

第九周补讲内容

陈欣鸿 陈昱夫 毛润泽

目录

- 问题
- 补充知识点
- 决策树实现细节
- 决策树优化
- 任务布置

报告

- 页数
- 原理
- 伪代码
- 结果
- 思考题

原理

(1) 贝叶斯定理 (Bayes rule)

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

其中 $P(A|B)$ 是在 B 发生的情况下 A 发生的可能性。

在贝叶斯定理中，每个名词都有约定俗成的名称：

- $P(A)$ 是 A 的先验概率或边缘概率。之所以称为“先验”是因为它不考虑任何 B 方面的因素。
- $P(A|B)$ 是已知 B 发生后 A 的条件概率，也由于得自 B 的取值而被称作 A 的后验概率。
- $P(B|A)$ 是已知 A 发生后 B 的条件概率，也由于得自 A 的取值而被称作 B 的后验概率。
- $P(B)$ 是 B 的先验概率或边缘概率，也作标准化常量 (*normalized constant*)。

按这些术语，Bayes 定理可表述为：

$$\text{后验概率} = (\text{相似度} * \text{先验概率}) / \text{标准化常量}$$

也就是说，后验概率与先验概率和相似度的乘积成正比。

另外，比例 $P(B|A)/P(B)$ 也有时被称作标准相似度 (*standardised likelihood*)，Bayes 定理可表述为：后验概率 = 标准相似度 * 先验概率

推导：

根据条件概率的定义。在事件 B 发生的条件下事件 A 发生的概率是

$$P(A|B) = \frac{P(A \cap B)}{P(B)}.$$

同样地，在事件 A 发生的条件下事件 B 发生的概率

$$P(B|A) = \frac{P(A \cap B)}{P(A)}$$

整理与合并这两个方程式，我们可以找到

$$P(A|B)P(B) = P(A \cap B) = P(B|A)P(A)$$

这个引理有时称作概率乘法规则。上式两边同除以 $P(B)$ ，若 $P(B)$ 是非零的，我们可以得到贝叶斯定理：

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

贝叶斯定理是关于随机事件 A 和 B 的条件概率（或边缘概率）的一则定理。

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

其中 $P(A|B)$ 是在 B 发生的情况下 A 发生的可能性。

在贝叶斯定理中，每个名词都有约定俗成的名称：

- $P(A)$ 是 A 的先验概率或边缘概率。之所以称为“先验”是因为它不考虑任何 B 方面的因素。
- $P(A|B)$ 是已知 B 发生后 A 的条件概率，也由于得自 B 的取值而被称作 A 的后验概率。
- $P(B|A)$ 是已知 A 发生后 B 的条件概率，也由于得自 A 的取值而被称作 B 的后验概率。
- $P(B)$ 是 B 的先验概率或边缘概率，也作标准化常量 (*normalized constant*)。

按这些术语，Bayes 定理可表述为：

$$\text{后验概率} = (\text{相似度} * \text{先验概率}) / \text{标准化常量}$$

也就是说，后验概率与先验概率和相似度的乘积成正比。

另外，比例 $P(B|A)/P(B)$ 也有时被称作标准相似度 (*standardised likelihood*)，Bayes 定理可表述为：

$$\text{后验概率} = \text{标准相似度} * \text{先验概率}$$

从条件概率推导贝叶斯定理

根据条件概率的定义。在事件 B 发生的条件下事件 A 发生的概率是

$$P(A|B) = \frac{P(A \cap B)}{P(B)}.$$

同样地，在事件 A 发生的条件下事件 B 发生的概率

$$P(B|A) = \frac{P(A \cap B)}{P(A)}.$$

整理与合并这两个方程式，我们可以找到

$$P(A|B)P(B) = P(A \cap B) = P(B|A)P(A).$$

这个引理有时称作概率乘法规则。上式两边同除以 $P(B)$ ，若 $P(B)$ 是非零的，我们可以得到贝叶斯定理：

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}.$$

(2) 朴素贝叶斯处理分类问题 (Classification for Naïve Bayes)

Bernoulli Model (伯努利模型) : a document is represented by a feature vector with **binary** elements taking value 1 if the corresponding word is present in the document and 0 if the word is not present.

$$p(x_k|e_i) = \frac{n_{e_i}(x_k)}{N_{e_i}} \quad p(e_i) = \frac{N_{e_i}}{N}$$

where $n_{e_i}(x_k)$ is the number of documents of emotion e_i in which x_k is observed, and N_{e_i} and N is the number of documents with emotion e_i and total documents, respectively.

Multinomial Model (多项式模型) : a document is represented by a feature vector with integer elements whose value is the **frequency** of that word in the document.

$$p(x_k|e_i) = \frac{nw_{e_i}(x_k)}{nw_{e_i}} \quad p(e_i) = \frac{N_{e_i}}{N}$$

where $nw_{e_i}(x_k)$ is the number of times word x_k occurs in documents with emotion e_i , and nw_{e_i} is the total number of words occurs in documents with emotion e_i .

(3) 朴素贝叶斯处理回归问题 (Regression for Naïve Bayes)

To predict emotion e_i of test document $X_3 = (x_3, x_4)$, we need to estimate:

$$\arg \max_{e_i} p(e_i|X) = \arg \max_{e_i} \sum_{j=1}^M \prod_{k=1}^{K'} p(x_k|e_i, d_j) p(d_j, e_i)$$

(4) 拉普拉斯平滑 (Laplace Smoothing)

Notice that if the word x_k in test document does not occur in the training set, $p(x_k|d_j, e_i)$ will be zero and thus cause the resulting value becoming 0.

Solution: **Laplace Smoothing!** (拉普拉斯平滑)

- regression model:

$$p(x_k|d_j, e_i) = \frac{x_k + 1}{\sum_{k=1}^K x_k + K}$$

- classification model:

$$\text{Bernoulli: } p(x_k|e_i) = \frac{n_{e_i}(x_k) + 1}{N_{e_i} + 2}$$

$$\text{Multinomial: } p(x_k|e_i) = \frac{nw_{e_i}(x_k) + 1}{nw_{e_i} + V_{e_i}}$$

where nw_{e_i} is the total number of words in documents with emotion e_i , and V_{e_i} is the number of non-repetitive words with label e_i .

伪代码反例

```
void AplusB() {
    ifstream "A", "B"; //读取文件 A, B
    rows = max{A.rows, B.rows};
    cols = max{A.cols, B.cols};

    push A -> C;
    push B -> C; //去重，词向量的值加 1 (相同项的值)
    sort(C); //排序，不重复按行再列
    ofstream fout << "C"; //写入文件 C
```

伪代码反例

```
void Div() {
    ifstream fin>>;           //读取数据集文件
    while(getline){             //取一条完整的文本
        strtok(",\t");         //去掉序号、情感权重，得到分析文本

        while(strtok(", ")){   //取一个单词

void smatrix() {
    ofstream fout<< //输出到文件
    cs                  //文本总数（行数）
    rs                  //单词总数（列数）
    for cs//遍历每个文本
        for ws //遍历每个单词
            if(wf != 0)//词频不为 0，即存在
                cnt++;
            cnt;          //输出三元表第三行
    //遍历，找到单词存在的位置，或是在 onehot 矩阵中的坐标
    fout << x << y << 1;
```

伪代码反例

第三步：构建所有文本的不重复词向量 `total_words(vector<string>)` 和每个文本的词向量
`train_set(vector<each_row>)`

将上一步分出的每个单词，与 `total_words` 中已有的单词进行比较，若非已有单词
则将其加入到 `total_words` 中

将单词与该(第 `i` 个)文本的词向量比较，若不存在，则将其加入到
`train_set[i].row_words` 中；若存在则增加相应的 `num` 值

第四步：获取 one-hot 矩阵

打开文件 `onehot.txt`

将 `i` 从 0 到 `row_size(文本数量)-1`：

对第 `i` 个文本的词向量 `train_set[i]` 根据单词在 `total_words` 中位置 `position` 来从

结果

Knn 回归-测试集结果:

A	B	C	D	E	F	G
textid	anger	disgust	fear	joy	sad	surprise
1	0.088756	0	0.218283	0.183867	0.289888	0.219206
2	0.070767	0	0.195792	0.200479	0.385266	0.147696
3	0.103633	0	0.24463	0.293582	0.197396	0.160759
4	0.104061	0	0.166424	0.39876	0.113612	0.217144
5	0.067391	0	0.10605	0.489761	0.130146	0.206652
6	0.110435	0	0.273227	0.127544	0.319231	0.169562
7	0.007549	0	0.107002	0.424915	0.015912	0.444622
8	0.11562	0	0.283973	0.191539	0.270856	0.138012
9	0	0.060011	0.199845	0.35603	0.076909	0.307205
10	0.117811	0	0.230617	0.272414	0.165707	0.213451
11	0.020326	0	0.143423	0.475613	0.13281	0.227829
12	0.148325	0	0.291524	0.22124	0.217365	0.121546
13	0.090285	0	0.096877	0.479275	0.10736	0.226203
14	0.11766	0	0.208597	0.347275	0.241376	0.085091
15	0.116727	0	0.379226	0.203046	0.171326	0.129674
16	0.116101	0	0.263185	0.25227	0.211457	0.156986
17	0.097866	0	0.207287	0.409234	0.125429	0.160184
18	0.113285	0	0.159483	0.326618	0.1561	0.244613
19	0.087718	0	0.169184	0.36579	0.146317	0.230992
20	0.016726	0	0.1779	0.35932	0.230259	0.215795
21	0.116467	0	0.260992	0.309613	0.172158	0.140769
22	0.066658	0	0.211246	0.32387	0.250918	0.147385
23	0.08492	0	0.183316	0.35213	0.243527	0.136107
24	0.118993	0	0.074374	0.022605	0.124245	0.659783

Knn 回归-验证集结果:

A	B	C	D	E	F	G
textid	anger	disgust	fear	joy	sad	surprise
1	0.153261	0	0.304928	0.181966	0.214596	0.145248
2	0.038158	0	0.159768	0.39959	0.118547	0.283937
3	0.091285	0	0.229783	0.292669	0.220007	0.166257
4	0.130093	0	0.216042	0.355166	0.173065	0.125633
5	0.025574	0	0.112089	0.429346	0.153048	0.279944
6	0.083212	0	0.160249	0.347983	0.142676	0.265879
7	0.142601	0	0.340099	0.047838	0.372533	0.09693
8	0.089098	0	0.177352	0.463886	0.11746	0.152204
9	0.113072	0	0.236335	0.300509	0.137332	0.212752

textid, lable	21, joy	278, joy
0, joy	22, joy	279, sad
1, joy	23, fear	280, joy
2, joy	24, fear	281, joy
3, joy	25, sad	283, joy
4, joy	26, fear	284, joy
5, joy	27, joy	285, joy
6, sad	28, sad	286, joy
7, joy	29, joy	287, joy
8, joy	30, joy	288, sad
9, joy	31, fear	289, joy
10, joy	32, joy	290, joy
11, joy	33, joy	292, joy
12, joy	34, sad	293, joy
13, sad	35, joy	294, joy
14, joy	36, joy	295, joy
15, joy	37, joy	296, joy
16, joy	38, joy	297, joy
17, joy	39, joy	298, joy
18, joy	40, joy	299, joy
19, joy	41, joy	301, joy
20, fear	42, joy	302, joy

上述为 k 取 18 时的部分结果

(2) Knn 回归

通过验证集得到的部分结果:

textid	anger	disgust	fear	joy	sad	surprise
0	0.0779	0.0759	0.1106	0.258	0.254	0.2236
1	0.0771	0.0432	0.1077	0.4079	0.0935	0.2706
2	0.0941	0.0781	0.1368	0.307	0.1562	0.2278
3	0.0784	0.0428	0.142	0.243	0.2326	0.2613
4	0.0866	0.0318	0.1655	0.3915	0.1192	0.2055
5	0.07	0.04	0.1142	0.4159	0.0274	0.3325
6	0.0834	0.055	0.2057	0.2205	0.285	0.1504
7	0.0791	0.0722	0.1099	0.3856	0.1017	0.2315
8	0.0903	0.0443	0.219	0.2396	0.1583	0.2484
9	0.0915	0.0659	0.2306	0.2616	0.1596	0.1908
10	0.0767	0.0531	0.145	0.3703	0.1692	0.1857
11	0.0795	0.0325	0.1757	0.3289	0.2105	0.1728
12	0.0865	0.0619	0.2087	0.1958	0.1789	0.2683

结果

b) KNN 回归 (K=5)

```
F:\KNN\回归.exe  
请输入KNN的K值:  
5  
0.0585883 0.0799963 0.0611823 0.37495 0.185824 0.239459  
0.0535945 0.0506644 0.163269 0.238342 0.203907 0.290223  
0.111688 0.0990018 0.0332846 0.221198 0.342274 0.192554  
0.146976 0.0737881 0.0476874 0.30172 0.0570793 0.372749  
0.111081 0.0517904 0.150995 0.209492 0.220285 0.256357  
0.11414 0.0369678 0.16657 0.412397 0.134213 0.135712  
0.15918 0.054 0.20966 0.15922 0.22834 0.1896  
0.22918 0.08128 0.2264 0.03734 0.25944 0.16636  
0.156932 0.07171 0.136509 0.259095 0.143778 0.231975  
0.0584983 0.0472921 0.316068 0.312217 0.135062 0.130863  
0.116886 0.0532645 0.158886 0.283589 0.140502 0.246855  
0.0994161 0.0581282 0.165934 0.227641 0.169475 0.279406  
0.1185 0.054 0.16108 0.25922 0.14246 0.26474  
0.110741 0.0459916 0.318581 0.155206 0.175252 0.194228  
0.107191 0.0347172 0.145707 0.116038 0.353809 0.242538  
0.1775 0.0376619 0.198341 0.111278 0.220069 0.25515  
0.0646676 0.0504398 0.122943 0.163871 0.259126 0.338952  
0.0545379 0.0901387 0.19063 0.0428058 0.42141 0.200478  
0.1185 0.054 0.16108 0.25922 0.14246 0.26474  
0.1185 0.054 0.16108 0.25922 0.14246 0.26474  
0.1185 0.054 0.16108 0.25922 0.14246 0.26474  
0.167214 0.0692051 0.217695 0.178546 0.175789 0.191461  
0.1185 0.054 0.16108 0.25922 0.14246 0.26474  
0.1185 0.054 0.16108 0.25922 0.14246 0.26474  
0.151461 0.0535097 0.2635 0.266382 0.0887667 0.17638  
0.124862 0.0591012 0.223504 0.159223 0.236725 0.196584  
0.124695 0.05209 0.207269 0.153588 0.30344 0.158913  
0.1185 0.054 0.16108 0.25922 0.14246 0.26474
```

d) NB 回归

F:\NB回归.exe

	0.0855954	0.0958038	0.240323	0.214904	0.186788	0.176586
0.0194957	0.0234991	0.188453	0.182396	0.115632	0.470525	
0.0367057	0.0225971	0.323116	0.10816	0.181709	0.327712	
0.0703912	0.0303988	0.236332	0.117943	0.410903	0.134033	
0.0720597	0.0533015	0.162984	0.186625	0.345808	0.179221	
0.101333	0.0614538	0.0919733	0.330214	0.173561	0.241465	
0.176457	0.082811	0.38477	0.0	0.279439	0.0708966	
0.0869913	0.0869913	0.0434957	0.521648	0.108689	0.152185	
0	0.0421887	0.249816	0.166923	0	0.538587	
0.0836757	0.0806639	0.258518	0.0866186	0.305938	0.184586	
0.0376984	0	0.104093	0.316584	0.192399	0.347565	
0.0265053	0.0442088	0.24785	0.0265053	0.557612	0.0973195	
0.0739247	0.121919	0.0869592	0.0325351	0.448779	0.235884	
0	0	0	0.790116	0.209339		
0.1065531	0.07735	0.206551	0.245147	0.204678	0.159743	
0	0	0.7386	0	0.2614		
0.102142	0.0469575	0.137864	0.327348	0.147905	0.237784	
0	0.0831314	0.579263	0.176752	0.157775		
0.0169842	0	0.347747	0.0581879	0.175462	0.365287	
0.0107097	0.280746	0.0278265	0.172928	0.0283593	0.479431	
0.133314	0.124412	0.222223	0.21782	0.137815	0.164416	
0	0.0933325	0.546667	0.0800017	0.279982		
0.0509408	0.0403064	0.0495311	0.465565	0.128695	0.264962	
0.070992	0.188844	0.109076	0.0804343	0.376739	0.173915	
0.0601736	0	0.123167	0.431898	0.0814108	0.295901	
0.102199	0.0593644	0.183455	0.213301	0.24207	0.19961	
0.195504	0.131177	0.234677	0.0127004	0.35957	0.0663719	
0.0131015	0.0124519	0.0136537	0.236215	0.0159847	0.708593	
0.0968692	0.0967073	0.206502	0	0.587019	0.0128998	
0.19224	0.0447109	0.22515	0.0447736	0.490762	0	

· 评测指标展示即分析（如果实验题目有特

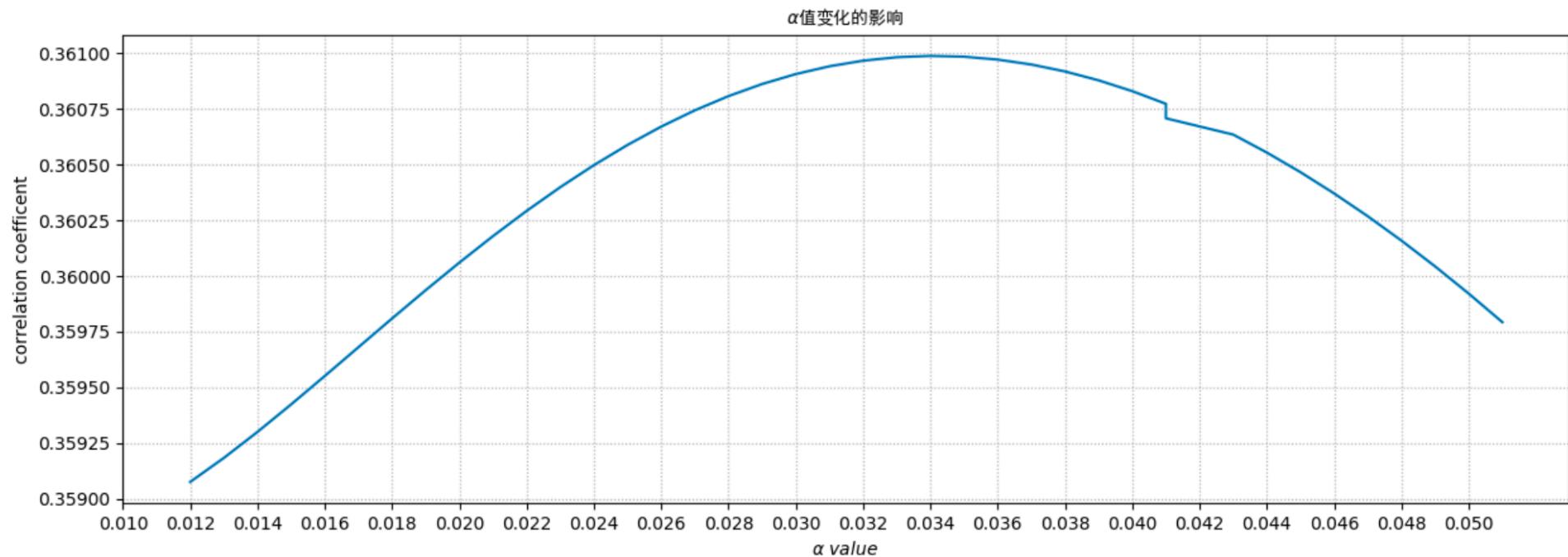
a) KNN 分类

	A	B	C	D	E	F	G	H	I	J
1	marijuana surprise	joy		FALSE	107					
2	thousands fear	joy		FALSE						
3	king coun joy	joy		TRUE						
4	rig threat fear	fear		TRUE						
5	new zeala sad	joy		FALSE						
6	england o joy	joy		TRUE						
7	kidnapper fear	joy		TRUE						
8	boy fear	joy		TRUE						
9	more sleep surprise	joy		FALSE						
10	a shot in fear	joy		FALSE						
11	lawmaker surprise	joy		FALSE						
12	snow brin sad	joy		FALSE						
13	opera can sad	joy		FALSE						
14	anna nico sad	joy		FALSE						
15	on myspa surprise	sad		FALSE						
16	on myspa surprise	sad		TRUE						

结果

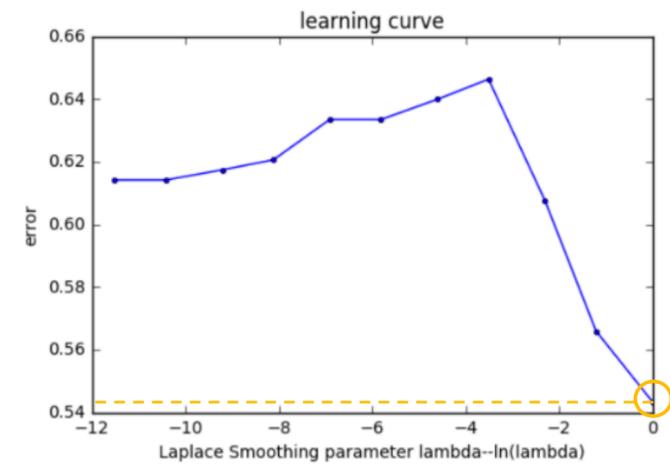
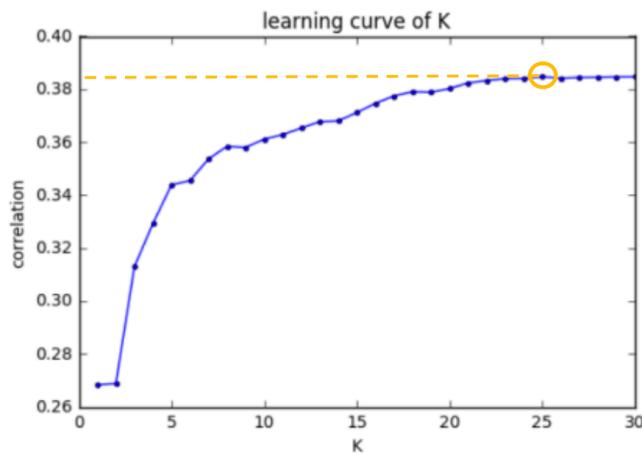
NB回归

在不同的 α 值下，相关系数变化为

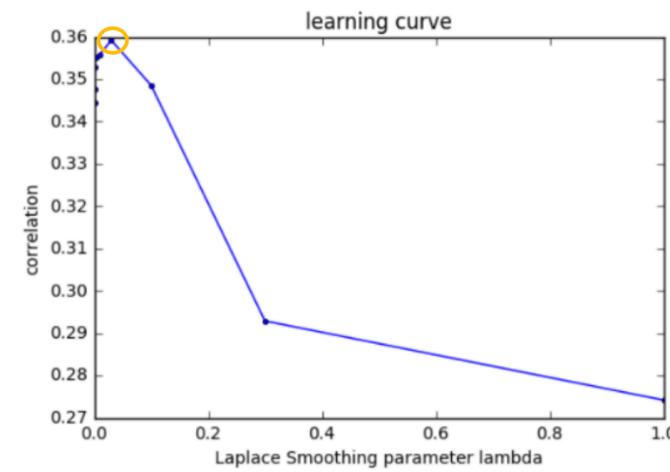
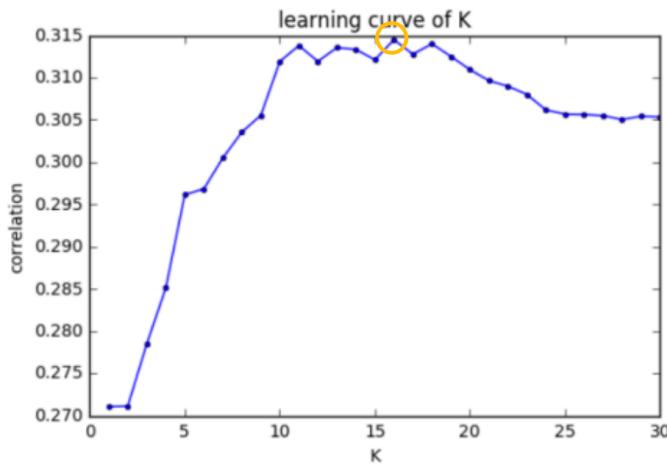


结果

曼哈顿



欧氏



思考题

3. IDF 数值有什么含义？TF-IDF 数值有什么含义

百度百科：“IDF 的主要思想是：如果包含词条 t 的文档越少，也就是 n 越小，IDF 越大，则说明词条 t 具有很好的类别区分能力。”

3. 在稀疏矩阵程度不同的时候，曼哈顿距离和欧氏距离表现有什么区别？为什么？

(好像) 无区别；

4、伯努利模型和多项式模型分别有什么优缺点?

思考题



对在情感 e 下，单词 x 出现的概率：

伯努利模型是这样算的：

$$P(x|e) = \frac{\text{情感为 } e \text{ 下包含单词 } x \text{ 的文本数}}{\text{情感为 } e \text{ 下的文本总数}}$$

而多项式模型是这样算的：

$$P(x|e) = \frac{\text{情感为 } e \text{ 下 } x \text{ 出现的总次数}}{\text{情感为 } e \text{ 下单词的总数}}$$

优缺点不明。

思考题

TF-IDF 算法是一种统计方法，用于统计一个字词在一个文件集中的一份文件的重要性。一般来说，字词的重要性随着它在文件中出现的次数成正比增加，但同时会随着它在语料库中出现的频率成反比下降。**TF-IDF** 算法正是结合这两者的特点，统计词频并用逆向文件频率进行加权来计算出该字词的重要程度。这样可以过滤掉常见的词语，保留重要的词语。（以上大部分内容来自网络）

TF-IDF 是一种统计方法，用以评估一字词对于一个文件集或一个语料库中的其中一份文件的重要程度。字词的重要性随着它在文件中出现的次数成正比增加，但同时会随着它在语料库中出现的频率成反比下降

简介

 编辑

TF-IDF是一种**统计方法**，用以评估一字词对于一个文件集或一个语料库中的其中一份文件的重要程度。字词的重要性随着它在文件中出现的次数成正比增加，但同时会随着它在**语料库**中出现的频率成反比下降。**TF-IDF**加权的各种形式常被**搜索引擎**应

归一化

- 归一化 \neq 让和等于1
- 让和等于 1 是一种归一化
- 归一化方法可参考 lab2 课件
- 对什么归一化？
- 为什么归一化？

归一化

	售出数量	评价(1-5分)	是否值得购买	与4距离	
1	20000	2	否	4002	✓
2	10000	5	是	6001	
3	10000	1	否	6003	
4	16000	4	?		

KNN决定		
如果是用街区距离，会由谁决定？		不值得购买

- 对列归一化？对行归一化？全局一起归一化？

归一化

	售出数量	评价(1-5分)	是否值得购买	与4距离	
1	1	0.25	否	0.9	
2	0	1	是	0.85	√
3	0	0	否	1.35	
4	0.6	0.75	?		

- 归一化会抵消由原数据可能产生的影响模型的效果
- **可能会提高准确度**

概率

$$\sum_x p(x|a, b) = ?$$

$$\sum_a p(x|a, b) = ?$$

贝叶斯公式

- $P(A|B)$ 是 ?
- $P(B|A)$ 是 ?
- $P(B)$ 是 ?
- $P(A)$ 是 ?

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

概率

$$\sum_x p(x|a, b) = ?$$

$$\sum_a p(x|a, b) = ? \quad p(x|b)$$

贝叶斯公式

- $P(A|B)$ 是后验概率
- $P(B|A)$ 是似然度
- $P(B)$ 是标准化常量
- $P(A)$ 是先验概率

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

拉普拉斯平滑

$$\text{Multinomial: } p(x_k|e_i) = \frac{n w_{e_i}(x_k) + 1}{n w_{e_i} + V}$$

- 如果平滑是为了防止概率乘积为 0，为什么要用拉普拉斯平滑的方式？为什么不直接默认成一个很小的数值，比如 0.001，也不会怎么影响其他的概率？
- 平滑的时候分子的 1 怎么理解？分母的 V 怎么理解？
- 如果分子加的是一个实数值 α ，分母加什么？为什么？

拉普拉斯平滑

$$\text{Multinomial: } p(x_k|e_i) = \frac{n_{w_{e_i}}(x_k) + 1}{n_{w_{e_i}} + V}$$

- 分子既然是该单词在这个词袋里面出现的词数，如果加上1，说明每个单词都多出现了一次
- 既然每个单词都在这个词袋里面多出现了一次，说明这个词袋现在多了V个单词

拉普拉斯平滑

$$p(x_k | d_j, e_i) = \frac{x_k + 1}{\sum_{k=1}^K x_k + K}$$

- 这个数值算出来是什么值？分子的 x_k 是什么？
- 如果要用 TFIDF 做特征，还是这样平滑吗？

词袋

- 文本被看成是无序的词汇集合，忽略语法或者是单词的顺序
 - I read a book today
 - today I read a book
 - read I a book today
 - book today a read I
-
- 什么时候就导致了模型是词袋模型？
 - 忽略单词的顺序或者语法，可能会有怎样的问题？

总结

- 不认真对待 = 没有分数
- 认真做了实验不好好写报告 = 没有分数
- 报告好好写了但是补交 = 降分
- 只知道照着PPT说的来做， 没有自己的思考 = 学完这门课还是等于没学
- Project 抱大腿的可能性不大， 就算组队也是分开打分， 一个高分一个没分是完全可能的

决策树实现细节

剪枝

- 预剪枝
 - 限制决策树深度
 - 为每个叶节点的数据个数设定阈值
 - 为准确率的增益设定阈值
- 后剪枝
 - 基于错误率剪枝
 - 基于模型复杂度剪枝

剪枝

- 后剪枝
 - 基于错误率剪枝

$$E(L) = \sum_{\text{All leaf } l} e(l) \quad E(L') = \sum_{\text{All leaf } t \text{ except } k} e(l)$$

- 基于模型复杂度剪枝

$$e_c = \frac{\sum_{l=1}^L (r(n_l) + \zeta(n_l))}{\sum_{l=1}^L m(n_l)}$$

优化

- 剪枝
- 尝试使用随机森林
- 改变叶子节点的基础模型

课堂任务

- 上课当天完成三种决策树的选择属性的函数中任意一种的编写
- ID3 => 根据信息增益选择
- C4.5 => 根据信息增益率选择
- C&RT => 根据 GINI 指数选择
- 输入：完整数据集
- 输出：在当前数据集下应该选择 **哪种属性** 作为根节点的属性
- 只提交代码文件，命名为“**1535xxxx_xiaoming.xx**”，xx视编程语言而定，多版本的在后面加“_v1”，数字视提交版本号而定
- 该任务一定程度参与实验报告评分，请务必认真完成
- DDL: **上课当天晚上十二点**，交到 ftp 上的 **Week 9 Mission**