Project presentation
Movie Recommendation System

Sai Vamshi Dobbali, u1266122
Abishek Krishnan, u1261980

Contents

Statistical Filtering

Step-1: Choose how to “score” a movie

E.g. Ratings doesn’t always gives us a true popularity measure of the movie

Step-2: Choose movie’s parameter to select it into the list

E.g. Movies less than 25 min should not qualify as top rated movies

Step-3: Calculate score of all the movies which pass Step-1 and 2 criterion

Step-4: The decreasing order of scores, gives us the top movies

Statistical Filtering (contd.)

IMDB Formula

$$Weighted\ Rating\ (WR) = (\frac{v}{v+m} \times R) + (\frac{m}{v+m} \times C)$$

v - number of votes for a movie

m - minimum number of votes for a movie (We choose the 75th %)

R - rating of the movie

C - mean rating of all the movies (From our dataset - ~5.6 on a scale of 10)

Statistical Filtering (contd.) - Results

```
The list of top 10 movies are:
                original_title  score
1881  The Shawshank Redemption    8.5
3337             The Godfather    8.4
2294                千と千尋の神隠し    8.3
3865                  Whiplash    8.3
2731    The Godfather: Part II    8.3
3232              Pulp Fiction    8.3
1818          Schindler's List    8.3
662                 Fight Club    8.3
2170                    Psycho    8.2
1847                GoodFellas    8.2
```

Content based Filtering

Step-1: What content to use to find the similarity among movies?

E.g. Plot description, Genre, Director, Cast

Step-2: For “plot” based content, need to convert description into feature vectors

Step-3: We can either use count vectorization or TF-IDF vectorization

Content based Filtering (contd.)

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

$w_{i,j}$ is the weight of word i in plot description j

df_i is the number of plot descriptions that contain the term i

N is the total number of movies in our dataset

Content based Filtering (contd.)

Step-1: Extract and clean the plot description data

Step-2: Build TF-IDF vectors for each movie’s plot description

Step-3: Calculate the pairwise **cosine similarity** among movies

Step-4: Functionality, which can take the movie name as its argument

Content based Filtering (contd.) Results

```
Movies similar to The Shawshank Redemption are:
4531                 Civil Brand
3785                      Prison
609                  Escape Plan
2868                    Fortress
4727                Penitentiary
1779     The 40 Year Old Virgin
2667             Fatal Attraction
3871            A Christmas Story
434              The Longest Yard
42                   Toy Story 3
Name: title, dtype: object
```

User Based Collaborative Filtering

- What is User-Based Collaborative Filtering for our use case?

  is a technique used to predict the movies that a user might like on the basis of ratings given to movies by the other users who have similar taste with that of the target user.

- How to find similarity between users?

  Create a user x movie rated matrix, then we can either apply

  - Pearson Coefficient
  - Cosine Similarity

User Based Collaborative Filtering (contd.)

Types which we have implemented,

- Hardcoded the predicted rating
- Average rating of all the users ratings
- Weighted average of the user ratings based on cosine similarity

User Based Collaborative Filtering (contd.) Results

- Hard coding the predicted rating gave RMSE value of **1.3**
- Mean of all the ratings of the movie gave RMSE value of **1.02**
- In the third step, we represented each user as vector of “movies the user rated” and performed pairwise cosine similarity on all the users.

user_id	1	2	3	4	5	6	7	8	9	10	...	934	935	936	937
user_id															
1	1.000000	0.118076	0.029097	0.011628	0.264677	0.312419	0.308729	0.224269	0.026017	0.286411	...	0.308475	0.055872	0.197862	0.131367
2	0.118076	1.000000	0.099097	0.107680	0.034279	0.152789	0.086705	0.078864	0.068940	0.092399	...	0.086927	0.259636	0.289092	0.318824
3	0.029097	0.099097	1.000000	0.252131	0.026893	0.062539	0.039767	0.089474	0.078162	0.037670	...	0.040918	0.019031	0.065417	0.055373
4	0.011628	0.107680	0.252131	1.000000	0.000000	0.045543	0.078812	0.095354	0.059498	0.053879	...	0.024226	0.050703	0.056561	0.107294
5	0.264677	0.034279	0.026893	0.000000	1.000000	0.202843	0.299619	0.163724	0.038474	0.153021	...	0.262547	0.048524	0.048312	0.022202
6	0.312419	0.152789	0.062539	0.045543	0.202843	1.000000	0.375963	0.131795	0.110944	0.400758	...	0.287549	0.080312	0.162988	0.182856

User Based Collaborative Filtering (contd.) Results

- Predicted rating (u , n) = ∑Sim(u, u’) * rating(u’)/∑Sim(u, u’)

- RMSE for cosine similarity: **1.01**

- All the RMSE values are calculated using k fold cross validation.
  (We divided the dataset into 5 folds and trained the model on four folds and tested on one fold.)

Top-N recommendations

- Used SVD algorithm on rated data
- This approximates the rating matrix R as a product of $P \cdot Q^T$

users × factors

.1	-.4	.2
-.5	.6	.5
-.2	.3	.5
1.1	2.1	.3
-.7	2.1	-2
-1	.7	.3

factors

P

•

items

factors

1.1	-.2	.3	.5	-2	-.5	.8	-.4	.3	1.4	2.4	-.9
-.8	.7	.5	1.4	.3	-1	1.4	2.9	-.7	1.2	-.1	1.3
2.1	-.4	.6	1.7	2.4	.9	-.3	.4	.8	.7	-.6	.1

Q^T

Top-N recommendations (contd)

One of the cleanup steps involved is,

- Before performing matrix factorization on rating data, we calculated average rating per movie, and subtracted this from each user / movie rating combination

- This subtracts movie bias from each user, we then go onto apply SVD on this data

Top-N recommendations (contd)

MovieID	1	10	100	1000	1002	1003	1004	1005	1006	1007	...	99	990	991
0	4.288861	-0.195950	0.059791	0.013902	-0.005269	0.068531	0.034708	0.051772	0.019438	0.069625	...	0.015478	0.025567	-0.089656
1	4.516912	0.942800	-0.185102	-0.070946	-0.031497	-0.132928	-0.088461	0.403236	-0.046937	0.974084	...	-0.207525	-0.057530	0.063368
2	-0.078546	0.907657	-0.012436	-0.018223	-0.004993	-0.009990	0.088658	-0.079309	-0.001477	0.027198	...	-0.010521	-0.017129	0.103189
3	3.800371	0.528449	-0.124235	-0.016057	-0.000831	-0.207614	-0.140680	0.048933	-0.088665	0.111042	...	0.062178	-0.091779	-0.176755
4	3.058320	0.926521	0.218438	0.169113	0.168728	0.271766	0.031149	0.133045	0.314324	-0.391314	...	0.126222	0.073419	1.065916
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

MovieID	1	10	100	1000	1002	1003	1004	1005	1006	1007	...	99	990	991	992	993	994	996	997	998	999
UserID																					
1	5	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
10	5	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
100	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
1000	5	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
1001	4	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

Top-N recommendations (contd.) Results

- We achieved RMSE of **0.8721** on MovieLens 1Million dataset using SVD technique.

- After reconstructing the rating matrix, we have ratings for all the users for every movie.

- Based on these rating we implemented topN recommendations per user, by sorting the predicted ratings and also making sure that these recommendations are not rated by user before.

Top-N recommendations (contd.) Results

Top 10 recommendations for user10:

Running Free (2000)	3.1943916666935435
Kansas City (1996)	2.56060769130894
Ladybird Ladybird (1994)	2.249825123017104
Big Blue, The (Le Grand Bleu) (1988)	2.132648461527276
Dog of Flanders, A (1999)	2.0915610203748427
Braindead (1992)	2.0132030671620145
Cell, The (2000)	1.7211436019468307
Class of Nuke 'Em High (1986)	1.642784583861926
Wonderland (1999)	1.5527635597957092
Alien (1979)	1.373468890382085

Thank you
Questions