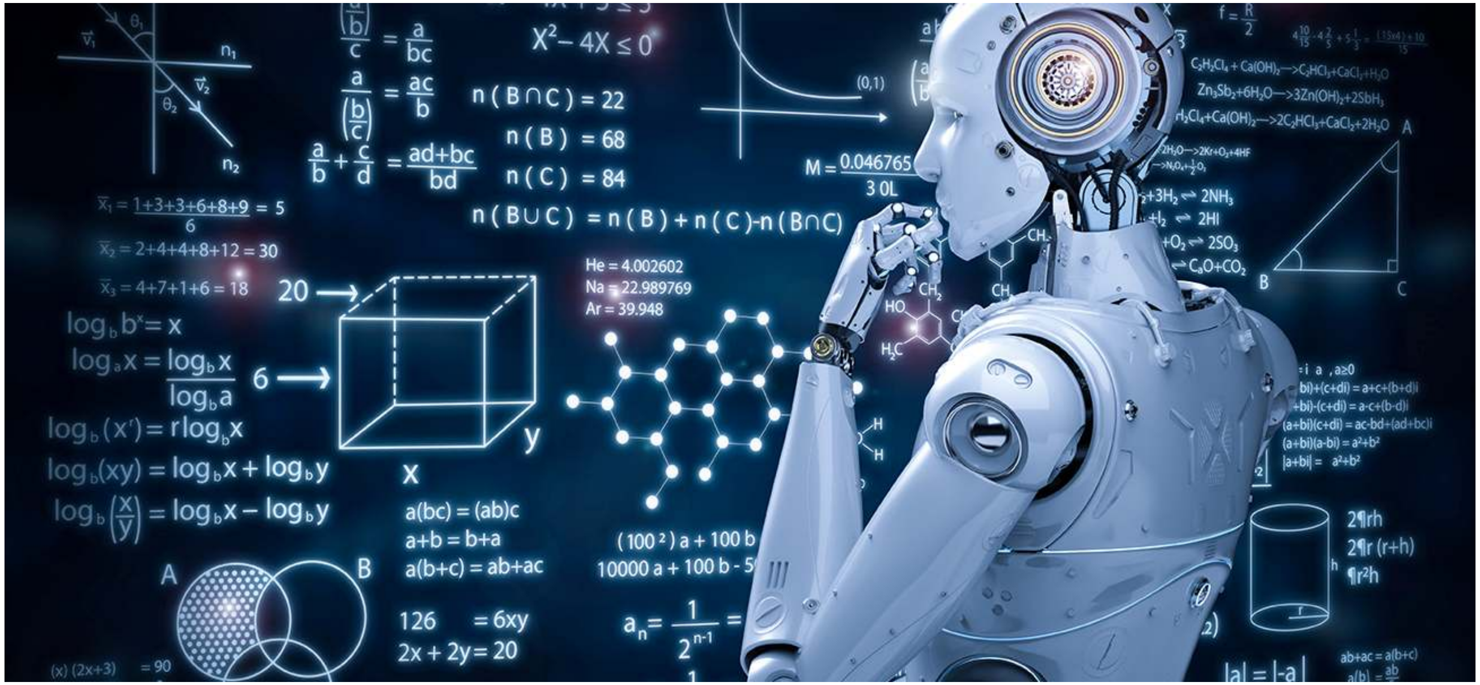
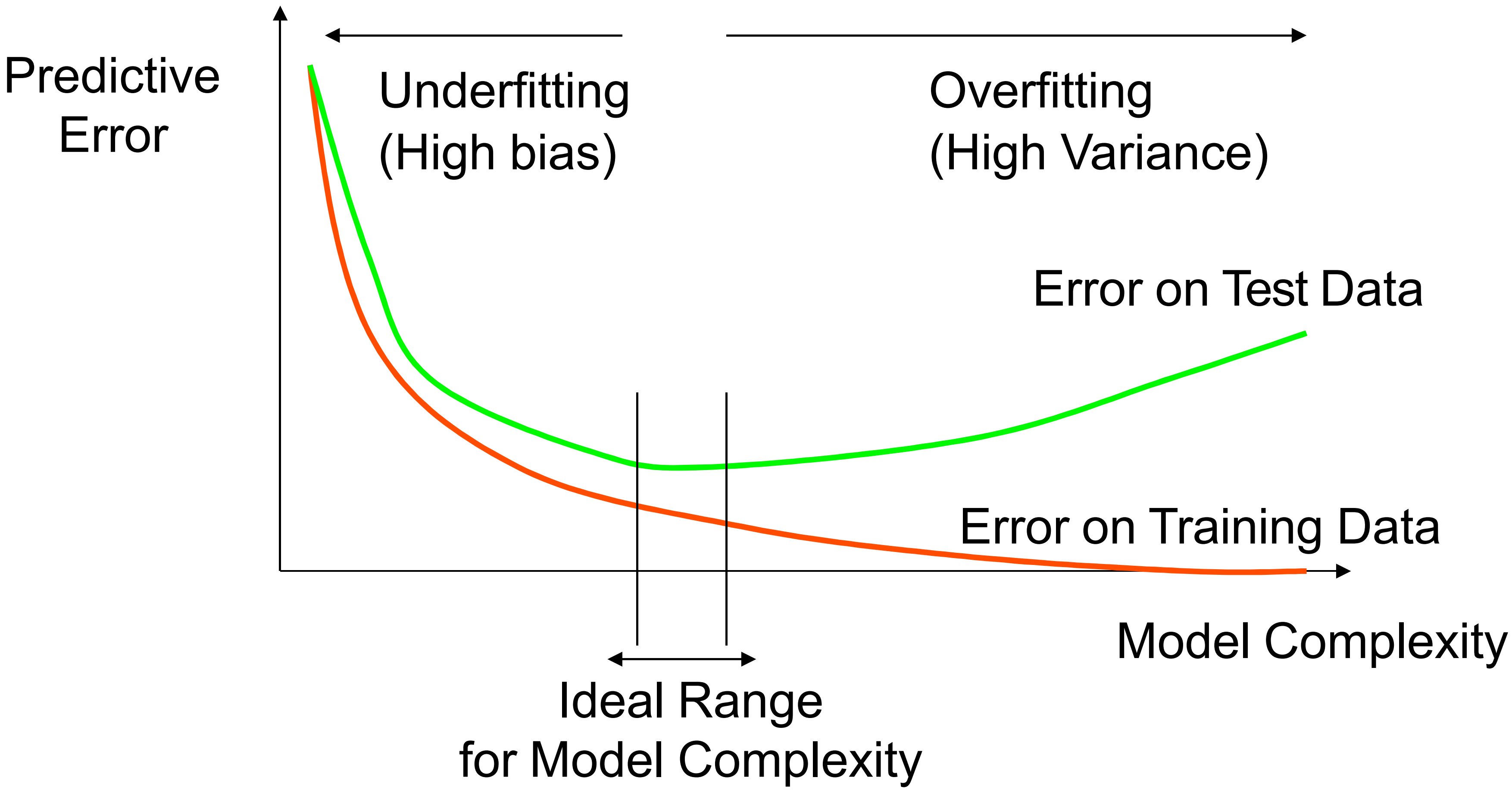


BBM406: Fundamentals of Machine Learning

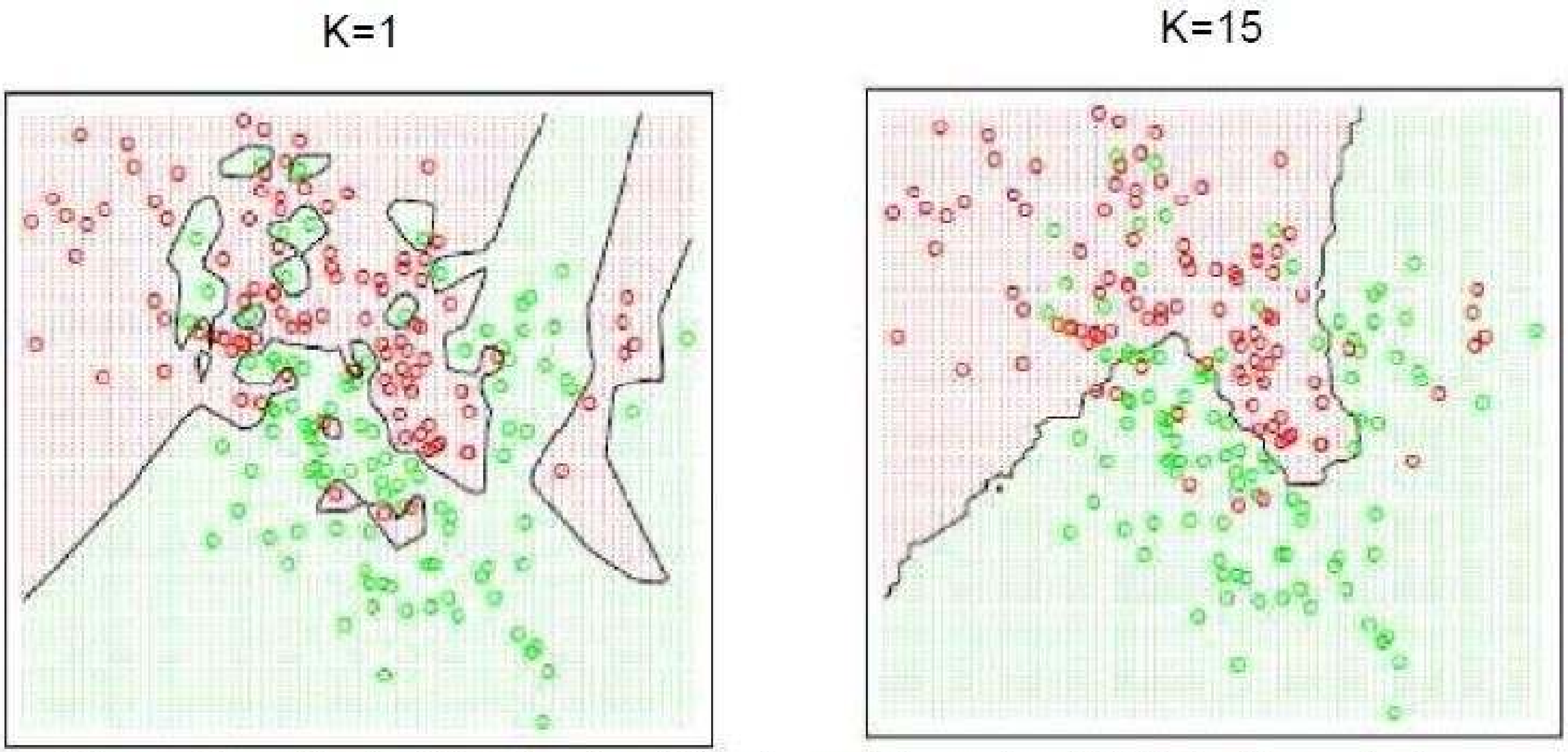
Support Vector Machines

Bias vs. Variance Trade off



- High bias (and low variance) indicates underfitting problem
- High variance (and low bias) indicates underfitting problem

Bias vs. Variance Trade off



Figures from Hastie, Tibshirani and Friedman (Elements of Statistical Learning)

Overfitting
(High Variance)

Underfitting
(High bias)

Regularization

- Linear Regression Cost Function

$$J(\theta) = \frac{1}{2n}\sum_{i=1}^{n}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

- Regularized Linear Regression Cost Function

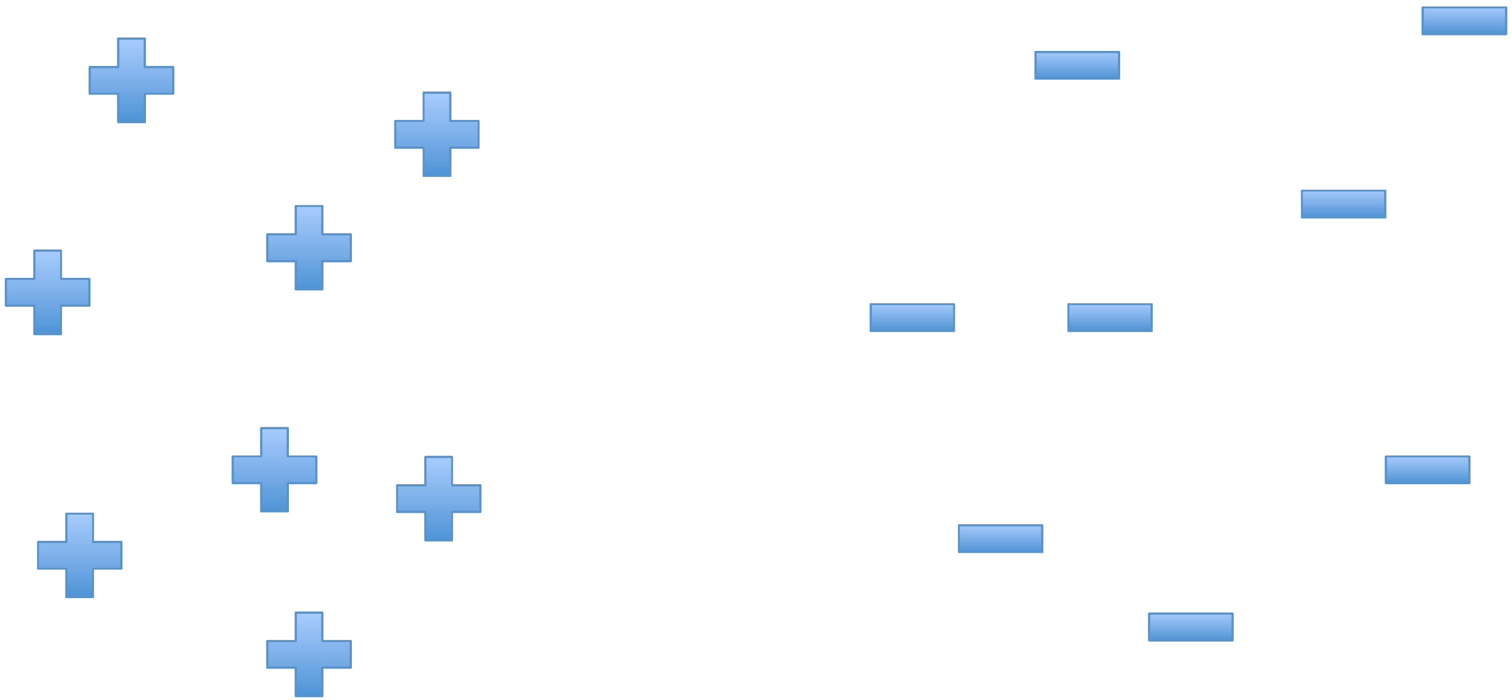
$$J(\theta) = \frac{1}{2n}\sum_{i=1}^{n}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{d}\theta_j^2$$

Regularization Parameter

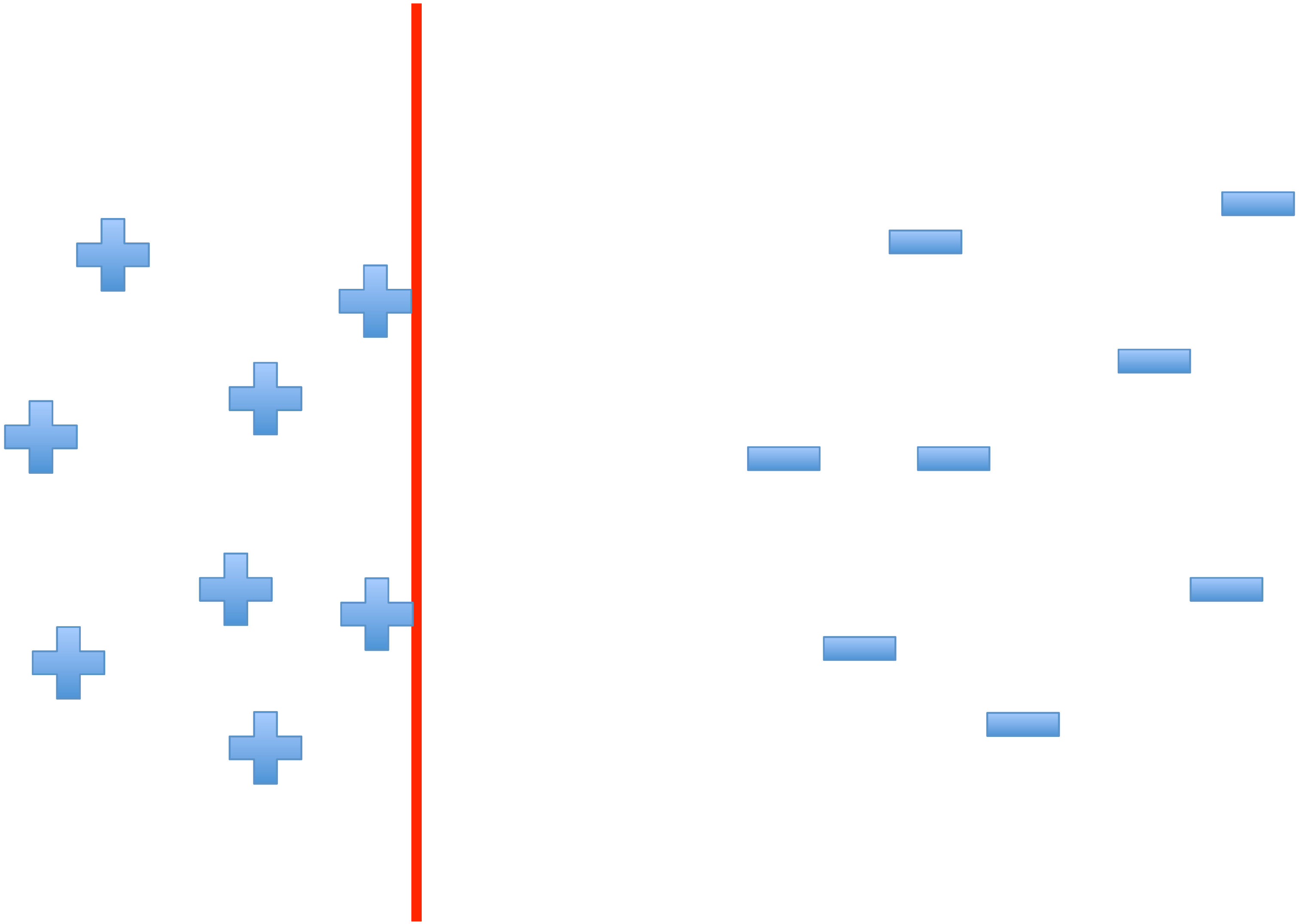
Strengths of SVMs

- Good generalization
  - in theory
  - in practice
- Works well with few training instances
- Find globally best model
- Efficient algorithms
- Amenable to the kernel trick

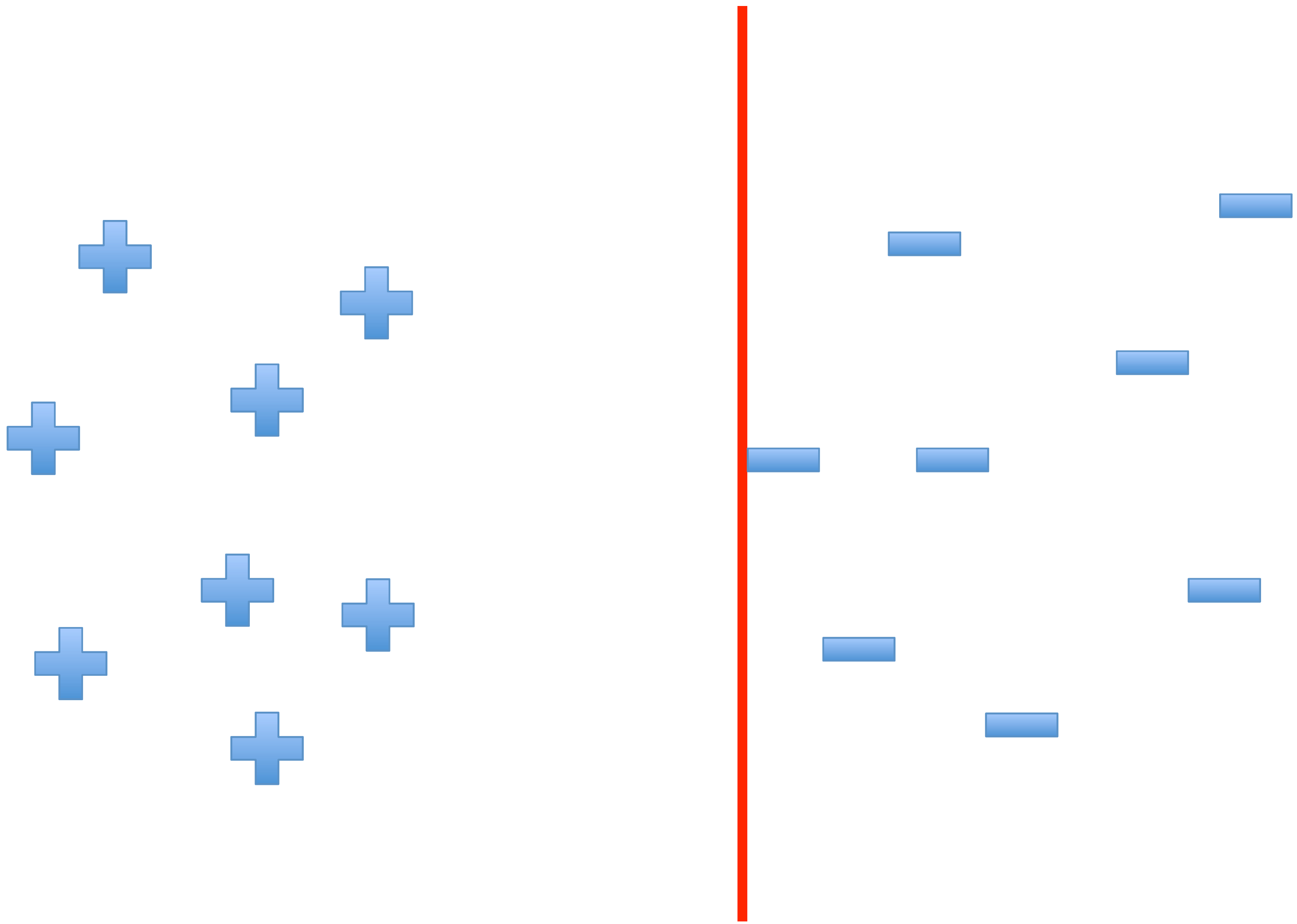
Intuitions



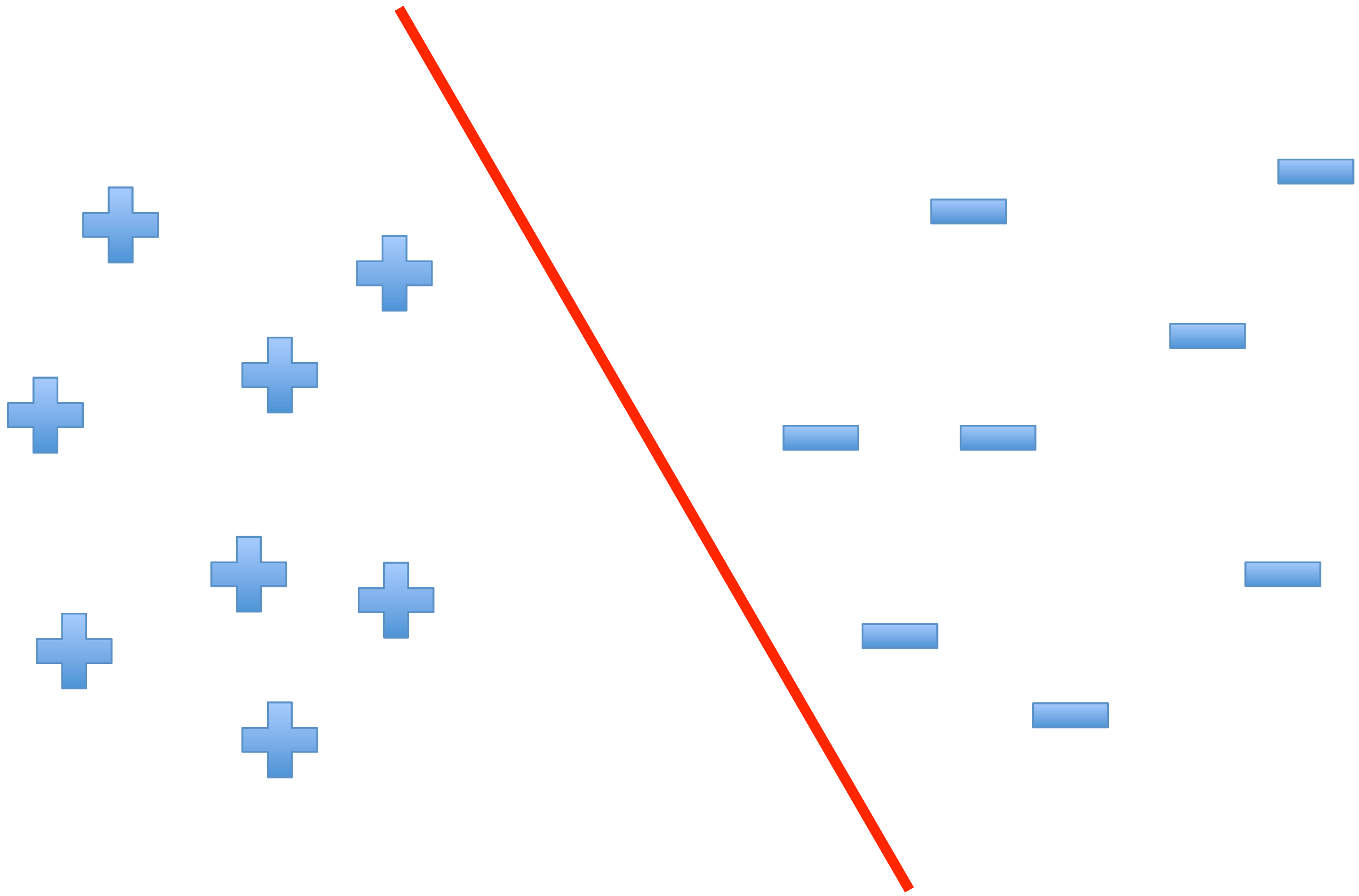
Intuitions



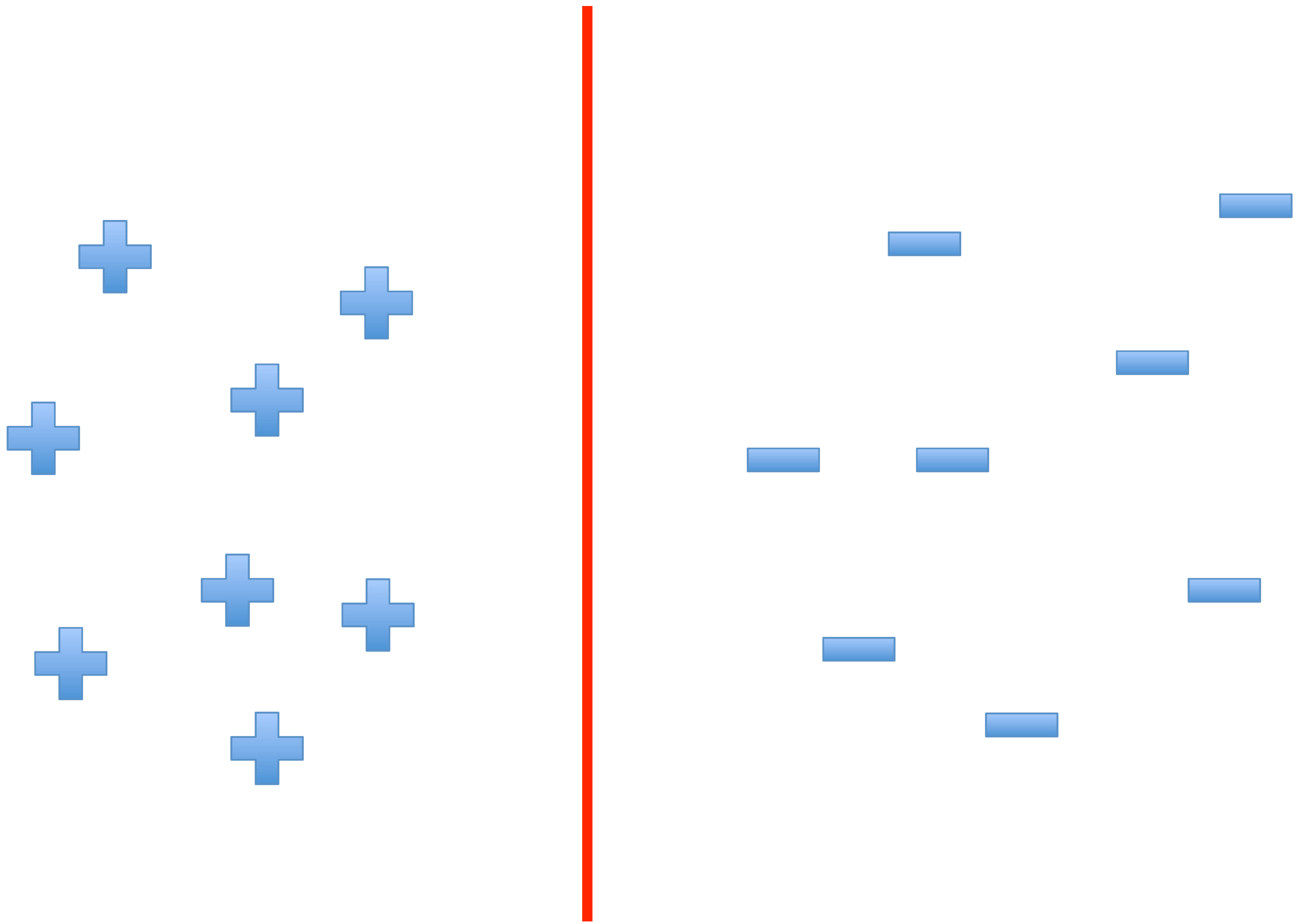
Intuitions



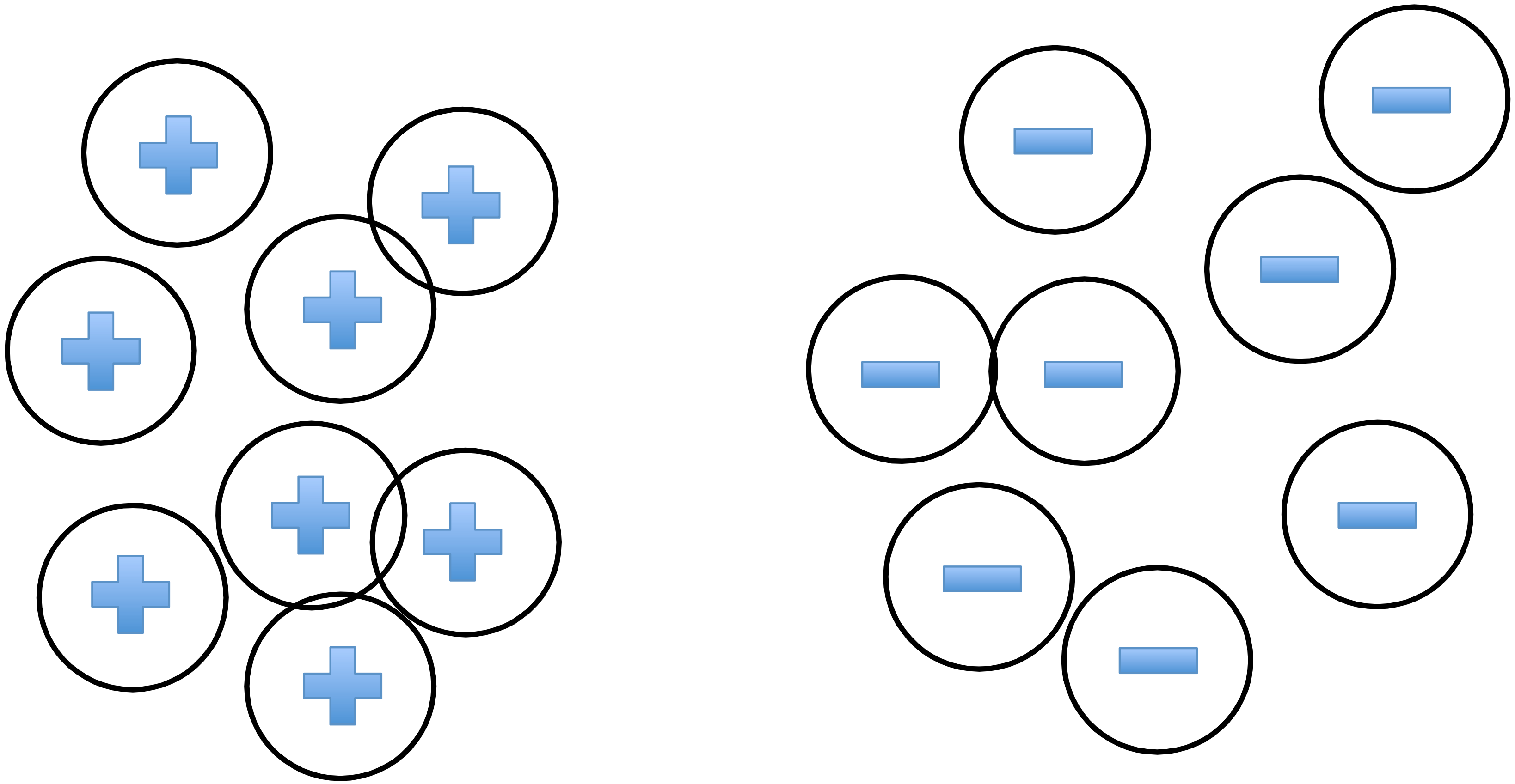
Intuitions



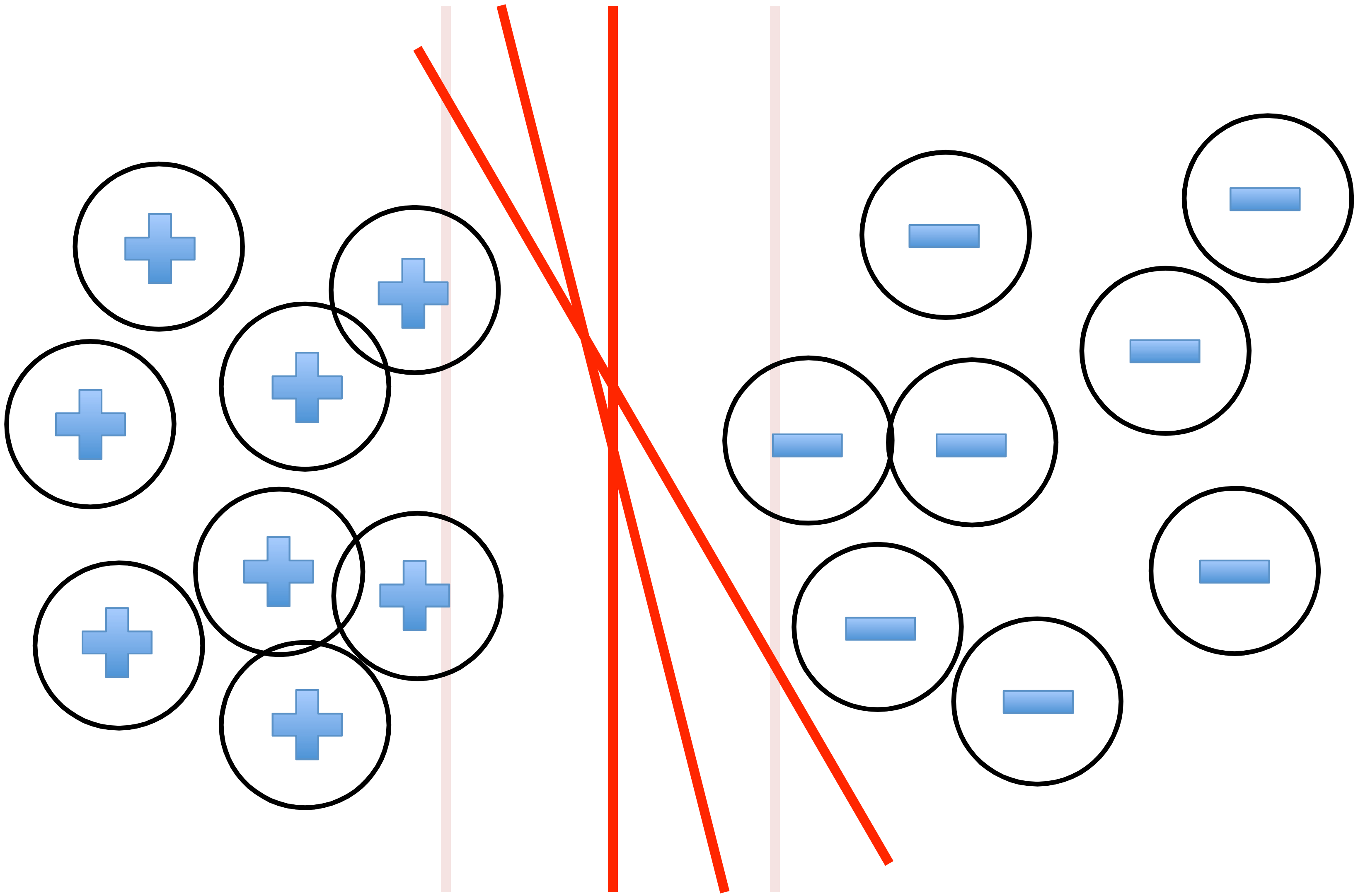
A “Good” Separator



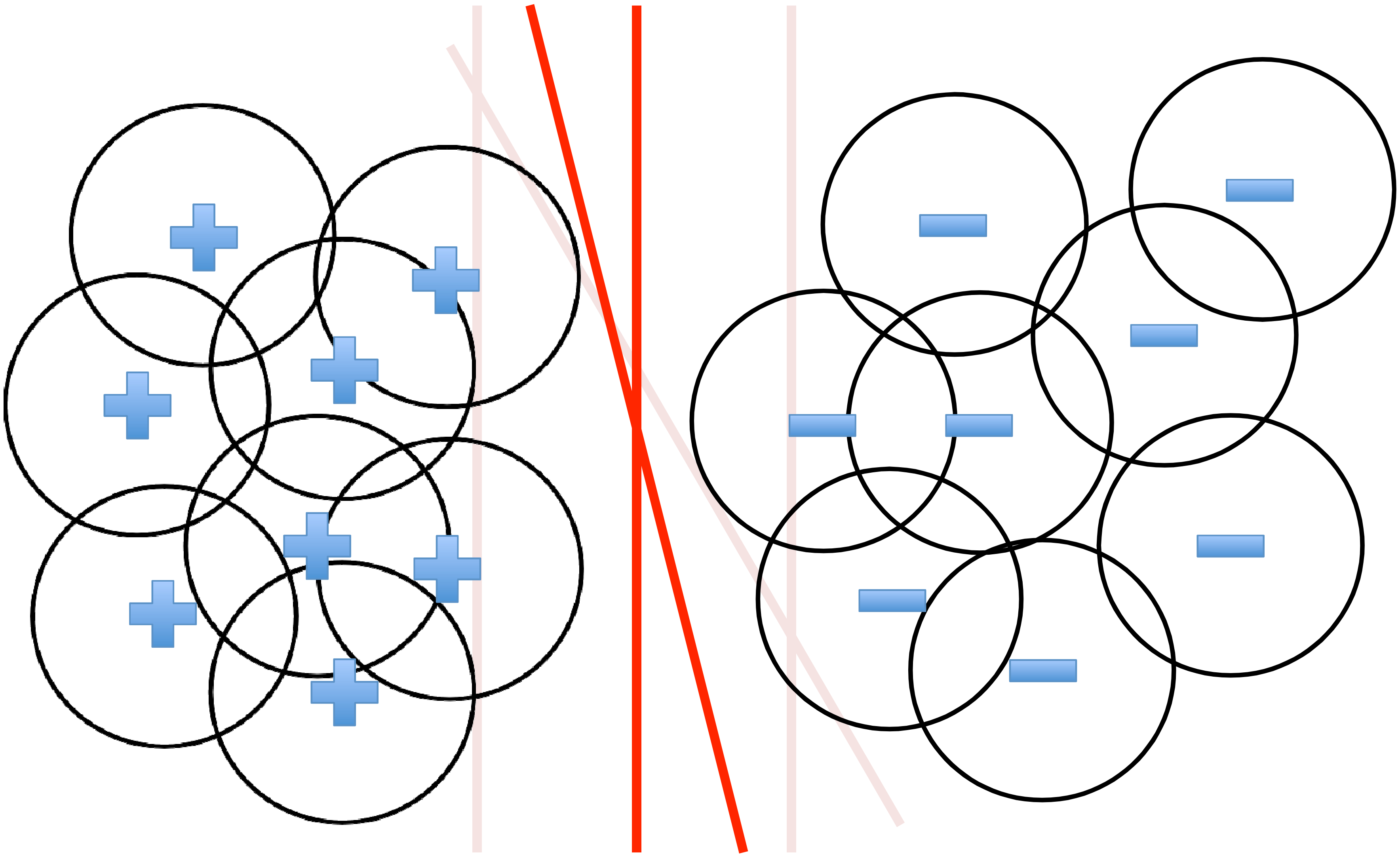
Noise in the Observations



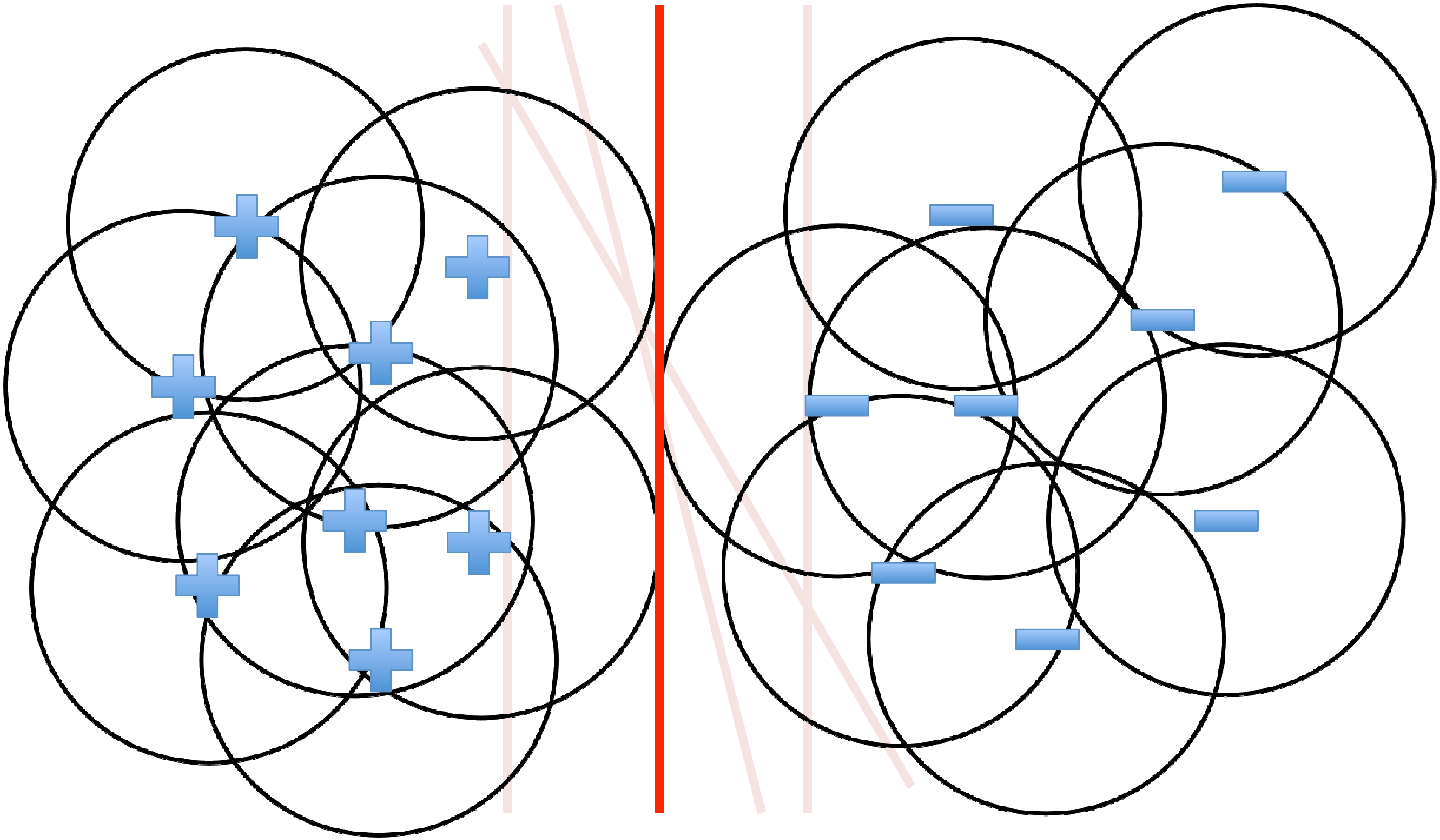
Ruling Out Some Separators



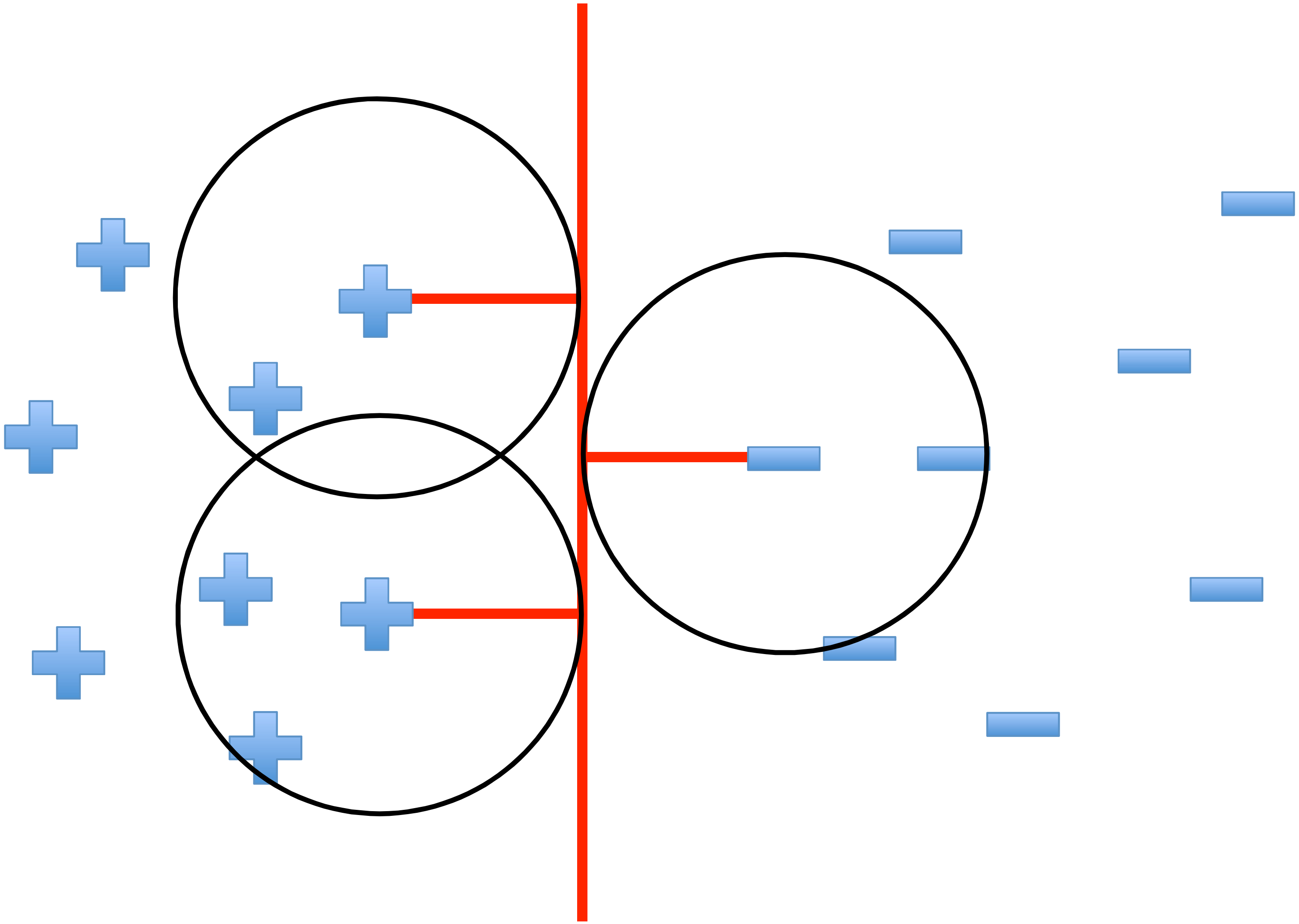
Lots of Noise



Only One Separator Remains



Maximizing the Margin



Why Maximize Margin

Increasing margin reduces *capacity*

- i.e., fewer possible models

Lesson from Learning Theory:

- If the following holds:
  - H is sufficiently constrained in size
  - and/or the size of the training data set n is large,

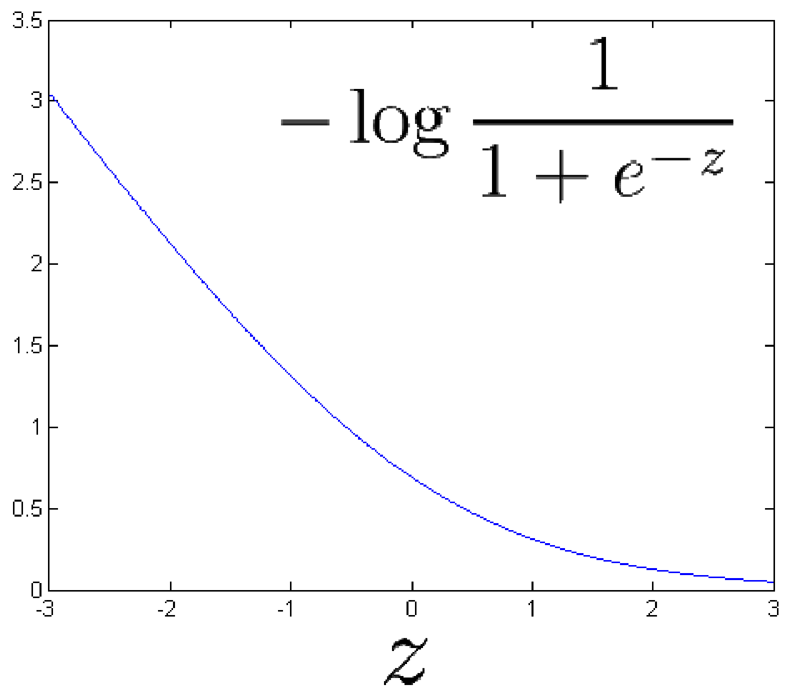
  then low training error is likely to be evidence of low generalization error

Alternate View of Logistic Regression

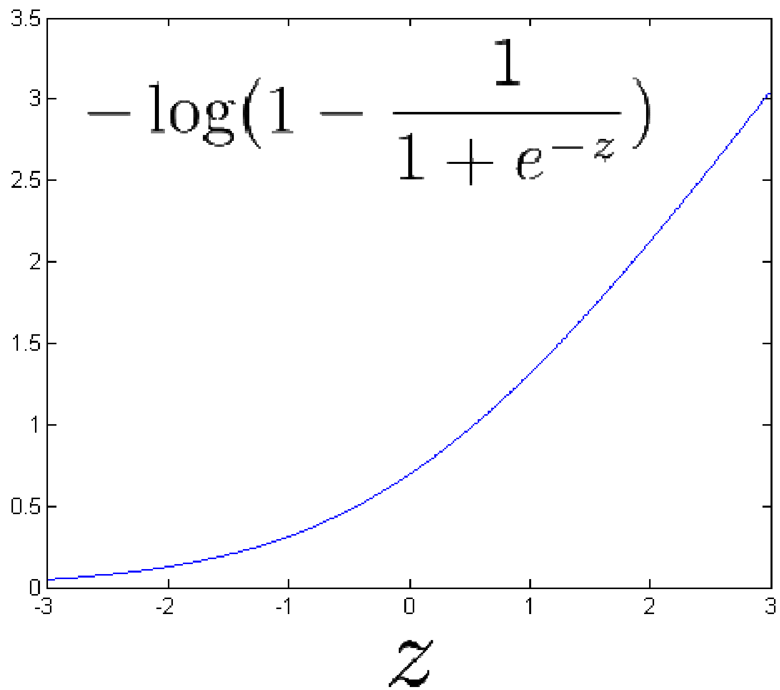
Cost of a sample: $y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)}))$

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}} \qquad z = \theta^T x$$

If y $= 1$ (we want $\theta^T x \gg 0$)



If y $= 0$ (we want $\theta^T x \ll 0$)



Based on slide by Andrew Ng

Logistic Regression to SVMs

Logistic Regression:

$$\min_{\theta} -\sum_{i=1}^{n}(y^{(i)} \log h_{\theta}(x^{(i)}) + (1-y^{(i)}) \log(1-h_{\theta}(x^{(i)}))) + \frac{\lambda}{2}\sum_{j=1}^{d}\theta_j^2$$

Support Vector Machines:

$$\min_{\theta} C\sum_{i=1}^{n}(y^{(i)} cost_1(\theta^T x^{(i)}) + (1-y^{(i)}) cost_0(\theta^T x^{(i)})) + \frac{1}{2}\sum_{j=1}^{d}\theta_j^2$$

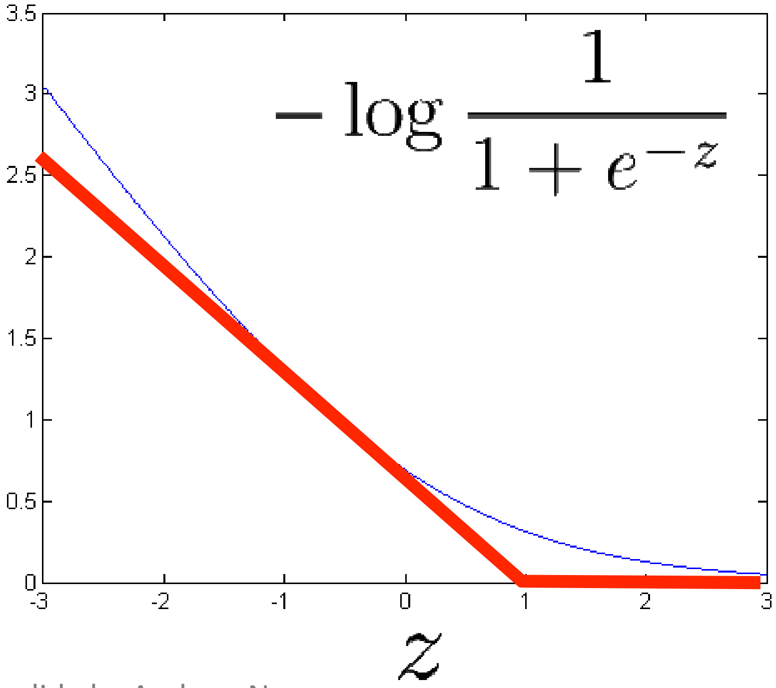
You can think of C as similar to $\frac{1}{\lambda}$

From Logistic Regression to SVM

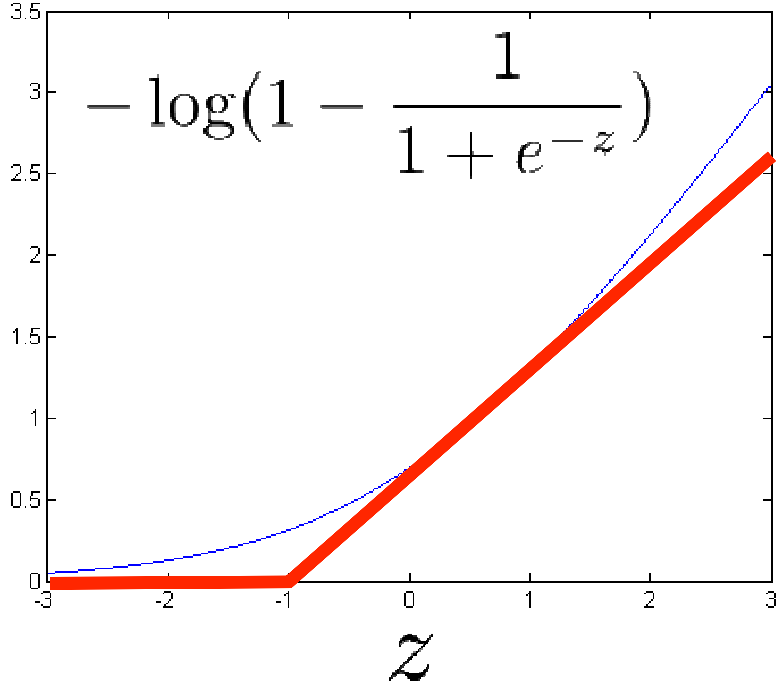
Support Vector Machines:

$$\min_{\theta} C \sum_{i=1}^{n} (y^{(i)} cost_1(\theta^T x^{(i)}) + (1 - y^{(i)}) cost_0(\theta^T x^{(i)})) + \frac{1}{2} \sum_{j=1}^{d} \theta_j^2$$

If y $= 1$ (we want $\theta^T x \gg 0$)



If y $= 0$ (we want $\theta^T x \ll 0$)

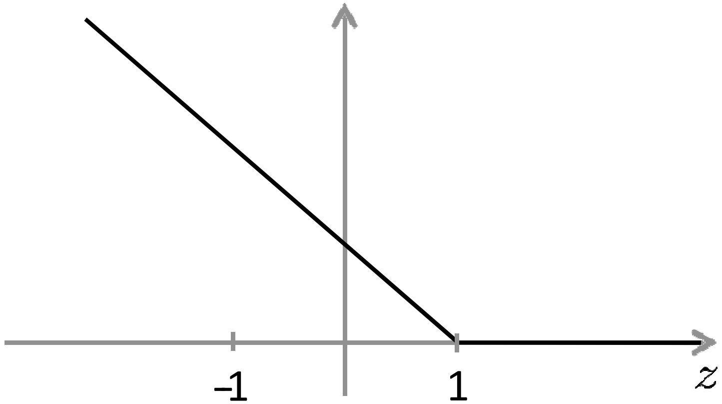


Based on slide by Andrew Ng

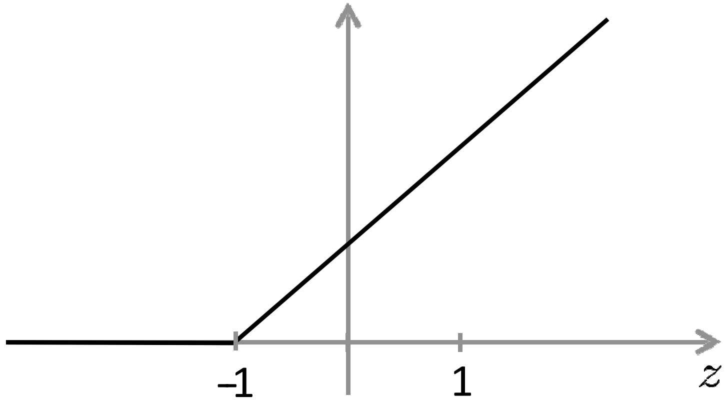
Support Vector Machine

$$\min_{\theta} C \sum_{i=1}^{n} (y^{(i)} cost_1(\theta^T x^{(i)}) + (1 - y^{(i)}) cost_0(\theta^T x^{(i)})) + \frac{1}{2} \sum_{j=1}^{d} \theta_j^2$$

If y $= 1$ (we want $\theta^T x \geq 1$)



If y $= 0$ (we want $\theta^T x \leq -1$)



Based on slide by Andrew Ng

Support Vector Machine

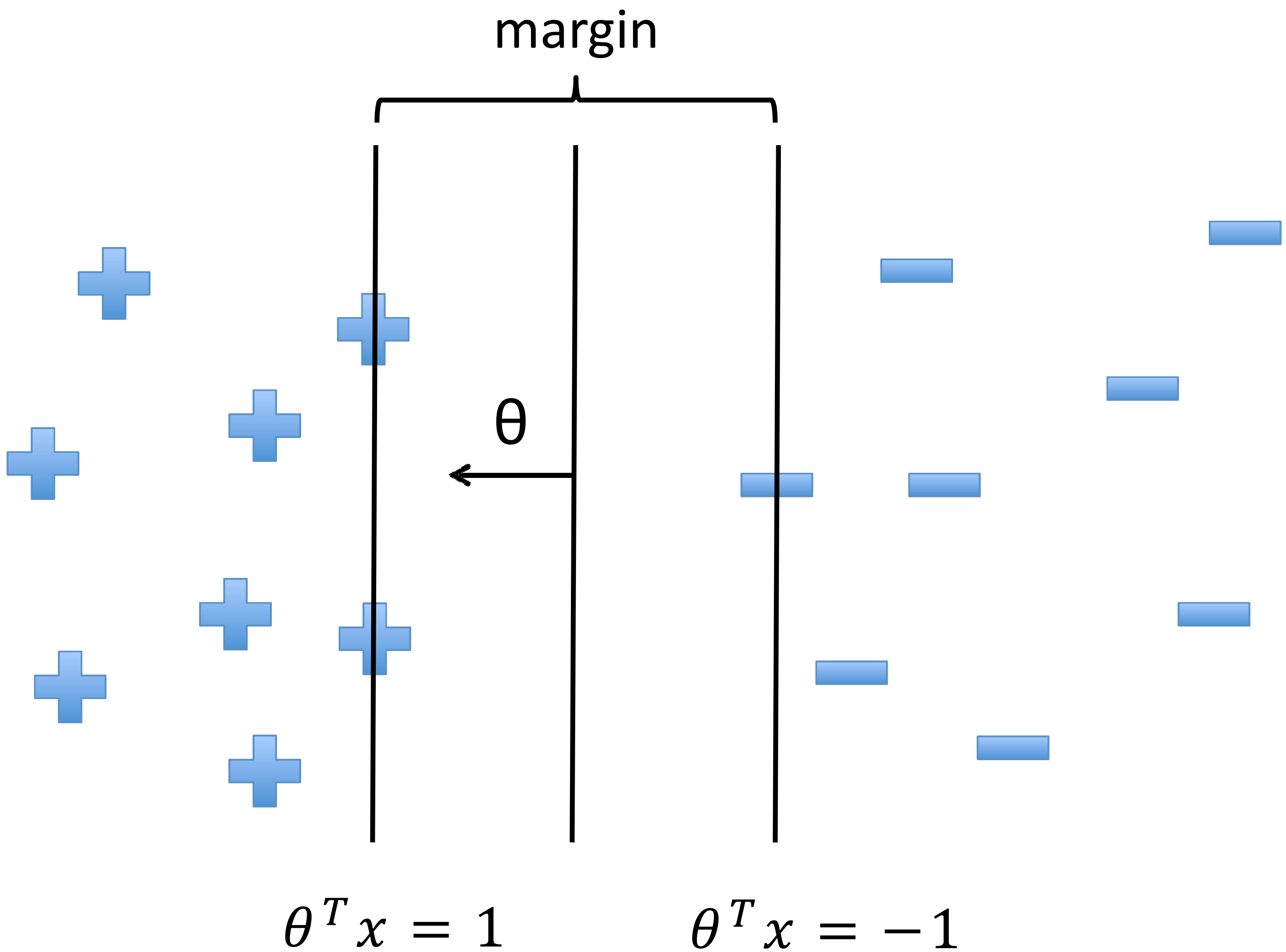
$$\min_{\theta} C \sum_{i=1}^{n} \underbrace{\left(y^{(i)} cost_1(\theta^T x^{(i)}) + (1 - y^{(i)}) cost_0(\theta^T x^{(i)})\right)}_{0} + \frac{1}{2} \sum_{j=1}^{d} \theta_j^2$$

$$\min_{\theta} \frac{1}{2} \sum_{j=1}^{d} \theta_j^2$$

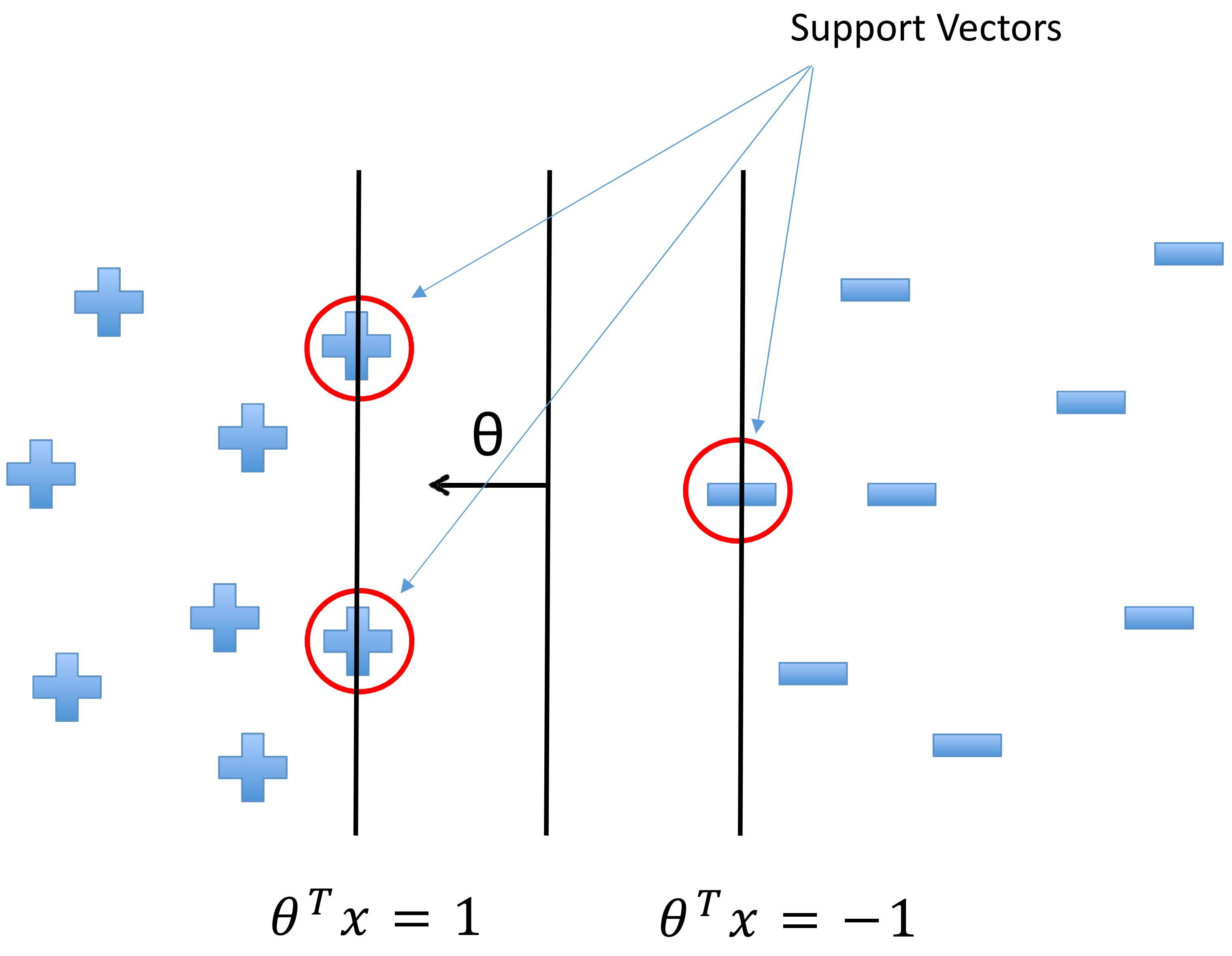
s.t. $\theta^T x^{(i)} \geq 1$ if $y^{(i)} = 1$

$\theta^T x^{(i)} \leq -1$ if $y^{(i)} = 0$

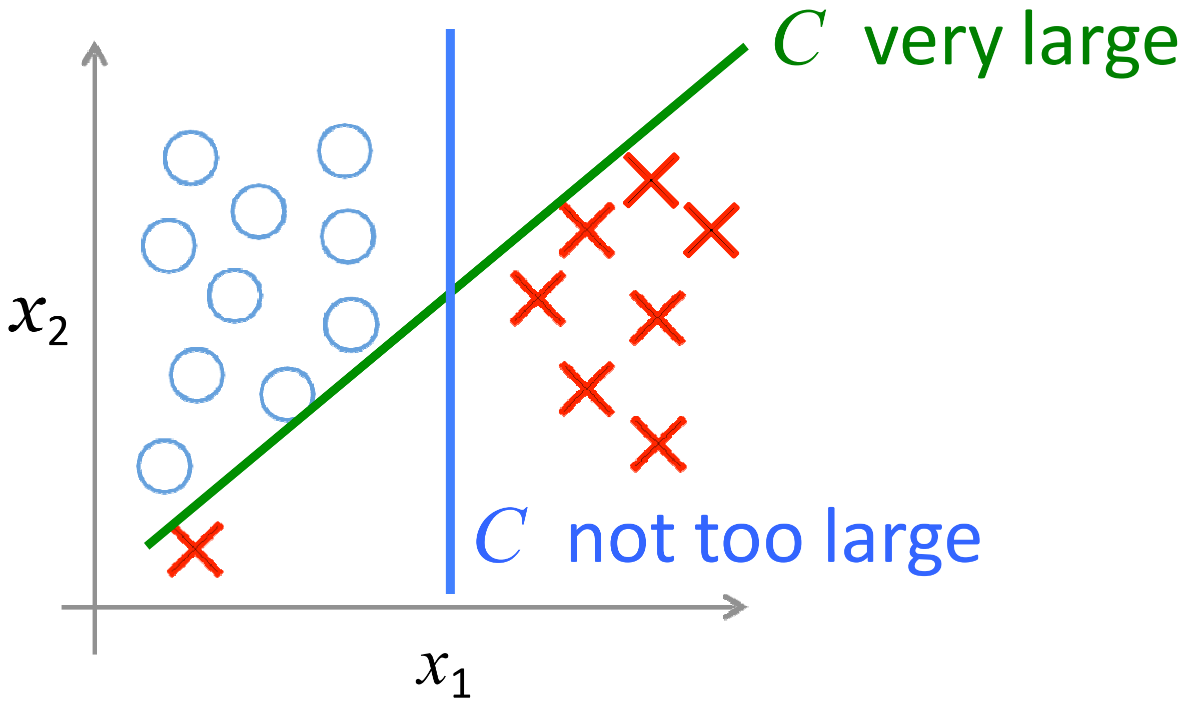
Maximum Margin Hyperplane



Support Vectors



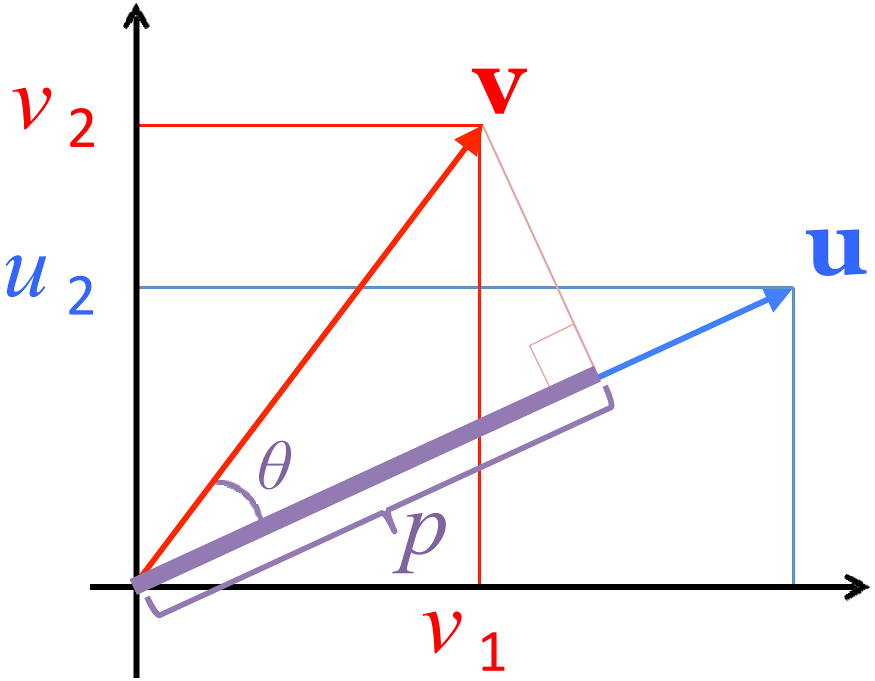
Large Margin Classifier in Presence of Outliers



$$\min_{\theta} C \sum_{i=1}^{n} (y^{(i)} cost_1(\theta^T x^{(i)}) + (1 - y^{(i)}) cost_0(\theta^T x^{(i)})) + \frac{1}{2} \sum_{j=1}^{d} \theta_j^2$$

Based on slide by Andrew Ng

Vector Inner Product



$$u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \qquad v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

$$\|u\|_2 = \sqrt{u_1^2 + u_2^2}$$

= length (u)

$$\begin{aligned} u^T v &= v^T u \\ &= u_1 v_1 + u_2 v_2 \\ &= \|u\|_2 \, \|v\|_2 \cos\theta \\ &= p\,\|u\|_2 \text{ where p } = \|v\|_2 \cos\theta \end{aligned}$$

Based on example by Andrew Ng

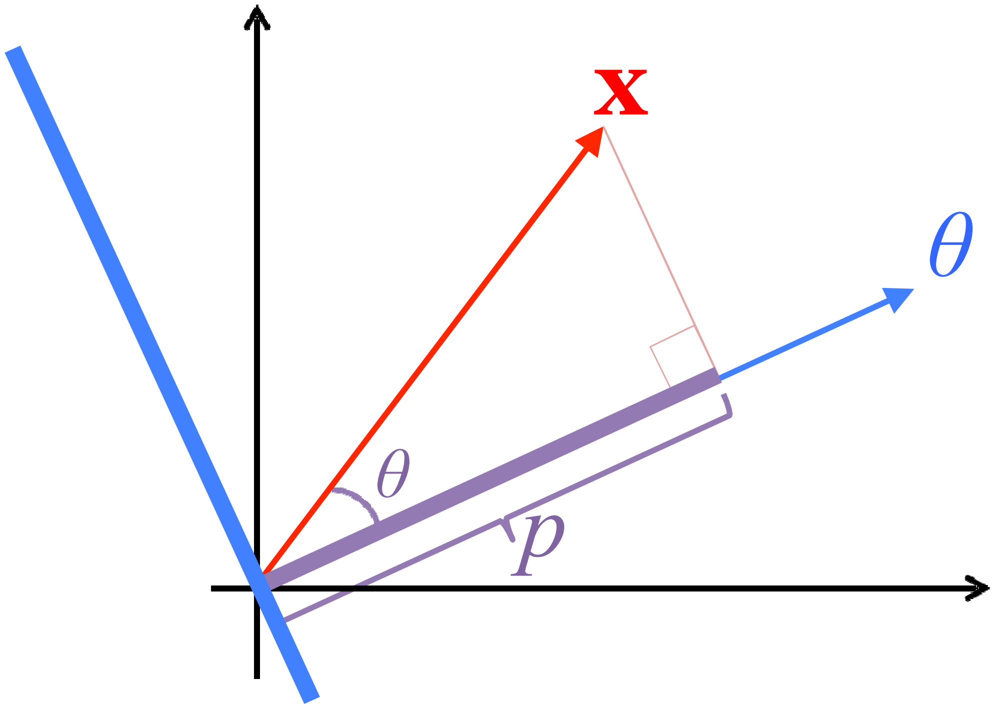
Understanding the Hyperplane

$$\min_{\theta} \frac{1}{2} \sum_{j=1}^{d} \theta_j^2$$

s.t. $\theta^T x^{(i)} \geq 1$ if $y^{(i)} = 1$

$\theta^T x^{(i)} \leq -1$ if $y^{(i)} = 0$

Assume $\theta_0 = 0$ so that the hyperplane is centered at the origin, and that $d = 2$



$$\theta^T x = \|\theta\|_2 \|x\|_2 \cos \theta$$

$$= p \|\theta\|_2$$

Based on example by Andrew Ng

Maximizing the Margin

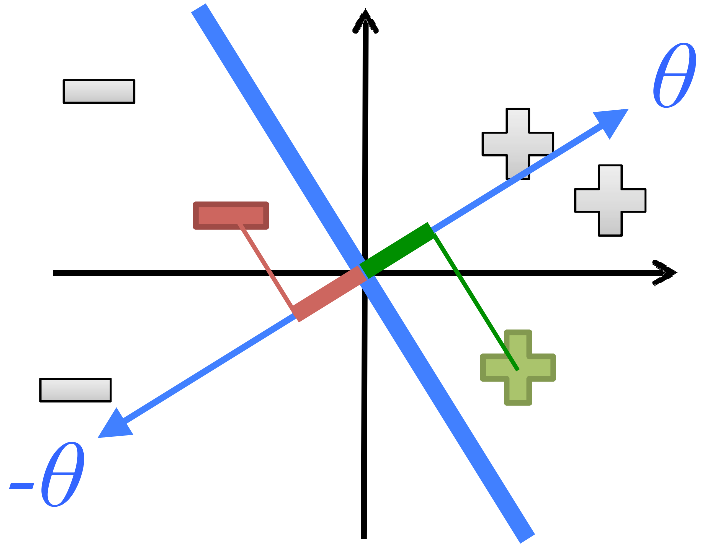
$$\min_{\theta} \frac{1}{2} \sum_{j=1}^{d} \theta_j^2$$

s.t. $\theta^T x^{(i)} \geq 1$ if $y^{(i)} = 1$

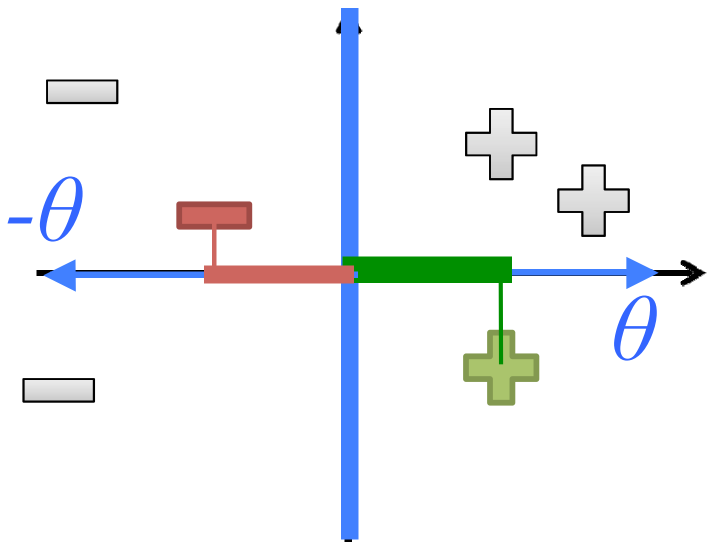
$\theta^T x^{(i)} \leq -1$ if $y^{(i)} = 0$

Assume $\theta_0 = 0$ so that the hyperplane is centered at the origin, and that $d = 2$

Let p_i be the projection of $\mathbf{x}_i$ onto the vector $\boldsymbol{\theta}$



Since p is small, therefore $\|\theta\|_2$ must be large to have $p \ \|\theta\|_2 \geq 1$ (or ≤ -1)

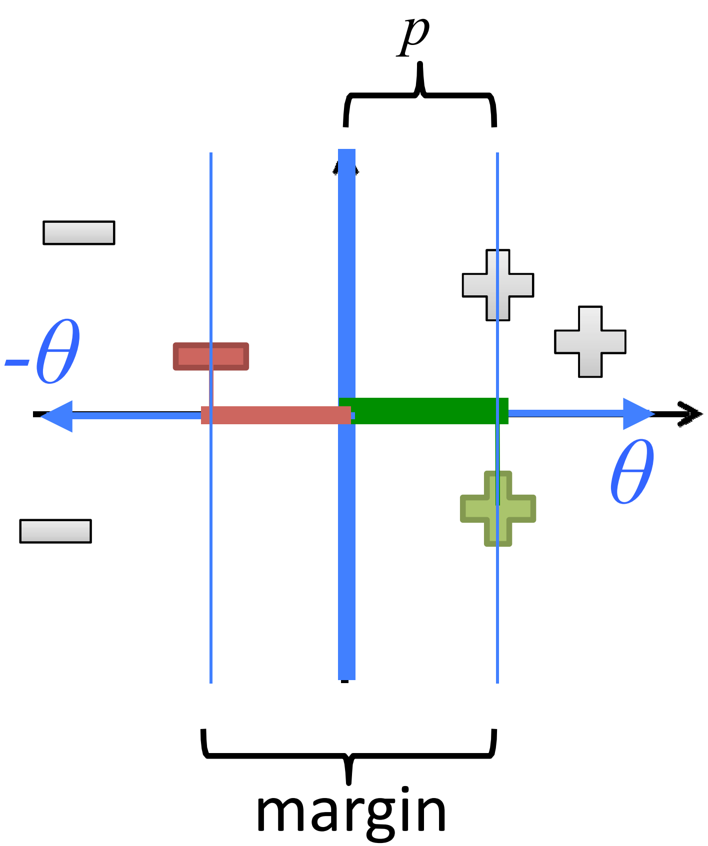


Since p is large, $\|\theta\|_2$ can be smaller to have $p \ \|\theta\|_2 \geq 1$ (or ≤ -1)

Size of the Margin

For the support vectors, we have $p \; \|\theta\|_2 \; = \; \pm 1$

- p is the length of the projection of the SVs onto $\boldsymbol{\theta}$

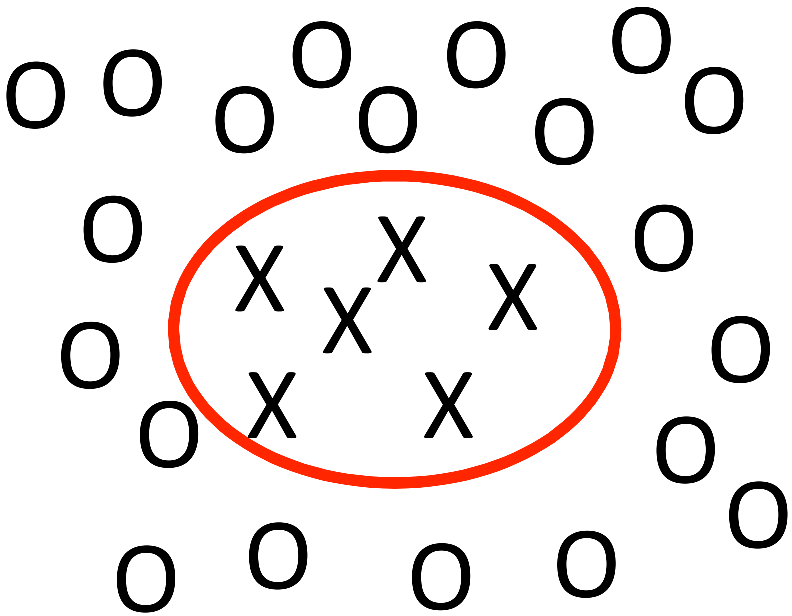


Therefore,

$$p = \frac{1}{\|\theta\|_2}$$

$$margin = 2p = \frac{2}{\|\theta\|_2}$$

What if Surface is Non--Linear?



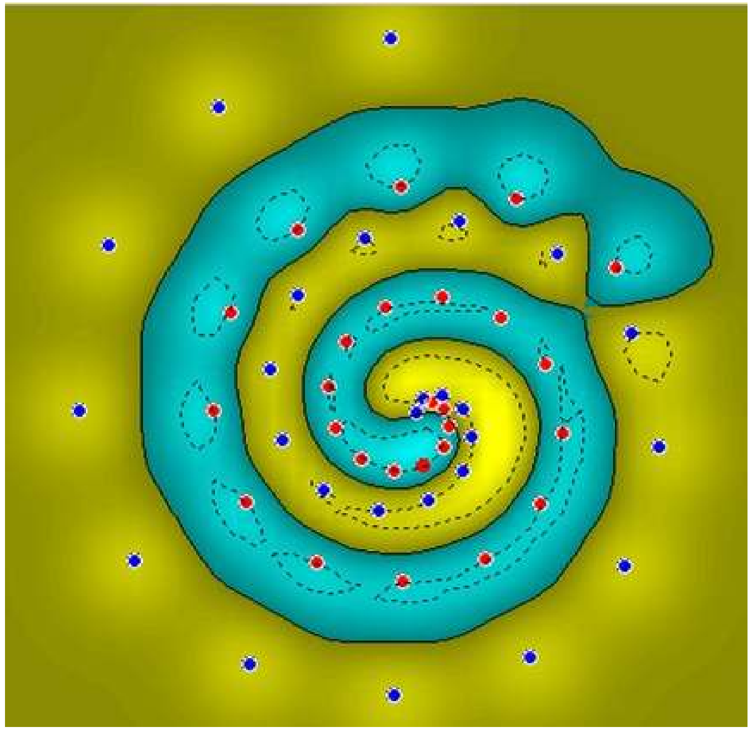
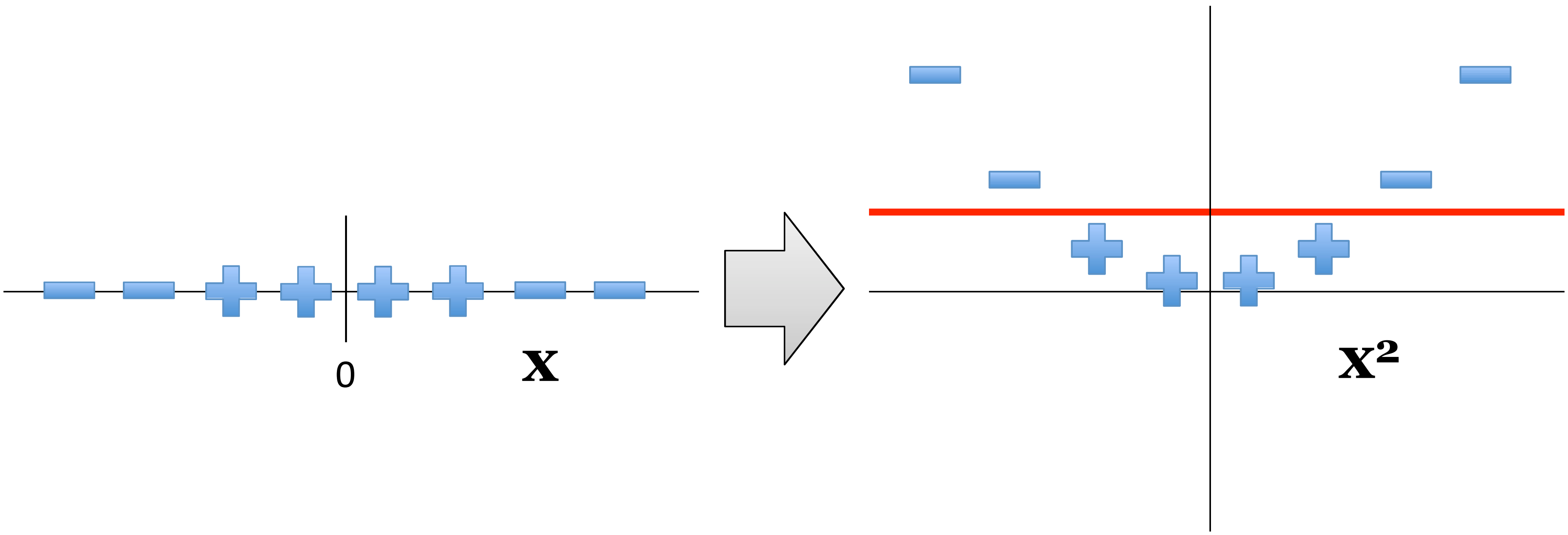


Image from http://www.atrandomresearch.com/iclass/

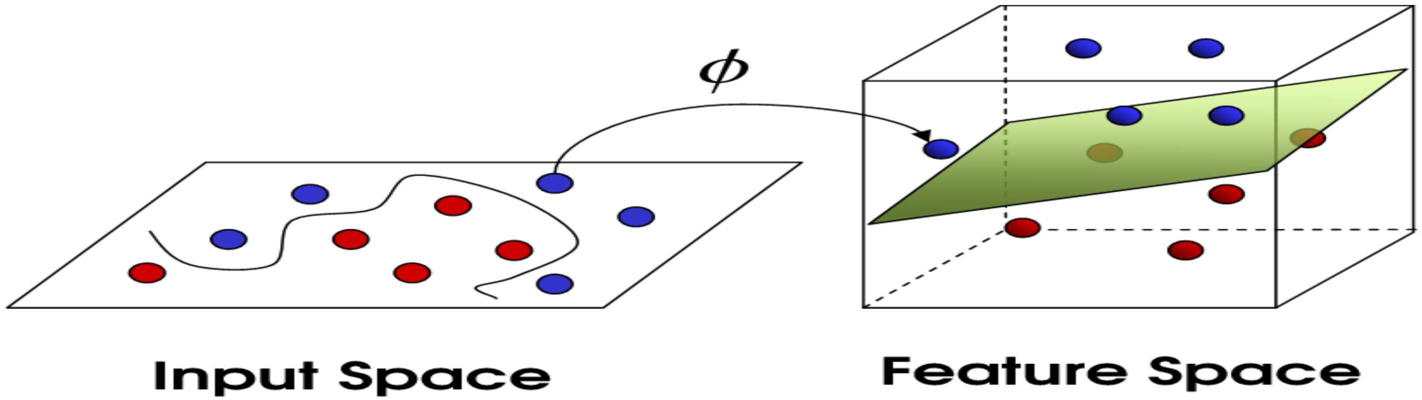
Kernel Methods

Making the Non-Linear Linear

When Linear Separators Fail



Mapping into a New Feature Space



$$\varphi: X \rightarrow \hat{X} = \varphi(x)$$

- For example, with $x^{(i)} \in R^2$

$$\varphi\left(\left[x_1^{(i)}, x_2^{(i)}\right]\right) = [x_1^{(i)}, x_2^{(i)}, x_1^{(i)}.x_2^{(i)}, (x_1^{(i)})^2, (x_2^{(i)})^2]$$

- Rather than run SVM on $\mathbf{x}_i$, run it on $\varphi(x^{(i)})$
  - Find non--linear separator in input space

- What if $\varphi(x^{(i)})$ is really big?
- Use kernels to compute it implicitly!

Image from http://web.engr.oregonstate.edu/ ~afern/classes/cs534/

Kernels

- Find kernel K such that
  $$\mathrm{K}\left(x^{(i)}, x^{(j)}\right) =< \varphi\left(x^{(i)}\right), \varphi\left(x^{(j)}\right) >$$
- Computing $\mathrm{K}\left(x^{(i)}, x^{(j)}\right)$ should be efficient, much more so than computing $\varphi\left(x^{(i)}\right)$ and $\varphi\left(x^{(j)}\right)$
- Use $\mathrm{K}\left(x^{(i)}, x^{(j)}\right)$ in SVM algorithm rather than $x^{(i)}, x^{(j)}$
- Remarkably, this is possible!

The Kernel Trick

"Given an algorithm which is formulated in terms of a positive definite kernel K_1, one can construct an alternative algorithm by replacing K_1 with another positive definite kernel K_2"

- SVMs can use the kernel trick

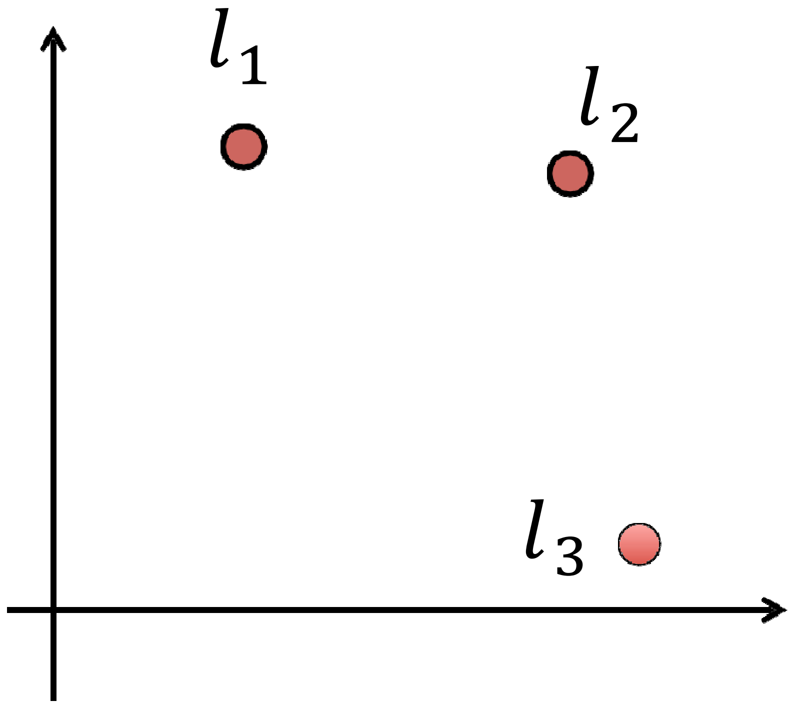
The Gaussian Kernel

- Also called Radial Basis Function (RBF) kernel

$$\mathrm{K}\left(x^{(i)}, x^{(j)}\right) = \exp\left(-\frac{\left\|x^{(i)} - x^{(j)}\right\|_2^2}{2\sigma^2}\right)$$

  - Has value 1 when $\mathbf{x}_i = \mathbf{x}_j$
  - Value falls off to 0 with increasing distance
  - Note: Need to do feature scaling <u>before</u> using Gaussian Kernel

Gaussian Kernel Example



$$K\left(x^{(i)}, x^{(j)}\right) = \exp\left(-\frac{\left\|x^{(i)} - x^{(j)}\right\|_2^2}{2\sigma^2}\right)$$

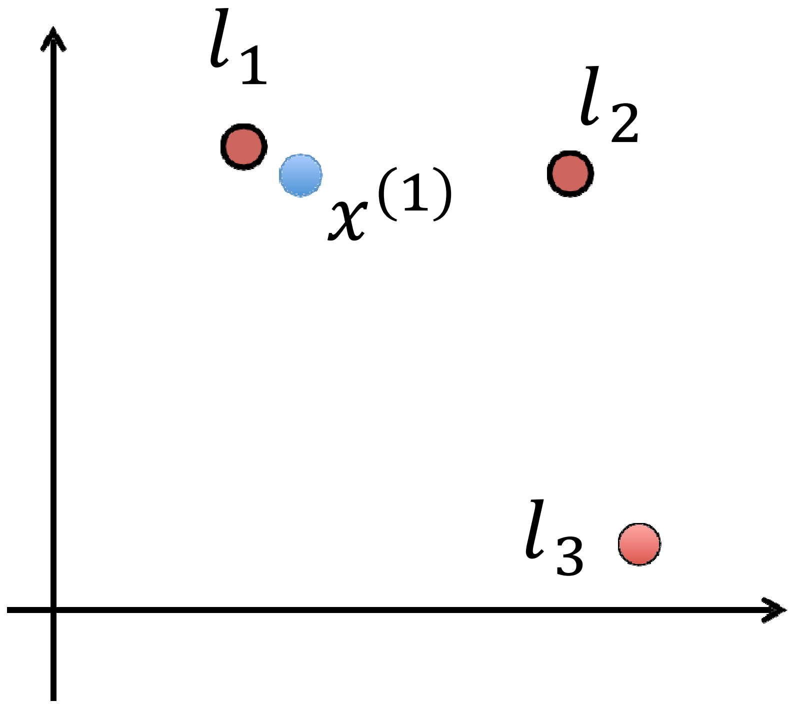
Imagine we've learned that:

$$\theta = [-0.5,1,1,0]$$

Predict +1 if $\theta_0 + \theta_1 K(x, l_1) + \theta_2 K(x, l_2) + \theta_3 K(x, l_3) \geq 0$

Based on example by Andrew Ng

Gaussian Kernel Example



$$K\left(x^{(i)}, x^{(j)}\right) = \exp\left(-\frac{\left\|x^{(i)} - x^{(j)}\right\|_2^2}{2\sigma^2}\right)$$

Imagine we've learned that:

$$\theta = [-0.5, 1, 1, 0]$$

Predict +1 if $\theta_0 + \theta_1 K(x, l_1) + \theta_2 K(x, l_2) + \theta_3 K(x, l_3) \geq 0$

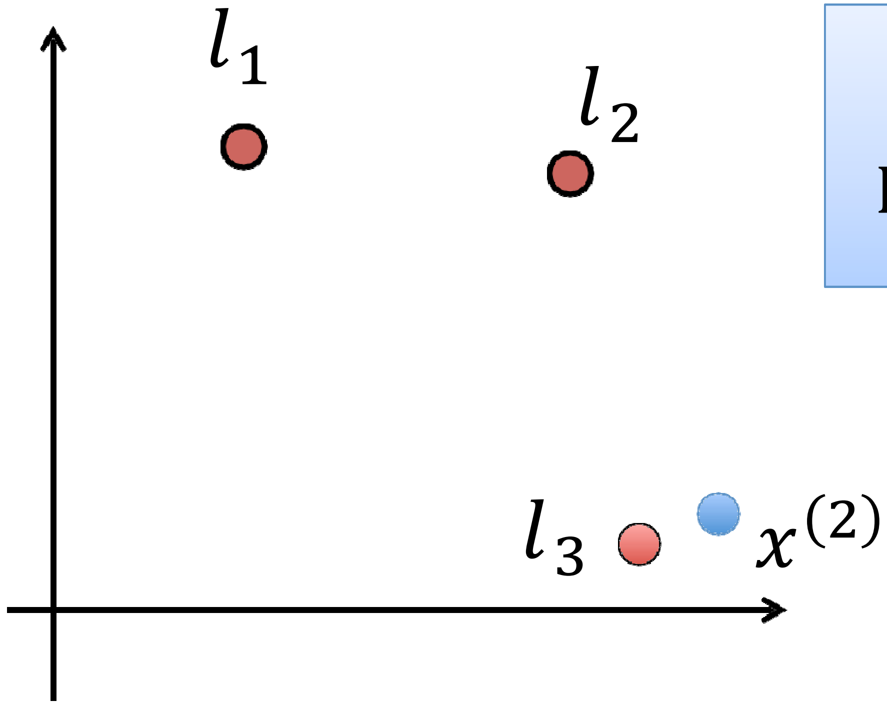
- For $x^{(1)}$, we have $K(x^{(1)}, l_1) \approx 1$, other similarities ≈ 0

$$\theta_0 + \theta_1 . 1 + \theta_2 . 0 + \theta_2 . 0 = -0.5 + 1.1 + 1.0 + 0.1$$

$= 0.5 \;\geq 0$, so predict +1

Based on example by Andrew Ng

Gaussian Kernel Example



$$K\left(x^{(i)}, x^{(j)}\right) = \exp\left(-\frac{\left\|x^{(i)} - x^{(j)}\right\|_2^2}{2\sigma^2}\right)$$

Imagine we've learned that:

$$\theta = [-0.5, 1, 1, 0]$$

Predict +1 if $\theta_0 + \theta_1 K(x, l_1) + \theta_2 K(x, l_2) + \theta_3 K(x, l_3) \geq 0$

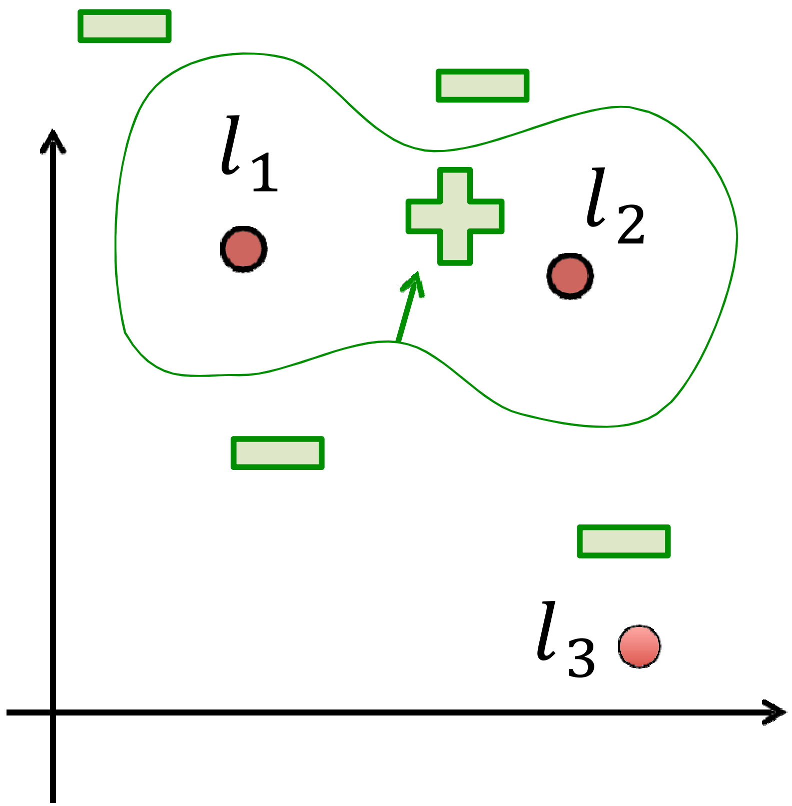
- For $x^{(2)}$, we have $K(x^{(2)}, l_3) \approx 1$, other similarities ≈ 0

$$\theta_0 + \theta_1 . 0 + \theta_2 . 0 + \theta_2 . 1 = -0.5 + 1.0 \ + 1.0 + 0.1$$

$$= -0.5 \ \leq 0$$, so predict −1

Based on example by Andrew Ng

Gaussian Kernel Example



$$K\left(x^{(i)}, x^{(j)}\right) = \exp\left(-\frac{\left\|x^{(i)} - x^{(j)}\right\|_2^2}{2\sigma^2}\right)$$

Imagine we've learned that:

$$\theta = [-0.5, 1, 1, 0]$$

Predict +1 if $\theta_0 + \theta_1 K(x, l_1) + \theta_2 K(x, l_2) + \theta_3 K(x, l_3) \geq 0$

Rough sketch of decision surface

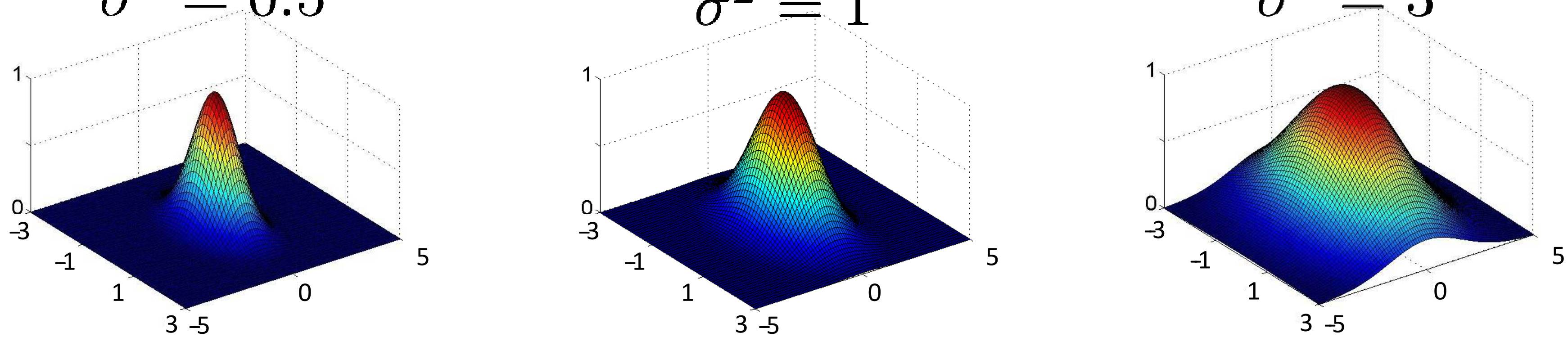
Based on example by Andrew Ng

The Gaussian Kernel

- Also called Radial Basis Function (RBF) kernel

$$\mathrm{K}\left(x^{(i)}, x^{(j)}\right) = \exp\left(-\frac{\left\|x^{(i)} - x^{(j)}\right\|_2^2}{2\sigma^2}\right)$$

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



lower bias,
higher variance

higher bias,
lower variance

Other Kernels

- Sigmoid Kernel

$$\mathrm{K}\left(x^{(i)}, x^{(j)}\right) = \tanh(\propto (x^{(i)})^T x^{(j)} + c)$$

  - Neural networks use sigmoid as activation function
  - SVM with a sigmoid kernel is equivalent to 2–layer perceptron

- Cosine Similarity Kernel

$$\mathrm{K}\left(x^{(i)}, x^{(j)}\right) = \frac{(x^{(i)})^T x^{(j)}}{\left\|x^{(i)}\right\| . \left\|x^{(j)}\right\|}$$

  - Popular choice for measuring similarity of text documents
  - L_2 norm projects vectors onto the unit sphere; their dot product is the cosine of the angle between the vectors

Other Kernels

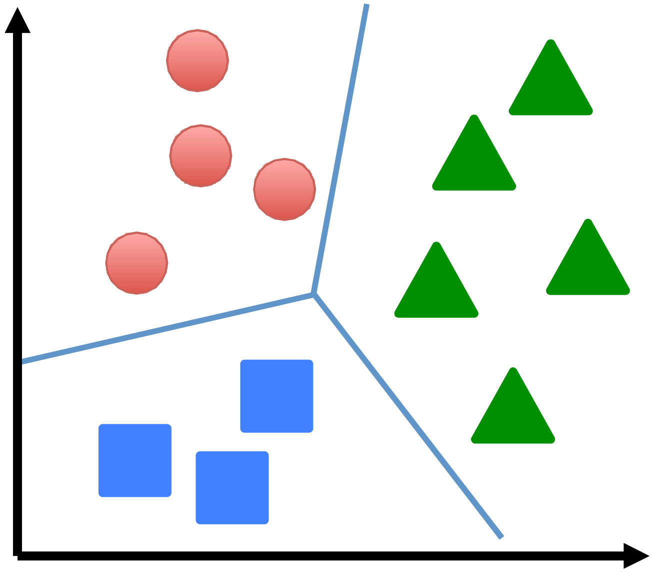
- Chi-squared Kernel

$$\mathrm{K}\left(x^{(i)}, x^{(j)}\right) = \exp\left(-\gamma \sum_k \frac{\left(x_k^{(i)} - x_k^{(j)}\right)^2}{x_k^{(i)} + x_k^{(j)}}\right)$$

  - Widely used in computer vision applications
  - Chi--squared measures distance between probability distributions
  - Data is assumed to be non--negative, often with L_1 norm of 1

- String kernels
- Tree kernels
- Graph kernels

Multi-Class Classification with SVMs



$y \in \{1, \dots, K\}$

- Many SVM packages already have multi-class classification built in
- Otherwise, use one-vs-rest
  - Train K SVMs, each picks out one class from rest, yielding $\theta^{(i)}, \dots, \theta^{(K)}$
  - Predict class i with largest $(\theta^{(i)})^T x$

Based on slide by Andrew Ng

Practical Advice for Applying SVMs

- Use SVM software package to solve for parameters
  - e.g., SVMlight, libsvm, cvx (fast!), etc.

- Need to specify:
  - Choice of parameter C
  - Choice of kernel function
    - Associated kernel parameters

e.g.,

$$\mathrm{K}\left(x^{(i)}, x^{(j)}\right) = \exp\left(-\frac{\left\| x^{(i)} - x^{(j)} \right\|_2^2}{2\sigma^2}\right)$$

SVMs vs Logistic Regression
(Advice from Andrew Ng)

n = # training examples d = # features

If d is large (relative to n) (e.g., $d > n$ with d = 10,000, n = 10--1,000)

- Use logistic regression or SVM with a linear kernel

If d is small (up to 1,000), n is intermediate (up to 10,000)

- Use SVM with Gaussian kernel

If d is small (up to 1,000), n is large (50,000+)

- Create/add more features, then use logistic regression or SVM without a kernel

Neural networks likely to work well for most of these settings, but may be slower to train

Based on slide by Andrew Ng

Other SVM Variations

- nu SVM
  - nu parameter controls:
    - Fraction of support vectors (lower bound) and misclassification rate (upper bound)
    - E.g., $v = 0.05$ guarantees that ≥ 5% of training points are SVs and training error rate is ≤ 5%
  - Harder to optimize than C-SVM and not as scalable
- SVMs for regression
- One-class SVMs
- SVMs for clustering

...

Conclusion

- SVMs find optimal linear separator
- The kernel trick makes SVMs learn non-‐linear decision surfaces

- Strength of SVMs:
  - Good theoretical and empirical performance
  - Supports many types of kernels

- Disadvantages of SVMs:
  - "Slow" to train/predict for huge data sets (but relatively fast!)
  - Need to choose the kernel (and tune its parameters)