Lecture 6

Recommender Systems

Personalization, Collaborative Filtering & Content-based recommendation

个性化、协同过滤和基于内容的推荐。

COMP 474/6741, Winter 2021

René Witte

René Witte
Department of Computer Science
and Software Engineering
Concordia University

Outline

1 **Introduction**
   Modeling Users

2 **Collaborative Filtering**
   Introduction
   Computing with Words
   Item Recommendation
   Items Related to other Items
   Items of Interest to a User
   Relevant Users for an Item
   Semantic User Profiles
   Evaluation

3 **Content-based Recommendations**
   Motivation
   TF*IDF weighting
   Term Vector Space Model
   Summary

4 **Notes and Further Reading**

René Witte

Concordia

Slides Credit

Includes slides by Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze [MRS08]

- Copyright © 2008 Cambridge University Press

Recommender Systems and Collaborative Filtering

René Witte

Concordia University

Collecting User Interactions

Purchase, Contribute, Recommend, View, Tag, Rate, Save, Bookmark

Item Metadata

Copyright 2009 by Manning Publications Co., [Ala09]

Netflix Recommendations

Why Netflix's Algorithm Is So Binge-Worthy | Mach | NBC News

https://www.youtube.com/watch?v=nq2QtatuF7U

1. **Introduction**

2. **Collaborative Filtering**
   Introduction
   Computing with Words
   Item Recommendation
   Items Related to other Items
   Items of Interest to a User
   Relevant Users for an Item
   Semantic User Profiles
   Evaluation

3. **Content-based Recommendations**

4. **Notes and Further Reading**

René Witte

Making Recommendations

Given Information about a User...

...we want to be able to have a system

- recommending items (books, movies, music, photos, videos, etc.)
- find users interested in a new item
- find similar items, based on interests of *other* users

Customers who bought this item also bought

Hands-On Unsupervised Learning Using Python: How to Build Applied...
Ankur A. Patel
★★★★☆ 2
Paperback
CDN$55.67
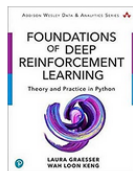
Foundations of Deep Reinforcement Learning: Theory and Practice in...
Laura Graesser
★★★★★ 1
Paperback
CDN$48.59
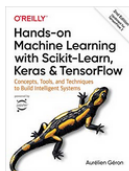
Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow:...
Aurélien Géron
★★★★★ 13
Paperback
CDN$69.16
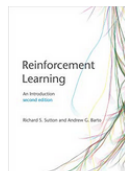
Reinforcement Learning: An Introduction
Richard S. Sutton
★★★★★ 9
Hardcover
CDN$86.18

Practical Time Series Analysis: Prediction with Statistics and Machine...
Aileen Nielsen
★★★☆☆ 1
Paperback
CDN$42.60

Collaborative Filtering

René Witte

Concordia
UNIVERSITY

Introduction
  Modeling Users
Collaborative Filtering
  Introduction
  Computing with Words
  Item Recommendation
  Items Related to other Items
  Items of Interest to a User
  Relevant Users for an Item
  Semantic User Profiles
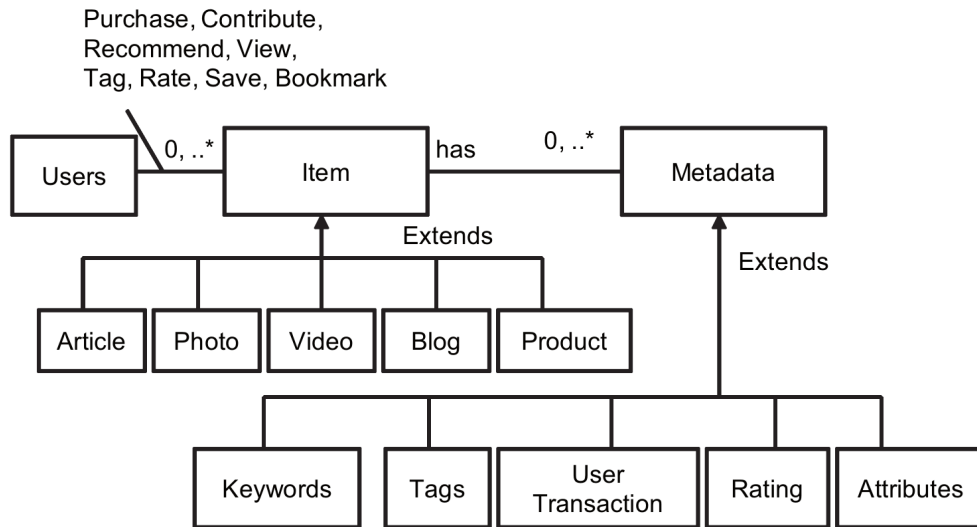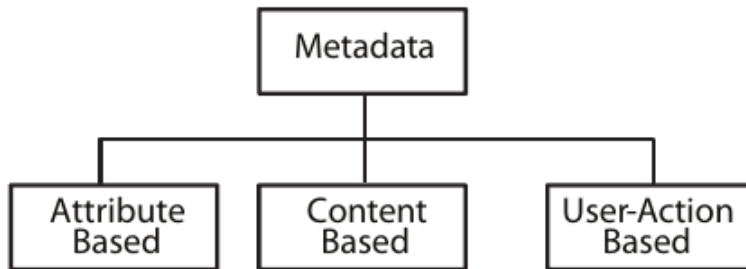 Evaluation
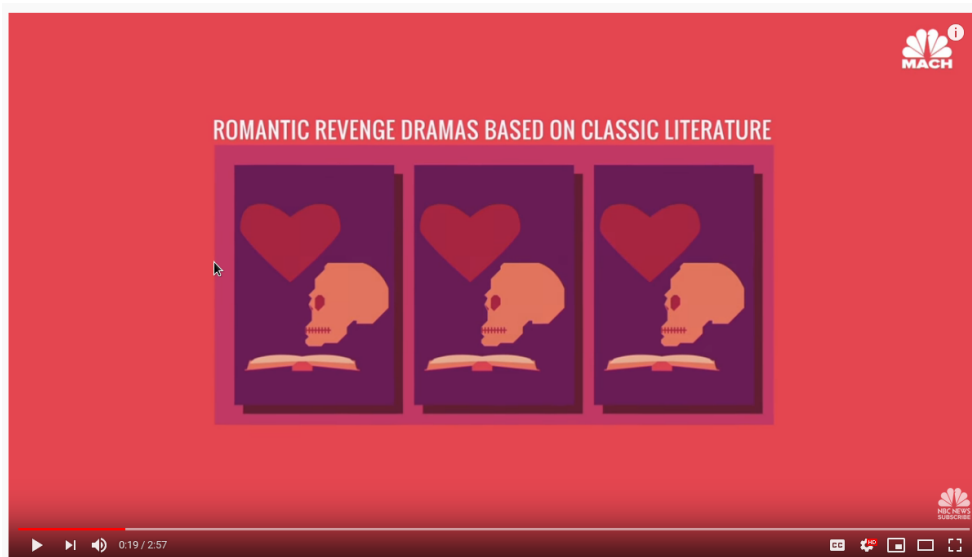Content-based Recommendations
  Motivation
  TF*IDF weighting
  Term Vector Space Model
  Summary
Notes and Further Reading

Data Collection

Date	User	Item
2015-01-24 15:01:29	Allison	Tunisia Sadie dress
2015-01-26 05:13:58	Christina	Gordon Monk stiletto
2015-02-18 10:28:37	David	Ravelli aluminum tripod
2015-03-17 14:29:23	Frank	Nikon digital camera
2015-03-26 18:11:01	Christina	Georgette blouse
2015-04-06 21:50:18	David	Canon 24 mm lens
2015-04-15 10:21:44	Frank	Canon 24 mm lens
2015-04-15 21:53:25	Brenda	Tunisia Sadie dress
2015-07-26 08:08:25	Elise	Nikon digital camera

Fun with ~~Flags~~ Vectors

Vectors

A vector $\vec{v}$ is an element of a vector space. 矢量$\vec{v}$是一个矢量空间的元素。

- For example, $\vec{v} \in \mathbb{R}^n$ with

$$\vec{v} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^n$$

Visualization

We can visualize vectors, e.g., in 2D:
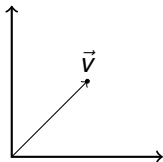
So what?

Vectors of words, users, products, . . .

We can represent (users, documents, products) as vectors, e.g., using the count of tags or the weight of words. This is called a vector space model.

- Vector operations on entities, e.g., to compute their similarity

Copyright 2008 by Cambridge University Press, [MRS08]

Movies as Vectors

The Little Mermaid is a cartoon but has no aliens or basketball.

Space Jam ranks highly in the dimensions of Cartoon-ness, Alien-ness, and Basketball-ness.

Cartoon-ness

Basketball-ness

The Godfather has nothing to do with cartoons, aliens, or basketball.

Aliens is all about aliens but has no basketball and is not a cartoon.

Alien-ness

Length normalization
归一化

René Witte

Concordia
UNIVERSITY

Introduction
Modeling Users
Collaborative Filtering
Introduction
Computing with Words
Item Recommendation
Items Related to other Items
Items of Interest to a User
Relevant Users for an Item
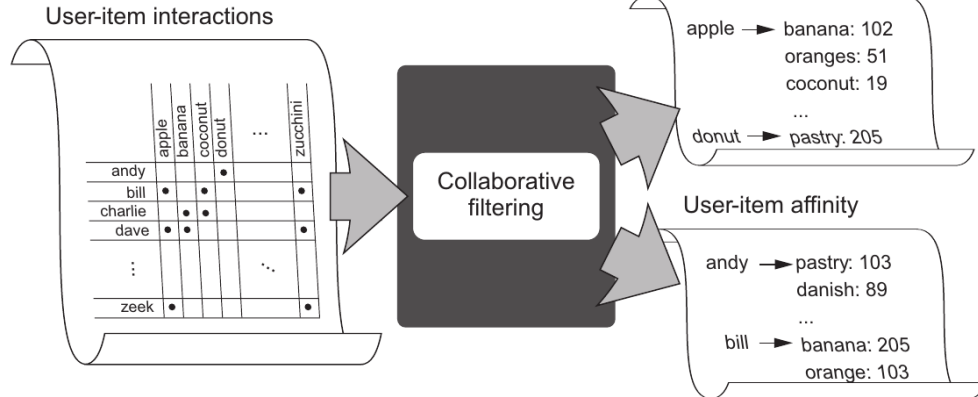Semantic User Profiles
Evaluation
Content-based Recommendations
Motivation
TF*IDF weighting
Term Vector Space Model
Summary
Notes and Further Reading
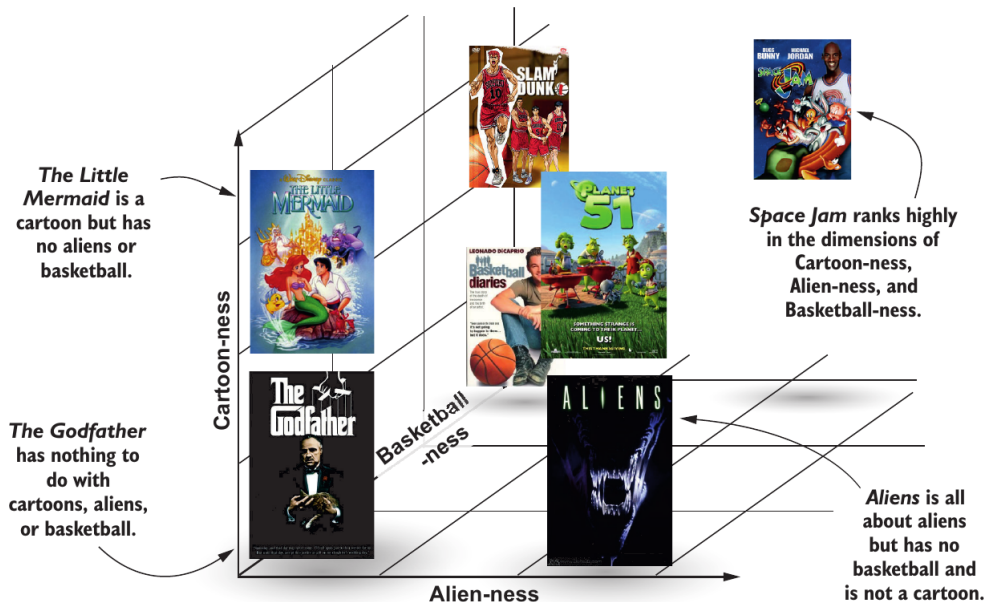
How do we compute the length of a vector?

- A vector can be (length-) normalized by dividing each of its components by its length – here we use the L_2 norm: $||x||_2 = \sqrt{\sum_i x_i^2}$

- This maps vectors onto the unit sphere ...

- ...since after normalization: $||x||_2 = \sqrt{\sum_i x_i^2} = 1.0$

- As a result, longer and shorter vectors (more/fewer tags) have weights of the same order of magnitude.

$\rightarrow$ **Worksheet #5: Tasks 1, 2**

How do we formalize vector space similarity?

René Witte

Computing the similarity

- <u>First cut:</u> (negative) distance between two points
- (= distance between the end points of the two vectors)
- Euclidean distance?
- Euclidean distance is a bad idea . . .
- . . . because Euclidean distance is large for vectors of different lengths.

Why Euclidian distance is a bad idea

René Witte

Concordia

Introduction
  Modeling Users
Collaborative Filtering
  Introduction
  Computing with Words
  Item Recommendation
  Items Related to other Items
  Items of Interest to a User
  Relevant Users for an Item
  Semantic User Profiles
  Evaluation
Content-based Recommendations
  Motivation
  TF*IDF weighting
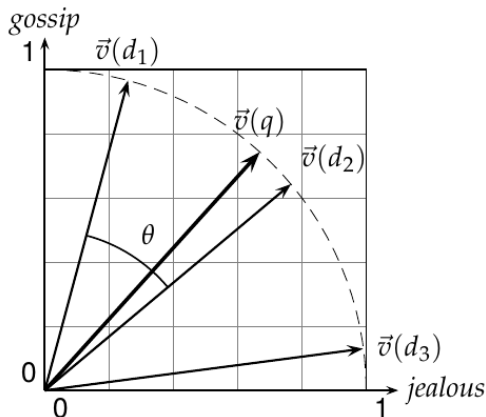 Term Vector Space Model
  Summary
Notes and Further Reading
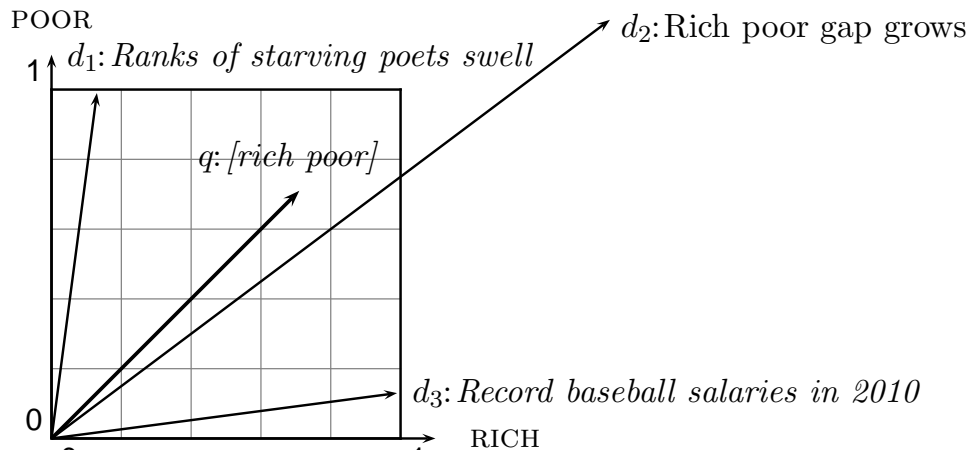
The Euclidean distance of $\vec{q}$ and $\vec{d_2}$ is large although the distribution of terms in the query q and the distribution of terms in the document d_2 are very similar.

虽然查询q中的术语分布和文档d2中的术语分布非常相似，但q和d2的欧氏距离很大。

From angles to cosines

Comparing vectors

- The following two notions are equivalent.
  - Compare item vectors according to the angle between them, in decreasing order
  - Rank item vectors according to cosine(item$_1$, item$_2$) in increasing order
- Cosine is a monotonically decreasing function of the angle for the interval $[0°, 180°]$

Cosine
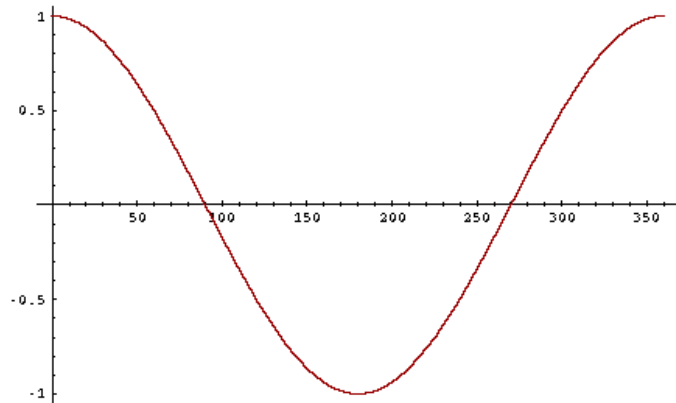
Cosine for normalized vectors

Computing similarity

- For normalized vectors, the cosine is equivalent to the dot product or scalar product.

- $\cos(\vec{q}, \vec{d}) = \vec{q} \cdot \vec{d} = \sum_i q_i \cdot d_i$
  - (if $\vec{q}$ and $\vec{d}$ are length-normalized).

→ **Worksheet #5: Task 3**

Item Recommendation

Simple Tag-based Recommendation

Collaborative tagging gives rise to simple recommender approaches:

- show other items (products, photos, videos, music) that were tagged similar by other users 显示被其他用户标记为类似的其他物品（产品、照片、视频、音乐）。
- exploited in many e-commerce/social networking web sites

Tags Customers Associate with This Product (What's this?)
Click on a tag to find related items, discussions, and people.

tagging (20)	social software (7)	semantic web (3)
metadata (16)	findability (6)	interaction design (2)
folksonomy (15)	ia (4)	
information architecture (11)	social media (4)	See all 28 tags…

Search Products Tagged with

[   ] (GO)

›See most popular Tags

Your tags: Add your first tag

Collaborative Filtering

Finding related content

When multiple users tag the same resource, content can be discovered based on the most frequent tags (example: Last.fm).

Tags

+ Add Tags

folk

Including Bob Dylan, Johnny Cash and Iron & Wine

singer-songwriter
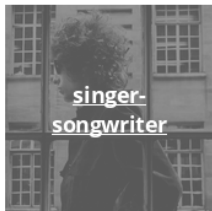
Including Bob Dylan, Tom Waits and Elliott Smith

classic rock
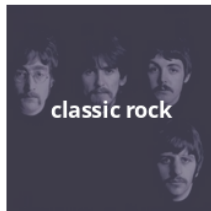
Including The Beatles, Led Zeppelin and Pink Floyd

folk rock

Including Bob Dylan, Jethro Tull and Neil Young

60s

rock

blues

bob dylan
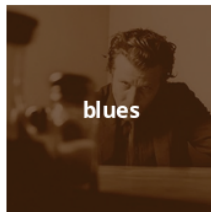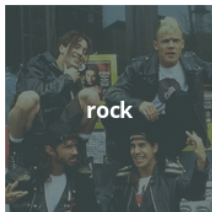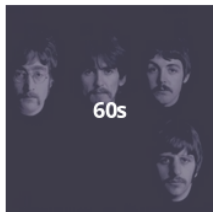
Recommendations

Recommendations based on tags

We can now exploit tags for a number of use cases:

- Recommend items related to other items
- Recommend items based on user's interest
- Find users interested in a new item

General Approach

- Represent users/items as (normalized) term vectors
- Compute cosine similarity between vectors; i.e., the angle between them (for normalized vectors, this is simply their dot product)
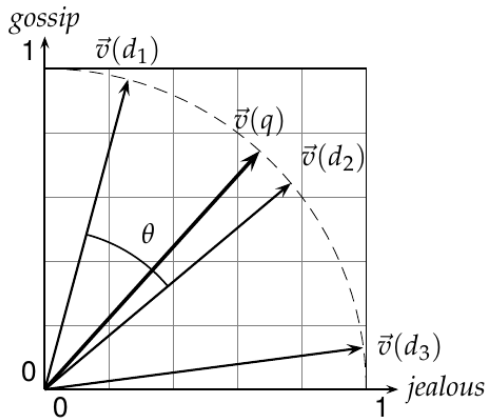
Items related to other items

Simple point-to-point recommendation engine

- Create item vectors using raw count
- Normalize vectors
- Compute cosine similarity

Result is a similarity matrix

Items of interest to a user

Personalization

- Item-to-item is the same for all users
- How can we recommend items for a particular user?

Solution: build user-specific similarity matrix

- computation of vectors, normalization as before
- this time, we calculate the cosine similarity between a user vector and article vector

→ Worksheet #5: Tasks 4, 5

Finding relevant users for an item

René Witte

Concordia
UNIVERSITY

Introduction
 Modeling Users

Collaborative Filtering
 Introduction
 Computing with Words
 Item Recommendation
 Items Related to other
 Items
 Items of Interest to a User
 Relevant Users for an Item
 Semantic User Profiles
 Evaluation

Content-based
Recommendations
 Motivation
 TF*IDF weighting
 Term Vector Space Model
 Summary

Notes and Further
Reading

Recommending items to users

- New item comes in (blog post, photo, article, product, . . .)
- Which users would be interested in it?

Similar to before, compute similarity matrix between metadata of new item and metadata of users.

Cold-Start Problem

General issue in recommender system deployment

- New user $\Rightarrow$ no user profile for recommendations
- New item $\Rightarrow$ no user interactions for this item

No general solution...

Some strategies:

- Ask user for preferences during sign-up
- Recommend top-n items (e.g., currently most popular movies/songs/products)

Semantic Vocabularies for User Modeling

Semantic User Profiles
Idea: Use vocabularies instead of keywords in the vector representation of a user profile
在用户资料的向量表示中使用词汇表而不是关键词。

Motivation

- Semantic recommendations (remember the "tree" example)
- Open knowledge bases:
  - interoperable between applications
  - controlled by users, not corporations

Vocabularies

Generic user modeling vocabularies

FOAF

- The most popular generic user model offering descriptions for basic user information
- No comprehensive classes for describing preferences or interests

GUMO

- A generic user model that offers several classes for users' characteristics
- Basic user dimensions like Emotional States, Characteristics and Personality

IntelLEO

- Several ontologies strongly focused on personalization
- Enables describing user and team modelling, preferences, tasks and interests

The $1m Netflix Prize Competition

6.30

General machine learning process

Performance Evaluation

Measuring performance

- Is our fancy model better than giving out random recommendations?
- We need metrics to evaluate and compare the performance of different approaches

Precision and Recall

https://www.jianshu.com/p/54ed63a7f816

The precision provides a measure of the quality of the generated recommendations:

$$\text{precision} = \frac{\#\text{relevant system recommendations}}{\#\text{all generated recommendations}}$$

The recall indicates how many relevant recommendations were found by a system:

$$\text{recall} = \frac{\#\text{correctly found recommendations}}{\#\text{all relevant recommendations}}$$

Generally, there is a trade-off between precision and recall.

→ **Worksheet #5: Task 6**

Precision@*k*

Precision at *cutoff k*

- Return a ranked list of recommendations (based on cosine similarity)
- Evaluate only top-*k* recommendations (e.g., top-10)

$$\text{precision@k} = \frac{1}{k} \cdot \sum_{c=1}^{k} \text{rel}(c),$$

where $\text{rel}(c)$ tells us if item at rank c was relevant (1) or not (0).

Intuitively...
The percentage of correct recommendations in the top-*k*.

Wait, what happened to *Recall*?
Well...in this application scenario, we don't really care (there are millions of potentially relevant items on Amazon or movies on Netflix)

→ **Worksheet #5: Task 7**

Average Precision

Average Precision at *N*

If we recommend *N* items to a user, where there are at most *m* relevant items in $1 \ldots N$,

$$\text{AP@N} = \frac{1}{m} \sum_{k=1}^{N} \text{precision@k} \cdot \text{rel}(k)$$

again, rel(*c*) is 1 if the recommendation at rank *c* is relevant, 0 otherwise

Note

AP "rewards" (gives a higher score to) higher-ranked, correct recommendations

→ **Worksheet #5: Task 8**

Mean Average Precision (MAP)

René Witte

Concordia
UNIVERSITY

Introduction
Modeling Users

Collaborative Filtering
Introduction
Computing with Words
Item Recommendation
Items Related to other Items
Items of Interest to a User
Relevant Users for an Item
Semantic User Profiles
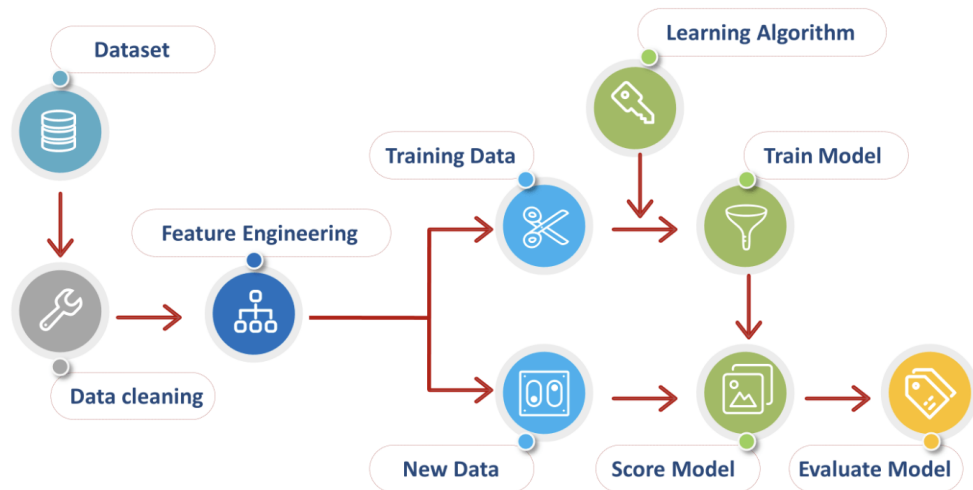Evaluation

Content-based Recommendations
Motivation
TF*IDF weighting
Term Vector Space Model
Summary

Notes and Further Reading

MAP

- So far, everything was calculated for one user $u \in U$
- But we want to know how well the system works across all users
- Hence, average the AP for all users:

$$\text{MAP@N} = \frac{1}{|U|} \sum_{u=1}^{|U|} \text{AP@N}(u)$$

But wait, there's more. . .

- Accuracy, Sensitivity, F-measure, . . .
- Non-binary ranked results (i.e., not just correct or wrong, but a *Likert*-scale): Compute the *discounted cumulative gain* (DCG),

$$\text{DCG}_u = rel_1 + \sum_{c=2}^{|C|} \frac{rel_c}{\log_2 c}$$

1 **Introduction**

2 **Collaborative Filtering**

3 **Content-based Recommendations**
    Motivation
    TF*IDF weighting
    Term Vector Space Model
    Summary

4 **Notes and Further Reading**

Content-based Recommendations

Motivation

- So far, we build our model using vectors of concepts (e.g., tags, movie categories, etc.)
- What if we want to create recommendations based on the content
  - Movie description/summary
  - Blog post
  - News article
  - Research publication
  - …

Approach

Same idea, but now we have to build vectors out of whole documents

- Basic idea of information retrieval (IR)
- Used in Internet search engines

Binary incidence matrix

	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth	...
ANTHONY	1	1	0	0	0	1	
BRUTUS	1	1	0	1	0	0	
CAESAR	1	1	0	1	1	1	
CALPURNIA	0	1	0	0	0	0	
CLEOPATRA	1	0	0	0	0	0	
MERCY	1	0	1	1	1	1	
WORSER	1	0	1	1	1	0	
...							

Each document is represented as a binary vector $\in \{0, 1\}^{|V|}$.

[from *Introduction to Information Retrieval*]

Count matrix

	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth	...
ANTHONY	157	73	0	0	0	1	
BRUTUS	4	157	0	2	0	0	
CAESAR	232	227	0	2	1	0	
CALPURNIA	0	10	0	0	0	0	
CLEOPATRA	57	0	0	0	0	0	
MERCY	2	0	3	8	5	8	
WORSER	2	0	1	1	1	5	
...							

Each document is now represented as a count vector $\in \mathbb{N}^{|V|}$.

Bag of words model

- We do not consider the order of words in a document.
- *John is quicker than Mary* and *Mary is quicker than John* are represented the same way.
- This is called a bag of words model.

Term frequency tf

The term frequency $\text{tf}_{t,d}$ of term t in document d is defined as the number of times that t occurs in d.

Frequency in document vs. frequency in collection

- In addition, to term frequency (the frequency of the term in the document) . . .
- . . . we also want to use the frequency of the term in the collection for weighting and ranking.
- Rare terms are more informative than frequent terms.
  - Consider a term in the query that is rare in the collection (e.g., ARACHNOCENTRIC).
  - A document containing this term is very likely to be relevant.
  - → We want high weights for rare terms like ARACHNOCENTRIC.

Desired weight for frequent terms

Weighting scheme

- Frequent terms are less informative than rare terms.
- Consider a term in the query that is frequent in the collection (e.g., GOOD, INCREASE, LINE).
- A document containing this term is more likely to be relevant than a document that doesn't . . .
- . . . but words like GOOD, INCREASE and LINE are not sure indicators of relevance.
- → For frequent terms like GOOD, INCREASE, and LINE, we want positive weights . . .
- . . . but lower weights than for rare terms.

Document Frequency

Document Frequency (df)

- We want high weights for rare terms like ARACHNOCENTRIC.

- We want low (positive) weights for frequent words like GOOD, INCREASE, and LINE.

- We will use document frequency to factor this into computing the matching score.

- The document frequency is the number of documents in the collection that the term occurs in.

idf weight

inverse document frequency (idf)

- df_t is the document frequency, the number of documents that t occurs in.
- df_t is an inverse measure of the informativeness of term t.
- We define the idf weight of term t as follows: dft是术语t的信息量的反向度量。

$$\text{idf}_t = \log_{10} \frac{N}{df_t}$$

  (N is the number of documents in the collection.)
- idf_t is a measure of the informativeness of the term.
- $[\log N/df_t]$ instead of $[N/df_t]$ to "dampen" the effect of idf
- Note that we use the log transformation for both term frequency and document frequency.

Examples for idf

Compute idf_t using the formula: $\text{idf}_t = \log_{10} \frac{1{,}000{,}000}{\text{df}_t}$

此单词只出现在一个document中

term	df_t	idf_t
calpurnia	1	6
animal	100	4
sunday	1000	3
fly	10,000	2
under	100,000	1
the	1,000,000	0

Effect of idf on ranking

- idf affects the ranking of documents for queries with at least two terms.
- For example, in the query "arachnocentric line", idf weighting increases the relative weight of ARACHNOCENTRIC and decreases the relative weight of LINE.
- idf has little effect on ranking for one-term queries.

tf-idf weighting

Computing tf-idf

- The tf-idf weight of a term is the product of its tf weight and its idf weight:

$$w_{t,d} = (1 + \log \mathrm{tf}_{t,d}) \cdot \log \frac{N}{\mathrm{df}_t}$$

- Best known weighting scheme in information retrieval
- Note: the "-" in tf-idf is a hyphen, not a minus sign!
- Alternative names: tf.idf, tf x idf

Summary: tf-idf

- Assign a tf-idf weight for each term t in each document d:
$$w_{t,d} = (1 + \log \text{tf}_{t,d}) \cdot \log \frac{N}{\text{df}_t}$$

- The tf-idf weight ...
  - ... increases with the number of occurrences within a document. (term frequency)
  - ... increases with the rarity of the term in the collection. (inverse document frequency)

$\rightarrow$ **Worksheet #5: Task 9**

Binary incidence matrix

	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth	...
ANTHONY	1	1	0	0	0	1	
BRUTUS	1	1	0	1	0	0	
CAESAR	1	1	0	1	1	1	
CALPURNIA	0	1	0	0	0	0	
CLEOPATRA	1	0	0	0	0	0	
MERCY	1	0	1	1	1	1	
WORSER	1	0	1	1	1	0	
...							

Each document is represented as a binary vector $\in \{0, 1\}^{|V|}$.

[from *Introduction to Information Retrieval*]

Count matrix

	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth	...
ANTHONY	157	73	0	0	0	1	
BRUTUS	4	157	0	2	0	0	
CAESAR	232	227	0	2	1	0	
CALPURNIA	0	10	0	0	0	0	
CLEOPATRA	57	0	0	0	0	0	
MERCY	2	0	3	8	5	8	
WORSER	2	0	1	1	1	5	
...							

Each document is now represented as a count vector $\in \mathbb{N}^{|V|}$.

Binary → count → weight matrix

	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth	...
ANTHONY	5.25	3.18	0.0	0.0	0.0	0.35	
BRUTUS	1.21	6.10	0.0	1.0	0.0	0.0	
CAESAR	8.59	2.54	0.0	1.51	0.25	0.0	
CALPURNIA	0.0	1.54	0.0	0.0	0.0	0.0	
CLEOPATRA	2.85	0.0	0.0	0.0	0.0	0.0	
MERCY	1.51	0.0	1.90	0.12	5.25	0.88	
WORSER	1.37	0.0	0.11	4.15	0.25	1.95	
...							

Each document is now represented as a real-valued vector of tf-idf weights $\in \mathbb{R}^{|V|}$.

Vector Space Model

Documents as vectors

- Each document is now represented as a real-valued vector of tf-idf weights $\in \mathbb{R}^{|V|}$.

- So we have a $|V|$-dimensional real-valued vector space.

- Terms are axes of the space.

- Documents are points or vectors in this space.

- Very high-dimensional: tens of millions of dimensions when you apply this to web search engines

- Each vector is very sparse – most entries are zero.

Queries as vectors

- Key idea 1: do the same for queries: represent them as vectors in the high-dimensional space
- Key idea 2: Rank documents according to their proximity to the query
- proximity = similarity
- proximity $\approx$ negative distance
- Recall: We're doing this because we want to get away from the you're-either-in-or-out, feast-or-famine Boolean model.
- Instead: rank relevant documents higher than nonrelevant documents

回顾一下。我们之所以这么做 是因为我们想摆脱非进即退、非快即退的布尔模型。
- 取而代之的是：将相关文档排在比非相关文档更高的位置。

Cosine similarity between query and document

René Witte

Concordia
UNIVERSITY

Introduction
Modeling Users

Collaborative Filtering
Introduction
Computing with Words
Item Recommendation
Items Related to other Items
Items of Interest to a User
Relevant Users for an Item
Semantic User Profiles
Evaluation

Content-based Recommendations
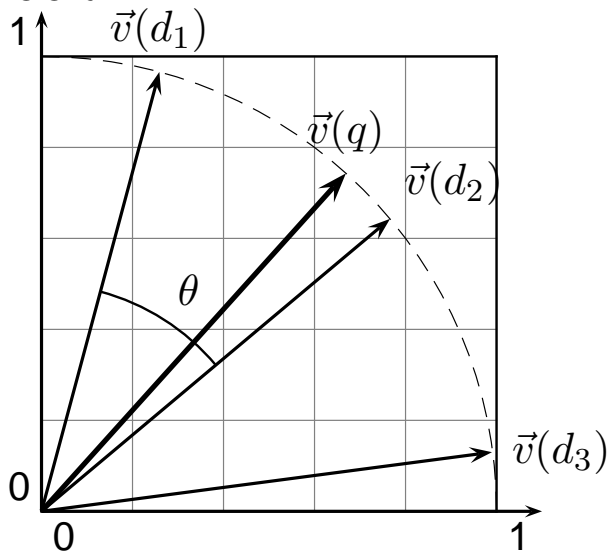Motivation
TF*IDF weighting
Term Vector Space Model
Summary

Notes and Further Reading

$$\cos(\vec{q}, \vec{d}) = \text{SIM}(\vec{q}, \vec{d}) = \frac{\vec{q} \cdot \vec{d}}{|\vec{q}||\vec{d}|} = \frac{\sum_{i=1}^{|V|} q_i d_i}{\sqrt{\sum_{i=1}^{|V|} q_i^2} \sqrt{\sum_{i=1}^{|V|} d_i^2}}$$

- q_i is the tf-idf weight of term i in the query.
- d_i is the tf-idf weight of term i in the document.
- $|\vec{q}|$ and $|\vec{d}|$ are the lengths of $\vec{q}$ and $\vec{d}$.
- This is the cosine similarity of $\vec{q}$ and $\vec{d}$ . . . . . . or, equivalently, the cosine of the angle between $\vec{q}$ and $\vec{d}$.

Cosine similarity illustrated

René Witte

Concordia
UNIVERSITÉ

Introduction
Modeling Users

Collaborative Filtering
Introduction
Computing with Words
Item Recommendation
Items Related to other Items
Items of Interest to a User
Relevant Users for an Item
Semantic User Profiles
Evaluation

Content-based Recommendations
Motivation
TF*IDF weighting
Term Vector Space Model
Summary

Notes and Further Reading

Basic Recommender Engine using Vector Space Model

Approach

- Represent all documents (movie descriptions, blog posts, research articles, . . .) as a weighted tf-idf vector
- Compute the cosine similarity between the target vector and each document vector
- Rank documents with respect to the target
- Return the top k (e.g., $k = 10$) to the user

  - 将所有文档（电影描述、博客文章、研究文章……）表示为一个加权的tf-idf向量。
  - 计算目标向量和每个文档向量之间的余弦相似性。
 - 根据目标对文件进行排序
  - 将前k名（如k=10）返回给用户。

Summary

Vector Space Model

- A mathematical model to portray an *n*-dimensional space
- Entities are described by vectors with *n* coordinates in a real space $\mathbb{R}^n$
- Given two vectors, we can compute a similarity coefficient between them
- Cosine of the angle between two vectors reflects their degree of similarity

$$tf = 1 + \log(tf_{t,d}) \tag{1}$$

$$idf = \log \frac{N}{df_t} \tag{2}$$

$$cos(\vec{q}, \vec{d}) = \frac{\sum_{i=1}^{|v|} q_i \cdot d_i}{\sqrt{\sum_{i=1}^{|v|} q_i{}^2} \cdot \sqrt{\sum_{i=1}^{|v|} d_i{}^2}} \tag{3}$$
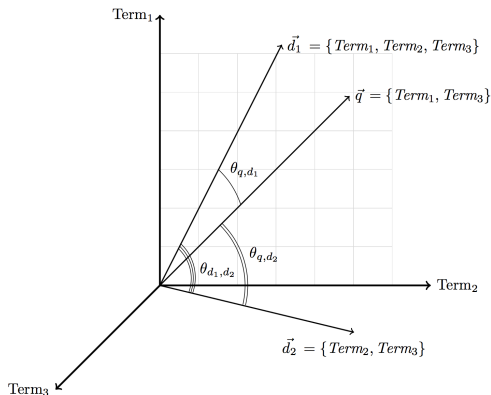
Ultra-fast Search Library and Server

◄ ‖ ►

Apache Lucene and Solr set the standard for search
and indexing performance

Welcome to Apache Lucene

DOWNLOAD

Apache Lucene 8.4.1

The Apache Lucene[TM] project develops open-source search software, including:

- Lucene Core, our flagship sub-project, provides Java-based indexing and search technology, as
  well as spellchecking, hit highlighting and advanced analysis/tokenization capabilities.

DOWNLOAD

Outline

1. **Introduction**

2. **Collaborative Filtering**

3. **Content-based Recommendations**

4. **Notes and Further Reading**

Reading Material

Required

- [Ala09, Chapters 2, 3] (Recommendations)
- [MRS08, Chapter 8] (Evaluation)

Supplemental

- [MRS08, Chapter 6] (Vector Space Model, tf-idf)

References

René Witte

Concordia

Introduction
Modeling Users

Collaborative Filtering
Introduction
Computing with Words
Item Recommendation
Items Related to other Items
Items of Interest to a User
Relevant Users for an Item
Semantic User Profiles
Evaluation

Content-based Recommendations
Motivation
TF*IDF weighting
Term Vector Space Model
Summary

Notes and Further Reading

[Ala09]   Satnam Alag.
Collective Intelligence in Action.
Manning, 2009.
https://concordiauniversity.on.worldcat.org/oclc/314121652.

[MRS08]   Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze.
Introduction to Information Retrieval.
Cambridge University Press, 2008.
http://informationretrieval.org.

[TB16]   Doug Turnbull and John Berryman.
Relevant Search.
Manning, 2016.
https://concordiauniversity.on.worldcat.org/oclc/954339855.