Bil 470 / YAP 470

Introduction to Machine Learning (Yapay Öğrenme)

Batuhan Bardak

SVM - Support Vector Machine

Date: 17.10.2022

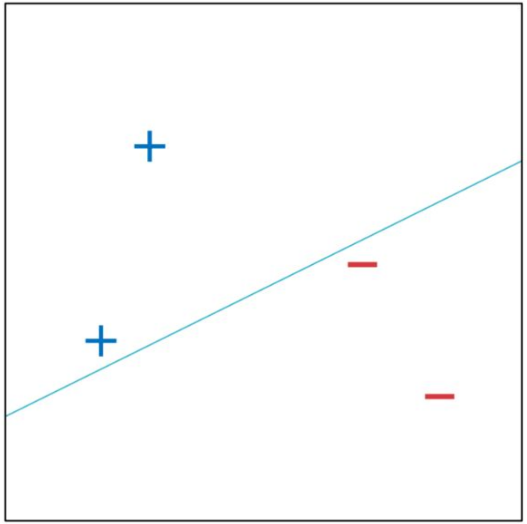
Plan for today

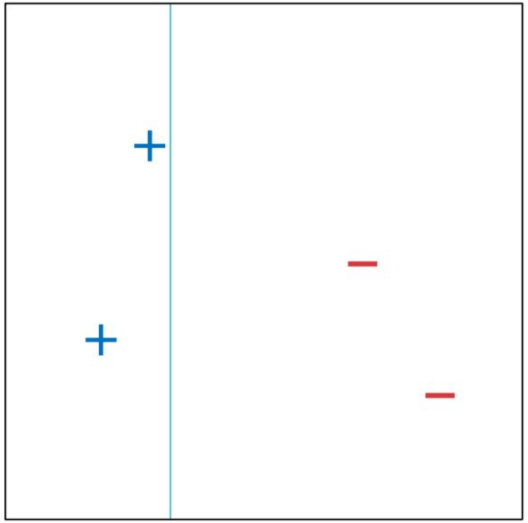
- SVM for linearly separable classes
- SVM for linearly inseparable classes
- SVM for nonlinear decision boundaries
    - Kernel functions

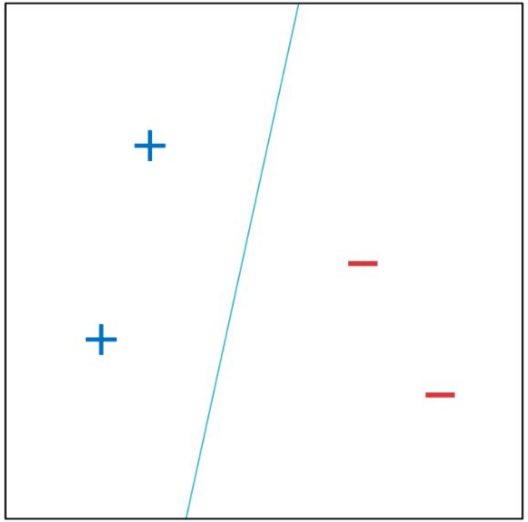
SVM

- The Support Vector Machine (SVM) is a linear classifier that can be viewed as an extension of the Perceptron.
- The Perceptron guaranteed that you find a hyperplane if it exists. The SVM finds the **maximum margin** separating hyperplane.

Which linear separator is best?

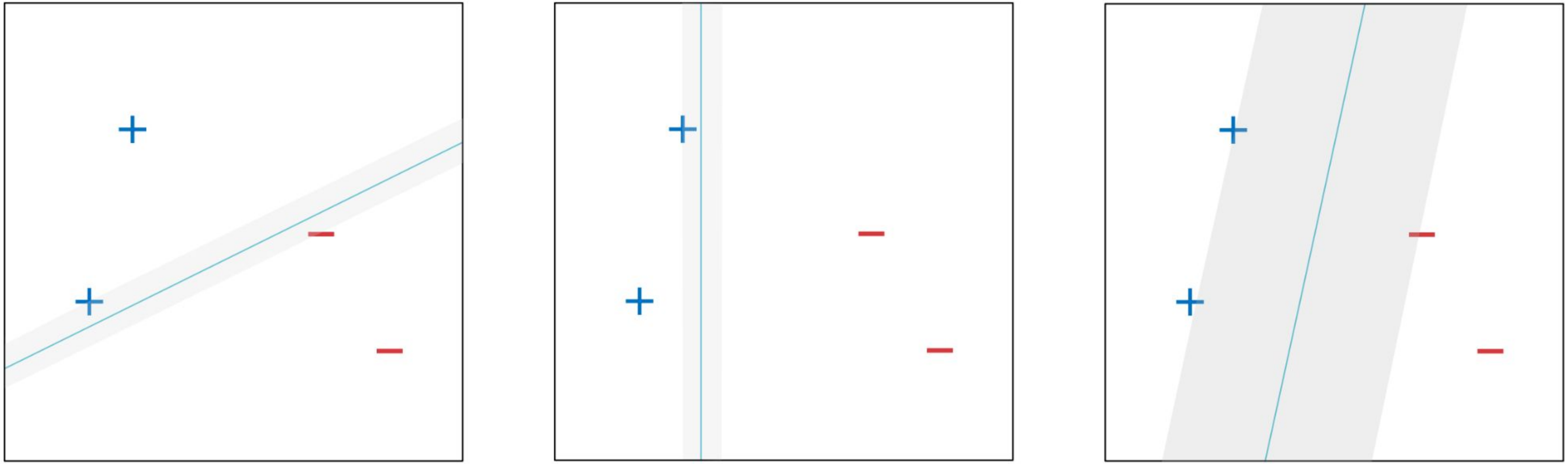






Slide Credit: Henry Chai

Which linear separator is best?



Slide Credit: Henry Chai

Maximal Margin Linear Separators

- The margin of a linear separator is the distance between it and the nearest training data point.
- Questions:
  - How can we efficiently find a maximal-margin linear separator?
  - Why are linear separators with larger margins better?
  - What can we do if the data is not linearly separable?

Slide Credit: Henry Chai

Hyperplanes

- For linear models, decision boundaries are D-dimensional ***hyperplanes*** defined by a weight vector, $[b, \boldsymbol{w}]$

$$\boldsymbol{w}^T \boldsymbol{x} + b = 0$$

- Problem: there are infinitely many weight vectors that describe the same hyperplane
  - $x_1 + 2x_2 + 2 = 0$ is the same line as $2x_1 + 4x_2 + 4 = 0$, which is the same line as $1000000x_1 + 2000000x_2 + 2000000 = 0$

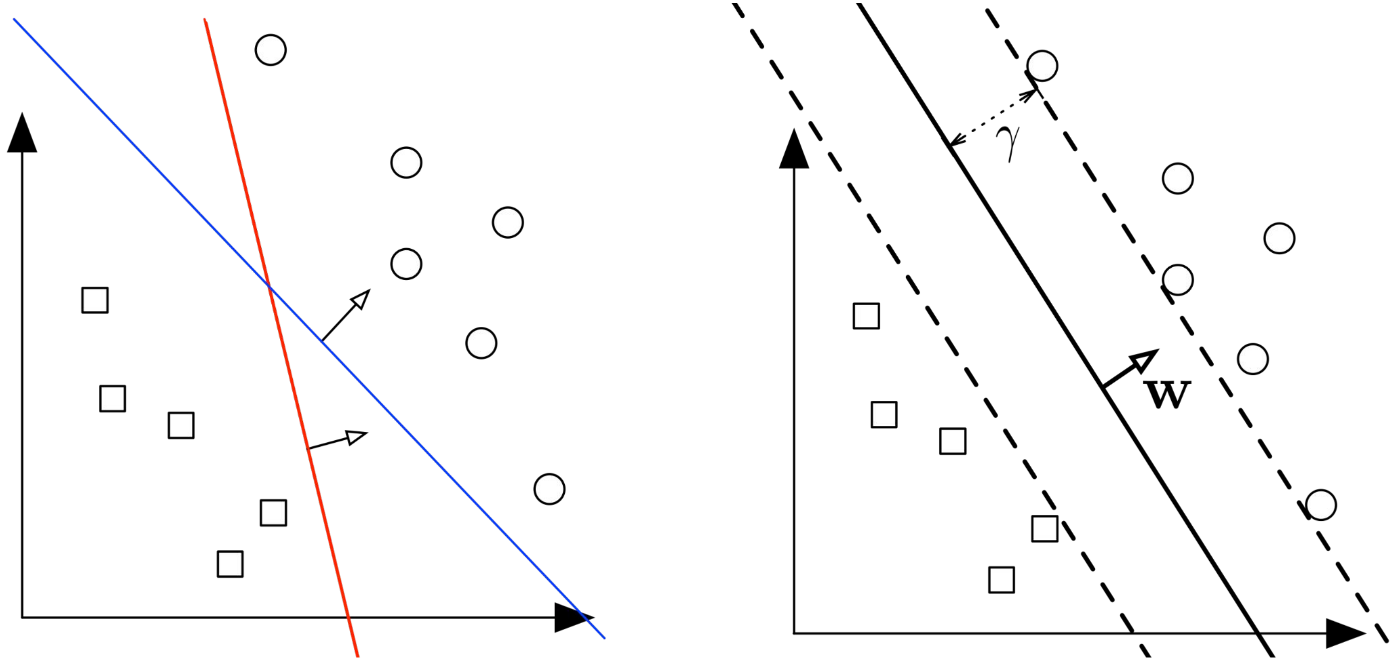
- Solution: normalize weight vectors *w.r.t. the training data*

Slide Credit: Henry Chai

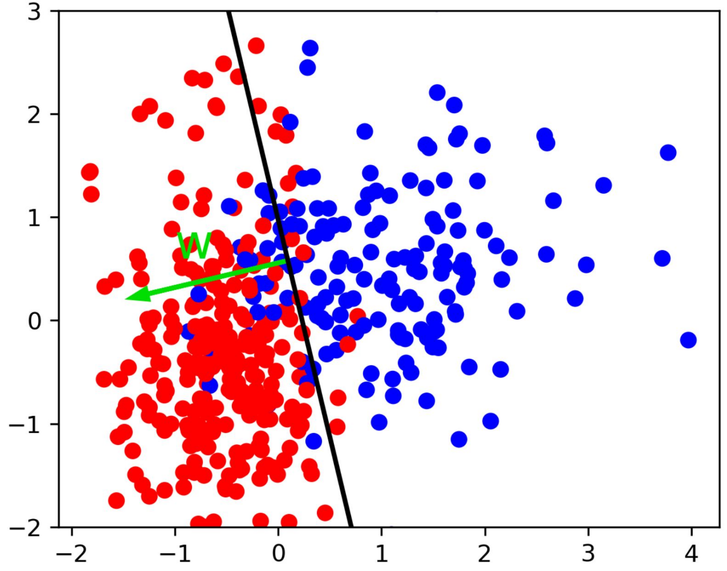
Maximal Margin Linear Separators

- The margin of a linear separator is the distance between it and the nearest training data point.
- Questions:
  - How can we efficiently find a maximal-margin linear separator?
  - Why are linear separators with larger margins better?
  - What can we do if the data is not linearly separable?

SVM



Linear models for **binary** classification



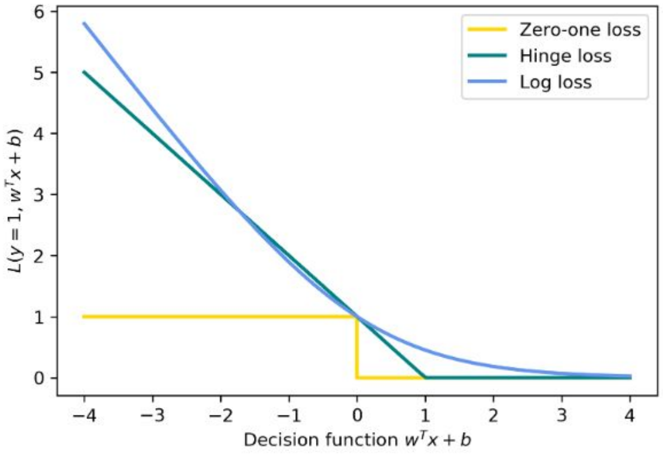
$$\hat{y} = \text{sign}(w^T \mathbf{x} + b) = \text{sign}\left(\sum_i w_i x_i + b\right)$$

Slide Credit: W4995, Andreas Müller

Picking a loss?

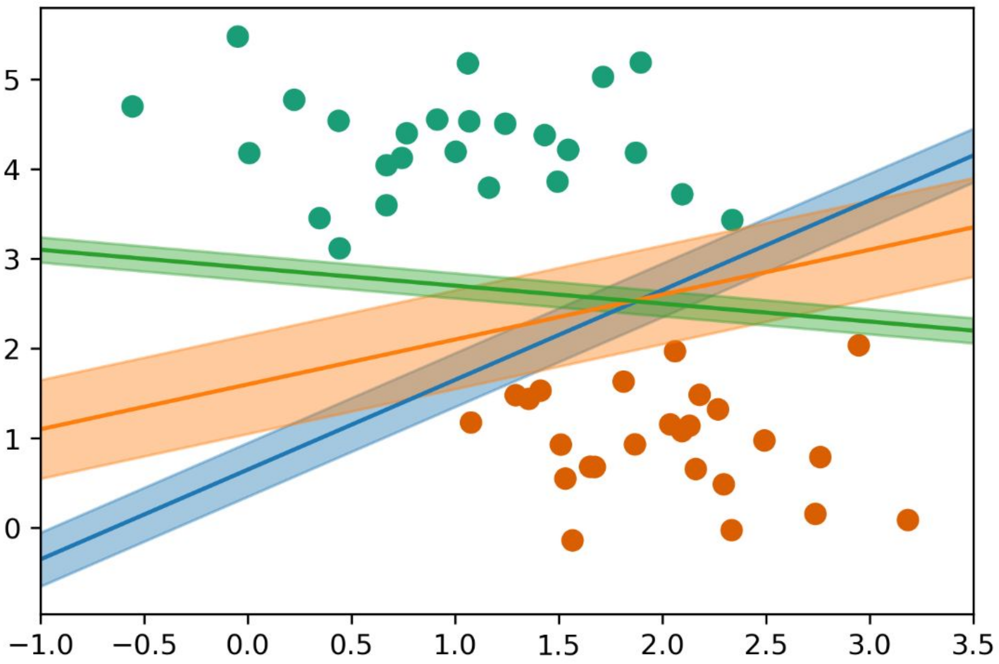
$$\hat{y} = \text{sign}(w^T\mathbf{x} + b)$$

$$\min_{w\in\mathbb{R}^p, b\in\mathbb{R}} \sum_{i=1}^{n} 1_{y_i \neq \text{sign}(w^T\mathbf{x}+b)}$$



Slide Credit: W4995, Andreas Müller

Max-Margin and Support Vectors



Slide Credit: W4995, Andreas Müller

Max-Margin and Support Vectors

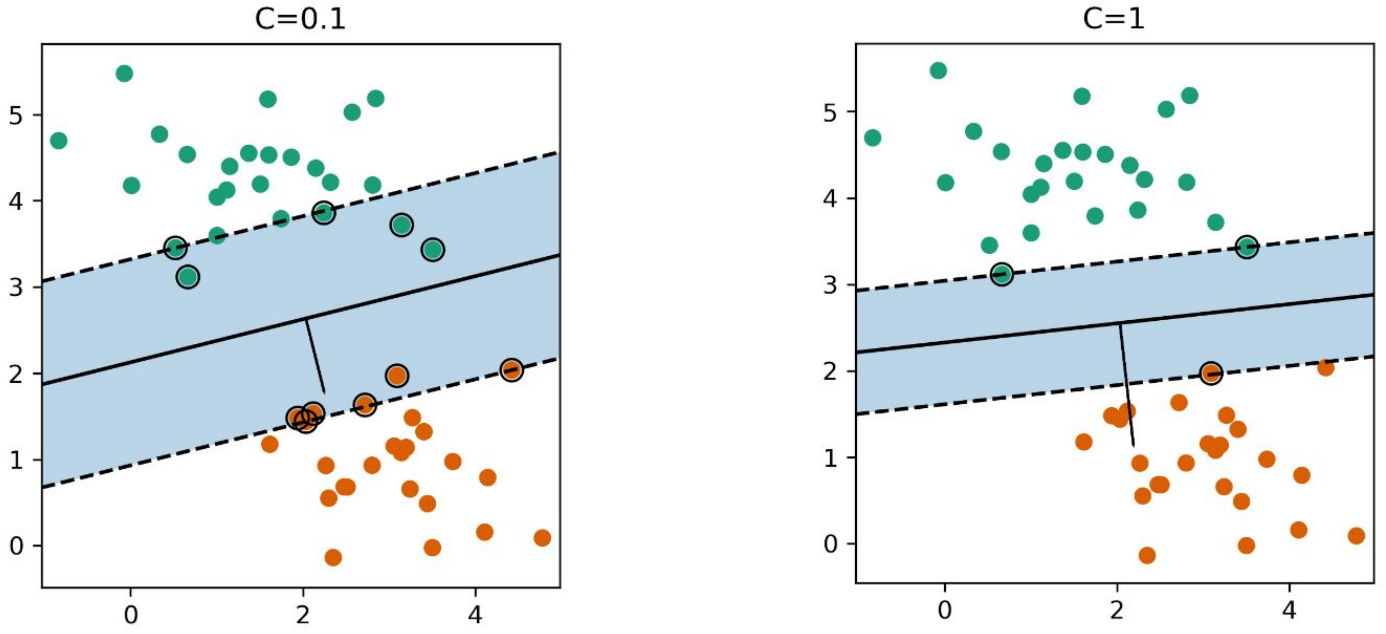
$$\min_{w \in \mathbb{R}^p, b \in \mathbb{R}} C \sum_{i=1}^{n} \max(0, 1 - y_i(w^T \mathbf{x} + b)) + ||w||_2^2$$

$$\text{Within margin} \Leftrightarrow y_i(w^T x + b) < 1$$

Smaller $w \Rightarrow$ larger margin

Slide Credit: W4995, Andreas Müller

Max-Margin and Support Vectors



Slide Credit: W4995, Andreas Müller

(soft margin) linear SVM

$$\min_{w \in \mathbb{R}^p, b \in \mathbb{R}} C \sum_{i=1}^{n} \max(0, 1 - y_i(w^T \mathbf{x}_i + b)) + ||w||_2^2$$

$$\min_{w \in \mathbb{R}^p, b \in \mathbb{R}} C \sum_{i=1}^{n} \max(0, 1 - y_i(w^T \mathbf{x}_i + b)) + ||w||_1$$

Slide Credit: W4995, Andreas Müller

Kernel SVMs

- Go from linear models to more powerful nonlinear ones.
- Keep convexity (ease of optimization).
- Generalize the concept of feature engineering.

Slide Credit: W4995, Andreas Müller

Linear SVM

$$\min_{w\in\mathbb{R}^p, b\in\mathbf{R}} C \sum_{i=1}^{n} \max(0, 1 - y_i(w^T\mathbf{x} + b)) + ||w||_2^2$$

$$\hat{y} = \text{sign}(w^T\mathbf{x} + b)$$

Reformulate Linear Models

- Optimization Theory

$$w = \sum_{i=1}^{n} \alpha_i \mathbf{x}_i$$

(alpha are dual coefficients. Non-zero for support vectors only)

$$\hat{y} = \text{sign}(w^T \mathbf{x}) \Longrightarrow \hat{y} = \text{sign}\left(\sum_{i}^{n} \alpha_i (\mathbf{x}_i^T \mathbf{x})\right)$$

$$\alpha_i <= C$$

Introducing Kernels

$$\hat{y} = \text{sign}\left(\sum_{i}^{n} \alpha_i (\mathbf{x}_i^T \mathbf{x})\right) \longrightarrow \hat{y} = \text{sign}\left(\sum_{i}^{n} \alpha_i (\phi(\mathbf{x}_i)^T \phi(\mathbf{x}))\right)$$

$$\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j) \longrightarrow k(\mathbf{x}_i, \mathbf{x}_j)$$

k positive definite, symmetric $\Rightarrow$ there exists a ϕ! (possilby ∞-dim)

Example of Kernels

$$k_{\text{linear}}(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T \mathbf{x}'$$

$$k_{\text{poly}}(\mathbf{x}, \mathbf{x}') = (\mathbf{x}^T \mathbf{x}' + c)^d$$

$$k_{\text{rbf}}(\mathbf{x}, \mathbf{x}') = \exp(-\gamma ||\mathbf{x} - \mathbf{x}'||^2)$$

$$k_{\text{sigmoid}}(\mathbf{x}, \mathbf{x}') = \tanh\left(\gamma \mathbf{x}^T \mathbf{x}' + r\right)$$

$$k_{\cap}(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^{p} \min(x_i, x'_i)$$

- If k and k' are kernels, so are $k + k', kk', ck', \ldots$

Polynomial Kernel vs Features

$$k_{\text{poly}}(\mathbf{x}, \mathbf{x}') = (\mathbf{x}^T \mathbf{x}' + c)^d$$

Primal vs Dual Optimization

Explicit polynomials $\rightarrow$ compute on `n_samples * n_features ** d`

Kernel trick $\rightarrow$ compute on kernel matrix of shape `n_samples * n_samples`

For a single feature:

$$(x^2, \sqrt{2}x, 1)^T (x'^2, \sqrt{2}x', 1) = x^2 x'^2 + 2xx' + 1 = (xx' + 1)^2$$

Kernels in Practice

- Dual coefficients less interpretable
- Long runtime for “large” datasets (100k samples)
- Real power in infinite-dimensional spaces: rbf!
- Rbf is “universal kernel” - can learn (aka overfit) anything.

Slide Credit: W4995, Andreas Müller

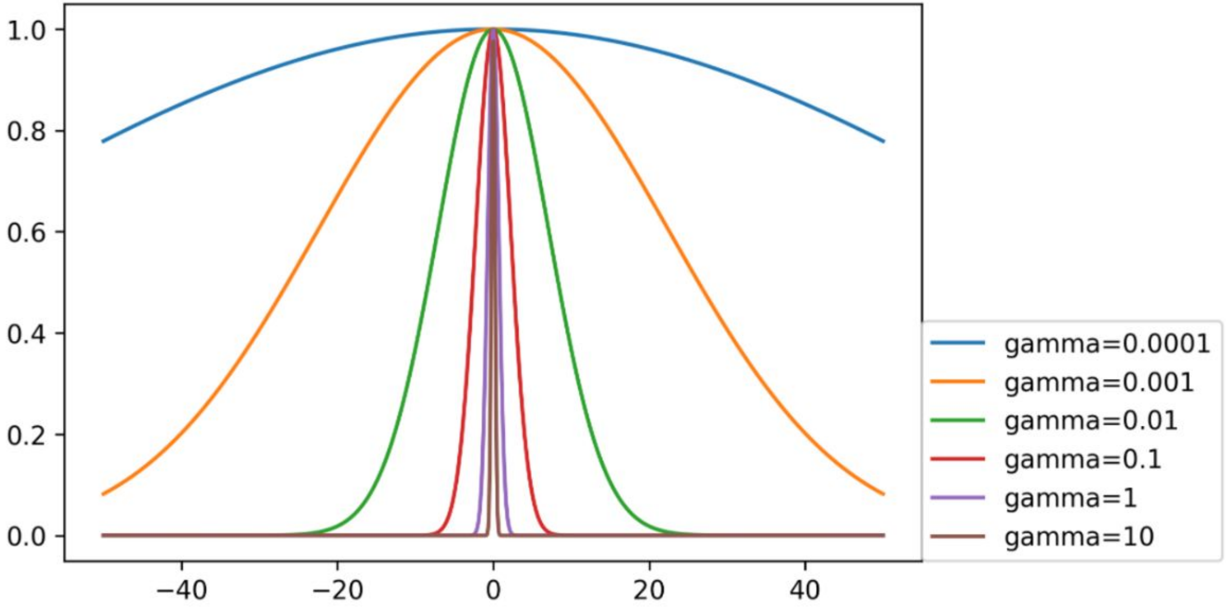
Preprocessing

- Kernel use inner products or distances.
- StandardScaler or MinMaxScaler ftw
- Gamma parameter in RBF directly relates to scaling of data and n_features – the default is `1/(X.var() * n_features)`

Slide Credit: W4995, Andreas Müller

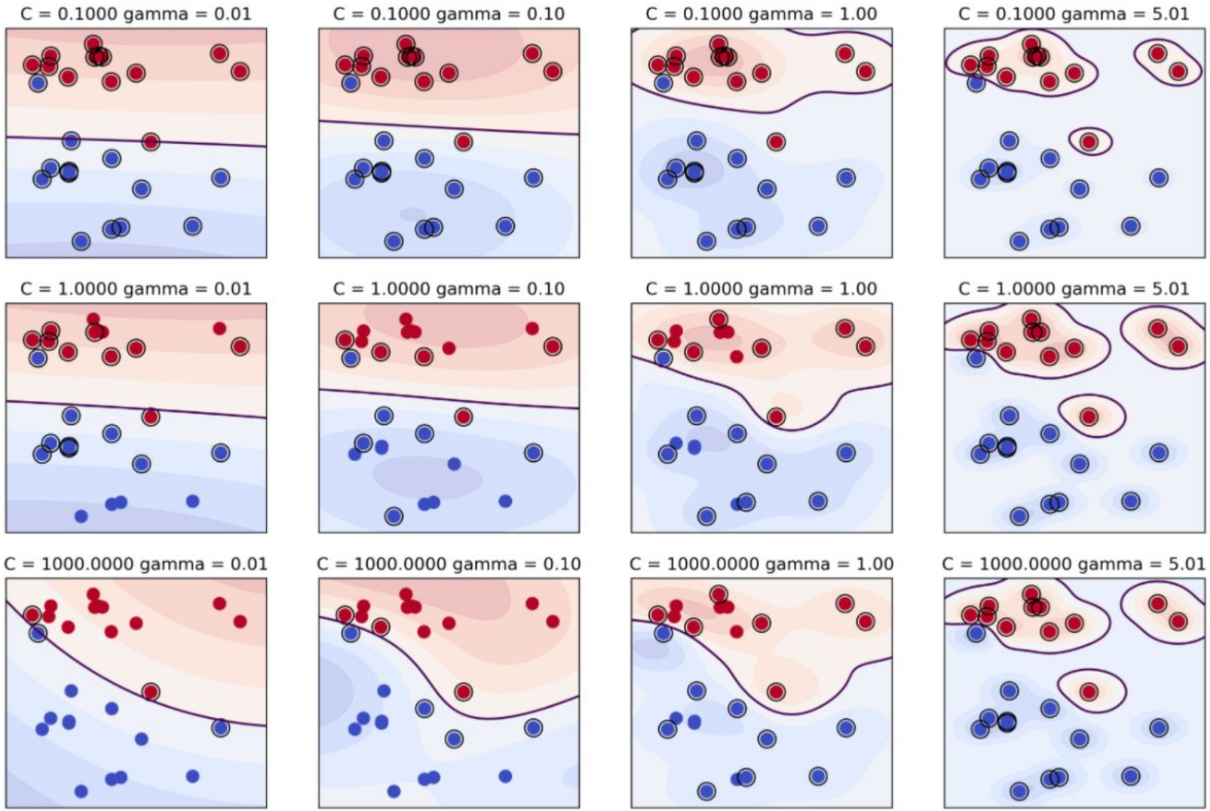
Parameters for RBF Kernels

- Regularization parameter C is limit on alphas (for any kernel)
- Gamma is bandwidth: $k_{\mathrm{rbf}}(\mathbf{x}, \mathbf{x}') = \exp(-\gamma ||\mathbf{x} - \mathbf{x}'||^2)$



Slide Cre

Parameters for RBF Kernels



Slide Credit: W4995, A

Next Class:

Decision Trees & Ensemble Learning