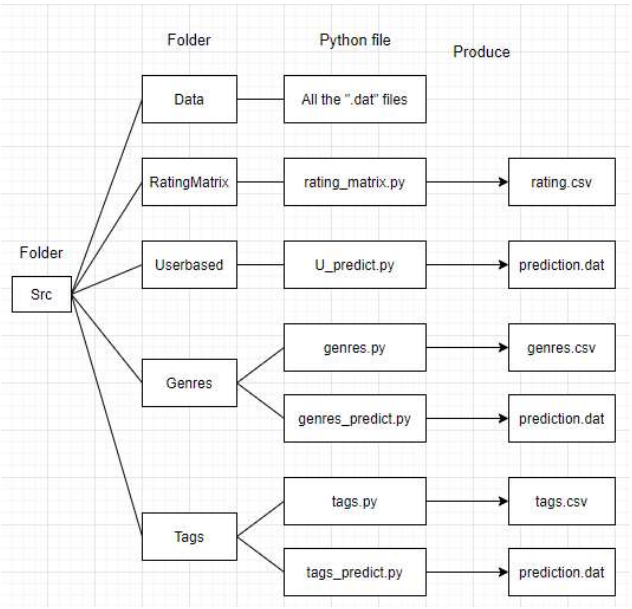
Miner2 username : Bear
Mason user id : chsiung2
Gnumber : G01272835
Best public score : 0.91

1. Introduction

- I only use “movie_genres.dat”、“movie_tags.dat” as my additional content.
- The following picture is the structure of my src folder :
  - Data folder : Contain all the “.dat” files.
  - RatingMatrix folder : Process the “train.dat” to get rating matrix “rating.csv”.
  - Userbased folder : Use user-based nearest-neighbor with rating matrix “rating.csv” to predict.
  - Genres folder : Process the “movie_genres.dat” to get additional content “genres.csv” and then use this content to predict.
  - Tags folder : Process the “movie_tags.dat” to get additional content “tags.csv” and then use this content to predict.



2. Approaches

- Handling data :
  - Use pandas and dataframe to store the content.
  - Use movie_id as row.
  - In “rating_matrix.py” : Use user_id as column and process the “train.dat” to get rating matrix.
  - In “genres.py” : Use genre as column and process the “movie_genres.dat”. Regard the missing values as 0.
  - In “tags.py” : Use tag as column and process the “movie_tags.dat”. Regard the missing values as 0.
- I checked the “train.dat” and “test.dat” beforehand. There is no cold start user problem. Therefore,

I don’t do the check in my code.

- In “U_predict.py” : I use “NearestNeighbors” in sklearn and do user-based nearest-neighbor. Use whole rating matrix to “fit()”. After finding k neighbors, ignore the missing values and simply average other ratings of a movie. Use it as prediction. If all the ratings of k neighbors are missing, predicted value is 0.
- In “genres_predict.py” : I use “svm” in sklearn and do regression with “genres.csv”. Use genres matrix as training data and the column of specific user_id in rating matrix as label to do “svm.fit()”. View all missing values as 0.
- In “tags_predict.py” : I use “NearestNeighbors” in sklearn and do nearest-neighbor with “genres.csv” according to their tags. Use tags matrix to “fit()”. After finding k neighbors, ignore the missing values and simply average other ratings of a movie. Use it as prediction. If all the ratings of k neighbors are missing, predicted value is 0.

3. Experimental Results

- “U_predict.py” : 2.76
- “genres_predict.py” : 0.91
- “tags_predict.py” : 2.57

4. Conclusion

- “U_predict.py”, “tags_predict.py” :
  - “rating.csv”, “tags.csv” are so large and sparse that they took few hours to predict the ratings.
  - Because I just simply average the ratings of nearest neighbors, the RMSE score are high. Sometimes, all the ratings of neighbors are missing, and it leads to the incorrect prediction : 0. Use the prediction function in lecture note would be much better.
- “genres_predict.py” :
  - “genres.csv” is denser(only 20 columns), so it also runs faster.
  - Because the affection of missing ratings is less than user-based nearest-neighbor (Most of rating matrix are missing compared to only one columns of rating matrix used for each test data in “genres_predict.py”), its prediction is the best in my codes.
- I don’t use “movie_directors” and “movie_actors”. Directors and actors are various. I think the information will generate a sparse matrix like tags. Genres are more straightforward to implement.