Note: Support Vector Machines

Sun Zhao

January 11, 2013

1 Support Vector Machines Intuition

A support vector machine(SVM for short) constructs a hyperplane or set of hyper-planes in a high- or infinite-dimensional space, which can be used for classification, regression, or other tasks. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training data point of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier. H_1, H_2, H_3 in Fig. 1 are three hypothesises trying to separate black and white pointers. Obviously, H_1 does not separate the classes. H_2 does, but only with a small margin. H_3 separates them with the maximum margin. SVM chooses the hyperplane so that the distance from it to the nearest data point on each side is maximized. Recall

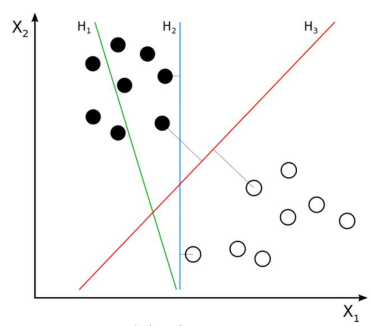

Figure 1:

that in logistic regression, we use $\Theta^T x = 0$ as the hyper-plane, and predict positive class if $\Theta^T x \geq 0$, otherwise, negative class. SVM is trying to keeps a minimum margin between the two classes and wants $\Theta^T x \geq 1$ if x is positive and $\Theta^T x \leq -1$ if x is negative. Let H_1 denotes $\Theta^T x = 1$, H_{-1} denotes $\Theta^T x = -1$ and H_0 denotes $\Theta^T x = 0$. The margin distance between H_1 and H_{-1} is $\frac{2}{||\Theta_{1...n}||}$ where $||\Theta_{1...n}||$ equals $\sqrt{\sum_{i=1}^{n} \Theta_i^2}$. The location relationships between training examples and H_1, H_0, H_{-1} is shown in Fig. 2. The optimization problem can be summarized as follow:

$$\min ||\Theta_{1...n}|| \tag{1}$$

Subject to(for any i = 1 ... m)

$$y^{(i)}(\Theta^T x^{(i)}) \geq 1 \tag{2}$$

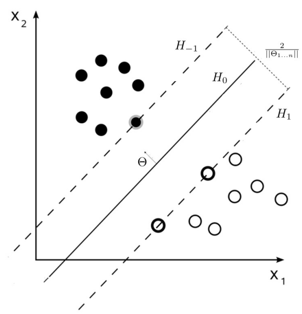

Figure 2:

2 Cost Function

Instead of following the logarithm example cost function, SVM introduces cost function show in Fig. 3. The new cost functions have following properties:

$$\text{if } y^{(i)}{=}1, \text{ cost} = 0 \text{ when } \Theta^T x^{(i)} \geq 1$$
$$\text{if } y^{(i)}{=}0, \text{ cost} = 0 \text{ when } \Theta^T x^{(i)} \leq -1$$

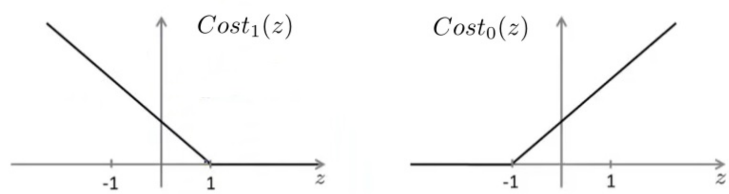

Figure 3:

The cost function is defined as (3). We can move and scale the const value, thus, have a equivalent cost function shown in (4). C is called the regular factor. If C is set to be very large, the optimization algorithm will try to set $Cost_1$ and $Cost_0$ to be 0 which is satisfied when $\Theta^T x^{(i)} \geq 1$ if $y^{(i)} = 1$ and $\Theta^T x^{(i)} \leq -1$ if $y^{(i)} = 0$. What's more, the minimizing term becomes $minimize \sum_{j=1}^{n} \Theta_j^2$ as the first term approaching to 0. This is consistent with optimization definition described in section 1.

$$\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} Cost_1(\Theta^T x^{(i)}) + (1-y^{(i)}) Cost_0(\Theta^T x^{(i)})\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\Theta_j^2 \tag{3}$$

$$C\left[\sum_{i=1}^{m} y^{(i)} Cost_1(\Theta^T x^{(i)}) + (1-y^{(i)}) Cost_0(\Theta^T x^{(i)})\right] + \sum_{j=1}^{n}\Theta_j^2 \tag{4}$$

3 Kernels

Kernels are used to select features and make the hypothesis having a form like $h_\Theta(x) = \Theta_0 + \Theta_1 f_1 + \Theta_2 f_2 + \ldots$. The features f_i is defined as (5) where $l^{(i)}$ is landmarks we should manually choose. The most common kernel function is Gaussian kernel shown in (6). Gaussian kernel measures the similarity of $x^{(i)}$ and landmarks $l^{(j)}$ which means that kernel values goes 1 while $x^{(i)}$ goes near to $l^{(j)}$ and goes 0 while $x^{(i)}$ goes far from $l^{(j)}$. The kernel is useful when the classification problem is non-linear which is not understood by me.

$$f_i = kernel(x, l^{(i)}) \tag{5}$$

$$GaussianKenel(x, l^{(i)}) = exp(\frac{||x - l^{(i)}||}{2\sigma^2}) \tag{6}$$

4 Using an SVM

While using an SVM, It is suggested that we should use a mature software package like liblinear, libsvm etc. What we need to specify is the choice of C and kernel function. Teacher gives some notations when using SVM listed as following:

- Do perform feature scaling whenever need.
- No all similarity functions make valid kernels.
- Polynomial, String, chi-square, histogram intersection kernel can be used.
- Using built-in multi-class SVM or One-VS-All method for multi-class classification.
- If n is large(relative to m) then use logistic regression or SVM without kernel function.
- If n is small and m is mediate then use SVM with Gaussian kernel function.
- If n is small and m is large then create more features.
- Use C to tune the trade off between bias and variance.

5 Summary

SVM is one of the most popular classifier among machine learning classifiers. It can handle linear and non linear classification problem. Mature packages are provided for efficient running SVM.