

# 台湾大学林轩田机器学习技法课程学习笔记5 -- Kernel Logistic Regression

作者：红色石头

微信公众号：AI有道 ( ID : redstonewill )

上节课我们主要介绍了Soft-Margin SVM，即如果允许有分类错误的点存在，那么在原来的Hard-Margin SVM中添加新的惩罚因子C，修正原来的公式，得到新的 $\alpha_n$ 值。最终的到的 $\alpha_n$ 有个上界，上界就是C。Soft-Margin SVM权衡了large-margin和error point之前的关系，目的是在尽可能犯更少错误的前提下，得到最大分类边界。本节课将把Soft-Margin SVM和我们之前介绍的Logistic Regression联系起来，研究如何使用kernel技巧来解决更多的问题。

## Soft-Margin SVM as Regularized Model

先复习一下我们已经介绍过的内容，我们最早开始讲了Hard-Margin Primal的数学表达式，然后推导了Hard-Margin Dual形式。后来，为了允许有错误点的存在（或者noise），也为了避免模型过于复杂化，造成过拟合，我们建立了Soft-Margin Primal的数学表达式，并引入了新的参数C作为权衡因子，然后也推导了其Soft-Margin Dual形式。因为Soft-Margin Dual SVM更加灵活、便于调整参数，所以在实际应用中，使用Soft-Margin Dual SVM来解决分类问题的情况更多一些。

Hard-Margin Primal	Soft-Margin Primal
$\min_{b, \mathbf{w}} \quad \frac{1}{2} \mathbf{w}^T \mathbf{w}$ $\text{s.t.} \quad y_n(\mathbf{w}^T \mathbf{z}_n + b) \geq 1$	$\min_{b, \mathbf{w}, \xi} \quad \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{n=1}^N \xi_n$ $\text{s.t.} \quad y_n(\mathbf{w}^T \mathbf{z}_n + b) \geq 1 - \xi_n, \xi_n \geq 0$
Hard-Margin Dual	Soft-Margin Dual
$\min_{\alpha} \quad \frac{1}{2} \alpha^T Q \alpha - \mathbf{1}^T \alpha$ $\text{s.t.} \quad \mathbf{y}^T \alpha = 0$ $0 \leq \alpha_n$	$\min_{\alpha} \quad \frac{1}{2} \alpha^T Q \alpha - \mathbf{1}^T \alpha$ $\text{s.t.} \quad \mathbf{y}^T \alpha = 0$ $0 \leq \alpha_n \leq C$

Soft-Margin Dual SVM有两个应用非常广泛的工具包，分别是Libsvm和Liblinear。Libsvm和Liblinear都是国立台湾大学的Chih-Jen Lin博士开发的，Chih-Jen Lin的个人网站为：[Welcome to Chih-Jen Lin's Home Page](#)

下面我们再来回顾一下Soft-Margin SVM的主要内容。我们的出发点是用 $\xi_n$ 来表示margin violation，即犯错值的大小，没有犯错对应的 $\xi_n = 0$ 。然后将有条件问题转化为对偶dual形式，使用QP来得到最佳化的解。

从另外一个角度来看， $\xi_n$ 描述的是点 $(x_n, y_n)$ 距离 $y_n(w^T z_n + b) = 1$ 的边界有多远。第一种情况是violating margin，即不满足 $y_n(w^T z_n + b) \geq 1$ 。那么 $\xi_n$ 可表示为： $\xi_n = 1 - y_n(w^T z_n + b) > 0$ 。第二种情况是not violating margin，即点 $(x_n, y_n)$ 在边界之外，满足 $y_n(w^T z_n + b) \geq 1$ 的条件，此时 $\xi_n = 0$ 。我们可以将两种情况整合到一个表达式中，对任意点：

$$\xi_n = \max(1 - y_n(w^T z_n + b), 0)$$

上式表明，如果有violating margin，则 $1 - y_n(w^T z_n + b) > 0$ ， $\xi_n = 1 - y_n(w^T z_n + b)$ ；如果not violating margin，则 $1 - y_n(w^T z_n + b) < 0$ ， $\xi_n = 0$ 。整合之后，我们可以把Soft-Margin SVM的最小化问题写成如下形式：

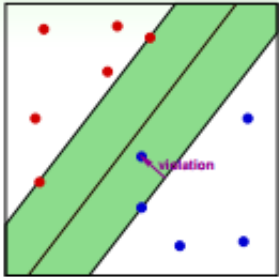
$$\frac{1}{2}w^T w + C \sum_{n=1}^N \max(1 - y_n(w^T z_n + b), 0)$$

经过这种转换之后，表征犯错误值大小的变量 $\xi_n$ 就被消去了，转而由一个max操作代替。

- record 'margin violation' by  $\xi_n$
- penalize with margin violation

$$\min_{b, w, \xi} \quad \frac{1}{2}w^T w + C \cdot \sum_{n=1}^N \xi_n$$

s.t.  $y_n(w^T z_n + b) \geq 1 - \xi_n$  and  $\xi_n \geq 0$  for all  $n$



on any  $(b, w)$ ,  $\xi_n = \text{margin violation} = \max(1 - y_n(w^T z_n + b), 0)$

- $(x_n, y_n)$  violating margin:  $\xi_n = 1 - y_n(w^T z_n + b)$
- $(x_n, y_n)$  not violating margin:  $\xi_n = 0$

'unconstrained' form of soft-margin SVM:

$$\min_{b, w} \quad \frac{1}{2}w^T w + C \sum_{n=1}^N \max(1 - y_n(w^T z_n + b), 0)$$

为什么要将把Soft-Margin SVM转换为这种unconstrained form呢？我们再来看一下转换后的形式，其中包含两项，第一项是w的内积，第二项关于y和w，b，z的表达式，似乎有点像一种错误估计 $e\hat{r}r$ ，则类似这样的形式：

$$\min \frac{1}{2} w^T w + C \sum e \hat{r}$$

看到这样的形式我们应该很熟悉，因为之前介绍的L2 Regularization中最优化问题的表达式跟这个是类似的：

$$\min \frac{\lambda}{N} w^T w + \frac{1}{N} \sum err$$

$$\min_{b, w} \quad \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{n=1}^N \max(1 - y_n(\mathbf{w}^T \mathbf{z}_n + b), 0)$$

familiar? :-)

$$\min \quad \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum \hat{err}$$

just L2 regularization

$$\min \quad \frac{\lambda}{N} \mathbf{w}^T \mathbf{w} + \frac{1}{N} \sum err$$

with shorter  $\mathbf{w}$ , another parameter, and special  $err$

这里提一下，既然unconstrained form SVM与L2 Regularization的形式是一致的，而且L2 Regularization的解法我们之前也介绍过，那么为什么不直接利用这种方法来解决unconstrained form SVM的问题呢？有两个原因。一个是这种无条件的最优化问题无法通过QP解决，即对偶推导和kernel都无法使用；另一个是这种形式中包含的max()项可能造成函数并不是处处可导，这种情况难以用微分方法解决。

我们在第一节课中就介绍过Hard-Margin SVM与Regularization Model是有关系的。Regularization的目标是最小化 $E_{in}$ ，条件是 $w^T w \leq C$ ，而Hard-Margin SVM的目标是最小化 $w^T w$ ，条件是 $E_{in} = 0$ ，即它们的最小化目标和限制条件是相互对调的。对于L2 Regularization来说，条件和最优化问题结合起来，整体形式写成：

$$\frac{\lambda}{N} w^T w + E_{in}$$

而对于Soft-Margin SVM来说，条件和最优化问题结合起来，整体形式写成：

$$\frac{1}{2} w^T w + C N \hat{E}_{in}$$

	minimize	constraint
regularization by constraint	$E_{in}$	$\mathbf{w}^T \mathbf{w} \leq C$
hard-margin SVM	$\mathbf{w}^T \mathbf{w}$	$E_{in} = 0$ [and more]
L2 regularization	$\frac{\lambda}{N} \mathbf{w}^T \mathbf{w} + E_{in}$	
soft-margin SVM	$\frac{1}{2} \mathbf{w}^T \mathbf{w} + C N \widehat{E}_{in}$	

通过对比，我们发现L2 Regularization和Soft-Margin SVM的形式是相同的，两个式子分别包含了参数 $\lambda$ 和 $C$ 。Soft-Margin SVM中的large margin对应着L2 Regularization中的short  $w$ ，也就是都让hyperplanes更简单一些。我们使用特别的 $\hat{err}$ 来代表可以容忍犯错误的程度，即soft margin。L2 Regularization中的 $\lambda$ 和Soft-Margin SVM中的 $C$ 也是相互对应的， $\lambda$ 越大， $w$ 会越小，Regularization的程度就越大； $C$ 越小， $\widehat{E}_{in}$ 会越大，相应的margin就越大。所以说增大 $C$ ，或者减小 $\lambda$ ，效果是一致的，Large-Margin等同于Regularization，都起到了防止过拟合的作用。

large margin  $\iff$  fewer hyperplanes  $\iff$  L2 regularization of short  $w$   
 soft margin  $\iff$  special  $\hat{err}$   
 larger  $C$  or  $C$   $\iff$  smaller  $\lambda$   $\iff$  less regularization

建立了Regularization和Soft-Margin SVM的关系，接下来我们将尝试看看是否能把SVM作为一个regularized的模型进行扩展，来解决其它一些问题。

## SVM versus Logistic Regression

上一小节，我们已经把Soft-Margin SVM转换成无条件形式：

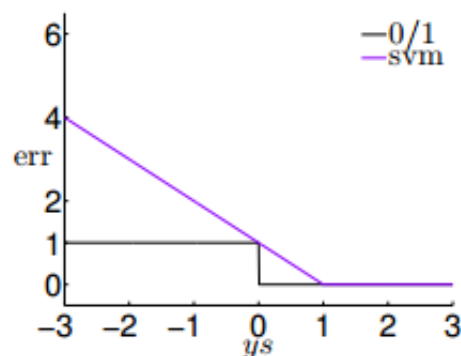
$$\min_{b, \mathbf{w}} \quad \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{n=1}^N \max(1 - y_n(\mathbf{w}^T \mathbf{z}_n + b), 0)$$

上式中第二项的 $\max(1 - y_n(\mathbf{w}^T \mathbf{z}_n + b), 0)$ 倍设置为 $\hat{err}$ 。下面我们来看看 $\hat{err}$ 与之前再二元分类中介绍过的 $err_{0/1}$ 有什么关系。

对于 $err_{0/1}$ ，它的linear score  $s = \mathbf{w}^T \mathbf{z}_n + b$ ，当 $ys \geq 0$ 时， $err_{0/1} = 0$ ；当 $ys < 0$ 时， $err_{0/1} = 1$ ，呈阶梯状，如下图所示。而对于 $\hat{err}$ ，当 $ys \geq 0$ 时， $err_{0/1} = 0$ ；当 $ys < 0$ 时， $err_{0/1} = 1 - ys$ ，呈折线状，如下图所示，通常把 $\hat{err}_{svm}$ 称为hinge error measure。比较两条error曲线，我们发现 $\hat{err}_{svm}$ 始终在 $err_{0/1}$ 的上面，则 $\hat{err}_{svm}$ 可作为 $err_{0/1}$ 的上界。所以，可以使用 $\hat{err}_{svm}$ 来代替 $err_{0/1}$ ，解决二元线性分类问题，而且 $\hat{err}_{svm}$ 是一个凸函数，使它在最佳化问题中有更好的性质。

linear score  $s = \mathbf{w}^T \mathbf{z}_n + b$

- $\text{err}_{0/1}(s, y) = \mathbb{I}[ys \leq 0]$
- $\hat{\text{err}}_{\text{SVM}}(s, y) = \max(1 - ys, 0)$ :  
upper bound of  $\text{err}_{0/1}$   
—often called **hinge error measure**

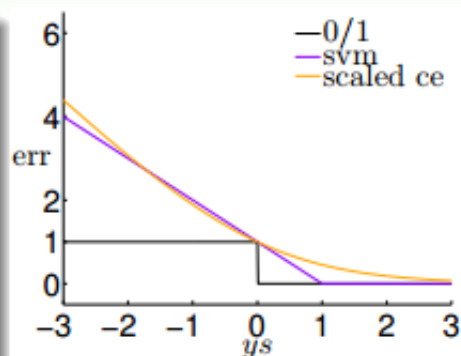


$\hat{\text{err}}_{\text{SVM}}$ : **algorithmic error measure**  
by **convex upper bound** of  $\text{err}_{0/1}$

紧接着，我们再来看一下logistic regression中的error function。逻辑回归中， $\text{err}_{\text{sce}} = \log_2(1 + \exp(-ys))$ ，当 $ys=0$ 时， $\text{err}_{\text{sce}} = 1$ 。它的err曲线如下所示。

linear score  $s = \mathbf{w}^T \mathbf{z}_n + b$

- $\text{err}_{0/1}(s, y) = \mathbb{I}[ys \leq 0]$
- $\hat{\text{err}}_{\text{SVM}}(s, y) = \max(1 - ys, 0)$ :  
upper bound of  $\text{err}_{0/1}$
- $\text{err}_{\text{SCE}}(s, y) = \log_2(1 + \exp(-ys))$ :  
another upper bound of  $\text{err}_{0/1}$  used in  
**logistic regression**



$-\infty$	$\leftarrow$	$ys$	$\rightarrow$	$+\infty$
$\approx -ys$		$\hat{\text{err}}_{\text{SVM}}(s, y)$		$= 0$
$\approx -ys$		$(\ln 2) \cdot \text{err}_{\text{SCE}}(s, y)$		$\approx 0$

很明显， $\text{err}_{\text{sce}}$ 也是 $\text{err}_{0/1}$ 的上界，而 $\text{err}_{\text{sce}}$ 与 $\hat{\text{err}}_{\text{svm}}$ 也是比较相近的。因为当 $ys$ 趋向正无穷大的时候， $\text{err}_{\text{sce}}$ 和 $\hat{\text{err}}_{\text{svm}}$ 都趋向于零；当 $ys$ 趋向负无穷大的时候， $\text{err}_{\text{sce}}$ 和 $\hat{\text{err}}_{\text{svm}}$ 都趋向于正无穷大。正因为二者的这种相似性，我们可以把SVM看成是L2-regularized logistic regression。

总结一下，我们已经介绍过几种Binary Classification的Linear Models，包括PLA，Logistic Regression和Soft-Margin SVM。PLA是相对简单的一个模型，对应的是 $\text{err}_{0/1}$ ，通过不断修正错误的点来获得最佳分类线。它的优点是简单快速，缺点是只对线性可分的情况有用，线性不可分的情况需要用到pocket算法。Logistic Regression对应的是 $\text{err}_{\text{sce}}$ ，通常使用GD/SGD算法求解最佳分类线。它的优点是凸函数 $\text{err}_{\text{sce}}$ 便于最优化求解，而且有regularization作为避免过拟合的保证；缺点是 $\text{err}_{\text{sce}}$ 作为 $\text{err}_{0/1}$ 的上界，当 $ys$ 很小（负值）时，上界变得更宽松，不利于最优化求解。Soft-Margin SVM对应的是 $\hat{\text{err}}_{\text{svm}}$ ，通常使用QP求解最佳分类线。它的优点和



Logistic Regression一样，凸优化问题计算简单而且分类线比较“粗壮”一些；缺点也和 Logistic Regression一样，当 $y_s$ 很小（负值）时，上界变得过于宽松。其实，Logistic Regression和Soft-Margin SVM都是在最佳化 $err_{0/1}$ 的上界而已。

PLA	soft-margin SVM	regularized logistic regression for classification
minimize $err_{0/1}$ specially <ul style="list-style-type: none"> <li>pros: <b>efficient if lin. separable</b></li> <li>cons: works only if lin. separable, otherwise needing <b>pocket</b></li> </ul>	minimize regularized $\widehat{err}_{SVM}$ by QP <ul style="list-style-type: none"> <li>pros: <b>'easy' optimization &amp; theoretical guarantee</b></li> <li>cons: loose bound of <math>err_{0/1}</math> for very negative <math>y_s</math></li> </ul>	minimize regularized $err_{SCE}$ by GD/SGD/... <ul style="list-style-type: none"> <li>pros: <b>'easy' optimization &amp; regularization guard</b></li> <li>cons: loose bound of <math>err_{0/1}</math> for very negative <math>y_s</math></li> </ul>

至此，可以看出，求解regularized logistic regression的问题等同于求解soft-margin SVM的问题。反过来，如果我们求解了一个soft-margin SVM的问题，那这个解能否直接为regularized logistic regression所用？来预测结果是正类的几率是多少，就像 regularized logistic regression做的一样。我们下一小节将来解答这个问题。

## SVM for Soft Binary Classification

接下来，我们探讨如何将SVM的结果应用在Soft Binary Classification中，得到是正类的概率值。

第一种简单的方法是先得到SVM的解 $(b_{svm}, w_{svm})$ ，然后直接代入到logistic regression中，得到 $g(x) = \theta(w_{svm}^T x + b_{svm})$ 。这种方法直接使用了SVM和logistic regression的相似性，一般情况下表现还不错。但是，这种形式过于简单，与logistic regression的关联不大，没有使用到logistic regression中好的性质和方法。

第二种简单的方法是同样先得到SVM的解 $(b_{svm}, w_{svm})$ ，然后把 $(b_{svm}, w_{svm})$ 作为logistic regression的初始值，再进行迭代训练修正，速度比较快，最后，将得到的 $b$ 和 $w$ 代入到 $g(x)$ 中。这种做法有点显得多此一举，因为并没有比直接使用logistic regression快捷多少。

### Naïve Idea 1

- 1 run SVM and get  $(b_{\text{SVM}}, \mathbf{w}_{\text{SVM}})$
- 2 return  $g(\mathbf{x}) = \theta(\mathbf{w}_{\text{SVM}}^T \mathbf{x} + b_{\text{SVM}})$

- 'direct' use of similarity —works reasonably well
- **no LogReg flavor**

### Naïve Idea 2

- 1 run SVM and get  $(b_{\text{SVM}}, \mathbf{w}_{\text{SVM}})$
- 2 run LogReg with  $(b_{\text{SVM}}, \mathbf{w}_{\text{SVM}})$  as  $\mathbf{w}_0$
- 3 return LogReg solution as  $g(\mathbf{x})$

- not really 'easier' than original LogReg
- **SVM flavor (kernel?) lost**

这两种方法都没有融合SVM和logistic regression各自的优势，下面构造一个模型，融合了二者的优势。构造的模型 $g(\mathbf{x})$ 表达式为：

$$g(\mathbf{x}) = \theta(A \cdot (\mathbf{w}_{\text{svm}}^T \Phi(\mathbf{x}) + b_{\text{svm}}) + B)$$

与上述第一种简单方法不同，我们额外增加了放缩因子A和平移因子B。首先利用SVM的解 $(b_{\text{svm}}, \mathbf{w}_{\text{svm}})$ 来构造这个模型，放缩因子A和平移因子B是待定系数。然后再用通用的logistic regression优化算法，通过迭代优化，得到最终的A和B。一般来说，如果 $(b_{\text{svm}}, \mathbf{w}_{\text{svm}})$ 较为合理的话，满足 $A > 0$ 且 $B \approx 0$ 。

$$g(\mathbf{x}) = \theta(A \cdot (\mathbf{w}_{\text{svm}}^T \Phi(\mathbf{x}) + b_{\text{svm}}) + B)$$

- **SVM flavor**: fix hyperplane direction by  $\mathbf{w}_{\text{svm}}$ —kernel applies
- **LogReg flavor**: fine-tune hyperplane to match maximum likelihood by scaling (A) and shifting (B)
  - often  $A > 0$  if  $\mathbf{w}_{\text{svm}}$  reasonably good
  - often  $B \approx 0$  if  $b_{\text{svm}}$  reasonably good

那么，新的logistic regression表达式为：

new LogReg Problem:

$$\min_{A, B} \frac{1}{N} \sum_{n=1}^N \log \left( 1 + \exp \left( -y_n \left( A \cdot \underbrace{(\mathbf{w}_{\text{svm}}^T \Phi(\mathbf{x}_n) + b_{\text{svm}})}_{\Phi_{\text{svm}}(\mathbf{x}_n)} + B \right) \right) \right)$$

这个表达式看上去很复杂，其实其中的 $(b_{\text{svm}}, \mathbf{w}_{\text{svm}})$ 已经在SVM中解出来了，实际上的未知参数只有A和B两个。归纳一下，这种Probabilistic SVM的做法分为三个步骤：

## Platt's Model of Probabilistic SVM for Soft Binary Classification

- 1 run **SVM** on  $\mathcal{D}$  to get  $(b_{\text{SVM}}, \mathbf{w}_{\text{SVM}})$  [or the equivalent  $\alpha$ ], and transform  $\mathcal{D}$  to  $\mathbf{z}'_n = \mathbf{w}_{\text{SVM}}^T \Phi(\mathbf{x}_n) + b_{\text{SVM}}$   
—actual model performs this step in a more complicated manner
- 2 run **LogReg** on  $\{(\mathbf{z}'_n, y_n)\}_{n=1}^N$  to get  $(A, B)$   
—actual model adds some special regularization here
- 3 return  $g(\mathbf{x}) = \theta(A \cdot (\mathbf{w}_{\text{SVM}}^T \Phi(\mathbf{x}) + b_{\text{SVM})) + B)$

这种soft binary classifier方法得到的结果跟直接使用SVM classifier得到的结果可能不一样，这是因为我们引入了系数A和B。一般来说，soft binary classifier效果更好。至于logistic regression的解法，可以选择GD、SGD等等。

## Kernel Logistic Regression

上一小节我们介绍的是通过kernel SVM在z空间中求得logistic regression的近似解。如果我们希望直接在z空间中直接求解logistic regression，通过引入kernel，来解决最优化问题，又该怎么做呢？SVM中使用kernel，转化为QP问题，进行求解，但是logistic regression却不是个QP问题，看似好像没有办法利用kernel来解决。

我们先来看看之前介绍的kernel trick为什么会work，kernel trick就是把z空间的内积转换到x空间中比较容易计算的函数。如果w可以表示为z的线性组合，即

$\mathbf{w}_* = \sum_{n=1}^N \beta_n \mathbf{z}_n$ 的形式，那么乘积项

$\mathbf{w}_*^T \mathbf{z} = \sum_{n=1}^N \beta_n \mathbf{z}_n^T \mathbf{z} = \sum_{n=1}^N \beta_n K(\mathbf{x}_n, \mathbf{x})$ ，即其中包含了z的内积。也就是w可以表示为z的线性组合是kernel trick可以work的关键。

我们之前介绍过SVM、PLA包扩logistic regression都可以表示成z的线性组合，这也提供了一种可能，就是将kernel应用到这些问题中去，简化z空间的计算难度。

SVM	PLA	LogReg by SGD
$\mathbf{w}_{\text{SVM}} = \sum_{n=1}^N (\alpha_n y_n) \mathbf{z}_n$ $\alpha_n$ from <b>dual solutions</b>	$\mathbf{w}_{\text{PLA}} = \sum_{n=1}^N (\alpha_n y_n) \mathbf{z}_n$ $\alpha_n$ by <b># mistake corrections</b>	$\mathbf{w}_{\text{LOGREG}} = \sum_{n=1}^N (\alpha_n y_n) \mathbf{z}_n$ $\alpha_n$ by <b>total SGD moves</b>

有这样一个理论，对于L2-regularized linear model，如果它的最小化问题形式为如下的话，那么最优解 $\mathbf{w}_* = \sum_{n=1}^N \beta_n \mathbf{z}_n$ 。



claim: for any L2-regularized linear model

$$\min_{\mathbf{w}} \quad \frac{\lambda}{N} \mathbf{w}^T \mathbf{w} + \frac{1}{N} \sum_{n=1}^N \text{err}(y_n, \mathbf{w}^T \mathbf{z}_n)$$

optimal  $\mathbf{w}_* = \sum_{n=1}^N \beta_n \mathbf{z}_n$ .

下面给出简单的证明，假如最优解  $\mathbf{w}_* = \mathbf{w}_{\parallel} + \mathbf{w}_{\perp}$ 。其中， $\mathbf{w}_{\parallel}$  和  $\mathbf{w}_{\perp}$  分别是平行  $\mathbf{z}$  空间和垂直  $\mathbf{z}$  空间的部分。我们需要证明的是  $\mathbf{w}_{\perp} = \mathbf{0}$ 。利用反证法，假如  $\mathbf{w}_{\perp} \neq \mathbf{0}$ ，考虑  $\mathbf{w}_*$  与  $\mathbf{w}_{\parallel}$  的比较。第一步先比较最小化问题的第二项：

$\text{err}(y, \mathbf{w}_*^T \mathbf{z}_n) = \text{err}(y_n, (\mathbf{w}_{\parallel} + \mathbf{w}_{\perp})^T \mathbf{z}_n) = \text{err}(y_n, \mathbf{w}_{\parallel}^T \mathbf{z}_n)$ ，即第二项是相等的。然后第二步比较第一项： $\mathbf{w}_*^T \mathbf{w}_* = \mathbf{w}_{\parallel}^T \mathbf{w}_{\parallel} + 2\mathbf{w}_{\parallel}^T \mathbf{w}_{\perp} + \mathbf{w}_{\perp}^T \mathbf{w}_{\perp} > \mathbf{w}_{\parallel}^T \mathbf{w}_{\parallel}$ ，即  $\mathbf{w}_*$  对应的 L2-regularized linear model 值要比  $\mathbf{w}_{\parallel}$  大，这就说明  $\mathbf{w}_*$  并不是最优解，从而证明  $\mathbf{w}_{\perp}$  必然等于零，即  $\mathbf{w}_* = \sum_{n=1}^N \beta_n \mathbf{z}_n$  一定成立， $\mathbf{w}_*$  一定可以写成  $\mathbf{z}$  的线性组合形式。

- let optimal  $\mathbf{w}_* = \mathbf{w}_{\parallel} + \mathbf{w}_{\perp}$ , where  $\mathbf{w}_{\parallel} \in \text{span}(\mathbf{z}_n)$  &  $\mathbf{w}_{\perp} \perp \text{span}(\mathbf{z}_n)$   
— want  $\mathbf{w}_{\perp} = \mathbf{0}$
  - what if **not**? Consider  $\mathbf{w}_{\parallel}$ 
    - of same err as  $\mathbf{w}_*$ :  $\text{err}(y_n, \mathbf{w}_*^T \mathbf{z}_n) = \text{err}(y_n, (\mathbf{w}_{\parallel} + \mathbf{w}_{\perp})^T \mathbf{z}_n)$
    - of smaller regularizer as  $\mathbf{w}_*$ :  
 $\mathbf{w}_*^T \mathbf{w}_* = \mathbf{w}_{\parallel}^T \mathbf{w}_{\parallel} + 2\mathbf{w}_{\parallel}^T \mathbf{w}_{\perp} + \mathbf{w}_{\perp}^T \mathbf{w}_{\perp} > \mathbf{w}_{\parallel}^T \mathbf{w}_{\parallel}$
- $\mathbf{w}_{\parallel}$  ‘more optimal’ than  $\mathbf{w}_*$  (contradiction!)

经过证明和分析，我们得到了结论是任何 L2-regularized linear model 都可以使用 kernel 来解决。

现在，我们来看看如何把 kernel 应用在 L2-regularized logistic regression 上。上面我们已经证明了  $\mathbf{w}_*$  一定可以写成  $\mathbf{z}$  的线性组合形式，即  $\mathbf{w}_* = \sum_{n=1}^N \beta_n \mathbf{z}_n$ 。那么我们就无需一定求出  $\mathbf{w}_*$ ，而只要求出其中的  $\beta_n$  就行了。怎么求呢？直接将  $\mathbf{w}_* = \sum_{n=1}^N \beta_n \mathbf{z}_n$  代入到 L2-regularized logistic regression 最小化问题中，得到：

solving L2-regularized logistic regression

$$\min_{\mathbf{w}} \quad \frac{\lambda}{N} \mathbf{w}^T \mathbf{w} + \frac{1}{N} \sum_{n=1}^N \log(1 + \exp(-y_n \mathbf{w}^T \mathbf{z}_n))$$

yields optimal solution  $\mathbf{w}_* = \sum_{n=1}^N \beta_n \mathbf{z}_n$

with out loss of generality, can solve for **optimal  $\beta$**  instead of **w**

$$\min_{\beta} \frac{\lambda}{N} \sum_{n=1}^N \sum_{m=1}^N \beta_n \beta_m K(\mathbf{x}_n, \mathbf{x}_m) + \frac{1}{N} \sum_{n=1}^N \log \left( 1 + \exp \left( -y_n \sum_{m=1}^N \beta_m K(\mathbf{x}_m, \mathbf{x}_n) \right) \right)$$

—how? GD/SGD/... for **unconstrained optimization**

上式中，所有的w项都换成 $\beta_n$ 来表示了，变成了没有条件限制的最优化问题。我们把这种问题称为kernel logistic regression，即引入kernel，将求w的问题转换为求 $\beta_n$ 的问题。

从另外一个角度来看Kernel Logistic Regression ( KLR )：

$$\min_{\beta} \frac{\lambda}{N} \sum_{n=1}^N \sum_{m=1}^N \beta_n \beta_m K(\mathbf{x}_n, \mathbf{x}_m) + \frac{1}{N} \sum_{n=1}^N \log \left( 1 + \exp \left( -y_n \sum_{m=1}^N \beta_m K(\mathbf{x}_m, \mathbf{x}_n) \right) \right)$$

上式中log项里的 $\sum_{m=1}^N \beta_m K(\mathbf{x}_m, \mathbf{x}_n)$ 可以看成是变量 $\beta$ 和 $K(\mathbf{x}_m, \mathbf{x}_n)$ 的内积。上式第一项中的 $\sum_{n=1}^N \sum_{m=1}^N \beta_n \beta_m K(\mathbf{x}_n, \mathbf{x}_m)$ 可以看成是关于 $\beta$ 的正则化项 $\beta^T K \beta$ 。所以，KLR是 $\beta$ 的线性组合，其中包含了kernel内积项和kernel regularizer。这与SVM是相似的形式。

但值得一提的是，KLR中的 $\beta_n$ 与SVM中的 $\alpha_n$ 是有区别的。SVM中的 $\alpha_n$ 大部分为零，SV的个数通常是比较少的；而KLR中的 $\beta_n$ 通常都是非零值。

## 总结

本节课主要介绍了Kernel Logistic Regression。首先把Soft-Margin SVM解释成Regularized Model，建立二者之间的联系，其实Soft-Margin SVM就是一个L2-regularization，对应着hinge error measure。然后利用它们之间的相似性，讨论了如何利用SVM的解来得到Soft Binary Classification。方法是先得到SVM的解，再在logistic regression中引入参数A和B，迭代训练，得到最佳解。最后介绍了Kernel Logistic Regression，证明L2-regularized logistic regression中，最佳解 $w_*$ 一定可以写成z的线性组合形式，从而可以将kernel引入logistic regression中，使用kernel思想在z空间直接求解L2-regularized logistic regression问题。

**注明：**

文章中所有的图片均来自台湾大学林轩田《机器学习技法》课程