

文档搜索无法达到绝对准确！

- 语言本身的局限性
- 关键词表达能力有限（查询语句所表达的用户需求不清晰、歧义）
- 计算机的语义理解能力不够

如何衡量搜索的准确性

- 结果集中的相关文档越多越好（Precision）
- 尽量多的相关文档囊括到结果集中（Recall）

自由检索

布尔检索的局限

- 运算难以被大众接受
- 太死板

自由检索

- 查询语句为一组词汇
- 结果按照相关性顺序返回
- 匹配程度可一定程度放松

词的预处理

1. A document → A bag of terms

2. 减少文档中的噪音、排除歧义

- 词干提取（Stemming）

Example: **automation** → **automate(s)** \ **automatic** \ **automation**

效果：增加 Recall

相关规则：sses→ss \ ies→i \ ational→ate \ tional→tion

- 删除停止词（Stop Words）

定义：出现频繁，但是对语义没有帮助的词

Example: a,an,the \ of,for,by \ it,they,them...

效果：提高 Precision

注意点：有些停止词有重要意义,不能直接去处 (let it be \ to be or not to be)

3. 不同语言需要些特殊处理

- 中文需要分词
- 德语有些词汇需要拆分

4. 标准化问题

Example: U.S.A , USA → USA / colour , color → color

补充内容

From *Data Science For Business*

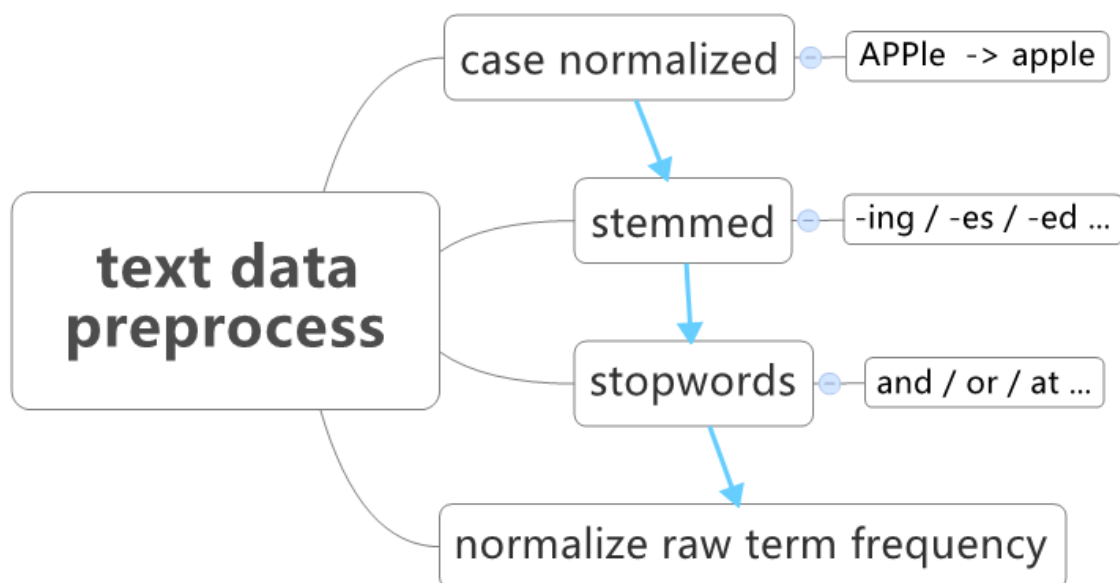
文本数据 (Text data) 的特点

- Unstructured data (非结构化数据)
- Linguistic structure (语言结构) ——NLP (自然语言处理)

文本数据的缺陷 (Text data's problem —dirty data) :

- 不按照语法规则书写, 例如很多用户留言或评论 (Ungrammatical)
- 拼写错误 (misspell)
- 各种缩写简写 (Abbreviate)
- 随意使用标点符号 (Punctuate randomly)
- 各种同义词的使用 (Synonyms)
- 同形异词 (Homograph)
- 同一词汇在不同领域的不同理解 (Different domain)
- 很多词汇的理解需要基于上下文 (Context)

Dirty data 需要经过预处理 (**preprocess**) 才能作为下一步模型的输入 (**input**)



搜索结果排序

Process:

1. 得到所有可能相关的文档
2. 对文档相关度进行评估
3. 按相关度大小返回文档

模型

Bag of Words Model

查询语句 (Q), 文档集 (D) \rightarrow (0,1)

Term Frequency (TF)

一个词在一篇文档中出现的频率

$$TF(t, d) = \frac{\text{文档 } d \text{ 中词汇 } t \text{ 的数量}}{\text{文档 } d \text{ 中词汇总量}}$$

Term frequency 有两点需要注意:

- 太过稀有的词汇往往不加入考虑范畴。对于聚类分析, 低频的词汇往往没有太大作用, 所以在预处理时, 我们会设置一个最低频率限制。(当然, 在有的应用场景下, 这些低频词汇会显得十分重要, 就和我们有时候需要关注outlier一样)
- 过于平凡的词汇也不用太过关注。在聚类分析中, 这种平凡词汇并不能提供区分点, 所以我们常常会设置一个最高频率限制

Document Frequency

给定一个词, 包含该词的文档的总数

Inverse Document Frequency (IDF)

若某些词汇集中出现在少数几个文档中, 那么这些词汇对这几个文档的重要性不言而喻。

$$IDF(t) = \log\left(\frac{\text{文档总数}}{\text{包含 } t \text{ 的文档数} + 1}\right)$$

TFIDF

$$TFIDF(t, d) = TF(t, d)IDF(t)$$

文档向量化

依据TFIDF我们便可以将文档转变为特征向量

随后我们往往需要进行特征选择（可以使用频率的上下阈值，或者Information Gain），之后我们便能够将特征向量用于各种模型之中。

例如：Cosine Similarity