CS512 FUN Projects - Spring 2016

Vivek Harilal Bapodra (vhb12)
Vedika Babu (vb305)
Yi Hou (yh523)

***Abstract*— The goal of the project is to provide movie recommendations to users. To accomplish this, the system uses the ratings given by the user to other movies, together with the ratings given by other similar users on similar movies. Personal information can be also be used for 'cold starts', referring to newer users who haven't rated enough movies to generate recommendations.**

I. Project Description

This project aims to develop a system which can recommend movies to users based on existing ratings the user has already given, along with ratings provided by other similar users on similar movies. This is a Massive Algorithmics project. This system can be useful because it helps users to acquire more usable information without having to deal with massive amounts of data. Barring unforeseen circumstances, it should be feasible enough to implement within the semester. The main problems we expect to face are mainly about the time required to process 1 million ratings from our dataset. It will be challenging to come up with an algorithm to accomplish such a task. Intensive study on the various types of algorithms has already begun, as this is the most crucial component of the project.

The project has four stages: Gathering, Design, Infrastructure Implementation, and User Interface.

A. Stage1 - The Requirement Gathering Stage.

- The general system description: The goal of the project is to provide movie recommendations to users. To accomplish this, the system uses the ratings given by the user to other movies, together with the ratings given by other similar users on similar movies. Demographic information can be also be used for 'cold starts', referring to newer users who haven't rated enough movies to generate recommendations.
- The three types of users (grouped by their data access/update rights):
  - System administrators: who add or delete movies.
    Access Rights: Add/Update/Delete movies.
  - Existing users: As they have rated many movies,we know their appetite (based on either content-based clustering or collaborative filtering).
    Access Rights: Add/Update their movie rating and personal information.
  - New users: we know their age, gender, etc. (cold start problem, based on demographic-based recommendation).
    Access Rights: Add/Update their movie rating and personal information.
- The user's interaction modes:
  - System administrators: view(movie rating statistics); edit (add or delete movie database).
  - Existing users: view (average movie ratings and their personal movie ratings); edit (personal information and personal movie ratings).
  - New users: view (average movie ratings and their personal movie ratings); edit (personal information and personal movie ratings).
- The real world scenarios:
  - Scenario1 description: The users interact with the application. Movies are recommended based on their movie rating.
  - System Data Input for Scenario1: Movie ratings.
  - Input Data Types for Scenario1: Float (Rating from 0 to 5 in 0.5 increments).
  - System Data Output for Scenario1: Recommended movies.
  - Output Data Types for Scenario1: String (Names of recommended movies).
  - Scenario2 description: System administrator requests movie statistics based on movie names.
  - System Data Input for Scenario2: Name of the movie.
  - Input Data Types for Scenario2: String.
  - System Data Output for Scenario2: User rating statistics.
  - Output Data Types for Scenario2: Float and String.
- Project Time line and Divison of Labor.
  Vivek Harilal Bapodra (vhb12): database manipulation, testing. Vedika Babu (vb305): algorithm design, evaluation. Yi Hou (yh523): interface, documentation.
  Project timeline: one week Design phase, three weeks Implementation phase, one week UI phase.

B. Stage2 - The Design Stage.

- Short Textual Project Description.
  Given a set of users U, set of movies M, the corresponding ratings R, predict the rating P a user would assign to a movie which has not been rated by the user. We use a Root Mean Square method to calculate the error in prediction.

$$RMSE = \sqrt{\frac{1}{|S_{test}|} \sum_{(m,u) \in S_{test}} (R_{m,u} - P_{m,u})^2} \quad (1)$$

The dataset will be partitioned into a test set and a training set. A UserID is given as an input. This input can come from the test dataset, or can be entered manually. The system then performs a cold start check. This is to ensure that the user has watched a certain number of movies to generate meaningful recommendations. If it is a cold start, i.e., the user has not watched enough movies, then in that case information like gender, age and occupation will be used to find the closest users which will be then used to generate recommendations. If it is a 'regular' user, then the k-most similar movies to the movies already watched by the user will be computed. Using this, the system predicts a rating P for these movies.

Before testing, a simple preprocessing will be done to find the optimal *k* value. Considering the following system flow and pseudo code of this project, the over all running time is O(|M||M||U|), with |M| representing the size of movie set and |U| representing the size of user set. A dictionary is used to store the ratings, users, movies datasets, and store *k*-nearest neighbours to each movie and some users, making the space complexity be O(|M||M|+|U||U|), O(|M||M|) for the storage of movie similarity and O(|U||U|) for the storage of user similarity.

- Flow Diagram.

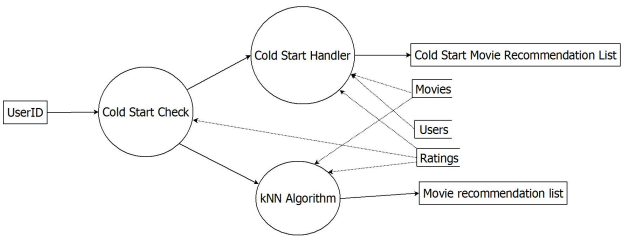

Fig. 1. System Flow Diagram

- High Level Pseudo Code System Description.

For training purposes, the dataset will be partitioned into two sets - a training set(P) and a test set(Q). High-level pseudo-code for the kNN algorithm:

```
for all Users i in Q
    if (New_User_Check(u_i) is FALSE)
        for all Movies j : (i,j,r) in P
            for all Movies l \in M
                calculate m_similarity(j,l)
            S=Set of k closest movies to j
            for all Movies m in S
                calculate P(m,i)
    else
        for each user a in U
            calculate u_similarity(i,a)
        S = set of all movies watched by
               k closest users to i
        for each movie m in S
            calculate P(m,i)
Calculate RMSE
```

```
New_User_Check(u_i)
    movie_count = 0
    for all Movies m_j : (u_i,j,r) in P
        movie_count = movie_count +1
    if(movie_count<threshold_movie_count)
        return TRUE
    return FALSE
```

- Algorithms and Data Structures.
  - kNN Algorithm: We use an adjusted cosine similarity to calculate the similarity between two movies M_a and M_b.

$$sim(a,b)=\frac{\sum_{u\in U(a)\cap U(b)}(R_{a,u}-\overline{R_u})(R_{b,u}-\overline{R_u})}{\sqrt{\sum_{u\in U(a)\cap U(b)}(R_{a,u}-\overline{R_u})^2\sum_{u\in U(a)\cap U(b)}(R_{b,u}-\overline{R_u})^2}} \quad (2)$$

where $R_{a,u}$ is User *u*'s rating on Movie a, $\overline{R}_u$ is User *u*'s average tating, U_a is the set of users that have rated Movie a. The advantage of the above-defined adjusted cosine similarity over standard similarity is that the differences in the rating scale between different users are taken into consideration. We find the *k*-nearest neighbours to the movies already watched by the user. We then calculate the predicted ratings based on the following formula.

$$P_{m,u}=\frac{\sum_{j\in N_u^K(m)} sim(m,j)R_{j,u}}{\sum_{j\in N_u^K(m)}|sim(m,j)|} \quad (3)$$

where $N^K{}_u$(m) = { *j* : *j* belongs to the *K* most similar movies to Movie m and User *u* has rated *j* }, and $P_{m,u}$ is the prediction of ratings (of User *u* on Movie *j*). We then find the top m movies sorted by predicted rating, and present them to the user [**?**].
  - Cold Start Check: This will be done by checking if the user has rated a minimum number of movies .
  - Cold Start Handler: Each user has three features, i.e. age, gender, and occupation. The similarity between two users should be determined by the score, which represent the similarity, of those features. For the feature age, score has positive correlation with the difference between ages of two users and is confined to [0,1]. For gender and occupation, score is 1 if the features of two users are the same, otherwise, score is 0. Then the similarity between two users equals to the summation of weighted score of each feature:

$$sim(u1,u2)=w_1s_1+w_2s_2+w_3s_3 \quad (4)$$

where w_i is the weight of feature *i*, and s_i is the scores. Calculate the similarity between this new user and all other users and find the closest *k* neighbouring users for him. The predicted ratings of

this new user u on the movie m is calculated using the following formula:

$$P_{m,u} = \frac{\sum_{i \in N_m^K(u)} sim(u,i) R_{m,i}}{\sum_{i \in N_m^K(u)} |sim(u,i)|} \quad (5)$$

where $N^K{}_m(u) = \{i$: User i belongs to the K most similar users to User u and User i has rated movie m $\}$, and $P_{m,u}$ is the prediction of ratings (of this new User u on Movie m). We then find the top m movies sorted by predicted rating, and present them to the user

- Data structures used: A dictionary is used to store the ratings, users, movies datasets, and store k-nearest neighbours to each movie and user. This is done to make it easier to fetch an entry. The final recommendation list will be a list of (Movie, Predicted rating) ordered pairs.

*The rating data sets used in this project is collected by the GroupLens Research Project at the University of Minnesota [**?**].

- Flow Diagram Major Constraints.
  - Primary Key Constraint: Column movie id in Table Movies. This is the id given to every movie available in the database.This has to be unique.
  - Primary Key Constraint: Column user id in Table Users. This is the id given to every user available in the database.This has to be unique.
  - Unique Key Constraint: Column movie id, User id in Table Ratings. This is the rating given by an user for a movie .This has to be unique.A single user can?t give 2 ratings for a single movie.
  - Foreign Key Constraint: Column movie id in Table Ratings [Primary Key: Column movie id in Table Movies]. Movie id is the foreign key from the table Table movies [column movie id].
  - Foreign Key Constraint: Column user id in Table Ratings [Primary Key: Column user id in Table Users]. User id is the foreign key from the table Table movies [column user id].

C. Stage3 - The Implementation Stage.

We use postgresql to manipulate the database and Python 2.7 to implement the recommend system.

- Sample small data snippet.
  - Rating information:(user id, movie id, rating, timestamp)
    Example:
    354 1063 3 891218230
    551 471 5 892783365
    894 535 4 879896920
    943 1044 3 888639903
  - User information:(user id, age, gender, occupation, zipcode)
    Example:
    354 29 F librarian 48197
    551 25 M programmer 55414
    894 47 M educator 74075
    943 22 M student 77841
  - Movie information:(movie id, movie title, release date, video release date, IMDb URL, movie genre)
    Example:
    1, Toy Story, (1995), 01-Jan-1995, http://us.imdb.com/M/title-exact?Toy%20Story%20(1995) 0001110000000000000
    2, GoldenEye (1995) 01-Jan-1995 http://us.imdb.com/M/title-exact?GoldenEye%20(1995) 0110000000000000100
    3, Four Rooms (1995) 01-Jan-1995 http://us.imdb.com/M/title-exact?Four%20Rooms%20(1995) 0000000000000000100
    4, Get Shorty (1995) 01-Jan-1995 http://us.imdb.com/M/title-exact?Get%20Shorty%20(1995) 0100010010000000000

- Sample small output
  - Predicted rating: for each record of rating information in test dataset, we give the predicted rating. Then RMSE of predicted rating and actual rating is used to examine this recommendation system.
    Example: movie id, user id, rating, predicted rating
    276 210 5 4.2303
    257 214 3 3.7678
    554 217 3 2.5500
    1134 221 4 4.0125
    339 222 4 2.5415
    The overall RMSE equals to 1.01499
  - Recommendation list
    Example: Recommendation list for user 2:

```
THE TOP 50 RECOMMENDED MOVIES
1606 Deceiver (1997)
119 Maya Lin: A Strong Clear Vision (1994)
373 Judge Dredd (1995)
850 Perfect Candidate, A (1996)
1408 Gordy (1995)
1253 Tie That Binds, The (1995)
850 Perfect Candidate, A (1996)
1288 Denise Calls Up (1995)
1288 Denise Calls Up (1995)
1484 Jerky Boys, The (1994)
1606 Deceiver (1997)
1510 Mad Dog Time (1996)
119 Maya Lin: A Strong Clear Vision (1994)
1528 Nowhere (1997)
1528 Nowhere (1997)
1081 Curdled (1996)
1293 Star Kid (1997)
1293 Star Kid (1997)
337 House of Yes, The (1997)
1122 They Made Me a Criminal (1939)
1408 Gordy (1995)
1606 Deceiver (1997)
1389 Mondo (1996)
1253 Tie That Binds, The (1995)
1314 Surviving the Game (1994)
1389 Mondo (1996)
1283 Out to Sea (1997)
1087 Bloodsport 2 (1995)
1662 Rough Magic (1995)
1453 Angel on My Shoulder (1946)
359 Assignment, The (1997)
1116 Mark of Zorro, The (1940)
```

Fig. 2. Recommendation list or user 2

- Working code

Fig. 3. Code 1

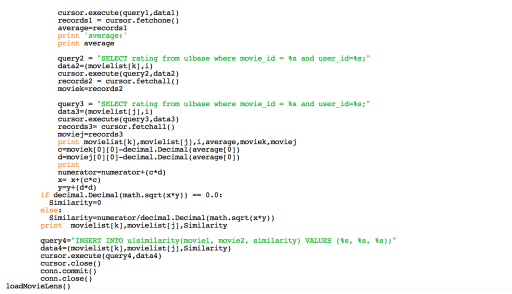

Fig. 4. Code 2

Fig. 5. Code 3

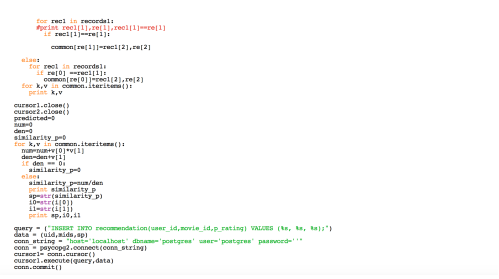

Fig. 6. Code 4

- Demo and sample findings
  - Data size: all the dataset resides on the disk and the file size is 1933kB;
  - Increasing value of K improves accuracy but increases time of execution as well.

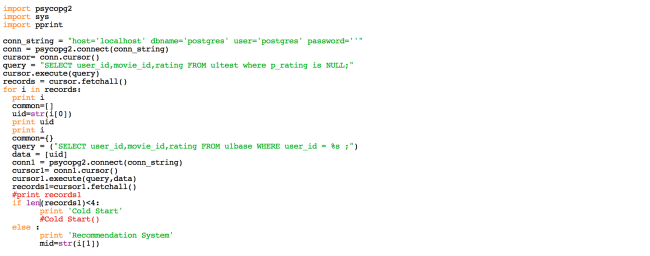

Fig. 7. Code 5

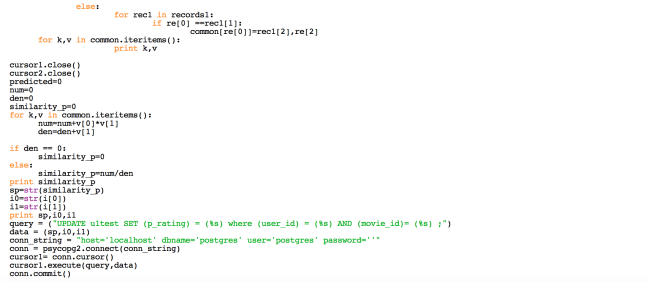

Fig. 8. Code 6

D. Stage4 - User Interface.

- Theoretically, when a user logs into the movie recommendation system, we fetch his/her user id, then the system automatically do the calculation and list top 50 movies. But in practice, we need to input user id manually into the input textbox on the left, and recommended movies show up in the textbox on the right. Figure 9 shows the User Interface and Figure 10 shows the User Interface with input and output.

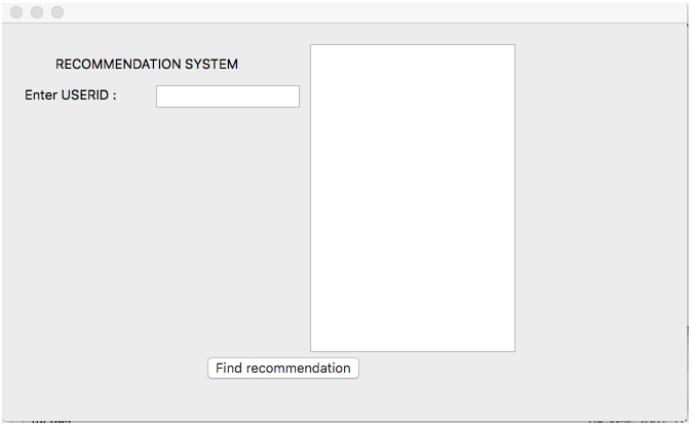

Fig. 9. User Interface

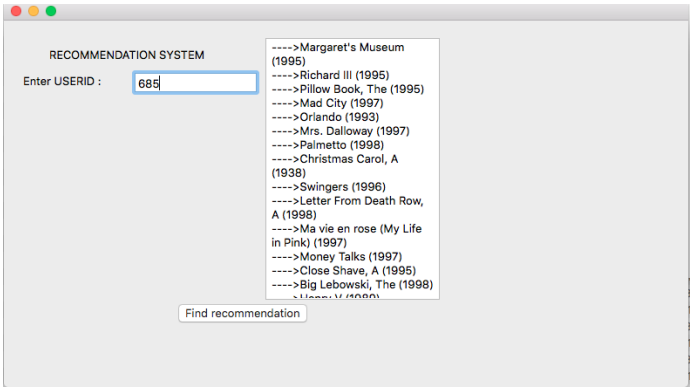

Fig. 10. User Interface with outcome

- The error messages popping-up when users access and/or updates are denied (along with explanations and examples):
  - The error message: When a new user without any features comes, we can not recommend proper movies to him. Figure 11 show the error message if we in put a user id that is not in the database.

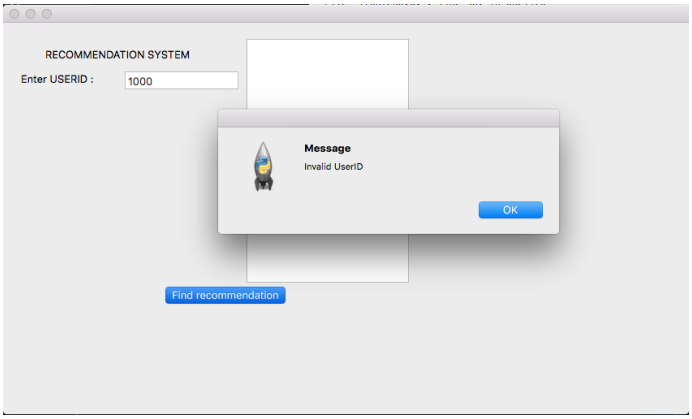

Fig. 11. Error Message

  - The error message explanation (upon which violation it takes place): Recall that when dealing with cold start, we use demographic information, such as user's age, gender and occupation, to make recommendation according to his/her similar users' favorite movies. If a new user dose not have any demographic information, we can not perform the cold start procedure.
  - The error message example according to user(s) scenario(s): The user id existing in database ranges between 1 and 943, if we input user id as 1000, there will be an error message. One possible solution to handle this problem is that we just recommend movies with highest average ratings, that is the most popular movers.
- The information messages or results that pop-up in response to user interface events.
  - The information message: Recommended movies for the specific users are presented.
  - The information message explanation and the corresponding event trigger: Each time there is a user connecting our server, we fetch the user's id and make corresponding recommendation to him/her.
- The interface mechanisms that activate different views.
  - The interface mechanism: The small textbox on the left serves the input, and the larger textbox on the right servers the output.

II. Project Highlights.

  - Only working applications will be acceptable at project completion. A running demo shoul be presented to your project advisor at a date to be specified after the second midterm. A version of your application shall be installed in a machine to be specifed later during the semester. Your final submissiom package will also include a final LaTeX report modeled after this document, as well as a Power Point Presentation.
  - The presentation (7 to 8 minutes) should include at least the following items (The order of the slides is important):
    1) Title: Project Names (authors and affiliations)
    2) Project Goal
    3) Outline of the presentation
    4) Description
    5) Pictures are essential. Please include Interface snapshots exemplyfing tthe different modes of users's interaction.
    6) Project Stumbling Blocks
    7) Data collection, Flow Diagram, Integrity Constraints
    8) Sample Findings
    9) Future Extensions
    10) Acknowledgements
    11) References and Resources used(libraries, languages, web resources)
    12) Demo(3 minutes)

    Please follow the sample presentation mock up that is posted on Sakai.
  - By Dec 1 your group should have completed the final submission. This includes a presentation (7 to 8 minutes) to your project advisor as well as a convincing demo of your project functionalities (3 minutes): every group member should attend the demo (and presentation) indicating clearly and specifically his/her contribution to the project. This wil allow us to evaluate all students in a consistent and fair manner.
  - Thank you, and best of luck!

REFERENCES

[1] Wen Z. Recommendation System Based on Collaborative Filtering[R]. Technical Report, CS229, Stanford University, USA, 2008.

[2] F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. ACM Transactions on Interactive Intelligent Systems (TiiS) 5, 4, Article 19 (December 2015), 19 pages. DOI=http://dx.doi.org/10.1145/2827872