SVM 支持向量机算法实验报告

学号：PB21000224　　姓名：陈鸿绪　　日期：11.23.2023

一．支持向量机理论简介

支持向量机是一类按监督学习方式对数据进行二元分类的广义线性分类器，其决策边界是对学习样本求解的最大边距超平面。其按照功能作用的不同大致可以分成软间隔 SVM、硬间隔 SVM、带有核技巧的 SVM。在机器学习书籍里，书中详细推导了硬间隔 SVM 的求解，将其等价于求解一个二次规划问题，即书中式 6.6：

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

$$\text{s.t. } y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b) \geqslant 1, \quad i = 1, 2, \ldots, m.$$

求解以上二次规划问题可以有许多不同方式。本实验中采取了两种解决方式，一种是转化原来问题，通过求解其朗格朗日对偶问题：

$$\max_{\boldsymbol{\alpha}} \ \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j \boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j$$

$$\text{s.t.} \ \sum_{i=1}^{m}\alpha_i y_i = 0 \ ,$$

$$\alpha_i \geqslant 0 \ , \quad i = 1, 2, \ldots, m \ .$$

利用 SMO 算法，对该对偶问题进行求解。另外一种方式就是对原问题加上惩罚项构造损失函数进行梯度下降求解。

二．SMO 算法求解思路

SMO 算法的原理是把整个二次规划问题分解为很多易于处理的小问题，然后逐一求解。SMO 算法把问题分解到可能达到的最小规模：每次优化只处理两

个样本的优化问题，并且用解析的方法进行处理。这种最小优化的好处是可以避免迭代算法，从而大大提高求解效率。

由于实验中要求需要使用硬间隔，然而实验数据存在错标，即数据集不一定线性可分，所以满足一般 KKT 条件并不能保证得到优解，所以这里直接设置最大循环次数，超出即返回。笼统来说，SMO 算法通过以下步骤进行：

1. 从数据集中选择两个样本进行优化。
   启发式 SMO 算法会选择违反 KKT 条件最大的样本和与其距离最大的样本进行优化，为了更好凸显出错标数据点对于解的影响，实验中也采用了随机选取样本进行优化的 SMO 算法。
2. 对这两个样本的 Lagrange 乘子进行优化，以最大化间隔。这是通过求解一个二次规划问题来实现的。
3. 将得到的新的 Lagrange 乘子应用于整个数据集，更新分类决策函数。
4. 超出最大循环次数返回。

可以看见对于不同的选取样本方式会有不同的 SMO 算法，本次实验采取了主要硬间隔启发式 SMO 算法、硬间隔随机 SMO 算法。实验过程中发现了错标数据对于最后分类准确率影响很大，一旦选取到了错标样本进行迭代更新的时候，模型的准确率就会下降很多，迭代一定次数后，更新样本的选取对于模型的影响要远大于迭代次数，所以实验中 SVM1 类对应的 SMO 算法要求是：模型每一次的更新必须使得模型关于训练集准确率不能下降。如此就可以有效避免错标数据对于模型的干扰。

三．梯度下降法求解思路

对于书中 6.6 式，由于数据不一定线性可分，所以并不能保证其所有的线性约束都是成立的，所以不妨构造损失函数，通过梯度下降尽量最大化使得线性约束成立，同时也使得 6.6 目标函数值尽可能的小。梯度下降的损失函数构造采用对率损失函数，对于$y_i(w^T x_i + b)$而言，由于大于 0 则分类正确，小于 0 则分类错

误，所以直观上来说，$y_i(w^T x_i + b)$越小会使得对损失函数的贡献越大，所以构造的损失函数为：

$$f(w,b) = C\|w\|^2 + \sum_{i=1}^{m} \log(1 + \exp(-y_i(w^T x_i + b)))$$

这里 C 为模型参数。采用梯度下降方式求解，即：

$$w = w - learning\ rate \times \frac{\partial f}{\partial w}$$

$$b = b - learning\ rate \times \frac{\partial f}{\partial b}$$

最后在两者梯度小于一定精度时迭代退出。

四．代码说明

1. SFOT_SVM 类是采取 SMO 算法的软间隔 SVM。同时保证了每次迭代使模型对于训练集的准确率不会下降。

2. RAND_SVM 类是采取随机 SMO 算法的硬间隔 SVM。

3. SMO_HARD_SVM 类是采取启发式 SMO 算法的硬间隔 SVM。

4. SVM1 类是采取随机 SMO 算法的硬间隔 SVM，同时保证了每次迭代使模型对于训练集的准确率不会下降。

5. SVM2 类是采取的梯度下降算法的 SVM。

6. accuracy 函数用于度量模型预测准确率

五．实验结果与比较分析

采用 generate_data 函数产生数据集，采用维度 dim 为 20，数据个数为 1000。对应的错标数据的占比为 0.036，所以理论准确率上限应该为 0.964。为了更好的呈现实验结果，对于每一个基于 SMO 算法的模型不仅给出最后的准确率，而且会给出循环过程中的准确率。

1. RAND_SVM 类 运行时间 30.7s 最终准确率 0.893 总迭代次数 2000

迭代次数	100	200	300	400	500	600
准确率	0.725	0.885	0.858	0.802	0.893	0.829

700	800	900	1000	1100	1200	1300
0.907	0.776	0.736	0.907	0.796	0.843	0.869

1400	1500	1600	1700	1800	1900	2000
0.832	0.868	0.830	0.632	0.901	0.847	0.893

根据以上迭代情况可以发现，采取随机 SMO 算法的硬间隔 SVM 的在每一次迭代后不一定准确率会提升，有时反而会下降，这是由于数据集是非线性可分，有错标数据，SMO 算法有概率选取到了错标数据后，会导致原本准确率相对较高的超平面在调整之后倾向于错误数据，所以会导致超平面划分准确率下降。

2. SMO_HARD_SVM 类 运行时间 40 s 最终准确率 0.754 总迭代次数 10

迭代次数	1	2	3	4	5	6
准确率	0.686	0.807	0.658	0.859	0.688	0.754
样本A序号	991	428	488	218	488	440
样本B序号	518	94	973	168	973	10

7	8	9	10
0.685	0.754	0.685	0.754
488	440	488	440
973	10	973	10

由于启发式 SMO 算法寻找违背 KKT 条件程度最大的样本所需要的时间较长，所以迭代次数选择为 10，但并不妨碍看出这个算法最后落入只调整固定两对的循环，即（440，10）与（488，973）两对样本。

定性分析可知，我们找到的违背 KKT 条件程度最大的样本很大的概率会是错标数据，假设选取的那个错标数据 A 是正错标成了负，那么调整之后的超平面必然会趋向样本 A，调整之后违背 KKT 条件程度最大的几乎不可能是负样本，所以很

大概率是某个正样本 B 违背 KKT 条件，而这个正样本很大概率是错标的，即 B 负错标成了正，所以接下来选取样本会选取到 B，超平面又趋向于样本 B，不难发现接着迭代下去很可能会导致超平面在趋于正确分类 A 和正确分类 B 之间不断震荡，而准确率会一直保持在较低的水平，实例中准确率是在 0.7 附近震荡。此时准确率这时就和错标数据的偏离程度有莫大的关系，而非迭代次数有关。

为了验证以上的分析猜测，考察数据集中错标的数据序号集合，其中就包括了 488 序号样本和 440 序号样本，且恰好 488 是负错标成正，440 是正错标成负。

3. SVM1 类 运行时间 29.9 s 最终准确率 0.953 总迭代次数 2000

迭代次数	100	200	300	400	500	600
准确率	0.939	0.945	0.947	0.947	0.95	0.952

700	800	900	1000	1100	1200	1300
0.952	0.952	0.952	0.952	0.952	0.952	0.952

1400	1500	1600	1700	1800	1900	2000
0.953	0.953	0.953	0.953	0.953	0.953	0.953

在经过启发式 SMO 算法的试错后可以发现，SMO 最好的方法还是随机选取优化样本，因为错标的数据占比只有 0.036，大概率在随机选取优化样本对的时候是可以避开错标样本。同时为了避免错标样本带来的准确率下降，依托训练集，要求每次迭代准确率不能下降，这样就可以带来准确率的提升，为了时间代价尽量减少，令最大迭代次数 2000 次。

4. SVM2 类 运行时间 9.65s 最终准确率 0.954

SVM2 采用梯度下降算法，损失函数采取对率损失，一般在梯度下降求解 SVM 中并不能得到最优解。由于该数据集存在低程度的错标，所以本人猜测大概率是这一方面的原因导致 SVM 的梯度下降算法比 SMO 算法的时间性能和准确率都要高。

5. sklearn 模块中的支持向量机 最终准确率：0.953，运行时间为 447ms。该准确率和实验中写的 svm1 与 svm2 相匹配，但是时间代价远远低于 svm1 与 svm2 类。

比较以上实验中若干类对应的算法时间代价与准确率代价，同样使用 SMO 算法的类的时间代价差不多，而准确率只有 SVM1 可以达到较高水平，使用梯度下降算法的类时间代价和准确率均是最优，这并非说明梯度下降优于 SMO，而是完全和数据集错标程度有关。