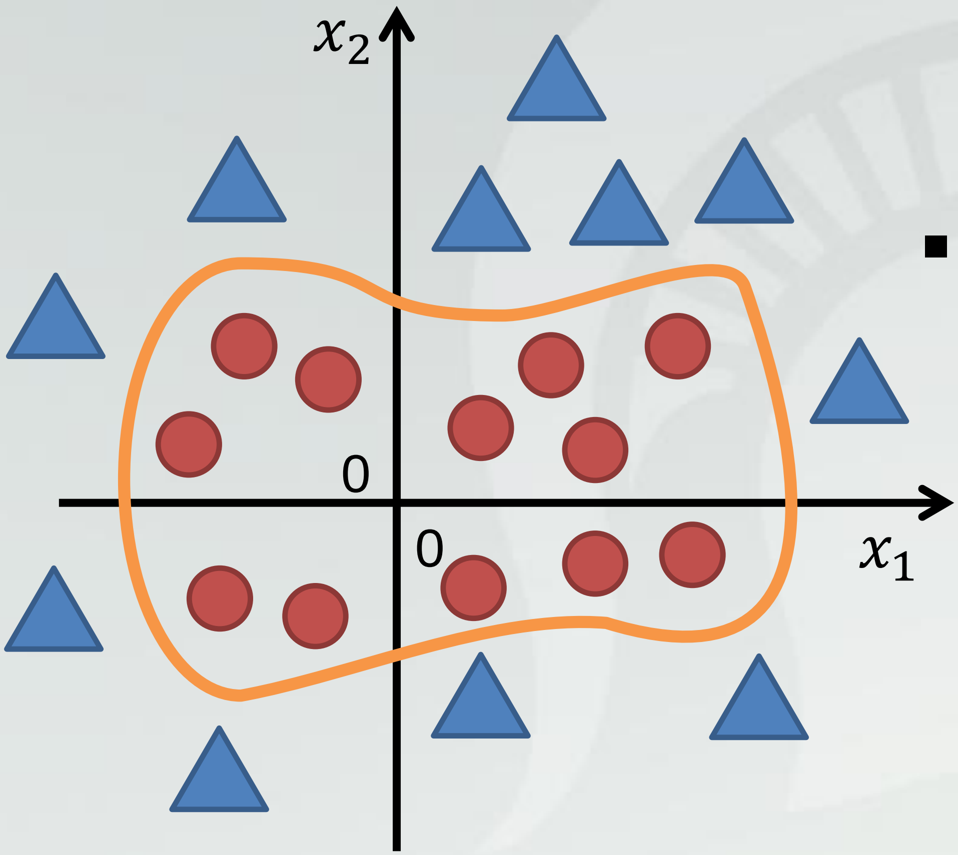
Support Vector Machine (SVM)-II
Nonlinear predictors

Guowei Wei
Department of Mathematics
Michigan State University

References:
Duc D. Nguyen's lecture notes
Wikipedia

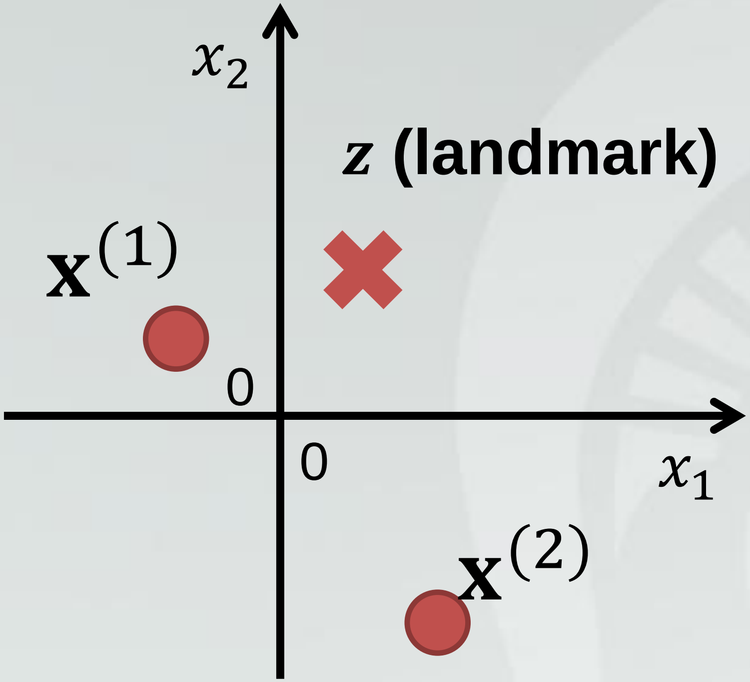
SVM for Nonlinear Classifiers



- Linear predictor:
  $p_{\mathbf{c}}(\mathbf{x}) = \mathbf{c}^T\mathbf{x}$
  $= c_0 + c_1 x_1 + \cdots + c_n x_n$
- Nonlinear predictor => nonlinear decision boundary:
  - $p_{\mathbf{c}}(\mathbf{x}) = c_0 + c_{11}x_1 + \cdots + c_{1k}x_1^k + c_{21}x_2 + \cdots$
  - $p_{\mathbf{c}}(\mathbf{x}) = c_0 + c_1 x_1 + c_2 x_1^2 + c_3 x_1 x_2 + \cdots + c_m x_{n-1} x_n$

Drawback: High risk of overfitting

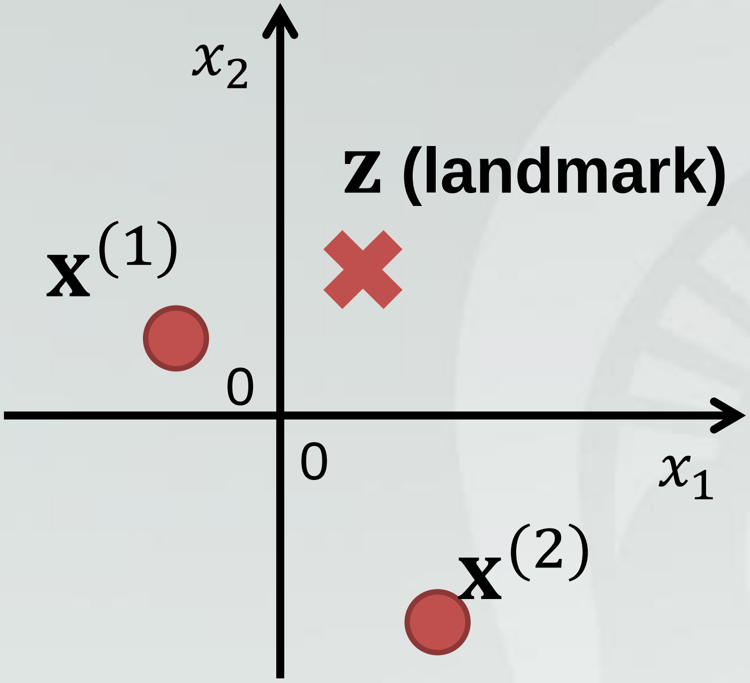
SVM for Nonlinear Classifiers



- Use kernel (Kernel method, Vapnik 1963)
  - A similarity function $k(\mathbf{x}, \boldsymbol{z})$
  - $k(\mathbf{x}, \mathbf{z})$ define how similar a given data point $\mathbf{x}$ to the pre-defined landmark $\boldsymbol{z}$
  - $\mathbf{x}^{(1)}$ is more similar (or close) to $\mathbf{z}$ than $\mathbf{x}^{(2)}$ if

$$k\left(\mathbf{x}^{(1)}, \mathbf{z}\right) > k\left(\mathbf{x}^{(2)}, \mathbf{z}\right)$$

SVM for Kernel Classifiers



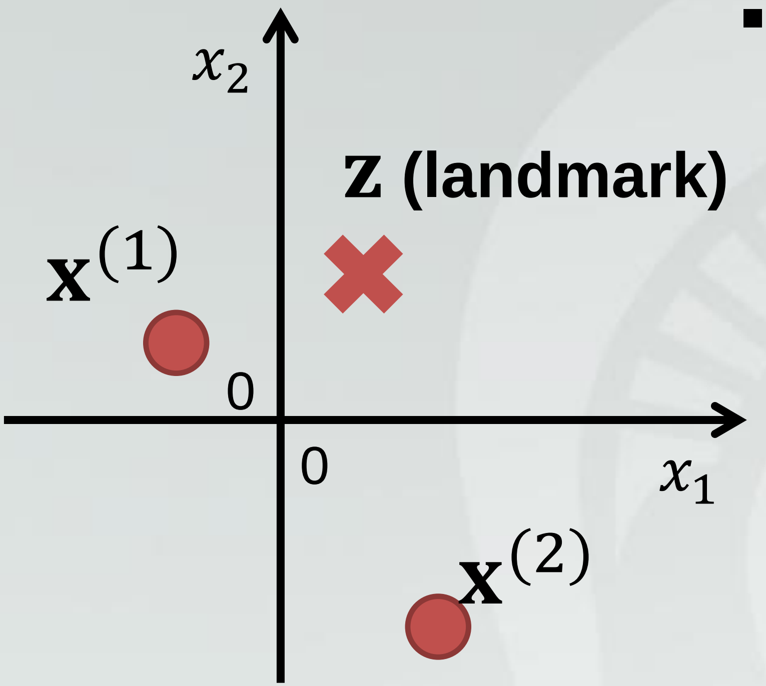
- Use kernel
  - A similarity function $k(\mathbf{x}, \mathbf{z})$
  - $k(\mathbf{x}, \mathbf{z})$ define how similar a given data point $\mathbf{x}$ to the pre-defined landmark $\boldsymbol{z}$
  - $\mathbf{x_1}$ is more similar (or close) to $\mathbf{z}$ than $\mathbf{x_2}$ if $k(\mathbf{x_1}, \mathbf{z}) > k(\mathbf{x_2}, \mathbf{z})$

Kernel functions:

$$k(\mathbf{x}, \mathbf{z}) = \frac{1}{1 + \|\mathbf{x} - \boldsymbol{z}\|}$$

$$k(\mathbf{x}, \mathbf{z}) = \frac{1}{1 + \left(\frac{\|\mathbf{x} - \boldsymbol{z}\|}{\eta}\right)^{\nu}} \quad \text{(lorentz)}$$

SVM for kernel Classifiers



- Use kernel
  - $\mathbf{x}^{(1)}$ is more similar (or close) to $\mathbf{z}$ than $\mathbf{x}^{(2)}$ if $k\left(\mathbf{x}^{(1)}, \mathbf{z}\right) > k\left(\mathbf{x}^{(2)}, \mathbf{z}\right)$
  - Kernel functions:

$$k(\mathbf{x}, \mathbf{z}) = \frac{1}{1 + \|\mathbf{x} - \mathbf{z}\|}$$

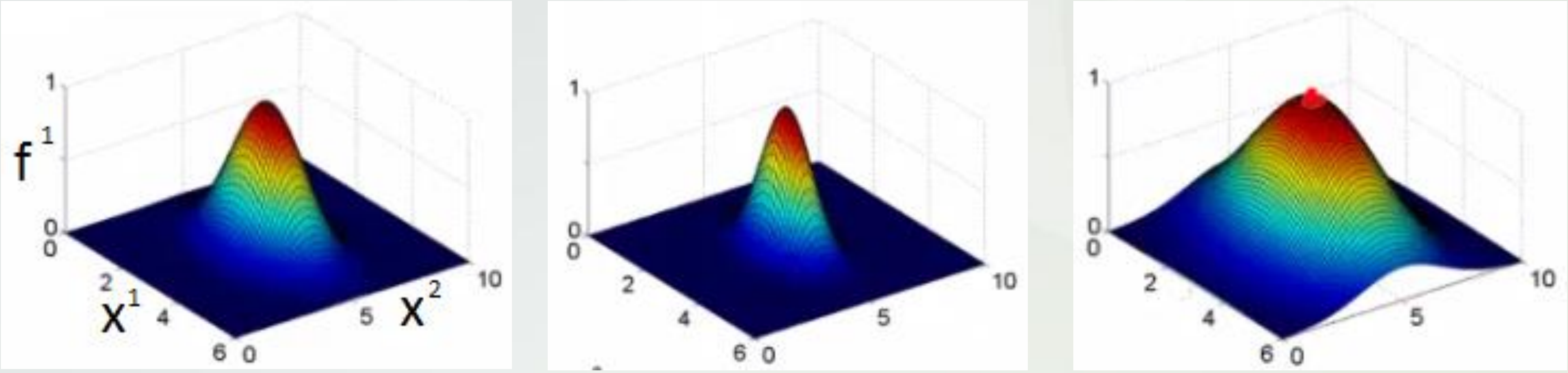
$$k(\mathbf{x}, \mathbf{z}) = \frac{1}{1 + \left(\frac{\|\mathbf{x} - \mathbf{z}\|}{\sigma}\right)^{\nu}} \quad \text{(Lorentz)}$$

$$k(\mathbf{x}, z) = e^{-\left(\frac{\|\mathbf{x} - \mathbf{z}\|}{\sigma}\right)^{\nu}} \quad \text{(exponential)}$$

- if $\mathbf{x}$ very close to $\mathbf{z} \Rightarrow \|\mathbf{x} - \mathbf{z}\| \to 0 \Rightarrow k(\mathbf{x}, \mathbf{z}) \to 1$
- if $\mathbf{x}$ far away from $\mathbf{z} \Rightarrow \|\mathbf{x} - \mathbf{z}\| \to \infty \Rightarrow k(\mathbf{x}, \mathbf{z}) \to 0$
- In exponential kernel, when $\nu = 2$ we get Gauss kernel $e^{-\left(\frac{\|x - z\|}{\sigma}\right)^{2}}$

SVM for Nonlinear Classifiers

- Gaussian kernel: $\mathrm{k}(\mathbf{x}, \boldsymbol{z}) = e^{-\left(\frac{\|\mathbf{x}-\mathbf{z}\|}{\sigma}\right)^2}$
- σ: standard deviation
- σ^2: variance, define how steep from the landmark (the top) to the ground
- $\mathbf{z} = (3,5)^T$ with three σ^2 values: 1, 0.5, and 3.0

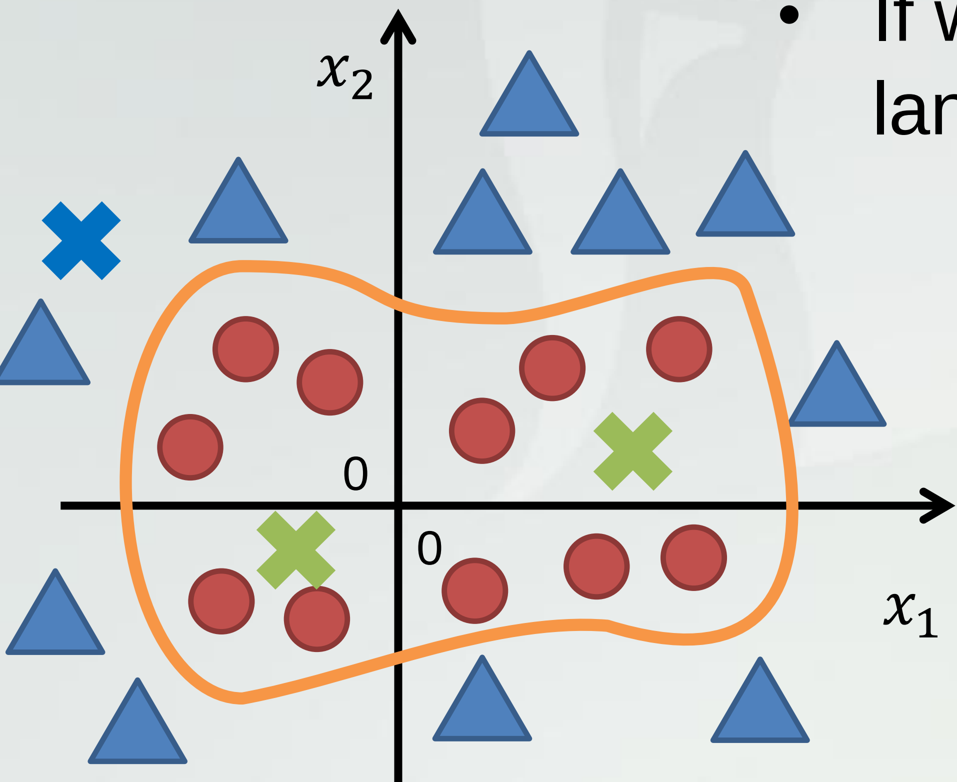


Predictor with Kernels

- Make use the predictor for linear classifier

$$p_{\mathbf{c}}(\mathbf{x}) = c_0 + c_1 x_1 + \cdots + c_n x_n$$

- If we use landmarks = use similarity functions = use kernels



- If we use three landmarks: $\boldsymbol{z}^{(1)}, \boldsymbol{z}^{(2)}, \boldsymbol{z}^{(3)}$

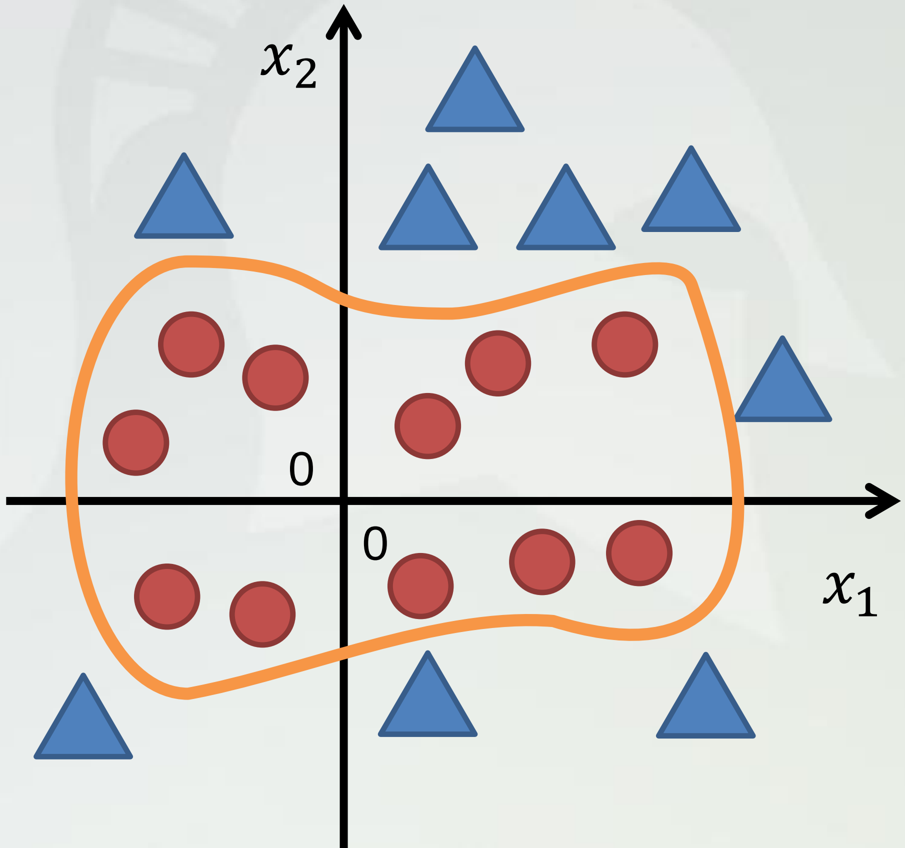
$$p_{\mathbf{c}}(\mathbf{x}) = c_0 + c_1 k\left(\mathbf{x}, \boldsymbol{z}^{(1)}\right) + c_2 k\left(\mathbf{x}, \boldsymbol{z}^{(2)}\right) + c_3 k\left(\mathbf{x}, \boldsymbol{z}^{(3)}\right)$$

How to Choose Landmarks?

- Assume our training data is

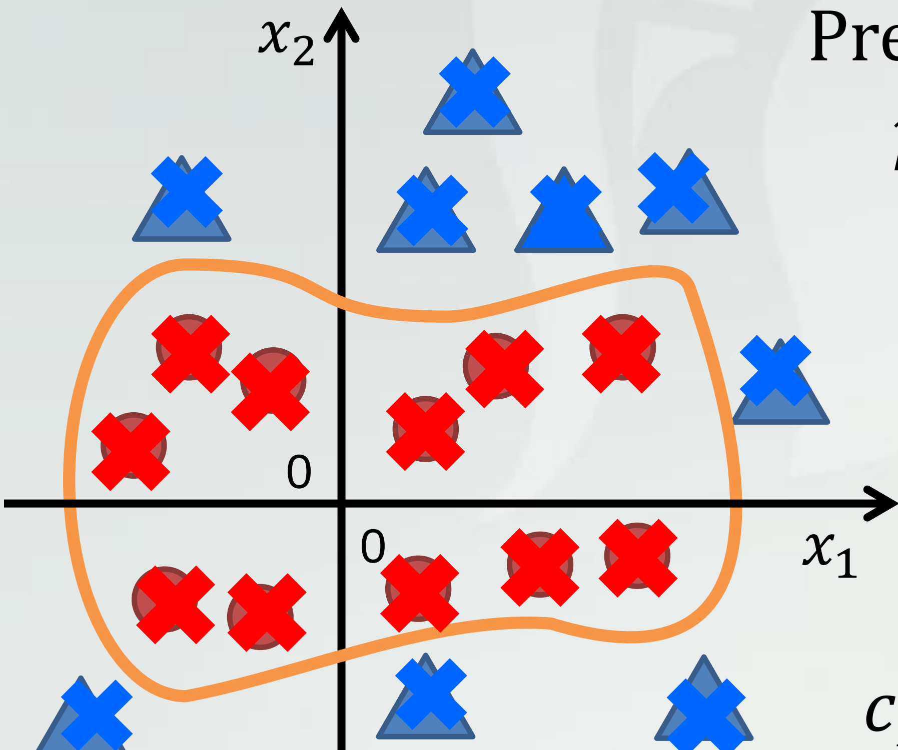
$$\left(\mathbf{x}^{(1)}, y^{(1)}\right), \left(\mathbf{x}^{(2)}, y^{(2)}\right), \ldots, \left(\mathbf{x}^{(M)}, y^{(M)}\right)$$

- How to choose landmarks for a given training data? (The kernel trick, Guyon and Vapnik, 1992)



How to Choose Landmarks?

- Assume our training data is
  $$\left(\mathbf{x}^{(1)}, y^{(1)}\right), \left(\mathbf{x}^{(2)}, y^{(2)}\right), \ldots, \left(\mathbf{x}^{(M)}, y^{(M)}\right)$$
- How to choose landmarks for a given training data? $\mathbf{z}^{(1)} = \mathbf{x}^{(1)}, \mathbf{z}^{(2)} = \mathbf{x}^{(2)}, \ldots, \mathbf{z}^{(M)} = \mathbf{x}^{(M)}$



Predictor for M landmarks

$$p_{\mathbf{c}}(\mathbf{x}) = c_0 + c_1 k\left(\mathbf{x}, \mathbf{z}^{(1)}\right) + c_2 k\left(\mathbf{x}, \mathbf{z}^{(2)}\right) + \cdots + c_M k\left(\mathbf{x}, \mathbf{z}^{(M)}\right)$$

We have

$$p_{\mathbf{c}}(\mathbf{x}) = c_0 + c_1 k\left(\mathbf{x}, \mathbf{x}^{(1)}\right) + c_2 k\left(\mathbf{x}, \mathbf{x}^{(2)}\right) + \cdots + c_M k\left(\mathbf{x}, \mathbf{x}^{(M)}\right)$$

Loss Function with Kernels

- Predictor

$$p_{\mathbf{c}}(\mathbf{x}) = c_0 + c_1 k\left(\mathbf{x}, \mathbf{x}^{(1)}\right) + \cdots + c_M k(\mathbf{x}, \mathbf{x}^{(M)})$$

- Loss function without kernel

$$L(\mathbf{c}) = \sqrt{c_1^2 + c_2^2 + \cdots + c_n^2} + \lambda \sum_{i=1}^{M} \max(0, 1 - y^{(i)} \mathbf{c}^T \mathbf{x}^{(i)})$$

- Loss function with kernels

$$L(\mathbf{c}) = \sqrt{c_1^2 + c_2^2 + \cdots + c_M^2} + \lambda \sum_{i=1}^{M} \max(0, 1 - y^{(i)} \mathbf{c}^T \mathbf{K}(\mathbf{x}^{(i)}))$$

Loss Function with Kernels

- Predictor

$$p_{\mathbf{c}}(\mathbf{x}) = c_0 + c_1 k\left(\mathbf{x}, \mathbf{x}^{(1)}\right) + \cdots + c_M k(\mathbf{x}, \mathbf{x}^{(M)})$$

- Loss function with kernels

$$L(\mathbf{c}) = \sqrt{c_1^2 + c_2^2 + \cdots + c_M^2} + \lambda \sum_{i=1}^{M} \max(0, 1 - y^{(i)} \mathbf{c}^T \mathbf{K}(\mathbf{x}^{(i)}))$$

$$\mathbf{K}(\mathbf{x}^{(i)}) \equiv \left(1, k(\mathbf{x}^{(1)}, \mathbf{x}^{(i)}), k(\mathbf{x}^{(2)}, \mathbf{x}^{(i)}), \ldots, k(\mathbf{x}^{(M)}, \mathbf{x}^{(i)})\right)^T$$

Kernel Selections for SVM

- Not all similarity kernels are valid. Must satisfy Mercer's theorem

$$k: \mathbf{x} \times \mathbf{x} \rightarrow \mathbb{R}$$

$$k(\mathbf{x}, \mathbf{z}) = k(\mathbf{z}, \mathbf{x}) \quad \text{(symmetric)}$$

$$\int\int g(\mathbf{x}) k(\mathbf{x}, \mathbf{y}) g(\mathbf{y}) d\mathbf{x} d\mathbf{y} \geq 0$$

(positive semidefinite)

for all vector $g \in \mathcal{H}$ and k

$$\int\int |k(\mathbf{x}, \mathbf{y})|^2 d\mathbf{x} d\mathbf{y} < \infty \quad \text{(Hilbert–Schmidt operator)}$$

Mercer's requirement ensures that the loss function is convex in the dual form when using quadratic optimization method.

Kernel Selections for SVM

If kernel does not meet the Mercer conditions, no global minimum is guarantee, but one can use gradient descent to find a local minimum.

Quadratic optimization:

Minimize $\frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathrm{Q}\mathbf{x} + \mathbf{c}^{T}\mathbf{x}$

Subject to $\mathrm{A}\mathbf{x} \leq \boldsymbol{b}$

Q $-$real symmetric matrix (n x n)

A $-$real matrix (m x n)

$\boldsymbol{b}$ – real vector (m)

Commonly used Kernels

- Linear kernel (or dot product kernel)

$$k(\mathbf{x}, \mathbf{z}) = \mathbf{x}^T\mathbf{z}$$

- Polynomial

$$k(\mathbf{x}, \mathbf{z}) = (\alpha\mathbf{x}^T\mathbf{z} + r)^d$$

- Radial basis function (RBF)

$$k(\mathbf{x}, \mathbf{z}) = e^{-\left(\frac{\|\mathbf{x}-\mathbf{z}\|}{\sigma}\right)^{\nu}}$$

- Sigmoid

$$\frac{1}{1 + e^{-\gamma\mathbf{x}^T\mathbf{z}}} \quad \text{or} \quad \tanh(\gamma\mathbf{x}^T\mathbf{z} + r)$$

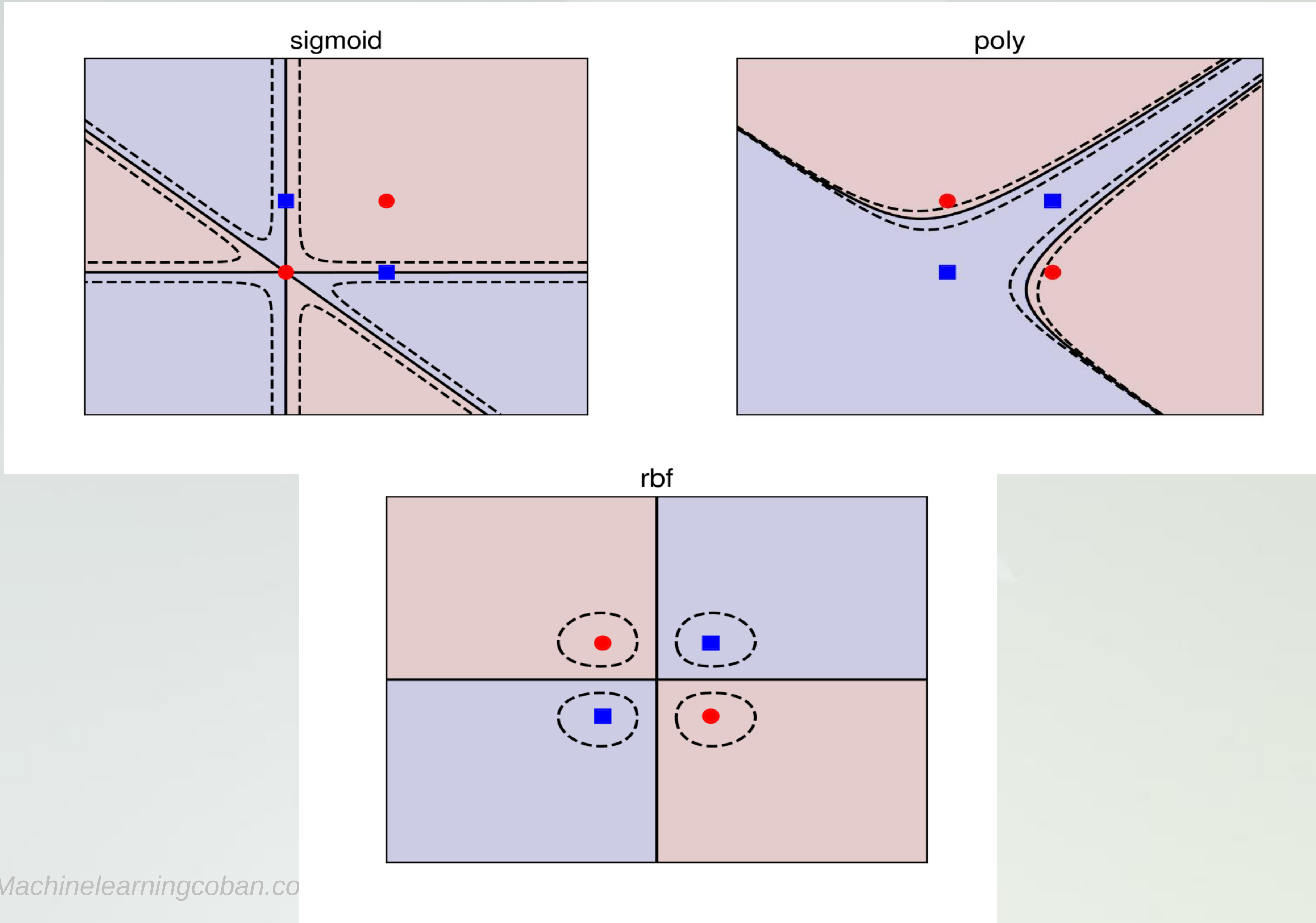
- Are these kernels Hilbert-Schmidt?

Discussions

How to Choose Kernel?

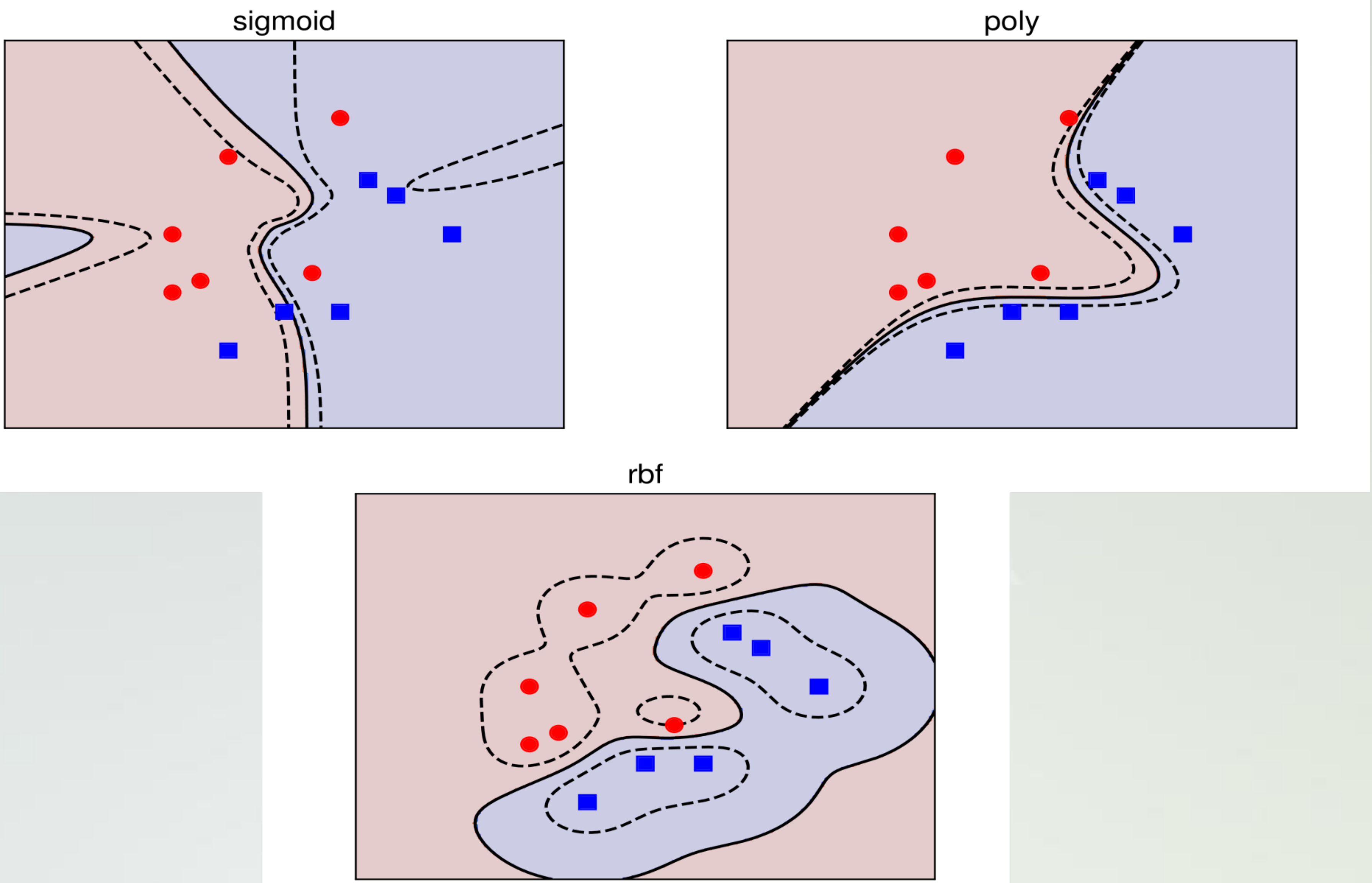
- Radial basic functions are commonly used
- Use polynomial for linear separation
- Sigmoid often performs worst
- Should try a variety of kernels for a given problem

Discussions -- Examples



(Machinelearningcoban.co

Discussions -- Examples



(Machinelearningcoban.co

Discussions

- Support vector clustering (for unsupervised learning), a fundamental method in data science
- Multiclass SVM:
  - ❖ multiple binary classification problems: https://link.springer.com/chapter/10.1007%2F11494683_28.
  - ❖single optimization problem: http://jmlr.csail.mit.edu/papers/volume2/crammer01a/crammer01a.pdf

Discussions

- **Support vector regression (SVR) (**Vladimir N. Vapnik)

  Minimize $\frac{1}{2}||\bar{\boldsymbol{c}}||^2$

  subject to $\begin{cases} y^{(i)} - \mathbf{c}^T\mathbf{x}^{(i)} \le \varepsilon \\ \mathbf{c}^T\mathbf{x}^{(i)} - y^{(i)} \le \varepsilon \end{cases}$ (where $\varepsilon \ge 0$)

- **Least squares support vector machine (LS-SVM): (**Suykens and Vandewalle)

Discussions

- **Mathematical issues?**
  1. Kernels (Reproducing-kernel-Hilbert-space kernels; Wavelets; Frames; Splines; Separable, etc.)
  2. Regularization and stability (Tikhonov)

$$\arg\min_{f\in\mathcal{H}} L(\mathbf{c}) + \mathcal{R}(\mathbf{K}), \quad \text{where } \mathcal{R}(f) = \gamma_{\boldsymbol{A}} \|f\|_{\mathcal{H}}^2$$

$$L(\mathbf{c}) = \sqrt{c_1^2 + \cdots + c_M^2} + \lambda \sum_{i=1}^{M} \max(0, 1 - y^{(i)} \mathbf{c}^T \mathbf{K}(\mathbf{x}^{(i)}))$$

$$f = \sum_{i=1}^{M} \mathbf{c}^T \mathbf{K}(\mathbf{x}^{(i)})$$

Discussions

- **Transductive support vector machines (semi-supervised learning):** The training and test sets are minimized together.

$$\text{Training set: } \mathcal{D} = \left\{\left(\mathbf{x}^{(i)}, y^{(i)}\right) \middle| \mathbf{x}^{(i)} \in \mathbb{R}^n, y^{(i)} \in \{-1,1\}\right\}_{i=1}^{M}$$

$$\text{Test set: } \mathcal{D}^* = \left\{\mathbf{x}^{(i)} \middle| \mathbf{x}^{(i)} \in \mathbb{R}^n \right\}_{i=1}^{N}$$

- **Manifold learning for semi-supervised learning:**

$$\arg\min_{f \in \mathcal{H}} L(\mathbf{c}) + \mathcal{R}(f),$$

$$\mathcal{R}(f) = \gamma_A \|f\|_{\mathcal{H}}^2 + \gamma_I \|f\|_I^2$$

$$\|f\|_I^2 = \frac{1}{(M+N)^2} \sum_{i,j=1}^{M+N} W_{ij} \left(f(\mathbf{x}_i) - f(\mathbf{x}_j)\right)$$

- This will be discussed further in future.