深度之眼
deepshare.net

法律声明

本课件包括演示文稿、示例、代码、题库、视频和声音等内容，深度之眼和讲师拥有完全知识产权；只限于善意学习者在本课程使用，不得在课程范围外向任何第三方散播。任何其他人或者机构不得盗版、复制、仿造其中的创意和内容，我们保留一切通过法律手段追究违反者的权利。

课程详情请咨询

- 微信公众号：深度之眼
- 客服微信号：deepshare0920

公众号

微 信

关注公众号深度之眼，后台回复 统计学 ，获取统计学习方法第二版电子书及其他AI必学书籍

deepshare.net
深度之眼

第17章：潜在语义分析

17.1 LSA导入

导师：Irene

本节学习内容

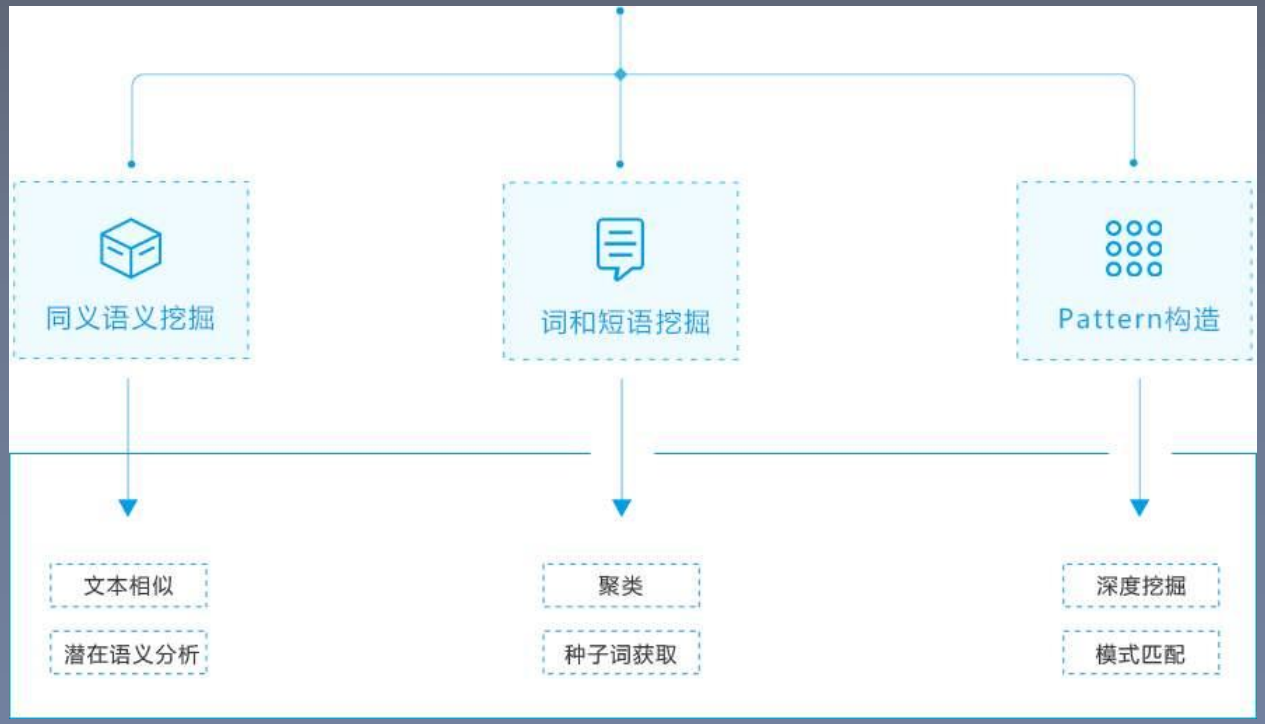
Learning content in this section

对应书本/课程章节	主要学习内容	学习目标
文本的话题分析	概念与相关表示法	理解文本的话题分析相关概念
语义相似度	语义相似度的内积表示法	学会计算语义相似度
非概率的话题分析模型	向量空间模型，语义内容表示	理解话题分析模型结构
单词向量空间	单词—文本矩阵的含义	掌握单词—文本矩阵的行列意义
单词向量空间实例分析	理解主题与本文相似度的关系	学会分析文本的相似度
话题向量空间	单词—话题矩阵的含义	掌握单词—话题矩阵的行列意义

潜在语义分析LSA导入

Tutorial of Latent Semantic Analysis

- 潜在语义分析（(latent semantic analysis, LSA）是一种无监督学习方法，主要用于**文本的话题分析**，通过矩阵分解发现文本与单词之间的基于话题的语义关系。
- 文本信息处理中，传统方法以单词向量表示文本的语义内容，以单词向量空间的度量表示文本之间的**语义相似度**。
- 基于**话题分析**（topic modeling）的基本想法，潜在语义分析旨在解决这种方法不能准确表示语义的问题，试图从大量的文本数据中发现潜在的话题，以话题向量表示文本的语义内容，以话题向量空间的度量更准确地表示文本之间的语义相似度。



潜在语义分析LSA导入

Tutorial of Latent Semantic Analysis

deepshare.net
深度之眼

- LSA使用**非概率的话题分析模型**。将文本集合表示为单词-文本矩阵，对单词-文本矩阵进行奇异值分解，从而得到话题向量空间，以及文本在话题向量空间的表示。
- 非负矩阵分解（non-negative matrix factorization, NMF）是另一种矩阵的因子分解方法，其特点是**分解的矩阵非负**，可用于话题分析
- 文本信息处理，比如文本信息检索、文本数据挖掘的一个核心问题是对文本的**语义内容进行表示**，并进行文本之间的**语义相似度**计算。
- 最简单的方法是利用**向量空间模型**（vector space model, VSM），也就是单词向量空间模型（word vector space model）。
- 向量空间模型的基本想法是，给定一个文本，用一个向量表示该文本的“语义”，向量的每一维对应一个单词，其数值为该单词在该文本中出现的**频数或权值。**
- 基本假设是文本中所有单词的出现情况表示了文本的语义内容，文本集合中的每个文本都表示为一个向量，存在于一个向量空间，向量空间的**度量**，如内积或标准化内积表示文本之间的**“语义相似度”**。

单词向量空间

Word vector space

- 给定一个含有n个文本的集合 $D = \{d_1, d_2, \cdots, d_n\}$ 以及在所有文本中出现的m个单词的集合 $W = \{w_1, w_2, \cdots, w_m\}$ 。
- 将单词在文本中出现的 数据用一个单词-文本矩阵（word-document matrix）表示，记作X:

$$X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{bmatrix}$$

- 这是一个 m x n 矩阵，元素 x_{ij} 表示单词 w_i 在文本.d_j 内中出现的频数或权值。
- 由于单 词的种类很多，而每个文本中出现单词的种类通常较少，所以单词-文本矩阵是一个稀疏矩阵。

单词向量空间

Word Vector Space

- 权值通常用单词频率-逆文本频率（term frequency-inverse document frequency, TF-IDF）表示，其定义是

$$\mathrm{TFIDF}_{ij} = \frac{\mathrm{tf}_{ij}}{\mathrm{tf}_{\cdot j}} \log \frac{\mathrm{df}}{\mathrm{df}_i}, \quad i = 1, 2, \cdots, m; \quad j = 1, 2, \cdots, n$$

- tf_{ij}：单词 w_i 出现在文本 d_j 中的频数
- $\mathrm{tf}_{\cdot j}$：是文本 d_j 中出现的所有单词的频 数之和
- df_i：含有单词 w_i 的文本数
- df：是文本集合D的全部文本数

单词向量空间

Word Vector Space

deepshare.net
深度之眼

- 直观上，一个单词在一个文本中出现的频数越高，这个单词在这个文本中的重要度就越高
- 一个单词在整个文本集合中出现的文本数越少，这个单词就越能表示其所在文本的特点，重要度就越高，单词在一个文本的TF-IDF是两种重要度的积，表示综合重要度。
- 单词向量空间模型直接使用单词-文本矩阵的信息。单词-文本矩阵的第j列向量 x_j 表示文本 d_j

$$x_j = \begin{bmatrix} x_{1j} \\ x_{2j} \\ \vdots \\ x_{mj} \end{bmatrix}, \quad j = 1, 2, \cdots, n$$

- x_{ij} :单词 w_i 在文本 d_j 的权值
- 权值越大，该单词在该文本中的重要度就越高

$$X = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}.$$

单词向量空间

Word Vector Space

- 两个单词向量的内积或标准化内积（余弦）表示对应的文本之间的语义相似度
- 因此，文本 d_i 与 d_j 之间的相似度为 $x_i \cdot x_j$，$\dfrac{x_i \cdot x_j}{\|x_i\| \, \|x_j\|}$
- 直观上，在两个文本中共同出现的单词越多，其语义内容就越相近，对应的单词向量同不为零的维度就越多，内积就越大（单词向量元素的值都是非负的），表示两个文本在语义内容上越相似
- 单词向量空间模型
  - 模型简单
  - 计算效率高
  - 有局限性，内积相似度未必能够准确表达两个文本的语义相似度
    - 一词多义性(polysemy)
    - 多词一义性(synonymy)

单词向量空间实例

Example of Word Vector Space

- 单词向量空间模型中，文本 d_1 与 d_2 相似度并不高，尽管两个文本的内容相似，这是因为同义词“airplane”与“aircraft”被当作了两个**独立**的单词，单词向量空间模型不考虑单 的同义性，在此情况下无法进行准确的相似度计算。
- 文本 d_3 与 d_4 有一定的相似度，尽管两个文本的内容并不相似，这是因为单词“apple”具有**多义**，可以表示“apple computer”和“fruit"，单词向量空间模型不考虑单词的多义性，在此情况下也无法进行准确的相似度计算。

	d_1	d_2	d_3	d_4
airplane	2			
aircraft		2		
computer			1	
apple			2	3
fruit				1
produce	1	2	2	1

话题向量空间

Topic Vector Space

deepshare.net
深度之眼

- 两个文本的语义相似度可以体现在两者的话题相似度上一个文本一般含有若干个话题。 如果两个文本的话题相似，那么两者的语义应该也相似。
- 话题可以由若干个**语义相关**的单词表示，同义词（如“airplane”与“aircraft"）可以表示同一个话题，而多义词（如“apple"）可以表示不同的话题。这样，基于话题的模型就可以解决上述基于单 词的模型存在的问题。
- 设想定义一种话题向量空间模型(topic vector space model)给定一个文本，用话题空间的一个向量表示该文本，该向量的每一分量对应一个话题，其数值为该话题在该文本中出现的权值，用两个向量的**内积或标准化内积**表示对应的两个文本的**语义相似度**。

- 注：单词向量空间模型与话题向量空间模型可以相互补充、同时使用。

话题向量空间

Topic Vector Space

- 给定一个文本集合 $D = \{d_1, d_2, \cdots, d_n\}$和一个相应的单词集合 $W = \{w_1, w_2, \cdots, w_m\}$ 。可以获得其单词-文本矩阵X，X构成原始的单词向量空间，每一列是一个文本 在单词向量空间中的表示

$$X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{bmatrix}$$

- 矩阵X也可以写作 $X = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}$

话题向量空间

Topic Vector Space

- 假设所有文本共含有k个话题。假设每个话题由一个定义在单词集合W上的m维向量表示，称为话题向量，即

$$t_l = \begin{bmatrix} t_{1l} \\ t_{2l} \\ \vdots \\ t_{ml} \end{bmatrix}, \quad l = 1, 2, \cdots, k$$

- t_{il}：单词 w_i 在话题 t_l 的权值，权值越大，该单词在该话题中的重要度就越高
- k个话题向量张成一个话题向量空间(topic vector space)，维数为k
- 话题向量空间T是单词向量空间X的一个子空间

话题向量空间

Topic Vector Space

- 话题向量空间T也可以表示为一个矩阵，称为单词-话题矩阵（word-topic matrix），记作

$$T = \begin{bmatrix} t_{11} & t_{12} & \cdots & t_{1k} \\ t_{21} & t_{22} & \cdots & t_{2k} \\ \vdots & \vdots & & \vdots \\ t_{m1} & t_{m2} & \cdots & t_{mk} \end{bmatrix}$$

- 矩阵T也可写作

$$T = \begin{bmatrix} t_1 & t_2 & \cdots & t_k \end{bmatrix}$$

结 语

在这次课程中，我们了解到了潜在语义分析导入、单词向量空间、话题向量空间的定义与表示

希望在课下，大家都能

掌握本节知识点并完成作业

deepshare.net

深度之眼

联系我们：

电话：18001992849

邮箱：service@deepshare.net

Q Q： 2677693114

公众号

客服微信

关注公众号深度之眼，后台回复 统计学 ， 获取统计学习方法第二版电子书及其他AI必学书籍

deepshare.net
深度之眼

第17章：潜在语义分析

17.2 LSA算法实现

导师：Irene

本节学习内容

Learning content in this section

对应书本/课程章节	主要学习内容	学习目标
文本在话题向量空间的表示	文本在话题向量空间的表示向量	理解文本在话题向量空间表示矩阵的意义
线性变换	文本由话题向量空间中的向量线性表示	理解线性变换系数的意义
矩阵的因子分解	单词文本矩阵分解为单词话题矩阵与话题文本矩阵	掌握单词文本矩阵的分解形式
潜在语义分析算法	矩阵的截断奇异值分解	掌握单词文本矩阵分解步骤
潜在语义分析实例	实际单词文本矩阵的因子分解	学会计算并分析单词文本矩阵的分解过程

文本在话题向量空间的表示

Text Expressed in Topic Vector Space

- 现在考虑文本集合D的文本 d_j，在单词向量空间中由一个向量 x_j 表示，将 x_j 投影到话题向量空间T中，得到在话题向量空间的一个向量 y_j，y_j 是一个k维向量，其表达式为

$$y_j = \begin{bmatrix} y_{1j} \\ y_{2j} \\ \vdots \\ y_{kj} \end{bmatrix}, \quad j = 1, 2, \cdots, n$$

- y_{ij}：文本 d_j 在话题 t_i 的权值，权值越大，该话题在该文本中的 重要度就越高

文本在话题向量空间的表示

Text Expressed in Topic Vector Space

- 矩阵Y表示话题在文本中出现的情况，称为话题-文本矩阵(topic-document matrix)，记作

$$Y = \begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1n} \\ y_{21} & y_{22} & \cdots & y_{2n} \\ \vdots & \vdots & & \vdots \\ y_{k1} & y_{k2} & \cdots & y_{kn} \end{bmatrix}$$

- 矩阵Y可一个写作

$$Y = \begin{bmatrix} y_1 & y_2 & \cdots & y_n \end{bmatrix}$$

线性变换

Linear Transformation

- 在单词向量空间的文本向量 x_j 可以通过它在话题空间中的向量 y_j 近似表示，具体地由k个话题向量以 y_j 为系数的线性组合近似表示

$$x_j \approx y_{1j}t_1 + y_{2j}t_2 + \cdots + y_{kj}t_k, \quad j = 1, 2, \cdots, n$$

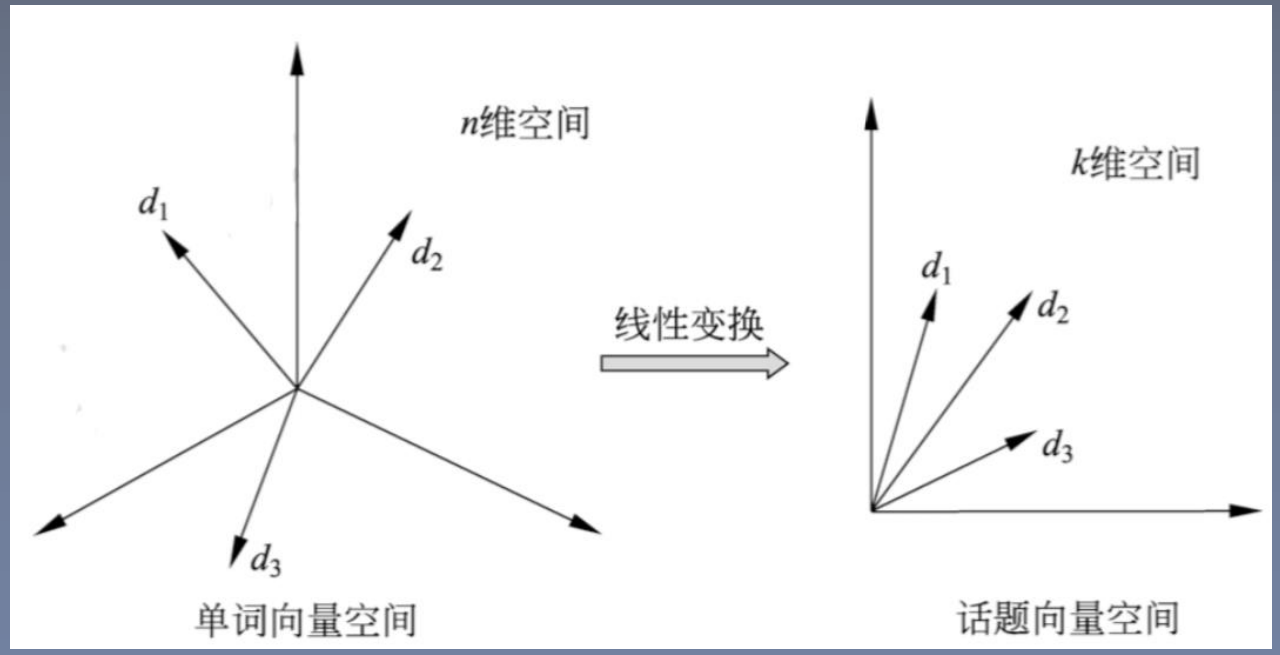
- 所以，单词-文本矩阵X可以近似的表示为单词-话题矩阵T与话题一文本矩阵Y的乘积形式。即是潜在语义分析。

$$X \approx TY$$

线性变换

Linear Transformation

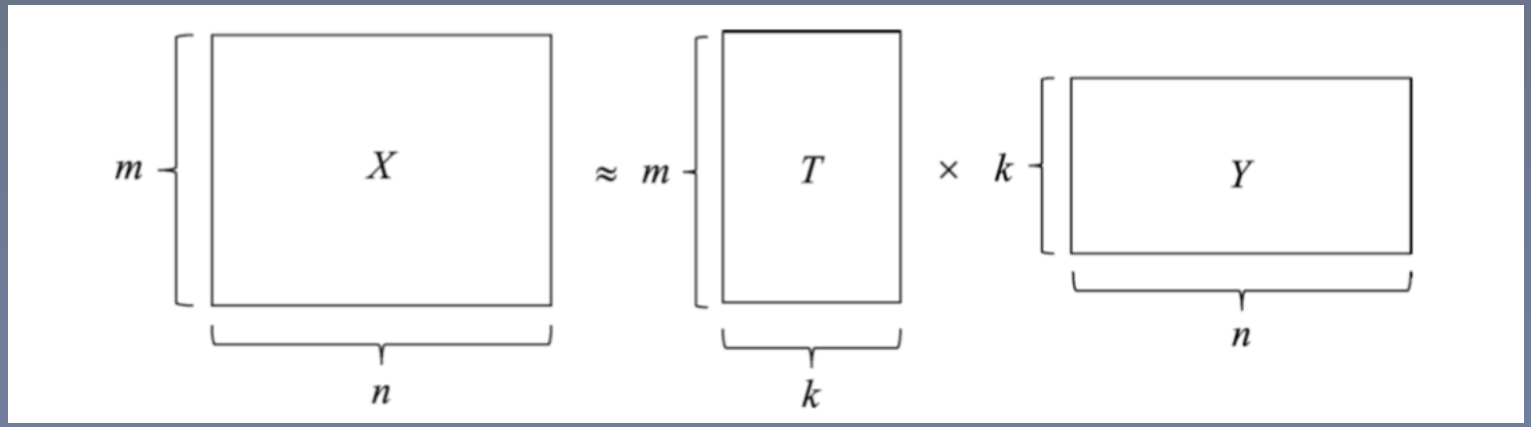
- 直观上，潜在语义分析是将文本在单词向量空间的表示通过线性变换转换为在话 题向量空间中的表示



线性变换

Linear Transformation

- 潜在语义分析通过矩阵因子分解实现，单词-文本矩阵X可以近似地表示为单词-话题矩阵T与话题-文本矩阵Y的乘积形式：



线性变换

Linear Transformation

- 在原始的单词向量空间中，两个文本 d_i 与 d_j 的相似度可以由对应的向量的内积 表示，即 $x_i \cdot x_j$。

- 经过潜在语义分析之后，在话题向量空间中，两个文本 d_i 与 d_j 的相似度可以由对应的向量的内积即 $y_i \cdot y_j$ 表示。

- 要进行潜在语义分析，需要同时决定两部分的内容，一是话题向量空间T，二是文本在话题空间的表示Y，使两者的乘积是原始矩阵数据的近似，而这一结果完全从话题-文本矩阵的信息中获得。

潜在语义分析算法

Algorithm of LSA

- 潜在语义对单词-文本矩阵进行奇异值分解， 将其左矩阵作为话题向量空间，将其对角矩阵与右矩阵的乘积作为文本在话题向量空间的表示。
- （1）单词-文本矩阵
- 给定文本集合 $D = \{d_1, d_2, \cdots, d_n\}$和单词集合 $W = \{w_1, w_2, \cdots, w_m\}$ 。
- 潜在语义分析首先将这些数据表成一个单词-文本矩阵

$$X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{bmatrix}$$

潜在语义分析算法

Algorithm of LSA

- （2）截断奇异值分解
- 潜在语义分析根据确定的话题个数k对单词-文本矩阵X进行截断奇异值分解

$$X \approx U_k \Sigma_k V_k^{\mathrm{T}} = \begin{bmatrix} u_1 & u_2 & \cdots & u_k \end{bmatrix} \begin{bmatrix} \sigma_1 & 0 & 0 & 0 \\ 0 & \sigma_2 & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \sigma_k \end{bmatrix} \begin{bmatrix} v_1^{\mathrm{T}} \\ v_2^{\mathrm{T}} \\ \vdots \\ v_k^{\mathrm{T}} \end{bmatrix}$$

潜在语义分析算法

Algorithm of LSA

- （3）话题向量空间
- 在单词—文本矩阵X的截断奇异值分解式中，矩阵U_k的每一个列向量 $\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_k$ 表示一个话题，称为话题向量。由这k个话题向量张成一个子空间

$$U_k = \begin{bmatrix} u_1 & u_2 & \cdots & u_k \end{bmatrix}$$

称为话题向量空间。

潜在语义分析算法

Algorithm of LSA

- （4） 文本的话题空间表示
- 有了话题向量空间，接着考虑文本在话题空间的表示

$$
\begin{aligned}
X &= \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix} \approx U_k \Sigma_k V_k^{\mathrm{T}} \\
&= \begin{bmatrix} u_1 & u_2 & \cdots & u_k \end{bmatrix} \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & 0 \\ & & \ddots & \\ & 0 & & \sigma_k \end{bmatrix} \begin{bmatrix} v_{11} & v_{21} & \cdots & v_{n1} \\ v_{12} & v_{22} & \cdots & v_{n2} \\ \vdots & \vdots & & \vdots \\ v_{1k} & v_{2k} & \cdots & v_{nk} \end{bmatrix} \\
&= \begin{bmatrix} u_1 & u_2 & \cdots & u_k \end{bmatrix} \begin{bmatrix} \sigma_1 v_{11} & \sigma_1 v_{21} & \cdots & \sigma_1 v_{n1} \\ \sigma_2 v_{12} & \sigma_2 v_{22} & \cdots & \sigma_2 v_{n2} \\ \vdots & \vdots & & \vdots \\ \sigma_k v_{1k} & \sigma_k v_{2k} & \cdots & \sigma_k v_{nk} \end{bmatrix}
\end{aligned} \tag{17.14}
$$

其中

$$
u_l = \begin{bmatrix} u_{1l} \\ u_{2l} \\ \vdots \\ u_{ml} \end{bmatrix}, \quad l = 1, 2, \cdots, k
$$

潜在语义分析算法

Algorithm of LSA

- 由式(17.14)知， 矩阵X的第j列向量 x_j 满足

$$
\begin{aligned}
x_j &\approx U_k(\Sigma_k V_k^{\mathrm{T}})_j \\
&= \begin{bmatrix} u_1 & u_2 & \cdots & u_k \end{bmatrix} \begin{bmatrix} \sigma_1 v_{j1} \\ \sigma_2 v_{j2} \\ \vdots \\ \sigma_k v_{jk} \end{bmatrix} \\
&= \sum_{l=1}^{k} \sigma_l v_{jl} u_l, \quad j = 1, 2, \cdots, n
\end{aligned}
$$

- $\left(\Sigma_k V_k^{\mathrm{T}}\right)_j$ 是矩阵 $\left(\sum_k \cdot V_k^T\right)$ 第j列向量
- 式(17.15)是文本 d_j 的近似表达式， 由 k个话题向量 u_l 的线性组合构成

潜在语义分析算法

Algorithm of LSA

- 矩阵$\left(\sum_k \cdot V_k^T\right)$的每一个列向量

$$\begin{bmatrix} \sigma_1 v_{11} \\ \sigma_2 v_{12} \\ \vdots \\ \sigma_k v_{1k} \end{bmatrix}, \begin{bmatrix} \sigma_1 v_{21} \\ \sigma_2 v_{22} \\ \vdots \\ \sigma_k v_{2k} \end{bmatrix}, \cdots, \begin{bmatrix} \sigma_1 v_{n1} \\ \sigma_2 v_{n2} \\ \vdots \\ \sigma_k v_{nk} \end{bmatrix}$$

- 是一个文本在话题向量空间的表示
- 综上，可以通过对单词—文本矩阵的奇异值分解进行潜在语义分析$X \approx U_k \Sigma_k V_k^{\mathrm{T}} = U_k \left(\Sigma_k V_k^{\mathrm{T}}\right)$得到话题空间$U_k$，以及文本在话题空间的表示$\left(\sum_k \cdot V_k^T\right)$

潜在语义分析例题

Example of LSA

- 假设有9个文本，11个单词，单词一文本矩 阵x为11 x 9矩阵，矩阵的元素是单词在文本中出现的频数，表示如下：

Index Words	Titles								
	T1	T2	T3	T4	T5	T6	T7	T8	T9
book			1	1					
dads						1			1
dummies		1						1	
estate							1		1
guide	1					1			
investing	1	1	1	1	1	1	1	1	1
market	1		1						
real							1		1
rich						2			1
stock	1		1					1	
value				1	1				

- 进行潜在语义分析。

潜在语义分析例题

Example of LSA

- 实施对矩阵的截断奇异值分解，假设话题的个数是3，截断奇异值分解结果为:

Book	0.15	−0.27	0.04
Dads	0.24	0.38	−0.09
Dummies	0.13	−0.17	0.07
Estate	0.18	0.19	0.45
Guide	0.22	0.09	−0.46
Investing	0.74	−0.21	0.21
Market	0.18	−0.30	−0.28
Real	0.18	0.19	0.45
Rich	0.36	0.59	−0.34
Stock	0.25	−0.42	−0.28
Value	0.12	−0.14	0.23

*

3.91	0	0
0	2.61	0
0	0	2.00

*

T1	T2	T3	T4	T5	T6	T7	T8	T9
0.35	0.22	0.34	0.26	0.22	0.49	0.28	0.29	0.44
−0.32	−0.15	−0.46	−0.24	−0.14	0.55	0.07	−0.31	0.44
−0.41	0.14	−0.16	0.25	0.22	−0.51	0.55	0.00	0.34

潜在语义分析例题

Example of LSA

- 左矩阵U有3个列向量（左奇异向量）。第1列向量 u_1 的值均为正，V_3^{T}第2列向量 u_2 和第3列向量 u_3 的值有正有负。中间的对角矩阵 $\sum_3$ 的元素是3个由 大到小的奇异值（正值）。右矩阵是 V_3^{T}，其转置矩阵V_3也有3个列向量（右奇异向量）。第1列向量 v_1 的值也都为正，第2列向量 v_2 和第3列向量 v_3 的值有正有负。现在，将 $\sum_3$ 与 V_3^{T} 相乘，整体变成两个矩阵乘积的形式

$$X \approx U_3(\Sigma_3 V_3^{\mathrm{T}})$$

$$= \begin{bmatrix} 0.15 & -0.27 & 0.04 \\ 0.24 & 0.38 & -0.09 \\ 0.13 & -0.17 & 0.07 \\ 0.18 & 0.19 & 0.45 \\ 0.22 & 0.09 & -0.46 \\ 0.74 & -0.21 & 0.21 \\ 0.18 & -0.30 & -0.28 \\ 0.18 & 0.19 & 0.45 \\ 0.36 & 0.59 & -0.34 \\ 0.25 & -0.42 & -0.28 \\ 0.12 & -0.14 & 0.23 \end{bmatrix} \begin{bmatrix} 1.37 & 0.86 & 1.33 & 1.02 & 0.86 & 1.92 & 1.09 & 1.13 & 1.72 \\ -0.84 & -0.39 & -1.20 & -0.63 & -0.37 & 1.44 & 0.18 & -0.81 & 1.15 \\ -0.82 & 0.28 & -0.32 & 0.50 & 0.44 & -1.02 & 1.10 & 0.00 & 0.68 \end{bmatrix}$$

矩阵U_3有3个列向量，表示3个话题，矩阵U_3表示话题向量空间。矩阵 $(\Sigma_3 V_3^{\mathrm{T}})$ 有9个列向量，表示9个文本，矩阵 $(\Sigma_3 V_3^{\mathrm{T}})$ 是文本集合在话题向量空间的表示。

结 语

在这次课程中，我们了解到了线性变换、矩阵的因子分解，潜在语义分析算法

希望在课下，大家都能

掌握本节知识点并完成作业

deepshare.net

深度之眼

联系我们：

电话：18001992849

邮箱：service@deepshare.net

Q Q： 2677693114

公众号

客服微信

关注公众号深度之眼，后台回复 统计学 ， 获取统计学习方法第二版电子书及其他AI必学书籍

deepshare.net
深度之眼

第17章：潜在语义分析

17.3 非负矩阵分解算法

导师：Irene

本节学习内容

Learning content in this section

对应书本/课程章节	主要学习内容	学习目标
非负矩阵分解算法	非负矩阵X分解为两个非负矩阵乘积	理解非负矩阵分解原理
潜在语义分析模型	W与H矩阵的具体行列意义	掌握非负矩阵分解在话题分析中的作用
非负矩阵分解的形式化	散度损失函数	理解散度损失函数的实际含义
非负矩阵分解定理	乘法更新规则非增定理	理解乘法更新规则非增的原因
非负矩阵分解迭代算法	初始化、迭代与更新W,H矩阵	掌握非负矩阵分解迭代算法步骤

非负矩阵分解算法

Non-Negative Matrix Factorization Algorithm

- 非负矩阵分解也可以用于话题分析，对单词—文本矩阵进行非负矩阵分解，将其左矩阵作为话题向量空间，将其右矩阵作为文本在话题向量空间的表示。注意通常单词-文本矩阵是非负的。
- 给定一个非负矩阵X≥0，找到两个非负矩阵W≥0和H≥ 0，使得

$$X \approx WH$$

- 即将非负矩阵X分解为两个非负矩阵W和H的乘积的形式，称为非负矩阵分解。因为WH与X完全相等很难实现，所以只要求WH与X近似相等。

非负矩阵分解算法

Non-Negative Matrix Factorization Algorithm

- 假设非负矩阵X是 m x n 矩阵，非负矩阵W和H分别为 m x k 矩阵和 k x n 矩阵。假设 k<min(m, n)，即W和H小于原矩阵X，所以非负矩阵分解是对原数据的压缩。
- 由 $X \approx WH$ 知，矩阵X的第j列向量 x_j 满足

$$x_j \approx Wh_j$$

$$= \begin{bmatrix} w_1 & w_2 & \cdots & w_k \end{bmatrix} \begin{bmatrix} h_{1j} \\ h_{2j} \\ \vdots \\ h_{kj} \end{bmatrix} = \sum_{l=1}^{k} h_{lj} w_l, \quad j = 1, 2, \cdots, n$$

- 矩阵X的第j列 x_j 可以由矩阵W的k个列 w_l 的线性组合逼近，线性组合的系数是矩阵H的第j列 h_j的元素。
- 非负矩阵分解旨在用较少的基向量、系数向量来表示较大的数据矩阵。

潜在语义分析模型

LSA Model

- 给定一个 m x n 非负的单词-文本矩阵X≥0
- 假设文本集合共包含k个话题， 对X进行非负矩阵分解。即求非负的 m x k 矩阵W≥0和 k x n 矩阵H≥0，使得

$$X \approx WH$$

- 令 $W = \begin{bmatrix} w_1 & w_2 & \cdots & w_k \end{bmatrix}$为话题向量空间，$w_1, w_2, \cdots, w_k$ 表示文本集合的k个 话题，令 $H = \begin{bmatrix} h_1 & h_2 & \cdots & h_n \end{bmatrix}$ 为文本在话题向量空间的表示，$h_1, h_2, \cdots, h_n$ 表示 文本集合的n个文本

非负矩阵分解的形式化

Expression of Non-Negative Matrix Factorization

- 非负矩阵分解可以形式化为最优化问题求解。首先定义损失函数或代价函数。

- 第一种损失函数是平方损失。设两个非负矩阵 $A = [a_{ij}]_{m \times n}$ ，和 $B = [b_{ij}]_{m \times n}$ ，平方损失函数定义为

$$\|A - B\|^2 = \sum_{i,j} (a_{ij} - b_{ij})^2$$

- 其下界是0，当且仅当A=B时达到下界。

非负矩阵分解的形式化

Expression of Non-Negative Matrix Factorization

- 另一种损失函数是散度（divergence)。设两个非负矩阵 $A=[a_{ij}]_{m\times n}$ 和$B=[b_{ij}]_{m\times n}$散度损失函数定义为

$$D(A\|B)=\sum_{i,j}\left(a_{ij}\log\frac{a_{ij}}{b_{ij}}-a_{ij}+b_{ij}\right)$$

- 其下界也是0，当且仅当A= B时达到下界。A和B不对称。

- 当 $\sum_{i,j}a_{ij}=\sum_{i,j}b_{ij}=1$ 时,散度损失函数退化为Kuliback-Leiber散度或相对熵,

这时A和B是概率分布。

非负矩阵分解的形式化

Expression of Non-Negative Matrix Factorization

- 目标函数 $\|X - WH\|^2$ 关于W和H的最小化，满足约束条件W, H≥0，即

$$\begin{aligned} \min_{W,H} \quad & \|X - WH\|^2 \\ \text{s.t.} \quad & W, H \geqslant 0 \end{aligned}$$

- 或者，目标函数 $D(X\|WH)$ 关于W和H的最小化，满足约束条件W, H≥0，即

$$\begin{aligned} \min_{W,H} \quad & D(X\|WH) \\ \text{s.t.} \quad & W, H \geqslant 0 \end{aligned}$$

非负矩阵分解定理

Non-Negative Matrix Factorization Theorem

- **定理17.1**：平方损失 $\|X - WH\|^2$ 对下列乘法更新规则

$$H_{lj} \leftarrow H_{lj} \frac{(W^{\mathrm{T}}X)_{lj}}{(W^{\mathrm{T}}WH)_{lj}}$$

$$W_{il} \leftarrow W_{il} \frac{(XH^{\mathrm{T}})_{il}}{(WHH^{\mathrm{T}})_{il}}$$

是非增的。当且仅当W和H是平方损失函数的稳定点时函数的更新不变。

非负矩阵分解定理

Non-Negative Matrix Factorization Theorem

- **定理17.2**：散度损失D(X-WH）对下列乘法更新规则

$$H_{ij} \leftarrow H_{ij} \frac{\sum_i [W_{il} X_{ij} / (WH)_{ij}]}{\sum_i W_{il}}$$
$$W_{il} \leftarrow W_{il} \frac{\sum_j [H_{lj} X_{ij} / (WH)_{ij}]}{\sum_j H_{lj}}$$

是非增的。当且仅当W和H是平方损失函数的稳定点时函数的更新不变。

非负矩阵分解算法

Non-Negative Matrix Factorization Algorithm

- 最优化目标函数是 $\|X - WH\|^2$，为了方便将目标函数乘以1/2，其最优解与原问 题相同，记作

$$J(W,H) = \frac{1}{2}\|X - WH\|^2 = \frac{1}{2}\sum [X_{ij} - (WH)_{ij}]^2$$

- 应用梯度下降法求解。首先求目标函数的梯度

$$\frac{\partial J(W,H)}{\partial W_{il}} = -\sum_j [X_{ij} - (WH)_{ij}] H_{lj}$$

$$= -\left[\left(XH^{\mathrm{T}}\right)_{il} - \left(WHH^{\mathrm{T}}\right)_{ii}\right]$$

- 同样可得

$$\frac{\partial J(W,H)}{\partial H_{lj}} = -\left[\left(W^{\mathrm{T}}X\right)_{lj} - \left(W^{\mathrm{T}}WH\right)_{lj}\right]$$

非负矩阵分解算法

Non-Negative Matrix Factorization Algorithm

- 然后求得梯度下降法的更新规则

$$W_{il} = W_{il} + \lambda_{il}\left[\left(XH^{\mathrm{T}}\right)_{il} - \left(WHH^{\mathrm{T}}\right)_{il}\right]$$
$$H_{lj} = H_{lj} + \mu_{lj}\left[\left(W^{\mathrm{T}}X\right)_{lj} - \left(W^{\mathrm{T}}WH\right)_{lj}\right]$$

- 式中 λ_{il}, μ_{lj} 是步长。选取

$$\lambda_{il} = \frac{W_{il}}{\left(WHH^{\mathrm{T}}\right)_{il}}, \quad \mu_{lj} = \frac{H_{lj}}{\left(W^{\mathrm{T}}WH\right)_{lj}}$$

- 即得乘法更新规则

$$W_{il} = W_{il}\frac{\left(XH^{\mathrm{T}}\right)_{il}}{\left(WHH^{\mathrm{T}}\right)_{il}}, \quad i = 1, 2, \cdots, m; \quad l = 1, 2, \cdots, k$$

$$H_{lj} = H_{lj}\frac{\left(W^{\mathrm{T}}X\right)_{lj}}{\left(W^{\mathrm{T}}WH\right)_{lj}}, \quad l = 1, 2, \cdots, k; \quad j = 1, 2, \cdots, n$$

非负矩阵分解迭代算法

Non-Negative Matrix Factorization Iteration Algorithm

- 输入：单词-文本矩阵X≥0，文本集合的话题个数k，最大迭代次数t；
- 输出：话题矩阵W，文本表示矩阵H。

（1）初始化

W≥0，并对W的每一列数据归一化；

H≥0；

（2）迭代

对迭代次数由1到t执行下列步骤：

(a)更新W的元素，对l从1到k，i从1到m按式（17.33）更新W_{il}

(b)更新H的元素，对l从1到k，j从1到n按式（17.34）更新H_{lj}

结 语

在这次课程中，我们了解到了潜在语义分析模型，非负矩阵分解迭代算法与乘法更新规则非增定理

希望在课下，大家都能

掌握本节知识点并完成作业

deepshare.net

深度之眼

联系我们：

电话：18001992849

邮箱：service@deepshare.net

Q Q： 2677693114

公众号

客服微信

关注公众号深度之眼，后台回复 统计学 ， 获取统计学习方法第二版电子书及其他AI必学书籍

deepshare.net
深度之眼

第17章：潜在语义分析

17.4 LSA案例分析与编程实现

导师：Irene

本节学习内容

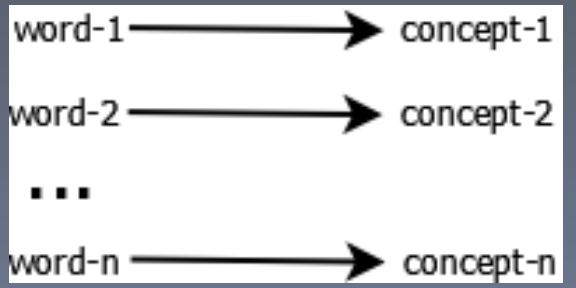
Learning content in this section

对应书本/课程章节	主要学习内容	学习目标
潜在语义分析工作原理	关键词，语义空间，对比分析	理解LSA的工作原理和相关概念
潜在语义分析实施步骤	奇异值分解法在具体案例的实现	理解案例中奇异值分解的方法和目的
LSA案例引入	索引词、类簇的相关定义	理解案例目的及相关Python代码
案例构建与语义分析	单词-标题矩阵，文章解析方法	掌握Python代码，实现文章解析与聚类
TFIDF指标调整权重	将单词-标题矩阵进行权重调整	掌握TFIDF指标计算方法和实际含义
奇异值分解与聚类结果	确定有效维度，实现标题聚类	掌握Python聚类及可视化相关代码
LSA优劣与应用	LSA优势与不足及应用领域	理解LSA的优缺点和适用范围

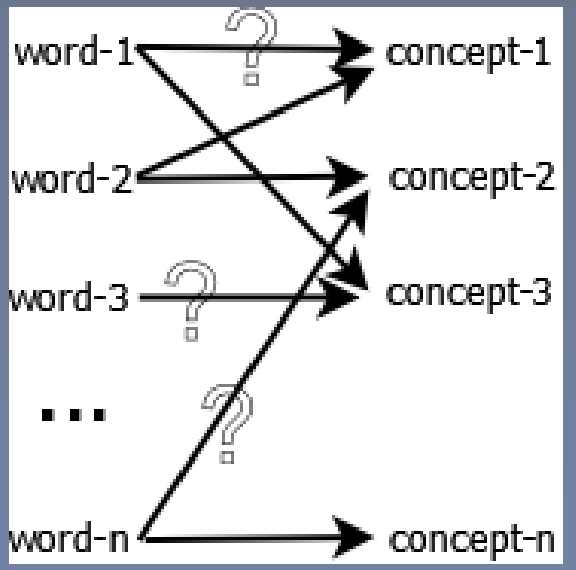
潜在语义分析LSA案例导入

Case Tutorial of Latent Semantic Analysis

潜在语义分析LSA（Latent Semantic Analysis）也叫作潜在语义索引LSI（Latent Semantic Indexing） 顾名思义是通过分析文章（documents）来挖掘文章的潜在意思或语义（concepts）。如果每个单词都仅以着一个语义，同时每个语义仅仅由一个单词来表示，那么LSA将十分简单，即简单地将进行语义和单词间的映射。



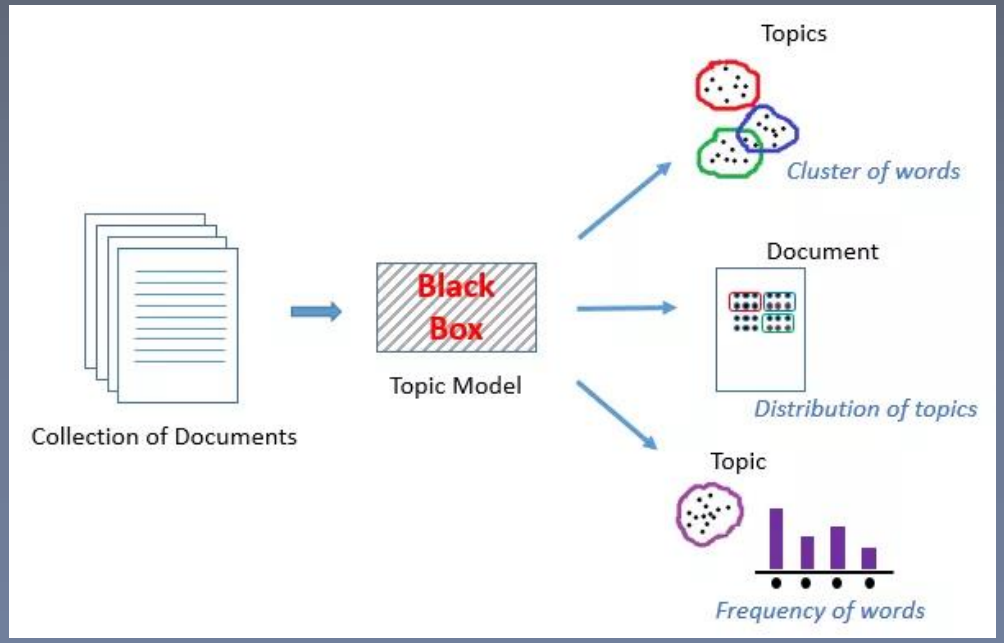
然而LSA并没有这么简单。因为不同的单词可以表示同一个语义，或一个单词同时具有多个不同的意思，这些的模糊歧义使**语义的准确识别**变得十分困难。例如，bank 这个单词如果和mortgage, loans, rates 这些单词同时出现时，bank 很可能表示金融机构的意思。可是如果bank 这个单词和lures, casting, fish一起出现，那么很可能表示河岸的意思。



LSA的工作原理

How Latent Semantic Analysis Works

LSA目的是解决如通过**搜索词/关键词**（search words）定位出相关文章。如何通过对比单词来定位文章是一个难点，因为我们正在要做的是对比单词背后的语义。潜在语义分析的基本原理是将文章和单词懂**映射**到语义空间（“concept” space）上，并在该空间进行**对比分析**。



由于作家在创作文章可以随意地选择各种单词来表达，因此不同的作家的词语选择风格都大不相同，表达的语义也因此变得模糊。这种单词选择的随机性必然将**噪声**的引入到“单词-语义关系”（word-concept relationship）。LSA能过滤掉一些噪声，同时能在语料库中找出一个最小的**语义子集**（to find the smallest set of concepts that spans all the documents）。

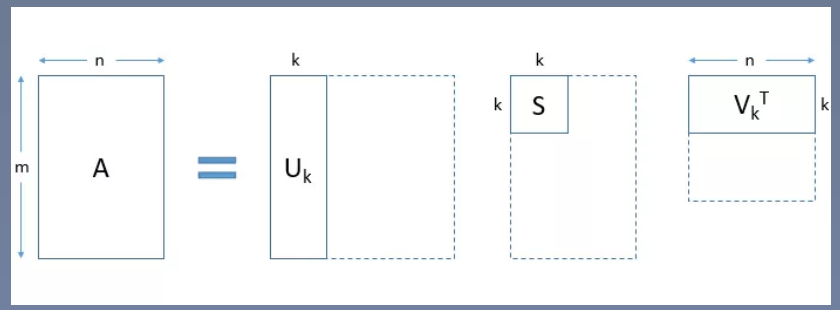
LSA的实施步骤

Realization Steps of Latent Semantic Analysis

假设有m篇文档，其中包含n个唯一词项（单词）。我们希望从所有文档的文本数据中提取出k个主题:

- 生成一个m×n维的文档-词项矩阵（Document-Term Matrix），矩阵元素为TF-IDF分数
- 然后使用**奇异值分解**（SVD）把上述矩阵的维度降到k（预期的主题数）维:$A=USV^T$
- SVD将一个矩阵分解为三个矩阵。假设我们利用SVD分解矩阵A，我们会得到矩阵U，矩阵S和矩阵V^T,矩阵U_k（document-term matrix）的每个行向量代表相应的文档。这些向量长度k是预期的主题数。代表数据中词项的向量可以在矩阵V_k（term-topic matrix）中找到。
- 因此，SVD为数据中的每篇文档和每个词项都提供了向量。每个向量的长度均为k。我们可以使用**余弦相似度**的方法通过这些向量找到相似的单词和文档。

	Terms				
Documents	T1	T2	T3	...	Tn
D1	0.2	0.1	0.5	...	0.1
D2	0.1	0.3	0.4	...	0.3
D3	0.3	0.1	0.1	...	0.5
...	...	...	...	...	...
Dm	0.2	0.1	0.2	...	0.1



LSA的实例

An Example of Latent Semantic Analysis

当在Amazon.com上搜索“investing”时将返回10个书名，这些书名都有共同一个**索引词**（index word）。一个索引词可以是符合以下条件的如何单词:

1.出现在2个或以上的文章题目中。
2.**停止词**：词意一般，如“and”, “the”等（(known as stop words)）。这些词对文章的语音并没起到突出的作用，应被过滤掉。

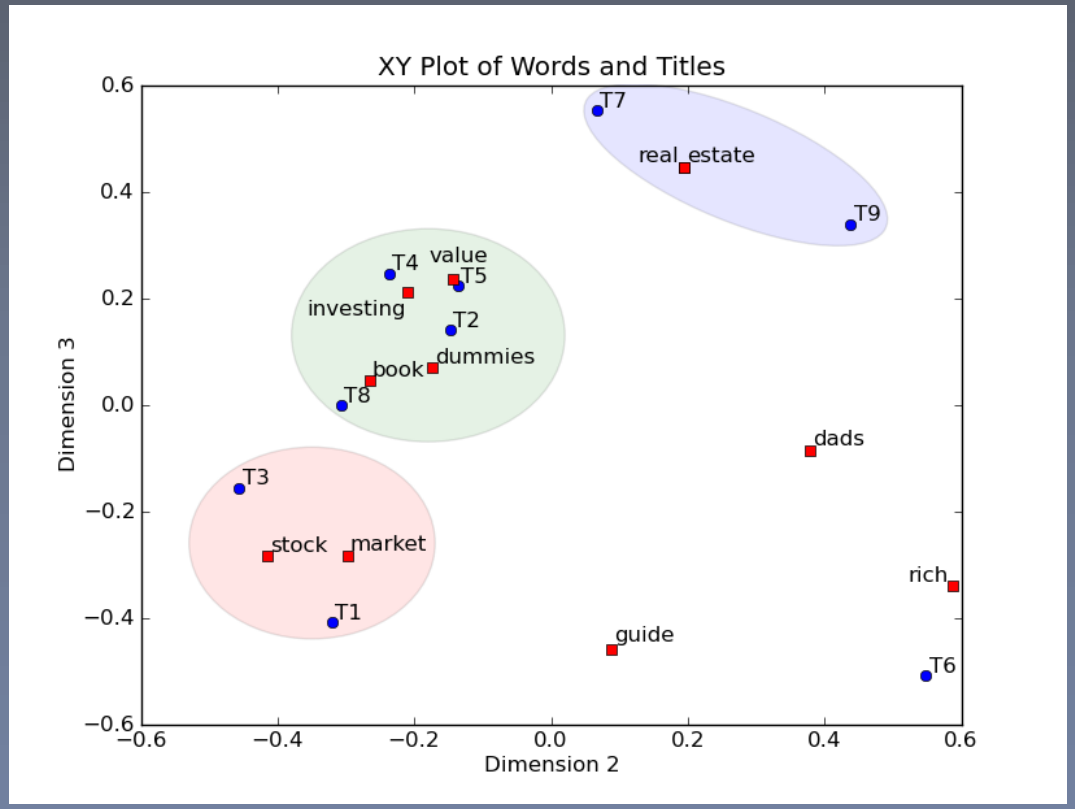
在本例中，我们将过滤以下单词: “and”, “edition”, “for”, “in”, “little”, “of”, “the”, “to”， 这里有9个标题:

1."The Neatest Little Guide to Stock Market Investing"
2."Investing For Dummies, 4th Edition"
3."The Little Book of Common Sense Investing: The Only Way to Guarantee Your Fair Share of Stock Market Returns"
4."The Little Book of Value Investing"
5."Value Investing: From Graham to Buffett and Beyond"
6."Rich Dad's Guide to Investing: What the Rich Invest in, That the Poor and the Middle Class Do Not!"
7."Investing in Real Estate, 5th Edition"
8."Stock Investing For Dummies"
9."Rich Dad's Advisors: The ABC's of Real Estate Investing: The Secrets of Finding Hidden Profits Most Investors Miss"

LSA的实例

An Example of Latent Semantic Analysis

上面的例子进过LSA后，我们能画出索引词和文章标题的二位XY图，并能标注出文章的类簇。9个标题用蓝点标注，11个索引词用红点标注。我们不仅能标注出文章的**类簇**，还能标注出相应的**索引词**，通过这些索引词来给类簇打上标签。



图中蓝色类簇包含了T7和T9，该类簇是有关real estate的文章。绿色类簇包含了T2, T4, T5, 和T8,，该类簇是有关 value investing的文章。最后红色类簇包含了T1 和T3, 该类簇是有关 stock market的文章。而T6是一个异常标题。

LSA的实例

An Example of Latent Semantic Analysis

Part 1 - Creating the Count Matrix

首先，LSA需要创建**单词-标题（或文章）矩阵**。在该矩阵中，行表示索引词，而列表示题目。每个元素表示对应的标题包含多少个相应的索引词。

例如，"book" 在T3和T4中出现了1次,而"investing" 出现在所有的表中。一般情况下LSA创建的单词-标题矩阵会相对巨大，而且十分稀疏（大部分元素为0），这是因为每个标题或文章一般只包含十分少的频繁单词。改进的LSA通过这种**稀疏性**能有效降低内存的损耗和算法复杂度。

Index Words	Titles								
	T1	T2	T3	T4	T5	T6	T7	T8	T9
book			1	1					
dads						1			1
dummies		1						1	
estate							1		1
guide	1					1			
investing	1	1	1	1	1	1	1	1	1
market	1		1						
real							1		1
rich						2			1
stock	1		1					1	
value				1	1				

LSA的实例

An Example of Latent Semantic Analysis

Python - Import Functions

首先需要加载几个python的数学运算模块。NumPy 是一个进行线性代数包，加载zeros函数，该函数用于创建0矩阵。通过线性代数包scipy能得到svd的算法函数，SVD是整个LSA算法的核心。

```
from numpy import zeros
from scipy.linalg import svd
```

Python - Define Data

9本书的标题包含8个停止词，在计算词频时将忽略这些停止词，再进行分词的标点符号也将被忽略。

```
titles =
[
"The Neatest Little Guide to Stock Market Investing", "Investing For Dummies, 4th Edition", "The Little Book of Common Sense Investing: The Only Way to Guarantee Your Fair Share of Stock Market Returns", "The Little Book of Value Investing", "Value Investing: From Graham to Buffett and Beyond", "Rich Dad's Guide to Investing: What the Rich Invest in, That the Poor and the Middle Class Do Not!", "Investing in Real Estate, 5th Edition", "Stock Investing For Dummies", "Rich Dad's Advisors: The ABC's of Real Estate Investing: The Secrets of Finding Hidden Profits Most Investors Miss"
]
stopwords = ['and','edition','for','in','little','of','the','to']
ignorechars = ''',:'!''
```

LSA的实例

An Example of Latent Semantic Analysis

deepshare.net
深度之眼

Python - Define LSA Class

LSA类的方法有初始化，文章解析，创建单词计数矩阵。初始化方法 __init__ 用于实例化LSA，并加载停止词集合和标点符号集合，还初始化词典（word dictionary）和文章计数变量。

```
class LSA(object):
def __init__(self, stopwords, ignorechars):
self.stopwords = stopwords
self.ignorechars = ignorechars
self.wdict = {}
self.dcount = 0
```

Python - Parse Documents

文章解析方法是对文章进行词汇筛选处理，并将所有字母变成小写（指英文文章），最后滤掉停止词和标点。该方法将非停止词加入到词典，并进行词频计数。例如，单词“book”出现在再T3和T4，设self.wdict[‘book’] = [3, 4] 。当处理完一篇文章的所有分词，就递增文章计数变量，并输入下一篇文章进行解析。

```
def parse(self, doc):
words = doc.split();
for w in words:
w = w.lower().translate(None, self.ignorechars)
if w in self.stopwords:
 continue
elif w in self.wdict:
 self.wdict[w].append(self.dcount)
else:
 self.wdict[w] = [self.dcount]
 self.dcount += 1
```

LSA的实例

An Example of Latent Semantic Analysis

Python - Build the Count Matrix

当所有的文章都被解析，所有的索引词都将被提取和保存，接着将构建单词-标题（或文章）矩阵。矩阵的行数等于索引词的数目，矩阵的列数等于文章/标题的数目。最后每个单词-文章的配对值将加载到矩阵相应的单元格中。

```
def build(self):
self.keys = [k for k in self.wdict.keys() if len(self.wdict[k]) > 1]
 self.keys.sort()
 self.A = zeros([len(self.keys), self.dcount])
for i, k in enumerate(self.keys):
 for d in self.wdict[k]:
 self.A[i,d] += 1
```

Python - Print the Count Matrix

printA()方法用于打印创建好的单词-标题（或文章）矩阵

```
def printA(self):
print self.A
```

LSA的实例

An Example of Latent Semantic Analysis

deepshare.net
深度之眼

Python - Test the LSA Class

现在我们用 LSA类来测试前面的9个标题。首先是实例化LSA为mylsa，然后初始化将忽略我们预先定义的停止词和标点，同时出书词典和文章计数变量，接着调用解析方法去解析各标题，从而提出索引词，并对词频进行计数，最后通过build() 方法构建单词-标题（或文章）矩阵。该矩阵过滤掉仅出现在1个标题的单词。

```
mylsa = LSA(stopwords, ignorechars)
for t in titles:
    mylsa.parse(t)
    mylsa.build()
    mylsa.printA()
```

以下是输出结果：

```
[[ 0. 0. 1. 1. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 0. 1. 0. 0. 1.]
 [ 0. 1. 0. 0. 0. 0. 0. 1. 0.]
 [ 0. 0. 0. 0. 0. 0. 1. 0. 1.]
 [ 1. 0. 0. 0. 0. 1. 0. 0. 0.]
 [ 1. 1. 1. 1. 1. 1. 1. 1. 1.]
 [ 1. 0. 1. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 0. 0. 1. 0. 1.]
 [ 0. 0. 0. 0. 0. 2. 0. 0. 1.]
 [ 1. 0. 1. 0. 0. 0. 0. 1. 0.]
 [ 0. 0. 0. 1. 1. 0. 0. 0. 0.]]
```

LSA的实例

deepshare.net
深度之眼

An Example of Latent Semantic Analysis

Part 2 - Modify the Counts with TFIDF

在LSA算法中，源单词-标题（或文章）矩阵一般会进行加权调整，其中稀少的词的的权重会大于一般性的单词。例如一个文章中出现率5%的单词，其权重应大于一个出现率90%的单词。TFIDF 是最常用的度量指标(Term Frequency - Inverse Document Frequency)，公式如下：

$$TFIDF_{ij} = \frac{N_{ij}}{N_{.j}} log\left(\frac{D}{D_i}\right)$$

N_{ij}是词i出现在文章j的次数，也就是源矩阵第ij个元素，$N_{.j}$是出现在文章j中所有索引词出现的次数，也就是源矩阵第j列的求和，D是语料库文章的总数，也就是源矩阵的列数，D_i是语料库文章出现索引词i的文章数，也就是源矩阵i行中非零元素的个数

从公式知，单词的词频越高且包含该单词的文章越少，则相应的TFIDF 值越大。 本例规模小，不对矩阵进行权重调整。

```
def TFIDF(self):
WordsPerDoc = sum(self.A, axis=0)
DocsPerWord = sum(asarray(self.A > 0, 'i'), axis=1)
rows, cols = self.A.shape
for i in range(rows):
 for j in range(cols):
 self.A[i,j] = (self.A[i,j] / WordsPerDoc[j]) * log(float(cols) / DocsPerWord[i])
```

LSA的实例

An Example of Latent Semantic Analysis

Part 3 - Using the Singular Value Decomposition

SVD的强大之处在于通过强调**强的相关关系**并**过滤掉噪声**来实现矩阵降维。换句话说，SVD使用尽可能少的信息来对原矩阵进行尽可能**失真少且噪声少**的重构。实现手段是减低噪声，同时增强强模式和趋势。
在LSA中使用SVD时为了确定单词-标题（或文章）矩阵有效维度数或包含“语义”数。经过压缩后，少量用于有用的维度或语义模式被留下，大量噪声将被过滤掉，这些噪声由作者的随机选择找出。

通过加装python的SVD函数，我们将矩阵分解成3个矩阵。矩阵U提供了每个单词在语义空间的坐标。而V^T提供了每篇文章在语义空间的坐标。奇异值矩阵S告诉我们有词-标题（或文章）矩阵的语义或含量语义空间的有效维度。

```
def calc(self):
self.U, self.S, self.Vt = svd(self.A)
```

LSA的实例

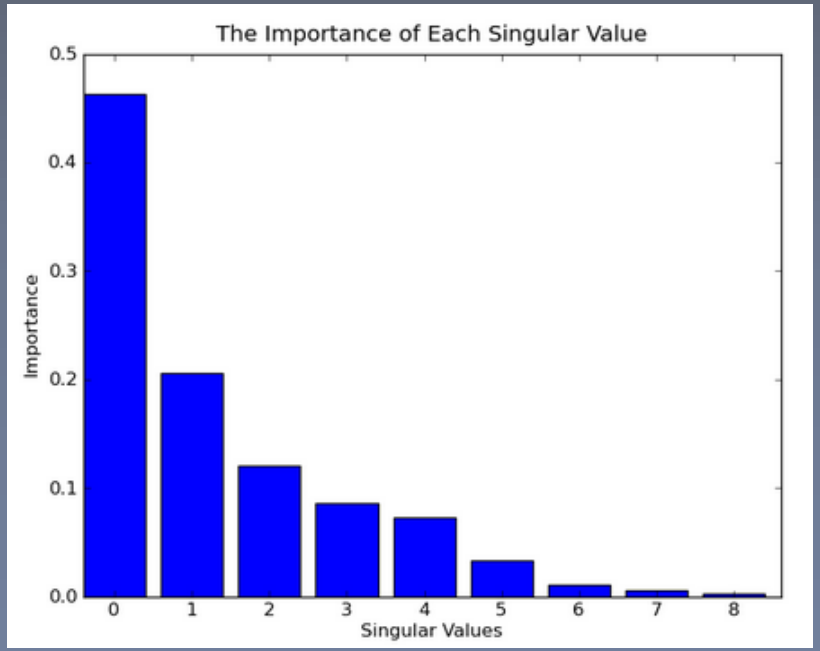
An Example of Latent Semantic Analysis

Part 3 - Using the Singular Value Decomposition

为了确定合适的**有效维度**，通过**奇异值**的平方的直方图来进行观察。右图演示出各奇异值的重要性:

对于大规模的语料库，压缩后的有效一般是100-500维。选择有效维度数为3 。最后将第2和第3维进行可视化。

从文章的角度，第1维表示文章的“长度”，即文章中索引词的数量。从单词的角度，第1维表示该次出现在语料库的次数。如果中心化单词-标题（或文章）矩阵，即每列都减去该列的均值，则可利用第1维进行可视化。



LSA的实例

An Example of Latent Semantic Analysis

Part 3 - Using the Singular Value Decomposition

为了避免将单词-标题（或文章）矩阵由稀疏矩阵变为稠密矩阵，增加内存的负荷和计算量，因此不对单词-标题（或文章）矩阵进行中心化，放弃第1维很高效。

这里计算出了3个奇异值，分别对应3个维度。每个单词与这个3个维度及这些奇异值相关，第1维表示该单词在语料库中的频繁程度，因此信息量不大。类似地，每篇文章也有3个维度分别对3个奇异值。第1维反映了文章所包含索引词的数量，信息量不大。

book	0.15	-0.27	0.04
dads	0.24	0.38	-0.09
dummies	0.13	-0.17	0.07
estate	0.18	0.19	0.45
guide	0.22	0.09	-0.46
investing	0.74	-0.21	0.21
market	0.18	-0.30	-0.28
real	0.18	0.19	0.45
rich	0.36	0.59	-0.34
stock	0.25	-0.42	-0.28
value	0.12	-0.14	0.23

*

3.91	0	0
0	2.61	0
0	0	2.00

*

T1	T2	T3	T4	T5	T6	T7	T8	T9
0.35	0.22	0.34	0.26	0.22	0.49	0.28	0.29	0.44
-0.32	-0.15	-0.46	-0.24	-0.14	0.55	0.07	-0.31	0.44
-0.41	0.14	-0.16	0.25	0.22	-0.51	0.55	0.00	0.34

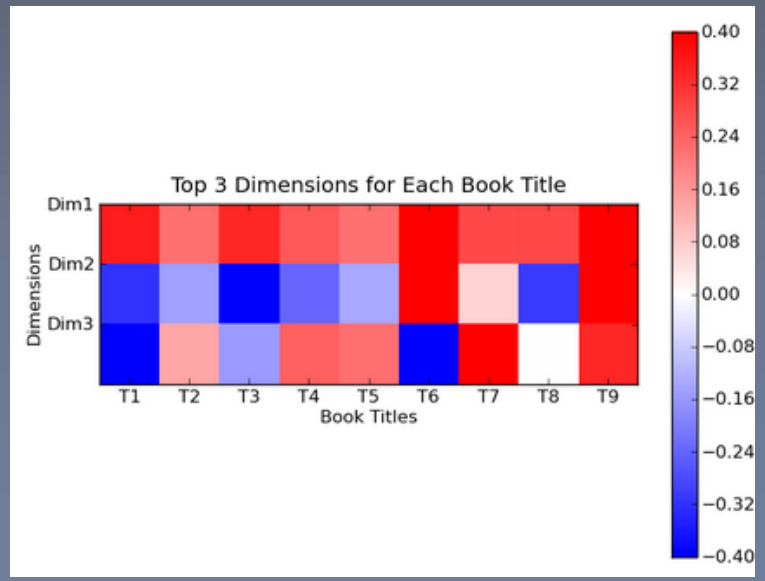
LSA的实例

An Example of Latent Semantic Analysis

Part 4 - Clustering by Color

将数字用不同的颜色进行表示。

例如，用同不的颜色来表示V^T矩阵的值，这个颜色表示的矩阵和原V^T矩阵反映的信息完全一致。蓝色表示负数，红色表示正数，白色表示0。如标题9，其3个维度的上值都正数，因此相应的颜色都是深红色。



LSA的实例

An Example of Latent Semantic Analysis

Part 4 - Clustering by Color

用这些颜色对聚类结果进行颜色标注。忽略第1维表示的颜色，因为所有文章在该维度上都是红色。

Dim2	Titles
red	6-7, 9
blue	1-5, 8

加上第3维，能用相同方法区分出不同的语义群。在第3维上，标题6是蓝色，而标题7和标题9依然是红色。
通过这种方法将标题集分成4个群:

Dim2	Dim3	Titles
red	red	7, 9
red	blue	6
blue	red	2,4，5,8
blue	blue	1,3

LSA的实例

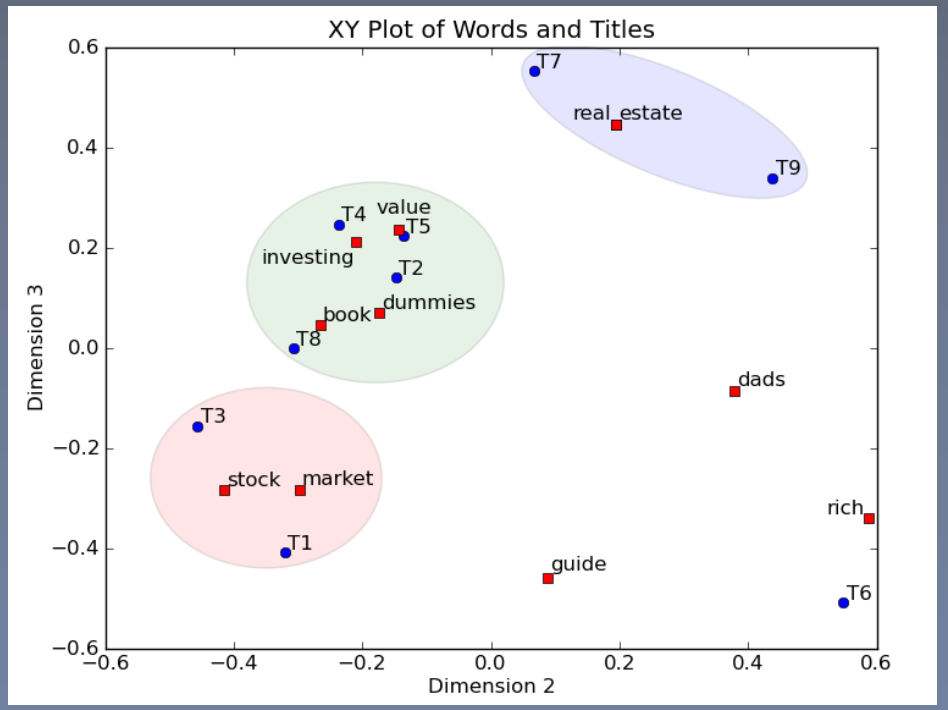
An Example of Latent Semantic Analysis

Part 5 - Clustering by Value

将矩阵U和V的第2,3维画在一个2维XY平面中，其中X表示第2维，Y表示第3维，并将所有索引词和标题画在该平面中。如图所示，单词“book”的坐标值为 (0.15, -0.27, 0.04)，忽略第一维的值 0.15 后，“book” 的坐标点为 (x = -0.27, y = 0.04)。标题的画法类似。

这种可视化方法的优点在于将单词和标题都画在同一个空间，能实现**标题的聚类**，还能通过索引词标注出**不同类簇的意义**。

例如，左下的簇包含含量了标题1和标题2，这两个标题均关于stock market investing。单词”stock”和”market”明显包含在标题1和标题2的簇中，这也很容易地理解这个语义簇所指代的意义。中间的簇包含了标题2,4,5,8。其中标题2,4,5与单词”value” 和”investing”代表的意思最为接近，因此 标题2,4,5 的语义可表示为：”value” 和”investing”



LSA的优劣与应用

Advantages, Disadvantages, and Applications of LSA

LSA的优势

1.文章和单词都映射到同一个语义空间。在该空间内即能对文章和单词进行聚类。能通过这些聚类结果实现基于单词的文献检索，反之亦然。

2.语义空间的维度明显少于源单词-文章矩阵。经过特定方式组合而成维度包含源矩阵的大量信息，同时降低了噪声影响，有助于后续其他算法的加工处理。

3.LSA 是一个全局最优化算法，其目标是寻找全局最优解而非局部最优解，因此它能求出基于局部求解算法得不到的全局信息。有时LSA会结合一些局部算法，如最近领域法，使得LSA性能得到进一步提升。

LSA的缺陷

1.LSA是假设服从高斯分布和2范数规范化的，因此它并非适合于所有场景。例如，单词在语料库中服从的是Poisson 分布而不是高斯分布。

2.LSA不能有效处理一词多义问题。因为LSA的基本假设之一是单词只有一个词义。

3.LSA的核心是SVD，而SVD的计算复杂度十分高并且难以更新新出现的文献。但最近已出现一些有效的方法用于解决SVD的基于文献更新问题。

LSA的应用

LSA被广泛用于文献检索，文本分类，垃圾邮件过滤，语言识别，模式检索以及文章评估自动化等场景。

结 语

在这次课程中，我们了解到了LSA案例：实现标题的聚类，TFIDF指标，Python编程实现语义分析

希望在课下，大家都能

掌握本节知识点并完成作业

deepshare.net

深度之眼

联系我们：

电话：18001992849

邮箱：service@deepshare.net

Q Q： 2677693114

公众号

客服微信

关注公众号深度之眼，后台回复 统计学 ， 获取统计学习方法第二版电子书及其他AI必学书籍