Movie Recommendation with MLlib - Collaborative Filtering

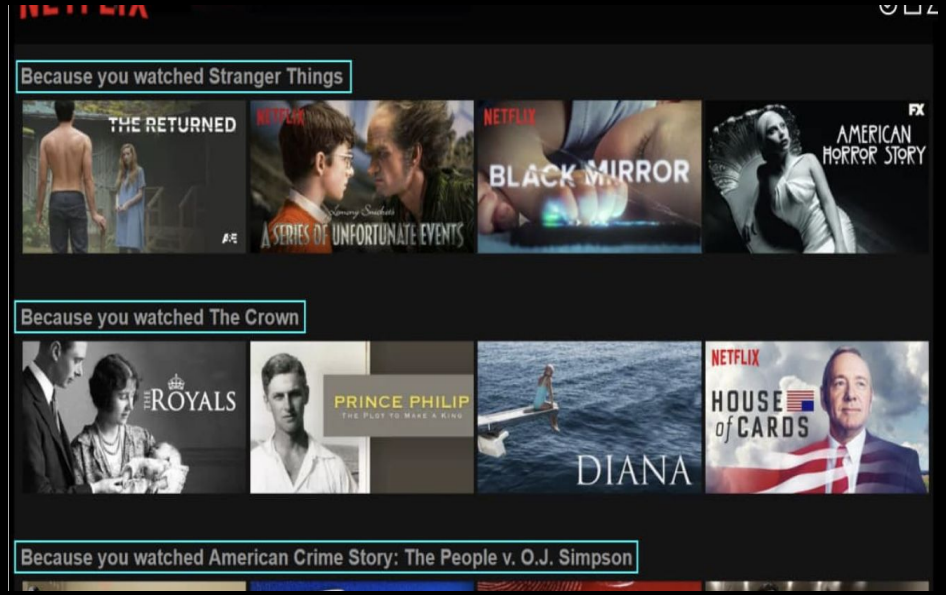
—

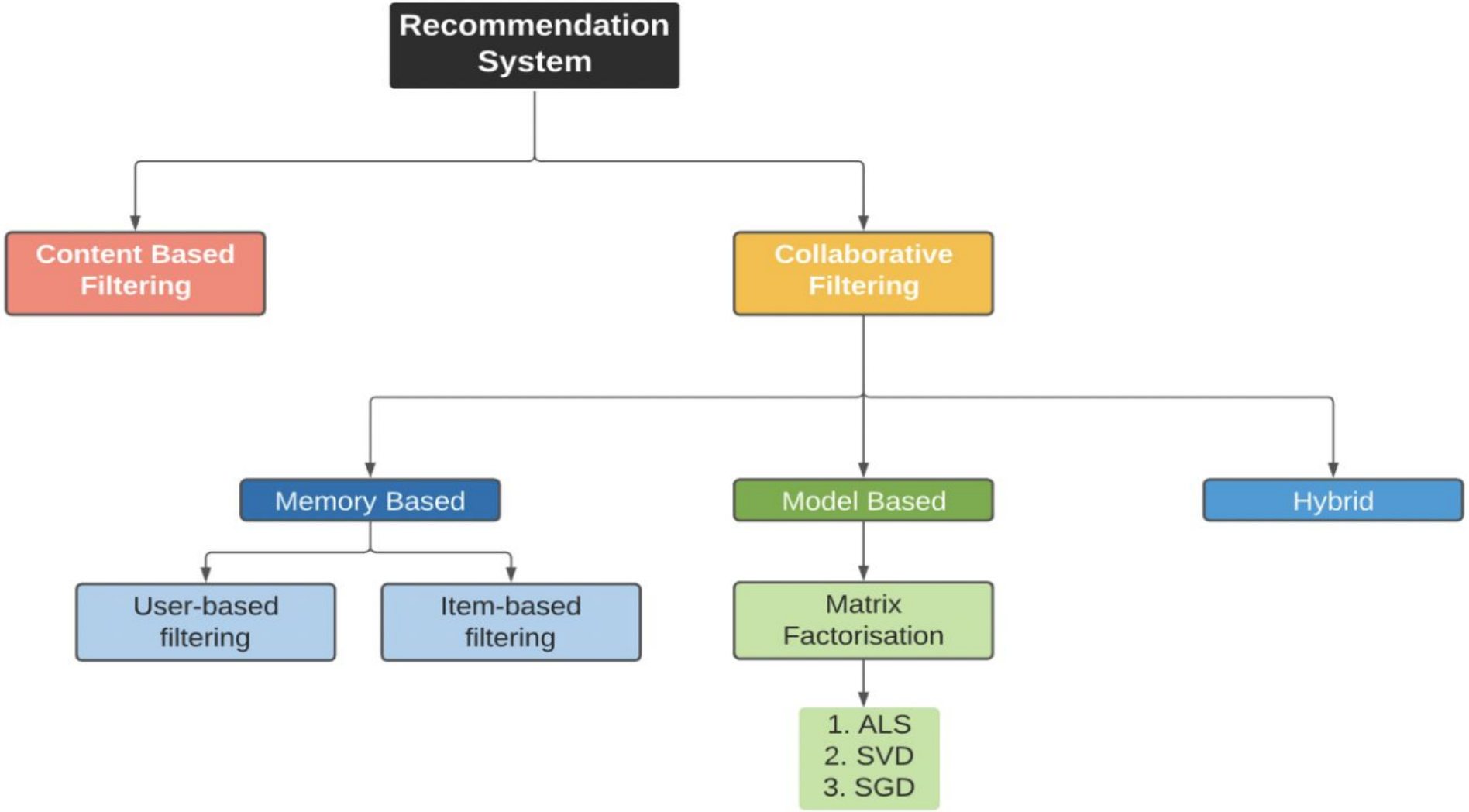
Norina Akhtar

Table Of Contents

Introduction

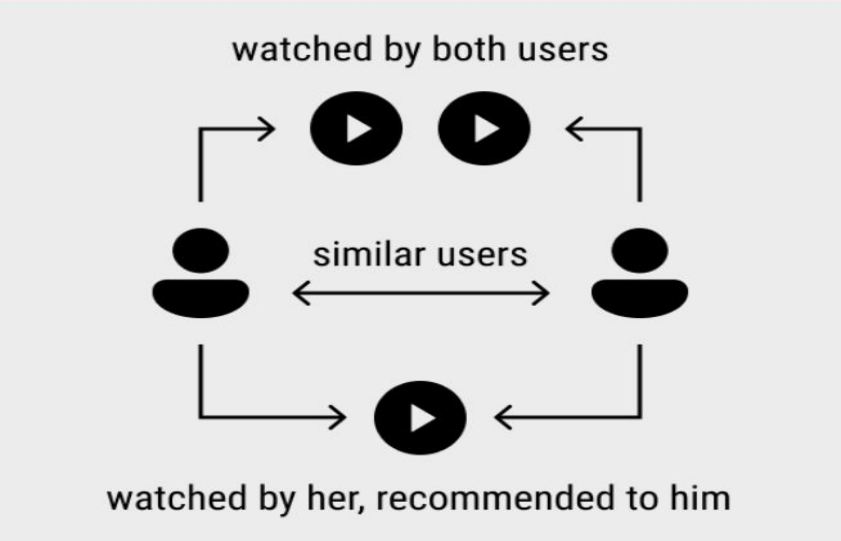
A Movie recommendation system , is an ML based approach to filtering or predicting the users film preferences based on their past choices and behaviour.



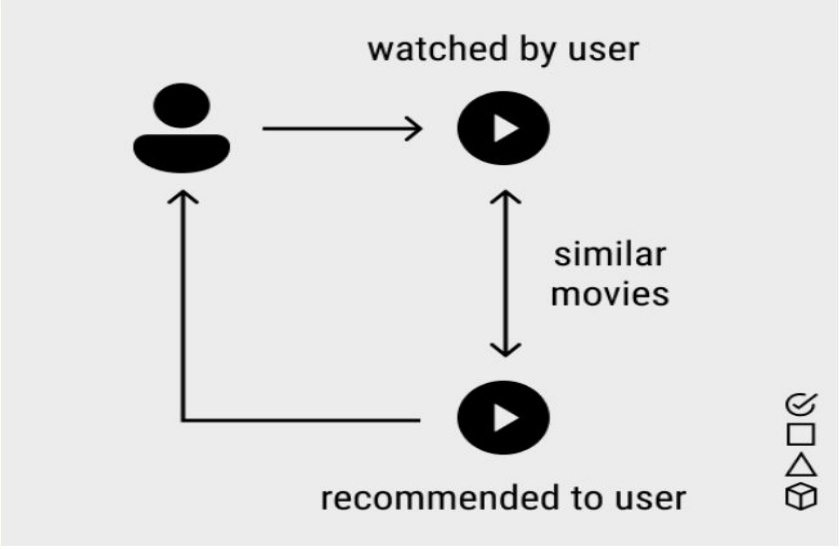

Recommendation System
Content Based Filtering
Collaborative Filtering
Memory Based
Model Based
Hybrid
User-based filtering
Item-based filtering
Matrix Factorisation
1. ALS
2. SVD
3. SGD

Categories of Movie Recommendation Includes

Collaborative Filtering



Content-Based Filtering



Collaborative Filtering

It is based on the combination of relevant users and other users behaviour. The system compares and contrasts these behaviours for the most optimal results. Its collaboration of the multiple users film preferences and behaviors.

The algorithm is divided into two categories:

A.**Memory base - User based filtering**: look for similar patterns in movie preferences in the target user and other users in the database

B. **Model Base-Item-based collaborative filtering**: look for similar items(movies) that target users rate or interact with.

Design

User Based Recommendation System

Rating Matrix A (users x items)

	Item 0	Item 1	Item 2	Item 3
User 0	5.00	5.00	2.00	--
User 1	2.00	--	3.00	5.00
User 2	--	5.00	--	3.00
User 3	3.00	--	--	5.00

Formula

R(u, i) = SUM(v in U')(sim(u, v) x R(v, i))

where U' is the user set who rated item i

For example,

The rating of user 2, item 2

= R(2, 2)

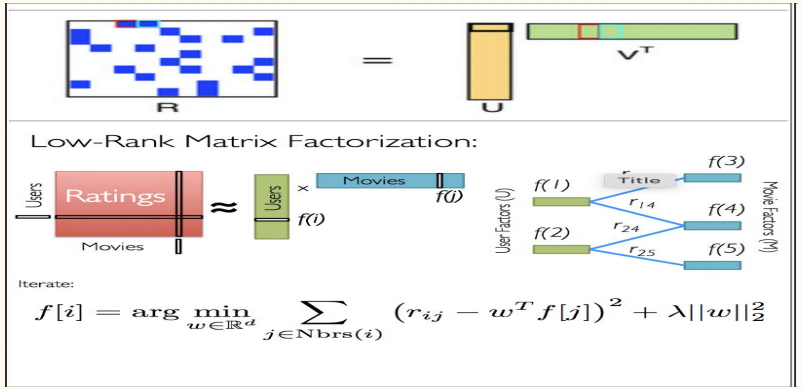
= sim(2, 0) x R(0, 2) + sim(2, 1) x R(1, 2)

= sim(2, 0) x 2 + sim(2, 1) x 3

Matrix Factorization Based Recommendation system
Alternating Least Square With Regularization(ALS-WR)

ALS-WR is for factorizing original rating matrix R(U x I) into 2 matrix U(U x F), and M(I x F) so that cost function is minimized.

- R: Rating
- U: User
- I: Item
- F: Feature



Input: Rating Matrix A (users x items)

	Item 0	Item 1	Item 2	Item 3
User 0	5.00	5.00	2.00	--
User 1	2.00	--	3.00	5.00
User 2	--	5.00	--	3.00
User 3	3.00	--	--	5.00

User Feature Matrix U (users x features)

	Feature 0	Feature 1	Feature 2
User 0	1.12	1.49	0.48
User 1	1.31	-0.52	0.59
User 2	1.13	0.67	-0.52
User 3	1.39	0.05	0.45

Item Feature Matrix M (items x features)

	Feature 0	Feature 1	Feature 2
Item 0	1.81	1.62	0.74
Item 1	2.66	1.71	-1.08
Item 2	1.73	-0.23	0.78
Item 3	3.16	-0.24	0.90

Output: Prediction Matrix A_k (users x items)

A_k = UM'

	Item 0	Item 1	Item 2	Item 3
User 0	4.78	4.98	1.97	3.61
User 1	1.98	1.97	2.85	4.81
User 2	2.75	4.71	1.40	2.94
User 3	2.94	3.32	2.13	4.56

Implementation

1.Recommendation using Alternating least square(ALS)

ALS attempts to estimate the ratings matrix R as the product of two lower-rank

Matrices , x and y , i-e x*yt = R. The general approach is iterative.

Steps:

- Build out an ALS model
- Hyperparameter tuning and cross validation: cross validator
- Check the best model parameter
- Fit the best model and evaluate predictions
- Make recommendations
- Convert recommendations into interpretable format

2. Study Colab

Colaboratory, or “Colab” for short, is a product from Google Research. Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education. More technically, Colab is a hosted Jupyter notebook service that requires no setup to use, while providing access free of charge to computing resources including GPUs.

3. ALS Recommender-pyspark

Recommendation_Engine_MovieLens.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

Files

- ..
- sample_data
  - README.md
  - anscombe.json
  - california_housing_test.csv
  - california_housing_train.csv
  - mnist_test.csv
  - mnist_train_small.csv
  - movies.csv
  - ratings.csv

+ Code + Text

RAM Disk Editing

Import libraries

[6]
```
#https://grouplens.org/datasets/movielens/
```

Initiate spark session

```
!pip install pyspark
import pandas as pd
from pyspark.sql.functions import col, explode
from pyspark import SparkContext
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting pyspark
  Downloading pyspark-3.3.1.tar.gz (281.4 MB)
     |████████████████████████████████| 281.4 MB 55 kB/s
Collecting py4j==0.10.9.5
  Downloading py4j-0.10.9.5-py2.py3-none-any.whl (199 kB)
     |████████████████████████████████| 199 kB 77.7 MB/s
Building wheels for collected packages: pyspark
  Building wheel for pyspark (setup.py) ... done
  Created wheel for pyspark: filename=pyspark-3.3.1-py2.py3-none-any.whl size=281845512 sha256=9e1759fb5bc8424ba7b99cc920
  Stored in directory: /root/.cache/pip/wheels/42/59/f5/79a5bf931714dcd201b26025347785f087370a10a3329a899c
Successfully built pyspark
Installing collected packages: py4j, pyspark
Successfully installed py4j-0.10.9.5 pyspark-3.3.1
```

Result

```
nrecommendations.join(movies, on='movieId').filter('userId = 100').show()
```

```
+-------+------+---------+--------------------+--------------------+
|movieId|userId|   rating|               title|              genres|
+-------+------+---------+--------------------+--------------------+
|  67618|   100|  5.06765|Strictly Sexual (...|Comedy|Drama|Romance|
|  33649|   100|5.0600514|  Saving Face (2004)|Comedy|Drama|Romance|
|   3379|   100| 4.961345| On the Beach (1959)|               Drama|
|  42730|   100| 4.961083|   Glory Road (2006)|               Drama|
|  26073|   100|4.9105597|Human Condition I...|           Drama|War|
|   7071|   100|4.9105597|Woman Under the I...|               Drama|
|  84273|   100|4.9105597|Zeitgeist: Moving...|         Documentary|
| 179135|   100|4.9105597|Blue Planet II (2...|         Documentary|
| 184245|   100|4.9105597|De platte jungle ...|         Documentary|
| 117531|   100|4.9105597|    Watermark (2014)|         Documentary|
+-------+------+---------+--------------------+--------------------+
```

Download the file Recommendation_Engine_MovieLens.ipynb as .py

Recommendation_Engine_MovieLens.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

Locate in Drive
Open in playground mode

New notebook
Open notebook ⌘/Ctrl+O
Upload notebook

Rename
Move
Move to trash

Save a copy in Drive
Save a copy as a GitHub Gist
Save a copy in GitHub

Save ⌘/Ctrl+S
Save and pin revision ⌘/Ctrl+M S
Revision history

Download
Print ⌘/Ctrl+P

Download .ipynb
Download .py

Code + Text

RAM Disk Editing

RegParam: 0.15

22]
```
# View the predictions
test_predictions = best_model.transform(test)
RMSE = evaluator.evaluate(test_predictions)
print(RMSE)
```

```
0.868142452308257
```

23]
```
test_predictions.show()
```

```
+------+-------+------+----------+
|userId|movieId|rating|prediction|
+------+-------+------+----------+
|   148|    356|   4.0| 3.5239072|
|   148|   4896|   4.0|  3.510039|
|   148|   4993|   3.0| 3.4699442|
|   148|   7153|   3.0| 3.4208226|
|   148|   8368|   4.0| 3.5653324|
|   148|  40629|   5.0|  3.301226|
|   148|  50872|   3.0| 3.6407778|
|   148|  60069|   4.5| 3.7034664|
|                     .4164522|
|                     .2564936|
|                     3.451922|
|                     .7482162|
|   148| 115617|   3.5| 3.6247354|
|   148| 122886|   3.5|  3.372223|
|   463|    296|   4.0| 4.1569595|
|   463|    527|   4.0| 3.7740083|
|   463|   2019|   4.0| 3.9213715|
|   471|    527|   4.5| 3.8166275|
|   471|   6016|   4.0|  3.990181|
|   471|   6333|   2.5| 3.2488956|
+------+-------+------+----------+
only showing top 20 rows
```

4. Set up pyspark on GCP

- Enable compute engine API
- Create cluster
- Connect ssh

Start your Free Trial with $300 in credit. Don't worry—you won't be charged if you run out of credits. Learn more

DISMISS ACTIVATE

Google Cloud CS570bigData Search Products, resources, docs (/)

Dataproc

Jobs on Clusters
- Clusters
- Jobs
- Workflows
- Autoscaling policies

Serverless
- Batches

Metastore Services
- Metastore
- Federation

Utilities
- Component exchange
- Workbench

Release Notes

Create a Dataproc cluster on Compute Engine

- **Set up cluster**
  Begin by providing basic information.
- **Configure nodes** (optional)
  Change node compute and storage capabilities.
- **Customize cluster** (optional)
  Add cluster properties, features, and actions.
- **Manage security** (optional)
  Change access, encryption, and security settings.

CREATE CANCEL

EQUIVALENT COMMAND LINE

Name

Cluster Name *
cluster-49f9

Location

Region *
us-west1

Zone *
us-west1-a

Cluster type

- (●) Standard (1 master, N workers)
- ( ) Single Node (1 master, 0 workers)
  Provides one node that acts as both master and worker. Good for proof-of-concept or small-scale processing
- ( ) High Availability (3 masters, N workers)
  Hadoop High Availability mode provides uninterrupted YARN and HDFS operations despite single-node failures or reboots

Autoscaling

Automates cluster resource management based on an autoscaling policy.

Policy
None

Enhanced Flexibility Mode

0bigData Search Products, resources, docs (/)

Clusters

CREATE CLUSTER REFRESH START STOP DELETE REGIONS + 5 RECOMMENDED ALERTS SHOW INFO PANEL

Filter Search clusters, press Enter

Name	Status	Region	Zone	Total worker nodes	Scheduled deletion	Cloud Storage staging bucket	Created
cluster-49f9	Provisioning	us-west1	us-west1-a	2	Off	dataproc-staging-us-west1-324065753903-h3ujxhnb	Nov 22, 2022, 3:45:52 PM

Connect to ssh and upload files on GCP

SSH-in-browser

UPLOAD FILE DOWNLOAD FILE

```
Linux cluster-49f9-m 5.10.0-0.deb10.16-amd64 #1 SMP Debian 5.10.127-2~bpo10+1 (2022-07-28) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
nakhtar@cluster-49f9-m:~$ ls
nakhtar@cluster-49f9-m:~$
```

Transferred 2 items

movies.csv

ratings.csv

SSH-in-browser

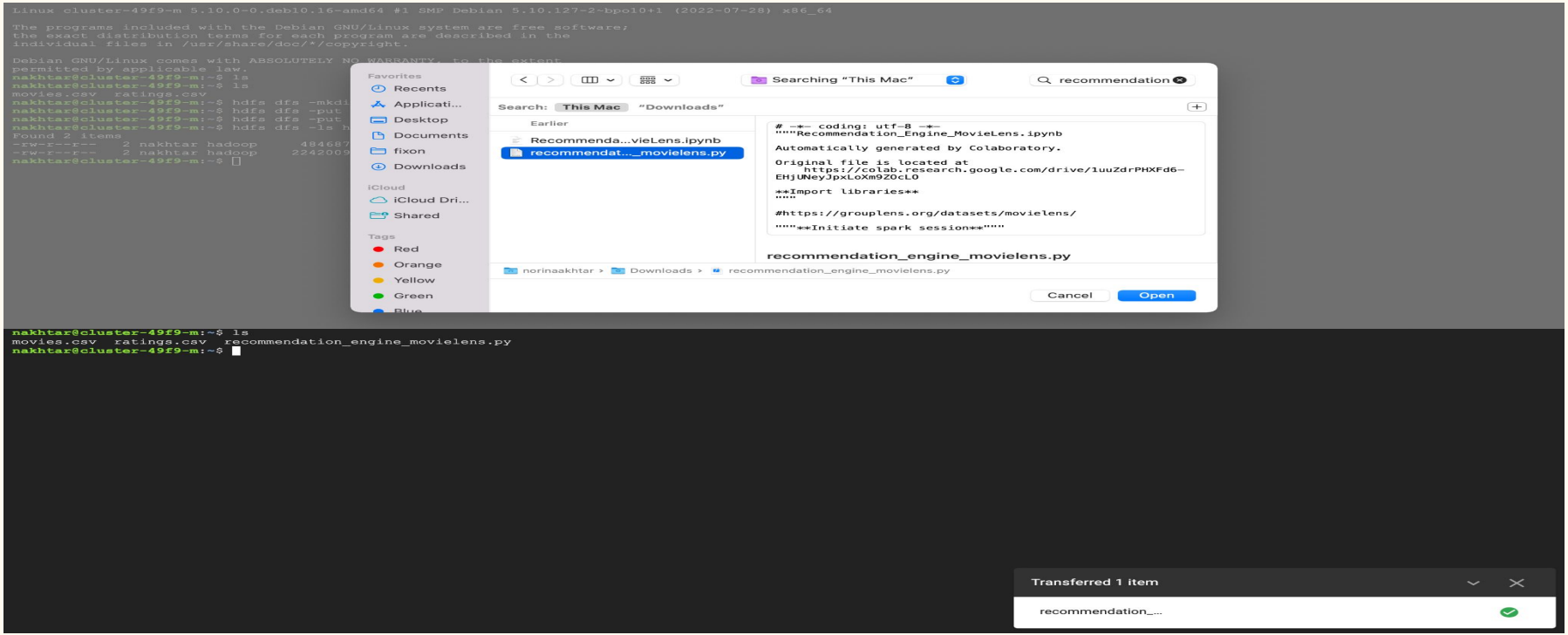
UPLOAD FILE DOWNLOAD FILE

```
Linux cluster-49f9-m 5.10.0-0.deb10.16-amd64 #1 SMP Debian 5.10.127-2~bpo10+1 (2022-07-28) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
nakhtar@cluster-49f9-m:~$ ls
nakhtar@cluster-49f9-m:~$ ls
movies.csv  ratings.csv
nakhtar@cluster-49f9-m:~$
```

Upload the .py file on GCP Cluster



Make directory on Gcp and put files in hdfs

```
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
nakhtar@cluster-49f9-m:~$ ls
nakhtar@cluster-49f9-m:~$ ls
movies.csv  ratings.csv
nakhtar@cluster-49f9-m:~$ hdfs dfs -mkdir hdfs:///mydata
nakhtar@cluster-49f9-m:~$ hdfs dfs -put movies.csv hdfs:///mydata
nakhtar@cluster-49f9-m:~$ hdfs dfs -put ratings.csv hdfs:///mydata
nakhtar@cluster-49f9-m:~$ hdfs dfs -ls hdfs:///mydata
Found 2 items
-rw-r--r--   2 nakhtar hadoop     484687 2022-11-22 23:51 hdfs:///mydata/movies.csv
-rw-r--r--   2 nakhtar hadoop    2242009 2022-11-22 23:51 hdfs:///mydata/ratings.csv
nakhtar@cluster-49f9-m:~$
```

Changes required on .py file

- Delete the —> !pip install pyspark
- Adjust the path of csv files

recommendation_engine_movielens.py UNREGISTERED

recommendation_engine_movielens.py

```python
# -*- coding: utf-8 -*-
"""Recommendation_Engine_MovieLens.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1uuZdrPHXFd6-EHjUNeyJpxLoXm9ZOcLO

**Import libraries**
"""

#https://grouplens.org/datasets/movielens/

"""**Initiate spark session**"""

# !pip install pyspark
import pandas as pd
from pyspark.sql.functions import col, explode
from pyspark import SparkContext

from pyspark.sql import SparkSession
sc = SparkContext
# sc.setCheckpointDir('checkpoint')
spark = SparkSession.builder.appName('Recommendations').getOrCreate()

"""**Load data**"""

movies = spark.read.csv("hdfs:///mydata/movies.csv",header=True)
ratings = spark.read.csv("hdfs:///mydata/ratings.csv",header=True)

ratings.show()

ratings.printSchema()

ratings = ratings.\
    withColumn('userId', col('userId').cast('integer')).\
    withColumn('movieId', col('movieId').cast('integer')).\
    withColumn('rating', col('rating').cast('float')).\
    drop('timestamp')
ratings.show()
```

```
nakhtar@cluster-49f9-m:~$ spark-submit recommendation_engine_movielens.py
```

Output

movieId	userId	rating	title	genres
67618	100	5.1201425	Strictly Sexual (...	Comedy|Drama|Romance
3379	100	5.064743	On the Beach (1959)	Drama
42730	100	5.042285	Glory Road (2006)	Drama
33649	100	5.021657	Saving Face (2004)	Comedy|Drama|Romance
117531	100	4.9267745	Watermark (2014)	Documentary
7071	100	4.9267745	Woman Under the I...	Drama
184245	100	4.9267745	De platte jungle ...	Documentary
26073	100	4.9267745	Human Condition I...	Drama|War
179135	100	4.9267745	Blue Planet II (2...	Documentary
84273	100	4.9267745	Zeitgeist: Moving...	Documentary

movieId	userId	rating	title	genres
1101	100	5.0	Top Gun (1986)	Action|Romance
1958	100	5.0	Terms of Endearme...	Comedy|Drama
2423	100	5.0	Christmas Vacatio...	Comedy
4041	100	5.0	Officer and a Gen...	Drama|Romance
5620	100	5.0	Sweet Home Alabam...	Comedy|Romance
368	100	4.5	Maverick (1994)	Adventure|Comedy|...
934	100	4.5	Father of the Bri...	Comedy
539	100	4.5	Sleepless in Seat...	Comedy|Drama|Romance
16	100	4.5	Casino (1995)	Crime|Drama
553	100	4.5	Tombstone (1993)	Action|Drama|Western

5. Shutdown the cluster

Google Cloud | CS570bigData | Search Products, resources, docs (/) | N

Dataproc

Jobs on Clusters
- Clusters
- Jobs
- Workflows
- Autoscaling policies

Serverless
- Batches

Metastore Services
- Metastore
- Federation

Utilities

Cluster details | SUBMIT JOB | REFRESH | START | STOP | DELETE | VIEW LOGS

Consider using Auto Zone rather than selecting a zone manually. See https://cloud.google.com/dataproc/docs/concepts/configuring-clusters/auto-zone MORE

Name	cluster-49f9
Cluster UUID	1700c920-a9a6-43ad-aa3a-6ef02e774b7b
Type	Dataproc Cluster
Status	Running

MONITORING | JOBS | VM INSTANCES | CONFIGURATION | WEB INTERFACES

Filter Filter instances

	Name	Role	
✓	cluster-49f9-m	Master	SSH
✓	cluster-49f9-w-0	Worker	
✓	cluster-49f9-w-1	Worker	

Enhancement Ideas

- For future work, we can change the training input parameters of the model by playing around with the *window* size, as well as the *min_count* parameters and see what the resulting recommendations looks like.

- We can use other algorithms(**Session-Based Recommender Systems with Word2Vec)**

Word2Vec is a model that, when well trained, is capable of capturing the meaning of words, based on their context.

Conclusion

Collaborative Filtering(CF) is a subset of algorithms that exploit other users and items along with their ratings(selection, purchase information could be also used) and target user history to recommend an item that target user does not have ratings for.

Hence, CF differs itself from content-based methods in the sense that user or the item itself does not play a role in recommendation but rather how(rating) and which users(user) rated a particular item.

References

- https://towardsdatascience.com/build-recommendation-system-with-pyspark-using-alternating-least-squares-als-matrix-factorisation-ebe1ad2e7679
- Collaborative filtering
- Collaborative Filtering for Implicit Feedback Datasets