**Web Analytics : Workshop 2 (Movielens Recommendation)**

Submitted By :

Aarcha JS (A0120053A)

Aditya Shankar (A0119938N)

Jayant Kumar (A0119993M)

Jing Sun (A0120104)

Ketki Ranade (A0119950B)

Prateek Singhal (A0119978H)

# Business Goal

Determine top 3 movies to recommend to a user. (Movies that the user has not seen)

# Introduction

The goal of the workshop is to predict the top 3 movies to each of the user in the given dataset.

Apache Mahout (Machine Learning and Data Mining API in java) and recommenderLab in R are used to perform analysis on the given dataset.

# Data Preparation

Data used for the purpose of study is (100Kfile -> ml-100k) ua.base and ua.test. “ua.base” contains the data of 943 user’s ratings over various movies. “ua.test” contains the data for testing purpose. Also “u\_item.txt” contains the movie names.

For the analysis purpose we have removed “datetime” field from “ua.base” and “ua.test”. We have realized that date & time won’t play important role in our analysis, hence we have removed it.

For use with recommender lab, we have merged the base and test data and we are doing a training and testing split using recommender lab’s evaluation scheme.

# Model Build

For the model building purpose we have used Mahout (Machine Learning and Data Mining API for JAVA).

For the model building purpose **Nearest Neighbor Collaborative Filtering technique** is used.

**No. of Nearest Neighbors selected for weighted averages** is 200.

**Input Parameters** are “User Id”, “Movie”, and “Rating”.

**Output Parameters** are “Top 3 predicted movies” and “Estimated Rating for Top 3 Predicted Movies”.

For recommender lab in R, we have built 4 types of evaluation scheme. User Based filtering with Jaccard and Cosine distance and Item based filtering with Jaccard and Cosine distance. The number of nearest neighbors is 30 for IBCF and 25 for UBCF.

# Test Regime

Our model is trained on “ua.base” dataset and tested on “ua.test” dataset.

**Test Case 1.**

For model testing purpose we have used the ratings from the ua.test dataset.

1. We have used our trained model to predict ratings for all unwatched movies by each users in the training set.
2. Then we have compared the predicted ratings for the unwatched movies with the movie ratings given in the test dataset.
3. Further we have calculated the Average Mean Absolute Error and confusion matrix. (Explained in next section)

**Test Case 2.**

In this testing approach we have merged our training and testing datasets. Randomly 70% (for each user) of the combined dataset is selected as training data and remaining 30% is used for testing purpose.

Avg. Mean Absolute Error: 0.8124667197414267(Using K-NN)

Root Mean Square Error: 1.027866232824244(Using K-NN)

For recommender lab in R, following are the test results with the testing set built by given 5 scheme and 90% training and 10% testing split.

|  |
| --- |
| > error  RMSE MSE MAE  UBCF\_Jaccard 39.54712 1563.9750 9.452474  UBCF\_Cosine 45.05876 2030.2918 9.623266  IBCF\_Jaccard 80.79609 6528.0089 13.147987  IBCF\_Cosine 17.82343 317.6748 1.510242 |
|  |
| |  | | --- | | > | |

We find that IBCF with cosine distance is the least error recommender and use it to give our predictions.

# Model Test Results

1. **Compute Predicted Rating Error**

For model evaluation purpose we have calculated Average Mean Absolute Error.

i.e.

**Avg. Mean Abs. Error** =

Our model has performed very well and we have achieved a very low **Avg. Mean Abs. Error** of **0.82467.( using KNN and Test Case 1)**

1. **Deriving Confusion Matrix**

Also we have calculated the confusion matrix (5x5) for validating the accuracy of recommendations made by our model. For comparing the actual versus predicted rating we have rounded our predicted ratings and confusion matrix is drawn for them.

Table 1 Confusion Matrix using KNN

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Predicted(1)** | **Predicted(2)** | **Predicted(3)** | **Predicted(4)** | **Predicted(5)** |
| **Actual(1)** | 14 | 13 | 8 | 10 | 5 |
| **Actual(2)** | 87 | 96 | 123 | 76 | 19 |
| **Actual(3)** | 263 | 509 | 1030 | 1008 | 402 |
| **Actual(4)** | 137 | 319 | 1178 | 2108 | 1552 |
| **Actual(5)** | 6 | 14 | 19 | 59 | 136 |

# Performance of Modeling Method

We have performed K-Nearest Neighbor collaborative filtering and matrix factorization approaches.

We have understood that K-nearest neighbor is performing better in our case since it is giving less

Avg. Mean Absolute Error as compared to Matrix Factorization approach.

**K-Nearest Neighbor** (Avg. Mean Absolute Error = 0.82467)

**Matrix Factorization** (Avg. Mean Absolute Error = 0.83229)

Hence, we recommend K-Nearest Neighbor using collaborative filtering modeling approach since K-Nearest Neighbor performed better over other approaches.

# Business Goal Performance

Our model has performed very well on the given dataset, since our dataset is very sparse still we have achieved low Average Mean Absolute Error of **0.82467.**

We are successful in recommending top 3 movies to the user hence we have achieved our business goal.

He have used K Nearest Neighbor (KNN) Collaborative Filtering method in which we have performed user-based collaborative filtering.

Our Test Case 1 and Test Case 2 suggest that our recommender system is very stable as its **Avg. Mean Abs. Error** is nearly equal in both the test cases.

Hence, we can confidently recommend our model for recommending top 3 movies for each user.

Here is a sample prediction for a user using recommender lab (User Ids 10 to 12)

> as(recom,"list")

[[1]]

[1] "1636" "1645" "1650"

[[2]]

[1] "1636" "1645" "1650"

[[3]]

[1] "1398" "1536" "1636"

# Attachments

* Java Project – **Recommender**: Project can be imported in the eclipse and following classes can be executed to replicate the results.
  1. MatrixFactorizationRecommender.java
  2. RecommederEvaluationTestCase1WithKNN. Java
  3. RecommederEvaluationTestCase1WithMatrixFActorization.java
  4. RecommenderEvaluationTestCase2WithKNN.java
  5. Top3PredictionsWithKNN.java
* **Prediction Error and Confusion Matrix.xlsx** - Shows the Prediction Error and Confusion Matrix for KNN.
* **TestingResults(Actual Rating vs Predicted Rating).csv**
* **Top3MoviesFor Recommendation with Rating.xlsx : Top 3 Predictions**
* **Reco.r :** Recommender Lab evaluation and prediction.