**【数据预处理】**

1. 去除所有表情符号 -> 首先**弄清楚表情符号的编码方式**
2. 删除所有空格
3. 删除所有标点符号（除了**“+”**,待定）
4. 删除所有特殊符号
5. 转换成简体
6. 检测到全角的数字，直接屏蔽（必须在删除标点符号之后），提示玩家“消息不符合发言规则”

如“正规渠道50送１００万元宝+ⅥP+银币+１３６＋５０８＋３２２到付。ctBWKn“”

以上1~4点基于以下理由：

1. 有的垃圾信息会故意以空格、标点符号、特殊符号、表情等来隔断可能被检测到的关键垃圾词汇；
2. 由于手误等原因，会有空格、表情等无意义字符插入到正常的文本中，删除这些有利于提取特征；
3. 聊天信息的长度一般非常短，因此要尽量提取到尽可能多的特征再进行筛选，避免漏掉一些可能会对分类有影响的特征。所以删去以上符号，使文本连在一起后再进行特征提取会更有利。

**本项目可使用：**

（1）特征单元：Bigram与分词相结合，两个特征集进行合并。若两个特征集中有相同的特征，次数不累加。

（2）特征选取方法：

* 用文本频率DF >= 2进行截断
* Chi-square/信息增益IG/互信息MI等方法选取前k个最具代表性的特征

（3）特征向量权重：使用TF-IDF表示；权重向量要进行规格化

**【文献中的经验】->可考虑**

* 将所有的空格、标点符号、特殊字符都删除
* **增加一个特征：文本中的数字串个数 或 文本中是否有数字串**。 考虑到垃圾信息中可能会含有qq号、群号等等，这个特征有一定价值。
* 注意到很多垃圾消息中包含“+”号，是否可以**将“+”号的数量也作为一个特征**呢。如果这么做，那么数据预处理的时候就要保留“+”号了。

例子：“50元给50万元宝+1亿银+V12+橙装+橙将Q272-387-060先货iN”

“ちO二1OO万圆寳+VIP+银币+橙武将十扣844一682一555故事従适裏開ㄝ”

【进一步想法】

1. 能否加入用户信息作为特征？

如：

1.用户好友数量  
2.用户的在线数据：如每日、每周平均在线时长，每次登录平均在线时长，活跃时间段等  
3.用户等级  
4.是否为VIP/是否为氪金玩家  
5.用户发言频率：发言次数/一天

6.该用户是否被标记为黑名单

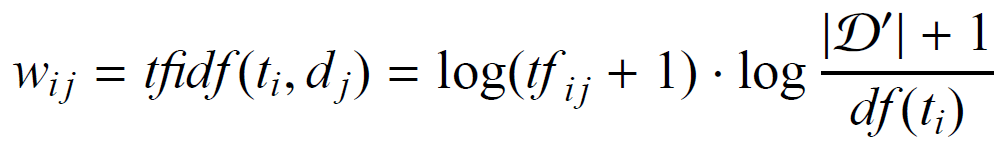
1. 能否增加用户反馈信息，由玩家反馈是否为垃圾信息，由此进行模型训练？

# 文本分类的步骤：

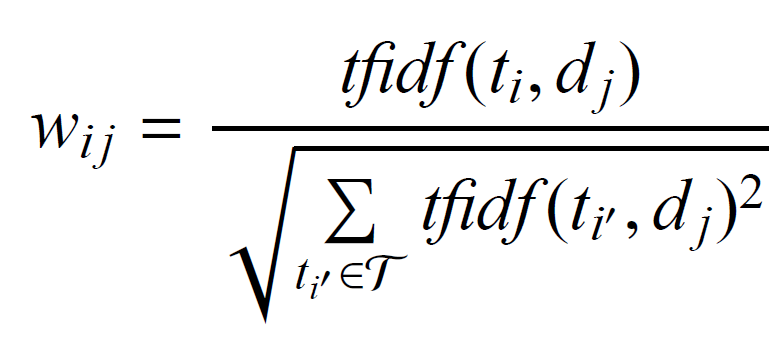
* + **建立数据集与预处理**
    - 收集文本，标注类别，去除非文本内容，编码转换，处理乱码
    - 处理**停用词（stopwords）**
    - 按照某种比例划分训练集和测试集
  + **文本向量化，即采用向量空间模型作为文本表示的数学模型**
    - **文本标引。即初步的特征提取：**
      * 划分**特征单元**，如**分词、bigram**等
      * **权重计算**，即计算特征在向量中的权重。为了消除文本长度的影响，通常还要进行向量规格化。

特征权重计算指为特征空间中的文本向量的每一维确定合适的数值，以表达对应特征在文本中的重要程度。

* + 布尔权重：某特征i在文本j中出现则为1；否则为0
  + 频度权重：以特征i在文本j中出现的次数表示权重
  + **TF-IDF权重：**



* + 权重向量的规格化——为了消除文本长度对特征的影响：



* + - **降维。降维通常利用训练集的信息来达成。**
      * 特征选择，即直接选出原始特征集的一个子集。特征选取方法见上。
        + 特征选择的必要性：

不相关特征的存在会使分类器对训练集过拟合，从而降低文本分类的性能

相关但是冗余的特征的存在也会影响文本分类的性能

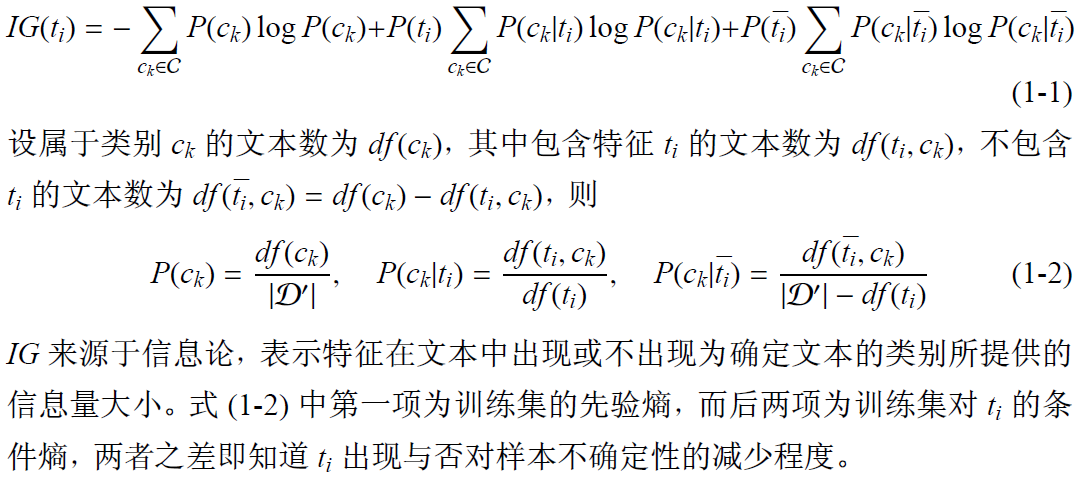
* + - * 特征抽取，一种方法是根据句法分析定义两个或多个具有特定句法关系的词为一个复合特征；另一种方法是通过聚类分析或统计分析发现相近的特征，合并为同一个特征。
  + **学习分类器——训练**
  + **测试与评价——测试模型**

**采用什么作为特征？**

1. **Bigram.** 在SVM上训练时，低维度分词效果更好，高维度bigram效果更好。
2. **使用分词系统进行分词**
3. **使用情感词典里的情感词作为特征**

**特征选取方法：**

1. **Chi-square：** 在标注了类别的文本数据上计算每个候选特征与类别之间的关系，选取每个类别最具有代表性的特征。见笔记本。
2. **信息增益（information gain, IG）：**特征ti在类别集C上的信息增益定义为：



1. 词频（Term frequency）
2. **文本频率**（Document Frequency, DF）：特征ti的文本频率即训练集中出现特征ti的文本数：它认为特征在越多的文本中出现，对分类的作用就越大。通过DF进行特征选择可以去除大量的低频特征，但是无法去除在类别间分布均匀的高频特征。实验表明，DF虽然简单，却相当有效；即使采用其他的特征选择指标，先使用DF截断来去掉一些少见特征通常也对分类性能有提高。**建议使用DF>=2来进行截断**。
3. **TF-IDF :** 某一特定文件内的高词语频率，以及该词语在整个文件集合中的低文件频率，可以产生出高权重的TF-IDF。因此，TF-IDF倾向于过滤掉常见的词语，保留重要的词语。
4. **互信息**（Mutual information）
5. **文本证据权**（the weight of evidence for text）
6. **几率比**（Odds ratio）

## Chi-square（卡方检验）

卡方检验最基本的思想就是通过观察**实际值**与**理论值**的偏差来确定理论的正确与否。

具体做的时候常常先假设两个变量确实是**独立的（“原假设”）**，然后观察实际值（观察值）与理论值（这个理论值是指“如果两者确实独立”的情况下应该有的值）的偏差程度，如果偏差足够小，我们就认为误差是很自然的样本误差，是测量手段不够精确导致或者偶然发生的，两者确确实实是独立的，此时就接受原假设；如果偏差大到一定程度，使得这样的误差不太可能是偶然产生或者测量不精确所致，我们就认为两者实际上是相关的，即否定原假设，而接受备择假设。

理论值为E，实际值为x，偏差程度的计算公式为：

这个式子就是卡方检验使用的差值衡量公式。当提供了数个样本的观察值x1，x2，……xi，……xn之后，代入到式中就可以求得开方值，用这个值与事先设定的阈值比较，如果大于阈值（即偏差很大），就认为原假设不成立，反之则认为原假设成立。

在文本分类的特征选择阶段，一般使用“词t与类别c不相关（即它们独立）”来做原假设，计算出的开方值越大，说明对原假设的偏离越大，我们越倾向于认为原假设的反面情况是正确的。选择的过程为每个词计算它与类别c的开方值，从大到小排个序（此时开方值越大越相关），取前k个就可以。

例如，N篇文档，分类有体育和非体育，考察特征词“篮球”与类别“体育”的相关性。

|  |  |  |  |
| --- | --- | --- | --- |
| 特征选择 | 属于“体育” | 不属于“体育” | 总计 |
| 包含“篮球” | A | B | A+B |
| 不包含“篮球” | C | D | C+D |
| 总数 | A+C | B+D | N |

根据原假设（体育类与“篮球”无关），“体育”类别中包含“篮球”的文档比例应与所有文档中包含“篮球”的文档比例相同。故Ａ（体育类中包含的“篮球”数）的理论值应为：

http://img.my.csdn.net/uploads/201211/07/1352272953_7306.gif

差值：

http://img.my.csdn.net/uploads/201211/07/1352273080_7298.gif

同样计算剩下三种情况的差值D12，D21，D22。最后计算“篮球”与“体育”类文章的卡方值：

http://img.my.csdn.net/uploads/201211/07/1352273293_1882.gif

进一步化简，注意如果给定了一个文档集合（例如我们的训练集）和一个类别，则N， A+C和B+D对同一类别文档中的所有词来说都是一样的，故实际计算的时候我们都使用

http://img.my.csdn.net/uploads/201211/07/1352273407_8936.gif

卡方检验的缺点是：它只统计文档是否出现词，而不管出现了几次。这会使得他对低频词有所偏袒（因为它夸大了低频词的作用）。

但是，根据大多文献的实验结果表明，**卡方检验是目前表现最好的特征选择指标之一**，甚至是最好的，除了在维数很低的情况下IG的表现更好一些。

## 信息增益

1. **信息量**

信息量的性质：

（1）越小概率的事件发生了，产生的信息量越大。因此一个具体事件的信息量应该是随着其发生概率而递减的，且不能为负；

（2）如果我们有俩个不相关的事件x和y，那么我们观察到的俩个事件同时发生时获得的信息应该等于观察到的事件各自发生时获得的信息之和，即：h(x,y) = h(x) + h(y)， 由于x，y是两个不相关的事件，那么满足p(x,y) = p(x)\*p(y). 根据上面推导，我们很容易看出h(x)一定与p(x)的对数有关

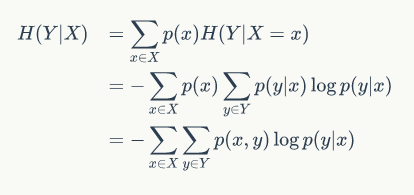
因此我们有信息量公式如下：

1. **信息熵**

信息量度量的是一个具体事件发生了所带来的信息，而信息熵则是在结果出来之前对可能产生的信息量的期望——考虑该随机变量的所有可能取值，即所有可能发生事件所带来的信息量的期望。即

1. **条件熵**

条件熵H（Y|X）表示在已知随机变量X的条件下随机变量Y的不确定性。



假设对分类系统来说，类别C是随机变量，它可能的取值是C1，C2，……，Cn，而每一个类别出现的概率是P(C1)，P(C2)，……，P(Cn)，那么此时分类系统的信息熵就是：

那么，当分类系统不包括某个特征t时，系统的条件熵是多少呢？答案是，将特征t固定住，使t不能变化，计算此时的系统条件熵。（可以这么想：一个班有很多学生，每个学生每天有两种可能情况，出勤或缺勤，因而整个班的出勤情况是非常多的；但是现在有一个学生，他被固定在座位上了/固定在家里，即他的情况一定是出勤/缺勤，那么，我们已经可以确定这个同学的出勤与否了，这个同学的出勤情况就不再影响这个班的出勤情况了。也就是说，固定这个学生的状态，就是在不包括这个学生的情况下，得到班级的信息熵）

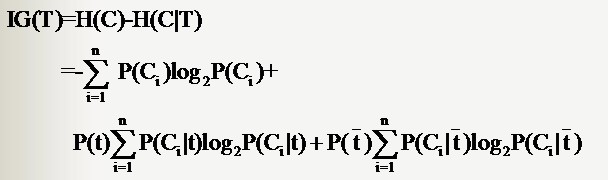
而t可以被固定为两个值：出现，或不出现；若我们用T表示该特征出不出现的变量，t表示出现，那么固定特征t的条件熵就为：

该条件熵可以理解为：当分类系统不包括该特征t时，系统的信息熵。

1. **信息增益**

**信息增益 = 信息熵 – 条件熵**

意思为：特征t给系统带来的的信息增益 = 整个系统的信息熵（包括特征t）- 不包括特征t的系统信息熵（固定t后的条件熵）



* 信息增益最大的问题在于它只能考察特征对整个系统的贡献，而不能具体到某个类别上，这就使得它只适合用来做所谓“全局”的特征选择（指所有的类都使用相同的特征集合），而无法做“本地”的特征选择（每个类别有自己的特征集合，因为有的词，对这个类别很有区分度，对另一个类别则无足轻重）。 不过对于本项目无影响，因为我们只做二分类。

**【互信息】：**

互信息和信息增益的数值是相等的。

**互信息**：两个随机变量的地位是相同的；

**信息增益**：把一个变量看成是减小另一个变量不确定度的手段

当我们尝试使用统计机器学习方法解决文本的有关问题时，第一个需要的解决的问题是，如何在计算机中表示出一个文本样本。一种经典而且被广泛运用的文本表示方法，即向量空间模型(VSM)，俗称“词袋模型”。

向量空间模型需要一个“字典”：文本的样本集中的特征词集合，这个字典可以在样本集中产生，也可以从外部导入。

有了字典后便可以表示出某个文本。先定义一个与字典长度相同的向量，向量中的每个位置对应字典中的相应位置的单词。然后遍历这个文本，对应文本中出现的某个单词，在向量中的对应位置，填入该特征词的**权重**。

文本分类中的特征集有两个特点：

1. 特征集很大，但是每个文本向量中大多数特征的值是零，即向量的高度稀疏；
2. 特征集冗余较大，同义和近义的特征非常多，这是由自然语言的特点决定的。

**如何抽取文章特征？**

前面这个回答着重是讲如何从海量候选中选取有代表性的特征。而特征也有很多不同类型，如单词、短语、隐含主题、概念，等等。文档的特征表示，是文本聚类和分类的共同基础。目前的主要尝试和结论如下：

1. 如开头所提到的问题中的回答，最常见的特征类型是词汇/词语，**利用Chi-squre进行特征选取，利用TFIDF作为特征权重**。这也是搜索引擎和文本分类中最通用鲁棒的做法。

2. 对于短文本，如微博、查询词、社区问答系统中的提问问题等，利用基于词汇的特征向量空间计算相似度会面临比较严重的数据稀疏性问题，解决的方案是利用隐含主题模型等方法，建立词汇之间的相似度。具体方案可以参考2008年发表在WWW上的一篇论文：Learning to classify short and sparse text & web with hidden topics from large-scale data collections。需要注意的是，在LDA的原始论文中作者就尝试仅用文档主题来作为特征进行分类，但经验表明，仅用隐含主题的特征表示效果明显弱于使用词汇的特征表示方案。实用的做法是将两者相结合。

**对于中文文本分类，基本有两个比较成熟的结论：**

（1）当特征数达到近1万维的时候，采用分词后的词作为特征还是采用字的bigram作为特征，效果相近。因此如果手头有比较靠谱的分词系统，就用分词作为特征，如果没有，用bigram也能实现不坏的效果。（2）特征选取的最好方法是Chi-Square，其基本思想是在标注了类别的文本数据上计算每个候选特征与类别之间的关系，选取每个类别最具代表性的特征。

* 对于大多数的中文文本分类任务，分词不是必要的，二字串特征是更简单而有效的选择。

svm是否使用了核函数，怎么选择的？

核函数是在数据线性不可分的情况下使用的，当样本的数量远大于特征的数量的时候就要考虑核函数了，但是当样本数量远大于特征数量的时候应该考虑线性核，因为非线性核的计算量太大了。

当数据量足够庞大时，feature足够多时，所有的分类算法最终的效果都差不多。

当训练集不大，feature比较多的时候，用线性的核。因为多feature的情况下就已经可以给线性的核提供不错的variance去fit训练集。

当训练集相对可观，而feature比较少，用非线性的核。因为需要算法提供更多的variance去fit训练集。

feature少，训练集非常大，用线性的核。因为非线性的核需要的计算量太大了。而庞大的训练集，本身就可以给非线性的核提供很好的分类效果。

Linux下machine learning的Python开发环境：

* OS：Ubuntu 16.04
* Python版本：Python 2.7.12
* 步骤：
  + 1.在pycharm官网下载免费的community版本的pycharm压缩包，解压后，在bin目录下运行./pycharm.sh进行安装，再将其bin目录添加到系统的PATH中去。
  + 2.利用sudo apt-get install 命令下载安装以下package：
    - python-pip
    - python-dev
    - python-numpy
    - python-numpy-dev
    - python-setuptools
    - python-scipy
    - python-matplotlib
    - libatlas-dev
    - g++
    - cython
    - python-sklearn
  + 这时发现import sklearn失败，原因是安装的numpy版本为9，版本太低，故需要升级。但是用pip（用pycharm进行下载实质上也是用pip下载）和apt-get的方法都升级失败了。而且用apt-get remove也无法卸载numpy。这时使用命令 sudo pip uninstall numpy成功卸载了低版本的numpy，再到GitHub上下载了numpy的源码，进入目录后使用命令 sudo python setup.py install 进行安装。安装完成后，到pycharm的package列表里查看，发现已经更新到最新版本了。现在，import sklearn就没有问题了。