Preference Learning in Recommender Systems

M. de Gemmis L. Iaquinta P. Lops C. Musto F. Narducci
G. Semeraro

Department of Computer Science
University of Bari, Italy

ECML/PKDD workshop on Preference Learning, 2009

Outline

Background and Motivation

Information Overload Problem

- Large quantity of information makes retrieval a hard task
- Dynamic and heterogeneous nature (such as in the Web) makes user overwhelmed

Needs

- Intelligent information access
- Personalized support in sifting through large amounts of available information
    - according user interests and preferences

Information Filtering and User Profile

- Information Filtering systems rely on user model (profile) to be tailored on user needs
- Personalized recommendations require
  - a user profile
  - an algorithm to update the profile given usage/input information
  - an adaptive tool that exploits the profile in order to provide personalization
- Profile content depends on the goals of the system and the recommendation approach

Recommender Systems

- Everyday we get advices from other people
  - “Hey, check out this Web site”
  - “I saw this book, you will like it”
  - “That restaurant is very good!”
- When making a choice in the absence of decisive first-hand knowledge, choosing as other like-minded people have chosen in the past may be a good strategy
- Recommender systems have the same role as human recommendations: they present information that they perceive to be useful and worth trying out

Collaborative Recommender Systems

- Recommendations are based on evaluations of users who share similar interests
- Assumption: a set of users which liked the same items in the past probably share the same preferences
- Thus, recommendations concern items
  - unseen by the active user
  - liked by other users with similar tastes
- Opinions on items expressed
  - as explicit user ratings
    on some scale ranging from bad to good
  - as implicit ratings given by user behavior
- User profiles consists of ratings

Recommendation approach

- User profiles are compared to find overlaps in interests among users
- The nearest neighbor approach
  - finds similar users
  - creates the nearest neighbors set for each user
  - infers the like degree for an unseen item based on the nearest neighbors behavior
- Essentials
  - many people must participate (increasing the likelihood that any one person will find other users with similar preferences)
  - easy way to represent user interests
  - matching algorithms for users with similar interests

Issues

- NEW USER PROBLEM - Accurate recommendations follow system preference learning from user ratings
- NEW ITEM PROBLEM (EARLY RATER) - Items would not be recommended until rated by a substantial number of users
- SPARSITY PROBLEM - The number of ratings obtained is usually very small compared to the number of predictions
- GREY SHEEP PROBLEM (UNUSUAL USER) - Individuals not consistently agree or disagree with any group of people
- SCALABILITY PROBLEM - A large amount of data from each user and a large number of users are require
- LACK OF TRANSPARENCY PROBLEM - *Black boxes* like oracles which give advice but cannot be questioned
  - no trust indicators
  - acceptance in low-risk content domains

Content-based Recommender Systems

- Recommendations based on contents describing the items
- Descriptions with attributes varying in number and type
  - same small number of attributes with known set of values
  - unstructured text
    - can be represent by the Vector Space Model
    - textual metadata represented by vectors in a *n*-dimensional space
    - each dimension corresponds to a term from the overall vocabulary
    - vector components are term weights that indicate the degree of association between the textual metadata and the term (dimension)

Recommendation approach

- User profiles based on some features of rated objects
- Item relevance by matching item representation against user profile
  - binary relevance judgment
  - continuous relevance judgment

    $\Rightarrow$ ranked list of potentially interesting items
- For instance, with the Vector Space Model, matching as cosine similarity of vectors
- Asking users for feedback on the recommended items

  $\Rightarrow$ matching performed according to the relevance feedback

Content-based Vs Collaborative Approaches

Advantages

- USER INDEPENDENCE - Solely ratings provided by the active user to build her own profile
- TRANSPARENCY - Explanations as list of content features or descriptions that caused the recommendation
- NEW ITEM - Recommendations rely on item descriptions even for items not yet rated

Content-based Vs Collaborative Approaches

Shortcomings

- LIMITED CONTENT ANALYSIS - Content-based techniques are limited by the features that are associated either automatically or manually with the items
  - Does content contain enough information to distinguish liked items from disliked items?
  - Does the representation capture all the aspects that influence the user experience?
- OVER-SPECIALIZATION - Recommendations of items (*too*) similar to those already (*highly*) rated
- NEW USER - Enough ratings have to be collected to understand user preferences and to provide accurate recommendations

Demographic Recommender Systems

- Recommendations based on demographic classes
- Categorization of users starting from personal attributes
- Data in user model can vary greatly
  - hand-crafted attributes with numeric confidence values [Grundy, 1994]
  - features from users' home pages [Pazzani, 1999]
- Benefit: the history of user ratings is not required
- Shortcoming: demographic data is difficult to collect

Knowledge-based Recommender Systems

- Recommendations based on a knowledge-based
  - about how a particular item meets a particular user need
  - reasoning about the relationship between *a need* and *a possible recommendation*
- Some system implements case-based reasoning
  - the recommender solves a new problem looking up a similar past solved one. Main steps:
    - [Retrieve] looks for a case similar to the new problem
    - [Reuse] reuses the retrieved solution
    - [Adaptation] makes some adaptation (if necessary)
    - [Retain] stores the new adapted case in the case-library
- User preference elicitation not required
  - recommendation algorithm $\sim$ retrieve the most similar case
- No ramp-up problem (“early rater” & “sparse ratings”)
  - since its recommendations do not depend on user ratings
- Approach is complementary to others

Hybrid Recommender Systems

- Combine two or more recommender algorithms in order to emphasize strengths and to level out weaknesses
- Hybridization methods
  - WEIGHTED - The recommending score is computed from the results of all of the available recommendation techniques, e.g., by linear combination
  - SWITCHING - Some criterion to switch between recommendation techniques
  - MIXED - Recommendations from several different recommenders are presented at the same time

Hybrid Recommender Systems

- Hybridization methods (cont.)
  - FEATURE COMBINATION - Features from different recommendation sources are thrown together into a single recommendation algorithm
    - e.g., collaborative information as additional feature of content-based data
  - CASCADE - one recommender refines the recommendations given by another one as a staged process
  - FEATURE AUGMENTATION - Output from one technique is used as an input feature to another
  - META-LEVEL - The model learned by one recommender is used as input to another

Learning User Preferences in Recommender Systems

- Preference: an ordering relation between two or more items to characterize which, among a set of possible choices, is the one that best fits user tastes [Brafman & Domshlak, 2009]
  - *Preferences* are something able to guide choices, discriminating items user likes from those she doesn't like (or she likes the least)
- Learning user preferences: a way to find the solution of a research (or optimization) problem whose space of possible solutions is represented by the set of the items the user can be recommended
  - The complexity is strictly related to the number of dimensions used to represent the set of possible choices
  - Gathering user feedbacks allows to generate user profiles catching information about user prefences

Feedback Gathering

- The Information Filtering and Information Retrieval systems rely on *relevance feedback* to capture an appropriate snapshot of user information needs
  - user is allowed to directly express her notion of relevance with respect to individual documents
- Relevance feedback employed in several classes of personalization systems
  - initially introduced to support basic query expansion
  - mean to elicit user information needs
  - based on an explicit or implicit feedback gathering
    - explicit: object ratings are provided explicitly by users
    - implicit: object relevance is inferred in a transparent fashion, by monitoring user interactions with the system

Explicit Ratings

- Explicit ratings is common in everyday
  - in free text form (e.g. book reviews)
  - on an agreed discrete scale (e.g. star ratings for restaurants, marks out of ten for films, etc.)
    - judgments easier to process statistically
- The evaluator has to examine an item and assign it a value on the rating scale
- Cognitive cost: the act of rating alters the user behavior from her normal interaction pattern
  - users tend to skip the rating task

Explicit Ratings

- Disregard user knowledge on the current topic
  - When users are unclear about their search interests
    - browse for more information to clarify their need
    - re-formulate their query accordingly
  - The uncertainty in search episodes increases the cognitive load during explicit relevance feedback
    - decide on the relevance of items with a lack of confidence
- Privacy issues
  - Users are not always comfortable in providing direct indications of their interests
  - Obtrusive nature of explicit ratings $\Rightarrow$ not many users are willing to provide them $\Rightarrow$ dearth of ratings $\Rightarrow$ the performance of profile capturing and recommendation algorithms degrades
    - the sparsity of judgments can often render collaborative filtering recommender systems unusable

Explicit Ratings by Critiquing Examples

- Users are required to critique possible solutions
  - Planing travel arrangements: “the arrival time of this flight leg is too late.”
- Cyclical interaction
  - the system provides example solutions
  - the user examines them and may state critiques
  - a critique becomes an additional preference in the model
  - the system refines the solution set
- Motivation: people usually cannot state preferences in advance; construct them according to the available options
  - The critiques come in response to shown examples $\Rightarrow$ the current solutions can hinder the user from refocusing the search in another direction (the anchoring effect)
  - A complete preference model can be acquired only if the system is able to stimulate the user

Implicit Ratings

- Proposed as unobtrusive alternative or supplement to explicit ratings in order to state (indirect) assessment about usefulness of any individual item
- Passive monitoring of user interactions with the system
  - Click-throughs, time spent viewing a document, mouse gestures, ...
- Removing the cognitive cost of explicit relevance feedback
- Implicit judgments are often considered less indicative than explicit ratings

Modeling User Preferences

- Feedback gathering techniques allow to collect information about user tastes and interests
- Collected information had to be modeled following a specific representation
  - structured data (e.g. generic ratings or some well-defined attribute-value pairs) can be represented through a matrix
  - unstructured data (usually exploited by content-based recommenders) had to be processed through some Information Retrieval-related techniques (e.g. stemming, lemmatization, indexing) to shift from a textual source to a structured one
  - more complex representations (semantic or neural networks, probabilistic models, etc.)

Techniques for Learning User Profiles

- Recommendations techniques can be grouped into two general classes [Adomavicius & Tuzhilin, 2005]: model-based and memory/heuristic based
- Same classification for for learning user profiles
  - offline learning techniques: used in model-based recommenders in domains where user preferences change slowly with respect to the time needed to build the model
  - online learning techniques: used in memory-based recommenders to build and update the model in order to make recommendations in real-time
- Machine Learning techniques allow to fulfil the task of learning user profiles in model-based recommenders

Machine Learning for Learning User Profiles

- Common approach: build a classifier, i.e. a model able to assign a category to a specific input
  - Learning user profiles problem becomes a binary categorization task: each item has to be classified as interesting or not with respect to the user preferences
- The classifier is learned by an inductive process from a training set, i.e. a collection of items labeled with the categories they belong to
- Classifiers implemented by machine learning strategies (e.g. probabilistic approaches, neural networks, decision trees, association rules and Bayesian networks)

Naïve Bayes

- Generate a probabilistic model based on previously observed data
- Used in content-based recommenders
- The model estimates the *a posteriori* probability, $P(c|d)$, of document d belonging to class c

  $P(c)$ the prob. of observing a document in class c
  $P(d|c)$ the prob. of observing the document d given c
  $P(d)$ the prob. of observing the instance d

  Bayes theorem $P(c|d) = \frac{P(c)P(d|c)}{P(d)}$
- To classify the document d, the class with the highest probability is chosen: $c = \text{argmax}_{c_j} \frac{P(c_j)P(d|c_j)}{P(d)}$
  - $P(d)$ is generally removed as it is equal for all c_j

Naïve Bayes

- $P(d|c)$ in $P(c|d) = \frac{P(c)P(d|c)}{P(d)}$ estimated by the training data
  - Estimating $P(d|c)$ in this way is problematic:
    it is very unlikely to see the same document more than once; the observed data is generally not enough to be able to generate good probabilities
  - Simplify the model by the independence assumption:
    all the words or tokens in the observed document d are conditionally independent of each other given the class
- Although naïve Bayes performances are not as good as some other statistical learning methods (e.g. nearest-neighbor classifiers or support vector machines), it has been shown that it can perform well in the classification tasks where the computed probability is not important
- Advantage: efficient and easy to implement

Rocchio's Method

- Some linear classifiers consist of an explicit profile (or prototypical document) of the category
- The Rocchio's method is used for inducing linear, profile-style classifiers
- Adaptation to text categorization of the Rocchio's formula for relevance feedback in the Vector Space Model
  - documents as vectors $\Rightarrow$ documents with similar content have similar vectors
- Learning is achieved by combining document vectors (of positive and negative examples) into a prototype vector for each class in the set of classes C
- Classify a new document d: assign d to the class with the most similar prototype vector
  - for example by using the cosine similarity measure

Rocchio's method

- Classifier for category c_i: $\overrightarrow{c_i} = \langle \omega_{1i}, \ldots, \omega_{|T|i} \rangle$
  - T is the *vocabulary*, i.e. the set of terms in the training set
  - $$\omega_{ki} = \beta \cdot \sum_{\{d_j \in POS_i\}} \frac{\omega_{kj}}{|POS_i|} - \gamma \cdot \sum_{\{d_j \in NEG_i\}} \frac{\omega_{kj}}{|NEG_i|}$$
  - ω_{kj} is the TF-IDF weight of the term t_k in document d_j
  - POS_i and NEG_i are the set of positive and negative examples in the training set for the specific class c_j
  - β and γ are control parameters that allow setting the relative importance of *all* positive and negative examples
- To assign a class $\widetilde{c}$ to a document d_j, the similarity between each prototype vector $\overrightarrow{c_i}$ and the document vector $\overrightarrow{d_j}$ is computed and $\widetilde{c}$ will be the c_i with the highest value of similarity

Decision Trees Learners

- Decision trees: trees in which
  - internal nodes are labeled by terms
  - branches departing from them are labeled by tests on the weight that the term has in the test document
  - leafs are labeled by categories
- Decision trees are built by recursively partitioning training data into subgroups
  - until subgroups contain only instances of a single class
- The test for partitioning data is run on the weights that the terms labeling the internal nodes have in the document
- The choice of the term on which to operate the partition is generally made according to an information gain or entropy criterion

Decision Rule Classifiers

- Similar to decision trees
  - operate a recursive data partitioning
- Tend to generate more compact classifiers
- Attempt to select from all the possible covering rules (i.e. rules that correctly classify all the training examples) the “best” one according to some minimality criterion.
- Examples of inductive learning techniques
  - Ripper [Cohen, 1995]
  - Slipper [Cohen & Singer, 1999]
  - CN2 [Clark & Niblett, 1989]
  - C4.5rules [Quinlan, 1994]

Neural Network

- Neural networks model complex relationships between input and output cells
- The user interests are represented by the output cells and each of them are achievable by a specific pattern in the network
- When an error occurs, there is a backward propagation until the responsible cell is achieved, so the cell parameters are adjusted

Bayesian Network

- It represents a probability distribution by a direct acyclic graph
  - nodes: random variables; represent attributes
  - arcs: relations among random variables; represent probability correlations
- Employed to model quantitative and qualitative relationships between items that users have liked
- Generally used in those situations where user interests change slowly

Nearest Neighbor Algorithms

- Store training data in memory and classify a new unseen item by comparing it to all stored items by using a similarity function
  - for example the cosine similarity measure is adopted when items are represented using the Vector Space Model
- The "nearest neighbor" or the "k-nearest neighbors" items are determined, and the class label for the unclassified item is derived from the class labels of the nearest neighbors
- Nearest neighbor algorithms are quite effective, although the most important drawback is their inefficiency at classification time, since they do not have a true training phase

Conclusions

- Survey of the methods for learning user profiles in recommender systems
  - Introduction to the basics of recommender systems
    - describing the main approaches presented in literature, namely the content-based and the collaborative one. We also introduced other important approaches, such as the demographic and the knowledge-based one, and some hybrid systems combining different types of recommendation strategies
  - Focus on the process of learning user preferences in recommender systems
    - techniques to get implicit or explicit user feedback
    - the most successful and widely used machine learning methods to learn user profiles in recommender systems