Lecture 9 – Support Vector Machines

Martin Spindler

2016-04-22

Introduction

- Outcome variable: $Y \in \{-1, +1\}$
- Generalization to multi-valued outcomes is straight forward.
- Basic Idea (for classification): Separating the space of features by hyperplanes into different regressions so that the dependent variable / outcome is separated.
- *Seperating* hyperplanes
- Graph: cf white board
- In this lecture we focus only on the basic idea.

Digression: Hyperplanes

- In a p-dimensional space a *hyperplane* is a flat affine subspace of dimension p-1-
- Example 1: In $p = 2$ a hyperplane is a line.
- Example 2: In $p = 3$ a hyperplane is a plane.
- A hyperplane is a p-dimensional space is defined by

$$\beta_0 + \beta_1 X_1 + \ldots \beta_p X_p = 0$$

.

- If $\beta_0 + \beta_1 X_1 + \ldots \beta_p X_p < 0$ than $X = (X_1, \ldots, X_p)$ lies on one side of the hyperplane.
- If $\beta_0 + \beta_1 X_1 + \ldots \beta_p X_p > 0$ than X lies on the other side of the hyperplane.
- Hence: Separation of the space into two parts.
- $f(x) = \beta_0 + \beta_1 x_1 + \ldots \beta_p x_p$ gives the signed distance from a point x to the hyperplane defined by $f(x) = 0$

Support Vector Classifier

- Training data: $(x_1, y_1), \ldots, (x_n, y_n)$, $x_i \in \mathbb{R}^p$ and $y_i \in \{-1, +1\}$
- Definition hyperplane: $\{x : f(x) = x^T\beta + \beta_0 = 0\}$ (β unit vector with $\|\beta\| = 1$)
- Classification rule: $G(x) = sign[x^T\beta + \beta_0]$

Support Vector Classifier: Separable Case

- Separable means: $y_i f(x_i) > 0 \forall i$ for a plane $f(x)$
- Find the hyperplane that creates the biggest margin between the classes for -1 and $+1$.
- Optimization problem: $\max_{\beta,\beta_0,\|\beta\|=1} M$ subject to

$$y_i(x_i^T \beta + \beta_0) \geq M, i = 1, \ldots, n$$

- Equivalent formulation: $\min_{\beta,\beta_0} \|\beta\|$ subject to

$$y_i(x_i^T \beta + \beta_0) \geq 1, i = 1, \ldots, n$$

Support Vector Classifiers: Non-separable Case

- Now: classes overlap in the feature space.
- Still maximize M, but allow for some points to be on the wrong side of the margin.
- Slack variables $\xi = (\xi_1, \ldots, x_n)$
- Modification of the constraints: $y_i({x_i}^T\beta + \beta_0) \geq M - \xi_i$ or $y_i({x_i}^T\beta + \beta 0 \geq M(1 - \xi_i)$
- $\forall \xi \geq 0, \sum_{i=1}^{n} \xi_i \leq constant$
- Interpretation: overlap in actual distance from the margin vs overlap in relative distance.
- Focus on the second case (b/c convex optimization problem)

Support Vector Classifiers: Non-separable Case

- ξ_i proportional amount by which the prediction $f(x_i)$ is on the wrong side of the margin.
- Missclassification occurs, if $\xi_i > 1$
- Bounding $\sum \xi_i$ at value K, bounds the number of training missclassifications at K.
- Equivalent formulation of the problem: min $\|\beta\|$ subject to

$$y_i({x_i}^T \beta + \beta_0) \geq 1 - \xi_i \forall i, \xi_i \geq 0, \sum \xi_i \leq K$$

Support Vector Machines

- Up to now: linear boundaries in the feature space.
- Flexibility by enlarging the feature space using basis expansions (e.g. polynomials, splines)
- Better training-class separation and nonlinear boundaries in the original space.
- Selection of basis functions $h_m(x), m = 1, \dots, M$ and fit of SV classifier using the input features $h(x_i) = (h_1(x_i), \dots, h_M(x_i))$.
- Nonlinear function $\hat{f}(x) = h(x)^T \hat{\beta} + \hat{\beta}_0$
- Classifier: $\hat{G} = sign(\hat{f}(x))$

Support Vector Machines

- SVM use a very large space of basis functions leading to computational problems.
- Problem of overfitting.
- SVM technology takes care of both problems.

Support Vector Machines

- (omitting technical details)
- Solution of the optimization problem involve $h(x)$ only through inner products.
- Knowledge of the kernel functions $K(x, x') = \langle h(x), h(x') \rangle$ is sufficient.
- Examples: $+$ dth degree polynomial: $K(x, x') = (1 + \langle x, x' \rangle)^d$
  $+$ Radial basis: $K(x, x') = \exp(-\gamma \|x - x'\|^2)$

Support Vector Machines | Example

- Consider two-dimensional space (X_1, X_2) and polynomial kernel of degree 2.
- $K(X, X') =$ $1 + 2X_1X_1' + 2X_2X_2' + (X_1X_1')^2 + (X_2X_2')^2 + 2X_1X_1'X_2X_2'$
- Then $M = 6$ and $h_1(X) = 1, h_2(X) = \sqrt(2)X_1, \ldots$. Then $K(X, X') = \langle h(X), h(X') \rangle$

SVM as a Penalization Method

- With $f(x) = h(x)^T\beta + \beta_0$, we consider the optimization problem:

$$\min_{\beta_0,\beta} \sum_{i=1}^{n} [1 - y_i f(x_i)]_+ + \frac{\lambda}{2}\|\beta\|^2$$

- loss + penalty
- Hinge loss function: $L(y, f) = [1 - yf]_+$
- Solution to the above optimization problem (with $\lambda = 1/C$) is the same as for the SVM problem.
- (C is a Cost parameter related to K)

SVM | An Illustration

We use the library *e1071* (alternative *LiblineaR* for very large linear problems)

```
library(e1071)
```

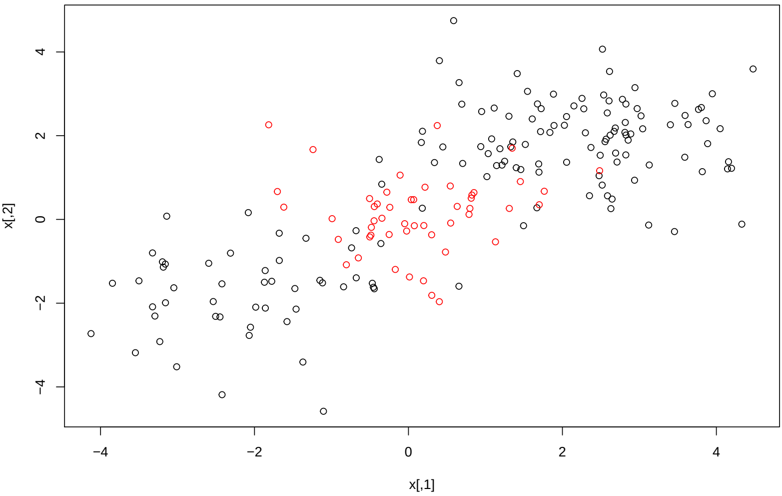
```
## Warning: package 'e1071' was built under R version 3.1.3
```

First, we generate data we would like to classify

```
set.seed(12345)
x = matrix(rnorm(200*2), ncol=2)
x[1:100,] = x[1:100,] + 2
x[101:150,] = x[101:150,] - 2
y = c(rep(1,150), rep(2,50))
dat = data.frame(x=x, y=as.factor(y))
```

SVM | An Illustration

```
plot(x, col=y)
```



SVM | An Illustration

Next, we split the data into training and testing sample

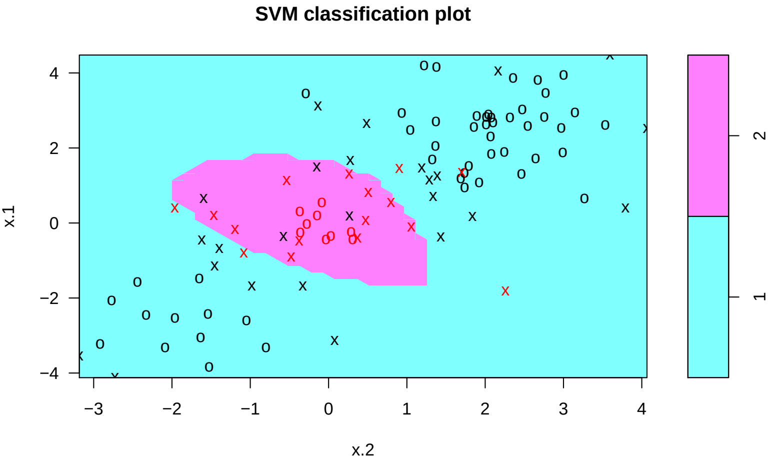
```
train = sample(200,100)
```

Then we fit a SVM with radial basis and plot the result

```
svmfit = svm(y~., data=dat[train,], kernel="radial", gamma=
# summary(smvfit)
```

SVM | An Illustration

```r
plot(svmfit, dat[train,])
```



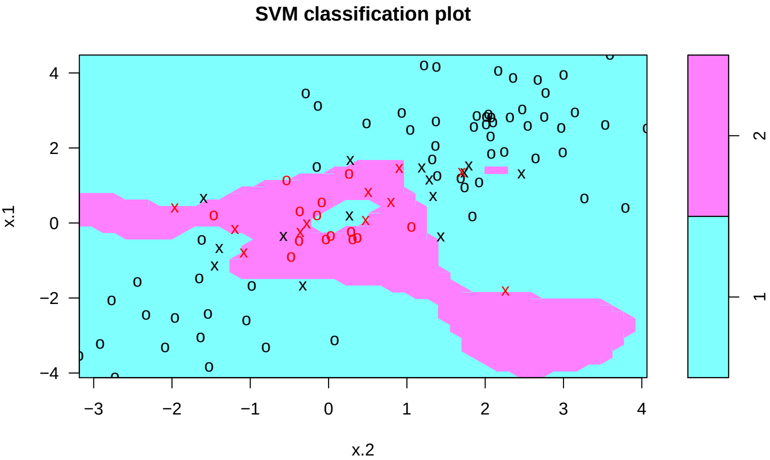
SVM | An Illustration

We can now increase the cost parameter to reduce the training errors

```
svmfit = svm(y~., data=dat[train,], kernel="radial", gamma=
#plot(svmfit, dat[train,])
# summary(smvfit)
```

SVM | An Illustration

```r
plot(svmfit, dat[train,])
```



SVM | An Illustration

Selection of the cost parameter and γ by CV

```
tune.out=tune(svm, y~., data=dat[train,], kernel="radial",
#summary(tune.out)
```

SVM | An Illustration

Finally, we test it on the testing data

```
table(true=dat[-train,"y"], pred=predict(tune.out$best.mode
```

```
##      pred
## true  1  2
##     1 70  5
##     2  4 21
```