作业 1 采用不同核函数的 SVM 二分类器

19335015 陈恩婷

1. SVM 模型的一般理论

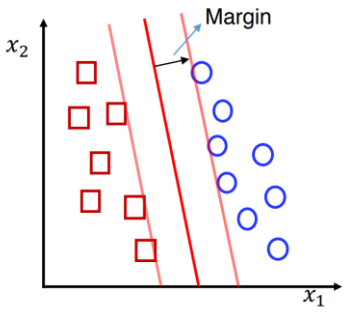
引自 MATLAB 官方文档：

> “当数据正好有两个类时，可以使用支持向量机（SVM）。SVM 通过找到将一个类的所有数据点与另一个类的所有数据点分离的最佳超平面对数据进行分类。SVM 的最佳超平面是指使两个类之间的边距最大的超平面。边距是指平行于超平面的内部不含数据点的平板的最大宽度。”

在线性分类器中，决策边界是一个超平面，其方程是

$$\{\boldsymbol{x}|\boldsymbol{w}^{*\boldsymbol{T}}\boldsymbol{x}+b^{*}=0\}$$

为了使分类器在未见过的数据上有较好的性能，需要找到一个让 margin 最大的超平面



可以通过求解以下优化问题来求出最佳超平面：

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

$$s.t.: y^{(l)}\cdot\left(\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{x}^{(l)}+b\right)\geq 1, for\ l=1,2,\dots,N$$

其最优解可以对向量进行如下分类：

$$\hat{y}(\boldsymbol{x})=sign(\boldsymbol{w}^{*\boldsymbol{T}}\boldsymbol{x}+b^{*})$$

对偶问题

为了求解以上优化问题，可以将其先转化为对偶问题。其拉格朗日函数如下：

$$L(\boldsymbol{w}, b, \boldsymbol{a}) = \frac{\mathbf{1}}{\mathbf{2}}\|\boldsymbol{w}\|^2 - \sum_{l=1}^{N} a_l(y^{(l)} \cdot \left(\boldsymbol{w^T x^{(l)}} + b\right) - 1)$$

拉格朗日对偶函数如下：

$$g(\boldsymbol{a}) = \min_{\boldsymbol{w},b} L(\boldsymbol{w}, b, \boldsymbol{a})$$

为了计算平稳点，令该函数的梯度等于零，得：

$$\boldsymbol{w} = \sum_{l=1}^{N} a_l y^{(l)} x^{(l)} \qquad \sum_{l=1}^{N} a_l y^{(l)} = 0$$

带入拉格朗日函数得到：

$$g(\boldsymbol{a}) = \sum_{l=1}^{N} a_l - \frac{1}{2}\sum_{l=1}^{N}\sum_{j=1}^{N} a_l a_j y^{(l)} y^{(j)} x^{(l)T} x^{(j)}$$

所以对偶问题为：

$$\max_{\boldsymbol{a}} g(\boldsymbol{a})$$

$$s.t.: \boldsymbol{a} \geq \mathbf{0}\ and \sum_{l=1}^{N} a_l y^{(l)} = 0$$

用二次规划方法求出$\boldsymbol{a}^*$后，就可以求出$\boldsymbol{w}^*$：

$$\boldsymbol{w}^* = \sum_{l=1}^{N} a_l^* y^{(l)} \boldsymbol{x}^{(l)}$$

对于边界上的所有样本$\left(\boldsymbol{x}^{(l)}, y^{(l)}\right)$，有$y^{(l)} \cdot \left(\boldsymbol{w^{*T} x^{(l)}} + b\right) = 1$，可得

$$b^* = \frac{1}{N_s}\sum_{n \in S}^{N}(y^{(l)} - \sum_{m} a_m^* y^{(m)} \boldsymbol{x}^{(n)T} \boldsymbol{x}^{(m)})$$

求出$\boldsymbol{w}^*$，b^*后，分类器就变为

$$\hat{y}(\boldsymbol{x}) = sign(\sum_{n=1}^{N} a_n^* y^{(n)}\, \boldsymbol{x}^{(n)T} \boldsymbol{x} + b^*)$$

这样就是一个简单的 SVM。

2. 采用不同核函数的模型和性能比较及分析

一些二分类问题没有办法用简单的超平面进行分类，但我们可以通过非线性化 SVM 模型解决这类问题，也就是说，将特征x通过 basis 函数映射到另一个空间：

$$\boldsymbol{\phi}: \boldsymbol{x} \rightarrow \boldsymbol{\phi}(\boldsymbol{x})$$

最大边距的优化问题就变成

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|^2$$
$$s.t.: y^{(l)} \cdot \left(\boldsymbol{w}^T \boldsymbol{\phi}\left(\boldsymbol{x}^{(l)}\right) + b\right) \geq 1, for\ l = 1, 2, \ldots, N$$

分类器：

$$\hat{y}(\boldsymbol{x}) = sign(\boldsymbol{w}^{*T}\boldsymbol{\phi}(\boldsymbol{x}^{(n)}) + b^*)$$

相应的对偶问题就变成

$$\max_{\boldsymbol{a}} g(\boldsymbol{a})$$

$$s.t.: \boldsymbol{a} \geq \boldsymbol{0}\ and \sum_{l=1}^{N} a_l y^{(l)} = 0$$

其中

$$g(\boldsymbol{a}) = \sum_{l=1}^{N} a_l - \frac{1}{2}\sum_{l=1}^{N}\sum_{j=1}^{N} a_l a_j y^{(l)} y^{(j)} \boldsymbol{\phi}\left(\boldsymbol{x}^{(l)}\right)^T \boldsymbol{\phi}(\boldsymbol{x}^{(j)})$$

核函数就是

$$k(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{\phi}(\boldsymbol{x})^T \boldsymbol{\phi}(\boldsymbol{x}')$$

考虑线性和函数和高斯核函数，其表达式和性能对比如下：

	线性核函数	高斯核函数
表达式	$k(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{x}^T \boldsymbol{x}'$	$k(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\|\boldsymbol{x} - \boldsymbol{x}'\|^2\right)$
MATLAB 训练代码	`SVMModel = fitcsvm(Train, "label",'KernelFunction', "linear",'KernelScale', 'auto');`	`SVMModel = fitcsvm(Train, "label",'KernelFunction', "gaussian",'KernelScale', 'auto');`

训练后的模型在测试集性能	>> accuracy accuracy = 0.999527186761229	>> accuracy accuracy = 0.999527186761229

可以看到，通过调用 MATLAB 库函数训练 SVM 模型，采用两种核函数的模型都得到了很好的正确率。

3. 采用hinge loss 的线性分类模型和 SVM 模型的关系

有的时候数据集并不存在一个分离超平面。在这种情况下可以使用软边距，对应的优化问题如下：

$$\min_{\boldsymbol{w},b,\xi} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{n=1}^{N}\xi_n$$
$$s.t.: y^{(n)} \cdot \left(\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{x}^{(n)} + b\right) \geq 1 - \xi_n,$$
$$\xi_n \geq 0, \text{for } n = 1, 2, \dots, N$$

将约束条件代入目标函数，得：

$$L(\boldsymbol{w}, b) = C\sum_{n=1}^{N} max\left(0, 1 - y^{(n)} \cdot \left(\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{x}^{(n)} + b\right)\right) + \frac{1}{2}\|\boldsymbol{w}\|^2$$
$$= C\sum_{n=1}^{N} E_{SV}\left(y^{(n)}h^{(n)}\right) + \frac{1}{2}\|\boldsymbol{w}\|^2$$
$$= \sum_{n=1}^{N} E_{SV}\left(y^{(n)}h^{(n)}\right) + \lambda\|\boldsymbol{w}\|^2$$

其中$E_{SV}(z) = \max(0, 1 - z)$就是 Hinge Loss。可见使用软边距的 SVM 的损失函数与 Hinge Loss 密切相关。

4. 采用 hinge loss 线性分类模型和 cross-entropy loss 线性分类模型比较

	Hinge loss	Cross-entropy loss
表达式	$L(\boldsymbol{w}) = \max(0, 1 - \mathrm{y}\mathbf{x}\mathbf{w})$	$L(\boldsymbol{w}) = -y\log(\sigma(\boldsymbol{x}\boldsymbol{w})) - (1-y)\log(\sigma(\boldsymbol{x}\boldsymbol{w}))$

代码	`function loss = hinge_loss(x, w, y)`
	`loss = max(0, 1-y*x*w);`
	`end`
	`function gradient = gradient_hinge_loss(x, w, y)`
	`if y*x*w<1`
	`  gradient = -y*x';`
	`else`
	`  gradient = zeros(size(w));`
	`end`
	`end`
		`function loss = cross_entropy(x, w, y)`
		`if y == 0`
		`  loss = -log(1-sigmoid(x*w));`
		`else`
		`  loss = -log(sigmoid(x*w));`
		`end`
		`end`
		`function gradient = gradient_cross_entropy(x, w, y)`
		`gradient = (sigmoid(x*w)-y)*x';`
		`end`
		`function y = sigmoid(x)`
		`y = 1/(1+exp(-x));`
		`end`

目标标签	0，1	-1，1
训练结果	测试集上正确率达到 1	测试集上正确率达到 1

可见不管是Hinge Loss还是Cross-entropy都在本实验的应用中取得了很好的训练效果。

关于超参数设置和训练方法等问题会在下一节讨论。

5. 训练过程（包括初始化方法、超参数选择、用到的训练技巧等）

	SVM	Hinge Loss 线性分类器	Cross-entropy 线性分类器
初始化方法	调用库函数，无需设置	`w = rand(size(X_Train, 2), 1)*0.01;`	`w = rand(size(X_Train, 2), 1)*0.001;`
超参数选择	采用默认设置	`epochs = 30;` `r = 0.000002;`	`epochs = 25;` `r = 0.000002;`
训练技巧	分别设置核函数为 linear 和 gaussian	梯度下降法	梯度下降法
其他设置	设置核尺度 (KernelScale) 参数为 auto	无	无

6. 实验结果、分析及讨论

6.1 实验结果

从前面的几小节可以看到，SVM 和线性分类器在测试集上都取得了很好的训练效果，其中 SVM 在 linear 和 gaussian 两种核函数上都达到了 2114/2115 的正确率，采用 hinge loss 和 cross-entropy 线性分类器更是达到了 2115/2115 的正确率。可见这几种分类方法都很适合本实验的任务。

6.2 分析及讨论

6.2.1 关于训练过程中预测结果和正确率出现 NaN 或 inf 的问题

经过尝试和查找资料，主要的解决方法有：

(1) 调整初始的权值矩阵 w，乘以一个小于 1 的数字（0.01 等）；

(2) 对于交叉熵函数，由于涉及到取对数的操作，在预测标签接近 0 或 1 时容易出现问题，所以选择不采用原本的表达式$L(\boldsymbol{w}) = -y\log\big(\sigma(\boldsymbol{xw})\big) - (1-y)\log\left(\sigma(\boldsymbol{xw})\right)$，改成分段函数的形式：

```
if y == 0
    loss = -log(1-sigmoid(x*w));
else
    loss = -log(sigmoid(x*w));
end
```

(3) 如果预测时出现了 inf 的结果，在程序中将其过滤出来，对于此样本不进行梯度下降操作，同时直接把损失记为零，避免影响平均损失的计算。

6.2.2 关于调用 fitcsvm 并把核函数设为 gaussian 时，训练完识别率仍为 50%的问题

这个主要涉及到参数的设计，一开始尝试了很久也没有弄出来，后面参考了这篇文章的参数设置，添加了'KernelScale','auto'这一选项，才开始有训练效果。这个参数的在官方文档的解释如下：

> 核尺度参数，指定为以逗号分隔的对组，其中包含 'KernelScale' 和 'auto' 或正标量。软件将预测变量矩阵 X 的所有元素除以 KernelScale 的值。然后，软件应用适当的核范数来计算 Gram 矩阵。
>
> 如果您指定 'auto'，则软件使用启发式过程选择适当的尺度因子。

7. 参考链接

1. https://ww2.mathworks.cn/help/stats/support-vector-machines-for-binary-classification.html
2. https://blog.csdn.net/qq_36108664/article/details/111117047
3. https://ww2.mathworks.cn/help/stats/fitcsvm.html