



Movie Recommendation with MLlib

CS570: Big-data Processing and Analytics

SFBU
Navya Kandimalla



Table Of Contents

- Introduction
- Design
- Implementation
- Test
- Enhancement Ideas
- Conclusion
- References



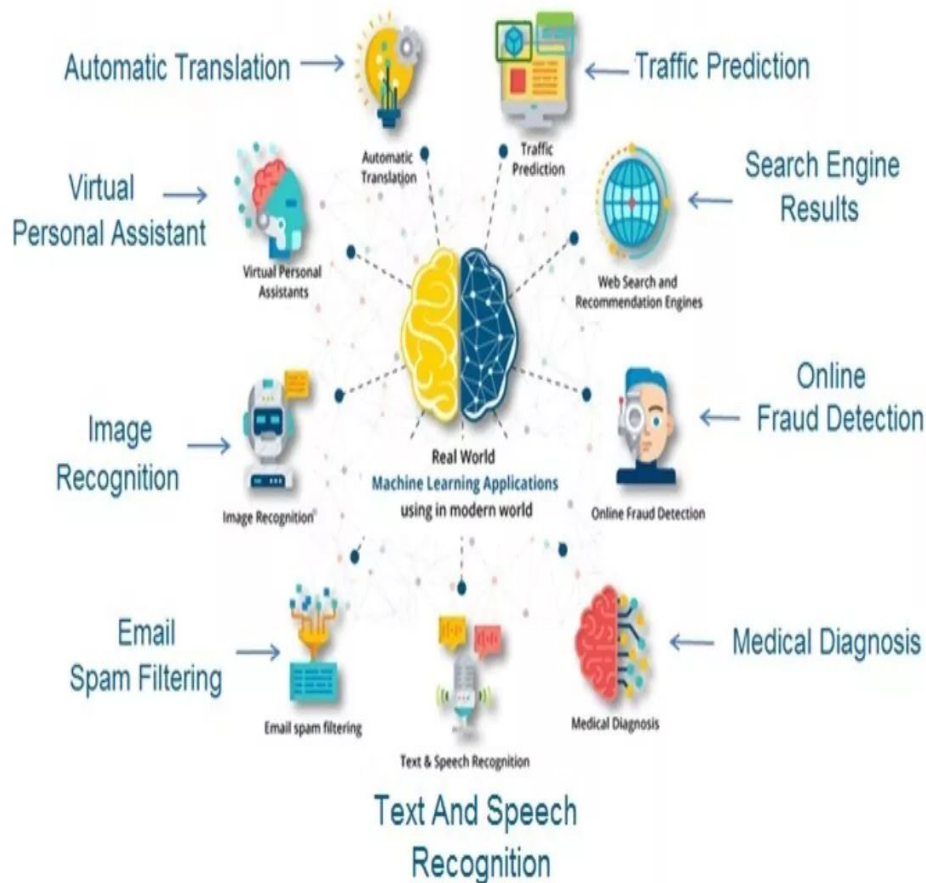
Introduction

- 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:

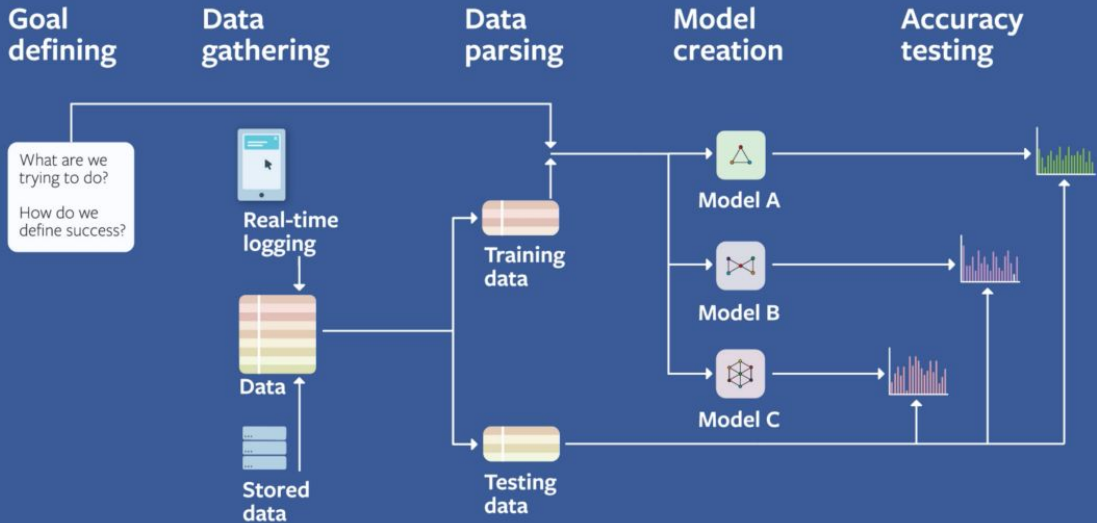
- Traffic Alerts.
- Social Media.
- Transportation and Commuting.
- Products Recommendations.
- Virtual Personal Assistants.
- Self Driving Cars.
- Dynamic Pricing.
- Google Translate.

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.

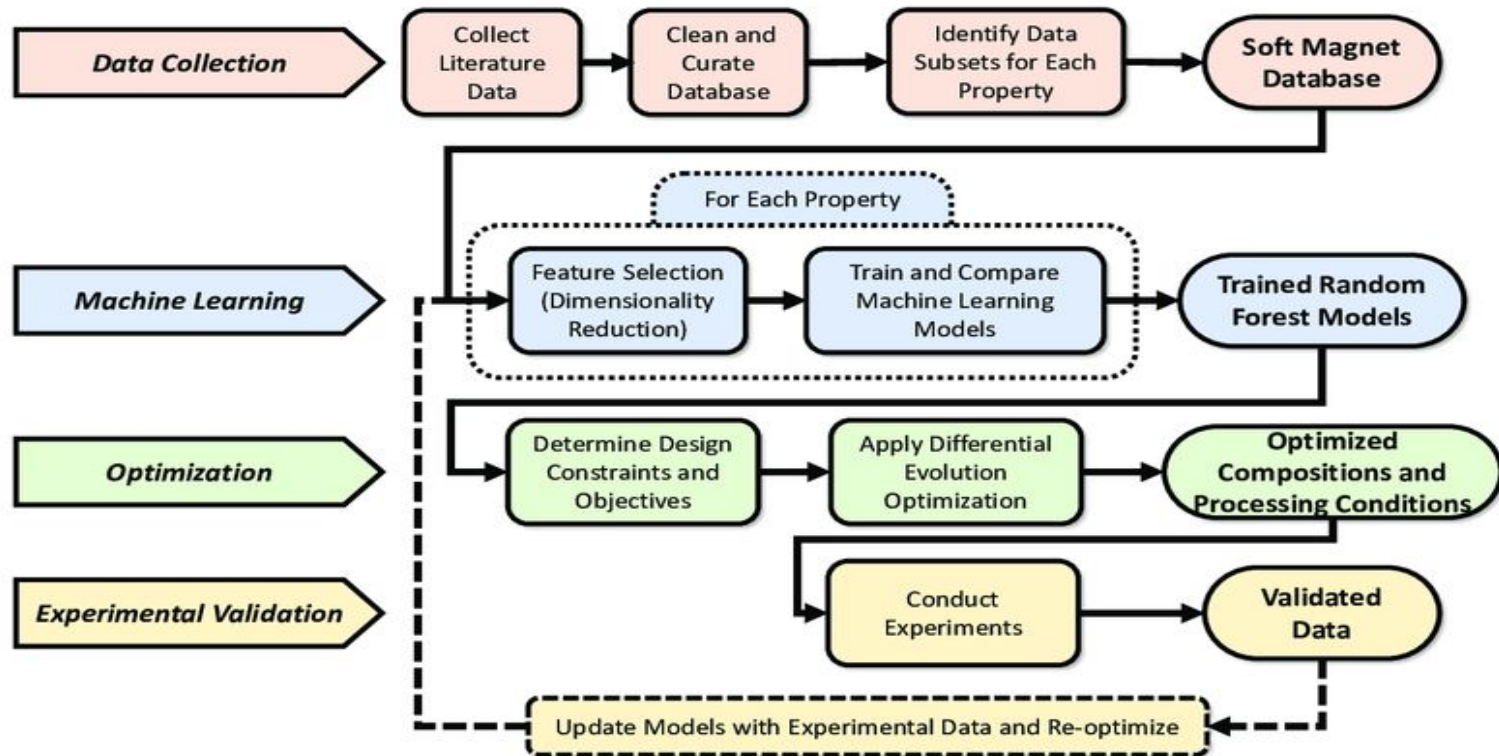


Design

For example:

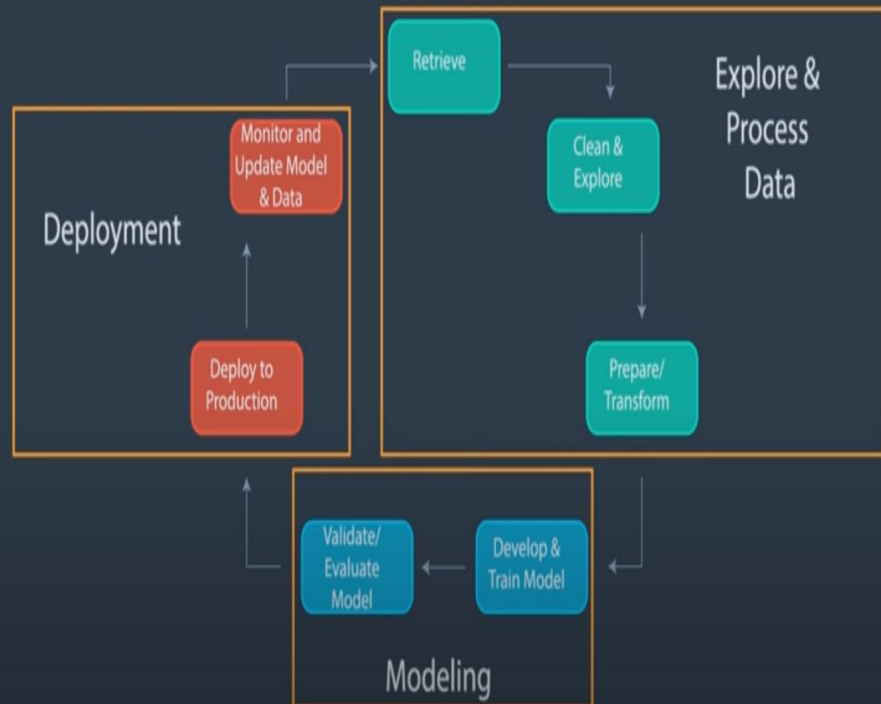
- If the user 'X' likes Titanic, Avatar and Elita
- While the user 'B' likes The Avengers, Batman and Spiderman then they have similar interests because we know that these movies belong to the super-hero genre.
- So, there is a high probability that the user 'A' would like Avengers and the user 'B' would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

Design

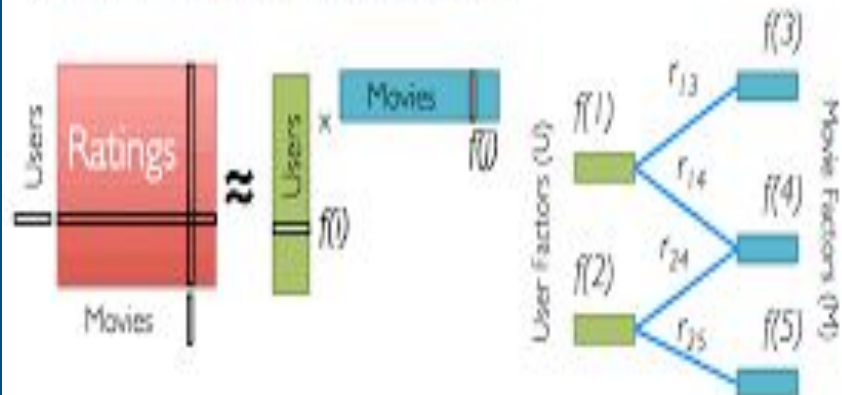


Design

Machine Learning Workflow



Low-Rank Matrix Factorization:



Iterate:

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

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

The screenshot shows a Google Colab notebook interface. The top bar includes the notebook title 'movielens.ipynb', a star icon, and navigation tabs: File, Edit, View, Insert, Runtime, Tools, and Help. A status bar on the right indicates 'All changes saved'. On the left, a 'Files' sidebar shows a message: 'Connecting to a runtime to enable browsing.' The 'Runtime' menu is open, displaying options such as 'Run all' (⌘/Ctrl+F9), 'Run before' (⌘/Ctrl+F8), 'Run the focused cell' (⌘/Ctrl+Enter), 'Run selection' (⌘/Ctrl+Shift+Enter), 'Run after' (⌘/Ctrl+F10), 'Interrupt execution' (⌘/Ctrl+M), 'Restart runtime' (⌘/Ctrl+M), 'Restart and run all', 'Disconnect and delete runtime', 'Change runtime type', 'Manage sessions', and 'View runtime logs'. The main editor area contains two code cells. The first cell contains a URL: <https://pypi.org/simple>, <https://us-python.pkg.dev/colab-wheels/public/simple/>, and a command: `pyspark in /usr/local/lib/python3.7/dist-packages (3.3.1)`. The second cell contains a command: `py4j==0.10.9.5 in /usr/local/lib/python3.7/dist-packages (from pyspark) (0.10.9.5)`. Below the code cells, there are two sections: 'Initiate spark session' and '1. Load data'. The 'Initiate spark session' section contains a code block:

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

Implementation: Google Colab

movielens.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

Locate in Drive
Open in playground mode
New notebook
Open notebook ⌘/Ctrl+O
Upload notebook
Rename
Move
Move to the bin
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

Code + Text

Reconnect Editing

Open in Colab

import libraries

```
1 #https://grouplens.org/datasets/movielens/
!pip3 install pyspark
```

Looking in indexes: <https://pypi.org/simple>, <https://us-python.pkg.dev/colab-wheels/public/simple/>
Requirement already satisfied: pyspark in /usr/local/lib/python3.7/dist-packages (3.3.1)
Requirement already satisfied: py4j==0.10.9.5 in /usr/local/lib/python3.7/dist-packages (from pyspark) (0.10.9.5)

```
1 import pandas as pd
from pyspark.sql.functions import col, explode
from pyspark import SparkContext
```

initiate spark session

```
sparkSession
sc = SparkContext
# sc.setCheckpointDir('checkpoint')
spark = SparkSession.builder.appName('Recommendations').getOrCreate()
```

1. Load data

Implementation: Go to your google cloud Dataproc cluster

The screenshot shows the Google Cloud console interface for VM instances. The left sidebar contains navigation links for Compute Engine, VM instances, Instance templates, Sole-tenant nodes, Machine images, TPUs, Committed use discounts, Migrate to Virtual Machine, Disks, Snapshots, Images, Instance groups, Health checks, Marketplace, and Release Notes. The main content area is titled 'VM instances' and includes buttons for 'CREATE INSTANCE', 'IMPORT VM', 'REFRESH', 'START / RESUME', and 'STOP'. Below these buttons is a section for 'INSTANCES' and 'INSTANCE SCHEDULES'. A message states: 'Get monitoring and logging insights for your VMs by installing Ops Agent. [Learn more](#)'. Below this, a description reads: 'VM instances are highly configurable virtual machines for running workloads on Google infrastructure. [Learn more](#)'. A filter bar is present above a table of VM instances. The table has columns: Status, Name, Zone, Recommendations, In use by, Internal IP, External IP, and Connect. The instances listed are: cluster-pagerank-m, cluster-pagerank-w-0, cluster-pagerank-w-1, clusterbigdata-m, clusterbigdata-w-0, clusterbigdata-w-1, and cs570. Below the table, there is a 'Related actions' section with five cards: 'Explore Backup and DR', 'View billing report', 'Monitor VMs', 'Explore VM logs', and 'Set up firewall rules'. A 'Patch management' card is also visible at the bottom left.

Google Cloud My First Project

Search Products, resources, docs (/)

Compute Engine VM instances

CREATE INSTANCE IMPORT VM REFRESH START / RESUME STOP OPERATIONS HELP ASSISTANT SHOW INFO PANEL LEARN

INSTANCES INSTANCE SCHEDULES

Get monitoring and logging insights for your VMs by installing Ops Agent. [Learn more](#) DISMISS

VM instances are highly configurable virtual machines for running workloads on Google infrastructure. [Learn more](#)

Filter Enter property name or value

Status	Name	Zone	Recommendations	In use by	Internal IP	External IP	Connect
<input type="checkbox"/>	cluster-pagerank-m	us-central1-b			10.128.0.8 (nic0)		SSH
<input type="checkbox"/>	cluster-pagerank-w-0	us-central1-b			10.128.0.9 (nic0)		SSH
<input type="checkbox"/>	cluster-pagerank-w-1	us-central1-b			10.128.0.10 (nic0)		SSH
<input type="checkbox"/>	clusterbigdata-m	us-central1-a			10.128.0.5 (nic0)		SSH
<input type="checkbox"/>	clusterbigdata-w-0	us-central1-a			10.128.0.6 (nic0)		SSH
<input type="checkbox"/>	clusterbigdata-w-1	us-central1-a			10.128.0.7 (nic0)		SSH
<input checked="" type="checkbox"/>	cs570	us-west4-b			10.182.0.2 (nic0)	34.125.19.193 (nic0)	SSH

Related actions

Explore Backup and DR NEW
Back up your VMs and set up disaster recovery

View billing report
View and manage your Compute Engine billing

Monitor VMs
View outlier VMs across metrics like CPU and network

Explore VM logs
View, search, analyze, and download VM instance logs

Set up firewall rules
Control traffic to and from a VM instance

Patch management
Schedule patch updates and view patch compliance on VM instances

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.RMPProxy: Connecting to ResourceMa
22/11/21 01:27:27 INFO org.apache.hadoop.yarn.client.AHSPProxy: 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.RMPProxy: Connecting to ResourceMa
```



Test: Google Colab

✓
8s



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

✓
3s

```
[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

The screenshot shows a Google Colab notebook interface. The top bar includes the Colab logo, the notebook title 'movielens.ipynb', and a star icon. Below this is a menu bar with 'File', 'Edit', 'View', 'Insert', 'Runtime', 'Tools', 'Help', and a link to 'All changes saved'. On the right side of the top bar are icons for 'Comment', 'Share', and 'Settings'. Below the menu bar is a toolbar with '+ Code' and '+ Text' buttons. On the right side of the toolbar are status indicators for 'RAM' and 'Disk' usage, and an 'Editing' button. The left sidebar shows a file explorer with a search bar and icons for file operations. The file list includes '..' (parent directory), 'sample_data' (a folder), and three CSV files: 'movies.csv', 'ratings.csv', and 'tags.csv'. The main area of the notebook contains a code cell with a play button icon and a status indicator '0s'. The code in the cell is: `ratings.join(movies, on='movieId').filter('userId = 100').sort('rating', ascending=False).limit(10).show()`. The output of the code is a pandas DataFrame displayed in a table format. The table has five columns: 'movieId', 'userId', 'rating', 'title', and 'genres'. The data is sorted by rating in descending order, showing the top 10 movies for user 100. The genres are listed as a pipe-separated string for each movie.

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

Disk 84.00 GB available

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-ebelad2e7679>
- <https://medium.com/edureka/spark-mllib-e87546ac268>

THANK
YOU