Pratima Kshetry, pkshetry@umd.edu

Collaborative Filtering Recommendation Engine Design

Modules:

1. **Data Reader**

We use a CustomInputFormat of FileInputFormat in hadoop to read data from the text file to generate the product profiles and customer profiles.

2. **Generate product profiles** — A text file with its name as the identifier of the product(ASIN) and the features listed as the content of the product. The product profile will consist of the features : group, sales rank, average rating and categories(this feature will be in form of tags or words with stop words removed).

3. **Classification of items:**

The products can be classified based on similar group categories using TF-IDF. We can generate TF-IDF for product files where terms would be category tags and documents will be the product profiles. We can save this TF-IDF matrix in a file. We can have a threshold value set for the TF*IDF calculation.
HOW:

1. We can use apache lucene to index all the product files with the categories(tags in product files) as the terms indexed.

Note: We could use Bayes classifier as well her but I am not sure about what is the best implementation technique for it.

4. **Customer profile**

This will be a hashtable of vectors of type {String, Vector<String,String>} where the key will be the customerID and the vector will be the ASIN of the products purchased and the rating given by the customer.

5. **Generate Utility Matrix**

The rows will represent customerID, ASIN's of the purchased products will be the columns and the value will be the rating given by the customer to the product. In case of no rating it will be blank(or null). We can use correlation similarity coeffcient to determine the similarity of user preference of products in the recommendation engine.

Use hadoop to generate this matrix. where the mapper receives an input of the customer profile vector of InputKey: {customerID} inputValue: {ASIN, rating}

6. Recommendation Engine

The recommendation engine will find matching items for a user where ranking will be based on following :

a. Item-Item similarity

For the items a customer purchases we will generate a list of products with the most similar category listing (based on step 4). The top 10 similar product ASINs will be retrieved. The similarity will be calculated using cosine similarity function.
The customer profile vector will indicate which files to be searched for terms. Only the terms listed under categories will be fetched from the product profiles and will be used for the TF-IDF calculation.

Pratima Kshetry, pkshetry@umd.edu

b. User-User similarity
Using Jaccard distance from the matrix in step 5, we will compute the top 5 similar profiles.
We can have a threshold of Jacquard distance
Say two customers A and B have a Jaccard distance of 0.7
We will generate list of products that a customer A has not bought but customer B has from the Customer profiles of A and B.

c. Boost parameters to generate ranked list
Once we have list of similar items from step (a) and (b) we will use sales rank, average rating (retrieved for all the listed products) as the boost parameters for each item to generate a list of products ranked by its sale rank and average rating.

This ranked list will be recommended to the customer.

7. **Evaluation - Not looked into this much.**
Training and Test data.
For evaluation,

1. 100 random customers with product purchased greater or equal to 20 and less than 30 are selected while datapoints for clustering are generated.

2. For each such customer only half of the products rated by them are fed into reco engine. These products are exported to include.txt file. This will serve as Training data.

3. For the same customers other half of the products rated are not fed into reco engine and hence they can serve as test data. These data are exported in exclude.txt. This will serve as Test data.
For testing purpose filter out recommended products for only products considered in include.txt

4. Check recommended products whether they fall in excluded list of that customer