Seminars in Databases

Course Overview and Motivation

F. Ricci
Contact Details

- Francesco Ricci
  - Room 212 (POS)
  - fricci@unibz.it
  - 0471 016971

- Availability Hours:
  - by prior arrangement via e-mail

- Course web site
  - http://www.inf.unibz.it/~ricci/SDB/
Course Structure

- **Lectures:** 24 hours
- **Labs:** 12 hours

**Timetable:**
- Lectures: Wed 14:00-16:00 Room E411
- Labs: Wed 18:00-19:00 Room E411

**Assessment:**
- Seminars: 60%
- Final oral exam: 40%
Motivations (I)

- Reading and UNDERSTANDING a scientific paper is not easy
  - 90% of the case you do not have all the background knowledge required to understand a paper
  - Understanding a paper is not a YES/NOT condition: you must decide when you have a reasonably good understanding of the content
  - Most of the papers contains: mistakes, do not define all the important concepts, do not enter into the details of all the presented material.
Motivations (II)

- Presenting a scientific paper is not easy
- You must decide how to better allocate the **various topics** of the paper in the allowed time
- You must decide the **level of details** to present for each section of the paper (normally not all the details can be presented)
- You must (select) introduce **additional material** that is not contained in the paper (in the references, for instance)
- You must **rise the attention** of the audience (examples, query the audience, etc.)
- You must be able to **adapt** the presentation exploiting the (implicit) **feedbacks** from the audience.
...Hence

- This course should enable you to **practice** and **learn** the difficult job of:
  - reading a scientific paper
  - deciding when you have understood (90%) of the content
  - browsing/reading additional material that is going to help you to master the paper content
  - preparing an effective presentation
  - presenting complex ideas to somebody
  - be able to derive your own conclusions, evaluations, further applications, from the presented ideas and techniques.
Course Format

- I have selected 8 papers for the seminars
- I will illustrate the topic of each paper and provide prerequisites (3 lectures)
- The students express their preferences on the papers
- Each student sends to fricci@unibz.it a sorted list of the papers (before October 9)
- I will collect the preferences and assign the papers to the teams
- *The number of papers assigned to each student could be different*
- The seminars are sorted respecting papers’ prerequisites.
Course Format

- The seminars will take place during the main lecture hours.
- In the lab/exercise – the week before a seminar – the team that will do the next seminar can discuss with me:
  - the paper main contribution and issues
  - the structure of the presentation
- Do not expect I will explain the paper – the objective is to test if you have understood the messages contained in the paper and are almost ready to make the presentation.
What a student must do to pass

- Every student **must** have read the paper **when it is presented at the seminar**
- If something is not clear he/she must make a note and rise a question during the seminar
- The team should start reading the paper they are going to present **two weeks before the seminar**
  - To have enough time to fully understand the content
  - To be able to have a useful meeting with me the week before the seminar (in the LAB)
- During the seminar every student must raise relevant **questions and comments**
- In the discussion time – after paper presentation- every student must **actively participate with comments** (prepare them before the seminar!)
Exam

- The final grade is obtained **evaluating the seminars** and the knowledge acquired about the seminar topics in an **oral final exam**

- **Oral exam**: illustrate one (random) of the papers not presented by you + my questions

- You will have two grades:
  - S for the seminar, from 12 to 18 (Fail below 12)
  - O for the oral part, from 6 to 12 (Fail below 6)

- The final grade is **F = S + O**
How seminars will be evaluated

- The presentation must follow the defined guidelines (see another slide)
- The presentation must cover (almost) all the topics contained in the paper – do not forget the important parts (*just because they are hard to understand*)
- The presentation must be understandable and raise the audience attention
- The presenters must be able to reply to the questions of the other participants
- The presenters must demonstrate that they have understood the paper content.
How to run the seminar (1)

- The presentation should take 1h (aprox.) without discussion.
- The number of slides must be determined in the rehearsal to fit the time constraint.
- The presentation should clearly describe:
  - The problems (technical and application) addressed in the paper.
  - Previously work on the subject.
  - A summary of the paper approach.
  - The detailed description of the approach.
  - The evaluation.
  - The discussion and conclusions.
How to run the seminar (2)

- After the presentation of the paper the team, to activate the discussion, must make a short summary of the paper and **formulate their opinion** regarding:
  - The techniques used in the paper
  - The problem addressed in the paper
  - If the problem has been solved, only partially, and what is still missing
  - The main advantages of the technique
  - The main disadvantages of the technique
  - How you believe you could further expand the work.
The Web has become a platform for service development and business innovation.

Software (browsers or communication clients) are becoming commodities, and value has moved to services and data.

The major Web players (Google or Yahoo) are companies that own and manage large databases (links or products) and can offer unique information services on top of that.

There is a need to develop advanced technologies for accessing large repositories of data typically generated in the web as a platform.

We shall investigate innovative techniques to fully exploit various kinds of web data, such as links, multimedia objects, consumer generated content.
Topics

- Information retrieval on the Web
- Ranking data items
- Managing implicit feedback in information search
- Learning to rank
- Rank integration
- Clustering
- Feature selection
Information retrieval (IR) is the science of searching for information in documents, searching for documents themselves, searching for metadata which describe documents, or searching within databases, whether relational stand-alone databases or hypertextually-networked databases such as the World Wide Web.
**Ranking** is the process of positioning items such as individuals, products or web pages on an ordinal scale in relation to others.

Other Examples of ranking?
Managing implicit feedback in information search

What can I derive from the fact that a user clicked on the 2nd link?
Learning to rank

- If all the users click on the third link then Yahoo should learn to do that search a bit better ...
Other Examples of rank integration?

- Also try: bolzano italy, bolzano handbags, bernard bolzano, bolzano bozen

1. Bolzano - Wikipedia, the free encyclopedia
   The South Tyrol Museum of Archeology in Bolzano is where the ice-mummy “Ötzi” is kept...
   en.wikipedia.org/wiki/Bolzano

2. Bolzano, Italy Forecast: Weather Underground
   Bolzano Mortgages, Mortgages, Refinance, Mortgage Rates, New York Mortgage ...
   www.wunderground.com/global/stations/16020.html

3. Bolzano summary
   Bernard Bolzano (1781-1848) ... A Poster of Bernard Bolzano. Mathematicians born in the same country. Show birthplace location...
   www.groups.dcs.st-and.ac.uk/~history/Mathematicians/Bolzano.html

4. Bolzano - Wikipedia, the free encyclopedia
   The South Tyrol Museum of Archeology in Bolzano is where the ice-mummy “Ötzi” is ...
   en.wikipedia.org/wiki/Bolzen-Bolzano

5. Bolzano -- Encyclopædia Britannica
   Bolzano: city, Trentino-Alto Adige regione, northern Italy. ... Italy, comprising the province of Bolzano-Bolzen (north) and Trento (south)...
   www.britannica.com/esb/article-9000520/Bolzano

6. Bolzano biography
   Biography of Bernard Bolzano (1781-1848) ... Bernard Placidus Johann Nepomuk Bolzano ... parents were Bernard Pompeius Bolzano and Maria Cecilia Maurer...
   www.groups.dcs.st-and.ac.uk/~history/Biographies/Bolzano.html
Clustering
Feature selection

- Features for a page
  - **PageRank value**
  - **Popularity**
  - **Anchor text and inlinks**
    - Total amount of text in links pointing to the page
    - Number of unique words
    - ..
  - **Page**
    - Number of words
    - Frequency of most common terms
    - ...
  - **Domain**
    - Average PageRank of the page domain
    - Average number of outlinks
    - ...

What of them are really important to predict the relevance of the page?
Lectures

- Lecture 1 - Introduction to the seminars: objectives, exam procedure, introduction to the syllabus and short presentation of the papers

- Lecture 2 - Linear Algebra and Markov Chains: linear algebra, matrices, eigenvalues and eigenvectors, markov chains, Google PageRank

- Lecture 3 - Information Retrieval: information retrieval, Web search, indexing, document model, relevance, evaluation of an information retrieval system

- Lecture 4 - Machine Learning for IR: machine learning, support vector machines, recommender systems
Papers


Problem

- Searching on the WWW – discovering pages that are relevant to a given query

- Hypothesis: The network structure of a hyperlinked environment can be a source of information about the content of the environment

- They consider three different types of queries
  - Specific: “what is the best search engine for the web”
  - Broad-topic: “find information about search engines”
  - Similar-pages: “find pages similar to” java.sun.com

- But they focus mostly on the second: the problem is to identify those pages that are more relevant/useful among a large number of available options.
Approach

- Develop a set of algorithms for **extracting information from the link structures**
- Hyperlinks encode a considerable amount of latent human judgment ("if I link a page I think this is relevant and authoritative")
- Balance the **relevance** of the page (refer explicitly to the query) with the **authority** (has many links from other pages)
- The approach is based by observing the relationships between **authorities** about a topic and those pages that link to many related authorities (**hubs**)
Hubs and Authorities

High-quality guides and resource lists
Few incoming links but many outgoing

Most prominent sources of primary content
They do not link to other authorities

unrelated page of large in-degree
Approach

- First, identify a set of pages that are related to a topic (using a text-based search engine) and are likely to contain the most authoritative pages for the searched topic.
- Then, exploit the link structure of the pages “around” this initial set to identify both Hub and Authorities (HITS algorithm).
  - Assign two numbers - a hub weight and an authority weight - to each page in such a way that a page's authority weight is proportional to the sum of the hub weights of pages that link to it; and a page's hub weight is proportional to the sum of the authority weights of pages that it links to.
- Finally, return the authorities for the given searched topic.
- Techniques: graph theory, linear algebra (eigenvalues and eigenvectors)
Problem

- Improve the **static ranking** produced by PageRank (Google)
- **Static ranking** = ranking independent from the user query
- Static ranking is **important**:
  1. General indication of page quality
  2. Search engine index is ordered by static ranking – the dynamic ranker can stop search when static rank is too low
  3. The Web is growing, crawler can avoid considering pages with very low static rank
Artificial Neural Networks (ANN)

\[ Y = I(0.3X_1 + 0.3X_2 + 0.3X_3 - 0.4 > 0) \]

where \( I(z) = \begin{cases} 
1 & \text{if } z \text{ is true} \\
0 & \text{otherwise} 
\end{cases} \)
Approach

- User **Machine Learning** to “learn” the right ranking function
- They use a **Neural Network**
  - Input of the NN is a **pair of pages** \(<i,j>\) represented with **features**
  - The NN computes the output (**score**) for each page: \(o_i\) and \(o_j\)
  - If \(i\) must be ranked higher than \(j\) and \((o_j - o_i)\) is positive then the weights of the NN are adapted to **make this error smaller**
  - The procedure requires that humans provides reference scores for a collection of pages (the \(o_i\) scores – ratings).
Results

- fRank is more accurate than PageRank.
- The features that are more important to predict the rank are those related to the page description, e.g.:
  - Number of words
  - Frequency of most common terms

<table>
<thead>
<tr>
<th>Technique</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>None (Baseline)</td>
<td>50.00</td>
</tr>
<tr>
<td>PageRank</td>
<td>56.70</td>
</tr>
<tr>
<td>fRank</td>
<td>67.43</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Feature Set</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PageRank</td>
<td>56.70</td>
</tr>
<tr>
<td>Popularity</td>
<td>60.82</td>
</tr>
<tr>
<td>Anchor</td>
<td>59.09</td>
</tr>
<tr>
<td>Page</td>
<td>63.93</td>
</tr>
<tr>
<td>Domain</td>
<td>59.03</td>
</tr>
<tr>
<td>All Features</td>
<td>67.43</td>
</tr>
</tbody>
</table>

Accuracy = percentage of correct pair ordering predictions (correct preferences)
Problems

- There are many search engines: *no one is satisfactory for all users*
- No one ranking algorithm can be considered the **best**
- No one search engine is sufficiently **comprehensive** in its coverage of the Web
- **Spam:** manipulation by authors of Web pages to achieve undeservedly high rank
  - How to avoid that?
- No single ranking function can be trusted to perform well for all queries.
Top Search Engines

- Yahoo rates higher in terms of customer satisfaction than Google (University of Michigan's American Customer Satisfaction Index - ACSI)
- "While Google does a great job in search, which is what they do, but [consumers] are seeing Google the same as three years ago."
- Ask.com registered a gain of 5.6 percent
- *Do not think that Google will be always the best!*

<table>
<thead>
<tr>
<th>Provider</th>
<th>Searches (000)</th>
<th>Share of Total Searches (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google</td>
<td>3,906,877</td>
<td>52.7</td>
</tr>
<tr>
<td>Yahoo</td>
<td>1,496,137</td>
<td>20.2</td>
</tr>
<tr>
<td>MSN/Windows Live</td>
<td>985,706</td>
<td>13.3</td>
</tr>
<tr>
<td>AOL</td>
<td>404,036</td>
<td>5.5</td>
</tr>
<tr>
<td>Ask.com</td>
<td>152,268</td>
<td>2.1</td>
</tr>
<tr>
<td>My Web Search</td>
<td>76,827</td>
<td>1.0</td>
</tr>
<tr>
<td>My Way</td>
<td>35,643</td>
<td>0.5</td>
</tr>
<tr>
<td>BellSouth</td>
<td>30,868</td>
<td>0.4</td>
</tr>
<tr>
<td>Comcast</td>
<td>30,467</td>
<td>0.4</td>
</tr>
<tr>
<td>Dogpile</td>
<td>30,452</td>
<td>0.4</td>
</tr>
<tr>
<td>Other</td>
<td>263,431</td>
<td>3.6</td>
</tr>
<tr>
<td>All search</td>
<td>7,412,712</td>
<td>100.0</td>
</tr>
</tbody>
</table>

Source: Nielsen//NetRatings, 2007
Approach

- Rankings can be **compared** using distance measures

**Example:** $t = [x_1 > x_2 > \ldots > x_d]$ a ranking of pages ($x_1 > x_2$ means that $x_1$ is ranked **higher** than $x_2$) and $t(i)$ is the position or rank of page $i$

- If $t$ and $s$ are two rankings on $S$ (set of pages) then the **Kendall tau distance** is:
  \[
  K(s, t) = |\{(i, j) \mid i < j, s(i) < s(j), \text{ but } t(i) > t(j)\}| 
  \]

**Problem solved:** Given a set of rankings $T = \{t_1, \ldots, t_k\}$ find a ranking $s$ that has the **minimal** Kendall tau distance from the rankings in $T$.

- Unfortunately this is a NP-Hard problem!
Consider a simpler problem called **Local Kemenization**:

- $s$ is a **locally Kemeny optimal** aggregation (for $T$) if there is no ranking that can be obtained by a single transposition of an adjacent pair of elements that makes the ranking closer to the rankings in $T$

**Theorem:** Every optimal aggregation is also locally Kemeny optimal

They build different kind of aggregated rankings using various methods (e.g. Borda)

Then they compute a locally Kemeny optimal (fast) and this is considered the best solution.

**Result:** the best aggregations are computed using **Markov Chain** methods:

- Similar to that used by Google but with a page-to-page transition probability that depends on the rankings
- **Idea:** the more rankings consider that page important the more likely is that a transition occurs to that page.
Problem

- Each time a user formulates a query or clicks on a search result, easily **observable feedback** is provided to the search engine.
- How to use this **implicit feedback** to learn a personalized or a better ranking?
- Different ranking methods could be learned for different contexts: user, application domain, ...
- The **sequence** of queries and the clicked links could also provide hints about what the user consider as relevant
  
  **Example:** If many users searching for “travel reimbursement” reformulate the query and eventually click on the “expense-report form”, why not to include this result in the original query?
If the user clicks on the indicated link, what does it mean?

The interpretation depends on:

- the other links
- The position of the link in the ranked list
The frequency of viewership is correlated to the rank
How to eliminate this bias from the interpretation of the click data?
Learning to Rank

- User actions can be interpreted as relative preferences: “for query $q$ the user prefers $d_1$ to $d_2$”
- They try to learn for each query $q$, user $u$, and document $d$ a utility function $h(q, u, d)$
- When they get a preference statement they interpret it as a constraint on the utility function
  - $h(q, u, d_1) > h(q, u, d_2)$
- They search a utility function that can be written as:
  - $h(q, u, d) = w \cdot \Phi(q, u, d)$
  - $w$ is a vector and is a set of features describing the query, the user and the document
- They exploit **Support Vector Machines** to determine the best vector $w$. 
Comparing Two Rankings

- A user reading from top to bottom will have seen the same number of links from the two rankings.
- If a user clicks more frequently on links from one ranking, we can conclude that that ranking is better.
Problem

- Learning a ranking is strongly **influenced by the item model**, i.e., the features used to represent the document

- Using a large number of features can cause "over-fitting" and poor generalization capability

- Reducing the number of features can improve the **efficiency** of the computation

- But feature selection for ranking is **different** from feature selection for classifying
  - Very large number of features
  - Existing methods are not applicable
  - The system evaluation is based on specific measures (e.g. precision-recall vs. prediction accuracy)
Approach

- Use of ranking information, i.e., how good is the feature for ranking \textit{when used alone}?
- It considers the \textbf{similarities} between features, and tries to avoid selecting redundant features.
- It models feature selection for ranking as a \textbf{multi-objective optimization} problem.
- The objective is to find a set of features with \textbf{maximum importance} (usefulness for ranking) and \textbf{minimum similarity}. 
Approach

- The **importance** score of each feature is defined by comparing the ranking produces using only that feature with respect to the target ranking.
- The similarity of two features is defined as the similarity (Kendall tau) of the rankings produced by the two features.
- Then the feature selection problem is cast into an integer programming problem:

\[
\begin{align*}
\text{max} & \quad \sum_i w_i x_i - c \sum_i \sum_{j \neq i} e_{i,j} x_i x_j \\
\text{s.t.} & \quad x_i \in \{0,1\} \quad i = 1, \ldots, m \\
& \quad \sum_i x_i = t
\end{align*}
\]

- \(w_i\) is the importance of feature \(i\) and \(e_{i,j}\) is the similarity of feature \(i\) and \(j\).
Results

- They compute an approximated solution to the optimization problem.
- Run experiments with different feature subset sizes.
- Use the features to run two rank learning methods (Ranking SVM and RankNet).
- Compared these learned ranking (together with their feature selection method) with other simpler feature selection methods:
  - **Information Gain**: reduction of entropy brought by the feature.
  - **Chi test**: correlation between feature and category (relevant/not-relevant).
- And ... got good results 😊
Problem

- Most online storefronts, given a user query (with multiple conditions, e.g. price and weight) present products in a ranked list

- By doing so, much information is lost about the mutual similarity between recommended products

- The top positions may be occupied by products better matching the price but with bad match for the weight

- These systems perform automatic rank aggregations

- How to let the user to have a clearer view of the tradeoffs?
Example

price

weight

Top products in the ranking

query

Third page

Second page
Approach

- Define a **product-to-product distance function** depending on product features.
- Exploit **Multi Dimensional Scaling Techniques**: project the original n-dimensional data (products) on a 2-dimensional space, such that the Euclidean distances of the projected points is **as close as possible to the original distances** in the n-dimensional space.
- Show only products that are **enough different** – select one representative for each cluster.
- Let the user to select one product and repeat the process considering the selected product as the user query.
Graphical Shopping Interface
Incremental search of the best case

User’s selection

Initial case
Incremental search of the best case
Problem

- An online catalogue of hotels may display hundreds of hotels in Rome – how to select one?
- **Collaborative-filtering** recommender systems exploit user ratings on hotels – made by a community of users – to guess if you will like a hotel
- Can we improve the rating prediction accuracy using **multicriteria ratings**?
  - Overall rating vs.
    - Rooms
    - Services
    - Cleanliness
    - Dining
Example

- Products are ranked
- Actually there are many ranks!
- An overall rating is displayed
More specific ratings are acquired

Recommendations (ratings) are predicted for group of users
## Approach

<table>
<thead>
<tr>
<th>Target user</th>
<th>Item $i_1$</th>
<th>Item $i_2$</th>
<th>Item $i_3$</th>
<th>Item $i_4$</th>
<th>Item $i_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>User $u_1$</td>
<td>5</td>
<td>7</td>
<td>5</td>
<td>7</td>
<td>?</td>
</tr>
<tr>
<td>User $u_2$</td>
<td>5</td>
<td>7</td>
<td>5</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>User $u_3$</td>
<td>5</td>
<td>7</td>
<td>5</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>User $u_4$</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>User $u_5$</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>5</td>
</tr>
</tbody>
</table>

Users most similar to the target user:

<table>
<thead>
<tr>
<th>Target user</th>
<th>Item $i_1$</th>
<th>Item $i_2$</th>
<th>Item $i_3$</th>
<th>Item $i_4$</th>
<th>Item $i_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>User $u_1$</td>
<td>5</td>
<td>7</td>
<td>5</td>
<td>7</td>
<td>?</td>
</tr>
<tr>
<td>User $u_2$</td>
<td>5</td>
<td>7</td>
<td>5</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>User $u_3$</td>
<td>5</td>
<td>7</td>
<td>5</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>User $u_4$</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>User $u_5$</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>5</td>
</tr>
</tbody>
</table>

Ratings to be used in prediction:

<table>
<thead>
<tr>
<th>Rating to be predicted</th>
</tr>
</thead>
<tbody>
<tr>
<td>User $u_1$</td>
</tr>
<tr>
<td>User $u_2$</td>
</tr>
<tr>
<td>User $u_3$</td>
</tr>
<tr>
<td>User $u_4$</td>
</tr>
<tr>
<td>User $u_5$</td>
</tr>
</tbody>
</table>
Approach

- $r_0$ is the overall rating
- $r_1, \ldots, r_k$ are the multicriteria ratings
- Combines recommendation techniques and rank aggregation.

**Known ratings**

$R(u, i) = (r_0, r_1, \ldots, r_k)$

(1) Predict $k$ multicriteria ratings using any traditional recommendation technique
Given: $r_i$ (for each $i = 1, \ldots, k$)
Compute: $r_i'$

(2) Learn aggregation function $f$ using statistical or machine learning techniques
Given: $(r_0, r_1, \ldots, r_k)$
Estimate: $f$ such that $r_0 = f(r_1, \ldots, r_k)$

(3) Predict an overall rating
Given: $(r_1', \ldots, r_k')$, $f$
Compute: $r_0'$ based on $r_0' = f(r_1', \ldots, r_k')$
End

- Now is your turn ...