Support Vector Machines & Kernels

Doing *really* well with linear decision surfaces

These slides were assembled by Eric Eaton, with grateful acknowledgement of the many others who made their course materials freely available online. Feel free to reuse or adapt these slides for your own academic purposes, provided that you include proper attribution. Please send comments and corrections to Eric.

Adapted from slides by Tim Oates

Outline

- Prediction
  - Why might predictions be wrong?
- Support vector machines
  - Doing really well with linear models
- Kernels
  - Making the non-linear linear

Why Might Predictions be Wrong?

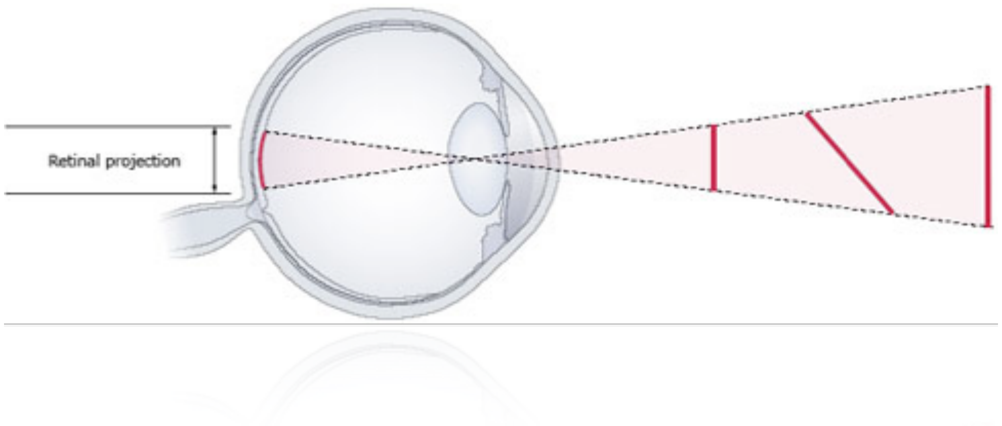
- True non-determinism
  - Flip a biased coin
  - p(heads) = θ
  - Estimate θ
  - If $\theta > 0.5$ predict ‘heads’, else ‘tails’

Lots of ML research on problems like this:

- Learn a model
- Do the best you can in expectation

Why Might Predictions be Wrong?

- Partial observability
  - Something needed to predict y is missing from observation x



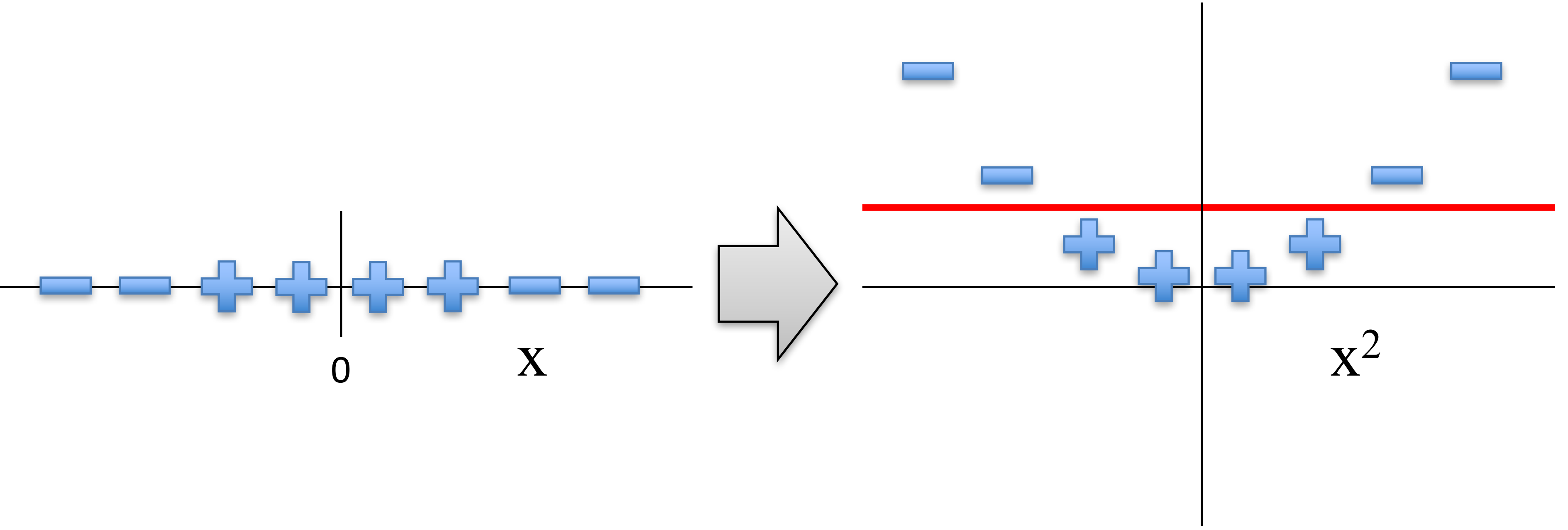
- Noise in the observation x
  - Measurement error
  - Instrument limitations

Why Might Predictions be Wrong?

- True non-determinism
- Partial observability
  - hard, soft
- Representational bias
- Algorithmic bias
- Bounded resources

Representational Bias

- Having the right features (x) is crucial



Support Vector Machines

Doing ***Really*** Well with Linear Decision Surfaces

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

Minor Notation Change

To better match notation used in SVMs

...and to make matrix formulas simpler

We will drop using superscripts for the i th instance

i th instance	$\boldsymbol{x}^{(i)}$	$\rightarrow$	$\mathbf{x}_i$
i th instance label	$y^{(i)}$	$\rightarrow$	y_i
j th feature of i th instance	$x_j^{(i)}$	$\rightarrow$	x_{ij}

Bold denotes vector

Non-bold denotes scalar

Linear Separators

- Training instances

$$\mathbf{x} \in \mathbb{R}^{d+1}, x_0 = 1$$

$$y \in \{-1, 1\}$$

- Model parameters

$$\boldsymbol{\theta} \in \mathbb{R}^{d+1}$$

- Hyperplane

$$\boldsymbol{\theta}^\intercal \mathbf{x} = \langle \boldsymbol{\theta}, \mathbf{x} \rangle = 0$$

- Decision function

$$h(\mathbf{x}) = \text{sign}(\boldsymbol{\theta}^\intercal \mathbf{x}) = \text{sign}(\langle \boldsymbol{\theta}, \mathbf{x} \rangle)$$

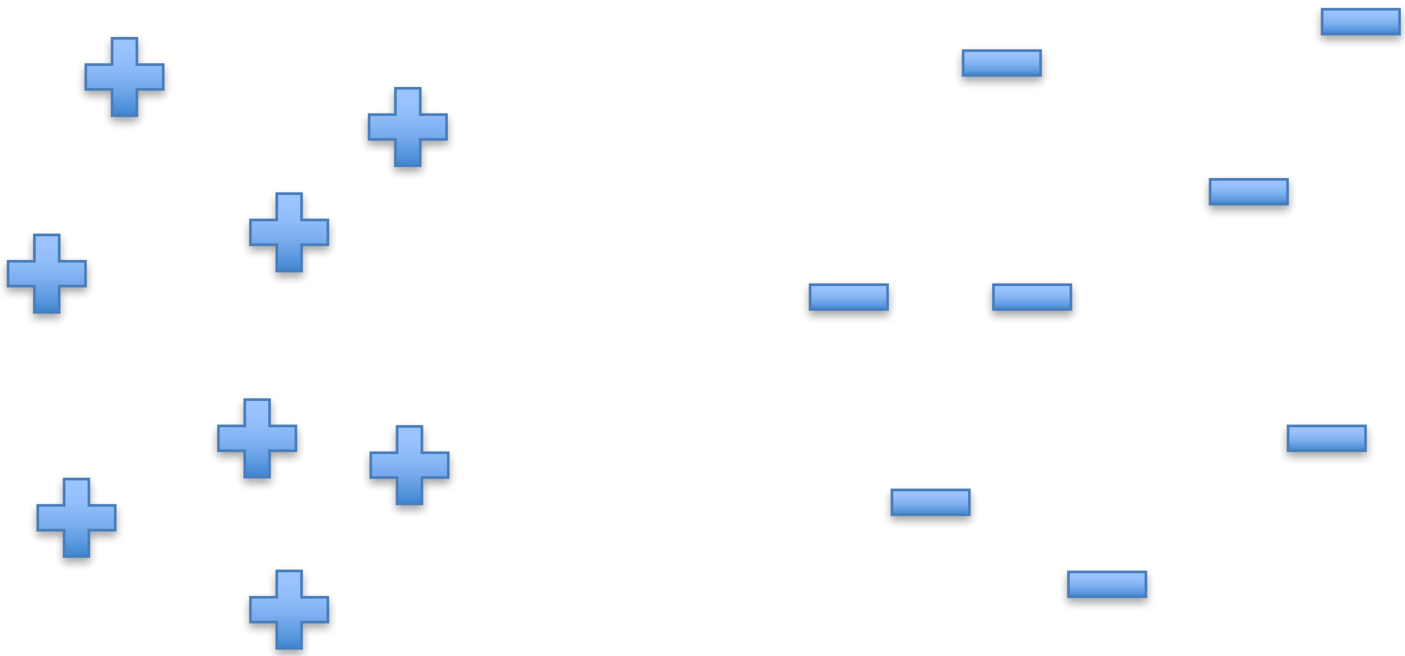
<u>Recall:</u>

Inner (dot) product:

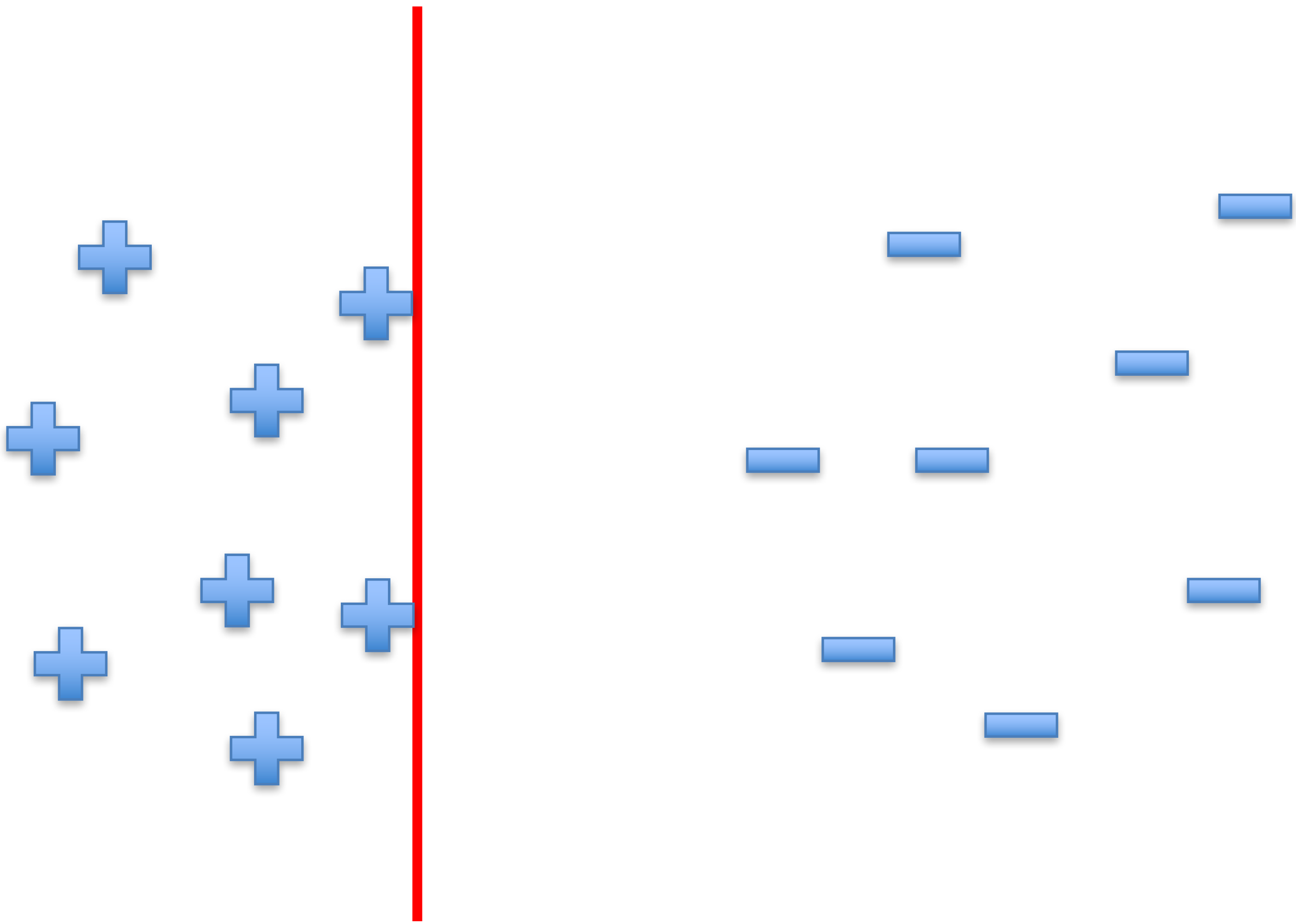
$$\langle \mathbf{u}, \mathbf{v} \rangle = \mathbf{u} \cdot \mathbf{v} = \mathbf{u}^\intercal \mathbf{v}$$

$$= \sum_i u_i v_i$$

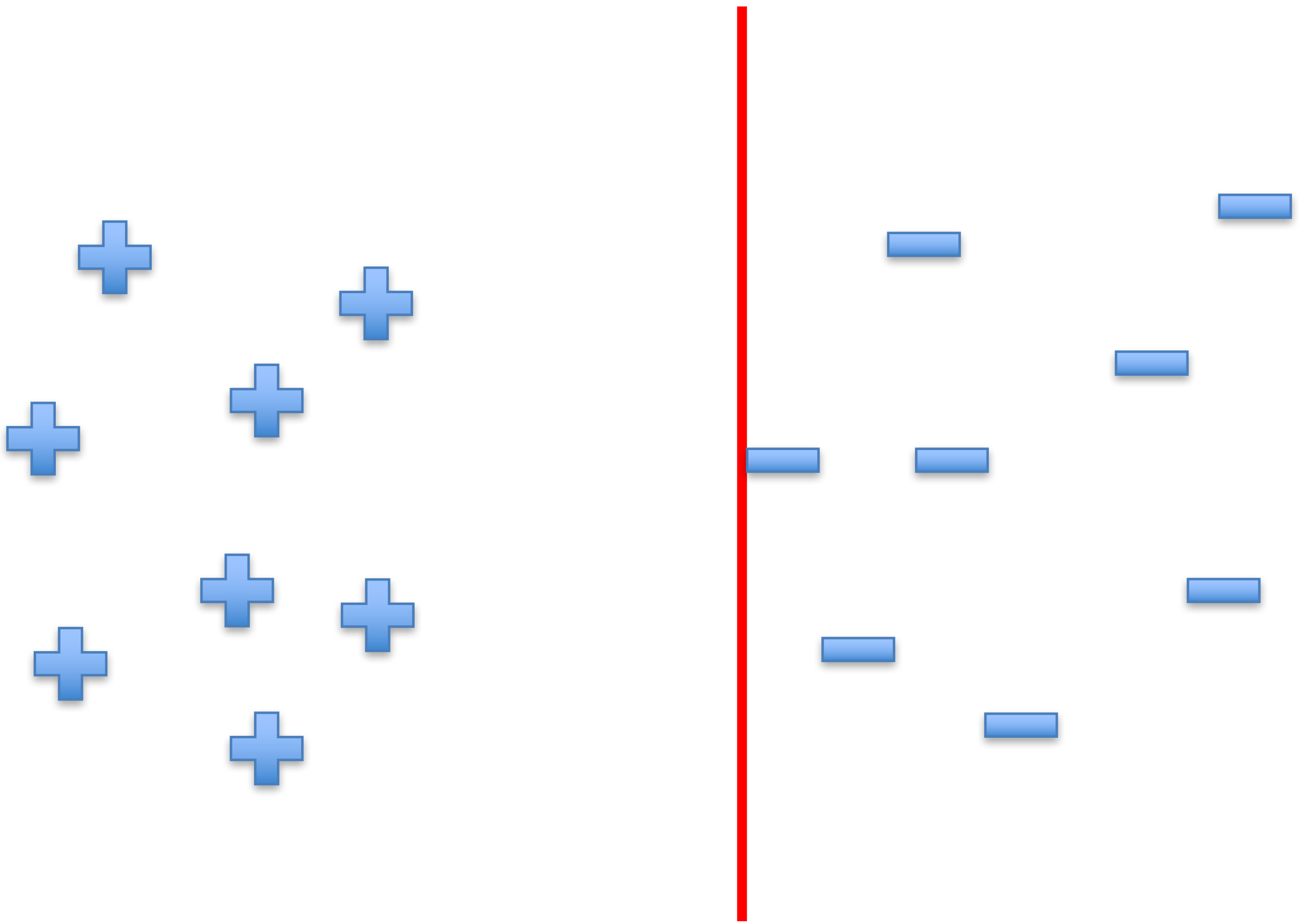
Intuitions



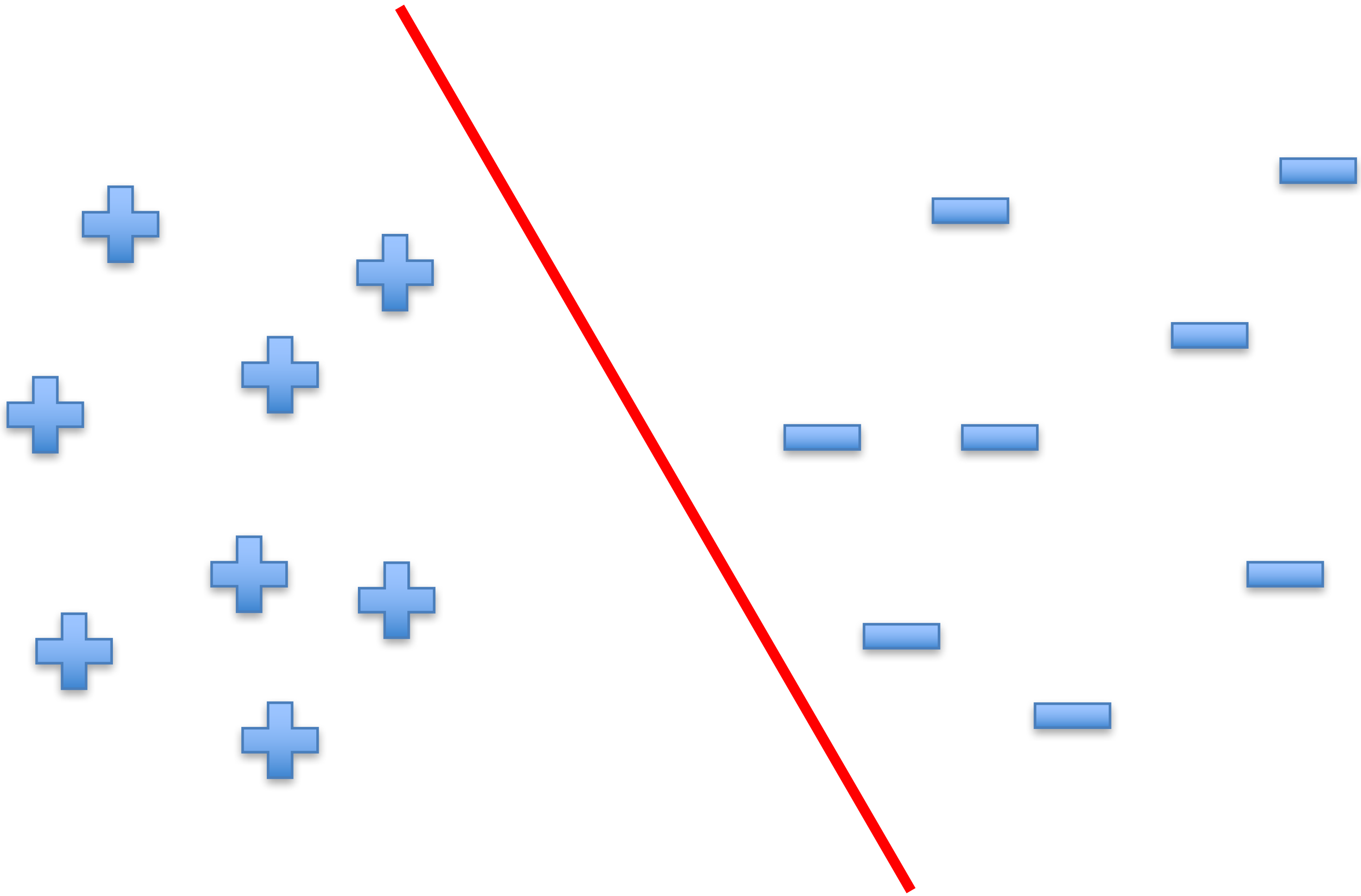
Intuitions



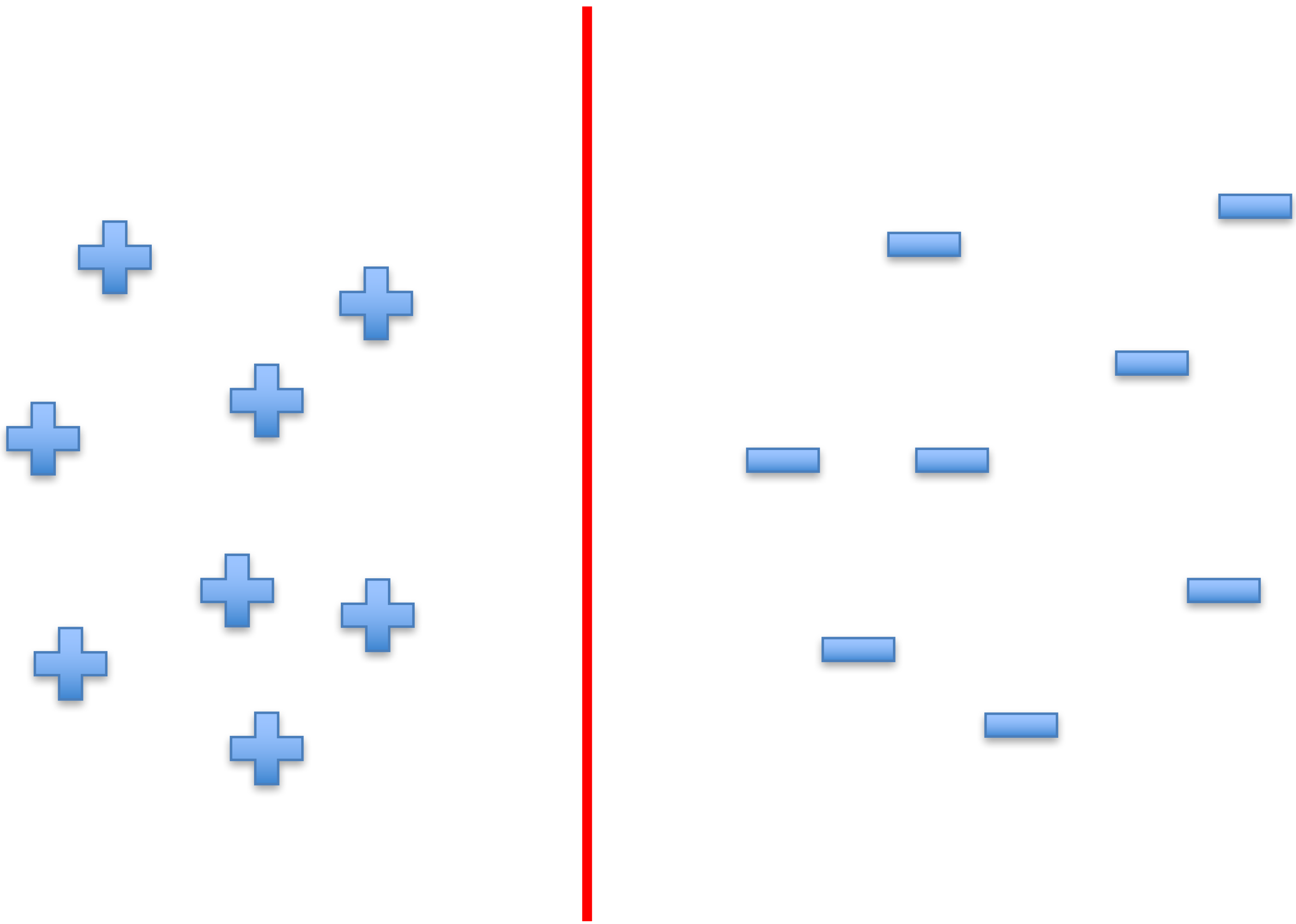
Intuitions



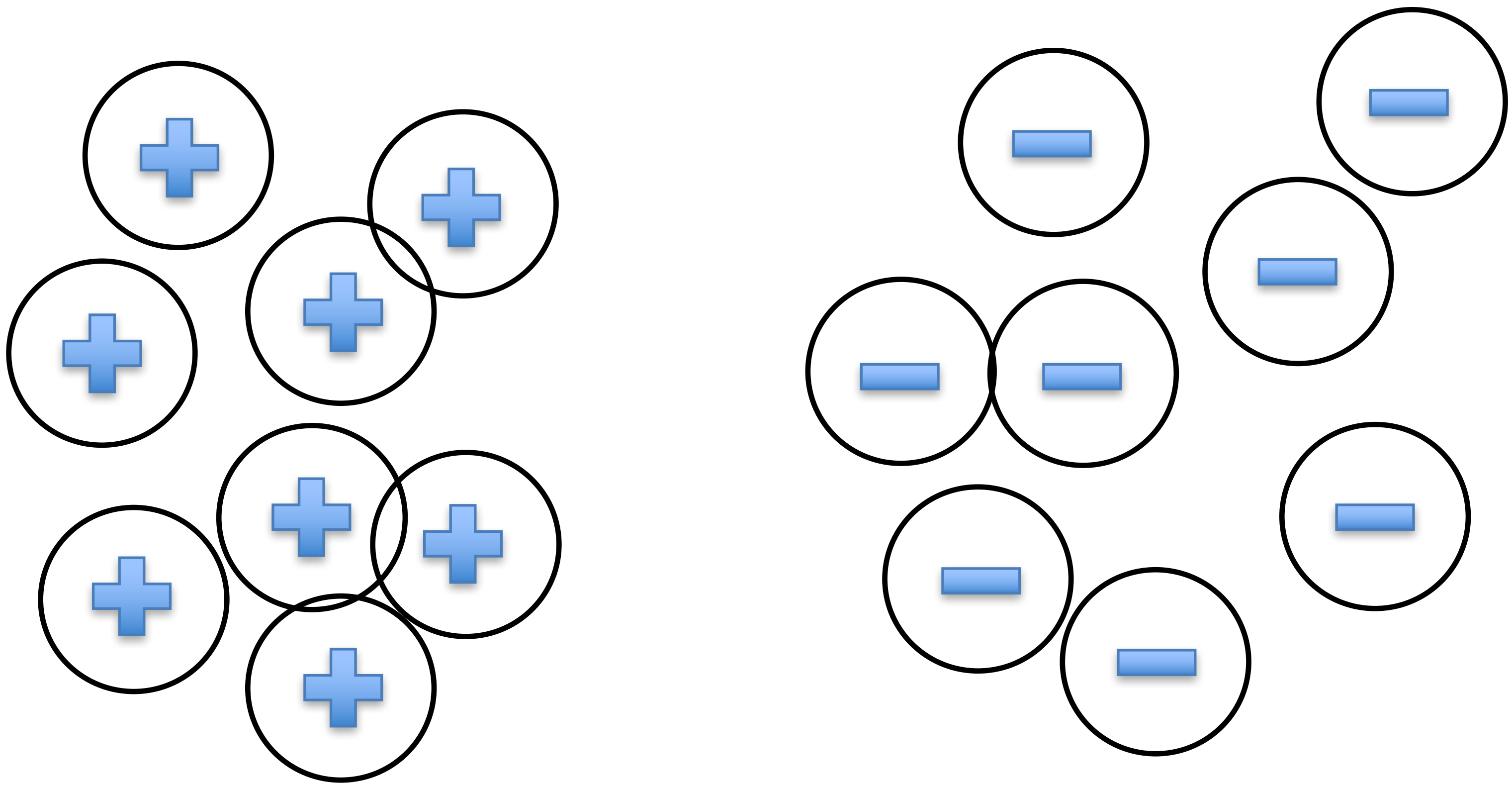
Intuitions



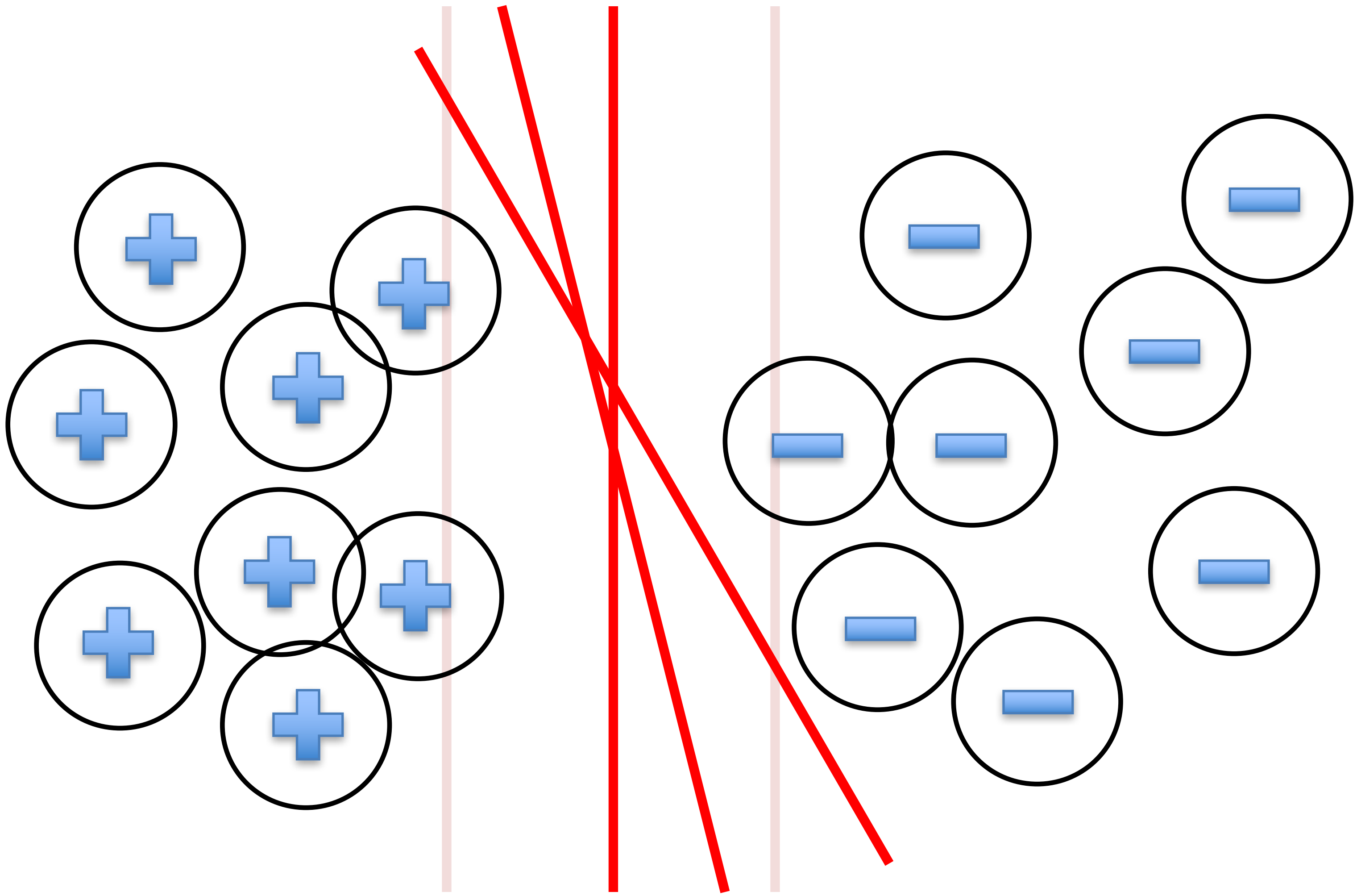
A “Good” Separator



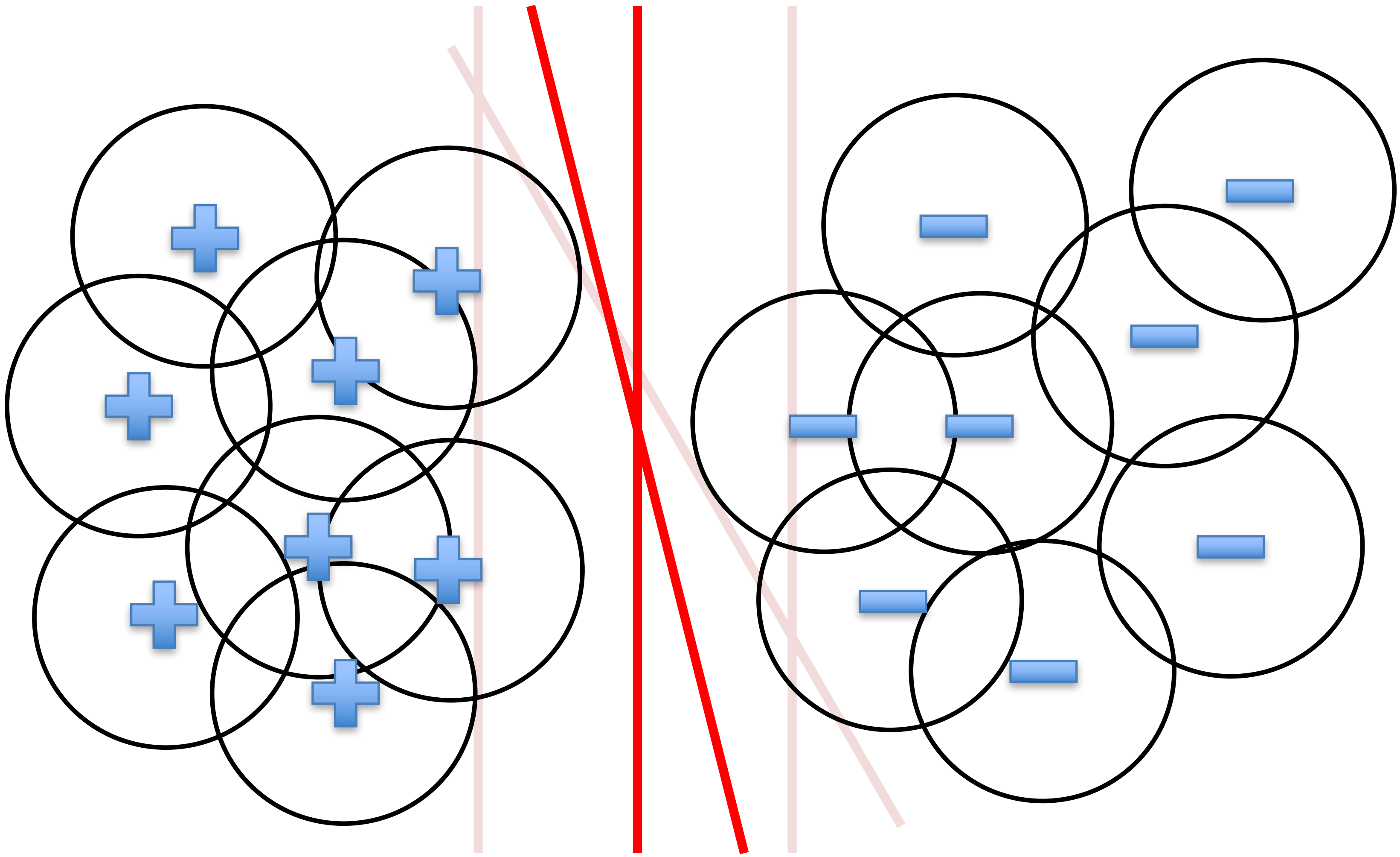
Noise in the Observations



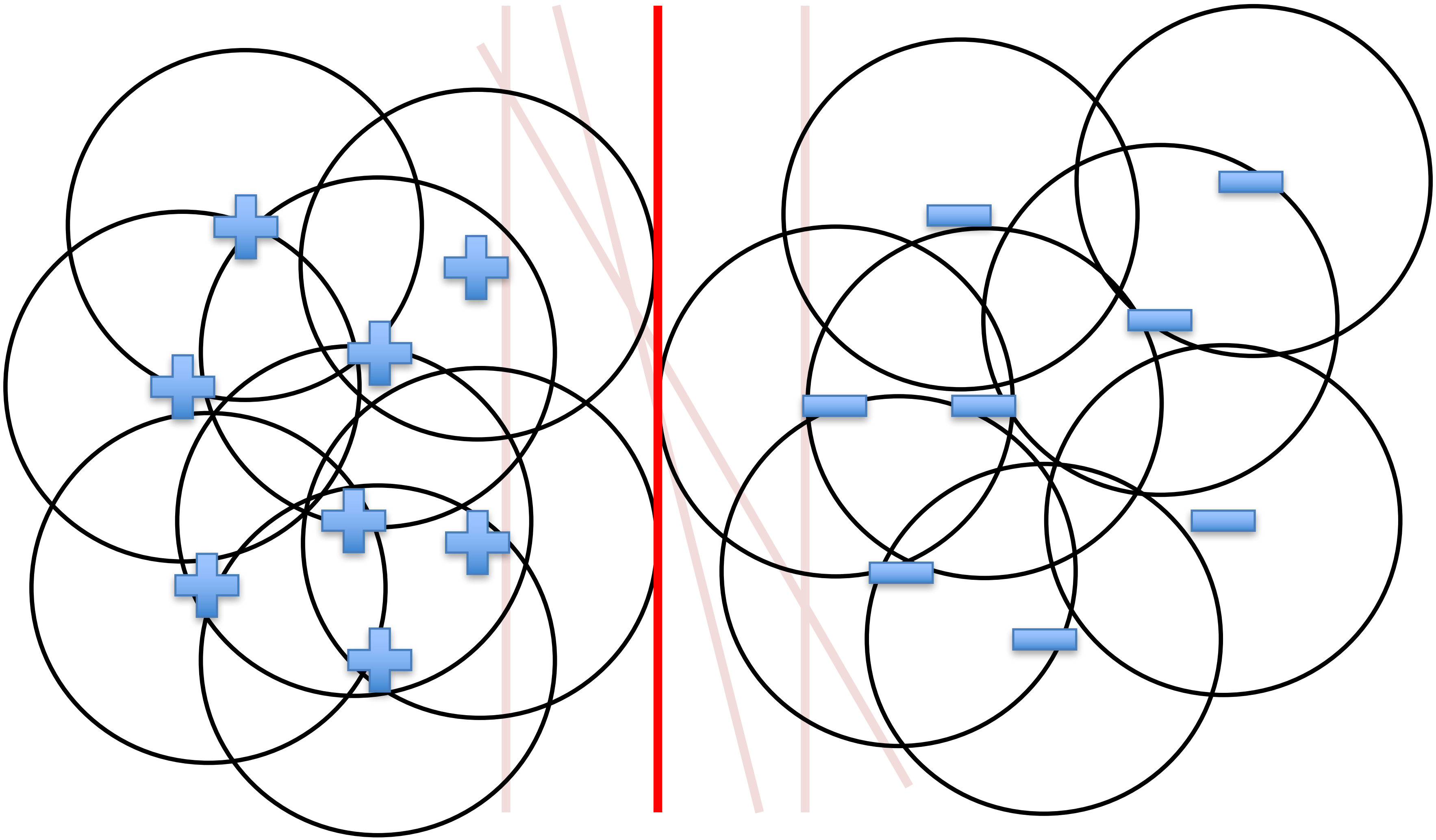
Ruling Out Some Separators



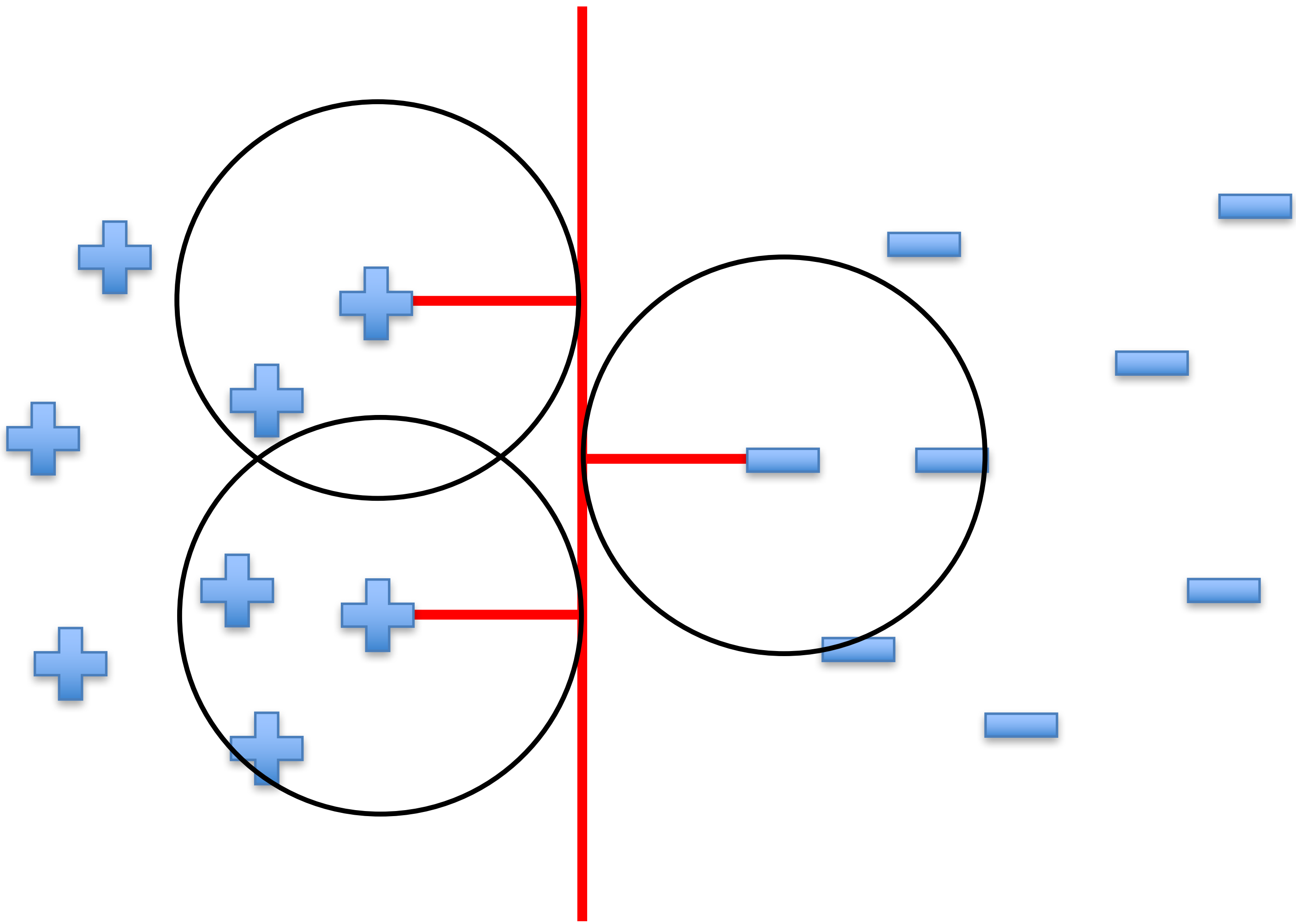
Lots of Noise



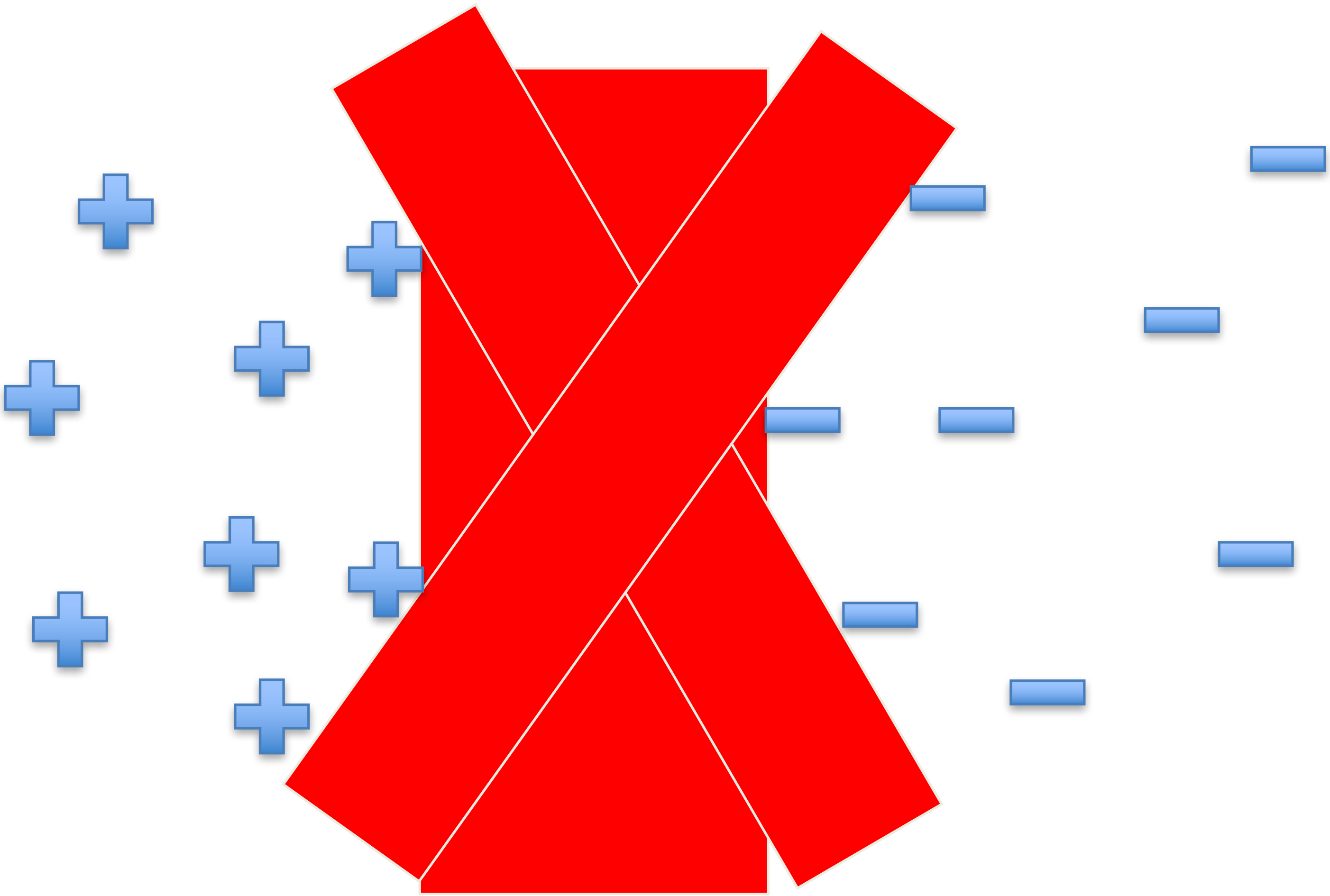
Only One Separator Remains



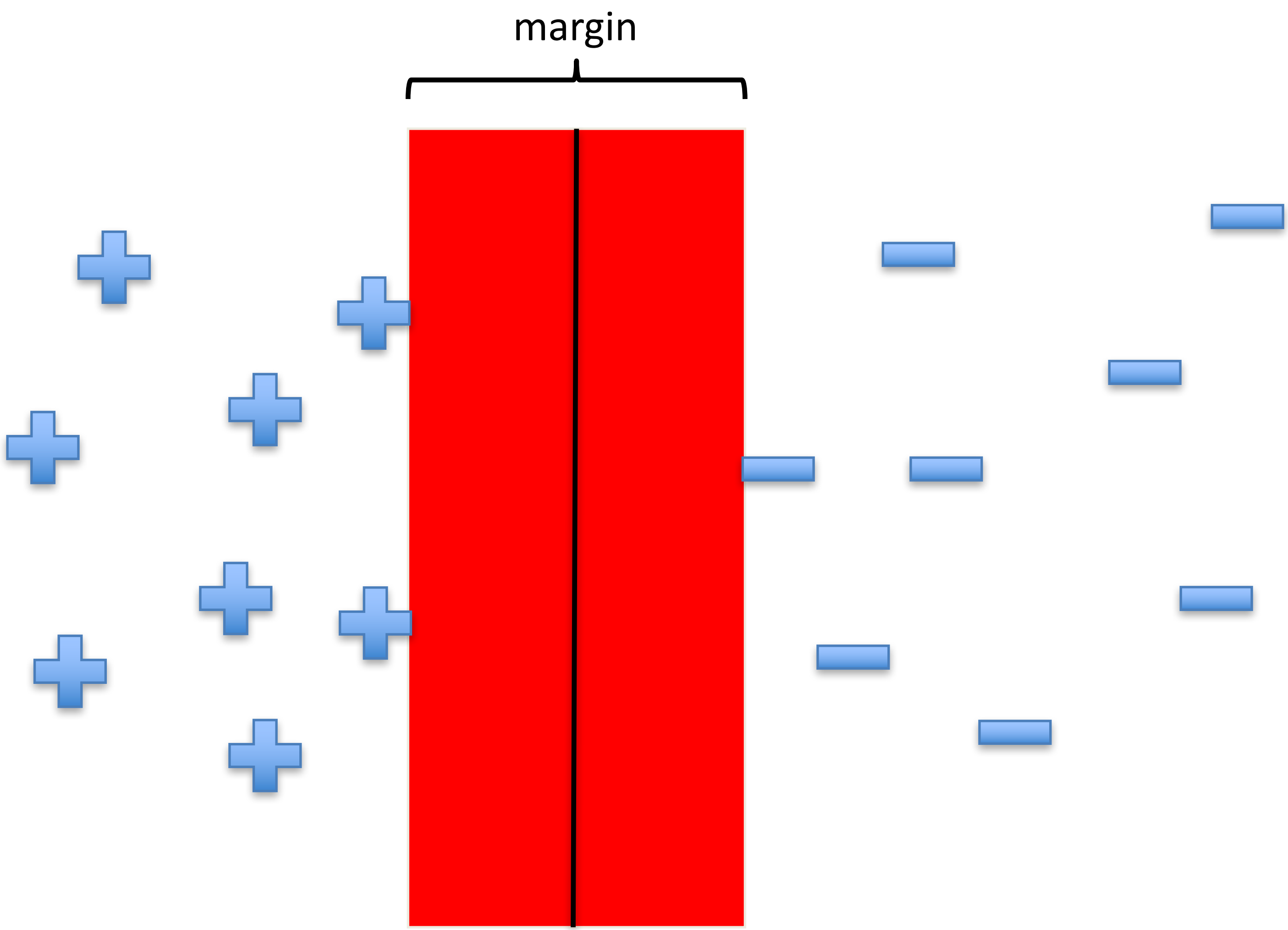
Maximizing the Margin



“Fat” Separators



“Fat” Separators



Why Maximize Margin

Increasing margin reduces *capacity*

- i.e., fewer possible models

Remember 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