Text and Data Mining In Innovation

Joseph Engler

Innovation Typology

- Generational Models
  1. Linear or Push (Baroque)
  2. Pull (Romantic)
  3. Cyclic (Classical)
  4. Strategic (New Age)
  5. Collaborative (Polyphonic)

Collaborative Authoritativeness

- Focused Web Mining for Papers/Articles
- Creation of Authoritativeness Matrix
- Apply Authoritativeness Metric
- Document Clustering

Focused Web Mining

- Standard Web Crawling Methodologies
- Download of Query Specific Files
- Forms the repository from which Text Mining takes place
Authoritativeness Matrix

- Authors names are parsed from the documents
- Cited references are parsed from the documents
- The publication date is parsed from the document

<table>
<thead>
<tr>
<th>Name</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kusiak</td>
<td>2006</td>
</tr>
<tr>
<td>Lin</td>
<td>2004</td>
</tr>
<tr>
<td>Stokic</td>
<td>1999</td>
</tr>
</tbody>
</table>

Parsing of Names

- Heuristics
  - Name should be first non-empty line after title
- Regular Expressions
  - `^\w+\s\w\[.\]\s*\w+\s*`  
  - `\w+\s\w\[.\]\s*\w+\[a\][n\]d\]`  
- Names Database with Dice Coefficient

Dice Coefficient

- Create Bigrams of the two words being compared.
- Night = {ni, ig, gh, ht} = X
- Nacht = {na, ac, ch, ht} = Y
- Calculate Similarity

\[
\text{Dice}_{\text{Coeff}} = \frac{2(X \cap Y)}{|X| + |Y|}
\]
Authoritativeness Metric

- Scan Authoritativeness Matrix
  - Create Hash of Authors (row)
  - Create Hash of Referenced Authors (column)
  - Create Hash of Average Age of Document for each author
- Authors Hash (rows) measures Out-Links
- Reference Hash (columns) measures In-Links

Authoritativeness Metric Cont.

- Calculate the initial authoritativeness for each author and referenced author.

\[ A_i = \ln(\lambda' \cdot \text{out}_i + \text{in}_i) \]

\( \lambda' \) is the average age of the document for author \( i \)

\( \text{out}_i \) is the number of out-links

\( \text{in}_i \) is the number of in-links

\( \lambda \) is a user defined weight parameter of document age in [0, 1]

Authoritativeness Boosting

- Similar to PageRank Algorithm
- Iterative Approach
- If an authoritative author references a paper, the in-link to that reference is increased
- In-Links of less authoritative authors pose no detriment

In-Link Boosting

- Calculate the mean of the in-links

\[ \overline{\text{in}}_i = \frac{\sum_{j=1}^{N} e^{i_j}}{N} \]

- Update the Authoritativeness Metric

\[ e^{i_j} = \begin{cases} \epsilon^{i_j} - \overline{\text{in}}_i & \text{if } e^{i_j} > \overline{\text{in}}_i \\ 0 & \text{otherwise} \end{cases} \]
Determining Authoritativeness

- Order the authors by Authoritativeness
- Select Top K authors
- Cluster the documents
- Find cluster closest to current issue
- Find most authoritative k authors for that issue
  - Authors that are authoritative overall may not be authoritative on specific topic
  - Possible application of Apriori Principal

Experimental Results

- 945 Articles on Genetic Algorithms
- Unclustered to determine overall authority
- 30 iterations of In-Link Boosting
- $\lambda$ set to 1 to not discount older authoritativeness

Experimental Results Cont.

<table>
<thead>
<tr>
<th>Author</th>
<th>Original Authoritativeness</th>
<th>Boosted Authoritativeness</th>
</tr>
</thead>
<tbody>
<tr>
<td>J. H. Holland</td>
<td>5.278</td>
<td>6.2122</td>
</tr>
<tr>
<td>J. R. Koza</td>
<td>4.8828</td>
<td>5.8105</td>
</tr>
<tr>
<td>F. H. Bennett</td>
<td>4.5849</td>
<td>5.1350</td>
</tr>
<tr>
<td>L. Altenberg</td>
<td>4.3438</td>
<td>4.4306</td>
</tr>
<tr>
<td>D. Andre</td>
<td>3.9512</td>
<td>4.0105</td>
</tr>
</tbody>
</table>

Note the minimal boost of the last two authors.

Cyclic Innovation & Data Mining

- Mining of Requirements
- Creation of Requirements Database
- Construction of Requirements Tree
Web Mining For Requirements

- Source of Requirements
  - Blogs
  - User Reviews
  - Expert Reviews
  - Patent Databases
  - Trade Journals
  - Stock Market Analysis (tricky at best)

Filtering Requirements

- Moaners and Praisers
  - “I hate this MP3 player and would never buy a product from this company again.”
  - “I just love Microsoft and everything they produce. It is all bug free”
- Attempt to assign a measure of success to the requirement
- Identify historic issues versus new requirements

Requirements Database

- Transactional Database of Sorts

<table>
<thead>
<tr>
<th>Product</th>
<th>Smaller Footprint</th>
<th>Increased Bandwidth</th>
<th>Interface with Stylus</th>
<th>Increased RPMs</th>
<th>Non-interfering legs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workstation Desk</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Turret Lather</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Smartphone</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Abstraction of Database

- Utilize Multidimensional Cubes
- Ability to Roll-up or Drill Down similar to OLAP
- Increased choices in levels of abstraction
Mining Frequent Requirements

- Select Product/Service type to mine the requirements for
  - IPod (includes all MP3 Players)
  - On-line Tax Service (includes all on-line)
- Utilize a Market Basket type of Analysis
  - Apriori Algorithm
  - FP-Growth

Frequent Itemset Metrics

- Support
  \[ \text{sup}(A \Rightarrow B) = \frac{\text{Number of tuples containing both } A \text{ and } B}{\text{Total number of tuples}} \]
- Confidence
  \[ \text{conf}(A \Rightarrow B) = \frac{\text{Number of tuples containing both } A \text{ and } B}{\text{Number of tuples containing } A} \]

Mining Frequent Requirements

- Discover Frequent Itemsets
- Itemset can be considered as a conjunction of items
  - \( A \land B \)
- Itemset can be considered as a predicate
  - \( A \Rightarrow B \)

Frequent Itemset Generation

1. Scan Database for frequent 1 items
2. Remove those items that have a support value of less than a given threshold
3. Join the remaining frequent items to form 2 item itemsets
4. Repeat steps 2 and 3 incrementing the itemset size each time until there are no itemsets left to join.
Frequent Itemset Example

- Workbench
  - Close to lathe and Desk
  - Items with legs and a top with stability
- Mine the frequent requirements from our Requirements database previously shown

Frequent Itemset Example

<table>
<thead>
<tr>
<th>Product</th>
<th>Smaller Footprint</th>
<th>Increased Bandwidth</th>
<th>Interface with Stylus</th>
<th>Increased RPMs</th>
<th>Non-interfering Legs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workstation Desk</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Turret Lathe</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Smartphone</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Choose items of similar abstraction to mine from.

Frequent Itemset Example

<table>
<thead>
<tr>
<th>Itemset</th>
<th>Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smaller Footprint</td>
<td>2</td>
</tr>
<tr>
<td>Increased Bandwidth</td>
<td>0</td>
</tr>
<tr>
<td>Interface with Stylus</td>
<td>0</td>
</tr>
<tr>
<td>Increased RPMs</td>
<td>1</td>
</tr>
<tr>
<td>Non-interfering Legs</td>
<td>2</td>
</tr>
</tbody>
</table>

Set support = 2

Frequent Itemset Example

<table>
<thead>
<tr>
<th>Itemset</th>
<th>Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smaller Footprint and Non-Interfering Legs</td>
<td>2</td>
</tr>
</tbody>
</table>

There remains no further itemsets to join.
Build Requirements Tree

- Built from frequent itemsets

```
  Req.
 /    /
|     |
Small Non-Int
Footprint Legs
```