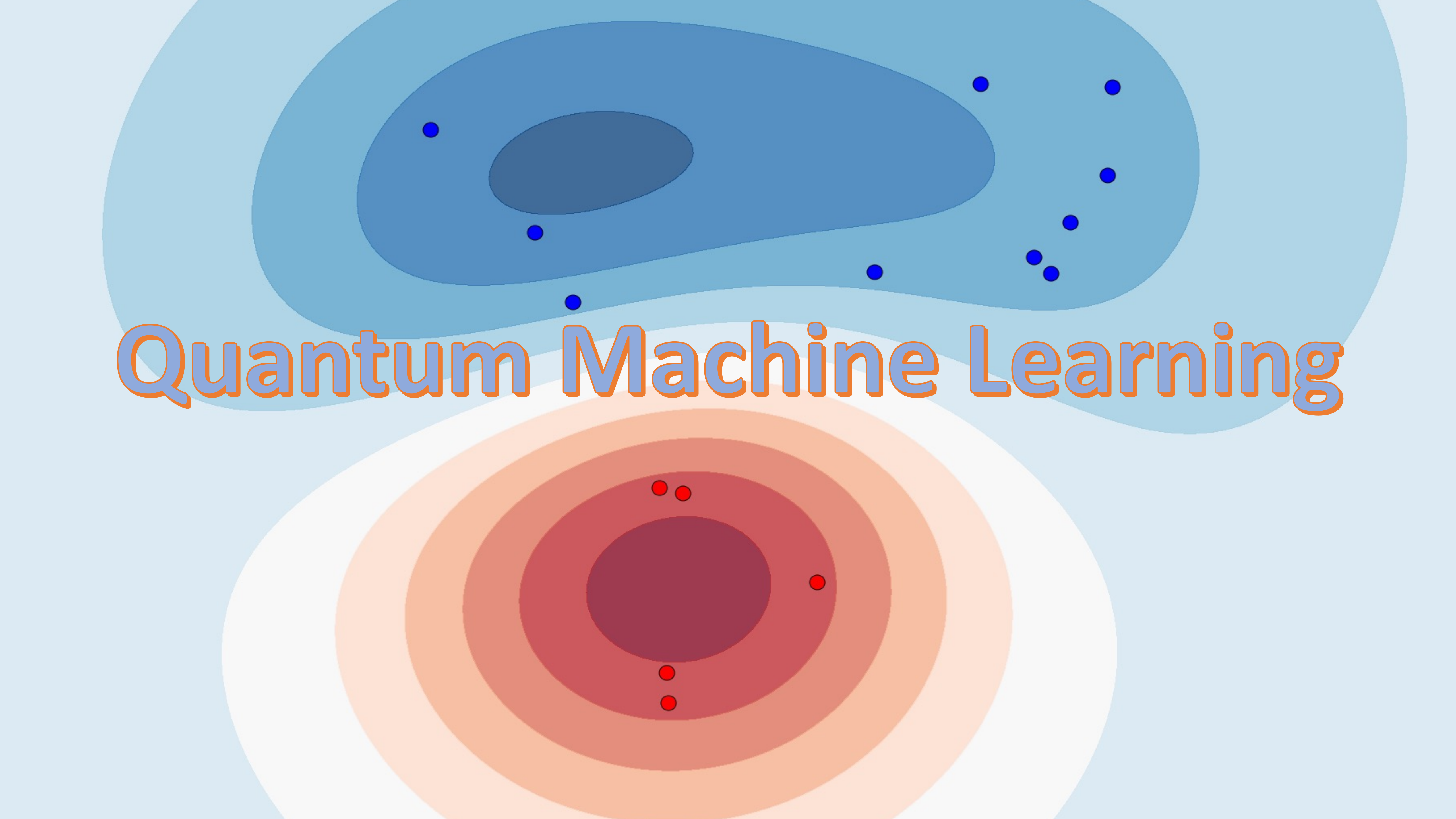
Quantum Machine Learning

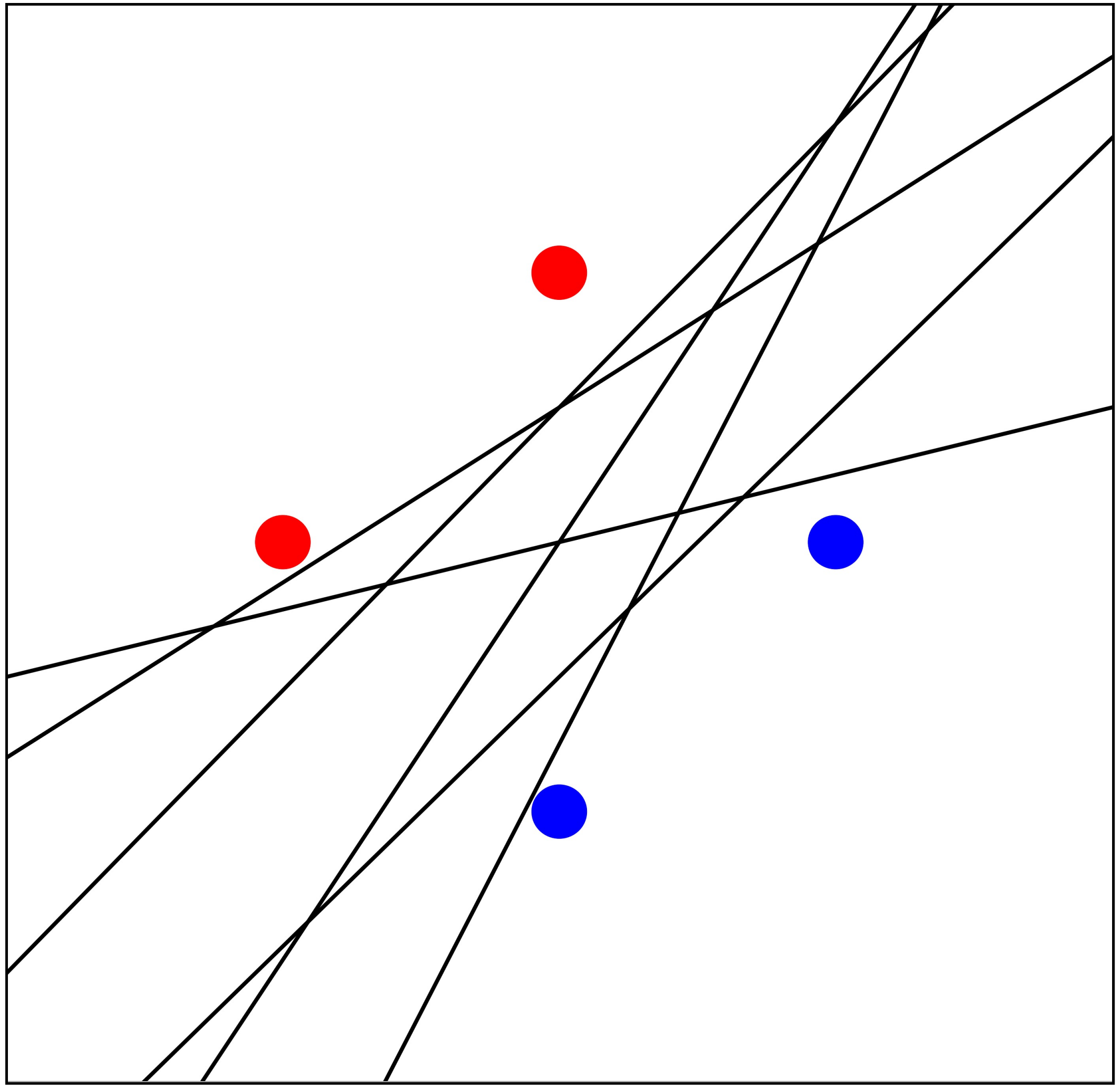


Overview

- Support Vector Machine
  - Pattern Analysis
  - XOR + Feature Space
  - Soft Margin SVM
- Kernel Method
- Quantum Machine Learning
  - Quantum Kernel Estimator
- Conclusion & Outlook

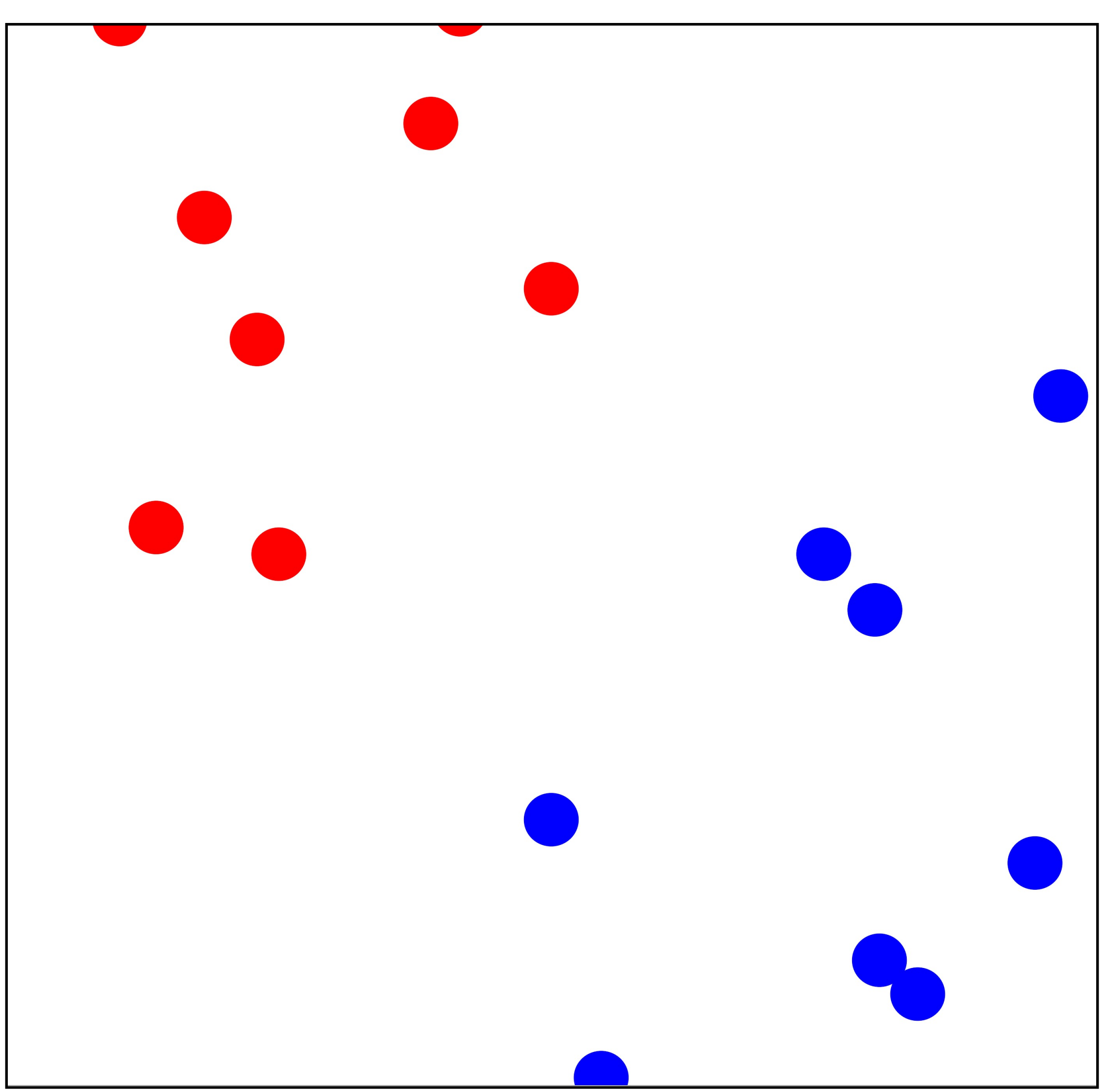
Pattern Analysis

- Binary Classification with $C = \{-1,1\}$ and input data $(\boldsymbol{x}_i, y_i)_{i=1,\ldots,t}$ with $\boldsymbol{x}_i \in T \subset \mathbb{R}^d$ and $y_i \in C$
- If data linearly separable we can separate by hyperplane
- Hyperplane: $(\boldsymbol{w}, b)$
- Decision Function: $\widetilde{m}(\boldsymbol{x}) = \operatorname{sign}(\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b)$
- Hyperplane is not unique



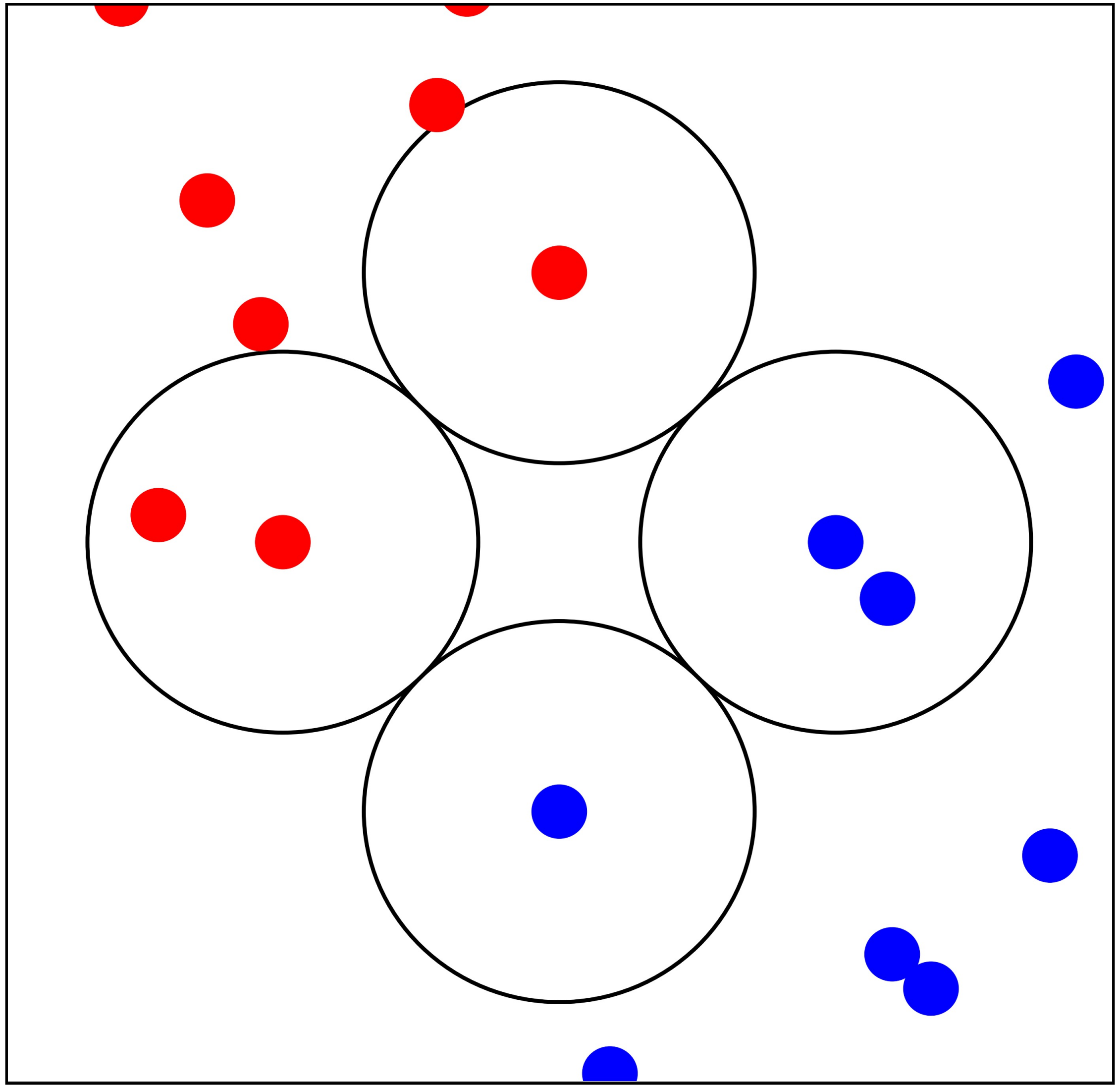
SVM

- SVM finds “optimal” hyperplane which is unique



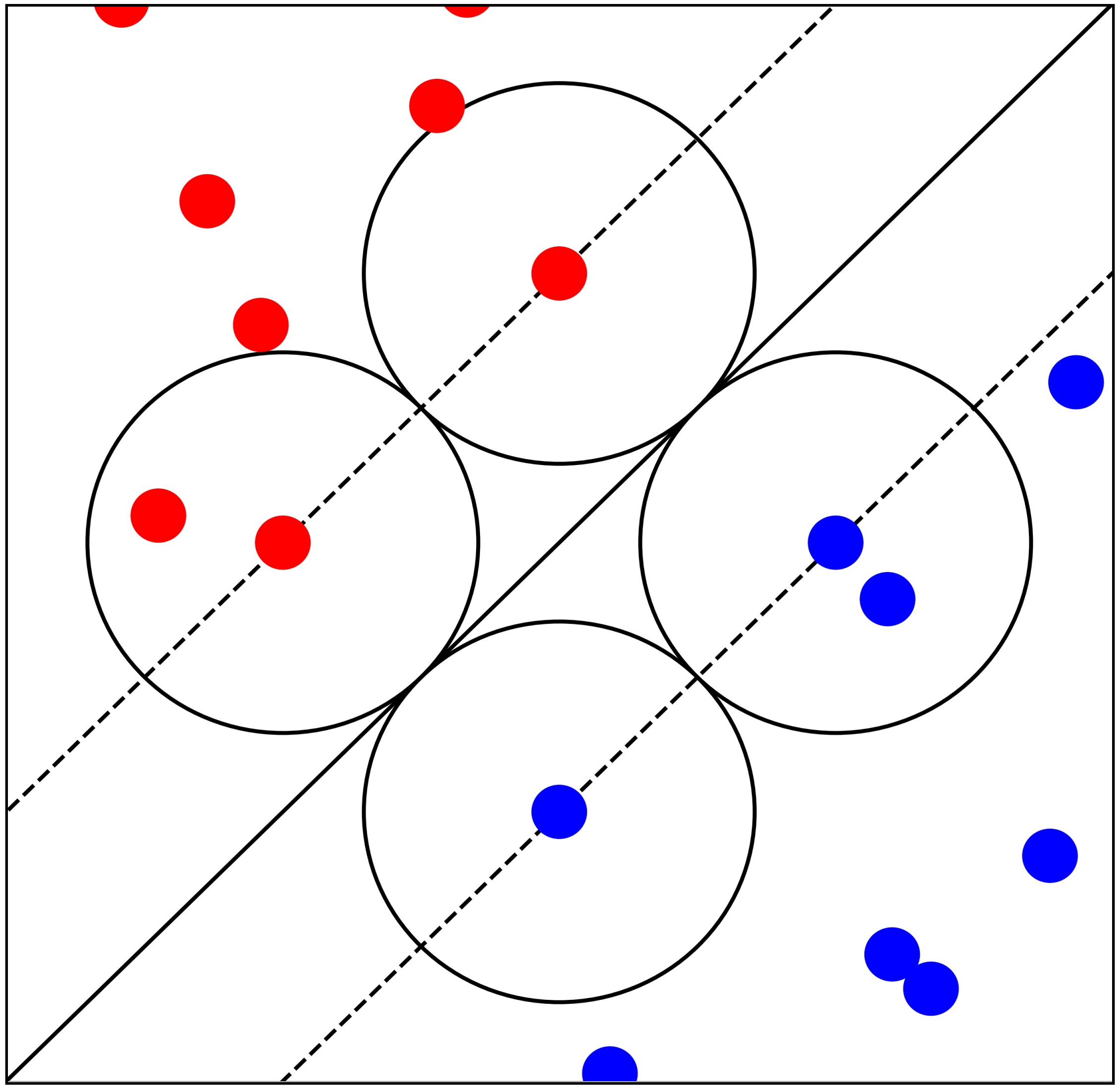
SVM

- SVM finds “optimal” hyperplane which is unique



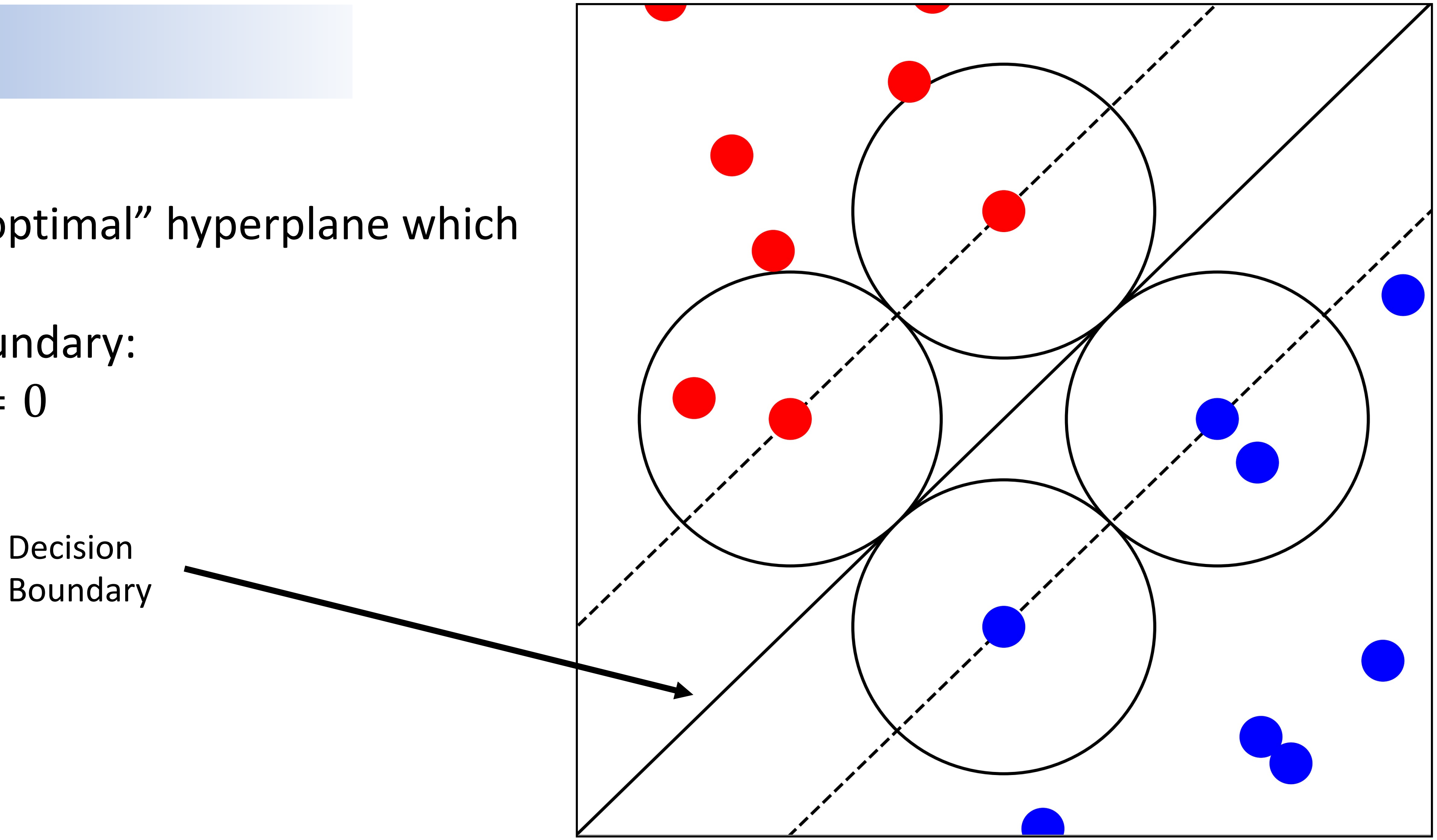
SVM

- SVM finds “optimal” hyperplane which is unique



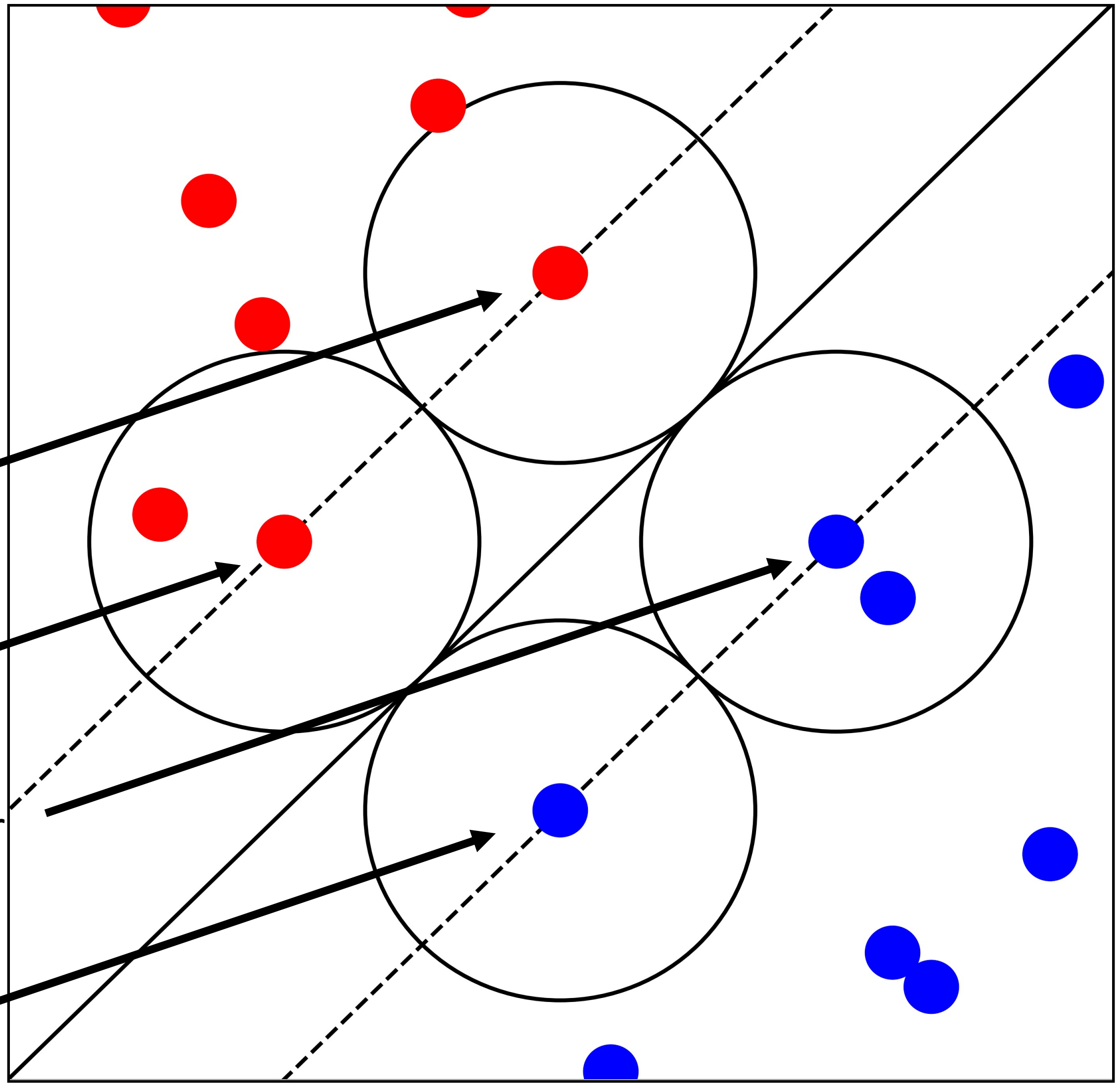
SVM

- SVM finds “optimal” hyperplane which is unique
- Decision Boundary:
  $\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b = 0$



SVM

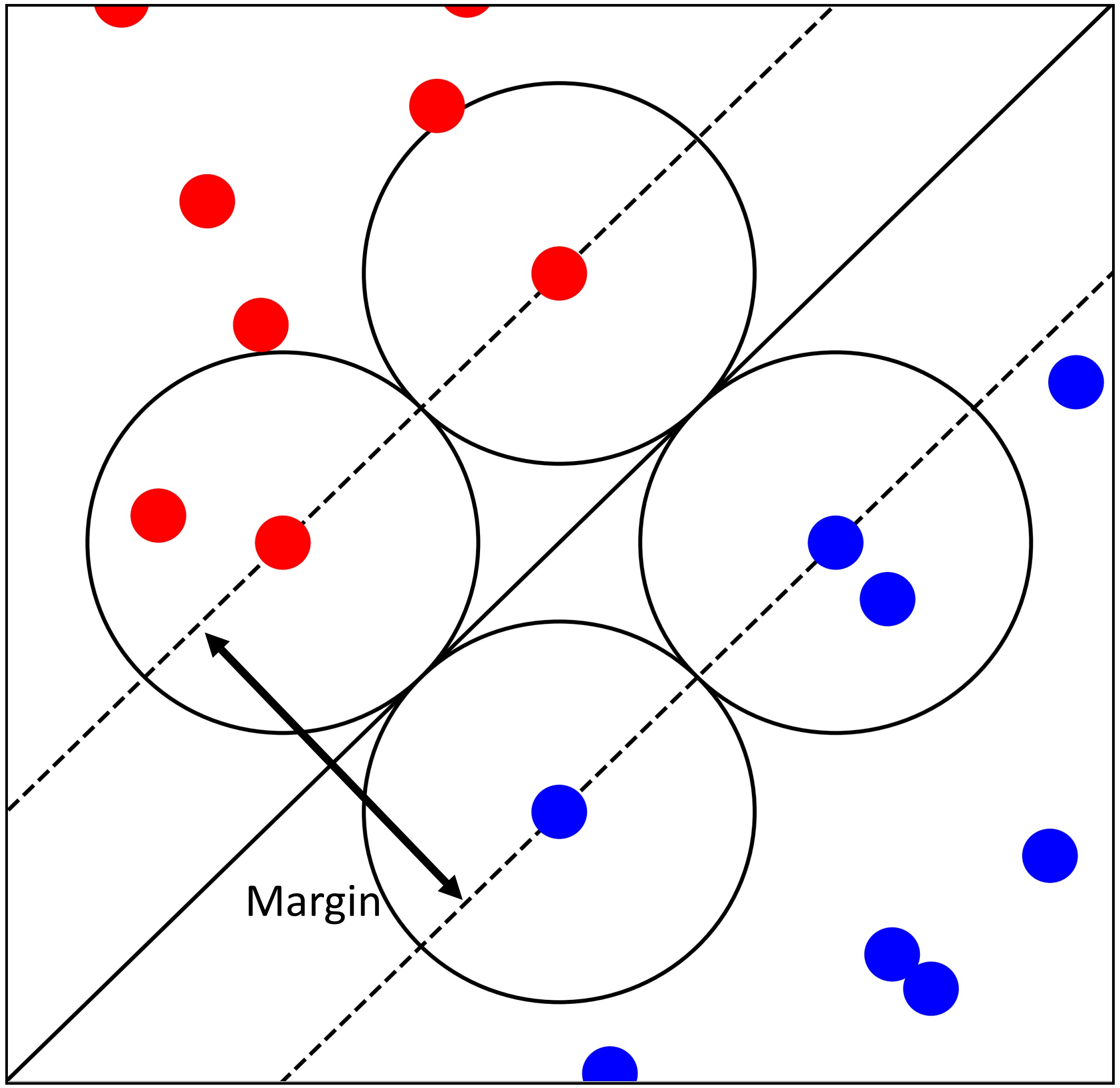
- SVM finds “optimal” hyperplane which is unique
- Decision Boundary: $\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b = 0$



SVM

- SVM finds "optimal" hyperplane which is unique
- Decision Boundary: $\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b = 0$
- Margin: $\gamma = \frac{2}{||w||}$

→ Maximize the margin

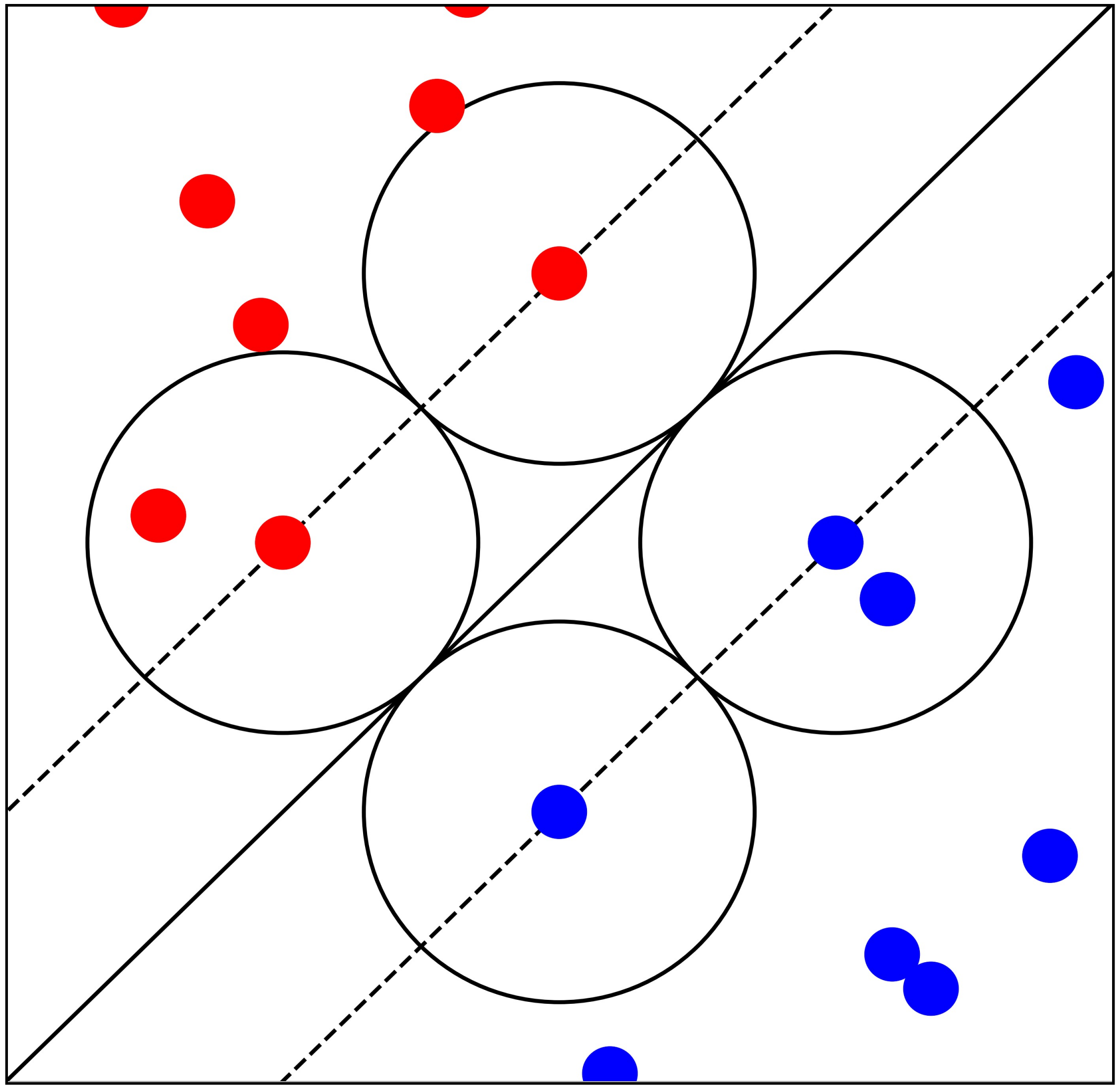


SVM

- Opimization Problem:
- minimize $L_p = \frac{1}{2}||\boldsymbol{w}||^2$
  subject to: $y_i(\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle) \geq 1,$
  $\forall i = 1, \ldots, t$
- In terms of the Lagrangian:

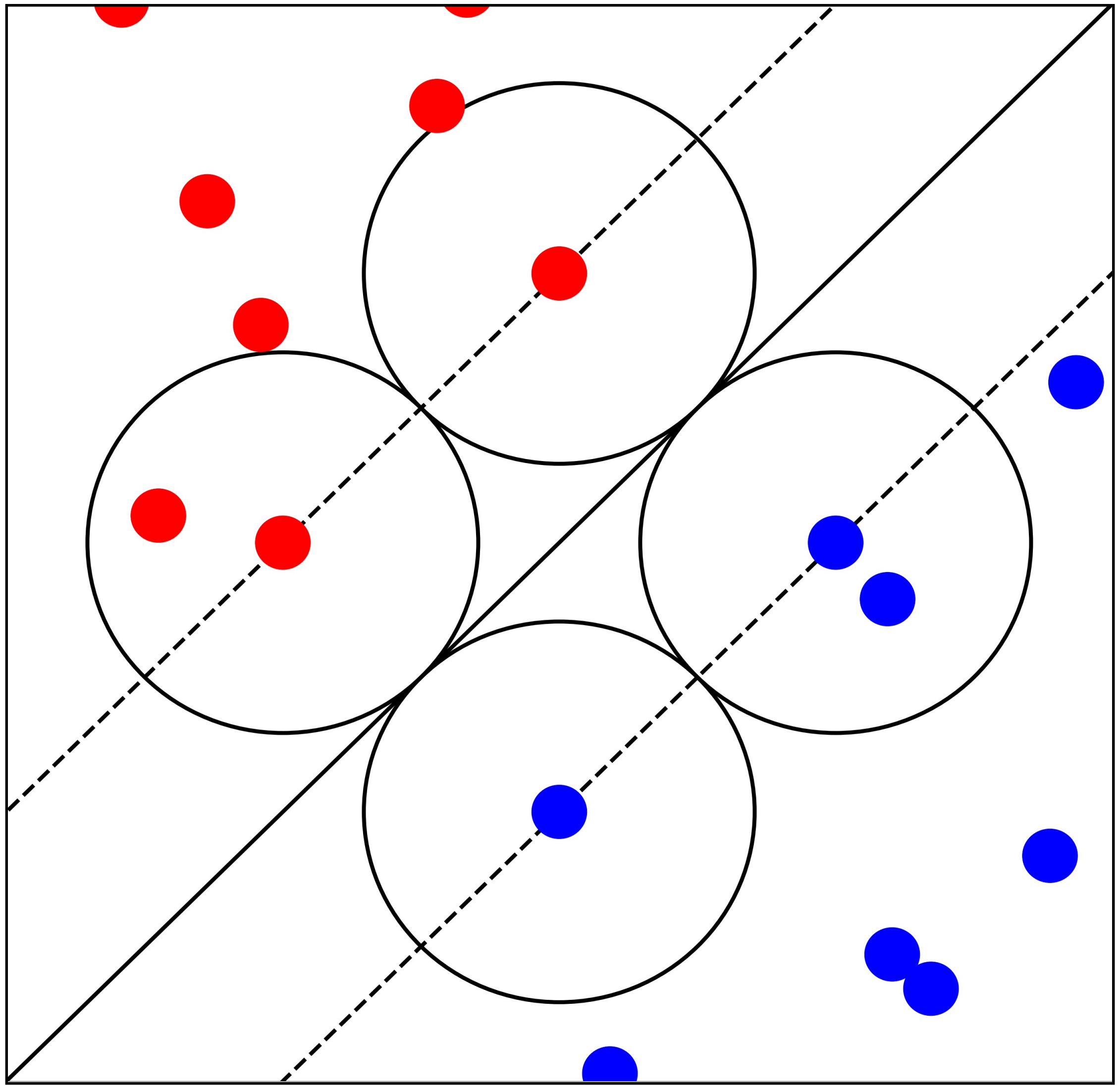
  $$\mathrm{L_D} = \frac{1}{2}||\boldsymbol{w}||^2 - \sum_{i=1}^{t} \alpha_i y_i(\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle) + \sum_{i=1}^{t} \alpha_i$$

  with $\alpha_i \geq 0$



SVM

- Solution of form $\boldsymbol{\alpha}^* = (\alpha_1^*, \dots, \alpha_t^*)$, only support vectors non-zero α_i^*
- $N_S = \{i \in [t] \mid \alpha_i^* > 0\}$
- Decision Function:
  $\widetilde{m}(\boldsymbol{s}) = \operatorname{sign}(\langle \boldsymbol{w}, \boldsymbol{s} \rangle + b)$
  with $\boldsymbol{w} = \sum_{i \in N_S} \alpha_i^* y_i \boldsymbol{x}_i$



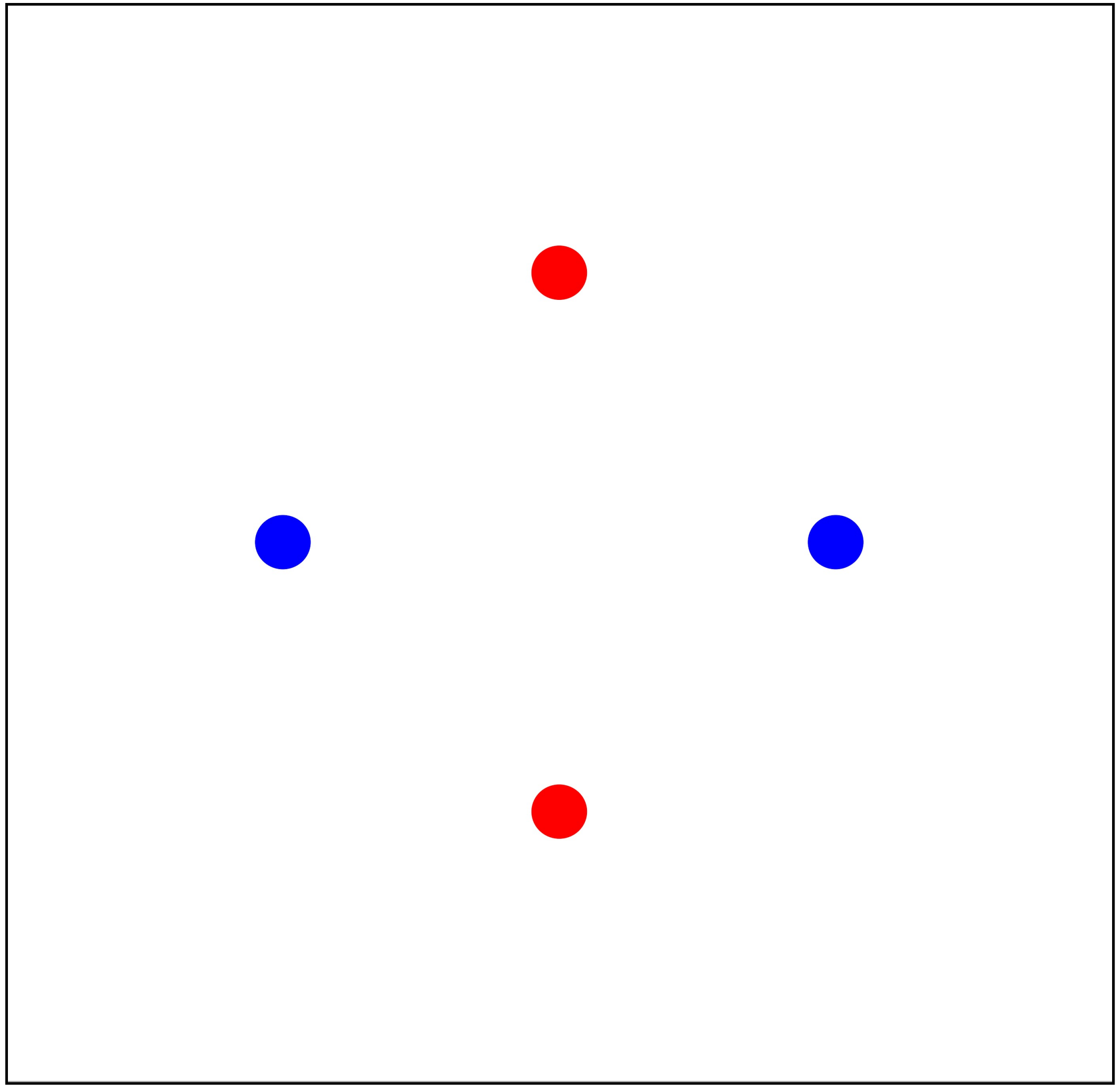
XOR-Problem

- What if data not linearly separable?
- Radon's Theorem:

  Any set of $d + 2$ *points in* $\mathbb{R}^d$ *can* ***always*** *partitioned into two subsets* S_1, S_2 *with*

  $$\mathrm{conv}(\mathrm{S}_1) \cap \mathrm{conv}(\mathrm{S}_2) \neq \emptyset$$

→ For any d-dimensional space we can find XOR dataset with at least $d + 2$ data points



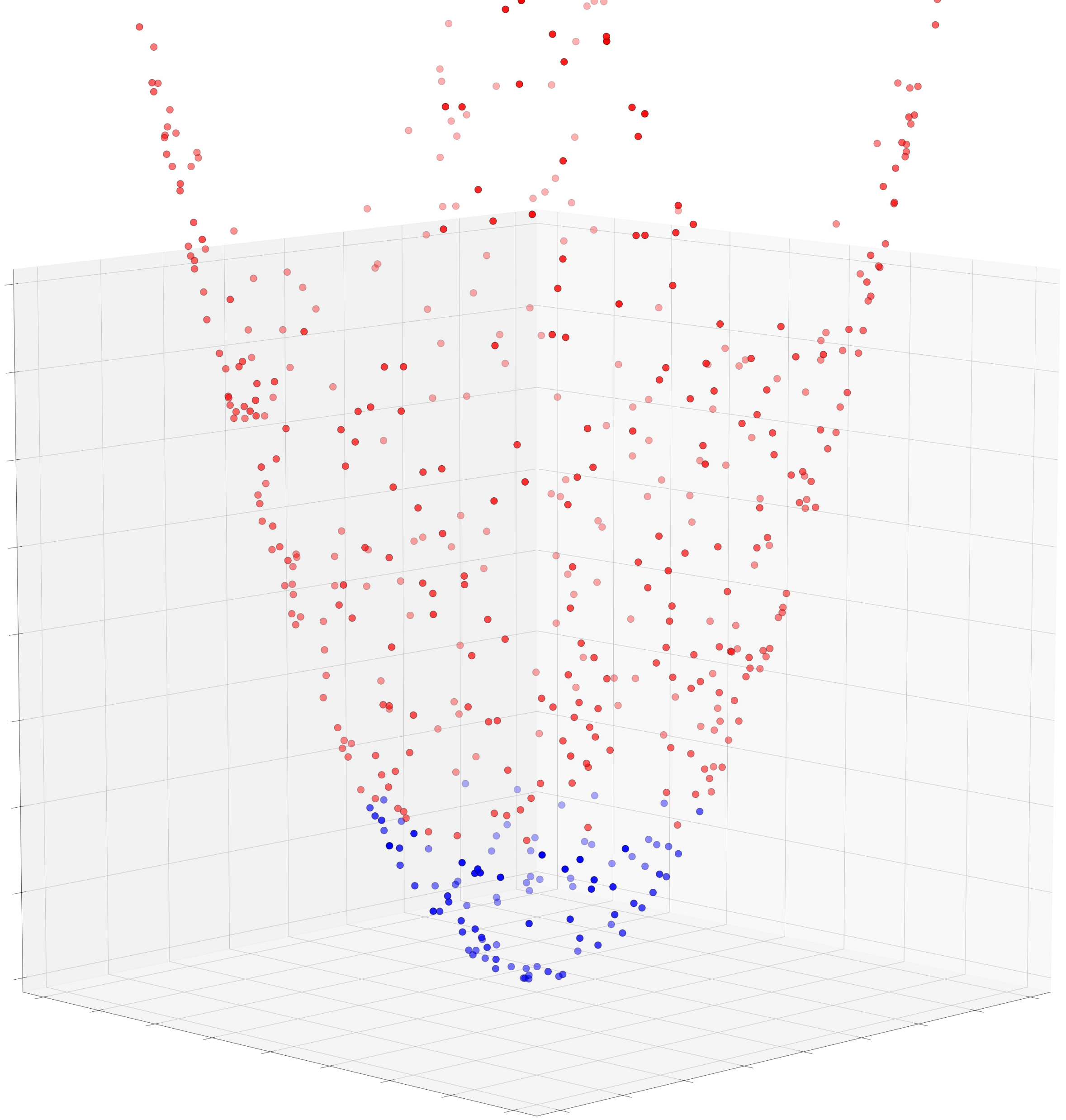
Feature Space

- Solution comes with Cover's Theorem
- Cover's Theorem:

  The number of linearly separable dichotomies of n points in general position in $\mathbb{R}^d$ is

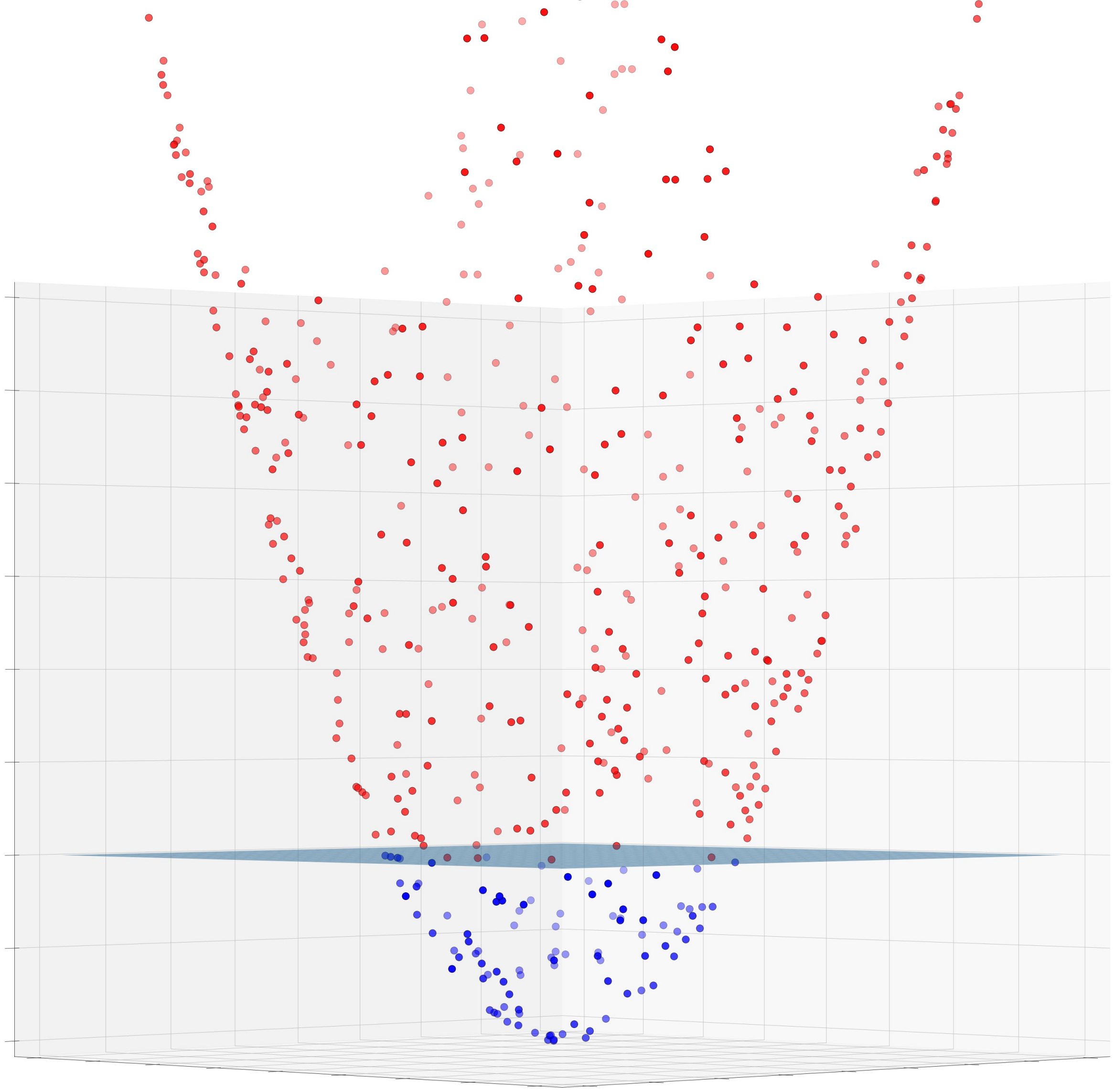
$$C(n,d) = 2\sum_{k=0}^{d} \binom{n-1}{d}$$

- If data not linearly separable than probably linearly separable in higher dimensional space



Feature Space

- Solution comes with Cover's Theorem
- Cover's Theorem:
  The number of linearly separable dichotomies of n points in general position in $\mathbb{R}^d$ is
  $$C(n,d) = 2\sum_{k=0}^{d}\binom{n-1}{d}$$
- If data not linearly separable than probably linearly separable in higher dimensional space



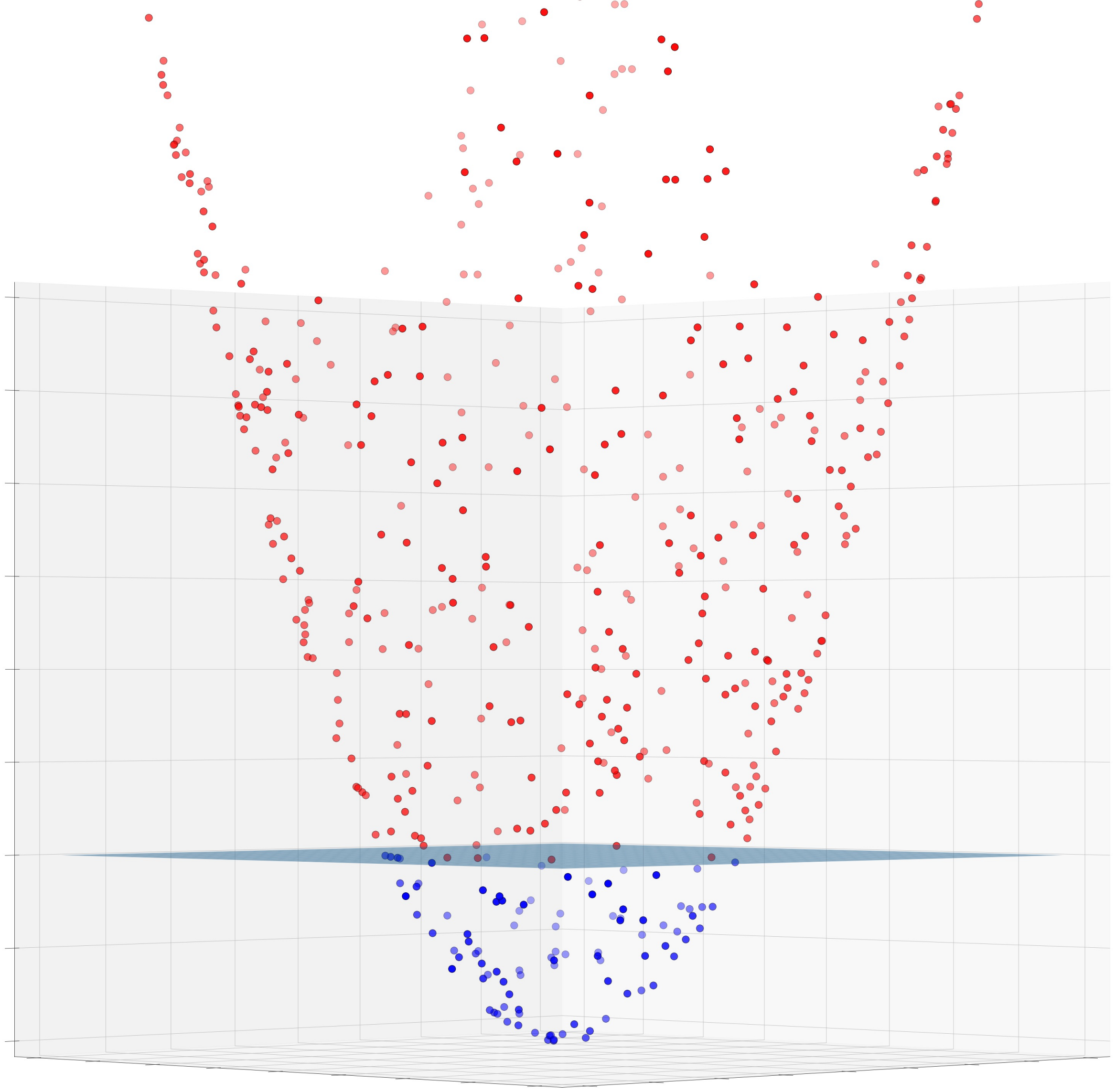
Feature Space

- Feature Map:

$$\phi: \mathbb{R}^d \to \mathcal{H}$$

- In our example:

$$(x, y) \mapsto (x, y, x^2 + y^2)$$



Feature Space

- Optimization Problem:

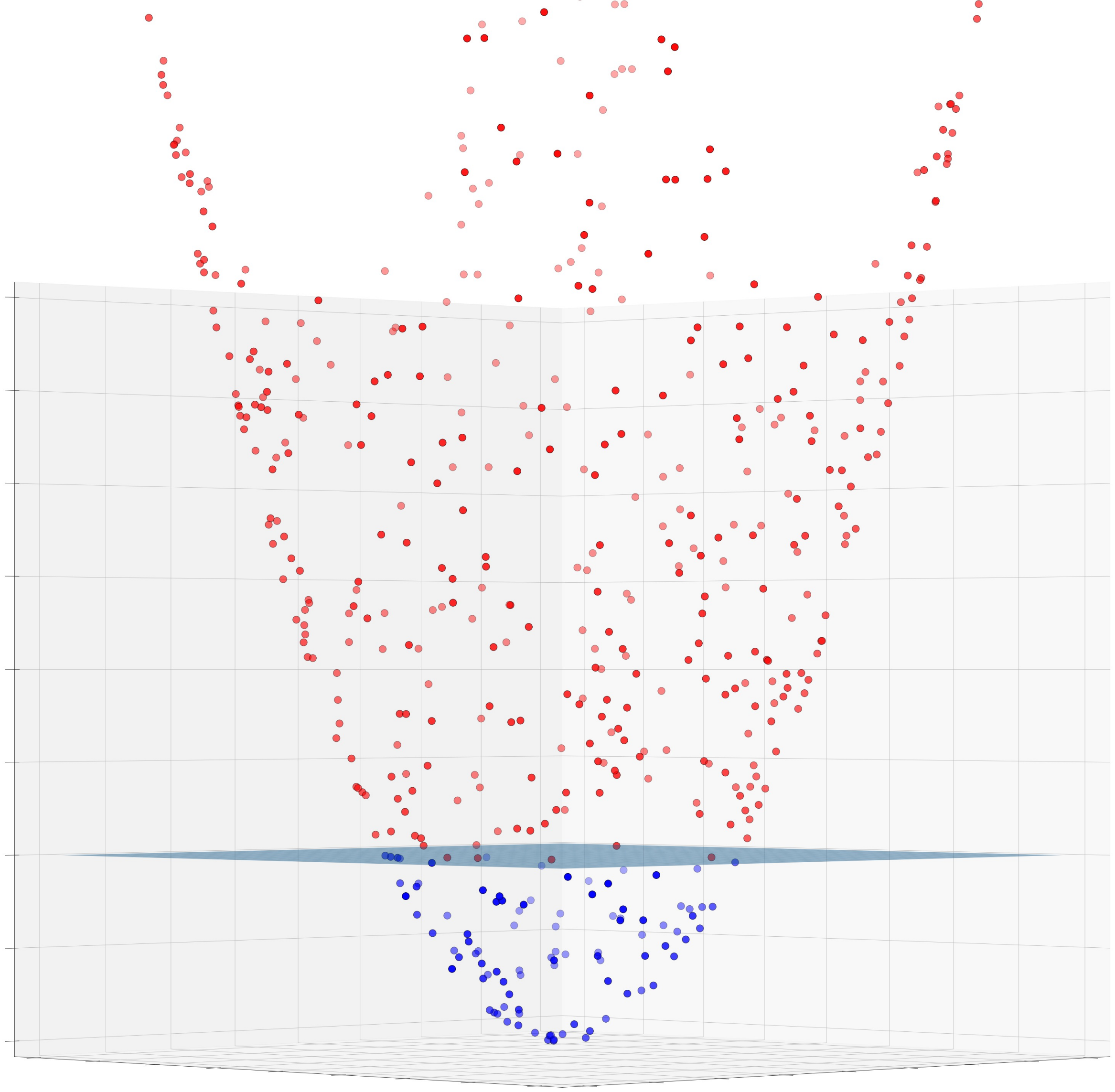
$$L_D = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle$$

subject to: $\sum_i \alpha_i y_i = 0,\ 0\ \leq \alpha_i$

- Decision Function:

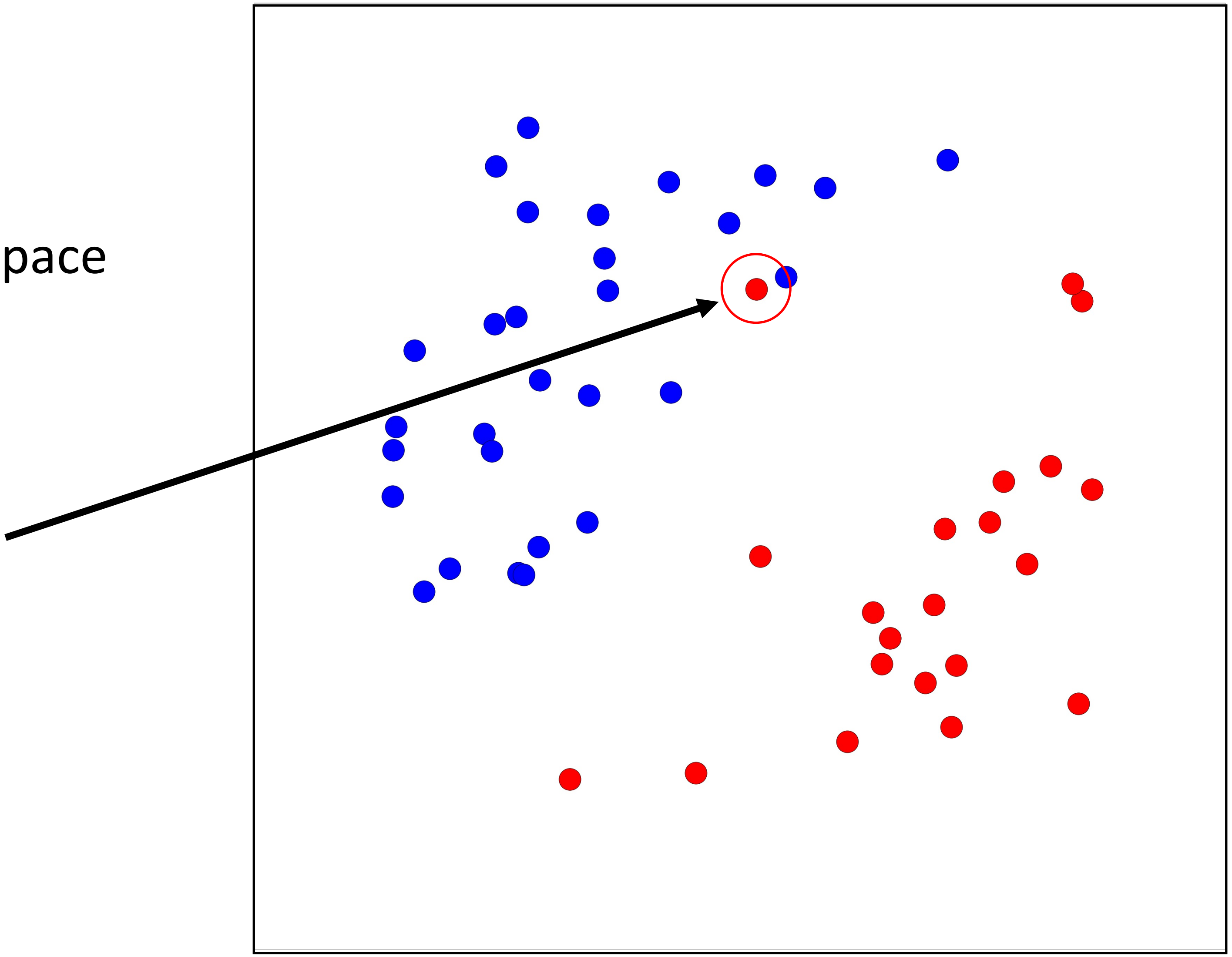
$$\widetilde{m}(\boldsymbol{s}) = \ \mathrm{sign}(\langle \boldsymbol{w}, \boldsymbol{s} \rangle + b)$$

with $\boldsymbol{w} = \sum_{i \in N_S} \alpha_i^* y_i \phi(\boldsymbol{x}_i)$



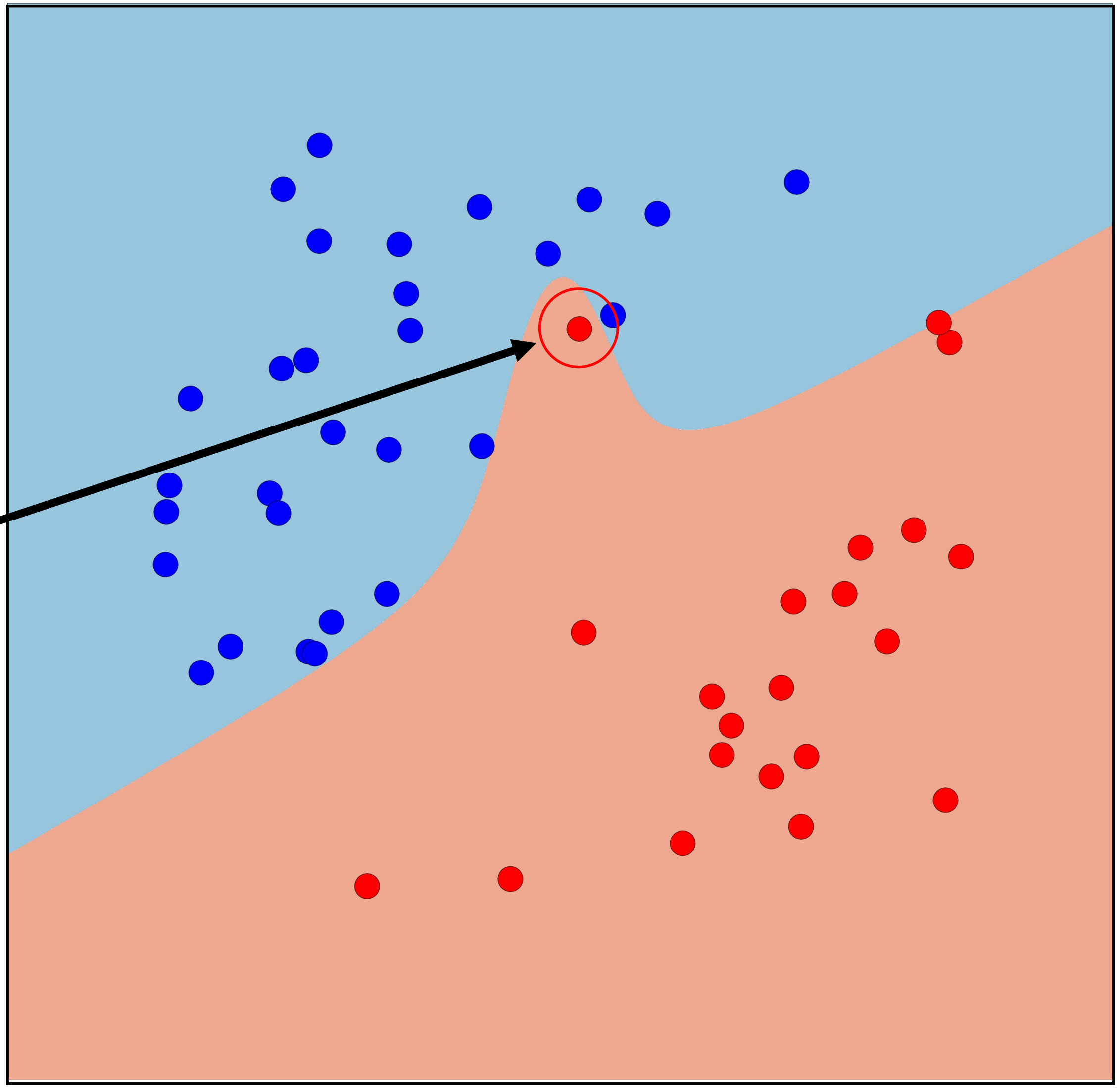
Soft Margin SVM

- What if dimension of feature space gets unnecessary high?



Soft Margin SVM

- What if dimension of feature space gets unnecessary high?



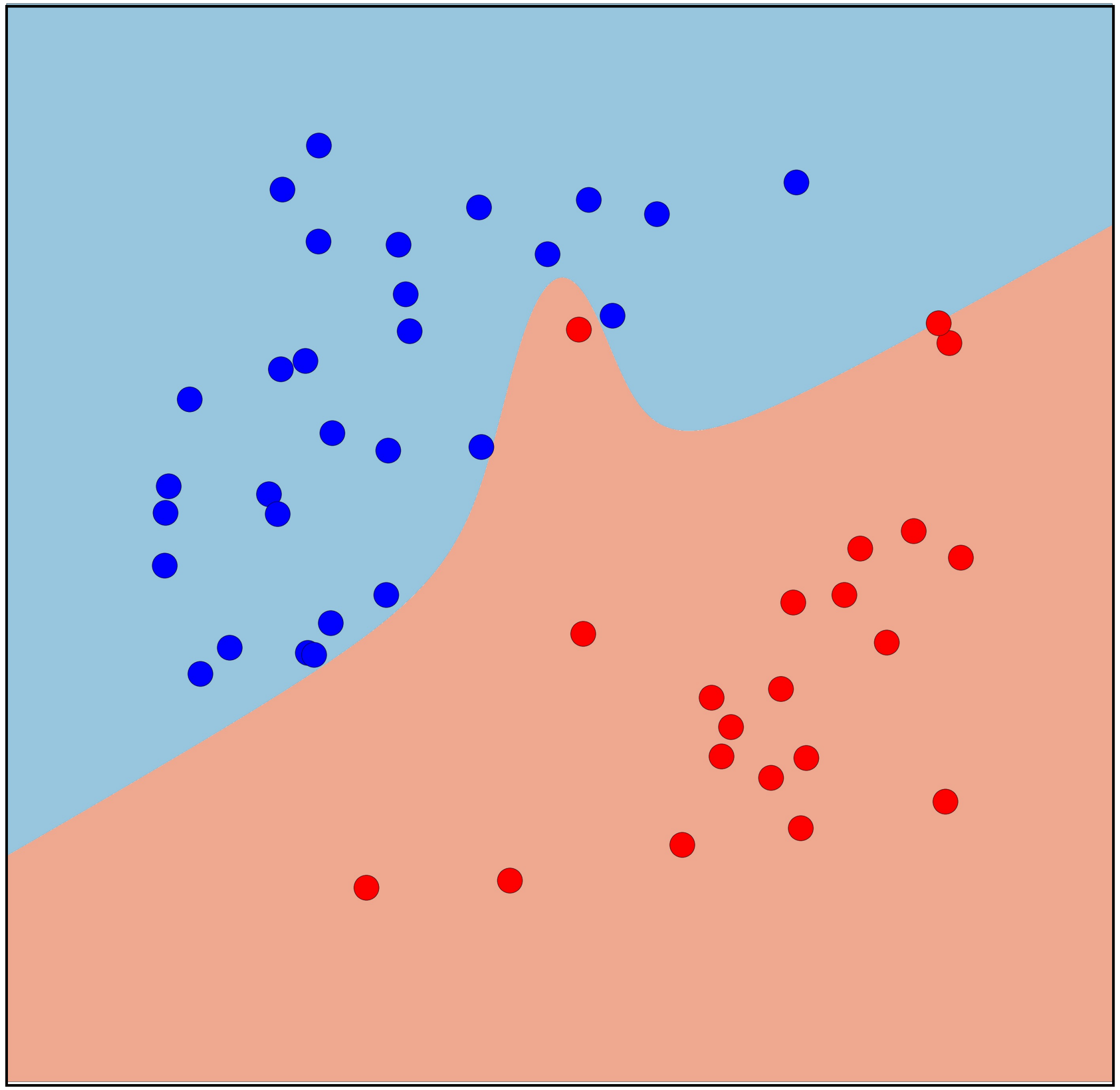
Soft Margin SVM

- What if dimension of feature space gets unnecessary high?
- Solution: Allow wrong classifications but punish them
- Optimization Problem:

$$L_D = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle$$

subject to: $\sum_i \alpha_i y_i = 0,\ 0 \leq \alpha_i \leq C$

- C is regularization parameter
- Overfitting: $C \rightarrow \infty$



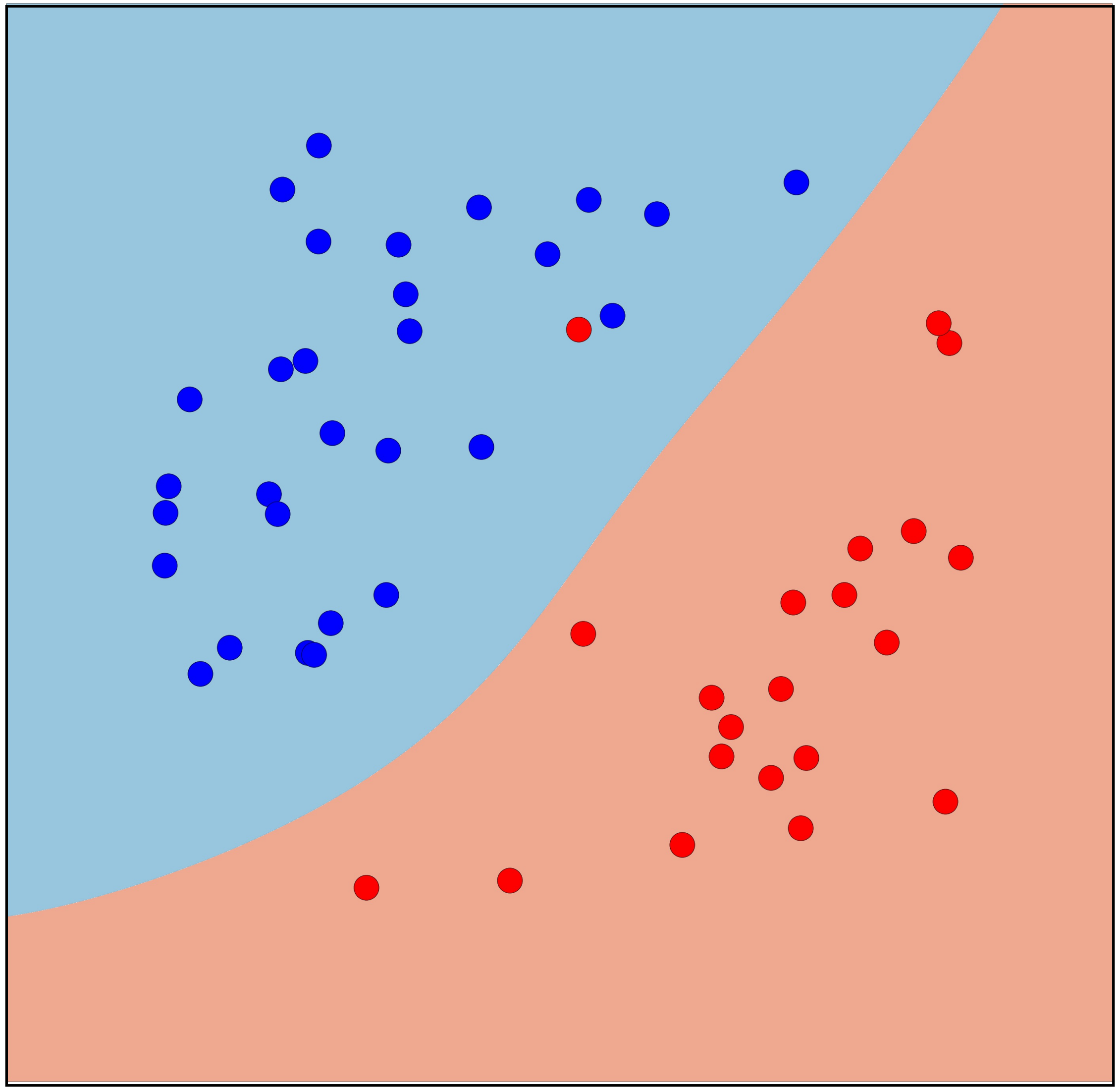
Soft Margin SVM

- What if dimension of feature space gets unnecessary high?
- Solution: Allow wrong classifications but punish them
- Optimization Problem:

$$L_D = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle$$

subject to: $\sum_i \alpha_i y_i = 0,\ 0 \leq \alpha_i \leq C$

- C is regularization parameter
- Overfitting: $C \rightarrow \infty$
- Underfitting: $C \rightarrow 0$



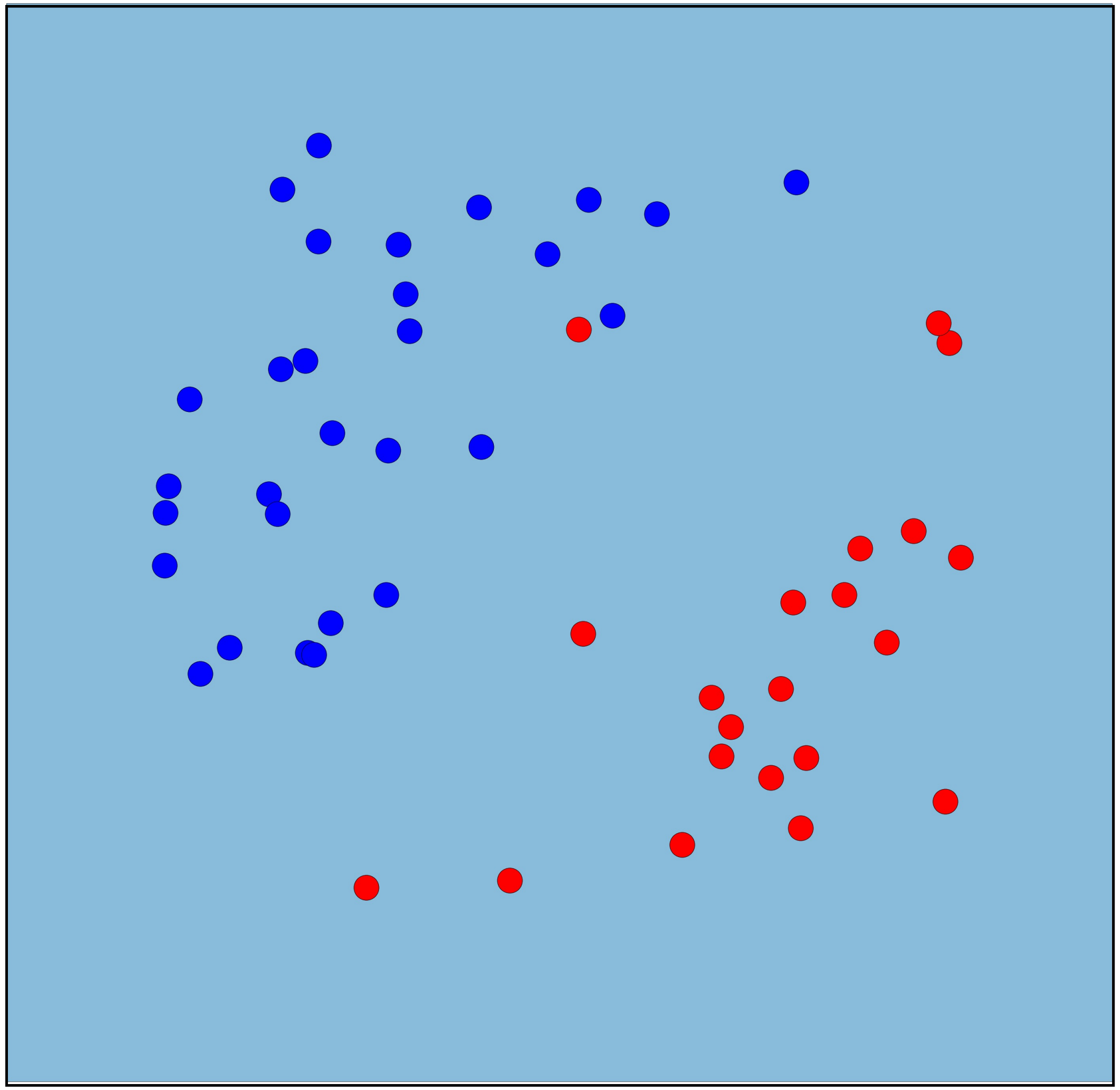
Soft Margin SVM

- What if dimension of feature space gets unnecessary high?
- Solution: Allow wrong classifications but punish them
- Optimization Problem:

$$L_D = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle$$

subject to: $\sum_i \alpha_i y_i = 0,\ 0 \le \alpha_i \le C$

- C is regularization parameter
- Overfitting: $C \rightarrow \infty$
- Underfitting: $C \rightarrow 0$



Kernel Method

- Even with Soft Margin SVM the dimension of the feature map becomes infeasible very fast

→ Kernel Trick

- Don't need to compute feature map explicitly
- Define kernel function:

$$k(\boldsymbol{x}, \boldsymbol{y}) = \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{y}) \rangle$$

- Kernel matrix K:

$$K_{ij} = k\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right)$$

- K positive semi-definite $\Leftrightarrow$ k is kernel

Kernel Method

- Use Kernel in SVM optimization problem
- Replace all inner products with kernel:

$$L_D = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j k(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

- Also replace inner products in decision function:

$$\widetilde{m}(\boldsymbol{s}) = \mathrm{sign}\left(\sum_{i \in N_S} \alpha_i^* y_i k(\boldsymbol{x}_i, \boldsymbol{s}) + b\right)$$

Kernel Method

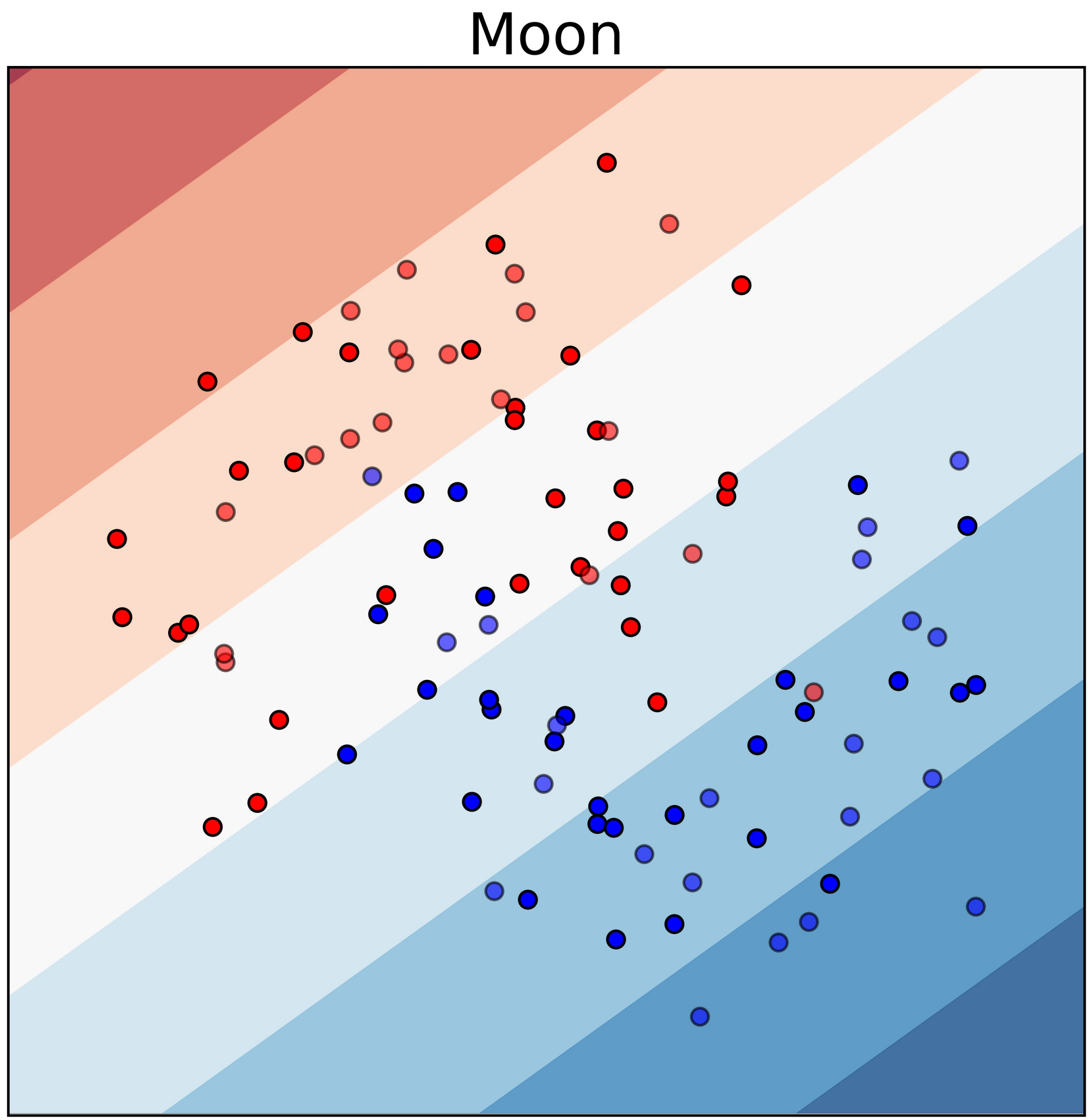
Example Kernels

- Linear Kernel

$$k(x, y) = \langle x, y \rangle$$

- Polynomial Kernel

$$k(x, y) = (\langle x, y \rangle + c)^k$$



Kernel Method

Example Kernels

- Linear Kernel

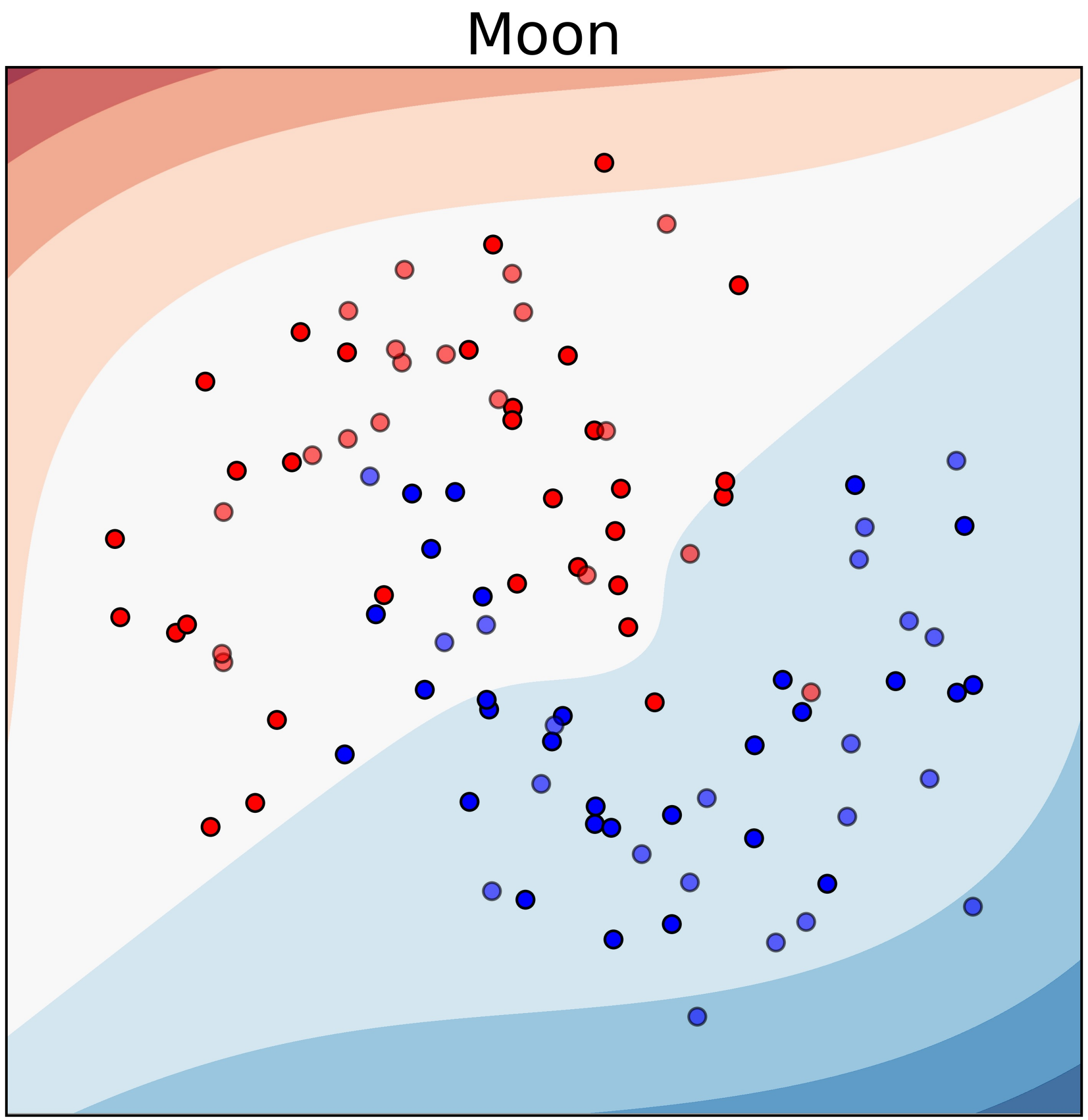
$$k(x, y) = \langle x, y \rangle$$

- Polynomial Kernel

$$k(x, y) = (\langle x, y \rangle + c)^k$$

- RBF Kernel

$$k(x, y) = \exp\left(\frac{-\|x - y\|^2}{2\sigma^2}\right)$$



Kernel Method

Example Kernels

- Linear Kernel

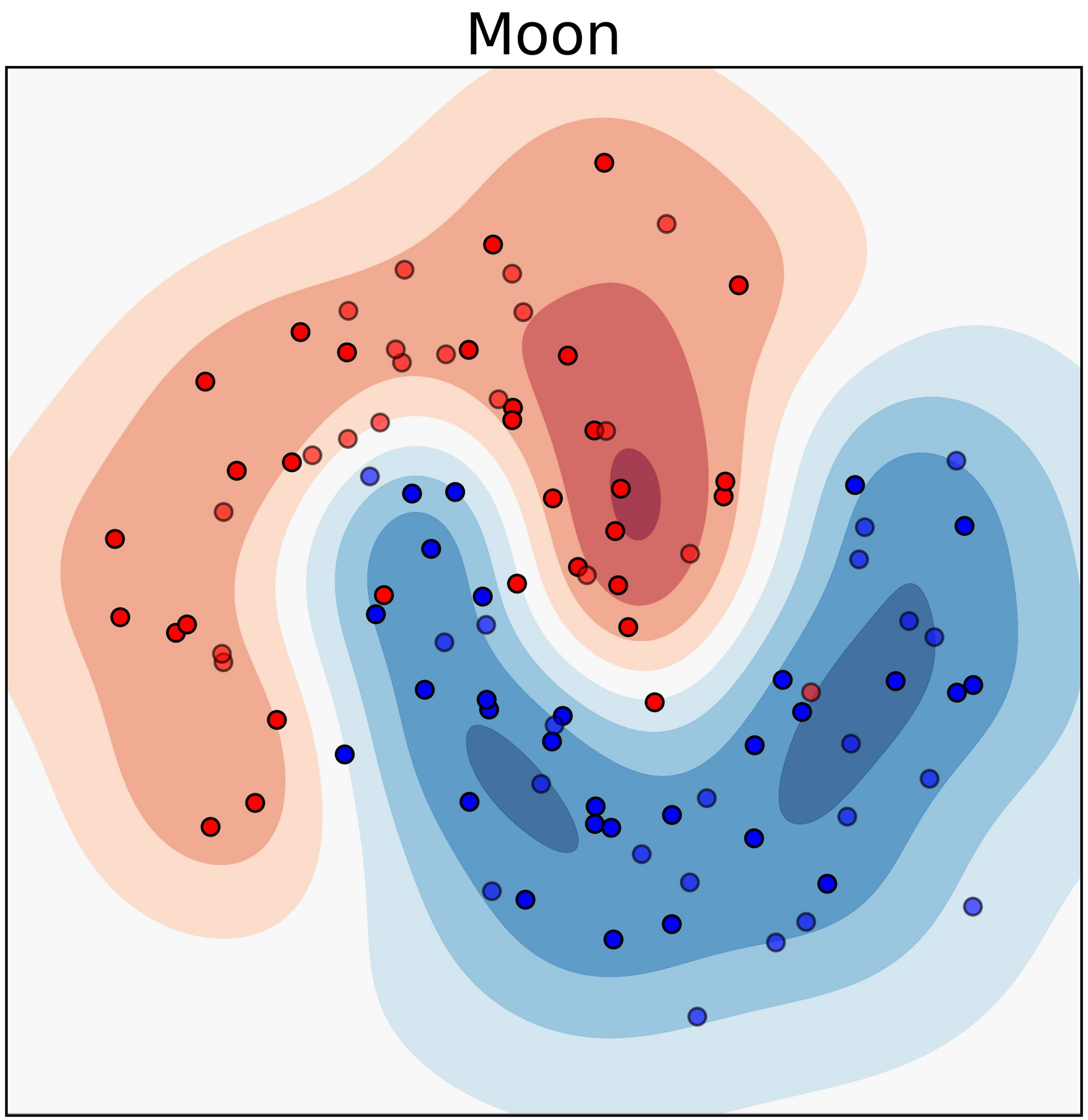
$$k(x,y) = \langle x, y \rangle$$

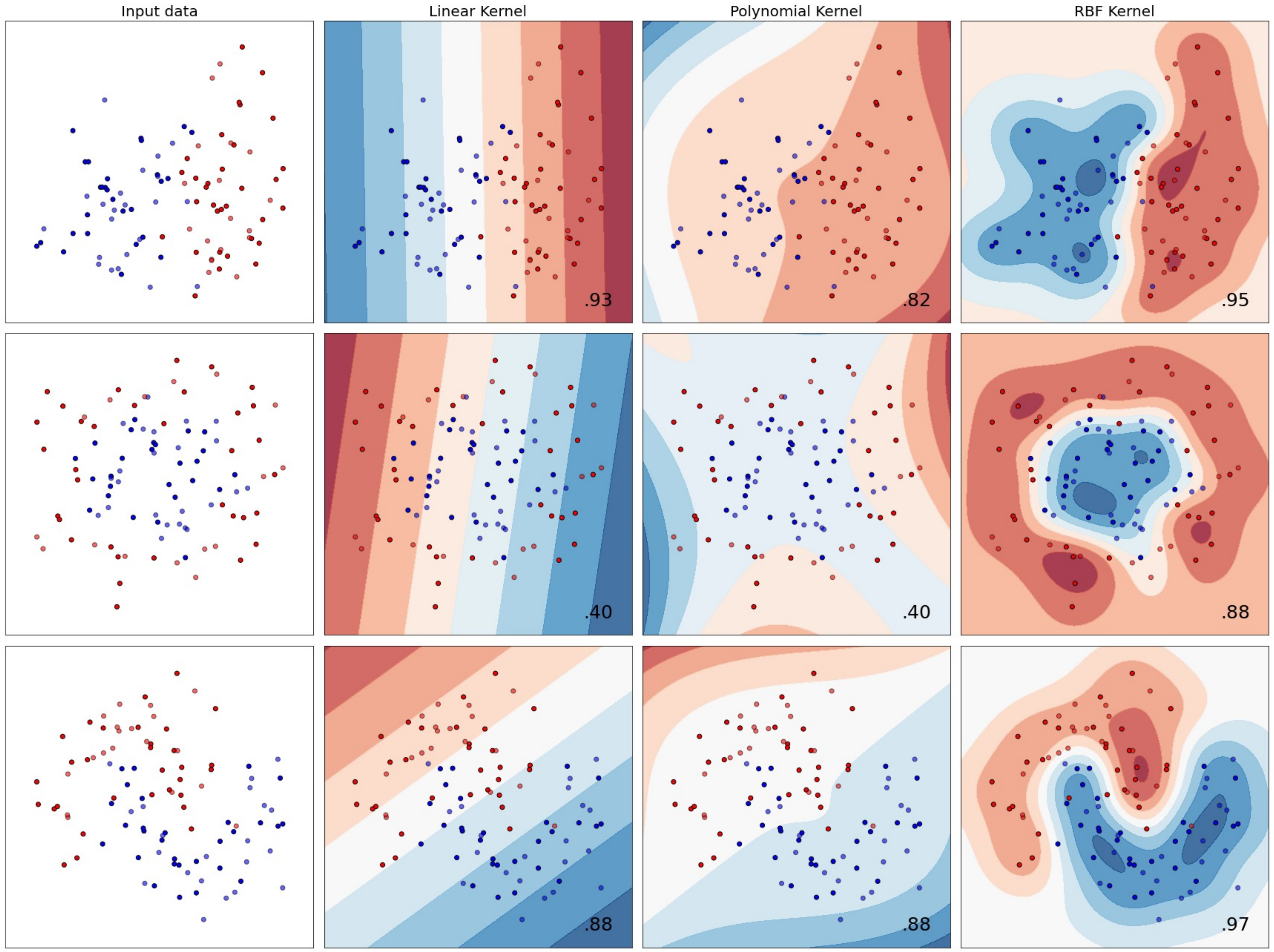
- Polynomial Kernel

$$k(x,y) = (\langle x, y \rangle + c)^k$$

- RBF Kernel

$$k(x,y) = \exp\left(\frac{-\|x-y\|^2}{2\sigma^2}\right)$$



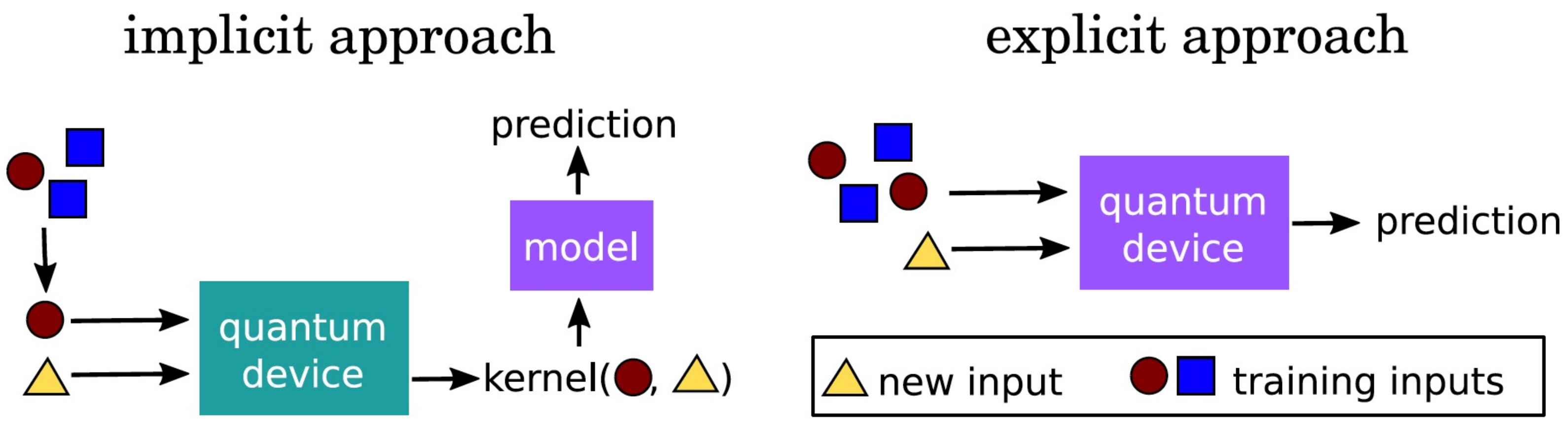

Input data
Linear Kernel
Polynomial Kernel
RBF Kernel
.93
.82
.95
.40
.40
.88
.88
.88
.97

QML

- Why quantum machine learning?
- Use feature map of form

$$\phi : x \longmapsto |\phi(x)\rangle\langle\phi(x)|$$

- Two methods
    - Quantum Variational Classifier
    - Quantum Kernel Estimator



Schuld et al. [2]

Quantum Kernel Estimator

- Use kernel of form

$$k(\boldsymbol{x}, \boldsymbol{y}) = |\langle \phi(\boldsymbol{x}) | \phi(\boldsymbol{y}) \rangle|^2$$

- Use Quantum computer twice:
  1. $k(\boldsymbol{x}_i, \boldsymbol{x}_j)\ \forall \boldsymbol{x}_i, \boldsymbol{x}_j \in T$
  2. for $\boldsymbol{s} \in S$: $k(\boldsymbol{s}, \boldsymbol{x}_i)\ \forall i \in N_S$

→ How to get quantum advantage?

Quantum Kernel Estimator

Havlíček et al. [1]

- Use map based on circuit, that is hard to compute classically
- Feature map on n-qubits by unitary:

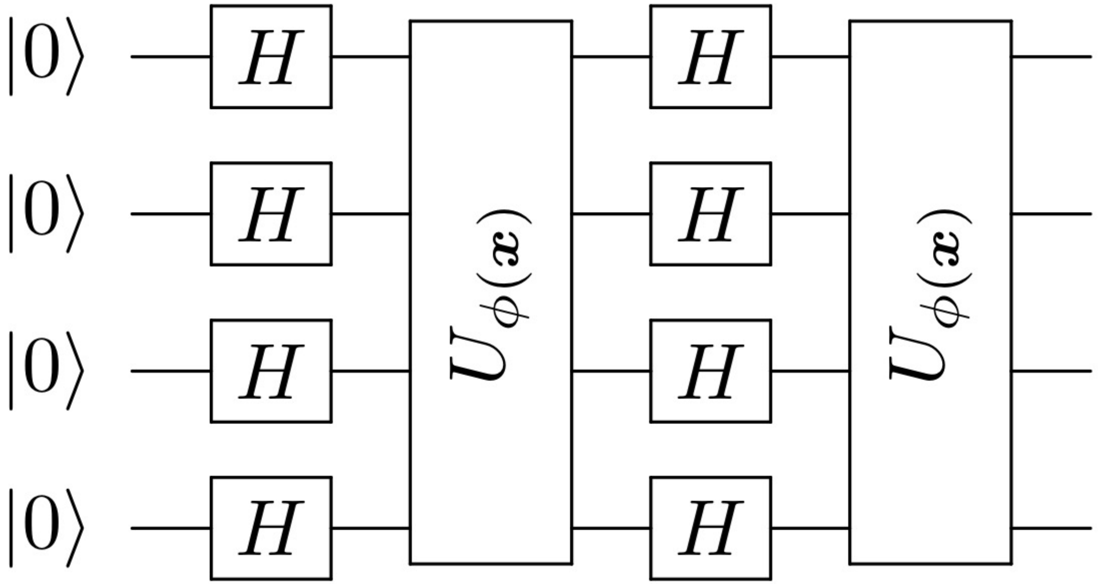
$$\mathcal{U}_{\phi(x)} = U_{\phi(x)} H^{\otimes n} U_{\phi(x)} H^{\otimes n}$$

- H being Hadamard gate and

$$U_{\phi(x)} = \exp\left[i \sum_{S \subseteq [n]} \phi_S(x) \prod_{i \in S} Z_i \right]$$

- Encode data in quantum state with

$$|\phi(x)\rangle = \mathcal{U}_{\phi(x)} |0^n\rangle$$

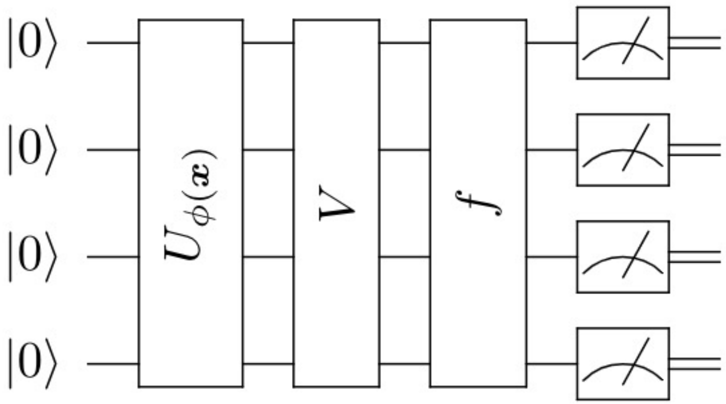


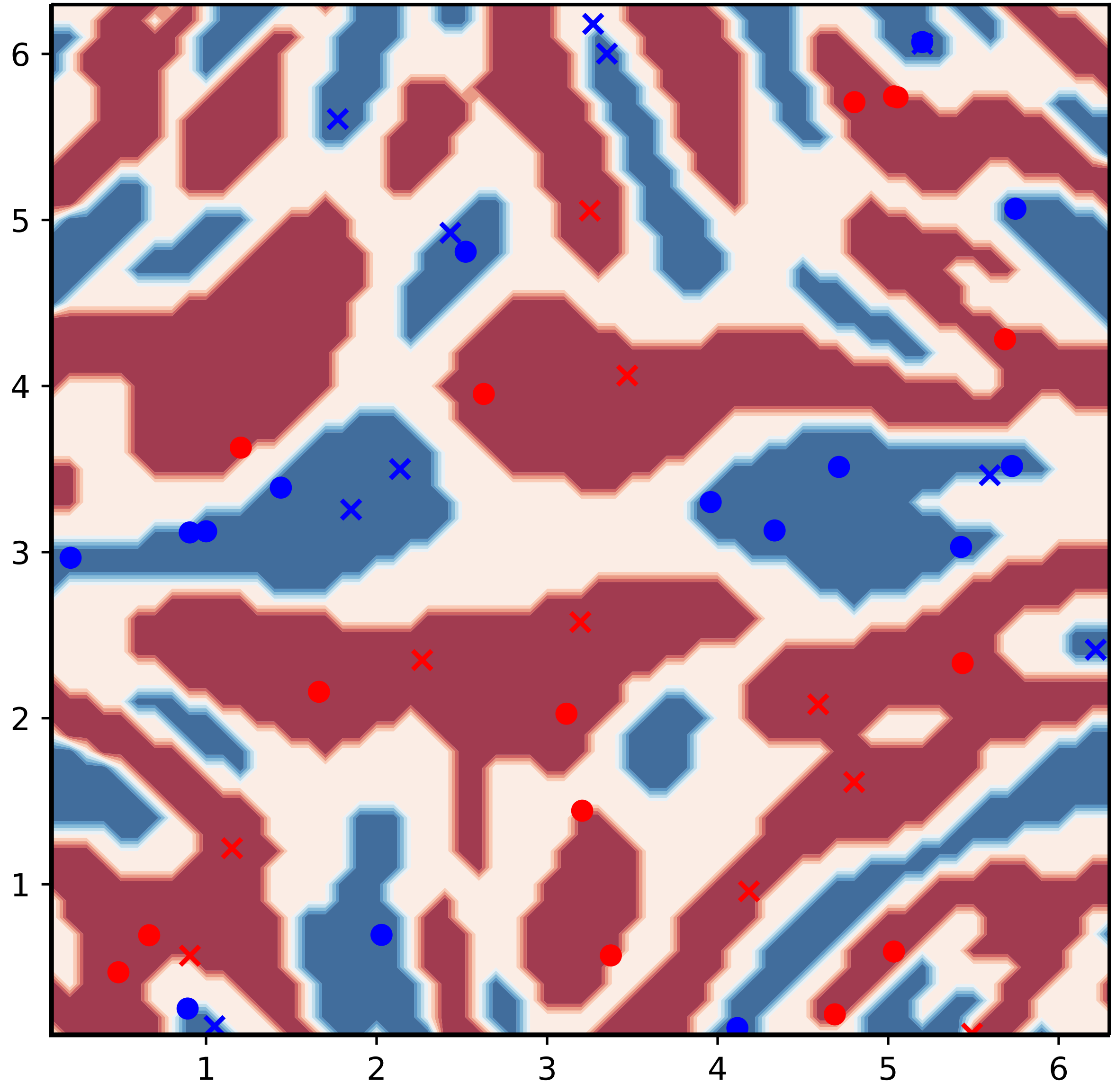
Quantum Kernel Estimator

Create Artificial Data:

- Data points:
  $$x \in T \cup S \subset (0,2\pi]^2$$
- $m(x) = \begin{cases} +1 : \langle \phi(x) | V^\dagger f V | \phi(x) \rangle \geq \Delta \\ -1 : \langle \phi(x) | V^\dagger f V | \phi(x) \rangle \leq -\Delta \end{cases}$

with $\Delta = 0.3, f = Z_1 Z_2, V \in SU(4)$

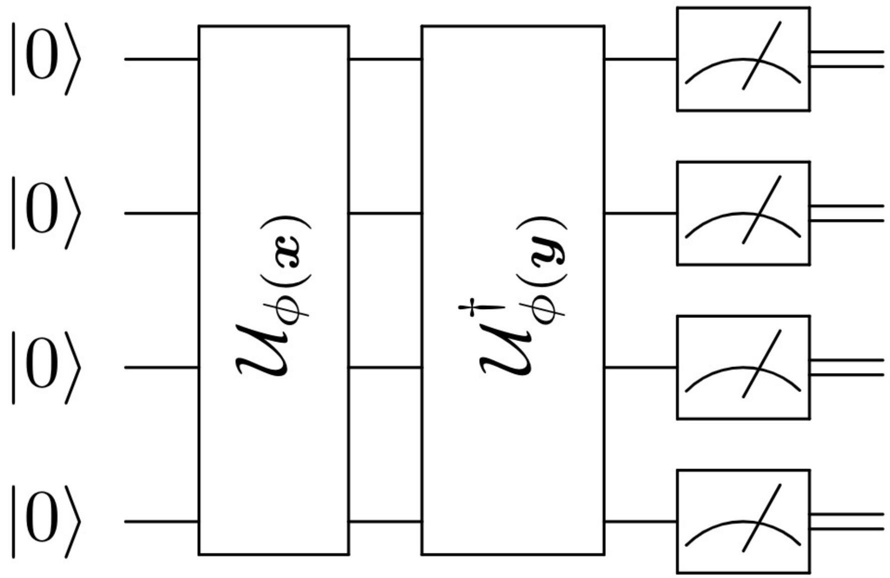


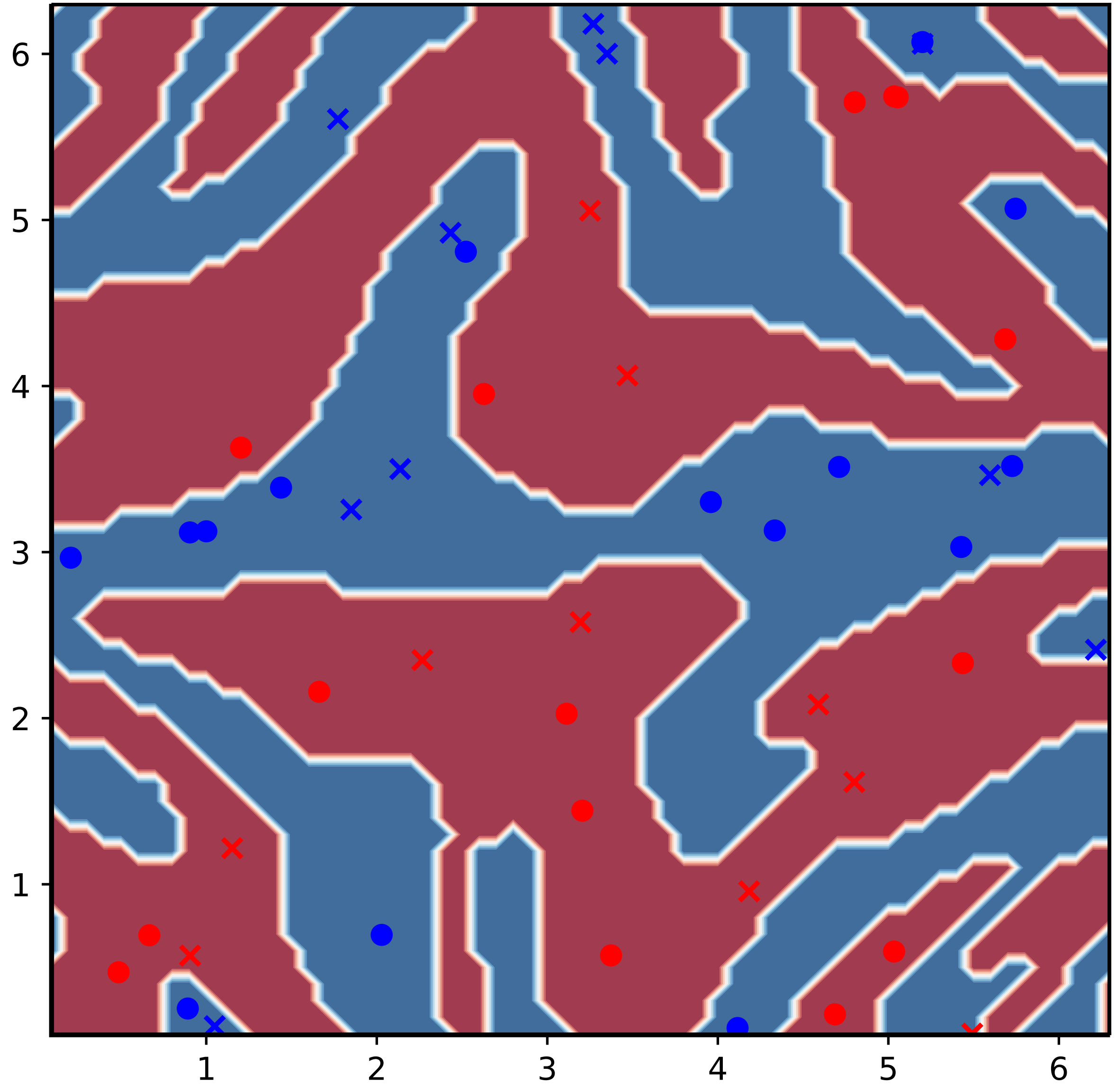


Quantum Kernel Estimator

- Compute quantum kernel

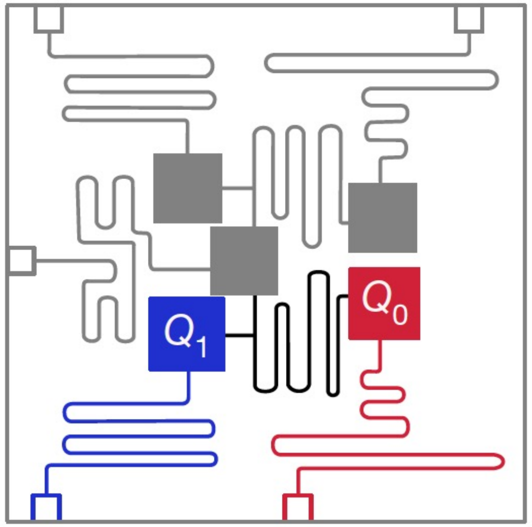
$$k(x, y) = |\langle \phi(x) | \phi(y) \rangle|^2$$
$$= \left| \left\langle 0^n \middle| \mathcal{U}^\dagger_{\phi(y)} \mathcal{U}_{\phi(x)} \middle| 0^n \right\rangle \right|^2$$





Havlíček et al. [1] results

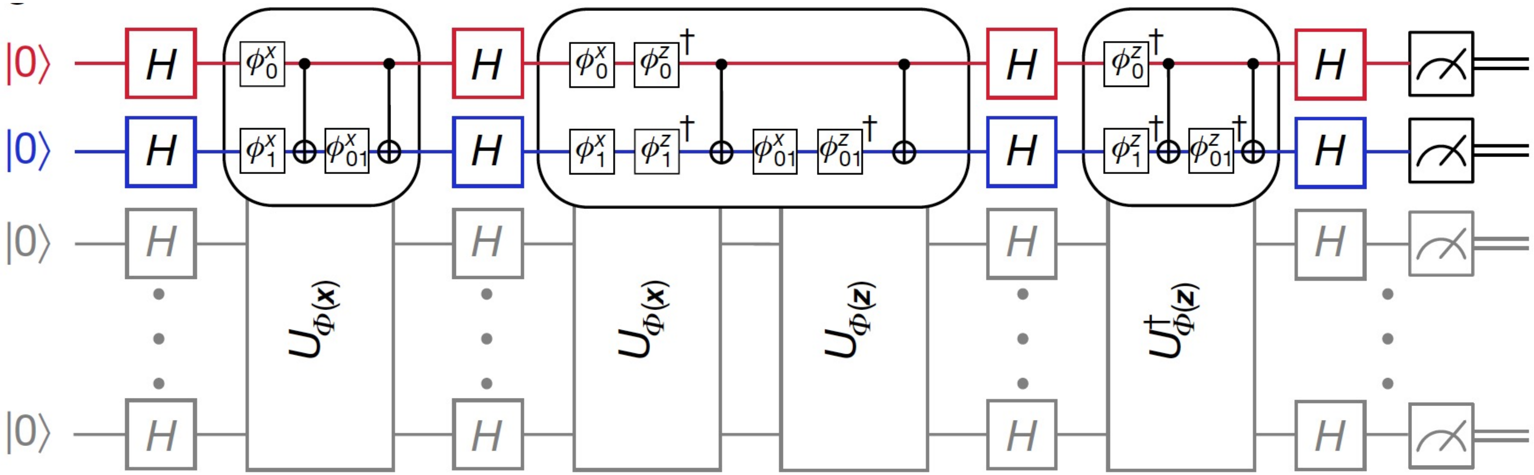
Five-qubit quantum processor



- Using same unitary

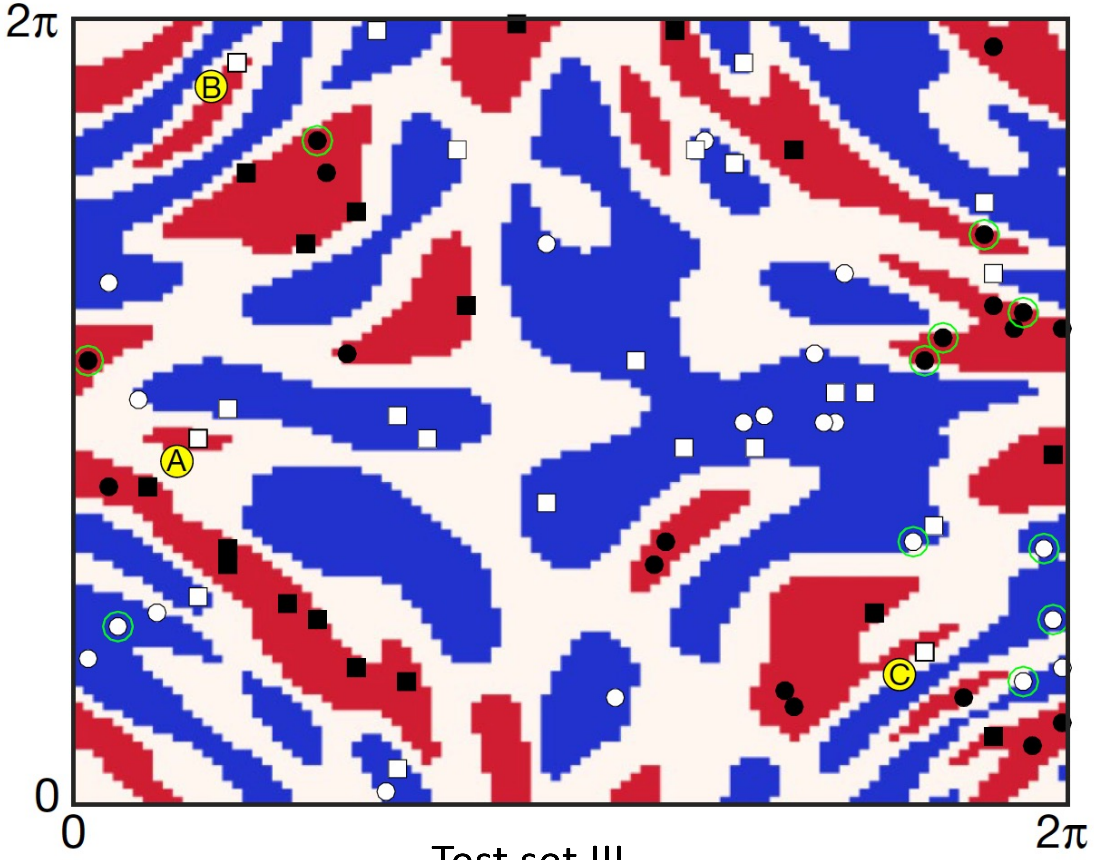
$$|\phi(\boldsymbol{x})\rangle = \mathcal{U}_{\phi(\boldsymbol{x})}|0^n\rangle$$

$$U_{\phi(\boldsymbol{x})} = \exp\left[i \sum_{S\subseteq[n]} \phi_S(\boldsymbol{x}) \prod_{i\in S} Z_i\right]$$



Havlíček et al. [1] results

Five-qubit quantum processor



Test set III

Test set success rate:

1. 100%
2. 100%
3. 94.75%

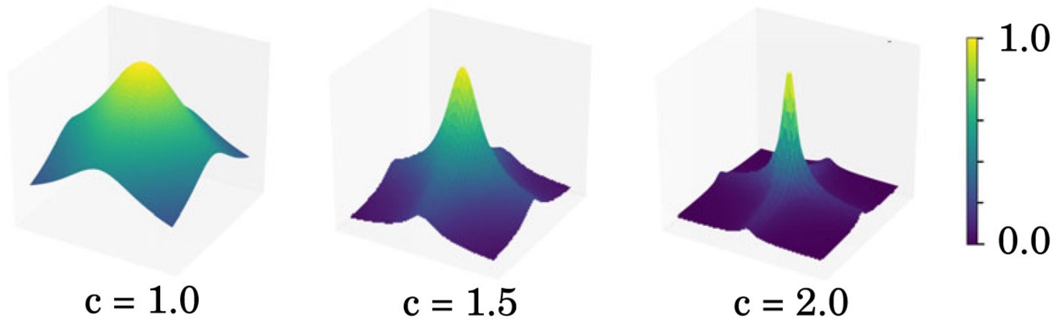
Schuld et al. [2]

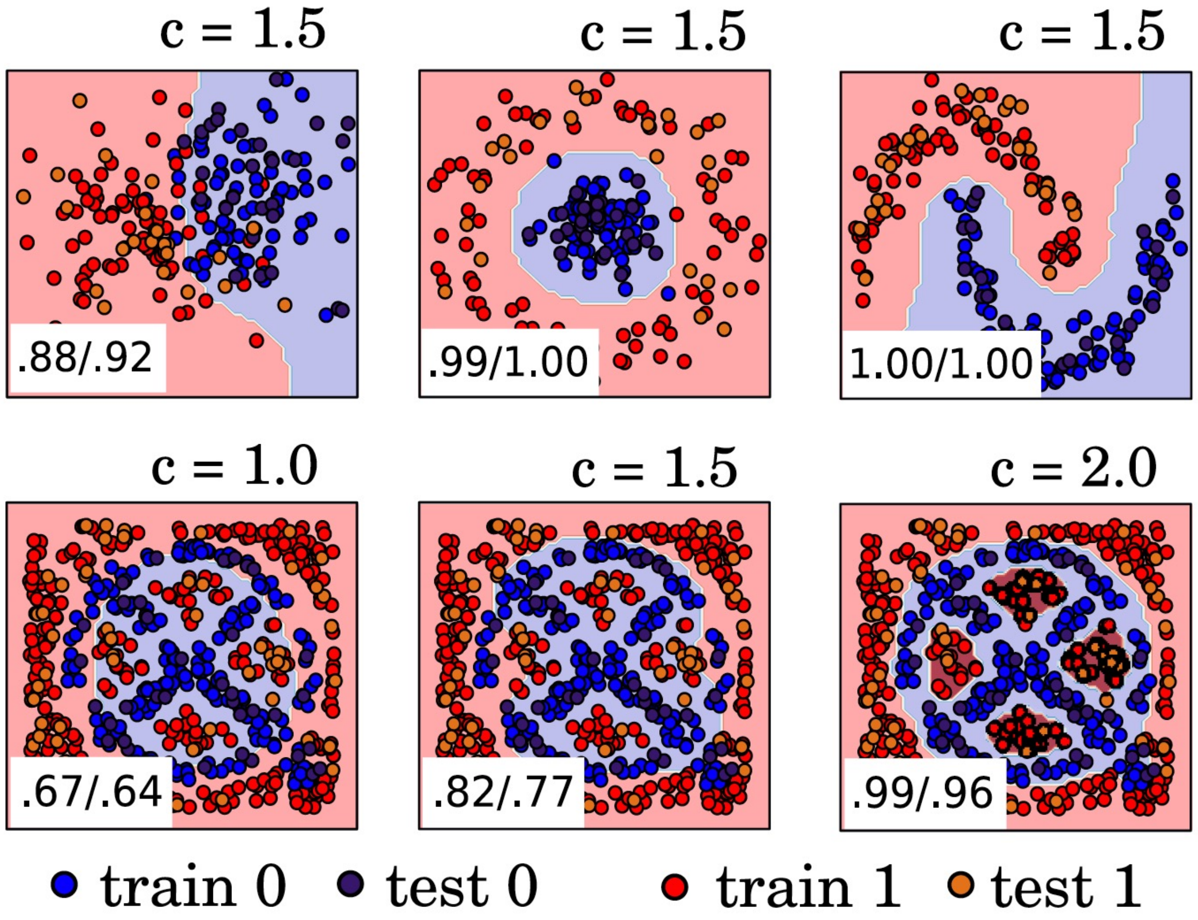
- Use squeezing as feature map

$$k(\boldsymbol{x}, \boldsymbol{y}; c) = \prod_{i=1}^{N} \langle (c, x_i) | (c, y_i) \rangle$$

with

$$\langle (c, x_i) | (c, y_i) \rangle = \sqrt{\frac{\operatorname{sech} c \operatorname{sech} c}{1 - e^{i(y_i - x_i)} \tanh c \tanh c}}$$





Conclusion & Outlook

- Possible quantum advantage
  - Find more classically hard to compute quantum kernels
- No real-world scenario for quantum machine learning
  - Find real-world applications for quantum machine learning

THANK YOU FOR LISTENING

References

- Havlíček et al.
  - Nature 567, 209-212 (2019)
- Schuld et al.
  - arXiv:1803.07128

Extra

Definition of $\phi_S(x)$ for $n = d = 2$:

$\phi_{\{i\}}(x) = x_i$

$\phi_{\{1,2\}}(x) = (\pi - x_1)(\pi - x_2)$

$$U_{\phi(x)} = \begin{bmatrix} e^{i(x_1+x_2+Q)} & 0 & 0 & 0 \\ 0 & e^{i(x_1-x_2-Q)} & 0 & 0 \\ 0 & 0 & e^{i(-x_1+x_2-Q)} & 0 \\ 0 & 0 & 0 & e^{i(-x_1-x_2+Q)} \end{bmatrix}$$

with $Q \coloneqq (\pi - x_1)(\pi - x_2)$

Extra

- Prediction with RBF Kernel on quantum data set