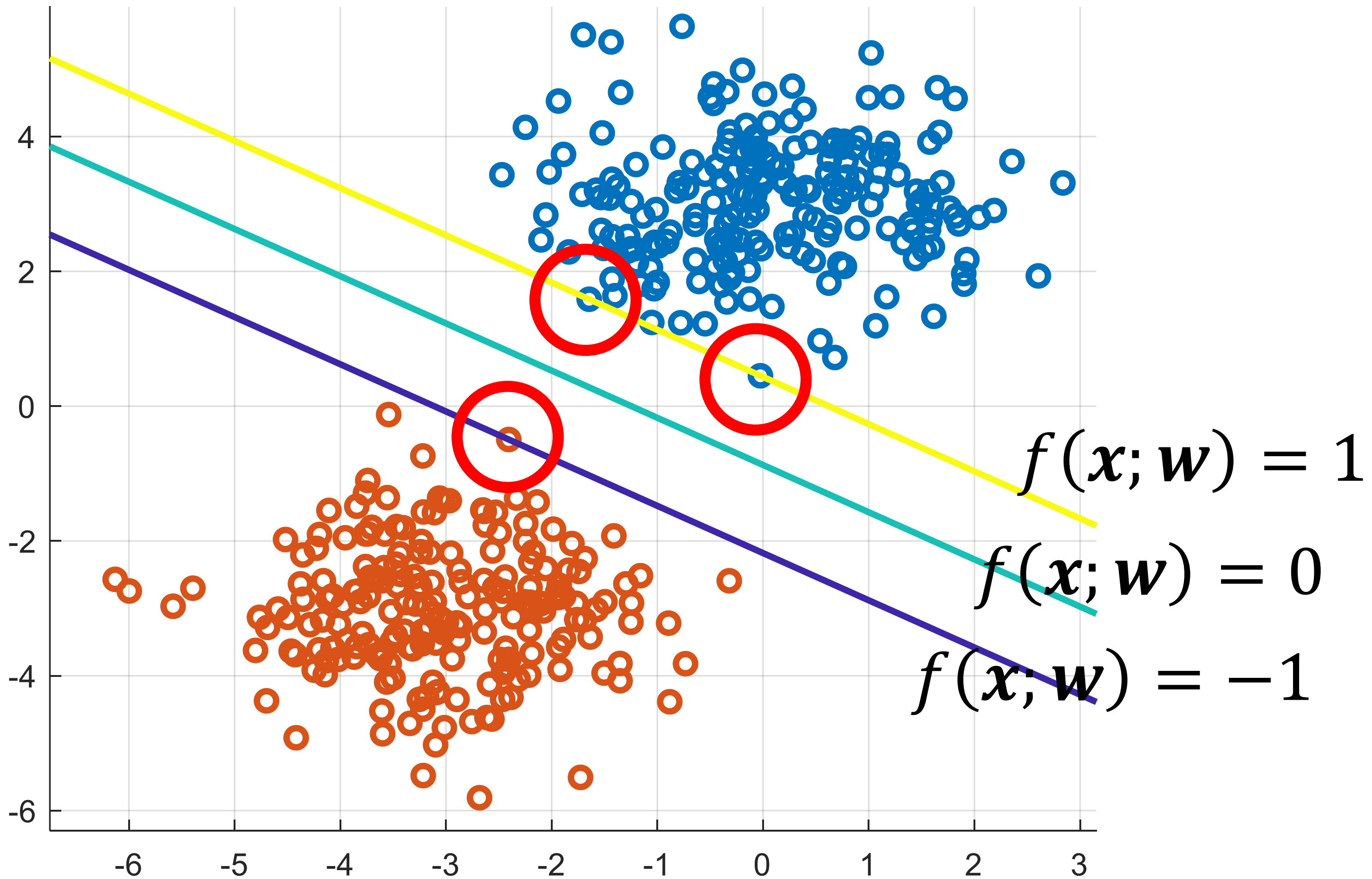
- Complex Constraints
- Quadratic w.r.t. $\boldsymbol{w} \in R^{d+1}$
- Slow when d is large

- $\max_{\boldsymbol{\lambda}} -\frac{\tilde{\boldsymbol{\lambda}}^\top \boldsymbol{X}^\top \boldsymbol{X} \tilde{\boldsymbol{\lambda}}}{4} + \langle \boldsymbol{\lambda}, \mathbf{1} \rangle$
- Subject to
- $0 \leq \lambda_i \leq 1$
- $\sum_{i=1} \lambda_i y_i = 0$
- Simpler Constraints
- Quadratic w.r.t. $\boldsymbol{\lambda} \in R^n$
- Slow when n is large
- Can use kernel!

Toy Example

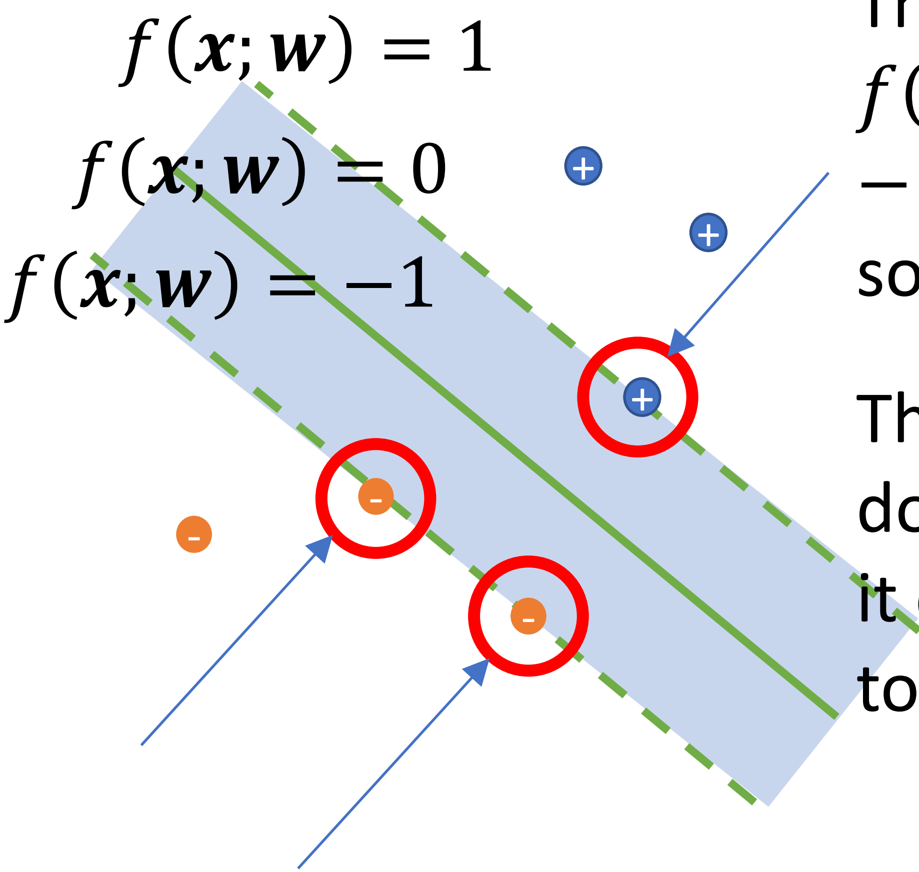

SVM classifier, original
6
4
2
0
-2
-4
-6
-6
-4
-2
0
2
4
6
$f(\boldsymbol{x}; \boldsymbol{w}) = 1$
$f(\boldsymbol{x}; \boldsymbol{w}) = 0$
$f(\boldsymbol{x}; \boldsymbol{w}) = -1$

Toy Example



4
2
0
-2
-4
-6
-6
-5
-4
-3
-2
-1
0
1
2
3
$f(\boldsymbol{x};\boldsymbol{w}) = 1$
$f(\boldsymbol{x};\boldsymbol{w}) = 0$
$f(\boldsymbol{x};\boldsymbol{w}) = -1$

"Support Vectors"

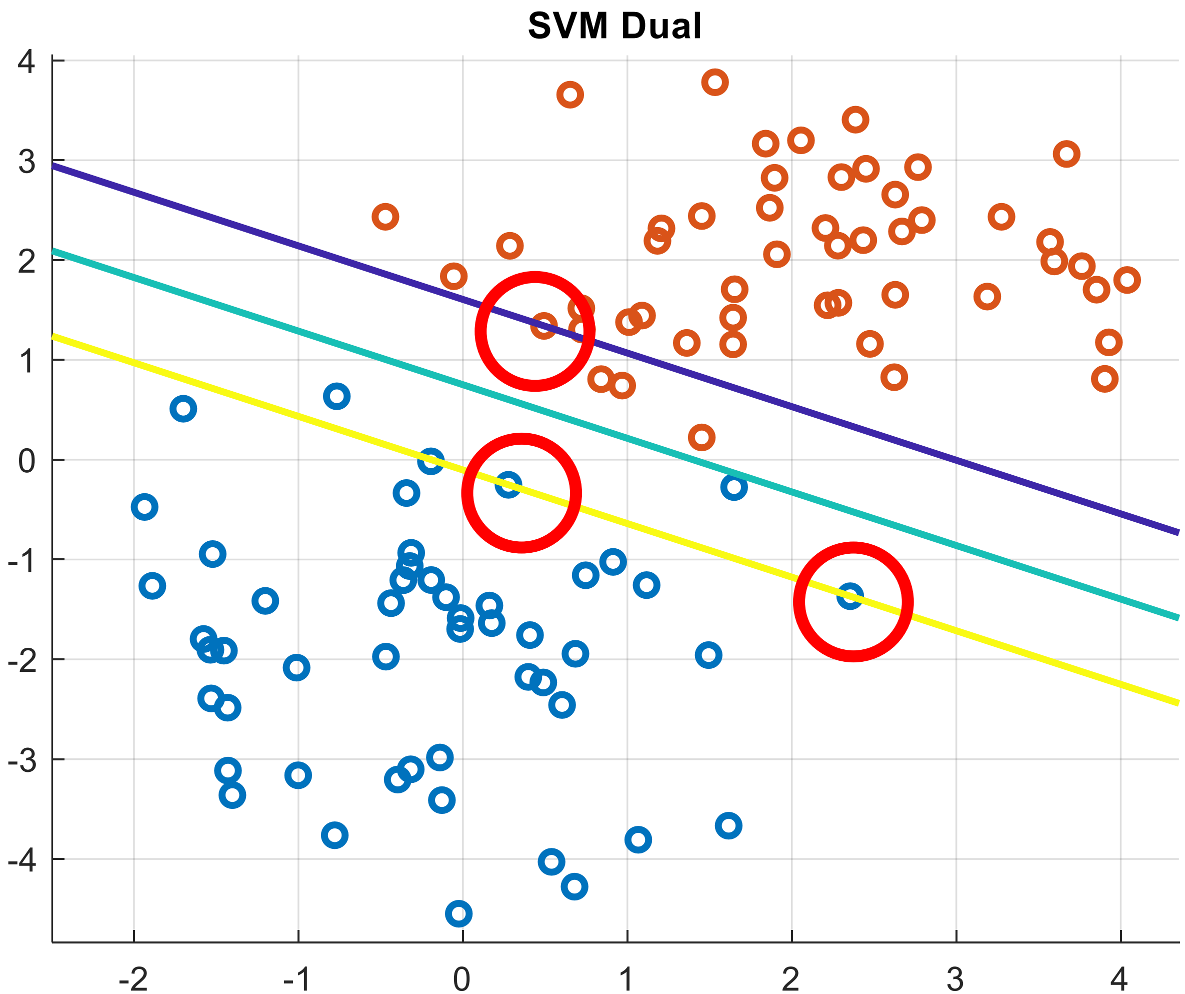


The border of the margin, $f(\boldsymbol{x};\boldsymbol{w}) = 1$ and $f(\boldsymbol{x};\boldsymbol{w}) = -1$ always passing through some data points.

This is natural, as if our margin does not touch any data points, it can further expand till touching data points.

These points as if they were resisting the expansion of the margin, are appropriately called "support vectors".

Toy, Soft-margin, Dual



$\boldsymbol{w}$ by solving Original:

$$w' = [-0.6287 \quad -1.1708]$$
$$w_0 = 0.8797$$

$\boldsymbol{w}$ recovered from $\boldsymbol{\lambda}$:

$$w' = [-0.6287 \quad -1.1708]$$
$$w_0 = 0.8797$$

Limitations of SVM

- SVM is **not** a probabilistic classifier
  - cannot be integrated with probabilistic classification models (generative or discriminative)
  - The decision function lacks probabilistic interpretation.

- Computational cost of SVM is high
  - Both original and dual requires solving constrained optimization.
  - Many other classifier, e.g. Logistic Regression, solves unconstrained optimization.

- Multi-class SVM classification is non-trivial.
  - SVM is motivated by the geometry of binary classification.

Conclusion

- SVM is motivated by "Maximum Margin" principle.
- Soft-margin SVM can classify overlapping pos/neg data.
- Dual of SVM can be derived using Lagrangian.
- SVM is not a probabilistic classifier.

Homework

- Derive the optimality condition in $l(\boldsymbol{\lambda})$ for $\boldsymbol{w}$ and $\boldsymbol{\epsilon}$.

- Represent prediction function $f(\boldsymbol{x};\boldsymbol{w})$ using dual parameter $\boldsymbol{\lambda}$, kernel function k and bias w_0.

Computing Lab

- See additional slides.