Web Usage Mining: Recommendation

Outline

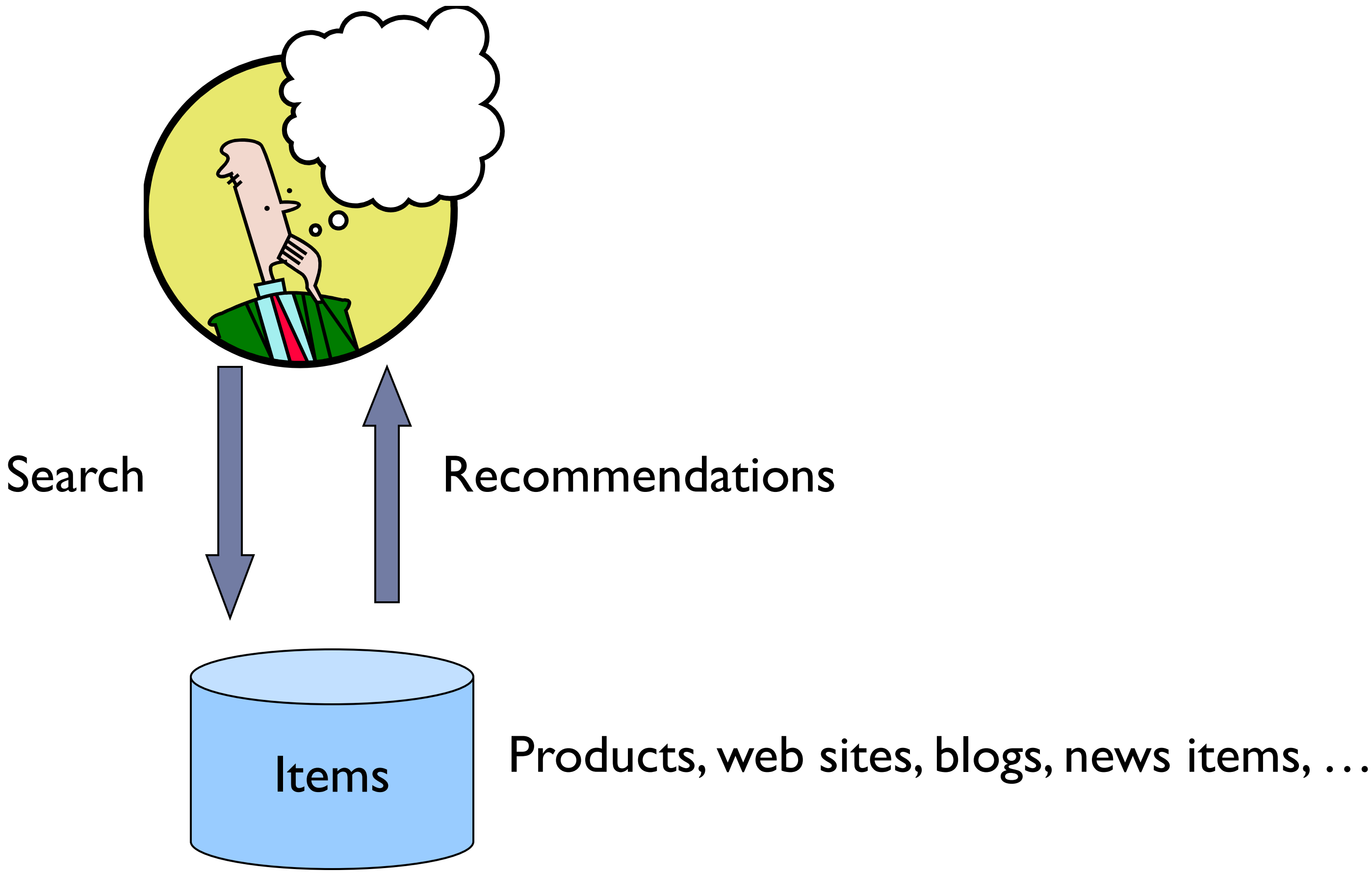
- Introduction to recommendation
- Content-based recommendation
- Collaborative filtering
- Remarks and practical tips

So, what do you want?

Why Recommendation?

- Large quantity
- Diversified quality

Recommendations



From scarcity to abundance

- Shelf space is a scarce commodity for traditional retailers
  - Also: TV networks, movie theaters,…
- The web enables near-zero-cost dissemination of information about products
  - From scarcity to abundance
- More choice necessitates better filters
  - Recommendation engines
  - How Into Thin Air made Touching the Void a bestseller (http://www.wired.com/wired/archive/12.10/tail.html)

Recommendation Types

- Editorial
  - List of favorites
  - Lists of “essential” items
- Simple aggregates
  - Top 10, Most Popular, Recent Uploads
- Tailored to individual users
  - Amazon, Netflix, Taobao.com…

Formal Model

- X = set of Customers
- S = set of Items
- Utility function $u: X \times S \rightarrow R$
  - R = set of ratings
  - R is a totally ordered set
  - e.g., 0-5 stars, real number in [0,1]

Utility Matrix

	King Kong	LOTR	Matrix	Pirates
Alice	1		0.2	
Bob		0.5		0.3
Carol	0.2		1	
David				0.4

Key Problems

- Gathering "known" ratings for matrix
  - How to collect the data in the utility matrix
- Extrapolate unknown ratings from known ratings
  - Mainly interested in high unknown ratings
    - We are not interested in knowing what you don't like but what you like
- Evaluating extrapolation methods
  - How to measure success/performance of recommendation methods

Gathering Ratings

- Explicit
  - Ask people to rate items
  - Doesn't work well in practice – people can't be bothered
- Implicit
  - Learn ratings from user actions

    e.g., purchase implies high rating
  - What about low ratings?

Extrapolating Utilities

- Key problem: matrix *U* is sparse
  - Most people have not rated most items
  - Cold start:
    - New items have no ratings
    - New users have no history
- Three approaches to recommender systems
  - Content-based
  - Collaborative
  - Latent factor based

Outline

- Introduction to recommendation
- Content-based recommendation
- Collaborative filtering
- Remarks and practical tips

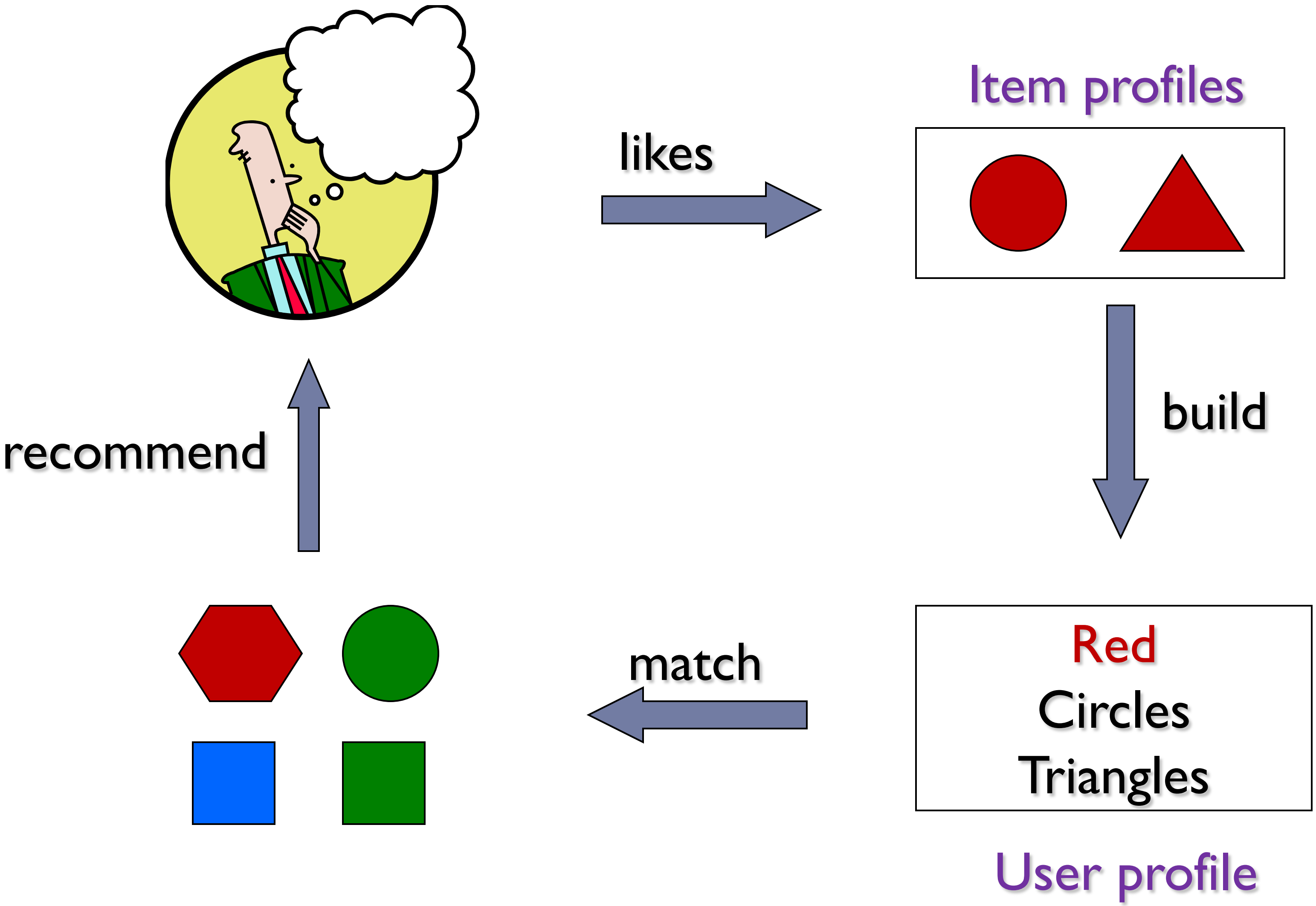
Content-based Recommendations

- Main idea: recommend items to customer *x* similar to previous items rated highly by *x*

Example:

- Movie recommendations
  - Recommend movies with same actor(s), director, genre, …
- Websites, blogs, news
  - Recommend other sites with "similar" content

Plan of Action



Item Profiles

- For each item, create an item profile
- Profile is a set (vector) of features
  - Movies: author, title, actor, director,…
  - Text: set of "important" words in document
- How to pick important words?
  - Usual heuristic from text mining is TF.IDF (Term Frequency * Inverse Doc Frequency)
    - Term … Feature
    - Document … Item

User Profiles and Prediction

- User profile possibilities:
  - Weighted average of rated item profiles
  - **Variation:** weight by difference from average rating for item
  - …
- Prediction heuristic
  - Given user profile *x* and item profile *i*, estimate

$$u(\mathbf{x}, \mathbf{i}) = \cos(\mathbf{x}, \mathbf{i}) = \frac{\mathbf{x} \cdot \mathbf{i}}{||\mathbf{x}|| \cdot ||\mathbf{i}||}$$

Pros: Content-based Approach

- No need for data on other users
  - No cold-start or sparsity problems
- Able to recommend users with unique tastes
- Able to recommend new and unpopular items
  - No first-rater problem
- Able to provide explanations
  - Can provide explanations of recommended items by listing content-features that caused an item to be recommended

Cons: Content-based Approach

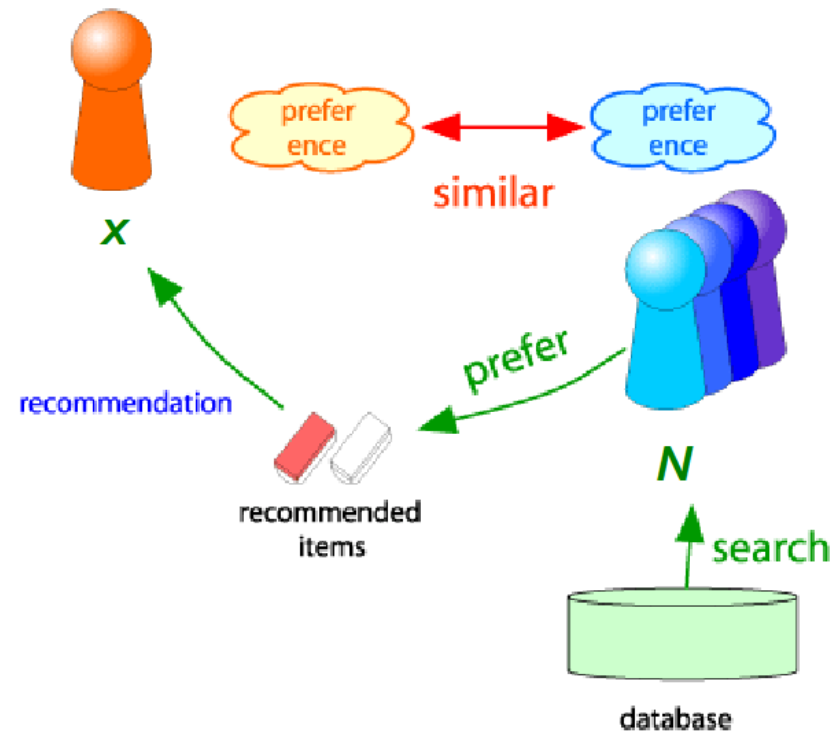
- Finding the appropriate features is hard
  - E.g., images, movies, music
- Overspecialization
  - Never recommends items outside user's content profile
  - People might have multiple interests
  - Unable to exploit quality judgments of other users
- Recommendations for new users
  - How to build a user profile?

Outline

- Introduction to recommendation
- Content-based recommendation
- Collaborative filtering
- Remarks and practical tips

Collaborative Filtering（协同过滤）

- Consider user *x*
- Find set *N* of other users whose ratings are "similar" to *x*'s ratings
- Estimate *x*'s ratings based on ratings of users in *N*



Similar Users

- Let r_x be the vector of user *x*'s ratings
- Jaccard similarity measure
  - Problem: ignores the value of the rating
- Cosine similarity measure
  - sim(x,y) = cos(r_x , r_y)
  - Problem: treats missing ratings as "negative"
- Pearson correlation coefficient
  - S_{xy} = items rated by both users ***x*** and ***y***

$$sim(x,y) = \frac{\sum_{s \in S_{xy}} (r_{xs} - \bar{r_x})(r_{ys} - \bar{r_y})}{\sqrt{\sum_{s \in S_{xy}} (r_{xs} - \bar{r_x})^2 (r_{ys} - \bar{r_y})^2}}$$

Rating Predictions

- Let r_x be the vector of user x's ratings
- Let N be the set of k users most similar to x who have rated item i

- Prediction for item s of user x:
  - $r_{xi} = \frac{1}{k} \sum_{y \in N} r_{yi}$
  - $r_{xi} = \frac{\sum_{y \in N} s_{xy} r_{yi}}{\sum_{y \in N} s_{xy}}$ $\quad s_{xy} = sim(x,y)$
  - Other options?
- Many other tricks possible…

Item-Item Collaborative Filtering

- So far: User-user collaborative filtering
- Another view: Item-item
  - For item *i*, find other similar items
  - Estimate rating for item *i* based on ratings for similar items
  - Can use same similarity metrics and prediction functions as in user-user model

$$r_{xi} = \frac{\sum_{j \in N(i;x)} s_{ij} r_{xj}}{\sum_{j \in N(i;x)} s_{ij}}$$

s_{ij}… similarity of items i and j
r_{xj}…rating of user u on item j
N(i;x)… set of items rated by x similar to i

CF: Common Practice

- Define similarity s_{ij} of item i and j
- Select k nearest neighbors $N(i; x)$
  - Items most similar to i, that were rated by x
- Estimate rating as the weighted average:

$$r_{xi} = b_{xi} + \frac{\sum_{j \in N(i;x)} s_{ij} \cdot (r_{xj} - b_{xj})}{\sum_{j \in N(i;x)} s_{ij}}$$

  - $\boldsymbol{b}_{xi}$: baseline estimate for r_{xi}
  - $\boldsymbol{b}_{xi} = \boldsymbol{\mu} + \boldsymbol{b}_x + \boldsymbol{b}_i$
  - $\boldsymbol{\mu}$=overall mean movie rating
  - $\boldsymbol{b}_x$=rating deviation of user x = (average rating of user x)-$\boldsymbol{\mu}$
  - $\boldsymbol{b}_i$=rating deviation of movie i

Item-Item vs. User-User

- In practice, it has been observed that item-item often works better than user-user

- Why?
  - Items are simpler, users have multiple tastes

Pros and Cons of Collaborative Filtering

☺Works for any kind of item

- No feature selection needed

☹Cold start:

- Need enough users in the system to find a match

☹Sparsity:

- The user/ratings matrix is sparse
- Hard to find users that have rated the same items

☹First rater:

- Cannot recommend an item that has not been previously rated
- New items, esoteric items

☹Popularity bias:

- Cannot recommend items to someone with unique taste
- Tends to recommend popular items

Hybrid Methods

- Implement or more different recommenders and combine predictions

- Add content-based methods to collaborative filtering
  - Item profiles for new item problem
  - Demographics（人口统计学） to deal with new user problem

Outline

- Introduction to recommendation
- Content-based recommendation
- Collaborative filtering
- Remarks and practical tips

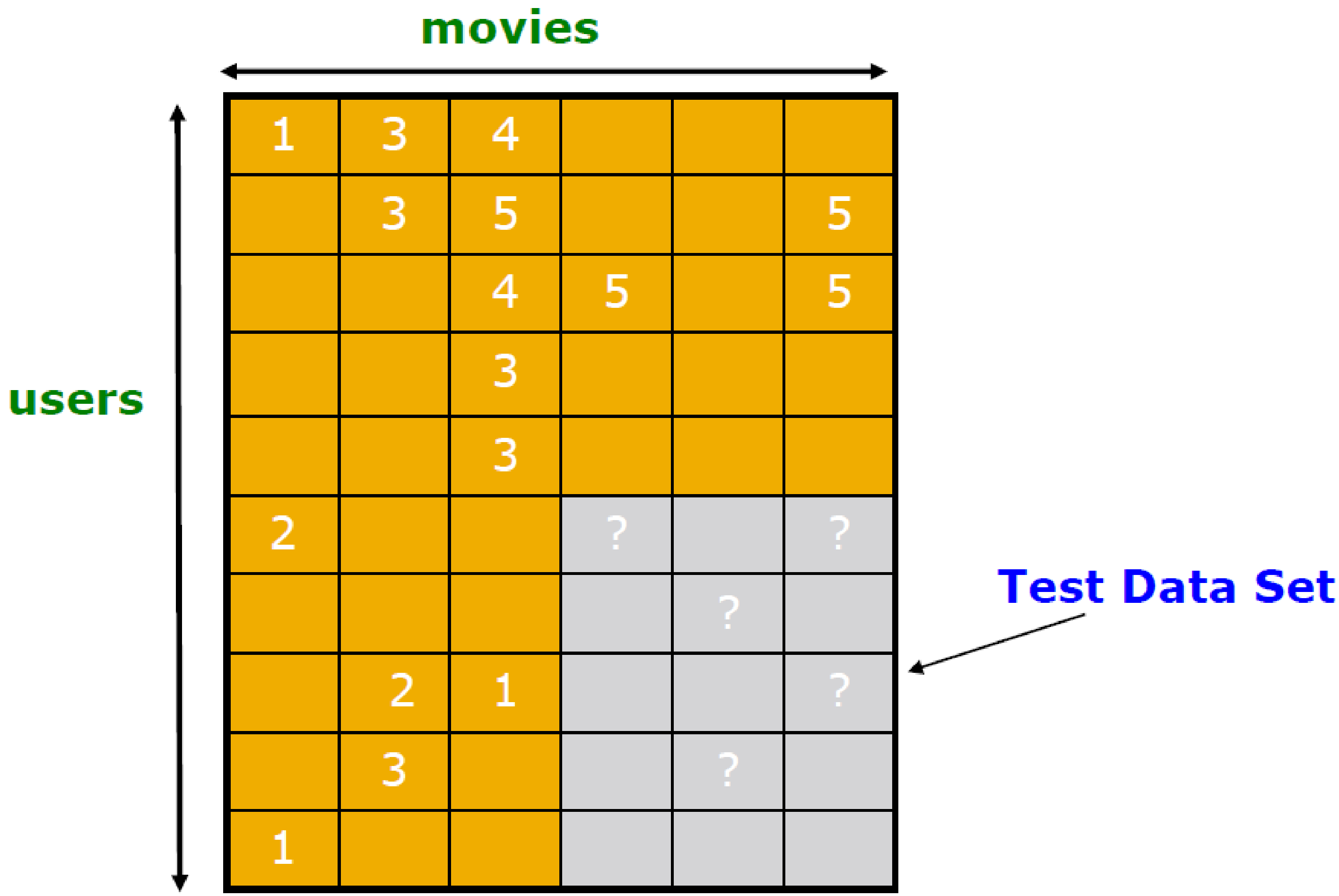
Evaluation

movies

users

1	3	4			
	3	5			5
		4	5		5
		3			
		3			
2			2		2
				5	
	2	1			1
	3			3	
1					

Evaluation



Evaluating Predictions

- Compare predictions with known ratings
  - Root-mean-square error (RMSE)
    - $\sqrt{\sum_{xi}(r_{xi} - r^*_{xi})^2}$ where r_{xi} is predicted, r^*_{xi} is the true rating
  - Precision at top 10:
    - % of those in top 10
- Another approach: 0/1 model
  - Coverage
    - Number of items/users for which system can make predictions
  - Precision
    - Accuracy of predictions
  - Receiver operating characteristic (ROC)
    - Tradeoff curve between false positives and false negatives

Problems with Measures

- Narrow focus on accuracy sometimes misses the point
  - Prediction Diversity
  - Prediction Context（情境）
  - Order of predictions
- In practice, we care only to predict high ratings
  - RMSE might penalize a method that does well for high ratings and badly for others

Collaborative Filtering: Complexity

- Expensive step is finding *k* most similar customers: $O(|X|)$
- Too expensive to do at runtime
  - Need to pre-compute
- Naïve pre-computation takes time $O(|S| \cdot |X|)$
  - Near-neighbor search in high dimensions
- Can use clustering, partitioning as alternatives, but quality degrades

The Netflix Prize

- Training data
  - 100 million ratings, 480,000 users, 17,770 movies
  - 6 years of data: 2000-2005
- Test data
  - Last few ratings of each user (2.8 million)
  - Evaluation criterion: root mean squared error (RMSE)
  - Netflix Cinematch RMSE: 0.9514
- Competition
  - 2700+ teams
  - $1 million prize for 10% improvement on Cinematch

Major Challenges in Recommendation

- Data Sparsity
- Scalability
- Cold Start
- Diversity vs. Accuracy
- Vulnerability to Attacks
- Value of Time
- Evaluation of Recommendations
- User Behavior mining
- User Interface
- Social-based recommendation
- Multi-resource data

Factors that Influence Recommendation

- Temporal
- Spatial / location
- Social
- Trust

…

Summary of Web Mining

Roadmap

- Web Mining
  - Content Mining
    - Classification, Clustering
  - Structure Mining
    - Social network analysis
  - Usage Mining
    - Recommendation …

Note: Helpful to combine usage with content and structure

Introduction

- 网络挖掘的概念，包含哪些方面的内容，分别有哪些重要应用?

Data

- 概念：数据对象(Objects)，属性(Attributes)，维度(Dimensions)，特征(features)

- 高维诅咒(Curse of dimensionality)现象。

- 对于数据的预处理有哪些方法？其中需要掌握采样(Sampling)，特征选择(Feature selection)及降维(Dimensionality reduction)的基本原理。

Classification

- 监督学习(Supervised learning)与无监督学习(Unsupervised learning)的关系与区别。
- 分类(Classification)的基本原理。
- 数据的向量表示(Vector space representation)
- 熟练掌握k近邻算法，包括影响算法性能的要素——近邻个数及距离（相似度）度量。
- 熟练掌握Logistic regression分类方法。
- 如何评价分类效果？理解训练错误率，测试错误率以及泛化错误率的区别。

Clustering

- 聚类(Clustering)的基本原理及准则。
  - High similarity within clusters
  - Low similarity between clusters
  - Important issues of clustering:
    - Number of clusters, Similarity measure

- 层次式聚类算法流程，两个类之间的距离定义。

- 熟练掌握K-means算法——算法流程，优化目标，收敛性分析。

- 聚类算法的评价标准。
  - With or without ground truth

Networks: Community

- 社区(Community)的概念
- 社区发现与聚类的关系。
- 如何计算结构相似度?
- 图分析的一些重要矩阵：邻接(Affinity)矩阵，拉普拉斯(Laplacian)矩阵，以及它们的一些重要性质。
- Cut概念；ratio cut以及normalized cut的定义及推导。
- Modularity概念及其推导。与spectral clustering的相同点及不同点。

Point: analyze the eigenvectors of a matrix to explore the structure of the graph

Networks: Influence

- 几种度量节点中心性的标准。
- 两种影响力传播模型——线性阈值模型(Linear Threshold Model)，层级传播模型(Independent Cascade Model)的传播过程及区别。
- 最大影响节点集(Most influential set)——问题建模，贪心算法以及算法的近似度。
- 子模性质(submodularity)。

Recommendation

- 推荐问题的形式。
- 基于内容的推荐：主要思想。
- 协同过滤：主要思想和基本方法。
- 各自的优缺点。