Support Vector Machines

Ryan Miller

Grinnell College
Statistics

Introduction

- Consider a binary classification task
  - Support Vector Machines (SVM) try to find a plane that separates the two classes in the space of our predictive features
- If no such plane exists, there are two possible solutions
  - Relaxing what we mean by "separate"
  - Expanding our feature space to facilitate separation

Hyperplanes

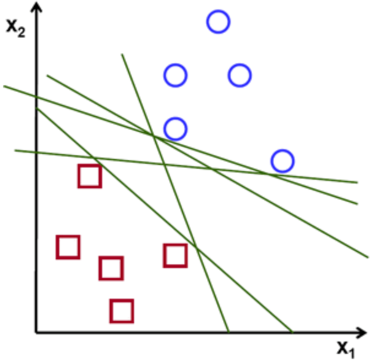
- A hyperplane is defined by a linear combination:

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p$$

- Recognize that *multivariable linear regression* involves a hyperplane
  - Its hyperplane represents the expected value of a continuous outcome, Y, estimated via least squares for a set of predictors
- Support vector machines seek a *separating hyperplane*

Separating Hyperplanes

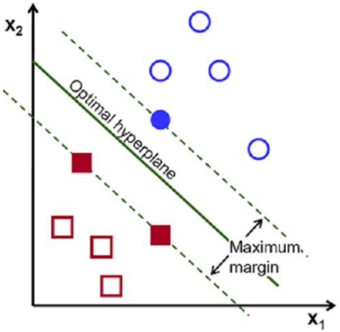
Consider two features, X_1 and X_2, and a binary outcome. It might be possible to draw several separating hyperplanes:



Which of these hyperplanes is the better classifier?

The Maximum Margin Classifier

A *hard margin* SVM finds the "maximum margin" hyperplane:



This plane represents the "widest street" between classes, and it is characterized by "support vectors", or training data-points that would change this hyperplane if removed

The Maximum Margin Classifier

- Consider the constraint: $\sum_{j=1}^{p} \beta_j^2 = 1$, which normalizes how our hyperplane is defined
  - This won't impact the direction of the plane, as $\{\beta_1 = 1, \beta_2 = 1\}$ and $\{\beta_1 = 3, \beta_2 = 3\}$ have the same orientation
- The SVM is defined by $\beta_1, \ldots, \beta_p$ which maximize M in the expression: $y_i(\beta_0 + \beta_1 x_{i,1} + \beta_2 x_{i,2} + \beta_p x_{i,p}) \geq M \;\; \forall i = 1, \ldots, n$
  - Here the binary outcome, y_i, is encoded as $+1$ or -1, so the left side of this expression is the distance from the current hyperplane to the i^{th} data-point

The Maximum Margin Classifier

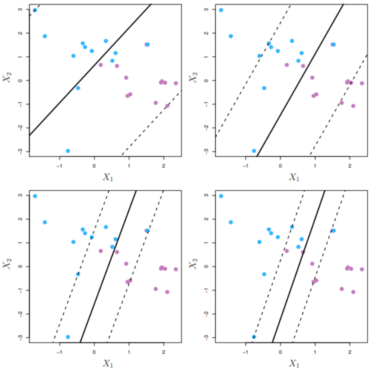
- The coefficients defining the SVM classifier can be found using the Lagrangian multiplier method
  - We will not cover this method in this course (as SVMs are the only classifier we'll study that use it)
- If you're interested in the mathematical details, I recommend Robert Berwick's (of MIT) "An Idiot's guide to support vector machines"

Soft Margin Classifiers

- For non-separable data, we can relax the maximum margin approach to find a *soft margin classifier*:
- Now we aim to find $\beta_1, \dots, \beta_p$ that maximize M where $y_i(\beta_0 + \beta_1 x_{i,1} + \beta_2 x_{i,2} + \beta_p x_{i,p}) \geq M(1 - \epsilon_i)$
  - Subject to $\epsilon \geq 0$ and $\sum_{i=1}^{n} \epsilon_i < s$
    - $\epsilon_i = 0$ if a point is on the correct side of the margin
    - $0 < \epsilon_i < 1$ if a point is within the margin
    - $\epsilon_i > 1$ if a point is on the wrong side of the margin
  - s is controls the total amount of "slack" that is allowed, with larger values allowing for more "slack"
    - As s decreases the tolerance for data-points being on the wrong side of the hyperplane diminishes

Soft Margin Classifiers

As s decreases (left to right), the margin M decreases:

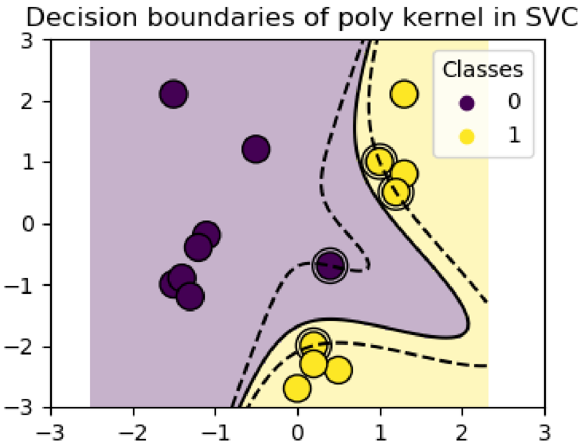


A larger s yields a more stable classifier, so the bias-variance trade-off can be manipulated via the value of s.

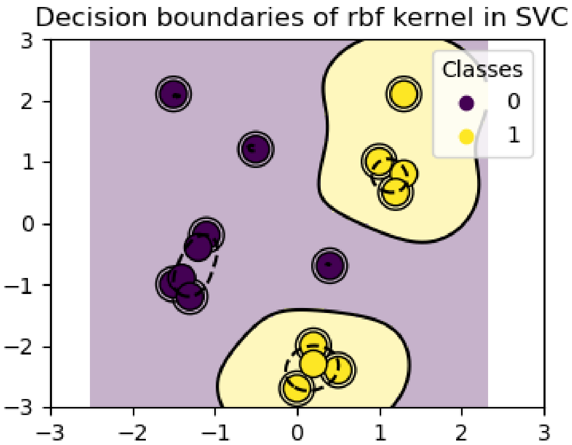
Feature Expansion and Kernel Functions

- Consider the features: $\{X_1, X_2\}$, and recall that the SVM classifier finds a decision boundary (separating hyperplane) of the form $\beta_0 + \beta_1 X_1 + \beta_2 X_2$
- We could apply transformations to create a new set of features: $\{X_1, X_2, X_1^2, X_2^2, X_1 X_2\}$
  - Now the decision boundary would have the form: $\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1^2 + \beta_4 X_2^2 + \beta_5 X_1 X_2$
- This corresponds to a *non-linear boundary* in the *original feature space*
  - *Kernel functions* allow for computationally efficient mappings of the original features to higher dimensions for the purpose of finding a non-linear decision boundary

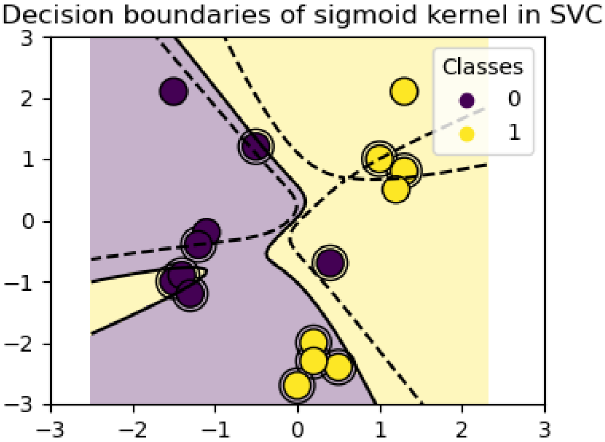
Polynomial Kernel ($d = 3$, $\gamma = 2$)



Radial Basis Function Kernel



Sigmoid Kernel



Practical guidance

- SVMs treat each feature equally, so standardization is an important data preparation step
- The kernel function (type of feature expansion) and "slack" parameter can be tuned via cross-validation to achieve optimal classification performance
  - `sklearn` represents "slack" using a parameter `C` that is inversely proportional to what we called s
  - Other hyperparameters affiliated with certain kernel functions, such as `\gamma` can also be tuned in this manner
- Support vector regression is also implemented in `sklearn`, the SVM lab will briefly cover this method
  - SVMs also have been generalized to multi-class tasks, and use a one-vs-one scheme in `sklearn`