Support Vector Machine(SGD-based)

Zhiliang Tian

2016.05.22

1 Introduction

1.1 What is Support Vector Machine

Support Vector Machine(SVM), is a two-categories classifier. The thought of original SVM is two norm:

- Classifing correctly on all observed samples. It means training data accuracy is 100%.
- Making the distance between different categories as much as they can. Actually, it required making the minimum of distance between different categories as big as they can. The minimum distance is the distance between samples on each categories' boundary.

As follow:

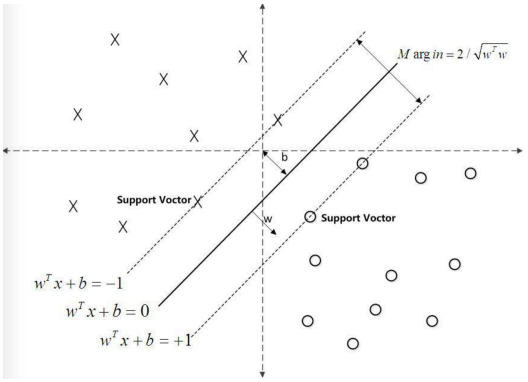

Figure 1: Figure Example

The solid line stands for separating hyperplane(called decision boundary), two dotted lines stand for categories' boundary. The minimum of distance between categories is the gap between dotted lines, called "margin". The sample on categories' boundary decide the margin, called Support Vector.

1.2 slack variable and non-linear

The original SVM can only deal with linearly separable data. If data has some noise(linearly non-separable), we need add slack variable. If data is non-linearly, we need use kernel trick, map data to high-dimension space, then do linear classification there.

2 classify formula and margin

2.1 classify

This is the formula for classify. given a input feature $\mathbf{x}$, f($\mathbf{x}$) is the output of classifier.

$$\begin{aligned} f(\mathbf{x}) &= sign(W * \mathbf{x} + b) \\ sign(t) &= \begin{cases} 1 & t \geq 0 \\ -1 & t < 0 \end{cases} \end{aligned} \tag{1}$$

$\mathbf{x}$ is a input feature, a vector with D dimension. W is weight parameter, connecting input feature with output score. b is bias. On data corpus, the output of each sample x_i is as follow:

$$f(x_i) = sign(\sum_{j=0}^{D} W_j * x_{ij} + b) \tag{2}$$

The first norm of SVM require:

$$\forall i \quad f(x_i) * y_i > 0 \tag{3}$$

y_i is the label of x_i.

2.2 margin

The second norm of SVM is maximizing the minimum of distance. We use functional margin and geometric margin to evaluate it:

2.2.1 functional margin

For all sample (x_i, y_i) in tranning data, γ' is the functional margin of classifier.

$$\begin{aligned} &\gamma_i{}' = y_i * (\sum_{j=0}^{D} W_j * x_{ij} + b) \\ &\gamma' = min(\gamma_i{}') \end{aligned} \tag{4}$$

2.2.2 geometric margin

For all sample (x_i, y_i) in tranning data, γ is the geometric margin of classifier.

$$\begin{aligned} &\gamma_i = y_i * f(x_i) = y_i * \frac{(\sum_{j=0}^{D} W_j * x_{ij} + b)}{||W||} \\ &\gamma = min(\gamma_i) \end{aligned} \tag{5}$$

We can see geometric margin is the "normalized version" of functional margin.

$$\gamma = \frac{\gamma'}{||W||} \tag{6}$$

SVM is to maximize the geometric margin.

2.3 goal of optimising

The goal of optimising:

$$\begin{aligned} &max \gamma \\ &s.t. \forall i \quad y_i * f(x_i) = y_i * \frac{(\sum_{j=0}^{D} W_j * x_{ij} + b)}{||W||} \geq \gamma \end{aligned} \tag{7}$$

$$\begin{aligned} &max \frac{\gamma'}{||W||} \\ &s.t. \forall i \qquad y_i * f(x_i) = y_i * \frac{(\sum_{j=0}^{D} W_j * x_{ij} + b)}{||W||} \geq \gamma' \end{aligned} \tag{8}$$

Notice that, maximizing $\frac{1}{||W||}$ is totally equal to minimizing $\frac{1}{2}||W||^2$ (????), so

$$\begin{aligned} &max \frac{1}{2}||W||^2 \\ &s.t. \forall i \qquad y_i * f(x_i) = y_i * \frac{(\sum_{j=0}^{D} W_j * x_{ij} + b)}{||W||} - 1 \geq 0 \end{aligned} \tag{9}$$

Equation 9 is the final goal of optimising, a convex quadiatic programming.

3 SGD-based optimising —- Pegasos

3.1 introduction

Stochastic gradient descent(SGD) ask for a specific loss function. Given a sample, this loss function tell us how to optimize, where should parameter go. Under this loss function, we can approach to final solution sample by sample, using gradient descent. We should balance two norms of SVM.

3.2 hinge loss and square loss

3.2.1 hinge loss

$$l_h = \begin{cases} 0 & y_i * f(x_i) - 1 \geq 0 \\ 1 - y_i * f(x_i) & y_i * f(x_i) - 1 < 0 \end{cases} \tag{10}$$

$$l_h = max(0, 1 - y_i * f(x_i)) \tag{11}$$

Hinge loss punishes the sample not obey the first norm(the restrict condition in Equation 9)

3.2.2 square loss

$$l_s = \frac{1}{2}||W||^2 \tag{12}$$

Square loss punishes the L2-norm. It maximizing Equation 9.

3.2.3 loss

$$l = \lambda l_s + l_h \tag{13}$$

Final loss is the combine of l_h and l_h. Pegasos algorithm is similar to it.

3.3 Pegasos algorithm

3.3.1 introduction

Pegasos algorithm is proposed in 2007, it is a SGD-based optimization of SVM. It seems like a mini-batch gradient descent.

3.3.2 original definition

We assume that t is the iterations of minibatch. For t-th mini-batch, A_t is the samples set in this batch, A_t contains k samples ($|A_t| = k$).

$$Loss = \frac{\lambda}{2}||W||^2 + \frac{1}{k}\sum_{(\mathbf{x}_i, y_i)\in A_t} y_i * f(\mathbf{x}_i) \tag{14}$$

3.3.3 gradient

At t-th iteration:

$$\frac{\alpha(Loss)}{\alpha(W_t)} = \frac{\lambda}{2} * \frac{\alpha(||W_t||^2)}{\alpha(W_t)} + \frac{1}{k} * \frac{\alpha(\sum_{(x_i,y_i)\in A_t} y_i * f(x_i)}{\alpha(W_t)} \tag{15}$$

$$\frac{\alpha(Loss)}{\alpha(W_t)} = \lambda * W_t - \frac{1}{k} * \sum_{(\mathbf{x}_i, y_i)\in A_t^+} y_i * \mathbf{x}_i \tag{16}$$

A_t^+ is a sample set whose hinge loss is non-zero in A_t. We set learning rate $\eta_t = \frac{1}{\lambda * t}$ (why???).

$$W'_{t+1} = W_t - \eta_t \frac{\alpha(Loss)}{\alpha(W_t)} \tag{17}$$

$$W'_{t+1} = (1 - \eta_t\lambda)W_t + \frac{\eta_t}{k}\sum_{(\mathbf{x}_i, y_i)\in A_t^+} y_i * \mathbf{x}_i \tag{18}$$

W'_{t+1} is a non-normalization W at $t+1$ iteration.

3.3.4 normalization

Then we normalize W'_{t+1} (why???). We must limit $||W|| \leq 1/\sqrt{\lambda}$. So,

$$W_{t+1} = min(1, \frac{1}{\sqrt{\lambda} * ||W'_{t+1}||})W'_{t+1} \tag{19}$$

Equation 18 and 19 is a complete mini-batch processing on Pagasos algorithm.

4 reference and appendix

- Original Pegasos. Pegasos: Primal Estimated sub-GrAdient SOlver for SVM
- Open source. https://github.com/liuhongjiang/blog_code/blob/master/pegasos
- My code. https://github.com/tianzhiliang/svm