Movie Recommendation with MLlib - Collaborative Filtering

FATEMA NAGORI 19635

TABLE OF CONTENT

INTRODUCTION

Recommender System

Generate meaningful recommendation to a collection of users for item or product that might interest them.

Where can RS be found?

- Movie recommendation (Netflix)
- Related product recommendation (Amazon)
- Web page ranking (Google)
- Social recommendation (Facebook)
- News content recommendation (Yahoo)
- Priority inbox & spam filtering (Google)
- Online dating (OK Cupid)
- Computational Advertising (Yahoo)

Taxonomy of Recommender System

Collaborative filtering(CF)

Content based filtering(CBF)

Knowledge based filtering(KBF)

Hybrid

Content Based Filtering

CREATES PROFILE FOR USER/MOVIE

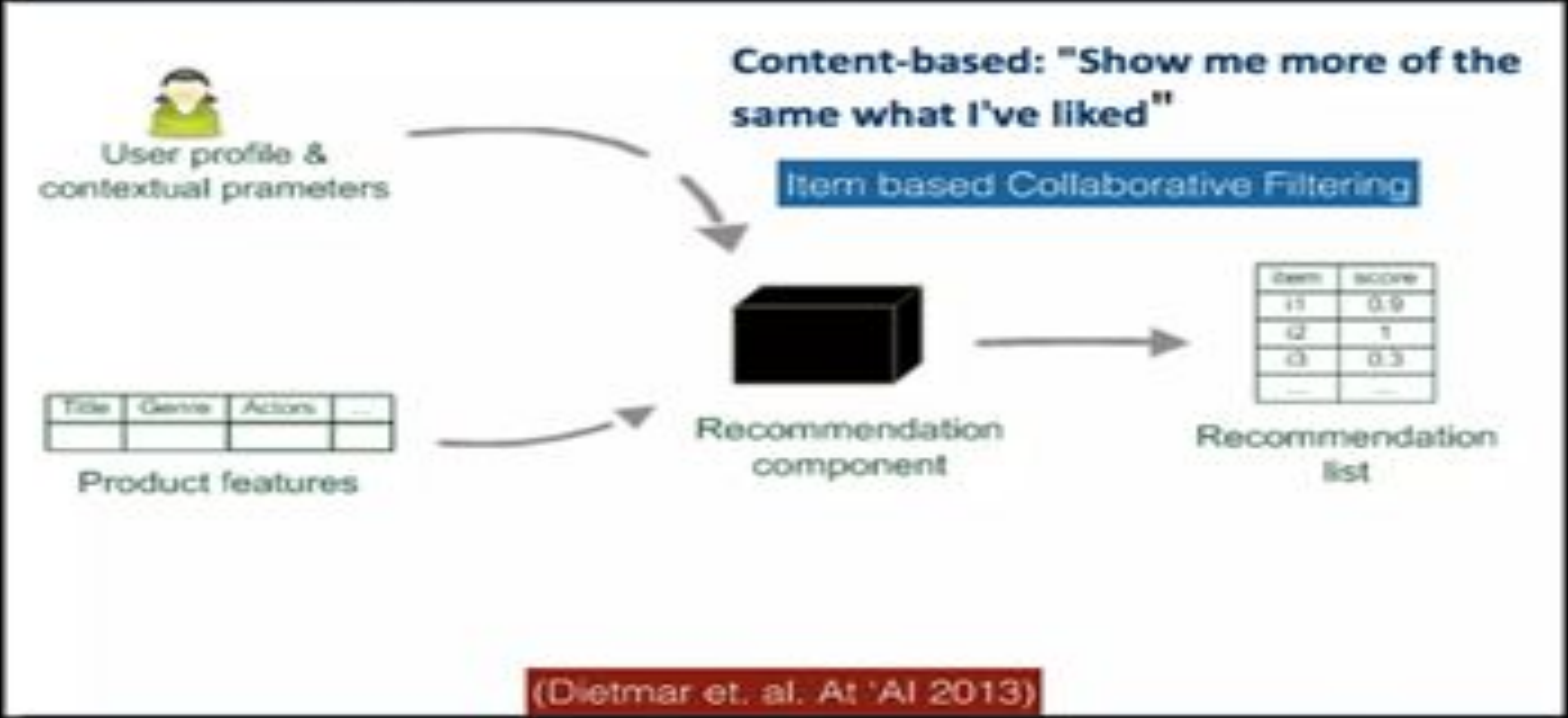
REQUIRES GATHERING EXTERNAL DATA

DENSE DATA

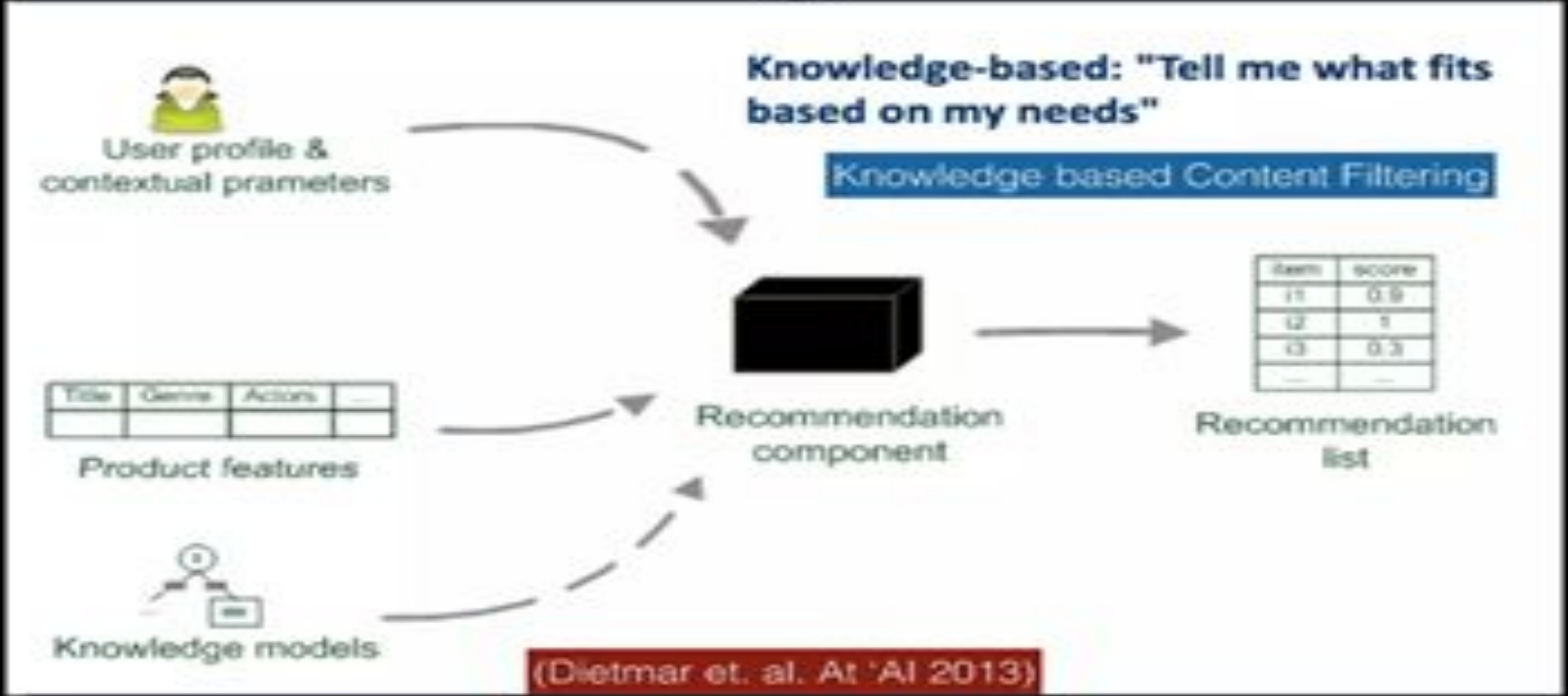
DOMAIN BOUNDED

NO COLD START PROBLEM

Content based

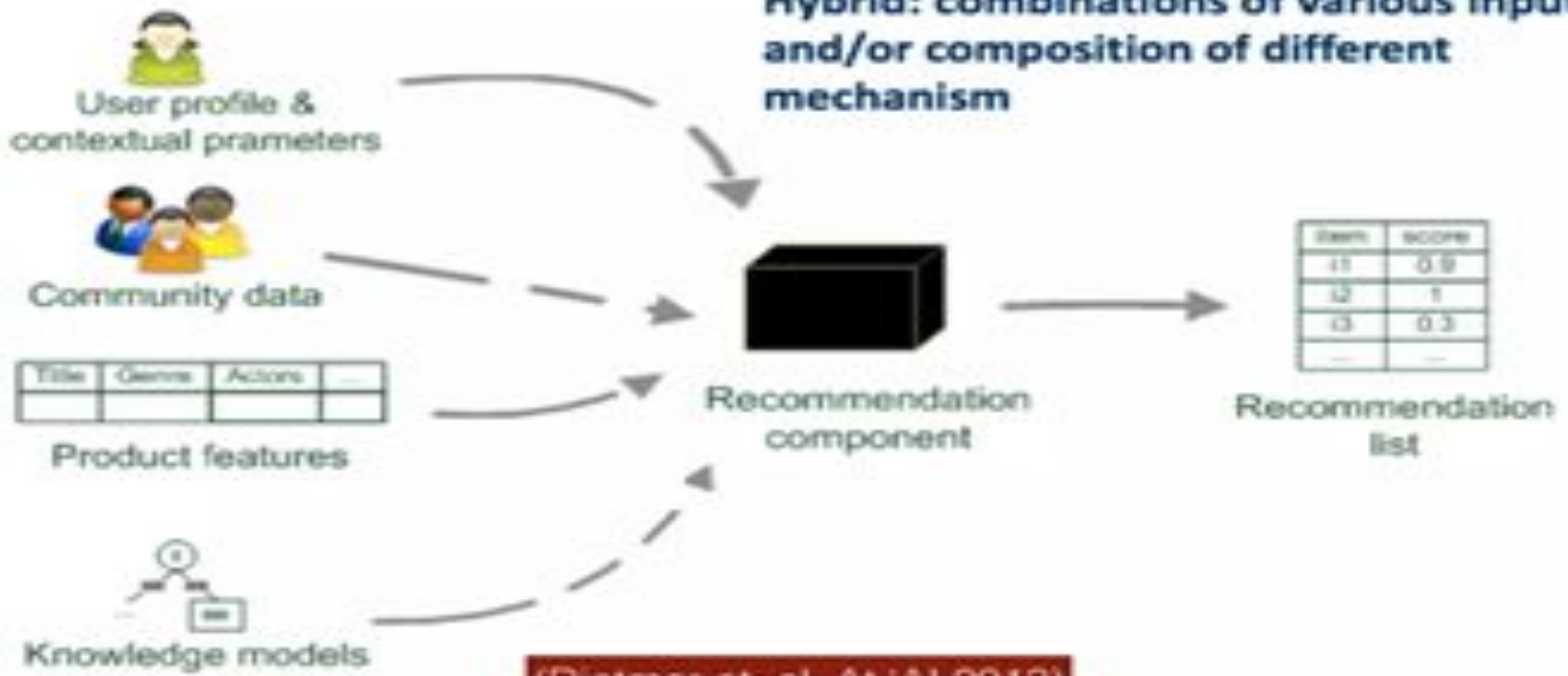


Knowledge based



Hybrid

Hybrid: combinations of various inputs and/or composition of different mechanism

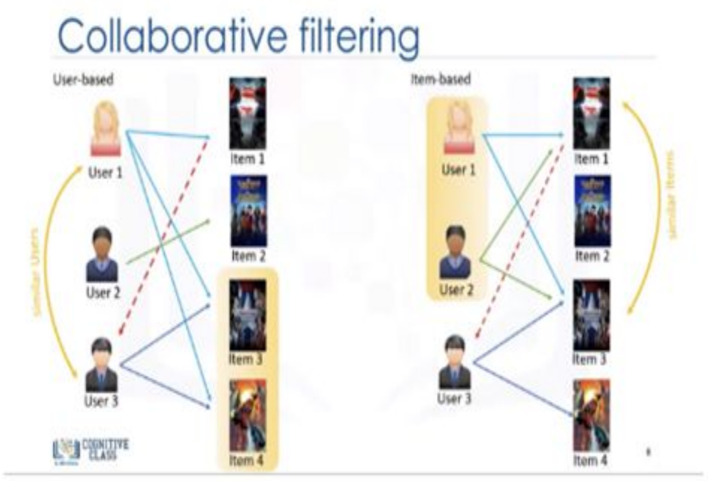


(Dietmar et. al. At 'AI 2013)

COLLABORATIVE FILTERING

Collaborative filtering is a system that predicts user behavior based on historical user data. From this, we can understand that this is used as a recommendation system.

For example, Amazon recommends products or gives discounts based on historical user data or YouTube recommends videos based on your history.



COLLABORATIVE FILTERING

Relies on past user behaviour
1. Implicit Feedback
2. Explicit Feedback

Requires no gathering of external data
1. Sparse data
2. Domain free
3. Cold start problem

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

Note:

- User-based Recommendation System (local copy)
  - It is user-based

```
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
```

Note:

- sim(u, v) is the similarity (formula) of user u and user v

- **Issue**
  - **Scalability of the user profiles.**
    - **The algorithm cannot be scaled to handle large data set**
  - **Sparseness of the user profiles.**
    - **The data in the Rating Matrix (users x items) is sparse.**

ALS_WR approach

- It makes lambda, λ, less dependent on the scale of the dataset.
  - So we can apply the best parameter learned from a sampled subset to the full dataset and expect similar performance.
- How
  - Scale the regularization parameter lambda, λ, in solving each least squares problem by
    - the number of ratings the user generated in updating user factors, or
    - the number of ratings the product received in updating product factors

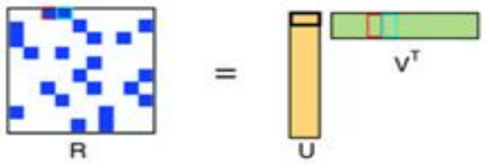
DESIGN

Matrix Factorization-based Recommendation System

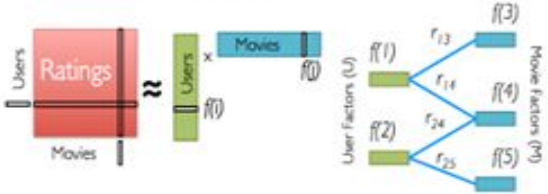
Alternating Least Square With Regularization (ALS-WR)

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



Low-Rank Matrix Factorization:



Iterate:

$$f[i] = \arg\min_{w \in \mathbb{R}^d} \sum_{j \in \mathrm{Nbrs}(i)} \left(r_{ij} - w^T f[j]\right)^2 + \lambda \|w\|_2^2$$

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)

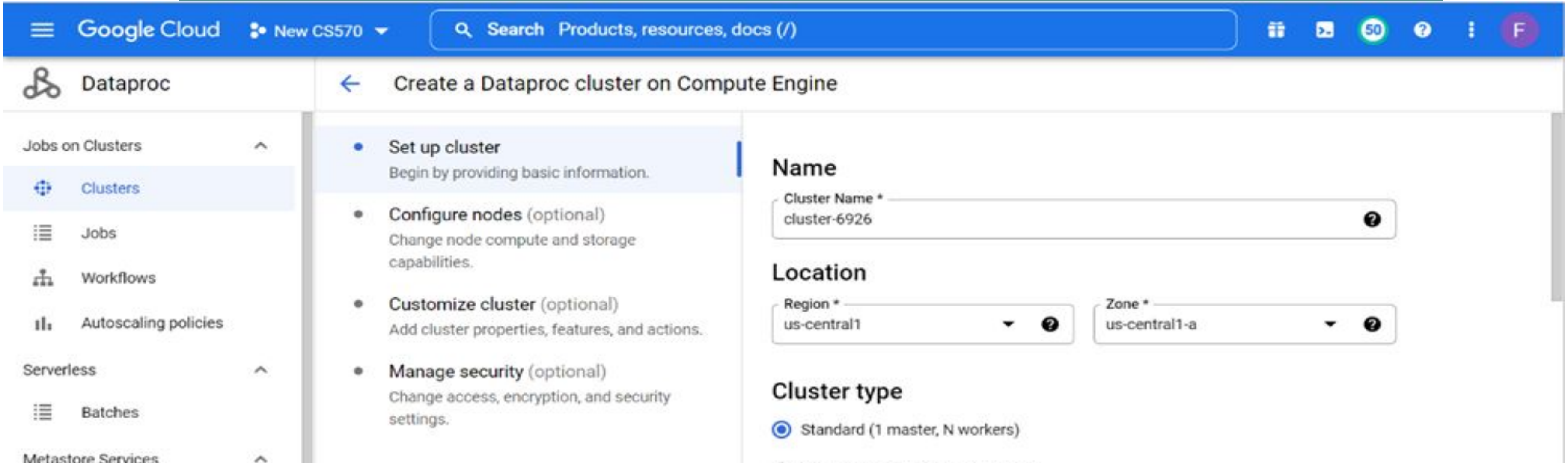
A_k = UM'

	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-Collaborative Filtering with ALS+PySpark+GCP

SET UP PySpark on GCP

- Enable the Google Cloud Compute Engine API
- Create, Configure and Launch a Google Cloud Dataproc cluster
- Connecting to the Master Node using Secure Shell (ssh)

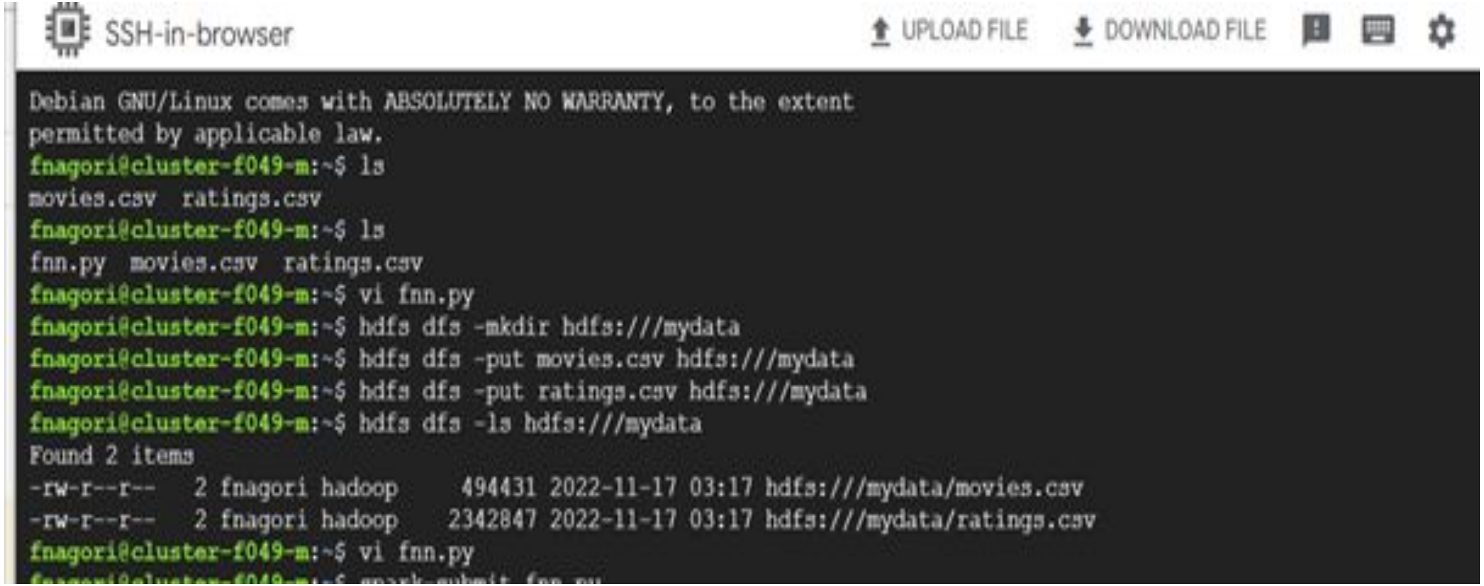


Implementation-Collaborative Filtering with ALS+PySpark+GCP

Steps To RUN the .py file at GCP

- Prepare Data:upload movies.csv and ratings.csv files at GCP
- Create a directory (folder) to store the data: hdfs dfs -mkdir hdfs:///mydata
- Copy the date to HDFS:
- hdfs dfs -put movies.csv hdfs:///mydata
- hdfs dfs -put ratings.csv hdfs:///mydata
- Prepare the program: vi fnn..py
- Modify the path of movies.csv and ratings.csv in program
- movies=hdfs:///mydata/movies.csv
- ratings=hdfs:///mydata/ratings.csv
- Running the program with Pyspark: spark-submit fnn.py

TEST-Collaborative Filtering with ALS+PySpark+GCP




```
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
fnagori@cluster-f049-m:~$ ls
movies.csv  ratings.csv
fnagori@cluster-f049-m:~$ ls
fnn.py  movies.csv  ratings.csv
fnagori@cluster-f049-m:~$ vi fnn.py
fnagori@cluster-f049-m:~$ hdfs dfs -mkdir hdfs:///mydata
fnagori@cluster-f049-m:~$ hdfs dfs -put movies.csv hdfs:///mydata
fnagori@cluster-f049-m:~$ hdfs dfs -put ratings.csv hdfs:///mydata
fnagori@cluster-f049-m:~$ hdfs dfs -ls hdfs:///mydata
Found 2 items
-rw-r--r--   2 fnagori hadoop     494431 2022-11-17 03:17 hdfs:///mydata/movies.csv
-rw-r--r--   2 fnagori hadoop    2342847 2022-11-17 03:17 hdfs:///mydata/ratings.csv
fnagori@cluster-f049-m:~$ vi fnn.py
```

TEST-Collaborative Filtering with ALS+PySpark+GCP

Check the best model parameters

Let us check, which parameters out of the 16 parameters fed into the

crossvalidator, resulted in the best model.

```
only showing top 20 rows

Num models to be tested:  16
CrossValidator_e062007b629f
<class 'pyspark.ml.recommendation.ALSModel'>
**Best Model**
  Rank: 50
  MaxIter: 10
  RegParam: 0.15
0.8685666272031624
```

TEST-Collaborative Filtering with ALS+PySpark+GCP

RESULT: Recommendations based on our search

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

```
22/11/17 04:11:10 INFO org.sparkproject.jetty.server.AbstractConnector: Stopp
.1)}{0.0.0.0:0}
fnagori@cluster-f049-m:~$
```

Enhancement Ideas

Two types of algorithms for collaborative filtering have been researched:

memory-based CF and model-based CF.

Memory-based approaches identify the similarity between two users by comparing their ratings on a set of items and have suffered from two fundamental problems: sparsity and scalability.

the model- based approaches have been proposed to alleviate these problems, but these approaches tend to limit the range of users.

The collaborative filtering recommendation method combining memory-based CF and model-based CF can provide better recommendation than traditional collaborative filtering.

Conclusion

Collaborative Filtering does not need features about the items or users to be known. Collaborative filtering recommender systems can help recommenders not specializing in a user’s profile. No domain knowledge is necessary in the case of collaborative filtering. It also has a great starting point and involves serendipity. Thus It is the best technique for recommendations.

References

Movie Recommendation with MLlib - Collaborative Filtering

COLAB

USE GCP

GCP common task

Write and run Spark Scala jobs on Dataproc

Spark Tutorial

THANK YOU!