Support vector machines

-Statistical Machine Learning-

Lecturer: Darren Homrighausen, PhD

Optimal separating hyperplanes

A main initiative in early computer science was to find separating hyperplanes among groups of data

(Rosenblatt (1958) with the perceptron algorithm)

The issue is that if there is a separating hyperplane, there is an infinite number

An optimal separating hyperplane can be generated by finding support points and bisecting them.

(Sometimes optimal separating hyperplanes are called maximum margin classifiers)

Basic linear geometry

A hyperplane in $\mathbb{R}^p$ is given by

$$\mathcal{H} = \{X \in \mathbb{R}^p : h(X) = \beta_0 + \beta^\top X = 0\}$$

(Usually it is assumed that $||\beta||_2 = 1$)

1. The vector β is normal to $\mathcal{H}$

   (To see this, let $X, X' \in \mathcal{H}$. Then $\beta^\top(X - X') = 0$)

2. Important: For any point $X \in \mathbb{R}^p$, the (signed) length of its orthogonal complement to $\mathcal{H}$ is $h(X)$

Support vector machines (SVM)

Let $Y_i \in \{-1, 1\}$

(It is common with SVMs to code Y this way. With logistic regression, Y is commonly phrased as $\{0, 1\}$ due to the connection with Bernoulli trials)

We will generalize this to supervisors with more than 2 levels at the end

A classification rule induced by a hyperplane is

$$g(X) = \mathrm{sgn}(X^\top \beta + \beta_0)$$

Separating hyperplanes

Our classification rule is based on a hyperplane $\mathcal{H}$

$$g(X) = \mathrm{sgn}(X^\top \beta + \beta_0)$$

A correct classification is one such that $h(X)Y > 0$ and $g(X)Y > 0$

(Why?)

The larger the quantity $Yh(X)$, the more "sure" the classification

(Reminder: The signed distance to $\mathcal{H}$ is $h(X)$)

Under classical separability, we can find a function such that $Y_i h(X_i) > 0$

(That is, makes perfect training classifications via g)

Optimal separating hyperplane

This idea can be encoded in the following convex program

$$\max_{\beta_0,\beta} M \text{ subject to}$$

$$Y_i h(X_i) \geq M \text{ for each } i \text{ and } ||\beta||_2 = 1$$

Intuition:

- We know that $Y_i h(X_i) > 0 \Rightarrow g(X_i) = Y_i$. Hence, larger $Y_i h(X_i) \Rightarrow$ "more" correct classification
- For "more" to have any meaning, we need to normalize β, thus the other constraint

Optimal separating hyperplane

Let's take the original program:

$$\max_{\beta_0, \beta} M \text{ subject to}$$

$$Y_i h(X_i) \geq M \text{ for each } i \text{ and } ||\beta||_2 = 1$$

and rewrite it as

$$\min_{\beta_0, \beta} \frac{1}{2} ||\beta||_2^2 \text{ subject to}$$

$$Y_i h(X_i) \geq 1 \text{ for each } i$$

(Replace $Y_i h(X_i) \geq M$ with $\frac{1}{||\beta||_2} Y_i h(X_i) \geq M$, which redefines β_0)

This is still a convex optimization program: quadratic criterion, linear inequality constraints

Optimal separating hyperplane

Again, we can convert this constrained optimization problem into the Lagrangian (primal) form

$$\min_{\beta_0,\beta} \frac{1}{2}||\beta||_2^2 - \sum_{i=1}^n \alpha_i [Y_i(X_i^\top \beta + \beta_0) - 1]$$

In contrast to the lasso problem, there are now n Lagrangian parameters $\alpha_1, \ldots, \alpha_n$

(There are n constraints, after all)

Everything is nice and smooth, so we can take derivatives..

Optimal separating hyperplane

$$\frac{1}{2}||\beta||_2^2 - \sum_{i=1}^{n} \alpha_i [Y_i(X_i^\top \beta + \beta_0) - 1]$$

Derivatives with respect to β and β_0:

- $\beta = \sum_{i=1}^{n} \alpha_i Y_i X_i$
- $0 = \sum_{i=1}^{n} \alpha_i Y_i$

Substituting into the Lagrangian:

$$\text{wolfe dual} = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{k=1}^{n} \alpha_i \alpha_k Y_i Y_k X_i^\top X_k$$

(this is all subject to $\alpha_i \geq 0$)

We want to maximize wolfe dual

Optimal separating hyperplane

A side condition, known as complementary slackness states[1]:

$$\alpha_i[1 - Y_i h(X_i)] = 0 \text{ for all } i$$

(The product of Lagrangian parameters and inequalty constraint equals 0)

This implies either:

- $\alpha_i = 0$, which happens if the constraint $Y_i h(X_i) > 1$

  (That is, when the constraint is non binding)
- $\alpha_i > 0$, which happens if the constraint $Y_i h(X_i) = 1$

  (That is, when the constraint is binding)

[1] See the Karush-Kuhn-Tucker (KKT) conditions

Optimal separating hyperplane

Taking this relationship

$$\alpha_i [Y_i h(X_i) - 1] = 0$$

we see that, for $i = 1, \ldots, n$,

- The points (X_i, Y_i) such that $\alpha_i > 0$ are support vectors
- The points (X_i, Y_i) such that $\alpha_i = 0$ are irrelevant for classification

(Why?)

End result: $\hat{g}(X) = \mathrm{sgn}(X^\top \hat{\beta} + \hat{\beta}_0)$

Support vector classifier

Support vector classifier

Of course, we can't realistically assume that the data are linearly separated (even in a transformed space)

In this case, the previous program has no feasible solution

We need to introduce slack variables, ξ, that allow for overlap among the classes

These slack variables allow for us to encode training missclassifications into the optimization problem

Support vector classifier

$$\max_{\beta_0, \beta, \xi_1, \ldots, \xi_n} M \text{ subject to}$$

$$Y_i h(X_i) \geq M \underbrace{(1 - \xi_i), \xi_i \geq 0, \sum \xi_i \leq t}_{new}, \text{ for each } i$$

Note that

- t is a tuning parameter. The literature usually refers to t as a budget

  (Think: lasso)

- The separable case corresponds to $t = 0$

Support vector classifier

We can rewrite the problem again:

$$\min_{\beta_0,\beta,\xi} \frac{1}{2} ||\beta||_2^2 \text{ subject to}$$

$$Y_i h(X_i) \geq 1 \underbrace{-\xi_i, \xi_i \geq 0, \sum \xi_i \leq t}_{new}, \text{ for each } i$$

(Convex optimization program: quadratic criterion, linear inequality constraints.)

Converting $\sum \xi_i \leq t$ to the Lagrangian (primal):

$$\min_{\beta_0,\beta} \frac{1}{2} ||\beta||_2^2 + \lambda \sum \xi_i \text{ subject to}$$

$$Y_i h(X_i) \geq 1 - \xi_i, \xi_i \geq 0, \text{ for each } i$$

(Think: lasso. $\lambda \sum \xi_i + \xi_i \geq 0 \Rightarrow \lambda ||\xi||_1$)

SVMs: slack variables

The slack variables give us insight into the problem

- If $\xi_i = 0$, then that observation is on correct the side of the margin
- If $\xi_i = \in (0, 1]$, then that observation is on the incorrect side of the margin, but still correctly classified
- If $\xi_i > 1$, then that observation is incorrectly classified

Support vector classifier

Continuing to convert constraints to Lagrangian

$$\min_{\beta_0,\beta,\xi} \frac{1}{2}||\beta||_2^2 + \lambda \sum \xi_i \underbrace{- \sum_{i=1}^{n} \alpha_i [Y_i(X_i^\top \beta + \beta_0) - (1 - \xi_i)] - \sum_{i=1}^{n} \gamma_i \xi_i}_{\text{remaining constraints}}$$

Necessary conditions (taking derivatives)

- $\beta = \sum_{i=1}^n \alpha_i Y_i X_i$
- $0 = \sum_{i=1}^n \alpha_i Y_i$
- $\alpha_i = \lambda - \gamma_i$

(As well as positivity constraints on Lagrangian parameters)

Support vector classifier

Substituting, we reaquire the Wolfe dual

This, combined with the KKT conditions uniquely characterize the solution:

$$\max_{\alpha \text{ subject to: KKT + Wolfe dual}} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{i'=1}^{n} \alpha_i \alpha_{i'} Y_i Y_{i'} X_i^\top X_{i'}$$

(See Chapter 12.2.1 in "Elements of Statistical Learning")

Note: the necessary conditions $\beta = \sum_{i=1}^{n} \alpha_i Y_i X_i$ imply estimators of the form

- $\hat{\beta} = \sum_{i=1}^{n} \hat{\alpha}_i Y_i X_i$
- $\hat{\beta}^\top X = \sum_{i=1}^{n} \hat{\alpha}_i Y_i X_i^\top X$

SVMs: tuning parameter

We can think of t as a budget for the problem

If $t = 0$, then there is no budget and we won't tolerate any margin violations

If $t > 0$, then no more than $\lfloor t \rfloor$ observations can be misclassified

A larger t then leads to larger margins

(we allow more margin violations)

SVMs: tuning parameter

Further intuition:

Like the optimal hyperplane, only observations that violate the margin determine $\mathcal{H}$

A large t allows for many violations, hence many observations factor into the fit

A small t means only a few observations do

Hence, t calibrates a bias/variance trade-off, as expected

In practice, t gets selected via cross-validation

SVMs: tuning parameter

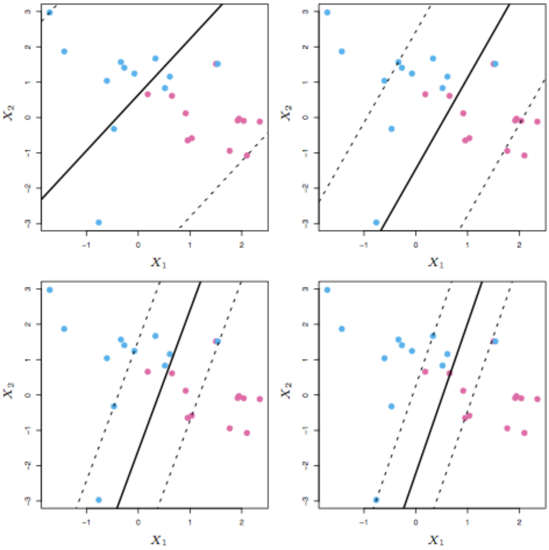


Figure 9.7 in ISL

Support vector classifier in R

A common package to use is e1071

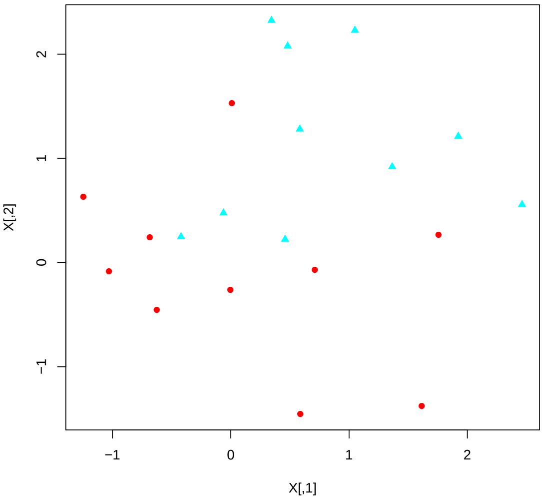
```
X = matrix(rnorm(20*2),ncol=2)
Y = c(rep(-1,10),rep(1,10))
X[Y == 1,] = X[Y == 1,] + 1

col = rep(0,length(Y))
col[Y == -1] = rainbow(2)[1]
col[Y == 1] = rainbow(2)[2]

pch = rep(0,length(Y))
pch[Y == -1] = 16
pch[Y == 1] = 17

plot(X,col=col,pch=pch)
```

Support vector classifier in R



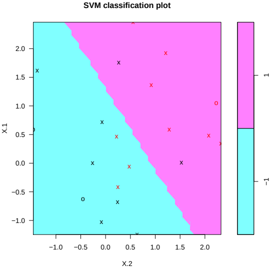
Support vector classifier in R

```
library(e1071)
dat  =data.frame(X=X, Y=as.factor(Y))
svmfit=svm(Y~., data=dat, kernel="linear", cost=cost)
```

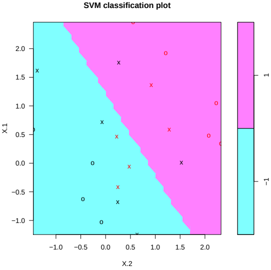
Important: Their definition of cost is the Lagrangian version, which we defined as λ

Hence, a small cost means a large t and a wider margin

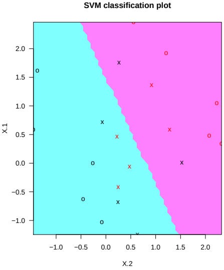
Support vector classifier in R



cost $= .1$



cost $= 1$

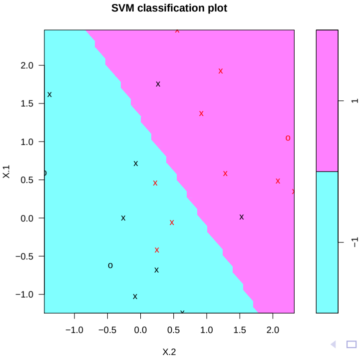


cost $= 10$

Support vector classifier in R

```
tune.out = tune(svm,Y~.,data=dat,kernel="linear",
     ranges=list(cost=c(0.001, 0.01, 0.1, 1,5,10,100)))
best.model = tune.out$best.model
```

Note that best.model is an svm object:



Next time: Kernel methods

Intuition: Many methods have linear decision boundaries

We know that sometimes this isn't sufficient to represent data

Example: Sometimes we need to included a polynomial effect or a log transform in multiple regression

Sometimes, a linear boundary, but in a different space makes all the difference..