高级机器学习
作业一

张逸凯 171840708

2020 年 11 月 8 日

学术诚信

本课程非常重视学术诚信规范，助教老师和助教同学将不遗余力地维护作业中的学术诚信规范的建立。希望所有选课学生能够对此予以重视。[1]

(1) 允许同学之间的相互讨论，但是**署你名字的工作必须由你完成**，不允许直接照搬任何已有的材料，必须独立完成作业的书写过程;

(2) 在完成作业过程中，对他人工作（出版物、互联网资料）中文本的直接照搬（包括原文的直接复制粘贴及语句的简单修改等）都将视为剽窃，剽窃者成绩将被取消。**对于完成作业中有关键作用的公开资料，应予以明显引用**；

(3) 如果发现作业之间高度相似将被判定为互相抄袭行为，**抄袭和被抄袭双方的成绩都将被取消**。因此请主动防止自己的作业被他人抄袭。

作业提交注意事项

(1) 请在LaTeX模板中**第一页填写个人的姓名、学号、邮箱信息**；

(2) 本次作业需提交该pdf文件、问题3可直接运行的源码(学号_.py)，将以上两个文件压缩成zip文件后上传。zip文件格式为**学号.zip**，例如170000001.zip；pdf文件格式为**学号_姓名.pdf**，例如170000001_张三.pdf。

(3) 未按照要求提交作业，或提交作业格式不正确，将会**被扣除部分作业分数**；

(4) 本次作业提交截止时间为**11月8日23:59:59**。除非有特殊情况（如因病缓交），否则截止时间后不接收作业，本次作业记零分。

[1] 参考尹一通老师高级算法课程中对学术诚信的说明。

1 [30pts] VC Dimensions

本题探讨 VC 维的性质。

(1) **[10pts]** 请在样本空间 $\mathcal{X} = [0, 1]$ 上构造一个有限假设空间 $\mathcal{H}$ 使得 $\mathrm{VC}(\mathcal{H}) = \lfloor \log_2(|\mathcal{H}|) \rfloor$.

(2) **[10pts]** 定义轴平行四边形概念类 $\mathcal{H} = \{h_{(a_1,a_2,b_1,b_2)}(x, y) : a_1 \le a_2 \wedge b_1 \le b_2\}$, 其中

$$h_{(a_1,a_2,b_1,b_2)}(x, y) = \begin{cases} 1 & \text{if } a_1 \le x \le a_2 \wedge b_1 \le y \le b_2 \\ 0 & \text{otherwise} \end{cases}$$

请证明 $\mathcal{H}$ 的 VC 维为 4.

(3) **[10 pts]** 请证明最近邻分类器的假设空间的 VC 维可以为无穷大.

Solution. 解:

- *(1) 构造:*

$$h_1 = \begin{cases} 1, & x < 0.5 \\ 0, & \text{其他} \end{cases}$$

$$h_2 = \begin{cases} 0, & x < 0.5 \\ 1, & \text{其他} \end{cases}$$

以及 $\forall x,\ h_3 = 1, \forall x,\ h_4 = 0.$

存在这样的集合: $x_1 = 0.1, x_2 = 0.9$, 对于任意对分均可被实现.

而对任意集合, 一定存在两个x都在某个半空间中, 若这两个x的*label*不一致, 则不能被实现.

所以如上假设空间有限, $VC(\mathcal{H}) = 2$.

实际上: 如果 $|\mathcal{H}| < \infty$, 则 $VC(\mathcal{H}) \le \log_2(|\mathcal{H}|)$, 反证法, 假设 $\log_2(|\mathcal{H}|) + 1$, 则所有对分有 $2^{(\log_2|\mathcal{H}|+1)} = 2|\mathcal{H}|$, 即矛盾, 仅有$|\mathcal{H}|$种方式.

- *(2)*

示例集大小为 *4* 时, 一共有2^4种对分, 如下列出了2^3种, 其他对分均可由下图中对分所有标签取反得到.

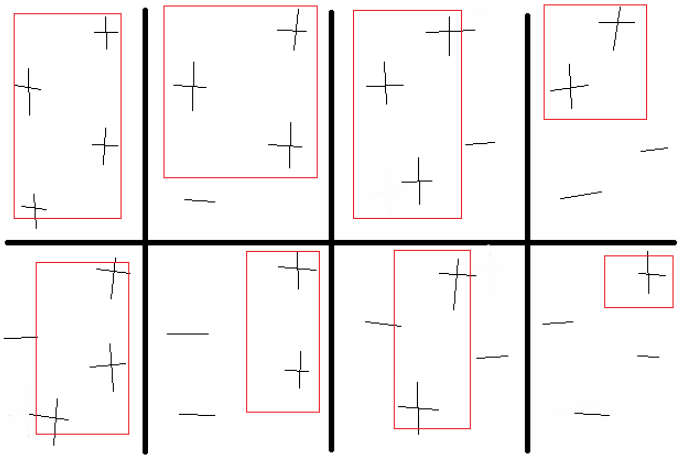


图 1: 实现假设空间的所有对分

这里构造示例集的方式为:

$$x_1 < x_2 < x_3 < x_4 \wedge y_1 < y_3 < y_2 < y_4$$

注意到, 示例集大小为5时, 构造对分: $\exists\ i, j,\ \{(x_i, -), (x_j, -), x_i < x_j\}$, 这给予了一个划分限制, **将$x$轴维度空间划分成了三个部分:**

$$x \leq x_i;\ x_i < x \leq x_j;\ x > x_j$$

在这三个部分指派**另外三个样本, 且标签为**$+$, 该对分不能被$\mathcal{H}$实现. 即不存在任何大小为5的示例集能被$\mathcal{H}$打散.

综上, $\mathcal{H}$ 的 VC 维为 4.

- *(3)*

即证明, $\mathcal{H}$ 能打散任意大小的示例集. 最近邻分类器的分类规则为距离测试集数据最近的那个$\mathcal{D}$标签. 由分类规则可知任一假设 h 由特定的训练集 $\mathcal{D}$ 所决定. 需证明: 假设空间$\mathcal{H}$能实现实例集D上的所有对分.

由于最近邻算法具有初始训练集, 并且标签已定, 反之不然, 因为如果所有标签都由一个数据生成, 则标签集为同一个标签, 由上基础:

$\forall$ 对分,$\forall$ 示例集样本个数n,有假设h(通过训练集定义, x 的最近邻为$Nearest(x)$) :

$D_{train} = \{x \mid y_{Nearest(x)} \neq y_x\}$

$D_{test} = D / D_{train}$

有上述构造, 可以发现对于任意大小的 D, $\mathcal{H}$ 能实现 D 上所有对分.

即 $\mathrm{VC}(\mathcal{H}) = \infty$.

$p(T=1)$	$p(T=2)$	$p(T=3)$
$\left(\frac{R+1}{2^N}\right)^2/C$	$\frac{R+1}{2^N}/C$	$1/C$
归一化因子: $C=\left(\frac{R+1}{2^N}\right)^2+\frac{R+1}{2^N}+1$		

2 [40pts] Understanding the Parameters in LVW from A Probabilistic Perspective

课程中，我们介绍了一种包裹式特征选择算法Las Vegas Wrapper（简称LVW），该算法的流程如教材中图11.1所示。首先请各位回顾一下LVW算法，接下来我们将对一种特殊情况下的LVW做一些分析，过程中复习一些简单的概率知识，并从更理性的角度理解LVW中的参数。

现在，我们获得了一个数据集D，该数据集中一共包含N个特征，设特征集合为$A=\{f_1,f_2,\cdots,f_N\}$。我们知道，对于某个具体任务，特征的质量参差不齐，高质量的特征会带来高质量的性能，低质量的特征会带来低质量的性能。我们设第i个特征的质量为2^{i-1}，即N个特征的质量分别为$1,2,4,\cdots,2^{N-1}$。我们规定：给定一个特征子集A'，学习器在A'上的性能恰好等于A'中包含的所有特征的质量之和。设特征子集A'中特征的质量之和记作$Q(A')$。

(1) **[5pts]** 现执行一次图11.1中第6行的语句，产生一个特征子集A'，试求数学期望$\mathbb{E}[Q(A')]$。

(2) **[10pts]** 现在，LVW算法已经执行了一段时间了，当前得到的最优特征子集为B，我们记$R=Q(B)$。设函数$better(n,r)$表示从前n个特征中随机产生一个特征子集且该特征子集的质量大于r的概率。试求$better(N,R)$。这里我们允许以递归式和递归边界来表达$better(N,R)$。

(3) **[10pts]** 从现在开始，我们设$better(n,r)$是一个已知函数。在LVW算法中，当连续T个随机生成的子集不比当前的最优子集更好时，算法结束。我们仍然设当前得到的最优特征子集为B，B的质量为$R=Q(B)$。在本小问中，我们还设T是一个已知参数。设布尔型随机变量e，$p(e=1|T)$表示经过恰好T次循环后LVW算法结束的概率，$p(e=0|T)$表示经过恰好T次循环后LVW算法没有结束的概率。试求分布$p(e|T)$。

(4) **[5pts]** 由第（3）小问可见，参数T越大，LVW算法结束就越（[回答“容易”或“困难”]）；当前最优子集质量越高，LVW算法结束就越（[回答“容易”或“困难”]）。

(5) **[10pts]** 现在，我们引入Bayes观点，认为参数T是服从某个先验分布$p(T)$的随机变量，且是未知的。我们仍然设当前得到的最优子集为B，B的质量为$R=Q(B)$。现在，LVW算法继续执行，在恰好执行了T个循环后成功退出了。如果我们设T的先验分布为整数区间$[1,10]$上的均匀分布，请对T做最大后验估计。这与最大似然估计的结果相同吗？为什么？

(6) **[Bonus 5pts]** 现在，设随机变量T只能在$\{1,2,3\}$中取值，且小明设置了一个先验$p(T)$。小红观察到，当前得到的最优子集为B，B的质量为$R=Q(B)$，且LVW在执行恰好T个循环后成功退出了，于是小红试图对T做最大后验估计。小红发现，T的后验概率在$\{1,2,3\}$上是均匀的，那么小明设置的先验$p(T)$有可能为：（填写一种可能的答案即可）

Solution. ***解:***

- *(1)*

  由题意以及期望的线性性质:

$$
\begin{aligned}
Q(A') &= \sum_{f_i \in A'} f_i = \sum_{f_i \in A'} 2^{i-1} \\
\text{又 } \mathbb{P}\{f_i \in A'\} &= \frac{1}{2} \\
\therefore \mathbb{E}[Q(A')] &= \sum_i \mathbb{P}\{f_i \in A'\} \cdot f_i \\
&= \frac{1}{2} \times \left(1 + \cdots + 2^{N-1}\right) \\
&= \frac{2^N - 1}{2}
\end{aligned}
$$

- *(2)*

  递归式*(*状态转移*)*, 根据随机选取中是否包含第n个特征:

$$better(n, r) = \frac{1}{2} \cdot better(n-1, r) + \frac{1}{2} \cdot better(n-1, r-2^{n-1})$$

  递归边界:

$$better(1, r) = \begin{cases} \frac{1}{2}, & r \le 1 \\ 0, & r > 1 \end{cases}$$

  – 这里给出非递归的解法:

  对于如下引理:

  > **myThm 1.** $\forall$ A' 是 A 的子集, 则 $Q(A')$ 是 A' 的某种二进制表达.
  > 例如$N = 4$时, 子集 $A' = \{f_2, f_4\}$, 则 $Q(A') = 1010_{(2)}$.

  其正确性是显然的.

  即对于特征总数为 N的情况, $Q(A') \le R$ 的子集有 $R+1$ 个, 即:

$$better(N, R) = 1 - \frac{R+1}{2^N}$$

- *(3)*

  一次循环后, 随机生成的子集 A' 不比当前的最优子集更好的概率:

$$\mathbb{P}\{\text{不比当前好的概率}\} = 1 - better(N, R)$$

  – 以下证明, 不存在生成的子集 A', $Q(A') = R$, 但 $d_{A'} < d_B$, 这等价于证明了书上图*11.1*第*9*行代码中的条件*2*在任意时刻不成立:

  $\because$ 每一种子集选择的 Q 为二进制表达. 且没有重复. *(*由 *1)*

由如上证明知 $\mathbb{P}\{\text{不比当前好的概率}\}$ 是正确的.

$$\begin{aligned}\therefore\ p(e=1\mid T)&=\mathbb{P}\{\text{不比当前好的概率}\}^T=(1-better(N,R))^T\\ p(e=0\mid T)&=1-(1-better(N,R))^T\end{aligned}$$

- *(4)*

  $T\uparrow$, 结束越困难.

  当前最优子集质量越高, 结束越容易.

- *(5)*

  T 的先验:

  $$\mathbb{P}\{T=i\}=\frac{1}{10},\ i\in\mathbb{Z}^+\cap[1,10]$$

  不妨令恰好执行T个循环后成功退出随机变量 $\xi\sim B(1,p(e=1\mid T))$, 不妨记 $p(e=1\mid T_i)$ 为 μ_{T_i}, 后验分布有如下相关性:

  $$\begin{aligned}\because\ P(T\mid\xi_1,\cdots,\xi_n)&\propto\pi(T)P(\xi_1,\cdots,\xi_n\mid T)\\ \therefore\ T_{MAP}&=\arg\max_T\mathbb{P}\{T=i\}\prod_{\xi_i\in\Xi}P(\xi_i\mid T)\\ &=\arg\max_T\sum_{\xi_i\in\Xi}\{\log P(\xi_i\mid T)\}+\log\frac{1}{10}\end{aligned}$$

  其中:

  $$\begin{aligned}\log P(\boldsymbol{\Xi}\mid T)&=\sum_{i=1}^{n}\log\left(\mu_{T_i}^{\xi_i}(1-\mu_{T_i})^{1-\xi_i}\right)\\ &=\sum_{i=1}^{n}\left[\xi_i\log\mu_{T_i}+(1-\xi_i)\log(1-\mu_{T_i})\right]\end{aligned}$$

  又

  $$\mu_{T_i}=p(e=1\mid T_i)=(1-better(N,R))^{T_i}=\left(\frac{R+1}{2^N}\right)^{T_i}$$

  请注意本题一次程序结束, 即观察量 ξ_i 只有一个, 并且为*1*, 代入上式并求导有:

  $$\xi log\left(\frac{R+1}{2^N}\right)+log2\cdot(1-\xi)\frac{-T\left(\frac{R+1}{2^N}\right)^{T-1}}{1-\left(\frac{R+1}{2^N}\right)^{T}}=0$$

  导函数小于零, $\xi=1\Rightarrow T_{MAP}=1$

  因为这里先验分布是均匀分布*(常值函数)*, *log*之后为加上常数, 所以最大后验估计与最大似然估计一致.

- *(6)*

$$\therefore\ T_{MAP} = \arg\max_{T} \left(\sum_{\xi_i \in \Xi} \{\log P(\xi_i \mid T)\} + \log(\mathbb{P}\{T = i\}) \right)$$

后验概率在 $\{1,2,3\}$ 是均匀的, 等价于它们相等.

不妨令 $T = i$ 处的先验是 P_i, $\xi_1 = 1$ 程序成功退出, 且仅有这一个观测量.

$$P_1 \left(\frac{R+1}{2^N} \right)^1 = P_2 \left(\frac{R+1}{2^N} \right)^2 = P_3 \left(\frac{R+1}{2^N} \right)^3$$

由上述式子, 小明可能设置的先验见上表.

3 [30pts] Semi-supervised SVM in practice

参照教材中图13.4所示的TSVM算法，在所提供的半监督数据集上进行训练，报告模型在未标记数据集以及测试集上的性能。

本次实验的数据集为一个二分类的数据集，已提前划分为训练数据和测试数据，其中训练数据划分为有标记数据和无标记数据。数据的特征维度为30，每一维均为数值类型。数据文件的具体描述如下:

- `label_X.csv`,`label_y.csv`分别是有标记数据的特征及其标签。
- `unlabel_X.csv`,`unlabel_y.csv`分别是无标记数据的特征及其标签。
- `test_X.csv`,`test_y.csv`分别是测试数据的特征及其标签。

注意，训练阶段只可以使用`label_X.csv`,`label_y.csv`,`unlabel_X.csv`中的数据，其他的数据只可以在测试阶段使用。

(1) 本次实验要求使用Python3编写，代码统一集中在`tsvm_main.py`中，通过运行该文件就可以完成训练和测试，并输出测试结果。

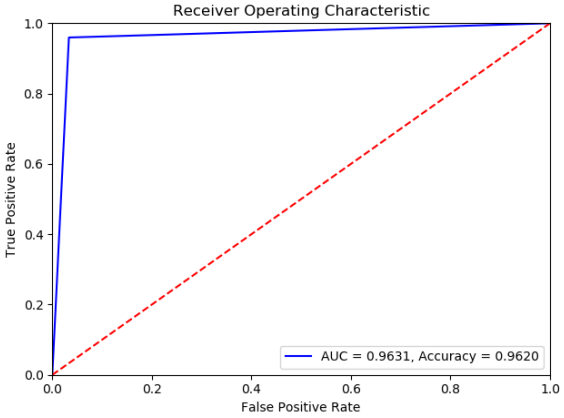
(2) 本次实验需要完成以下功能:

- **[10pts]** 参照教材中图13.4，使用代码实现TSVM算法。要求:
  1. 不允许直接调用相关软件包中的半监督学习方法。
  2. 可以直接调用相关软件包的SVM算法。
  3. 可以使用诸如`cvxpy`等软件包求解QP问题。
- **[10pts]** 使用训练好的模型在无标记数据和测试数据上进行预测，报告模型在这两批数据上的准确率和ROC曲线以及AUC值。
- **[10pts]** 尝试使用各种方法提升模型在测试集上的性能，例如数据预处理，超参数调节等。报告你所采取的措施，以及其所带来的提升。

Solution. *解:*

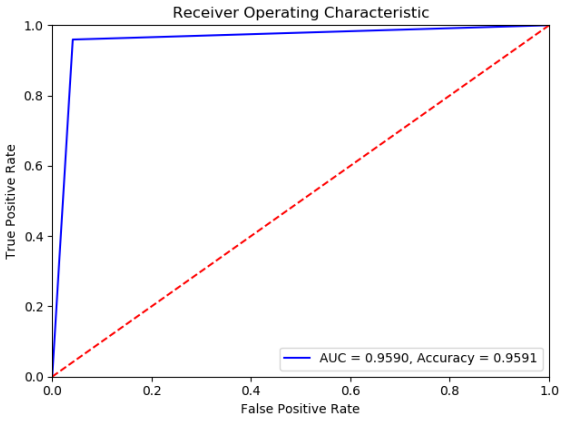
在无标记数据上进行预测:

$$AUC = 0.9631, Accuracy = 0.9620$$



在测试集上进行预测:

$$AUC = 0.9590, Accuracy = 0.9591$$



(3)

这里我尝试了调节各种超参数, 例如未标记样本训练样本上两个 C 的初始值, *kernel*类型*(*高斯, 多项式等*)*, 归一化标准化.

性能得到了很大程度的提升.

附录: 本题实现关键代码

```python
def fit(self):
    l, u = self.X_l.shape[0], self.X_u.shape[0]

    import tqdm
    while self.C_u < self.C_l:
        self.qp_solve()
        for i in tqdm.trange(u):
            for j in range(u):
                if self.y_all[l + i] * self.y_all[l + j] < 1e-5 and \
                        self.a[l + i] > 1e-5 and \
                        self.a[l + j] > 1e-5:

                    if self.y_all[l + i] * self.predict(self.X_u[i]) \
                            + self.y_all[l + j] * self.predict(self.X_u[j]) <= 0:
                        self.y_all[l + i] = -self.y_all[l + i]
                        self.y_all[l + j] = -self.y_all[l + j]
                        self.qp_solve()
        self.C_u = min(2 * self.C_u, self.C_l)

    return self.y_all[l : l + u]
```

图 2: TSVM算法

```python
def qp_solve(self):
    from cvxopt import matrix, solvers
    m, n = self.X_all.shape
    l, u = self.X_l.shape[0], self.X_u.shape[0]

    K = np.zeros((m, m))
    for i in range(m):
        for j in range(m):
            K[i, j] = linear_kernel(self.X_all[i], self.X_all[j])
    P = matrix(np.outer(self.y_all, self.y_all) * K)
    q = matrix(np.ones(m) * -1)
    A = matrix(self.y_all.reshape(1, -1).astype('float'))
    b = matrix(0.0)
    tmp1 = np.diag(np.ones(m) * -1)
    tmp2 = np.identity(m)
    G = matrix(np.vstack((tmp1, tmp2)))
    tmpC = np.zeros(2 * m)
    tmpC[m: m + l] = self.C_l
    tmpC[m + l: 2 * m] = self.C_u
    h = matrix(tmpC)

    solution = solvers.qp(P, q, G, h, A, b)
```

图 3: QP问题求解