

Support Vector Machine & Kernels

Bahan Kuliah SD3104 Machine Learning

Sevi Nurafni

Fakultas Sains dan Teknologi

Universitas Koperasi Indonesia 2024

Kenapa prediksi bisa salah?



Ketidakpastian yang Sebenarnya

- Bayangkan melempar koin yang tidak seimbang.
- Peluang munculnya 'kepala' disebut θ
- Kita mencoba memperkirakan nilai θ
- Jika $\theta > 0.5$, kita memprediksi 'kepala'; jika tidak, kita memprediksi 'ekor'.

Banyak penelitian Machine Learning menangani masalah seperti ini:

- Mempelajari sebuah model.
- Melakukan yang terbaik berdasarkan harapan hasil.

Kenapa prediksi bisa salah?



○ Pengamatan yang Tidak Lengkap

- Ada sesuatu yang diperlukan untuk memprediksi y tetapi tidak ada dalam pengamatan x
- Masalah paritas dengan N
 - x berisi $N-1$ bit (pengamatan yang sulit/tidak lengkap secara keras)
 - x berisi N bit, tetapi model mengabaikan beberapa bit (pengamatan yang tidak lengkap secara lunak).

○ Gangguan dalam Pengamatan

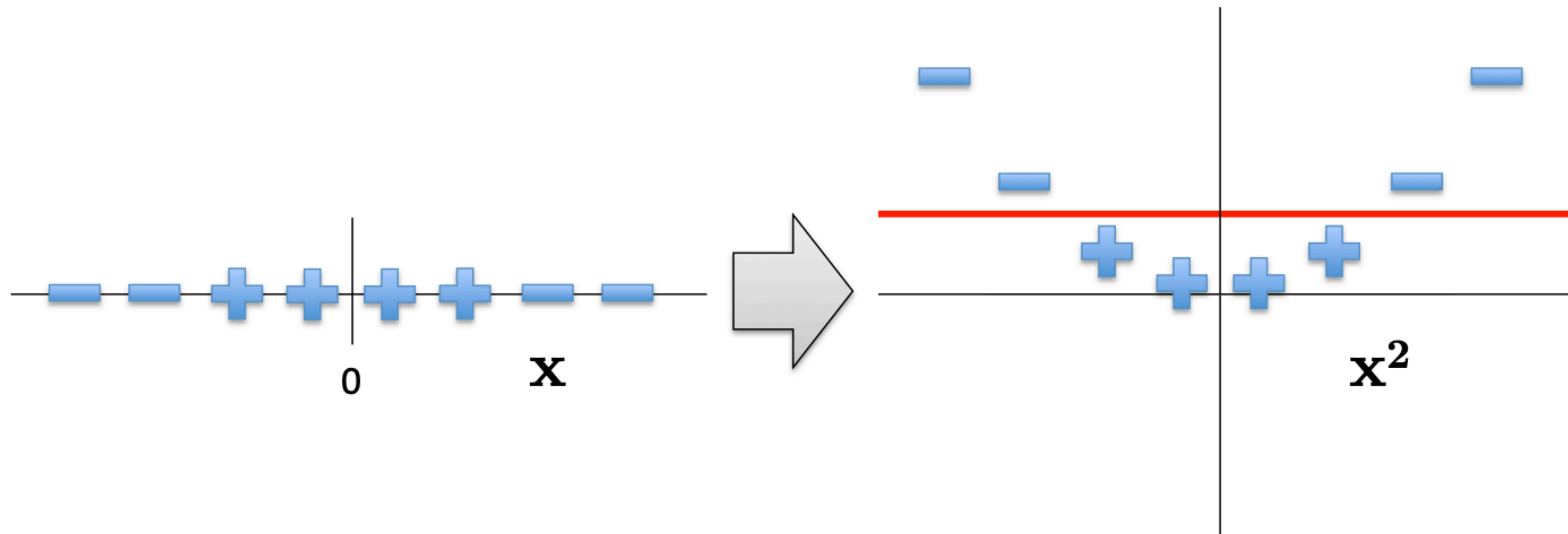
- Kesalahan pengukuran.
- Keterbatasan alat ukur.

Kenapa prediksi bisa salah?

- Ketidakpastian yang Sebenarnya
- Pengamatan yang Tidak Lengkap
 - Sulit/Tidak lengkap secara keras.
 - Tidak lengkap secara lunak.
- Bias Representasi
- Bias Algoritma
- Sumber Daya yang Terbatas

Bias Representasi

- pemilihan fitur yang tepat sangat penting agar model bisa memisahkan data dengan



Support Vector Machines

Doing Really Well with Linear Decision Surfaces

Kelebihan dari SVM



- Generalization yang baik
 - secara teori
 - dan dalam praktiknya
- Bekerja dengan baik meskipun hanya menggunakan sedikit data pelatihan
- Mencari model terbaik secara keseluruhan
- Algoritma yang efisien
- Dapat menggunakan trik kernel

Linear Separator

- Training

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

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

- Parameter model

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

- Hyperlane

$$\theta^\top x = (\theta, x) = 0$$

- Decision Function

$$h(x) = \text{sign}(\theta^\top x) = \text{sign}((\theta, x))$$

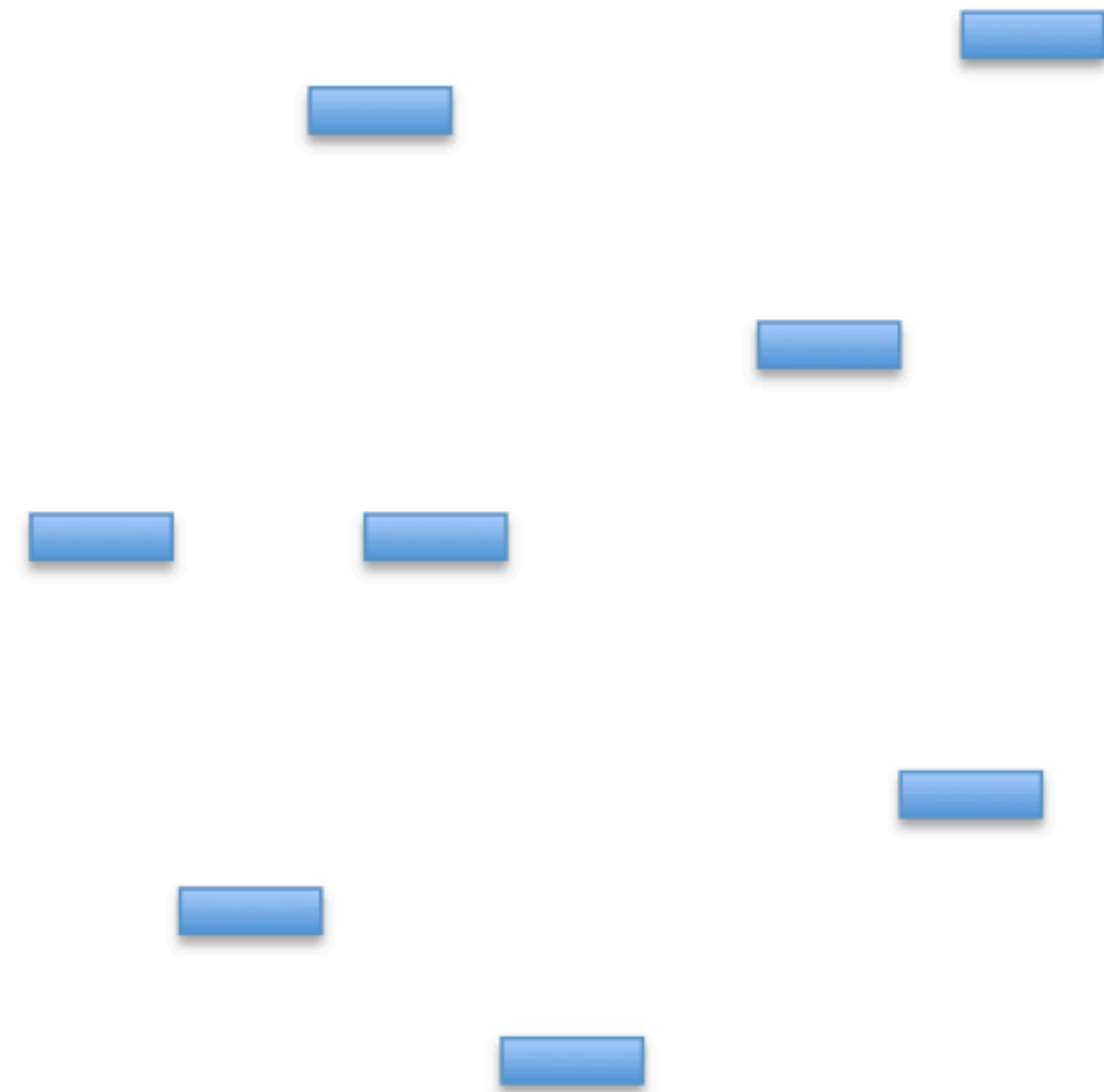
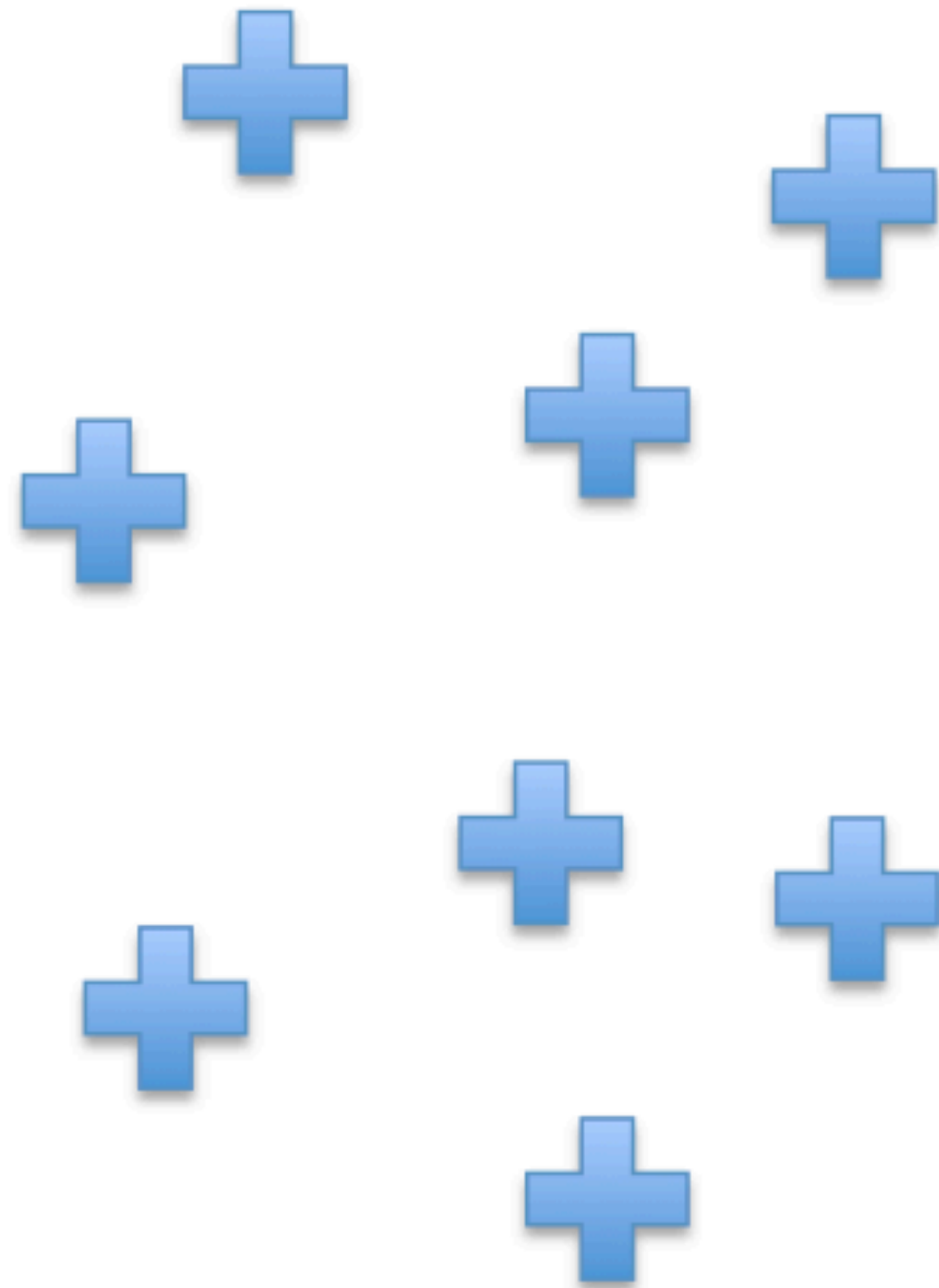
Recall:

Inner (dot) product:

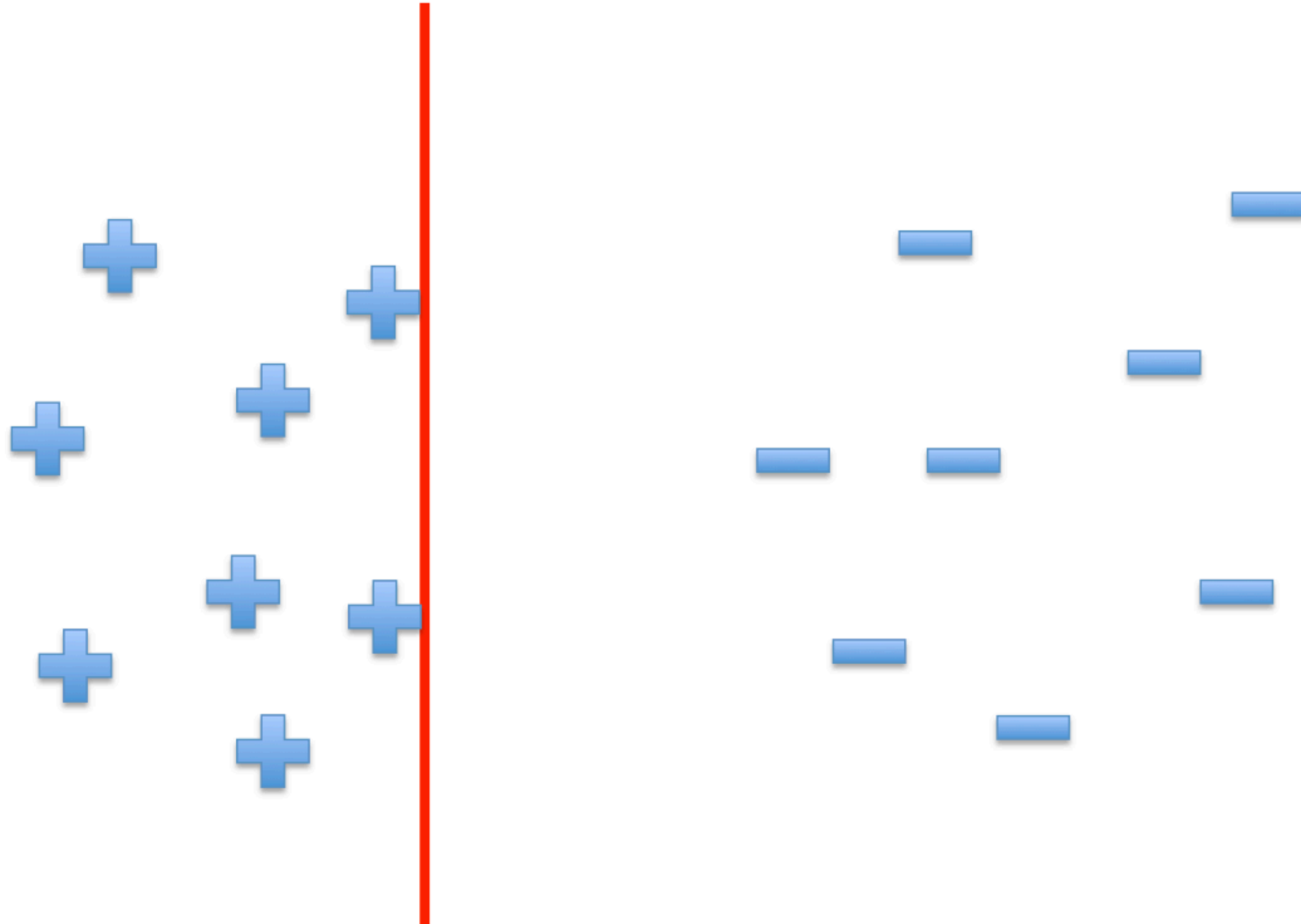
$$\langle \mathbf{u}, \mathbf{v} \rangle = \mathbf{u} \cdot \mathbf{v} = \mathbf{u}^\top \mathbf{v}$$

$$= \sum_i u_i v_i$$

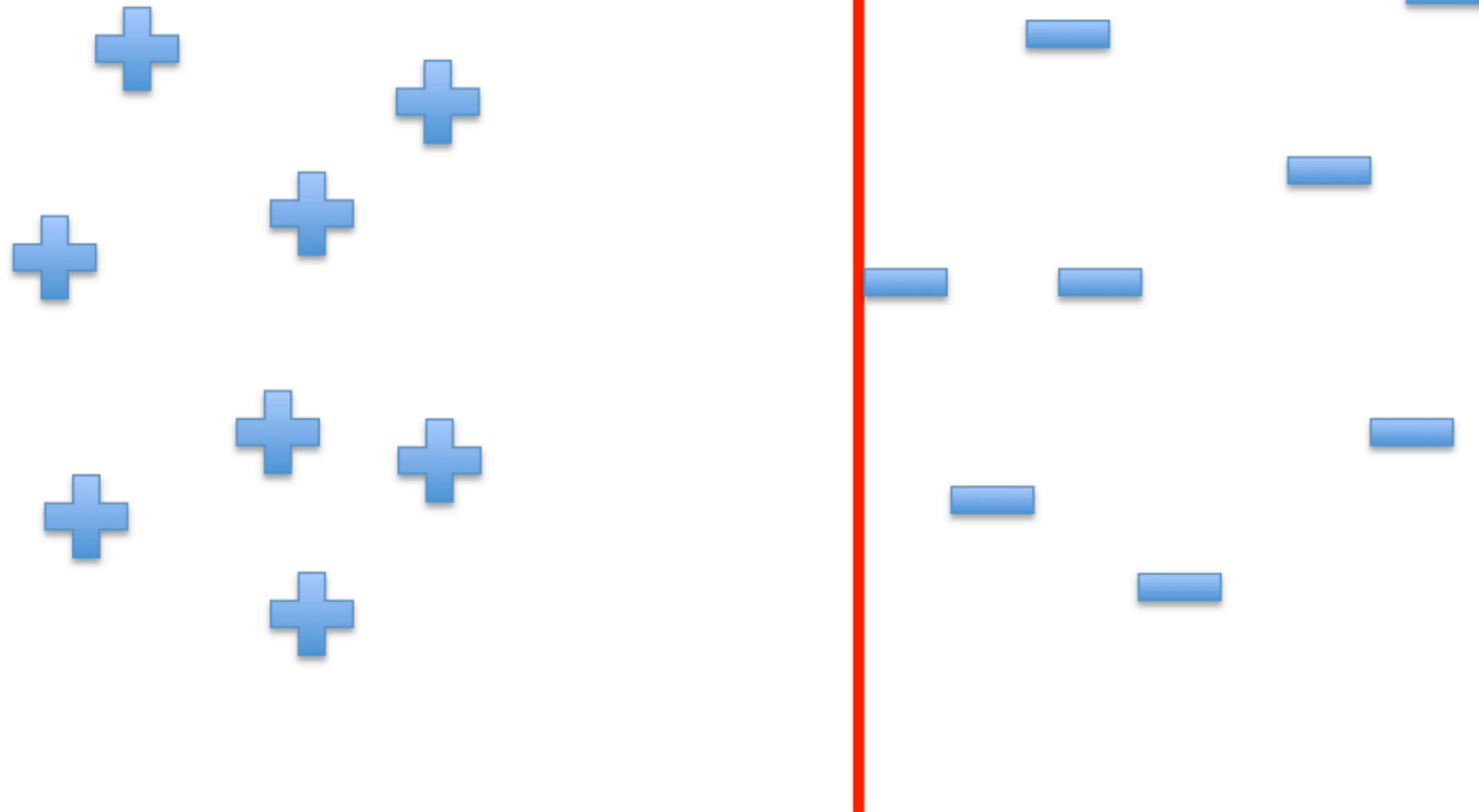
Intuitions



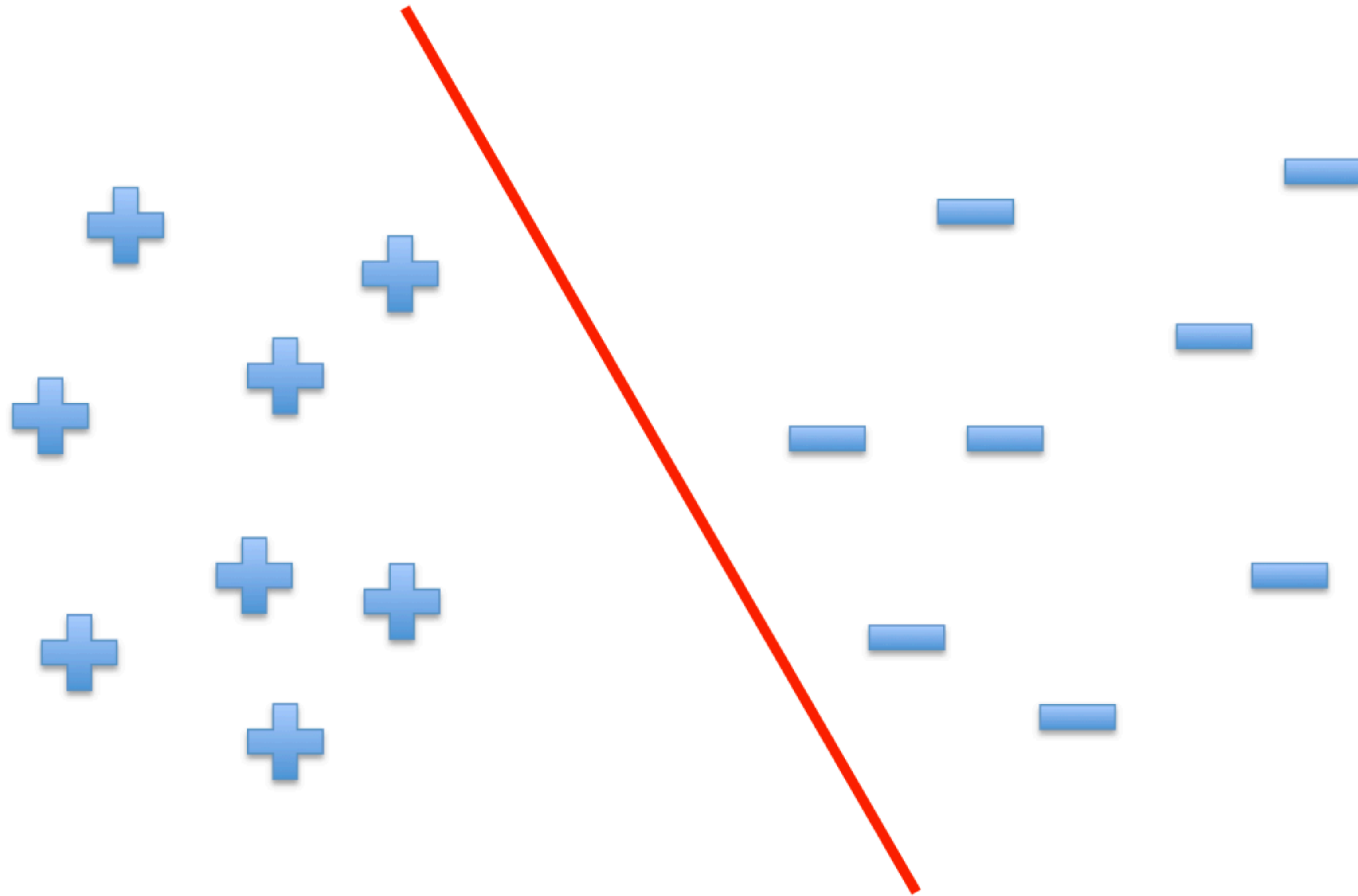
Intuitions



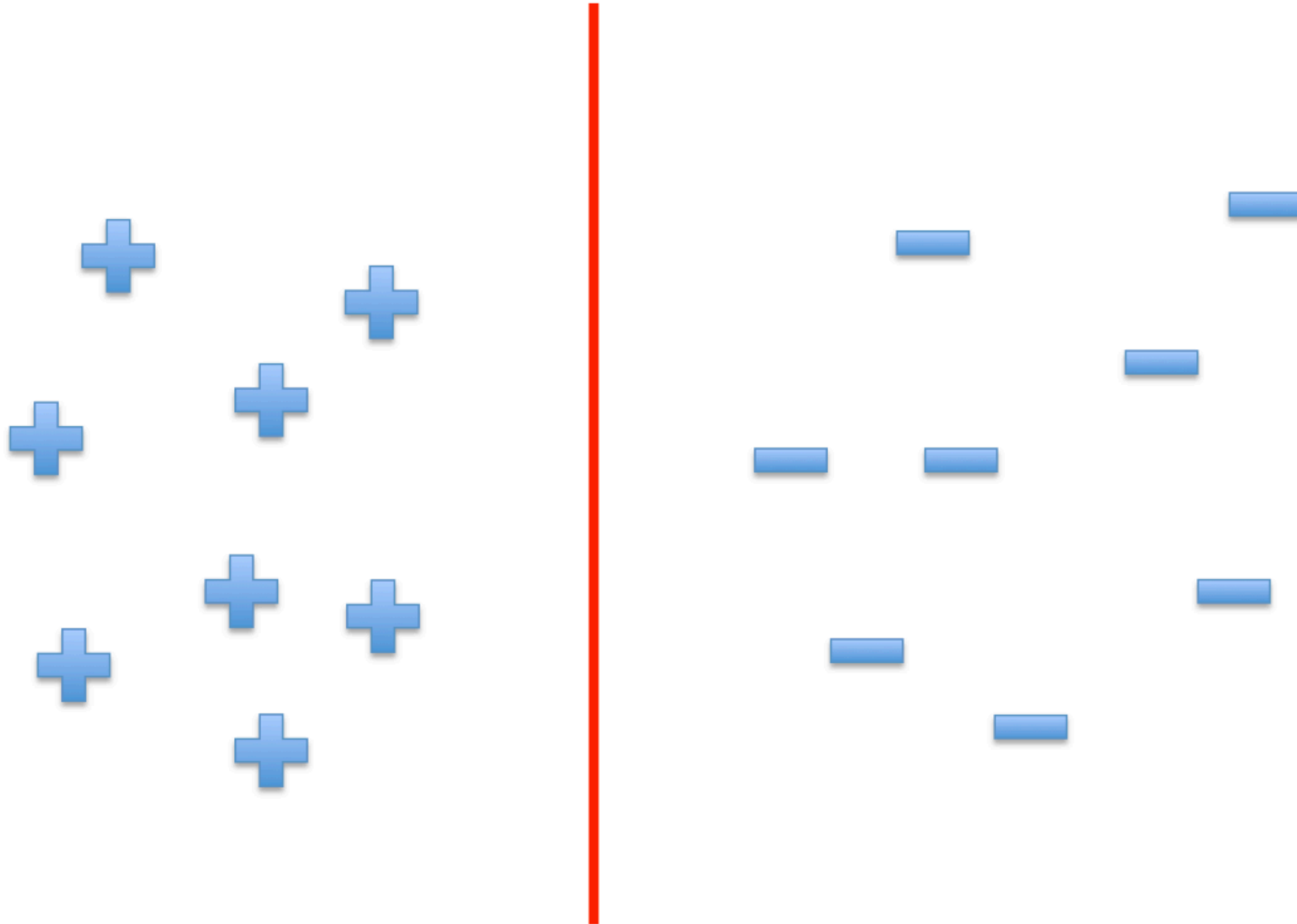
Intuitions



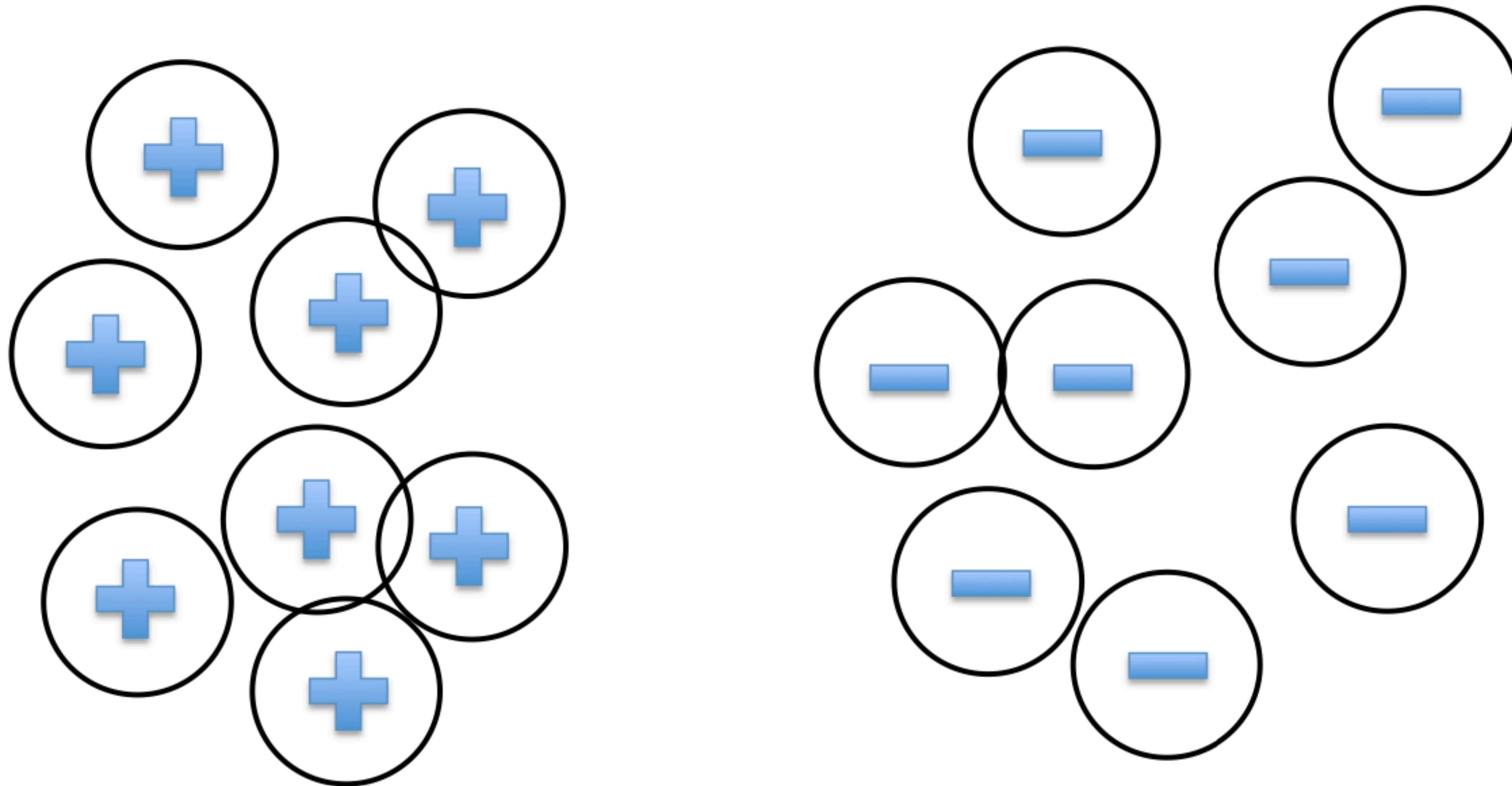
Intuitions



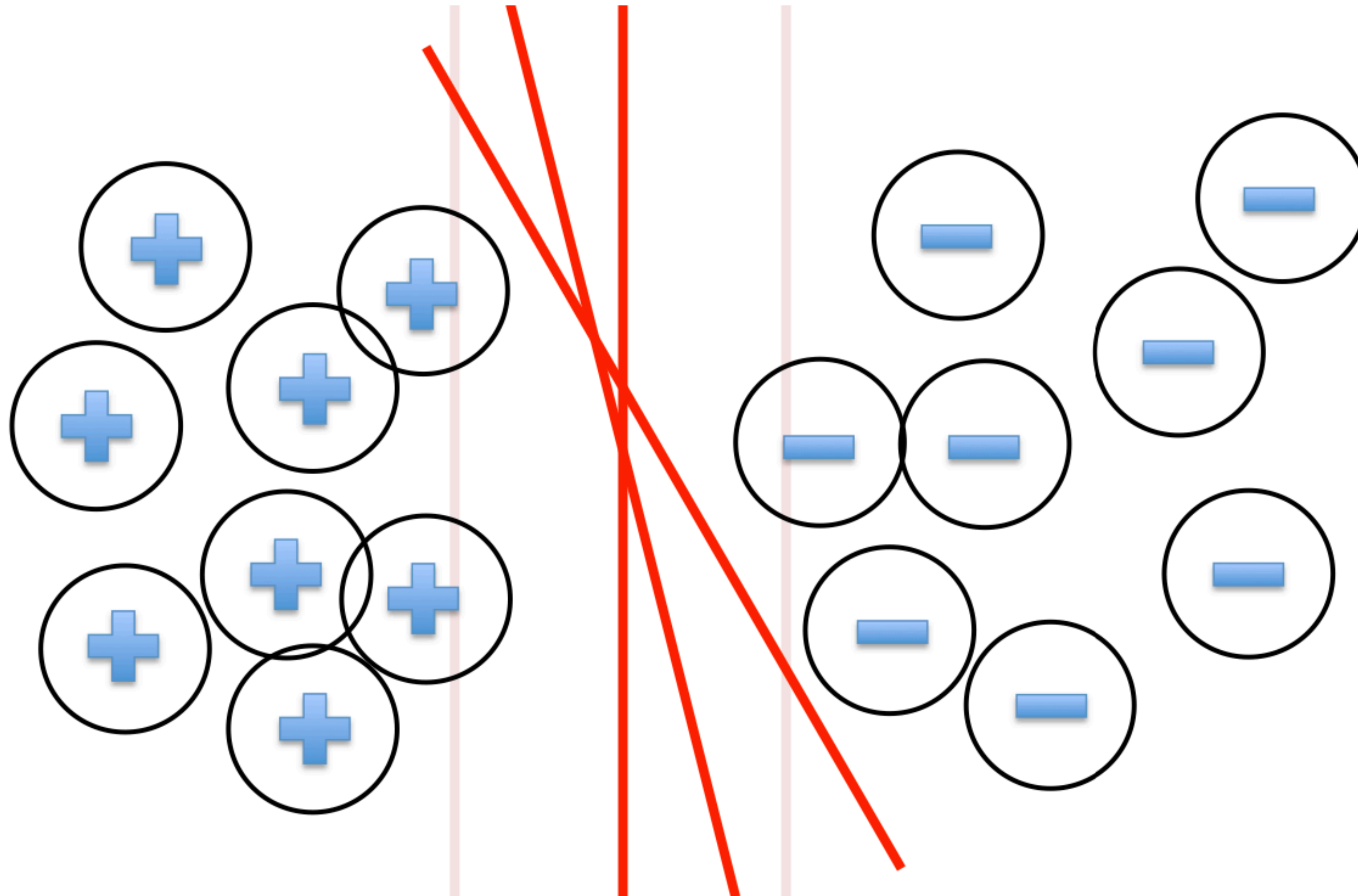
Separator yang “Baik”



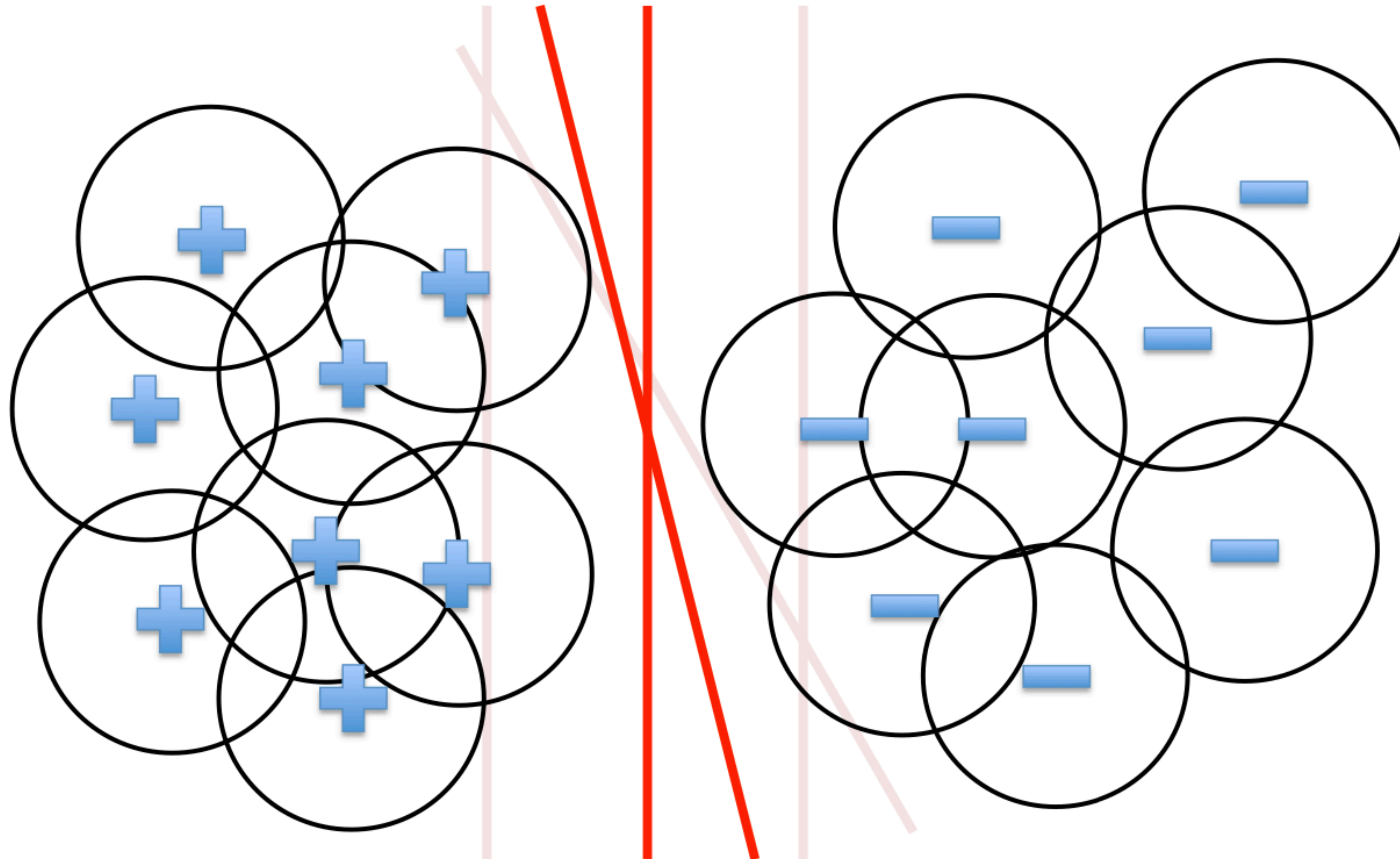
Gangguan dalam Pengamatan



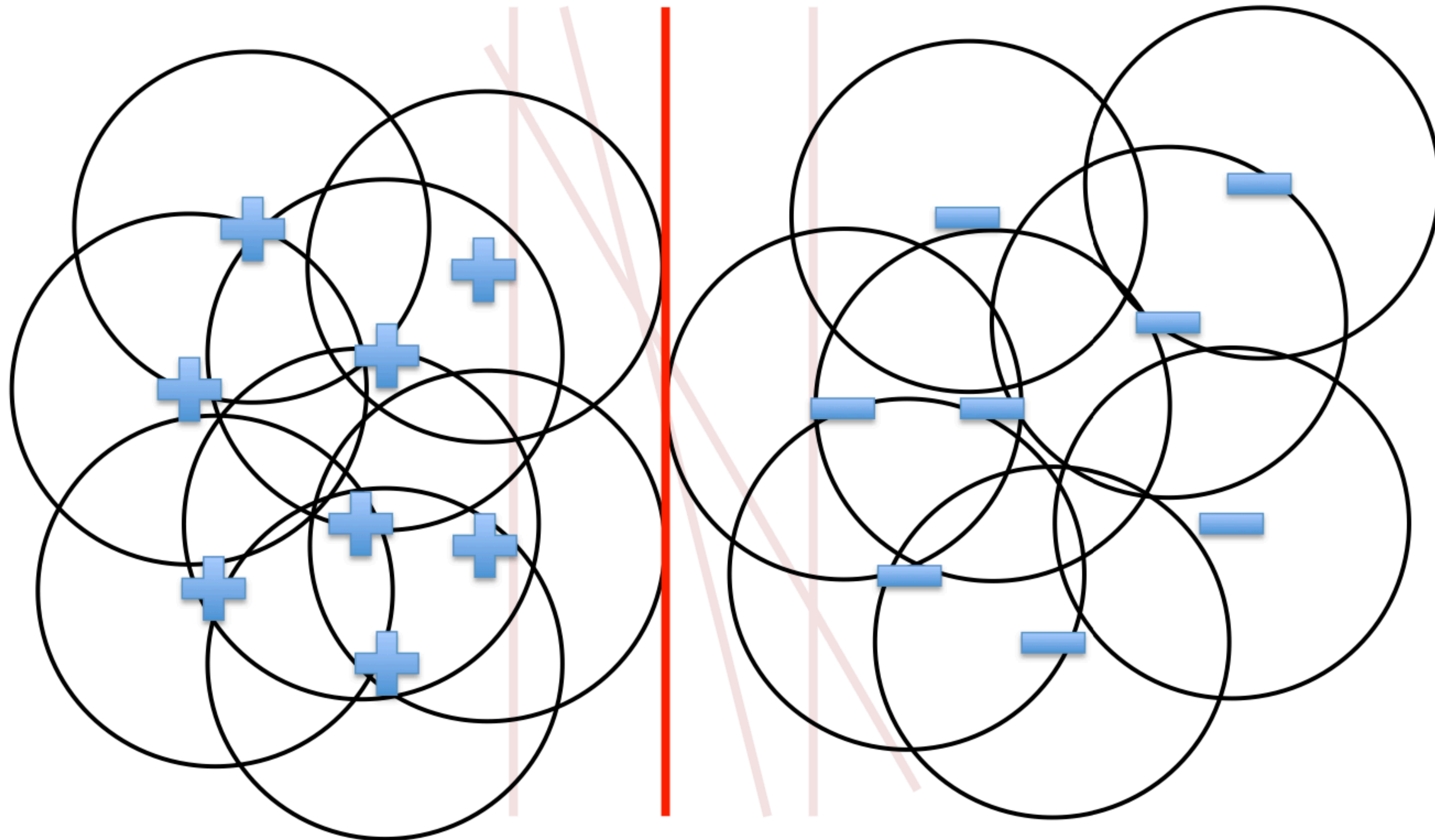
Mengeliminasi beberapa Garis Pemisah



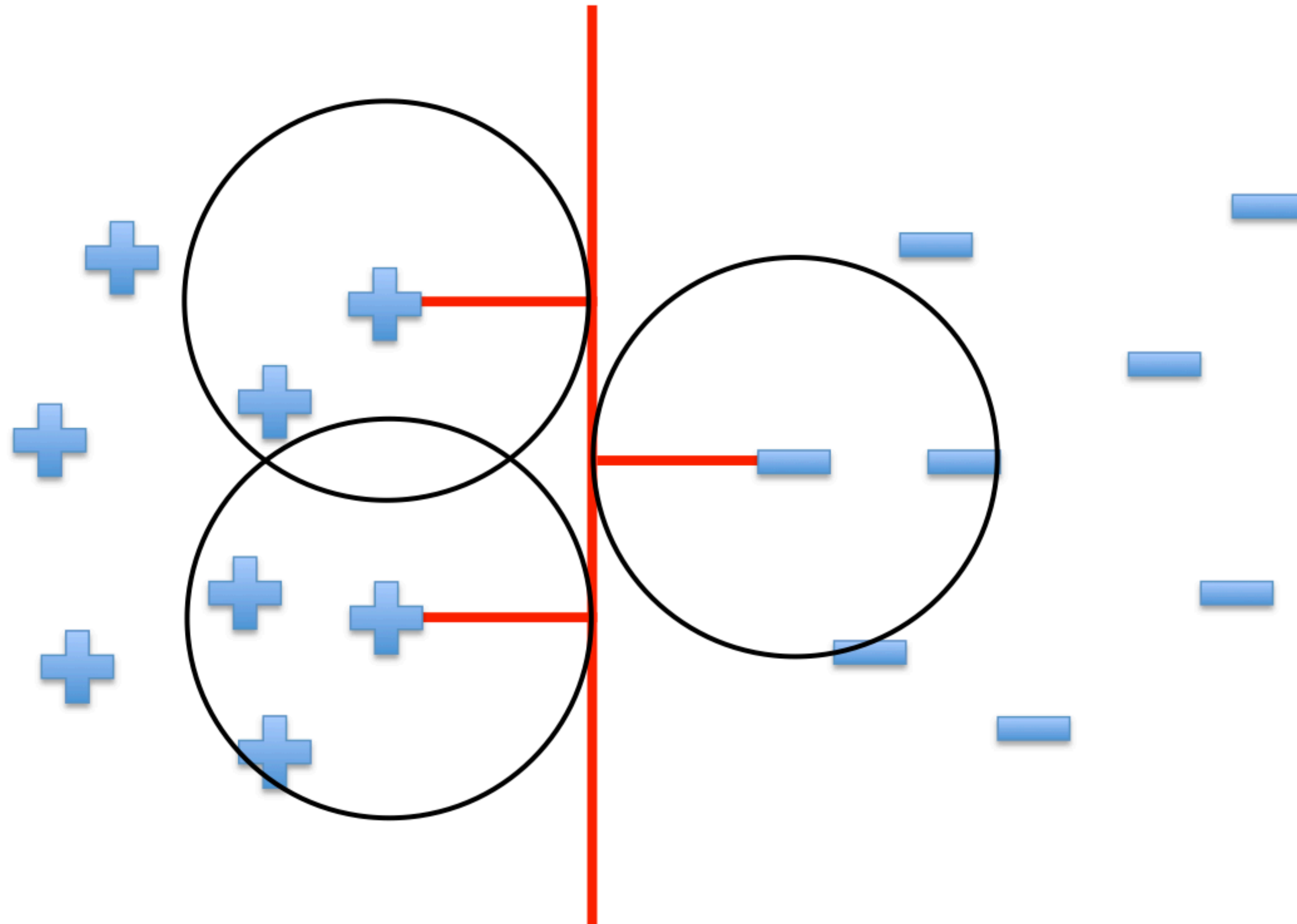
Banyak Noise



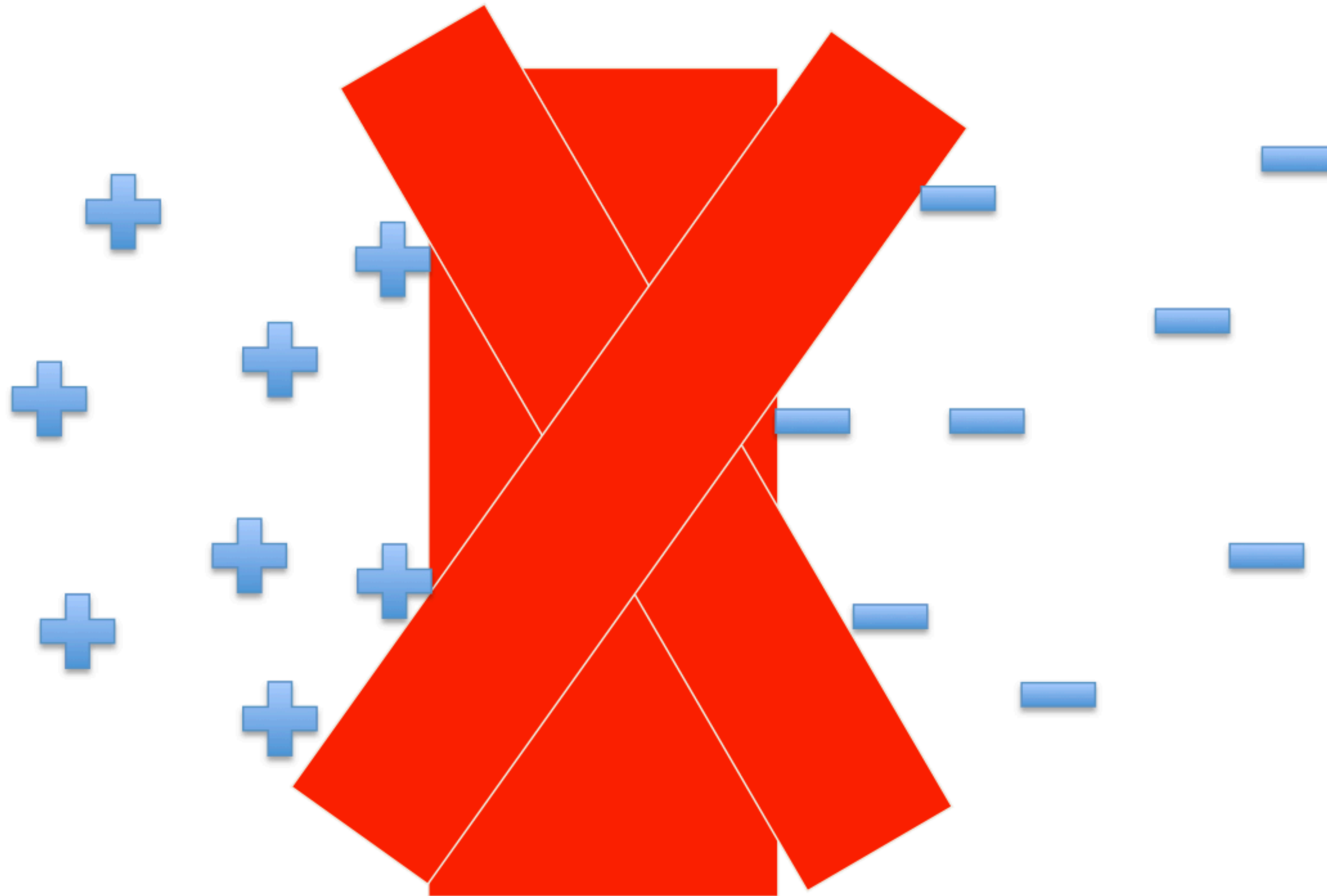
Hanya Tersisa satu Pemisah



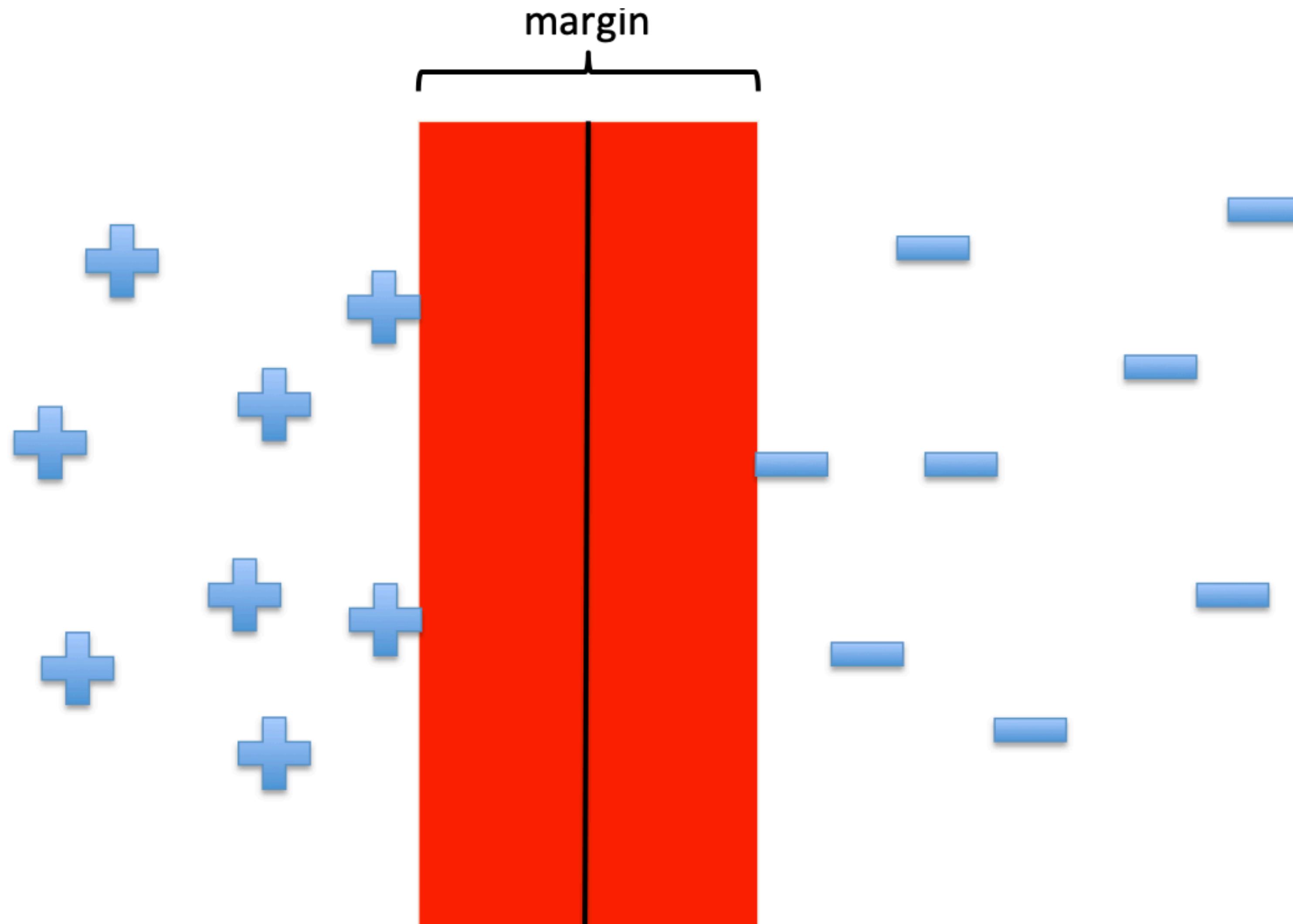
Memaksimalkan Margin



Separator yang “Gemuk”



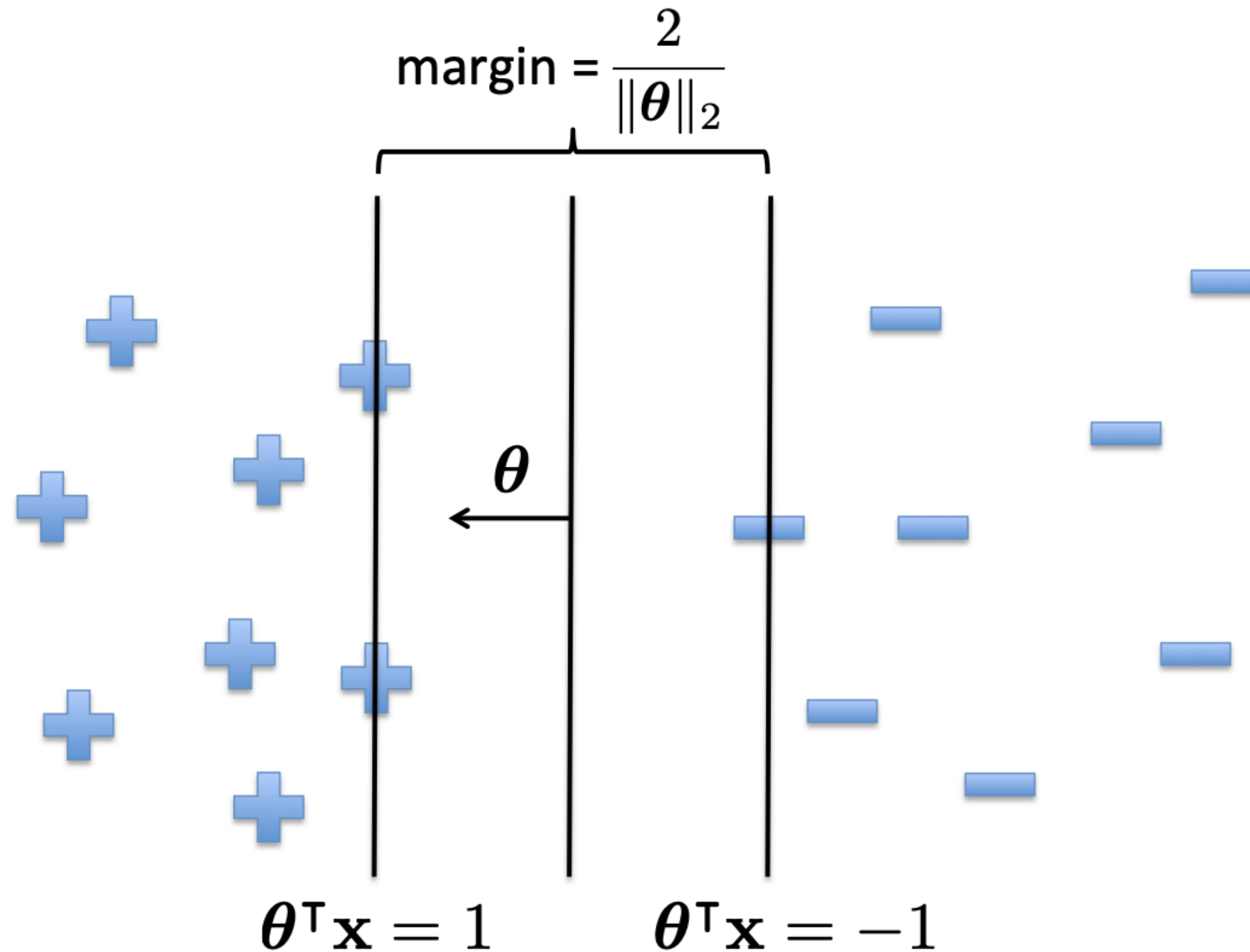
Separator yang “Gemuk”



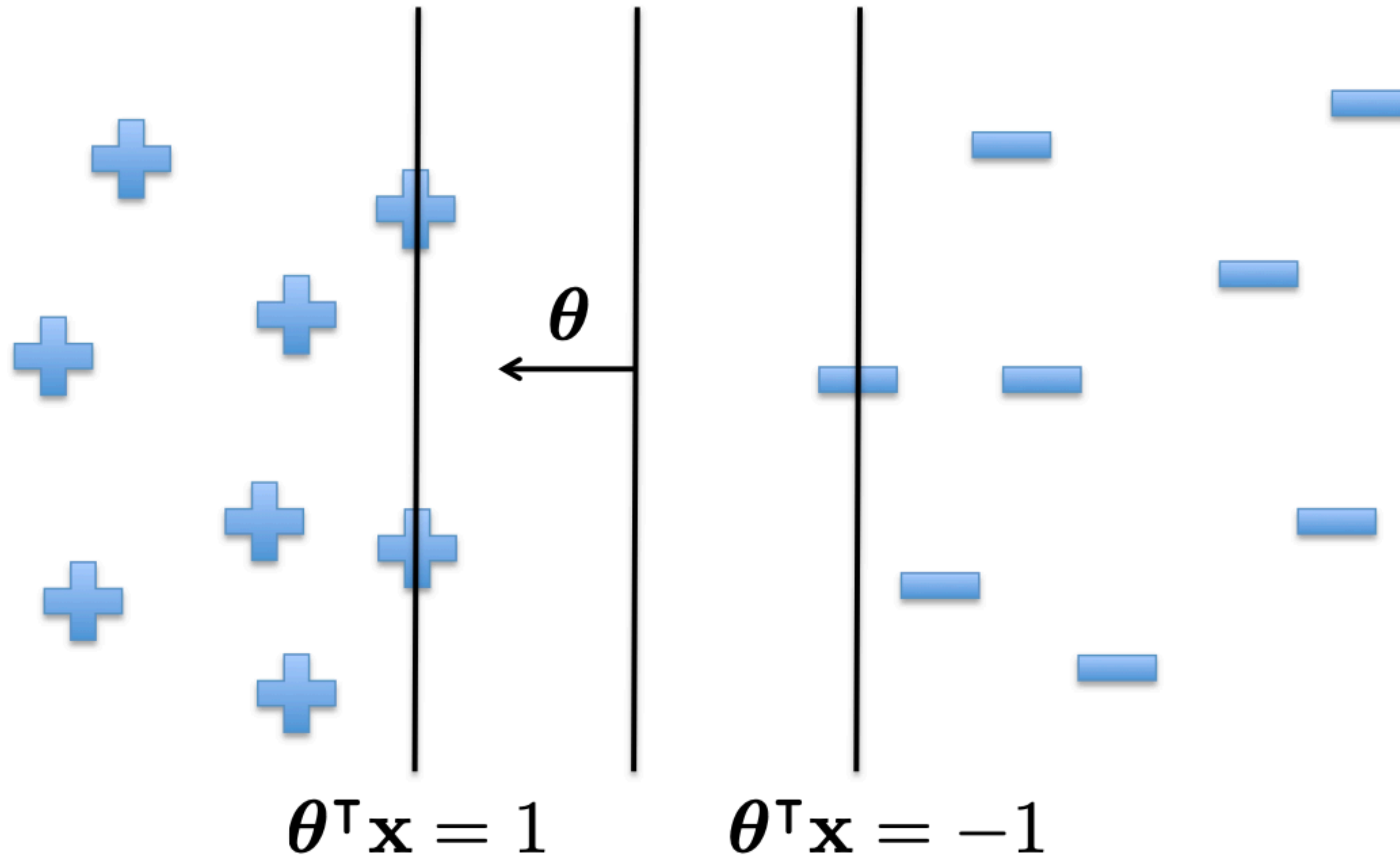
Kenapa Memaksimalkan Margin?

- SVM menjadi lebih tahan terhadap overfitting terutama ketika data memiliki beberapa outlier.
- Data dari kedua kelas berada pada jarak yang aman dari hyperplane, sehingga mengurangi risiko kesalahan klasifikasi.
- mencoba menghindari pembentukan hyperplane yang terlalu kompleks, sehingga lebih sesuai untuk masalah yang tidak linier atau rumit.

Hyperplan dengan Margin Maksimum



Support Vectors



Bagaimana jika Permukannya Non-Linear

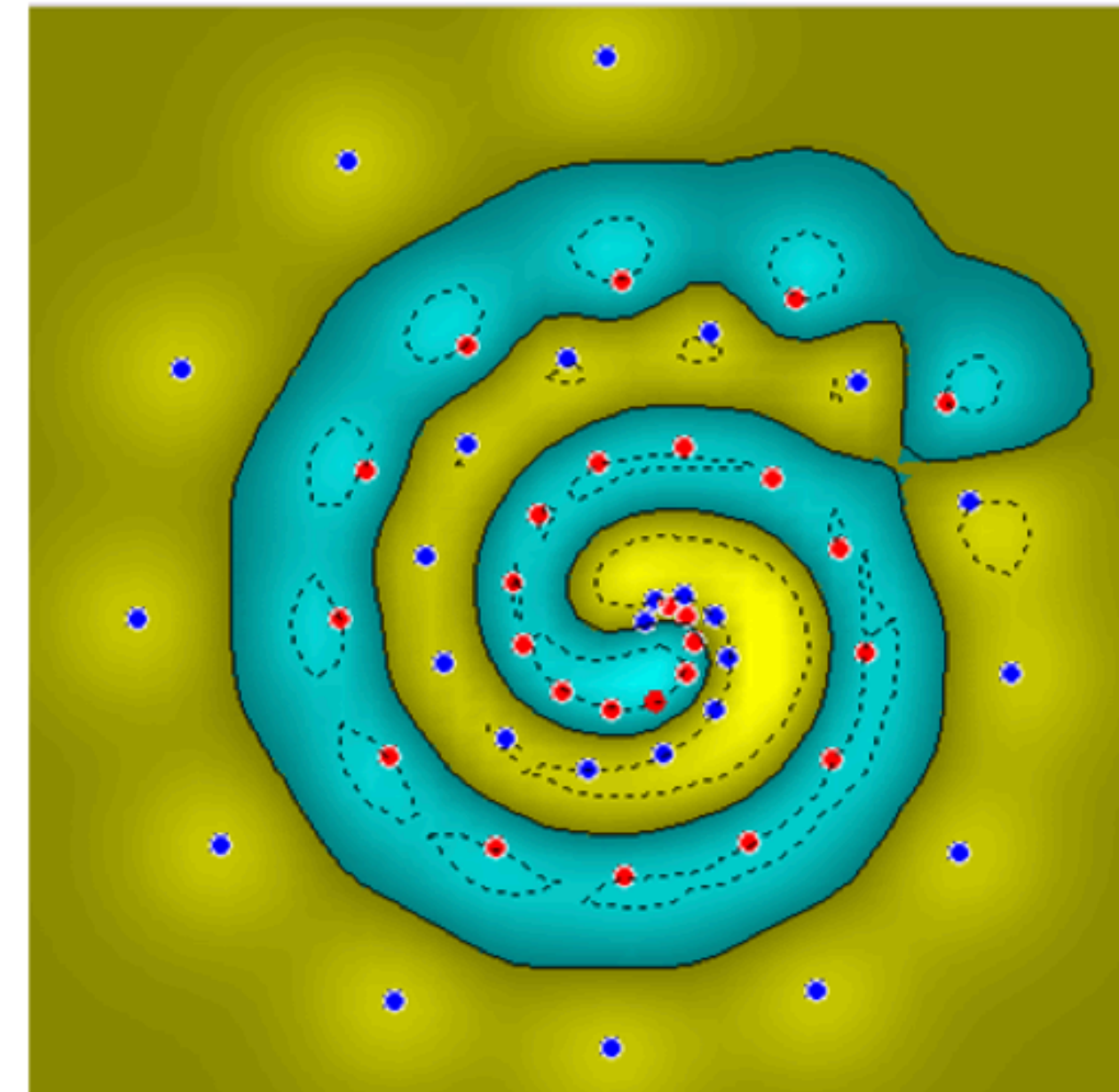
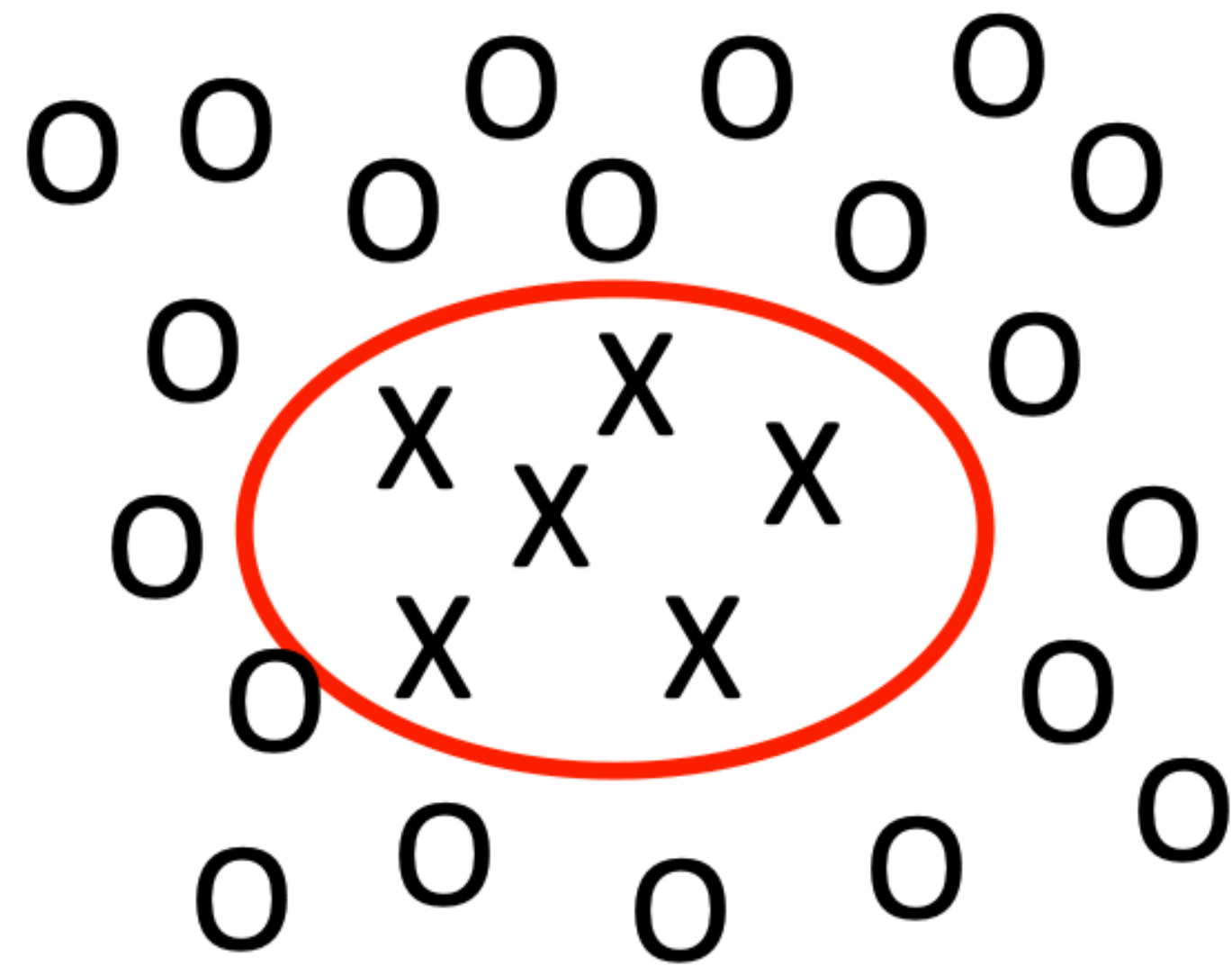
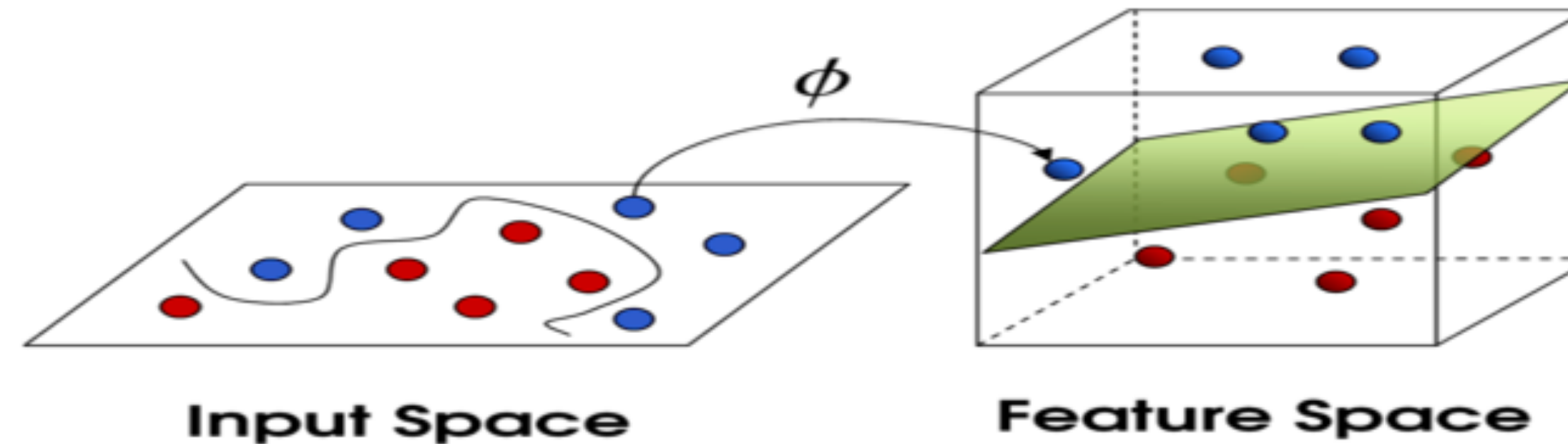


Image from <http://www.atrandomresearch.com/iclass/>

Kernal Methods

Making the Non-Linear Linear

Ketika Garis Pemisah Gagal



$$\Phi : \mathcal{X} \mapsto \hat{\mathcal{X}} = \Phi(\mathbf{x})$$

- Sebagai contoh $x_i \in \mathbb{R}^2$
 - $\Phi([x_{i1}, x_{i2}]) = [x_{i1}, x_{i2}, x_{i1}x_{i2}, x_{i1}^2, x_{i2}^2]$
- Alih-alih mencoba memisahkan data di x_i , kita pindahkan ke $\Phi(x_i)$
 - kita bisa menemukan pemisah non-linear yang akan memisahkan data lebih baik dibandingkan di ruang asli
- Bagaimana jika $\Phi(x_i)$ sangat tinggi?
 - Solusinya: gunakan **Kernel**

Kernel

- Menemukan kernel K dengan

$$K(x_i, x_j) = (\Phi(x_i), \Phi(x_j))$$

- Menghitung $K(x_i, x_j)$ seharusnya jauh lebih efisien dibanding menghitung secara langsung transformasi $\Phi(x_i)$ dan $\Phi(x_j)$
- Gunakan $K(x_i, x_j)$ pada algoritma SVM sebagai pengganti dot product data asli (x_i, x_j)
- Menariknya, hal ini memang memungkinkan! Dengan kernel, kita bisa mengatasi data non-linear dengan cara yang efisien.

Kernel Polynomial

- Didefinisikan

$$K(x_i, x_j) = ((x_i, x_j) + c)^d$$


- Nilai c menentukan seberapa banyak model akan memperhatikan pola-pola sederhana dibandingkan pola-pola yang lebih rumit
- Fungsi $\Phi(x)$ berisi semua suku polinomial hingga derajat d
- Berguna untuk pengenalan pola visual
- Contoh:
 - Gambar 16x16 piksel
 - Mencakup 10^{10} suku polinomial dengan derajat 5
 - Fungsi $\Phi(x)$ tidak perlu dihitung secara eksplisit

Kernel Trick

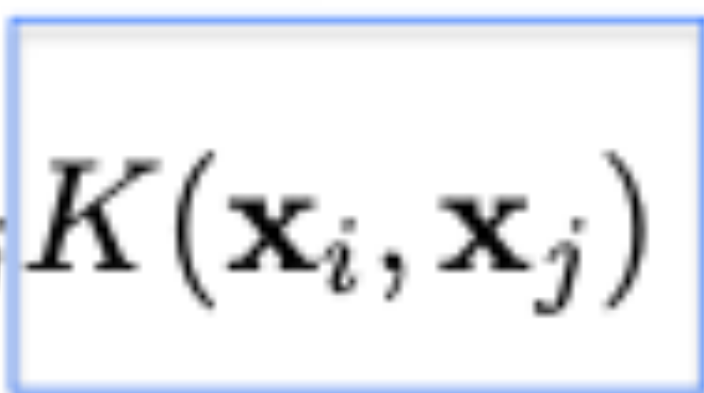


"Jika ada sebuah algoritma yang dirumuskan menggunakan kernel positif definit K_1 , kita bisa membuat algoritma alternatif dengan mengganti K_1 dengan kernel positif definit lainnya, K_2 ."

Memasukan Kernel ke SVM

$$J(\boldsymbol{\alpha}) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle$$




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


$$\text{s.t. } \alpha_i \geq 0 \quad \forall i$$

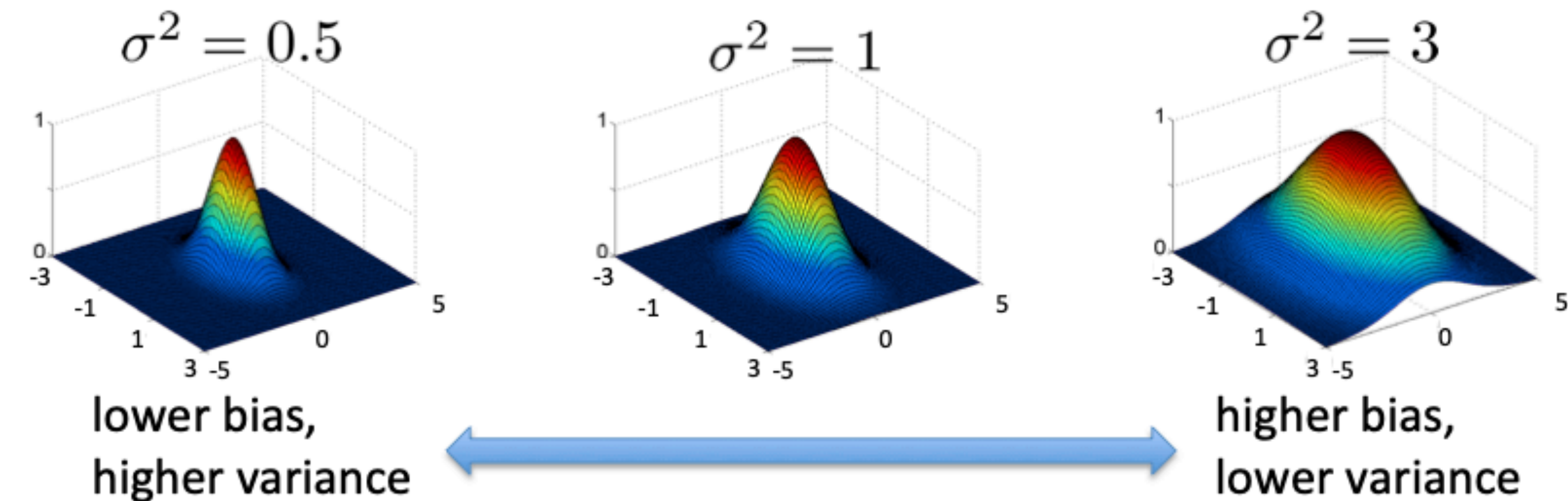
$$\sum_i \alpha_i y_i = 0$$

Kernel Gaussian

- Disebut juga Radial Basis Function (RBF) kernel

$$K(x_i, x_j) = \exp \frac{-||x_i - x_j||_2^2}{2\sigma^2}$$

- Menghasilkan nilai 1 ketika $x_i = x_j$
- Nilai akan turun hingga 0 seiring dengan meningkatnya jarak.
- Catatan: Pastikan melakukan feature scaling terlebih dahulu sebelum menggunakan Gaussian Kernel.



Kernel Sigmoid

$$K(x_i, x_j) = \tanh(\alpha x_i^\top x_j + c)$$

- Jaringan saraf (neural networks) menggunakan sigmoid sebagai fungsi aktivasi
- SVM dengan kernel sigmoid setara dengan perceptron 2-layer

Kernel Cosine Similarity

$$K(x_i, x_j) = \frac{x_i^\top x_j}{||x_i|| ||x_j||}$$

- Pilihan populer untuk mengukur kesamaan dokumen teks
- Norma L_2 memproyeksikan vektor ke dalam lingkup satuan; hasil perkalian dot-nya adalah kosinus dari sudut di antara kedua vektor

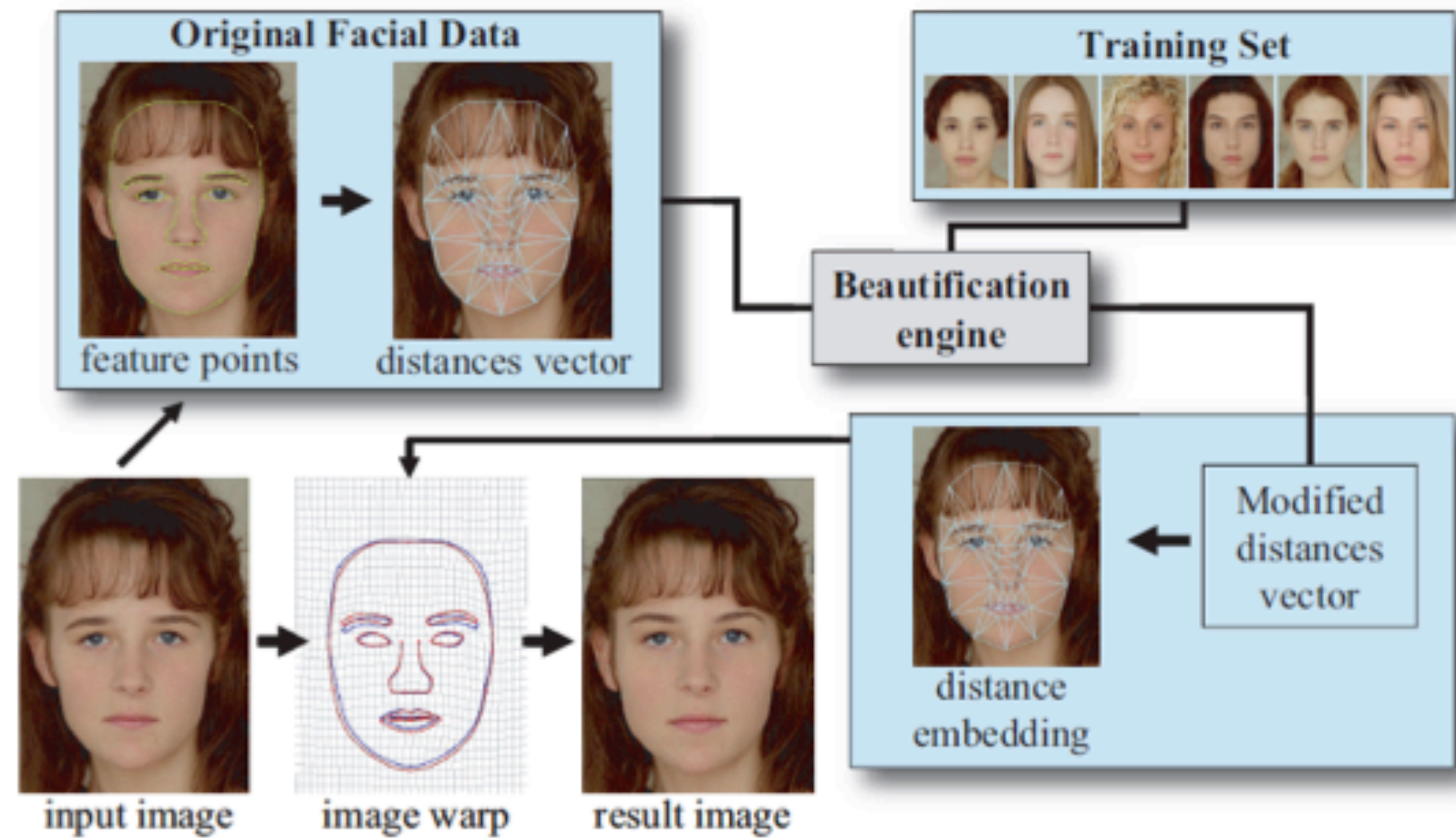
Kernel Chi-squared

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp \left(-\gamma \sum_k \frac{(x_{ik} - x_{jk})^2}{x_{ik} + x_{jk}} \right)$$

- Banyak digunakan dalam aplikasi *computer vision*
- *Chi-squared* mengukur jarak antara distribusi probabilitas
- Data diasumsikan bernilai non-negatif, sering kali dengan norma L_1 sebesar 1

Application: Automation Photo Retouching

(Leyvand et al., 2008)



SVM vs Logistic Regression

(Saran dari Andrew Ng)

- n = jumlah contoh pelatihan d = jumlah fitur
- Jika d besar (relatif terhadap n)
 - Contoh $d > n$ dengan $d = 10.000$, $n = 10 - 1.000$
 - Gunakan regresi logistik atau SVM dengan kernel linear
- Jika d kecil (hingga 1.000), n menengah (hingga 10.000)
 - Gunakan SVM dengan kernel Gaussian
- Jika d kecil (hingga 1.000), n menengah (hingga 50.000+)
 - Tambahkan lebih banyak fitur, lalu gunakan logistic regression atau SVM tanpa kernel

Kesimpulan

- SVM menemukan pemisah linear yang optimal.
- *Kernel trick* membuat SVM mampu mempelajari pemisah yang tidak linear.
- Kelebihan SVM:
 - Memiliki performa yang baik, baik secara teori maupun praktek.
 - Mendukung berbagai jenis kernel.
- Kekurangan SVM:
 - Relatif "lambat" dalam melatih/memprediksi jika dataset sangat besar (meskipun masih cukup cepat!).
 - Memerlukan pemilihan kernel yang tepat (dan pengaturan parameter kernel).

Video Andrew Ng



- SVM : <https://www.youtube.com/watch?v=IDwow4aOrtg&list=PLoROMvodv4rMiGQp3WXShtMGgzqpfVfbU&index=7>
- Kernel : <https://www.youtube.com/watch?v=8NYoQiRANpg&list=PLoROMvodv4rMiGQp3WXShtMGgzqpfVfbU&index=8>
- The math: <https://www.youtube.com/watch?v=wBVSbVktLIY>

**SELAMAT
BELAJAR**