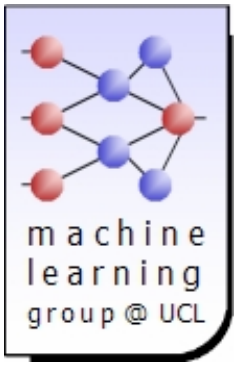
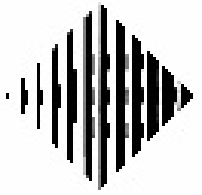
# Recommender systems: an overview

François Fouss and Marco Saerens
{francois.fouss,marco.saerens}@ucLouvain.be
Université catholique de Louvain (UCL)
Institut d'Administration et de Gestion (IAG)
Unité de Systèmes d'Information (ISYS)



UCL Université catholique de Louvain

Université partenaire de l'Académie universitaire 'Louvain'

# Outline

- Introduction
- Typology of recommender systems
  - Preference indicators
  - Recommendation approach
  - Recommendation technique
  - Direct or indirect method
  - Provided results
- Algorithms
  - Feature-analysis based
  - Link-analysis based
- Validation
  - Cross-validation
  - Performance measures
- Experimental results

# Outline

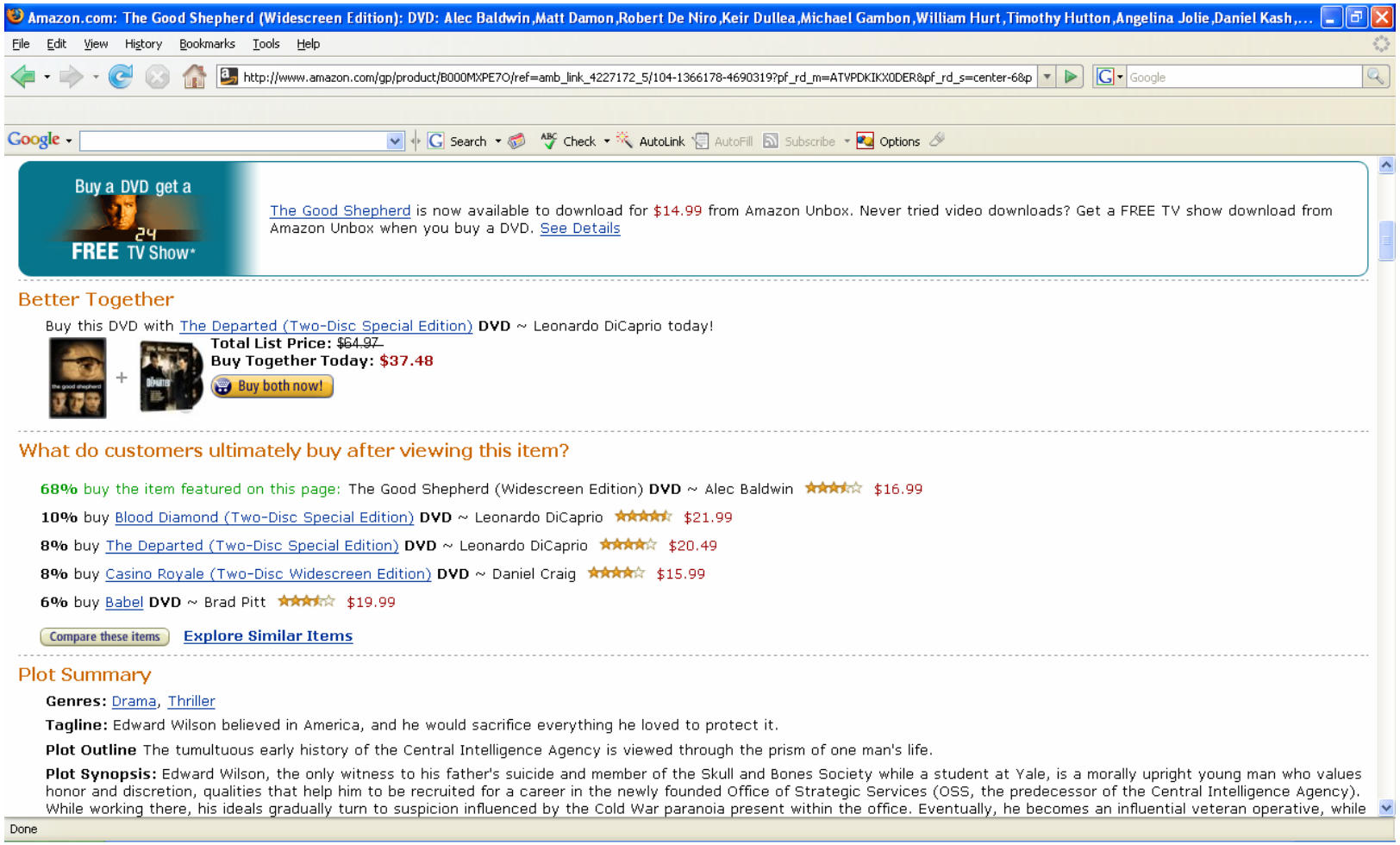
- **Introduction**
- Typology of recommender systems
- Algorithms
- Validation
- Experimental results

# Definition

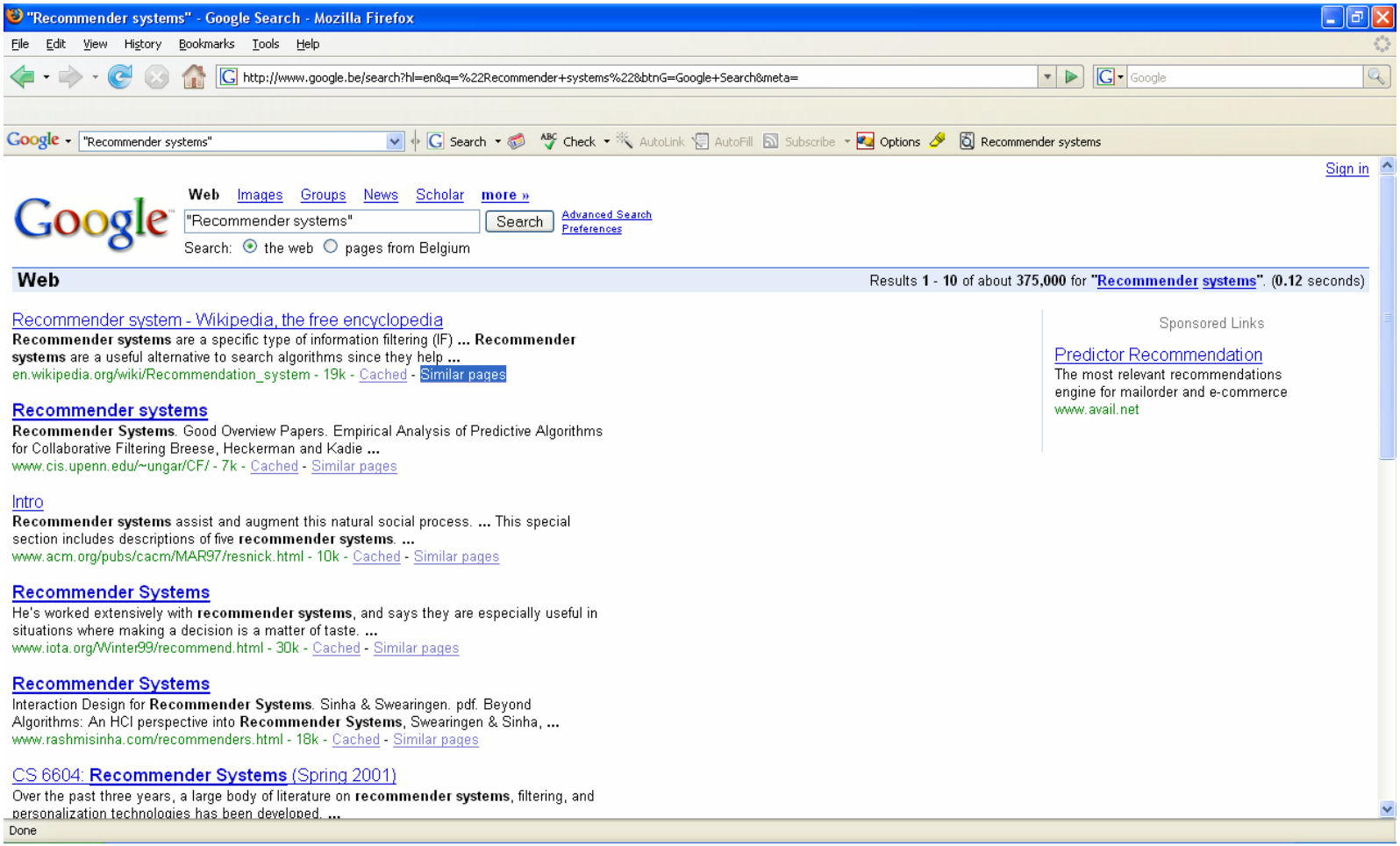
« Recommender systems try to provide people with recommendations of items they will appreciate, based on their past preferences, history of purchase, and demographic information »

for a review of the state-of-the-art on recommender systems: *Adomavicius and Tuzhilin, 2005*

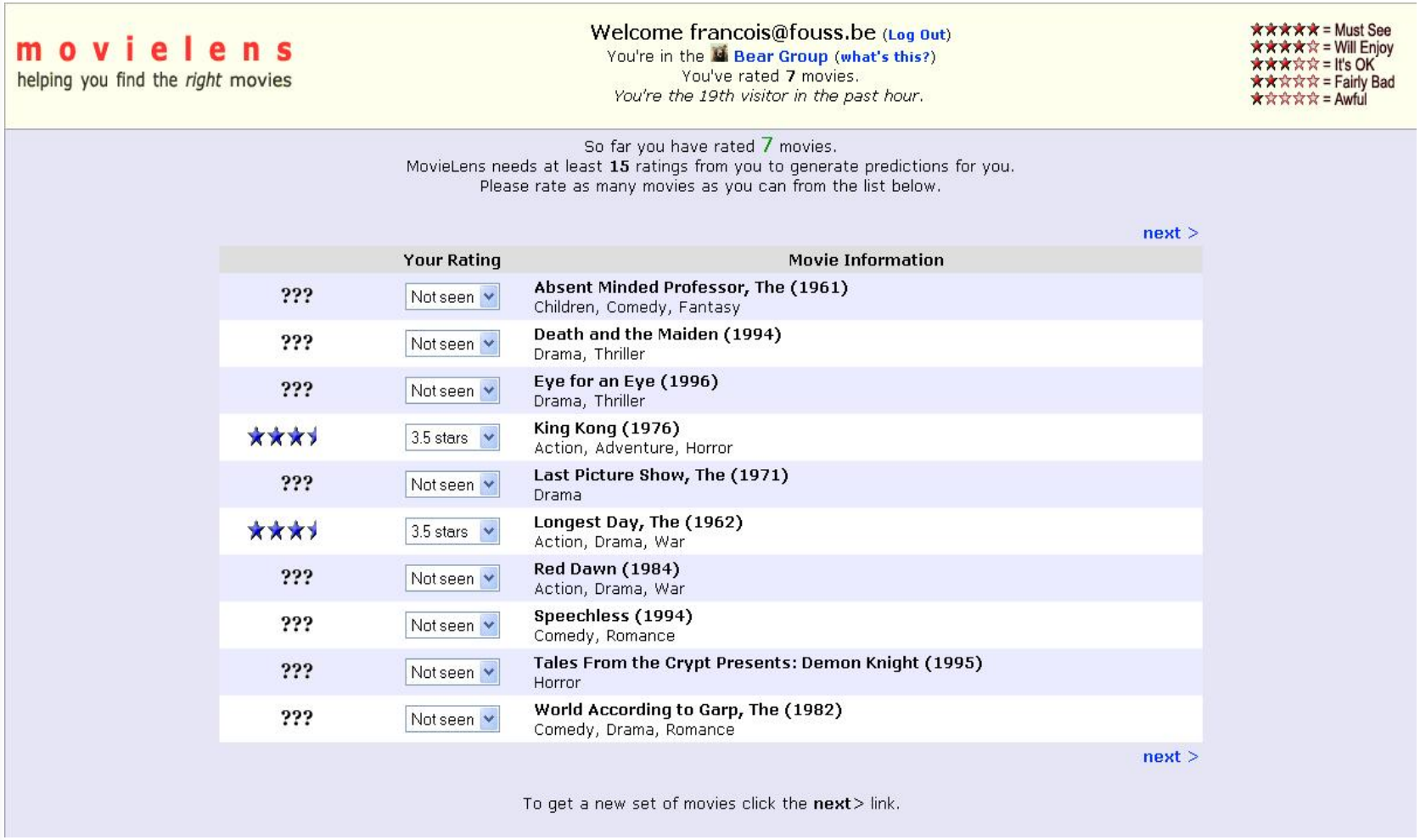
# Everyday examples of recommender systems



# Everyday examples of recommender systems

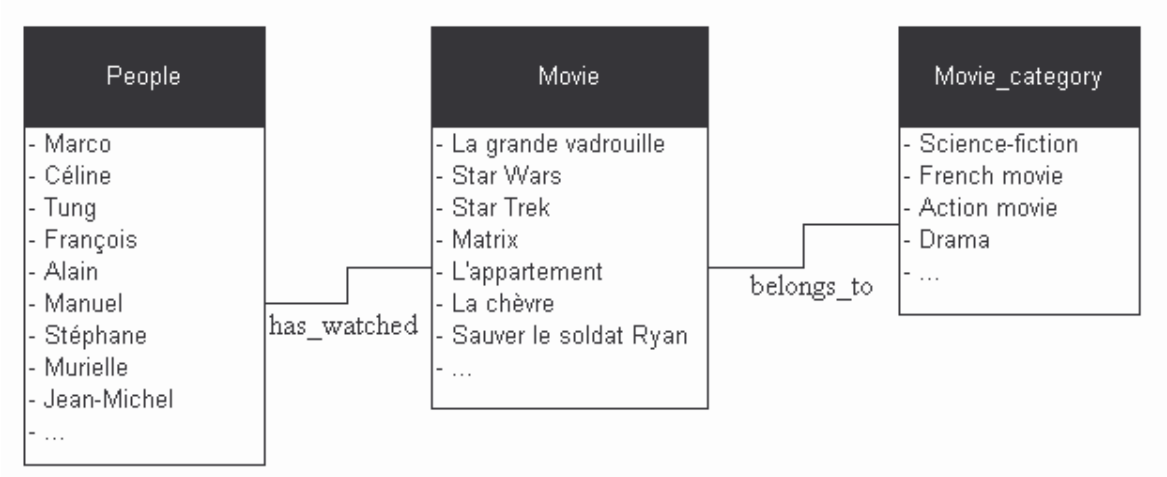


# MovieLens


movielens
helping you find the right movies
Welcome francois@fouss.be (Log Out)
You're in the Bear Group (what's this?)
You've rated 7 movies.
You're the 19th visitor in the past hour.
★★★★★ = Must See
★★★★☆ = Will Enjoy
★★★☆☆ = It's OK
★★☆☆☆ = Fairly Bad
★☆☆☆☆ = Awful
So far you have rated 7 movies.
MovieLens needs at least 15 ratings from you to generate predictions for you.
Please rate as many movies as you can from the list below.
next >
Your Rating
Movie Information
??? Not seen Absent Minded Professor, The (1961) Children, Comedy, Fantasy
??? Not seen Death and the Maiden (1994) Drama, Thriller
??? Not seen Eye for an Eye (1996) Drama, Thriller
★★★½ 3.5 stars King Kong (1976) Action, Adventure, Horror
??? Not seen Last Picture Show, The (1971) Drama
★★★½ 3.5 stars Longest Day, The (1962) Action, Drama, War
??? Not seen Red Dawn (1984) Action, Drama, War
??? Not seen Speechless (1994) Comedy, Romance
??? Not seen Tales From the Crypt Presents: Demon Knight (1995) Horror
??? Not seen World According to Garp, The (1982) Comedy, Drama, Romance
next >
To get a new set of movies click the next> link.

# Other examples

- MovieLens (movielens.umn.edu)




- Hollywood Video (www.hollywoodvideo.com)
- Netflix (www.netflix.com)
- Jester (shadow.ieor.berkeley.edu/humor)
- WikiLens (www.wikilens.org)
- etc.

# Outline

- Introduction
- **Typology of recommender systems**
  - Preference indicators
  - Recommendation approach
  - Recommendation technique
  - Direct or indirect method
  - Provided results
- Algorithms
- Validation
- Experimental results

# Preference indicators

- Rating triplet or co-occurrence pair
- Explicit or implicit

# Rating triplet

- A rating triplet has the form $(u,i,r)$

where

- $u$ corresponds to the index of a user
- $i$ corresponds to the index of an item
- $r$ corresponds to the rating provided

by user $u$ to item $i$

| | | |
|---|---|---|
| 8 | 689 | 4 |
| 9 | 7 | 4 |
| 9 | 50 | 5 |
| 9 | 201 | 5 |
| 9 | 242 | 4 |
| 9 | 276 | 4 |
| 9 | 294 | 4 |
| 9 | 371 | 5 |
| 9 | 385 | 5 |
| 9 | 402 | 4 |
| 9 | 483 | 5 |
| 9 | 615 | 4 |
| 9 | 690 | 1 |
| 10 | 1 | 4 |
| 10 | 4 | 4 |
| 10 | 9 | 4 |

# Co-occurrence pair

- A co-occurrence pair has the form $(u,i)$

where

- $u$ corresponds to the index of a user
- $i$ corresponds to the index of an item
- the occurrence of the pair $(u,i)$ means that user $u$ has bought / watched / consumed item $i$

| | |
|---|---|
| 22 | 2 |
| 22 | 4 |
| 22 | 17 |
| 22 | 21 |
| 22 | 24 |
| 22 | 29 |
| 22 | 50 |
| 22 | 53 |
| 22 | 62 |
| 22 | 68 |
| 22 | 85 |
| 22 | 89 |
| 22 | 94 |

# Explicit preference indicators

- Explicitly provided by the user

Examples:

- asking a person to rate an item
- asking a person to rank a set of items
- asking a person to choose between items
- etc.

# Implicit preference indicators

- Implicitly gathered

  *Claypool 2001*

  Examples:

  - observing the items that a person views in his/her online shopping
  - saving information about the time spent on a page
  - observing the mouse clicks
  - etc.

# Recommendation approaches

- Content-based approaches
- Collaborative approaches
- Hybrid approaches

# Content-based approaches

*Belkin 1992, Baeza-Yates 1999*

1. Discover patterns among items
2. Find similar items

have their roots in the information retrieval and information filtering communities

Shortcomings:

- hardly deal with new users
- difficulties to distinguish items
- overspecialization

# Collaborative approaches

*Goldberg 1992, Maes 1995*

1. Analyze the ratings previously given by the person of interest
2. Find neighbours of users or items

Shortcomings:

- hardly deal with new users
- hardly recommend new items
- sparsity

# Hybrid approaches

*Balabanovic 1997, Basilico 2004, Basu 1998*

- combine content-based and collaborative approaches
- take advantage of both approaches
- various kinds of combination

# Recommendation techniques

- Model-based techniques
- Memory-based techniques

# Model-based techniques

- develop a model of users ratings
- apply the model to new information

- *Examples*
  - *decision trees models*
  - *latent class models*
  - *artificial neural networks models*
  - *etc.*

# Memory-based techniques

- use various statistical techniques to recommend items to users
- by determining a neighbourhood for users or items, or not

- *Examples:*
  - *Pearson correlation coefficient*
  - *cosine correlation coefficient*
  - *link-based techniques*
  - *etc.*

# Direct or indirect methods

- **Direct method**

  compute directly the similarities between a given person and the items

- **User-based indirect method**

  find the nearest neighbours of the user of interest and proceed from there

- **Item-based indirect method**

  *Karypis 2001*

  find the nearest neighbours of each item rated by the user of interest and proceed from there

# Provided results

- Prediction

  Numerical value expressing the predicted likelihood that a user will like an item

- Recommendation

  List of items that a user will like the most

# Outline

- Introduction
- Typology of recommender systems
- **Algorithms**
  - Feature-analysis based
  - Link-analysis based (random-walk, kernel-based, and web-mining algorithms)
- Validation
- Experimental results

# Traditional approach

- Data are in the form

| Individuals | Variables | | | | | |
|---|---|---|---|---|---|---|
| | $X_1$ | $X_2$ | | $X_j$ | | $X_p$ |
| 1 | | | | | | |
| 2 | | | | | | |
| | | | | | | |
| $i$ | | | | $x_{ij}$ | | |
| | | | | | | |
| $n$ | | | | | | |

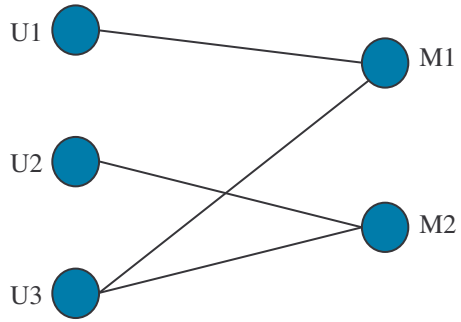
Individual *1*: age=22, gender='F', occupation='student',…
Individual *2*: age=53, gender='M', occupation='artist',…
…
Individual *n*: age=35, gender='F', occupation='lawyer',…

- Traditional algorithms of data mining, machine learning, and statistics exploit these features for clustering / classification / recommendation / etc.

→ **feature-analysis based**

# Another approach

- Data are viewed as links, leading to a graph (very popular for web pages ranking – PageRank, HITS)




→ **link-analysis based**

# Feature-analysis based algorithms

- Cosine correlation coefficient
- Binary similarities
- Latent class model
- etc.

- For comparison:

  'Basic' algorithm which recommends first, for each user, the most rated item (best-seller recommendation)

# Cosine correlation coefficient

$$\cos(i,j) = (\mathbf{v}_i{}^{\mathrm{T}}\mathbf{v}_j)/(\|\mathbf{v}_i\|\,\|\mathbf{v}_j\|)$$

where $\mathbf{v}_i$ ($\mathbf{v}_j$) is a binary vector containing the items user $i$ ($j$) has rated (or not)

$$\mathbf{v}_i = [0\ 1\ 1\ 0 \ldots 0]^{\mathrm{T}} \qquad \mathbf{v}_j = [1\ 1\ 0\ 0 \ldots 0]^{\mathrm{T}}$$

# Binary similarities

*Johnson and Wichern, 2002*

- Define a frequency table

| | | Individual $j$ 1 | 0 | Totals |
|---|---|---|---|---|
| Individual $i$ | 1 | $a$ | $b$ | $a+b$ |
| | 0 | $c$ | $d$ | $c+d$ |
| Totals | | $a+c$ | $b+d$ | $p=a+b+c+d$ |

and use, for example, the « ratio of matches to mismatches with 0-0 matches excluded »

$$sim(i,j) = s(\mathbf{v}_i, \mathbf{v}_j) = \frac{a}{b+c}$$

# Binary similarities: examples

| | Coefficient | Rationale |
|---|---|---|
| 1. | $\frac{a+d}{p}$ | Equal weights for 1-1 matches and 0-0 matches. |
| 2. | $\frac{2(a+d)}{2(a+d)+b+c}$ | Double weight for 1-1 matches and 0-0 matches. |
| 3. | $\frac{a+d}{a+d+2(b+c)}$ | Double weight for unmatched pairs. |
| 4. | $\frac{a}{p}$ | No 0-0 matches in numerator. |
| 5. | $\frac{a}{a+b+c}$ | No 0-0 matches in numerator or denominator. (The 0-0 matches are treated as irrelevant). |
| 6. | $\frac{2a}{2a+b+c}$ | No 0-0 matches in numerator or denominator. Double weight for 1-1 matches. |
| 7. | $\frac{a}{a+2(b+c)}$ | No 0-0 matches in numerator or denominator. Double weight for unmatched pairs. |
| 8. | $\frac{a}{b+c}$ | Ratio of matches to mismatches with 0-0 matches excluded. |

# Latent Class

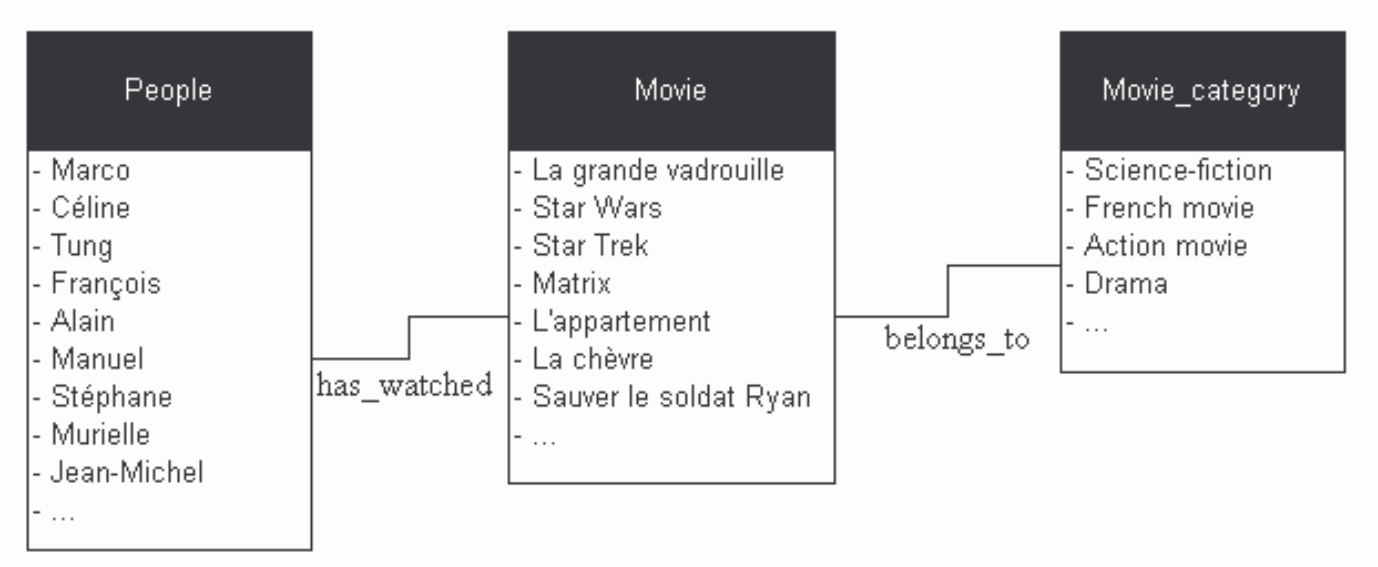
*Hofman et al. 1999, Delannay 2006*

- clustering model
- assumes that the preferences of a user are established through a latent variable (i.e., a non-observable variable)
- standard procedure: Expectation-Maximization (EM) algorithm

# Link-analysis based algorithms

- Random-walk based algorithms
- Kernel-based algorithms
- Web-mining based algorithms

# Some definitions:
# Remember our movie example



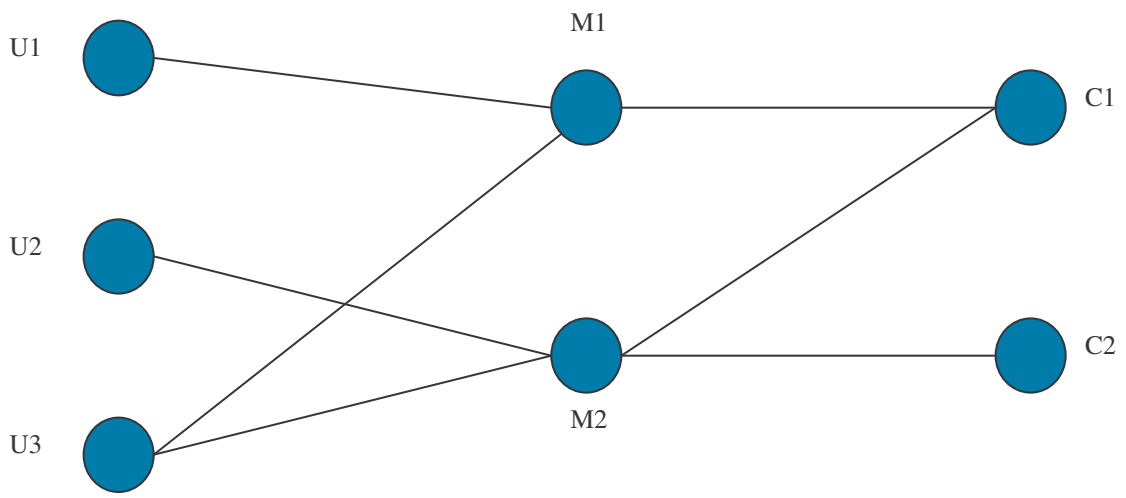

from which, we define

- a weighted graph
- an adjacency matrix
- the transition matrix
- a random-walk model on the graph

# The weighted graph associated with a database

- Database elements correspond to nodes of the graph
- Database links correspond to edges

  Example: a database containing 3 users, 2 movies and 2 movie categories:



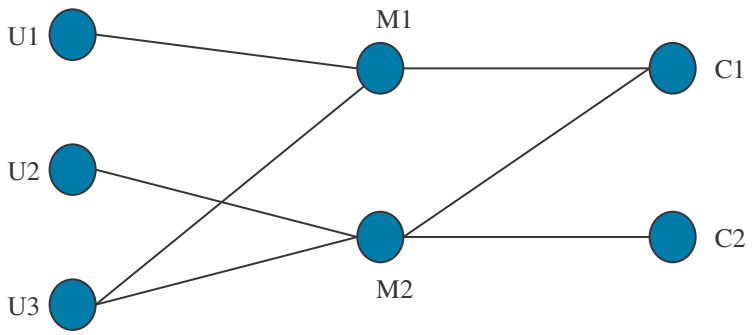
# The adjacency matrix

- The elements $a_{ij}$ of the adjacency matrix $\mathbf{A}$ of a weighted, undirected, graph are defined as

$$a_{ij} = \begin{cases} w_{ij} \text{ if node } i \text{ is connected to node } j \\ 0 \text{ otherwise} \end{cases}$$

  where $\mathbf{A}$ is symmetric

- The $w_{ij} \geq 0$ represent the strength of relationship between node $i$ and node $j$




$$\mathbf{A} = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

# The transition matrix

- We define $\mathbf{P}$ as the transition matrix whose entries are

$$p_{ij} = \frac{a_{ij}}{a_{i.}} \quad \text{where} \quad a_{i.} = \sum_{j=1}^{n} a_{ij}$$

- Remember our example:

$$\mathbf{A} = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \qquad \mathbf{P} = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{3} & 0 & \frac{1}{3} & 0 & 0 & \frac{1}{3} & 0 \\ 0 & \frac{1}{4} & \frac{1}{4} & 0 & 0 & \frac{1}{4} & \frac{1}{4} \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

# Link-analysis based algorithms

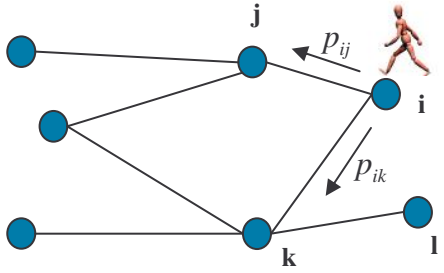
- **Random-walk based algorithms**
  - The average first-passage time
  - The average commute time
  - The pseudoinverse of the Laplacian matrix of the graph
- Kernel-based algorithms
- Web-mining based algorithms

# A random-walk model on the graph

- Every node is associated to a state of a Markov chain
- We define a random variable, $s(t)$, representing the state of the Markov model at time step $t$
- The random walk is defined by the single-step transition probabilities

$$\mathrm{P}(s(t+1)=j|s(t)=i)=p_{ij}=\frac{a_{ij}}{a_{i.}}$$

where $a_{i.}=\sum_{j=1}^{n}a_{ij}$




- In other words, to any state or node $i$, we associate a probability of jumping to an adjacent node, $s(t+1)=j$, which is proportional to the weight $w_{ij}$ of the edge connecting $i$ and $j$

# Average first-passage time

- $m(k|i)$ = average number of steps a random walker, starting in state $i$, will take to enter state $k$ for the first time

$$\begin{cases} m(k|i) = 1 + \sum_{\substack{j=1 \\ j \neq k}}^{n} p_{ij}\, m(k|j), \text{ for } i \neq k \\ \\ m(k|k) = 0 \end{cases}$$

- These equations can be used in order to iteratively compute the first-passage times.

# Average commute time

- $n(i,j) = m(j|i) + m(i|j)$ = average number of steps a random walker, starting in state $i \neq j$, will take before entering a given state $j$ for the first time, and go back to $i$
- Note: while $n(i,j)$ is symmetric by definition, $m(i|j)$ is not.

# The Laplacian matrix

- The Laplacian matrix $\mathbf{L}$ of the graph is defined by

$$\mathbf{L} = \mathbf{D} - \mathbf{A}$$

where $\mathbf{D} = \mathrm{diag}(a_{i.})$ with $d_{ii} = [\mathbf{D}]_{ii} = a_{i.} = \sum_{j=1}^{n} a_{ij}$

Properties:

- $\mathbf{L}$ is doubly centered
- $\mathbf{L}$ is symmetric
- $\mathbf{L}$ is positive semidefinite
- We suppose that the graph is connected, so that the rank of $\mathbf{L}$ is $n - 1$, where $n$ is the number of nodes

# Pseudoinverse of the Laplacian matrix ($\mathbf{L}^+$)

- The rank of $\mathbf{L}$ is n-1
- $\mathbf{L}$ is not inversible
  - ⇨use of the pseudoinverse ($\mathbf{L}^+$)
- It can be shown that $\mathbf{L}^+$ matrix contains the inner products of node vectors in a transformed space and can then be considered as a similarity measure between the nodes.

# Link-analysis based algorithms

- Random-walk based algorithms
- **Kernel-based algorithms**
- Web-mining based algorithms

# Kernel-based algorithms

- In a few words, a kernel is simply
  - An inner product matrix
- That is, a matrix containing inner products as entries,

$$[\mathbf{K}]_{ij} = k_{ij} = \mathbf{x}_i^{\mathrm{T}} \mathbf{x}_j$$

defined in some abstract inner product space, called the feature space

# Kernel-based algorithms examples

- The exponential diffusion kernel

$$\mathbf{K}_{\mathrm{ED}} = \sum_{k=0}^{\infty} \frac{\alpha^k \mathbf{A}^k}{k!} = \exp(\alpha \mathbf{A})$$

- The Laplacian exponential diffusion kernel

$$\mathbf{K}_{\mathrm{LED}} = \exp(-\alpha \mathbf{L})$$

- The von Neumann diffusion kernel

$$\mathbf{K}_{\mathrm{VND}} = \sum_{k=0}^{\infty} \alpha^k \mathbf{A}^k = (\mathbf{I} - \alpha \mathbf{A})^{-1}$$

- The regularized Laplacian kernel

$$\mathbf{K}_{\mathrm{RL}} = (\mathbf{I} + \alpha \mathbf{L})^{-1}$$

- The commute time kernel

$$\mathbf{K}_{\mathrm{CT}} = \mathbf{L}^{+}$$

# Kernel-based algorithms: examples

- The Markov diffusion kernel

$$\mathbf{K}_{\mathrm{MD}}(t) = \mathbf{Z}(t)\mathbf{Z}^{\mathrm{T}}(t) \text{ with } \mathbf{Z}(t) = \frac{1}{t}\sum_{\tau=1}^{t}\mathbf{P}^{\tau}$$

- The cross-entropy diffusion matrix

$$\mathbf{K}_{\mathrm{CED}}(t) = \mathbf{Z}(t)\log(\mathbf{Z}^{\mathrm{T}}(t)) + \log(\mathbf{Z}(t))\mathbf{Z}^{\mathrm{T}}(t)$$

Each of these 7 quantities provides similarity measures between the nodes of the graph

# Link-analysis based algorithms

- Random-walk based algorithms
- Kernel-based algorithms
- **Web-mining based algorithms**
  - A variant of HITS algorithm
  - A variant of PageRank algorithm (called ItemRank)

# HITS algorithm

- The model proposed by Kleinberg is based on two concepts
  - **Hub** pages
  - **Authority** pages
- These are two categories of web pages
- These two concepts are strongly connected

# HITS algorithm

Original algorithm

- A page's **authority score** is proportional to the sum of the hub scores that link to it
- A page's **hub score** is itself proportional to the sum of the authority scores that it links to

Application to collaborative recommendation

- A **user's score** is proportional to the sum of the scores of the items rated by the user.
- An **item's score** is itself proportional to the sum of the scores of the users that have rated this item

# HITS algorithm

- Leading to the following iterative procedure

$$\mathbf{U}_{ij}^{(k)} = \sum_{l=1}^{I}\sum_{m=1}^{I} a_{il}a_{jm}\mathbf{I}_{lm}^{(k-1)}$$

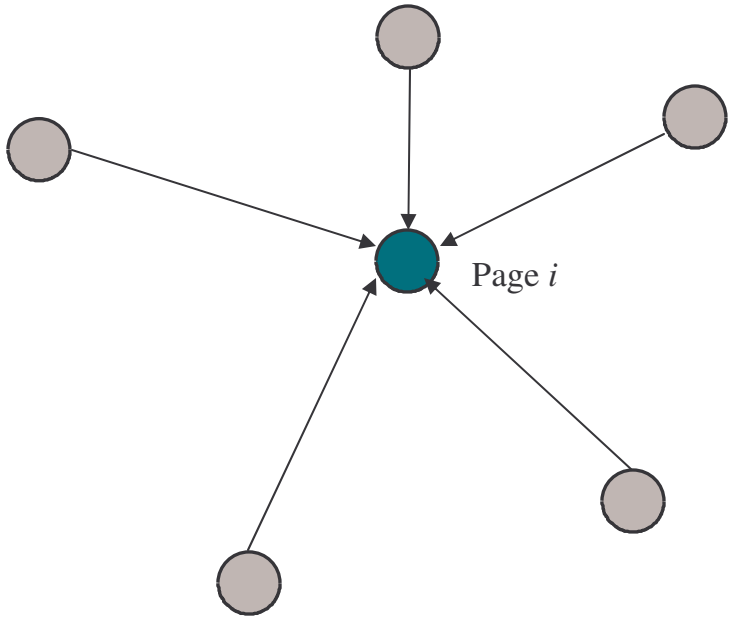
$$\mathbf{I}_{ij}^{(k)} = \sum_{l=1}^{U}\sum_{m=1}^{U} a_{li}a_{mj}\mathbf{U}_{lm}^{(k-1)}$$

where $k$ denotes the $k$th iteration, $U$ is the number of users, $I$ the number of items, $\mathbf{U}$ is a $U \times U$ matrix containing users' scores, $\mathbf{I}$ is a $I \times I$ matrix containing items' scores, and $a_{ij}$ is extracted from the adjacency matrix.

- Notice that to ensure the convergence of the iterative procedure, we have to normalize the scores at each step

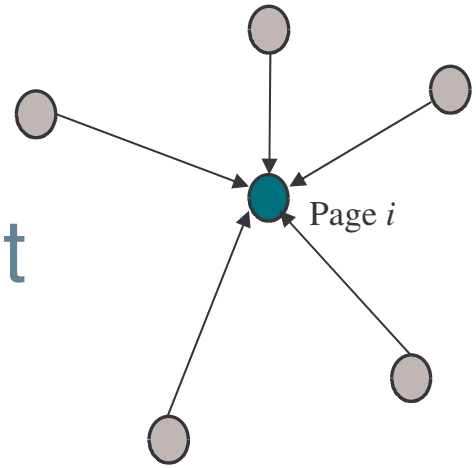
# PageRank algorithm

- To each web page we associate a score, $x_i$
  - The score of page $i$, $x_i$, is proportional to the weighted averaged score of the pages pointing to page $i$



# PageRank algorithm

- A page with a high score is a page that is pointed by
  - many pages
  - having each a high score
- Thus a page is an important page if
  - it is pointed by many, important, pages



# PageRank

- Remember PageRank iterative procedure:

$$\mathbf{PR}_{i+1} = \alpha \mathbf{B}^{\mathrm{T}} \mathbf{PR}_i + (1-\alpha)\frac{1}{N}\mathbf{1}_N$$

where $\mathbf{PR}_{\mathrm{i}}$ is the vector containing the pages' scores at step $\mathrm{i}$, $\mathbf{B}$ is a stochastic matrix, $0 < \alpha < 1$, and $\mathbf{1}_N$ is a column vector made of $N$ ones.

# ItemRank

*Gori and Pucci 2006*

$$\mathbf{IR}_{U_i} = \alpha \mathbf{C}\, \mathbf{IR}_{U_i} + (1-\alpha)\mathbf{d}_{U_i}$$

where the stochastic matrix $\mathbf{B}$ will be the correlation matrix $\mathbf{C}$ (whose element $i$,$j$ contain the number of items co-rated by users $i$ and $j$), vector $\mathbf{d}_{U_i}$ has been build according to user $U_i$ preferences

$$\mathbf{d}_{U_i}(k) = \begin{cases} 0 \text{ if user } U_i \text{ has not rated item } I_k \\ R_{U_i I_k}/N_{U_i} \text{ otherwise} \end{cases}$$

where $R_{U_i I_k}$ is the rating provided by user $U_i$ for item $I_k$ and $N_{U_i}$ is the number of items rated by user $U_i$.

# Outline

- Introduction
- Typology of recommender systems
- Algorithms
- **Validation**
  - Cross-validation
  - Performance measures
- Experimental results

# Cross-validation

- 10-fold cross-validation (using a training set and a test set)
  - Each test set contains 10% of the ratings (10,000 ratings)
  - The average result across all 10 trials is computed
- In other words, we ‘hide’ some existing links for the computation of the recommendations (these hidden links form the test set)

# Performance measures

- Mean Absolute Error (MAE)
- Precision
- Recall
- Degree of agreement
- etc.

# Performance evaluation

## For each user

1. take only the non-watched movies
2. rank them according to the corresponding algorithm
3. compute the performance measure between this ranking and watched movies from the test set

## Compute the average result

| | Degree of agreement | Recall |
|---|---|---|
| Random ranking | **0.5** | **close to 0.0** |
| Ideal ranking | **1.0** | **close to 1.0** |

# Outline

- Introduction
- Typology of recommender systems
- Algorithms
- Validation
- **Experimental results**

# Experimental results

- Results on the real MovieLens database
  - 943 users, 1682 movies, 19 movie categories
  - 100,000 ratings
- Experiment: suggest movies to people

  (for each algorithm the closest non-watched movie is proposed first to the user)

# Some results

| MovieLens dataset | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Direct method (in %)** | | | | | | | | | |
| | **Basic** | **Binary** | **Cosine** | **Latent Class** | **$L^+$** | **$K_{RL}$** | **$K_{MD}$** | **ItemRank** | **HITS** |
| **Agreement** | 85.98 | / | / | **93.16** | 91.11 | 91.17 | 92.74 | 89.29 | / |
| **Recall 10** | 11.02 | / | / | **20.84** | 16.31 | 18.85 | **20.66** | 15.71 | / |
| **Recall 20** | 17.43 | / | / | **31.94** | 26.39 | 28.43 | 31.07 | 24.17 | / |
| **User-based indirect method (in %)** | | | | | | | | | |
| **Agreement** | 90.13 | 92.64 | 92.66 | / | **92.90** | **92.93** | 92.58 | / | 90.07 |
| **Neighbours** | 20 | 100 | 70 | / | 100 | 100 | 90 | / | 30 |
| **Recall 10** | 17.23 | 20.76 | 20.68 | / | **21.43** | 21.00 | 20.66 | / | 17.05 |
| **Neighbours** | 50 | 50 | 60 | / | 40 | 30 | 50 | / | 50 |
| **Recall 20** | 25.60 | 31.13 | 31.29 | / | **32.20** | 31.80 | 31.27 | / | 25.37 |
| **Neighbours** | 30 | 50 | 40 | / | 50 | 40 | 30 | / | 30 |
| **Item-based indirect method (in %)** | | | | | | | | | |
| **Agreement** | 88.03 | **93.27** | 92.78 | / | 92.13 | 92.29 | 92.43 | / | 86.81 |
| **Neighbours** | 100 | 100 | 100 | / | 70 | 100 | 70 | / | 100 |
| **Recall 10** | 6.28 | **18.92** | 17.29 | / | 16.41 | 16.93 | 16.57 | / | 5.81 |
| **Neighbours** | 100 | 20 | 20 | / | 40 | 30 | 30 | / | 100 |
| **Recall 20** | 11.59 | **30.81** | 28.72 | / | 27.98 | 27.11 | 26.67 | / | 11.07 |
| **Neighbours** | 80 | 20 | 30 | / | 40 | 30 | 20 | / | 100 |

# Analyzing the results

- The best results overall are provided by ‘Binary’, ‘Latent Class’, $\mathbf{L}^+$ ($K_{CT}$), $K_{RL}$, and $K_{MD}$.
- Laplacian-based kernels perform better than adjacency-matrix based kernels
- Web-mining based algorithms (i.e., HITS and ItemRank) provide poor results (as well as most random-walk algorithms).
- Link-analysis algorithms provide results comparable to feature-based algorithms.

# References


- G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, pages 734–749, 2005.
- M. Balabanovic and Y. Shoham. Fab: Content-based, collaborative recommendation. *Communications of the ACM*, 40:66–72, 1997.
- P. Baldi, P. Frasconi, and P. Smyth. *Modeling the Internet and the Web: Probabilistic Methods and Algorithms*. John Wiley & Sons, 2003.
- J. Basilico and T. Hofmann. Unifying collaborative and content-based filtering. *Proceedings of the Twenty-first International Conference on Machine Learning*, page 9, 2004.
- C. Basu, H. Hirsh, and W. Cohen. Recommendation as classification: Using social and content-based information in recommendation. *Recommender System Workshop'98*, pages 11–15, 1998.
- J. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, 1998.
- S. Chakrabarti. *Mining the Web: Discovering Knowledge from Hypertext Data*. Elsevier Science, 2003.
- M. Claypool, P. Le, M.Waseda, and D. Brown. Implicit interest indicators. *Proceedings of the ACM Intelligent User Interfaces Conference*, pages 33–40, 2001.
- M. Deshpande and G. Karypis. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems*, 22(1):143–177, 2004.
- P. Domingos. Prospects and challenges for multi-relational data mining. *ACM SIGKDD Explorations Newsletter*, 5(1):80–83, 2003.
- F. Fouss, A. Pirotte, J.-M. Renders, and M. Saerens. Random-walk computation of similarities between nodes of a graph, with application to collaborative recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 19(3):355–369, 2007.
- K. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, 1992.
- M. Gori and A. Pucci. Research paper recommender systems: A random walk based approach. *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, 2006.
- J. Herlocker, J. Konstan, L. Terveen, and J. Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, 22(1):5–53, 2004.

# References


- T. Ito, M. Shimbo, T. Kudo, and Y. Matsumoto. Application of kernels to link analysis. *Proceedings of the eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 586–592, 2005.
- G. Karypis. Evaluation of item-based top-n recommendation algorithms. *Proceedings of the tenth International Conference on Information and Knowledge Management*, pages 247–254, 2001.
- J.M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.
- J. Konstan, B. Miller, D. Maltz, J. Herlocker, L. Gordon, and J. Riedl. Grouplens: Applying collaborative filtering to usenet news. *Communications of the ACM*, 40(3):77–87, 1997.
- B. Marlin. *Collaborative Filtering: A Machine Learning Perspective*. University of Toronto, 2004.
- A. Nakamura and N. Abe. Collaborative filtering using weighted majority prediction algorithms. *Proceedings of the 15th International Conference on Machine Learning*, 1998.
- M. O'Mahony, N. Hurley, N. Kushmerick, and G. Silvestre. Collaborative recommendation: A robustness analysis. *ACM Transactions on Internet Technology*, 4(4):344–377, 2004.
- L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. *Technical Report, Computer System Laboratory, Stanford University*, 1998.
- M. Rashid, I. Albert, D. Cosley, S. Lam, S.McNee, J. Konstan, and J. Riedl. Getting to know you: Learning new user preferences in recommender systems. *Proceedings of the 7th International Conference on Intelligence User Interfaces*, pages 127–134, 2002.
- P. Resnick, I. Neophytos, S. Mitesh, P. Bergstrom, and J. Riedl. Grouplens: An open architecture for collaborative filtering of netnews. *Proceedings of the Conference on Computer Supported CooperativeWork*, pages 175–186, 1994.
- B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. *Proceedings of the International World Wide Web Conference*, pages 285–295, 2001.
- B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering. *Proceedings of the Fifth International Conference on Computer and Information Technology*, 2002.
- U. Shardanand and P. Maes. Social information filtering: Algorithms for automating 'word of mouth'. *Proceedings of the Conference on Human Factors in Computing Systems*, pages 210–217, 1995.