**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**

**本科生实验报告**

**（2017-2018学年秋季学期）**

课程名称：**Artificial Intelligence**

# 实验题目

1. **实现KNN算法的分类和回归**
2. **实现NB算法的分类和回归**

# 实验内容

1. 算法原理
   1. K-邻近算法（KNN）

KNN算法，其中NN为最近邻(NearestNeighbor)，K是我们选择的参数。所谓KNN，就是取K个最近的邻居的意思，意思是每个样本都可以用它最接近的K个邻居来代表，就是一种近朱者赤近墨者黑的判断方法。

KNN算法的核心思想是K个最相邻的样本中的大多数属于某一个类别，那么该样本也属于这个类别，并具有这个类别的特性。KNN主要的决策主要就是确定K个最近邻中哪个类别占得多，那么我们需要判断的样本就判给那个类别，实际中K值一般不大，所以KNN判别时都只依赖几个样本。

KNN算法中最核心的两个影响判决结果的是K和距离计算方式，K值的不同直接导致了判决结果的差异，因为这种算法基本就是K个内投票表决，数量变换将很大程度影响到表决结果。其次是距离的计算方式，这里的距离指数要就是用来判断两个对象之间的相似度，不同的距离计算方式，会决定谁离得最近。

KNN算法的用于回归也是可行的，通过找出K个最近邻，将这些最近邻的属性均值，赋值给样本即可得到样本的属性。当然这么做需要优化，例如以距离倒数做加权，通过距离确定影响，使得最近的在样本属性中所占的属性比重越大。也就是最近的影响最大。

* 1. 朴素贝叶斯（NB）

朴素叶贝斯基于叶贝斯定理：

主要思想就是通过先验概率计算后验概率，以训练集作为先验概率和条件概率的计算来源，通过这两个概率再结合叶贝斯定理计算后验概率。

在我们进行叶贝斯分类的时候上述公式变成(此处用了全概率公式)。

其中为先验概率为条件概率。

朴素贝叶斯条件概率独立。此时条件概率变为：

所以此时的分类器为：

此时我们只要最大的那么我们可以直接忽视分母，因为分母都是一致的。

最终我们的公式为：

那么此时我们只需要求解先验概率和条件概率

分类中有两种模型，多项式模型和伯努利模型，多项式模型的计算是基于词而伯努利模型式基于文件。

首先解释一下为什么我的多项式模型改用了单词而不是文件作为先验概率公式，主要就是在我看了很多blog后发现了这个公式，认为这样更加合理，因为多项式模式的条件概率使用的是以单词为单位的，那么理论上先验概率也应该以单词为单位，单位统一比较好。

这两个个模型从公式就可以看出多项式模型更适合我们基于词对于情感判断，如果只是基于文件数，这个判断肯定是不准确的。

计算后验概率时，对于一个文档，多项式模型中，只有在出现过的单词，才会参与后验概率计算，伯努利模型中，没有出现的，但是在全局单词表中出现的单词，也会参与计算，不过是作为1-部分参与计算的，也就是一个惩罚项。但是伯努利模型不会去考虑单词重复的情况，理效果不会有多项式那么好。

我们使用多项式模型计算出各个情感中的概率之后取最大值就是其类别。但是考虑到会有在没有出现过的词，所以要加上拉普拉斯平滑，否则后验概率会被拖累变成0。

这里的拉普拉斯平滑是严谨的拉普拉斯平滑，为0-1间的数值，此处需要根据数据集的大小来调整，如果数据集较小就该小一点不然此时所占比重过大影响判断。

回归在基础上去除先验概率，增加乘上各情感的出现概率，所以此时就变为：

之后只需要把各种情感的和归一化即可。

此处还需要说拉普拉斯平滑：

拉普拉斯平滑是为了解决零概率问题，就是在计算概率训练集中没有出现过，导致概率是0。这是不合理的，不能因为一个词没有就直接的认为这个样本在此处的概率是0。

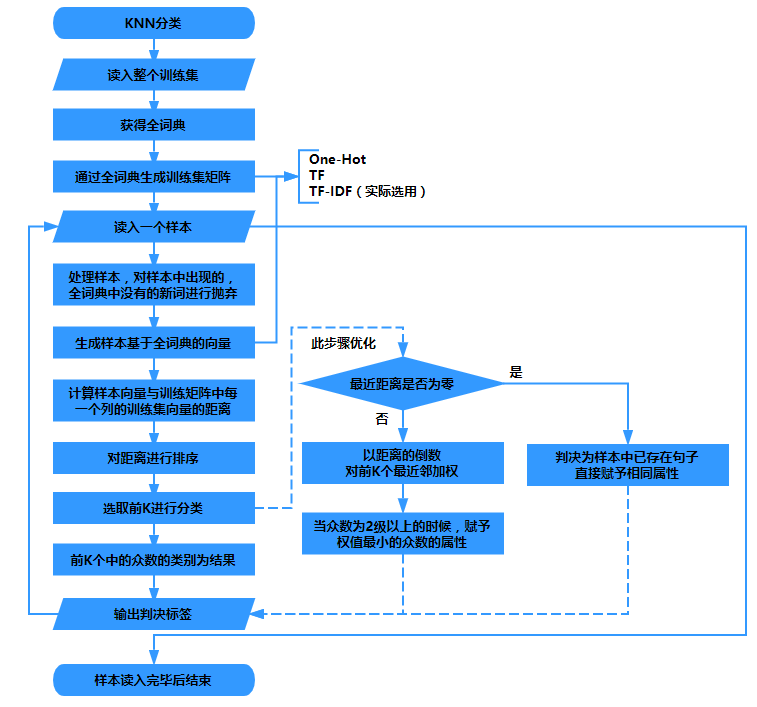
1. 流程图

此处所有的数据处理问题不在本次实验范围内，直接使用上次实验的成果就可以获得所需的数据，此处不在展示数据处理。

同时说明一下，数据处理包括在代码中的DATA.c文件中，KNN和NB都直接调用其中的数据处理类获得所需要的数据。所有的对于矩阵的有归一化等都在数据处理类中。KNN和NB都是直接使用现成数据。

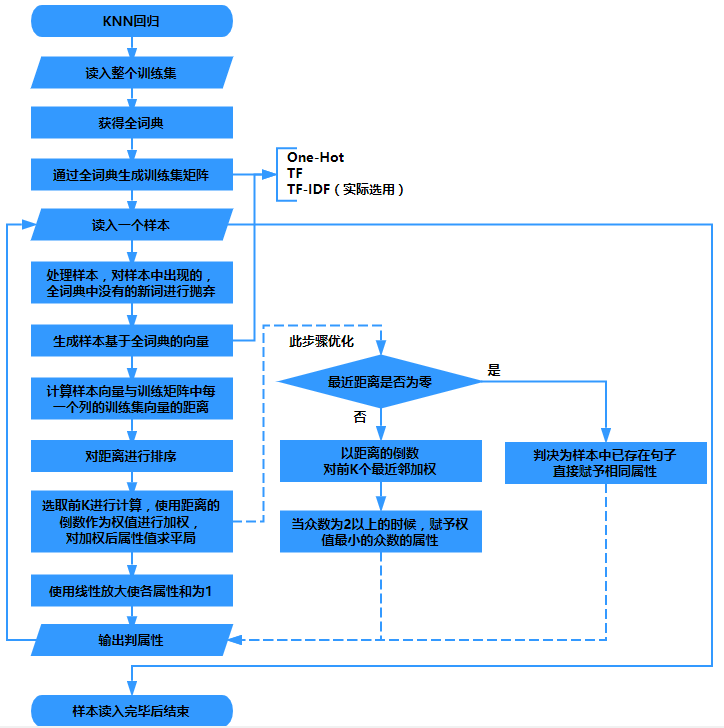
* 1. KNN算法

KNN分类算法。



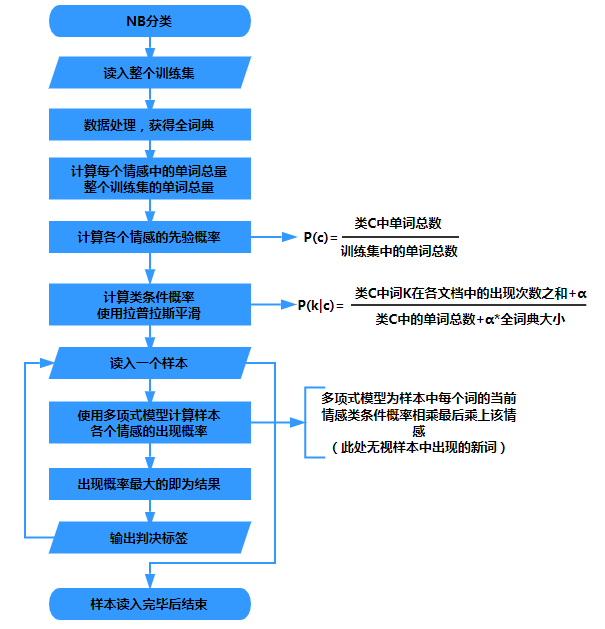
流程中不包括对训练集矩阵进行优化（如归一化等）和不同距离计算的流程。距离计算不是算法的核心，所以不纳入。

KNN回归算法：



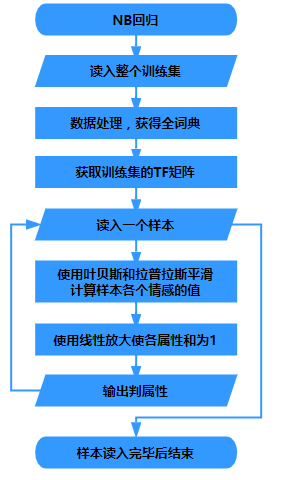
* 1. NB算法

NB分类，使用多项式模型



NB回归：

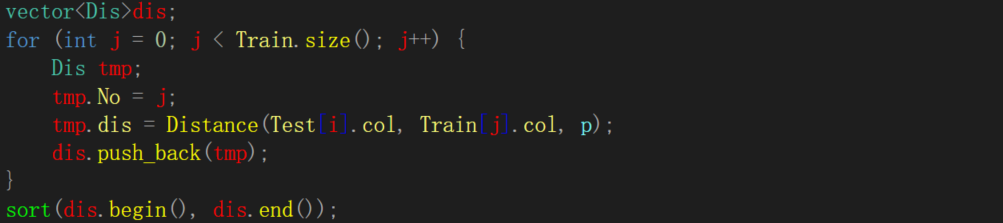
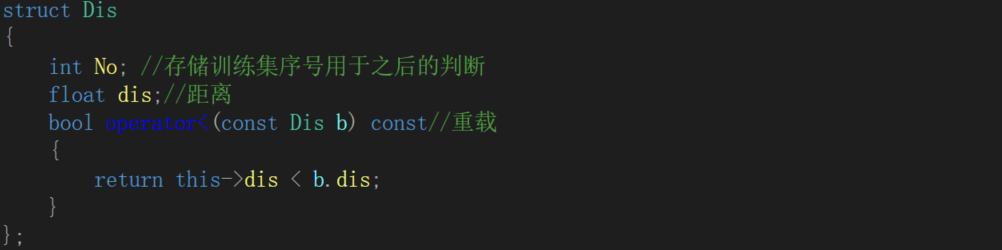
回归计算公式为实验原理中的公式

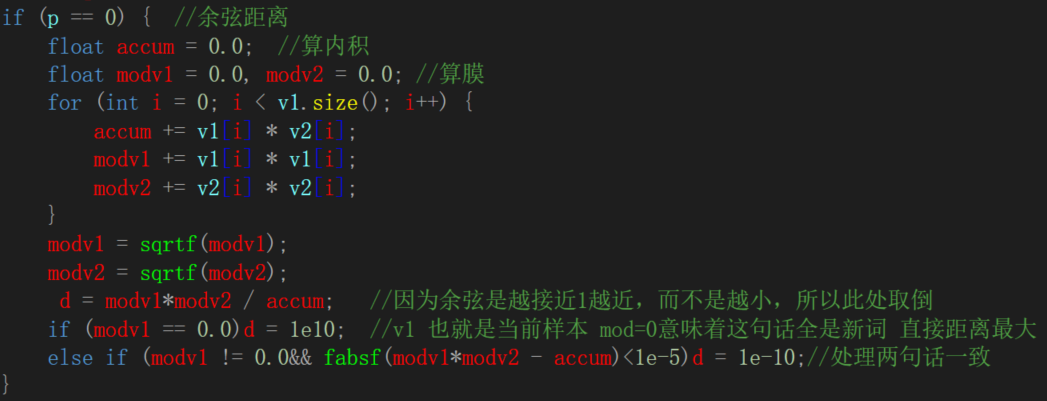
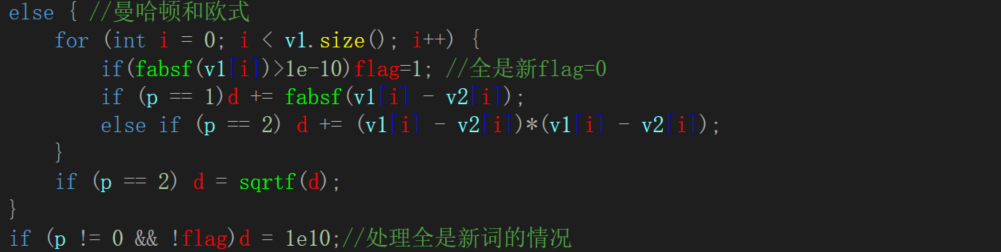


1. 关键代码截图（带注释）

实验中代码分为两个类KNN和NB，下面代码都是类函数。

* 1. KNN算法
     1. KNN分类：

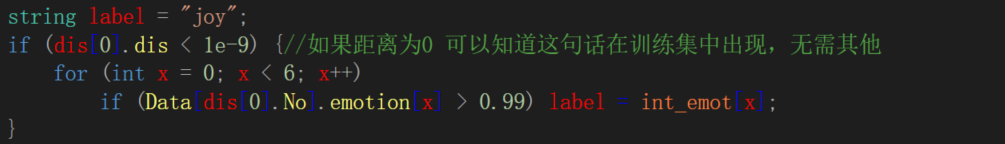
首先计算测试样本和训练样本中所有样本的距离然后根据距离排序。其中p=0为余弦，1为曼哈顿，2为欧式。 此处解释一下Dis这个结构体为什么是储存序号，因为之后回归中排序也需要的到训练集中前K个距离最小的，所以此时存储序号之后再回去取情感，如果只存情感，那么回归需要重新构造。下面为距离计算代码v1为当前样本，v2为训练集中的一个样本，处理余弦距离的时候因为余弦是越接近1，说明两个样本越相似，但是为了使得之后的判断不用附加，余弦距离取倒，此时余弦距离也是越小越近，同时处理一下两句话完全相等的情况。

 在曼哈顿和欧式距离中也要处理样本全是新词的情况。

然后开始KNN算法的核心取前K个距离最近的训练集样本，进

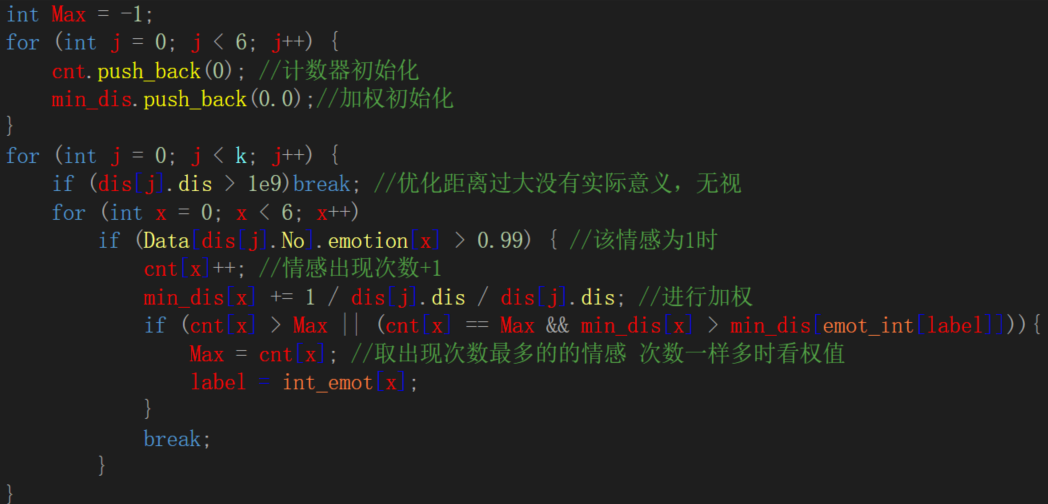
行投票。

首先先给个默认值为joy(因为他出现最多。同时判断距离最近的是不是0,0意味着这两句相同，那么此时我们就可以不要K个了，直接输出相同句的属性。



接着是K个最近邻的投票表决，这个部分特殊处理了两个部分：

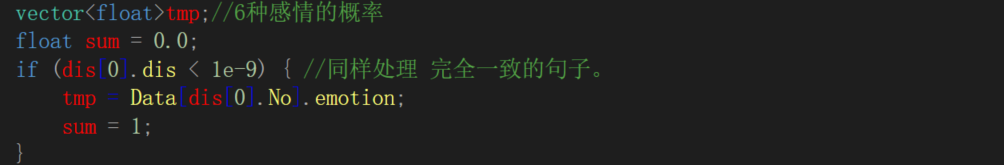
* 对于距离过大的直接无视，退出投票表决，因为此时距离已经过大几乎意味着这两句话没有关系，所以就不需要了。
* 处理众数有多个的情况，使用加权处理众数多的情况，取权值小的作为最终结果，此处加权为距离平方的倒数，这个加权是在测试回归的时候测试出来的。



* + 1. KNN回归

KNN的回归实际上主要部分已经在分类中实现了，分类使用了回归的加权处理。距离计算部分也是相同的。

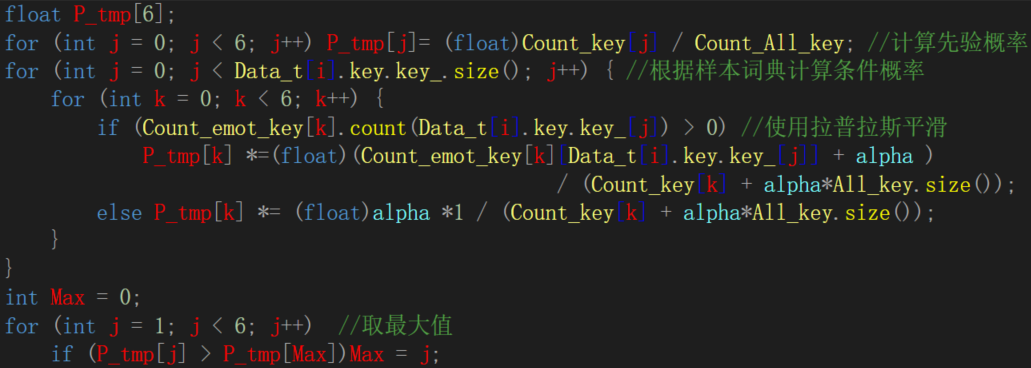
此处展示算法核心部分。

首先还是排除部分一致

之后前K个的平均值和使概率为1的处理，此处一样抛弃距离过大，同时对于全新的句子不用前k个了，直接赋予平均值。测试发现平均值回归比直接1/6处理的效果好0.5%。最后这个线性放大最终版多排除了小于0.02的情况，然后迭代做了一次。但是影响非常微弱，不过总体还是有所提升的。

* 1. NB算法
     1. NB分类

分类算法使用之前理论中的带拉普拉斯平滑的公式，同时将先验概率改成以单词为最小单位而不是以文本。

首先是数据准备，此处的数据准备主要用于分类，回归只能因为公式的原因临时计算更好准备会更加麻烦。分类算法的核心(其实就是实现公式的计算：

* + 1. NB回归

NB回归一样实现公式：



1. 创新点&优化
   1. KNN
      1. KNN算法使用TF-IDF和Z-SCORE标准化加Min-Max线性放缩，再使用余弦距离以达到最优效果。
      2. 算法中针对距离为0进行了直接赋值输出，距离过大的情况优化掉，不在计算。
      3. 对于分类算法引入了回归中的加权方式对计算出每个情感前K个的权值，以权值比较代替众数随机选择。
      4. 回归中针对加权方式进行改进，加权为倒数的平方，实际中效果增加2%。
   2. NB
2. 优化了PPT上给出的拉普拉斯平滑换成更严谨的进行调参
3. 更换多项式模型的先验概率的计算方式使其与条件概率的计算方式一样都是以单词为精度。

(优化代码已经在上面展示，优化结果下方对比)

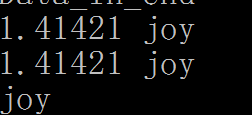
# 实验结果及分析

1. 实验结果展示示例

展示一下小数据集运行结果。

1. KNN分类：

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Words (split by space) | | | | | | | label | 理论距离One-hot 欧式 K=2 |
| this | is | small | | | test | | joy | 1.414 |
| this | is | a | | | test | | joy | 1.414 |
| new | big | test | | | | | sad | 2 |
| have | a | small | | | test | | joy | 1.732 |
| test | is | wrong | | | | | sad | 2 |
| this | is | a | new | small | | test | ? | 所以此处理论上为joy |

实际输出结果

1. KNN回归

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Words (split by space) | | | | | | | anger | disgust | fear | joy | sad | surprise |
| this | is | small | | | test | | 0 | 0 | 0 | 0.8 | 0 | 0.2 |
| this | is | a | | | test | | 0 | 0.2 | 0 | 0.7 | 0.1 | 0 |
| new | big | test | | | | | 0.2 | 0.2 | 0.1 | 0 | 0.5 | 0 |
| have | a | small | | | test | | 0 | 0 | 0 | 0.7 | 0 | 0.3 |
| test | is | wrong | | | | | 0.3 | 0.1 | 0.1 | 0 | 0.5 | 0 |
| this | is | a | new | small | | test | ? | ? | ? | ? | ? | ? |
| One-hot 欧式 K=2 理论值 | | | | | | | 0 | 0.1 | 0 | 0.75 | 0.05 | 0.1 |

实际输出结果

1. NB分类

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Words (split by space) | | | | label |
| this | is | small | test | joy |
| new | big | test | | sad |
| have | a | small | test | joy |
| this | is | | test | ? |

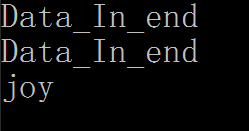
使用修改后的多项式模型无拉普拉斯平滑

P(joy)=8/11,P(sad)=3/11

P(d4|joy)=1/8 \* 1/8 \* 2/8 \* 8/11=0.00284

P(d4|sad)=0 \* 0 \* 1/3 \* 3/11=0

所以理论上为joy

实际

1. NB回归

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Words (split by space) | | | | anger | disgust | fear | joy | sad | surprise |
| this | is | small | test | 0 | 0 | 0 | 0.8 | 0 | 0.2 |
| new | big | test | | 0 | 0 | 0.5 | 0 | 0.5 | 0 |
| have | a | small | test | 0 | 0 | 0 | 0.7 | 0 | 0.3 |
| this | is | test | | ? | ? | ? | ? | ? | ? |

无拉普拉斯平滑理论值为：

P(d4|fear)=0

P(d4|joy)=0.25 \* 0.25 \* 0.25 0.8 = 0.0125

P(d4|sad)=0

P(d4| surprise)=0.25 \* 0.25 \* 0.25 \* 0.2 = 0.003125

概率为1后

P(d4|joy)=0.8

P(d4| surprise)= 0.2

实际值：



**2. 评测指标展示即分析**

1. KNN分类

* One-Hot矩阵部分优化(加权和0距离判断)，欧式距离K（1,30）
* 使用TF-IDF加上所有优化：

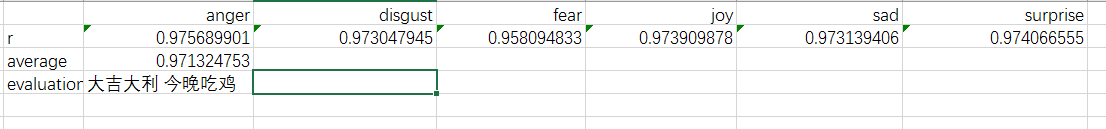
此处最高值为50.1608% k=11

实际测试中正确率上到过50.58%，但是因为此时是使用的stable\_sort(),且没有做距离排除优化，所以此时的值是因为训练集的顺序偶然上去的，并不是真正算法。所以此处排除。

1. KNN回归

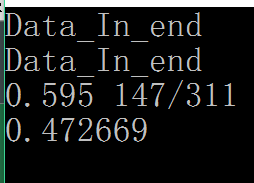
* One-Hot矩阵部分优化(加权和0距离判断)，欧式距离K（1,30）
* TF-IDF优化全部启用，修改加权方式。

和同学的test回归进行相关性检测：

结果表明没有出现过拟合。

1. NB分类

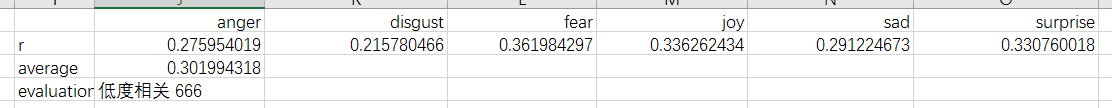
NB分类没有太多优化就是调节拉普拉斯α的值。一下为α在（0,2）对准确率的影响，主要就是一个凸函数，所以我们可以通过二分法直接定位一下最大值，此处定位的最大值为α=0.595验证集准确率为47.26%

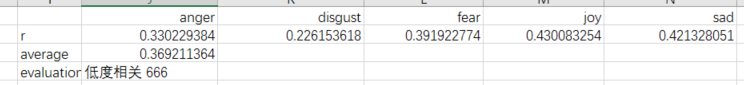


1. NB回归

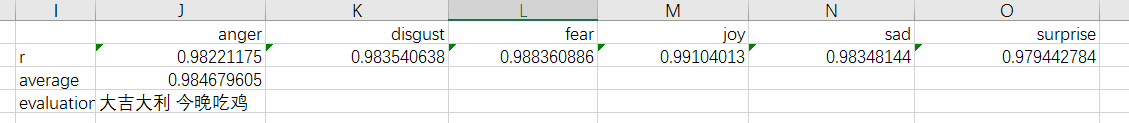
此处也和NB分类类似调整拉普拉斯平滑中的α的值，调整过程不在展示，此处相关性和α的值也和上面差不多是一个凸函数。

α=1时，

调整过程实在是太麻烦了…直接展示最优值，为α=0.0059



和同学的test回归进行相关性检测：

结果表明优化后没有出现过拟合。

# 思考题

**1. KNN回归把距离的倒数作为权值进行加权？**

首先距离越近说明他的和测试样本越接近，同时KNN回归的时候需要加权，因为如果K个最近邻在测试中所占的一致显然是不合理的，所以需要加权。加权方式此处选择倒数是为了使得距离近权值越大。 使得最终结果中比重根据距离进行配。

**2. 回归中同一测试样本的各个情感概率总和应该为1 如何处理？**

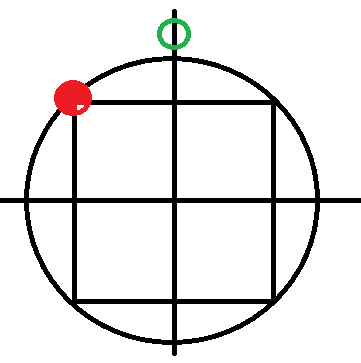
回归中个感情和为1，此处只需要把他们线性放大一下就好了，就是求和在分别除和。

实验中也考虑过非线性放大的方式，但是最终结果并没有提升，最终被这个做法被排除了。

1. **在矩阵稀疏程度不同的时候，欧式距离和曼哈顿距离表现有什么区别，为什么？**

矩阵越稀疏欧式距离和曼哈顿距离的计算出来的距离差距越大，换句话说就是，矩阵越稀疏的情况的曼哈顿距离和欧式距离一个根号的差距这个值越大，举个例子可能就是10000 和 100 之间这种差距。

这种差距的缺点源于特征向量的各分量的量纲不一致。

比如对身高（cm）和体重（kg）两个单位不同的指标使用欧式距离可能使结果失效。而曼哈顿不存在这种问题。如右图，出现了曼哈顿距离时绿色更近，而欧式距离更远。此时如果垂直方向为同一属性就反而出现误判。

在实验中也出现了种情况，使用曼哈顿TF-IDF能到44%+，而在使用欧式距离的时候只能到41%+。

同时还有效率问题O(n)和O(n^2)的差距。

1. **伯努利模型和多项式模型分别有什么优缺点？**

伯努利模型是一种以文档为基础的模型，他的计算最小单位为文本，不考虑单词重复的情况类似One-Hot矩阵，这种方式决定了他就有One-Hot矩阵的不足，主要问题就是对高频词的体现不足，如果文本词语数较多或者差异较大的情况效果不好，所以伯努利模型用来处理一些特征向量少的，重复单词少的短文档效果会比较好。同时其实现也比较简单，算法复杂度低。

多项式模型式一种以词为基础的模型，最小计算单位为词，考虑了词的重复情况，类似TF。好处就是能处理更长的文档和较多的特征向量。实际使用中伯努利模型更加准确。多项式模型和伯努利模型都是适合处理离散的特征向量对于连续，对于连续的多项式模型无法处理。

1. **如果测试集中出现了一个之前全词典中没有出现过的词该如何解决？**

个人的处理方式是抛弃掉这个词主要原因是，在最优解的情况下，我使用了TF-IDF矩阵，此时的IDF向量对于某个测试样本的新词是不存在的，所以这个新词的TF-IDF为0，也就是说没有存在的意义。在NB中没有的词如果没有拉普拉斯平滑那会导致乘上0使得算法效果下降，就算有了拉普拉斯，也会只乘上一个0平滑后的值，对最终判断基本没有影响。

KNN和NB都没有涉及到自主学习，所以我们并不需要因为某个训练中没有的新词去修改训练集的词典。

在实际测试中发现抛弃会提高效果，大概就1%左右的影响。

当然此处我也可选择保留通过修改一下即可选择保留。