Movie Recommendation with MLlib

CS570: Big-data Processing and Analytics

SFBU
Navya Kandimalla

Table Of Contents

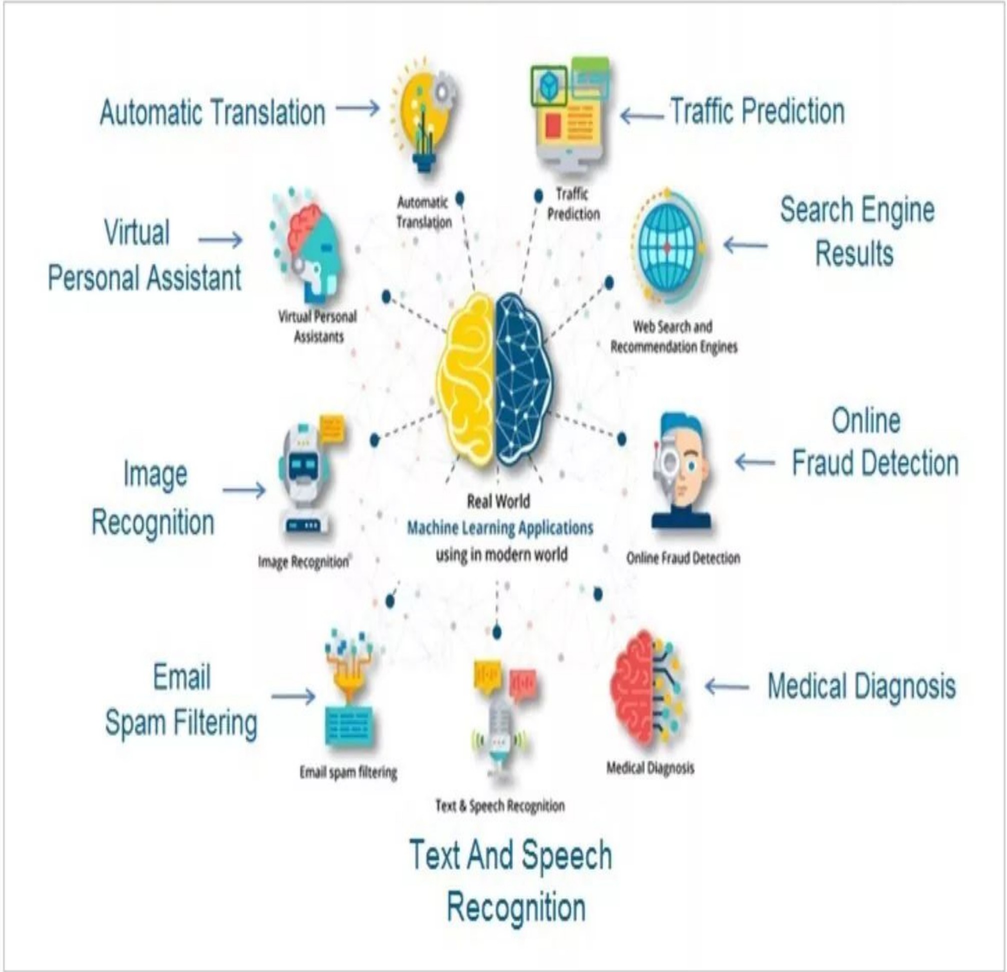
Introduction

- Machine learning (ML) is an area of artificial intelligence (AI) that enables computers to "learn" for themselves over time from training data and develop without explicit programming.
- Machine learning algorithms are able to detect patterns in data and learn from them, in order to make their own predictions.
- It is used in a variety of contexts, such as those involving movies, music, news, books, research articles,

Applications:

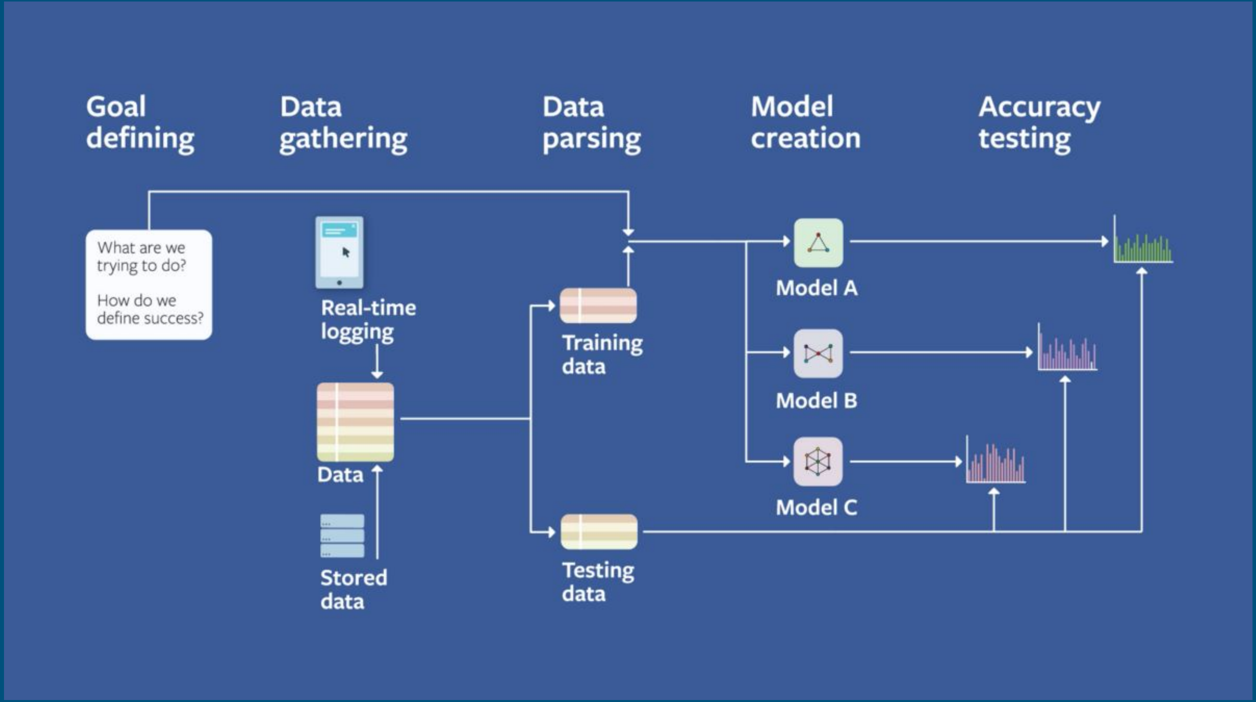
- Traffic Alerts.
- Social Media.
- Transportation and Commuting.
- Products Recommendations.
- Virtual Personal Assistants.
- Self Driving Cars.
- Dynamic Pricing.
- Google Translate.

Top Real-World Examples of Machine Learning



Design

- To quickly process vast amounts of data, the best Big Data technology is required. As a result, Apache Spark is the ideal tool for putting our movie recommendation system into practice.

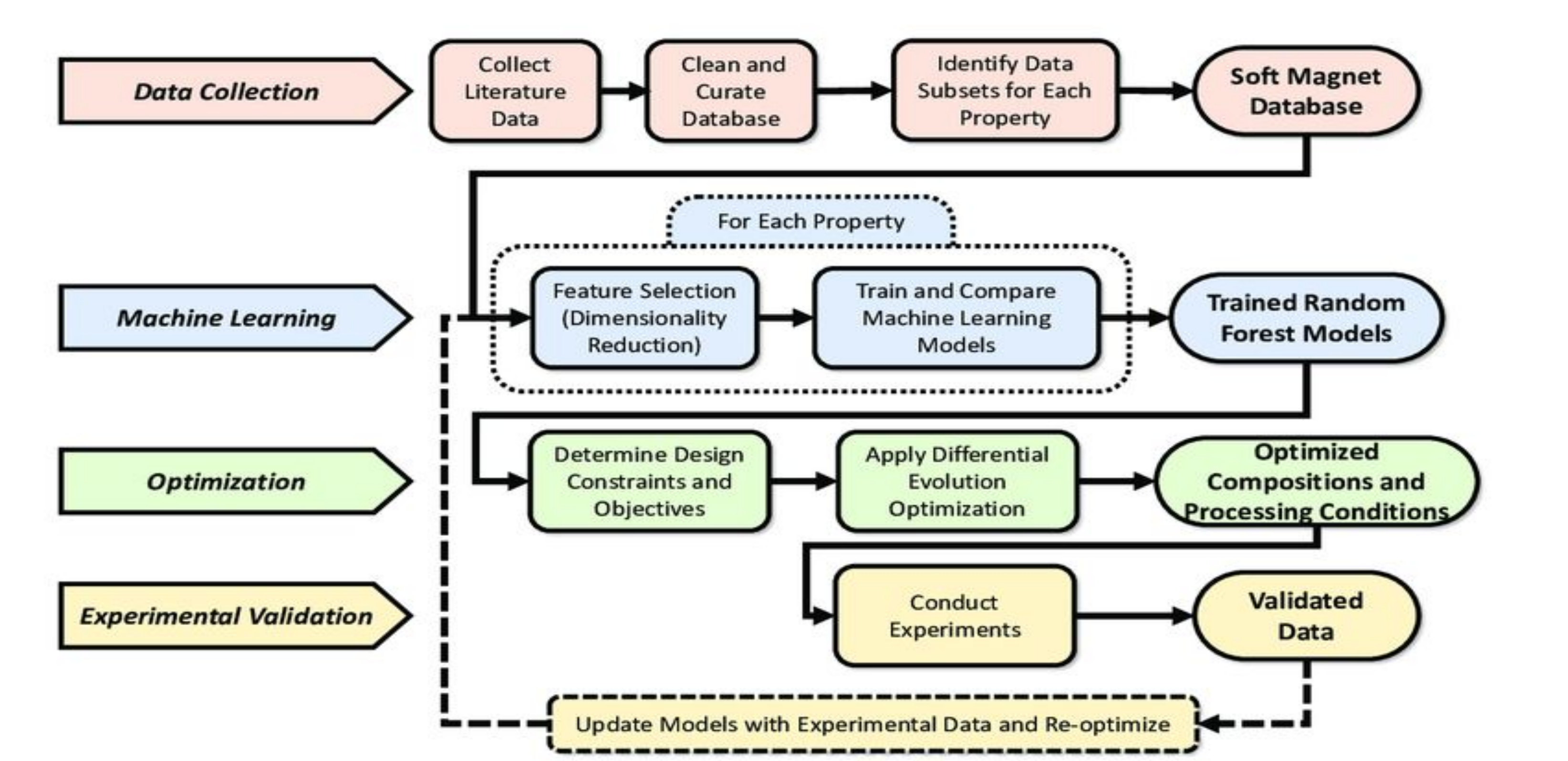


Design

For example:

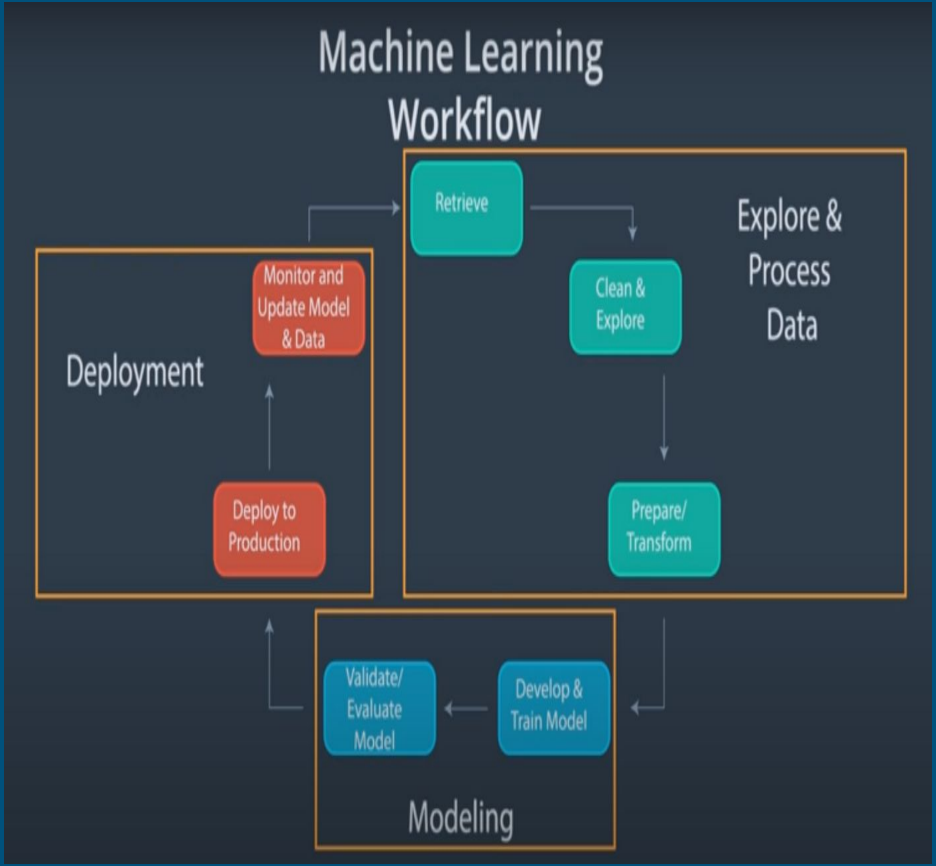
- If the user ‘X’ likes Titanic, Avatar and Elita
- While the user ‘B’ likes The Avengers, Batsman and Spiderman then they have similar interests because we know that these movies belong to the super-hero genre.
- So, there is a high probability that the user ‘A’ would like Avengers and the user ‘B’ would like The Avatar.

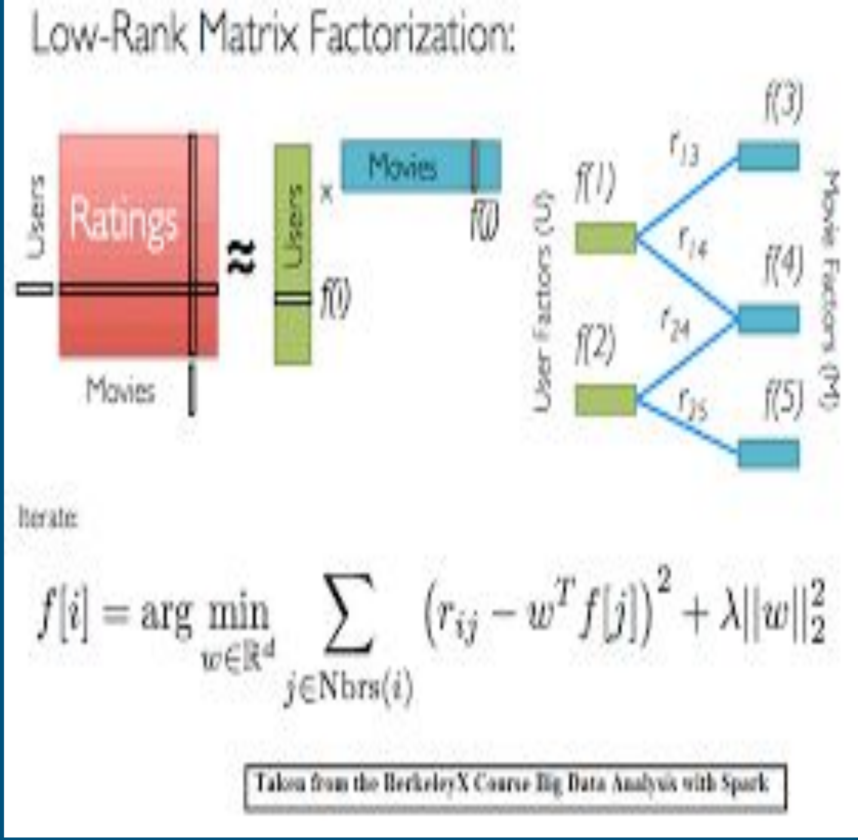
Design



Data Collection
Collect Literature Data
Clean and Curate Database
Identify Data Subsets for Each Property
Soft Magnet Database
For Each Property
Machine Learning
Feature Selection (Dimensionality Reduction)
Train and Compare Machine Learning Models
Trained Random Forest Models
Optimization
Determine Design Constraints and Objectives
Apply Differential Evolution Optimization
Optimized Compositions and Processing Conditions
Experimental Validation
Conduct Experiments
Validated Data
Update Models with Experimental Data and Re-optimize

Design


Machine Learning Workflow
Retrieve
Clean & Explore
Explore & Process Data
Prepare/ Transform
Monitor and Update Model & Data
Deployment
Deploy to Production
Validate/ Evaluate Model
Develop & Train Model
Modeling


Low-Rank Matrix Factorization:
Users
Ratings
Movies
Users
Movies
f(i)
f(j)
User Factors (U)
Movie Factors (M)
f(1)
f(2)
f(3)
f(4)
f(5)
Iterate:
$f[i] = \arg\min_{w \in \mathbb{R}^d} \sum_{j \in \mathrm{Nbrs}(i)} (r_{ij} - w^T f[j])^2 + \lambda \|w\|_2^2$
Taken from the BerkeleyX Course Big Data Analysis with Spark

Implementation

- Google Colab:

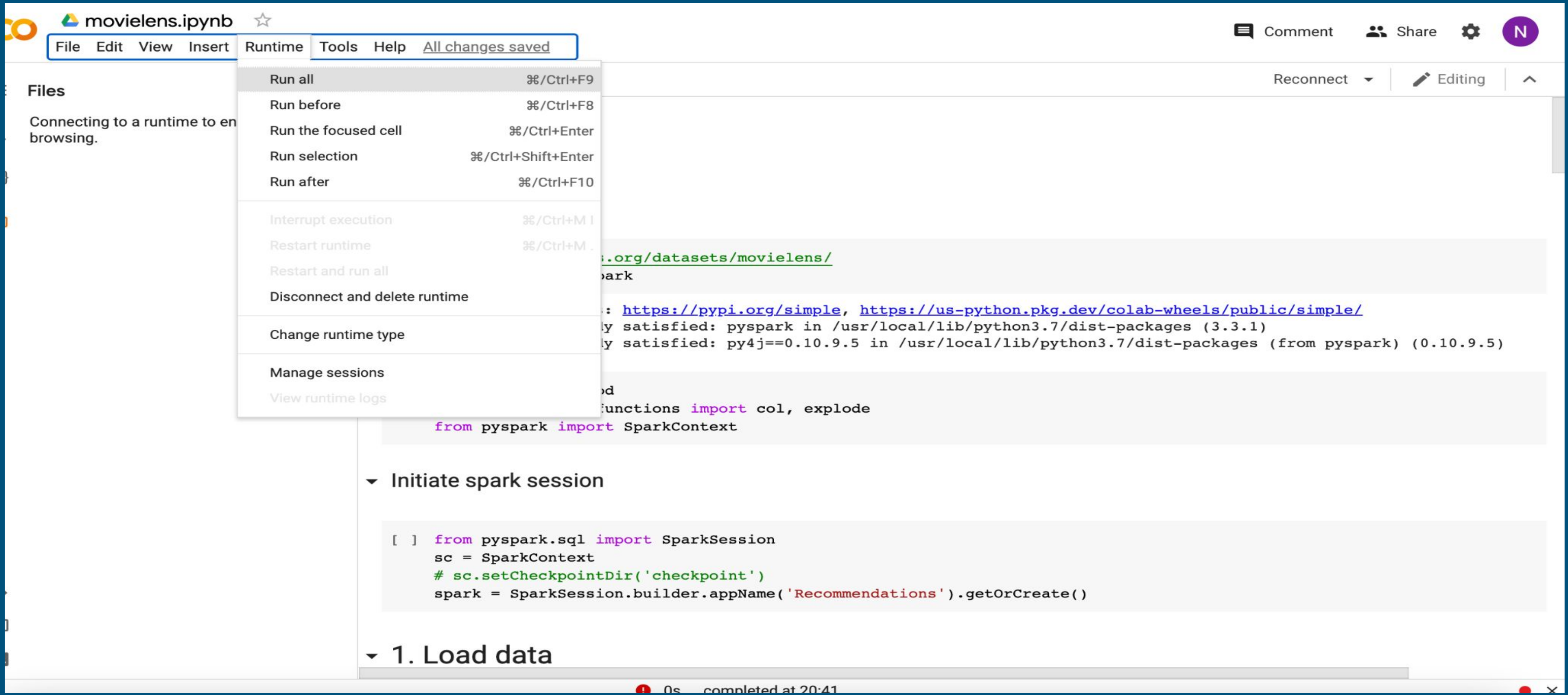
https://colab.research.google.com/notebooks/intro.ipynb#recent=true

Upload ipynb file

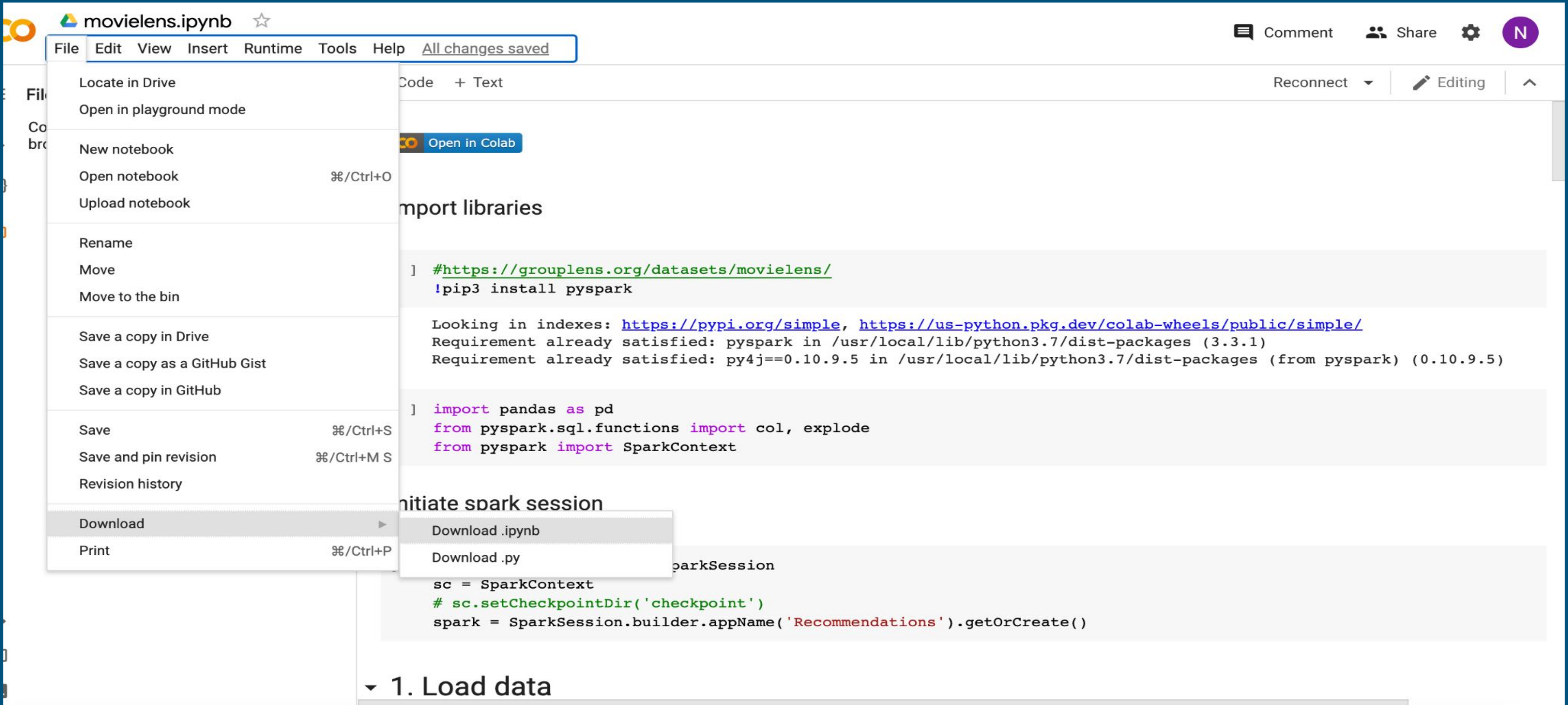
Upload
Movies.csv,
ratings.csv,
tags.csv file

Run All

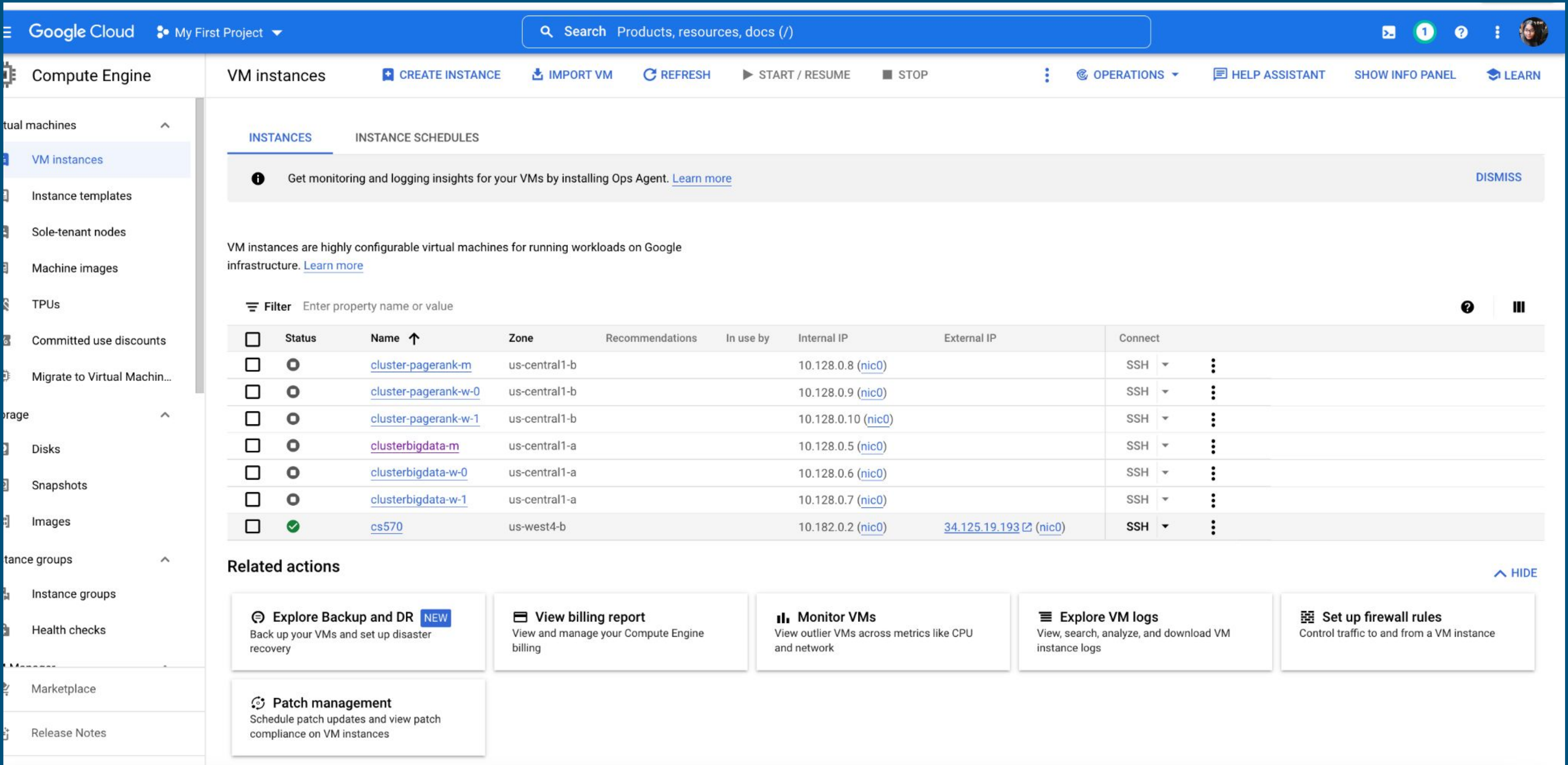
Implementation: Google Colab



Implementation: Google Colab



Implementation: Go to your google cloud Dataporc cluster



Implementation

- Publish the files movies.csv, ratings.csv and movielens.py
- Create the HDFS Directory;
- hdfs dfs -mkdir hdfs:/movielens
- Movies.csv and Ratings.csv should be copied into HDFS Directory:
- movies.csv hdfs:/movielens hdfs dfs -put
- hdfs dfs -put ratings.csv movielens ❖ Run movielens.py under spark

Implementation:

```
navyaannampelly@cluster-pagerank-m:~$ spark-submit movielens.py
22/11/21 01:27:26 INFO org.apache.spark.SparkEnv: Registering MapOutputTracker
22/11/21 01:27:26 INFO org.apache.spark.SparkEnv: Registering BlockManagerMaster
22/11/21 01:27:26 INFO org.apache.spark.SparkEnv: Registering BlockManagerMasterHeartb
22/11/21 01:27:26 INFO org.apache.spark.SparkEnv: Registering OutputCommitCoordinator
22/11/21 01:27:26 INFO org.sparkproject.jetty.util.log: Logging initialized @4138ms to
22/11/21 01:27:26 INFO org.sparkproject.jetty.server.Server: jetty-9.4.40.v20210413; b
b74; jvm 1.8.0_352-b08
22/11/21 01:27:26 INFO org.sparkproject.jetty.server.Server: Started @4277ms
22/11/21 01:27:26 INFO org.sparkproject.jetty.server.AbstractConnector: Started Server
22/11/21 01:27:27 INFO org.apache.hadoop.yarn.client.RMProxy: Connecting to ResourceMa
22/11/21 01:27:27 INFO org.apache.hadoop.yarn.client.AHSProxy: Connecting to Applicati
22/11/21 01:27:29 INFO org.apache.hadoop.conf.Configuration: resource-types.xml not fo
22/11/21 01:27:29 INFO org.apache.hadoop.yarn.util.resource.ResourceUtils: Unable to f
22/11/21 01:27:29 INFO org.apache.hadoop.yarn.client.api.impl.YarnClientImpl: Submitte
22/11/21 01:27:30 INFO org.apache.hadoop.yarn.client.RMProxy: Connecting to ResourceMa
```

Test: Google Colab

```
# Generate n Recommendations for all users
nrecommendations = best_model.recommendForAllUsers(10)
nrecommendations.limit(10).show()
```

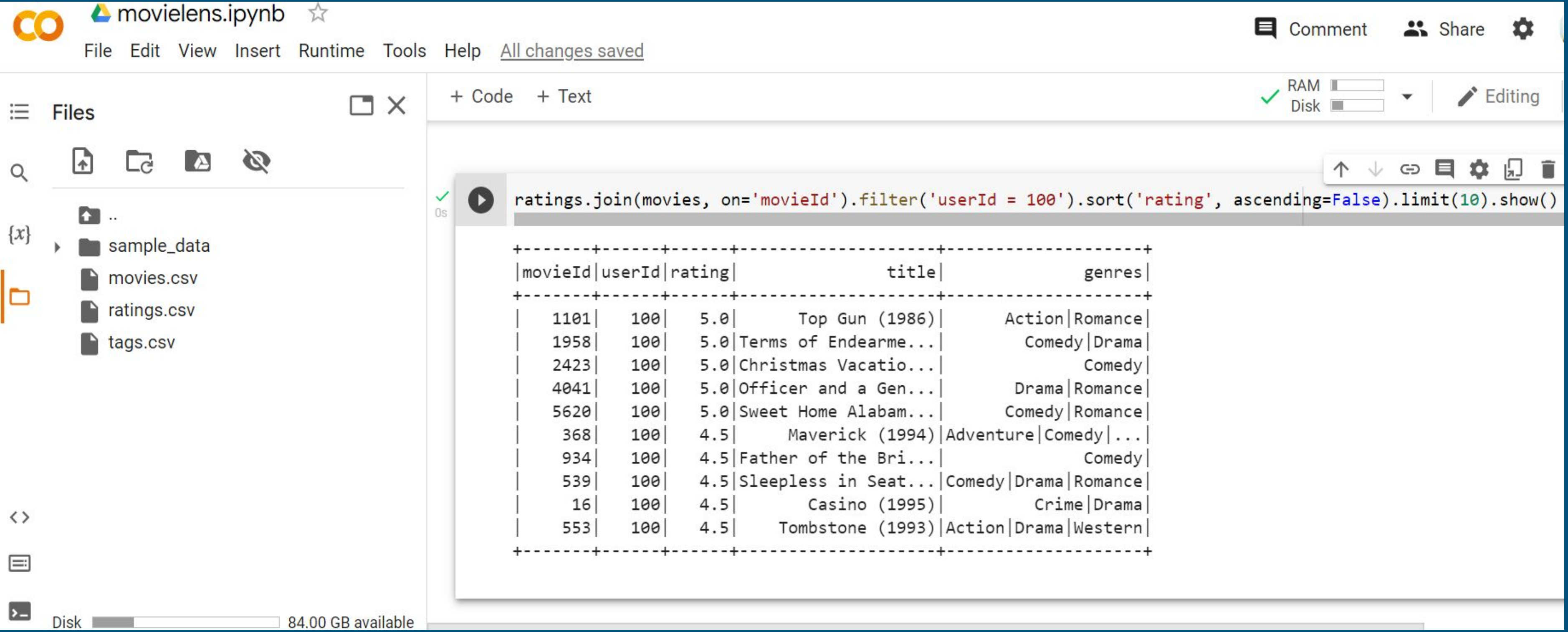
```
+------+--------------------+
|userId|     recommendations|
+------+--------------------+
|     1|[{3379, 5.729532}...|
|     3|[{5746, 4.8612}, ...|
|     5|[{3379, 4.529168}...|
|     6|[{42730, 4.758306...|
|     9|[{3379, 4.921908}...|
|    12|[{42730, 5.673235...|
|    13|[{3379, 5.06624},...|
|    15|[{3379, 4.448795}...|
|    16|[{3379, 4.6072893...|
|    17|[{3379, 5.1776514...|
+------+--------------------+
```

Test: Google Colab

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

```
+-------+------+---------+--------------------+--------------------+
|movieId|userId|   rating|               title|              genres|
+-------+------+---------+--------------------+--------------------+
|  67618|   100| 5.108419|Strictly Sexual (...|Comedy|Drama|Romance|
|  33649|   100|5.0640206| Saving Face (2004)|Comedy|Drama|Romance|
|   3379|   100|5.0374746| On the Beach (1959)|               Drama|
|  74282|   100|4.9346504|Anne of Green Gab...|Children|Drama|Ro...|
|  42730|   100|4.9183536|   Glory Road (2006)|               Drama|
|  93008|   100| 4.881788|Very Potter Seque...|      Comedy|Musical|
|  25906|   100| 4.881788|Mr. Skeffington (...|       Drama|Romance|
|  77846|   100| 4.881788| 12 Angry Men (1997)|         Crime|Drama|
|   7121|   100|4.8749967|   Adam's Rib (1949)|      Comedy|Romance|
| 171495|   100| 4.874456|              Cosmos|  (no genres listed)|
+-------+------+---------+--------------------+--------------------+
```

Test: Google Colab



Test: Google Colab

```
+------+--------------------+
|userId|     recommendations|
+------+--------------------+
|    91|[{3379, 4.9286127...|
|   601|[{3379, 5.447586}...|
|   111|[{128914, 4.82704...|
|   291|[{87234, 5.526545...|
|   581|[{3379, 5.1550307...|
|     1|[{3379, 5.7632384...|
|   223|[{33649, 4.224575...|
|   333|[{3567, 4.7874923...|
|   493|[{876, 4.8167825}...|
|    93|[{3379, 5.7609735...|
+------+--------------------+
```

Test: Google Colab

userId	movieId	rating
471	3379	4.822564
471	8477	4.6659493
471	33649	4.5504856
471	102217	4.5333
471	92494	4.5333
471	33779	4.5333
471	171495	4.527984
471	7096	4.4821672
471	84273	4.4345856
471	117531	4.4345856

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

Test: Google Colab

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

Enhancement Ideas:

- Use deep learning to implement a recommendation system for movies.
- Implement a content-based filtering movie recommendation system.
- The training dataset can be expanded in a scalable way by using data augmentation techniques. The type of data will determine whether data augmentation techniques are used.

Conclusion:

- The technique for suggesting movies has enormous potential. For certain people, movie recommendations have been fairly accurate, and movie titles have been successfully clustered based on their plot summaries.
- After learning about recommendation engines theoretically, we first looked at the movie lens dataset.
- Then, after learning how to use MLlib to implement collaborative filtering, we divided the dataset into training and testing sets for the transform

References:

- https://hc.labnet.sfbu.edu/~henry/npu/classes/mllib/collaborative_filtering/PySpark_Recommender_System_with_ALS.pdf
- https://towardsdatascience.com/build-recommendation-system-with-pyspark-using-alternating-least-squares-als-matrix-factorisation-ebe1ad2e7679
- https://medium.com/edureka/spark-mllib-e87546ac268


ThaNK
YOU