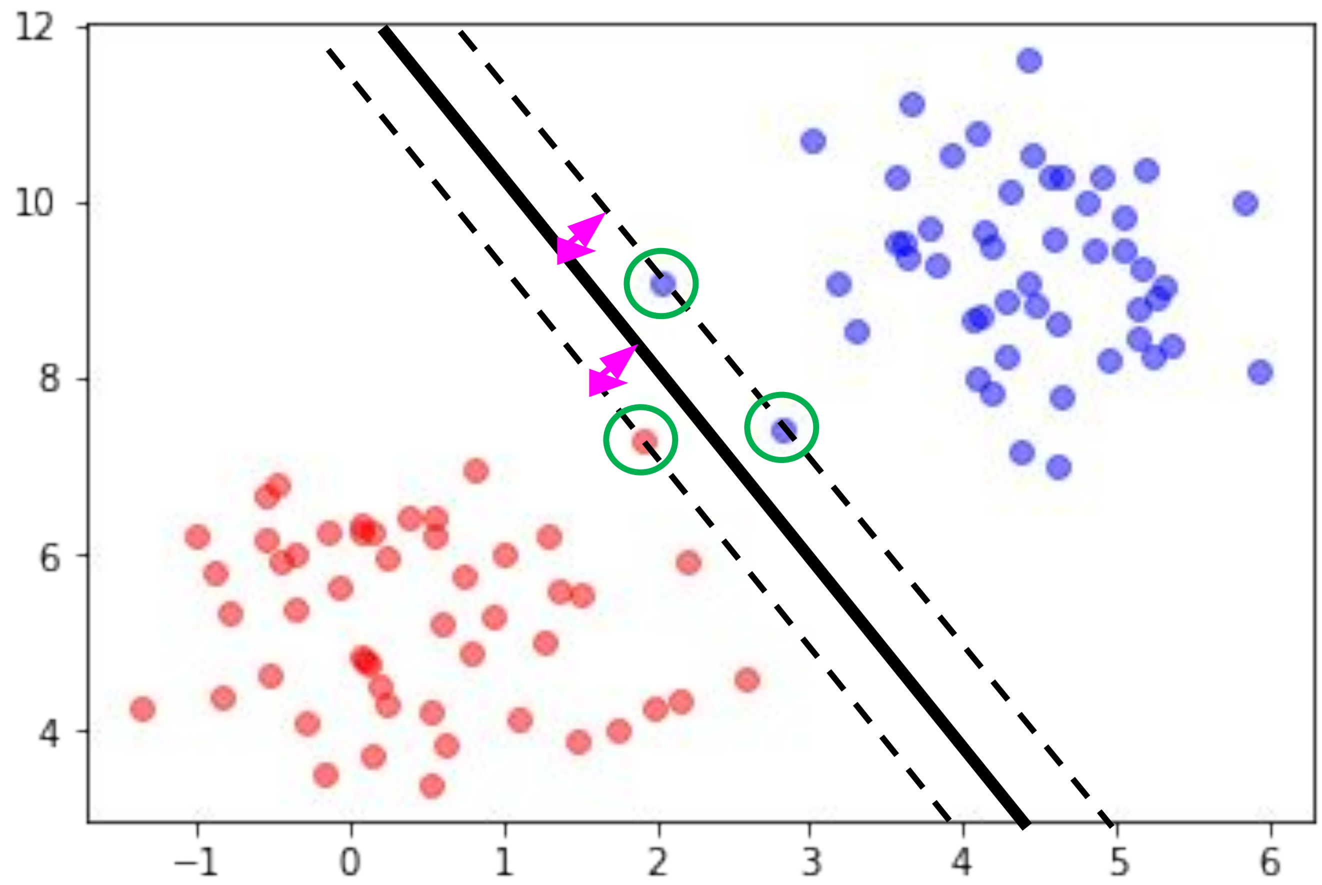
Support Vector Machine

Kernel trick

Recap

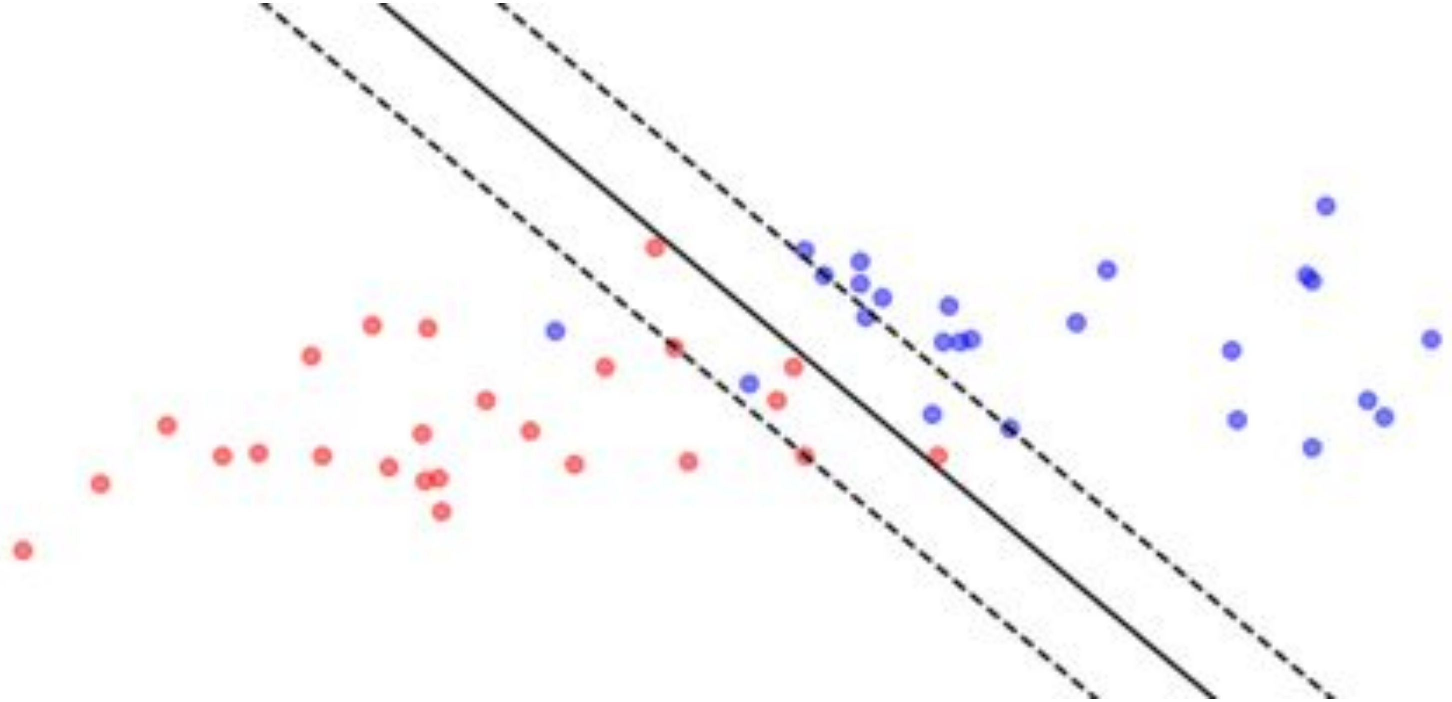


Support

Margin

$$y_i\left(\beta_0 + \beta_1 x_{i,1} + \beta_2 x_{i,2} + \beta_3 x_{i,3} + \cdots\right) \geq M$$

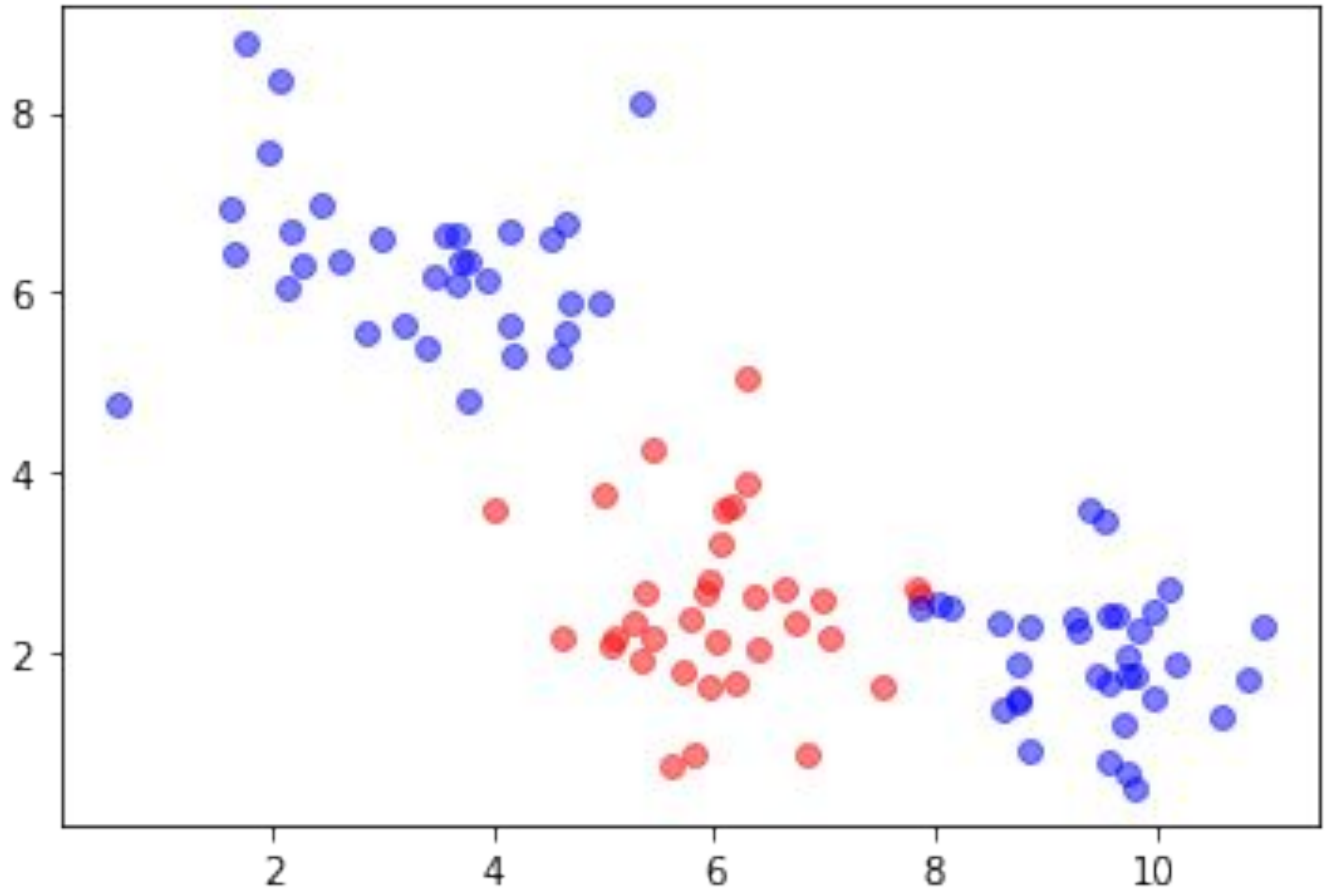
Recap



$$y_i\left(\beta_0 + \beta_1 x_{i,1} + \beta_2 x_{i,2} + \beta_3 x_{i,3} + \cdots\right) \geq M(1 - \epsilon_i) \qquad \epsilon_i \geq 0 \qquad \sum_{i=1}^{n} \epsilon_i \leq C$$

Beyond linearly separable data

How can we separate this kind of data with SVC?



SVM using Kernels

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M(1 - \epsilon_i)$$

$$\epsilon_i \geq 0$$

$$\sum_{i=1}^{n} \epsilon_i \leq C$$

Why SVM called non-parameteric when there are coefficients?

How SVM finds a solution?

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i3} + \cdots \beta_p x_{ip}) \geq M\ (1 - \epsilon_i)$$

Inner product: $$\langle x_i, x_{i'} \rangle = \sum_{j=1}^{p} x_{ij} x_{i'j}$$

$$f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i \langle x, x_i \rangle$$

How SVM finds a solution?

It computes inner product between observations

$$\langle x_i, x_{i'} \rangle = \sum_{j=1}^{p} x_{ij} x_{i'j}$$

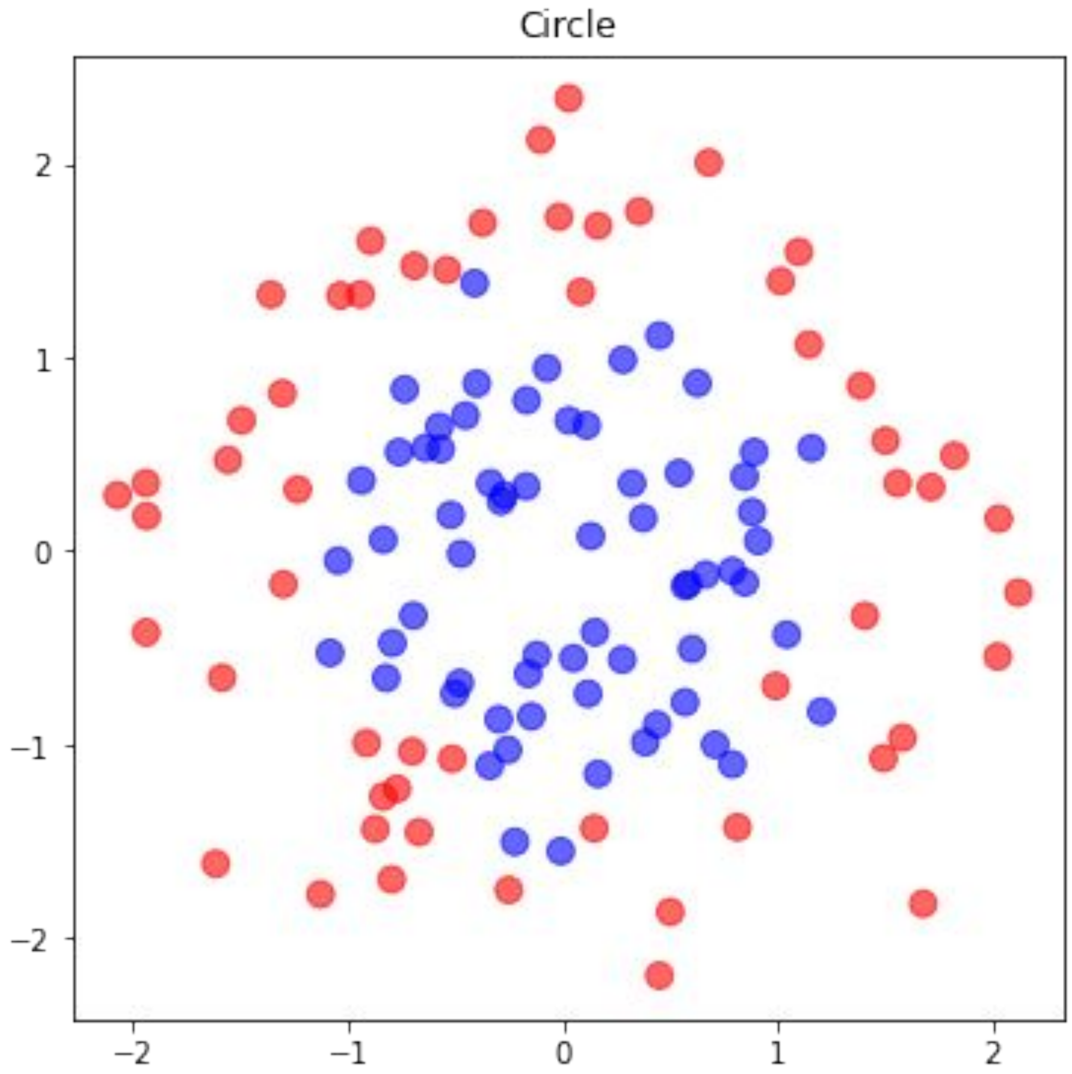
The original function $f(X) = \beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p$ can be rewritten to

$$f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i \langle x, x_i \rangle$$

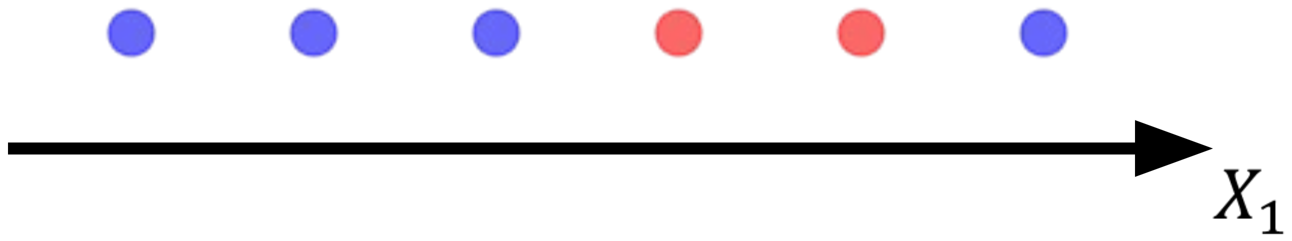
(Caution: SVM needs inputs normalized)

We need $n(n-1)/2$ inner products to calculate

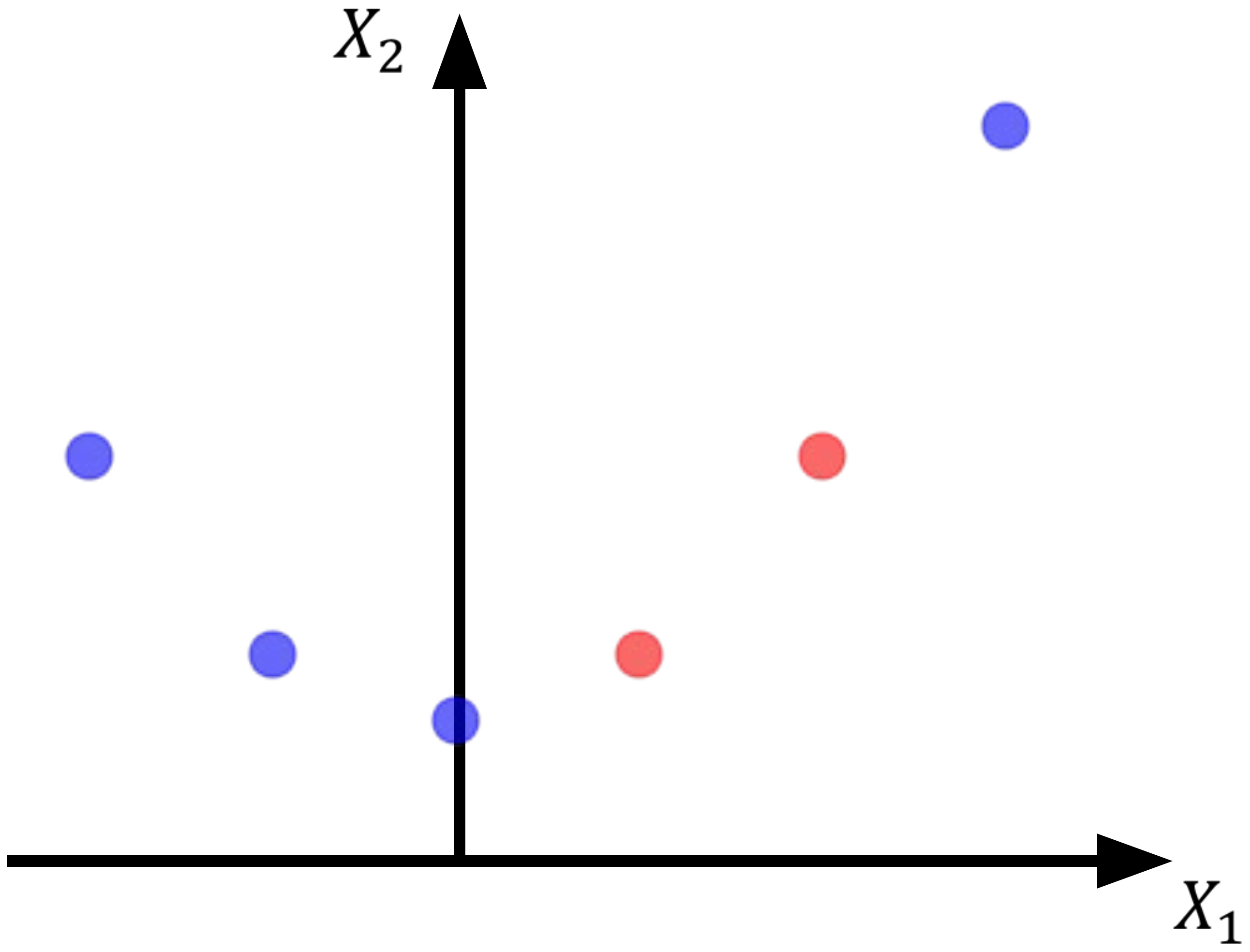
What about this data?



When data is not linearly separable

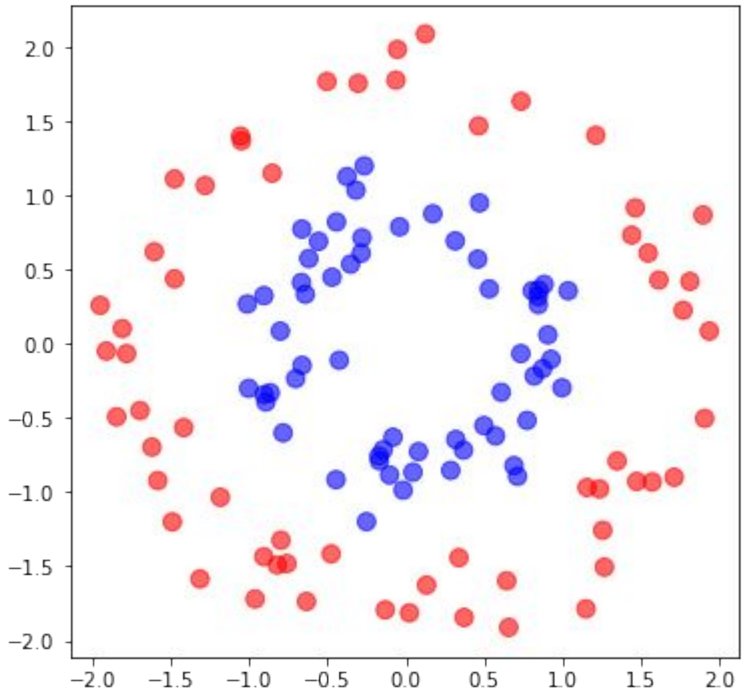


When data is not linearly separable

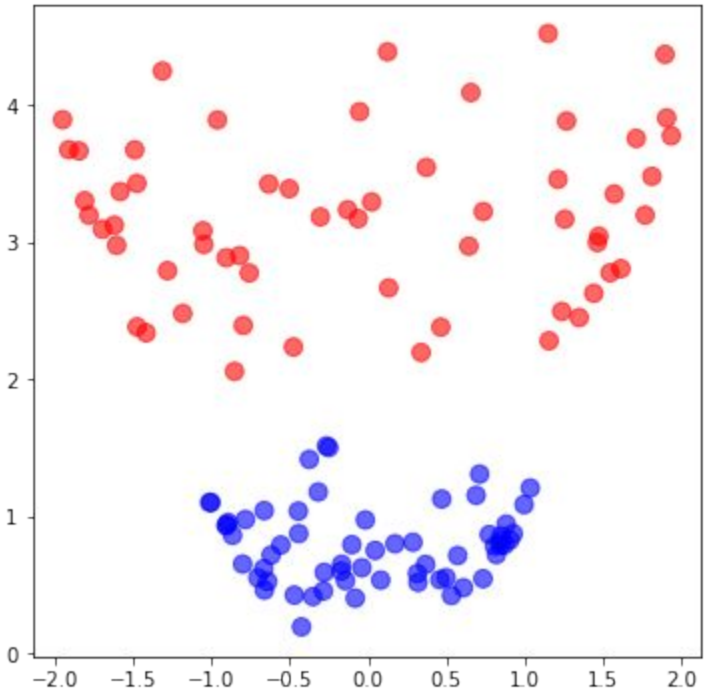


When data is not linearly separable

Not linearly separable in 2D



We can separate in 3D



Adding higher order terms

$$X_1, X_2, \ldots, X_p$$

Great! I can add higher order terms…

$$X_1, X_1^2, X_2, X_2^2, \ldots, X_p, X_p^2$$

But…

.

$$\underset{\beta_0,\beta_{11},\beta_{12}\ldots,\beta_{p1},\beta_{p2},\epsilon_1,\ldots,\epsilon_n,M}{\text{maximize}} \quad M$$

$$\text{subject to } y_i\left(\beta_0 + \sum_{j=1}^{p}\beta_{j1}x_{ij} + \sum_{j=1}^{p}\beta_{j2}x_{ij}^2\right) \geq M(1-\epsilon_i)$$

$$\sum_{i=1}^{n}\epsilon_i \leq C, \quad \epsilon_i \geq 0, \quad \sum_{j=1}^{p}\sum_{k=1}^{2}\beta_{jk}^2 = 1.$$

The Kernel trick

Let's generalize this function (the inner product)

$$\langle x_i, x_{i'} \rangle = \sum_{j=1}^{p} x_{ij} x_{i'j}$$

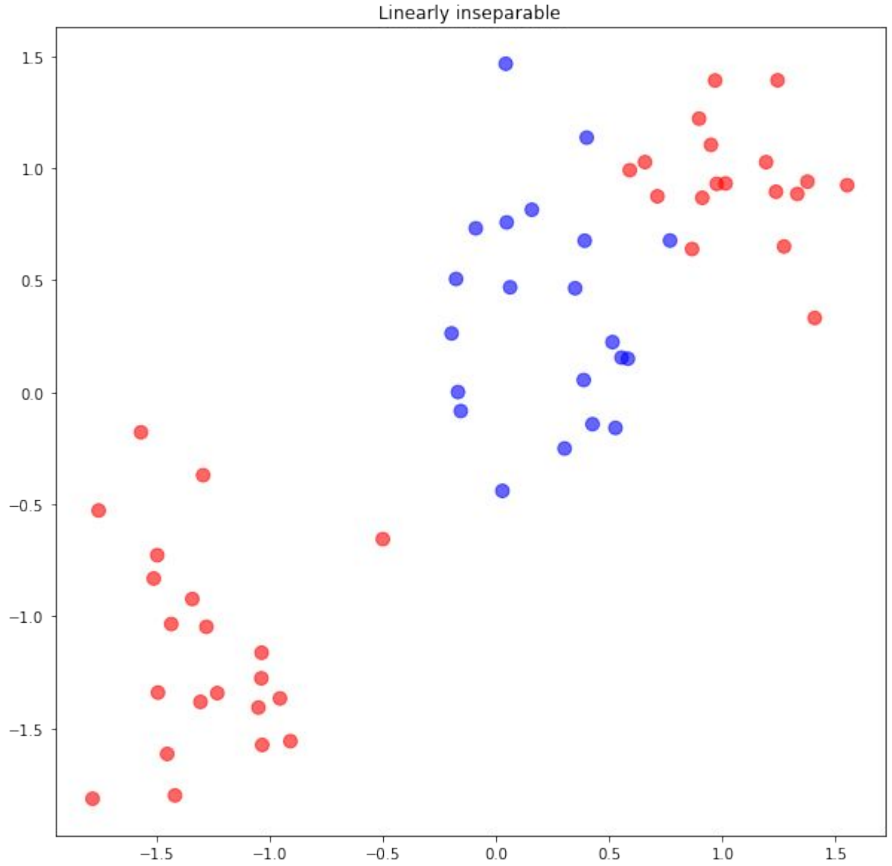
to a kernel $K(x_i, x_{i'})$

$$K(x_i, x_{i'}) = (1 + \sum_{j=1}^{p} x_{ij} x_{i'j})^d$$

Then, we get $f(x) = \beta_0 + \sum_{i \in \mathcal{S}} \alpha_i K(x, x_i)$

The Kernel trick

Non-linear kernels can take care of non-linear decision boundary

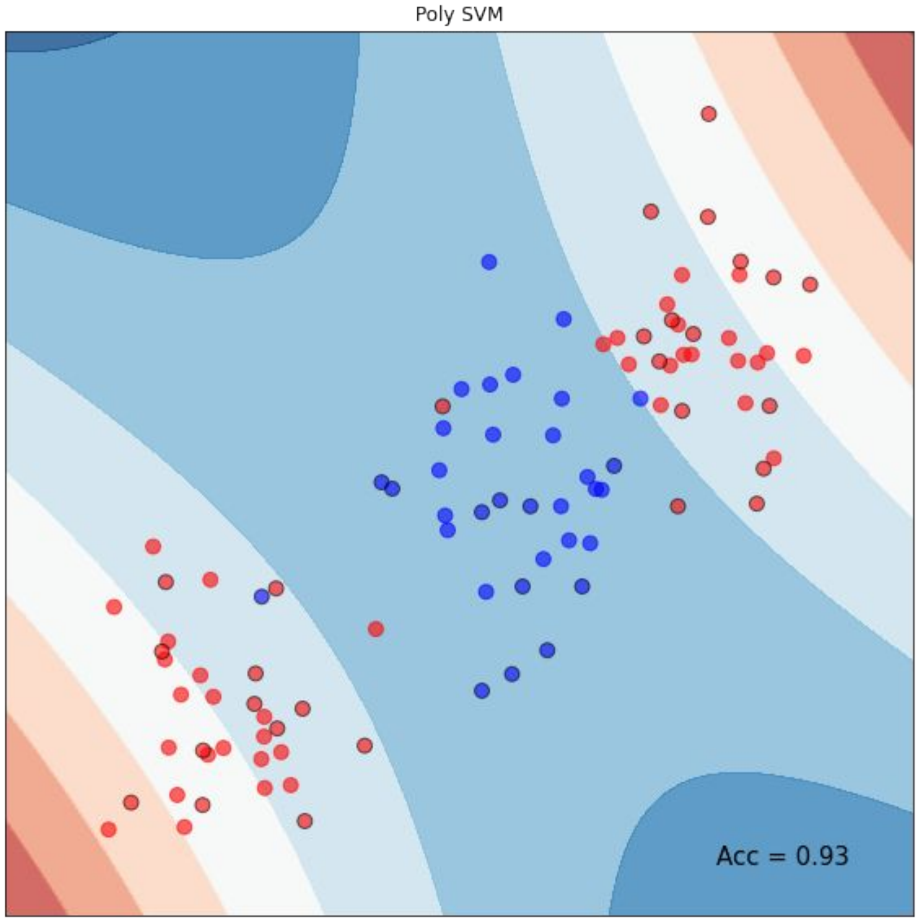


Polynomial Kernel

Non-linear kernels can take care of non-linear decision boundary

Polynomial kernel

$$K(x_i, x_{i'}) = (1 + \sum_{j=1}^{p} x_{ij} x_{i'j})^d$$



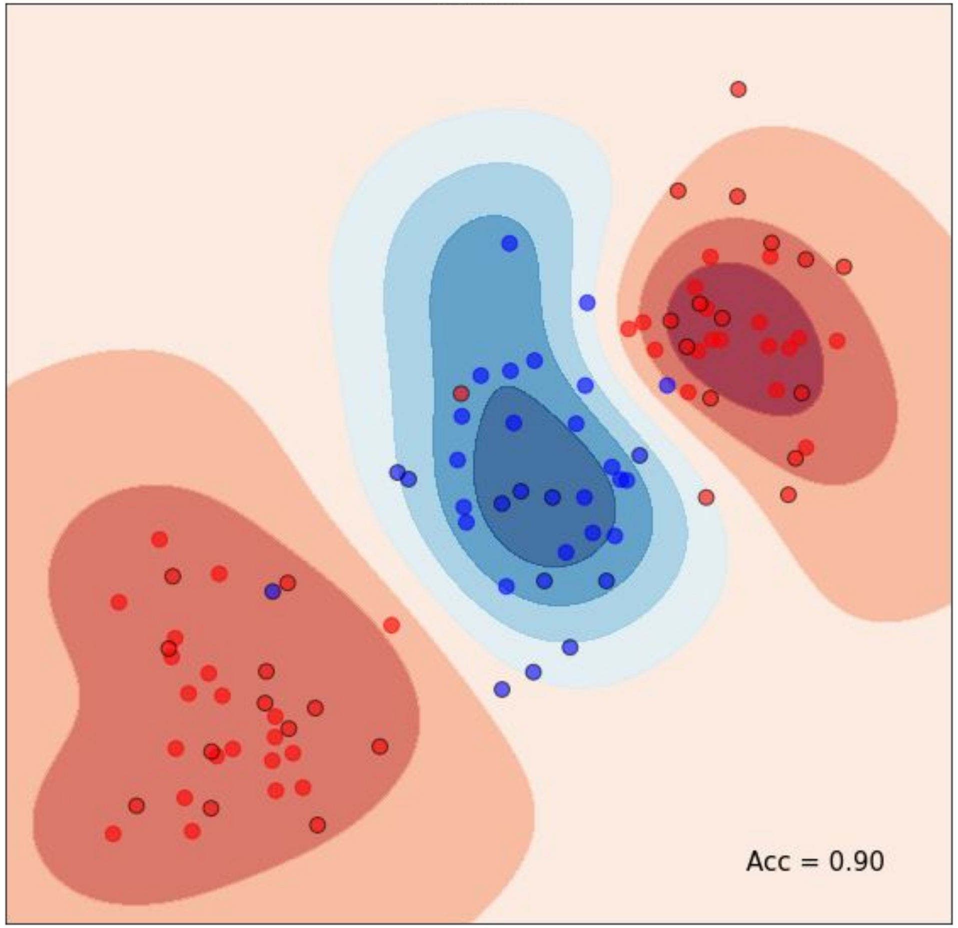
Radial Kernel

Non-linear kernels can take care of non-linear decision boundary

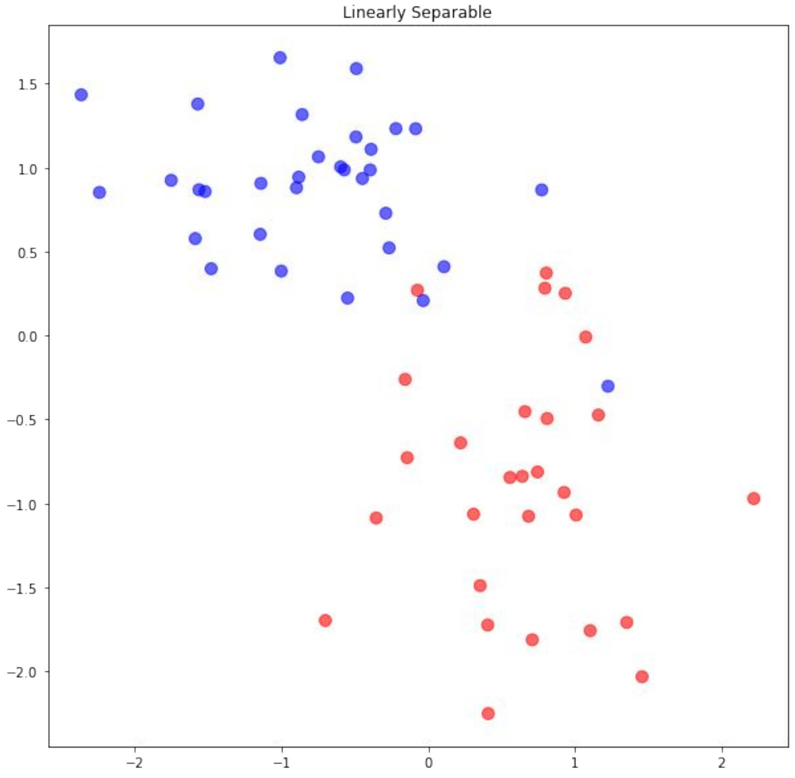
Radial Basis Function Kernel

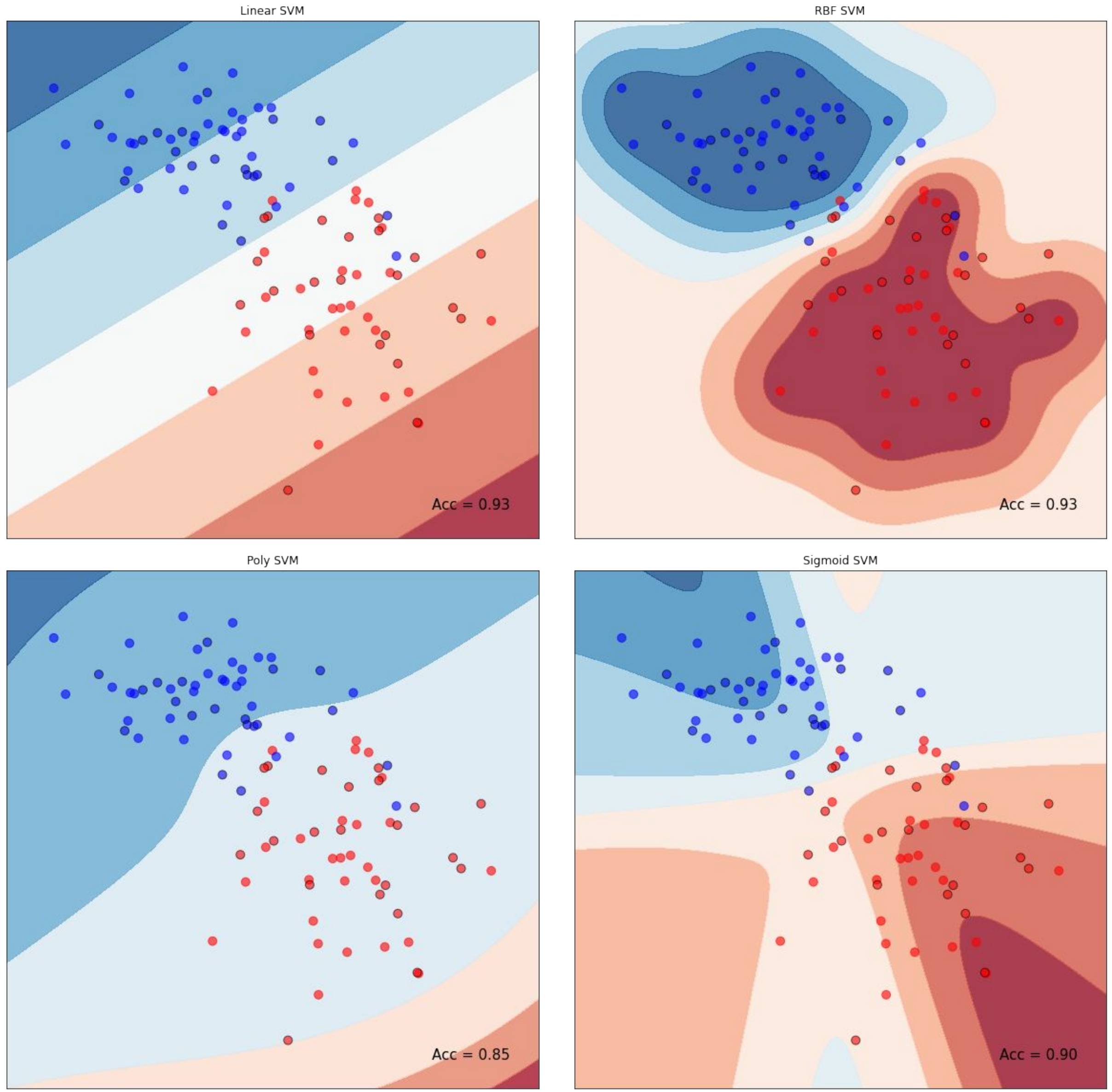
$$K(x_i, x_{i'}) = \exp(-\gamma \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2)$$

RBF SVM

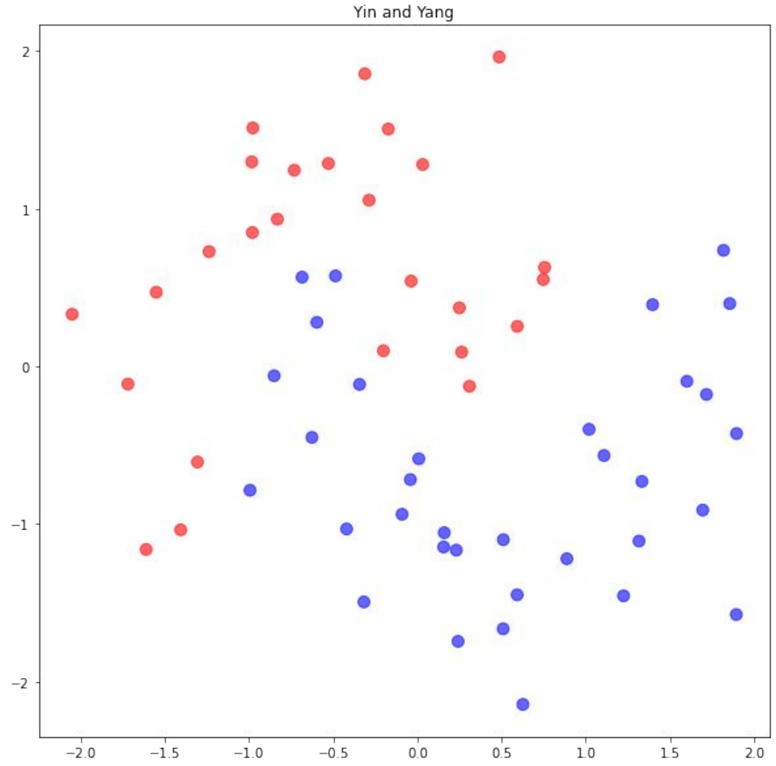


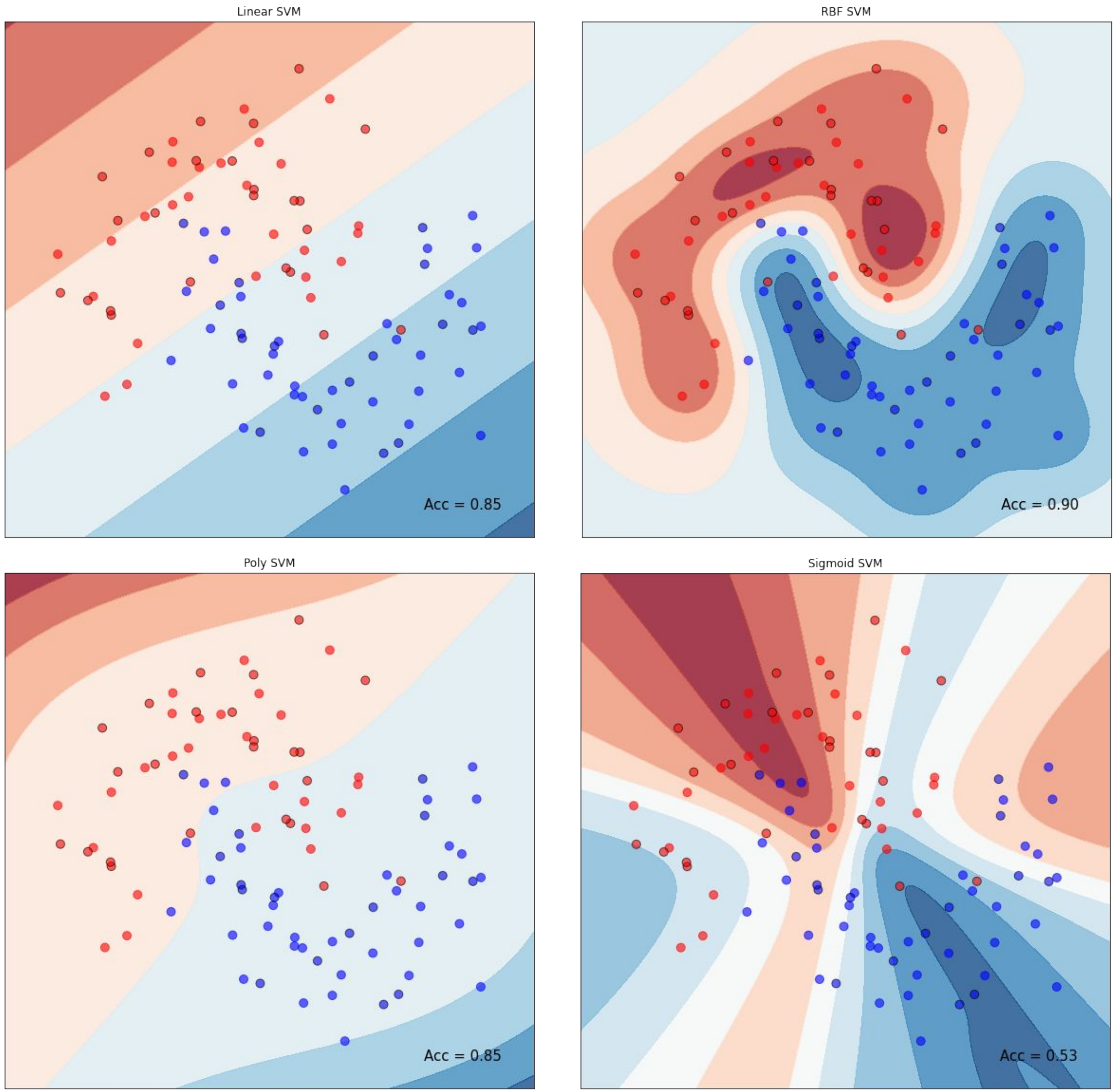
Choice of Kernels



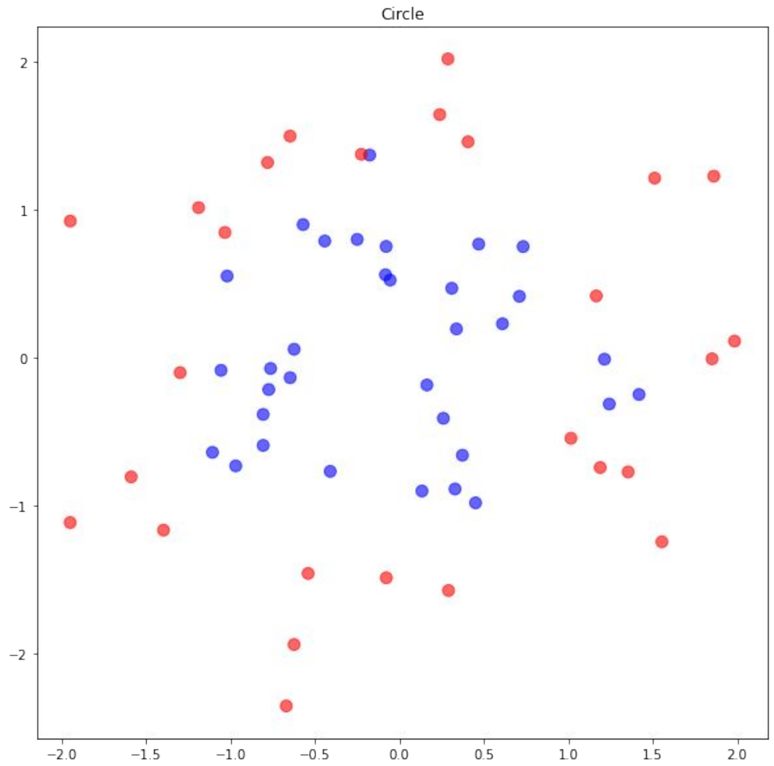


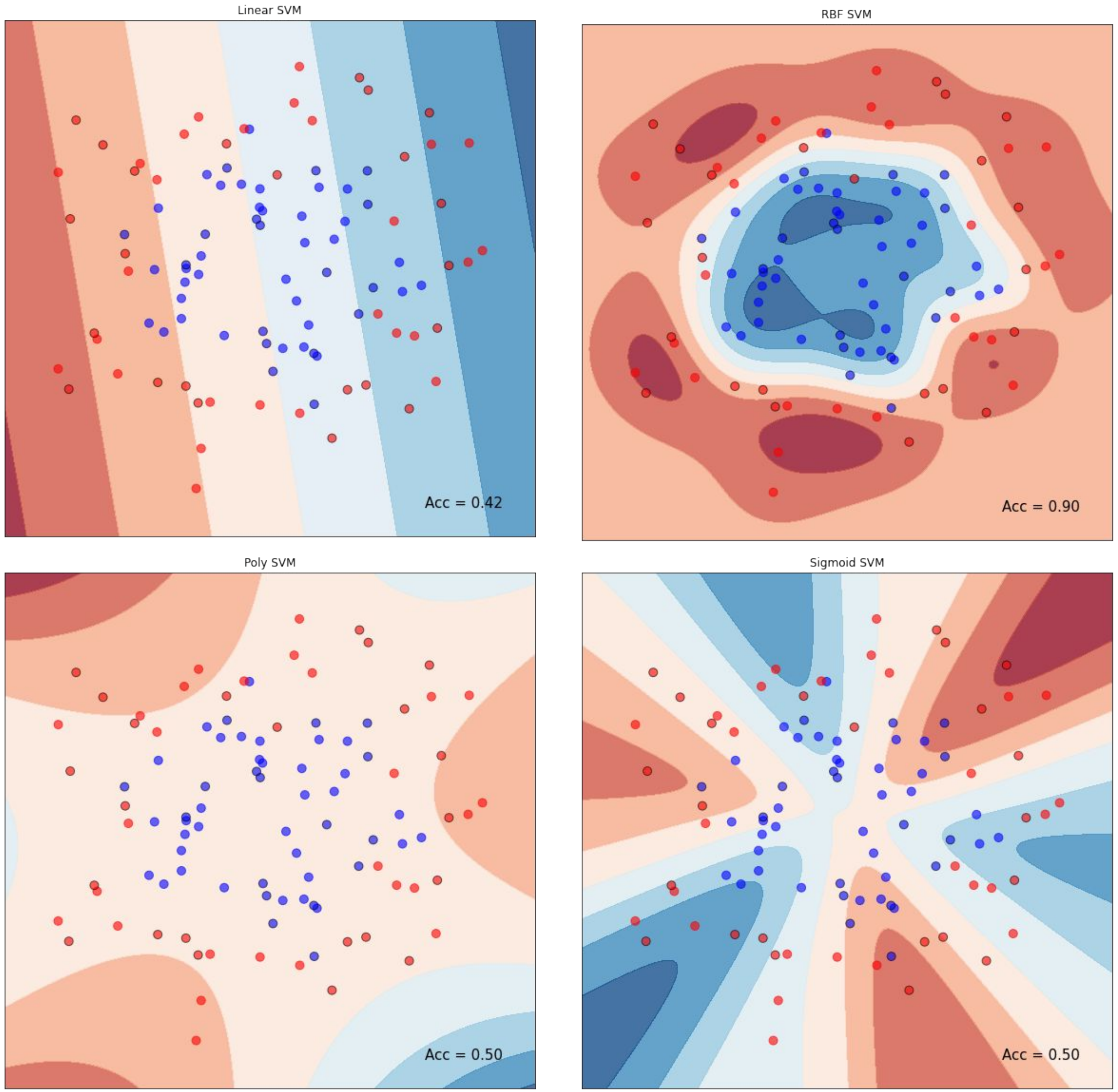
Choice of Kernels





Choice of Kernels





Hinge Loss

$$\underset{\beta_0,\beta_1,\ldots,\beta_p}{\text{minimize}} \left\{ \sum_{i=1}^{n} \max\left[0, 1 - y_i f(x_i)\right] + \lambda \sum_{j=1}^{p} \beta_j^2 \right\}$$

When to use which model?

For Binary classification

Logistic regression

SVM