ENGR-E 511; ENGR-E 399

Machine Learning for Signal Processing

Module 09:

Support Vector Machines

Minje Kim

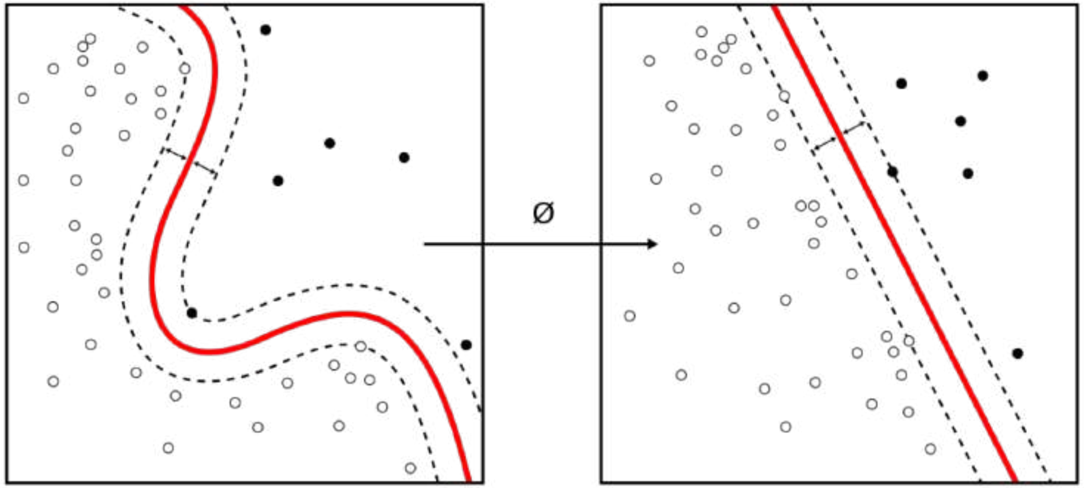
Department of Intelligent Systems Engineering

Email: minje@indiana.edu

Website: http://minjekim.com

Research Group: http://saige.sice.indiana.edu

Meeting Request: http://doodle.com/minje



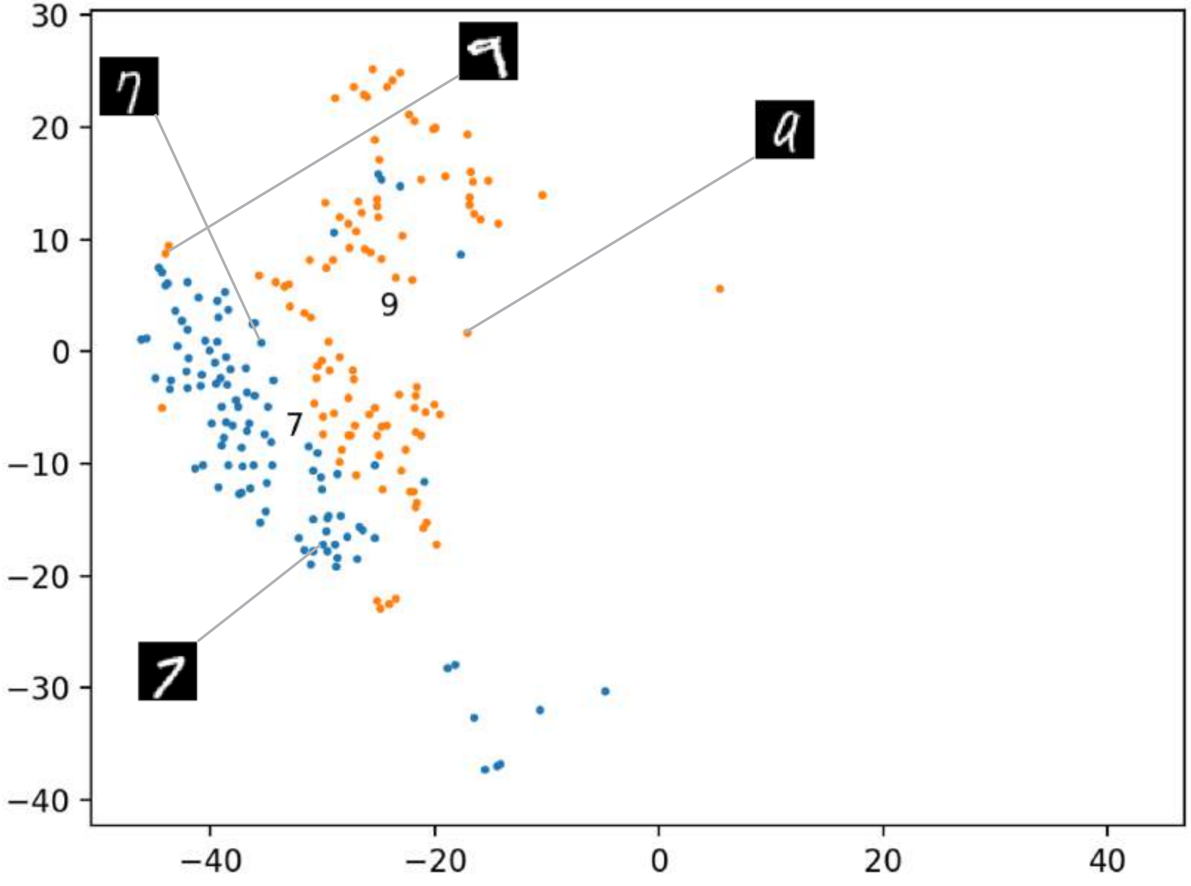
INDIANA UNIVERSITY

SCHOOL OF INFORMATICS, COMPUTING, AND ENGINEERING

Fisher's Linear Discriminant Analysis

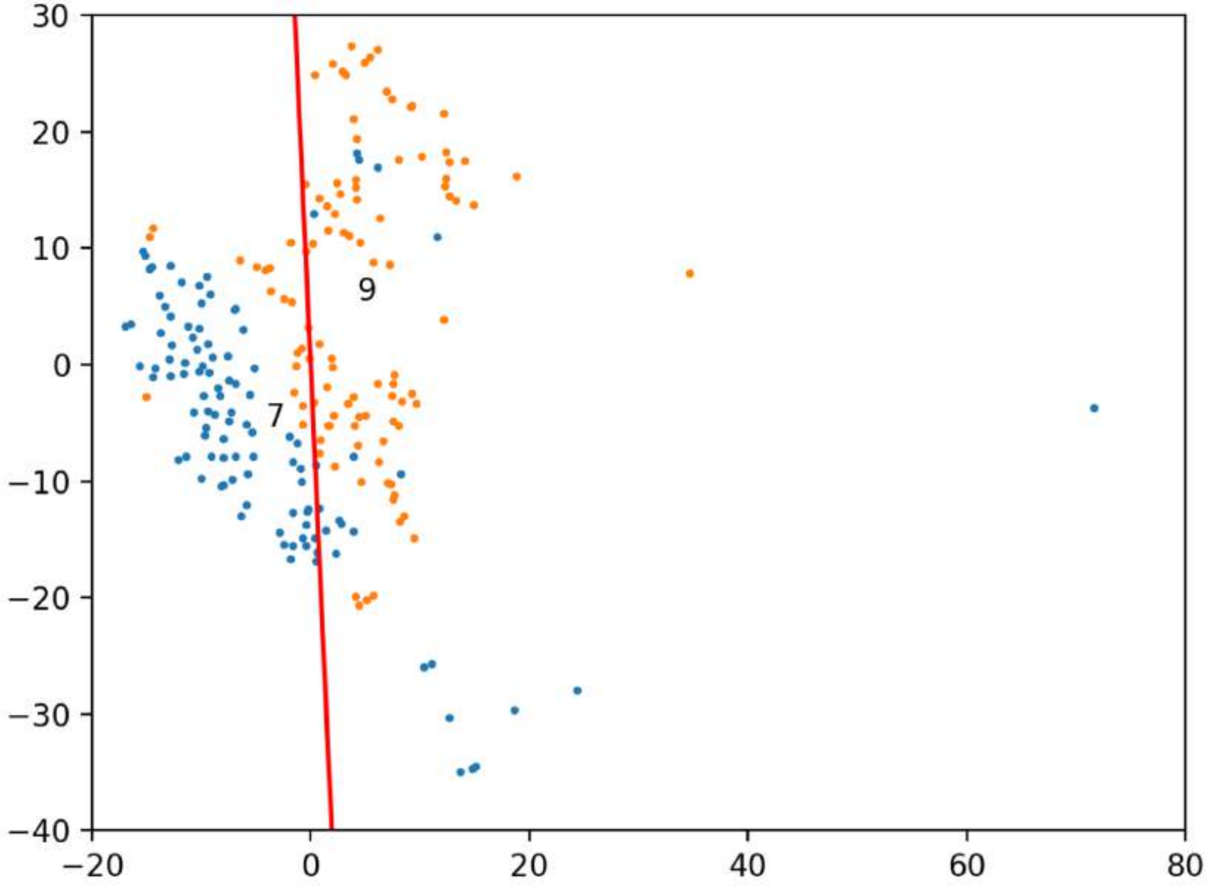
- t-distributed Stochastic Neighbor Embedding

- ○ MNIST handwritten digits
  - □ 28 X 28 pixels = 784 dimensions
- ○ tSNE
  - □ Yet another manifold learning technique
  - □ Just for visualization
- ○ Imagine this 2D space
  is your original data space
- ○ There are some confusing examples
- ○ What would be the best projection vector?
  - □ i.e. You are reducing them into a 1D space?



Fisher's Linear Discriminant Analysis
- PCA?

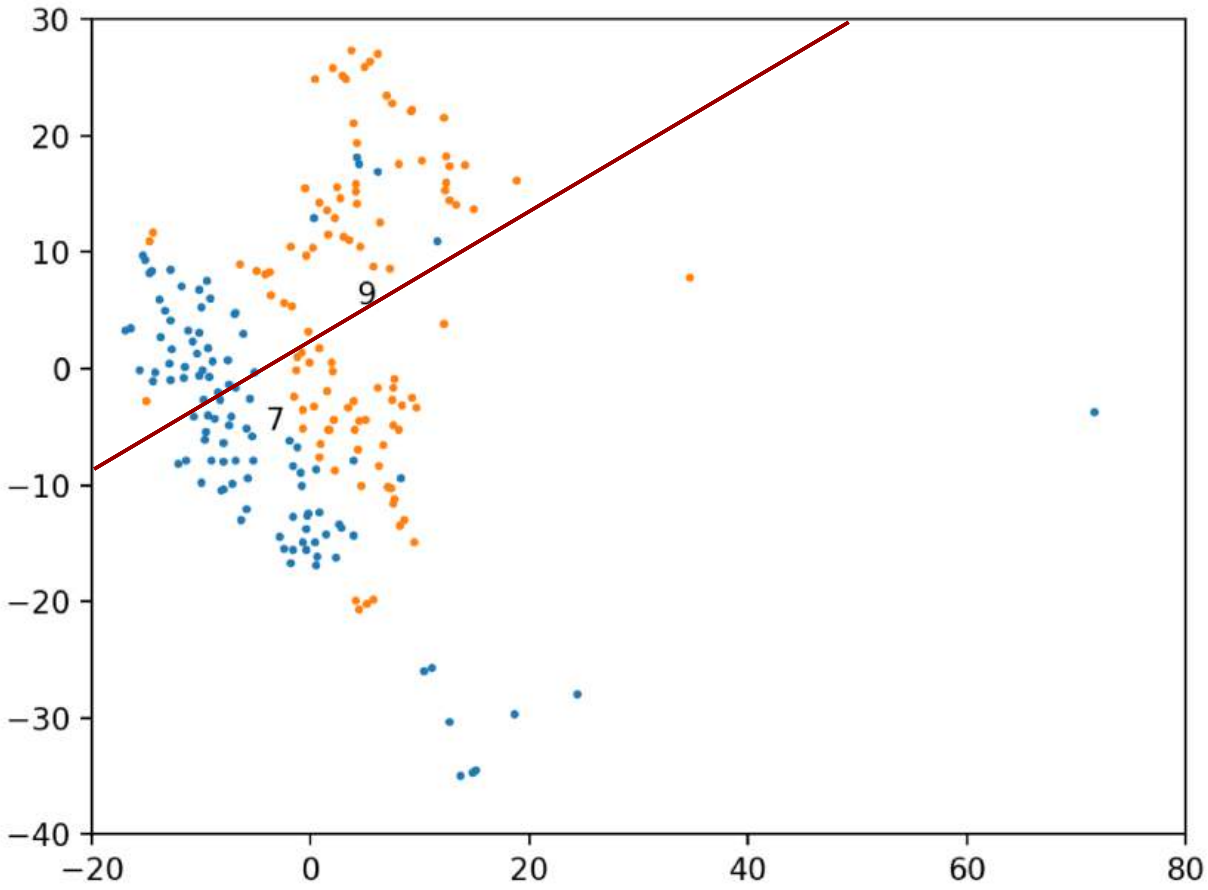
- Your first choice will be PCA
- Do you like it?
- Why not?
- PCA knew nothing about the classes
  - The first PC happens to be a very bad choice
  - When it comes to classification
- Any better projection than PCA?



Fisher's Linear Discriminant Analysis

- Optimal reduced rank decision boundary

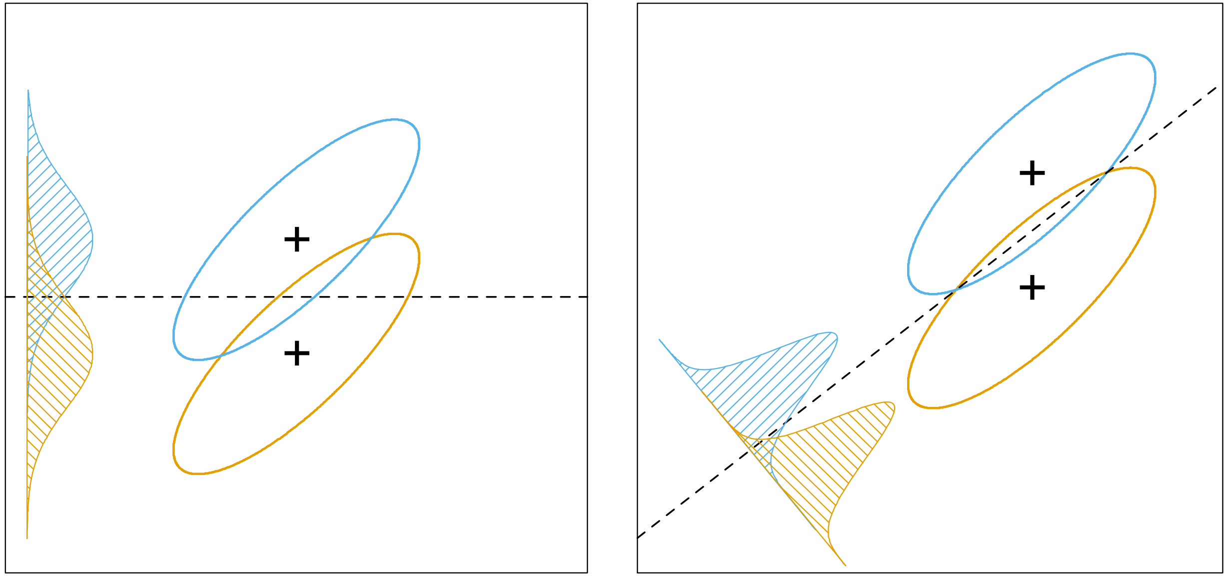
- How about this one?
- How do I find this one?
- What do I want from this projection?



Fisher's Linear Discriminant Analysis

- Optimal reduced rank decision boundary

- Let's assume Gaussian
- After the projection,
- We want the means to be far
  - **Maximum between-class scatter**
  - Projection onto the difference vector b/w means
- Is that all?
- Maximum between-class variance is not enough
  - We want the **within-class scatter** to be **small**, too
- Why do we care?
  - To minimize the overlap between the class-specific distributions
- Once again, how do we find this projection?



Fisher's Linear Discriminant Analysis

- Within-class scatter

- $\boldsymbol{a}$ is the projection vector you're looking for
- Within-class scatter?
  - Class-specific mean $\boldsymbol{\mu}_k = \frac{1}{|\mathcal{C}_k|} \sum_{t \in \mathcal{C}_k} \boldsymbol{x}_t$
  - Class-specific mean after projection $\tilde{\boldsymbol{\mu}}_k = \boldsymbol{a}^\top \boldsymbol{\mu}_k = \boldsymbol{a}^\top \frac{1}{|\mathcal{C}_k|} \sum_{t \in \mathcal{C}_k} \boldsymbol{x}_t = \frac{1}{|\mathcal{C}_k|} \sum_{t \in \mathcal{C}_k} \boldsymbol{a}^\top \boldsymbol{x}_t$
  - Within-class scatter matrix $\boldsymbol{W}_k = \sum_{t \in \mathcal{C}_k} (\boldsymbol{x}_t - \boldsymbol{\mu}_k)(\boldsymbol{x}_t - \boldsymbol{\mu}_k)^\top$
  - **Within-class scatter after projection** $\boldsymbol{a}^\top \boldsymbol{W}_k \boldsymbol{a} = \boldsymbol{a}^\top \left(\sum_{t \in \mathcal{C}_k} (\boldsymbol{x}_t - \boldsymbol{\mu}_k)(\boldsymbol{x}_t - \boldsymbol{\mu}_k)^\top \right) \boldsymbol{a} =$

$$\sum_{t \in \mathcal{C}_k} (\boldsymbol{a}^\top \boldsymbol{x}_t - \boldsymbol{a}^\top \boldsymbol{\mu}_k)(\boldsymbol{a}^\top \boldsymbol{x}_t - \boldsymbol{a}^\top \boldsymbol{\mu}_k)^\top$$

- You want this to be small

Fisher's Linear Discriminant Analysis

- Between-class scatter

- Let's start from the total scatter

$$\boldsymbol{T} = \sum_t (\boldsymbol{x}_t - \boldsymbol{\mu})(\boldsymbol{x}_t - \underbrace{\boldsymbol{\mu}}_{\text{Total mean}})^\top = \sum_k \sum_{t \in \mathcal{C}_k} (\boldsymbol{x}_t - \boldsymbol{\mu})(\boldsymbol{x}_t - \boldsymbol{\mu})^\top$$

$$= \sum_k \sum_{t \in \mathcal{C}_k} (\boldsymbol{x}_t - \boldsymbol{\mu}_k + \boldsymbol{\mu}_k - \boldsymbol{\mu})(\boldsymbol{x}_t - \boldsymbol{\mu}_k + \boldsymbol{\mu}_k - \boldsymbol{\mu})^\top$$

$$= \sum_k \sum_{t \in \mathcal{C}_k} \Big((\boldsymbol{x}_t - \boldsymbol{\mu}_k) + (\boldsymbol{\mu}_k - \boldsymbol{\mu})\Big)\Big((\boldsymbol{x}_t - \boldsymbol{\mu}_k) + (\boldsymbol{\mu}_k - \boldsymbol{\mu})\Big)^\top$$

$$\sum_{t \in \mathcal{C}_k} \boldsymbol{x}_t - \boldsymbol{\mu}_k = 0$$

$$= \sum_k \sum_{t \in \mathcal{C}_k} (\boldsymbol{x}_t - \boldsymbol{\mu}_k)(\boldsymbol{x}_t - \boldsymbol{\mu}_k)^\top + (\boldsymbol{\mu}_k - \boldsymbol{\mu})(\boldsymbol{\mu}_k - \boldsymbol{\mu})^\top + \cancel{(\boldsymbol{x}_t - \boldsymbol{\mu}_k)(\boldsymbol{\mu}_k - \boldsymbol{\mu})^\top} + \cancel{(\boldsymbol{\mu}_k - \boldsymbol{\mu})(\boldsymbol{x}_t - \boldsymbol{\mu}_k)^\top}$$

$$= \sum_k \boldsymbol{W}_k + \sum_k \sum_{t \in \mathcal{C}_k} (\boldsymbol{\mu}_k - \boldsymbol{\mu})(\boldsymbol{\mu}_k - \boldsymbol{\mu})^\top$$

- Between-class scatter $\boldsymbol{B} = \sum_k |\mathcal{C}_k| (\boldsymbol{\mu}_k - \boldsymbol{\mu})(\boldsymbol{\mu}_k - \boldsymbol{\mu})^\top$

- **Between-class scatter after projection** $\boldsymbol{a}^\top \boldsymbol{B} \boldsymbol{a}$

Fisher's Linear Discriminant Analysis

- Rayleigh quotient

- We want to maximize the between-class scatter, while minimize the within-class scatter
- To maximize (generalized) Rayleigh quotient $R(\boldsymbol{a}) = \dfrac{\boldsymbol{a}^\top \boldsymbol{B}\boldsymbol{a}}{\boldsymbol{a}^\top \boldsymbol{W}\boldsymbol{a}}$
- What's next?

$$\frac{\partial R(\boldsymbol{a})}{\partial \boldsymbol{a}} = \frac{2\boldsymbol{B}\boldsymbol{a}(\boldsymbol{a}^\top \boldsymbol{W}\boldsymbol{a}) - 2\boldsymbol{W}\boldsymbol{a}(\boldsymbol{a}^\top \boldsymbol{B}\boldsymbol{a})}{(\boldsymbol{a}^\top \boldsymbol{B}\boldsymbol{a})^2} = 0$$

$$\leftrightarrow \boldsymbol{B}\boldsymbol{a}(\boldsymbol{a}^\top \boldsymbol{W}\boldsymbol{a}) = \boldsymbol{W}\boldsymbol{a}(\boldsymbol{a}^\top \boldsymbol{B}\boldsymbol{a})$$

$$\leftrightarrow \boldsymbol{B}\boldsymbol{a} = R(\boldsymbol{a})\boldsymbol{W}\boldsymbol{a}$$

$$\leftrightarrow \boldsymbol{B}\boldsymbol{a} = \lambda \boldsymbol{W}\boldsymbol{a} \qquad\qquad \lambda = R(\boldsymbol{a})$$

$$\leftrightarrow \boldsymbol{W}^{-1}\boldsymbol{B}\boldsymbol{a} = \lambda \boldsymbol{a}$$

- **Maximum Rayleigh quotient: largest eigenvalue**
- **fLDA projection: the eigenvector with the largest eigenvalue**

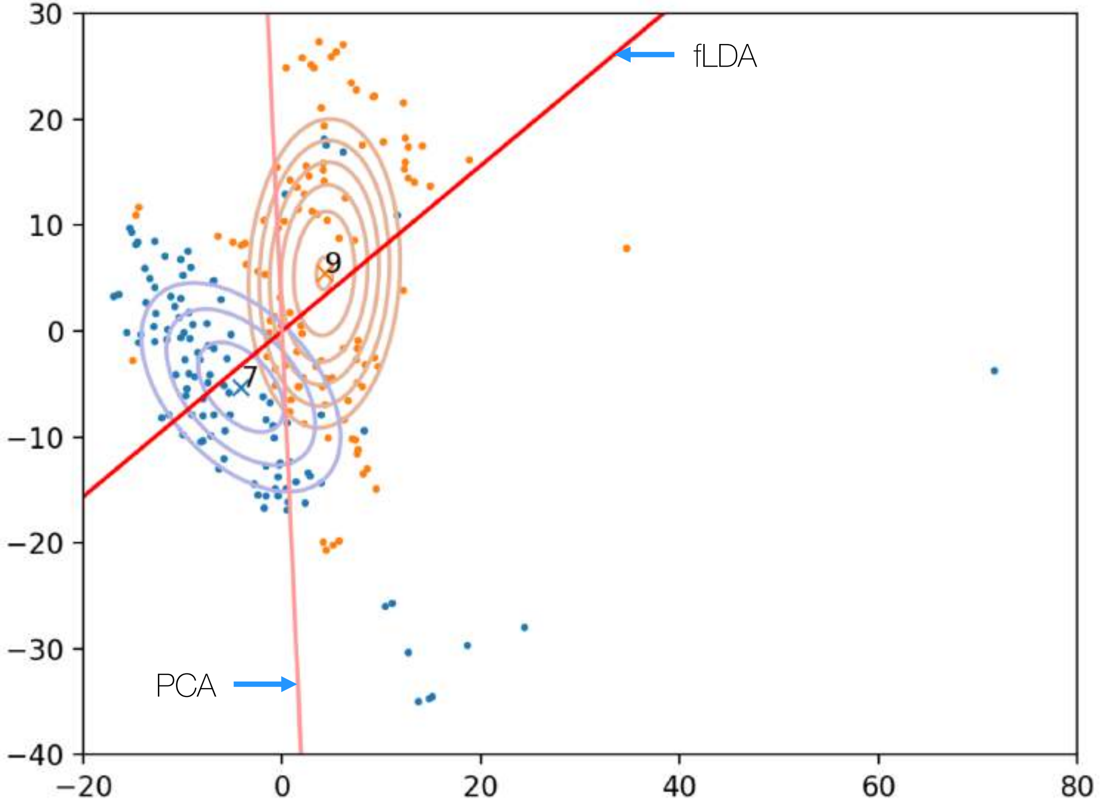
Fisher's Linear Discriminant Analysis

- Supervised dimension reduction

- It's not the line that connects the means!

- fLDA finds projection that
  - Makes the class-specific distribution compact
  - Makes the class-specific distributions farther from each other

- Samples near the decision boundary are less confused
  - Linearly separable case: the closest sample gets farther
  - Not-linearly separable case: a lessor number of confusing samples



The Shortest Distance Between a Line and a Dot

- Warm-up

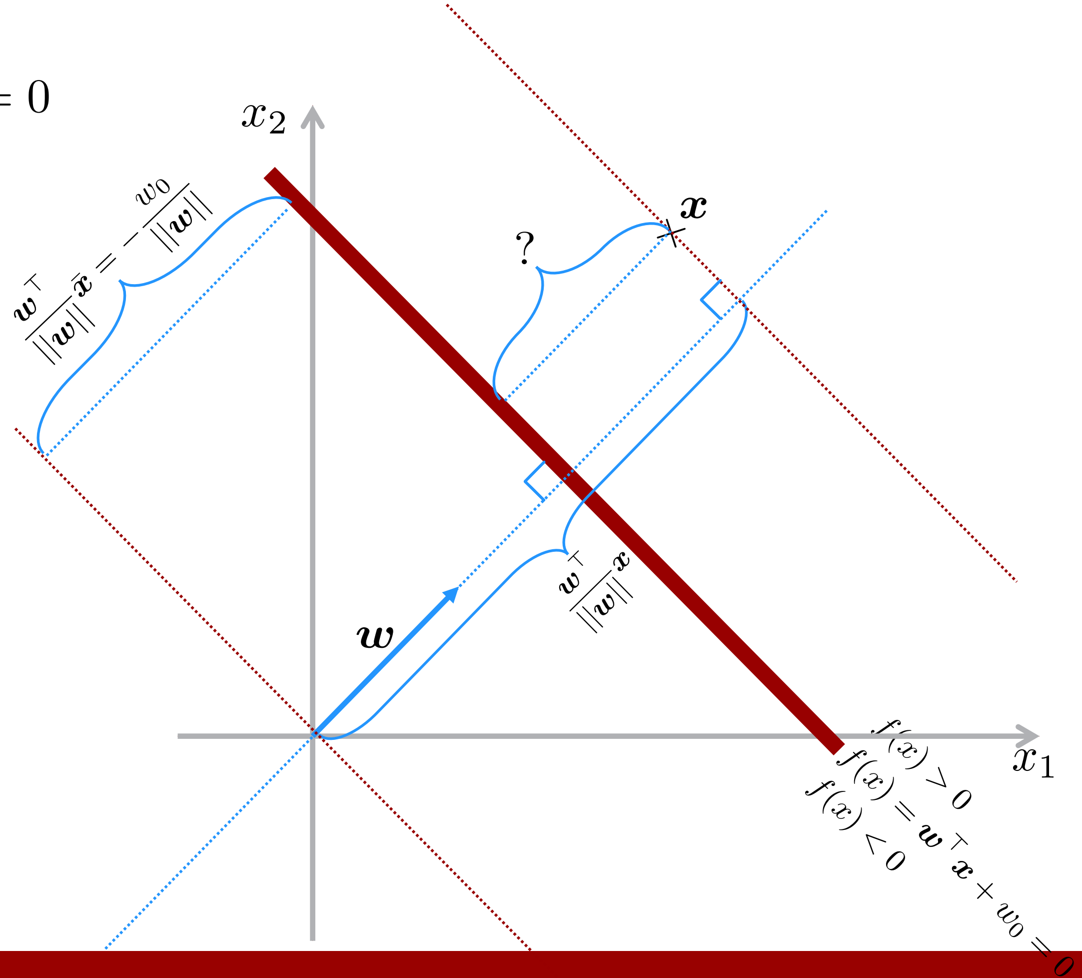
- $\boldsymbol{w}$ is orthogonal to the line
  $\boldsymbol{w}^\top \boldsymbol{x}_A + w_0 = \boldsymbol{w}^\top \boldsymbol{x}_B + w_0 = 0 \leftrightarrow \boldsymbol{w}^\top (\boldsymbol{x}_A - \boldsymbol{x}_B) = 0$

- The projection of a sample on the line to the normalized $\boldsymbol{w}$ defines the amount of the shift from the origin $\frac{\boldsymbol{w}^\top}{||\boldsymbol{w}||}\bar{\boldsymbol{x}} = -\frac{w_0}{||\boldsymbol{w}||}$

- The projection of an arbitrary sample $\boldsymbol{x}$ to the normalized $\boldsymbol{w}$ defines the amount of the shift from the origin $\frac{\boldsymbol{w}^\top}{||\boldsymbol{w}||}\boldsymbol{x}$

- Distance b/w $\boldsymbol{x}$ and $f(\boldsymbol{x})$?

$$\frac{\boldsymbol{w}^\top}{||\boldsymbol{w}||}\boldsymbol{x} + \frac{w_0}{||\boldsymbol{w}||} = \frac{f(\boldsymbol{x})}{||\boldsymbol{w}||}$$



The Shortest Distance Between a Line and a Dot

- Warm-up

- Hyperplane (decision boundary) is defined by $\boldsymbol{x}$ that meets $f(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} + w_0 = 0$
- Hyperplane doesn't change however you scale the function $af(\boldsymbol{x}) = a\boldsymbol{w}^\top \boldsymbol{x} + aw_0 = 0$
- Distance between a data point and the hyperplane is invariant to the scale $\dfrac{\boldsymbol{w}^\top}{||\boldsymbol{w}||}\boldsymbol{x} + \dfrac{w_0}{||\boldsymbol{w}||} = \dfrac{f(\boldsymbol{x})}{||\boldsymbol{w}||}$
- To clarify, the "distance" is this: $\dfrac{|f(\boldsymbol{x})|}{||\boldsymbol{w}||}$
- Or, by introducing bipolar binary labels: $y_t \in \{-1, +1\}$

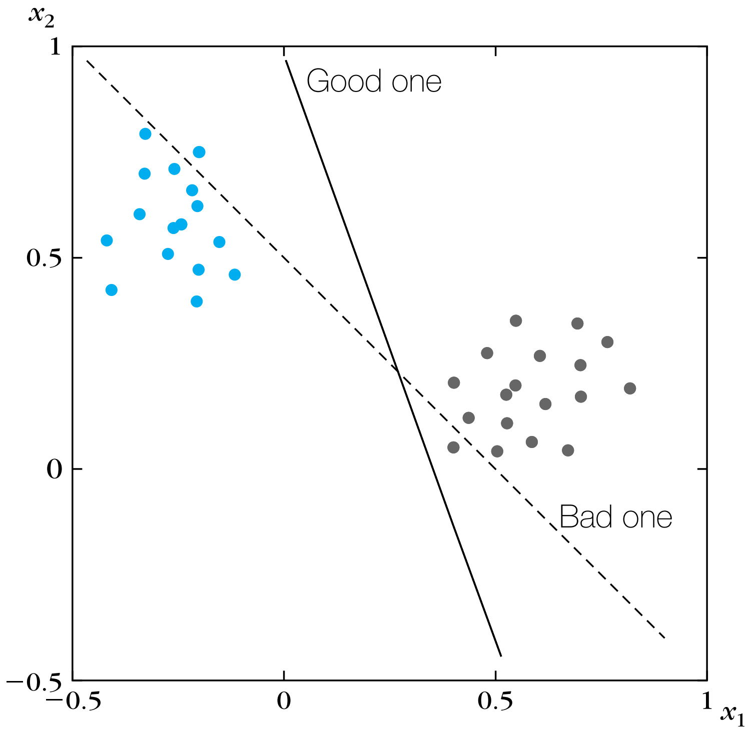
$$\frac{|f(\boldsymbol{x})|}{||\boldsymbol{w}||} = y_t\left(\frac{f(\boldsymbol{x})}{||\boldsymbol{w}||}\right) = y_t\left(\frac{\boldsymbol{w}^\top}{||\boldsymbol{w}||}\boldsymbol{x} + \frac{w_0}{||\boldsymbol{w}||}\right)$$

  - (Easier to handle than the absolute function during optimization)
- So what?
  - You're ready to learn **Maximum Margin Classifiers**

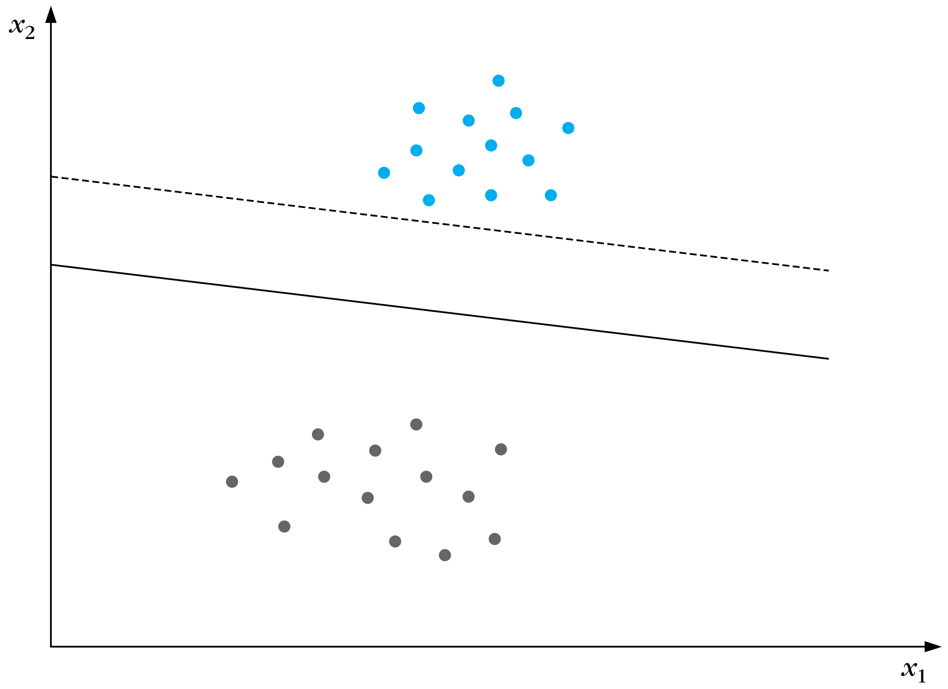
Maximum Margin Classifiers

- Optimal separation

- There can be many hyperplanes
  - Misclassification is something to prevent



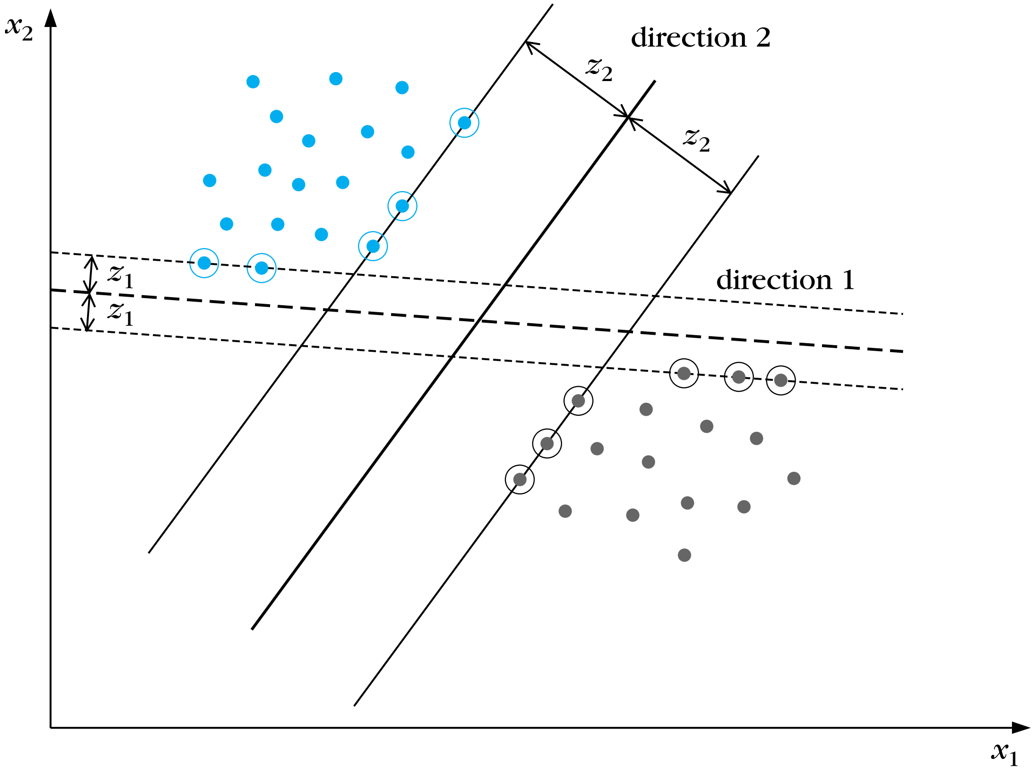
- Is that all?
  - Which one do you prefer?



Maximum Margin Classifiers

- Margins?

- Which one do you prefer?
  - You prefer the one with larger margin $z_1 < z_2$
- Margin
  - The shortest perpendicular distance between the decision boundary and the data points
- You find the hyperplane that **maximizes the margin**
- By the way, those closest data samples are **support vectors**
- *"I've heard a lot about this concept by now, so could you please teach me how to learn the decision boundary that maximizes the margin?"*
  - I think you've learned that part somewhere else, too, but let me see what I can do.
  - It involves ugly math…



Maximum Margin Classifiers

- Margins?

- I prepped you as to how to calculate the distance between a line and a dot

$$\frac{|f(\boldsymbol{x})|}{||\boldsymbol{w}||} = y_t\left(\frac{f(\boldsymbol{x})}{||\boldsymbol{w}||}\right) = y_t\left(\frac{\boldsymbol{w}^\top}{||\boldsymbol{w}||}\boldsymbol{x} + \frac{w_0}{||\boldsymbol{w}||}\right)$$

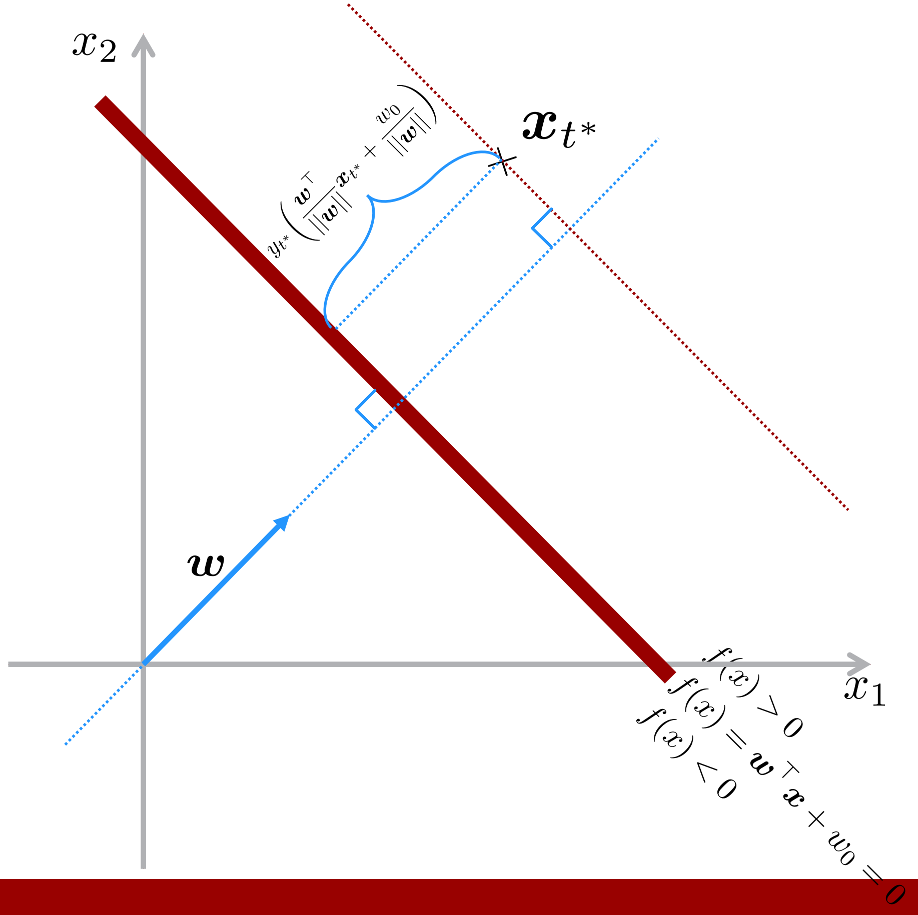
- But I'm talking about the margin, the shortest distance

$$\min_t y_t\left(\frac{\boldsymbol{w}^\top}{||\boldsymbol{w}||}\boldsymbol{x}_t + \frac{w_0}{||\boldsymbol{w}||}\right)$$

- And I want to maximize this

$$\arg\max_{\boldsymbol{w},w_0}\left(\frac{1}{||\boldsymbol{w}||}\min_t \left[y_t\left(\boldsymbol{w}^\top\boldsymbol{x}_t + w_0\right)\right]\right)$$

- This is kind of ugly
  - i.e. I don't know how to optimize this



Maximum Margin Classifiers

- The objective function

- First, “the distance between a data point and the hyperplane is invariant to the scale”

$$y_t\left(\frac{\boldsymbol{w}^\top}{||\boldsymbol{w}||}\boldsymbol{x} + \frac{w_0}{||\boldsymbol{w}||}\right) = y_t\left(\frac{a\boldsymbol{w}^\top}{a||\boldsymbol{w}||}\boldsymbol{x} + \frac{aw_0}{a||\boldsymbol{w}||}\right)$$

- Imagine we always scale properly so that

$$\min_t y_t(a\boldsymbol{w}^\top\boldsymbol{x}_t + aw_0) = 1$$

- Or you can just simply drop the specific scaling factor

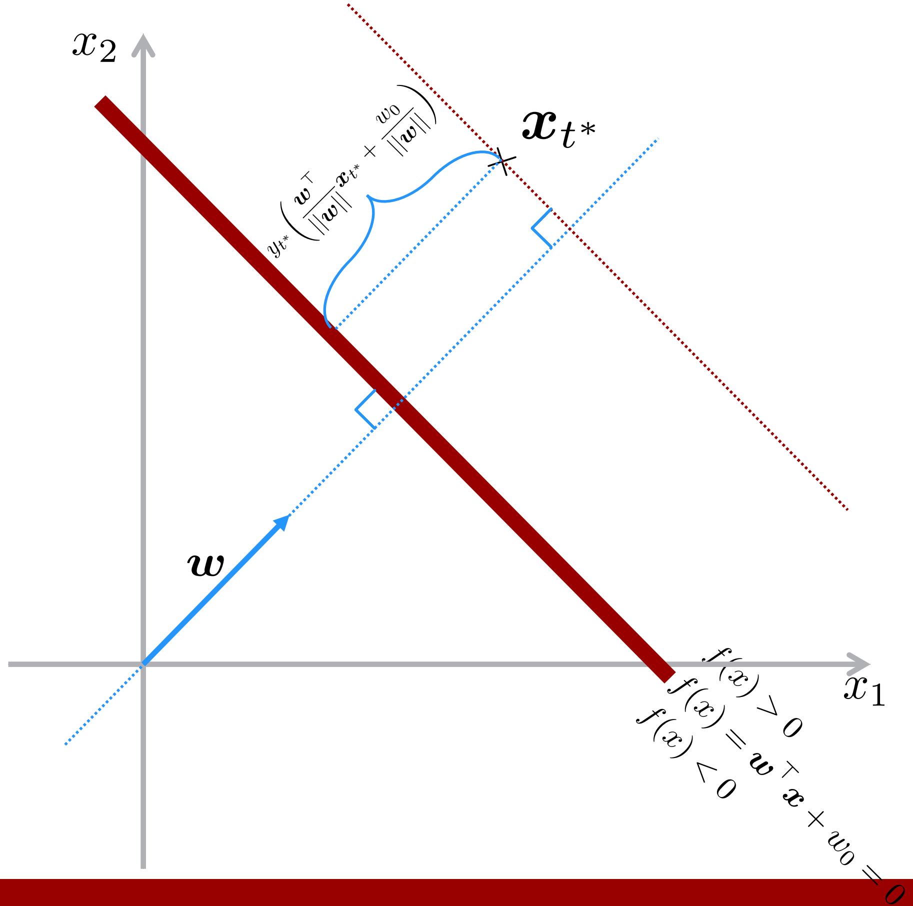
$$\min_t y_t(\boldsymbol{w}^\top\boldsymbol{x}_t + w_0) = 1$$

  - This doesn’t change the hyperplane nor the margin

- Then what?

$$\arg\max_{\boldsymbol{w},w_0}\left(\frac{1}{||\boldsymbol{w}||}\min_t\ \left[y_t(\boldsymbol{w}^\top\boldsymbol{x}_t + w_0)\right]\right)$$

$$= \arg\min_{\boldsymbol{w},w_0}\frac{1}{2}||\boldsymbol{w}||^2 \qquad \text{s.t. } y_t(\boldsymbol{w}^\top\boldsymbol{x}_t + w_0) \geq 1, \quad \forall t$$



Maximum Margin Classifiers

- Optimization

$$\arg\min_{\boldsymbol{w},w_0} \frac{1}{2}||\boldsymbol{w}||^2, \text{ s.t. } y_t(\boldsymbol{w}^\top \boldsymbol{x}_t + w_0) \geq 1, \quad \forall t$$

- The objective function

$$\mathcal{L}(\boldsymbol{w}, w_0, \boldsymbol{a}) = \frac{1}{2}||\boldsymbol{w}||^2 - \sum_t \boxed{a_t} \overset{?}{\{} y_t(\boldsymbol{w}^\top \boldsymbol{x}_t + w_0) - 1\}$$

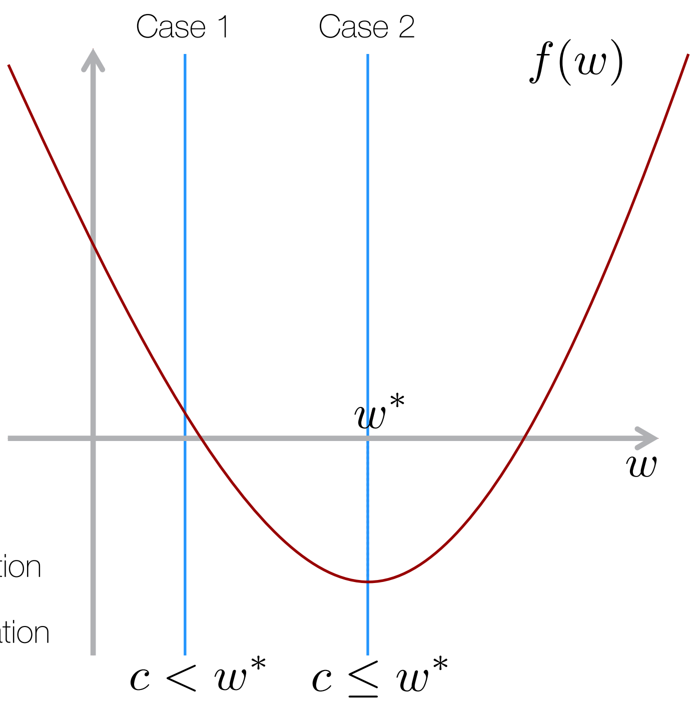
- This comes from the **inequality** constraint, i.e. Lagrange multiplier $f(w) - \lambda g(w)$
  - Case 1: Inequality constraint is **inactive**
    - The Lagrange multiplier is **zero→** stationary point: $\nabla f(w) = 0$
  - Case 2: Inequality constraint is **active**
    - Minimum is when $g(w) = 0$ (same with the equality constraint case)
    - The Lagrange multiplier is ~~not zero~~ **POSITIVE**
    - **Sign matters!** $\nabla f(w) > 0, \ \nabla g(w) > 0 \ \therefore \nabla f(w) - \lambda \nabla g(w) = 0$ ← Minimization

      $\nabla f(w) < 0, \ \nabla g(w) > 0 \ \therefore \nabla f(w) + \lambda \nabla g(w) = 0$ ← Maximization

- $\min f(w) \text{ s.t. } g(w) \geq 0$

  $\min f(w) - \lambda g(w) \text{ s.t. } \boxed{g(w) \geq 0, \ \lambda \geq 0, \ \lambda g(w) = 0}$ KKT condition

$$g(w) = w - c \ > 0$$



Dual Representation

- Incorporating kernels

- Let's eliminate $\boldsymbol{w}, w_0$

$$\mathcal{L}(\boldsymbol{w}, w_0, \boldsymbol{a}) = \frac{1}{2}||\boldsymbol{w}||^2 - \sum_t a_t\{y_t(\boldsymbol{w}^\top \boldsymbol{x}_t + w_0) - 1\} = \frac{1}{2}\boldsymbol{w}^\top\boldsymbol{w} - \sum_t a_t y_t \boldsymbol{w}^\top \boldsymbol{x}_t + a_t y_t w_0 - a_t$$

$$\boxed{\frac{\partial \mathcal{L}}{\partial w_0} = \sum_t a_t y_t = 0}$$

$$= \frac{1}{2}\boldsymbol{w}^\top\boldsymbol{w} - \boldsymbol{w}^\top \underbrace{\cancel{\sum_t a_t y_t \boldsymbol{x}_t}}_{\boldsymbol{w}} + w_0 \underbrace{\cancel{\sum_t a_t y_t}}_{0} + \sum_t a_t = -\frac{1}{2}\boldsymbol{w}^\top\boldsymbol{w} + \sum_t a_t$$

$$\boxed{\frac{\partial \mathcal{L}}{\partial \boldsymbol{w}} = \boldsymbol{w} - \sum_t a_t y_t \boldsymbol{x}_t = 0}$$

$$= -\frac{1}{2}(a_1 y_1 \boldsymbol{x}_1^\top + a_2 y_2 \boldsymbol{x}_2^\top + \cdots + a_T y_T \boldsymbol{x}_T^\top)(a_1 y_1 \boldsymbol{x}_1 + a_2 y_2 \boldsymbol{x}_2 + \cdots + a_T y_T \boldsymbol{x}_T) + \sum_t a_t$$

$$= -\frac{1}{2}\sum_t a_t y_t \boldsymbol{x}_t^\top \left(\sum_l a_l y_l \boldsymbol{x}_l\right) + \sum_t a_t = -\frac{1}{2}\sum_t\sum_l a_t y_t a_l y_l \boxed{\boldsymbol{x}_t^\top \boldsymbol{x}_l} + \sum_t a_t$$

$$= -\frac{1}{2}\sum_t\sum_l a_t y_t a_l y_l \underline{\mathcal{K}(\boldsymbol{x}_t, \boldsymbol{x}_l)} + \sum_t a_t \quad \text{s.t.} \sum_t a_t y_t = 0, \quad a_t \geq 0, \ \forall t$$

$\phi(\boldsymbol{x}_t)^\top \phi(\boldsymbol{x}_l)$ You don't have to deal with the nonlinear transformation once you use a kernel function that implies it

- The SVM dual representation naturally extends to the nonlinear case!
- KKT conditions $\quad a_t \geq 0 \qquad y_t(\boldsymbol{w}^\top \boldsymbol{x}_t + w_0) - 1 \geq 0 \qquad a_t\{y_t(\boldsymbol{w}^\top \boldsymbol{x}_t + w_0) - 1\} = 0$

Prediction

- Prediction using kernel

$$\boxed{\frac{\partial \mathcal{L}}{\partial \boldsymbol{w}} = \boldsymbol{w} - \sum_t a_t y_t \boldsymbol{x}_t = 0 \rightarrow \boldsymbol{w} = \sum_t a_t y_t \boldsymbol{x}_t}$$

$$\boxed{\boldsymbol{w} = \sum_t a_t y_t \phi(\boldsymbol{x}_t)}$$

- So, how do we make prediction for the new test sample?

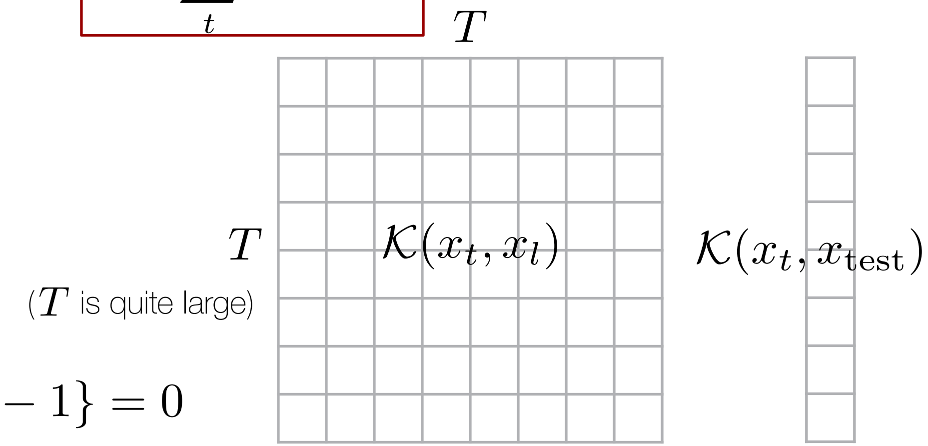
$$\boldsymbol{w}^\top \boldsymbol{x}_{\text{test}} + w_0 > 0? \quad \text{Doable}$$

$$\boldsymbol{w}^\top \phi(\boldsymbol{x}_{\text{test}}) + w_0 > 0? \quad \text{Not doable}$$

$$\sum_t a_t y_t \phi(\boldsymbol{x}_t)^\top \phi(\boldsymbol{x}_{\text{test}}) + w_0 > 0?$$

$$\sum_t a_t y_t \phi(\boldsymbol{x}_t)^\top \phi(\boldsymbol{x}_{\text{test}}) + w_0 = \sum_t a_t y_t \mathcal{K}(\boldsymbol{x}_t, \boldsymbol{x}_{\text{test}}) + w_0$$

$$= \sum_t a_t y_t \mathcal{K}(\boldsymbol{x}_t, \boldsymbol{x}_{\text{test}}) + w_0 > 0$$



- Recall $a_t \geq 0 \quad y_t(\boldsymbol{w}^\top \boldsymbol{x}_t + w_0) - 1 \geq 0 \quad a_t\{y_t(\boldsymbol{w}^\top \boldsymbol{x}_t + w_0) - 1\} = 0$
  - Not all of the terms in the summation matter (e.g. when $a_t = 0$)
  - The ones that don't matter: the ones that are surely classified $y_t(\boldsymbol{w}^\top \boldsymbol{x}_t + w_0) > 1$
  - The ones that matter: **support vectors**! $a_t > 0 \quad y_t(\boldsymbol{w}^\top \boldsymbol{x}_t + w_0) = 1$
- Prediction $\sum_{t \in \mathcal{S}} a_t y_t \mathcal{K}(\boldsymbol{x}_t, \boldsymbol{x}_{\text{test}}) + w_0 > 0?$

  ← Set of SVs

Estimation of the Bias

- Using kernels

- What if you feed a support vector as if it's a test sample?

$$y_{\text{testSV}}(\boldsymbol{w}^\top \phi(\boldsymbol{x}_{\text{testSV}}) + w_0) = 1 \longleftarrow$$ Equation holds because $\boldsymbol{x}_{\text{testSV}}$ is a support vector

$$y_{\text{testSV}}\left(\sum_t a_t y_t \phi(\boldsymbol{x}_t)^\top \phi(\boldsymbol{x}_{\text{testSV}}) + w_0\right) = 1 \qquad \boxed{\boldsymbol{w} = \sum_t a_t y_t \phi(\boldsymbol{x}_t)}$$

- Recall $a_t \geq 0 \quad y_t(\boldsymbol{w}^\top \boldsymbol{x}_t + w_0) - 1 \geq 0 \quad a_t\{y_t(\boldsymbol{w}^\top \boldsymbol{x}_t + w_0) - 1\} = 0$
  - Not all of the terms in the summation matter (e.g. when $a_t = 0$)
  - The ones that don't matter: the ones that are surely classified $y_t(\boldsymbol{w}^\top \boldsymbol{x}_t + w_0) > 1$
  - The ones that matter: **support vectors**! $a_t > 0 \quad y_t(\boldsymbol{w}^\top \boldsymbol{x}_t + w_0) = 1$
- Therefore $y_{\text{testSV}}\left(\sum_{t\in\mathcal{S}} a_t y_t \mathcal{K}(\boldsymbol{x}_t, \boldsymbol{x}_{\text{testSV}}) + w_0\right) = 1 \leftrightarrow y^2_{\text{testSV}}\left(\sum_{t\in\mathcal{S}} a_t y_t \mathcal{K}(\boldsymbol{x}_t, \boldsymbol{x}_{\text{testSV}}) + w_0\right) = y_{\text{testSV}}$

$$\leftrightarrow w_0 = y_{\text{testSV}} - \sum_{t\in\mathcal{S}} a_t y_t \mathcal{K}(\boldsymbol{x}_t, \boldsymbol{x}_{\text{testSV}})$$

- Want to be more careful? $w_0 = \frac{1}{|\mathcal{S}|} \sum_{\text{testSV}\in\mathcal{S}} \left\{ y_{\text{testSV}} - \sum_{t\in\mathcal{S}} a_t y_t \mathcal{K}(\boldsymbol{x}_t, \boldsymbol{x}_{\text{testSV}}) \right\}$

Soft Margin Classifiers

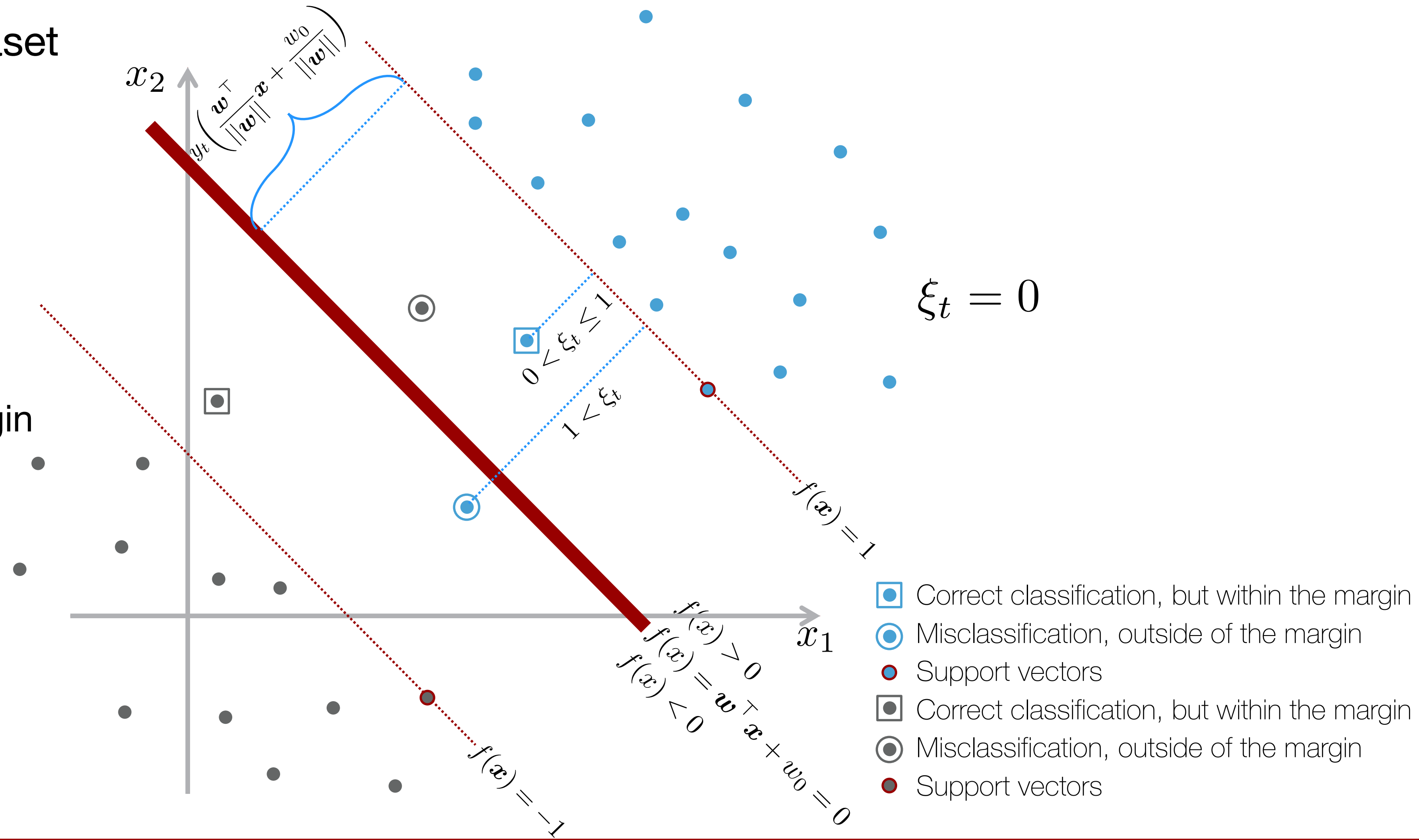
- Slack variables

- What if the training dataset is not separable?

$$y_t f(\boldsymbol{x}_t) \geq 1 - \xi_t$$

Slack variable

- $\xi_t = 0$
  - Support vectors
  - Samples inside the margin
- $0 < \xi_t \leq 1$
  - Inside the margin
- $1 < \xi_t$
  - Outside of the margin



Soft Margin Classifiers

- Objective function with the slack variables

- Objective $\mathcal{L}(\boldsymbol{w}, w_0, \boldsymbol{\xi}, \boldsymbol{a}, \boldsymbol{\mu}) = \frac{1}{2}\boldsymbol{w}^\top \boldsymbol{w} + C\sum_t \xi_t - \sum_t a_t\{y_t(\boldsymbol{w}^\top \boldsymbol{x}_t + w_0) - 1 + \xi_t\} - \sum_t \mu_t \xi_t$
  - KKT Constraints

$$
\begin{array}{ccc}
y_t f(\boldsymbol{x}_t) \geq 1 - \xi_t & a_t \geq 0 & a_t\{y_t f(\boldsymbol{x}_t) - 1 + \xi_t\} = 0 \\
\xi_t \geq 0 & \mu_t \geq 0 & \xi_t \mu_t = 0
\end{array}
$$

- Eliminating variables

$$\mathcal{L} = \frac{1}{2}\boldsymbol{w}^\top \boldsymbol{w} + C\sum_t \xi_t - \boldsymbol{w}^\top \cancelto{\boldsymbol{w}}{\sum_t a_t y_t \boldsymbol{x}_t} + w_0 \cancelto{0}{\sum_t a_t y_t} + \sum_t a_t - \sum_t a_t \xi_t - \sum_t \mu_t \xi_t$$

$$\mathcal{L} = -\frac{1}{2}\boldsymbol{w}^\top \boldsymbol{w} + \sum_t a_t + \cancelto{0}{\sum_t (C - a_t - \mu_t)}\xi_t$$

$$= -\frac{1}{2}\sum_t \sum_l a_t y_t a_l y_l \mathcal{K}(\boldsymbol{x}_t, \boldsymbol{x}_l) + \sum_t a_t$$

$$\frac{\partial \mathcal{L}}{\partial w_0} = \sum_t a_t y_t = 0$$

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{w}} = \boldsymbol{w} - \sum_t a_t y_t \boldsymbol{x}_t = 0 \leftrightarrow \boldsymbol{w} = \sum_t a_t y_t \boldsymbol{x}_t$$

$$\frac{\partial \mathcal{L}}{\partial \xi} = -a_t + C - \mu_t = 0 \leftrightarrow a_t = C - \mu_t$$

- Same with the hard margin case!

Soft Margin Classifiers

- Dual representation and prediction

- Dual representation $\mathcal{L}(\boldsymbol{a}) = -\frac{1}{2}\sum_t \sum_l a_t y_t a_l y_l \mathcal{K}(\boldsymbol{x}_t, \boldsymbol{x}_l) + \sum_t a_t$
  - With new constraints

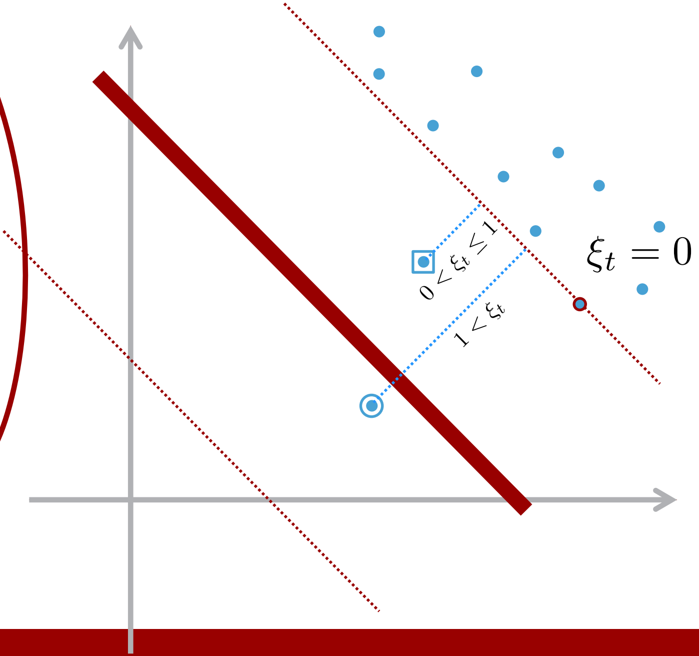
$$a_t \geq 0 \qquad \mu_t \geq 0 \qquad \frac{\partial \mathcal{L}}{\partial \xi} = -a_t + C - \mu_t = 0 \leftrightarrow a_t = C - \mu_t \quad \Leftrightarrow \quad 0 \leq a_t \leq C$$

$$\frac{\partial \mathcal{L}}{\partial w_0} = \sum_t a_t y_t = 0$$

- Prediction $\sum_{t \in \mathcal{S}} a_t y_t \mathcal{K}(\boldsymbol{x}_t, \boldsymbol{x}_{\text{test}}) + w_0 > 0?$
- Recall KKT $y_t f(\boldsymbol{x}_t) \geq 1 - \xi_t \qquad a_t \geq 0 \qquad a_t\{y_t f(\boldsymbol{x}_t) - 1 + \xi_t\} = 0$

$$\xi_t \geq 0 \qquad \mu_t \geq 0 \qquad \xi_t \mu_t = 0$$

  - Not all of the terms in the summation matter (e.g. when $a_t = 0$)
  - The ones that matter: $a_t > 0 \quad y_t(\boldsymbol{w}^\top \phi(\boldsymbol{x}_t) + w_0) = 1 - \xi_t$
    - Too early to call
  - If $0 < a_t < C$, then x_t is a support vector
    - Because $\mu_t > 0 \quad \xi_t = 0$
  - If $a_t = C$, then $\mu_t = 0, \ \xi_t > 0$



SVM Error Function

- Sparsity of SVM

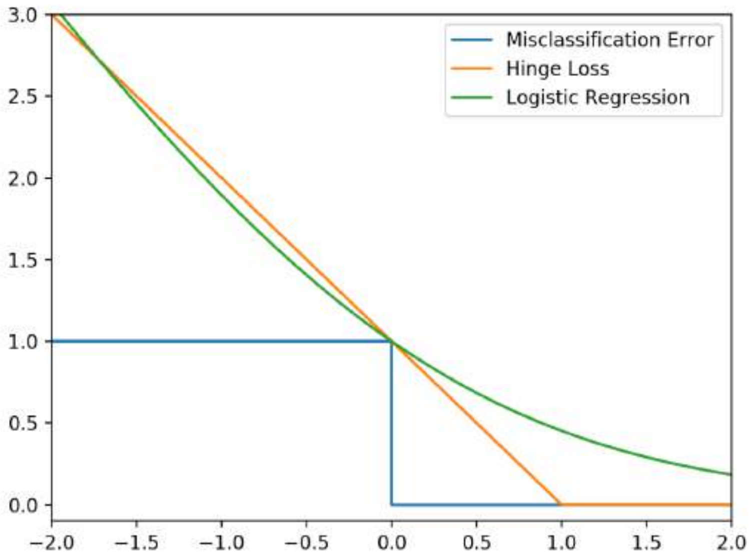
- Regularized error function

$$\frac{1}{2}\boldsymbol{w}^\top \boldsymbol{w} + C\sum_t \xi_t \quad \leftrightarrow \sum_t \mathcal{E}\big(y_t, f(x_t)\big) + \lambda||\boldsymbol{w}||^2$$

  - Where error is defined by the slack variable

$$\mathcal{E}\big(y_t, f(x_t)\big) = \xi_t = \begin{cases} 0 & \text{if } y_t f(x_t) \geq 1 \\ 1 - y_t f(x_t) & \text{otherwise} \end{cases}$$

    - We call this **hinge loss**
  - The zero part makes the solution sparse



- Logistic regression (in comparison)

$$p(y=1|f(x)) = \sigma(f(x))$$

$$p(y=-1|f(x)) = 1 - \sigma(f(x))$$

$$1 - \sigma(f(x)) = 1 - \frac{1}{1+\exp(-f(x))} = \frac{1+\exp(-f(x)) - 1}{1+\exp(-f(x))}$$

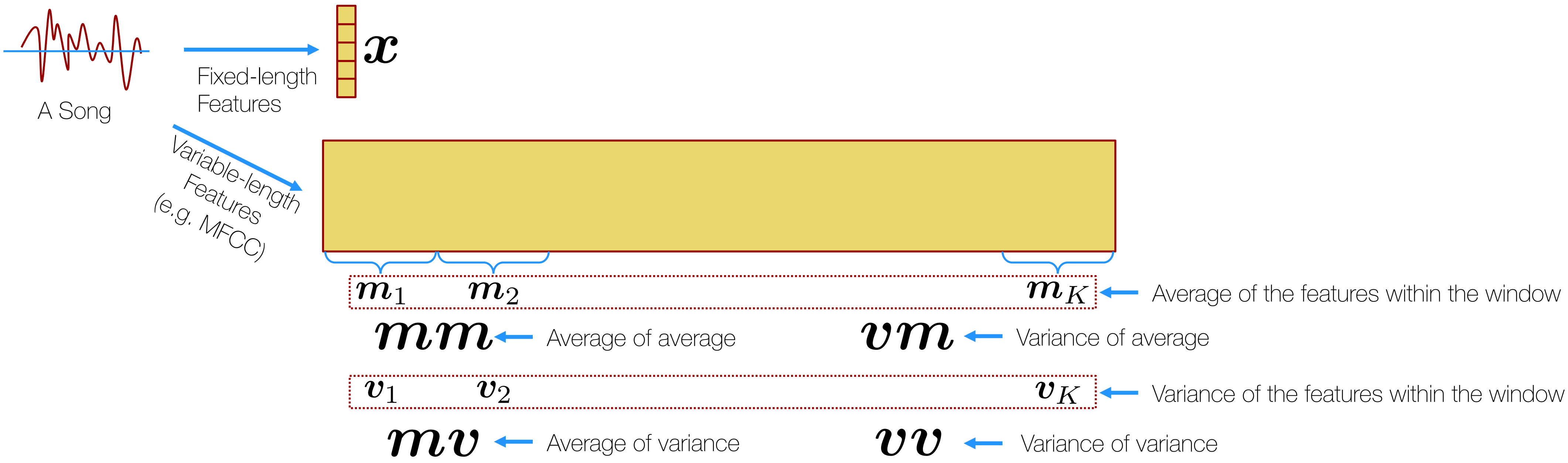
$$= \frac{\exp(-f(x))\exp(f(x))}{\exp(f(x)) + \exp(-f(x))\exp(f(x))} = \frac{1}{\exp(f(x)) + 1}$$

  - By the way, $1 - \sigma(f(x)) = \sigma(-f(x))$
  - Therefore $p(y|f(x)) = \sigma(yf(x))$
  - Negative log likelihood $-\log p\big(y|f(x)\big) = -\log \sigma\big(yf(x)\big) = \log\Big(1 + \exp\big(y(fx)\big)\Big)$
- We don't want to include correct classified examples in the loss

Music Classification

- MARSYAS

- Genre classification



- The number of features: $|\boldsymbol{x}| + |\boldsymbol{mm}| + |\boldsymbol{vm}| + |\boldsymbol{mv}| + |\boldsymbol{vv}|$
- This sliding windowing technique has been used extensively for music classification
  - e.g. mood classification

Music Classification

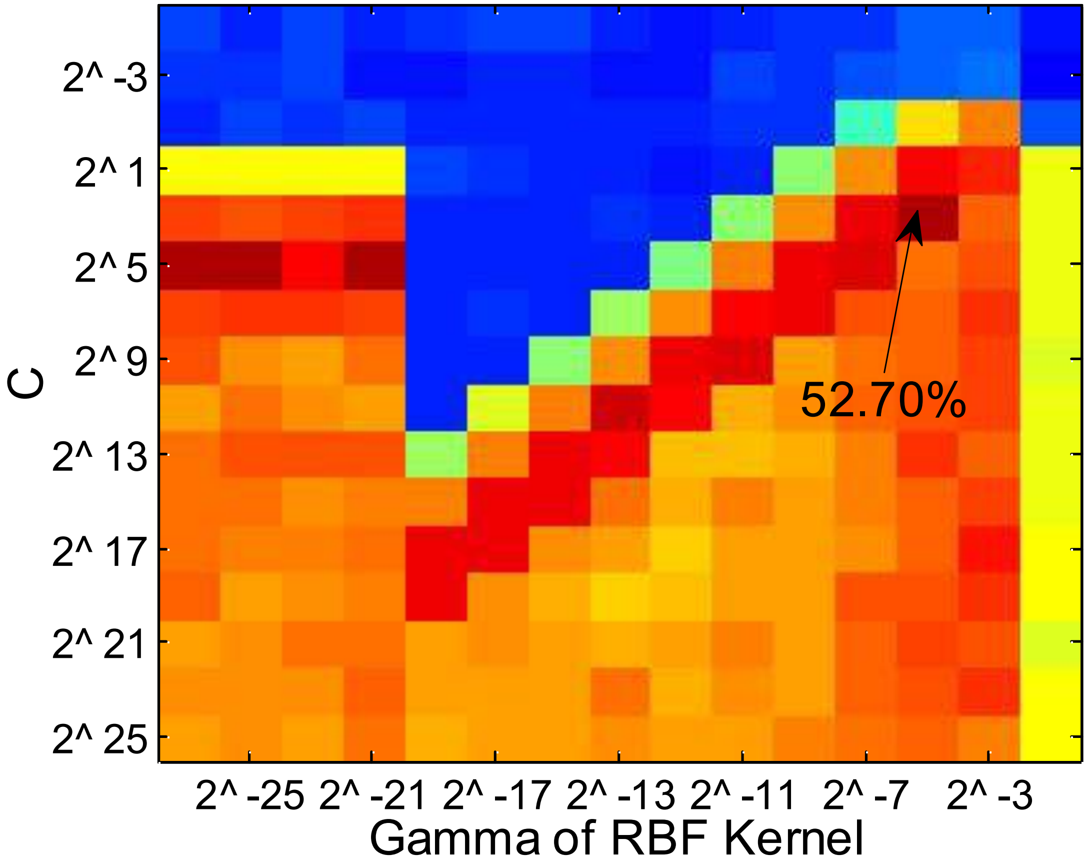
- SVM hyperparameter search

- ○ N-fold Cross validation
  - □ Divide the training set into N exclusive subsets

Frames

	Features		
1st fold	Train	Train	Test
2nd fold	Train	Test	Train
3rd fold	Test	Train	Train

  - □ N different train-validation pairs
  - □ Each pair is used to train a classifier and to evaluate it
  - □ Average the N results
  - □ The average shows the performance of your choice
- ○ Usually there are many combinations to try out



Recap

- We want the hyperplane that best discriminates the two classes
  - fLDA can linearly do this for Gaussian generative models
- No generative models
  - Boundary can be affected by noise
  - Needs slack variables and soft margins to handle this
- Sparse: focuses on the examples that count
  - Need to see all data anyway
- Naturally based on the kernels with its dual representation
  - Easy to build a nonlinear version

Reading

- Textbook Chapter 3 and Chapter 4.18
- Haste et. al, "The Elements of Statistical Learning," Chapter 4.3.3
- C. Bishop, "Pattern Recognition and Machine Learning," Chapter 7

Thank You!