2019-03-28

Support Vector Machines

参考资料：[Andrew Ng] CS229 Lecture notes 3

Nankai-Baidu
Joint Laboratory

Parallel and Distributed
Software Technology Lab

Support Vector Machines

SVMs are among the best (and many believe are indeed the best) “off-the-shelf” supervised learning algorithms.

Support Vector Machines

- margins and the idea of separating data with a large “gap”
- the optimal margin classifier, which will lead us into a digression on Lagrange duality
- kernels, which give a way to apply SVMs efficiently in very high dimensional (such as infinitedimensional) feature spaces
- SMO algorithm, which gives an efficient implementation of SVMs

Support Vector Machines

- margins and the idea of separating data with a large "gap"
  边缘，余地，页边空白

- Consider logistic regression:

$$y = \begin{cases} 1 \; if \; p(y = 1|x, \theta) \geq 0.5, or \; \theta^T x \geq 0 \\ 0 \; if \; p(y = 1|x, \theta) < 0.5, or \; \theta^T x < 0 \end{cases}$$

- Consider a positive training example: $y = 1$

The larger $\theta^T x$ is, the larger also is $p(y = 1|x, \theta)$, and thus also the higher our degree of "confidence" that the label is 1.

Support Vector Machines

- margins and the idea of separating data with a large “gap”
  边缘，余地，页边空白

- Consider logistic regression:

$$y = \begin{cases} 1 \ if \ p(y=1|x,\theta) \geq 0.5, or \ \theta^T x \geq 0 \\ 0 \ if \ p(y=1|x,\theta) < 0.5, or \ \theta^T x < 0 \end{cases}$$

- Consider a positive training example: $y = 1$

Thus, informally we can think of our prediction as being a very confident one that $y = 1$ if $\theta^T x \gg 0$

Support Vector Machines

- margins and the idea of separating data with a large “gap”
  边缘，余地，页边空白

- Consider logistic regression:

$$y = \begin{cases} 1\ if\ p(y=1|x,\theta) \geq 0.5, or\ \theta^T x \geq 0 \\ 0\ if\ p(y=1|x,\theta) < 0.5, or\ \theta^T x < 0 \end{cases}$$

- Consider a positive training example: $y = 1$

Similarly, we think of logistic regression as making a very confident prediction of $y = 0$, if $\theta^T x \ll 0$

Support Vector Machines

- margins and the idea of separating data with a large “gap”
  边缘，余地，页边空白

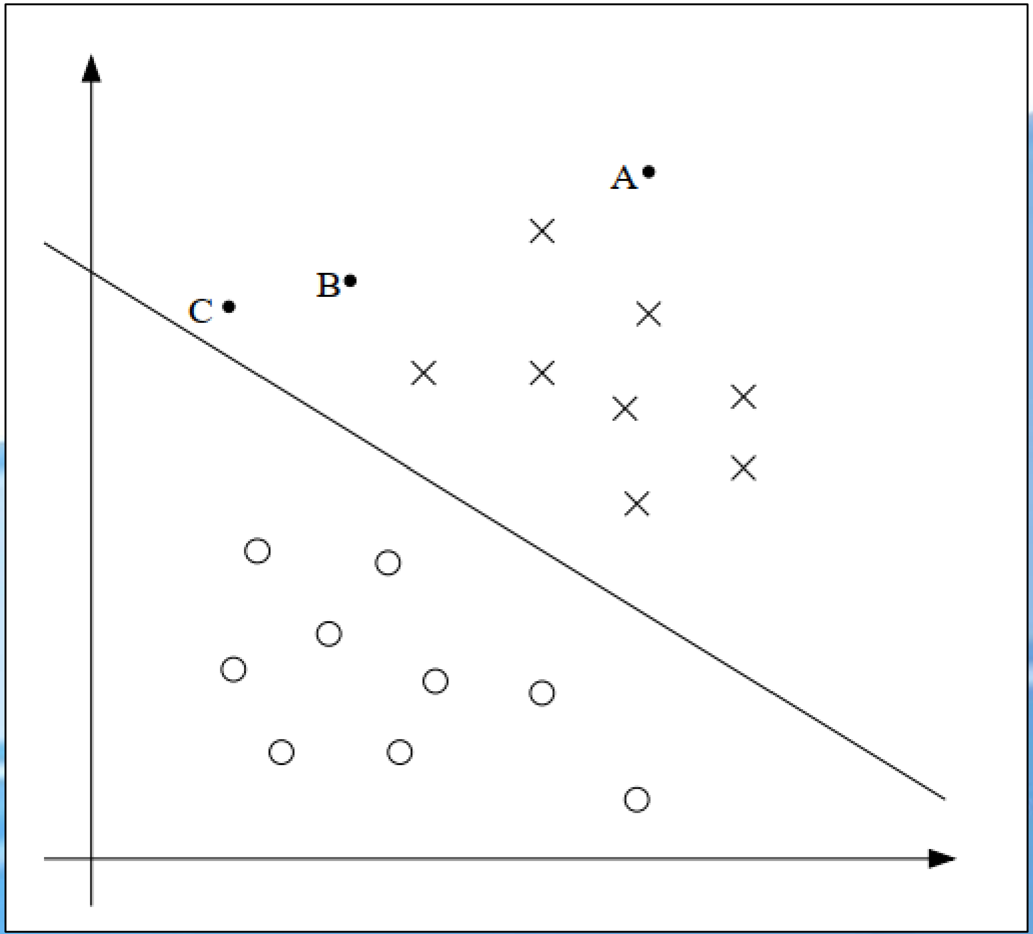
- Thus, a very confident one that $y = 1$ if $\theta^T x \gg 0$
- Similarly, a very confident prediction of $y = 0$, if $\theta^T x \ll 0$

- Given a training set: informally, a good fit to the training data if find θ s.t. $\theta^T x_i \gg 0$ whenever $y_i = 1$, $\theta^T x_i \ll 0$ whenever $y_i = 0$,

Since this would reflect a very confident (and correct) set of classifications for all the training examples.

Support Vector Machines

- For a different type of intuition, consider the following figure:
- 'x' : positive training examples
- 'o' : negative training examples
- a decision boundary : $\theta^T x = 0$
- three points : A, B, C



- quite confident that A : positive
- C positive negative ?

\- much confident about our prediction at A than at C.

Support Vector Machines

- We see that if a point is far from the separating hyperplane, then we may be significantly more confident in our predictions.
- Again, informally we think it'd be nice if, given a training set, we manage to find a decision boundary that allows us to make all correct and confident predictions on the training examples.

- From now, we'll use $y \in \{-1, +1\}$ to denote the class labels.

  Also, we will use parameters w, b, and write our classifier as

$$h_{w,b} = g(w^T x + b) = \begin{cases} +1, w^T x + b \geq 0, \\ -1, w^T x + b < 0. \end{cases}$$

the functional and geometric margins

- From now, we'll use $\mathrm{y} \in \{-1, +1\}$ to denote the class labels.

  Also, we will use parameters w, b, and write our classifier as

$$h_{w,b} = g(w^T x + b) = \begin{cases} +1, w^T x + b \geq 0, \\ -1, w^T x + b < 0. \end{cases}$$

- Given a training example (x_i, y_i), we define the functional margin of $(w,\ \ b)$ with respect to the training example

$$\gamma_i = y_i \cdot (w^T x + b)$$

- if y_i= +1, for our prediction to be confident and correct, we need $w^T x + b$ to be a large positive number.
- if y_i= -1, then for the functional margin to be large, we need $w^T x + b$ to be a large negative number.

the functional and geometric margins

- $$h_{w,b} = g(w^T x + b) = \begin{cases} +1, w^T x + b \geq 0, \\ -1, w^T x + b < 0. \end{cases}$$
- Given a training example (x_i, y_i), we define the functional margin
$$\gamma_i = y_i \cdot (w^T x + b)$$
- For our prediction to be confident and correct, for the functional margin to be large
  - if y_i= +1, need $w^T x + b$ to be a large positive number.
  - if y_i= -1, need $w^T x + b$ to be a large negative number.

Hence, a large functional margin represents a confident and a correct prediction.

the functional and geometric margins

- $$h_{w,b} = g(w^T x + b) = \begin{cases} +1, w^T x + b \geq 0, \\ -1, w^T x + b < 0. \end{cases}$$
- Given a training example (x_i, y_i), we define the functional margin
$$\gamma_i = y_i \cdot (w^T x + b)$$
- A large functional margin represents a confident and a correct prediction.

If we replace w with $2w$ and b with $2b$, then since $g(w^T x + b) = g(2w^T x + 2b)$, this would not change $h_{w,b}$ at all. (depends only on the sign, but not on the magnitude of $w^T x + b$) 同一个输入样本x_c

Intuitively, it might therefore make sense to impose some sort of normalization condition such as that $||w|| = 1$

Nankai-Baidu Joint Laboratory

Parallel and Distributed Software Technology Lab

the functional and geometric margins

- $$h_{w,b} = g(w^T x + b) = \begin{cases} +1, w^T x + b \geq 0, \\ -1, w^T x + b < 0. \end{cases}$$
- Given a training set $S = \{(x_i, y_i); i = 1, \ldots, m\}$, we define the functional margin of $(w,\ \ b)$ with respect to the training set

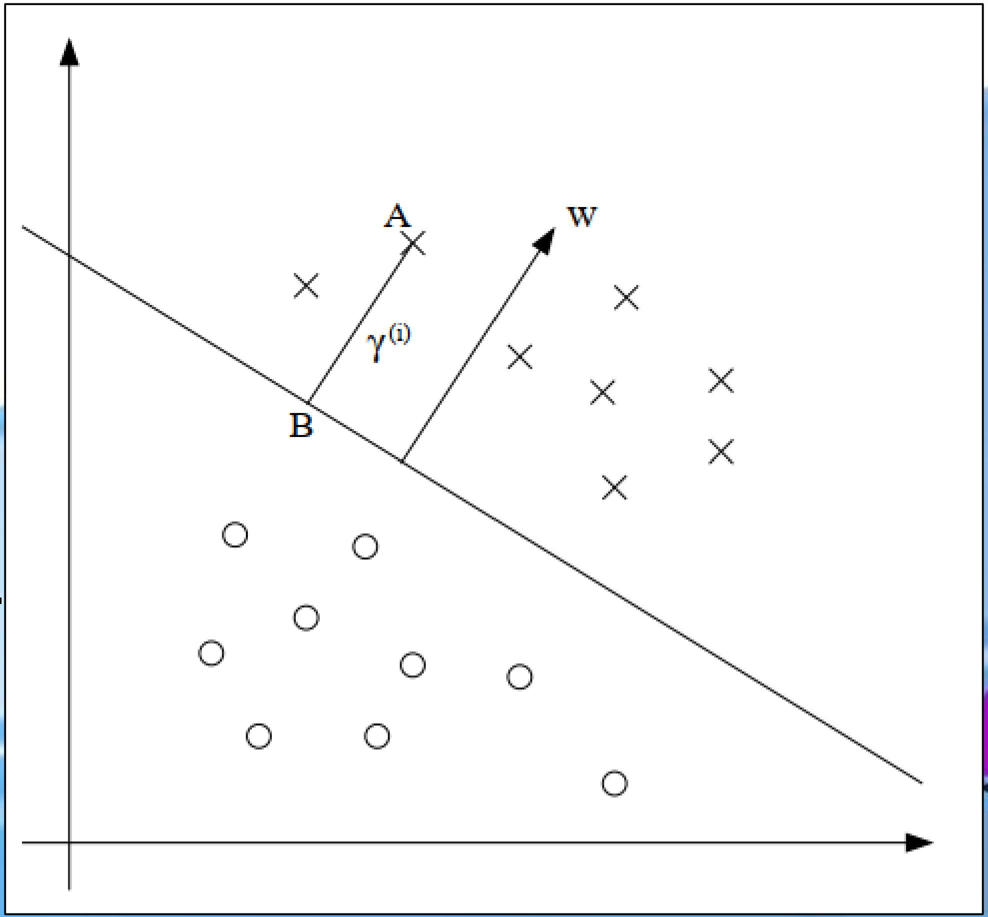
$$\hat{\gamma} = \min_{i=1,\ldots,m} \gamma_i \qquad \left(\gamma_i = y_i \cdot (w^T x + b)\right)$$

Nankai-Baidu Joint Laboratory

Parallel and Distributed Software Technology Lab

geometric margins

- Consider the picture below:
- The decision boundary correspoding to $(w,\ b)$
- w is orthogonal to the hyperplane



- Consider the point at A$(x_i, y_i = 1)$
- γ_i is given by the line segment AB.
- $\gamma_i = \left(\frac{w}{\|w\|}\right)^T x_i + \frac{b}{\|w\|}$

For a positive training example at A.

geometric margins

- Consider the point at A$(x_i, y_i = 1)$
- $\gamma_i = \left(\frac{w}{||w||}\right)^T x_i + \frac{b}{||w||}$ For a positive training example at A.
- More generally, the geometric margin of $(w,\ \ b)$ with respect to a training example (x_i, y_i) to be

$$\gamma_i = y_i \cdot \left(\left(\frac{w}{||w||}\right)^T x_i + \frac{b}{||w||}\right)$$

If $||w|| = 1$, then the functional margin equals the geometric margin.

the geometric margin is invariant to rescaling of the parameters

The optimal margin classifier

- Given a training set, it seems from our previous discussion that a natural desideratum is to try to find a decision boundary that maximizes the geometric margin, since this would reflect a very confident set of predictions on the training set and a good "fit" to the training data. Specifically, this will result in a classifier that separates the positive and the negative training examples with a "gap" (geometric margin).

The optimal margin classifier

- For now, we will assume that we are given a training set that is linearly separable.
- How will we find the one that achieves the maximum geometric margin? the following optimization problem: I.e., we want to maximize γ,subject to each training example having functional margin at least γ. The $||w|| = 1$ constraint moreover ensures that the functional margin equals to the geometric margin, so we are also guaranteed that all the geometric margins are at least γ.
- $\max_{\gamma,w,b} \gamma$ s.t $y_i \cdot \left(\left(\frac{w}{||w||}\right)^T x_i + \frac{b}{||w||}\right) \geq \gamma, i = 1, \dots, m$

The optimal margin classifier

- For now, we will assume that we are given a training set that is linearly separable.
- How will we find the one that achieves the maximum geometric margin? the following optimization problem: I.e., we want to maximize γ,subject to each training example having functional margin at least γ. The $||w|| = 1$ constraint moreover ensures that the functional margin equals to the geometric margin, so we are also guaranteed that all the geometric margins are at least γ.
- $\max_{\gamma,w,b} \gamma$ s.t $y_i \cdot (w^T x + b) \cdot \frac{1}{||w||} \geq \gamma, i = 1, \ldots, m$

The optimal margin classifier

- For now, we will assume that we are given a training set that is linearly separable.
- How will we find the one that achieves the maximum geometric margin? the following optimization problem: I.e., we want to maximize γ,subject to each training example having functional margin at least γ. The $\|w\| = 1$ constraint moreover ensures that the functional margin equals to the geometric margin, so we are also guaranteed that all the geometric margins are at least γ.

set $y_i \cdot (w^T x + b) = 1$

- $\max\limits_{\gamma,w,b} \gamma$ s.t $\frac{1}{\|w\|} \geq \gamma, i = 1, \ldots, m$

The optimal margin classifier

- For now, we will assume that we are given a training set that is linearly separable.
- How will we find the one that achieves the maximum geometric margin?

The "$||w|| = 1$" constraint is a nasty (non-convex) one.

Recall: we can add an arbitrary scaling constraint on w and b without changing anything. Consider:

- $\max_{w,b} \frac{1}{||w||}$

  s.t. $y_i \cdot (w^T x_i + b) \geq 1, i = 1, \ldots, m$

The optimal margin classifier

- For now, we will assume that we are given a training set that is linearly separable.
- How will we find the one that achieves the maximum geometric margin?

Consider:

- $\min_{w,b} \frac{1}{2} ||w||^2$

$$\text{s.t.} \quad y_i \cdot (w^T x_i + b) \geq 1, i = 1, \dots, m$$

- Lagrange duality

The optimal margin classifier

- For now, we will assume that we are given a training set that is linearly separable.
- How will we find the one that achieves the maximum geometric margin?
- Lagrange duality

$$w^T x + b = \sum_{i=1}^{m} \alpha_i y_i \langle x_i, x \rangle + b$$

Kernels

- attributes : the “original” input value x
- features : $\emptyset(x)$ $\qquad\qquad$ $\emptyset$: the feature mapping

$$\emptyset(x) = \begin{bmatrix} x \\ x^2 \\ x^3 \end{bmatrix}$$

Rather than applying SVMs using the original input attributes x, we may instead want to learn using some features $\emptyset(x)$.

To do so, we simply need to go over our previous algorithm, and replace x everywhere in it with $\emptyset(x)$.

Kernels

- attributes : the "original" input value x
- features : $\emptyset(x)$ $\emptyset$: the feature mapping
- Since the algorithm can be written entirely in terms of the inner products $\langle x, z\rangle$, this means that we would replace all those inner products with $\langle\emptyset(x), \emptyset(z)\rangle$. Define the corresponding Kernel to be:

$$K(x, z) = \emptyset(x)^T \emptyset(z)$$

everywhere we previously had $\langle x, z\rangle$ in our algorithm, we could simply replace it with $K(x, z)$, and our algorithm would now be learning using the features $\emptyset$.

Kernels

- we could easily compute $K(x,z)$ by finding $\emptyset(x)$ and $\emptyset(z)$ and taking their inner product.
- Whatis more interesting is that often, $K(x,z)$ may be very inexpensive to calculate, even though $\emptyset(x)$ itself may be very expensive to calculate(perhaps because it is an extremely high dimensional vector).
- In such settings, an efficient way to calculate $K(x,z)$,

we can get SVMs to learn in the high dimensional feature space given by $\emptyset$, but without ever having to explicitly find or represent vectors $\emptyset(x)$.

Kernels

- Intuitively, we can think of $K(x, z)$ as some measurement of how similar are $\emptyset(x)$ and $\emptyset(z)$, or of how similar are x and z.

if $\emptyset(x)$ and $\emptyset(z)$ are close together, then we might expect $K(x, z)$ to be large. Conversely, if $\emptyset(x)$ and $\emptyset(z)$ are far apart, ⋯ be small.

- Gaussian kernel :

$$K(x, z) = exp\left(-\frac{||x - z||^2}{2\sigma^2}\right)$$

A reasonable measure of x and z's similarity, and is close to 1 when x and z are close, and near 0 when x and z are far apart.

Corresponds to an infinite dimensional feature mapping $\emptyset$

Kernels

- some finite set of m points $\{x_1, x_2, \dots, x_m\}$,

  Define the Kernel Matrix to be a square, m by m matrix K.

$$K_{ij} = K(x_i, x_j)$$

if K is a valid kernel (i.e., if it corresponds to some feature mapping $\emptyset$), then the corresponding Kernel matrix K is symmetric positive semidefinite.

Theorem (Mercer). Necessary and Sufficient

Given a function K, apart from trying to find a feature mapping $\emptyset$ that corresponds to it, this theorem therefore gives another way of testing if it is a valid kernel.

Kernels

- Consider the digit recognition problem, in which given an image (16x16 pixels) of a handwritten digit (0-9)

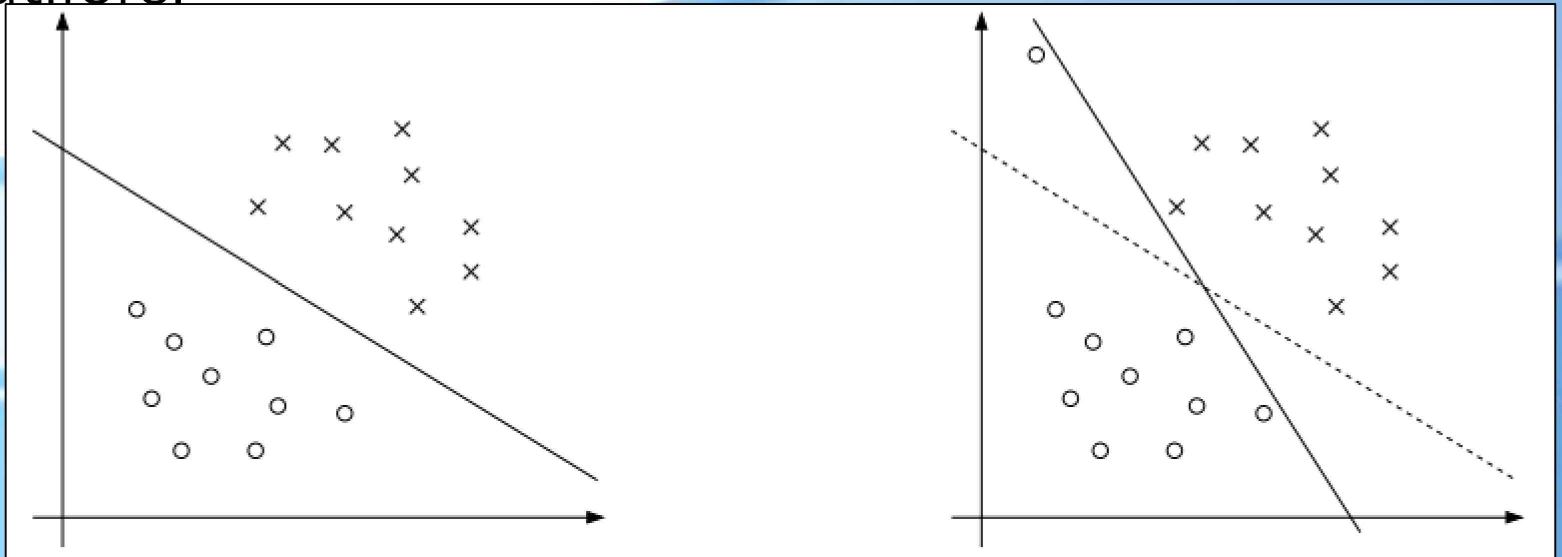
Using the Gaussian kernel, SVMs are able to obtain extremely good performance on this problem.

This was particularly surprising since the input attributes x were just 256-dimensional vectors of the image pixel intensity values, and the system had no prior knowledge about vision, or even about which pixels are adjacent to which other ones.

- Keep in mind that the idea of kernels has significantly broader applicability than SVMs. : algorithm only inner products bwtween input attribute vectors

Regularization and the non-separable case

- While mapping data to a high dimensional feature space via Ø does generally increase the likelihood that the data in separable, we can't guarantee that it always will be so.
- Also, in some cases it is not clear that finding a separating hyperplane is exactly what we'd want to do, since that might be susceptible to outliers.



Regularization and the non-separable case

- for non-linearly separable datasets + be less sensitive to outliers
- $\min_{w,b} \frac{1}{2}\|w\|^2$

  s.t. $y_i \cdot (w^T x_i + b) \geq 1, i = 1, \dots, m$
- $\min_{w,b} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m} \xi_i$

  s.t. $y_i \cdot (w^T x_i + b) \geq 1 - \xi_i, i = 1, \dots, m$

  $\xi_i \geq 0, i = 1, \dots, m$

Thus, examples are now permitted to have margin less than 1. if an example has functional margin $1 - \xi_i$, we would pay a cost of the objective function being increased by $C\xi_i$.

Nankai-Baidu Joint Laboratory

Parallel and Distributed Software Technology Lab

The SMO algorithm

- Coordinate Ascent

$$W(\alpha_1, \ldots, \alpha_{i-1}, \alpha_i, \alpha_{i+1}, \ldots, \alpha_m)$$

Loop until convergence: {

For $i = 1, \ldots, m,$ {

$$\alpha_i := \arg\max_{\hat{\alpha}_i} W(\alpha_1, \ldots, \alpha_{i-1}, \alpha_i, \alpha_{i+1}, \ldots, \alpha_m).$$

}

}

The SMO algorithm

- the SMO algorithm

Repeat till convergence {

1. Select some pair α_i and α_j to update next (using a heuristic that tries to pick the two that will allow us to make the biggest progress towards the global maximum).
2. Reoptimize $W(\alpha)$ with respect to α_i and α_j, while holding all the other α_k's ($k \neq i, j$) fixed.

}

- 机器学习实战　libsvm