中山大学数据科学与计算机学院

移动信息工程专业-人工智能

本科生项目报告

（2017-2018 学年秋季学期）

课程名称：**Artificial Intelligence**

教学班级	15M1	专业（方向）	移动信息工程（互联网）
组号	26	组名	阿通队
学号	15352116	姓名	洪子淇

一、 Project 最终结果展示

1. 最终结果

26	阿通队	0.897016	0.632199	105.9321	39	13	69	48

2. 组内分工

二元分类	多元分类	回归问题
黄铖杰	洪子淇	黄国正

3. 个人工作

效果：差（0~20%），一般（20%~40%），较好（40%~55%），好（55%~）

日期	工作	是否完成	效果（模型）
12/12~12/18	文本分析	完成	/
	简单进行文本处理	完成	/
12/19	尝试简单统计模型	完成	较好
12/20~12/22	实现独热下的 KNN	完成	一般
	实现 tf 矩阵下的 KNN，LR	完成	一般
12/23~1/1	利用 word2vec 实现 KNN，LR 算法（清洁）	完成	一般，较好
	利用 word2vec 实现 KNN，LR 算法（不清洁）	完成	一般，较好
1/2	在上一次基础上实现 BPNN	完成	一般
1/3~1/9	改进简单统计模型，准备 pre	完成	好
1/10~1/12	写报告	完成	/

二、 工作流程

1. 数据集分析

- 二元分类

数据集包含 48000 个样本, 十三列属性, 其中 10 列是连续值, 3 列是离散值：

特征	A	B	C	D	E	F	G
Max	3856	/	5455	6970	6958	7111	5292
Min	1882	/	1	-67	-165	29	3
Var	77727.82	/	7412.952	303843.8	277393.1	2556073	4180.666

特征	H	I	J	K	L	M
Max	7063	6831	7139	1	3	2.999988
Min	2	0	0	0	0	2.000026
Var	204648.3	194961.8	1882836	0.041399	0.640604	0.083295

离散列分别为 B、K、L，分别具有 5，2，2 特征。观察数据发现有些列的属性方差较大，极差较大。可见数据分布差别较大，需要对数据进行归一化，减少属性之间距离的影响。而对于离散列，由于不知道属性，难以说明属性下的特征是否等价，需要通过训练得出特征之间的距离，若等价就需要进行编码，否则不需要进行编码。

- 多元分类

数据集包含 62522 个样本，每个样本有多个句子，文本中出现较多关于餐厅服务及食物味道等单词，推测这是一个关于餐厅评论的数据集，其中 LOW,MID,HIG 分别代表差评，中评，好评。

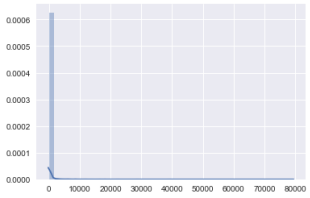
① 从字符串意义上对于评论没有影响的单词或字符等：

样本中存在变音字母以及非 ASCII 字符，如”åãâàäáçæœèéêèëïíîñôóòöûùüú®死八婆°¿™¡ ”等，为无效字符。

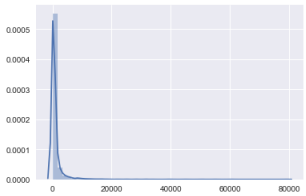
② 观测训练集下每个单词出现的频数：

图一为所有单词出现的频数分布图，图二，图三在图一上约束分别为频数大于 100，200，可观测到在评论中经常出现的单词大约只有前面 5000

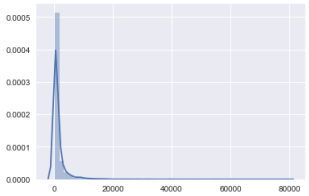
个左右，出现少的单词可能是俗语或者与大众评论相关度低。



图一



图二



图三

③ 对于样本单词进行计数分析（已除去①中提到的无效字符以及人称代词）表一以 low 降序排列，表二对前面的 low，mid，hig 归一化后，对行取方差，以方差降序排列，排在前面表示单词在标签下的离散程度越高，即对类别区分度越高。（表中数据取前 3 与后 3）

表一：

id	word	total num	low num	mid num	hig num
1	the	502062	173927	201810	126325
2	and	323406	101973	133384	88049
3	to	215867	78717	83398	53752
...	...	...	...	...	...
76883	riceyummy	1	0	0	1
76884	loch	1	0	0	1
76885	tne	1	0	0	1

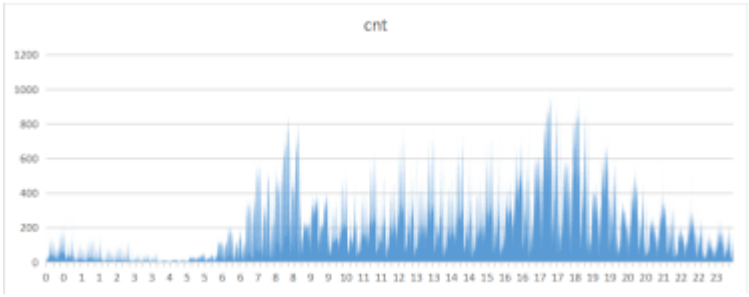
表二：

id	word	low nor	mid nor	hig nor	var
1	was	0.022539852	0.018289989	0.014961964	9.62E-06
2	!	0.004523203	0.008984064	0.0120463	9.54E-06
3	and	0.035142442	0.039386373	0.040454046	5.26E-06
...	...	...	...	...	...
76883	perk	9.65E-06	9.74E-06	9.65E-06	2.03E-15
76884	mccormick	4.14E-06	4.13E-06	4.14E-06	3.95E-19
76885	multitude	4.14E-06	4.13E-06	4.14E-06	3.95E-19

分析可知，训练集中 the，and，to 出现次数很高，离散程度也高，这是因为生活习惯中会经常用到这些单词，但不具备什么情感色彩，由于样本本来不均衡，所以不能直接从上面评价单词对分类的影响程度。

- 回归问题

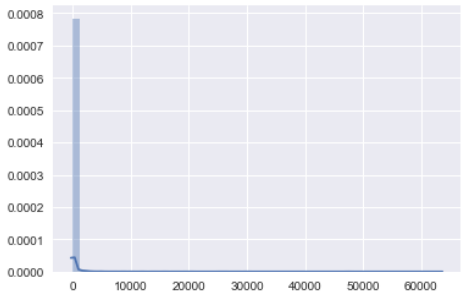
回归数据集具有 16637 个样本，具有 4 列连续属性 :temp、atemp、hum、windspeed 以及三列离散属性 dteday、hr、weathersit。dteday 含有较多的信息，可以将其分为年份或这季度的属性。hr 属性较为重要，由于出行与时间息息相关，早上七八点和傍晚五六点的时候车辆数目最高，符合实际情况：

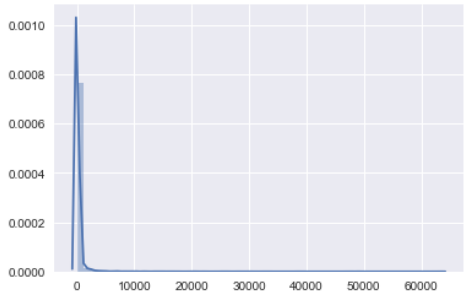


三、 算法简介

1. KNN——LR——BPNN

训练集采用前 10000 个训练样本，验证集选取后 500 个训练样本，onehot 矩阵通过索引下标节省内存及时间的消耗。由于单词出现次数较少对于文本类别区别度不高，在构建 tf 矩阵前先去掉大部分低频词，下图是频数大于 0 及大于 10 的参考图：





使用 KNN 算法预测验证集效果最好的 K 值以及准确率如下：

Type	Onehot	TF
KNN/ac	K=1/ 0.31	K=10/ 0.468

由于 onehot 矩阵中忽略了词频，由于词频更能反映单词之间的区别以及情

感色彩，所以预测效果上不如 tf 矩阵好。

在 tf 矩阵下，使用 LR 以及 BPNN 算法：

LR 要点：加入正则项； 权重向量初始化为 0；动态学习：当损失增加，步长减半；

BPNN 要点：单层神经网络；随机初始化权重向量；对数损失函数；隐层，和输出层采用 Sigmoid 函数激活；输出节点个数为三，分别对应 LOW，MID，HIG，每遍历一次样本更新权重向量，多次迭代..

下面 LR 进行 5000 次迭代, 初始步长为 1 及 BPNN 进行了 300 次迭代的准确率：

LR/ac	0.508
BPNN/ac	0.468

辅助手段Word2Vec, Word2vec直接使用TA提供的文件, 其中LR以及BPNN与上文设置参数一致：

KNN	LR	BPNN
K=30/0.408	0.5919	0.5920

2. 原始的简单统计模型：

文本的基本组成单元是词汇。词汇的特征决定了文本的特征，文本的特征反映了词汇的特征。但是微观上，构成文本的词汇具有相对较大的不确定性，难以准确预测某个具体的词汇会出现在哪种情感类型的文本中，只能通过统计手段，大概描述它出现在不同情感类型的文本中的概率。而在宏观上，大量词汇组合成一个句子，再由句子组合成文本后，文本的情感类型就能够比较好的确定。

一般地，预测多元分类问题模型的计算结果是一个概率向量$\boldsymbol{p}$，$\boldsymbol{p}$的每个分量表示对应类别的概率。

由柯尔莫果洛夫公理，$\boldsymbol{p}$ 满足：$\boldsymbol{p}_{\mathrm{i}} \geq 0$，$\sum \boldsymbol{p}_{\mathrm{i}} = 1$

使用这个向量的来表达预测结果比标签结果包含更多的信息。它保留了数值结果，可以用来计算 logloss，表达每个分类结果的准确度，了解不同标签的相对可能性大小，有利于后期模型的融合处理。

这个模型有以下假设和前提：

1. 某些单词是有特定情感色彩的。比如 delicious 一般出现在好评中，而 just 则一般出现在中差评中。因此这些单词在不同分类的文本中，出现的概率一般不会是均匀的。

2. 每个单词关于类别的分布是独立的，不会受到上下文的影响。

3. 每个单词关于类别的分布是固定的，忽略不同的顾客对餐馆的评价标准的偏差。

4. 忽略否定词的影响。

模型在数据清洗的基础上，主要由**训练词频矩阵**、**预测句子分类**和**预测文本分类**组成。首先，需要对经过清洗的文本数据进行词汇层面上的统计，归纳出每个单词对在不同类型文本中出现的概率，即分类概率向量。然后根据词汇的分类概率向量，计算出句子的分类概率向量。最后再由句子的分类概率向量投票决定文本的概率分布向量，进一步确定文本的分类。

训练词频矩阵：

简单统计每个单词关于各类别的频数，得到一个频数矩阵 D（dictionary）。D 的行数是词汇表的大小，列数为 3，分别对应 LOW, MID, HIG。$\mathrm{D}_{\mathrm{wd,LOW}}$表示包含单词 wd 且分类标签为 LOW 的句子中，wd 的总个数为$\mathrm{D}_{\mathrm{wd,LOW}}$。对每一行进行归一化，就得到了概率分布矩阵（使用频率表示概率是根据最大

似然原则，证明参考附录）。但是，由于每个类别的文本长度不同，如中差评的评论字数一般比好评的多，这样会导致好评的概率分布偏低。为了修正这种偏差，在行归一化之前，先对每一列进行归一化。这样我们就初步得到了**词频矩阵**。

预测句子分类：

一个句子Sen由许多单词$\{v_1, v_2, \cdots, v_n\}$构成。定义：

$$\mathbf{p}(\text{sen}) = (\Pr(\text{LOW}|\text{sen}), \Pr(\text{MID}|\text{sen}), \Pr(\text{HIG}|\text{sen}))$$

$$\mathbf{p}(v) = (\Pr(\text{LOW}|v), \Pr(\text{MID}|v), \Pr(\text{HIG}|v))$$

根据我们的假设，每个单词对分类的结果是独立的，不会受到其他单词的影响。因此，

$$\mathbf{p}(\text{sen}) \propto \prod \mathbf{p}(v_i)$$

由于$\mathbf{p}(v_i) \leq 1$，$\prod \mathbf{p}(v_i)$容易出现浮点数下溢，因此需要在运算过程中作对数变换。

$$\ln \mathbf{p}(\text{sen}) \propto \sum \ln \mathbf{p}(v_i)$$

$$\mathbf{p}(\text{sen}) = \lambda \exp(\sum \ln \mathbf{p}(v_i))$$

令

$$(\alpha, \beta, \gamma) = \left(\sum_{v_i \in \text{sen}} \ln \Pr(\text{LOW}|v_i), \sum_{v_i \in \text{sen}} \ln \Pr(\text{MID}|v_i), \sum_{v_i \in \text{sen}} \ln \Pr(\text{HIG}|v_i) \right)$$

那么

$$\lambda = \exp(\alpha) + \exp(\beta) + \exp(\gamma)$$

$$\mathbf{p}(\text{sen}) = \lambda^{-1}(\exp(\alpha), \exp(\beta), \exp(\gamma))$$

由于$\exp(\alpha), \exp(\beta), \exp(\gamma)$和$\lambda$很接近 0，直接计算还是有可能出现下溢的情况。为了避免小数除法误差[1]，以$\Pr(\text{LOW}|\text{sen})$为例，处理如下：

$$\Pr(\text{LOW}|\text{sen}) = \lambda^{-1}\exp(\alpha)$$

$$= \frac{\exp(\alpha)}{\exp(\alpha) + \exp(\beta) + \exp(\gamma)}$$
$$= (1 + \exp(\beta - \alpha) + \exp(\gamma - \alpha))^{-1}$$

预测测试集：

每个测试文本有多个句子。我们可以对每个句子的进行预测，然后对这些预测结果进行投票决策，票数最大的分类即为文本的分类。

4. 改进版的简单模型

① 在训练词频矩阵阶段，尝试用$\mathbf{p}(\text{sen}) \propto \sum \mathbf{p}(v_i)$代替$\mathbf{p}(\text{sen}) \propto \prod \mathbf{p}(v_i)$

② 在预测训练集阶段，保留概率向量作为预测结果，对这些概率向量进行加权平均，代替投票决策。

③ 由于训练文本中可能有与分类无关的句子，因此要想办法除去。1. 一开始先把统计所有句子内的单词，得到词频矩阵D_0。2. 用D_n预测训练文本内的句子，如果预测结果和原来的类别不一致，就屏蔽这个句子，统计所有未被屏蔽句子里的单词，得到D_{n+1}。3.在最大迭代次数内，当验证集的预测准确率达到最大时，停止迭代 D，用 D 预测测试集。

④ 改进③中的屏蔽条件，假设文本的分类是 MID，那么当句子的概率向量的 MID 分类的概率值低于θ时，屏蔽该句子。

伪代码：

Algorithm 1 Predicting Corpus (adding)

```
for item in test_set do
    pred ← [ ]
    for sen in item.corpus do
        pred += [ ∑_{wd in sen} D_matrix[wd] ]
    end for
    pred ← pred.mean('by row')
    item.res ← pred
    item.label ← ARGMAX(pred)
end for
```

Algorithm 2 Predicting Corpus (multiplying)

```
for item in test_set do
    pred ← [ ]
    for sen in item.corpus do
        sigma ← ∑_{wd in sen} LOG(D_matrix[wd])
        a ← EXP(sigma[1]-sigma[2])
        b ← EXP(sigma[2]-sigma[0])
        c ← EXP(sigma[0]-sigma[1])
        x ← 1/(1+b+c^{-1})
        y ← 1/(1+a^{-1}+c)
        z ← 1/(1+a+b^{-1})
        pred += [[x,y,z]]
    end for
    pred ← pred.mean('by row')
    item.res ← pred
    item.label ← ARGMAX(pred)
end for
```

Algorithm 3 Training Vocabulary Probability Matrix (adding)

```
wd_list ← {'LOW':[ ], 'MID':[ ], 'HIG':[ ]}
for item in train_set do
    wd_list[item.label] += [wd for wd in sen for sen in item.corpus]
end for
D_matrix ← COUNTER(wd_list)
for t←0:max_iteration do
    if t == max_iteration then
        break
    end if
    wd_list ← {'LOW':[ ], 'MID':[ ], 'HIG':[ ]}
    for item in train_set do
        for sen in item.corpus do
            sen_pred ← ∑_{wd in sen} D_matrix[wd]
            if sen_pred[item.label] > θ then
                wd_list[item.label] += [wd for wd in sen]
            end if
        end for
        D_matrix ← COUNTER(wd_list)
    end for
end for
```

最好实验结果：

见下文调参过程中的模型 3.0.1，其准确率为 63.22%。

四、 调参过程

KNN：间断设置不同的 K 值，在准确率较高的附近 K 值设置距离减少。

LR：改变步长，当损失比上一次高的时候步长减半，调节迭代次数。

简单统计模型：由于简单模型具有多个子模型，所以有多重结合方式，调参主要围绕着对句子的加权参数，以及迭代训练矩阵的次数（0-），是否删除停用词展开。

下表是测试集的准确率，其中阈值表示当预测准确类别的概率大于该阈值便保留句子；除去停用词中，0 表示没有去除,2 表示 1 中的停用词除去部分介词；

	除去停用词	训练矩阵次数/阈值	句子权重	原始频数/概率相加/相乘	Accuracy
模型 0.0.0	0	/	相等	原始频数	42.01%
模型 1.0.0	0	/	相等	相加	63.03%
模型 1.0.1	0	/	句子方差	相加	60.64%
模型 1.0.2	1	/	相等	相加	63.18%
模型 1.0.3	2	/	相等	相加	63.21%
模型 1.3.0	0	1/1.0	相等	相加	61.08%
模型 1.3.1	0	1/1.0	句子方差	相加	59.43%
模型 1.3.2	1	1/1.0	相等	相加	61.42%
模型 1.3.3	1	1/1.0	句子方差	相加	59.89%
模型 1.3.7	0	7/1.0	相等	相加	53.49%
模型 2.0.0	0	/	相等	相乘	62.31%
模型 2.0.1	1	/	相等	相乘	62.30%
模型 2.0.2	2	/	相等	相乘	62.17%
模型 3.0.0	2	1/0.3	相等	相加	63.21%
模型 3.0.1	2	1/0.1	相等	相加	63.22%
模型 4.0.0	2	1/0.2	相等	相乘	60.95%

分析：该模型还有多种组合，以及对于阈值的设置还有很大的空间。从上面结果分析可以知道频率相加的效果会比理论中推导相乘的效果好一些。版本 1 的停用词包括了一些主语和介词，版本 2 在版本 1 中除去部分介词，这是考虑到动词与

介词的组合形式。实验验证不除介词准确率会高 0.1%。当训练矩阵的时候单纯考虑预测错误就把句子屏蔽，多次迭代后会过渡清洁样本导致准确率下降。

五、 集成学习方法(AdaBoost)

- AdaBoost

AdaBoost 核心思想是针对同一训练集训练不同的分类器（弱分类器），然后将这些弱分类器组合起来构造出一个强分类器。[1]

小组中在二分类问题使用该算法。弱分类器中使用了 PLA，当设置 1000 个分类器的时候可以得到 82%的准确率，分类器数目越多，时间成本越高，最终准确率在 84%左右，难以上升。

六、 引用

[1]: http://blog.csdn.net/haidao2009/article/details/7514787
[2]: http://blog.csdn.net/omade/article/details/17023091
[3]: https://www.cnblogs.com/zyxu1990/p/3209407.html

七、 课程总结

实验心得：这次的多分类数据涉及自然语言，所以解决办法不能对等具有属性特征下的分类问题。一开始简单从词义层面上单词，没有从语义层面上面仔细观察数据，盲目地套用了课堂上的机器学习模型。前期的模型只是为了刷 rank…后面认真观察之后，发现单词的色彩以及英语语言的习惯，在处理上结合思考，明确自己的目的后改善了一开始的简单统计模型，效果比前期的模型都要好。所以说，没有最好的模型，只有适应数据集的模型，对于本次多分类上的文本分析，简单统计模型效果更好。

课程建议 :虽然严格要求可以促进学习, 不过我觉得课程作业要求可以适当宽松。通过这次实验，我发现比起学习多个模型，多个算法，不如简单入门，针对性对数据进行分析，开放性思考模型与数据之间的关系的学习模式，后者比前者更能诞生新的东西。希望实验或者理论可以开设几节课，专门针对一个问题（分类或回归) 下各种情况进行探讨 :在哪一种情况下适用哪一种模型, 以及适用的理由。或者大作业就是针对一个问题，做出一个库，对应不同的情况进行调用。这样队友之间也可以有更多的交流。模型适用性也较强。（当然这是我个人的想法，这个想法源于…这次大作业设置多个问题，组队和个人感觉不到太大的差别，组内交流较片面，又以 rank 的形式进行，队伍间的交流可能也会被阻塞。）

人工智能的理解和体会 : 这门课学习了很多，针对性来说，这门课可以说是人工智能下的分支——机器学习。学习了很多模型，针对模型进行调参，严格来说还没有脱离人为的挑选模型, 人为创建新模型的步骤。个人理解就是人工下的智能，阶跃性的发展应该就是脱离人工的智脑诞生吧。(希望有生之年可以看到)

八、 附录

多项式分布（似然函数）：

$$f(x, N;\ \theta) = \binom{N}{x_1, x_2, \dots, x_n} \prod_{i=1}^{n} \theta_i^{x_i}$$

其中$X = (x_1, x_2, \dots, x_n)$，$\sum \theta_i = 1$

令 N = 1 ，根据贝叶斯法则，似然函数为：

$$L(\theta) = \prod_i^T f(x^i, 1; \theta) = \prod_{i=1}^{T} \theta_i^{l_i}$$

ln $L(\theta) = \sum_j^m x_j \, ln\theta_j$

其中， x^i表示样本下的第 i 个单词， l_i表示其发生的次数

最大化似然函数 ln $L(\theta)$，已知 $\sum \theta_j = 1$，根据拉格朗日乘数法：

令$F(\theta, \lambda) = lnL(\theta) - \lambda g(\theta)$，其中$g(\theta) = \sum \theta_j = 1$:

那么

$$\frac{\partial F(\theta,\lambda)}{\partial \theta_j} = \frac{l_j}{\theta_j} - \lambda = 0$$

$$\theta_j = \lambda^{-1} l_j$$

所以

$$\theta_j \propto l_j, \quad \theta = \frac{l_j}{\sum l}$$

即每个单词在类别下的概率近似于频率。