第15讲 支持向量机及排序机器学习

SVM & Learning to Rank

*节选自王斌老师《现代信息检索》课程的课件

提纲

1. 上一讲回顾
2. 支持向量机
3. 文本分类中的问题
4. 基于布尔权重的学习
5. 基于实数权重的学习
6. 基于序回归的排序学习

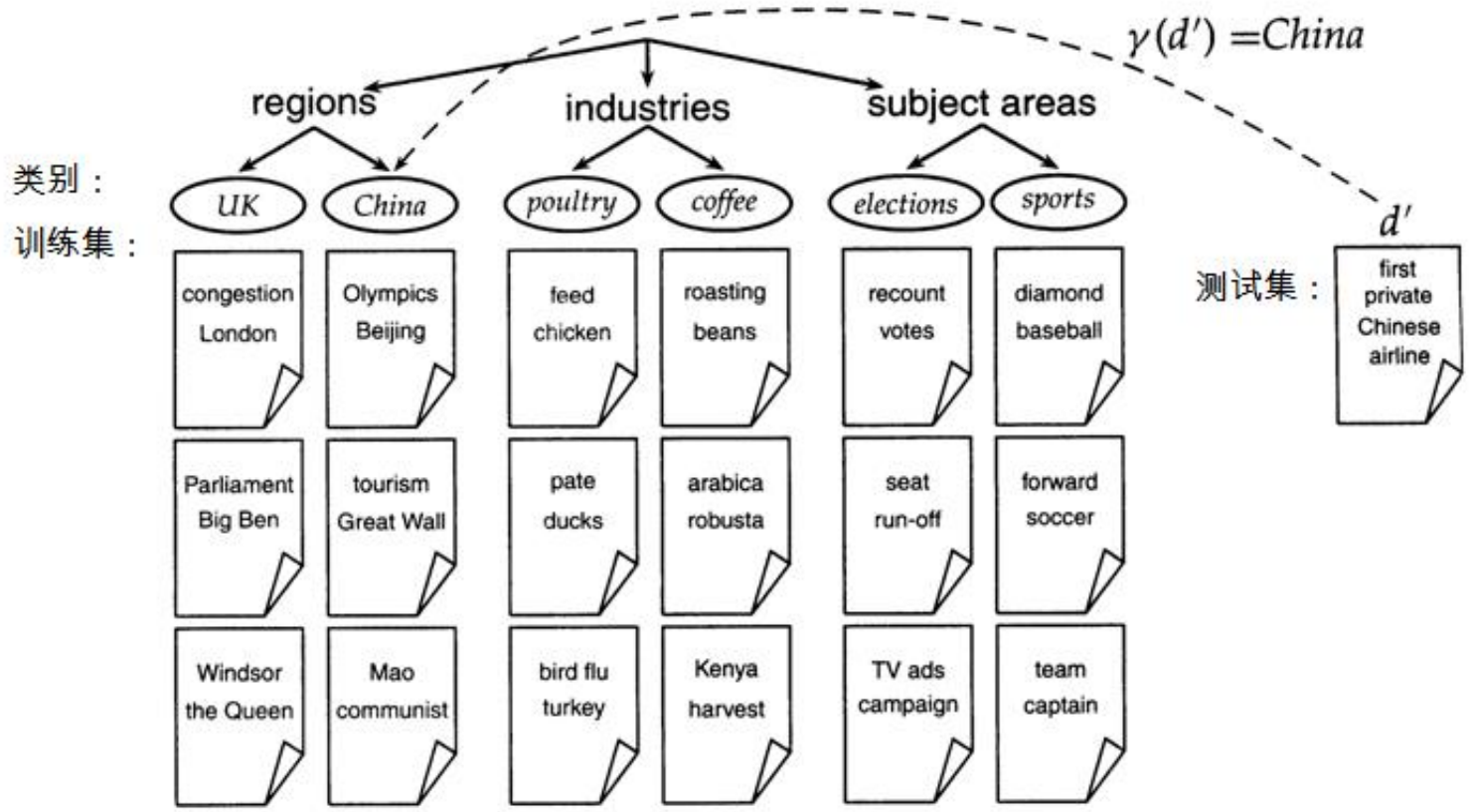
提纲

1. 上一讲回顾
2. 支持向量机
3. 文本分类中的问题
4. 基于布尔权重的学习
5. 基于实数权重的学习
6. 基于序回归的排序学习

上一讲内容

- 文本分类的概念及其与IR的关系
- 朴素贝叶斯分类器(朴素贝叶斯)
- 文本分类的评价
- 特征选择

文本(主题)分类


$\gamma(d') = China$
regions
industries
subject areas
类别：
训练集：
UK
China
poultry
coffee
elections
sports
congestion London
Olympics Beijing
feed chicken
roasting beans
recount votes
diamond baseball
Parliament Big Ben
tourism Great Wall
pate ducks
arabica robusta
seat run-off
forward soccer
Windsor the Queen
Mao communist
bird flu turkey
Kenya harvest
TV ads campaign
team captain
d'
测试集：
first private Chinese airline

朴素贝叶斯分类器

- 朴素贝叶斯是一个概率分类器
- 文档 d 属于类别 c 的概率计算如下：

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

- n_d 是文档的长度(词条的个数)
- $P(t_k|c)$ 是词项t_k 出现在类别c中文档的概率
- $P(t_k|c)$ 度量的是当c是正确类别时t_k 的贡献
- $P(c)$ 是类别c的先验概率
- 如果文档的词项无法提供属于哪个类别的信息，那么我们直接选择$P(c)$最高的那个类别

具有最大后验概率的类别

- 朴素贝叶斯分类的目标是寻找“最佳”的类别
- 最佳类别史具有最大后验概率(maximum a posteriori -MAP)的类别 c_{map}:

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

参数估计 1: 极大似然估计

- 如何从训练数据中估计 $\hat{P}(c)$ 和$\hat{P}(t_k|c)$ ？
- 先验:

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

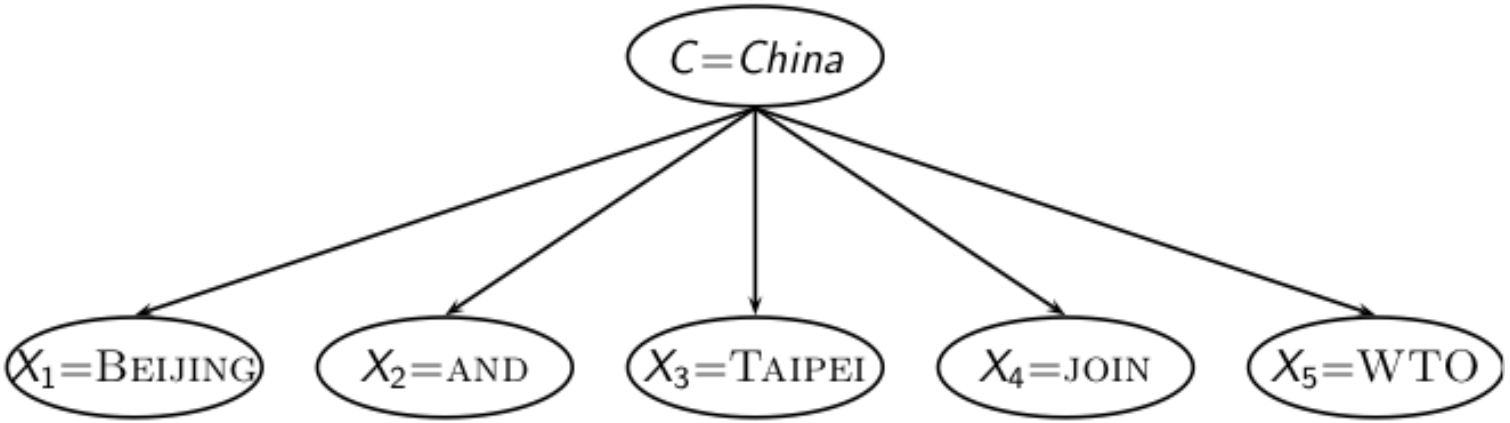
- N_c : 类c中的文档数目; N: 所有文档的总数
- 条件概率:

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

- T_{ct} 是训练集中类别c中的词条t的个数 (多次出现要计算多次)
- 给定如下的 朴素贝叶斯 独立性假设(independence assumption):

$$\hat{P}(t_{k_1}|c) = \hat{P}(t_{k_2}|c)$$

MLE估计中的问题：零概率问题



$$P(China|d) \propto P(China) \cdot P(\texttt{BEIJING}|China) \cdot P(\texttt{AND}|China) \cdot P(\texttt{TAIPEI}|China) \cdot P(\texttt{JOIN}|China) \cdot P(\text{WTO}|China)$$

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

MLE估计中的问题：零概率问题（续）

- 如果 WTO 在训练集中没有出现在类别 China中，那么就会有如下的零概率估计：

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

- → 那么，对于任意包含WTO的文档，P(China|d) = 0。
- 一旦发生零概率，将无法判断类别

避免零概率: 加一平滑

- 平滑前：

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

- 平滑后: 对每个量都加上1

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

- B 是不同的词语个数 (这种情况下词汇表大小 $|V|$ = B)

避免零概率: 加一平滑（续）

- 利用加1平滑从训练集中估计参数
- 对于新文档，对于每个类别，计算
  - (i) 先验的对数值之和以及
  - (ii) 词项条件概率的对数之和
- 将文档归于得分最高的那个类

分类评价

- 评价必须基于测试数据进行，而且该测试数据是与训练数据完全独立的 (通常两者样本之间无交集)
- 很容易通过训练可以在训练集上达到很高的性能 (比如记忆所有的测试集合)
- 指标: 正确率、召回率、 F_1值、分类精确率(classification accuracy)等等

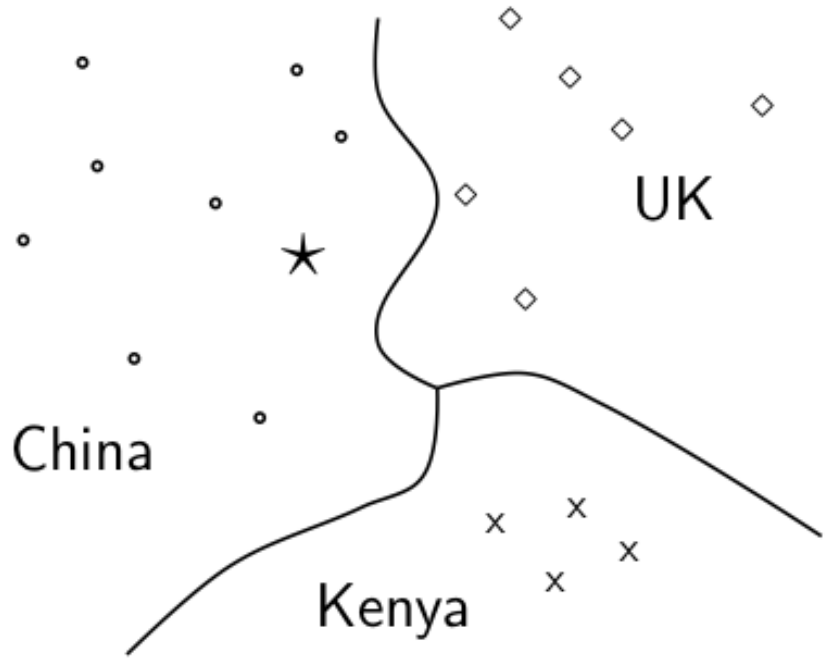
微平均 vs. 宏平均

- 对于一个类我们得到评价指标F_1
- 但是我们希望得到在所有类别上的综合性能
- 宏平均(Macroaveraging)
  - 对类别集合C 中的每个类都计算一个F_1 值
  - 对C个结果求平均Average these C numbers
- 微平均(Microaveraging)
  - 对类别集合C 中的每个类都计算TP、FP和FN
  - 将C中的这些数字累加
  - 基于累加的TP, FP, FN计算P、R和F_1

特征选择

- 文本分类中，通常要将文本表示在一个高维空间下，每一维对应一个词项
- 本讲义中，我们不特意区分不同的概念: 每个坐标轴 = 维 = 词语 = 词项 = 特征
- 许多维上对应是罕见词
- 罕见词可能会误导分类器
- 这些会误导分类器的罕见词被称为噪音特征（noise feature）
- 去掉这些噪音特征会同时提高文本分类的效率和效果
- 上述过程称为特征选择（feature selection）

向量空间中的类别



- 文档*到底是属于UK、China还是Kenya类？首先找到上述类别之间的分类面，然后确定文档所属类别，很显然按照图中分类面，文档应该属于China类
- 如何找到分类面并将文档判定给正确类别是本讲的重点。

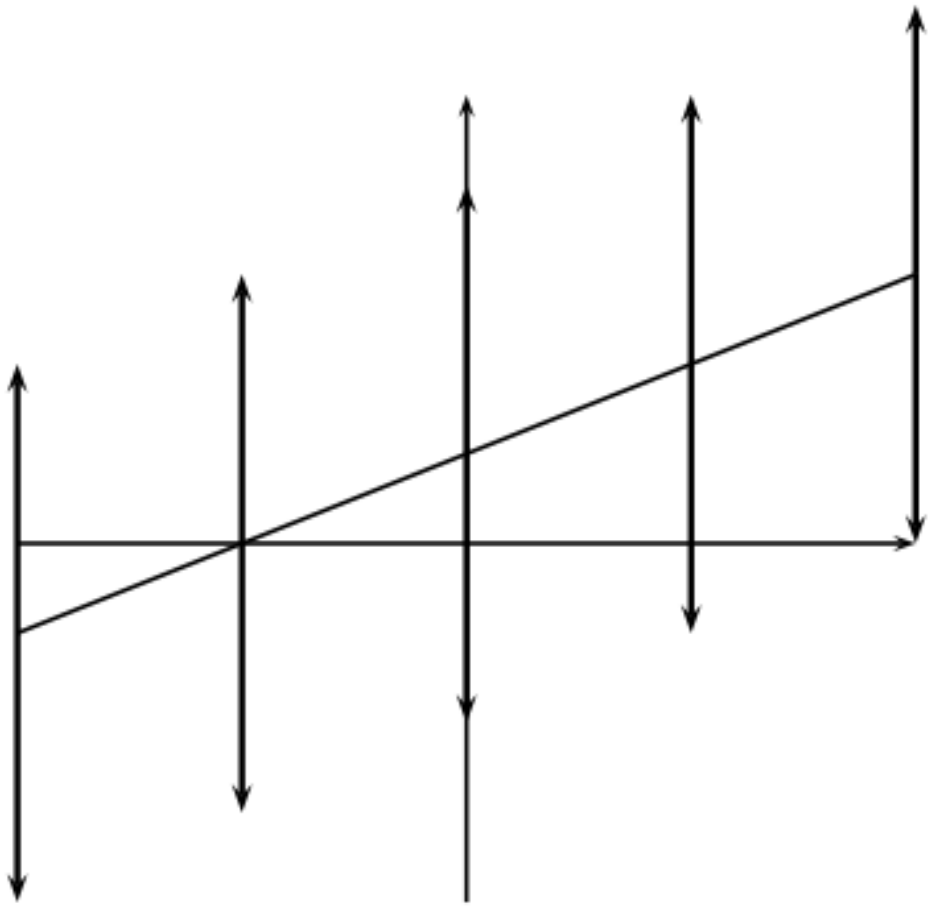
线性分类器

- 定义：
  - 线性分类器计算特征值的一个线性加权和 $\sum_i w_i x_i$
  - 决策规则：$\sum_i w_i x_i > \theta ?$
  - 其中，θ 是一个参数
- 首先，我们仅考虑二元分类器
- 从几何上说，二元分类器相当于二维平面上的一条直线、三维空间中的一个平面或者更高维下的超平面，称为分类面
- 基于训练集来寻找该分类面
- 寻找分类面的方法：感知机(Perceptron)、 Rocchio, Na ïve Bayes – 我们将解释为什么后两种方法也是二元分类器
- 假设：分类是线性可分的

一维下的线性分类器

- 一维下的分类器是方程$w_1 d_1 = \theta$对应的点
- 点的位置是θ / w_1
- 那些满足$w_1 d_1 \geq \theta$ 的点d_1属于类别c
- 而那些$w_1 d_1 < \theta$的点d_1属于类别 $\bar{c}$.

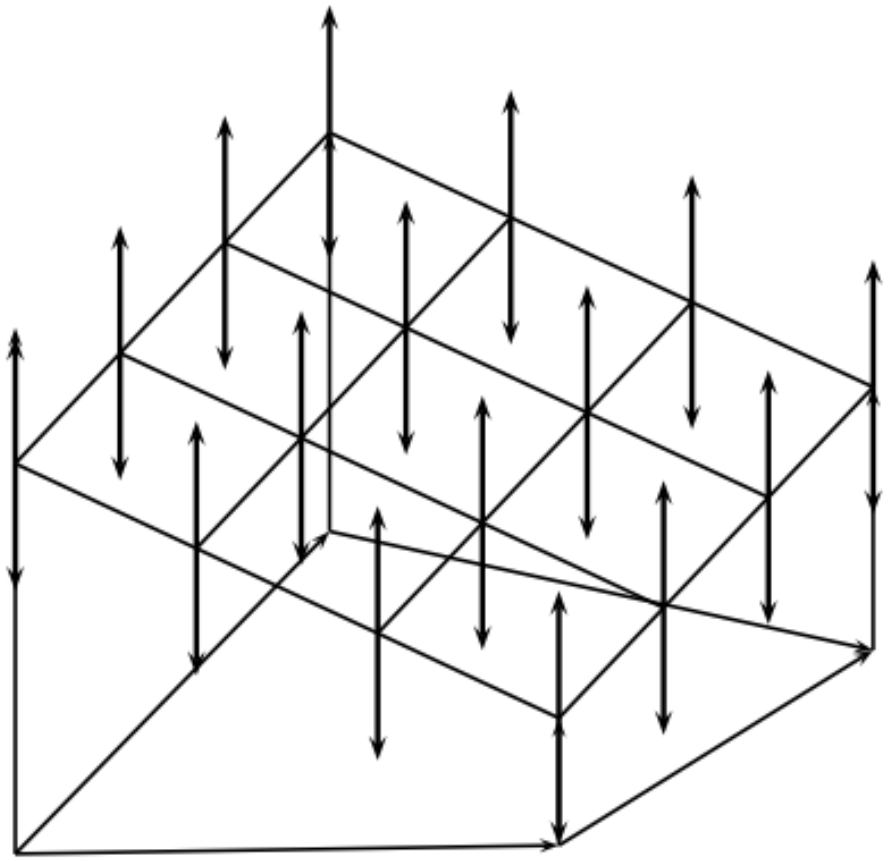
二维平面下的线性分类器



- 二维下的分类器是方程$w_1d_1 + w_2d_2 = \theta$对应的直线
- 那些满足$w_1d_1 + w_2d_2 \geq \theta$的点$(d_1\ d_2)$属于类别$c$
- 那些满足$w_1d_1 + w_2d_2 < \theta$的点$(d_1\ d_2)$属于类别

$\overline{c}$.

三维空间下的线性分类器



- 三维空间下的分类器是方程$w_1d_1 + w_2d_2 + w_3d_3 = \theta$对应的平面
- 那些满足$w_1d_1 + w_2d_2 + w_3d_3 \geq \theta$ 的点$(d_1\, d_2\, d_3)$ 属于类别c
- 那些满足$w_1d_1 + w_2d_2 + w_3d_3 < \theta$ 的点$(d_1\, d_2\, d_3)$ 属于类别 $\overline{c}$.

朴素贝叶斯是线性分类器

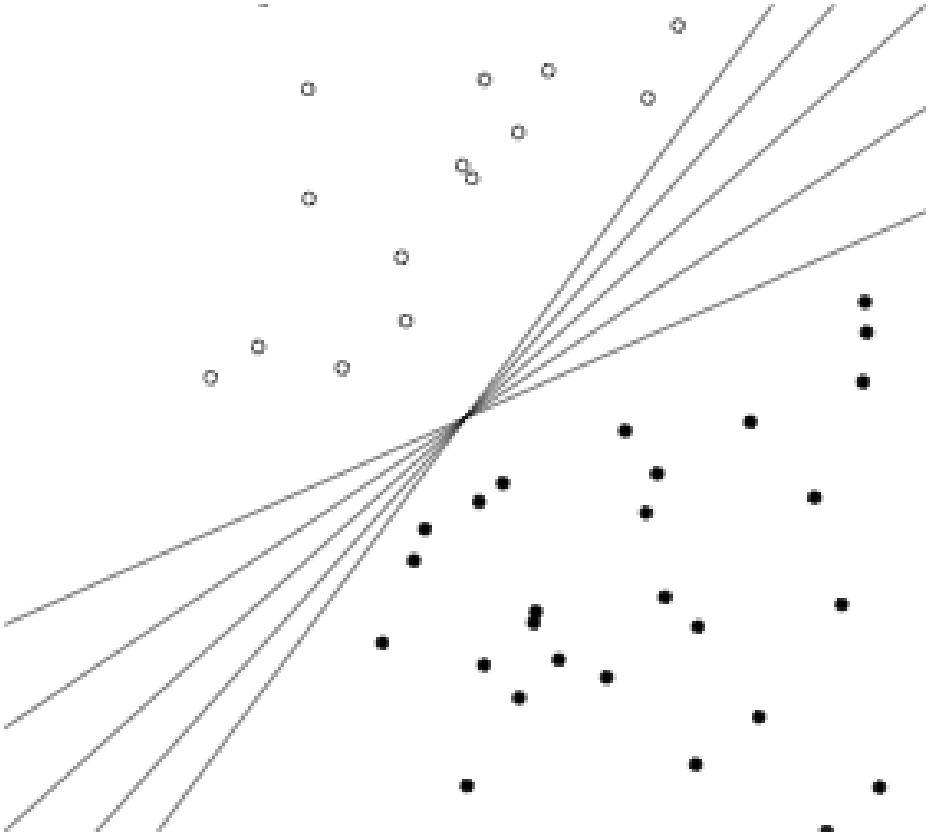
多项式模型的朴素贝叶斯也是线性分类器，其分类面定义为：

$$\sum_{i=1}^{M} w_i d_i = \theta$$

其中 $w_i = \log[\hat{P}(t_i|c)/\hat{P}(t_i|\bar{c})]$, $d_i = t_i$ 在d中的出现次数, $1 \le i \le M$, $\theta = -\log[\hat{P}(c)/\hat{P}(\bar{c})]$

(注意：这里的t_i指的是所有词汇表中的词项，而不是上一讲中出现在文档d中的词项)，

应该选哪个超平面?



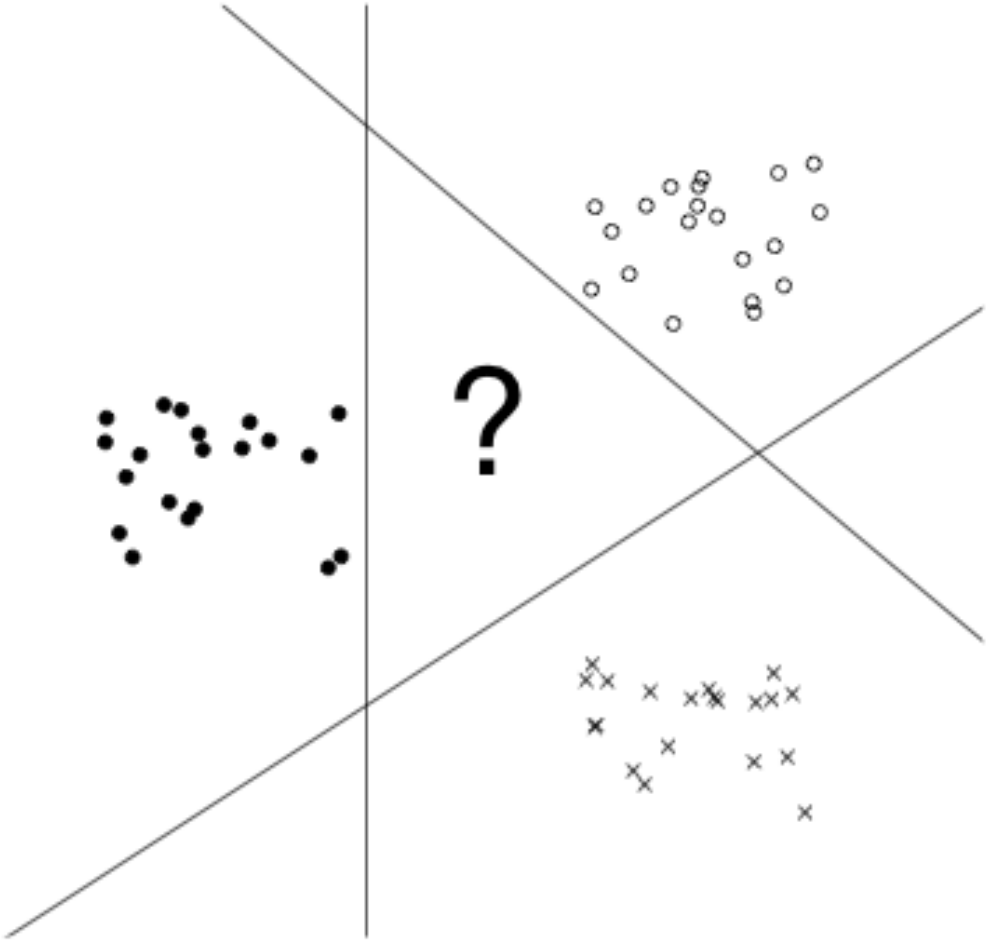
超平面的选择

- 对于线性可分的训练集而言，肯定存在无穷多个分类面可以将两类完全正确地分开
- 但是不同的分类面在测试集的表现完全迥异. . .
- 对于新数据，有些分类器的错误率很高，有一些却很低
- 感知机：通常很差；朴素贝叶斯、Rocchio：一般；线性SVM：好

给定文本分类问题下的分类器选择

- 是否存在某个学习方法对于任何分类问题都最优？
- 答案显然是否，这是因为存在着偏差(bias)和方差(variance)之间的折中
- 需要考虑的因素:
  - 有多少训练数据？
  - 问题的复杂性如何？ (分类面是线性还是非线性？)
  - 问题的噪音有多大？
  - 随时间推移问题的稳定性如何？
    - 对于一个不稳定的问题，最好使用一个简单鲁棒的分类器

多类(>2)的情况下的超平面组合



单标签问题（One-of problem）

- 单标签分类问题，也称single label problem
  - 类别之间互斥
  - 每篇文档属于且仅属于某一个类
  - 例子：文档的语言类型 (假定：任何一篇文档都只包含一种语言)

基于线性分类器的单标签分类

- 对于单标签分类问题（比如A、B、C三类），可以将多个二类线性分类器(A vsBC、B vs. AC、C vs. AB)进行如下组合：
  - 对于输入文档，独立运行每个分类器
  - 将分类器排序 (比如按照每个分类器输出的在A、B、C上的得分)
  - 选择具有最高得分的类别

多标签问题(Any-of problem)

- 多标签分类问题，也称multilabel classification
  - 一篇文档可以属于0、1或更多个类
  - 针对某个类的决策并不影响其他类别上的决策
  - 一种 “独立”类型，但是不是统计意义上的“独立”
  - 例子：主题分类
  - 通常：地区、主题领域、工业等类别之上的决策是互相独立的

基于线性分类器的多标签分类

- 对于多标签分类问题（比如A、B、C三类），可以将多个二类线性分类器(A vsBC、B vs. AC、C vs. AB)进行如下组合：
  - 对测试文档独立地运行每个分类器
  - 按照每个分类器自己的输出结果进行独立决策

本讲内容

- 支持向量机
  - 线性可分：最大间隔
  - 非线性可分：最大间隔+最小错误
  - 空间转换：核函数及核技巧

- 排序机器学习
  - 基于基于布尔权重的学习
  - 基于基于实数权重的学习
  - 基于序回归的排序学习

提纲

1. 上一讲回顾
2. 支持向量机
3. 文本分类中的问题
4. 基于布尔权重的学习
5. 基于实数权重的学习
6. 基于序回归的排序学习

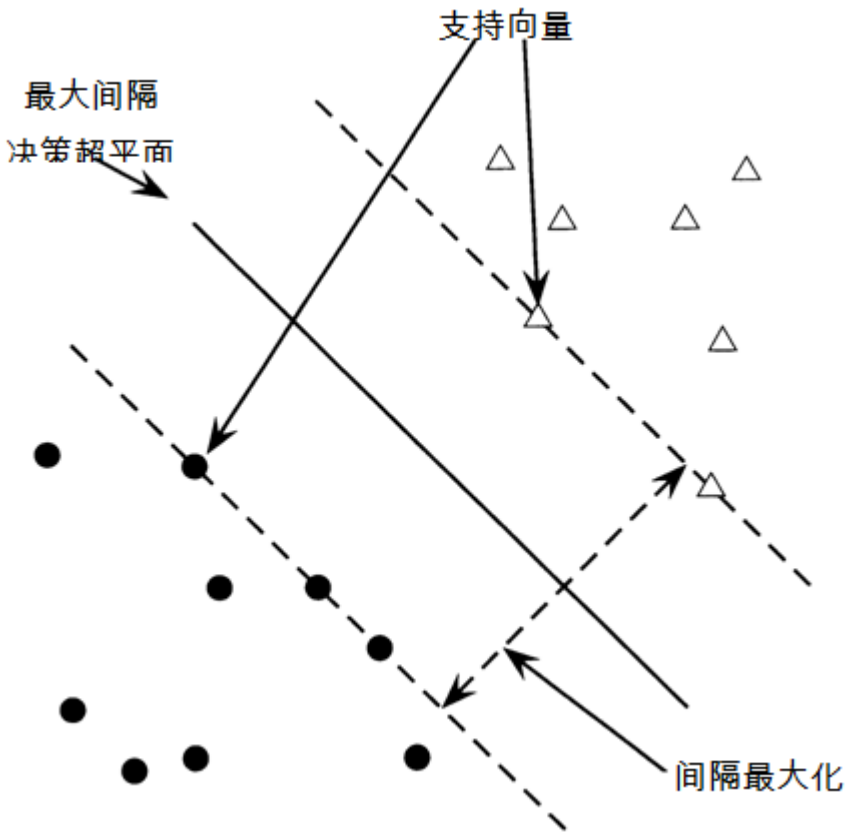
支持向量机

- 2-类训练数据
- 决策面

  → 线性分类面
- 准则:离任何数据点最远

  → 确定分类器

  间隔(margin)
- 线性分类面的位置基于支持向量(support vector)来定义
- 除支持向量外的其他数据点对于最终分界面的确定并不起作用。

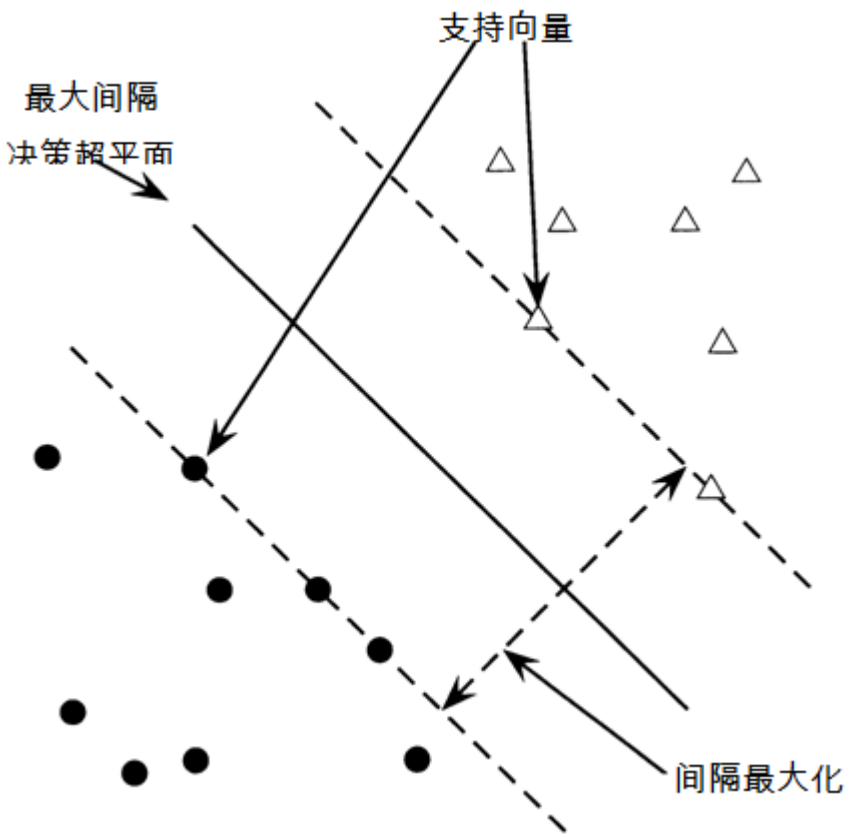


为什么要使间隔最大化？

分界面附近的点代表了不确定的分类决策，分类器会以两边各50%的概率做出决策

具有很大分类间隔的分类器不会做出确定性很低的决策，它给出了一个分类的安全间隔

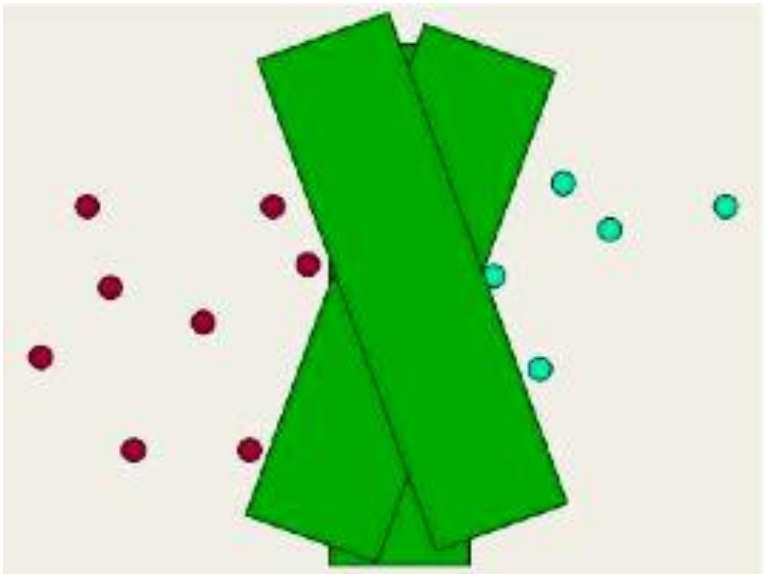
度量中的微小错误和文档中的轻微变化不会导致错误分类



为什么要使间隔最大化？

SVM 分类器：在决策面周围有大的间隔

- 与放置(无穷的)决策超平面相比，如果要在类别间放置一个宽间隔，那么选择会少很多
- 减少记忆容量
- 增加测试文档分类泛化能力



SVM的形式化

超平面(Hyperplane)

n维超平面(n=1时对应点，n=2时对应直线，n=3时对应普通平面)

决策超平面

定义为:

- 截距 b
- 法向量 $\vec{w}$(**权重向量**)，它与超平面正交

所有超平面上的点 $\vec{x}$ 满足:

(1) $$\vec{w}^{\mathsf{T}}\vec{x} = -b$$

SVM形式化

预备知识

考虑一个二类分类问题：

- $\vec{x}_i$ 是输入向量
- y_i 是相应的标签

$\overrightarrow{x_i}$ 代表的是一系列带标签点，称为输入空间

对于SVM，两类的标签分别用+1 和 −1来标记，截距用b来代表

线性分类器定义为：

(2)

$$f(\vec{x}) = \text{sign}(\vec{w}^{\mathsf{T}}\vec{x} + b)$$

采用符号函数，结果为−1 表示属于一个类，为+1 表示属于另一个类

函数间隔(Functional Margin)

如果某个点远离决策分类面，那么可以确信该点的类别归属

函数间隔

第i个样例 $\vec{x_i}$ 到超平面$<\vec{w},b>$的距离为：

$$y_i(\vec{w}^T\vec{x_i}+b)$$

对于决策平面来说，某个数据集的函数间隔是数据集点中最小函数间隔的2倍

2倍这个因子来自对整个间隔宽度的计算

但是，我们可以通过放大$\vec{w}$和b来获得我们所需要的任意函数间隔，因此，我们需要对向量 $\vec{w}$的大小加以限制

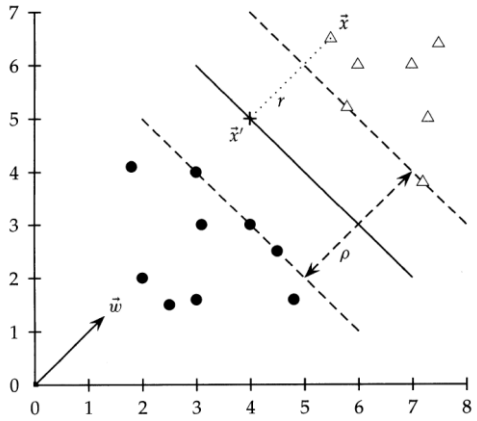


图 15-3　点到决策边界（ρ）的几何间隔（r）

一个点 $\vec{x}$ 到决策边界的欧氏距离是多少？在图 15-3 中，我们把这个距离记为 r。我们知道，点到超平面的最短距离垂直于该平面，也就是说和 $\vec{w}$ 平行。这个方向的单位向量是 $\vec{w}/|\vec{w}|$。图中的点线也就是向量 $r\vec{w}/|\vec{w}|$的平移结果。将超平面上离 $\vec{x}$ 最近的点标记为 $\vec{x}'$，于是有：

$$\vec{x}' = \vec{x} - yr\frac{\vec{w}}{|\vec{w}|'}。 \tag{15-2}$$

其中，乘以 y 的结果将改变在决策面上下的点 $\vec{x}$ 所对应的符号。另外，由于 $\vec{x}'$ 在决策边界上，因此有 $\vec{w}^{\mathrm{T}}\vec{x}' + b = 0$，于是：

$$\vec{w}^{\mathrm{T}}\left(\vec{x} - yr\frac{\vec{w}}{|\vec{w}|}\right) + b = 0。 \tag{15-3}$$

对r求解得[①]：

$$r = y\frac{\vec{w}^{\mathrm{T}}\vec{x} + b}{|\vec{w}|}。 \tag{15-4}$$

几何间隔(Geometric margin)

分类器的几何间隔: 中间空白带的最大宽度，该空白带可以用于将两类支持向量分开

$$r = y\frac{\vec{w}^{\mathsf{T}}\vec{x} + b}{|\vec{w}|} \tag{3}$$

很显然，不管参数$\vec{w}$和b如何缩放，几何间隔总是一个不变量。比如，即使将$\vec{w}$和b分别替换成$5\vec{w}$和$5b$，其几何间隔仍然保持不变，这是因为它已经基于w的长度做了归一化处理。

线性SVM的数学推导

规范化距离：
假定所有数据到分类面的距离至少是1

$$y_i(\vec{w}^{\mathsf{T}}\vec{x}_i + b) \geq 1 \qquad (4)$$

每个点到分类面的距离为 $r_i = y_i(\vec{w}^{\mathsf{T}}\vec{x}_i + b)/|\vec{w}|$ ，几何间隔为：

$$\rho = 2/|\vec{w}|$$

我们的目标就是最大化 $\rho = 2/|\vec{w}|$ ，也就是说在满足

$$(\vec{x}_i, y_i) \in \mathbb{D},\ y_i(\vec{w}^{\mathsf{T}}\vec{x}_i + b) \geq 1$$

的条件下，寻找$\vec{w}$和b，使得 $\rho = 2/|\vec{w}|$ 最大

线性 SVM的数学推导

最大化 $2/|\vec{w}|$ 相当于最小化$|\vec{w}|/2.$，于是可以将SVM求解过程转化为如下的一个求最小化的问题：

问题：

寻找 $\vec{w}$ 及b 使得：

$\frac{1}{2}\vec{w}^{\mathrm{T}}\vec{w}$ 最小 (因为 $|\vec{w}| = \sqrt{\vec{w}^{\mathrm{T}}\vec{w}}$)

且所有数据点满足 $\{(\vec{x}_i, y_i)\},\ y_i(\vec{w}^{\mathrm{T}}\vec{x}_i + b) \geq 1$

上述问题实际上是在线性约束条件下的二次函数优化问题。该问题是数学中的基本问题之一，存在很多解决算法 (比如，有一些二次规划库)

线性 SVM的结果形式

线性SVM的结果分类器为：

$$f(\vec{x}) = \text{sign}(\sum_i \alpha_i y_i \vec{x}_i \cdot \vec{x} + b)$$

其中上式中的求和只对支持向量求和，也就是说，对于新来的文档向量 ，与所有支持向量求内积之后加权求和，然后与阈值-b进行比较，最后根据结果的符号判断文档的类别。

SVM要点重述

给定训练数据集

- 该数据集定义了最佳的分类超平面
- 基于该数据集，通过二次优化过程寻找该超平面
- $f(\vec{x})$ ：对于待分类的新数据点 $\vec{x}$ ，分类函数计算该点到超平面法向量的投影
- 上述函数的符号决定了该数据点类别的归属

- 如果该点在分类器的间隔之内，分类器可以在原来的两个类之外，返回“类别未知”
- $f(\vec{x})$ 的值可以转化为一个分类概率

软间隔(Soft margin)分类

如果数据不线性可分？

- 标准做法：允许在宽间隔条件下犯少许错误
  - 某些点、离群点或者噪音点可以在间隔之内或者在间隔的错误一方
- 计算每个错误点的代价，具体的计算它们到分类面的距离。

松弛变量 ξ_i : 允许点 $\vec{x}_i$ 不满足间隔要求，但是其错误代价正比于ξ_i

优化问题：在间隔的宽度 和 那些需要在计算间隔时去掉的点数之间折中

ξ_i 的和给出了所有训练错误的上界

软间隔SVM主要在最小化训练错误和最大化间隔之间折中

支持软间隔的 SVM的数学推导

引入软间隔，可以将SVM求解过程转化为如下的一个求最小化的问题：

问题：

寻找 $\vec{w}$ 及 b 使得：

$\frac{1}{2}\vec{w}^{\mathrm{T}}\vec{w} + C\sum\xi_i$ 最小

且所有数据点满足 $\{(\vec{x}_i, y_i)\}, y_i(\vec{w}^T\vec{x}_i + b) + \xi_i \geq 1$

C是加权系数，上述问题仍然是在线性约束条件下的二次函数优化问题，同样可以求解。

非线性SVM

通过空间映射将原始空间映射到新空间，为避免显式的映射函数，引入核函数(定义在原始空间下但是结果是新空间下的内积函数)

常用核函数	
多项式核	$K(\vec{x},\vec{z})=(1+\vec{x}\cdot\vec{z})^d$
径向基核	$K(\vec{x},\vec{z})=e^{-(\vec{x}-\vec{z})/(2\sigma^2)}$

支持多类的支持向量机

SVMs: 是一个天生的二类分类器

- 实际中存在一些常用的技巧： 构造 |C|个一对多(one-versus-rest 或one-versus-all或OVA)的二类分类器， 选择那个在测试数据上具有最大间隔的分类器所给出的类别
- 另一种方法：建立一系列一对一（one-versus-one）的分类器，选择这些分类器中给出的最多的那个类别。虽然包含 $|C|(|C|-1)/2$ 个分类器的构建过程，但是分类器的训练时间可能实际上会降低，这是因为每个分类器训练的语料都小很多

SVM用于支持多类问题

更好的解决方法：结构化SVM

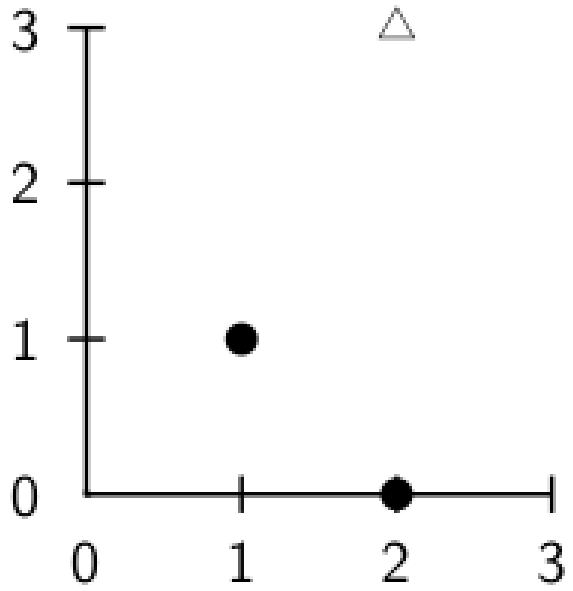
- 将分类问题一般化为如下问题：类别不再只是多个独立的类别标签集合，而可以是相互之间具有关系定义的任意结构化对象的集合。
- 在基于机器学习的排序中将进一步介绍该方法

一个SVM的例子

几何上看:

- 最大间隔权重向量将和两类中距离最短的那条线段(直线)平行，即与连接点(1, 1)和(2, 3)的直线平行，这可以得到权重向量 (1,2).
- 最优的分类直线与上述线段垂直并相交与其中点(中垂线)，因此它经过点 (1.5, 2).
- 于是，可以求得SVM的决策直线方程为：

$$y = x_1 + 2x_2 - 5.5$$

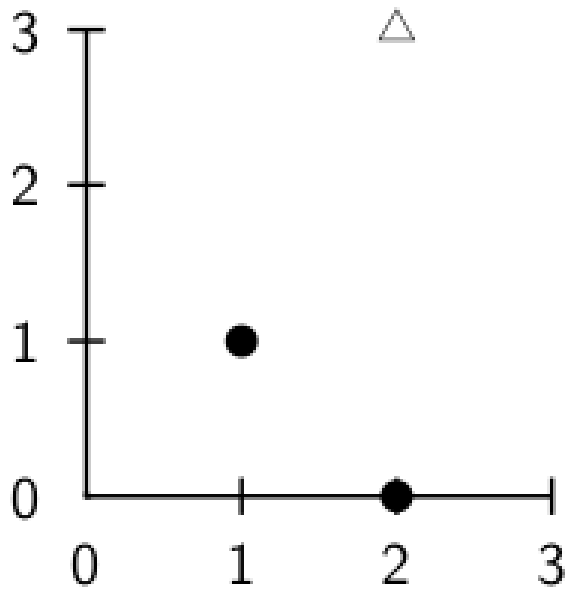


一个SVM的例子（续）

代数法求解:

- 在约束条件 $(y_i(\vec{w}^{\mathsf{T}}\vec{x}_i + b)) \geq 1$ 下，寻找最小的 $|\vec{w}|$.

- 我们知道解的形式为：
  $\vec{w} = (a, 2a)$ 于是有：
  $a + 2a + b = -1,\ 2a + 6a + b = 1$
- 解得， $a = 2/5$ 及 $b = -11/5$
  因此，最优超平面的参数为：
  $\vec{w} = (2/5, 4/5)$
  $b = -11/5.$
- 此时间隔ρ为：
  $2/|\vec{w}| = 2/\sqrt{4/25 + 16/25} =$
  $2/(2\sqrt{5}/5) = \sqrt{5}.$



提纲

1. 上一讲回顾
2. 支持向量机
3. 文本分类中的问题
4. 基于布尔权重的学习
5. 基于实数权重的学习
6. 基于序回归的排序学习

文本分类

许多商业应用

- “能够基于内容对文档进行自动分类的商业价值毋庸置疑，在公司内网、政府机构及互联网出版等机构或领域中存在大量的潜在应用”

采用领域相关的文本特征在性能上会比采用新的机器学习方法获得更大的提升

- “对数据的理解是分类成功的关键之一，然而这又是大部分分类工具供应商非常不擅长的领域。市场上很多所谓的通用分类工具并没有在不同类型的内容上进行广泛的测试。”

分类器的选择

当面对一个建立分类器的需求时，第一个要问的问题就是：训练数据有多少？

实际中的挑战: 建立或获取足够的训练语料

为了获得高性能的分类器，每个类都需要成百上千的训练文档，而且现实当中的类别体系也非常庞大

- 一点都没有?
- 很少?
- 挺多?
- 量很大，而且每天都在增长?

如果没有任何训练数据

采用人工撰写规则的方法

例子

IF (wheat OR grain) AND NOT (whole OR bread) THEN
c = grain

实际中的规则要比这个例子长很多，并且可以采用更复杂的表示方式。经过精心调整（也就是说，人们可以在开发集上调整规则）之后，利用这些规则分类的精度可以非常高。然而，要构造非常好的人工规则需要做大量的工作。一个基本合理的估计数字是每个类别需要两天的时间，由于类别中的文档内容会发生漂移，所以必须还要利用很多额外的时间去维护规则。

如果拥有较少的训练数据，又希望训练一个有监督的分类器

如何尽快地获得更多的标注数据

- 最佳方法：将自己也放入到标注的过程中去，在这个过程中人们自愿为你标注数据，并且这也是他们正常工作的一部分。

例子

很多情况下，人们会根据需要整理或者过滤邮件数据，这些动作能够提供类别相关的信息

主动学习(Active Learning)

建立一个系统来确定应该标注的那些文档。通常情况下，这些文档主要指那些分类器不确定能否正确分类的文档。

如果拥有训练数据

拥有适量的标注数据

能够使用我们在前面所介绍的任何文本分类技术

通常优先考虑混合方法

拥有极大规模的标注数据

分类器的选择也许对最后的结果没有什么影响，目前我们还不清楚是否有最佳的选择方法。也许最好的方法是基于训练的规模扩展性或运行效率来选择。为达到这个目的，需要极大规模的数据。一个通用的经验法则是，训练数据规模每增加一倍，那么分类器的效果将得到线性的提高。但是对于极大规模的数据来说，效果提高的幅度会降低成亚线性。

大规模高难度分类体系

如果文本分类问题仅包含少量具有区分度的类别，那么很多分类算法都可能取得很好的结果。但是实际的文本分类问题往往包含大量非常类似的类别。

例子

Web目录（如Yahoo!目录或ODP（Open Directory Project）目录）、图书馆分类机制（杜威十进制分类法或美国国会图书馆分类法），或者用于法律和医学领域的分类机制。

对大量相近的类别进行精确分类是一个固有的难题

提纲

1. 上一讲回顾
2. 支持向量机
3. 文本分类中的问题
4. 基于布尔权重的学习
5. 基于实数权重的学习
6. 基于序回归的排序学习

基本思路

- 词项权重(如tfidf)的目标是为了度量词项的重要性
  - 将一篇文档中所有词项的权重加起来便可以计算文档和查询的相关度，基于该相关度可以对所有文档排序

- 上述过程可以想象成一个文本分类问题
  - 词项权重可以从已判定的训练集合中学习得到

- 上述研究方法被归入一类称为机器学习的相关度(**machine learned relevance**)或排序学习(**learning to rank**)

权重学习

主要方法：

- 给定训练样例集合，每个样例表示为三元组<q, d, R(d,q)>
  - 最简单的情况： 相关性判定结果$R(d, q)$要么为1 (相关)，要么为0 （不相关）
  - 更复杂的情况：多级相关
- 从上述样例中学习权重，使得学到的评分接近训练集中的相关性判定结果。

- 下面以域加权评分(*Weighted zone scoring*)为例来介绍

域加权评分

- 给定查询以及具有3个域(author、title、body)的文档集合

- 域加权评分对每个域都有个独立的权重，比如 g_1, g_2, g_3
- 并非所有域的重要性都完全一样:

  比如： author ¡title ¡body

  $\rightarrow g_1 = 0.2, g_2 = 0.3, g_3 = 0.5$ (系数总和为1)
- 如果查询词项出现在某个域中，那么该域上的得分为1，否则为0 (布尔权重)

例子

查询词项仅仅在title和body域中出现
于是文档得分为： $(0.3 \cdot 1) + (0.5 \cdot 1) = 0.8.$

域加权评分的一般化

给定 q 和 d, 域加权评分方法通过计算所有文档域得分的线性组合，赋予(q,d)一个[0,1]内的得分

- 考虑一系列文档，每篇文档包含 l 个域
- 令 $g_1, ..., g_l \in [0, 1]$, 且有 $\sum_{i=1}^{l} g_i = 1$
- 对于 $1 \leq i \leq l$，令 s_i 为q和文档第i个域的 布尔匹配得
  - 比如， s_i可以是将域当中查询词项出现与否映射为0或1的任意布尔函数。

域加权评分也称为排序式布尔检索

$$\sum_{i=1}^{l} g_i s_i$$

域加权评分及权重学习

- 域加权评分可以看成基于布尔匹配值的线性函数学习，每个布尔匹配值对应一个域
- 坏消息：权重学习需要训练集，而人工产生训练集中的相关性判断费时费力
  - 特别是在动态语料库环境下 (比如Web)
- 好消息：将权重g_i的学习简化为一个简单的优化问题

域加权评分中的权重学习

- 最简单的情况：每篇文档有两个域---- title, body
- 域加权评分的公式如下：

$$\sum_{i=1}^{l} g_i s_i \qquad (2)$$

- 给定 q, d, 要根据d的不同域与查询q的匹配情况计算 $s_T(d, q)$ 及$s_B(d, q)$
- 利用 $s_T(d, q)$ 、 $s_B(d, q)$ 及常数 $g \in [0, 1]$我们可以得到(q,d)的得分：

$$score(d, q) = g \cdot s_T(d, q) + (1 - g) \cdot s_B(d, q) \qquad (3)$$

基于训练样例学习权重g

例子

Example	DocID	Query	s_T	s_B	Judgment
Φ_1	37	linux	1	1	Relevant
Φ_2	37	penguin	0	1	Nonrelevant
Φ_3	238	system	0	1	Relevant
Φ_4	238	penguin	0	0	Nonrelevant
Φ_5	1741	kernel	1	1	Relevant
Φ_6	2094	driver	0	1	Relevant
Φ_7	3194	driver	1	0	Nonrelevant

- 训练样例: 三元组形式 $\Phi_j = (d_j , q_j , r(d_j , q_j))$
- 给定训练文档 d_j 和训练查询qj，通过人工判定得到$r(d_j , q_j)$ (要么相关要么不相关)

基于训练样例学习权重g

样例

Example	DocID	Query	s_T	s_B	Judgment
Φ_1	37	linux	1	1	Relevant
Φ_2	37	penguin	0	1	Nonrelevant
Φ_3	238	system	0	1	Relevant
Φ_4	238	penguin	0	0	Nonrelevant
Φ_5	1741	kernel	1	1	Relevant
Φ_6	2094	driver	0	1	Relevant
Φ_7	3194	driver	1	0	Nonrelevant

- 对每个训练样例 Φ_j ，我们有布尔值 $s_T(d_j, q_j)$和$s_B(d_j, q_j)$，利用这些布尔值可以计算：

$$score(d_j, q_j) = g \cdot s_T(d_j, q_j) + (1 - g) \cdot s_B(d_j, q_j) \qquad (4)$$

权重学习

- 比较该得分和人工判定结果的差异
  - 人工判定结果中，相关记为1，不相关记为0
- 评分函数的错误定义为：

$$\epsilon(g,\Phi_j) = (r(d_j, q_j) - \text{score}(d_j, q_j))^2 \quad (5)$$

- 于是，整个训练集上的总错误为：

$$\sum_j \epsilon(g,\Phi_j) \quad (6)$$

- 权重g的学习归结为选择使得上述总错误率最小的那个g

课堂练习: 寻找使总错误ϵ 最小的g值

训练样例

Example	DocID	Query	s_T	s_B	Judgment
Φ_1	37	linux	1	1	Relevant
Φ_2	37	penguin	0	1	Nonrelevant
Φ_3	238	system	0	1	Relevant
Φ_4	238	penguin	0	0	Nonrelevant
Φ_5	1741	kernel	1	1	Relevant
Φ_6	2094	driver	0	1	Relevant
Φ_7	3194	driver	1	0	Nonrelevant

1. 相关记为 1，不相关记为0
2. 计算得分:

$$score(d_j, q_j) = g \cdot s_T(d_j, q_j) + (1 - g) \cdot s_B(d_j, q_j)$$

3. 计算总错误: $\sum_j \epsilon(g, \Phi_j)$，其中

$$\epsilon(g, \Phi_j) = (r(d_j, q_j) - \text{score}(d_j, q_j))^2$$

4. 选择使得总错误最小的g值

习题解答

1. 计算评分 score(d_j , q_j)

$score(d_1, q_1) = g \cdot 1 + (1-g) \cdot 1 = g + 1 - g = 1$

$score(d_2, q_2) = g \cdot 0 + (1-g) \cdot 1 = 0 + 1 - g = 1 - g$

$score(d_3, q_3) = g \cdot 0 + (1-g) \cdot 1 = 0 + 1 - g = 1 - g$

$score(d_4, q_4) = g \cdot 0 + (1-g) \cdot 0 = 0 + 0 = 0$

$score(d_5, q_5) = g \cdot 1 + (1-g) \cdot 1 = g + 1 - g = 1$

$score(d_6, q_6) = g \cdot 0 + (1-g) \cdot 1 = 0 + 1 - g = 1 - g$

$score(d_7, q_7) = g \cdot 1 + (1-g) \cdot 0 = g + 0 = g$

2. 计算总错误 $\sum_j \epsilon(g, \Phi_j)$

$(1-1)^2 + (0-1+g)^2 + (1-1+g)^2 + (0-0)^2 + (1-1)^2 +$

$(1-1+g)^2 + (0-g)^2 = 3g^2 + (1-g)^2 = 4g^2 - 2g + 1$

3. 选择使得总错误最小的g值

求解g=0.25

提纲

1. 上一讲回顾
2. 支持向量机
3. 文本分类中的问题
4. 基于布尔权重的学习
5. 基于实数权重的学习
6. 基于序回归的排序学习

一个简单的机器学习评分的例子

- 迄今为止，我们都是考虑一种非常简单的情况，即我们考虑的是布尔相关值的组合
- 现在考虑更一般的情况

一个简单的机器学习评分的例子

- 设置: 评分函数是两个因子的线性组合:
  1. 1 查询和文档的向量空间相似度评分 (记为 α)
  2. 2 查询词项在文档中存在的最小窗口宽度 (记为 ω)
     - 查询词项的邻近度(proximity)往往对文档的主题相关性具有很强的指示作用
     - 查询词项的邻近度给出了隐式短语的实现

- 因此，我们的一个因子取决于查询词项在文档中的词袋统计量，另一个因子取决于邻近度权重

一个简单的机器学习评分的例子

给定训练集，对每个样例计算

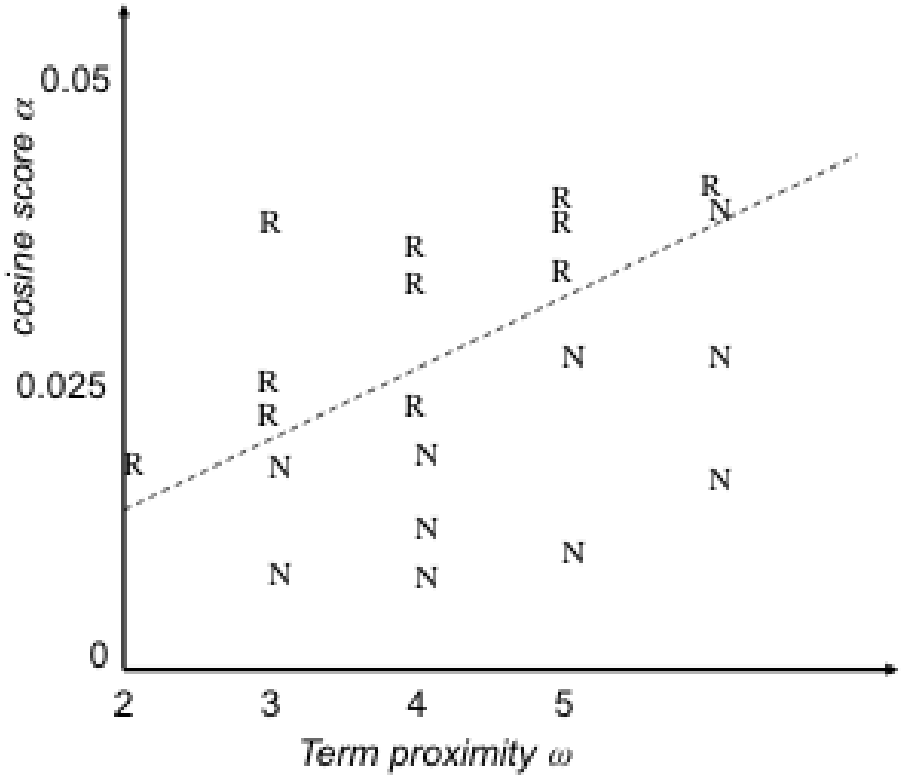
- 向量空间余弦相似度 α
- 窗口宽度 ω

上述结果构成训练集，与前面不同的是，我们引入的是两个实数特征因子 (α, ω)

例子

Example	DocID	Query	Cosine score	ω	Judgment
Φ_1	37	linux operating system	0.032	3	*relevant*
Φ_2	37	penguin logo	0.02	4	*nonrelevant*
Φ_3	238	operating system	0.043	2	*relevant*
Φ_4	238	runtime environment	0.004	2	*nonrelevant*
Φ_5	1741	kernel layer	0.022	3	*relevant*
Φ_6	2094	device driver	0.03	2	*relevant*
Φ_7	3191	device driver	0.027	5	*nonrelevant*
...	...	...	...	...	...

2-D 平面上的图展示



一个简单的机器学习评分的例子

- 同样，相关记为1，不相关记为0
- 我们的目标是寻找一个评分函数，该函数能够组合特征因子的值，并尽量接近0或1
- 希望该函数的结果尽量与训练集上的结果保持一致

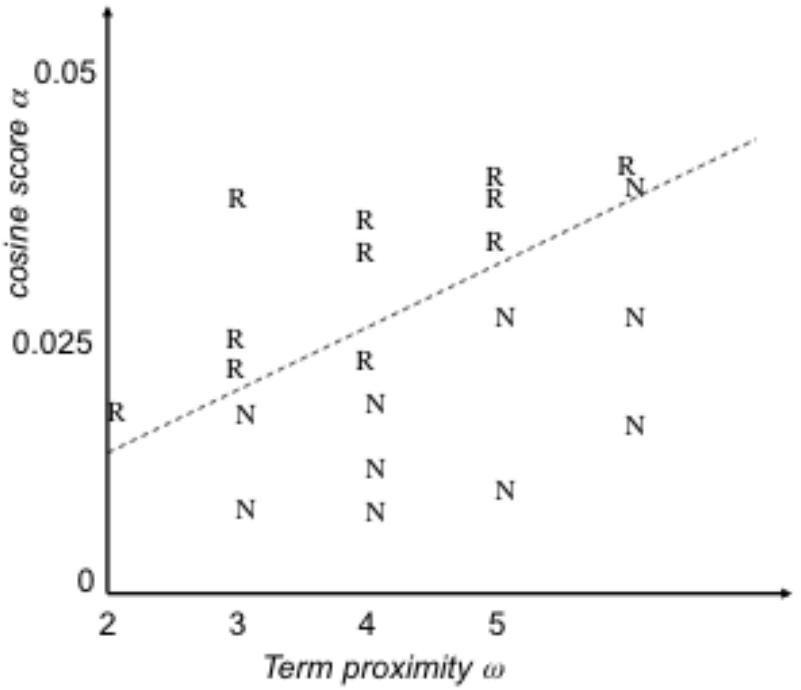
不失一般性，线性分类器可以采用下列通用形式：

$$Score(d, q) = Score(\alpha, \omega) = a\alpha + b\omega + c, \qquad (7)$$

公式中的系数 a, b, c 可以从训练语料中学到

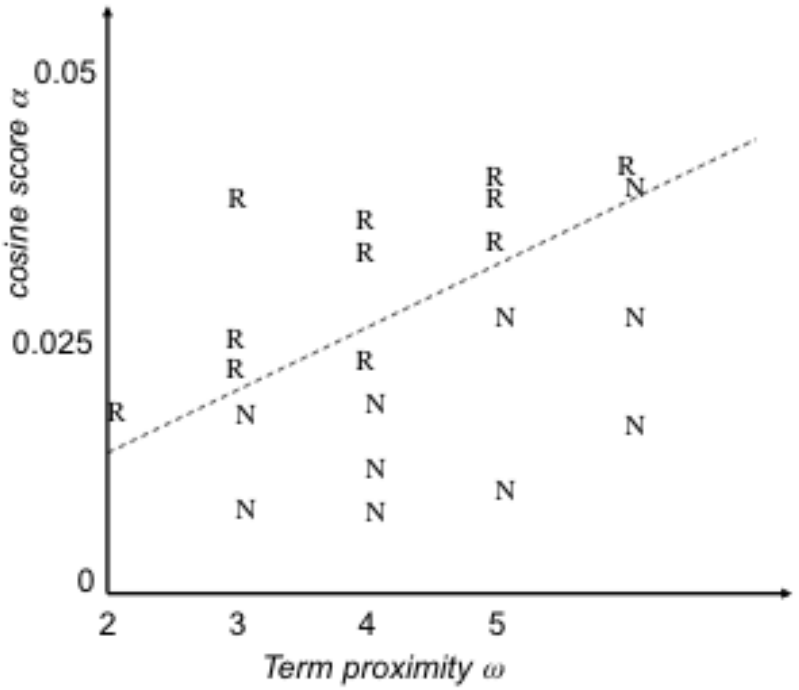
一个简单的机器学习评分的例子

- 函数$Score(\alpha,\omega)$可以看成悬挂在右图上空的一个平面
- 理想情况下，该平面在R表示的点之上的函数值接近于1，而在N表示的点之上的函数值接近于0



一个简单的机器学习评分的例子

- 引入阈值 θ
- 如果$Score(\alpha, \omega) > \theta$, 则认为文档相关，否则认为不相关
- 从上面的 SVM我们可以知道, $Score(\alpha, \omega) = \theta$ 的点构成一条直线 (图中虚线) → 它能将相关和不相关文档线性地分开



一个简单的机器学习评分的例子

因此，给定训练集情况下判定相关/不相关的问题就转变成一个学习问题，如前面提到，相当于学习一条能够将训练集中相关文档和不相关文档分开的虚线

- 在α-ω 平面上, 该直线可以写成基于α 和ω (分别表示斜率和截距)的一个方程
- 前面已经介绍选择直线的线性分类方法
- 如果能够构建大规模训练样例，那么就可以避免手动去调节评分中的参数
- 瓶颈： 维护一个合适的有代表性的训练样例需要较大的卡西开销，而且这些样例的相关性判定需要专家来进行

基于机器学习的检索结果排序

- 上面的例子可以很容易地推广到包含多个变量的函数
- 除余弦相似度及查询词项窗口之外，存在大量其他的相关性度量方法，如 PageRank之类的指标、文档时间、域贡献、文档长度等等
- 如果训练文档集中的这些指标可以计算出来，加上相关性判定，就可以利用任意数目的指标来学习分类器。

基于机器学习的检索结果排序

然而，利用上述方法来进行IR的排序未必是正确的问题处理方法

- 统计学家通常将问题分成分类问题 (预测一个类别型变量) 和回归问题 (预测一个实数型变量)
- 在这两者之间，有一个特别的称为序回归(**ordinal regression)的领域，其目标是预测一个序**
- 基于机器学习的Ad hoc检索可以看成是一个序回归问题，这是因为检索的目标是，给定q的情况下，对所有的文档进行排序

提纲

1. 上一讲回顾
2. 支持向量机
3. 文本分类中的问题
4. 基于布尔权重的学习
5. 基于实数权重的学习
6. 基于序回归的排序学习

将IR排序问题看成序回归

为什么将IR排序问题看成一个序回归问题？

- 对于同一查询，文档之间可以按照相对得分排序即可，并不一定要求每篇文档有一个全局的绝对得分
- 因此，只需要一个排序，而不要得到相关度的绝对得分，问题空间可以减小

这对于Web检索来说特别重要，Web检索中排名最高的一些结果非常重要

利用结构化SVM 进行IR排序

利用结构化SVM框架可以处理IR排序问题，对于一个查询，预测的类别是结果的排序

排序SVM的构建

- 给定一些已经判定的查询
- 对训练集中的每条查询q, 我们都有针对该查询的一系列文档集合，这些文档已经由人工按照其与查询的相关度排序
- 对每个文档、查询对，构造特征向量 $\psi_j = \psi(d_j , q)$，这里的特征可以采用前面讨论的特征
- 对于两篇文档d_i 和d_j ，可以计算特征向量之间的差异向量：:

$$\Phi(d_i , d_j , q) = \psi(d_i , q) - \psi(d_j , q) \qquad (8)$$

排序SVM的构建

- 依据假设， d_i 、d_j 中的一个更相关
- 如果 d_i 比 d_j更相关，记为$d_i \prec d_j$ (在检索结果中，d_i 应该出现在 d_j 前面), 那么分配给$\Phi(d_i, d_j, q)$向量的类别为 $y_{ijq} = +1$，否则为 -1

- 学习的目标是建立一个分类器，满足：

$$\vec{w}^{\mathrm{T}} \Phi(d_i, d_j, q) > 0 \text{ iff } d_i \prec d_j \qquad (9)$$

排序SVM(Ranking SVM)

- 该方法已经被用于构建排序函数，在标准数据集的IR评测中表现的性能优于普通的人工排序函数
- 参考《信息检索导论》第239页的一些参考文献

注意: 线性 vs. 非线性权重计算

- 前面介绍的方法也代表当前研究中的主要做法，即将特征进行线性组合
- 但是很多传统IR方法中还包括对基本量的非线性放缩方法，比如对词项频率或IDF取对数
- 当前来说，机器学习方法很擅长对线性组合的权重进行优化，但是并不擅长基本量的非线性放缩处理。
- 该领域仍然存在大量人工特征工程的方法

本讲内容

- 支持向量机
  - 线性可分：最大间隔
  - 非线性可分：最大间隔+最小错误
  - 空间转换：核函数及核技巧

- 排序机器学习
  - 基于基于布尔权重的学习
  - 基于基于实数权重的学习
  - 基于序回归的排序学习