PQHS 471 Lecture 10: Support Vector Machines

Support Vector Machines



- Dr. Vladimir Vapnik (1936 present).
- Moved from USSR to USA in 1990, and worked at AT&T Bell Lab.
- Inventor of the Support Vector Machines.

- Here we approach the two-class classification problem in a direct way:
 - We try and find a plane that separates the classes in feature space.

- Here we approach the two-class classification problem in a direct way:
 - We try and find a plane that separates the classes in feature space.
- If we cannot, we get creative in two ways:
 - We soften what we mean by "separates", and
 - 2 We enrich and enlarge the feature space so that separation is possible.

- A hyperplane in p dimensions is a flat affine subspace of dimension p-1.
- In general the equation for a hyperplane has the form

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_p X_p = 0$$

- In p = 2 dimensions a hyperplane is a line.
- If $\beta_0 = 0$, the hyperplane goes through the origin, otherwise not.
- The vector $\beta = (\beta_1, \beta_2, ..., \beta_p)$ is called the normal vector it points in a direction orthogonal to the surface of a hyperplane.

イロン イヨン イヨン

Hyperplane example in 2D



æ

Separating Hyperplanes



- If $f(X) = \beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p$, then f(X) > 0 for points on one side of the hyperplane, and f(X) < 0 for points on the other.
- If we code the colored points as $Y_i = +1$ for blue, say, and $Y_i = -1$ for mauve, then if $Y_i \cdot f(X_i) > 0$ for all i, f(X) = 0 defines a separating hyperplane.

Among all separating hyperplanes, find the one that makes the biggest gap or margin between the two classes.



Constrained optimization problem

 $\underset{\beta_0,\beta_1,\ldots,\beta_p}{\operatorname{maximize}} M$

subject to
$$\sum_{j=1}^{p} \beta_j^2 = 1,$$
$$y_i(\beta_0 + \beta_1 x_{i1} + \ldots + \beta_p x_{ip}) \ge M$$
for all $i = 1, \ldots, N.$

Non-separable data



The data on the left are not separable by a linear boundary.

This is often the case, unless N < p.



- Sometimes the data are separable, but noisy. This can lead to a poor solution for the maximal-margin classifier.
- The support vector classifier maximizes a soft margin.

Support Vector Classifier



$$\begin{array}{l} \underset{\beta_{0},\beta_{1},\ldots,\beta_{p},\epsilon_{1},\ldots,\epsilon_{n}}{\operatorname{maximize}} M \quad \text{subject to} \quad \sum_{j=1}^{p} \beta_{j}^{2} = 1, \\ y_{i}(\beta_{0} + \beta_{1}x_{i1} + \beta_{2}x_{i2} + \ldots + \beta_{p}x_{ip}) \geq M(1 - \epsilon_{i}), \\ \epsilon_{i} \geq 0, \quad \sum_{i=1}^{n} \epsilon_{i} \leq C, \end{array}$$

• C: a budget

æ

- When the data are not perfectly separable in the feature space, we may allow some observations to be on the "wrong" side of the margin.
- We give "allowances" to the observations but control the total "budget".
- A large C: the margin is large and many observations can be support vectors.

Budget C: regularization parameter



Feature Expansion and Kernels

Linear boundary can fail



Sometime a linear boundary simply won't work, no matter what value of C.

The example on the left is such a case.

What to do?

Feature Expansion

- Enlarge the space of features by including transformations; e.g. $X_1^2, X_1^3, X_1X_2, X_1X_2^2, ...$ Hence go from a *p*-dimensional space to a M > p dimensional space.
- Fit a support-vector classifier in the enlarged space.
- This results in non-linear decision boundaries in the original space.

Example: Suppose we use $(X_1, X_2, X_1^2, X_2^2, X_1X_2)$ instead of just (X_1, X_2) . Then the decision boundary would be of the form

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1^2 + \beta_4 X_2^2 + \beta_5 X_1 X_2 = 0$$

This leads to nonlinear decision boundaries in the original space (quadratic sections).

Cubic polynomials example

- Here we use a basis expansion of cubic polynomials
- From 2 variables to 9 variables
- The support-vector classifier in the enlarged space solves the problem in the lower-dimensional space



 $\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1^2 + \beta_4 X_2^2 + \beta_5 X_1 X_2 + \beta_6 X_1^3 + \beta_7 X_2^3 + \beta_8 X_1 X_2^2 + \beta_9 X_1^2 X_2 = 0$

When domain knowledge is available, sometimes we could use explicit transformations. But often we cannot.



- 2D classification.
- Separable (and linear!) in features spaces of x_1^2, x_2^2, x_1x_2
- The linear hyperplane \rightarrow nonlinear ellipsoidal decision boundary in the original space.

- Polynomials (especially high-dimensional ones) get wild rather fast.
- There is a more elegant and controlled way to introduce nonlinearities in support-vector classifiers through the use of kernels.
- Before we discuss these, we must understand the role of inner products in support-vector classifiers.

Inner products and support vectors

Inner product between two vectors:

$$\langle \mathbf{x}_i, \mathbf{x}_j \rangle = \sum_{k=1}^p x_{ik} x_{jk}$$

• The linear support vector classifier can be represented as (n parameters):

$$f(x) = \beta_0 + \sum_{i=1}^n \alpha_i \langle \mathbf{x}, \mathbf{x}_i \rangle$$

To estimate the parameters α₁, ..., α_n and β₀, all we need are the ⁿ₂ inner products (x, x_i) between all pairs of training observations.
It turns out that most of the â_i can be zero:

$$f(x) = \beta_0 + \sum_{i \in \mathcal{S}} \hat{\alpha}_i \langle \mathbf{x}, \mathbf{x}_i \rangle$$

 ${\cal S}$ is the support set of indices i such that $\hat{lpha}_i > 0$. See slides 10.

Kernels and Support Vector Machines

- If we can compute inner-products between observations, we can fit a SV classifier. Can be quite abstract!
- Some special kernel functions can do this for us. E.g.

$$K(\mathbf{x}_i, \mathbf{x}_j) = (1 + \sum_{k=1}^p x_{ik} x_{jk})^d$$

can computes the inner-products needed for d dimensional polynomials — $\binom{p+d}{d}$ basis functions!

• The solution has the form:

$$f(x) = \beta_0 + \sum_{i \in \mathcal{S}} \hat{\alpha}_i K(\mathbf{x}, \mathbf{x}_i)$$

• h(x) is involved ONLY in the form of inner product! So, as long as we define the kernel function

$$K(\mathbf{x}_i, \mathbf{x}_j) = \langle h(\mathbf{x}_i), h(\mathbf{x}_j) \rangle$$

which computes the inner product in the transformed space, we don't need to know what h(x) itself is! (Kernel trick)

• Some commonly used Kernels:

dth-Degree polynomial: $K(x, x') = (1 + \langle x, x' \rangle)^d$, Radial basis: $K(x, x') = \exp(-\gamma ||x - x'||^2)$, Neural network: $K(x, x') = \tanh(\kappa_1 \langle x, x' \rangle + \kappa_2)$

Radial Kernel



- Implicit feature space; very high dimensional.
- Controls variance by squashing down most dimensions severely.

SVM - Degree-4 Polynomial in Feature Space



- $\bullet\,$ Radial basis kernel with $\gamma=1$
- C was tuned and picked = 1
- Radial kernel performs the best here (close to Bayes optimal), as might be expected give the data arise from mixtures of Gaussians.

• The function svm() in package e1071 provides svm solutions efficiently.

Example: Heart Data



・ロト・日本・日本・日本・日本・今日・

Example: Heart Data



can we make conclusion using this ROC? NO.



・ロト・日本・モト・モー シック



can we make conclusions? Yes we are ready now.

The SVM as defined works for K=2 classes. What do we do if we have K>2 classes?

• **OVA**:

One versus All. Fit K different 2-class SVM classifiers $\hat{f}_k(x), k = 1, ..., K$; each class versus the rest. Classify x^* to the class for which $\hat{f}_k(x^*)$ is largest.

• **OVO**:

One versus One. Fit all $\binom{K}{2}$ pairwise classifiers. Classify x^* to the class that wins the most pairwise competitions.

Which to choose? If K is not too large, use OVO.

How to select kernel and parameters?

- Domain knowledge:
 - How complex should the space partition be?
 - Should the surface be smooth?
- Compare the models by their approximate testing error rate cross-validation:
 - Fit data using multiple kernels/parameters
 - Estimate error rate for each setting
 - Select the best-performing one

Strengths of SVM:

- flexibility
- scales well for high-dimensional data
- can control complexity and error trade-off explicitly
- as long as a kernel can be defined, non-traditional(vector) data, like strings, trees can be input

Weakness:

• how to choose a good kernel?

(a low degree polynomial or radial basis function can be a good start)

• ISLR: chapter 9: 9.1 - 9.4