

上讲回顾：统计思维与机器学习

- 统计思维
 - 描述统计 (Descriptive Statistics)
 - 推论统计 (Inferential Statistics)
 - 实用机器学习
 - 研究设计 (Study Design)
 - 核心概念 (Conceptual Issues)
 - 评测方法 (Evaluation Methods)
- 
- LAB01 - EDA
Exploratory
Data Analysis



中國人民大學
RENMIN UNIVERSITY OF CHINA

计算传播理论与实务

2019-2020秋季学期

第三讲

文本分析

授课教师：范举副教授、塔娜讲师

时间：2019年10月14日

文本覆盖了人类生活的方方面面

Topics:

People
Events
Products
Services, ...



Sources:

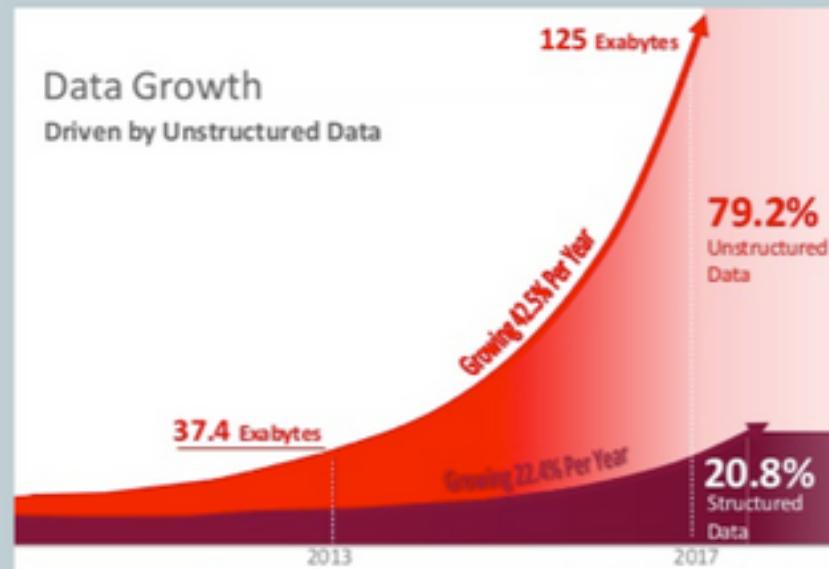
Blogs
Microblogs
Forums
Reviews, ...



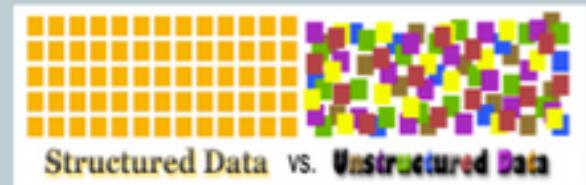
文本数据体量迅速增长

Structured and Unstructured Data Growth

IDC Study: Structured Versus Unstructured Data: The Balance of Power Continues to Shift



"80% of business-relevant information originates in unstructured form, primarily text."



典型的文本处理与分析场景

• 搜索引擎



• 推荐系统



• 自然语言处理

- 机器翻译
- 问答系统

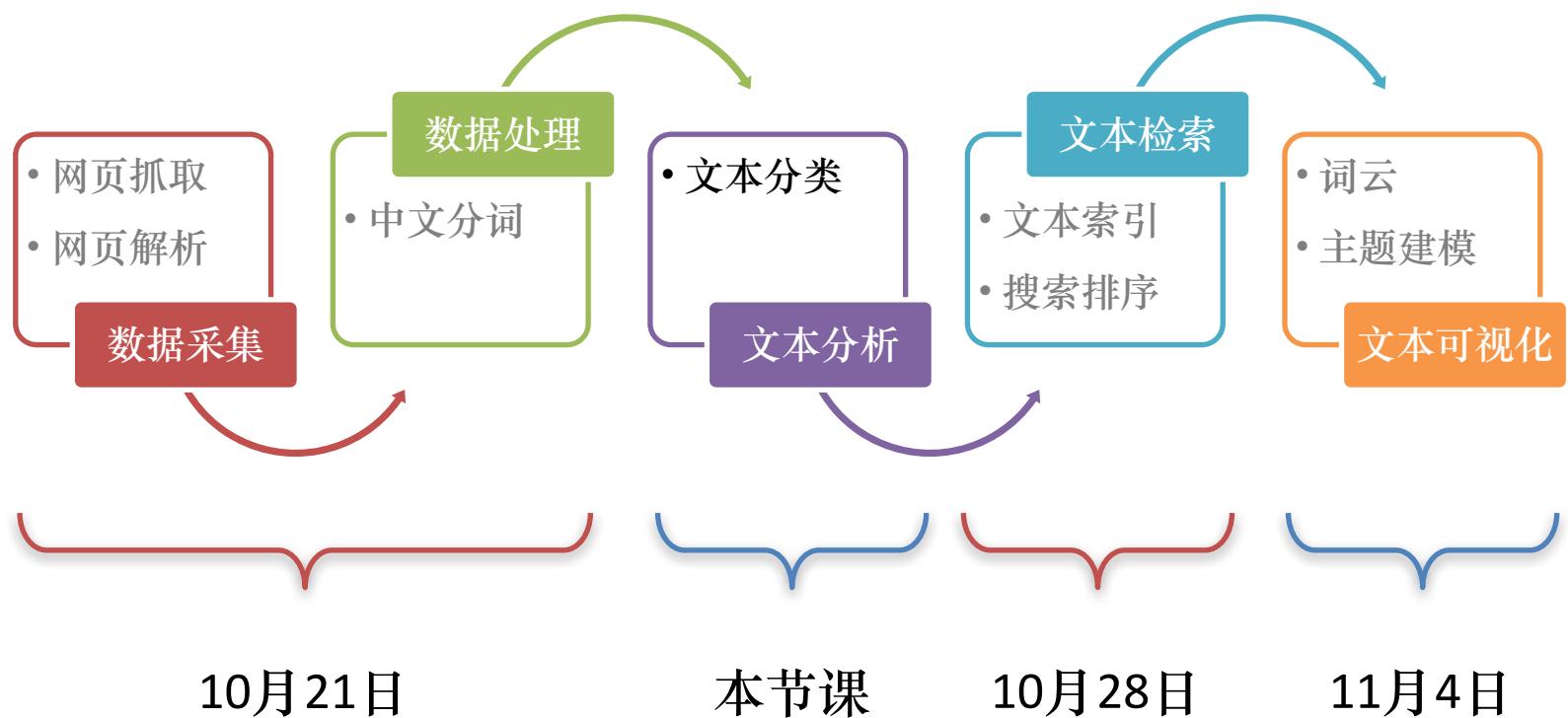
在被**潮汐锁定**的星球上生活是一种怎样的体验?
计算传播理论与实务
ace zh: 太阳长期在天空同一个位置。太阳直射点称为赤道，朝向赤道的方向称为南方，相对的方向为北方。

自动问答系统的应用

- 二哈：阿里巴巴的智能防骚扰电话技术

文本模块主要内容

- 目标：自动分析出人们在“说”什么
 - 大多新闻与传播领域的数据以文本形态存在



第3.1节

文本分类

文本分类举例——垃圾信息过滤

From: “” <takworlld@hotmail.com>
Subject: real estate is the only way... gem oalvgkay

Anyone can buy real estate with no money down

Stop paying rent TODAY !

There is no need to spend hundreds or even thousands for similar courses

I am 22 years old and I have already purchased 6 properties using the methods outlined in this truly INCREDIBLE ebook.

Change your life NOW !

=====
Click Below to order:
<http://www.wholesaledaily.com/sales/nmd.htm>
=====

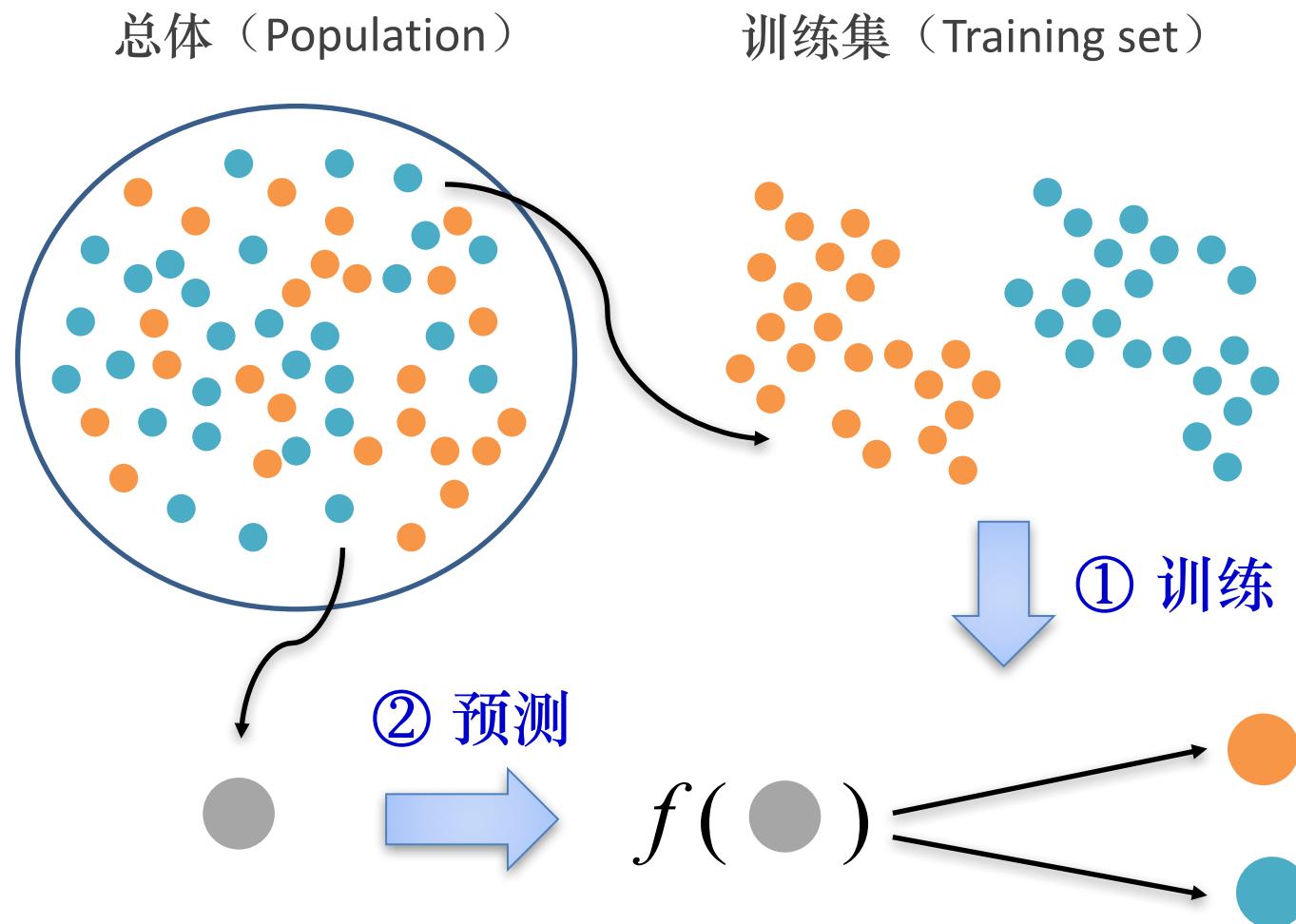
垃圾邮件

标签	短信内容
0	商业秘密的秘密性那是维系其商业价值和垄断地位的前提条件之一
1	南口阿玛施新春第一批限量春装到店啦 春暖花开淑女裙、冰蓝色公主衫 气质粉小西装、冰丝女王长半裙
0	带给我们大常州一场壮观的视觉盛宴
0	23年从盐城拉回来的麻麻的嫁妆
1	感谢致电杭州萧山全金釜韩国烧烤店，本店位于金城路xxx号。韩式烧烤等，价格实惠、欢迎惠顾【全金釜韩国烧烤店】
0	这款UVe智能杀菌机器人是扫地机的最佳伴侣
1	一次价值xxx元王牌项目；可充值xxx元店内项目卡一张；可以参与V动好生活百分百抽奖机会一次！预约电话：xxxxxxxxxxxx



垃圾短信

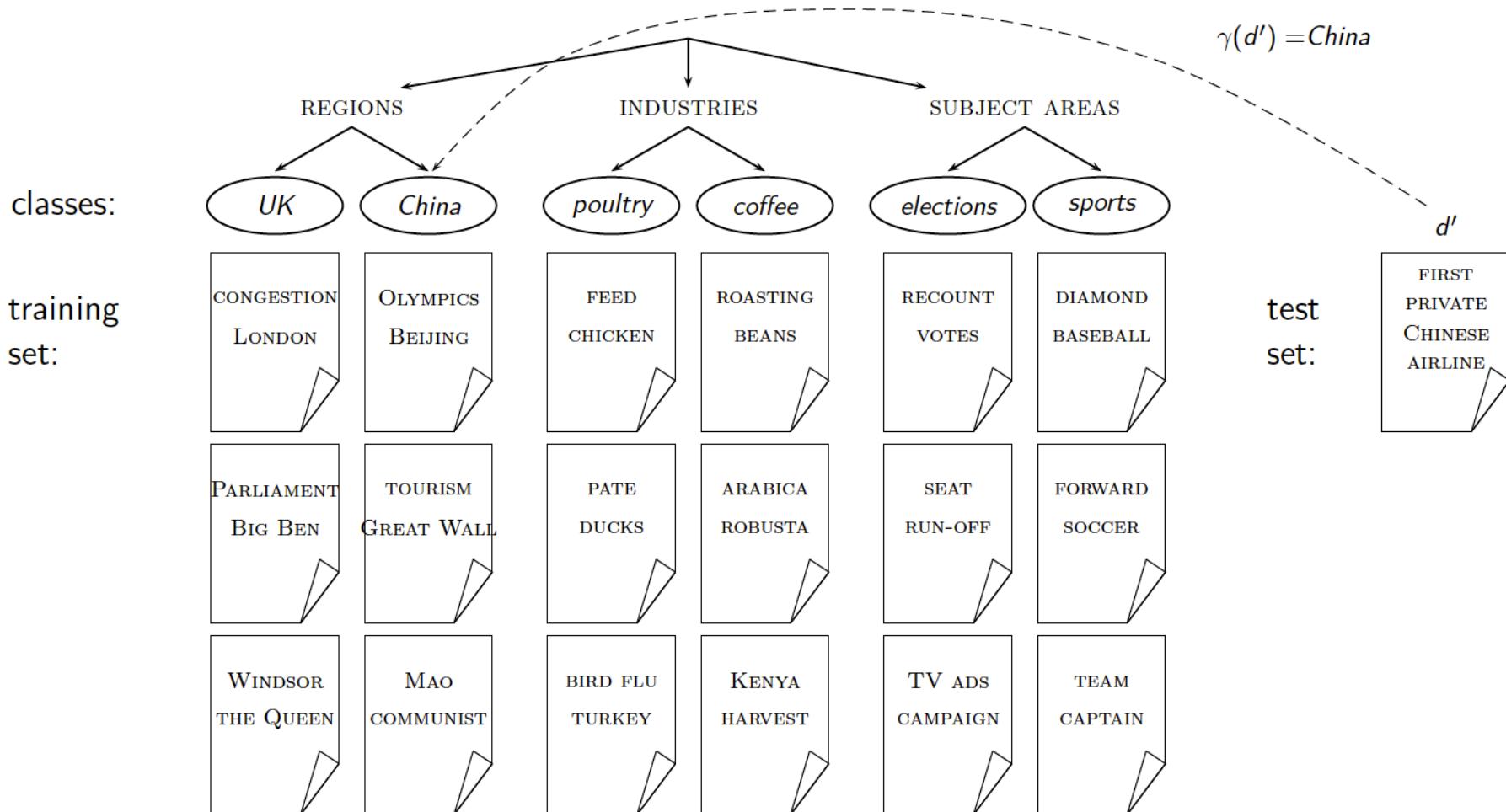
回顾：机器学习如何预测



文本分类的形式化定义

- 训练过程：给定
 - 文档空间 X
 - 所有的文档都在此空间进行表示，一般表示为高维向量
 - 文档类别 $C = \{c_1, \dots, c_J\}$
 - 依据目标分类任务人工定义，比如 $C = \{\text{垃圾信息}, \text{正常信息}\}$
 - 带标签的训练集合 $D = \{(d_i, c_i)\}_{i=1}^N, (d_i, c_i) \in X \times C$
 - 基于学习算法，训练得到一个分类器 γ
$$\gamma: X \mapsto C$$
- 应用/测试过程
 - 给定：任意一个（可能在训练集合D中未出现）文档 $d \in X$
 - 确定： $\gamma(d) \in C$ ，即对 d 而言最合适的类别

举例：话题分类



提问

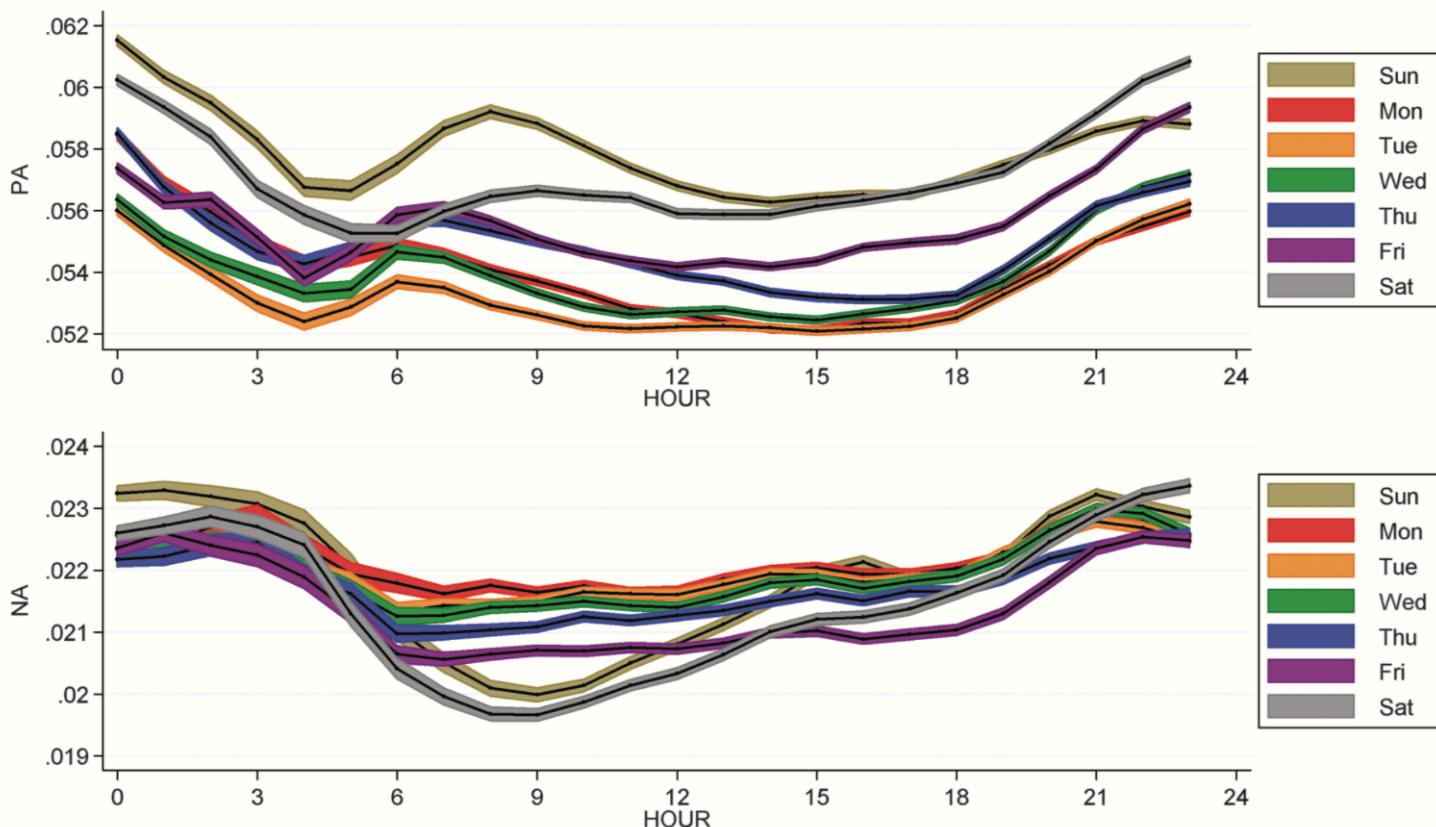
- 列举新闻传播领域文本分类的应用案例

文本分类举例

- 语言识别（中文，英文，日文，...）
 - 垃圾网页识别（垃圾网页，正常网页）
 - Fake News识别（虚假信息，正常信息）
 - 情感识别（正面、负面）
 - 主页识别（主页、一般网页）
-

回顾：使用Tweet做情绪分析

- 收集个体持续产生的行为数据
 - 针对用户每天发布的Tweets进行文本分析



Source: Golder and Macy: Diurnal and Seasonal Mood Vary with Work, Sleep, and Daylength Across Diverse Cultures. Science, VOL 333, 2011

文本分类方法#1：人工分类

- 雅虎最先提出将网页进行手工分类，ODP、PubMed也进行了类似的实践
 - 准确率高（如果雇佣专家进行分类）
 - 稳定（如果问题和团队规模可控）
- 规模化难题
 - 费用太高
 - 速度慢
 - 专家团队人数太多后，准确率难以保证

The screenshot shows the Yahoo! Directory homepage. At the top, there's a search bar and a link to 'YAHOO! DIRECTORY'. Below the header, there's a section titled 'Yahoo! Directory' with several category links arranged in a grid:

Arts & Humanities Photography, History, Literature...	News & Media Newspapers, Radio, Weather, Blogs...
Business & Economy B2B, Finance, Shopping, Jobs...	Recreation & Sports Sports, Travel, Autos, Outdoors...
Computer & Internet Hardware, Software, Web, Games...	Reference Phone Numbers, Dictionaries, Quotes...
Education Colleges, K-12, Distance Learning...	Regional Countries, Regions, U.S. States...
Entertainment Movies, TV Shows, Music, Humor...	Science Animals, Astronomy, Earth Science...
Government Elections, Military, Law, Taxes...	Social Science Languages, Archaeology, Psychology...
Health Disease, Drugs, Fitness, Nutrition...	Society & Culture Sexuality, Religion, Food & Drink...

The screenshot shows the 'CATEGORIES (What's This?)' page. It has a sidebar with 'Top Categories' and a main content area with 'Additional Categories'.

Top Categories

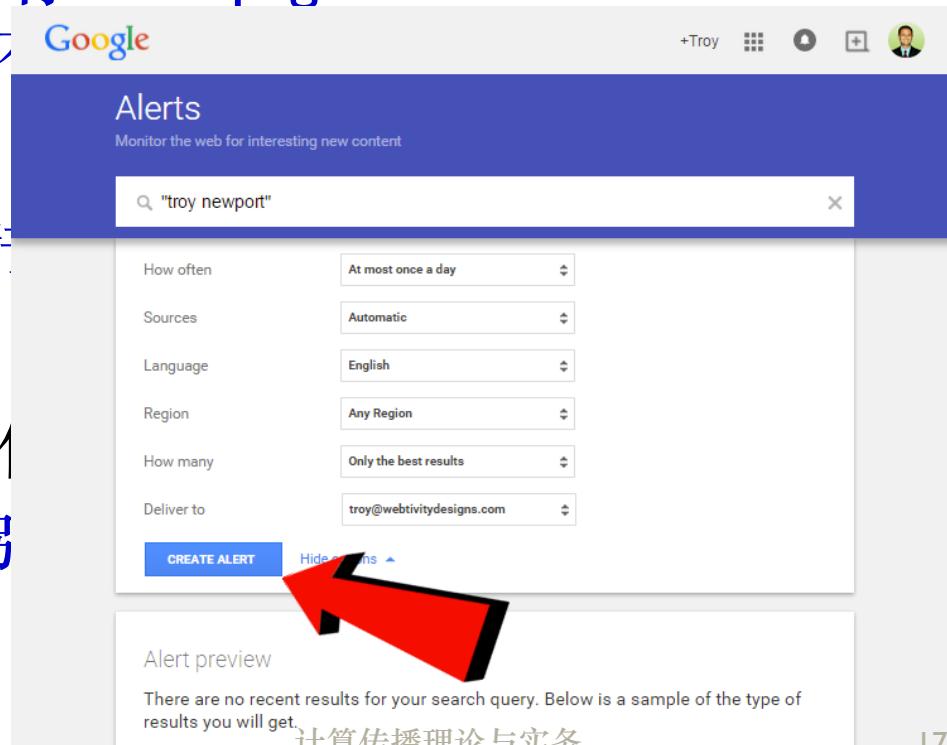
- Communications and Networking (1284)
- Data Formats (439)
- Graphics (260)
- Hardware (1753)
- Internet (4083)

Additional Categories

- Business to Business@
- Chats and Forums@
- Computer Generated Art@
- Computer Generated Music@
- Computer Science@
- Multimedia (495)
- News and Media (495)
- Programming and Development (958)
- Security and Encryption (567)
- Software (3235) NEW
- Information Technology@
- Issues (11)
- Macintosh (391)
- Magazines@
- Mobile Computing (52)

文本分类方法#2：基于规则分类

- 例如：Google alerts是基于规则的分类
 - 通过辅助工具创建分类规则
 - 通常规则为布尔表达式
如 “” 标题中含有 homepage”
AND ” URL中含”
- 优点：准确率高
 - 如果雇佣专家进行
 - 易于理解
- 缺点：规则维护困难
 - 分类规则甚至类别
 - 规则冲突问题



文本分类方法#3：基于机器学习

- 把文本分类问题看成一个机器学习问题
 - 基于监督学习算法，训练一个分类器 $\gamma: X \mapsto C$
 - 把 γ 应用于新文档 d : $\gamma(d) \in C$
- 没有免费的午餐
 - 需要手工标注的训练数据（数据驱动）
 - 需要把文本表达为合适的特征向量

文本分类方法

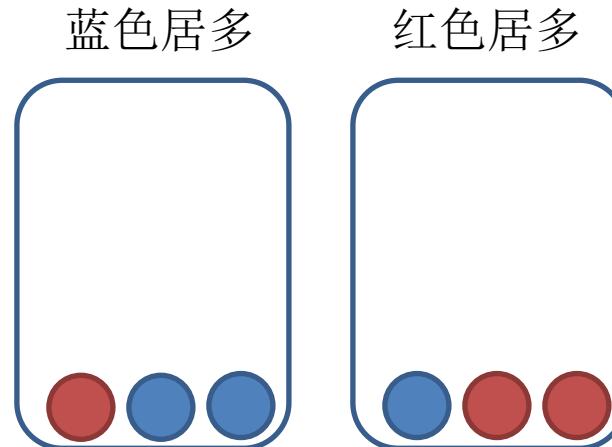
- 朴素贝叶斯Naïve Bayes – 掌握
- 支持向量机SVM – 了解
- 分类效果评价 – 掌握
- 文本分类的Python实现 – 掌握

第3.1.1节 文本分类方法

朴素贝叶斯 Naïve Bayes

什么是贝叶斯？

- 一个例子：红蓝石子游戏（Marble Game）
 - 在一个罐子里面放三颗石子
 - 可能有两种石子的放法
 - 蓝色居多：2颗蓝色、1颗红色
 - 红色居多：2颗红色、1颗蓝色
 - 如果你不知道具体放法，只让你随机摸出一颗
 - 如果摸出的是蓝色（红色），判断放法是蓝色居多还是红色居多的概率是多少？



贝叶斯公式

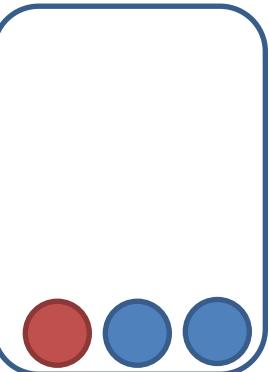
- 定义随机变量
 - 随机变量 C : 表示罐子的类型
 - $C = c_1$: 表示罐子属于蓝色为主型
 - $C = c_2$: 表示罐子属于红色为主型
 - 随机变量 X : 表示摸出石子的颜色
 - $X = r$: 表示摸出了红色石子
 - $X = b$: 表示摸出了蓝色石子
- 需要计算的概率

$$P(C = c_1 | X = r) = \frac{P(X = r | C = c_1)P(C = c_1)}{P(X = r | C = c_1)P(C = c_1) + P(X = r | C = c_2)P(C = c_2)}$$

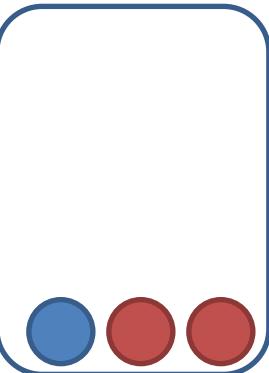
什么是贝叶斯?

- 一个例子：红蓝石子游戏 (Marble Game)
 - 问题2：如果一组同学一个接一个的摸石子，每个同学公布颜色后，再把石子放回
 - 第一个同学摸出蓝色，怎么猜？
 - 第二个同学摸出红色，怎么猜？
 - 第三个同学摸出蓝色，怎么猜？

蓝色居多



红色居多



朴素贝叶斯 (Naïve Bayes)

- 朴素贝叶斯属于概率分类器，给定一个文档d，其属于类别c的概率为

$$P(c|d) \propto P(c) \prod_{1 \leq k \leq n_d} P(t_k|c)$$

- n_d : 文档长度
- $P(t_k|c)$: 给定类别c，出现单词 t_k 的概率
 - 通过单词 t_k 来度量类别c是否是文档d的正确类别
- $P(c)$: 类别c的先验概率
 - 如果一个文档没有任何的迹象（单词）表明它属于类别c的概率要大于属于 c' 的概率，那么则选择先验概率较大的类别

贝叶斯公式

- 分类目标：找出文档最可能属于的类别

$$c_{\text{map}} = \arg \max_{c \in \mathcal{C}} P(c|d)$$

- 应用贝叶斯公式 $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$

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

- 由于 $P(d)$ 与类别选取没有关系，因此可以去除

$$c_{\text{map}} = \arg \max_{c \in \mathcal{C}} P(d|c)P(c)$$

最大化后验概率

- 朴素贝叶斯的目标
 - 为文档找到“最好”的类
- 什么是最好的类?
 - 能够最大化后验概率的类别 (maximum a posteriori (MAP) class), 记为 c_{map}

$$c_{\text{map}} = \arg \max_{c \in \mathcal{C}} \hat{P}(c|d) = \arg \max_{c \in \mathcal{C}} \hat{P}(c) \prod_{1 \leq k \leq n_d} \hat{P}(t_k|c)$$

通过log把乘法变成加法

- 计算机表达概率相乘所遇到的问题
 - 如果把很多个概率（0~1之间的数字）相乘，会很快得到低于计算机所能表达精度的数字（floating point underflow），从而得到0概率
- 解决方案： $\log(xy) = \log(x) + \log(y)$
 - 通过求log和的方式表达积的log
- 注意：log是一个单增函数，取log后的函数并不改变其最大值点
- 因此，Naïve Bayes实际去计算

$$c_{\text{map}} = \arg \max_{c \in \mathcal{C}} \left[\log \hat{P}(c) + \sum_{1 \leq k \leq n_d} \log \hat{P}(t_k | c) \right]$$

朴素贝叶斯分类器

$$c_{\text{map}} = \arg \max_{c \in \mathcal{C}}$$

$$\left[\log \hat{P}(c) + \sum_{1 \leq k \leq n_d} \log \hat{P}(t_k | c) \right]$$

选取概率最大的类别

Log先验概率与词权重的和：总体而言文档属于此类别的概率

c的先验概率，即类别c在总体数据中出现的频率

单词 t_k 对类别c的权重，表明当出现此单词时表明其类别应该为c的信号强度

参数估计

- 给定训练数据之后，如何估计上述概率（参数）？
- 需要估计的参数包括
 - 先验概率： $P(c)$, for all $c \in C = \{c_1, \dots, c_J\}$
 - 条件概率： $P(t_k|c)$, for all $t_k \in V$, and $c \in C = \{c_1, \dots, c_J\}$, 其中 V 为所有可能的单词

$$c_{\text{map}} = \arg \max_{c \in \mathcal{C}} \left[\log \hat{P}(c) + \sum_{1 \leq k \leq n_d} \log \hat{P}(t_k|c) \right]$$

参数估计方法：最大似然估计

- 先验概率：

$$\hat{P}(c) = \frac{N_c}{N}$$

- N_c : 训练集合中属于类别c的文档数目；N: 训练集合中所有文档数目

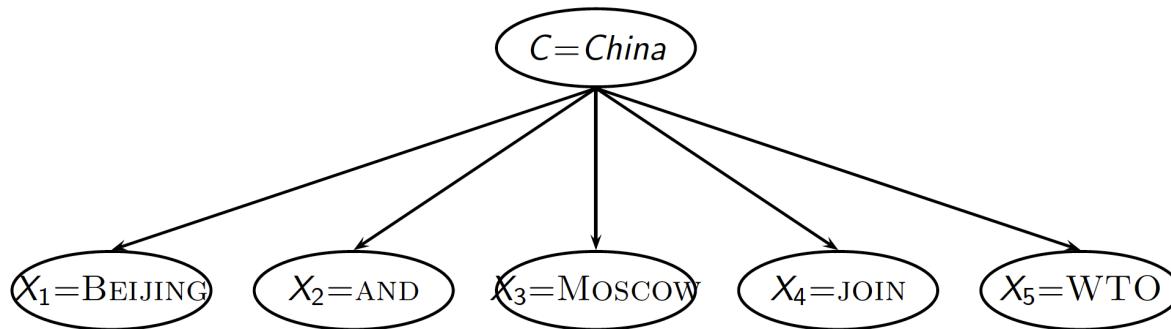
- 条件概率：

$$\hat{P}(t|c) = \frac{T_{ct}}{\sum_{t' \in V} T_{ct'}}$$

- T_{ct} : 在训练集合中，属于类别c的文档中包含单词t的次数（包括重复出现）

Naïve Bayes独立性假设：单词t在类别c中出现的概率其所出现的位置和上下文无关

最大似然估计的问题：零概率



$$P(China|d) \propto P(China) \cdot P(BEIJING|China) \cdot P(AND|China) \\ \cdot P(MOSCOW|China) \cdot P(JOIN|China) \cdot P(WTO|China)$$

- 如果WTO从来没有在类别China中出现过

$$\hat{P}(WTO|China) = \frac{T_{China,WTO}}{\sum_{t' \in V} T_{China,t'}} = \frac{0}{\sum_{t' \in V} T_{China,t'}} = 0$$

最大似然估计的问题：零概率

- 在训练数据中，属于China的文档中从来没有出现过单词WTO，则

$$\hat{P}(WTO|China) = \frac{T_{China,WTO}}{\sum_{t' \in V} T_{China,t'}} = \frac{0}{\sum_{t' \in V} T_{China,t'}} = 0$$

- 依据分类规则，则所有含有WTO的文档属于China类别的概率为0：
 - $P(China|d) = 0$ ，如果 d 中含有单词WTO

$$c_{\text{map}} = \arg \max_{c \in \mathcal{C}} \hat{P}(c|d) = \arg \max_{c \in \mathcal{C}} \hat{P}(c) \prod_{1 \leq k \leq n_d} \hat{P}(t_k|c)$$

如何避免零概率：概率平滑

- 最大似然条件概率估计：

$$\hat{P}(t|c) = \frac{T_{ct}}{\sum_{t' \in V} T_{ct'}}$$

- 平滑化：在所有的统计次数上加一次

$$\hat{P}(t|c) = \frac{T_{ct} + 1}{\sum_{t' \in V} (T_{ct'} + 1)} = \frac{T_{ct} + 1}{\sum_{t' \in V} T_{ct'} + B}$$

- $B=|\mathcal{V}|$: 字典中所含有不同的单词数目

朴素贝叶斯小结

- 训练：平滑化基于训练数据的概率数值，避免零概率
- 应用/测试：给定一个文档，
 - 计算每一个类别的log先验概率
 - 计算每一个文档中所出现的单词的log后验概率
 - 求和
 - 将分值最高的类别作为预测的类

朴素贝叶斯训练算法

TRAINMULTINOMIALNB(\mathbb{C}, \mathbb{D})

```
1  $V \leftarrow \text{EXTRACTVOCABULARY}(\mathbb{D})$ 
2  $N \leftarrow \text{COUNTDOCS}(\mathbb{D})$ 
3 for each  $c \in \mathbb{C}$ 
4 do  $N_c \leftarrow \text{COUNTDOCSINCLASS}(\mathbb{D}, c)$ 
5    $prior[c] \leftarrow N_c/N$ 
6    $text_c \leftarrow \text{CONCATENATETEXTOFTALLDOCSINCLASS}(\mathbb{D}, c)$ 
7   for each  $t \in V$ 
8     do  $T_{ct} \leftarrow \text{COUNTTOKENSOFTERM}(text_c, t)$ 
9     for each  $t \in V$ 
10    do  $condprob[t][c] \leftarrow \frac{T_{ct}+1}{\sum_{t'}(T_{ct'}+1)}$ 
11 return  $V, prior, condprob$ 
```

朴素贝叶斯应用/测试

APPLYMULTINOMIALNB($\mathbb{C}, V, prior, condprob, d$)

- 1 $W \leftarrow \text{EXTRACTTOKENSFROMDOC}(V, d)$
- 2 **for each** $c \in \mathbb{C}$
- 3 **do** $score[c] \leftarrow \log prior[c]$
- 4 **for each** $t \in W$
- 5 **do** $score[c] += \log condprob[t][c]$
- 6 **return** $\arg \max_{c \in \mathbb{C}} score[c]$

练习

	docID	words in document	in $c = China?$
training set	1	Chinese Beijing Chinese	yes
	2	Chinese Chinese Shanghai	yes
	3	Chinese Macao	yes
	4	Tokyo Japan Chinese	no
test set	5	Chinese Chinese Chinese Tokyo Japan	?

$$\hat{P}(c) = \frac{N_c}{N}$$

$$\hat{P}(t|c) = \frac{T_{ct} + 1}{\sum_{t' \in V} (T_{ct'} + 1)} = \frac{T_{ct} + 1}{(\sum_{t' \in V} T_{ct'}) + B}$$

$$c_{\text{map}} = \arg \max_{c \in \mathcal{C}} \hat{P}(c|d) = \arg \max_{c \in \mathcal{C}} \hat{P}(c) \prod_{1 \leq k \leq n_d} \hat{P}(t_k|c)$$

训练过程 - I

- 步骤1：提取词典 (Vocabulary)
 - $V = \{\text{Chinese}, \text{Beijing}, \text{Shanghai}, \text{Macao}, \text{Tokyo}, \text{Japan}\}$
- 步骤2：计算训练文档个数
 - $N = 4$
- 步骤3：针对正例文档，表示为 c
 - 计算正例文档个数 $N_c = 3$
 - 计算先验概率 $P(c) = \frac{3}{4} = 0.75$
 - 将正例文档进行合并形成文本 text_c

训练过程 - 2

- 步骤3：针对正例文档，即 $c=China$
 - 针对词典 V 中的每个单词，进行如下计算

$$\bullet P(Chinese|c) = \frac{5+1}{8+6} = \frac{3}{7}$$

$$\bullet P(Beijing|c) = \frac{1+1}{8+6} = \frac{1}{7}$$

$$\bullet P(Shanghai|c) = \frac{1+1}{8+6} = \frac{1}{7}$$

$$\bullet P(Macao|c) = \frac{1+1}{8+6} = \frac{1}{7}$$

$$\bullet P(Tokyo|c) = \frac{0+1}{8+6} = \frac{1}{14}$$

$$\bullet P(Japan|c) = \frac{0+1}{8+6} = \frac{1}{14}$$

训练过程 - 3

- 步骤4：针对负例文档，表示为 \bar{c}

- 计算先验概率 $P(\bar{c}) = \frac{1}{4} = 0.25$

- 将负例文档进行合并形成文本text $_{\bar{c}}$

- $P(Chinese|\bar{c}) = \frac{1+1}{3+6} = \frac{2}{9}$

- $P(Beijing|\bar{c}) = \frac{0+1}{3+6} = \frac{1}{9}$

- $P(Shanghai|\bar{c}) = \frac{0+1}{3+6} = \frac{1}{9}$

- $P(Macao|\bar{c}) = \frac{0+1}{3+6} = \frac{1}{9}$

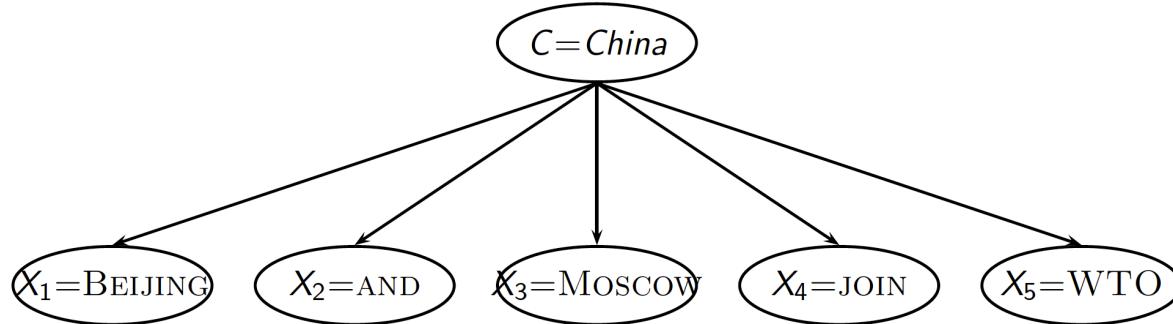
- $P(Tokyo|\bar{c}) = \frac{1+1}{3+6} = \frac{2}{9}$

- $P(Japan|\bar{c}) = \frac{1+1}{3+6} = \frac{2}{9}$

测试（预测）过程

- 给定测试文本
 - Chinese Chinese Chinese Tokyo Japan
- 计算正例的后验概率
 - $P(c|d) = \frac{3}{4} \cdot \left(\frac{3}{7}\right)^3 \cdot \frac{1}{14} \cdot \frac{1}{14} \approx 0.0003$
- 计算负例的后验概率
 - $P(\bar{c}|d) = \frac{1}{4} \cdot \left(\frac{2}{9}\right)^3 \cdot \frac{2}{9} \cdot \frac{2}{9} \approx 0.0001$
- 由于 $P(c|d) > P(\bar{c}|d)$, 预测测试文本的类别为正例
- 请你计算以下文本
 - Beijing Tokyo Japan

生成模型 – Generative Model



- $P(c|d) \propto P(c) \prod_{1 \leq k \leq n_d} P(t_k|c)$
 - 依据 $P(c)$ 生成一个类别 c
 - 给定类别 c ，依据概率 $P(t_k|c)$ 独立地生成每一个单词 t_k
- 为了给文档进行分类，“reengineering”上述过程，寻找能够有最大生成概率的 c

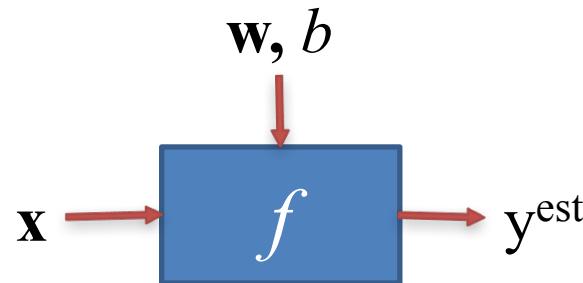
并不“朴素”的朴素贝叶斯

- 赢得KDD-CUP 1997年冠军
- 相对于大部分传统的机器学习模型
 - 对无关的特征比较鲁棒
 - 对topic drift（比如：类别的定义随着时间而变化）鲁棒
- 相对决策树
 - 如果一些特征同样重要，则朴素贝叶斯表现更好
- 如果独立性假设成立，会有惊艳的表现
- 文本分类中一类重要的模型，经常被用作基线模型
 - 简单：没有超参数
 - 时间：速度快、效率高
 - 空间：需要存储空间少

第3.1.1节 文本分类方法

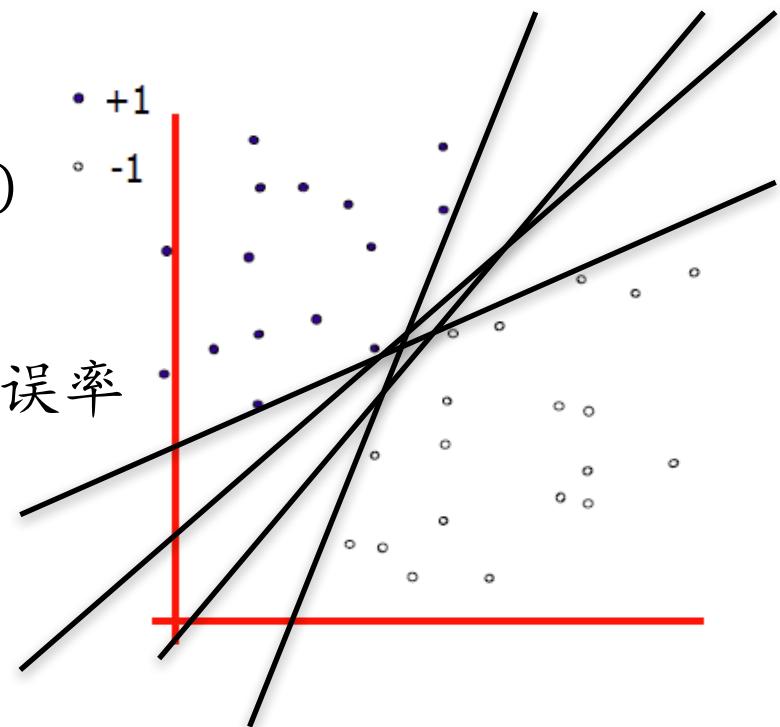
支持向量机SVM

线性分类器



$$f(\mathbf{x}; \mathbf{w}, b) = \text{sgn}(\langle \mathbf{w}, \mathbf{x} \rangle + b)$$

所有的模型(分类面)错误率为0, 哪个最好?



简单的学习规则

- 线性感知机(Perceptron)算法
- 输入: 训练数据 $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$, 学习步长 $\eta > 0$
- 输出: 感知机模型 \mathbf{w}, b

1. $(\mathbf{w}, b) \leftarrow$ 随机值

2. $R \leftarrow \max_i |\mathbf{x}_i|$

3. **repeat**

4. 选取数据 $(x_i, y_i) \in D$

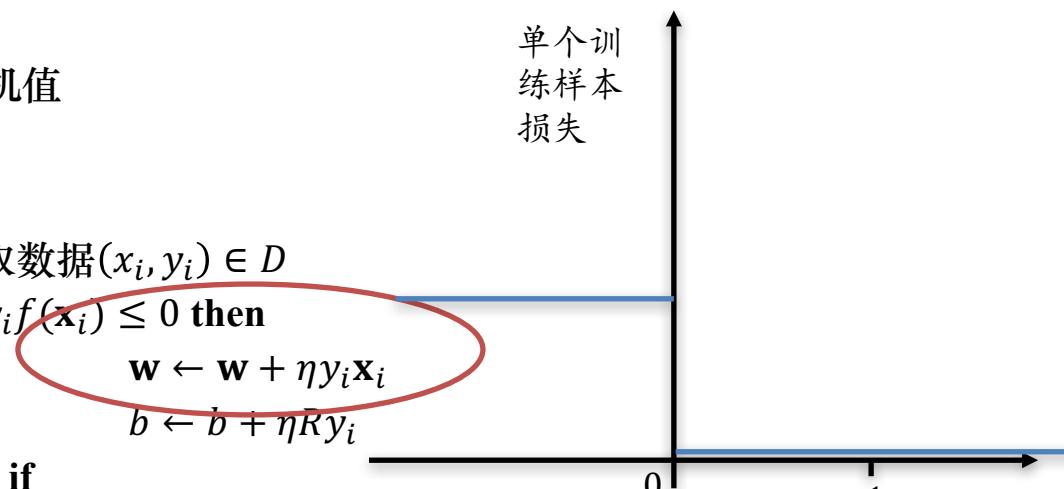
5. **if** $y_i f(\mathbf{x}_i) \leq 0$ **then**

6. $\mathbf{w} \leftarrow \mathbf{w} + \eta y_i \mathbf{x}_i$

7. $b \leftarrow b + \eta R y_i$

8. **end if**

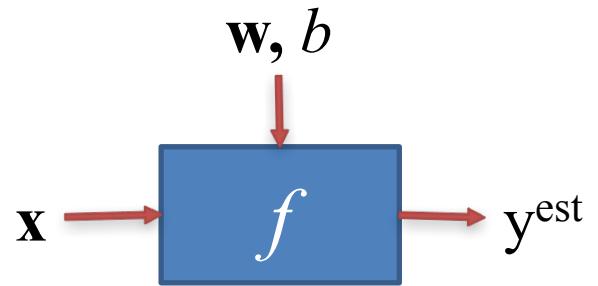
9. **until** convergence



关于更新规则的说明

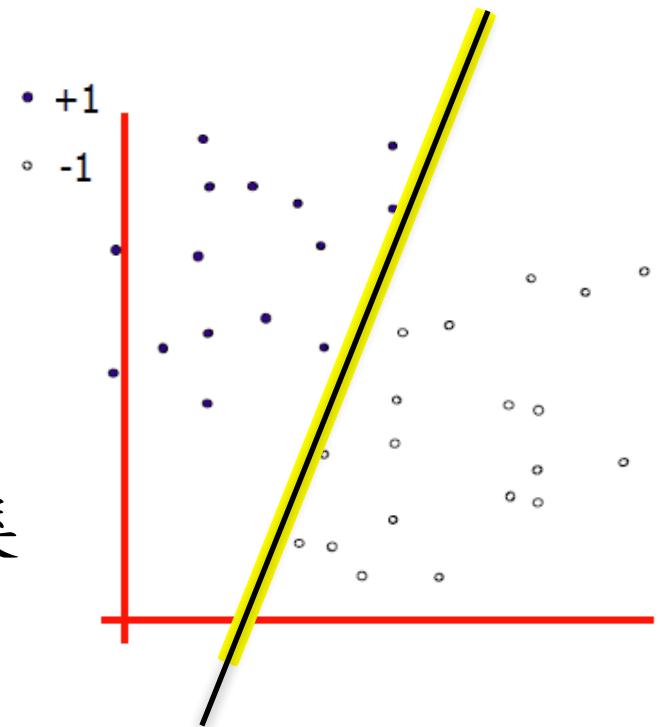
- $\mathbf{w}^{t+1} = \mathbf{w}^t + \eta y_i \mathbf{x}_i, b^{t+1} \leftarrow b^t + \eta R y_i$
- $y_i f^{t+1}(\mathbf{x}_i)$
 $= y_i (\langle \mathbf{w}^{t+1}, \mathbf{x}_i \rangle + b^{t+1})$
 $= y_i (\langle \mathbf{w}^t + \eta y_i \mathbf{x}_i, \mathbf{x}_i \rangle + b^t + \eta R y_i)$
 $= y_i \cancel{\langle \mathbf{w}^t, \mathbf{x}_i \rangle} + y_i \langle \eta y_i \mathbf{x}_i, \mathbf{x}_i \rangle + \cancel{y_i b^t} + \eta R y_i y_i$
 $= y_i f^t(\mathbf{x}_i) + \eta (\langle \mathbf{x}_i, \mathbf{x}_i \rangle + R)$
 $> y_i f^t(\mathbf{x}_i)$
- 每次更新都将减少当前的错误
- 收敛性：在线性可分的数据上，在有限步更新后收敛(Novikoff (1962))

分类边距(Margin)



$$f(\mathbf{x}; \mathbf{w}, b) = \text{sgn}(\langle \mathbf{w}, \mathbf{x} \rangle + b)$$

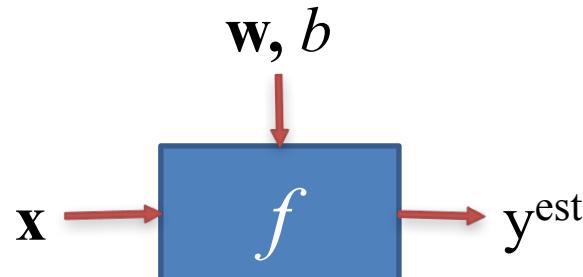
分类边距(Margin):对线性分类器而言，边距是从分类面到最近的训练样本的距离。



带边距感知机 (Perceptron with Margin)

- 输入: 训练数据 $D = \{(x_i, y_i)\}_{i=1}^N$, 学习步长 $\eta > 0$, 边距 $\tau > 0$
 - 输出: 感知机模型 \mathbf{w}, b
1. $(\mathbf{w}, b) \leftarrow$ 随机值
 2. $R \leftarrow \max_i |\mathbf{x}_i|$
 3. **repeat**
 4. 选取数据 $(x_i, y_i) \in D$
 5. **if** $y_i f(\mathbf{x}_i) \leq \tau$ **then**
 6. $\mathbf{w} \leftarrow \mathbf{w} + \eta y_i \mathbf{x}_i$
 7. $b \leftarrow b + \eta R y_i$
 8. **end if**
 9. **until** convergence

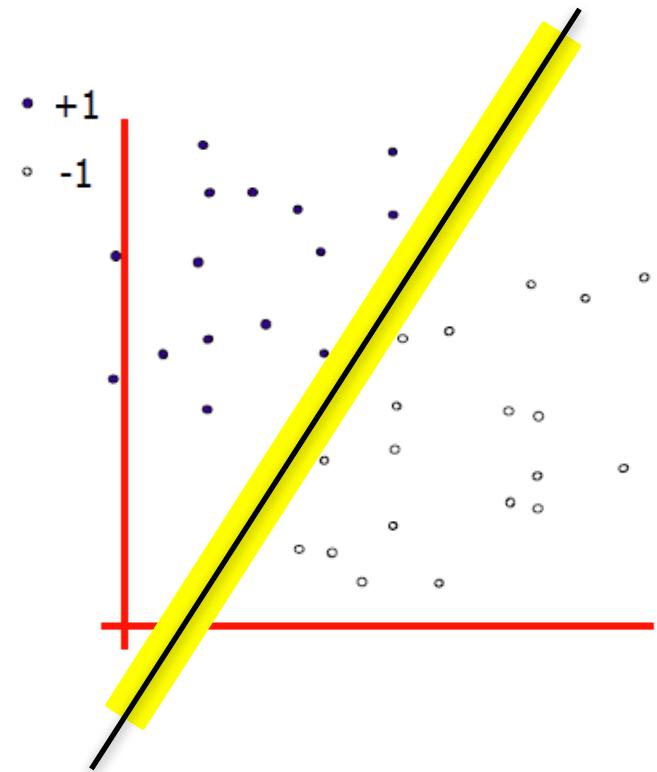
SVM: 最大边距分类器



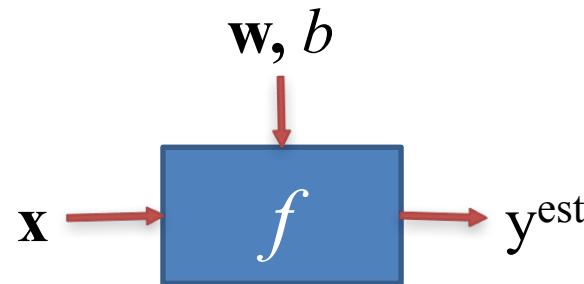
$$f(\mathbf{x}; \mathbf{w}, b) = \text{sgn}(\langle \mathbf{w}, \mathbf{x} \rangle + b)$$

不同的模型有不同大小的边距。

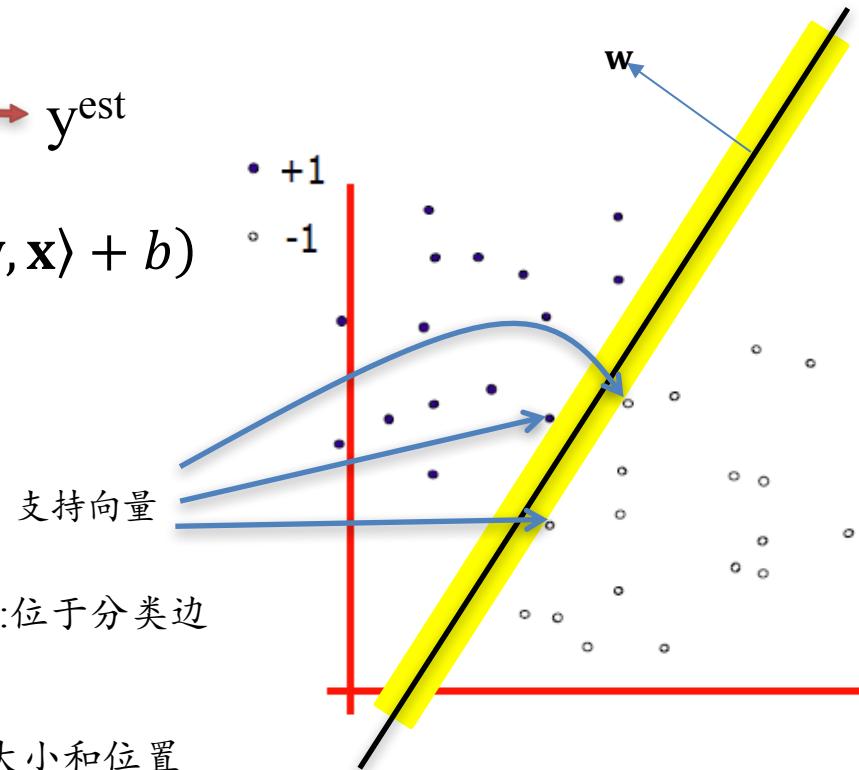
SVM (support vector machine) 的目标:
最大化分类边距



支持向量



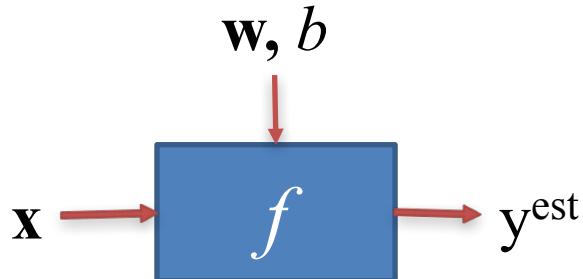
$$f(\mathbf{x}; \mathbf{w}, b) = \text{sgn}(\langle \mathbf{w}, \mathbf{x} \rangle + b)$$



支持向量(support vector):位于分类边距上的数据点

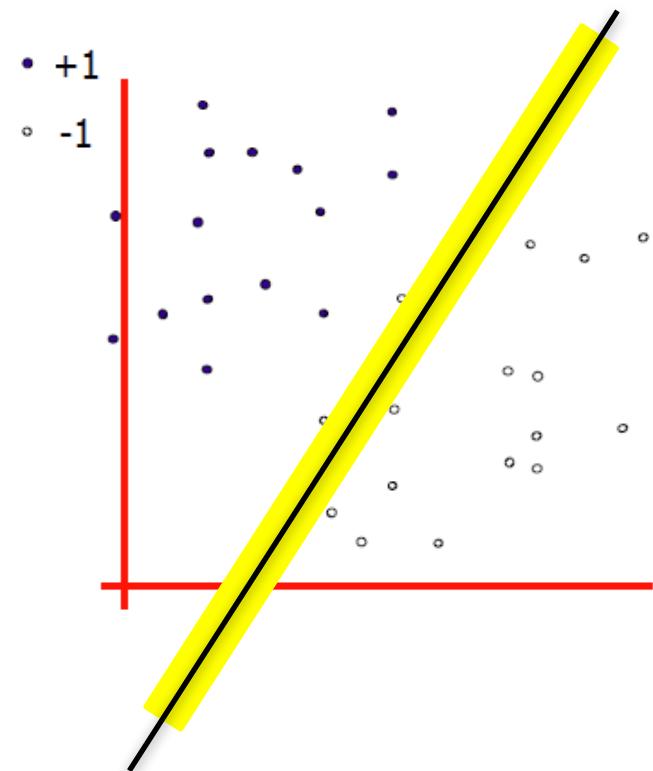
支持向量决定了边距的大小和位置
给定训练数据集合, \mathbf{w} 的**方向**决定了边距大小

为什么最大化边距?



$$f(\mathbf{x}; \mathbf{w}, b) = \text{sgn}(\langle \mathbf{w}, \mathbf{x} \rangle + b)$$

- 直觉上，比其它分类界面更不容易出错
 - 分类面的微小扰动不会导致错误发生
- 理论上，VC理论证明最大边距分类器具有较好的泛化能力(generalization ability)
- 实际应用：SVM在很多应用上取得了很好的效果



第3.1.3节 文本分类方法

分类效果评测

评价分类效果

- 对模型的评价必须在与训练数据独立的**测试数据**上进行，即：训练数据与测试数据不能有交集
 - 如果违背，则测试结果虚高
 - 极端情况：记住所有的训练数据，则可以得到100%正确的测试结果
 - 记住：我们目的是为了分类“**未曾见过**”文档
- 评价指标
 - Accuracy
 - Precision (P), Recall (R), F-score (F)

分类模型评价—评价准则

	预测为正样本	预测为负样本
标注为正样本	TP (true positive)	FN (false negative)
标注为负样本	FP (false positive)	TN (true negative)

- 常用的分类评价准则

- Accuracy = $\frac{\text{正确分类的样本数}}{\text{总样本数}} = \frac{TP+TN}{TP+FN+FP+TN}$
- Precision = $\frac{\text{正确预测的正样本数}}{\text{预测为正例的样本数}} = \frac{TP}{TP+FP}$
- Recall = $\frac{\text{正确预测的正样本数}}{\text{标注的正样本数}} = \frac{TP}{TP+FN}$
- $F = \frac{1}{\frac{1}{2}(\frac{1}{P} + \frac{1}{R})} = \frac{2PR}{P+R}$
 - F是P和R的harmonic mean: $\frac{1}{F} = \frac{1}{2} \left(\frac{1}{R} + \frac{1}{P} \right)$

问题：在什么情形下不能使用accuracy作为评价指标？

Accuracy存在的问题

	预测为正样本	预测为负样本	
标注为正样本	0	10	10
标注为负样本	0	990	990

$$\text{Accuracy} = \frac{0+990}{0+10+0+990} = 0.99$$

$$\text{Precision} = \frac{0}{0+0} = 0$$

$$\text{Recall} = \frac{0}{0+10} = 0$$

1. 数据不平衡，正负比例为1:99
2. 分类器将**所有的**样本预测为负例！

	预测为正样本	预测为负样本	
标注为正样本	490	5	495
标注为负样本	5	500	505

$$\text{Accuracy} = \frac{490+500}{490+5+5+500} = 0.99$$

$$\text{Precision} = \frac{490}{490+5} = 0.9899$$

$$\text{Recall} = \frac{490}{490+5} = 0.9899$$

Precision/Recall/F1评价对目标类别（正例）的分类效果

第3.1.4节 文本分类方法

文本分析代码实现

自然语言分析工具

- Jieba: 中文分词
- NLTK: 处理自然语言数据的Python程序和平台
- Gensim: 可扩展的统计语义、分析纯文本文档的语义结构、检索语义相似的文档
- TextBlob: 用于处理文本数据的Python库
- MBSP for Python: 基于TiMBL和MBT内存的学习应用程序
- xTAS: 基于Celery的分布式文本分析套件
- Pattern: Python编程语言的Web挖掘模块

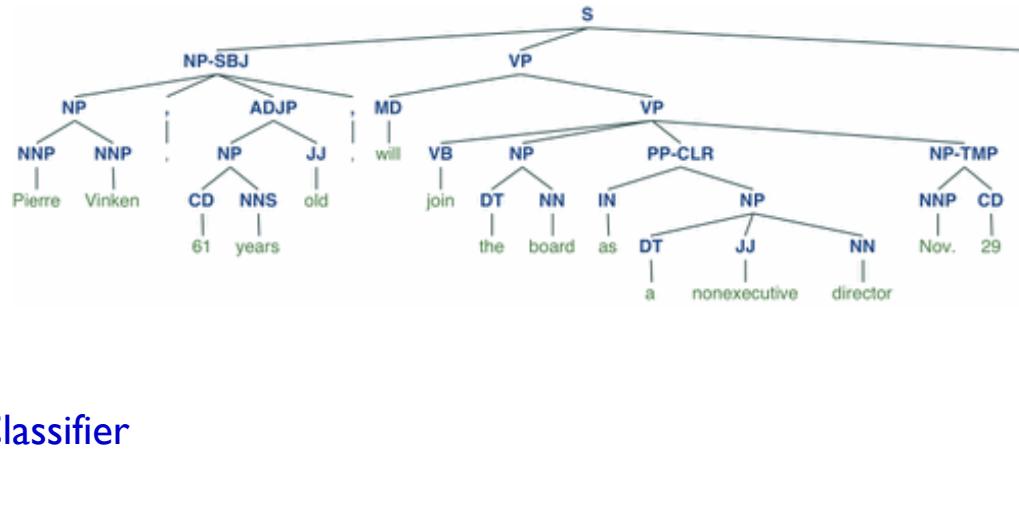
<https://www.jiqizhixin.com/articles/2018-10-29-26>

Natural Language Toolkit (NLTK)

- <http://www.nltk.org/index.html>
<https://github.com/nltk/nltk>
- 安装
 - NLTK requires Python versions 2.7, 3.4, 3.5, 3.6, or 3.7
 - 1.安装NLTK：执行sudo pip install -U nltk
 - 2.安装Numpy（可选）：执行sudo pip install -U numpy
 - 3.安装测试：执行python，进入python环境后再执行import nltk
- 数据下载（可选）
 - NLTK附带了许多语料库，toy grammar以及训练模型等。完整的列表发布在：http://nltk.org/nltk_data/
 - 要安装这些数据，首先安装NLTK，再利用NLTK的数据下载器进行下载

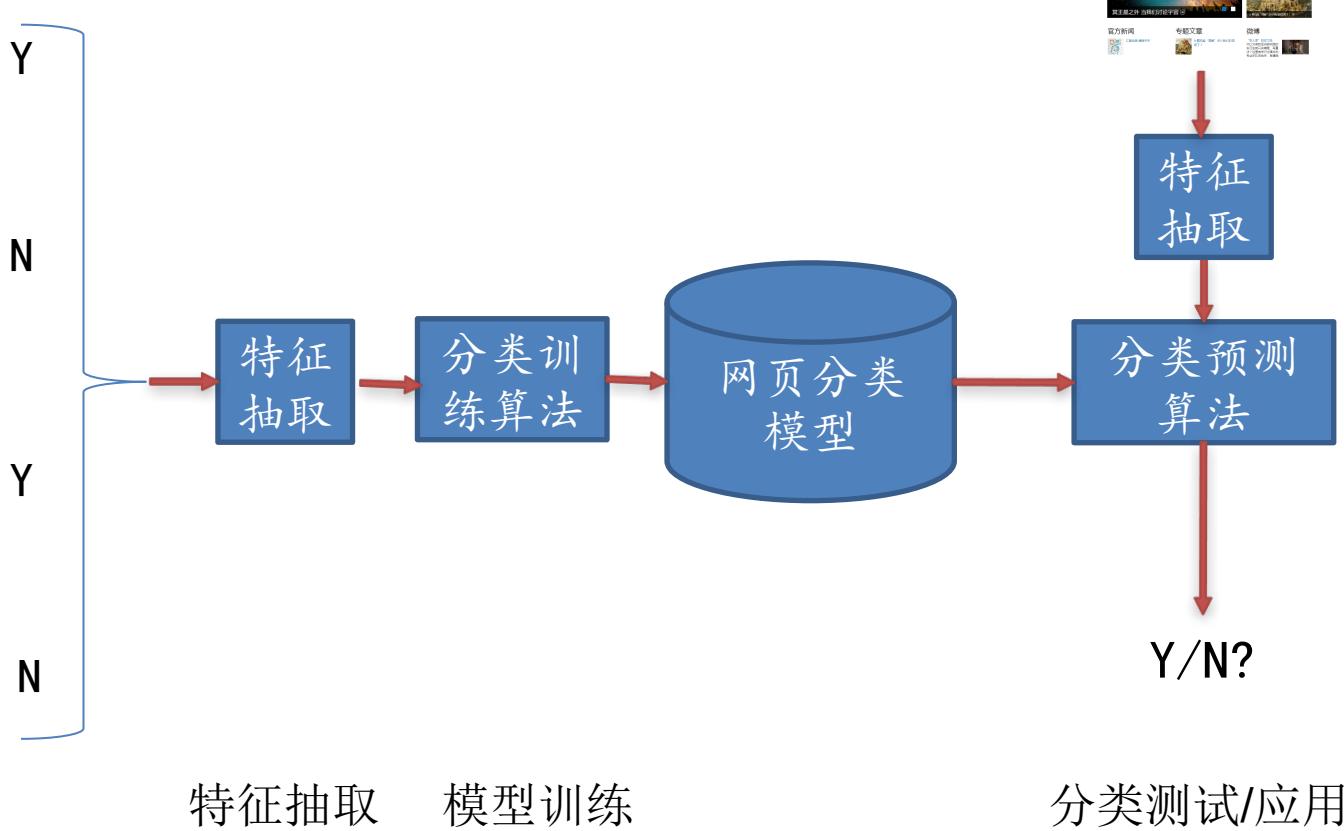
Naïve Bayes仅是NLTK的一部分

- 自然语言处理
 - Tokenization and POS tagging
 - Identify named entities
 - parse tree
 - Stemming
 - 情感分类
 - 语言模型
- 机器翻译
- 分类
 - ConditionalExponentialClassifier
 - DecisionTreeClassifier
 - MaxentClassifier
 - NaiveBayesClassifier
- 聚类
 - EM
 - kMeans



监督学习模型训练使用流程

训练样本 人工标注



- 训练和测试数据集合分别都被表示为list
- List中每一个元素为一个pair：第一个为特征，第二个为分类标签
- 每一个特征（一个文档）都被表示成一个dict
- 训练模型：
 - `classifier = NaiveBayesClassifier.train(train)`
- 测试模型：
 - `labels = classifier.classify_many(test)`
- 得到分类的概率：
 - `probs = classifier.prob_classify_many(test)`
- 查看每一维特征的作用：
 - `classifier.show_most_informative_features()`

文本二值分类

```
train_corpus= [('The team dominated the game', True),
 ('The game was intense', True),
 ('The ball went off the court', True),
 ('They had the ball for the whole game', True),
 ('The President did not comment', False),
 ('The show is over', False),
]

#build features
v = voc(train_corpus)
#train
train_feature_label = feature(train_corpus, v)
NBC = NaiveBayesClassifier.train(train_feature_label)
#test
test_corpus = [('I lost the keys', False),
 ('The goalkeeper caughted the ball', True),
 ('The other team controlled the ball', True),
 ('Sara has two kids', False),
 ('this is a book', True),
]
test_feature = []
test_labels=[]
for (ftr, label) in feature(test_corpus, v):
    test_feature.append(ftr)
    test_labels.append(label)

pred_labels = NBC.classify_many(test_feature)
print(test_labels)
print(pred_labels)

print('Precision = %.4f, Recall = %.4f, F-score = %.4f, Accuracy = %.4f'%
      classify_eval(test_labels, pred_labels))

# show probabilities
probs = NBC.prob_classify_many(test_feature)
for pdist in probs:
    print('%.4f %.4f' % (pdist.prob(True), pdist.prob(False)))
```

```
from nltk.classify import NaiveBayesClassifier

def voc(data):
    voc = {}
    for (sentence, val) in data:
        words = sentence.lower().split()
        for w in words:
            voc[w] = True
    return voc

def feature(data, v):
    ftr = []
    for (sentence, label) in data:
        f = dict((w, 0) for w in dict.keys(v))
        words = sentence.lower().split()
        for w in words:
            f[w] = 1
        ftr.append((f, label))
    return ftr
```

特征（字典）构建 和 特征抽取

结果评价

```
def classify_eval(truth, pred):
    idx = 0
    (TP, FP, TN, FN) = (0, 0, 0, 0)
    for truth_label in truth:
        pred_label = pred[idx]
        if (truth_label == True and pred_label == True):
            TP = TP + 1
        elif (truth_label == False and pred_label == False):
            TN = TN + 1
        elif (truth_label == True and pred_label == False):
            FN = FN + 1
        elif (truth_label == False and pred_label == True):
            FP = FP + 1
        idx = idx + 1
    P = 0 if TP == 0 else TP / (TP + FP)
    R = 0 if TP == 0 else TP / (TP + FN)
    F = 0 if (P == 0 or R == 0) else 2 * P * R / (P + R)
    Acc = 0 if (TP + TN == 0) else (TP + TN) / (TP + TN + FP + FN)
    return (P, R, F, Acc)
```

```
[False, True, True, False, True]
[True, True, True, True, False]
Precision = 0.5000, Recall = 0.6667, F-score = 0.5714, Accuracy = 0.4000
0.7776 0.2224
0.9459 0.0541
0.9740 0.0260
0.6602 0.3398
0.1775 0.8225
Most Informative Features
    game = 0           False : True   = 2.8 : 1.0
    comment = 0         True : False  = 1.8 : 1.0
    not = 0             True : False  = 1.8 : 1.0
    show = 0             True : False  = 1.8 : 1.0
    over = 0             True : False  = 1.8 : 1.0
    president = 0        True : False  = 1.8 : 1.0
    did = 0               True : False  = 1.8 : 1.0
    is = 0                 True : False  = 1.8 : 1.0
    ball = 0               False : True  = 1.7 : 1.0
    whole = 0              False : True  = 1.2 : 1.0
```

哪些单词对分类最有意义？

NBC.show_most_informative_features(10)

```
[False, True, True, False, True]
[True, True, True, True, False]
Precision = 0.5000, Recall = 0.6667, F-score = 0.5714, Accuracy = 0.4000
0.7776 0.2224
0.9459 0.0541
0.9740 0.0260
0.6602 0.3398
0.1775 0.8225
Most Informative Features
    game = 0          False : True   =
    comment = 0        True : False  =
    not = 0           True : False  =
    show = 0           True : False  =
    over = 0           True : False  =
    president = 0      True : False  =
    did = 0            True : False  =
    is = 0             True : False  =
    ball = 0           False : True  =
    whole = 0          False : True  =
```

使用SVM进行文本分类

```
from sklearn.svm import LinearSVC
```

```
SVMC = nltk.classify.SklearnClassifier(LinearSVC())
SVMC.train(train_feature_label)
pred_labels_SVM = []
for f in test_feature:
    pred_labels_SVM.append(SVMC.classify(f))

print(pred_labels_SVM)
print('Precision = %.4f, Recall = %.4f, F-score = %.4f, Accuracy = %.4f' %
      classify_eval(test_labels, pred_labels_SVM))
```