Movie Recommendation System

DA 331 : Mid Presentention

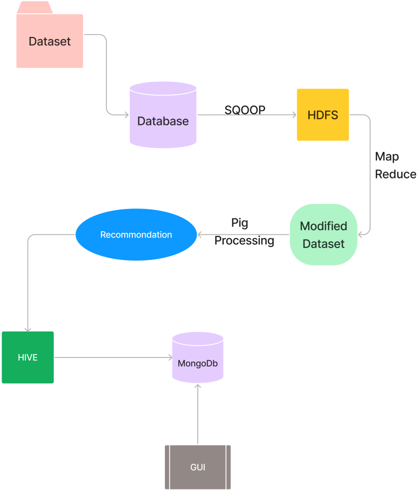
Vikram Singh 200121063
Apurv Kushwaha 200121008

05-11-2023

Outline

- Workflow
- Data-set Explanation
- Recommendation Systems
- How Recommendation Systems Work?
- Conclusion

Workflow



Dataset
Database
SQOOP
HDFS
Map
Reduce
Modified
Dataset
Pig
Processing
Recommondation
HIVE
MongoDb
GUI

Dataset - User

```
1|24|M|technician|85711
2|53|F|other|94043
3|23|M|writer|32067
4|24|M|technician|43537
5|33|F|other|15213
6|42|M|executive|98101
7|57|M|administrator|91344
8|36|M|administrator|05201
9|29|M|student|01002
10|53|M|lawyer|90703
```

Figure 1: user id | age | gender | occupation | zip code

Dataset - Movie

```
1|Toy Story (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Toy%20Story%20(1995)|0|0|0|1|1|1|0|0|0|0|0|0|0|0|0|0|0|0|0
2|GoldenEye (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?GoldenEye%20(1995)|0|1|1|0|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0
3|Four Rooms (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Four%20Rooms%20(1995)|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0
4|Get Shorty (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Get%20Shorty%20(1995)|0|1|0|0|0|1|0|0|1|0|0|0|0|0|0|0|0|0|0
5|Copycat (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Copycat%20(1995)|0|0|0|0|0|0|1|0|1|0|0|0|0|0|0|0|1|0|0
```

Figure 2: | movie id | title | release date | IMDb URL | (GENRE) |

Dataset - Genre

```
unknown|0
Action|1
Adventure|2
Animation|3
Children's|4
Comedy|5
Crime|6
Documentary|7
Drama|8
Fantasy|9
Film-Noir|10
Horror|11
Musical|12
Mystery|13
Romance|14
Sci-Fi|15
Thriller|16
War|17
Western|18
```

Figure 3: | Genre | ID |

Dataset

196	242	3	881250949
186	302	3	891717742
22	377	1	878887116
244	51	2	880606923
166	346	1	886397596
298	474	4	884182806
115	265	2	881171488
253	465	5	891628467
305	451	3	886324817

Figure 4: | User ID | Movie ID | Rating | TimeStamp |

Recommendation Systems

Types of Recommendations:

1. **Editorial and Hand Curated**
   - List of Favourites
   - List of "essential" items
   - No input from user
2. **Simple Aggregates**
   - Depends on aggregate users not individuals
   - Top 100, Most Popular, Recent Uploads
   - No input from user
3. **Tailored to Individual Users**
   - Recommendation according to individual user

Formal Model

- C = Set of Customers
- S = Set of Items
- Utility function $u : C \times S \to \mathbb{R}$
  - $\mathbb{R}$ = Set of Ratings
  - $\mathbb{R}$ is a totally ordered set
  - For example, 0-5 stars, real numbers in $[0, 1]$

Utility Matrix

	Avatar	LOTR	Matrix	Avengers
A	1	?	0.2	?
B		0.5		0.3
C	0.2		1	
D				0.4

Table 1: Ratings

Problems

1. Gathering Data
   - Explicit Methods : Asking users to rate
   - Implicit Methods : Assuming Possibilities
2. Extrapolating Utilities: 3 Approaches
   - Collaborative Filtering
   - Content-Based Recommending
   - Hybrid Modeling

Content-based Recommendations

Main Idea: Recommend items to customer x similar to previous items rated highly by x.

Examples:

- **Movies:**
  - Same actor(s), director, genre.
- **Websites, Blogs, News:**
  - Articles with "similar" content.

Item Profiles

For each item, create an item profile.
Profile is a set of features:

- **Movies:** author, title, actor, director,...
- **Images, Videos:** metadata and tags.

Convenient to think of the item profile as a vector:

- One entry per feature (e.g., each actor, director,...).
- Vector might be boolean or real-valued.

Selecting Important Words for Profiles

Profile: A set of "important" words in an item (document).
How to pick important words?

- Usual heuristic from text mining is **TF-IDF (Term Frequency-Inverse Document Frequency)**.

Sidenote: TF-IDF

TF (Term Frequency):

$$TF_{ij} = \frac{f_{ij}}{\max_k f_{kj}}$$

Where f_{ij} is the frequency of term (feature) i in document (item) j.
IDF (Inverse Document Frequency):

$$IDF_i = \log\left(\frac{N}{n_i}\right)$$

Where n_i is the number of documents that mention term i, and N is the total number of documents.
TF-IDF Score:

$$W_{ij} = TF_{ij} \times IDF_i$$

Note: We normalize TF to discount for "longer documents.

Creating Document Profiles

Doc Profile: A set of words with the highest TF-IDF scores, together with their scores.

User Profiles

User has rated items with profiles $i_1, i_2, \ldots, i_n$.

- **Simple Approach:** Compute a (weighted) average of rated item profiles.

Variant:

- Normalize weights using the average rating of the user.

Note: More sophisticated aggregations are possible.

Example: Star Ratings

Same Example, 1-5 Star Ratings

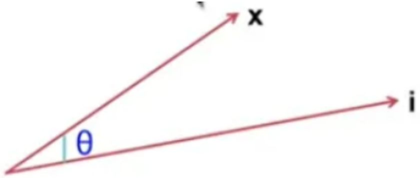
- Actor A's movies rated 3 and 5.
- Actor B's movies rated 1, 2, and 4.

Useful Step: Normalize Ratings

- Normalize ratings by subtracting the user's mean rating (3).
- Actor A's normalized ratings = 0, 2.
- Profile weight $= \frac{0+2}{2} = 1$.
- Actor B's normalized ratings = -2, -1, +1.
- Profile weight $= -\frac{2}{3}$.

Cosine Similarity Measure

User Profile x**, Item Profile** i
Estimate $U(x, i) = \cos(\theta) = \frac{x \cdot i}{|x| \cdot |i|}$



Theta:

- Technically, the cosine distance is the angle θ.
- The cosine similarity is the angle $180^\circ - \theta$.

For convenience, we use $\cos(\theta)$ as our similarity measure and refer to it as the "cosine similarity" in this context.

Pros of Content-based Approach

- **No need for data on other users:** Content-based recommendations rely solely on user preferences and item features.
- **Able to recommend to users with unique tastes:** Customized recommendations based on user profiles.

Cons of Content-based Approach

- **Overspecialization:** Content-based systems may overly specialize, leading to recommendations limited to the user's content profile. Users may have multiple interests.
- **Cold-Start Problem for New Users:** Building an accurate user profile for new users can be a challenge.

How to Build a User Profile?

- This depends on the specific content-based recommendation system and the type of items being recommended.
- User profiles are typically constructed based on user interactions, item attributes, and various techniques like TF-IDF, vector representations, or other feature extraction methods.

Collaborative Filtering

- Does not build item profiles or user profiles.
- In place of item-profile (user-profile), we use its row (column) in the utility matrix.

Comes in Two Flavors:

1. **User-User Collaborative Filtering**
2. **Item-Item Collaborative Filtering**

User-User Collaborative Filtering

- **Consider User x.**
- **Find Set N of Other Users Whose Ratings Are "Similar" to x's Ratings.**
- **Estimate x's Ratings Based on Ratings of Users in N.**

Similar Users (1)

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

- **Consider users x and y with rating vectors r_x and r_y.**
- **We need a similarity metric $sim(x, y)$.**
- **Capture intuition that $sim(A, B) > sim(A, C)$.**

Option 1: Jaccard Similarity

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

- **Similarity Metric: Jaccard Similarity**
- $sim(A, B) = \frac{|r_A \cap r_B|}{|r_A \cup r_B|}$
- $sim(A, B) = \frac{1}{5}$
- $sim(A, C) = \frac{2}{4}$
- $sim(A, B) < sim(A, C)$
- **Problem:** Ignores rating values!

Option 2: Cosine Similarity

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4	0	0	5	1	0	0
B	5	5	4	0	0	0	0
C	0	0	0	2	4	5	0
D	0	3	0	0	0	0	3

- **Similarity Metric: Cosine Similarity**
- $sim(A, B) = \cos(r_A, r_B)$
- $sim(A, B) \approx 0.38$
- $sim(A, C) \approx 0.32$
- $sim(A, B) \geq sim(A, C)$, but not by much
- **Problem:** Treats missing ratings as negative.

Option 3: Centered Cosine

Normalise rating by subtracting the row mean

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	2/3			5/3	-7/3		
B	1/3	1/3	-2/3				
C				-5/3	1/3	4/3	
D		0					0

Centered Cosine Similarity (2)

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	2/3			5/3	-7/3		
B	1/3	1/3	-2/3				
C				-5/3	1/3	4/3	
D		0					0

- **Similarity Metric: Centered Cosine (Pearson Correlation)**
- $sim(A, B) = \cos(r_A, r_B) \approx 0.09$
- $sim(A, C) = \cos(r_A, r_C) \approx -0.56$
- $sim(A, B) > sim(A, C)$
- **Captures intuition better.**
- **Missing ratings treated as "average."**
- **Handles "tough raters" and "easy raters."**
- **Also known as Pearson Correlation.**

Rating Predictions

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

Prediction for User x and Item i:

1. **Option 1:**
$$r_{xi} = \frac{1}{k} \sum_{y \in N} r_{yi}$$
2. **Option 2:**
$$r_{xi} = \frac{\sum_{y \in N} s_{xy} r_{yi}}{\sum_{y \in N} s_{xy}}$$

where $s_{xy} = \text{sim}(x, y)$

Item-Item Collaborative Filtering

- So far: User-User Collaborative Filtering
- **Another View: Item-Item**
- For item i, find other similar items.
- **Estimate rating for item i based on ratings for similar items.**
- Can use the same similarity metrics and prediction functions as in the user-user model.
- To estimate a user's rating r_{xi} for item i, we can use the following formula:

$$r_{xi} = \frac{\sum_{j \in N(i,x)} S_{ij} \cdot r_{xj}}{\sum_{j \in N(i,x)} S_{ij}}$$

  where:
  - S_{ij} represents the similarity of items i and j.
  - r_{xj} is the rating of user x on item j.
  - $N(i,x)$ is the set of items which were rated by user x and are similar to item i.

Result: Verifying the Output

- We removed 100 ratings from the original data set and stored them separately.
- Predicted these removed ratings using Collaborative Filtering and Hybrid model.
- For comparison, we calculated the Root Mean Squared Error (RMSE) in the Predicted Ratings with respect to the original ratings.