Inf553 – Foundations and Applications of Data Mining

Fall 2018 Assignment 2
Recommendation System

Deadline: 10/12 2018 11:59 PM PST

1 Overview of the Assignment

This assignment contains two parts. First, you will implement a Model-based Collaborating Filtering(CF) recommendation system using Spark MLlib. Second, you will implement either a User-based CF system or Item-based CF system without using a library. The dataset you are going to use are the Yelp challenge dataset. The task sections below will explain the assignment instructions in detail. The goal of the assignment is to make you understand how different types of recommendation systems work and more importantly, try to find a way to improve the accuracy of the recommendation system yourself.

Environment Requirements

Python: 2.7 Scala: 2.11 Spark: 2.3.1

Student can use Python or Scala to complete both Task1 and Task2.

There will be 10% bonus if you use Scala for both Task1 and Task2 (i.e. 10 - 11; 9 - 10).

IMPORTANT: We will use these versions to compile and test your code. If you use other versions, there will be a 20% penalty since we will not be able to grade it automatically.

You can only use Spark RDD.

In order for you to understand more deeply of the Spark, use RDD only, you won't get any point if you use Dataframe or Dataset.

Write your own code!

For this assignment to be an effective learning experience, you must write your own code! I emphasize this point because you will be able to find Python implementations of most or perhaps even all of the required functions on the web. Please do not look for or at any such code!

TA will combine some python code on Github which can be searched by keyword ”INF553” and every students’ code, using some software tool for detecting Plagiarism.

Do not share code with other students in the class!!

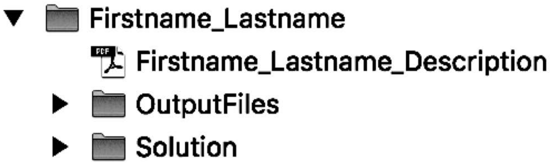
Submission Details

For this assignment you will need to turn in a Python or Scala program depending on your language of preference. This assignment will surely need some time to be implemented so please plan accordingly and start early!

Your submission must be a .zip file with name: **Firstname_Lastname_hw2.zip**. The structure of your submission should be identical as shown below.

The Firstname_Lastname_Description.pdf file contains helpful instructions on how to run your code along with other necessary information as described in the following sections.

The OutputFiles directory contains the deliverable output files for each problem and the Solution directory contains your source code.



▼ Firstname_Lastname
Firstname_Lastname_Description
▶ OutputFiles
▶ Solution

Figure 1: Submission Structure

Data

In this assignment, we will use the yelp challenge dataset, the ”yelp challenge dataset” can be download from this link: Yelp Challenge. In order to download the dataset, you need to use your email to sign up individually in the Yelp challenge website. Detailed introduction of the data can also be found through the link, in the document tab. After download and unzip the data, the dataset contain 6 .json file and two .pdf file. In this assignment, the reviews.json file and three columns in the review file will be used: user_id, business_id, stars.

The yelp dataset contains more than 6 million review record between millons of the user and business. Because the huge volume and the sparseness between the user and business, the recommendation system will take a lot of computation, we extract the subset of the whole dataset so that the assignment can end on a reasonable time for every students’ laptop. In order to finish this assignment, you only need the two data file in the Data/ folder.

We recommend you download the whole dataset as the playground of the data mining or any other area.

Yelp Dataset Description

yelp_academic_dataset_business.json : 188,593 records
Attributes: Business ID, address, name, city, Business hours, Categories, rating and reviews_count
yelp_academic_dataset_review.json : 5,996,996 records
Attributes: review ID, user ID, business ID, rating, comments
yelp_academic_dataset_user.json : 1,518,169 records
Attributes: user ID, name, review_count, Yelp_join_date
yelp_academic_dataset_checkin.json : 157,075 records
Attributes: Business ID, time
yelp_academic_dataset_tip.json : 1,185,348 records
Attributes: user ID, Business ID, text, likes, date
yelp_academic_dataset_photo.json : 280,992 records
Attributes: photo ID, Business ID, text

Dataset for Assignment

In this assignment, we extract the subset of the whole dataset contains 452353 reviews between 30,000 user and 30,000 business and split them to train data (90%) and test data (10%). you can get two files in the Data/: train_review.csv and test_review.csv, each file contain three conlumns: user_id, business_id, and stars. And we will use these two files to finish and test our recommendation system.

Task of Recomendation System

The task of this the recommendation system is to use the records in the train.csv to **predict** the stars for users and businesses in the test.csv. Then, you need to use the stars in testing data as the ground truth to evaluate the accuracy of your recommendation system.

Example: Assuming train.csv contains 1 million records and the test.csv contains two records: (12345, 2, 3) and (12345, 13, 4). You will use the records in the train.csv to train a recommendation system (1 million). Finally, given the user_id 12345 and business_id 2 and 13, your system should produce rating predictions as close as 3 and 4, respectively.

2 Task1: Model-based CF Algorithms (30 Points)

In task1, you are required to implement a Model-based CF recommendation system by using Spark MLlib. You can learn more about Spark MLlib by this link: MLlib

You are going to predict the testing datasets mentioned above. In your code, you can set the parameters yourself to reach a better performance. You can make any improvement to your recommendation system: **speed**, **accuracy**.

After achieving the prediction for ratings, you need to compare your result to the correspond ground truth and **compute the absolute differences**. You need to divide the absolute differences into 5 levels and count the number of your prediction for each level as following:

>=0 and <1: 12345 (there are 12345 predictions with a <1 difference from the ground truth)
>=1 and <2: 123
>=2 and <3: 1234
>=3 and <4: 1234
>=4: 12

Additionally, you need to compute the RMSE (Root Mean Squared Error) by using following formula:

$$RMSE = \sqrt{\frac{1}{n}\sum (Pred_i - Rate_i)^2}$$

Where $Pred_i$ is the prediction stars for business i, $Rate_i$ is the true stars for business i, n is the total number of the review. Read the Microsoft paper mentioned in class to know more about how to use RMSE for evaluating your recommendation system.

Tips: For model-based CF, you may need to index the user_id and business_id to integer.

BaseLine

After implementing the model-based CF, you can try to change the parameters of the model and see the change of the RMSE of the recommendation. You need to find the parameters that can beat the baseline to get the full grade.

Here is the baseline of the Model-Based CF:

$$RMSE = 1.08$$

3 User-based or Item-based CF Algorithm (70 Points)

In this part, you are required to implement a User-based CF or Item-based recommendation system with Spark. For the detail of the User-Based CF and the Item-Based CF, you can find it from the slides of the lecture or from many tutorial from the Internet.

You are going to predict for the testing datasets mentioned above. Based on the User-based or Item-based CF, you can make any improvement to your recommendation system: **speed**, **accuracy** (e.g., Hybird approaches). It's your time to design the recommendation system yourself, but first you need to beat the baseline.

After achieving the prediction for ratings, you need to compute the accuracy in the same way mentioned in Model-based CF.

Result Format

1. Save the predication results in a text file. The result is ordered by **user_id** and **business_Id** in ascending order.

Example Format:
$user_1$, $business_2$, $prediction_{12}$
$user_1$, $business_3$, $prediction_{13}$
...
$user_n$, $business_k$, $prediction_{nk}$

2. **Print the accuracy information** in terminal, and **copy this value** in your description file.

```
>=0 and <1: 12345
>=1 and <2: 123
>=2 and <3: 1234
>=3 and <4: 1234
>=4: 12
RMSE: 1.23456789
Time: 123 sec
```

Baseline & Time Threshold

Same as the Model-Based CF, in order to get the full point of the grade, you need to beat the baseline first. And this task has the time threshold, make sure that your program can give the result within a reasonable time.

You can use any method (based on user-based or item-based CF) to improve the performance of your recommendation system, for example, you can find someway to refine the result from the User-Based or Item-Based CF, or combine the result from User-Based and Item-Based CF.

$$RMSE = 1.11$$

$$Time\ Threshold: \ 450\ Second$$

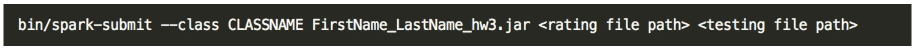
Execution Example

The first argument passed to our program (in the below execution) is the training csv file. The second input is the testing csv file. Following we present examples of how you can run your program with spark-submit both when your application is a Java/Scala program or a Python script.

Example of running application with spark-submit

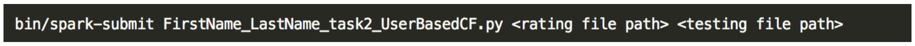
Notice that the argument class of the spark-submit specifies the main class of your application and it is followed by the jar file of the application.

Please use ModelBasedCF, UserBasedCF, ItemBasedCF as class name



```
bin/spark-submit --class CLASSNAME FirstName_LastName_hw3.jar <rating file path> <testing file path>
```

Figure 2: CF: Command Line Format for Scala



```
bin/spark-submit FirstName_LastName_task2_UserBasedCF.py <rating file path> <testing file path>
```

Figure 3: CF: Command Line Format for python

You don't need to specify the path of the output file in the commandline, you only need to save the file with the name format Firstname_Lastname_XXXXBasedCF.txt. in the same path your program run (Relative Path).

Description File

Please include the following content in your description file:

1. Mention the Spark version and Python version
2. Describe how to run your program for both tasks
3. Same baseline table as mentioned in task 1 to record your accuracy and run time of programs in task 2
4. If you make any improvement in your recommendation system, please also describe it in your description file.

Submission Details

Your submission must be a .zip file with name: Firstname_Lastname_hw2.zip
Please include all the files in the right directory as following:
1. A description file: Firstname_Lastname_desription.pdf
2. All Scala scripts:
Firstname_Lastname_ModelBasedCF.scala
Firstname_Lastname_UserBasedCF.scala
Firstname_Lastname_ItemBasedCF.scala

3. A jar package for all Scala file: Firstname_Lastname_hw2.jar
If you use Scala, please make all *.scala file into ONLY ONE Firstname_Lastname_hw2.jar file and strictly follow the class name mentioned above. And DO NOT include any data or unrelated libraries into your jar.
4. If you use Python, then all python scripts:
Firstname_Lastname_ModelBasedCF.py
Firstname_Lastname_UserBasedCF.py
Firstname_Lastname_ItemBasedCF.py
5. Required result files for task1 & 2:
Firstname_Lastname_ModelBasedCF.txt
Firstname_Lastname_UserBasedCF.txt
Firstname_Lastname_ItemBasedCF.txt

Grading Criteria

1. If your programs cannot be run with the commands you provide, your submission will be graded based on the result files you submit and 80% penalty for it.
2. If the files generated by your programs are not sorted based on the specifications, there will be 20% penalty.
3. If your program generates more than one file, there will be 20% penalty.
4. If you don't provide the source code and just the .jar file in case of a Java/Scala application there will be 80% penalty.
5. If the running time or RMSE is greater than the base line, there'll be 20% penalty for each part as mentioned above.
6. If your prediction result files miss any records, there will be 30% penalty
7. There will be 20% penalty for late submission within a week and 0 grade after a week.
8. You can use your free 5-day extension.
9. There will be 10% bonus if you use both Scala for the entire assignment.
10. **There will 0 grade if you use Dataframe or Dataset.**