**NLP**

1. **语言处理与python**

**1.1 语言计算:文本和单词**

搜索文本

.concordance（） 显示一行（固定字符数，搜寻字符位于中心位置）上下文，不区分大小写但区分不同时态

.similar（） 寻找与括号内的词出现在相似的上下文的词，即意思用法相近的词

.common\_contexts（） 显示两个或两个以上的词共同的上下文

文字和标点符号是彼此独立的

用离散图表示位置信息：

>>> text4.dispersion\_plot(["citizens", "democracy", "freedom", "duties", "America"])

len（） 获取文章标示符数量（包括标点符号）

set (text3) 获取text3的词汇表，包括标点符号

text.count() 计某个词在文本中出现的次数

sorted() 按首字母顺序进行排列,包括标点符号，大写字母在小写字母之前，标点符号在字母之前

**1.2 近观 Python:将文本当做词链表**

索引

text[num] 显示对应位置的字符

text.index() 显示对应字符在文中的位置

切片： list[m:n] m:n 表示元素 m...n-1。如果切片从链表第一个元素开始，我们可以省略第一个数字;如果切片到链表最后一个元素处结尾，我们可以省略第二个数字

字符串

把词用链表连接起来组成单个字符串，或者把字符串分割成一个链表：

>>> ' '.join(['Monty', 'Python'])

'Monty Python'

>>> 'Monty Python'.split()

['Monty', 'Python']

**1.3 计算语言:简单的统计**

Frequency Distributions:

>>> fdist1 = FreqDist(text1)

>>> print(fdist1)

<FreqDist with 19317 samples and 260819 outcomes>

>>> fdist1.most\_common(50) /显示出现频率最高的50个字符及其数目

>>> fdist1['whale'] /显示某个字符的数目

fdist1.hapaxes() 查看只出现过一次的标识符

细粒度的选择词： 下例为筛选出text1词汇表中长度大于15的标识符并排列打印

>>> V = set(text1)

>>> long\_words = [w for w in V if len(w) > 15]

>>> sorted(long\_words)

下例为筛选出长度大于7并且出现次数也大于7的标识符并排列打印

>>> fdist5 = FreqDist(text5)

>>> sorted([w for w in set(text5) if len(w) > 7 and fdist5[w] > 7])

提取文本中的词汇即双连词

>>> list(bigrams(['more','is','said','than','done']))

text\*.collocations() 寻找文本中的双连词

**NLTK频率分布类中定义的函数**例子 描述

fdist = FreqDist(samples) 创建包含给定样本的频率分布

fdist[sample] += 1 increment the count for this sample

fdist['monstrous' ] 计数给定样本出现的次数

fdist.freq('monstrous ' ) 给定样本的频率

fdist.N() 样本总数

fdist.most\_common(n) the n most common samples and their frequencies

for sample in fdist: 以频率递减的顺序遍历样本

fdist.max() 数值最大的样本

fdist.tabulate() 绘制频率分布表

fdist.plot() 绘制频率分布图

fdist.plot(cumulative=True) 绘制累积频率分布图

fdist1 |= fdist2 update fdist1 with counts from fdist2

fdist1 < fdist2 测试样本在 fdist1 中出现的频率是否小于 fdist2

**1.4 回到 Python:决策与控制**

一些词比较运算符函数 含义

s.startswith(t) 测试 s 是否以 t 开头

s.endswith(t) 测试 s 是否以 t 结尾

t in s 测试 s 是否包含 t

s.islower() 测试 s 中所有字符是否都是小写字母

s.isupper() 测试 s 中所有字符是否都是大写字母

s.isalpha() 测试 s 中所有字符是否都是字母

s.isalnum() 测试 s 中所有字符是否都是字母或数字

s.isdigit() 测试 s 中所有字符是否都是数字

s.istitle() 测试 s 是否首字母大写(s 中所有的词都首字母大写)

>>> len(set([word.lower() for word in text1 if word.isalpha()]))

16948

这个例子稍微有些复杂:将所有纯字母组成的词小写。也许只计数小写的词会更简单一

些，但这却是一个错误的答案(因为这样会漏掉只有大写没有对应小写的词)。

**1.5 自动理解自然语言**

词义消歧

指代消解

自动生成语言：

自动问答 机器翻译: 文本对齐

人机对话系统

文本的含义