

# Project: Movie Recommendation with MLlib - Collaborative

## Filtering (implementaiton 2)

[TianzeKang2000/Movie-Recommendation-System \(github.com\)](https://github.com/TianzeKang2000/Movie-Recommendation-System)

Step 1: Convert [MoveLens' data](#) (UserID, MovieID, rating, Timestamp) into the [format](#) of (UserID, MovieID, rating)

```
# Load the data file and convert it to the required format
```

```
input_file_path = '/mnt/data/New data.txt'
```

```
output_file_path = '/mnt/data/formatted_data.txt'
```

```
# Read the input file
```

```
with open(input_file_path, 'r') as file:
```

```
    data = file.readlines()
```

```
# Process the data
```

```
formatted_data = []
```

```
for line in data:
```

```
    parts = line.split()
```

```
    if len(parts) == 4:
```

```
        formatted_data.append(f'{parts[0]},{parts[1]},{parts[2]}\n')
```

```
# Write the formatted data to a new file
```

```
with open(output_file_path, 'w') as file:
```

```
    file.writelines(formatted_data)
```

```
output_file_path
```

```
196 242 3 881250949
186 302 3 891717742
22 377 1 878887116
244 51 2 880606923
166 346 1 886397596
298 474 4 884182806
115 265 2 881171488
253 465 5 891628467
305 451 3 886324817
6 86 3 883603013
62 257 2 879372434
286 1014 5 879781125
200 222 5 876042340
210 40 3 891035994
224 29 3 888104457
303 785 3 879485318
122 387 5 879270459
194 274 2 879539794
291 1042 4 874834944
234 1184 2 892079237
119 392 4 886176814
167 486 4 892738452
299 144 4 877881320
291 118 2 874833878
308 1 4 887736532
95 546 2 879106556
```

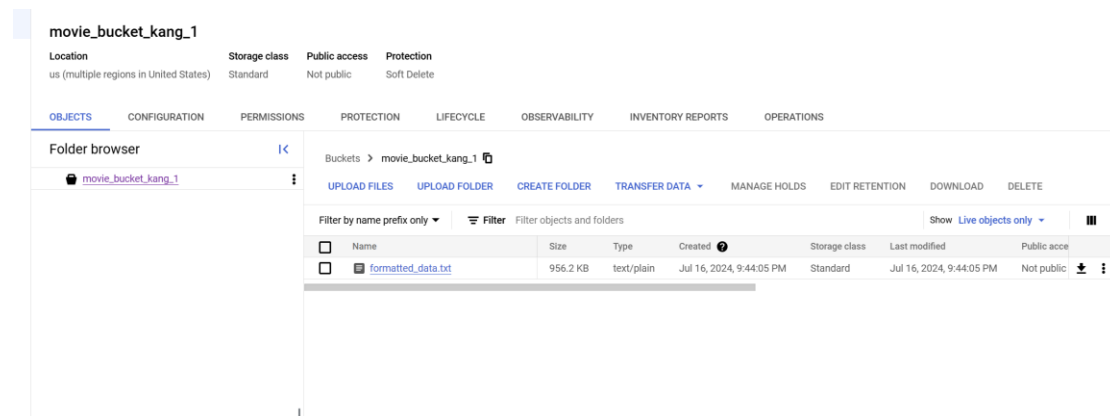
```
196,242,3
186,302,3
22,377,1
244,51,2
166,346,1
298,474,4
115,265,2
253,465,5
305,451,3
6,86,3
62,257,2
286,1014,5
200,222,5
210,40,3
224,29,3
303,785,3
122,387,5
194,274,2
291,1042,4
234,1184,2
119,392,4
167,486,4
299,144,4
291,118,2
308,1,4
```

Step 2 Implement this version of [MLlib - Collaborative Filtering Examples](#)

### Using the GCP Console to Upload the File:

Navigate to Cloud Storage, create a new bucket.

Click the "Upload Files" button and select C:\Users\KANG\Downloads\formatted\_data.txt to upload.



### Create a new Python script named recommendation.py:

```
from pyspark.sql import SparkSession
from pyspark.mllib.recommendation import ALS, MatrixFactorizationModel, Rating
```

```
# Initialize Spark session
spark = SparkSession.builder.appName("MovieRecommendation").getOrCreate()
sc = spark.sparkContext
```

```
# Load and parse the data
data = sc.textFile("gs://movie_bucket_kang_1/formatted_data.txt")
ratings = data.map(lambda l: l.split(',')\
                    .map(lambda l: Rating(int(l[0]), int(l[1]), float(l[2]))))
```

```
# Train the recommendation model
rank = 10
numIterations = 10
model = ALS.train(ratings, rank, numIterations)
```

```
# Predict user ratings for movies
testdata = ratings.map(lambda p: (p[0], p[1]))
predictions = model.predictAll(testdata).map(lambda r: ((r[0], r[1]), r[2]))
```

```
# Calculate Mean Squared Error
ratesAndPreds = ratings.map(lambda r: ((r[0], r[1]), r[2])).join(predictions)
MSE = ratesAndPreds.map(lambda r: (r[1][0] - r[1][1]) ** 2).mean()
print("Mean Squared Error = " + str(MSE))
```

```
model.save(sc, "gs://movie_bucket_kang_1/model/myCollaborativeFilter")
sameModel = MatrixFactorizationModel.load(sc,
"gs://movie_bucket_kang_1/model/myCollaborativeFilter")
```

```
KLORER ...
ANG20000627
ml-100k
combined_sorted.txt
Dockerfile
input.txt
ml-100k.zip
ml-100k.zip.1
pagerank.py
part-00000
part-00001
README-cloudshell.txt
recommendation.py
u.data
word.py
wordcount.py

recommendation.py > %$ SparkSession
1 from pyspark.sql import SparkSession
2 from pyspark.mllib.recommendation import ALS, MatrixFactorizationModel, Rating
3
4 # Initialize Spark session
5 spark = SparkSession.builder.appName("MovieRecommendation").getOrCreate()
6 sc = spark.sparkContext
7
8 # Load and parse the data
9 data = sc.textFile("gs://movie_bucket_kang_1/formatted_data.txt")
10 ratings = data.map(lambda l: l.split(',')\
11 | | | | |.map(lambda l: Rating(int(l[0]), int(l[1]), float(l[2]))))
12
13 # Train the recommendation model
14 rank = 10
15 numIterations = 10
16 model = ALS.train(ratings, rank, numIterations)
17
18 # Predict user ratings for movies
19 testdata = ratings.map(lambda p: (p[0], p[1]))
20 predictions = model.predictAll(testdata).map(lambda r: ((r[0], r[1]), r[2]))
21
22 # Calculate Mean Squared Error
23 ratesAndPreds = ratings.map(lambda r: ((r[0], r[1]), r[2])).join(predictions)
24 MSE = ratesAndPreds.map(lambda r: (r[1][0] - r[1][1]) ** 2).mean()
25 print("Mean Squared Error = " + str(MSE))
26
27 # Save the model
28 model.save(sc, "gs://movie_bucket_kang_1/model/myCollaborativeFilter")
29 sameModel = MatrixFactorizationModel.load(sc, "gs://movie_bucket_kang_1/model/myCollaborativeFilter")
30
31 spark.stop()
32
```

In Cloud Shell, submit the PySpark job to your Dataproc cluster using the following command:

```
nerr: 60, Product: 1060, Rating: 3.7722818326989  
nerr: 648, Product: 1060, Rating: 2.4444493310426972  
nerr: 637, Product: 1060, Rating: 2.2057676459558113  
nerr: 381, Product: 1060, Rating: 3.04013668693882  
nerr: 83, Product: 1060, Rating: 3.0015552990122543  
nerr: 919, Product: 1060, Rating: 2.530305321243043  
nerr: 181, Product: 1060, Rating: 0.9504543294932521  
nerr: 189, Product: 1060, Rating: 4.4775761265816705  
nerr: 911, Product: 1060, Rating: 2.8923239955081566  
nerr: 821, Product: 1060, Rating: 4.54534501332398  
nerr: 921, Prod2d[07/1] 05f05f53 INFO GoogleCloudStorageFileSystemImpl: Successfully repaired "gs://movie_bucket_kang_1/predictions/" directory.  
job b7a34f683d405ea2be30fd91c0087b finished successfully  
state: true  
hiveControlFilesDir: gs:/dataproc-staging-us-central-1-d25505028363-7ewrrdz/google-cloud-dataproc-metainfo/0cb87e0-2a91-4ffa-ab1b-534ad70ee32e/jobs/b7a34f683d405ea2be30fd91c0087b/  
hiveConfResourceDir: gs:/dataproc-staging-us-central-1-d25505028363-7ewrrdz/google-cloud-datapro-metainfo/0cb87e0-2a91-4ffa-ab1b-534ad70ee32e/jobs/b7a34f683d405ea2be30fd91c0087b/driveroutpu  
t@id: 6094TeSe-3Aa2-37E-a067-a4f1f87fflfc  
location:  
clusterName: cluster-a96  
clusterUdid: 0cb87e0-2a91-4ffa-ab1b-534ad70ee32e  
sparkJobId:  
mainPythonFileUri: gs:/dataproc-staging-us-central-1-d25505028363-7ewrrdz/google-cloud-datapro-metainfo/0cb87e0-2a91-4ffa-ab1b-534ad70ee32e/jobs/b7a34f683d405ea2be30fd91c0087b/staging/recommendation_examp  
le:  
reference:  
jobId: b7a34f683d405ea2be30fd91c00eb7  
trackingUrl: enhanced-mote-424120-m9  
status:  
state: DOWN  
startedAtTime: '2024-07-17T05:05:17.11364Z'  
timestampLastory:  
state: FINISHING  
startedAtTime: '2024-07-17T05:05:08.209739Z'  
state: SETUP_DOWN  
startedAtTime: '2024-07-17T05:05:08.269087Z'  
details: Agent reported job success  
state: RUNNING  
startedAtTime: '2024-07-17T05:05:08.529993Z'  
arnApplication:  
name: MovieRecommendation  
progress: 1.0  
state: FINISHED  
trackingUrl: http://cluster-a96-nus-us-central-f.c.enhanced-mote-424120-m9.internal:8088/proxy/application%2F172118882903%2F0002/
```

User: 49, Product: 320, Rating: 5.3506158193725835  
User: 617, Product: 320, Rating: 4.82065962165874  
User: 833, Product: 320, Rating: 3.9459219576166227  
User: 13, Product: 320, Rating: 1.1587396127296636  
User: 747, Product: 320, Rating: 5.490706479128973  
User: 234, Product: 1330, Rating: 2.936149990875719  
User: 181, Product: 1330, Rating: 1.0629440294165278  
User: 943, Product: 1330, Rating: 2.9578486159627584  
User: 211, Product: 1330, Rating: 3.0941604634584916  
User: 14, Product: 408, Rating: 5.364819477762742  
User: 514, Product: 408, Rating: 4.149183501579902  
User: 226, Product: 408, Rating: 4.423223163562723  
User: 160, Product: 408, Rating: 4.153093371830105  
User: 420, Product: 408, Rating: 4.316801740306228  
User: 58, Product: 408, Rating: 4.436220926400643  
User: 338, Product: 408, Rating: 4.424361776020332  
User: 216, Product: 408, Rating: 4.227886496138721  
User: 292, Product: 408, Rating: 4.636627592094855  
User: 92, Product: 408, Rating: 4.320357428047913  
User: 344, Product: 408, Rating: 5.106607836349356  
User: 664, Product: 408, Rating: 4.80135455349273  
User: 84, Product: 408, Rating: 4.875224892218625  
User: 592, Product: 408, Rating: 5.092428295030623  
User: 536, Product: 408, Rating: 4.99543059017358  
User: 402, Product: 408, Rating: 4.608248581782899  
User: 334, Product: 408, Rating: 4.474194719977783  
User: 312, Product: 408, Rating: 4.519703965268387  
User: 158, Product: 408, Rating: 4.8338565484401625  
User: 286, Product: 408, Rating: 4.053583716249065  
User: 634, Product: 408, Rating: 3.7177468168353447  
User: 70, Product: 408, Rating: 4.466932630415672  
User: 906, Product: 408, Rating: 4.223861211925433  
User: 658, Product: 408, Rating: 4.042194351769586  
User: 738, Product: 408, Rating: 4.4810835522805945  
User: 822, Product: 408, Rating: 4.80934150577896  
User: 214, Product: 408, Rating: 4.316519452228186  
User: 654, Product: 408, Rating: 4.72090460672466  
User: 852, Product: 408, Rating: 4.284863745363509  
User: 56, Product: 408, Rating: 3.712554365528061  
User: 32, Product: 408, Rating: 3.2876072775417784  
User: 684, Product: 408, Rating: 4.406452492186036  
User: 806, Product: 408, Rating: 4.709603233936851  
User: 526, Product: 408, Rating: 4.609398672485364  
User: 412, Product: 408, Rating: 4.095676586727922  
User: 622, Product: 408, Rating: 5.080670713431014  
User: 838, Product: 408, Rating: 4.397468513950832  
User: 458, Product: 408, Rating: 4.876656682291041  
User: 868, Product: 408, Rating: 5.0349678214073545  
User: 18, Product: 408, Rating: 4.735741669085218  
User: 342, Product: 408, Rating: 4.127261017238123  
User: 269, Product: 408, Rating: 4.34540327151419  
User: 864, Product: 408, Rating: 5.092805315337992  
User: 142, Product: 408, Rating: 3.9740423146469883  
User: 6, Product: 408, Rating: 4.580530130400435

You can see the results by open predictions.

OBJECTS

CONFIGURATION

PERMISSIONS

PROTECTION

LIFECYCLE

OBSERVABILITY

INVENTORY REPORTS

OPERATIONS

Folder browser

movie\_bucket\_kang\_1

model/

predictions/

Buckets > movie\_bucket\_kang\_1 > predictions

UPLOAD FILES

UPLOAD FOLDER

CREATE FOLDER

TRANSFER DATA

MANAGE HOLDS

EDIT RETENTION

DOWNLOAD

DELETE

Filter by name prefix only

Filter

Filter objects and folders

Show

Live objects only

<input type="checkbox"/>	Name	Size	Type	Created	Storage class	Last modified	
<input type="checkbox"/>	_SUCCESS	0 B	application/octet-stream	Jul 16, 2024, 10:05:53 PM	Standard	Jul 16, 2024, 10:05:53 PM	
<input type="checkbox"/>	part-00000	1.4 MB	application/octet-stream	Jul 16, 2024, 10:05:52 PM	Standard	Jul 16, 2024, 10:05:52 PM	
<input type="checkbox"/>	part-00001	1.4 MB	application/octet-stream	Jul 16, 2024, 10:05:52 PM	Standard	Jul 16, 2024, 10:05:52 PM	