Using Zoom for Lectures

Sign in using:

your name

Please mute both:

your video cameras for the entire lecture

your audio/mics unless asking or answering a question

Asking/answering a question, option 1:

click on Participants

use the hand icon to raise your hand

I will call on you and ask you to unmute yourself

Asking/answering a question, option 2:

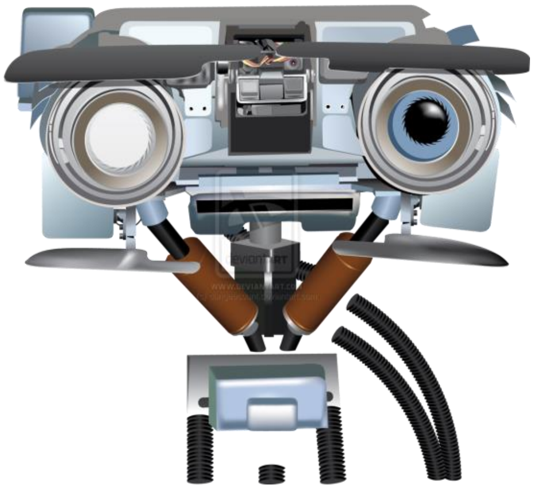
click on Chat

type your question, and I will answer it

Today: Outline

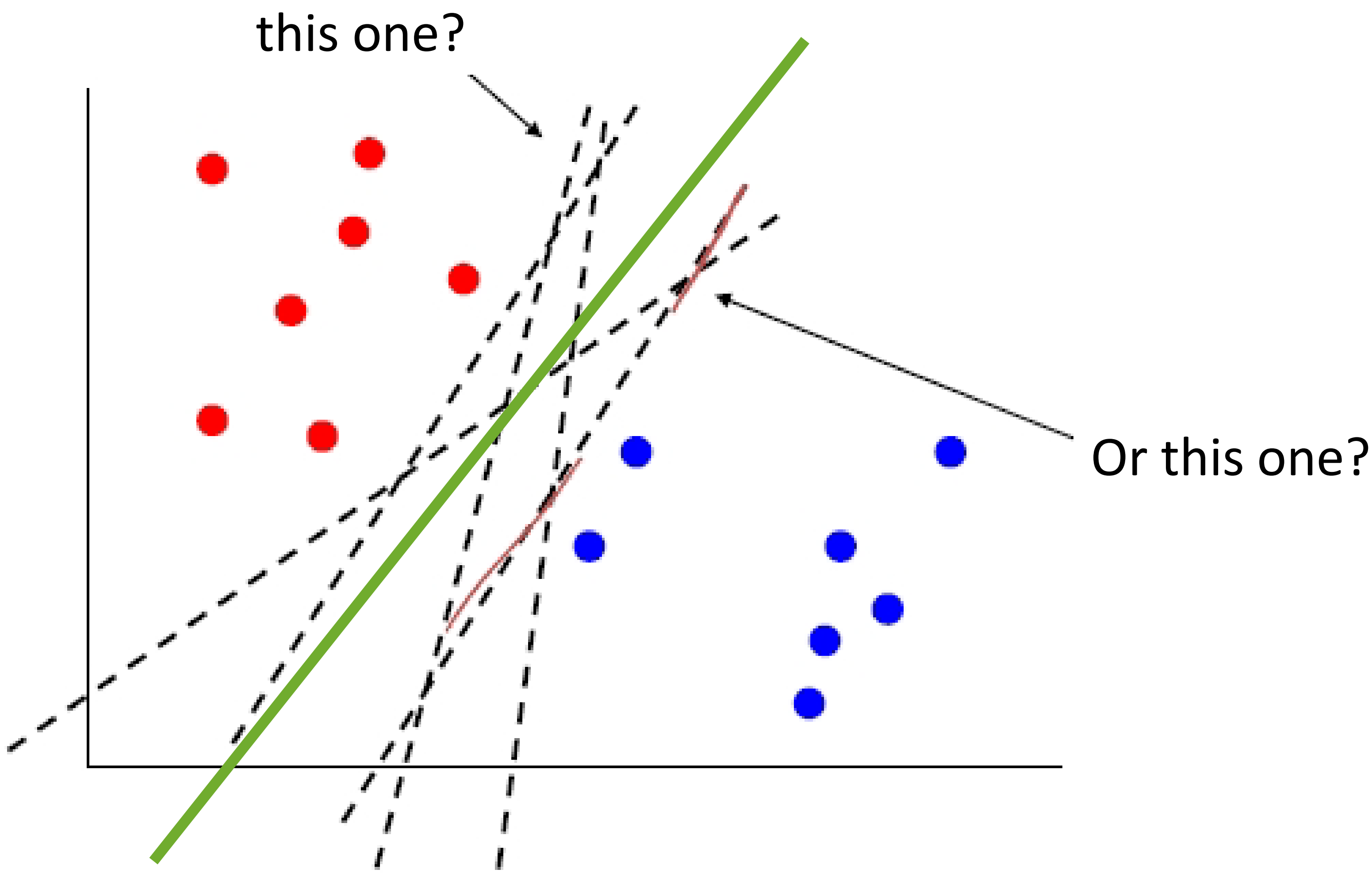
Support Vector Machines cont'd

Reminder: PS3 Self Score due Mar 23



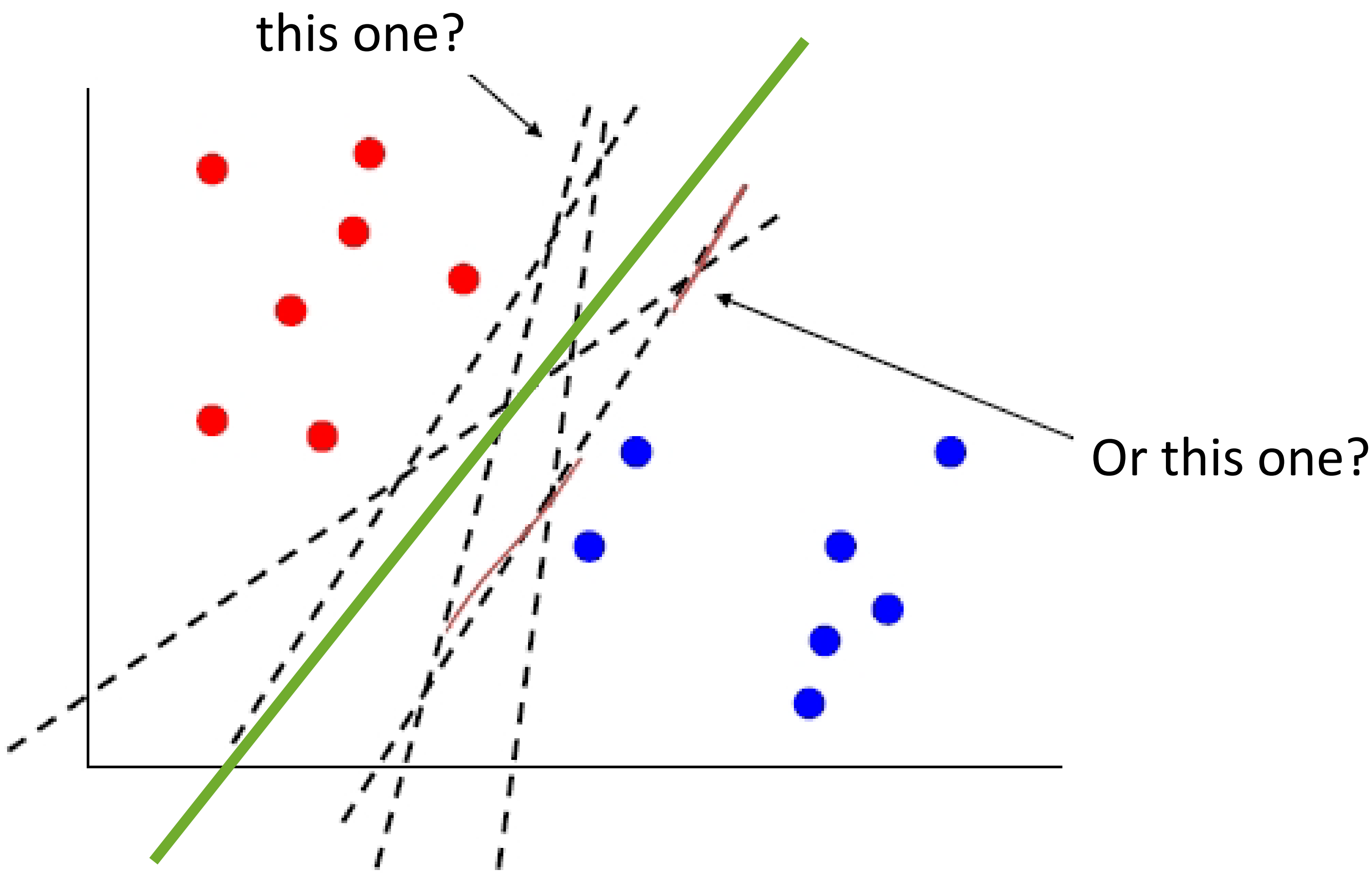
Maximum Margin

How about the one in the middle?



Intuitively, this classifier avoids misclassifying new test points generated from the same distribution as the training points

What is special about the green line?



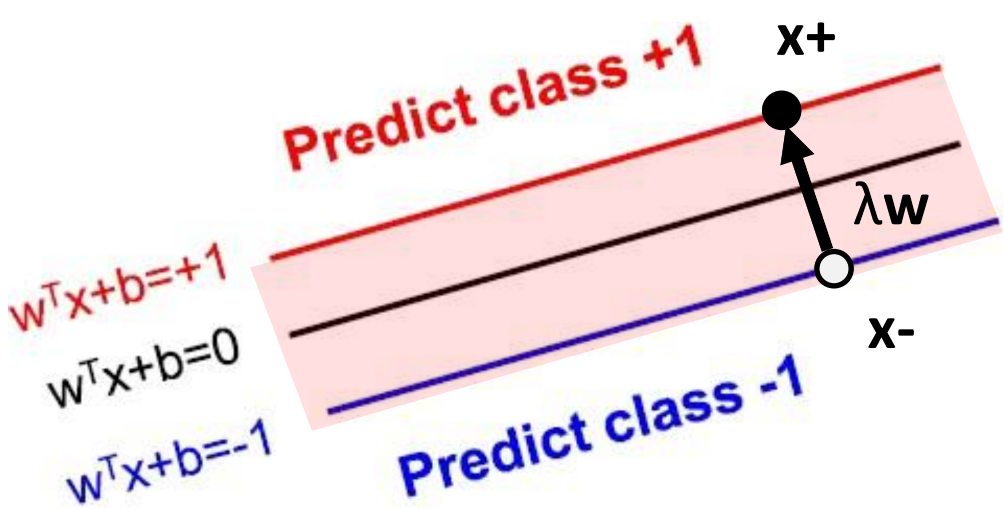
It maximizes the margin between the two classes.

Computing the Margin

Define the margin M to be the distance between the +1 and -1 planes

We can now express this in terms of **w** →

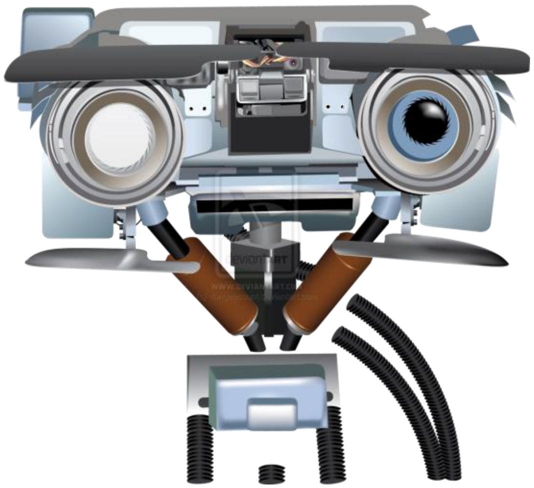
to maximize the margin we minimize the length of **w**



$$M = \left\| \mathbf{x}^+ - \mathbf{x}^- \right\|$$

$$= \left\| \lambda \mathbf{w} \right\| = \lambda \sqrt{\mathbf{w}^T \mathbf{w}}$$

$$= 2 \frac{\sqrt{\mathbf{w}^T \mathbf{w}}}{\mathbf{w}^T \mathbf{w}} = \frac{2}{\sqrt{\mathbf{w}^T \mathbf{w}}}$$

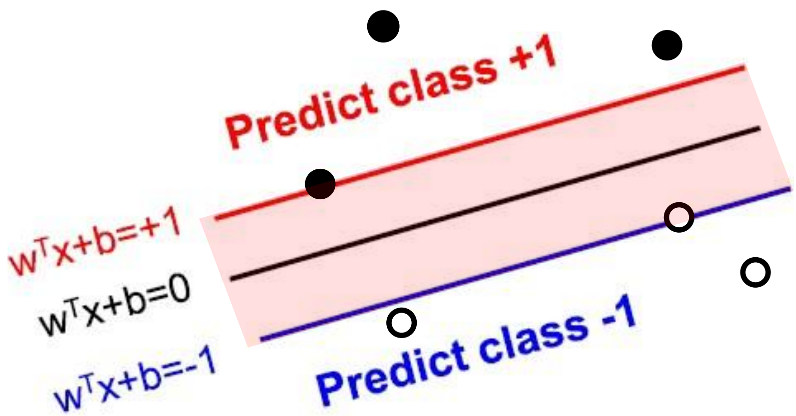


Linear SVM

Linear SVM Formulation

We can search for the optimal parameters (**w** and b) by finding a solution that:

1. Correctly classifies the training examples: $\{x_i, y_i\}$, i=1,..,n
2. Maximizes the margin (same as minimizing $\|\boldsymbol{w}\|^2$)



$$\min \ \frac{1}{2}\|\mathbf{w}\|^2$$

$$s.t. \ (\mathbf{w}^T\mathbf{x}_i + b)y_i \geq 1 \ \ \forall i$$

This is the primal formulation

Apply Lagrange multipliers: formulate equivalent problem

Lagrange Multipliers

Convert the primal constrained minimization to an unconstrained optimization problem: represent constraints as penalty terms:

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|^2 + \boldsymbol{\mathit{penalty\_term}}$$

For data $\{(x_i,y_i)\}$ use the following penalty term:

$$\begin{Bmatrix} 0 & \text{if } (\mathbf{w}^T\mathbf{x}_i + b)y_i \geq 1 \\ \infty & \text{otherwise} \end{Bmatrix} = \max_{\alpha_i \geq 0} \alpha_i [1 - (\mathbf{w}^T\mathbf{x}_i + b)y_i]$$

≤ 0 if constraint satisfied

Introduced Lagrange variables $\alpha_i \geq 0$; find ones that maximize term:

- If a constraint is satisfied, large α_i ensures smaller penalty
- If a constraint is violated, large α_i ensures larger penalty

Note, we are now minimizing with respect to **w** and b, and maximizing with respect to $\boldsymbol{\alpha}$ (additional parameters)

Lagrange Multipliers

Convert the primal constrained minimization to an unconstrained optimization problem: represent constraints as penalty terms:

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|^2 + \boldsymbol{penalty\_term}$$

For data $\{(x_i,y_i)\}$ use the following penalty term:

$$\begin{Bmatrix} 0 & \text{if } (\mathbf{w}^T\mathbf{x}_i + b)y_i \geq 1 \\ \infty & \text{otherwise} \end{Bmatrix} = \max_{\alpha_i \geq 0} \alpha_i[1-(\mathbf{w}^T\mathbf{x}_i + b)y_i]$$

Rewrite the minimization problem:

$$\min_{\mathbf{w},b} \{\frac{1}{2}\|\mathbf{w}\|^2 + \sum_{i=1}^{n} \max_{\alpha_i \geq 0} \alpha_i[1-(\mathbf{w}^T\mathbf{x}_i + b)y_i]\}$$

Where $\{\alpha_i\}$ are the Lagrange multipliers

$$\min_{\mathbf{w},b} \max_{\alpha_i \geq 0} \{\frac{1}{2}\|\mathbf{w}\|^2 + \sum_{i=1}^{n} \alpha_i[1-(\mathbf{w}^T\mathbf{x}_i + b)y_i]\}$$

Solution to Linear SVM

Swap the 'max' and 'min':

$$\max_{\alpha_i \geq 0} \min_{\mathbf{w},b} \{\frac{1}{2}\|\mathbf{w}\|^2 + \sum_{i=1}^{n} \alpha_i [1 - (\mathbf{w}^T \mathbf{x}_i + b) y_i]\}$$

$$= \max_{\alpha_i \geq 0} \min_{\mathbf{w},b} J(\mathbf{w}, b; \alpha)$$

First minimize J() w.r.t. {**w**,b} for any fixed setting of the Lagrange multipliers:

$$\frac{\partial}{\partial \mathbf{w}} J(\mathbf{w}, b; \alpha) = \mathbf{w} - \sum_{i=1}^{n} \alpha_i \mathbf{x}_i y_i = 0$$

$$\frac{\partial}{\partial b} J(\mathbf{w}, b; \alpha) = -\sum_{i=1}^{n} \alpha_i y_i = 0$$

Then substitute back into J() and simplify to get final optimization:

$$L = \max_{\alpha_i \geq 0} \{\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} y_i y_j \alpha_i \alpha_j (\mathbf{x}_i \cdot \mathbf{x}_j)\}$$

Dual Problem

Final optimization: maximize this loss over α_i's:

only dot products of data points needed

$$L = \max_{\alpha_i \geq 0} \{ \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} y_i y_j \alpha_i \alpha_j (\mathbf{x}_i \cdot \mathbf{x}_j) \}$$

subject to $\alpha_i \geq 0; \quad \sum_{i=1}^{n} \alpha_i y_i = 0$

Then use the obtained α_i's to solve for the weights and bias

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \qquad b = y_i - \mathbf{w}^{\mathrm{T}} \mathbf{x}_{\mathrm{i}} \quad \forall i$$

Dual vs Primal SVM

n is the number of training points, d is dimension of $\mathbf{x}, \mathbf{w}$

Primal problem: for $\mathbf{w} \in \mathbb{R}^{\boldsymbol{d}}$, hyperparameter C, the unconstrained version is

$$\min \frac{1}{2}\|\mathbf{w}\|^2 \quad s.t.\ (\mathbf{w}^T\mathbf{x}_i + b)y_i \geq 1$$

Dual problem: for $\boldsymbol{\alpha} \in \mathbb{R}^{\boldsymbol{n}}$

$$L = \max_{\alpha_i \geq 0}\{\sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{n} y_i y_j \alpha_i \alpha_j (\mathbf{x}_i \cdot \mathbf{x}_j)\} \quad s.t.\ \alpha_i \geq 0;\ \sum_{i=1}^{n}\alpha_i y_i = 0$$

- Efficiency: need to learn d parameters for primal, n for dual
- Dual form only involves data terms $\mathbf{x}_i^{\mathbf{T}}\mathbf{x}_{\mathbf{j}}$

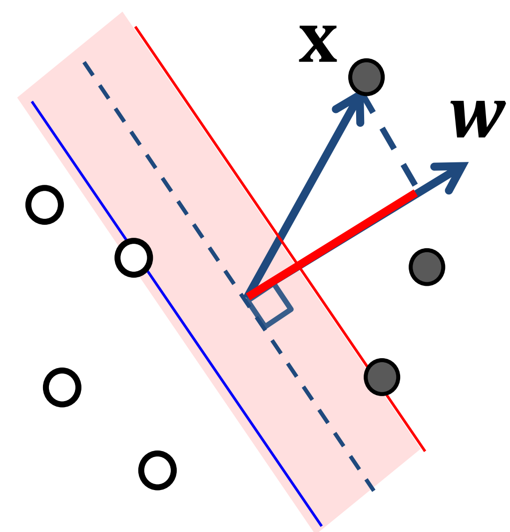
Prediction on Test Example

Now we have the solution for the weights and bias

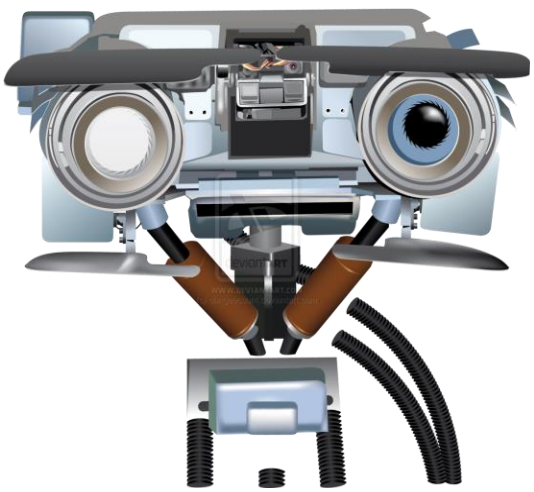
$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \qquad\qquad b = y_i - \mathbf{w}^{\mathrm{T}}\mathbf{x}_{\mathrm{i}} \quad \forall i$$

Given a new input example $\mathbf{x}$, classify it as

$+1$ if $\mathbf{w}^{\mathrm{T}}\mathbf{x} + b \geq 1$, or
-1 if $\mathbf{w}^{\mathrm{T}}\mathbf{x} + b \leq -1$



In practice, predict $y = \mathrm{sign}[\mathbf{w}^{\mathrm{T}}\mathbf{x} + b]$



Soft-margin SVM

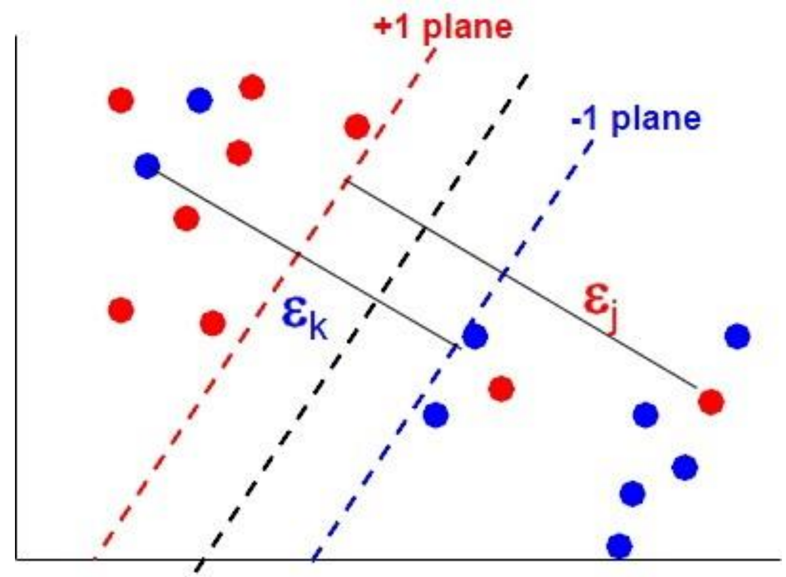
What if the data is not linearly separable?

What if data is not linearly separable?

- We will end up with a constraint satisfaction problem that is NOT satisfiable.

- What do we do? Relaxation.

- *i.e.* We will allow for some violation of constraints, but will minimize how that happens.



- We ask: How far is the bad sample from its margin?

What if data is not linearly separable?

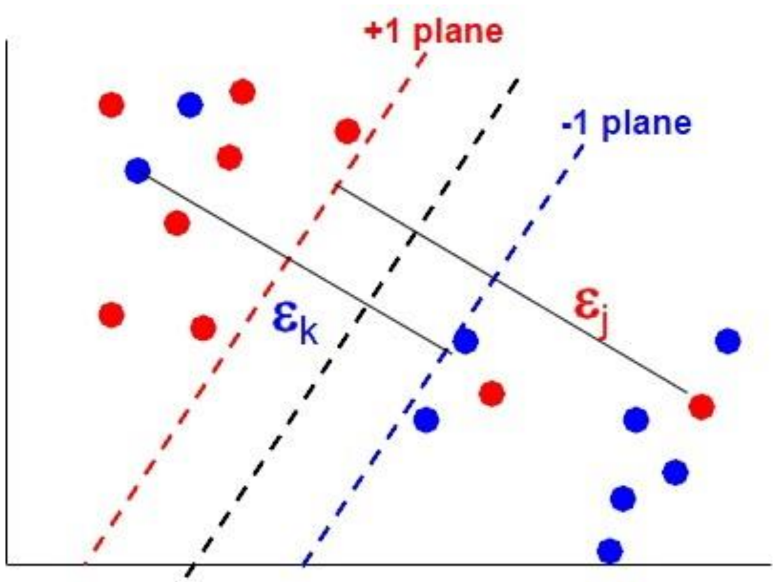
- Introduce slack variables ξ_i

$$\min\left[\frac{1}{2}\|\mathbf{w}\|^2 + \lambda \sum_{i=1}^{n} \xi_i\right]$$

subject to constraints (for all *i*):

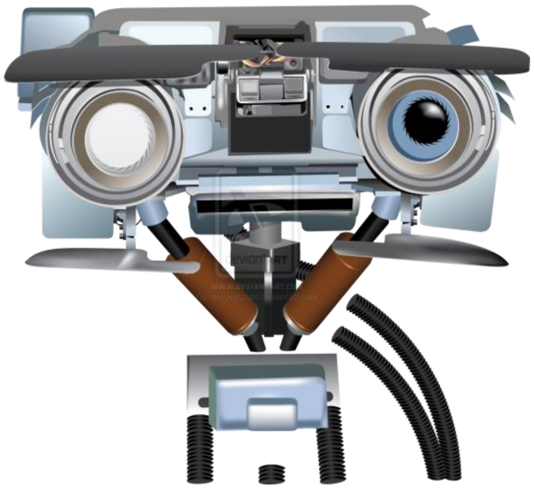
$$y_i(\mathbf{w}\cdot\mathbf{x}_i + b) \geq 1 - \xi_i$$

$$\xi_i \geq 0$$



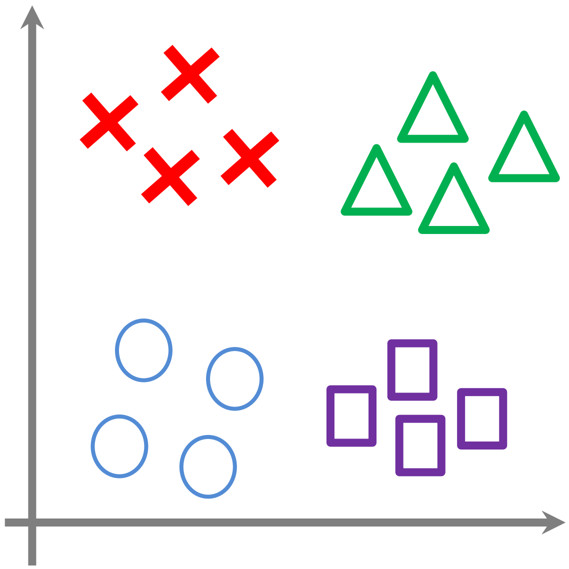
- Example lies on wrong side of hyperplane: $\xi_i > 1 \Rightarrow \sum_i \xi_i$ is upper bound on number of training errors

- This is known as the soft-margin extension



Multi-class SVMs

Multi-class classification

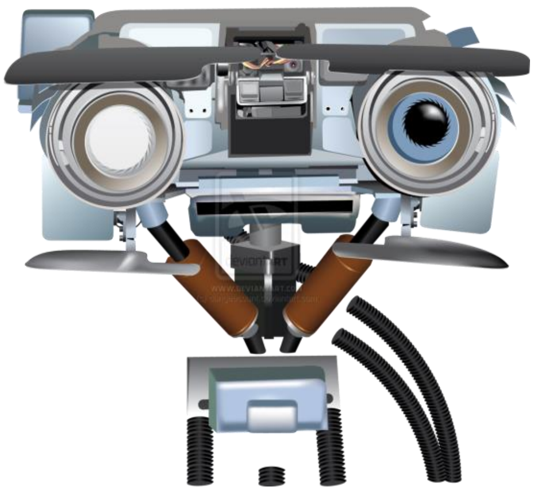


$$y \in \{1, 2, 3, \ldots, K\}$$

Many SVM packages already have built-in multi-class classification functionality.

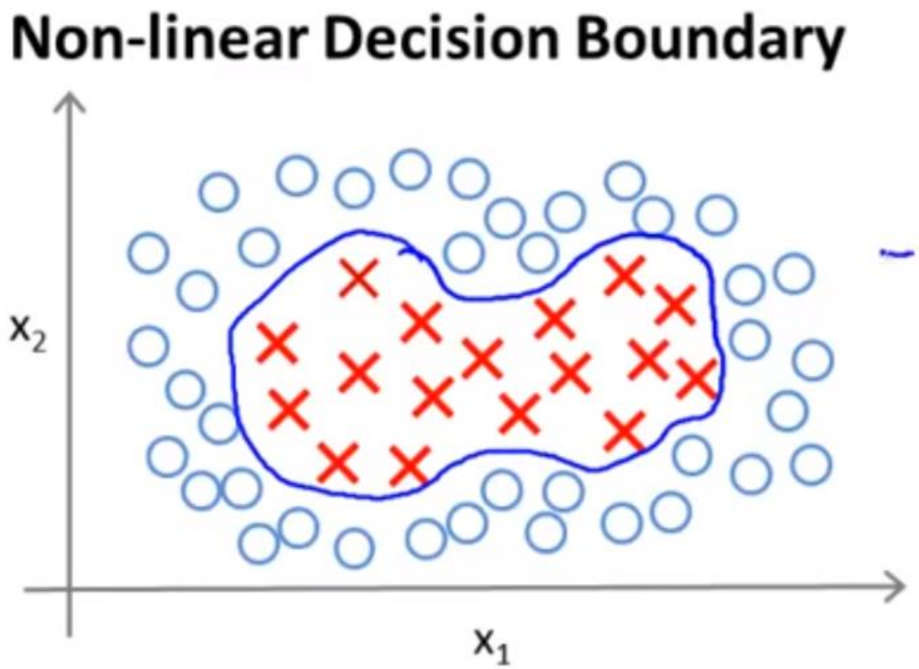
Otherwise, use one-*vs.*-all method. (Train K SVMs, one to distinguish class i from the rest), for $i = 1, \ldots, K$, get $\mathbf{w}^{(1)}, b^{(1)}, \ldots, \mathbf{w}^{(K)}, b^{(K)}$

Pick class $y = i$ with largest score ${\mathbf{w}^{(\mathrm{i})}}^T \boldsymbol{x} + b^{(i)}$



Kernel SVM

Complex Non-linear Boundary



f_1

Predict $y = 1$ if

$$\theta_0 + \theta_1 \boxed{x_1} + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 + \theta_5 x_2^2 + \cdots \geq 0$$

Is there a better choice of features than these high order polynomials?

Non-linear decision boundaries

- Note that both the learning objective and the decision function depend only on dot products between patterns

$$L = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} y_i y_j \alpha_i \alpha_j (\mathbf{x}_i \cdot \mathbf{x}_j) \qquad y = \mathrm{sign}[b + \mathbf{x} \cdot (\sum_{i=1}^{n} y_i \alpha_i \mathbf{x}_i)]$$

- How to form non-linear decision boundaries in input space?

- Basic idea:

1. Map data into feature space $\mathbf{x} \rightarrow \phi(\mathbf{x})$

2. Replace dot products between inputs with feature points

$$\mathbf{x}_i \cdot \mathbf{x}_j \rightarrow \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$$

3. Find linear decision boundary in feature space

- Problem: what is a good feature function φ(**x**)?

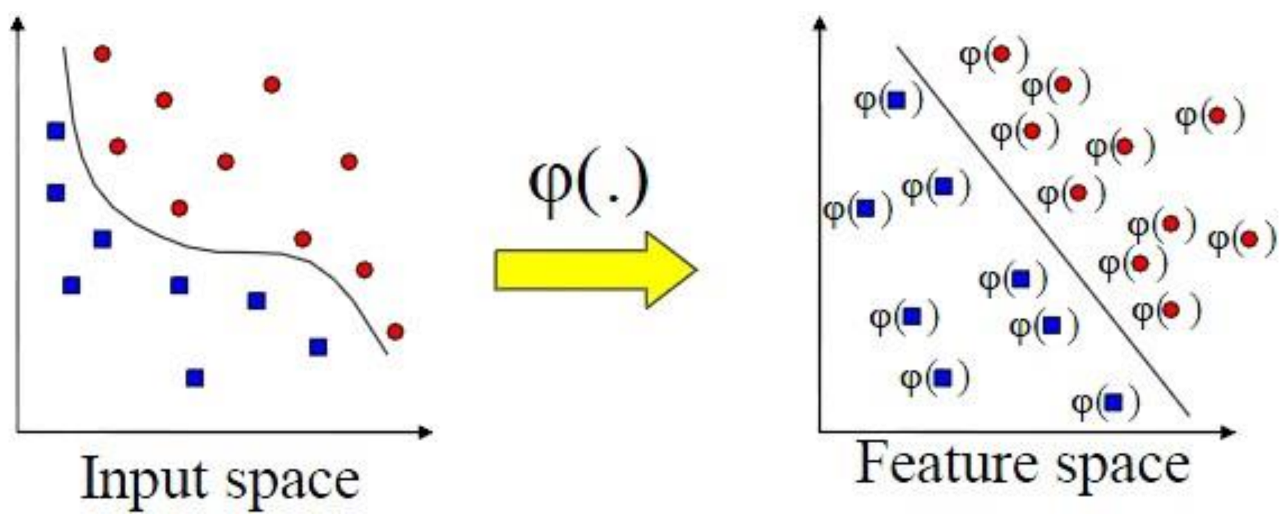
Input transformation

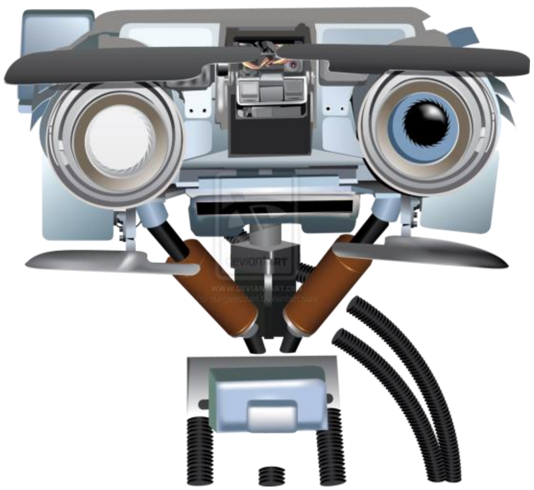
Mapping to a feature space can produce problems:

- High computational burden due to high dimensionality
- Many more parameters

SVM solves these two issues simultaneously

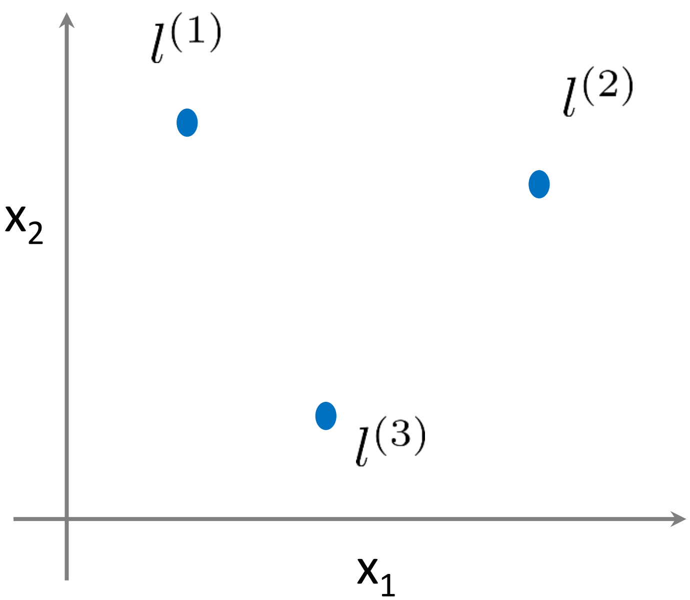
- Kernel trick produces efficient classification
- Dual formulation only assigns parameters to samples, not features





Kernels

Kernels and Similarity



Given x, compute new feature depending on proximity to landmarks $l^{(1)}, l^{(2)}, l^{(3)}$

Example: Gaussian kernel

$$f_1 = similarity(x, l^{(1)}) = \exp\left(-\frac{\left\| x - l^{(1)} \right\|^2}{2\sigma^2}\right)$$

If $x \approx l^{(1)}$:

similarity is high

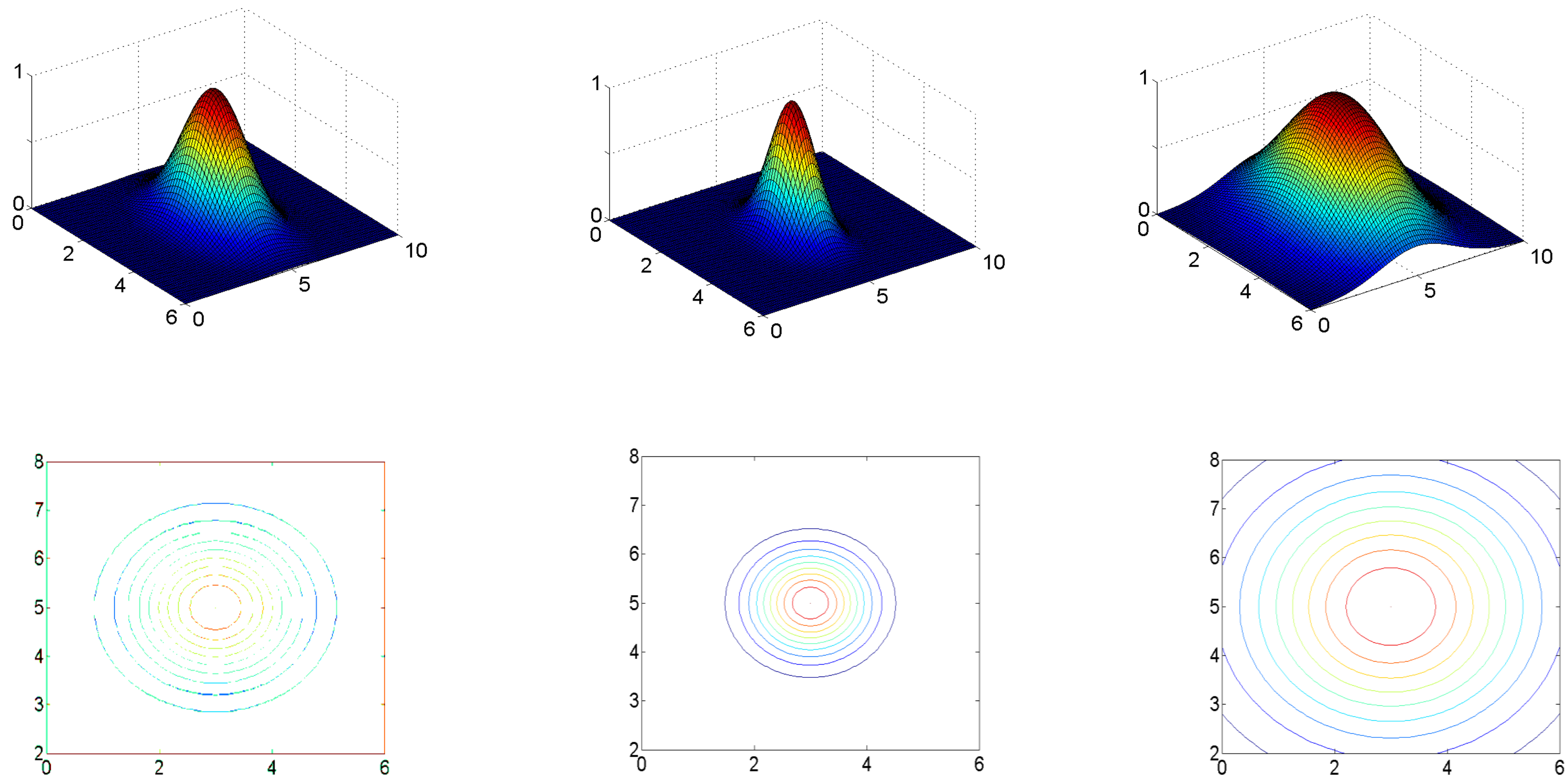
If x if far from $l^{(1)}$:

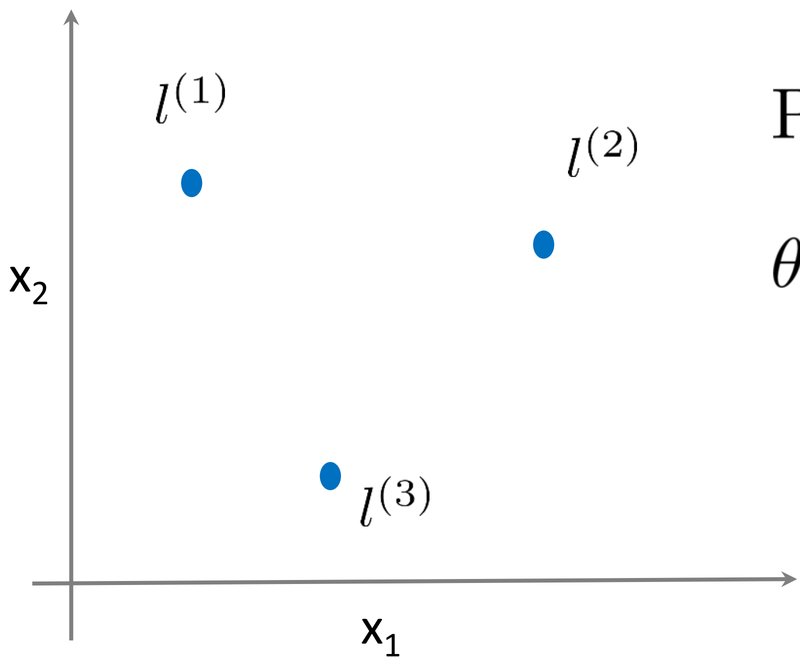
similarity is low

Example:

$$l^{(1)} = \begin{bmatrix} 3 \\ 5 \end{bmatrix}, \quad f_1 = \exp\left(-\frac{\|x - l^{(1)}\|^2}{2\sigma^2}\right)$$

$\sigma^2 = 1$ $\qquad$ $\sigma^2 = 0.5$ $\qquad$ $\sigma^2 = 3$

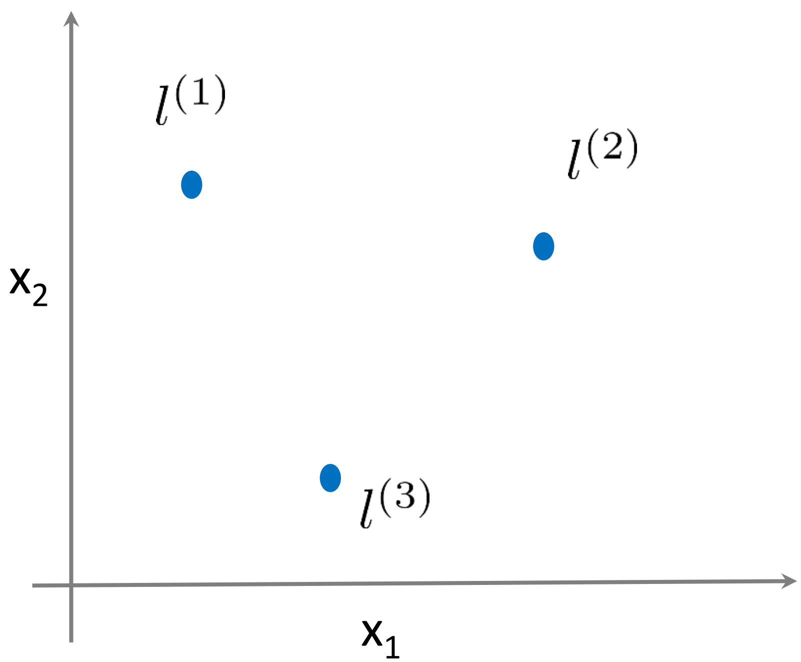




Predict "1" when

$$\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 \geq 0$$

Choosing the landmarks

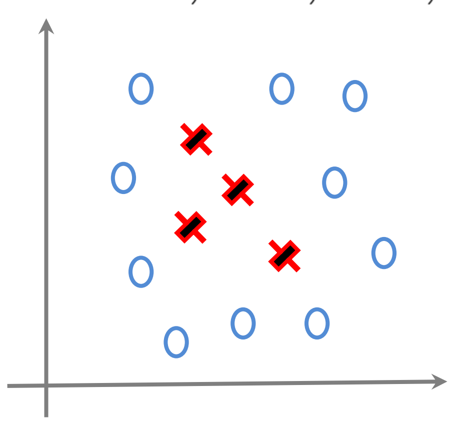


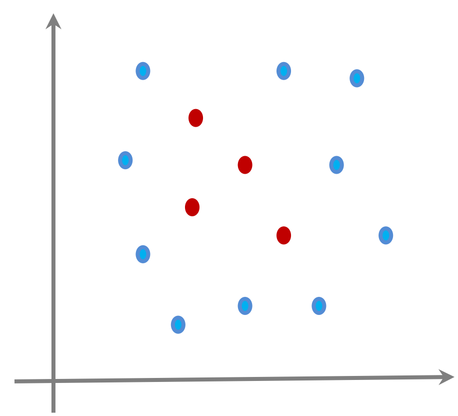
Given x:

$$f_i = \text{similarity}(x, l^{(i)})$$
$$= \exp\left(-\frac{||x - l^{(i)}||^2}{2\sigma^2}\right)$$

Predict $y = 1$ if $\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 \geq 0$

Where to get $l^{(1)}, l^{(2)}, l^{(3)}, \ldots$?





SVM with Kernels

Given $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})$,
choose $l^{(1)} = x^{(1)}, l^{(2)} = x^{(2)}, \ldots, l^{(m)} = x^{(m)}$.

Given example x:

$$f_1 = \text{similarity}(x, l^{(1)})$$
$$f_2 = \text{similarity}(x, l^{(2)})$$
$$\ldots$$

Kernels

Examples of kernels (kernels measure similarity):

1. Polynomial $K(\mathbf{x}_1, \mathbf{x}_2) = (\mathbf{x}_1 \cdot \mathbf{x}_2 + 1)^2$
2. Gaussian $K(\mathbf{x}_1, \mathbf{x}_2) = \exp(-\left\|\mathbf{x}_1 - \mathbf{x}_2\right\|^2 / 2\sigma^2)$
3. Sigmoid $K(\mathbf{x}_1, \mathbf{x}_2) = \tanh(\kappa(\mathbf{x}_1 \cdot \mathbf{x}_2) + a)$

Each kernel computation corresponds to dot product calculation for particular mapping φ(x): implicitly maps to high-dimensional space

Why is this useful?

1. Rewrite training examples using more complex features
2. Dataset not linearly separable in original space may be linearly separable in higher dimensional space

Classification with non-linear SVMs

Non-linear SVM using kernel function *K()*:

$$L_K = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} y_i y_j \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j)$$

Maximize L_K w.r.t. {α}, under constraints α≥0

Unlike linear SVM, cannot express w as linear combination of support vectors – now must retain the support vectors to classify new examples

Final decision function:

$$y = \mathrm{sign}[b - \sum_{i=1}^{n} y_i \alpha_i K(\mathbf{x}, \mathbf{x}_i)]$$

Kernel SVM Summary

Advantages:

- Kernels allow very flexible hypotheses
- Poly-time exact optimization methods rather than approximate methods
- Soft-margin extension permits mis-classified examples
- Excellent results (1.1% error rate on handwritten digits vs. LeNet's 0.9%)

Disadvantages:

- Must choose kernel parameters
- Very large problems computationally intractable

Kernelizing

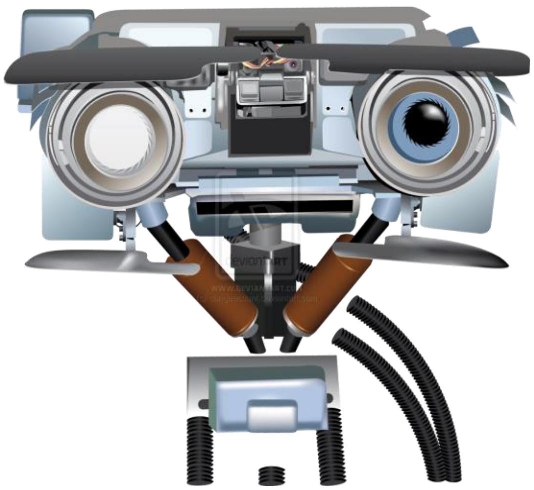
A popular way to make an algorithm more powerful is to develop a kernelized version of it

- We can rewrite a lot of algorithms to be defined only in terms of inner product

- For example: k-nearest neighbors

$$\mathbf{z} = \varphi(\mathbf{x})$$

$$(\mathbf{z}_i - \mathbf{z}_j)^2 = K(\mathbf{x}_i, \mathbf{x}_i) + K(\mathbf{x}_j, \mathbf{x}_j) - 2K(\mathbf{x}_i, \mathbf{x}_j)$$



Summary of SVMs

Summary

Software:

- A list of SVM implementations can be found at http://www.kernel-machines.org/software.html
- Some implementations (such as LIBSVM) can handle multi-class classification
- SVMLight is among the earliest implementations
- Several Matlab toolboxes for SVM are also available

Key points:

- Difference between logistic regression and SVMs
- Maximum margin principle
- Slack variables for mis-classified points
- Kernel trick allows non-linear generalizations

Logistics

- Pre-lecture material for next lecture
- Next lecture is given by Katia on using the SCC cluster (recorded lecture, not live)
- Walk through Piazza
- Examinations and other concerns