文本分类模型

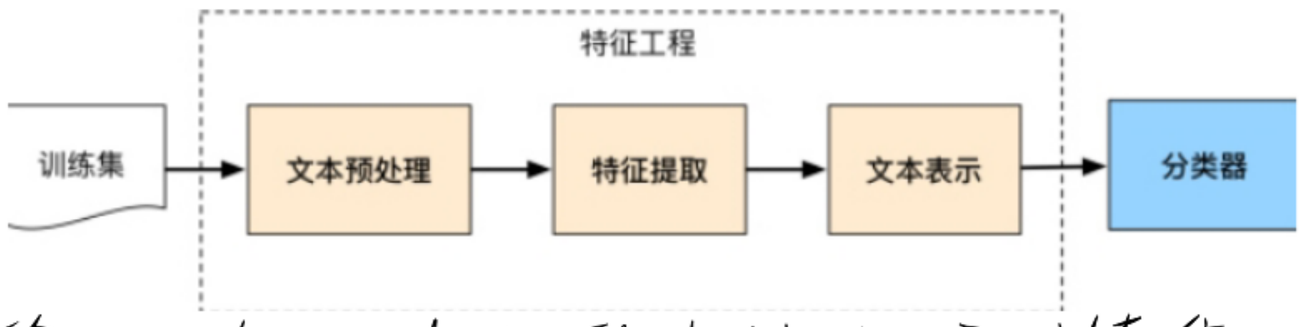
输入文章d，指定类别集合 $C=\{c_1, c_2, \cdots, c_j\}$，输出预测的类别 $c \in C$

基于规则的模型：由词和其它特征构成的规则来鉴别文章的类别

优点是只要规则合适精确率就高

缺点是构建与维护规则成本高

监督分类方法：



输入m个标注好的文档构成训练集 $(d_1, c_1) \cdots (d_m, c_m)$，最后分类器会根据输入的d映射到对应的c上

分类器有几种类别：
- Naive 贝叶斯分类
- Logistic回归
- SVM
- K-近邻

Naive贝叶斯分类

这是机器学习中最基础与易实现的分类算法，它基于一个假设：文本中所有词(特征)之间完全独立。

词袋模型：将整篇文章看作一个装单词的袋子，忽略词的顺序语法，上下文，只统计每个词在文章中出现的频次，将文本转化为一个词频向量作为分类器的输入特征

特征向量：用一串数字来表示某个东西，就称为特征向量

如一个影评"我喜欢这部电影"，而在统计词频中

"我"：6次 "喜欢"：3次 "这部"：2次 "电影"：9次
那么该影评的特征向量就是 [6, 3, 2, 9]

训练集中所有词汇会构成一个词汇表 $V=[w_1, \cdots, w_n]$
其中，$w_1 \cdots w_n$ 顺序是固定的（特征向量有序性）
词汇表的长度 n 被称为维度。对于一个输入 d，首先要做清洗（去除标点，统一大小写等），然后按词汇表将 d 变成一个长度为 n，顺序与 V 一致的特征向量 $X=[x_1, \cdots, x_n]$，每个维度 x_i 对应词汇表中第 i 个词的词频

文本分类流程：以电影好/差评为例

① 收集标注好/差评的电影影评

② 将所有影评转化为词频向量，构建词汇表

③ 模型训练：统计先验概率 $P(\text{好评})$ $P(\text{差评})$ 即好评与差评分别在训练集中的占比，统计条件概率 $P(w_i|\text{好评})$ $P(w_i|\text{差评})$ 即每个词在好评与差评中出现的概率

④ 分类预测：输入新影评 d，计算 $P(\text{好评}|d)$ 与 $P(\text{差评}|d)$，输出概率更大的类别

数学语言：对于文档 d 和类别 c

$$C_{MAP} = \arg\max_{c \in C} P(c|d)$$

MAP：最大后验概率 即概率最大的 c

$$= \arg\max_{c \in C} \frac{P(d|c)P(c)}{P(d)}$$

贝叶斯准则

$$= \arg\max_{c \in C} P(d|c)P(c)$$

$$= \arg\max_{c \in C} P(x_1, x_2, \cdots, x_n|c)P(c)$$

d 被表示为特征向量 $x_1, \cdots, x_n$

$$= \underset{c \in C}{\operatorname{argmax}} \prod_{x \in X} P(x_i \mid c)$$

词袋假设：特征与位置无关，各特征条件概率独立。

多元朴素贝叶斯模型：将文档中每个词出现的位置都当作一个独立特征（不论词是否重复）

$$C_{NB} = \underset{c_j \in C}{\operatorname{argmax}} P(c_j) \prod_{i \in positions} P(x_i \mid c_j)$$

解释：文档中每个出现的词就是一个特征，有多少词就有多少特征，如[我 爱 这部 电影] 我→x_1 爱→x_2 …

公式中的positions就是文档中每个词的位置，位置i对应特征i。对于重复的词，不计较其重复。

$P(c_j)$：先验概率，训练集中类别c_j的占比，如好评60%，那么P(好评)=60%，这是为了给分类加一个倾向，就算文章偏中性，也会倾向于判断为好评

$\prod_{i \in positions} P(x_i \mid c_j)$：给定类别$c_j$，这篇影评中每个词在训练集里$c_j$类别文档里出现的概率全部乘起来。

如"我爱这部电影"偏好评的概率为：

P(我|好评) P(爱|好评) P(这部|好评) P(电影|好评)

对数优化：$C_{NB} = \underset{c_j \in C}{\operatorname{argmax}} \left[\log P(c_j) + \sum_{i \in positions} \log P(x_i \mid c_j) \right]$

模型学习方法：

最初想法：最大似然估计

$$\hat{P}(c_j) = \frac{doccount(C = c_j)}{Ndoc}$$

类别为c_j的文档数 / 文档总数

$$\hat{P}(w_i|c_j) = \frac{count(w_i, c_j)}{\sum_{w \in V} count(w, c_j)}$$

w_i在类别c_j里出现次数 / 类别c_j里所有词出现总次数

即标注为c_j的文档中的词w_i的频次在该文档里所有词项($w \in V$)中所占比例，即在类别c_j文档的总词数里，w_i出现频次的占比

问题：未在训练集中出现的词会导致$c_{MAP} = \arg\max P(c_j) \prod P(x_i|c_j)$直接为0

解决：Laplace平滑

$$P(w_i|c) = \frac{count(w_i, c) + 1}{\sum_{w \in V} count(w, c) + |V|}$$

Naive贝叶斯分类与语言模型

（词袋模型）

如果Naive贝叶斯分类仅使用词项特征，且使用的是文档中的所有词项而不是其子集，那么它与语言模型基本相同。一个类别就是一个语言模型。

对于类别c：每个词出现概率：$P(word|c)$

每个句子出现概率：$P(s|c) = \prod P(word|c)$

换句话说，假设有n个语言模型，每个模型生成一种类型的文档，那么Naive贝叶斯分类就是在求一个文档来自哪个模型的概率最大。

文本分类评估

混淆矩阵：

	标签正向	标签负向
分类结果正向	tp	fp
分类结果负向	fn	tn

精确率：对于正向类别，分类结果中正确的比例

$$P = \frac{tp}{tp + fp}$$

召回率：对于正向类别，正向标签中被正确分类的比率

$$R = \frac{TP}{TP + FN}$$

混合度量下：P/R折中（加权调和平均）

$$F = \frac{1}{\alpha\frac{1}{P} + (1-\alpha)\frac{1}{R}} = \frac{(\beta^2+1)PR}{\beta^2 P + R}$$

通常使用F_1度量，即 $\beta = 1$（$\alpha = \frac{1}{2}$）

$$F_1 = \frac{2PR}{P+R}$$

明元分类评价：对于类别对 $<C_i, C_j>$，有多少C_i文档被错误分类为C_j

真值 \ 预测	C_1	C_2	…	C_i	…	C_n
C_1	C_{11}	C_{12}	…	C_{1i}	…	C_{1n}
C_2	C_{21}	C	…	C_{2i}	…	C_{2n}
⋮	⋮	⋮	⋱	⋮	⋰	⋮
C_i	C_{i1}	C_{i2}	…	C_{ii}	…	C_{in}
⋮	⋮	⋮	⋰	⋮	⋱	⋮
C_n	C_{n1}	C_{n2}	…	C_{ni}	…	C_{nn}

C_{ij}：有多少个真值为C_i的文档被识别为C_j

重点在C_{ii}上（即对角线）

召回率：类别为C_i的文档中被正确分类的比例

$$R = \frac{C_{ii}}{\sum_j C_{ij}}$$

精确率：分类为C_i的文档中被正确分类的比例

$$P = \frac{C_{ii}}{\sum_j C_{ji}}$$

准确率：文档被正确分类的比例

$$\frac{\sum_i C_{ii}}{\sum_i \sum_j C_{ij}}$$

对角线 / 总数