Mining Web Data

Lijun Zhang
zlj@nju.edu.cn
http://cs.nju.edu.cn/zlj
Outline

- Introduction
- Web Crawling and Resource Discovery
- Search Engine Indexing and Query Processing
- Ranking Algorithms
- Recommender Systems
- Web Usage Mining
- Summary
Introduction

- Web is an unique phenomenon
  - The scale, the distributed and uncoordinated nature of its creation, the openness of the underlying platform, and the diversity of applications

- Two Primary Types of Data
  - Web content information
    - Document data, Linkage data (Graph)
  - Web usage data
    - Web transactions, ratings, and user feedback, Web logs
Applications on the Web

- **Content-Centric Applications**
  - Data mining applications
    - Cluster or classify web documents
  - Web crawling and resource discovery
  - Web search
    - Linkage and content
  - Web linkage mining

- **Usage-Centric Applications**
  - Recommender systems
  - Web log analysis
    - Anomalous patterns, and Web site design
Outline

- Introduction
- **Web Crawling and Resource Discovery**
- Search Engine Indexing and Query Processing
- Ranking Algorithms
- Recommender Systems
- Web Usage Mining
- Summary
Web Crawling

- Web Crawlers or Spiders or Robots
- Motivations
  - Resources on the Web are dispensed widely across globally distributed sites
  - Sometimes, it is necessary to download all the relevant pages at a central location

- Universal Crawlers
  - Crawl all pages on the Web (Google, Bing)

- Preferential Crawlers
  - Crawl pages related to a particular subject or belong to a particular site
Crawler Algorithms

- A real crawler algorithm is complex
  - A selection Algorithm, Parsing, Distributed, multi-threads

- A Basic Crawler Algorithm

Algorithm BasicCrawler(Seed URLs: $S$, Selection Algorithm: $A$)

begin
  $FrontierList = S$;
  repeat
    Use algorithm $A$ to select URL $X \in FrontierSet$;
    $FrontierList = FrontierList - \{X\}$;
    Fetch URL $X$ and add to repository;
    Add all relevant URLs in fetched document $X$ to end of $FrontierList$;
  until termination criterion;
end
Selection Algorithms

- Breadth-first
- Depth-first

- Frequency-Based
  - Most universal crawlers are incremental crawlers that are intended to refresh previous crawls

- PageRank-Based
  - Choose Web pages with high PageRank
Preferential Crawlers

- User-defined Criteria
  - Keyword presence in the page
  - A topical criterion defined by a machine learning algorithm
  - A geographical criterion about page location
  - A combination of the different criteria

- Modify the approach for updating the frontier list
  - The web page or pages that it points to need to satisfy the criteria

- Modify the selection algorithm
Multiple Threads

- Network is slow
  - The system is idle when a crawler issues a request for a URL and waits for it

- Concurrency
  - Use multiple threads to update a shared data structure for visited URLs and the page repository (locking or unlocking)
  - The crawler may also distributed geographically with each “sub-crawler” collecting pages in its geographical proximity
Combatting Spider Traps

- The crawling algorithm maintains a list of previously visited URLs for comparison purposes
  - So, it always visits distinct Web pages

- However, many sites create dynamic URLs
  - http://www.example.com/page1
  - http://www.example.com/page1/page2
  - Limit the maximum size of the URL
  - Limit the number of URLs from a site
Near Duplicate Detection

- Many duplicates of the same page may be crawled
- A \( k \)-shingle (\( k \)-gram)
  - A string of \( k \) consecutively occurring words
  
  "Mary had a little lamb, its fleece was white as snow."
  
  "Mary had", "had a", "a little", ...

- The Shingle-Based Similarity

  \[
  J(S_1, S_2) = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|}
  \]

  - \( S_1 \) and \( S_2 \) be the \( k \)-shingles extracted from two documents \( D_1 \) and \( D_2 \)
Outline

- Introduction
- Web Crawling and Resource Discovery
- **Search Engine Indexing and Query Processing**
- Ranking Algorithms
- Recommender Systems
- Web Usage Mining
- Summary
The Process of Search

- **Offline Stage**
  - The search engine preprocesses the crawled documents to extract the tokens and constructs an index
  - A quality-based ranking score is also computed for each page

- **Online Query Processing**
  - The relevant documents are accessed and then ranked using both their relevance to the query and their quality
Offline Stage

☐ The Preprocessing Steps
- The relevant tokens are extracted and stemmed
- Stop words are removed

☐ Construct the Inverted Index
- Maps each word identifier to a list of document identifiers containing it
  - Document ID, Frequency, Position

☐ Construct the Vocabulary Index
- Access the storage location of the inverted word
Content-Based Score

- A word is given different weights, depending upon whether it occurs in the title, body, URL token, or the anchor text.
- The number of occurrences of a keyword in a document will be used in the score.
- The prominence of a term in font size and color may be leveraged for scoring.
- When multiple keywords are specified, their relative positions in the documents are used as well.
Limitations of Content-Based Score

- It does not account for the reputation, or the quality, of the page
  - A user may publish incorrect material

Web Spam

- Content-spamming: The Web host owner fills up repeated keywords in the hosted Web page
- Cloaking: The Web site serves different content to crawlers than it does to users

Search Engine Optimization (SEO)

- The Web site owners attempt to optimize search results by using their knowledge
Ranking (3)

- **Reputation-Based Score**
  - Page *citation* mechanisms: When a page is of high quality, many other Web pages point to it
  - User *feedback* or behavioral analysis mechanisms: When a user chooses a Web page, this is clear evidence of the relevance of that page to the user

- **The Final Ranking Score**
  \[ \text{RankScore} = f(\text{IRScore}, \text{RepScore}). \]
  - Spams always exist
Outline

- Introduction
- Web Crawling and Resource Discovery
- Search Engine Indexing and Query Processing
- **Ranking Algorithms**
- Recommender Systems
- Web Usage Mining
- Summary
Google’s PageRank (1)

- Random Walk Model
  - A random surfer who visits random pages on the Web by selecting random links on a page
  1. The long-term relative frequency of visits to any particular page is clearly influenced by the number of in-linking pages to it
  2. The long-term frequency of visits to any page will be higher if it is linked to by other frequently visited pages
Google’s PageRank (2)

- **Random Walk Model**
  - Dead ends: pages with no outgoing links
  - Dead-end component

![Diagram showing dead-end node and dead-end component]
Google’s PageRank (3)

- Random Walk Model
  - Dead ends: pages with no outgoing links
    - Add links from the dead-end node (Web page) to all nodes (Web pages), including a self-loop to itself
  - Dead-end component
    - A teleportation (restart) step: The random surfer may either jump to an arbitrary page with probability $\alpha$, or it may follow one of the links on the page with probability $1 - \alpha$
Steady-state Probabilities (1)

- \( G = (N,A) \) be the directed Web graph
  - Nodes correspond to pages
  - Edges correspond to hyperlinks
    - Include added edges for dead-end nodes
  - \( \pi(i) \): the steady-state probability at \( i \)
  - \( In(i) \): set of nodes incident on \( i \)
  - \( Out(i) \): the set of end points of the outgoing links of node \( i \)
  - Transition matrix \( P \) of the Markov chain
    
    \[
    p_{ij} = \frac{1}{|Out(i)|} \quad \text{if there is an edge form } i \text{ to } j
    \]
Steady-state Probabilities (2)

- The probability of a teleportation into $i$ is $\frac{\alpha}{n}$.

- The probability of a transition into $i$ is $(1 - \alpha) \sum_{j \in \text{In}(i)} \pi(j) \cdot p_{ji}$.

- Then, we have

$$\pi(i) = \frac{\alpha}{n} + (1 - \alpha) \cdot \sum_{j \in \text{In}(i)} \pi(j) \cdot p_{ji}$$
Steady-state Probabilities (3)

- Let \( \bar{\pi} = [\pi(1), \ldots, \pi(n)]^T \)

\[
\bar{\pi} = \frac{\alpha \bar{e}}{n} + (1 - \alpha) P^T \bar{\pi}
\]

- With the constraint \( \sum_{i=1}^{n} \pi(i) = 1 \)

- Optimization
  - \( \bar{\pi}(0) = \frac{\bar{e}}{n} \)
  - \( \bar{\pi}(t+1) = \frac{\alpha \bar{e}}{n} + (1 - \alpha) P^T \bar{\pi}(t) \)
  - \( \bar{\pi}(t+1) \leftarrow \frac{\bar{\pi}(t+1)}{|\bar{\pi}(t+1)|_1} \)
Topic-Sensitive PageRank

- **The Motivation**
  - Provide greater importance to some topics than others

- **The Procedure**
  - Fix a list of topics, and determine a high-quality sample of pages from each topic
  - Teleportation is only performed on this sample set of Web documents belonging to a specific topic

\[
\bar{\pi} = \alpha \bar{e}_p / n_p + (1 - \alpha) P^T \bar{\pi}
\]

✓ $\bar{e}_p$ is an indicator vector for the specific topic
SimRank (1)

□ An Asymmetric Ranking Problem
- Given a target node $i_q$ and a subset of nodes $S \subseteq N$ from graph $G = (N, A)$, rank the nodes in $S$ in their order of similarity to $i_q$
  - Very popular in bipartite graph
- A limiting case of topic-sensitive PageRank
  - The teleportation is performed to the single node $i_q$
    \[
    \bar{\pi} = \alpha \bar{e}_q + (1 - \alpha) P^T \bar{\pi},
    \]
  - $\bar{e}_q$ is a vector of all 0s, except for a single 1, corresponding to the node $i_q$
SimRank (2)

- **The Goal**
  - Compute the **structural/symmetric** similarity between nodes

- **The Definition**
  \[
  \text{SimRank}(i, j) = \frac{C}{|\text{In}(i)| \cdot |\text{In}(j)|} \sum_{p \in \text{In}(i)} \sum_{q \in \text{In}(j)} \text{SimRank}(p, q)
  \]

  - \(\text{In}(i)\): in-linking nodes of \(i\)
  - \(C \in (0,1)\) is a constant

- **Optimization**
  - \(\text{SimRank}(i, j) = 1\) if \(i = j\)
  - Apply the above equation iteratively
Hypertext Induced Topic Search (HITS)

- Authority
  - A page with many in-links
  - It contains authoritative content on a particular subject

- Hub
  - A page with many out-links to authorities
The Insight of HITS

- Good hubs point to many good authorities
- Good authority pages are pointed to by many hubs
The Procedure of HITS (1)

- Collect the top-\(r\) most relevant results to the search query at hand
  - This defines the root set \(R\)
  - \(r = 50\)

- Determine all nodes immediately connected (either in-linking or out-linking) to \(R\)
  - This provides a larger base set \(S\)
  - The number of in-linking nodes is restricted to \(k\)
  - \(k = 50\)
The Procedure of HITS (2)

- \( G = (S, A) \) be the subgraph of the Web graph defined on the base set \( S \), where \( A \) is the set of edges between nodes in the root set \( S \)

- Each page \( i \) is assigned both a hub score \( h(i) \) and authority score \( a(i) \)

\[
\begin{align*}
    h(i) &= \sum_{j: (i, j) \in A} a(j) \quad \forall i \in S \\
    a(i) &= \sum_{j: (j, i) \in A} h(j) \quad \forall i \in S.
\end{align*}
\]

- Reward hubs for pointing to good authorities and reward authorities for being pointed to by good hubs
The Procedure of HITS (3)

- An Iterative Algorithm
  - \( h^0(i) = a^0(i) = 1/\sqrt{|S|} \)
  - for each \( i \in S \) set \( a^{t+1}(i) = \sum_{j : (j, i) \in A} h^t(j) \);
  - for each \( i \in S \) set \( h^{t+1}(i) = \sum_{j : (i, j) \in A} a^{t+1}(j) \);
  - Normalize \( L_2 \)-norm of each of hub and authority vectors to 1;

- \( \vec{h} = [h(1), ..., h(n)]^\top \) and \( \vec{a} = [a(1), ..., a(n)]^\top \)

  \[
  \vec{a} = A^\top \vec{h} \quad \vec{h} = A \vec{a}
  \]

- \( \vec{a} = A^\top A \vec{a} \quad \vec{h} = AA^\top \vec{h} \)

- Eigenvectors or singular vectors
Recommender Systems

- **Data About User Buying Behaviors**
  - User profiles, interests, browsing behavior, buying behavior, and ratings about various items

- **The Goal**
  - Leverage such data to make recommendations to customers about possible buying interests
Utility Matrix (1)

For \( n \) users and \( d \) items, there is an \( n \times d \) matrix \( D \) of utility values.

- The utility value for a user-item pair could correspond to either the buying behavior or the ratings of the user for the item.

- Typically, a small subset of the utility values are specified.
Utility Matrix (2)

- For $n$ users and $d$ items, there is an $n \times d$ matrix $D$ of utility values
  - Positive preferences only
    - A specification of a “like” option on a social networking site, the browsing of an item at an online site, the buying of a specified quantity of an item, or the raw quantities of the item bought by each user
  - Positive and negative preferences (ratings)
    - The user specifies the ratings that represent their like or dislike for the item
Utility Matrix (3)

For \( n \) users and \( d \) items, there is an \( n \times d \) matrix \( D \) of utility values.

![Utility Matrix Diagram]

(a) Ratings-based utility
(b) Positive-preference utility
Types of Recommendation

- **Content-Based Recommendations**
  - The users and items are both associated with feature-based descriptions
    - The text of the item description
    - The interests of user in a profile

- **Collaborative Filtering**
  - Leverage the user preferences in the form of ratings or buying behavior in a “collaborative” way
  - The utility matrix is used to determine either relevant users for specific items, or relevant items for specific users
Content-Based Recommendations (1)

- User is associated with some documents that describe his/her interests
  - Specified demographic profile
  - Specified interests at registration time
  - Descriptions of the items bought

- The items are also associated with textual descriptions

1. If no utility matrix is available
   - $k$-nearest neighbor approach: find the top-$k$ items that are closest to the user
     - The cosine similarity with tf-idf can be used
Content-Based Recommendations (1)

- User is associated with some documents that describe his/her interests
  - Specified demographic profile
  - Specified interests at registration time
  - Descriptions of the items bought
- The items are also associated with textual descriptions

1. If no utility matrix is available
   - $k$-nearest neighbor approach: find the top-$k$ items that are closest to the user
     - The cosine similarity with tf-idf can be used
Content-Based Recommendations (2)

2. If a utility matrix is available

- Classification-Based Approach
  - Training documents representing the descriptions of the items for which that user has specified utilities
  - The labels represent the utility values.
  - The descriptions of the remaining items for that user can be viewed as the test documents

- Regression-Based Approach

☐ Limitations

- Depends on the quality of features
Collaborative Filtering

- Missing-value Estimation or Matrix Completion

\[ M = \begin{bmatrix}
\text{●} & \text{●} & \text{●} \\
\text{●} & \text{●} & \text{●} \\
\text{●} & \text{●} & \text{●}
\end{bmatrix} \in \mathbb{R}^{n \times d} \]

- The Matrix is extremely large
- The Matrix is extremely sparse
Algorithms for Collaborative Filtering

- Neighborhood-Based Methods for Collaborative Filtering
  - User-Based Similarity with Ratings
  - Item-Based Similarity with Ratings

- Graph-Based Methods

- Clustering Methods
  - Adapting $k$-Means Clustering
  - Adapting Co-Clustering

- Latent Factor Models
  - Singular Value Decomposition
  - Matrix Factorization
  - Matrix Completion
User-Based Similarity with Ratings

- A Similarity Function between Users
  - $\bar{X} = (x_1, ..., x_s)$ and $\bar{Y} = (y_1, ..., y_s)$ be the common ratings between a pair of users
  - The Pearson correlation coefficient
    \[
    \text{Pearson}(\bar{X}, \bar{Y}) = \frac{\sum_{i=1}^{s}(x_i - \hat{x}) \cdot (y_i - \hat{y})}{\sqrt{\sum_{i=1}^{s}(x_i - \hat{x})^2} \cdot \sqrt{\sum_{i=1}^{s}(y_i - \hat{y})^2}}
    \]
    - $\hat{x} = \frac{\sum_{i=1}^{s} x_i}{s}$ and $\hat{y} = \frac{\sum_{i=1}^{s} y_i}{s}$

1. Identify the peer group of the target user
   - Top-$k$ users with the highest Pearson coefficient

2. Return the weighted average ratings of each of the items of this peer group
   - Normalization is needed
Item-Based Similarity with Ratings

- A Similarity Function between Items
  - The average of each row in the ratings matrix is subtracted from that row
  - \( \bar{U} = (u_1, \ldots, u_s) \) and \( \bar{V} = (v_1, \ldots, v_s) \) are two columns of the matrix

\[
\text{Cosine}(\bar{U}, \bar{V}) = \frac{\sum_{i=1}^{s} u_i \cdot v_i}{\sqrt{\sum_{i=1}^{s} u_i^2} \cdot \sqrt{\sum_{i=1}^{s} v_i^2}}
\]

1. Determine the top-\( k \) most similar items to item \( j \)
2. Among those items, identify the ones for which user \( i \) provides ratings
3. Return the weighed average value of those ratings
Graph-Based Methods (1)

- A Bipartite User-Item Graph \( G = (N_u \cup N_i) \)
  - \( N_u \) is the set of nodes representing users
  - \( N_i \) is the set of nodes representing items
  - Each nonzero entry in the utility matrix corresponds an edge in \( A \)
Graph-Based Methods (1)

- A Bipartite User-Item Graph $G = (N_u \cup N_i)$
  - $N_u$ is the set of nodes representing users
  - $N_i$ is the set of nodes representing items
  - Each nonzero entry in the utility matrix corresponds an edge in $A$

- Combine with Previous Methods
  - Similarity Between Users/Items
    - Topic-Sensitive PageRank
    - SimRank
  - Return the weighted average
Graph-Based Methods (2)

- A Positive and Negative Link Prediction Problem
  - The normalized rating of a user for an item, after subtracting the user-mean, can be viewed as either a positive or negative weight on the edge

- A Positive Link Prediction Problem
  - Random Walk Model

1. The top ranking items for the user $i$ can be determined by returning the item nodes with the largest $PageRank$ in a random walk with restart at node $i$.

2. The top ranking users for the item $j$ can be determined by returning the user nodes with the largest $PageRank$ in a random walk with restart at node $j$. 
Clustering Methods (1)

- **Motivations**
  - Reduce the computational cost
  - Address the issue of data sparsity to some extent

- **The Result of Clustering**
  - Clusters of users
    - User-user similarity recommendations
  - Clusters of items
    - Item-item similarity recommendations
Clustering Methods (2)

- **User-User Recommendation Approach**
  1. Cluster all the users into $n_g$ groups of users using any clustering algorithm
  2. For any user $i$, compute the average (normalized) rating of the specified items in its cluster
  3. Report these ratings for user $i$

- **Item–Item Recommendation Approach**
  1. Cluster all the items into $n_g$ groups of items
  2. The rest is the same as “Item-Based Similarity with Ratings”
Adapting $k$-Means Clustering

1. In an iteration of $k$-means, centroids are computed by averaging each dimension over the number of specified values in the cluster members
   - Furthermore, the centroid itself may not be fully specified

2. The distance between a data point and a centroid is computed only over the specified dimensions in both
   - Furthermore, the distance is divided by the number of such dimensions in order to fairly compare different data points
Adapting Co-Clustering

- User-neighborhoods and item-neighborhoods are discovered simultaneously

(a) Co-cluster

(b) User-item graph
Latent Factor Models

The Key Idea

- Summarize the correlations across rows and columns in the form of lower dimensional vectors, or latent factors.
- These latent factors become hidden variables that encode the correlations in the data matrix in a concise way and can be used to make predictions.
- Estimation of the $k$-dimensional dominant latent factors is often possible even from incompletely specified data.
Modeling

- The $n$ users are represented by $n$ factors: $\overline{U}_1, ..., \overline{U}_n \in \mathbb{R}^k$
- The $d$ items are represented by $d$ factors: $\overline{I}_1, ..., \overline{I}_d \in \mathbb{R}^k$
- The rating $r_{ij}$ for user $i$ and item $j$

$$r_{ij} \approx \langle \overline{U}_i, \overline{I}_j \rangle = \overline{U}_i^T \overline{I}_j = \overline{I}_j^T \overline{U}_i$$

- The rating matrix $D = [r_{ij}]_{n \times d}$

$$D \approx F_{user} F_{item}^T$$

- $F_{user} \in \mathbb{R}^{n \times k}$ and $F_{item} \in \mathbb{R}^{d \times k}$
Singular Value Decomposition

- **SVD of** $D \in \mathbb{R}^{n \times d}$
  \[ D = Q\Sigma P^T \]
  - $Q^T Q = I$, $P^T P = I$
  - $\Sigma = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_d) \in \mathbb{R}^{d \times d}$, $\sigma_1 \geq \cdots \geq \sigma_d$

- **Truncated SVD**
  \[ D \approx Q_k \Sigma_k P_k^T \]
  - $\Sigma_k = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_k) \in \mathbb{R}^{k \times k}$, $\sigma_1 \geq \cdots \geq \sigma_k$

- **Discussions**
  - SVD is undefined for incomplete matrices
  - PLSA may be used for nonnegative matrices
Matrix Factorization (MF)

- SVD is a special form of MF
  \[ D \approx UV^T \]

- The objective when \( D \) is fully observed
  \[ J = ||D - UV^T||_F^2 \]

- The objective when \( D \) is partially observed
  \[ J = \sum_{(i,j) \in \Omega} \left( D_{ij} - U_i^T V_j \right)^2 \]

  - \( \Omega \) is the set of observed indices
  - Constrains can be added: \( U \geq 0 \) and \( V \geq 0 \)
Matrix Factorization (MF)

- SVD is a special form of MF
  \[ D \approx UV^T \]

- The objective when \( D \) is fully observed
  \[ J = \|D - UV^T\|_F^2 \]

- The objective when \( D \) is partially observed
  \[ J = \sum_{(i,j) \in \Omega} (D_{ij} - U_i^T V_j)^2 + \lambda (\|U\|_F^2 + \|V\|_F^2) \]

  - \( \Omega \) is the set of observed indices
  - Constrains can be added: \( U \geq 0 \) and \( V \geq 0 \)
  - Regularization can also be introduced
Matrix Completion

- Assuming the Utility matrix is low-rank

\[ M = \begin{bmatrix}
\text{\ldots} & \text{\ldots} & \text{\ldots} \\
\text{\ldots} & \text{\ldots} & \text{\ldots} \\
\text{\ldots} & \text{\ldots} & \text{\ldots} \\
\end{bmatrix} \in \mathbb{R}^{n \times d} \]

- The Optimization Problem

\[
\begin{aligned}
\min_{X \in \mathbb{R}^{n \times d}} & \quad \text{rank}(X) \\
\text{s.t.} & \quad X_{ij} = M_{ij}, \forall (i,j) \in \Omega
\end{aligned}
\]

\[
\begin{aligned}
\min_{X \in \mathbb{R}^{n \times d}} & \quad \|X\|_* \\
\text{s.t.} & \quad X_{ij} = M_{ij}, \forall (i,j) \in \Omega
\end{aligned}
\]

- \( \Omega \) is the set of observed indices
Outline

- Introduction
- Web Crawling and Resource Discovery
- Search Engine Indexing and Query Processing
- Ranking Algorithms
- Recommender Systems
- Web Usage Mining
- Summary
Types of Logs

- **Web Server Logs**
  - User activity on Web servers
  - Stored in *NCSA common log format* or its variants

```
```

- **Query Logs**
  - Queries posed by a user during search
Data Preprocessing

- Data in the Log File
  - A continuous sequence of entries that corresponds to the user accesses
  - The entries for different users are typically interleaved with one another randomly

- Distinguish between different user sessions
  - Client-side cookies, IP address, user agents

- A subset of users can be identified
  - A set of sequences in the form of page views (click streams), or search tokens
Applications

- **Recommendations**
  - Recommend Web pages based on browsing patterns

- **Frequent Traversal Patterns**
  - Web site reorganization

- **Forecasting and Anomaly Detection**
  - Forecast future clicks of the user
  - Identify unusual clicks or patterns

- **Classification**
  - Label (shopping, intrusion) the sequence
Outline

- Introduction
- Web Crawling and Resource Discovery
- Search Engine Indexing and Query Processing
- Ranking Algorithms
- Recommender Systems
- Web Usage Mining

Summary
Summary

- **Web Crawling and Resource Discovery**
  - Universal, Preferential, Multiple Threads, Spider Traps, Near Duplicate Detection

- **Search Engine Indexing and Query Processing**
  - Content-based score, reputation-based scores

- **Ranking Algorithms**
  - PageRank and its variants, HITS

- **Recommender Systems**
  - Content-Based, Collaborative Filtering (Neighborhood-Based, Graph-Based, Clustering, Latent Factor Models)

- **Web Usage Mining**
  - Data Preprocessing, Applications