SVM

Mia Feng

2018 年 3 月 4 日

1 概述

SVM 用于求解分类超平面。不同于 Logistic Regression 用概率来描述数据所属类别，SVM 是基于几何距离的确定性方法。当数据近似线性可分时，模型可以采用软间隔最大化求解超平面。当数据线性不可分时，可以采用 kernel trick 和软间隔最大化求解超平面。所以 SVM 和 GPR 一样，都是 kernel machine 的一种。kernel trick 的使用使得 SVM 的适用范围更广。但是，SVM 适用于小数据集。

求解目标：分类超平面，以二维空间为例，求解一条直线

$$y = w^{\mathrm{T}} x + b \tag{1}$$

求解思路：求解 θ 基于几何间隔最大化。即，分类结果需要使得最难分的点（离分类超平面最近的点，亦即 support vector）也有足够大的信度将其分开。选用几何距离的原因是为了避免函数间隔的尺度影响

求解方法：SMO 算法。Rather than 对所有参数对全局优化（当维度很大时全局优化计算效率极低），SMO 每次只选取两个变量构建二次规划问题（子问题）。变量更新后，原始问题的目标函数值会更小。其实 SMO 每次只求解了一个变量，只是固定了其他变量后，一个变量求解更新，另一个值也随之确定而已。子问题的两个变量，一个是违反 KKT 条件最严重的那个，另一个由约束条件自动确定

1.1 推导

给定训练数据集 $T = \Big\{\big(x_i, y_i\big)\Big\}$ 和超平面 $\big(w, b\big)$

函数间隔

$$\hat{\gamma}_i = y_i\big(w \cdot x_i + b\big) \tag{2}$$

乘以类标 y_i 只是为了衡量分类结果。分正确此值为正，反之为负。则函数间隔的最小值就是分类结果最差的点所对应的函数间隔。所以，SVM 的关注点在于分类结果最差的那些点，亦即 support vector。记函数间隔最小值为 $\hat{\gamma}$，即 $\hat{\gamma} = \min\limits_{i=1,\dots,n} \hat{\gamma}_i$

几何间隔

$$\hat{\gamma}_i = y_i\big(\frac{w}{\|w\|} \cdot x_i + \frac{b}{\|w\|}\big) \tag{3}$$

参考点到直线的距离公式，其实函数间隔对应那个公式的分子部分。几何间隔对应点到直线的距离公式

关注最难分的点，使最难分的点被有效分开。则需要最大化最小几何间隔，由此，SVM 的目标函数为

$$\arg\max_{w,b} \left\{ \min_{n} y_i\big(\frac{w}{\|w\|} \cdot x_i + \frac{b}{\|w\|}\big) \right\} \tag{4}$$

假定函数间隔为 1，进一步简化之后，优化目标为

$$\begin{aligned} &\min_{w,b} \quad \tfrac{1}{2}\|w\|^2 \\ &s.t. \quad y_i\big(w \cdot x_i + b\big) - 1 \geq 0, \quad i = 1, 2, \cdots, n \end{aligned} \tag{5}$$

如下图所示，中间的实线便是寻找到的最优超平面（Optimal Hyper Plane），其到两条虚线的距离相等，这个距离便是几何间隔$\tilde{\gamma}$，两条虚线之间的距离等于$2\tilde{\gamma}$，而虚线上的点则是支持向量。由于这些支持向量刚好在边界上，所以它们满足$y(w^Tx+b)=1$（还记得我们把 functional margin 定为 1 了吗？上节中：处于方便推导和优化的目的，我们可以令$\hat{\gamma}=1$），而对于所有不是支持向量的点，则显然有$y(w^Tx+b)>1$。

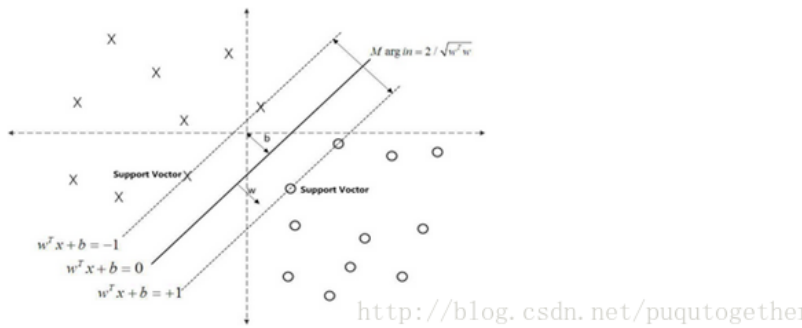

图 1: Hyperplane and support vector

构造拉格朗日函数求解对偶问题，从而得到原始问题的最优解。

$$\max_{\alpha}\min_{w,b}\mathcal{L}\big(w,b,\alpha\big)=\max_{\alpha}\min_{w,b}\frac{1}{2}\left\|w\right\|^2-\sum_{i=1}^{n}\alpha_i\Big(y_i\big(w\cdot x_i+b\big)-1\Big) \tag{6}$$

通过 KKT 条件和对偶问题求解（先求极小，导数为 0 后求解得到 w 和 b 的表达式，再代入原式即得对偶问题）由对偶问题可以很容易的引入核函数。

$$\begin{aligned}\min_{w,b}\mathcal{L}\big(w,b,\alpha\big)=\min_{w,b}\Bigg(&\tfrac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j x_i^{\mathrm{T}}x_j\\ &-\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j x_i^{\mathrm{T}}x_j-b\sum_{i=1}^{n}\alpha_i y_i+\sum_{i=1}^{n}\alpha_i\Bigg)\end{aligned} \tag{7}$$

对偶问题优化目标为

$$\begin{aligned}\min_{\alpha}\quad & \tfrac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j x_i^{\mathrm{T}}x_j-\sum_{i=1}^{n}\alpha_i\\ s.t.\quad & \sum_{i=1}^{n}\alpha_i y_i=0\\ & \alpha_i\geq 0,\quad i=1,2,\cdots,n\end{aligned} \tag{8}$$

解得 α 之后，就可以得到 w 和 b

$$
\begin{aligned}
w^* &= \sum_{i=1}^{n} \alpha_i^* y_i x_i \\
b^* &= y_j - \sum_{i=1}^{n} \alpha_i^* y_i \left(x_i \cdot x_j\right)
\end{aligned}
\tag{9}
$$

注意这里的下标索引 j 指的是任意一个令 $\alpha_j > 0$ 的下标，任取一个，计算出的 b^* 值是相同的。可以观察到，每一个样本都有一个拉格朗日乘子，当样本数增加的时候，需要求解的参数维度太大，很难优化。SMO 算法可以降低求解难度。

分离超平面：

$$
w^* \cdot x + b^* = 0 \tag{10}
$$

分类决策函数

$$
f\left(x\right) = \mathrm{sign}\left(w^* \cdot x + b^*\right) \tag{11}
$$

1.2 SMO

论文 [4]，具体推导参考 [3]，推导比李航的书上更详细

2 算法实现

见李航书 [1]

算法 7.5（SMO 算法）

输入：训练数据集 $T=\{(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)\}$，其中，$x_i \in \mathcal{X}=\mathbf{R}^n$，$y_i \in \mathcal{Y}=\{-1,+1\}$，$i=1,2,\cdots,N$，精度 ε；

输出：近似解 $\hat{\alpha}$.

（1）取初值 $\alpha^{(0)}=0$，令 $k=0$；

（2）选取优化变量 $\alpha_1^{(k)},\alpha_2^{(k)}$，解析求解两个变量的最优化问题 (7.101) ~ (7.103)，求得最优解 $\alpha_1^{(k+1)},\alpha_2^{(k+1)}$，更新 α 为 $\alpha^{(k+1)}$；

（3）若在精度 ε 范围内满足停机条件

$$\sum_{i=1}^{N}\alpha_i y_i=0$$

$$0\leqslant\alpha_i\leqslant C,\quad i=1,2,\cdots,N$$

$$y_i\cdot g(x_i)=\begin{cases}\geqslant 1, & \{x_i \mid \alpha_i=0\}\\ =1, & \{x_i \mid 0<\alpha_i<C\}\\ \leqslant 1, & \{x_i \mid \alpha_i=C\}\end{cases}$$

其中，

$$g(x_i)=\sum_{j=1}^{N}\alpha_j y_j K(x_j,x_i)+b$$

则转 (4)；否则令 $k=k+1$，转 (2)；

（4）取 $\hat{\alpha}=\alpha^{(k+1)}$.

图 2: SMO algorithm

实现中的一个小技巧：把 α_2 的索引值搜索范围设置在支持向量索引内，因为非支持向量对应的拉格朗日乘子 $\alpha=0$。详细推导见博客 [2]。

3 Implementation

线性核测试

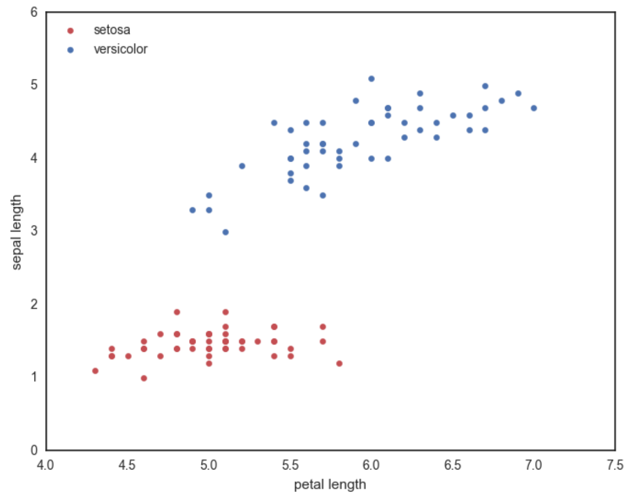

图 3: 鸢尾花数据

```
the 33-th iter is running
the 34-th iter is running
the 35-th iter is running
the 36-th iter is running
the 37-th iter is running
the 38-th iter is running
the 39-th iter is running
the 40-th iter is running
the 41-th iter is running
the 42-th iter is running
3 SupportVectors:

[ 4.8  1.9]
[ 5.1  1.9]
[ 5.1  3. ]
the #error_case is   0
0 errors
```

图 4: SVM 运行结果

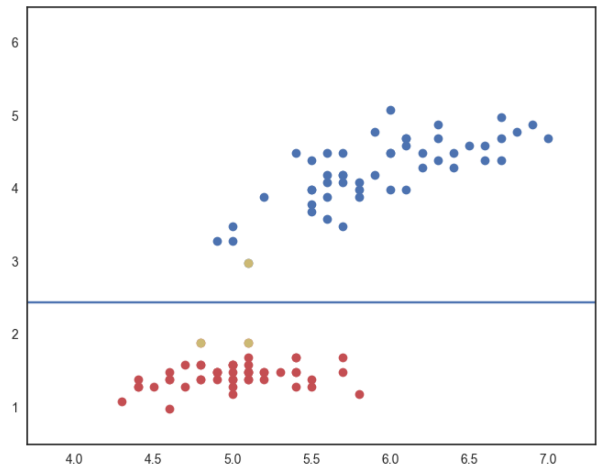

图 5: SVM 分类平面。黄色点为支持向量

参考文献

[1] 李航. 统计学习方法. 清华大学出版社, 2012.

[2] 风之忧伤. 支持向量机（svm）的详细推导过程及注解（一）. `http://blog.sina.com.cn/s/blog_4298002e010144k8.html`, 2012. Blog, 2012.

[3] July. 理解 svm 的三层境界. `http://blog.csdn.net/v_july_v/article/details/7624837`, 2012. Blog, 2012.

[4] John C. Platt. Sequential minimal optimization: A fast algorithm for training support vector machines. In *Advances in Kernel Methods-support Vector Learning*, pages págs. 212–223, 1998.