Nonlinear Support Vector Machine

COMP 4211 - Tutorial 10

Chun-Kit Yeung

Hong Kong University of Science and Technology

2018-04-27

Objective

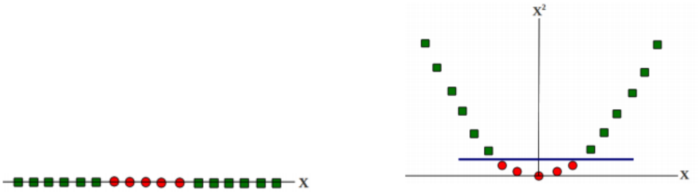
In this tutorial, we will review the nonlinear SVM and apply several kernel functions in scikit-learn.

Why we need kernel?

Kernels: Make linear models work in nonlinear settings

1. by mapping data to a higher dimensions as the new input space where it exhibits linear patterns; and then
2. apply the linear model in the new input space.

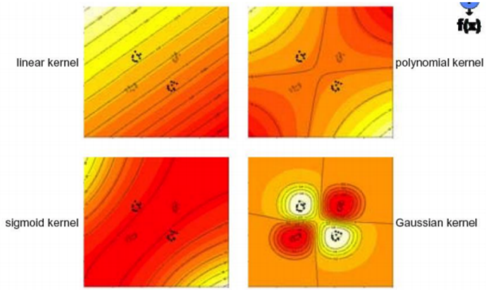
For example, consider this binary classification problem



What are the options for kernel?

Examples of Kernels

- inhomogeneous polynomial: $k(\mathbf{x}, \mathbf{y}) = (\mathbf{x}'\mathbf{y} + 1)^d$
- Gaussian: $k(\mathbf{x}, \mathbf{y}) = \exp(-\|\mathbf{x} - \mathbf{y}\|^2/(2\sigma^2))$
  - radial basis function (RBF) network
  - corresponds to an infinite-dimensional feature space
- sigmoid: $k(\mathbf{x}, \mathbf{y}) = \tanh(\kappa(\mathbf{x}'\mathbf{y}) + \theta)$
  - a valid kernel only for certain κ and θ



What are the options for kernel?

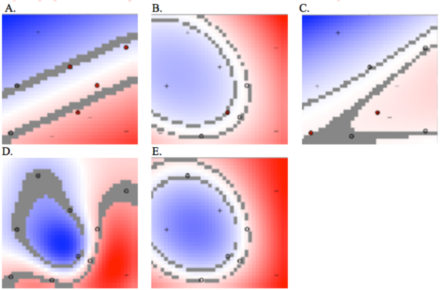
There are several popular kernel that have introduced in the lecture:

- Polynomial: $K(\mathbf{a}, \mathbf{b}) = (\gamma \cdot \mathbf{a}^T \mathbf{b} + r)^d$
- Gaussian: $K(\mathbf{a}, \mathbf{b}) = \exp(\frac{\|\mathbf{a}-\mathbf{b}\|^2}{2\sigma^2})$, or it could be simply expressed as $K(\mathbf{a}, \mathbf{b}) = \exp(\gamma \cdot \|\mathbf{a} - \mathbf{b}\|^2)$
- Sigmoid: $K(\mathbf{a}, \mathbf{b}) = \tanh(\gamma \cdot \mathbf{a}^T \mathbf{b} + r)$

where d, γ and r is the kernel parameters.

Question

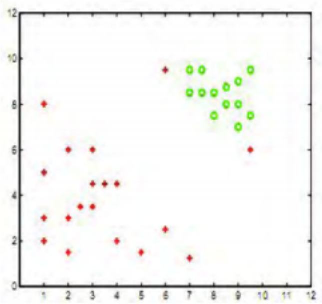
The following diagrams represent graphs of support vector machines trained to separate pluses (+) from minuses (-) for the same data set.



Match the diagrams with the following kernels:
(1) RBF with $\sigma = 0.08$, (2) RBF with $\sigma = 0.5$, (3) RBF with $\sigma = 2.0$, (4) linear, (5) second order polynomial.

Question

For this question assume that we are training an SVM with a quadratic kernel - i.e., our kernel function is a polynomial kernel of degree 2. This means the resulting decision boundary in the original feature space may be parabolic in nature. The dataset on which we are training is given below:



1. Where would the decision boundary be for very large values of C?
2. Where would the decision boundary be for C nearly equal to 0?

Let's code

To better understand today tutorial, the following .ipynb is covered:

- T10_Linear_SVM.ipynb