机器学习 (周志华) 参考答案 第一章 绪论

机器学习 (周志华西瓜书) 参考答案 总目录

- http://blog.csdn.net/icefire_tyh/article/details/52064910

1. 表 1.1 中若只包含编号为 1，4 的两个样例，试给出相应的版本空间。

假设空间指的是问题所有假设组成的空间，我们可以把学习过程看作是在假设空间中搜索的过程，搜索目标是寻找与训练集“匹配”的假设。

1

假设数据集有 n 种属性，第 i 个属性可能的取值有t_i种，加上该属性的泛化取值 (*)，所以可能的假设有$\prod_i(t_i+1)$。再用空集表示没有正例，假设空间中一共$\prod_i(t_i+1)+1$种假设。

现实问题中常面临很大的假设空间，我们可以寻找一个与训练集一致的假设集合，称之为版本空间。版本空间从假设空间剔除了与正例不一致和与反例一致的假设，它可以看成是对正例的最大泛化。

版本空间的可以通过搜索假设空间来得到，这样需要遍历完整的假设空间。如果数据集中有正例，则可以先对一个正例进行最大泛化，得到2^n个假设，然后再对这些假设进行剔除操作，可以适当精简计算量。

西瓜数据集（精简）

编号	色泽	根蒂	敲声	好瓜
1	青绿	蜷缩	浊响	是
2	乌黑	稍蜷	沉闷	否

数据集有 3 个属性，每个属性 2 种取值，一共$3*3*3+1=28$种假设，分别为

- 1. 色泽 = 青绿 根蒂 = 蜷缩 敲声 = 浊响
- 2. 色泽 = 青绿 根蒂 = 蜷缩 敲声 = 沉闷
- 3. 色泽 = 青绿 根蒂 = 稍蜷 敲声 = 浊响
- 4. 色泽 = 青绿 根蒂 = 稍蜷 敲声 = 沉闷
- 5. 色泽 = 乌黑 根蒂 = 蜷缩 敲声 = 浊响
- 6. 色泽 = 乌黑 根蒂 = 蜷缩 敲声 = 沉闷
- 7. 色泽 = 乌黑 根蒂 = 稍蜷 敲声 = 浊响
- 8. 色泽 = 乌黑 根蒂 = 稍蜷 敲声 = 沉闷
- 9. 色泽 = 青绿 根蒂 = 蜷缩 敲声 =*
- 10. 色泽 = 青绿 根蒂 = 稍蜷 敲声 =*
- 11. 色泽 = 乌黑 根蒂 = 蜷缩 敲声 =*
- 12. 色泽 = 乌黑 根蒂 = 稍蜷 敲声 =*
- 13. 色泽 = 青绿 根蒂 =* 敲声 = 浊响
- 14. 色泽 = 青绿 根蒂 =* 敲声 = 沉闷
- 15. 色泽 = 乌黑 根蒂 =* 敲声 = 浊响
- 16. 色泽 = 乌黑 根蒂 =* 敲声 = 沉闷
- 17. 色泽 =* 根蒂 = 蜷缩 敲声 = 浊响
- 18. 色泽 =* 根蒂 = 蜷缩 敲声 = 沉闷
- 19. 色泽 =* 根蒂 = 稍蜷 敲声 = 浊响
- 20. 色泽 =* 根蒂 = 稍蜷 敲声 = 沉闷
- 21. 色泽 = 青绿 根蒂 =* 敲声 =*
- 22. 色泽 = 乌黑 根蒂 =* 敲声 =*
- 23. 色泽 =* 根蒂 = 蜷缩 敲声 =*

- 24. 色泽 =* 根蒂 = 稍蜷 敲声 =*
- 25. 色泽 =* 根蒂 =* 敲声 = 浊响
- 26. 色泽 =* 根蒂 =* 敲声 = 沉闷
- 27. 色泽 =* 根蒂 =* 敲声 =*
- 28. 空集 Ø

编号 1 的数据可以删除 $2-8$，$10-12$，$14-16$，$18-20$，22，24，26，28(不包含数据 1)
编号 1 的数据可以删除 27(包含了数据 2)
所以版本空间为:

- 1. 色泽 = 青绿 根蒂 = 蜷缩 敲声 = 浊响
- 9. 色泽 = 青绿 根蒂 = 蜷缩 敲声 =*
- 13. 色泽 = 青绿 根蒂 =* 敲声 = 浊响
- 17. 色泽 =* 根蒂 = 蜷缩 敲声 = 浊响
- 21. 色泽 = 青绿 根蒂 =* 敲声 =*
- 23. 色泽 =* 根蒂 = 蜷缩 敲声 =*
- 25. 色泽 =* 根蒂 =* 敲声 = 浊响

一般情况下版本空间是正例的泛化，但由于数据集中只有 1 个正例，所以在版本空间中依然包含了这个样本的假设 (假设 1)。

2. 与使用单个合取式来进行假设表示相比，使用 “析合范式” 将使得假设空间具有更强的表示能力。若使用最多包含 k 个合取式的析合范式来表达 1.1 的西瓜分类问题的假设空间，试估算有多少种可能的假设。

- http://blog.csdn.net/icefire_tyh/article/details/52065626

3. 若数据包含噪声，则假设空间中可能不存在与所有训练样本都一致的假设。在此情形下，试设计一种归纳偏好用于假设选择

通常认为两个数据的属性越相近，则更倾向于将他们分为同一类。若相同属性出现了两种不同的分类，则认为它属于与他最临近几个数据的属性。也可以考虑同时去掉所有具有相同属性而不同分类的数据，留下的数据就是没误差的数据，但是可能会丢失部分信息。

4. 本章 1.4 节在论述 “没有免费的午餐” 定理时，默认使用了 “分类错误率” 作为性能度量来对分类器进行评估。若换用其他性能度量l, 试证明没有免费的午餐” 定理仍成立

还是考虑二分类问题，NFL 首先要保证真是目标函数 f 均匀分布，对于有 X 个样本的二分类问题，显然 f 共有2^X种情况。其中一半是与假设一致的，也就$P(f(x)=h(x))=0.5$。
此时，$\sum_f l(h(x),f(x))=0.5*2^X*(l(h(x)=f(x))+l(h(x)\neq f(x)))$
$l(h(x)=f(x))+l(h(x)\neq f(x))$应该是个常数，隐含的条件就该是 (一个比较合理的充分条件) $l(0,0)=l(1,1),l(1,0)=l(0,1)$。如果不满足， NFL 应该就不成立了 (或者不那么容易证明)。

5. 试述机器学习在互联网搜索的哪些环节起什么作用

1. 最常见的，消息推送，比如某东经常说某些商品我可能会感兴趣，然而并没有。
2. 网站相关度排行，通过点击量，网页内容进行综合分析。
3. 图片搜索，现在大部分还是通过标签来搜索，不过基于像素的搜索也总会有的吧。

机器学习 (周志华) 参考答案 第二章 模型评估与选择

机器学习 (周志华西瓜书) 参考答案 总目录

- http://blog.csdn.net/icefire_tyh/article/details/52064910

1. 数据集包含 1000 个样本，其中 500 个正例，500 个反例，将其划分为包含 70% 样本的训练集和 30% 样本的测试集用于留出法评估，试估算共有多少种划分方式。

一个组合问题，从500正反例中分别选出150正反例用于留出法评估，所以可能取法应该是$(C_{500}^{150})^2$种。

2. 数据集包含 100 个样本，其中正反例各一半，假定学习算法所产生的模型是将新样本预测为训练样本数较多的类别（训练样本数相同时进行随机猜测），试给出用 10 折交叉验证法和留一法分别对错误率进行评估所得的结果。

10 折交叉检验：由于每次训练样本中正反例数目一样，所以讲结果判断为正反例的概率也是一样的，所以错误率的期望是50%。
留一法：如果留下的是正例，训练样本中反例的数目比正例多一个，所以留出的样本会被判断是反例；同理，留出的是反例，则会被判断成正例，所以错误率是100%。

3. 若学习器 A 的 F1 值比学习器 B 高，试析 A 的 BEP 值是否也比 B 高。

~~F1 值的大小与 BEP 值并没有明确的关系。~~
两个分类器的F_1值得大小与他们的 BEP 值大小并没有明确的关系 (没去找)
这道题这里用反推，设计两个 BEP 值相同的分类器，如果他们的F_1值不一样，那么这道题的结论就是否定的
再加点我看了评论后的疑惑：
BEP 值就是F_1值吗?
BEP 值是在 P=R 时取到的，也就是 BEP=P=R。如果在计算 F 时也要定义 P=R，那么F_1和F_β将会恒等于 BEP，那么 P,R,F 在这里有什么意义呢?
这里分两种情况：
第一就是我的理解，在计算 F1 时就是按照分类器真实的分类结果来计算 P,R，再根据 PR 计算 F1。当这个分类器正好 P=R 时，有 P=R=BEP=F1。否则 BEP 的计算不能用当前的 PR，而是通过一步一步尝试到查准率 = 查全率时，P’ =R’ =BEP。
第二种就是不存在我下面假设的分类器，分类器始终会在 P=R 的位置进行截断 (截断指的是分类器将所有样本按分为正例的可能性排序后，选择某个位置。这个位置前面分类为正，后面分类为负)。但是这个可能吗？这种情况下$F_1 = F_\beta = BEP$恒成立，分类器的评价本质将会变成了样本的正例可能性排序，而不是最终的样本划分结果。

分类器将所有训练样本按自己认为是正例的概率排序，排在越前面分类器更可能将它判断为正例。按顺序逐个把样本标记为正，当查准率与查全率相等时，BEP= 查准率 = 查全率。当然分类器的真实输出是在这个序列中的选择一个位置，前面的标记为正，后面的标记为负，这时的查准率与查全率用来计算$F1$值。可以看出有同样的 BEP 值的两个分类器在不同位置截断可能有不同的$F1$值，所以$F1$值高不一定BEP值也高。
比如：

1/+	2/+	3/+	4/+	5/+	6/-	7/-	8/-	9/-	10/-
1/+	2/+	3/+	4/+	6/-	5/-	7/-	8/-	9/-	10/-
1/+	2/+	3/+	4/+	6/+	5/-	7/-	8/-	9/-	10/-

第一行是真实的测试样本编号与分类，第二三行是两个分类器对所有样本按为正例可能性的排序，以及判断的结果。显然两个分类器有相同的 BEP 值，但是他们的$F1$值一个是0.89，一个是0.8。

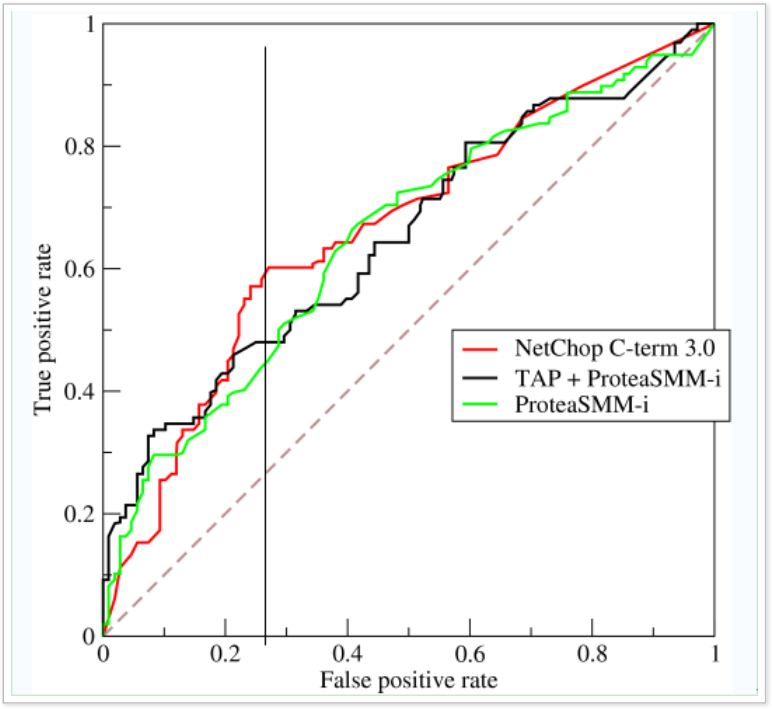
4. 试述真正例率（TPR）、假正例率（FPR）与查准率（P）、查全率（R）之间的联系。

查全率: 真实正例被预测为正例的比例
真正例率: 真实正例被预测为正例的比例
显然查全率与真正例率是相等的。
查准率: 预测为正例的实例中真实正例的比例
假正例率: 真实反例被预测为正例的比例
两者并没有直接的数值关系。

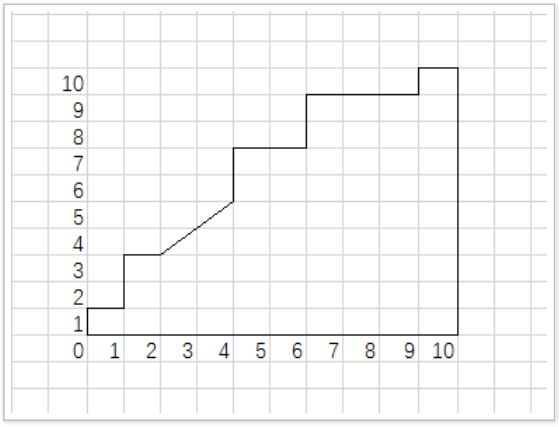
5. 试证明 (2.22)$AUC = 1 - l_{rank}$

从书34页 b 图看来，AUC的公式不应该写的这么复杂，后来才发现原来这个图并没有正例反例预测值相等的情况。当出现这种情况时，ROC曲线会呈斜线上升，而不是这种只有水平和垂直两种情况。

由于一开始做题时并没有想过 ROC 曲线不可以是斜线，所以画了这张图，如果不存在正例反例预测值相等的情况，那么斜线也没必要存在。
但是在维基百科上看到一副图，貌似也存在斜线的 ROC，但是不知道含义是否和我这里写的一样。
https://en.wikipedia.org/wiki/Receiver_operating_characteristic
引用一幅有斜线的 ROC 曲线



与BEP一样，学习器先将所有测试样本按预测概率排序，越可能是正的排在越前面。然后依次遍历，每扫描到一个位置，里面如果只有正例，则ROC曲线垂直向上，如果只有反例，曲线水平往右，如果既有正例也有反例，则斜向上。如图所示



由于TPR与FPR的分母是常数，所以这里按比例扩大了坐标 (分别是真实正例和真实反例的数目倍)，可以更好看出曲线走势。

可以看出一共有20个测试样本，10个正，10个反。学习器排序的结果是
$+, -, (+, +), (+, -), (+, -), (+, +), (-, -), (+, +), (-, -, -), +, -$。其中括号内的样本排在相同的位置。
$<(+, +, -, -)$ 与 $(+, -), (+, -$是同样的效果 >

公式2.21累加了所有不在正例的反例数目，其中同样的位置标记为0.5，在正例前面标记为1。从图中可以看出，折线每次向右 (右上) 延伸，表示扫描到了反例，折线上方对应的面积，就是该反例后面有多少个正例，每个正例是一个正方形，对应的面积是1。同位置上的正例是个三角形，对

应的面积是0.5。计算出总面积后，由于ROC图的坐标是归一化的，所以总面积要除以一开始放大的倍数，也就是m^+m^-。

6. 试述错误率与 ROC 曲线之间的关系

ROC曲线每个点对应了一个TPR与FPR，此时对应了一个错误率。
$E_{cost} = (m^+ * (1 - TPR) * cost_{01} + m^- * FPR * cost_{10})/(m^+ + m^-)$
学习器会选择错误率最小的位置作为截断点。

7. 试证明任意一条 ROC 曲线都有一条代价曲线与之对应，反之亦然。

由定义可以知道TPR与FPR都是由0上升到1，那么FNR则是由1下降到0。
每条ROC曲线都会对应一条代价曲线，由于第一条代价线段的是$(0,0),(1,1)$，最后是$(0,1)(1,0)$,
所有代价线段总会有一块公共区域，这个区域就是期望总体代价，而这块区域的边界就是代价曲线，且肯定从$(0,0)$到$(1,0)$。
在有限个样本情况下，ROC是一条折线，此时根据代价曲线无法还原ROC曲线。但若是理论上有无限个样本，ROC是一条连续的折线，代价曲线也是连续的折线，每个点的切线可以求出TPR与FNR，从而得到唯一的ROC曲线。

8.Min-Max 规范化与 z-score 规范化如下所示。试析二者的优缺点。

$Min - max$规范化方法简单，而且保证规范化后所有元素都是正的，每当有新的元素进来，只有在该元素大于最大值或者小于最小值时才要重新计算全部元素。但是若存在一个极大 (小) 的元素，会导致其他元素变的非常小(大)。
$z - score$标准化对个别极端元素不敏感，且把所有元素分布在0的周围，一般情况下元素越多，0周围区间会分布大部分的元素，每当有新的元素进来，都要重新计算方差与均值。

9. 试述卡方检验过程。

略 (……)

10. 试述在使用$Friedman$检验中使用式 (2.34) 与(2.35)的区别

书上说$Friedman$检验，在Nk比较大时，平均序值r_i近似于正态分布，均值为$\frac{k+1}{2}$,
方差为$\frac{k^2-1}{12}$~~(其实我觉得r_i的方差是$\frac{k^2-1}{12N}$)。~~

即：r_i~$N(\frac{k+1}{2},\frac{k^2-1}{12})$

所以$\frac{12N}{k^2-1}(r_i-\frac{k+1}{2})^2$~$\chi^2(1)$

统计量$\frac{12N}{k^2-1}\sum_k(r_i-\frac{k+1}{2})^2$由于$k$个算法的平均序值$r_i$是有关联的，知道其中$k-1$个就能推出最后一个，所以自由度为$k-1$, 在前面乘上$\frac{k-1}{k}$，最终得到$Friedman$统计量为
$fri = \frac{k-1}{k} * \frac{12N}{k^2-1}\sum_k(r_i-\frac{k+1}{2})^2$

猜测: 由于$Friedman$统计量只考虑了不同算法间的影响，而没去考虑不同数据集 (其他方差) 所带来的影响，所以书上说这个 Friedman 统计量太保守。
对序值表做方差分析:
总方差$SST = N*(E(X^2)-(EX)^2) = N*k*(k^2-1)/12$自由度$N*(k-1)$
算法间方差$SSA = N*\sum_k(r_i-\frac{k+1}{2})^2$ 自由度$k-1$
其他方差 $SSE = SST - SSA$自由度$(N-1)*(k-1)$

做统计量$f = \frac{SSA/(k-1)}{SSE/((N-1)*(k-1))} = \frac{(N-1)fri}{N(k-1)-fri}$,f服从$(k-1)$和$(N-1)*(k-1)$的F分布

机器学习(周志华) 参考答案 第三章 线性模型

机器学习(周志华西瓜书) 参考答案 总目录

- http://blog.csdn.net/icefire_tyh/article/details/52064910

1.试分析在什么情况下，在以下式子中不比考虑偏置项b。

线性模型$y = w^t x + b$,两个实例相减得到$y_i - y_0 = w^t(x_i - x_0)$,以此消除了$b$。所以可以对训练集每个样本都减去第一个样本，然后对新的样本做线性回归，只需要用模型$y = w^t x$。

2.试证明，对于参数w，对率回归（logistics回归）的目标函数（式1）是非凸的，但其对数似然函数（式2）是凸的。

如果一个多元函数是凸的，那么它的Hessian矩阵是半正定的。

> $y = \frac{1}{1+e^{-(w^T x+b)}}$
> $\frac{dy}{dw} = \frac{xe^{-(w^T x+b)}}{(1+e^{-(w^T x+b)})^2} = x(y-y^2)$
> $\frac{d}{dw^T}(\frac{dy}{dw}) = x(1-2y)(\frac{dy}{dw})^T = xx^T y(y-1)(1-2y)$

xx^T合同于单位矩阵，所以xx^T是半正定矩阵
y的值域为$(0,1)$,当$y \in (0.5,1)$时，$y(y-1)(1-2y) < 0$,导致$\frac{d}{dw^T}(\frac{dy}{dw})$半负定，所以$y = \frac{1}{1+e^{-(w^T x+b)}}$是非凸的。

> $l(\beta) = \sum_{i=1}^{m}(-y_i\beta^T x_i + ln(1+e^{\beta^T x}))$
> $\frac{d}{d\beta^T}(\frac{dl}{d\beta}) = xx^T p1(x;\beta)(1-p1(x;\beta))$

显然概率$p1 \in (0,1)$，则$p1(x;\beta)(1-p1(x;\beta)) \geq 0$，所以$l(\beta) = \sum_{i=1}^{m}(-y_i\beta^T x_i + ln(1+e^{\beta^T x}))$是凸函数。

3.编程实现对率回归，并给出西瓜数据集3.0α上的结果

- http://blog.csdn.net/icefire_tyh/article/details/52068844

4.选择两个UCI数据集，比较10折交叉验证法和留一法所估计出的对率回归的错误率。

- http://blog.csdn.net/icefire_tyh/article/details/52068900

5.编程实现线性判别分析，并给出西瓜数据集3.0α上的结果。

- http://blog.csdn.net/icefire_tyh/article/details/52069003

6. LDA仅在线性可分数据上能获得理想结果，试设计一个改进方法，使其能较好地用于非线性可分数据。

在当前维度线性不可分，可以使用适当的映射方法，使其在更高一维上可分，典型的方法有$KLDA$，可以很好的划分数据。

7.令码长为9，类别数为4，试给出海明距离意义下理论最优的EOOC二元码并证明之。

对于$ECOC$二元码，当码长为2^n时，至少可以使$2n$个类别达到最优间隔，他们的海明距离为$2^{(n-1)}$。比如长度为8时，可以的序列为

1	1	1	1	-1	-1	-1	-1
1	1	-1	-1	1	1	-1	-1
1	-1	1	-1	1	-1	1	-1
-1	-1	-1	-1	1	1	1	1
-1	-1	1	1	-1	-1	1	1
-1	1	-1	1	-1	1	-1	1

其中$4,5,6$行是对$1,2,3$行的取反。若分类数为4，一共可能的分类器共有2^4-2种(排除了全1和全0)，在码长为8的最优分类器后添加一列没有出现过的分类器，就是码长为9的最优分类器。

8.EOOC编码能起到理想纠错作用的重要条件是：在每一位编码上出错的概率相当且独立。试析多分类任务经ECOC编码后产生的二类分类器满足该条件的可能性及由此产生的影响。

理论上的$ECOC$码能理想纠错的重要条件是每个码位出错的概率相当，因为如果某个码位的错误率很高，会导致这位始终保持相同的结果，不再有分类作用，这就相当于全0或者全 1的分类器，这点和NFL的前提很像。但由于事实的样本并不一定满足这些条件，所以书中提到了有多种问题依赖的$ECOC$被提出。

9.使用OvR和MvM将多分类任务分解为二分类任务求解时，试述为何无需专门针对类别不平衡性进行处理。

书中提到，对于OvR，MvM来说，由于对每个类进行了相同的处理，其拆解出的二分类任务中类别不平衡的影响会相互抵消，因此通常不需要专门处理。以$ECOC$编码为例，每个生成的二分类器会将所有样本分成较为均衡的二类，使类别不平衡的影响减小。当然拆解后仍然可能出现明显的类别不平衡现象，比如一个超级大类和一群小类。

10.试推出多分类代价敏感学习(仅考虑基于类别的错误分类代价)使用“再缩放”能获得理论最优解的条件。

题目提到仅考虑类别分类的误分类代价，那么就默认正确分类的代价为0。
于是得到分类表,(假设为3类)

0	c_{12}	c_{13}
c_{21}	0	c_{23}
c_{31}	c_{32}	0

对于二分类而言，将样本为正例的后验概率设为是p,那么预测为正的代价是$(1-p)*c_{12}$，
预测为负的代价是$p*c_{21}$。当$(1-p)*c_{12}\leq p*c_{21}$样本会被预测成正例，因为他的代价更小。当不等式取等号时，得到了最优划分，这个阀值$p_r=\frac{c12}{c12+c21}$，这表示正例与反例的划分比例应该是初始的$\frac{c12}{c21}$倍。假设分类器预设的阀值是p_o,不考虑代价敏感时，当$\frac{y}{1-y}>\frac{p_o}{1-p_o}$时取正例。当考虑代价敏感，则应该是$\frac{y}{1-y}>\frac{1-p_r}{p_r}*\frac{p_o}{1-p_o}=\frac{c21}{c12}*\frac{p_o}{1-p_o}$。
推广到对于多分类，任意两类的最优再缩放系数$t_{ij}=c_{ij}/c_{ji}$,然而所有类别的最优缩放系数并不一定能同时满足。当代价表满足下面条件时，能通过再缩放得到最优解。设$t_{ij}=w_i/w_j$，则$w_i/w_j=c_{ij}/c_{ji}$对所有i,j成立，假设有k类，共C_k^2个等式，此时代价表中$k*(k-1)$个数，最少只要知道$2*(k-1)$就能推出整张表。

机器学习 (周志华) 参考答案 第四章 决策树

机器学习 (周志华西瓜书) 参考答案 总目录

- http://blog.csdn.net/icefire_tyh/article/details/52064910

```
如果说决策树这章还有什么遗憾，就是实在没找到一个好的树形图生成器，结果只能用Matlab上那么丑的图
```

1. 试证明对于不含冲突数据（即特征向量完全相同但标记不同）的训练集，必存在与训练集一致（即训练误差为 0）的决策树。

因为决策树是通过属性来划分，相同属性的样本最终肯定会进入相同的叶节点。一个叶节点只有一个分类，如果样本属性相同而分类不同，必然产生训练误差。反之，决策树只会在当前样本集合是同一类或者所有属性相同时才会停止划分，最终得到训练误差为 0 的决策树。

2. 试析使用 “最小训练误差” 作为决策树划分选择的缺陷。

从机器学习最开始就讲起，最小训练误差并不可靠，由于过度学习样本特性最终导致严重的过拟合，而没有泛化能力。

3. 试编程实现基于信息熵进行划分选择的决策树算法，并为表 4.3 中数据生成一棵决策树。

- http://blog.csdn.net/icefire_tyh/article/details/52081556
  重写的不剪枝的决策树
- http://blog.csdn.net/icefire_tyh/article/details/54575527

4. 试编程实现基于基尼指数进行划分选择的决策树算法，并为表 4.2 中数据生成预剪枝、后剪枝决策树，并与未剪枝决策树进行比较。

- http://blog.csdn.net/icefire_tyh/article/details/52081879

5. 试编程实现基于对率回归进行划分选择的决策树算法，并为表 4.3 中数据生成一棵决策树。

- http://blog.csdn.net/icefire_tyh/article/details/52081770

6. 试选择 4 个 UCI 数据集，对上述 3 种算法所产生的未剪枝、预剪枝、后剪枝决策树进行实验比较，并进行适当的统计显著性检验。

这里要对上面三种实现的算法进行未剪枝，预剪枝，后剪枝做比较，对率回归划分就算了，都不知道是个什么情况，信息增益和基尼指数的差别并不大，其实就是为了比较未剪枝，预剪枝，后剪枝对测试样本的输出结果。显著性分析，对 2 种算法，3 种剪枝方式的错误数做方差分析，信息增益和基尼指数有显著区别是拒绝的，未剪枝，预剪枝，后剪枝有显著区别是接受的。

7. 图 4.2 是一个递归算法，若面临巨量数据，则决策树的层数会很深，使用递归方法易导致 “栈” 溢出，试使用 “队列” 数据结构，以参数 maxDepth 控制数的最大深度，写出与图 4.2 等价、但不使用递归的决策树生成算法。

直接用递归会导致大量的临时变量被保存，当层数过深时会导致 “栈” 溢出。
用队列对决策树进行层次遍历来生成，用 Max_Depth 来控制树的最大层数。队列中每个元素代表着决策树的每个节点，它必要的属性有：样本集合、剩余属性集合，当前层数指示，父节点序号。队列一开始里面只有一个元素，就是最初初始化，带着所有样本的根节点。然后当队列不为空的时候开始循环，每次取出一个元素，判断是否需要划分，如果不要，就是一个叶节点，出队列就不用管了；如果需要划分，那么找出最好的划分属性，然后划分成 n 个子区间，依次送入队列，继续循环，直到队列为空。
是否需要划分有 3 个依据:

- 当前所有样本属于一类
- 当前所有样本属性完全相同
- 达到了 Max_Depth 的深度

这样就完成了层次遍历 (广度优先搜索) 对决策树的构建。
显然由于每次出队的元素要先完全划分，那么如果是进行预剪枝算法的决策树，用队列结构是非常方便的。
如果是后剪枝，那必须要等到最终整棵树完全生成，才能进行。

8. 试将决策树生成的深度优先搜索过程修改为广度优先搜索，以参数 MaxNode 控制树的最大结点数，将题 4.7 中基于队列的决策树算法进行改写。对比题 4.7 中的算法，试分析哪种方式更易于控制决策树所需储存不超过内存。

队列结构看着不错，但是极端情况下，比如层树很低，但是分叉多，会导致队列内有大量元素。另外不能在决策树生成过程中很好的进行后剪枝，全树生成后再次统计会浪费大量时间。
用栈结构把递归转换为非递归是很好的方法，设置最大节点数 MaxNode 来保证节点数不会爆炸。栈结构内的元素组成与列队的基本相同: 样本集合、剩余属性集合，当前层数指示，父节点序号。初始化也是一样，将一个包含所有样本的集合压入栈中，当栈不为空时，每次拿出栈顶的元素进行操作，如果不可以划分，则不做操作；如果可以划分，则划分出 1 个子集，先将剩余子集压回栈，再将划分出的子集压回栈。然后再取出一个元素，知道栈内没有元素为止。
是否需要划分有 3 个依据:

- 当前所有样本属于一类
- 当前所有样本属性完全相同
- 达到了 Max_node 上限

栈也有他的问题，极端情况下会生成一棵畸形的二叉树，但就算这样内部元素也只是队列结构极端情况的一半。栈可以很好的进行后剪枝操作，当非叶节点所有叶节点生成后可以做后剪枝，由于划分子集的时候就需要计算各个属性样本数，所以这些操作代价并不高。对于一棵完整的树，后剪枝要对所有只拥有叶节点的非叶节点进行遍历，如果某个非叶节点可以剪枝，还需要进一步考虑它的父节点。虽然在某些剪枝需求不大的情况下生成树后剪枝有不错的效果，但是我觉得生成中判断更加方便，而且不需要太多额外的代码。

9. 试将 4.4.2 节对缺失值的处理机制推广到基尼指数的计算中去。

只需要把信息增益的公式换成基尼指数就行，包括扩展到连续参数，缺失参数，都是很直观的方法。

10. 从网上下载或自己编程实现任意一种多变量决策树算法，并观察其在西瓜数据集 3.0 上产生的结果。

- http://blog.csdn.net/icefire_tyh/article/details/52082051

机器学习 (周志华) 参考答案 第五章 神经网络

机器学习 (周志华西瓜书) 参考答案 总目录

- http://blog.csdn.net/icefire_tyh/article/details/52064910

```
这章就是真正的两句话讲完一个知识点，然后后面还出个题让你编程
```

神经网络作为一个相当大的，多学科交叉的学科领域，并不是仅仅应用于机器学习。书上 3 张纸介绍了 6 种神经网络，都是泛泛几句话介绍。选了几本关于神经网络的书放进某东的购物车，等着下次 300-200 买起来慢慢研究。

1. 试述将线性函数$f(x) = w^t x$用作神经元激活函数的缺陷。

必须要强调的是，神经网络中必须要有非线性的激活函数，无论是在隐层，还是输出层，或者全部都是。如果单用$w^t x$作为激活函数，无论多少层的神经网络会退化成一个线性回归，只不过是把他复杂化了。

2. 试述使用图 5.2(b) 激活函数的神经元与对率回归的联系。

两者都是希望将连续值映射到 {0,1} 上，但由于阶跃函数不光滑，不连续的性质，所以才选择了 sigmoid 作为映射函数。不同之处在于激活函数不一定要使用 sigmoid，只要是非线性的可导函数都可以使用。

3. 对于图 5.7 中v_{ih}，试推导出 BP 算法中的更新公式。

$-\frac{\partial E_k}{\partial v_{ih}} = -\frac{\partial E_k}{\partial b_h}\frac{\partial b_h}{\partial a_h}\frac{\partial a_h}{\partial v_{ih}}$
$\frac{\partial a_h}{\partial v_{ih}} = x_i$
$e_h = -\frac{\partial E_k}{\partial b_h}\frac{\partial b_h}{\partial a_h}$在书中 5.15 已经证明
所以得到更新公式$\upsilon e_h x_i$

4. 试述学习率的取值对神经网络训练的影响。

如果学习率太低，每次下降的很慢，使得迭代次数非常多。
如果学习率太高，在后面迭代时会出现震荡现在，在最小值附近来回波动。

5. 试编程实现标准 BP 算法与累积 BP 算法，在西瓜数据集 3.0 上分别用这个算法训练一个单隐层网络，并进行比较。

- http://blog.csdn.net/icefire_tyh/article/details/52106069

6. 试设计一个 BP 改进算法，能通过动态学习率显著提升收敛速度。

```
这真是一个蛋疼的题，本来以为方法很多，结果没有一个能用的。
```

固定的学习率要么很慢要么在后期震荡，设计一种自适应的动态学习率算法是有必要的。

- 对四种参数的最速方向做一位搜索
  这是很直观的一种方法，已知 f(x) 在x_0的导数为 d, 那么下降方向就是 - d。一位搜索就是求试$f(x+td)$最小的 t，也就是当前的学习率。
  然而这方法的 t 用解析法并不好求，$f_t'(x+td)=0$也是无解的。
  使用近似方法尝试了下收敛速度并没有显著提升

- 对四种参数做牛顿迭代
  虽然不符合题目改学习率的要求，但是牛顿法肯定能大大提高收敛速度，只是没有了学习率这个概念。
  然而 Hessian 矩阵求的我想吐血，最后也没去继续了。

书中给出了两篇文献供参考

- Neural Networks: Tricks of the Trade
- Neural Smithing:Supervised learning in Feedforward Artificial Neural Networks

  暂时不解这个题，等以后遇到好的方法再来更新。

7. 根据式 5.18 和 5.19，试构造一个能解决异或问题的 RBF 神经网络。

- http://blog.csdn.net/icefire_tyh/article/details/52106074

8. 从网上下载或自己编程实现一个 SOM 网络，并观察在西瓜数据集$3.0a$上产生的结果。

SOM 网络书上真的就是草草介绍了下，顺便配了个看不懂的图。我去翻了下《MATLAB 神经网络 43 个案例分析》这本书，里面对 SOM 讲的还是很明白的，包括原理与训练方法。
简单的说，SOM 网络在平面放 M*N 个神经元 (M,N 自定义)，神经元按 8 邻接来连接。网络将输入层映射到平面上 M*N 个神经元上 (M,N 自定义)，每个输入与每个神经元都有一个初始权值，然后计算各个神经元的权值向量与输入向量的距离，选择距离最小的神经元更新他以及他 8 邻接神经元的权值向量，循环这个过程直到收敛。
如果神经元比样本数多，足够多次迭代后所有样本会映射到了不同的神经元上。
网络生成后就通过输入算出离他最近的神经元，将它标记为该神经元的分类。
matlab 中有现成的 SOM 网络，我就没自己编程了。

最终得到 SOM 网络为

64 个神经元最终的位置，并可以查看样本映射到了哪个神经元上。

9. 试推导用于 Elman 网络的 BP 算法。

Elman 比正常网络多了个反馈，把前一次的b_h作为隐层的输入来调节隐层。
假设用u_{ih}来表示反馈输入与隐层连接的参数，由于前一次计算的b_h作为常数输入，uij与vij的计算方法一样，
$\Delta u_{ih}=\eta e_h b_h$，其中 e_h 书上 5.15 给出。就是相当于多了几个输入会变的输入层神经元。

10. 实现一个卷积神经网络。

CNN 作为深度学习一个重要工具，Stanford 专门有个 CS231n 来讲 CNN。等我把西瓜书弄完再回头解决这个问题，我感觉一时半会搞不定，还是先把机器学习基础的做完再说。

机器学习 (周志华) 参考答案 第六章 支持向量机

机器学习 (周志华西瓜书) 参考答案 总目录

- http://blog.csdn.net/icefire_tyh/article/details/52064910

> 总的来说支持向量机这章书上讲的比较全面，公式和推到都比较详细，可惜的就是没有具体的SMO算法过程

1. 试证明样本空间中任意点 x 到超平面(w,b)的距离为式 (6.2)。

超平面(w,b)的平面法向量为w， 任取平面上一点x_0，有$w^T x_0 + b = 0$。x到平面的距离就是x到x_0的距离往w方向的投影，就是$\frac{|w^T(x-x_0)|}{|w|} = \frac{|w^T x+b|}{|w|}$。

2. 使用 libsvm, 在西瓜数据集3.0α上分别用线性核和高斯核训练一个 SVM, 并比较其支持向量的差别。

```
由于电脑的vs是2015的，而matlab上最高只支持2013来编译libsvm，所以只能在vs上用libsvm了。
1
```

训练的结果是线性核与高斯核得到了完全一样的支持向量，由于没去分析 libsvm 内部是如何计算的，这里只贴结果。
第一列是支持向量的权值，后面则是支持向量对应的属性

a_i	x_1	x_2
1	0.697	0.46
1	0.744	0.376
1	0.634	0.264
1	0.608	0.318
1	0.556	0.215
1	0.403	0.237
1	0.481	0.149
1	0.437	0.211
-1	0.666	0.091
-1	0.243	0.267
-1	0.343	0.099
-1	0.639	0.161
-1	0.657	0.198
-1	0.36	0.37
-1	0.593	0.042
-1	0.719	0.103

3. 选择两个 UCI 数据集，分别用线性核和高斯核训练一个 SVM，并与 BP 神经网络和 C4.5 决策树进行实验比较。

使用的是 iris 数据集，选取其中分类为 1，2 的样本，各 50 个，4 属性。
每类选前 40 个样本训练，后 10 个样本作为测试
线性核：找出 3 个支持向量

a_i	x_1	x_2	x_3	x_4
0.04	5.1	3.3	1.7	0.5
0.16	4.8	3.4	1.9	0.2
-0.20	4.9	2.5	4.5	1.7

偏置为 1.50709
高斯核：找出 9 个支持向量

a_i	x_1	x_2	x_3	x_4
0.48	4.4	2.9	1.4	0.2
0.45	4.3	3	1.1	0.1
0.90	5.7	4.4	1.5	0.4
-0.17	6.3	3.3	6	2.5
-0.66	4.9	2.5	4.5	1.7
-0.03	6.5	3.2	5.1	2
-0.40	7.7	2.6	6.9	2.3
-0.15	6	2.2	5	1.5
-0.41	7.9	3.8	6.4	2

偏置为 - 0.212437
编写一个验证的 matlab 程序

tmp.xlsx 验证数据表格

得到结果表格如下

y1	y2	y
1.155205016	1.164557907	1
0.997242309	0.780919564	1
1.148298718	1.068992278	1
0.906038093	1.085988234	1
0.842638654	1.042299416	1
1.035492502	1.062906963	1
1.062585631	1.141980465	1
1.09304642	1.100071803	1
1.101546038	1.141056647	1
1.103671899	1.13405161	1
-1.839224592	-1.049900524	0
-1.542242776	-0.95432202	0
-1.471406411	-1.085122464	0

y1	y2	y
-1.958761346	-1.084832335	0
-1.895362864	-1.029274823	0
-1.608120552	-1.002438466	0
-1.448029593	-1.0262247	0
-1.519044109	-1.03794725	0
-1.658415695	-0.995288562	0
-1.402518712	-1.058306748	0

这里没和神经网络和决策树做对比了，这个数据集的数据线性可分，应该都是 0 误差。

4. 讨论线性判别分析与线性核支持向量机在何种情况下等价。

在线性可分的情况下, LDA 求出的w_l与线性核支持向量机求出的w_s有$w_l * w_s = 0$，即垂直，此时两者是等价的。

当初在做这个题的时候也没细想，就想当然的认为在线性可分时两者求出来的 w 会垂直，现在看来并不一定。
首先，如果可以使用软间隔的线性 SVM，其实线性可分这个条件是不必要的，如果是硬间隔线性 SVM，那么线性可分是必要条件。这个题只说了是线性 SVM，就没必要关心数据是不是可分，毕竟 LDA 是都可以处理的。
第二，假如当前样本线性可分，且 SVM 与 LDA 求出的结果相互垂直。当 SVM 的支持向量固定时，再加入新的样本，并不会改变求出的 w，但是新加入的样本会改变原类型数据的协方差和均值，从而导致 LDA 求出的结果发生改变。这个时候两者的 w 就不垂直了，但是数据依然是可分的。所以我上面说的垂直是有问题的。
我认为这个题的答案应该就是，当线性 SVM 和 LDA 求出的 w 互相垂直时，两者是等价的，SVM 这个时候也就比 LDA 多了个偏移 b 而已。

5. 试述高斯核 SVM 与 RBF 神经网络的联系

RBF 网络的径向基函数与 SVM 都可以采用高斯核，也就分别得到了高斯核 RBF 网络与高斯核 SVM。
神经网络是最小化累计误差，将参数作为惩罚项，而 SVM 相反，主要是最小化参数，将误差作为惩罚项。
在二分类问题中，如果将 RBF 中隐层数为样本个数，且每个样本中心就是样本参数，得出的 RBF 网络与核 SVM 基本等价，非支持向量将得到很小的w.
使用 LIBSVM 对异或问题训练一个高斯核 SVM 得到α，修改第 5 章 RBF 网络的代码，固定β参数为高斯核 SVM 的参数，修改每个隐层神经元的中心为各个输入参数，得到结果w,w 与α各项成正比例。

6. 试析 SVM 对噪声敏感的原因。

SVM 的目的是求出与支持向量有最大化距离的直线，以每个样本为圆心，该距离为半径做圆，可以近似认为圆内的点与该样本属于相同分类。如果出现了噪声，那么这个噪声所带来的错误分类也将最大化，所以 SVM 对噪声是很敏感的。

7. 试给出式 (6.52) 的完整 KT 条件。

非等式约束写成拉格朗日乘子式，取最优解要满足两个条件

- 拉格朗日乘子式对所有非拉格朗日参数的一阶偏导为 0
- 非等式约束对应的拉格朗日项，要么非等式的等号成立，要么对应的拉格朗日参数为 0

所以得到完整 KT 条件
$w = \sum_i(\alpha_i' - \alpha_i)x_i$
$0 = \sum_i(\alpha_i' - \alpha_i)$
对所有的i
$C = \alpha_i + u_i$
$C = \alpha_i' + u_i'$
$\alpha_i(f(x_i) - y_i - \varepsilon - \xi_i) = 0$
$\alpha_i'(y_i - f(x_i) - \varepsilon - \xi_i') = 0$
$(C - \alpha_i)\xi_i = 0$
$(C - \alpha_i')\xi_i' = 0$

8. 以西瓜数据集 3.0α的 “密度” 属性为输入，“含糖率”为输出，使用 LIBSVM 训练一个 SVR。

含糖率和密度有什么必然联系吗？训练后得到的支持向量为

密度x_i	
1	0.697
1	0.744
0.798	0.608
-1	0.666
0.452	0.243
-1	0.245
-0.25	0.343
1	0.36
-1	0.593
-1	0.719

偏置为 0.213589
得到含糖率与密度的关系
假设密度为x
含糖率$(x) = \sum_i \alpha_i e^{-(x-x_i)^2} + 0.213589$

9. 试使用和技巧推广对率回归，产生 “核对率回归” 。

- http://blog.csdn.net/icefire_tyh/article/details/52135526

10. 设计一个显著减少 SVM 中支持向量数目而不显著降低泛化性能的方法。
对于线性的 SVM，三个属性不完全一样的支持向量就能确定这个 SVM，而其他的落在边缘上的点都可以舍弃。
还没太想明白，以后有机会更新。

机器学习 (周志华) 参考答案 第七章 贝叶斯分类器

机器学习 (周志华西瓜书) 参考答案 总目录

- http://blog.csdn.net/icefire_tyh/article/details/52064910

```
看了半天依然没看懂如何去优化贝叶斯网，9，10题先空着
```

1. 试使用极大似然法估算西瓜数据集 3.0 中前 3 个属性的类条件概率。

极大似然法要先假定一种概率分布形式。
色泽：
对于好瓜，假设
P(色泽 = 青绿 | 好瓜)=σ_1
P(色泽 = 乌黑 | 好瓜)=σ_2
P(色泽 = 浅白 | 好瓜)=σ_3=$1-\sigma_1-\sigma_2$
$L(\sigma)=\bigcap_i P(\text{色泽}=x_i|\text{好瓜})=\sigma_1^3\sigma_2^4(1-\sigma_1-\sigma_2)$
$L'(\sigma_1)=\sigma_2^4\sigma_1^2(3-4\sigma_1-3\sigma_2)$
$L'(\sigma_2)=\sigma_1^3\sigma_2^3(4-4\sigma_1-5\sigma_2)$
令$L'(\sigma_1)=0$，$L'(\sigma_2)=0$得$\sigma_1=\frac{3}{8}$，$\sigma_1=\frac{1}{2}$，$\sigma_3=\frac{1}{8}$
可以看出$\sigma_1,\sigma_2,\sigma_3$分别对应他们在样本中出现的频率。

对于坏瓜以及另外两种属性计算方式相同，得出类似的结果。

2. 试证明：条件独立性假设不成立时，朴素贝叶斯分类器任有可能产生最优分类器。

朴素贝叶斯分类器就是建立在条件独立性假设上的。当有不独立的属性时，假如所有样本不独立的属性取值相同时分类也是相同的，那么此时朴素贝叶斯分类器也将产生最优分类器。

3. 试编程实现拉普拉斯修正的朴素贝叶斯分类器，并以西瓜数据集 3.0 为训练集，并对 “测 1” 样本进行分类。

- http://blog.csdn.net/icefire_tyh/article/details/52167211

4. 实践中用式 (7.15) 决定分类类别时，若数据的维度非常高，则连乘的概率结果会非常接近 0 并导致下溢。试述防止下溢的可能方案。

若连乘的式子太多，导致乘积接近 0。由于属性个数是已知的，可以对每个乘式做适当次的开方处理，可以保证结果不会为 0。另外也可以对各项取对数，当累加太多时，可能导致和接近负无穷。可以对每个加数除以属性的个数，来防止溢出。

5. 试证明: 二分类任务中两类数据满足高斯分布且方差相同时，线性判别分析产生最优贝叶斯分类器。

假设 1 类样本均值为u_1，2 类样本均值为u_2
由于数据满足同方差的高斯分布，当样本足够大时，可以认为

线性判别分析公式$J=\frac{|w^T(u_1-u_2)|^2}{w^T(\Sigma_1+\Sigma_2)w}$求最大值
对$\frac{1}{J}=\frac{w^T(\Sigma_1+\Sigma_2)w}{|w^T(u_1-u_2)|^2}=\sum_i\frac{(1-y_i)|w^T(x_i-u_1)|^2+y_i|w^T(x_i-u_2)|^2}{|w^T(u_1-u_2)|^2}$求最小值

最优贝叶斯分类器使每个训练样本的后验概率$P(c|x)$最大，对应线性判别分析中，即离对应分类的中心距离 (平方) 除以两个分类中心的距离 (平方) 越小。
即求$\sum_i\frac{(1-y_i)|w^T(x_i-u_1)|^2+y_i|w^T(x_i-u_2)|^2}{|w^T(u_1-u_2)|^2}$的最小值

两个式子相同，所以线性判别分析产生最优贝叶斯分类器。

6. 试编程实现 AODE 分类器，并以西瓜数据集 3.0 为训练集，并对 “测 1” 样本进行分类。

- http://blog.csdn.net/icefire_tyh/article/details/52167263

7. 给定 d 个二值属性的分类任务，假设对于任何先验概率的估算需要 30 个样本。试估计 AODE 中估算先验概率$p(c, x_i)$所需要的样本数。

显然对于正负样本，各属性对应的取值x_i需要出现 30 次。
最好的情况下，只需要 60 个样本就能就能估算概率。其中 30 个x_i属性的样本取值为 1，30 个x_i属性的样本取值为 0。尽管这不符合实际情况(相同属性取值不同)。
最坏的情况下，要 60d 个样本才能估算。其中每个样本只有一个属性和测试样本x_i相同，其余都是另一个取值。

8. 考虑图 7.3，证明：在同父结构中，若x_1的取值未知，则$x_3 \perp x_4$不成立。在顺序结构中，$y \perp z|x$成立，但$y \perp z$不成立。

①.x_1已知时，$p(x_1, x_3, x_4) = p(x_1)p(x_3|x_1)p(x_4|x_1)$
$p(x_3, x_4|x_1) = \frac{p(x_1,x_3,x_4)}{p(x_1)} = p(x_3|x_1)p(x_4|x_1)$
所以$x_3 \perp x_4|x_1$。

x_1未知时，$p(x_1, x_3, x_4) = p(x_1)p(x_3|x_1)p(x_4|x_1)$
p(x_3,x_4)=\sum_x_1{p(x_1,x_3,x_4)}=\sum_x_1{p(x_1)p(x_3|x_1)p(x_4|x_1)}
由于不知道$p(x_3|x_1)p(x_4|x_1)$，所以无法得出$p(x_3, x_4) = p(x_3)p(x_4)$。

②.x已知时,$p(x, y, z) = p(z)p(x|z)p(y|x)$
$p(y, z|x) = \frac{p(x,y,z)}{p(x)} = \frac{p(z)p(x|z)}{p(x)}p(y|x) = p(z|x)p(y|x)$
所以$y \perp z|x$

x未知时,$p(x, y, z) = p(z)p(x|z)p(y|x)$
$p(y, z) = \sum_x p(x, y, z) = p(z)\sum_x p(x|z)p(y|x)$
无法得出$p(y, z) = p(y)p(z)$

机器学习 (周志华) 参考答案 第八章 集成学习

机器学习 (周志华西瓜书) 参考答案 总目录

- http://blog.csdn.net/icefire_tyh/article/details/52064910

```
学了那么多分类器，但真正应用起来都是将这些分类器进行集成学习，达到更好的泛化效果。
```

1. 假设硬币正面朝上的概率为 p，反面朝上的概率为 1-p。令 H(n) 代表抛 n 次硬币所得正面朝上的次数，则最多 k 次正面朝上的概率为 $P(H(n) \leqslant k) = \sum_{i=0}^{k} \binom{n}{i} p^i (1-p)^{n-i}$。对$\delta > 0$，$k = (p-\delta)n \leqslant e^{-2\delta^2 n}$试推导出 (8.3)。

取$p - \delta = \frac{1}{2}$，则$\delta = p - \frac{1}{2} = \frac{1}{2} - \varepsilon$
$P(H(n) \leqslant \frac{n}{2}) = \sum_{i=0}^{\frac{n}{2}} \binom{n}{i} p^i (1-p)^{n-i} \leqslant e^{-2(\frac{1}{2}-\varepsilon)^2 n} = e^{-\frac{1}{2}(1-2\varepsilon)^2 n}$

2. 对于 0/1 损失函数来说，指数损失函数并非仅有的一致替代函数。考虑式 (8.5)，试证明: 任意随机函数$l(-H(x)f(x))$，若对于$H(X)$在区间$[-\infty, \delta](\delta > 0)$上单调递减，则$l$是 0/1 损失函数的一致替代函数。

总损失$L = l(-H(x)f(x))P(f(x)|x) = l(-H(x))P(f(x) = 1|x) + l(H(x))P(f(x) = 0|x)$
$H(x) \in -1, 1$，要使 L 最小，当$P(f(x) = 1|x) > P(f(x) = 0|x)$时，会希望$l(-H(x)) < l(H(x))$, 由于$l$是递减的，得$H(x) > -H(x)$，的$H(x) = 1$。同理当$P(f(x) = 1|x) < P(f(x) = 0|x)$时,$H(x) = -1$。

$l(-H(x)f(x))$是对$H(X)$的单调递减函数，那么可以认为$l(-H(x)f(x))$是对$(-H(X))$的单调递增函数
此时$H(x) = argmax_{y \in 0,1} P(f(x) = y|x)$, 即达到了贝叶斯最优错误率，说明$l$是 0/1 损失函数的一致替代函数。

3. 自己编程实现一个 AdaBoost, 以不剪枝决策树为基学习器，在西瓜数据集 3.0α上训练一个 AdaBoost 集成，并与图 8.4 比较。

- http://blog.csdn.net/icefire_tyh/article/details/52193816

4.GradientBoosting 是一种常用的 Boosting 算法，是分析其与 AdaBoost 的异同。

GradientBoosting 与 AdaBoost 相同的地方在于要生成多个分类器以及每个分类器都有一个权值，最后将所有分类器加权累加起来
不同在于:
AdaBoost 通过每个分类器的分类结果改变每个样本的权值用于新的分类器和生成权值，但不改变每个样本不会改变。
GradientBoosting 将每个分类器对样本的预测值与真实值的差值传入下一个分类器来生成新的分类器和权值 (这个差值就是下降方向)，而每个样本的权值不变。

5. 试编程实现 Bagging, 以决策树桩为学习器，在西瓜数据集 3.0α上训练一个 Bagging 集成，并与 8.6 进行比较。

- http://blog.csdn.net/icefire_tyh/article/details/52194516

6. 试述为什么 Bagging 难以提升朴素贝叶斯分类器的性能。

Bagging 主要是降低分类器的方差，而朴素贝叶斯分类器没有方差可以减小。对全训练样本生成的朴素贝叶斯分类器是最优的分类器，不能用随机抽样来提高泛化性能。

7. 试述随即森林为什么比决策树 Bagging 集成的训练速度快。

随机森林不仅会随机样本，还会在所有样本属性中随机几种出来计算。这样每次生成分类器时都是对部分属性计算最优，速度会比 Bagging 计算全属性要快。

8.MultiBoosting 算法与 Iterative Bagging 的优缺点。

MultiBoosting 由于集合了 Bagging，Wagging，AdaBoost，可以有效的降低误差和方差，特别是误差。但是训练成本和预测成本都会显著增加。
Iterative Bagging 相比 Bagging 会降低误差，但是方差上升。由于 Bagging 本身就是一种降低方差的算法，所以 Iterative Bagging 相当于 Bagging 与单分类器的折中。

9. 试设计一种可视化多样性度量，并与 k - 误差图比较。

暂无

10. 试设计一种能提升 k 近邻分类器性能的集成学习算法。

可以使用 Bagging 来提升 k 近邻分类器的性能，每次随机抽样出一个子样本，并训练一个 k 近邻分类器，对测试样本进行分类。最终取最多的一种分类。

机器学习 (周志华) 参考答案 第九章 聚类

机器学习 (周志华西瓜书) 参考答案 总目录

- http://blog.csdn.net/icefire_tyh/article/details/52064910

```
聚类由于不存在客观标准，任何有点道理的角度都能提出新的聚类算法。
```

1. 试证明：$P \geq 1$时，闵可夫斯基距离满足度量的四条基本性质；$0 \leq P < 1$时，闵可夫斯基距离不满足直递性；P 趋近于无穷大时，闵可夫斯基距离等于对应分量的最大绝对距离，即
$lim_{p\to\infty}(\sum_{u=1}^{n}|x_{iu}-x_{ju}|^p)^{\frac{1}{p}} = max_u|x_{iu}-x_{ju}|$。

显然$(\sum_{u=1}^{n}|x_{iu}-x_{ju}|^p)^{\frac{1}{p}}$满足非负性，同一性和对称性。只考虑直递性：
$dist_{mk}(x_i,x_j) = (\sum_{u=1}^{n}|x_{iu}-x_{ju}|^p)^{\frac{1}{p}}$
$dist_{mk}(x_i,x_k) = (\sum_{u=1}^{n}|x_{iu}-x_{ku}|^p)^{\frac{1}{p}}$
$dist_{mk}(x_j,x_k) = (\sum_{u=1}^{n}|x_{ju}-x_{ku}|^p)^{\frac{1}{p}}$
显然当x_{ku}不在x_{iu}和x_{ju}之间，$|x_{iu}-x_{ku}|+|x_{ju}-x_{ku}|$会明显大于$|x_{iu}-x_{ju}|$, 使得直递性成立。
取特殊情况：对所有的 u，都有$x_{ju} \leq x_{ku} \leq x_{iu}$，
设$a_u = |x_{iu}-x_{ku}|$，$b_u = |x_{ku}-x_{ju}|$，则$|x_{iu}-x_{ju}| = a_u + b_u$，且$a_u, b_u \geq 0$
所以根据闵可夫斯基不等式
$(1).p \geq 1$时，有$(\sum a_u^p)^{\frac{1}{p}} + (\sum b_u^p)^{\frac{1}{p}} \geq (\sum(a_u+b_u)^p)^{\frac{1}{p}}$，直递性成立。
$(2).p \leq 1$时，有$(\sum a_u^p)^{\frac{1}{p}} + (\sum b_u^p)^{\frac{1}{p}} \leq (\sum(a_u+b_u)^p)^{\frac{1}{p}}$，直递性不成立。
(3). 根据极限法则可以得出
$lim_{p\to\infty}(\sum_{u=1}^{n}|x_{iu}-x_{ju}|^p)^{\frac{1}{p}} = (max_u|x_{iu}-x_{ju}|)lim_{p\to\infty}(\sum_{u=1}^{n}(\frac{|x_{iu}-x_{ju}|}{max_u|x_{iu}-x_{ju}|})^p)^{\frac{1}{p}} = max_u|x_{iu}-x_{ju}|$
由于$p \to \infty$，$\sum_{u=1}^{n}(\frac{|x_{iu}-x_{ju}|}{max_u|x_{iu}-x_{ju}|})^p = 1$
所以得证。

2. 同一样本空间中的集合 X 与 Z 之间的距离可以通过 “豪斯多夫距离” 计算：
$dist_H(X,Z) = max(dist_h(X,Z), dist_h(Z,X))$，其中
$dist_h(X,Z) = max_{x\in X} minz \in Z||x-z||_2$证明豪斯多夫距离满足四条距离度量基本性质。

这个距离有点不好理解，形象点说$dist_h(X,Z)$是对 X 内所有点做圆并慢慢扩大，遇到的第一个属于 Z 的点时的半径，就是当前点的$minz \in Z||x-z||_2$，而所有半径中最大的一个，就是 dist_h(X,Z)。由此可以看出$dist_h(X,Z)$是 X 中的样本往 Z 做圆，$dist_h(Z,X)$是 Z 中的样本往 X 做圆，所以两者不一定相当，取较大的一个作为距离。
非负性，同一性和对称性是很明显的，省略。
直递性：由于表达式太抽象，解析法不知道怎么泛化去解。取个简单的特殊情况，假设集合是连续的区间，在平面上用圆来表示

如图可知，
$dist_h(X,Y)$是 X,Y 两个圆的距离加上直径，也就是圆心距加上 X 的半径减去 Y 的半径。
即$dist_h(X,Y) = |o_x - o_y| + r_x - r_y$
那么$dist_H(X,Y)$就是圆心距加上 X，Y 中较大的半径减去较小的半径

$dist_H(X,Y) = |o_x-o_y| + max(r_x,r_y) - min(r_x,r_y)$
$dist_H(X,Z) = |o_x-o_z| + max(r_x,r_z) - min(r_x,r_z)$
$dist_H(Y,Z) = |o_y-o_z| + max(r_y,r_z) - min(r_y,r_z)$

显然$|o_x-o_z| + |o_y-o_z| \geq |o_x-o_y|$
假设$r_x \geq r_y$
当$r_z \geq r_x$时
$max(r_x,r_y) - min(r_x,r_y) = r_x - r_y \leq r_z - r_y = max(r_y,r_z) - min(r_y,r_z)$
当$r_z \leq r_y$时

$max(r_x, r_y) - min(r_x, r_y) = r_x - r_y \leq r_x - r_z = max(r_x, r_z) - min(r_x, r_z)$
当$r_x \geq r_z \geq r_y$时
$max(r_x, r_y) - min(r_x, r_y) = r_x - r_y = (r_x - r_z) + (r_z - r_y) = max(r_y, r_z) - min(r_y, r_z) + max(r_x, r_z) - min(r_x, r_z)$
所以$dist_H(X, Z) + dist_H(Y, Z) \geq dist_H(X, Z)$

3. 试析 k 均值算法能否找到最小化 (9.24) 的最优解。

不能，因为 k 均值算法只是局部最有的近似算法，只能找到初始化均值附近的局部最优解，无法找到全局最优解。

4. 编程实现 k 均值算法，设置三组不同的 k 值，三组不同的初始中心点，在西瓜数据集 4.0 上进行实验，并讨论什么样的初始中心有利于取得好结果。

- http://blog.csdn.net/icefire_tyh/article/details/52224394

5. 基于 DBSCAN 的概念定义，若 x 为核心对象，有 x 密度可达的所有样本构成的集合 X，试证明：X 满足连接性和最大性。

由题意显然最大性是满足的。
连接性：假设x_i为核心对象，由于x_j可以由x_i密度可达。则存在核心对象x_k，使得x_i与x_k密度直达，x_k与x_j密度直达。由于x_k是核心对象，则x_k与x_i密度直达。且密度直达是密度可达的子集，所以x_k与x_j密度可达，x_k与x_i密度可达，所以x_i与x_j密度相连。

6. 试析 AGNES 算法使用最小距离和最大距离的区别。

最大距离可以认为是所有类别先生成一个能包围所有类内样本的最小圆，然后所有圆同时慢慢扩大相同的半径，哪个类圆能完全包围另一个类则停止，并合并这两个类。由于此时的圆已经包含另一个类的全部样本，所以称为全连接。
最小距离则是扩大时遇到第一个非自己类的点就停止，并合并这两个类。由于此时的圆只包含另一个类的一个点，所以称为单连接。

7. 聚类结果中若每个簇都有一个凸包，且凸包不相交，则称为凸聚类。试析本章介绍的哪些聚类方法只能产生凸聚类，哪些能产生非凸聚类。

显然高斯混合聚类是一种可能产生非凸的聚类方式。
高斯混合聚类并不是去最小化类间均方误差，而是通过概率模型来计算每个样本属于每个分类的概率，最后概率最大的。高斯混合概率模型不再单纯与均值相关，而且和方差 (协方差) 有关，所以不再一定得到凸聚类。
其余如 k-means,LVQ,DBSCAN,AGNES 都是凸聚类。

8. 试设计一个聚类性能度量指标，并与 9.2 比较。

略。

9. 是设计一个能用于混合属性的非度量距离。

混合属性中的连续属性，可以标准的距离为距离参数。
对于非连续属性的距离参数，可以将属性看成是字符串，计算距离时，一对一对比两个字符串各个位置的值相同的次数，并通过计算求出距离。比如两个字符串长度分别为l_i, l_2，一共对比$l_i l_2$次，相同的字符为 k, 那么距离参数可以认为是$t(1 - \frac{k}{l_i l_2})$，t 为一个合适缩放倍数。
然后通过指数函数将距离参数影射成真正的距离，此时的距离是非度量距离。

10. 实现一种能自动确定聚类数的改进 k 均值算法，编程实现并在西瓜数据集上运行。

- http://blog.csdn.net/icefire_tyh/article/details/52224612

机器学习 (周志华) 参考答案 第十章 降维与度量学习

机器学习 (周志华西瓜书) 参考答案 总目录

- http://blog.csdn.net/icefire_tyh/article/details/52064910

1. 编程实现 k 邻近分类器，在西瓜数据集 3.0α上比较其与决策树分类边界的异同。

- http://blog.csdn.net/icefire_tyh/article/details/52243081

2. 令err, err^*分别表示最近邻分类器与贝叶斯最优分类器的期望错误率，试证明：$err^* \leq err \leq err^*(2 - \frac{|Y|}{|Y|-1} * err^*)$。

由书 226 页可知 $err = 1 - \sum_{c \in Y} P^2(c|x)$
$err* = 1 - max_{c \in Y} P(c|x)$
设$c^* = argmax_{c \in Y} P(c|x)$
则$err* = 1 - P(c^*|x)$
左边：
由于$P(c^*|x) = max_{c \in Y} P(c|x)$

$\sum_{c \in Y} P^2(c|x)$可以看出$P(c|x)$的带权线性组合，总权值为 1，结果肯定会小于他们的最大值$P(c^*|x)$

即：$P(c^*|x) > \sum_{c \in Y} P^2(c|x)$

所以$err^* \leq err$

右边：
$err = 1 - \sum_{c \in Y} P^2(c|x) = 1 - P^2(c^*|x) - \sum_{c \in Y - c^*} P^2(c|x) = (2 - err^*)err^* - \sum_{c \in Y - c^*} P^2(c|x)$

当剩余的$P(c|x)$全部相等时，$\sum_{c \in Y - c^*} P^2(c|x)$取最小值，即$P_{c \in Y - c^*}(c|x) = \frac{err^*}{|Y|-1}$

$\sum_{c \in Y - c^*} P^2(c|x) \geq \frac{(err^*)^2}{|Y|-1}$

所以$err \leq (2 - err^*)err^* - \frac{(err^*)^2}{|Y|-1} = err^*(2 - \frac{|Y|}{|Y|-1} * err^*)$

3. 在对高维数据降维前应该先进性 “中心化”，常见的方法是将协方差阵XX^T转换为XHH^TX^T，其中$H = I - \frac{1}{m}11^T$，试讲述原因。

假设 X 是 k*m 矩阵，其中 m 是样本数，k 是维度。
中心化即使每个样本减去中心$\bar{x}$，即$\bar{X} = X - \bar{x} * (1^{m*1})^T$
又$\bar{x} = \frac{1}{m} X * 1^{m*1}$
所以$\bar{X} = X - \frac{1}{m} X * 1^{m*1} * (1^{m*1})^T = X(I - \frac{1}{m}11^T) = XH$
其中 1 是1^{m*1}。

4. 在实践中，协方差阵 XX^T 的特征值分解常由中心化后的样本矩阵 X 的奇异值分解替代，试讲述原因。

假设样本阵 X 是 k*m 矩阵，其中 m 是样本数，k 是维度。
使用协方差阵求特征值分解时，协方差阵与属性的维度成平方比，这需要占用大量的空间。当属性维度与样本数差距巨大时，这种不必要的开销更加明显。
对样本矩阵进行奇异值分解，很明显非 0 奇异值的个数 m’ ，肯定不会大于样本数和属性维度较小的一个 (一般情况 k>>m)，这样使得求出来的特征向量阵为$k * m'(m' \leq m)$，显然当$m << k$时，$m'k$的开销会远远小于k^2。

5. 降维中涉及的投影矩阵通常要求是正交的，试述正交非正交投影矩阵用于降维的优缺点。

当特征向量两两正交时，任何两种属性都是相互独立的，其中一个的取值不会影响另一个。但是属性并非全部不相关，比如书上说的，西瓜的体积和重量，显然是正相关的。这时如果两个属性的特征向量不成交会有更好的效果。

6. 试使用 matlab 的 PCA 函数对人脸数据进行降维，并观察前 20 个特征向量对应的图像。

- http://blog.csdn.net/icefire_tyh/article/details/52243639

7. 试述核化线性降维与流型学习之间的联系与优缺点。

非线性核的线性降维与流型学习都属于非线性降维。
核化线性降维有线性降维的优点，比如 KPCA 与保留了最主要的特征，计算方法简单，使用非线性核可以实现非线性降维。缺点一个是核化后的缺点，复杂度与样本总数成正比，当样本很多时复杂度会很高；另外由于 PCA 使用的正交空间，如果属性相关性比较大，会出现不好的结果。
流型学习：流形在局部具有欧式空间的性质，能用欧氏距离来进行距离计算。它的优点就是把高维中不能直接计算的距离使用局部距离来累计表示。比如 Isomap, 它使用测地线距离来表示高维距离。缺点一是如果本的分布不均匀，导致设置的 k 近邻或 e 距离近邻中存在短路与断路的存在，不利于计算全局距离。二是并没有特别好的方法去计算新样本的低维坐标。

8.k 近邻与 e 近邻图存在短路和断路问题会给 Isomap 造成困扰，设计一个来缓解。

短路是由于 k 与 e 设置过大造成的，断路是因为 k 与 e 太小或者样本分布问题造成的。
比如 5 个远离其他样本，但他们 5 个靠的很近，导致 5 近邻时他们与其他所有样本距离无穷远而导致断路。
这里设计一条规则来解决这个问题：
假设每个点寻找到一个近邻，就连上一条边
那么对每个点遍历寻找近邻的时候，至少要加入一条新的边。
这样可以解决断路问题。至于新增加的边，就是该样本未连边的样本中离它最近的样本。

9. 设计一个方法为新样本找到 LLE 降维后的低维坐标。

如书 236 的方法，为新样本x寻找它的近邻，设集合为Q
通过最小化平方误差$min|x-\sum_{i\in Q}w_ix_i|^2$求出各近邻点的权值。
把求出权值与近邻点在低维坐标线性组合求出新样本的坐标。
$z=\sum_{i\in Q_z}w_iz_i$

10. 试述如何确保度量学习产生的距离能满足距离度量四条基本性质。

要保证度量对称性与同一性，需要保证度量与$|x_i-x_j|$相关。
假设度量为$dist_M(x_i,x_j)=(x_i-x_j)^TM(x_i-x_j)$，M 是各属性的相关阵。
要保证非负性，使得 M 必须是非负定的，而且属性相关满足交换率，也就是说 M 是对称矩阵。
所以 M 可以写成PP^T，P 是正交矩阵。
$dist_M(x_i,x_j)=(x_i-x_j)^TM(x_i-x_j)=(P^T(x_i-x_j))^T(P^T(x_i-x_j))$
此时相当于将初始的 x 映射成了P^Tx，显然三角不等式
$|P^T(x_i-x_k)|+|P^T(x_j-x_k)|\geq|P^T(x_i-x_j)|$恒成立
当P^Tx_k在P^Tx_i和P^Tx_j之间时等号成立。

机器学习 (周志华) 参考答案 第十三章 半监督学习

机器学习 (周志华西瓜书) 参考答案 总目录

- http://blog.csdn.net/icefire_tyh/article/details/52064910

1. 试推导出式 (13.5)~(13.8)。

Υ_{ji}为样本x_j属于第 i 个高斯混合成分的后验概率。
可知$\Upsilon_{ji} = p(\theta = i|x_j)$
所以推出 (13.5)：$\Upsilon_{ji} = \frac{\alpha_i p(x_j|u_i,\Sigma_i)}{\sum_i^N \alpha_i p(x_j|u_i,\Sigma_i)}$

使用 (13.4)：$LL(D_l \cup D_u)$对u_i求偏导
让$\frac{\partial LL(D_l \cup D_u)}{\partial u_i} = 0$
化简得$\sum_{x_j \in D_u} \Upsilon_{ji}(u_i - x_j) + \sum_{y_j \in D_l \cap y_j = i}(u_i - x_j) = 0$
求出u_i得到式 13.6
同理推出式 (13.7)，式 (13.8)。

2. 试基于朴素贝叶斯模型推导出生成式半监督学习算法。

朴素贝叶斯模型假设样本所有属性相互独立。
参数表示：

- a 表示属性集合
- x 样本属性
- y 表示有标记样本的分类
- c 表示样本的生成伪分类
- θ表示属性的类条件概率，θ_{ijk}表示第 i 个属性值为a_{ij}分类为 k 的概率
  对于每个样本，将它分类为 k 的概率为
  $P(c = k|x;\theta) = \prod_i P(c = k|x_i = a_{ij};\theta) = \prod \theta_{ijk}$
  贝叶斯判定为$h_{nb}(x) = argmax_{k \in Y} P(c = k|x;\theta)$
  初始化：
  根据训练样本计算出最初的θ，并对无标记样本生成最初的伪标记。
  使用 EM 算法来求解伪标记：
  E 步：使用拉普拉斯平滑标记对已经有标记的样本进行属性类概率估计，求出θ。
  M 步：使用当前的θ对无标记样本集合重新进行分类，获得新的伪标记。
  直到无标记样本的伪标记不再变化。

3. 假设数据由混合专家模型生成，即数据是基于 k 个成分混合的概率密度生成：$p(x|\theta) = \sum_i^k \alpha_i p(x|\theta_i)$，其中θ为模型参数。假设每个混合成分对应一种类别，但每个类别可能包含多个混合成分。试推导出生成式半监督学习算法。

与书上高斯混合成分不同的是，混合专家模型的分类可以对应多个混合成分。
由于与高斯混合只是混合成分与类别对应不同，可以列出相同的目标函数，如式 (13.4)
$LL(D_l \cup D_u) = \sum_{x_j,y_j \in D_l} ln(\sum_i^n \alpha_i p(x|\theta_i)p(y_j|\theta = i, x_j)) + \sum_{x_j \in D_u} ln(\sum_i^n \alpha_i p(x|\theta_i))$
假设第 i 个混合成分生成分类 j 的概率为β_{ij}
则 (13.4) 改写为$LL(D_l \cup D_u) = \sum_{x_j,y_j \in D_l} ln(\sum_i^n \alpha_i p(x|\theta_i)\beta_{i(k=y_j)} + \sum_{x_j \in D_u} ln(\sum_i^n \alpha_i p(x|\theta_i))$
如果β_{ij}定义为常数，则与高斯混合模型进行相同的 EM 算法就行。
否则加入对β参数的优化。
具体初始化与公式参考本章引用 [Miller and Uyar,1997]。

4. 实现 TSVM 算法，选择两个 UCI 数据集，将其中 30% 作为测试样本，10% 作为训练样本，60% 作为无标记样本，分别训练出利用无标记样本的 TSVM 和仅利用有标记样本的 SVM，并比较其性能。

- http://blog.csdn.net/icefire_tyh/article/details/52345053

5. 对未标记样本进行标记指派与调整过程中可能出现类别不平衡问题，试给出该问题改进的 TSVM 算法。

将C_u拆分为C_+与C_-，将C_+作为参数，使$C_- = \frac{m_+}{m_-}C_+$。
将式 (13.9) 改为
min $\frac{1}{2}||w||_2^2 + C_l \sum_{i \in D_l} \xi_i + C_+ \sum_{i \in D_u \cap \hat{y}_i = 1} \xi_i + C_- \sum_{i \in D_u \cap \hat{y}_i = -1} \xi_i$
约束条件做相应更改。

6.TSVM 对未标记样本进行标记指派与调整过程涉及很大的计算开销，试设计一个高效的改进算法。

在标记调整过程中，可以考虑每次将最有可能指派错误的样本进行调整，即正负伪标记样本中松弛变量最大且大于 1 的样本进行标记更改，可以减少迭代的次数。

7. 试设计一个能对新样本进行分类图半监督算法。

图半监督算法不会直接对新样本进行分类，可行的办法一是将新样本作为无标记样本再次进行图半监督算法。或者使用已有标记的样本训练一个学习器，再对新样本分类。

8. 自训练是一种比较原始的半监督学习方法：它现在有标记的样本上学习，然后在无标记的样本上获得伪标记，再在全部样本上进行重复训练，分析该方差有何缺陷。

由于训练样本远远少于无标记样本，如果将全部无标记样本的伪标记直接作为训练样本，将导致很多样本属于噪声样本，十分影响分类器的准确度。应该进行局部伪标记调整来优化分类器，而不是直接使用全部的伪标记重复训练分类器。

9. 给定一个数据集，假设属性集包含两个示图，但事先并不知道哪些属性属于哪个示图，试设计一个算法将两示图分离出来。

根据已有的数据集将属性集分成二个集合，若遍历求最优解是指数级的复杂度。
考虑使用一种局部最优的方法：
设置两个集合，初始时一个集合包含全部属性，另一个为空。
从集合 1 随机选取一个集合放入集合 2。
然后开始迭代：
每轮迭代从集合 1 选择一个属性放入集合 2，使得集合 2 属性的训练误差减小量与集合 1 属性的训练误差增加量最小。
当两个属性集合训练结果符合停止标准时，停止迭代。

10. 试为图 13.7 算法第 10 行写出违约检测算法。

- http://blog.csdn.net/icefire_tyh/article/details/52345055