Indian Institute of Information Technology, Allahabad

Project Report

Implementing and exploring Hybrid Recommender System on Linked Open Data Case Study: Music Recommender System

Under the guidance of Prof. O.P. Vyas

Group Members:

Vijendra Singh
IIT2011151

Vignan Lavu
IIT2011119

Yash Vardhan SIngh
IIT2011124

Semester- VIth

Certificate

I do hereby recommend that the project work entitled "Implementing and exploring Hybrid Recommender System on Linked Open Data Case Study: Music Recommender System" is being submitted in fulfilment of Mini Project for the academic session Jan 2014 - May 2014. The declaration made by the candidates is true to the best of my knowledge.

Place:
IIIT Allahabad
Jhalwa Campus
Date: 11th March, 2014

Project Mentor
Prof. O.P. Vyas

Declaration

We declare that this written submission represents our ideas and reference has been quoted where others' ideas or words have been included. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Place:
IIIT Allahabad
Jhalwa Campus
Date: 10th March, 2014

Vijendra Singh
IIT2011151

Vignan Lavu
IIT2011119

Yash Vardhan Singh
IIT2011124

Acknowledgements

We would like to convey our deepest gratitude to Prof. O.P. Vyas, who guided us through this project. His keen awe-inspiring personality, superb guidance, and constant encouragement is the motivating force behind this project work. Also, we would like to thank Ms Nidhi Kushwaha for her enormous help and support throughout the project duration.

Table of Contents

1. Introduction

1.1 Semantic Web

The Semantic Web[1] is a collaborative movement led by the World Wide Web Consortium (W3C) that promotes common formats for data on the World Wide Web. By encouraging the inclusion of semantic content in web pages, the Semantic Web aims at converting the current web of unstructured documents into a "web of data". It builds on the W3C's Resource Description Framework (RDF).

According to the W3C, "The Semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries."

The term was coined by Tim Berners-Lee[2], the inventor of the World Wide Web and director of the World Wide Web Consortium ("W3C"), which oversees the development of proposed Semantic Web standards. He defines the Semantic Web as "a web of data that can be processed directly and indirectly by machines."

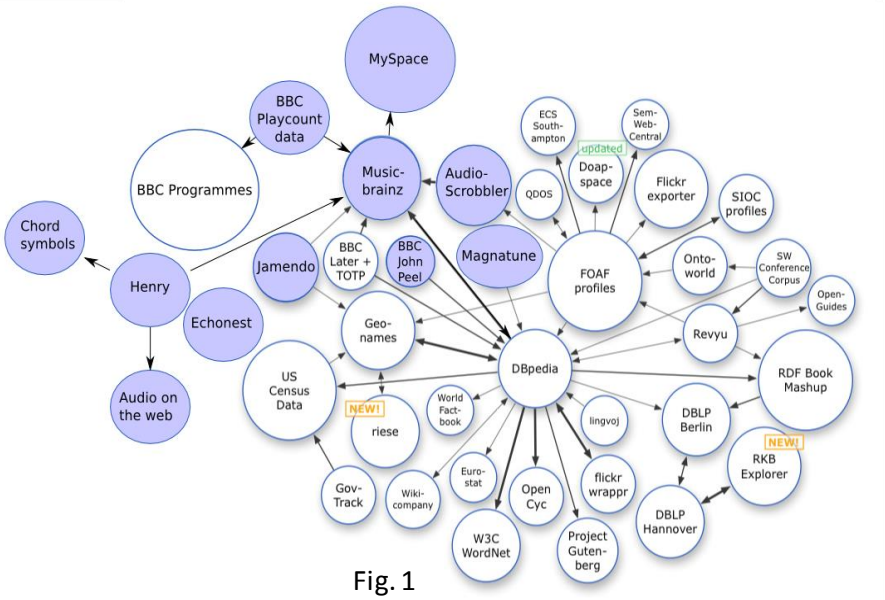

Fig. 1

The main purpose of the Semantic Web is driving the evolution of the current Web by enabling users to find, share, and combine information more easily.

Humans are capable of using the Web to carry out tasks such as finding the Irish word for "folder", reserving a library book, and searching for the lowest price for a DVD. However, machines cannot accomplish all of these tasks without human direction, because web pages are designed to be read by people, not machines. The semantic web is a vision of information that can be readily interpreted by machines, so machines can perform more of the tedious work involved in finding, combining, and acting upon information on the web.

The Semantic Web, as originally envisioned, is a system that enables machines to "understand" and respond to complex human requests based on their meaning. Such an "understanding" requires that the relevant information sources is semantically structured, a challenging task.

1.2
Linked Data

In computing, linked data[3] describes a method of publishing structured data so that it can be interlinked and become more useful. It builds upon standard Web technologies such as HTTP and URIs, but rather than using them to serve web pages for human readers, it extends them to share information in a way that can be read automatically by computers. This enables data from different sources to be connected and queried.

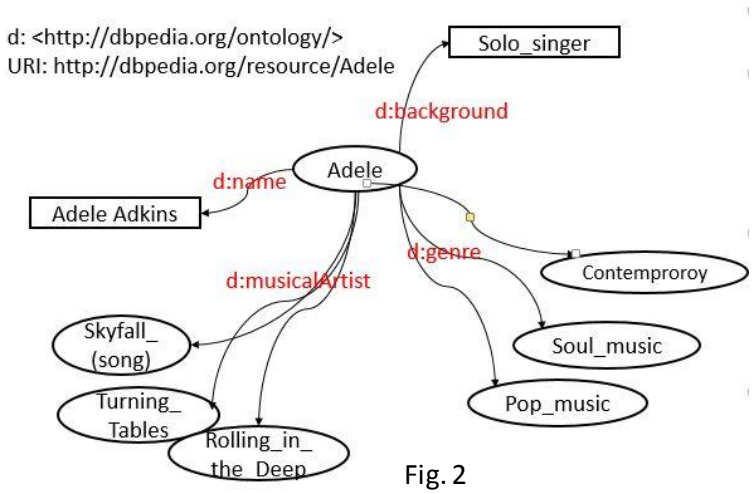

Fig. 2

Linked Data is about using the Web to connect related data that wasn't previously linked, or using the Web to lower the barriers to linking data

currently linked using other methods. More specifically,Wikipedia defines Linked Data as "a term used to describe a recommended best practice for exposing, sharing, and connecting pieces of data, information, and knowledge on the Semantic Web using URIs[4] and RDF[5]."

Tim Berners-Lee outlined four principles of linked data in his Design Issues: Linked Data note, paraphrased along the following lines:

1. Use URIs to identify things.

2. Use HTTP URIs so that these things can be referred to and looked up ("dereferenced") by people and user agents.

3. Provide useful information about the thing when it's URI is dereferenced, using standard formats such as RDF/XML.

4. Include links to other, related URIs in the exposed data to improve discovery of other related information on the Web.

1.3 Resource Description Framework

The Resource Description Framework (RDF) is a family of World Wide Web Consortium (W3C) specifications originally designed as a metadata data model. It has come to be used as a general method for conceptual description or modelling of information that is implemented in web resources, using a variety of syntax formats.

The RDF data model is similar to classic conceptual modelling approaches such as entity-relationship or class diagrams, as it is based upon the idea of making statements about resources (in particular Web resources) in the form of subject-predicate-object expressions. These expressions are known as *triples* in RDF terminology.

Triple: The subject denotes the resource, and the predicate denotes traits or aspects of the resource and expresses a relationship between the subject and the object.

For example, one way to represent the notion "The sky has the color blue" in RDF is as the triple: a subject denoting "the sky", a predicate denoting "has the color", and an object denoting "blue". RDF is an abstract model with several serialization formats (i.e., file formats), and so the particular way in which a resource or triple is encoded varies from format to format.

RDF's simple data model and ability to model disparate, abstract concepts has also led to its increasing use in knowledge management applications unrelated to Semantic Web activity.

A collection of RDF statements intrinsically represents a labelled, directed multi-graph. As such, an RDF-based data model is more naturally suited to certain kinds of knowledge representation than the relational model and other ontological models. However, in practice, RDF data is often persisted in relational database or native representations also called Triple stores, or Quad stores if context (i.e. the named graph) is also persisted for each RDF triple. As RDFS and OWL demonstrate, additional ontology languages can be built upon RDF.

The subject of an RDF statement is either a Uniform Resource Identifier (URI) or a blank node, both of which denote resources. Resources indicated by blank nodes are called anonymous resources.

They are not directly identifiable from the RDF statement. The predicate is a URI which also indicates a resource, representing a relationship. The object is a URI, blank node or a Unicode string literal.

The following example shows how such simple claims can be elaborated on, by combining multiple RDF vocabularies. Here, we note that the primary topic

of the Wikipedia page is a "Person" whose name is "Tony Benn":

```
<rdf:RDF
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:foaf="http://xmlns.com/foaf/0.1/"
 xmlns:dc="http://purl.org/dc/elements/1.1/">
     <rdf:Description rdf:about="http://en.wikipedia.org/wiki/Tony_Benn">
          <dc:title>Tony Benn</dc:title>
          <dc:publisher>Wikipedia</dc:publisher>
          <foaf:primaryTopic>
            <foaf:Person>
               <foaf:name>Tony  Benn</foaf:name>
            </foaf:Person>
          </foaf:primaryTopic>
     </rdf:Description>
</rdf:RDF
>
```

Triples Formed:

Number	Subject	Predicate	Object
1	http://en.wikipedia.org/wiki/Tony_Benn	http://purl.org/dc/elements/1.1/title	"Tony Benn"
2	http://en.wikipedia.org/wiki/Tony_Benn	http://purl.org/dc/elements/1.1/publisher	"Wikipedia"
3	genid:A178435	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://xmlns.com/foaf/0.1/Person
4	http://en.wikipedia.org/wiki/Tony_Benn	http://xmlns.com/foaf/0.1/primaryTopic	genid:A178435
5	genid:A178435	http://xmlns.com/foaf/0.1/name	"Tony Benn"

Fig. 3

1.4

Ontology

Ontologies are considered one of the pillars of the Semantic Web, although they do not have a universally accepted definition. A (Semantic Web) vocabulary can be considered as a special form of

dbpedia-owl:birthDate	1988-05-05 (xsd:date)
dbpedia-owl:birthPlace	dbpedia:Tottenham dbpedia:North_London
dbpedia-owl:genre	dbpedia:Contemporary_R&B dbpedia:Pop_music dbpedia:Soul_music
dbpedia-owl:occupation	dbpedia:Singer-songwriter
dbpedia-owl:recordLabel	dbpedia:Columbia_Records dbpedia:XL_Recordings
dbpedia-owl:thumbnail	http://upload.wikimedia.org/wikipedia/commons/thumb/9/97/AdeleLo
dbpedia-owl:wikiPageExternalLink	http://adele.tv/
dbpedia-owl:wikiPageID	13041163 (xsd:integer)
dbpedia-owl:wikiPageRevisionID	547814786 (xsd:integer)
dbpprop:after	dbpedia:Zac_Brown_Band dbpedia:Little_Boots incumbent
dbpprop:align	left
dbpprop:alternativeNames	Adele
dbpprop:background	solo_singer
dbpprop:before	dbpedia:Mika_(singer) dbpedia:Amy_Winehouse Jack White and Alicia Keys Another Way to Die, 2008

Fig. 4

(usually light-weight) ontology, or sometimes also merely as a collection of URIs with an (usually informally) described meaning. Ontology is a formal specification of a shared conceptualization.

SPARQL

SPARQL[5] (a recursive acronym for SPARQL Protocol and RDF Query Language) is an RDF query language, that is, a query language for databases, able to retrieve and manipulate data stored in Resource Description Framework format. It was made a standard by the *RDF Data Access Working Group* (DAWG) of the World Wide Web Consortium, and considered as one of the key technologies of semantic web. On 15 January 2008, SPARQL 1.0 became an official W3C Recommendation.

SPARQL allows for a query to consist of triple patterns, conjunctions , disjunctions, and optional patterns. SPARQL allows users to write unambiguous queries. For example, the following query returns names and emails of every person in the dataset:

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name ?email
WHERE {
  ?person a foaf:Person.
  ?person foaf:name ?name.
  ?person foaf:mbox ?email.
}
```

Fig. 5

The SPARQL language specifies four different query variations for different purposes.

SELECT query:
Used to extract raw values from a SPARQL endpoint, the results are returned in a table format.

CONSTRUCT query:
Used to extract information from the SPARQL endpoint and transform the results into valid RDF.

ASK query:
Used to provide a simple True/False result for a query on a SPARQL endpoint.

DESCRIBE query:
Used to extract an RDF graph from the SPARQL endpoint, the contents of which is left to the endpoint to decide based on what the maintainer deems as useful information.

Each of these query forms takes a WHERE block to restrict the query although in the case of the DESCRIBE query the WHERE is optional. To summarize things:

- The SPARQL language is used for constructing queries that extract data from RDF specifications.

- SPARQL is not based on XML. It is based on a roughly SQL-like syntax, and represents RDF graphs as triples.
- The building blocks of a SPARQL queries are graph patterns that include variables. The result of the query will be the values that these variables must take to match the RDF graph.
- A SPARQL query can return results in several different ways, as determined by the query.
- SPARQL queries can be used for OWL querying.

SPARQL End Point:

A sparql end point is a place where you can query the remote linked data sets from your PC by using SPARQL language for querying.

For a bunch of sparql endpoints, see this { http://esw.w3.org/SparqlEndpoints}

SPARQL:

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX : <http://dbpedia.org/resource/>
PREFIX dbpedia2: <http://dbpedia.org/property/>
PREFIX dbpedia: <http://dbpedia.org/>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
PREFIX d: <http://dbpedia.org/ontology/>

select ?o ?song
  where {
      ?s d:musicalArtist ?o
       FILTER (regex(?s, "Skyfall","i"))
       ?song  d:musicalArtist ?o
}
```

Results: Browse Go! Reset

Fig. 6

SPARQL results:

o	song
:Adele	:Chasing_Pavements
:Adele	:Rolling_in_the_Deep
:Adele	:Hometown_Glory
:Adele	:Make_You_Feel_My_Love
:Adele	:Rumour_Has_It_(Adele_song)
:Adele	:Someone_like_You_(Adele_song)
:Adele	:Turning_Tables
:Adele	:Cold_Shoulder_(song)
:Adele	:Set_Fire_to_the_Rain
:Adele	:Skyfall_(song)

Fig. 7

1.5
Recommender system

Recommender system[6] is a personalized information filtering technology used to either predict whether a particular user will like a particular item (prediction problem) or to identify a set of N items that will be of interest to a certain user. Recommender systems form or work from a specific type of information filtering system technique that attempts to recommend information items (movies, TV program/show/episode, video on demand, music, books, news, images, web pages, scientific literature etc.) or social elements (e.g. people, events or groups) that are likely to be of interest to the user.

Typically, a recommender system compares a user profile to some reference characteristics, and seeks to predict the 'rating' or preference' that a user would give to an item they had not yet considered. These characteristics may be from the information item (the content-based approach) or the user's social environment (the collaborative filtering). The recommender system apply data mining techniques and prediction algorithms to predict user's interest on information, product and services user.

Recommender systems apply techniques and methodologies from another neighbouring areas-such as Human computer interaction (HCI) or Information Retrieval (IR). However, most of these systems bear in their core an algorithm that can be understand as a particular instance of a data mining (DM) technique. The process of data mining consists of 3 steps, carried out in succession:

- Data Pre-processing
- Data Analysis
- Result Interpretation.

Examples of recommender system are amazon.com, Reel.com, CDNOW, eBay, Levis, Moviefinder.com.

1.6 Types of Recommender systems

1.6.1 Content based recommender system

These are the recommender system which work with profiles of users that are created at the beginning. A profile has information about a user and his taste. Taste is based on how user rated items. In the recommendation process, the engine compares the items that were already positively rated by the user with the items he didn't rate and looks for similarities. Those items that are mostly similar to the positively rated ones, will be recommended to the user.

Content based recommendation systems recommend an item to a user based upon a description of the item and a profile of the user's interests. Such systems are used in recommending web pages, TV programs and news articles etc. All content based recommender systems has few things in common like means for description of items, user profiles and techniques to compare profile to items to identify what is the most suitable recommendation for a particular user. In content-based recommendation methods, the utility u(c, s) of item s for user c is estimated based on the utilities $u(c, s_i)$ assigned by user c to items $s_i \in S$ that are "similar" to item s. Content-Based

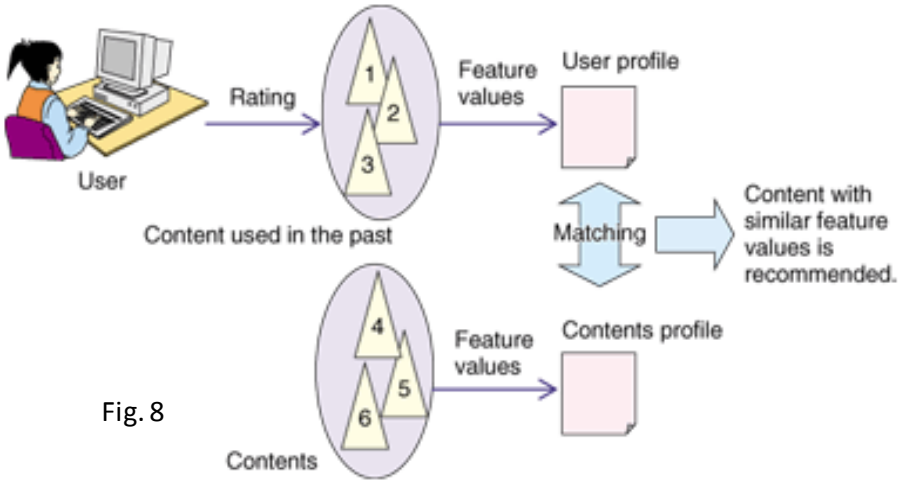

Fig. 8

recommender systems make suggestions upon item features and user interest profiles. Typically, personalized profiles are created automatically through user feedback, and describe the type of items a person likes. In

order to determine what items to recommend, collected user information is compared against content features of the items to examine. As shown below

• System has a huge database consisting of the items to be recommended and the features of these items and it is termed as Item Profile.
• The users provide some sort of information about their preferences to the system. Combining the item information with the user preferences, the system builds a profile of the users.
• According to the information existing in a target user's profile, the system recommends suitable items to the user.

One can make better personalized recommendation by utilizing the features of items and users. An item profile is defined by its important features. For instance, a movie can be described by its title, genre, language, country, actors etc. Depending upon the weighing procedure, similarity between two items can be calculated. Depending on the domain, features can be represented either by Boolean values or by a set of restricted values.

For example, imagine we want to analyse a set of newspaper articles about different kind of topics. While a Boolean value could indicate whether a word is contained in an article or not, an integer value could express the number of times a word appears.

To build a user profile, information of a user can be used. In MovieLens dataset, users are described using demographic information that includes age, gender, occupation and zip code. User information might be provided explicitly by the individual person or gathered implicitly by a software agent.

• Explicit user information collection basically relies on personal input by the user. A common feedback technique is the one that allows users to express their opinions by selecting a value of range.

However, filling out forms or clicking checkboxes places a burden on the user. Profiles might be imprecise, because the user is not willing to spend a lot of time providing personal information or the user information is already outdated.

• Implicit feedback does not require any additional intervention by the user during the process of constructing profiles. Moreover, it automatically updates as the user interacts with the system. Thus, systems that collect implicit feedback are more likely to be used in practice

1.6.2 Collaborative recommender system

The Collaborative filtering (CF) systems work by collecting user feedback in the form of ratings for items in a given domain and exploiting similarities in rating behaviour amongst several users in determining how to recommend an item. CF systems recommend an item to a user based on opinions of other users. For example, in a movie recommendation application, CF system tries to find other like-minded users and then recommends the movies that are most liked by them. The task of traditional collaborative filtering recommender algorithm concerns the prediction of the target user's rating for the target item that the user has not given the rating, based on the users' ratings on observed items.

• CF algorithms represent the entire user-item space as a rating matrix 'R'. Each entry R_{ij} in matrix represents the preference score (rating) if the i^{th} user on the j^{th} item. Each individual rating is within a numerical scale and it can be 0 as well, indicating that user has not yet rated this item.

• CF problem includes the estimation or prediction of rating for the yet unrated item. For the prediction of rating, similarities between items and users are calculated using different approaches. Thus, the two related problems consist in finding set of K users that are most similar to a given user and finding set of K items that are most similar to a given item.

• Finally using these similarities, recommendations that are produced at output interface can be of two types: Prediction and Recommendation.

• Prediction is a numerical value, R_{ij}, expressing the predicted score of item j for the user i. The predicted value is within the same scale that is used by all users for rating.

a) User-based approach: This approach was proposed in the end of 1990s by the professor of University of Minnesota Jonathan L.Herlocker. In the user-based approach, the users perform the main role. If certain majority of the customers has the same taste then they join into one group. Recommendations are given to user based on evaluation of items by other users form the same group, with whom he/she shares common preferences. If the item was positively rated by the community, it will be recommended to the user.

b) Item-based approach: This approach was proposed by the researchers of University of Minnesota in 2001. Referring to the fact that the taste of users remains constant or change very slightly similar items build neighbourhoods based on appreciations of users. Afterwards the system generates recommendations with items in the neighbourhood that a user would prefer.

1.6.3 Hybrid recommendation approaches

Hybrid recommenders are systems that combine multiple recommendations techniques together to achieve a synergy between them. Several researchers have attempted to combine collaborative filtering and content based approaches in order to smoothen their disadvantages and gain better performance while recommendations. Depending on domain and data characteristics, several hybridization techniques are possible to combine CF and CB techniques which may generate different outputs. Different ways of hybridization are:

• Implementing CF and CB separately and combine their predictions.
• Incorporating some content based characteristics into collaborative approach.
• Incorporating some collaborative characteristics into content based approach.
• Constructing a general unifying model that incorporates both content-based and collaborative characteristics.

Many hybrid approaches are based on CF but CB methods are used to maintain the user profiles and such profiles are used to find similar users.

1.7 Various Approaches in Recommendation

1.7.1 Memory-based

This mechanism uses user rating data to compute similarity between users or items. This is used for making recommendations. This was the earlier mechanism and is used in many commercial systems. It is easy to implement and is effective.

Typical examples of this mechanism are neighbourhood based CF and item-based/user-based top-N recommendations The neighbourhood-based algorithm calculates the similarity between two users or items, produces a

prediction for the user taking the weighted average of all the ratings.

Similarity computation between items or users is an important part of this approach. Multiple mechanisms such as Pearson correlation and vector cosine based similarity are used for this.

The user based top-N recommendation algorithm identifies the k most similar users to an active user using similarity based vector model. After the k most similar users are found, their corresponding user-item matrices are aggregated to identify the set of items to be recommended. A popular method to find the similar users is the Locality-sensitive hashing, which implements the nearest neighbour mechanism in linear time.

The advantages with this approach include: the explain ability of the results, which is an important aspect of recommendation systems; it is easy to create and use; new data can be added easily and incrementally; it need not consider the content of the items being recommended; and the mechanism scales well with co-rated items.

There are several disadvantages with this approach. First, it depends on human ratings. Second, its performance decreases when data gets sparse, which is frequent with web related items. This prevents the scalability of this approach and has problems with large datasets. Although it can efficiently handle new users because it relies on a data structure, adding new items becomes more complicated since that representation usually relies on a specific vector space. That would require to include the new item and re-insert all the elements in the structure.

1.7.2 Model-based

Models are developed using data mining, machine learning algorithms to find patterns based on training data. These are used to make predictions for real data. There are many model-based CF algorithms. These include Bayesian networks, clustering models, latent semantic models such as singular value decomposition, probabilistic latent semantic analysis, Multiple Multiplicative Factor, Latent Dirichlet allocation and markov decision process based models.

This approach has a more holistic goal to uncover latent factors that explain

observed ratings. Most of the models are based on creating a classification or clustering technique to identify the user based on the test set. The number of the parameters can be reduced based on types of principal component analysis. There are several advantages with this paradigm. It handles the sparsity better

than memory based ones. This helps with scalability with large data sets. It improves the prediction performance. It gives an intuitive rationale for the recommendations.

The disadvantages with this approach are in the expensive model building. One needs to have a trade-off between prediction performance and scalability. One can lose useful information due to reduction models. A number of models have difficulty explaining the predictions.

1.8 Imputation

In statistics, imputation is the process of replacing missing data with substituted values. When substituting for a data point, it is known as "unit imputation"; when substituting for a component of a data point, it is known as "item imputation".

Because missing data can create problems for analysing data, imputation is seen as a way to avoid pitfalls involved with list wise deletion of cases that have missing values. That is to say, when one or more values are missing for a case, most statistical packages default to discarding any case that has a missing value, which may introduce bias or affect the representativeness of the results. Imputation preserves all cases by replacing missing data with a probable value based on other available information.

Once all missing values have been imputed, the data set can then be analysed using standard techniques for complete data.

We changed the problem of predicting the item ratings to classification problem where class labels refer to the types of rating that are possible. To solve this, we can use machine learning algorithms like SVM, SVD etc.

1.9 Support vector machines

Support vector machines (SVMs) are supervised learning models with associated learning algorithms that analyze data and recognize patterns, used for classification and regression analysis. Given a set of training examples, each

marked as belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples into one category or the other, making it a non-probabilistic binary linear classifier. An SVM model is a representation of the points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on.
In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.

Bagging

Bootstrap aggregating (bagging) is a machine learning ensemble meta-algorithm designed to improve the stability and accuracy of machine learning algorithms used in statistical classification and regression. It also reduces variance and helps to avoid overfitting. Although it is usually applied to decision tree methods, it can be used with any type of method. Bagging is a special case of the model averaging approach.

Problems in Recommender Systems

1. Sparsity Problem

Sparsity problem is one of the major problems encountered by recommender system and data sparsity has great influence on the quality of recommendation. Generally, data of system like MovieLens is represented in form of user-item matrix populated by ratings given to movies and as no. of users and items increases the matrix dimensions and sparsity evolves. The main reason behind data sparsity is that most users do not rate most of the items and the available ratings are usually sparse. Collaborative filtering suffers from this problem because it is dependent over the rating matrix in most cases. Many researchers have attempted to reduce this problem; still this area demands more research.

2. Cold Start problem

Cold start problem refers to the situation when a new user or item just enters the system. Three kinds of cold start problems are: new user problem, new item problem and new system problem. In such cases, it is really very difficult to provide recommendation

as in case of new user, there is very less information about user that is available and also for a new item, no ratings are usually available and thus collaborative filtering cannot make useful recommendations in case of new item as well as new user.

However, content based methods can provide recommendation in case of new item as they do not depends on any previous rating information of other users to recommend the item.

3. Scalability

Scalability is the property of system indicates its ability to handle growing amount of information in a graceful manner. With enormous growth in information over internet, it is obvious that the recommender systems are having an explosion of data and thus it is a great challenge to handle with continuously growing demand. Some of the recommender system algorithms deal with the computations which increase with growing number of users and items. In CF computations grow exponentially and get expensive, sometimes leading to inaccurate results. Methods proposed for handling this scalability problem and speeding up recommendation formulation are based on approximation mechanisms. Even if they improve performance, most of the time they result in accuracy reduction

2.Problem Definition and Objective

Objective

Given a dataset containing information about

- Items liked by user
- Features of each item
- Friends of each user

Recommend new item that a user will like.

Problem Definition

Content based and collaborative recommender system have various disadvantages associated with them like scalability, cold start, sparsity etc.

Hybrid recommender systems give us the best of both approaches. They make recommendations by comparing the usage habits of similar users (i.e. collaborative filtering) as well as by offering items that share characteristics with items that a user has rated highly (content-based filtering).

Our Lastfm dataset includes:

artists.dat
This file contains information about music artists listened and tagged by the users.

tags.dat
This file contains the set of tags available in the dataset.

user_artists.dat
This file contains the artists listened by each user. It also provides a listening count for each [user, artist] pair.

user_taggedartists.dat - user_taggedartists-timestamps.dat
These files contain the tag assignments of artists provided by each particular user. They also contain the timestamps when the tag assignments were done.

user_friends.dat
These files contain the friend relations between users in the database.

Item (I) = Artist
Feature (F) = Attributes of Artist
User (U) = Users

Problem: I x F data set of Lastfm is not rich
Solution: Obtain more features by using the extensive Dbpedia Linked open data cloud

Problem: U x I matrix is sparse.
Solution: Perform Imputation by applying SVM

Problem: Simple Top-N recommendation looks only in a user's neighbourhood
Solution: We also consider features of Item along with nearest neighbours of User. Reduces problem of scalability.

3. Literature Survey

S. No.	Publisher Name & Year	Paper Title	Objective	Technique/ Algorithm	Merit/ Advantages	Demerit/ Disadvantages
1	Panagiotis Symeonidis, Alexandros Nanopoulos, and Yannis Manolopoulos, 2007	Feature-weighted User Model for Recommender Systems[7]	To construct a feature-weighted user profile to disclose the duality between users and features. The approach correlates user ratings with item features bringing to surface the actual reasons of user preferences.	(i) Construct a novel feature weighted user model, which discloses the duality between users and features (ii) Include the Term Frequency Inverse Document Frequency (TFIDF) weighting scheme in CF (iii) Propose a new top-N generation list algorithm based on features frequency	This is a memory based approach. Explainabilty of results. Easy to create and use. New data can be added easily and incrementally.	Problems of Sparsity and Scalabiltiy exists. Time complexity will be more than model based approach. It depends on human ratings.
2	Kamakshi Lakshminarayan, Steven A. Harp, Robert Goldman and Tariq Samad 1996.	Imputation of missing data using machine learning techniques.[9]	This paper explores the use of machine - learning based alternatives to standard statistical data completion (data imputation) methods, for dealing with missing data.	(i) The first is an unsupervised dustering strategy which uses a Bayesian approach to cluster the data into classes. The dasses so obtained are then used to predict multiple choices for the attribute of interest. (ii) The second technique involves modelling missing variables by supervised induction of a decision tree-based classifier. This predicts the most likely value for the attribute of interest.		
3)	Zhonghang Xia, Yulin Dong. Guangming Xing.	Support Vector Machines For Collaborative Filtering[8]	In this paper, a heuristic method, is proposed to improve the predictive accuracy of SVMs by repeatedly correcting the missing values in the user-item matrix.	In this section, a SSVM-based heuristic (SSVMBH) Method to overcome these problems by iteratively estimating	This is a model based approach of collaborative filtering. Helps scalabilty	Expensive model building. There can be loss of data due to reduction models.

				missing elements in the user-item matrix A. For each element a_{mn} belongs to A we assign x_{mn} if n belongs to I_m otherwise assign p_{mn}. Initially, randomly assign 0 or 1 to p_{mn}. Then, for each user u_m and item n where $n I_m \notin$, a linear classifier f_{mn} is trained by a SVM algorithm according to feature vector. According to F_{mn}, a new p_{mn} is given. After each p_{mn} is re-computed, we test the current classifiers with the test data, denoted by T. Let $\|T_c\|$ be the total number of correct labels computed with current classifiers. The accurate rate is defined as $\|Tc\|/\|T\|$ If the difference of accurate rates between two consecutive steps is less than a predefined value ε, the algorithm stops. Otherwise, this procedure is repeated. The details of the algorithm are given as follow	with larger datasets. Improves the prediction performances.	

4. Proposed approach

4.1 Algorithm

a) Obtain item-feature matrix (I x F) from DBpedia and, create U x I (user-item) and U x F (user-feature) matrix from the Lastfm data.

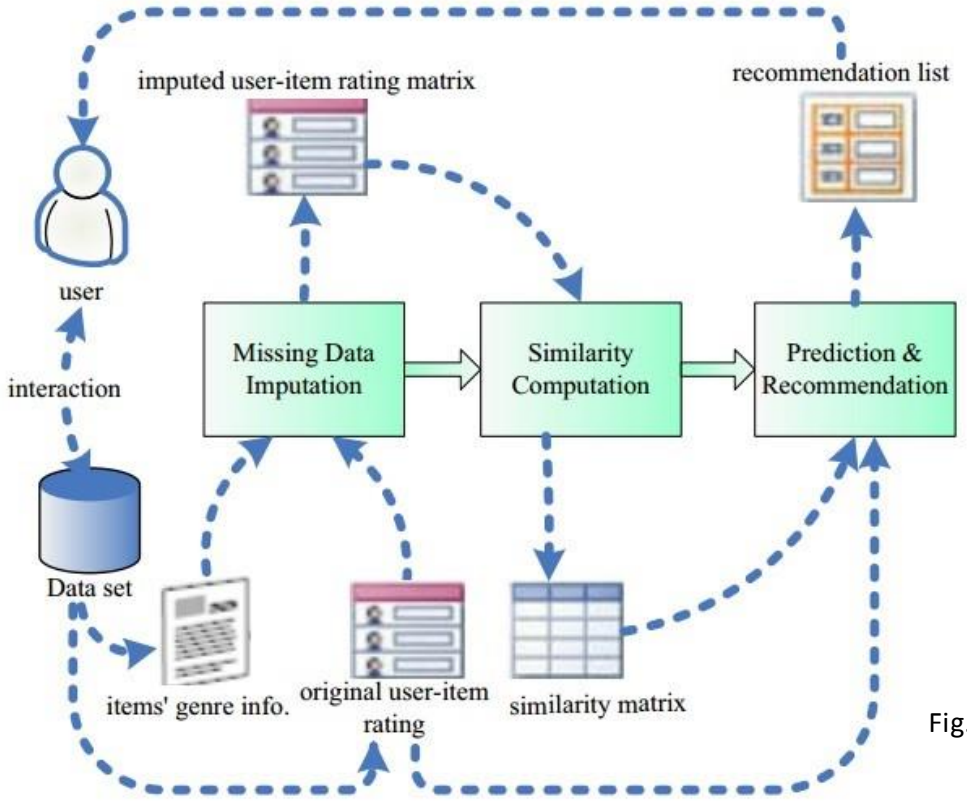

Fig. 9

b) Utilize the above matrices to calculate the imputed value of missing data in (U x I), forming a modified full rating matrix $(U \times I)_{modified}$.

c) Find the weighted User-Feature matrix $(U \times F)_{weighted}$ using (U x F) matrix and Inverse User Frequency IUF. Then obtain the (U x U) similarity matrix with the help of $(U \times F)_{weighted}$.

d) Finally an item is recommended to the user based on not only the neighbourhood but also on the features of the item in the neighbourhood.

Block Diagram

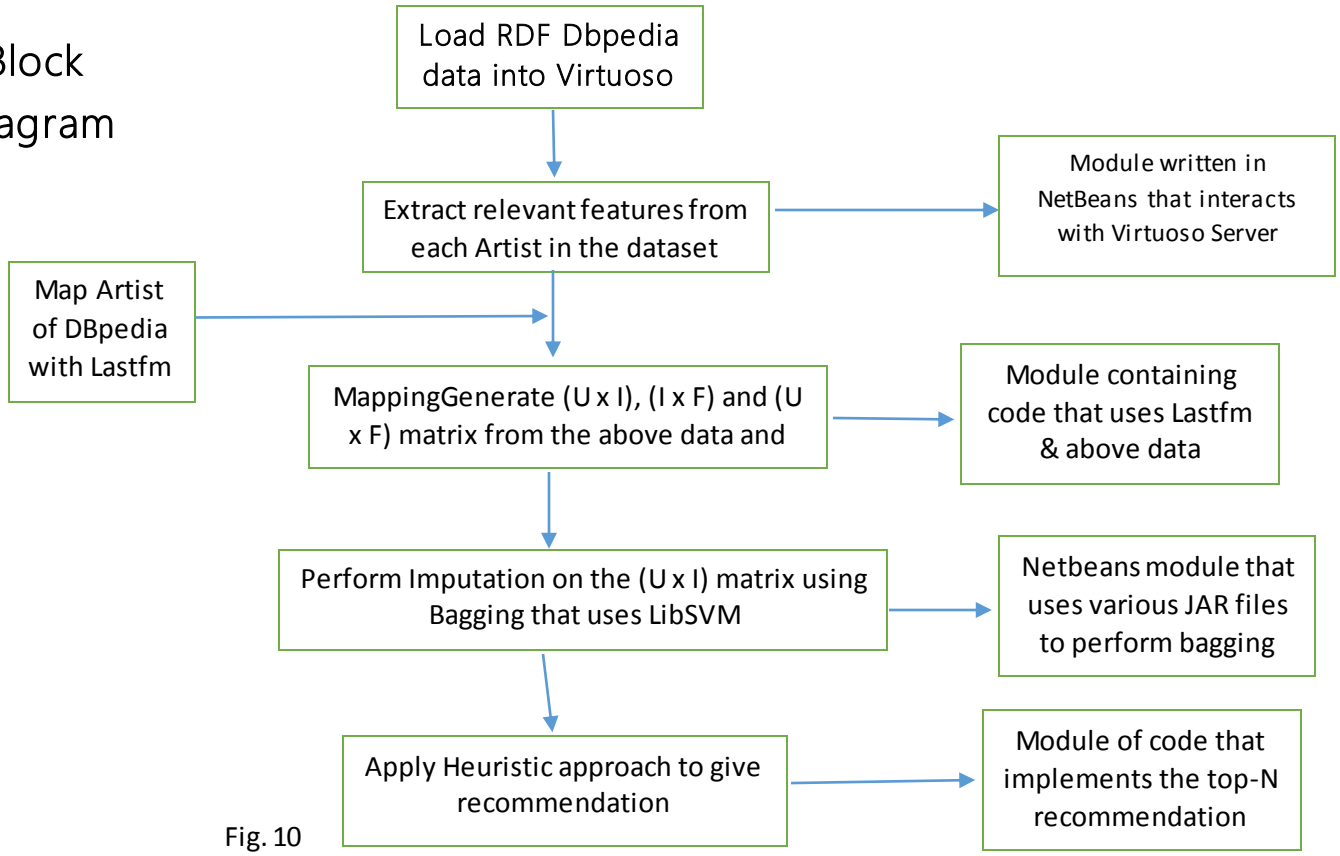

Fig. 10

Data loading, collection and pre-processing

a) Dbpedia 3.9 data of size 8 GB was loaded into Virtuoso 7.1. The data set later expanded to 20 GB in graph format when it was fully loaded.

Virtuoso is an innovative enterprise grade multi-model data server for agile enterprises & individuals. It delivers an unrivalled platform agnostic solution for data management, access, and integration.

The unique hybrid server architecture of Virtuoso enables it to offer traditionally distinct server functionality within a single product offering that covers the following areas:

- Relational Data ManagementRDF Data Management
- XML Data Management

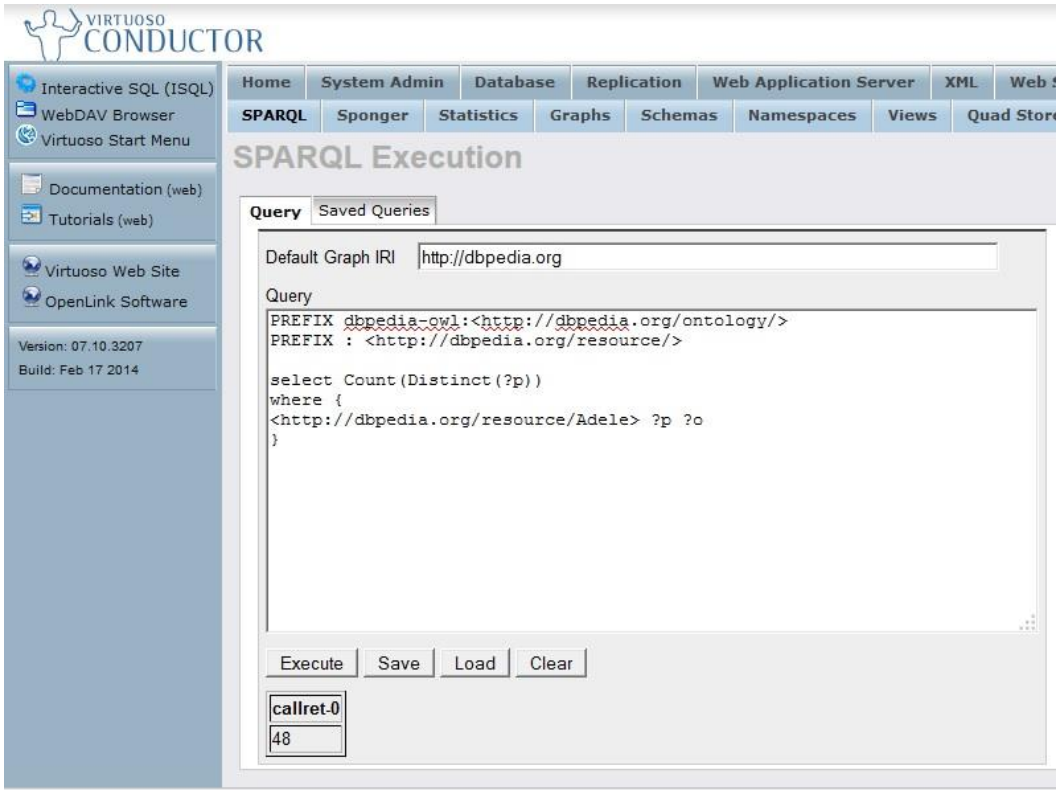

Fig. 11

- Free Text Content Management & Full Text Indexing
- Document Web Server
- Linked Data Server

b) Each Artist in DBpedia has 54 distinct attributes but all of them were not relevant to our recommendation process. We hence select only those attributes that give us "useful" knowledge. We selected
   - dbpedia-owl:genre
   - dcterms:subject
   - is dbpedia-owl:artist of
   - is dbpedia-owl:associatedMusicalArtist of

c) Using the above extracted data we create an item-feature matrix (I x F). Similarly we can create U x I (user-item) and U x F (user-feature) matrix from the Lastfm data.

4.2 Imputation

Why we need Imputation?

A serious problem in mining industrial data bases is that they are often incomplete, and a significant amount of data is missing. This paper explores the use of machine-learning based alternatives to standard statistical data completion (data imputation) methods, for dealing with missing data. We have approached the data completion problem using well-known machine learning techniques.

Validity of Imputation

This imputation approach is based on the hypothesis that the given user probably ranks closely on the items in the same genre, which is also accordant with common sense.

For example, Elma likes romance movies, while dislikes horror ones, well then she is likely to give high ratings on her unrated romance movies and low ratings on the horror ones.

It suggests that making use of item's genre information and user's original rating to fill up the missing data is an appropriate way to smooth the sparsity problem in CF, so we try to calculate user's mean rating on every item genre as the imputation values turning the original sparse rating into a full-rating matrix

How we perform Imputation?

To smooth the data sparsity problem, we propose a new CF framework, which uses missing data imputation as a pre-treatment stage before similarity computation.

In the imputation, we combine item's genre information and original ratings to generate the proper value to fill up the user-item rating matrix.

For training and developing the model of predicting user item ratings, we are using software tool WEKA which helps us to use various classifier algorithms and to choose best one which suits our dataset.

With the help of WEKA we show performance measures of various classifiers.

4.3 Recommendation part

Brief description of the approach:
(i) We construct a novel feature-weighted user model, which discloses the duality between users and features.
(ii) Based on Information Retrieval, we include the Term Frequency Inverse Document Frequency (TFIDF) weighting scheme in CF.
(iii) We propose a new top-N generation list algorithm based on features frequency.

Neighbourhood size: The number, k, of nearest neighbours used for the neighbourhood formation is important because it can affect substantially the system's accuracy. In most related works, k has been examined in the range of values between 10 and 100. The optimum k depends on the data characteristics (e.g., sparcity). Therefore, CF and CB algorithms should be evaluated against varying k, in order to tune it.

Positive rating threshold: It is evident that recommendations should be "positive", as it is not success to recommend an item that will be rated with, e.g., 1 in 1-5 scale. Thus, "negatively" rated items should not contribute to the increase of accuracy. We use a rating-threshold, P, to recommended items whose rating is not less than this value. If we do not use a P_ value, then the results become misleading.

Train/Test data size: There is a clear dependence between the training set's size and the accuracy of CF and CB algorithms. Therefore, these algorithms should be evaluated against varying training data sizes.

Recommendation list's size: The size, N, of the recommendation list corresponds to a tradeoff: With increasing N, the absolute number of relevant items (i.e., recall) is expected to increase, but their ratio to the total size of the recommendation list (i.e., precision) is expected to decrease.

Proposed methodology

The outline of our approach consists of four steps:
1. The content-based user profile construction step: It constructs a content-based user profile from both collaborative and content features.

2. The feature-weighting step: We quantify the affect of each feature inside the user's profile (find important intra-user features) and among the users (find important inter-users features).

3. The formation of user's neighbourhood algorithm: To provide recommendations, we create the user's neighbourhood, calculating the similarity between each user.

5. The top-N list generation algorithm: We provide for each test user a Top-N recommendation list based on the most frequent features in his neighbourhood.

In the following, we analyse each step in detail. To ease the discussion, we will use the running example illustrated in Figure 1(a), where I_1–6 are items and U_1–4 are users. The null (not rated) cells are presented with dash.
Moreover, in Figure 1(b), for each item we have four features that describe its characteristics.

	I_1	I_2	I_3	I_4	I_5	I_6
U_1	-	4	-	-	5	-
U_2	-	3	-	4	-	-
U_3	-	-	-	-	-	4
U_4	5	-	3	-	-	-

(a)

	F_1	F_2	F_3	F_4
I_1	0	1	0	0
I_2	1	1	0	0
I_3	0	1	1	0
I_4	0	1	0	0
I_5	1	1	1	0
I_6	0	0	0	1

(b)

	F_1	F_2	F_3	F_4
U_1	2	2	1	0
U_2	1	2	0	0
U_3	0	0	0	1
U_4	0	2	1	0

(c)

Fig. 1. (a) User-Item matrix R, (b) Boolean Item-Feature matrix F (c) User-Feature matrix P

Fig. 12

The content-based user profile construction step

We construct a feature profile for a user from both user ratings and item features. In particular, for a user u who rated positively (above P) some items, we build a feature profile to find his favourite features. In particular, matrix R(u, i) denotes the ratings of user u on each item i. We use a Boolean matrix F, where F (i,f) element is one, if item i contains feature f and zero otherwise. In our running example, matrices R and F are illustrated in Figures 1 (a) and 1(0b, respectively. For a user u, his profile is constructed with matrix P(u, f), with elements given as follows:

$$P(u,f) = \sum_{\forall R(u,i) > P_{\tau}} F(i,f)$$

Therefore, P(u,f) denotes the correlation between user u and feature f. Notice that we use only the positively rated items i (i.e., R(u,i)> P_{τ}) by user u. In our running example (with P_ = 2), we construct the P matrix by combining information from R and F matrices. As we can see in Figure 1c, the new matrix P reveals a strong similarity (same feature preferences) between users U1 and U4. This similarity could not be derived from the corresponding user ratings in the R matrix.

Feature-weighting step

Let U be the domain of all users and Fu the set of features that are correlated with user u, i.e.,

Fu ={f € F | P(u,f)> 0}

Henceforth, user u and feature f are correlated when P(u,f)> 0.

We will weight the features of matrix P, in order to find

(i) those features which better describe user u (describe the Fu set).

(ii) those features which better distinguish him from the others (distinguishing him from the remaining users in the U domain). The first set of features provides quantification of intra-user similarity, while the second set of features provides quantification of inter-use dissimilarity.

In our model, motivated from the information retrieval field and the TFIDF scheme [1], intra-user similarity is quantified by measuring the frequency of each feature f for a user u. Henceforth, this factor is referred as Feature Frequency (FF) factor. Furthermore, inter-user dissimilarity is quantified by measuring the inverse of the frequency of a feature f among all users.

Henceforth, this factor is referred as Inverse User Frequency(IUF) factor.

Thus, Feature Frequency FF(u,f) is the number of times feature f occurs in the profile of user u. In our model, it holds that FF(u,f)=P(u,f). The User Frequency UF(f) is the number of users in which feature f occurs at least once. Finally, the Inverse User Frequency IUF(f) can be calculated from UF(f) as follows:

$$IUF(f) = \log (|U| / UF(f)) \qquad (2)$$

In Equation 2, |U| is the total number of users. The Inverse User Frequency of a feature is low, if it occurs in many users' profiles, whereas it is high, if the feature occurs in few users profiles. Finally, the new weighted value of feature f for user u is calculated as following:

$$W(u, f) = FF(u, f) \_ IUF(f) \qquad (3)$$

This feature weighting scheme represents that a feature f is an important indexing element for user u, if it occurs frequently in it. On the other hand, features which occur in many users' profiles are rated as less important indexing elements due to the low inverse user frequency.

In our running example, the matrix P of Figure 1c is transformed into the matrix W in Figure 2a. As it can be noticed in matrix P, features F1 and F2 for user U1 have the same value, equal to two. In contrast, in matrix W, the same features are weighted differently (0.60 and 0.24). It is obvious now that feature

	F_1	F_2	F_3	F_4
U_1	0.60	0.24	0.30	-
U_2	0.30	0.24	0	-
U_3	-	-	-	0.60
U_4	0	0.24	0.30	-

(a)

	U_1	U_2	U_3	U_4
U_1	-	0.96	0	1
U_2	0.96	-	0	1
U_3	0	0	-	0
U_4	1	1	0	-

(b)

Fig. 2. (a) weighted User-Feature matrix W (b) User-User similarity matrix

F1 for user U1 is an important discriminating feature, whereas this could not be notice in matrix P.

Fig. 13

User's neighbourhood formation step

To provide recommendations, we need to find similar users. In our model, as it is expressed by equation 4, we apply cosine similarity in the weighted user-feature W matrix. We adapt cosine similarity to take into account only the set of features, that are correlated with both users. Thus, in our model the similarity between two users is measured as follows:

$$\mathrm{sim}(u,v) = \frac{\sum_{\forall f \in X} W(u,f)W(v,f)}{\sqrt{\sum_{\forall f \in X} W(u,f)^2}\sqrt{\sum_{\forall f \in X} W(v,f)^2}}, X = \mathcal{F}_u \cap \mathcal{F}_v.$$

In our running example, we create a user-user matrix according to equation 4, where we can find the neighbours of each user (those which have the higher value, are the nearest ones). In Figure 2b, we can see that the nearest neighbour of user U2 is U4 with similarity 1, and U1 follows with similarity value 0.96.

Top-N list generation step

The most often used technique for the generation of the top-N list, is the one that counts the frequency of each positively rated item inside the found neighbourhood, and recommends the N most frequent ones. Our approach differentiates from this technique by exploiting the item features. In particular, for each feature f inside the found neighbourhood, we add its frequency. Then, based on the features that an item consists of, we count its weight in the neighbourhood. Our method, takes into account the fact that, each user has his own reasons for rating an item.

In our running example, assuming that we recommend a top-1 list for U2 (with k=2 nearest neighbours), we work as follows:

1. We get the nearest neighbours of U2: {U4, U1}.
2. We get the items in the neighbourhood: {I1, I3, I5}.
3. We get the features of each item: I1: {F2}, I3: {F2, F3}, I5: {F1, F2, F3}.
4. We find their frequency in the neighbourhood: fr(F1)=1, fr(F2)=3, fr(F3)=2.
5. For each item, we add its features frequency finding its weight in the neighbourhood: w(I1) = 3, w(I3) = 5, w(I5) = 6.

Thus, I5 is recommended, meaning that it consists of features that are prevalent in the feature profiles of U2's neighbours.

5. Tools, Software and Hardware requirements

5.1 Software and Hardware

1. Virtuoso 7.1
2. Weka 3
3. NetBeans IDE 7.4
4. LibSVM
5. Adaboost

5.2 Datasets

1. Last.fm 2011
2. Dbpedia 3.9

Ontologies used

The DBpedia Ontology is a shallow, cross-domain ontology, which has been manually created based on the most commonly used infoboxes within Wikipedia. The ontology currently covers 529 classes which form a subsumption hierarchy and are described by 2,333 different properties.

Since the DBpedia 3.7 release, the ontology is a directed-acyclic graph, not a tree. Classes may have multiple superclasses, which was important for the mappings to schema.org.

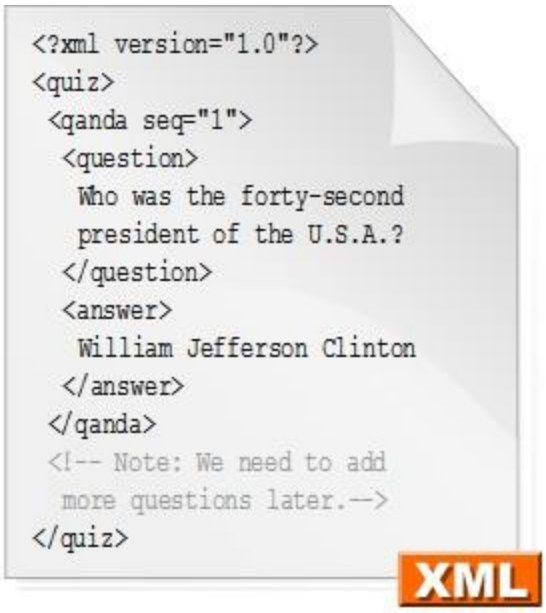

Fig. 14

Fig. 15

For many resources there are values for properties under two namespaces: dbpprop (http://dbpedia.org/property/) and dbpedia-owl (http://dbpedia.org/ontology/).

The first (dbprop) contains the value extracted from the infobox (as raw as the extraction framework can make it).

http://dbpedia.org/property/birthDate

http://dbpedia.org/property/homeTown

http://dbpedia.org/property/occupation

	▪ dbpedia:West_Norwood
dbpedia-owl:occupation	▪ dbpedia:Adele__1
dbpedia-owl:viafId	▪ 21955977
dbpedia-owl:wikiPageCharacterSize	▪ 83667 (xsd:integer) ▪ 84719 (xsd:integer)
dbpedia-owl:wikiPageEditLink	▪ http://en.wikipedia.org/w/index.php?title=Adele&action=edit
dbpedia-owl:wikiPageExternalLink	▪ http://adele.tv/ ▪ http://adele.tv
dbpedia-owl:wikiPageExtracted	▪ 2014-04-19 03:01:44 (xsd:date) ▪ 2014-05-08 18:25:46 (xsd:date)
dbpedia-owl:wikiPageHistoryLink	▪ http://en.wikipedia.org/w/index.php?title=Adele&action=history
dbpedia-owl:wikiPageID	▪ 13041163 (xsd:integer)
dbpedia-owl:wikiPageModified	▪ 2014-04-17 08:22:14 (xsd:date) ▪ 2014-05-06 10:13:13 (xsd:date)
dbpedia-owl:wikiPageOutDegree	▪ 311 (xsd:integer) ▪ 313 (xsd:integer)
dbpedia-owl:wikiPageRevisionID	▪ 604566926 (xsd:integer) ▪ 607304523 (xsd:integer)
dbpedia-owl:wikiPageRevisionLink	▪ http://en.wikipedia.org/w/index.php?title=Adele&oldid=604566926 ▪ http://en.wikipedia.org/w/index.php?title=Adele&oldid=607304523
dbpprop:after	▪ dbpedia:Little_Boots ▪ dbpedia:Zac_Brown_Band ▪ incumbent
dbpprop:alternativeNames	▪ Adele
dbpprop:before	▪ dbpedia:Amy_Winehouse ▪ dbpedia:Mika_(singer) ▪ Jack White and Alicia Keys ▪ (Another Way to Die, 2008)
dbpprop:birthDate	▪ 1988-05-05 (xsd:date)
dbpprop:birthName	▪ Adele Laurie Blue Adkins
dbpprop:birthPlace	▪ Tottenham, Greater London, England
dbpprop:caption	▪ Adele in concert, January 2009
dbpprop:children	▪ Angelo Konecki
dbpprop:dateOfBirth	▪ 1988-05-05 (xsd:date)
dbpprop:description	▪ On "Don't You Remember", the song contains bluesy guitar riffs and instrumental b examples of the country music influences that permeate the album. ▪ One of the more pop-influenced songs on the album, the "epic" power ballad "Set F she alternatively affirms and laments the end of her relationship.
dbpprop:filename	▪ AdeleDon't You Remember.ogg ▪ Set Fire to Rain.ogg
dbpprop:hasPhotoCollection	▪ http://wifo5-03.informatik.uni-mannheim.de/flickrwrappr/photos/Adele

Fig. 16

The second (dbpedia-owl) contains the mapped property.

http://dbpedia.org/ontology/Stadium

http://dbpedia.org/ontology/Arena

http://dbpedia.org/ontology/Building

Since different infoboxes may use different property names for the same attribute, we map them to a consistent ontology via mappings.dbpedia.org.

6.Activity Time chart

Before Mid-Sem
1st Feb. Literature Survey which included learning about various approaches to our problem.
15th Feb. Loading dbpedia music database into virtuso 7.1
5th March Understood basic querying mechanism from Netbeans to Virtuso SPARQL endpoint.
After Midsem
25th April Implemented the algorithm and got the recommendations.
6th May Tested the Performance of Algorithm and checked how good the recommendations are.
7th may Designed the GUI and checked the working of all modules

7. Performance study And Results

In this section, we study the performance of our (feature-weighted user) model against the well-known CF, CB and a hybrid algorithm, by means of a thorough experimental evaluation. For the experiments, the F eatured-Weighted User Model is denoted as FWUM, the collaborative filtering algorithm as CF and the contentbased algorithm as CB. Finally, as representative of the hybrid algorithms, we have implemented an algorithm, Factors that are treated as parameters, are the following, the neighborhood size (k, default value 10), the size of the recommendation list (N , default value 20) and the size of train set (default value 75%). The metrics we use are recall, precision, and F1. Pτ threshold is set to 3. Finally, we consider the division between not hidden and hidden data. For each transaction of a test user we keep the 75% as hidden data (the data we want to predict) and use the rest 25% as not hidden data (the data for modeling new users).

7.1 Comparative Results for CF algorithms :

Firstly, we compare the two main CF algorithms, denoted as user-based (UB) and item-based (IB) algorithms. The basic difference between these two CF algorithms is that, the former constructs a user-user similarity matrix while the latter, builds an item-item similarity matrix. Both of them, exploit the user ratings information(user-item R matrix). Figure 3 demonstrates that item-based CF compares favourably against user-based CF for small values of k. For large values of k, both algorithms converge, but never exceed the limit of 40% in terms of precision.

The reason is that as the k values increase, both algorithms tend to recommend the most popular items. In the sequel, we will use the IB algorithm as a representative of CF algorithms.

7.2 Comparative Results for CB algorithm

As it is already discussed, we have extracted 4 different classes of features from the imdb database. We test them using the pure content-based CB algorithm to reveal the most effective in terms of accuracy. Pure CB algorithm exploits information derived only from document or item features.

Thus, we create an item-item similarity matrix based on cosine similarity applied on features of items (by exploiting information only from the item-feature F matrix). In Figure 3b, we see results in terms of precision for the four different classes of extracted features.

As it is shown, the best performance is attained for the "keyword" class of content features.

7.3 Comparative Results for CF, CB, CFCB and FWUM algorithms

We test the FWUM algorithm against CF, CB and CFCB algorithms using the best options as they have resulted from the previous measurements. In Figures 4a and 4b, we see results for precision and recall. FWUM presents the best performance in terms of precision (above 60%) and recall(above 20%). The reason is two-fold:

(i) the sparsity has been downsized through the features and

(ii) the applied weighting-schema reveals the actual user preferences.

7.4 Examination of Additional Factors

Recommendation list's size: We examine the impact of N . The results of our experiments are depicted in Figures 5a and 5b. As expected, with increasing N, recall increases and precision decreases. Notice that the FWUM outperforms CF, CB and CFCB in all cases. The relative differences between the algorithms are coherent with those in our previous measurements.

7.5 Why Boosting and Not LIBSVM

LIBSVM is a hard classifier. All classifiers including LIBSVM are performing poorly for our dataset, because of less information in our dataset. Accuracy of LIBSVM is poor so we experimented with various classifiers and finally we found out Ensemble learners are yielding better results than normal classifiers. In ensemble techniques we opted Boosting with decision stumps.

Ensemble Learners

Ensembles combine multiple hypotheses to form a (hopefully) better hypothesis. In other words, an ensemble is a technique for combining many weak learners in an attempt to produce a strong learner. The term ensemble is usually reserved for methods that generate multiple hypotheses using the same base learner. The broader term of multiple classifier systems also covers hybridization of hypotheses that are not induced by the same base learner.

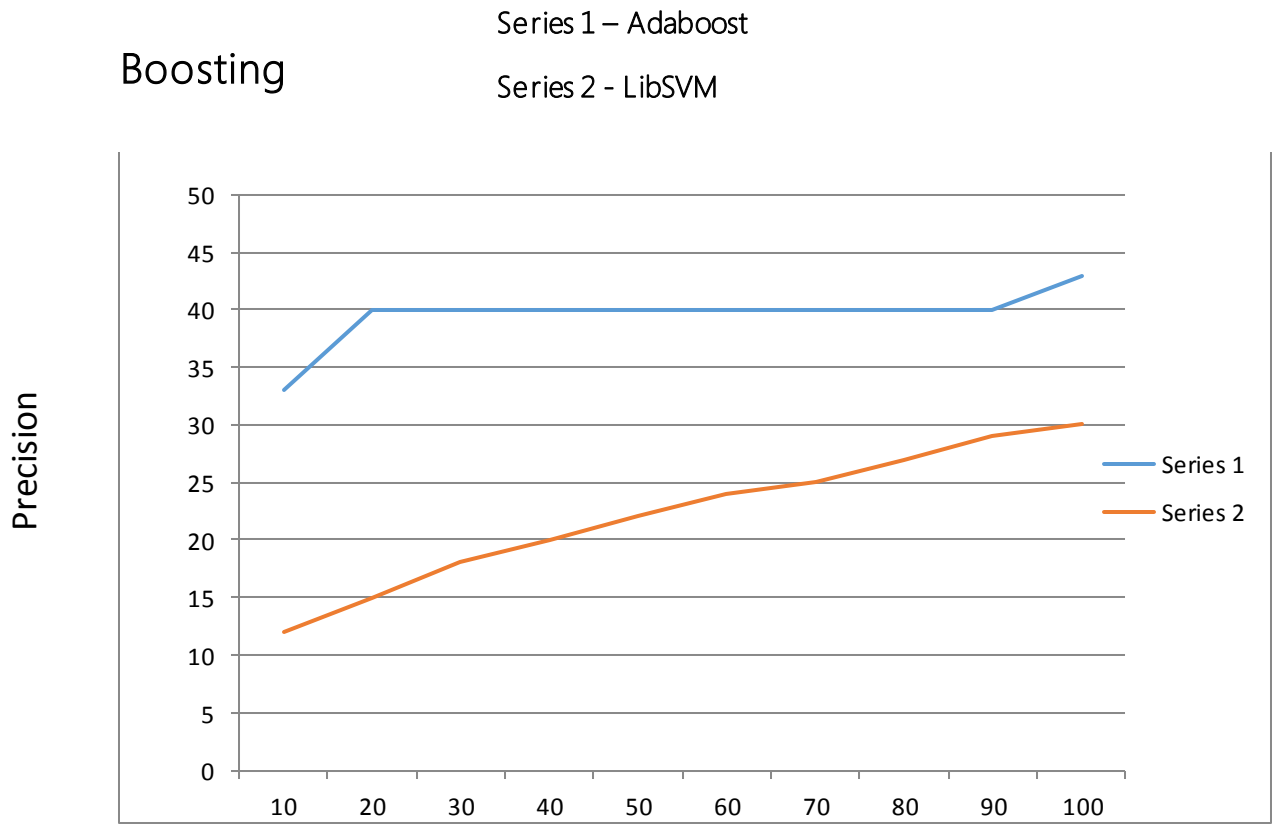

Fig. 17

Boosting involves incrementally building an ensemble by training each new model instance to emphasize the training instances that previous models mis-classified

Decision Stump

A decision stump is a machine learning model consisting of a one-level decision tree. That is, it is a decision tree with one internal node (the root) which is immediately connected to the terminal nodes (its leaves). A decision stump makes a prediction based on the value of just a single input feature. Sometimes they are also called 1-rules.

Decision stumps are often used as components (called "weak learners" or "base learners") in machine learning ensemble techniques such as bagging and boosting. For example, a state-of-the-art Viola–Jones face detection algorithm employs AdaBoost with decision stumps as weak learners.

Series 1 – CB

Series 2- CF

Series 3 – CFCB

Series 4 – FWUM

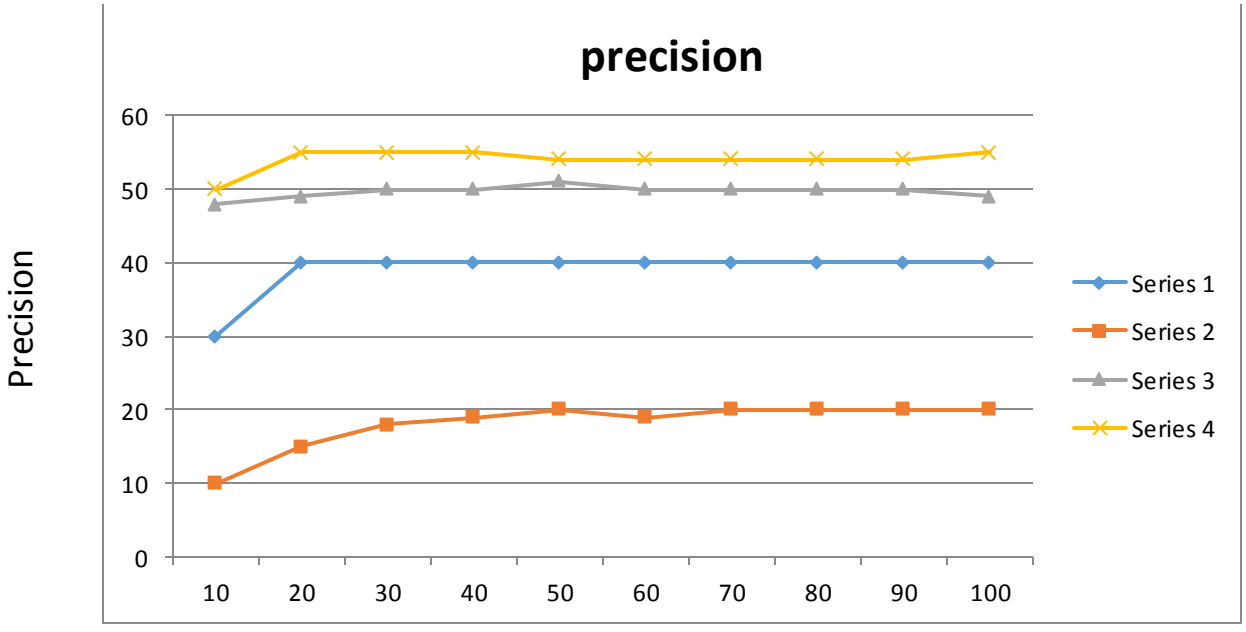

Fig. 18

RECALL:

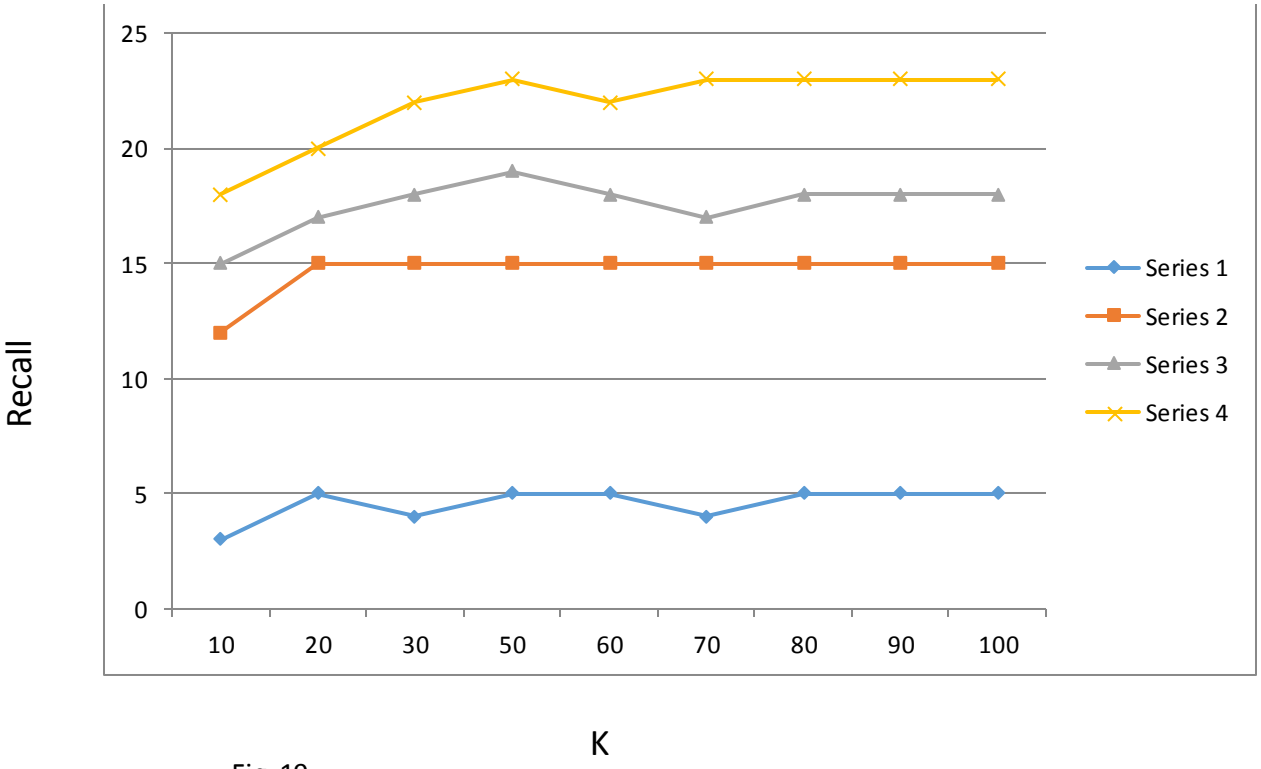

Fig. 19

Series 1 – CB

Series 2- CF

Series 3 – CFCB

Series 4 – FWUM

7.6 Results

We have got the top-N recommendations for a user number 678.

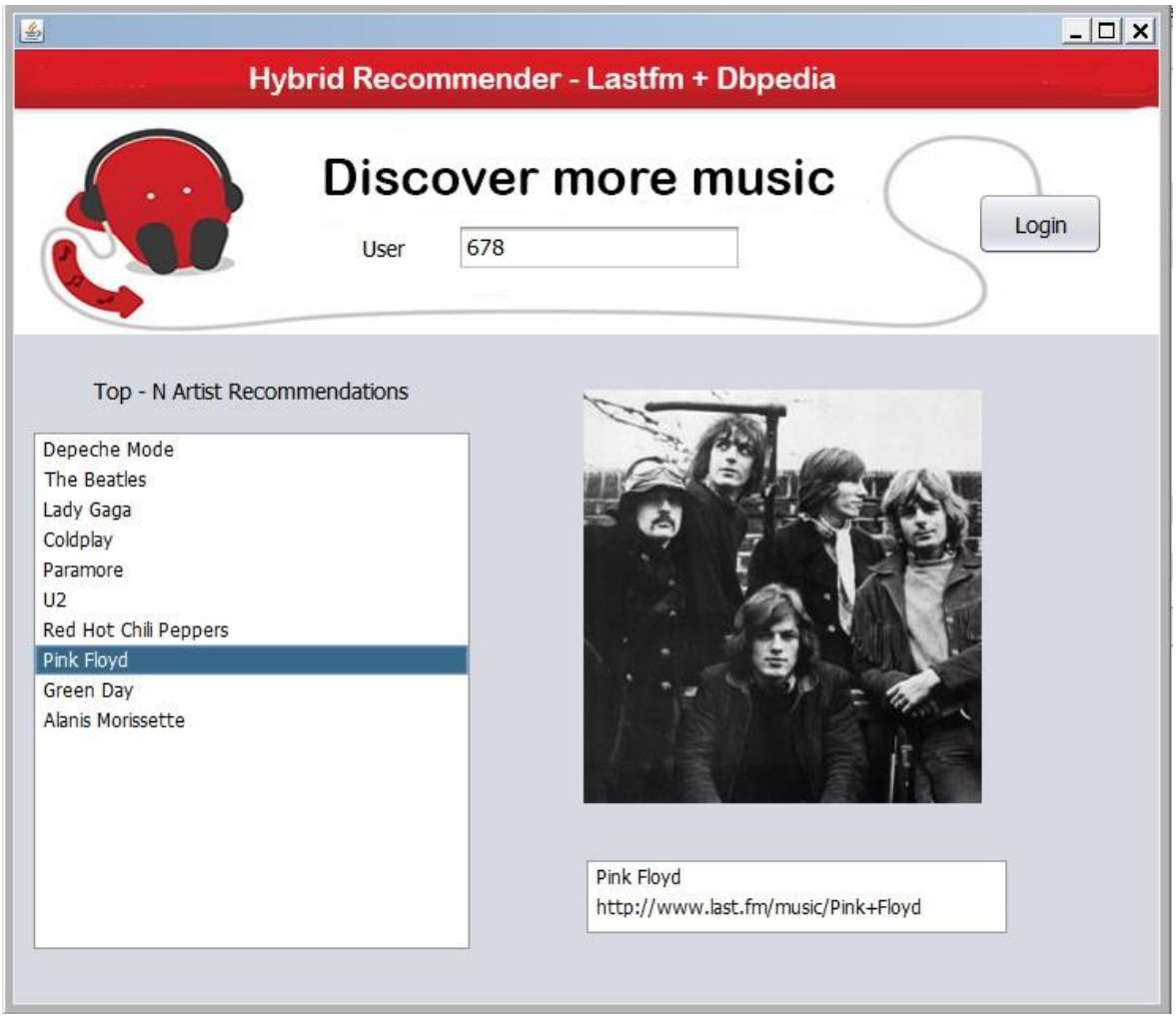

Fig. 20

8. Conclusion and future scope

We proposed a feature-weighted user model for recommender systems. We perform experimental comparison of our method against well known CF, CB and a hybrid algorithm with a real data set.

– Our approach builds a feature profile for the users, that reveals the real reasons of their rating behavior.

– Our approach shows significant improvements in accuracy of recommendations over existing CF, CB and a hybrid algorithm.

– The generation of the top-N recommendation list which is based on features reveals the favourite features of a user and recommends those items that are composed of these features.

Summarizing the aforementioned conclusions, our feature-weighted user model is a promising approach for getting more robust recommender systems.

In our future work, we will consider the fusion of different classes of features in a multivariate user model for getting more precise recommendations.

Currently our data set does not has any user information apart from the artist he has heard and the tags given. But in future if we get information like demographic, social etc. about the user we would be able to provide better recommendations.

For ex. a user whose nationality is French and is also of age less than 25, then we can give recommend French songs that have been released in the past 5 years.

Future works in terms of implementations can be done by translating this Java application into an Android app like Beats Music App etc. which takes use of dynamic user listening history and the rich social data available to give extremely good recommendations.

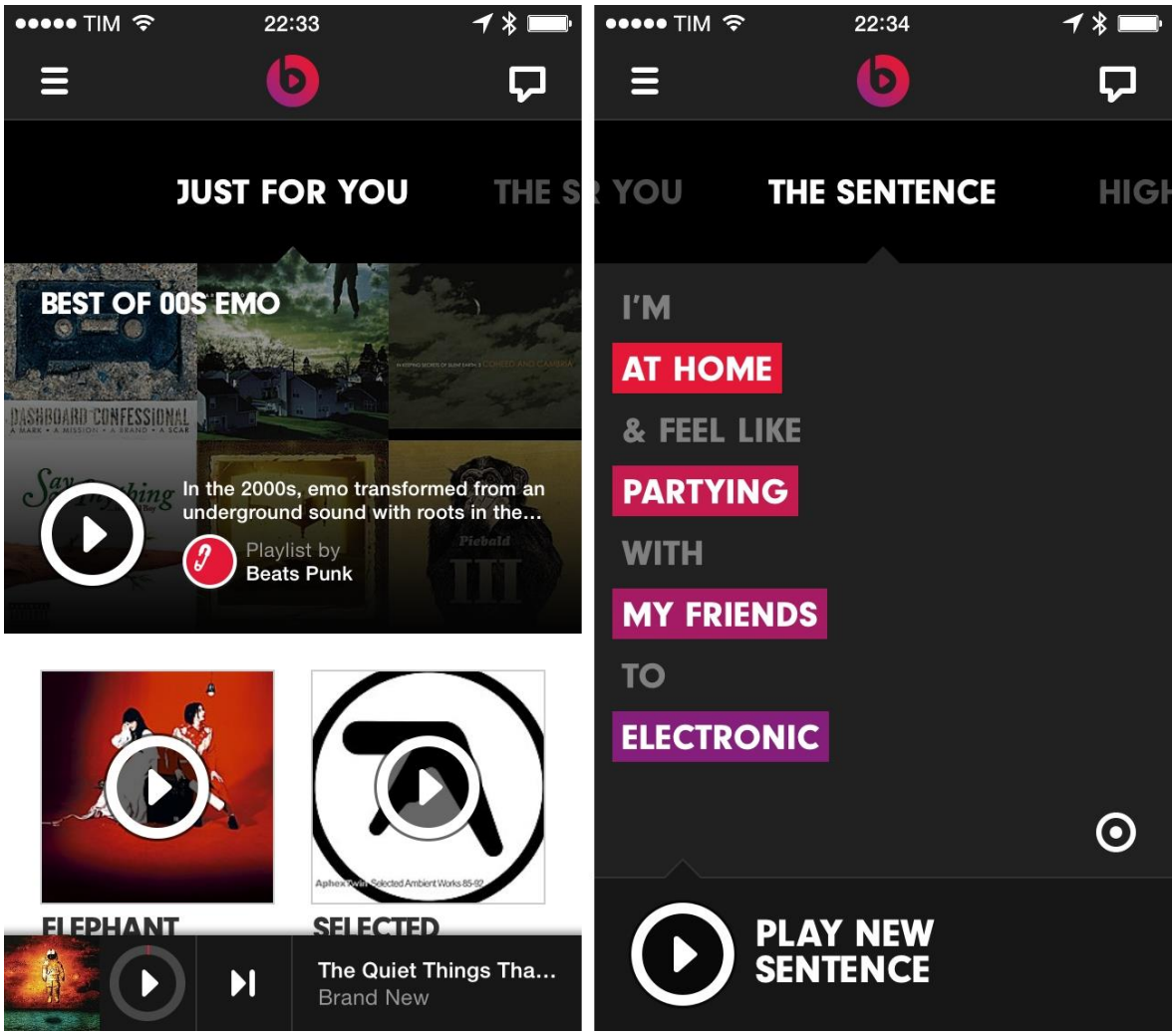

Fig. 21

References

[1] Nikos Bikakis, "XML and Semantic Web W3C Standards Timeline". 2012-02-04

[2] Tim Berners-Lee, "world wide web", Internet: www.w3.org/People/Berners-Lee.

[3] Bizer, Christian,Heath, Tom, Berners-Lee, Tim . "Linked Data—The Story So Far". *International Journal on Semantic Web and Information Systems* vol.no. 3, pp. 1–22.

[4] M. Mealling,URI Planning Interest Group, W3C/IETF (21 September 2001). "URIs, URLs, and URNs: Clarifications and Recommendations 1.0". Internet: http://www.ietf.org/rfc/rfc3305.txt.

[5] Segaran, Toby, Evans, Colin ,Taylor, Jamie . "*Programming the Semantic Web*"., 1005 GravensteinHighway North, Sebastopo : O'Reilly Media, Inc.

[6] Carlson Sch, Minneapolis, "Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions", *Knowledge and Data Engineering, IEEE Transactions on* (Volume:17 , Issue: 6), June 2005, pp. 734 - 749

[7] Panagiotis Symeonidis, Alexandros Nanopoulos, and Yannis Manolopoulos,"Feature-weighted user model for recommender systems", In *Proc. 11th international conference on User Modeling, Lecture Notes in Computer Science*, Vol. 4511, Springer (2007), pp. 97–106.

[8] Zhonghang Xia, Yulin Dong, Guangming Xing, "Support Vector Machines For Collaborative Filtering". In *Proc. (ACMSE) 44th annual Southeast regional conference*, pp. 169-174, 2006.

[9] K. Lakshminarayan, S.A. Harp, R. Goldman, and T. Samad, "Imputation of missing data using machine learning techniques," in *Proceedings: Second International Conference on Knowledge Discovery and Data Mining*, pp. 140–145, 1996.

[10] Yoav Freund, Robert E. Schapire, "A Short Introduction to Boosting", *Journal of Japanese Society for Artificial Intelligence,14(5):771-780, September, 1999.*

[11] Yongli Ren, Tianqing Zhu, Gang Li, Wanlei Zhou, "Top-N Recommendations by Learning User Preference Dynamics ", *Springer Berlin Heidelberg, 17th Pacific-Asia Conference, PAKDD 2013, Gold Coast, Australia*, April 14-17, 2013, Proceedings, Part II, pp 390-401.

Suggestions of the Board members