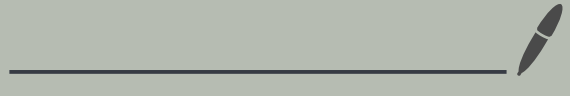
Notes for Recommender Systems Handbook



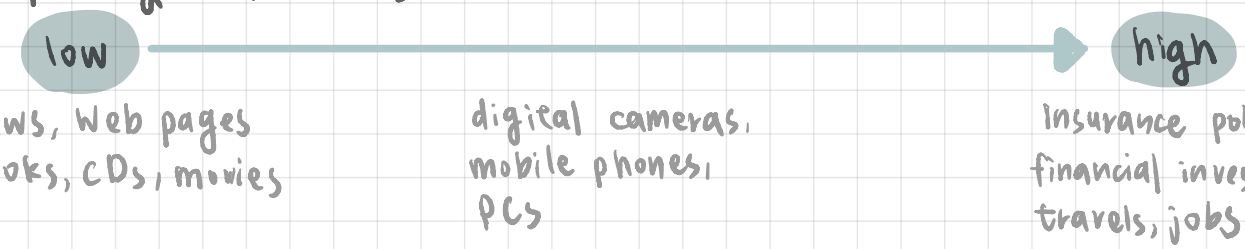
1. Introduction

1.2. Recommender systems functions

1. **Increase the number of item sold**: increase the conversion rate.
2. **Sell more diverse items**: suggest or advertises unpopular items to users.
3. **Increase the user satisfication**: improve the user experience.
4. **Increase user fidelity**
5. **Better understand what user wants**: can be leveraged to many other applications, is the description of the user's preferences, either collected explicity or predicted by system.

1.3. Data and Knowledge source.

1. **Items**.
   - complexity and values



   - can be represented as a **single id code**, **set of attributes**, **ontological representation**.
2. **Users**.
   - profiles users. e.g. **demographic attributes**, **behavior pattern data**.
3. **Transactions**
   - interaction between user and RS
   - **ratings** are most popular form of transaction data that RS collect.
     1. numerical ratings ex. 1~5 star
     2. ordinal ratings ex. strongly agree, agree, neutral, disagree, ...
     3. binary ratings ex. good / bad
     4. unary ratings ex. user has observed or purchased an item, or rated the item positively.
     5. tags ex. in MovieLens users feel about a movie "too long", "acting"

1.4. Recommendation Techiques.

- a simple, non-personalized recommendation algorithm is **recommend the most popular item.**
- **model the degree of utility** of user u for the item i as a function R(u,i), predict the value of R over pairs of users and items $\hat{R}(u,i)$, having computed prediction for user u on a set of items $\hat{R}(u,i_1), \ldots \hat{R}(u,i_N)$, the system will recommend the items $i_{j_1}, \ldots, i_{j_K}$ with largest $\hat{R}$, and K is typically much smaller than the number of the item.
- **some recommender systems do not fully estimate the utility**, e.g. knowledge-based system.
- taxonomy of recommendation approaches.
  1. **Content-based**
     - recommend items that are similar to the ones that user liked in the past.
  2. **Collaborative filtering**.
     - most popular and widely implement.
     - recommend to user the items that other users with similar tastes liked in the past.
     - similarity in taste: based on the similarity in the rating history.
     - refers to **people-to-people correlation**.
  3. **Demographic**
     - recommends items **based on the demographic profile** of the user.
     - relatively little research into demographic systems.
  4. **Knowledge-based**
     - recommends items **based on specific domain knowledge**.
     - **case-based**: **similarity function estimates** how much the user needs match the recommendation (utility)
     - **constraint-based**: predominately exploit **predefined knowledge** base that contain **explicit rules** about how to relate users and items.
     - tends to **work better than others at the beginning**, but may be surpassed by other methods that can exploit the logs of human / computer interaction.

5. community-based. (social recommender systems)
   - recommends items based on the preferences of the users friends.
   - "Tell me who your friends are, and I will tell your who you are."
6. Hybrid.
   - combination of the above techniques, use the advantages of A to fix the disadvantage of B.

- paradigms for corporating contextual information into the recommendation process.
  1. reduction-based (pre-filtering): only the information that matches the current usage context are used to compute the recommendation.
  2. contextual post filtering: recommendation algorithm ignores the context information. the output the algorithm is filtered / adjusted to include only the recommendations that are relavent.
  3. contextual modeling: context data is explicitly used in the prediction model.

1.5, Application and Evaluation.

- common applications:
  - Entertainment - movies, music, IPTV.
  - Content - news, documents, Web pages, e-learning applications, e-mail.
  - E-commerce - books, cameras, PCs
  - Services - travel services, experts for consultation, house to rent, matchmaking services.
- analysis of the available sources of knowledge → decide algorithm.
- evaluation is required at different stages of the system life cycle.
  - At design time: verify the selection of the appropriate recommender approach.
  - In the design face: off-line evaluation consists of running several algorithms and comparing their performance. off-line evaluation should follow experiment design.

- After system launched: on-line evaluation. If on-line evaluation is not feasible or too risky, a controlled experiment can be planned.

1.6. Recommender Systems and Human Computer Interaction.

- 7 roles that can be played by explaination in RSs.
  1. transparency - explaining how the system works.
  2. scrutability - allowing users to tell the system it is wrong.
  3. trust - increasing user confidence in the system.
  4. effectiveness - helping users make good decisions.
  5. persuasiveness - convincing users to try or buy.
  6. efficiency - helping users make decision faster.
  7. satisfaction - increasing the ease of use and enjoyment.

- Conversational RSs.
  - critiquing-based interface (dialogues model): given an initial set of user preferences, have attracted great interest in domain where there is a need for more sophisticated and interactive decision/recommendation.
  - preference-based.

Part I Basic Techniques

2. Data Mining Methods for Recommender Systems.

2.2. Data Preprocessing.

- similarity measures.
  - **Euclidean distance**: $d(x,y)=\sqrt{\sum_{k=1}^{n}(x_k-y_k)^2}$
  - **Minkowski distance**: $d(x,y)=\left(\sum_{k=1}^{n}|x_k-y_k|^r\right)^{\frac{1}{r}}$

    r: degree of distance

    $r=1$, city block / Manhattan / taxicab / L_1 norm

    $r=2$, euclidean distance

    $r\to\infty$, supremum / L_{max}-norm / L_∞-norm
  - **Mahalanobis distance**: $d(x,y)=\sqrt{(x-y)\Sigma^{-1}(x-y)^T}$

    Σ: covariance matrix of data
  - **cosine similarity**: $\cos(x,y)=\frac{(x\cdot y)}{\|x\|\|y\|}$
  - **Pearson correlation**: $\text{Pearson}(x,y)=\frac{\Sigma(x,y)}{\sigma_x\sigma_y}$, Σ: covariance, σ: s.d.
- similarity measures (binary)
  - **Simple matching coefficient**

    $SMC=\frac{\#\text{ of matches}}{\#\text{ of attributes}}=\frac{M_{00}+M_{11}}{M_{01}+M_{10}+M_{00}+M_{10}}$

	$y=0$	$y=1$
$x=0$	M_{00}	M_{01}
$x=1$	M_{10}	M_{11}
  - Jaccard coefficient $JC=\frac{M_{11}}{M_{01}+M_{10}+M_{11}}$
  - Extended Jaccard (Tanimoto) coefficient

    $d=\frac{x\cdot y}{\|x\|^2+\|y\|^2-x\cdot y}$
- Reducing Dimesionality.

  It is common in RS to have a dataset with features that defines a **high dimensional** space with very **sparse information**.
  - **Principal Component Analysis (PCA)**

    PCA allows to obtain an **ordered list of components that account**

for the largest amount of the variance.

PCA allows us to retrieve the original data matrix by projecting the data onto the new coordinate system: $X'_{n\times m'} = X_{n\times m} W_{m\times m'}$.

The new data matrix X' contains most of the information of the original X with a dimensionality reduction of $m-m'$.

limitations 1. empirical dataset should be a linear combination of a certain basis.

2. original dataset has been drawn from a Gaussian distribution.

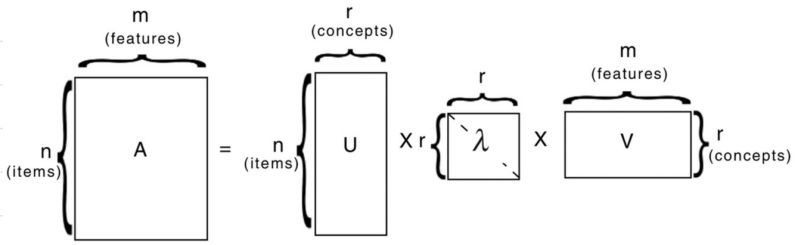
- Singular Value Decomposition (SVD)

SVD is to find a lower dimensional feature space where the new features represent "concept" and each concept is computable.

It can be used as the basis of latent-semantic analysis.

It is always possible to decompose a given matrix A into

$$A = U\lambda V^T$$



- U: item-to-concept.
- λ: strength of each concept.
  a diagonal matrix contains the singular values, which will alway be positive and sorted in decreasing order.
- V: term-to-concept.

The original matrix can be approximated by simply truncating the eigenvalue at a given k. $A_k = U_k \lambda_k V_k^T$. A_k is the closest rank-k matrix to A, "closest" means that A_k minimizes the sum of the squares of A_k and A.

- Denoising

1. natural noise: unvoluntarely introduced by users.
2. malicious noise: deliberately introduced in a system in order to bias the results.

2.3. Classification.

- **Nearest Neighbors (kNN)**

  kNN classifier finds the **k closest points** (nearest neighbors) from the training records, it then assigns the class label according to the class labels of its nearest neighbors.

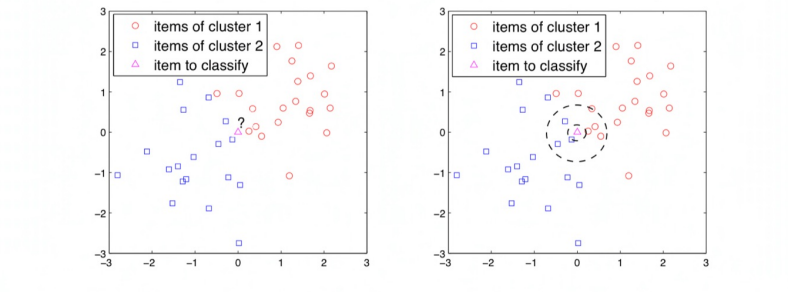


Fig. 2.4: Example of k-Nearest Neighbors. The left subfigure shows the training points with two class labels (circles and squares) and the query point (as a triangle). The right sub-figure illustrates closest neighborhood for $k = 1$ and $k = 7$. The query point would be classified as square for $k = 1$, and as a circle for $k = 5$ according to the simple majority vote rule. Note that the query points was just on the boundary between the two clusters.

challenging: How to choose the value of k?

- k too small: classifier will be sensitive to noise point.
- k too large: neighborhood might include too many points from other classes.

- **Decision Trees.**

advantages:

- inexpensive to construct.
- extremely **fast**.
- can be used to produce a set of rules that are **easy to interpret**.

- usage in RS:

  1. used as a **filter** to select which users should be targeted with recommendations.
  2. used as a tool for **item ranking**.

- **Ruled-based Classifiers.**

- a collection of **"if... then..."** rules.

definition

- **cover**: if the attributes of the instance satisfy the rule condition.
- **coverage**: fraction of records that satisfy its antecedent.

- **accuracy**: fraction of records that satisfy both the antecedend and the consequent.
- **mutually exclusive rules**: rules are independent of each other. every record is covered by at most one rule.
- **exhaustive rules**: each record is covered by at least one rule.

advantages:
- easy to interpret.
- easy to generate.
- classify new instances efficiently.

- **Bayesian Classifier**
  - based on **conditional probability** and **Bayes theorem**.
  - **posterior ∝ likelihood × prior**
    likelihood: effect of data
    prior: belief in the model before the data was observed.
    posterior: probability of a model given the data.
  - Given N attributes $(A_1, A_2, \dots, A_N)$, the goal is to predict class C_k by finding the value of C_k that maximizes the posterior of the class given the data $P(C_k | A_1, A_2, \dots, A_N)$. Applying Bayes' theorem, $P(C_k | A_1, A_2, \dots, A_N) \propto P(A_1, A_2, \dots, A_N | C_k) P(C_k)$
  - **Naive Bayes Classifier**
    - assumes the **probabilistic indepence** of the attributes.
      $\Rightarrow P(A_1, A_2, \dots, A_N | C_k) = P(A_1 | C_k) P(A_2 | C_k) \dots P(A_N | C_k)$

    advantage:
    - **robust** to isolated noise points and irrelevant attributes.
    - can **handle missing values** by ignoring the instance during probability estimation.

    disadvantage: independence assumption is too strong.
  - often used for **content-based** RS.
- **Artificial Neural Network**

advantage: can perform **non-linear** classification task.

disadvantage: ANN belongs to sub-symbolic classifier, they provide no sematics for inferring knowledge.

- Support Vector Machines (SVM)
  - find a linear hyperplane (decision boundary) that seperates the data in such way that the margin is maximized.

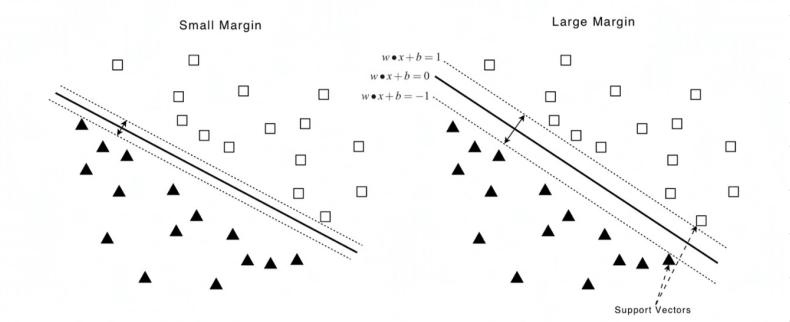


Fig. 2.6: Different boundary decisions are possible to separate two classes in two dimensions. Each boundary has an associated margin.

$$f(x) = \begin{cases} 1, & \text{if } w \cdot x + b \geq 1 \\ -1, & \text{if } w \cdot x + b < -1 \end{cases}$$

$\Rightarrow$ constrained optimization problem.

$$\text{margin} = \frac{2}{\|w\|^2}$$

  - If not linear seperable, introduce slack variable for soft margin, or use kernel trick.
  - soft margin

$$f(x) = \begin{cases} 1, & \text{if } w \cdot x + b \geq 1 - \xi \\ -1, & \text{if } w \cdot x + b \leq -1 + \xi \end{cases}$$

$$\text{minimize } L(w) = \frac{\|w\|^2}{2} + C \sum_{i=1}^{N} \xi$$ ← slack variable.

- Essembles of Classifiers.
  - construct a set of classifiers → aggregating their predictions.
  - Bagging
    - perform sampling with replacement, build classifier on each bootstrap sample.
    - each sample has probability $(1 - \frac{1}{N})^N$ of being selected.
      if $N \to \infty$, $(1 - \frac{1}{N})^N \to 0.623$.
  - Boosting
    - iterative procedure

- adaptively change distribution of training data
- focusing more on previously misclassified records.

- Evaluating Classifiers.
  - focus only on classification performance.

		Acutual	
		Neg	Pos
pred	Neg	TN	FN
	Pos	FP	TP

- Accuracy: $\frac{TP+TN}{TP+TN+FP+FN}$
- Precision: $\frac{TP}{TP+FP}$
- Recall: $\frac{TP}{TP+FN}$
- F_1-score: $\frac{2RP}{R+P} = \frac{2TP}{2TP+FN+FP}$

  - Receiver Operating Characteristic Curve (ROC curve)

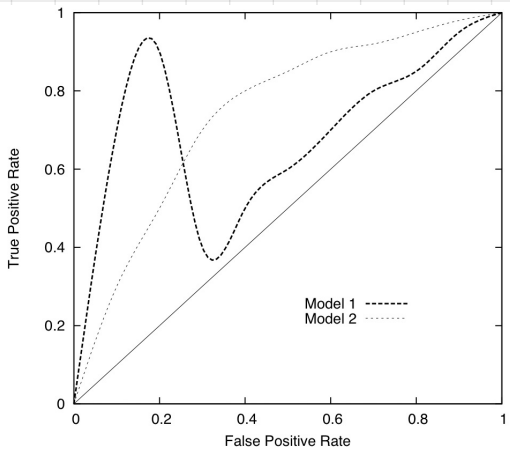


Fig. 2.7: Example of ROC Curve. Model 1 performs better for low False Positive Rates while Model 2 is fairly consistent throughout and outperforms Model 1 for False Positive Rates higher than 0.25

2.4 Cluster Analysis.

- assigning items into groups, items in the same groups are more similar than items in different groups.
- minimize intra-cluster distances.
  maximize inter-cluster distances.
- { partitional : divide data into non-overlapping clusters.
  { hierarchical : producing nested cluster organized as a hierarchical tree.

- k-Means
  - belongs to partitioning method.

- iterative process to minimize $E = \sum_{j=1}^{k} \sum_{n \in s_j} d(x_n, \lambda_j)$

  x_n: vector representation of n-th item.

  λ_j: centroid of the item in s_j

  d: distance measure.

  step 1: randomly selecting k centroids.

  step 2: all items are assigned to the cluster whose centroid is the closest to them.

  step 3: update new cluster centroids.

  step 4: repeat 2,3 until there are no further items that change cluster. stopping condition is often change to "until relatively few points change clusters" to improve convergence speed.

  advantage: extremely simple and efficient.

  disadvantage: (1) how to choose appropriate k.
  (2) sensitive to the initial centroids.
  (3) may produce empty cluster.
  (4) can not handle clusters with differing size, density and non-globular shape.
  (5) not robust to outlier.

- Density-based clustering.

  DBSCAN

  density: number of points within a specified radius

  (1) core point: more than a specified number of neighbors within a given distance.

  (2) border point: fewer than a specified number of neighbors but belong to core points' neighbor.

  (3) noise point: neither core or border.

  iteratively removes noise points and perform clustering.

- Message-passing clustering.
  - graph-based clustering methods.
  - Affinity Propagation

- Hierarchical Clustering.
  - does not have to assume number of cluster in advance.
  - agglomerative: start with individual point as per cluster, then merge the closest pair of cluster.
    divisive: start with one, all-inclusive cluster, then split cluster.

2.5 Association Rule Mining.

- means co-occurrence but not causality.

definition: (1) **itemset**: collection of one or more items. e.g. (Milk, Beer, Diaper)

(2) **k-itemset**: itemset that contains k items.

(3) **support count**: frequency of an itemset.
e.g. (Milk, Beer, Diaper) = 131.

(4) **support**: the fraction of transactions that contain a itemset
e.g. (Milk, Beer, Diaper) = 0.12

(5) **minsup**: minimum support, a threshold.

(6) **frequent itemset**: itemset with support ≥ minsup.

(7) **rule**: expression of the form $X \Rightarrow Y$, X, Y are itemsets.

(8) **support of rule**: the fraction of transactions that both have X and Y.

(9) **confidence**: how often items in Y appear in transaction that contain X.

- Given a set of transaction T, finding all rules that:
  (1) **support ≥ minsup**
  (2) **confidence ≥ minconf**
  - brute-force approach: list all possible association rules, and prune rules that do not satisfy conditions.
    ⇒ **computationally expensive** !!
  - take two-step approach:
    (1) **Frequent Itemset Generation**
    generate all itemsets whose support ≥ minsup
    (2) **Rule Generation**
    generate high confidence rules from frequent itemsets.

- Apriori Principle.
  - use to reduce the number of candidate.
  - if an itemset is frequent ⇒ all subsets are frequent
  - the support of an itemset never exceeds that of its subset.