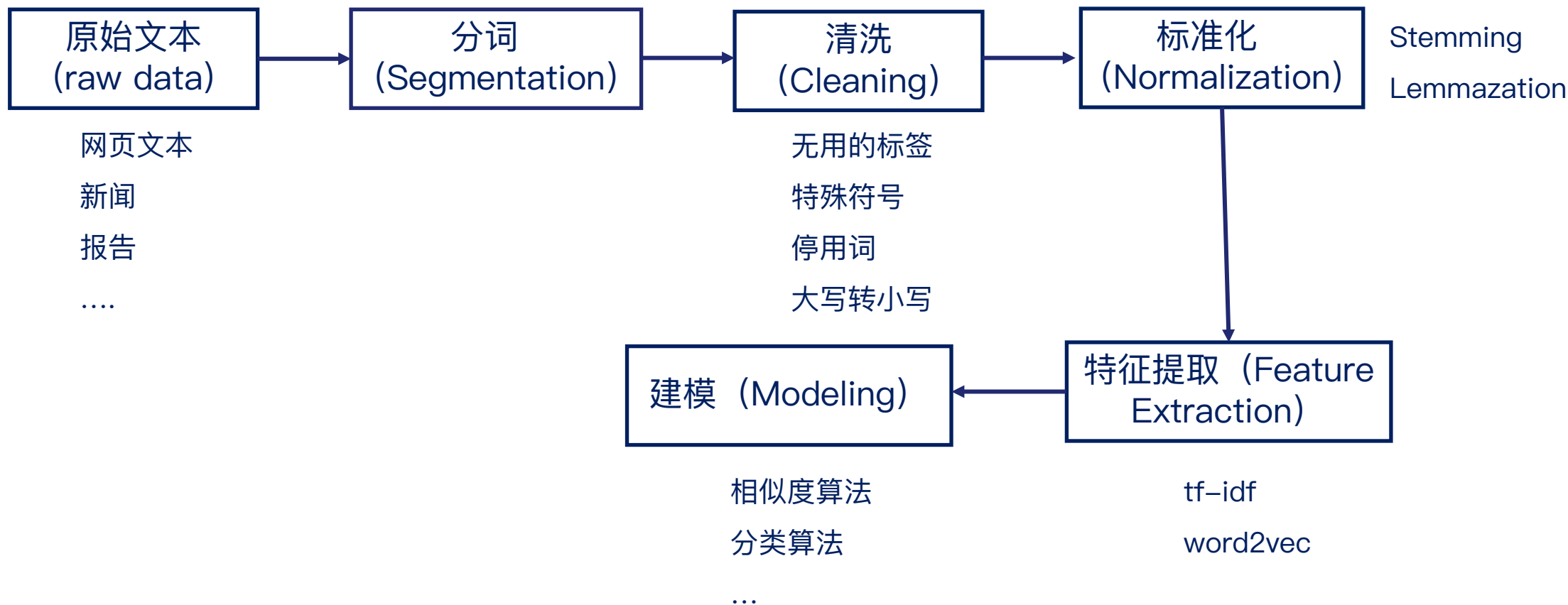


## 自然语言处理训练营 (3)

点此获取更多资源

# Pipeline



# What we cover today

- Word Segmentation
- Spell Correction
- Stop Words Removal
- Stemming

# Word Segmentation (分词)

# Word Segmentation Tools

Jieba分词 <https://github.com/fxsjy/jieba>

SnowNLP <https://github.com/isnowfy/snownlp>

LTP <http://www.ltp-cloud.com/>

HanNLP <https://github.com/hankcs/HanLP/>

.....

# Segmentation Method 1: Max Matching(最大匹配)

前向最大匹配 (forward-max matching)

例子：我们经常有意见分歧

词典：[“我们”，“经常”，“有”，“有意见”，“意见”，“分歧”]

# Segmentation Method 1: Max Matching(最大匹配)

后向最大匹配 (backward-max matching)

例子：我们经常有意见分歧

词典：[“我们”，“经常”，“有”，“有意见”，“意见”，“分歧”]

# Segmentation Method 1: Max Matching(最大匹配)

## 最大匹配的缺点？

例子：我们经常有意见分歧

词典：[“我们”，“经常”，“有”，“有意见”，“意见”，“分歧”]



# Segmentation Method 2: Incorporate Semantic (考虑语义)

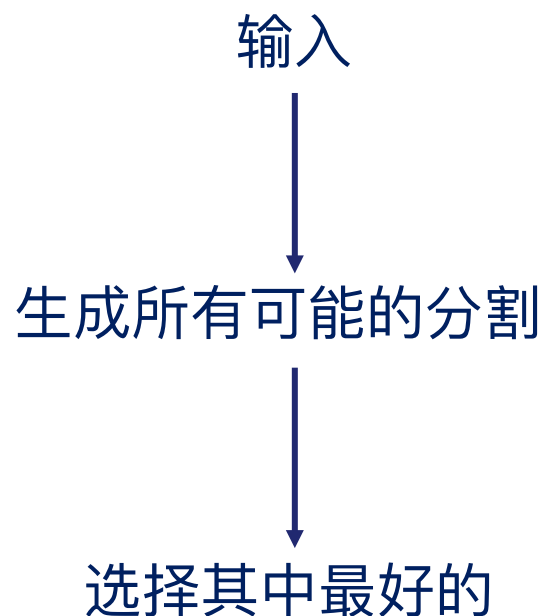
例子：经常有意见分歧

词典：[“有”，“有意见”，“意见”，“分歧”，“见”，“意”]

# Segmentation Method 2: Incorporate Semantic (考虑语义)

例子：经常有意见分歧

词典：[“有”，“有意见”，“意见”，“分歧”，“见”，“意”]



# Segmentation Method 2: Incorporate Semantic (考虑语义)

怎么解决效率问题?

Ask him! —————→



例子：“经常有意见分歧”

词典：[“有”，“有意见”，“意见”，“分歧”，“见”，“意”]

解决方案：

# Word Segmentation Summary

## 知识点总结:

- 基于匹配规则的方法
- 基于概率统计方法 (LM, HMM, CRF..)
- 分词可以认为是已经解决的问题

## 需要掌握什么?

- 可以自行实现基于最大匹配和Unigram LM的方法

# Spell Correction (拼写错误纠正)

# Spell Correction (拼写错误纠正)

用户输入 (input)

用户输入(correction)

天起



天气

theris



theirs

机器学系



机器学习

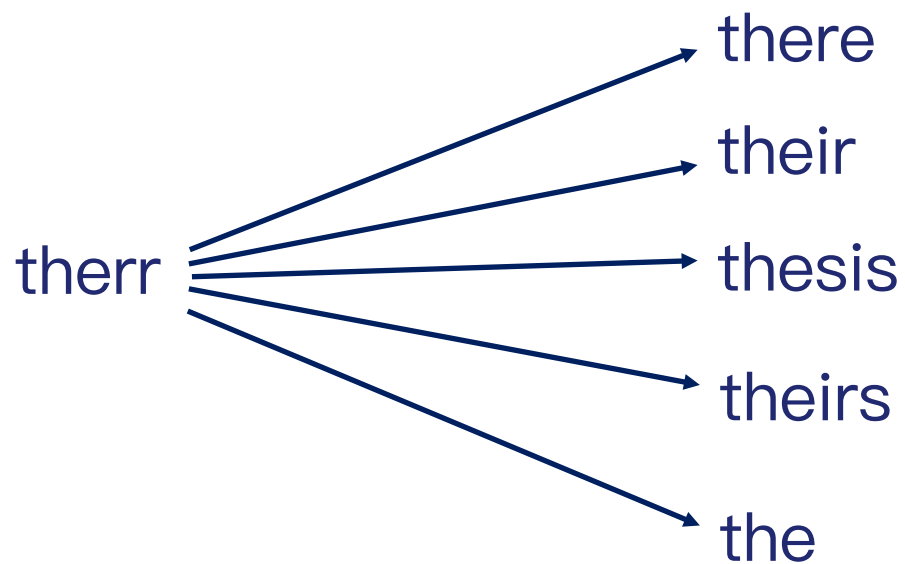
怎么做?

# Find the words with smallest edit distance

用户输入 (input)

候选 (candidates)

编辑距离 (edit distance)



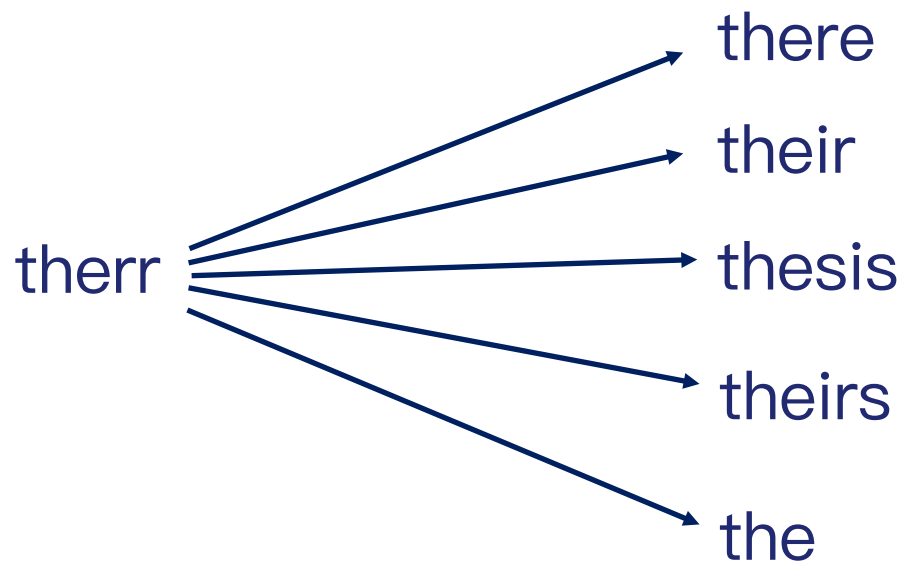
词典



# Find the words with smallest edit distance

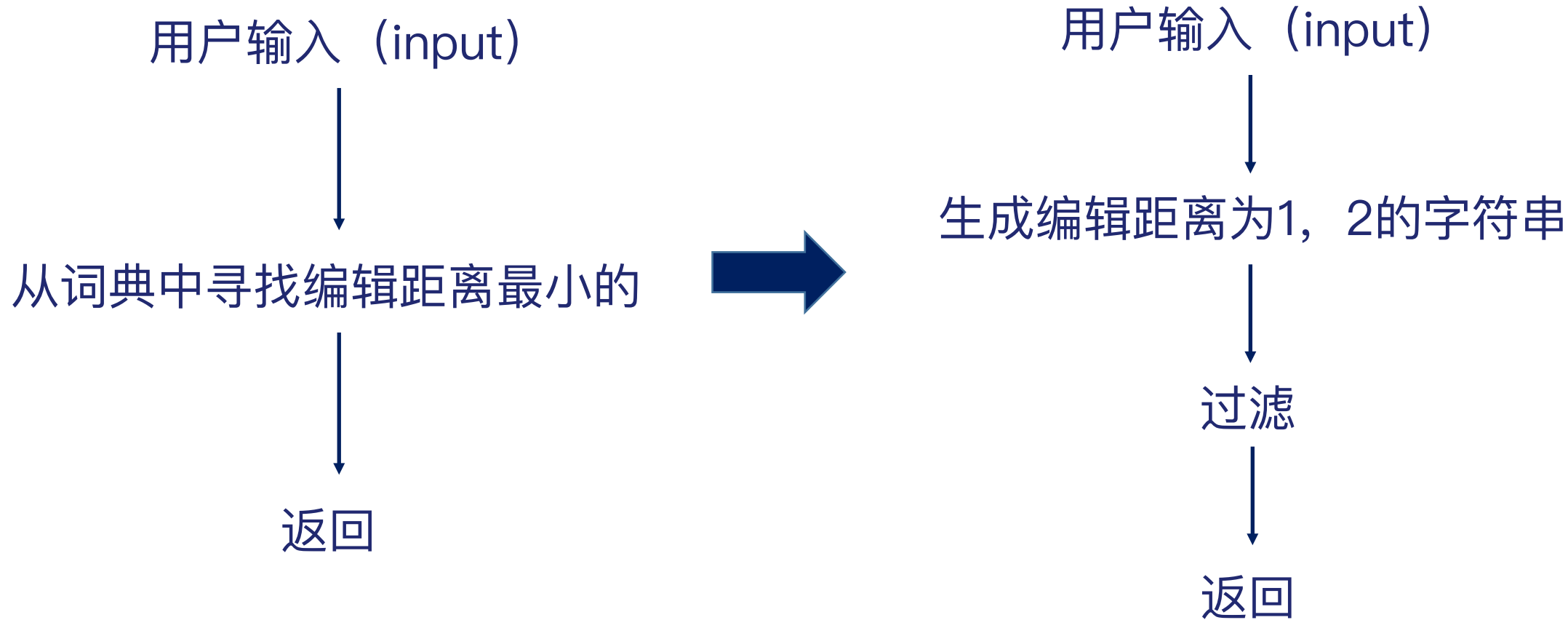
用户输入 (input)

候选 (candidates)

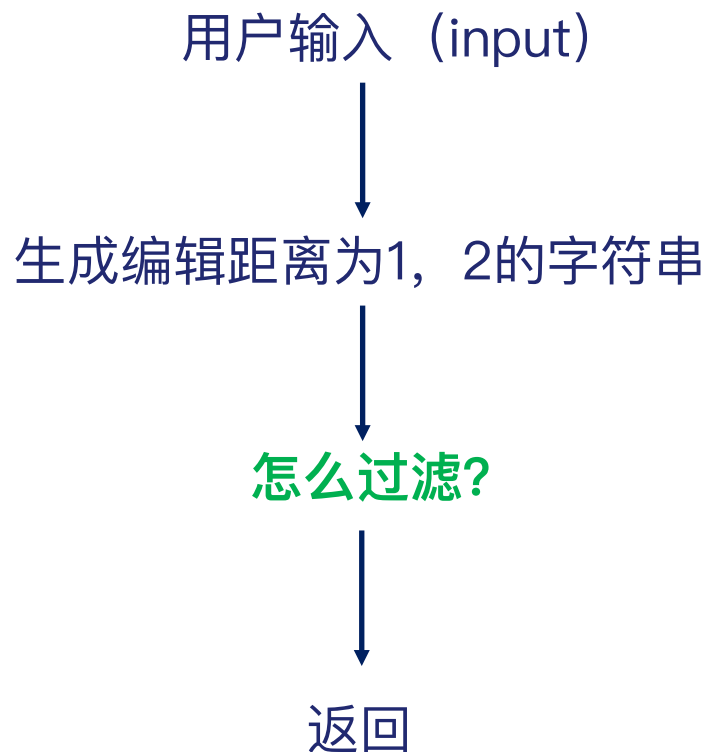


词典

# Better Way



# How to Select?



**问题定义:** 给定一个字符串  $s$ , 我们要找出最有可能修改后的字符串  $c$ 。  
并且  $c \in [w_1, w_2, \dots, w_N]$ 。也就是  $\hat{c} = \operatorname{argmax}_{c \in \text{candidates}} p(c|s)$

$$\begin{aligned}\hat{c} &= \operatorname{argmax}_{c \in \text{candidates}} p(c|s) \\ &= \operatorname{argmax}_{c \in \text{candidates}} p(s|c) * p(c)/p(s) \\ &= \operatorname{argmax}_{c \in \text{candidates}} p(s|c) * p(c)\end{aligned}$$

# Words Filtering

# Filtering Words

对于NLP的应用，我们通常先把停用词、出现频率很低的词汇过滤掉

这其实类似于特征筛选的过程

# Removing Stop Words

在英文里，比如“the”，“an”，“their”这些都可以作为停用词来处理。但是，也需要考虑自己的应用场景

# Low Frequency Words

出现频率**特别低的**词汇对分析作用不大，所以一般也会去掉。把停用词、出现频率低的词过滤之后，可以得到一个我们的词典库。

# Words Normalization



# Stemming: one way to normalize

went, go, going

fly, flies,

deny, denied, denying

fast, faster, fastest

意思都类似，怎么合并？

<https://tartarus.org/martin/PorterStemmer/java.txt>

# Porter Stemmer

## Step 1a

sses	→ ss	caresses	→ caress
ies	→ i	ponies	→ poni
ss	→ ss	caress	→ caress
s	→ ∅	cats	→ cat

## Step 1b

(*v*)ing	→ ∅	walking	→ walk
		sing	→ sing
(*v*)ed	→ ∅	plastered	→ plaster
...			

## Step 2 (for long stems)

ational	→ ate	relational	→ relate
izer	→ ize	digitizer	→ digitize
ator	→ ate	operator	→ operate
...			

## Step 3 (for longer stems)

al	→ ∅	revival	→ reviv
able	→ ∅	adjustable	→ adjust
ate	→ ∅	activate	→ activ
...			

# 单词的表示 (representation)

- 最常用的表示方式： **词袋模型** (Bag-of-word Model)
- 假设我们的词典里有7个单词：[我们，去，爬山，今天，你们，昨天，运动]
- **每个单词的表示：**
  - 我们： [1, 0, 0, 0, 0, 0, 0]
  - 爬山： [0, 0, 1, 0, 0, 0, 0]
  - 运动： [0, 0, 0, 0, 0, 0, 1]
  - 昨天： [0, 0, 0, 0, 0, 1, 0]

# 句子的表示 (representation)

最常用的表示方式： **词袋模型** (Bag-of-word Model)

假设我们的词典里有7个单词：[我们，去，爬山，今天，你们，昨天，运动]

我们今天去爬山： [1, 1, 1, 1, 0, 0, 0]

你们昨天运动： [0, 0, 0, 0, 1, 1, 1]

你们去爬山： [0, 1, 1, 0, 1, 0, 0]

# 怎么表示句子之间的相关性？

计算距离第一种方法：  $d = |s1-s2|$

例子： 给定“我们今天去爬山”和“你们昨天运动”：

$s1 = [1, 1, 1, 1, 0, 0, 0]$ ,  $s2 = [0, 0, 0, 0, 1, 1, 1]$

$d = \text{sqrt}(1+1+1+1+1+1+1) = \text{sqrt}(7)$

例子： 给定“我们今天去爬山”和“你们去爬山”：

$s1 = [1, 1, 1, 1, 0, 0, 0]$ ,  $s2 = [0, 1, 1, 0, 1, 0, 0]$

$d = \text{sqrt}(1+0+0+1+1+0+0) = \text{sqrt}(3)$

所以，“我们今天去爬山”和“你们去爬山”更接近

# 怎么表示句子之间的相关性？

计算距离第一种方法：  $d = |s1-s2|$

这种距离计算方法有什么缺点？

# 怎么表示句子之间的相关性？

计算距离第二种方法：  $d = (s1*s2) / |s1|*|s2|$  （余弦相似度）

例子： 给定“我们今天去爬山”和“你们昨天运动”：

$s1 = [1, 1, 1, 1, 0, 0, 0]$ ,  $s2 = [0, 0, 0, 0, 1, 1, 1]$

$d = 0 / \sqrt{2} * \sqrt{3} = 0$

例子： 给定“我们今天去爬山”和“你们去爬山”：

$s1 = [1, 1, 1, 1, 0, 0, 0]$ ,  $s2 = [0, 1, 1, 0, 1, 0, 0]$

$d = 2 / \sqrt{2} * \sqrt{3} = 2 / \sqrt{6}$

所以，“我们今天去爬山”和“你们去爬山”更接近

# 回顾句子的表示方式

```
corpus = [  
    'He is going from Beijing to Shanghai.',  
    'He denied my request, but he actually lied.',  
    'Mike lost the phone, and phone was in the car.',  
]
```

[	0	0	1	0	0	0	1	1	1	0	1	0	0	0	0	0	1	0	1	0]	
[	1	0	0	1	0	1	0	0	2	0	0	1	0	0	1	0	1	0	0	0]	
[	0	1	0	0	1	0	0	0	0	1	0	0	1	1	0	2	0	0	2	0	1]]

denied      he




# 回顾句子的表示方式

[	[	0	0	1	0	0	0	1	1	1	0	1	0	0	0	0	0	1	0	1	0]	
	[	1	0	0	1	0	<b>1</b>	0	0	<b>2</b>	0	0	1	0	0	1	0	1	0	0	0]	
	[	0	1	0	0	1	0	0	0	0	1	0	0	1	1	0	2	0	0	2	0	1]]
	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

denied      he

并不是出现的越多就越重要！  
并不是出现的越少就越不重要！

# tf-idf表达方法

$$\text{tf-idf}(w) = \text{tf}(d, w) * \text{idf}(w)$$


词频

$\log \frac{N}{N(w)}$

N: 语料库中文档总数

N(w): 词语w出现在多少个文档?

# 例子