最优化大作业实验报告

15336140 庞泽雄

2018 年 1 月 26 日

摘要

本次大作业内容主要分为两部分，一是实现二次规划中的起始集法，也被称为有效集法；二是应用有效集算法实现 SVM（支持向量机），并与 MATLAB 中的 libsvm 函数库的运行结果进行比较。

其中有效集法的主要原理是在迭代点处，确定起作用约束的集合，然后将二次规划问题转化为等式约束最优化问题，然后根据 KKT 条件判断每一步的迭代点是否是最优解。若其不是最优解，求出迭代方向和迭代步长，进行迭代点的修正开始下一步迭代。

支持向量机是机器学习中的一个重要算法，其基本原理也是源自于二次规划的最优化方法，在本次的作业中，有两个数据集，我们将用二次规划的最优化算法对其进行求解，同时还会与其他的算法进行比较，分析 SVM 算法的优劣。

目录

1 等式约束的二次规划算法

等式约束的二次规划问题的一般形式为：

$$\min q(x) = \frac{1}{2}x^T Gx + h^T x \tag{1.1a}$$

$$\text{s.t. } A^T x = b \tag{1.1b}$$

1.1 变量消去法

将 x 的分量分成基本变量 x_B 与非基本变量 x_N 两部分，通过等式约束将基本变量用非基本变量线性标出；再将基本变量代入目标函数，从而消去基本变量，把问题化为一个关于非基本变量的无约束最优化问题；最后用求解无约束最优化问题的方法解之。

$$x = \begin{bmatrix} x_B \\ x_N \end{bmatrix} \; A = \begin{bmatrix} A_B \\ A_N \end{bmatrix}$$

$$G = \begin{bmatrix} G_{BB} & G_{BN} \\ G_{NB} & G_{NN} \end{bmatrix} \; h = \begin{bmatrix} h_B \\ h_N \end{bmatrix}$$

1.2 零空间方法

零空间方法又称为广义变量消去法，将 R_N 分成两个互补的子空间

$$R_N = R(A) \oplus N(A^T)$$

从两个子空间中选择 $Y, Z,$ 使得 $[YZ]$ 非奇异且满足

$$A^T Y = I \tag{1.2a}$$

$$A^T Z = 0 \tag{1.2b}$$

约束 $A^T x = b$ 转化为

$$A^T Y x_y + A^T Z x_z = b$$

问题（1.1）的可行点表示成

$$x = Yb + Zx_z \tag{1.3}$$

根据 KKT 条件可知，最优化问题唯一解为：

$$(Z^TGZ)x_z = -Z^T(h + GYb) \tag{1.4}$$

用 cholesky 分解求得 x_z^*, 得到最优解

$$x^* = Yb + Zx_z^*$$

相应的拉格朗日乘子 λ^* 为：

$$\lambda^* = Y^T(Gx^* + h)$$

而在本次实验中，关于选取 Y, Z 的方法我使用的是 QR 分解

$$A = Q\begin{bmatrix} R \\ 0 \end{bmatrix} = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix}\begin{bmatrix} R \\ 0 \end{bmatrix} = Q_1R$$

取 $Y = Q_1R^{-T}, Z = Q_2$

1.3 Lagrangian 方法

等式约束二次规划问题（1.1）的拉格朗日函数为：

$$L(x, \lambda) = \frac{1}{2}x^TGx + h^Tx - \lambda(A^Tx - b)$$

其 KKT 条件为：

$$Gx + h - A\lambda = 0$$

$$A^Tx - b = 0$$

可表示为：

$$\begin{bmatrix} G & -A \\ -A^T & 0 \end{bmatrix}\begin{bmatrix} x \\ \lambda \end{bmatrix} = -\begin{bmatrix} h \\ b \end{bmatrix} \tag{1.5}$$

利用初等变换，将 KKT 矩阵准三角化后，代入（1.5）式可以得到下面三个方程组

$$Gw = -h \tag{1.6a}$$

$$A^TG^{-1}A\lambda = -A^Tw + b \tag{1.6b}$$

$$Gx = A\lambda - h \tag{1.6c}$$

拉格朗日的具体算法如下：

LAGRANGIAN(G, h, A, b)

1 step 1 对矩阵 G 作 cholesky 分解，求得下三角阵 L
2 $G = LL^T$
3 step 2 计算 $V = A^T G^{-1} A$
4 step 3 $V = \tilde{L}\tilde{L}^T$
5 step 4 $Lu = -h, L^T w = u$
6 解得 $\tilde{b} = -A^T w + b$
7 step 5 $\tilde{L}v = \tilde{b}, \tilde{L}^T \lambda = v$
8 解得 $\tilde{h} = A\lambda^* - h$
9 step 6 $Ly = \tilde{h}, L^T x = y$
10 解得 x^*

2 起作用集方法（有效集法）

二次规划问题的一般形式为：

$$\min q(x) = \frac{1}{2} x^T G x + h^T x \tag{2.1}$$

$$\text{s.t. } a_i^T x = b_i, i \in \epsilon$$
$$a_i^T x = 0, i \in I$$

有效集方法是针对中小规模的凸二次规划问题提出的，主要原理是在每一步的迭代点处，把未有效的不等式约束剔除掉并把原命题转化成更易求解的等式约束命题，工作集可能与最优解的有效集相同，也可能不同。如果相同，我们可以通过计算对偶变量了解到此时已经是最优点从而退出迭代。如果不同，我们会对工作集进行更新，从现有工作集中删除一条约束或者增加一条新的约束到工作集。记工作集为 W_k。

假定在第 k 步，x_k 和相应的 W_k 已知，下面我们考虑在迭代点处，迭代方向 p 和步长 α_k 的计算及 W_k 的修正。我们定义：

$$p = x - x_K$$

$$g_k = G x_k + h$$

代入问题（2.1）中，我们得到每次迭代需要求解的等式约束最优化问题

$$\min_p q(x) = \frac{1}{2} p^T G p + g_k^T p \tag{2.2a}$$

$$\text{s.t. } a_i^T p = 0, i \in W_k \tag{2.2b}$$

如果 p = 0, 计算出 λ_k，如果满足以下条件

$$\lambda_i^T \geq 0, i \in I(x_k)$$

则 x_k 是二次规划问题（2.1）的 KKT 点；否则，求出工作集中删除的约束 q 并修正 W_k

$$\lambda_q = \min_{i \in I(x_k), \lambda_i^k < 0} \lambda_i^k$$

$$W_k := W_k \backslash \{q\}$$

如果 $p_k \neq 0$，则更新迭代点

$$x_{k+1} = x_k + \alpha_k p_k$$

其中 α_k 应该在 $[0, 1)$ 的最大值，使得在 x_{k+1} 处，所有约束满足，

$$\alpha_k = \min\{1, \min_{i \notin W_k, a_i^T p < 0} \frac{b_i - a_i^T x_k}{a_i^T p_k}\} \tag{2.3}$$

若在上式中确定一个约束，由不起作用约束变为起作用约束，我们设这个约束为 m，往工作集中添加约束

$$W_{k+1} := W_k \cup \{m\}$$

Lagrangian(G, h, A, b)

1 确认可行点 x_0, 进而确定初始工作集 W_0

2 step 1 根据（2.2）确定可行方向 p_k

3 step 2 若 $p_k = 0$，判断是否 $\lambda_i \geq 0$，如果是终止迭代；

4 不是的话，$\lambda_q = \min_{i \in I(x_k), \lambda_i^k < 0} \lambda_i^k$

5 $W_k := W_k \backslash \{q\}$，转 step 1

6 step 3 若 $p_k \neq 0$, 计算 $x_{k+1} = x_k + \alpha_k p_k$

7 其中 $\alpha_k = \min\{1, \min_{i \notin W_k, a_i^T p < 0} \frac{b_i - a_i^T x_k}{a_i^T p_k}\}$, 如果 $a \neq 1$ 转 step 4，否则转 step 1

8 step 4 $W_{k+1} := W_k \cup \{m\}$ 转 step 1

9 解得 x^*

3 支持向量机 (SVM)

3.1 SVM 的定义

首先，对于一个线性可分的数据集，我们可以通过一些特定的算法，得到一个超平面，把数据集中的元素分成两部分。对于同一个数据集而言，可以实现线性分隔的超平面有很多个，而且每个分类器的鲁棒性也不尽相同。因为我们的数据集一般都不是严格线性可分的，所以我们应该从中选取最好的分类器，使得它即使在噪声的影响下，也能够保证分类的正确性。

为了定义这样一个最优的超平面，我们介绍两个概念，函数距离（functional margin）和几何距离（geometric margin）。

数据样本相对于超平面的函数距离为：

$$\gamma_i = y_i(wx_i + b) \tag{3.1}$$

几何距离，即样本点到超平面的欧式距离

$$\hat{\gamma}_i = \frac{1}{\|w\|} y_i(wx_i + b) \tag{3.2}$$

显然我们得到下面的一个关系：

$$\gamma_i = \frac{1}{\|w\|} \hat{\gamma}_i \tag{3.3}$$

非常直观的，如果距离分类超平面（hyperlane）最近的数据点的几何距离越大，分类也就越精确，该分类器的噪声容忍度就越大，也就是越 robust。

所以分类器的目标就是在保证训练样本分类正确的前提下，最大化几何点到超平面最小欧式距离，因此，SVM 也被称为最大间隔分类器。

3.2 SVM 的标准形式

由（3.1）可知，当 w，b 乘以一个任意常数，函数距离也随之变化，但几何距离保持不变。我们可以通过一定的数值变化，使得最小函数距离为 1，最优分类器的最优化问题就可以定义为：

$$\max_{w,b} \hat{\gamma} \tag{3.4}$$

$$\text{s.t. } y_i(wx_i + b) \geq 1 \quad i = 1, 2, ...m$$

从（3.3）我们又知道最小几何距离实际等于 w 的模长的倒数，从而我们得到了与最优间隔分类器等价的凸二次规划问题

$$\min_{w,b} \frac{1}{2} w^T w \tag{3.5}$$

$$\text{s.t. } y_i(wx_i + b) \geq 1 \quad i = 1, 2, ...m$$

但是，数据集中总是会存在一些噪声极大的离群点，这些特殊的样本使得原本线性可分的数据集变成线性不可分，上述的分类器就特别容易受到离群点的影响。所以为了提高分类器的容错性，我们给硬性的函数距离引入一个松弛变量，同时给目标函数增加一个惩罚项，这种做法在机器学习中被称为正则化。改动之前的超平面称为硬间隔分类器，引入松弛变量后的分类器称为软间隔分类器。

这里，我们采用 $l1$ 正则化，得到一个一阶软间隔分类器：

$$\min_{w,b} \frac{1}{2} w^T w + C \sum_{i=1}^{m} \xi_i \tag{3.6}$$

$$\text{s.t. } y_i(wx_i + b) \geq 1 - \xi_i \quad i = 1, 2, ...m$$
$$\xi_i \geq 0 \quad i = 1, 2, ...m$$

其中惩罚因子 C 表示了你对离群点带来损失的重视程度，C 越大，分类的精度也越高，C 趋于无穷的时候，软间隔分类也就变成了硬分隔分类器。

3.3 SVM 的求解

显然，由 SVM 的标准形式可知，我们可以使用二次规划的方法来直接求解问题（3.5）和问题（3.6）。对于硬间隔分类器，我们有：

$$u = \begin{bmatrix} w_1 \\ \vdots \\ w_n \\ b \end{bmatrix} \qquad G = \begin{bmatrix} I_n & 0 \\ 0 & 0 \end{bmatrix} \qquad h = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$$

$$A = \begin{bmatrix} a_1 & a_2 & \cdots & a_n \end{bmatrix} \qquad a_n^T = y_n \begin{bmatrix} x_n^T & 1 \end{bmatrix}$$

而对于软间隔分类器，则：

$$u = \begin{bmatrix} w_1 \\ \vdots \\ w_n \\ b \\ \xi_1 \\ \vdots \\ \xi_m \end{bmatrix} \qquad G = \begin{bmatrix} I_n & 0 \\ 0 & 0 \end{bmatrix} \qquad h = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ C \\ \vdots \\ C \end{bmatrix}$$

$$A = \begin{bmatrix} A_0 & I_m \\ 0 & I_m \end{bmatrix} \qquad A_0 = \begin{bmatrix} a_1 & a_2 & \cdots & a_n \end{bmatrix} \qquad a_n^T = y_n \begin{bmatrix} x_n^T & 1 \end{bmatrix}$$

把上述的系数代入到我们所实现的二次规划算法中，便可求得 SVM 的超平面分类器

4 实验分析与结果

4.1 课本作业题

本次大作业的前三题是课本上的课后练习题，要求编写求解等式约束二次规划问题的程序以及有效集方法解二次规划问题的程序，求出相应问题的最优解。

4.1.1 问题一

$$\min f(x) = (x_1 - 1)^2 + (x_2 - x_3)^2 + (x_4 - x_5)^2$$

$$\begin{aligned} \text{s.t. } & x_1 + x_2 + x_3 + x_4 + x_5 - 5 = 0 \\ & x_3 - 2(x_4 + x_5) + 3 = 0 \end{aligned}$$

转化成类似问题（2.1）的二次规划形式

$$G = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & -2 & 0 & 0 \\ 0 & -2 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & -2 \\ 0 & 0 & 0 & -2 & 2 \end{bmatrix} \qquad h = \begin{bmatrix} -2 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$A^T = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & -2 & -2 \end{bmatrix} \qquad b = \begin{bmatrix} 5 \\ -3 \end{bmatrix}$$

由上可知，问题一只有等式约束，所以只需要直接用等式约束的二次规划最优化方法求解即可。又因为矩阵 G 半正定，$Lagrangian$ 方法求解会出现矩阵奇异的情况，所以问题一采用的算法是零空间方法，QR 分解实现。

求解的结果为：

$$x^* = (1,1,1,1,1)^T$$

$$\lambda^* = (0,0)^T$$

$$f(x^*) = 0$$

运行时间为：$t = 0.015625$

4.1.2 问题二

$$\min f(x) = 9 - 8x_1 - 6x_2 - 4x_3 + 2x_1^2 + 2x_2^2 + x_3^2 + 2x_1x_2 + 2x_1x_3$$

$$\text{s.t. } 3 - x_1 - x_2 - 2x_3 \geq 0$$

$$x_i \geq 0, i = 1,2,3$$

转化成相应的二次规划问题形式：

$$G = \begin{bmatrix} 4 & 2 & 2 \\ 2 & 4 & 0 \\ 2 & 0 & 2 \end{bmatrix} \qquad h = \begin{bmatrix} -8 \\ -6 \\ -4 \end{bmatrix}$$

$$A^T = \begin{bmatrix} -1 & -1 & -2 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad b = \begin{bmatrix} -3 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

矩阵 G 正定，等式约束二次规划最优化算法采用 $Lagrangian$ 算法，约束条件中存在不等式约束，初始点 $x_0 = (0.5, 0.5, 0.5)^T$ 为可行点，调用有效集方法进行求解。

每次迭代求解等式约束的二次规划问题结果如下：

迭代次数	x_k	λ_k	p_k	W_k	f_k
1	$(0.5, 0.5, 0.5)^T$	$(0,0,0,0)^T$	$(0.5, 0.5, 0.5)^T$	$\emptyset$	2.25
2	$(0.75, 0.75, 0.75)^T$	$(1,0,0,0)^T$	$(0.583333, 0.027778, -0.305556)$	$\{1\}$	0.5625
3	$(\frac{4}{3}, \frac{7}{9}, \frac{4}{9})^T$	$(\frac{2}{9}, 0, 0, 0)$	$(0,0,0)^T$	$\{1\}$	0.1111

求解的结果为：

$$x^* = (\frac{4}{3}, \frac{7}{9}, \frac{4}{9})^T$$
$$\lambda^* = (0.222, 0, 0, 0)^T$$
$$f(x^*) = 0.1111111$$

总迭代次数为：3
运行时间为：$t = 0.156250$

4.1.3 问题三

$$\min f(x) = x_1 - x_2 - x_3 - x_1x_3 + x_1x_4 + x_2x_3 - x_2x_4$$

$$\begin{aligned}
\text{s.t. } & 8 - x_1 - 2x_2 \geq 0 \\
& 12 - 4x_1 - x_2 \geq 0 \\
& 12 - 3x_1 - 4x_2 \geq 0 \\
& 8 - 2x_3 - x_4 \geq 0 \\
& 8 - x_3 - 2x_4 \geq 0 \\
& 5 - x_3 - x_4 \geq 0 \\
& x_i \geq 0, i = 1, 2, 3, 4
\end{aligned}$$

转化为二次规划的形式

$$G = \begin{bmatrix} 0 & 0 & -1 & 1 \\ 0 & 0 & 1 & -1 \\ -1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \end{bmatrix} \qquad h = \begin{bmatrix} 1 \\ -1 \\ -1 \\ 0 \end{bmatrix}$$

$$A^T = \begin{bmatrix} -1 & -2 & 0 & 0 \\ -4 & -1 & 0 & 0 \\ -3 & -4 & 0 & 0 \\ 0 & 0 & -2 & -1 \\ 0 & 0 & -1 & -2 \\ 0 & 0 & -1 & -1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad b = \begin{bmatrix} -8 \\ -12 \\ -12 \\ -8 \\ -8 \\ -5 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

显然矩阵 G 不定，且约束条件数大于 x 的维数，所以不能采用 $Lagrangian$ 方法和零空间法。同时问题三也不是凸二次规划问题，所以不用起作用集方法进行求解。

作为代替，使用了 $Matlab$ 自带的库函数 $fmincon$，通过序列二次规划算法（SQP）和 $BFGS$ 迭代算法求解问题。

求解的结果为：

$$x^* = (0,3\ 0,4)^T$$

$$f(x^*) = -27$$

运行时间为：$t = 0.187500$

4.2 SVM 求解问题

本次大作业中用于检验 SVM 算法的数据集共有两个，为了更好验证其效果，特地把 $libsvm$（台湾大学林智仁教授等开发设计的一个简单、易于使用和快速有效的 SVM 模式识别与回归的软件包）的运行结果作为参考，以下则是本次实验的基本情况。

4.2.1 数据集（australian）

该数据集训练样本有 552 条，测试样本有 138 条，其中特征维数为 14。对于硬间隔分类器而言，二次规划问题的维数为 15，而对于软间隔分类器，维数则是 567。

分类结果为：

方法	iteration	训练集精度	测试集精度
Libsvm	9841	85.6884%	85.5072%
SVM(硬间隔)	15	67.03%	71.01%
SVM(软间隔)	24	85.6884%	85.5072%

4.2.2 数据集（asonar）

该数据集训练样本有 138 条，测试样本有 42 条，其中特征维数为 60。对于硬间隔分类器而言，二次规划问题的维数为 61，而对于软间隔分类器，维数则是 199。

分类结果为：

方法	iteration	训练集精度	测试集精度
Libsvm	94	78.3133%	80.9524%
SVM(硬间隔)	10	100%	71.43%
SVM(软间隔)	9	78.9157%	80.9524%

4.2.3 基线算法

本次实验采用的 *baseline* 为 *logisticregression* 和多元高斯生成模型,其中 *logisticregression* 也使用了 $l1$ 正则化。

logisticregression 的代价函数和相应的梯度为：

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}[-y^{(i)}log(h_\theta(x^{(i)})) - (1-y^{(i)})log(1-h_\theta(x^{(i)}))] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

$$\frac{\partial J}{\partial \theta_0} = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}, \text{for j} = 0$$

$$\frac{\partial J}{\partial \theta_j} = (\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}) + \frac{\lambda}{m}\theta_j, \text{for j} > 0$$

然后用梯度下降法进行最优化，并更新参数。而多元高斯概率函数密度为：

$$p(x, \mu, \sum) = \frac{1}{(2\pi)^{n/2}[\sum|}exp[-\frac{1}{2}(x-\mu)^T\sum^{-1}(x-\mu)$$

几种不同算法的比较如下：

算法	SVM 软间隔	$LogisticRegression$	多元高斯生成
数据集（australian）	85.5072%	67.39%	84.06%
数据集（sonar）	80.9524%	61.90%	76.19%

4.3 实验反思与总结

4.3.1 等式约束的二次规划算法

在课本第八章关于二次规划的最优化算法内容中，等式约束的算法共有三种，变量消去，零空间法和 *Lagrangian* 方法。而在本次实验中，我也只是实现了零空间和 *Lagrangian* 方法两种算法，主要原因是因为我没有找到关于如何实现变量消去的具体算法，课本上对于一些相关的推导也解释的比较含糊。

关于这一块内容的另外一个问题是，实现出来的算法程序鲁棒性不够。*e.g.*，在课后作业习题中，第一题二次规划形式的矩阵是半正定的，第三题则是不定的，因此 *Lagrangian* 算法难以求解，而零空间算法又有着特征维数应该小于约束条件数的前提限制，同样无法奏效。就是因为这一块的短板，导致有效集方法也没能在 SVM 的求解中发挥应有的作用。

4.3.2 SVM 及 *Baseline algorithm*

其实在这门课开始之前，我就开始接触机器学习这一方面的东西，所以到期中的时候，我对 SVM 这一些基础的算法都有大概的了解。在我个人看来，这些机器学习的基础算法，大部分都是以统计学模型作为基础，用最优化的方法求解模型，从而实现对某一类数据的分类或回归预测。所以说本学期的两门专业课都加深了我对这些算法的理解应用。一门是数值最优化，另一门则是数理统计。

正好最优化课程最后的大作业要求实现 SVM，借此机会，我就想把本学期课外学到的一些粗浅的东西加以应用，看看自己到底有什么收获，也算是对本学期的一个总结。

参考文献

1 高立. 数值最优化方法，2014

2 Nocedal J, Wright S J. Numerical Optimization Second Edition[J]. 1999

3 吴承恩，斯坦福大学 CS229（机器学习）网络公开课，2003