Cloud-based Data Analytics

Lecture 4

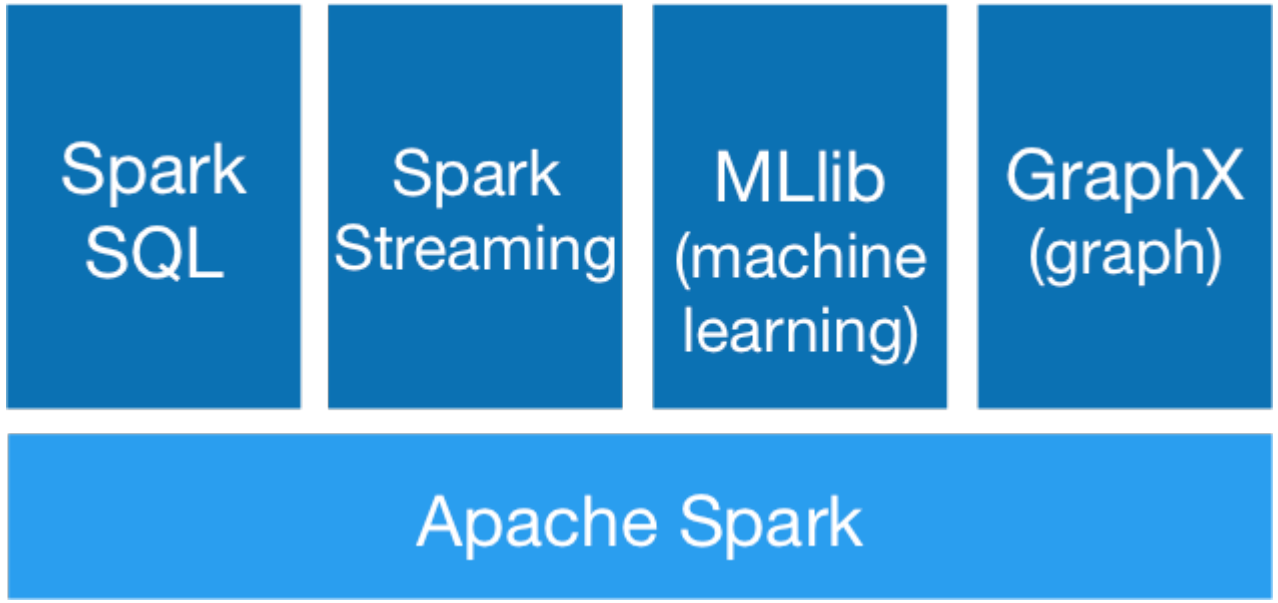
Content

- Spark Components
- Recommender Systems
- Collaborative Filtering based Recommender Systems
- ALS Algorithm
- Databricks

Apache Spark Components

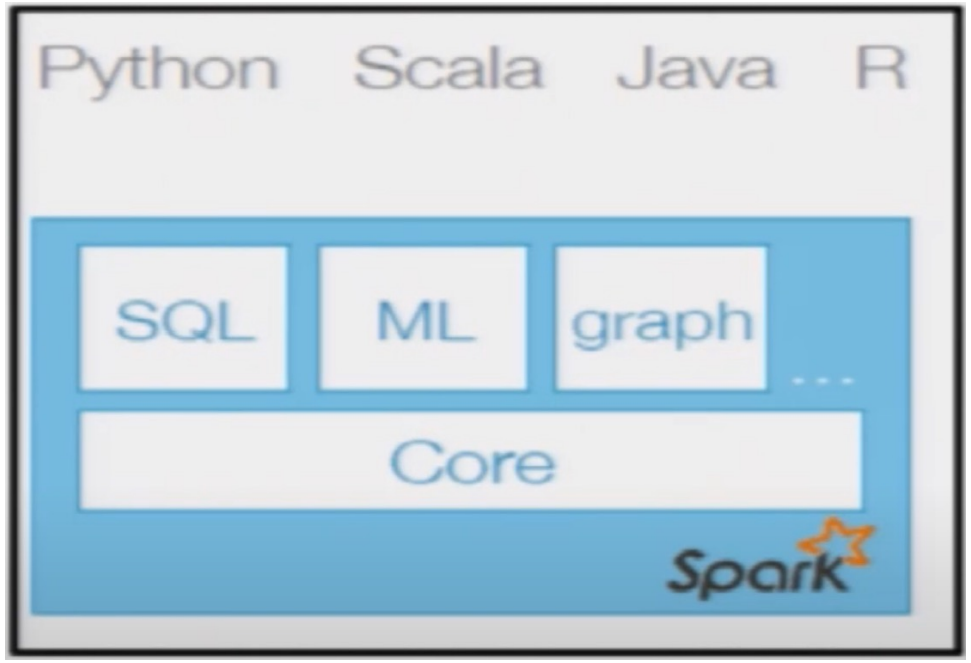
Introduction

- Apache Spark is fast and general-purpose cluster computing system for large scale data processing
- High-level APIs in Java, Scala, Python and R

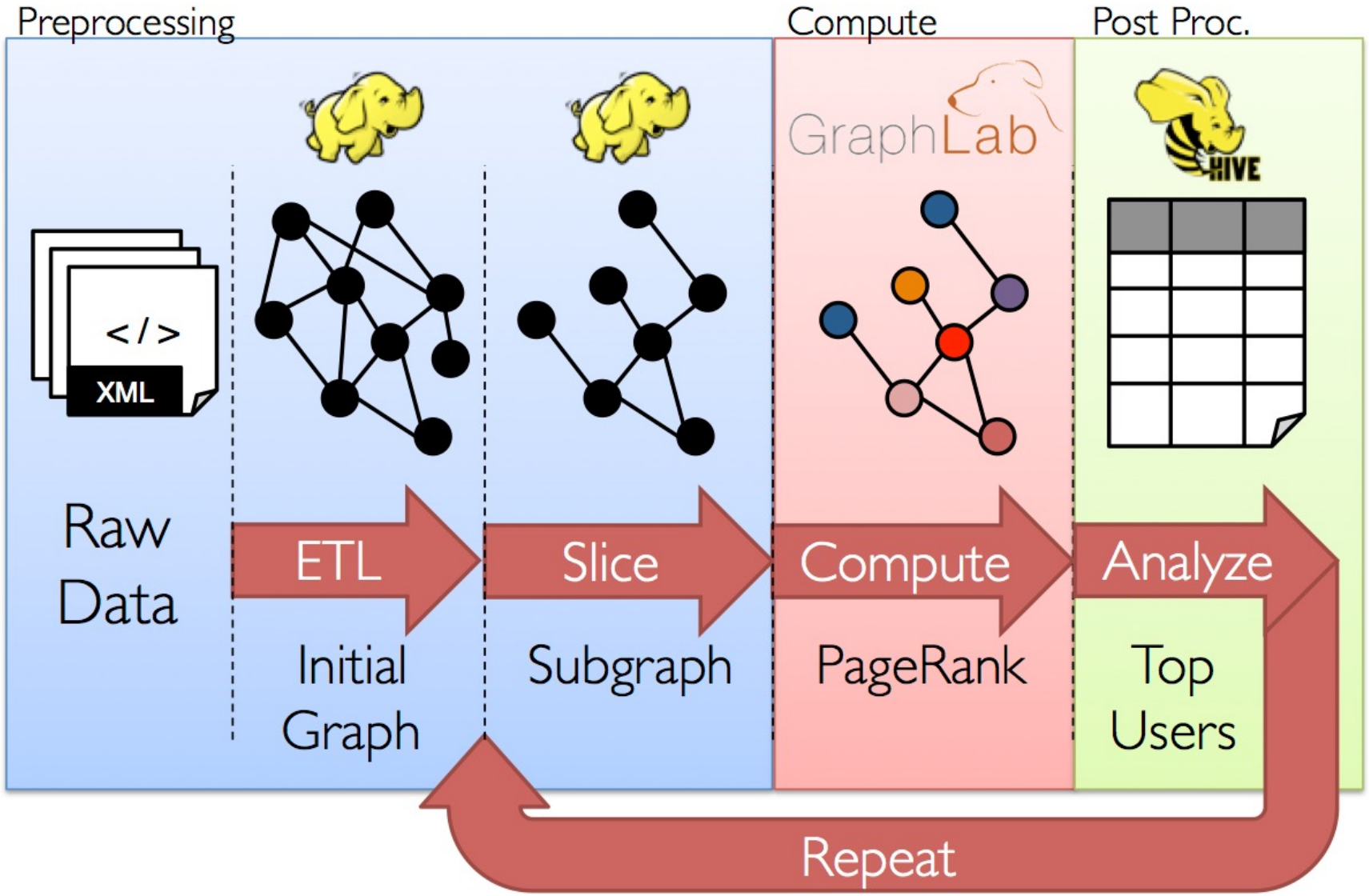


Standard library for Big Data

- Big data apps lack libraries of common algorithms
- Spark's generality + support for multiple languages make it suitable to offer this
- Much of future activity will be in these libraries



GraphX


Preprocessing
Compute
Post Proc.
GraphLab
HIVE
</>
XML
Raw Data
ETL
Slice
Compute
Analyze
Initial Graph
Subgraph
PageRank
Top Users
Repeat

GraphX

GENERAL GRAPH
PROCESSING LIBRARY

BUILD GRAPH USING RDDS
OF NODES AND EDGES

LARGE LIBRARY OF GRAPH
ALGORITHMS WITH
COMPOSABLE STEPS

GraphX

- Structured Prediction
  - Loopy Belief Propagation, Max-Product Linear Programs, Gibbs Sampling
- Semi-Supervised ML
  - Graph SSL, CoEM
- Community Detection
  - Triangle-Counting, K-core Decomposition, K-Truss
- Graph Analytics
  - PageRank, Personalized PageRank, Shortest Path, Graph Coloring
- Classification
  - Neural Networks

Spark Streaming

- Large scale streaming
- Ensure exactly one semantics
- Integrated with Spark -> unifies batch, interactive, and streaming computations!

Spark SQL

- Enables loading and querying structured data in Spark
- From Hive:

```
c = HiveContext(sc)
rows = c.sql(“select text, year from hivetable”)
rows.filter(lambda r: r.year >2013).collect()
```

- From JSON

```
c.jsonFile(“tweets.json”).registerAsTable(“tweets”)
c.sql(“select text, user.name from tweets”)
```

Machine Learning Library (MLlib)

- **MLlib Algorithms**
  - *Classification*: Naïve Bayes, Decision Trees, Logistic Regression, Linear SVM
  - *Regression*: Generalized Linear Models (GLM's), Regression tree
  - *Collaborative Filtering*: Alternating Least Squares (ALS), Non-negative Matrix Factorization (NMF)
  - *Clustering*: K-Means
  - *Decomposition*: SVD, PCA
  - *Optimization*: SGD (Stochastic Gradient Descent), L-BFGS

PySpark with MLlib - Naïve bayes

```
from pyspark.ml.classification import NaiveBayes
from pyspark.ml.evaluation import MulticlassClassificationEvaluator
from pyspark.sql import SparkSession
spark = SparkSession.builder.appName('rec').getOrCreate()

# Load training data
data = spark.read.format("libsvm").load("sample_libsvm_data.txt")
# Split the data into train and test
splits = data.randomSplit([0.6, 0.4], 1234)
train = splits[0]
test = splits[1]
# create the trainer and set its parameters
nb = NaiveBayes(smoothing=1.0, modelType="multinomial")
# train the model
model = nb.fit(train)
# select example rows to display.
predictions = model.transform(test)
predictions.show(30)
# compute accuracy on the test set
evaluator = MulticlassClassificationEvaluator(labelCol="label", predictionCol="prediction",
                                              metricName="accuracy")
accuracy = evaluator.evaluate(predictions)
print("Test set accuracy = " + str(accuracy))
```

PySpark with MLlib - Naïve bayes

Run: navie_bayes

```
+-----+--------------------+--------------------+----------+----------+
|label|            features|       rawPrediction|probability|prediction|
+-----+--------------------+--------------------+----------+----------+
|  0.0|(692,[95,96,97,12...|[-172664.79564650...|  [1.0,0.0]|       0.0|
|  0.0|(692,[98,99,100,1...|[-176279.15054306...|  [1.0,0.0]|       0.0|
|  0.0|(692,[122,123,124...|[-189600.55409526...|  [1.0,0.0]|       0.0|
|  0.0|(692,[124,125,126...|[-274673.88337431...|  [1.0,0.0]|       0.0|
|  0.0|(692,[124,125,126...|[-183393.03869049...|  [1.0,0.0]|       0.0|
```

Run: navie_bayes

```
|  1.0|(692,[125,126,127...|[-82975.247866607...|  [0.0,1.0]|       1.0|
|  1.0|(692,[126,127,128...|[-120943.63701145...|  [0.0,1.0]|       1.0|
|  1.0|(692,[126,127,128...|[-82901.559516310...|  [0.0,1.0]|       1.0|
|  1.0|(692,[127,128,129...|[-135090.62334761...|  [0.0,1.0]|       1.0|
|  1.0|(692,[127,128,129...|[-110448.10961252...|  [0.0,1.0]|       1.0|
|  1.0|(692,[127,128,154...|[-60341.201337307...|  [0.0,1.0]|       1.0|
+-----+--------------------+--------------------+----------+----------+
only showing top 30 rows

Test set accuracy = 1.0
```

Summary

We have seen Spark Components

PySpark implementation for Naive Bayes Theorem

Recommender Systems

Content

- Introductory concepts
- Framework for understanding recommender system

Recommender Systems

Some popular examples

35 %
Revenue due to RS

33.3 %
Increase in monthly subscriptions thanks to RS

60 %
Amount of clicks due to recommendations

23.7 %
Increase in revenue after adopting RS

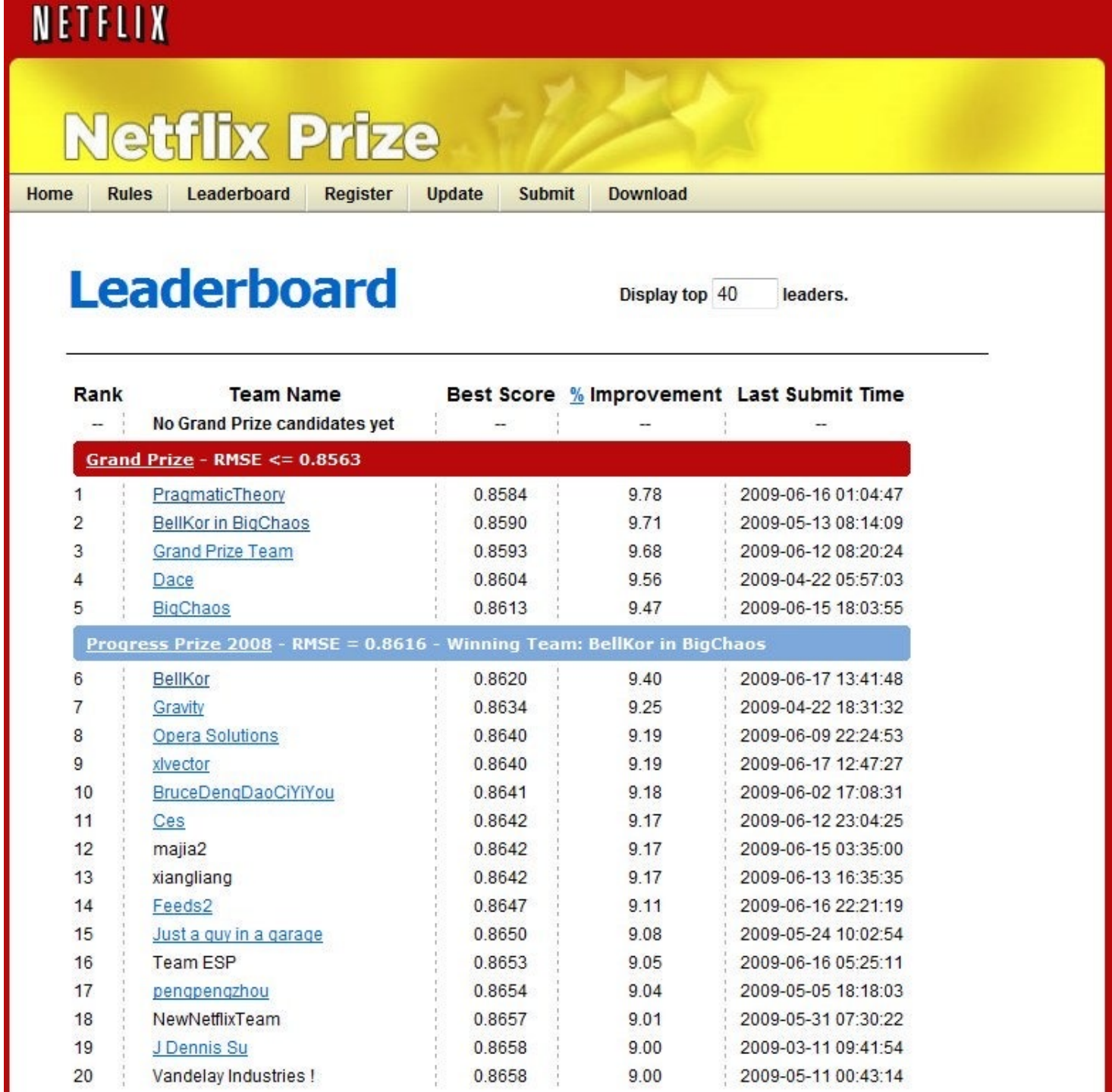
amazon

Spotify

YouTube

BEST BUY

Netflix Prize



NETFLIX

Netflix Prize

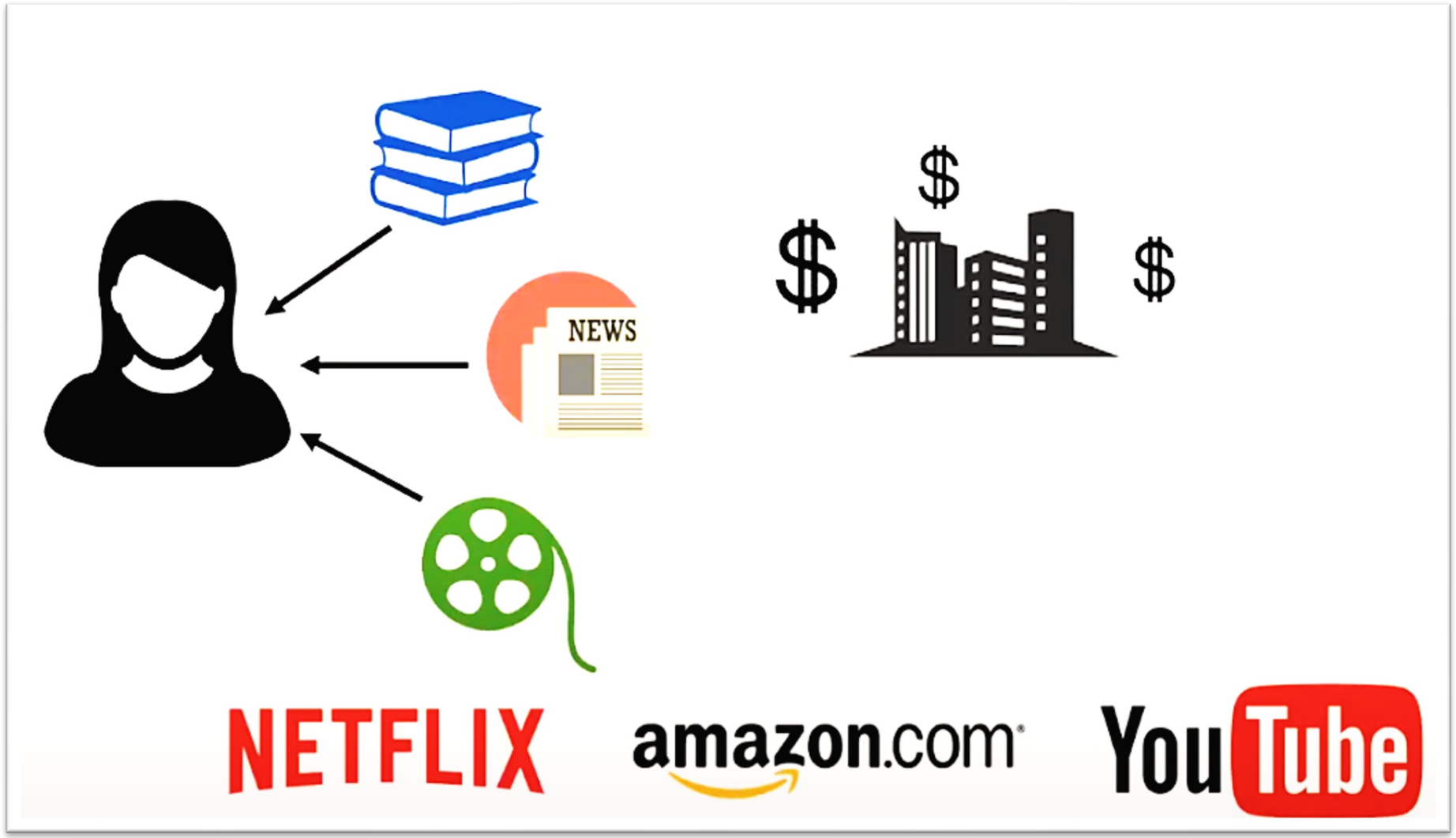
Home | Rules | Leaderboard | Register | Update | Submit | Download

Leaderboard

Display top 40 leaders.

Rank	Team Name	Best Score	% Improvement	Last Submit Time
--	No Grand Prize candidates yet	--	--	--
Grand Prize - RMSE <= 0.8563				
1	PragmaticTheory	0.8584	9.78	2009-06-16 01:04:47
2	BellKor in BigChaos	0.8590	9.71	2009-05-13 08:14:09
3	Grand Prize Team	0.8593	9.68	2009-06-12 08:20:24
4	Dace	0.8604	9.56	2009-04-22 05:57:03
5	BigChaos	0.8613	9.47	2009-06-15 18:03:55
Progress Prize 2008 - RMSE = 0.8616 - Winning Team: BellKor in BigChaos				
6	BellKor	0.8620	9.40	2009-06-17 13:41:48
7	Gravity	0.8634	9.25	2009-04-22 18:31:32
8	Opera Solutions	0.8640	9.19	2009-06-09 22:24:53
9	xlvector	0.8640	9.19	2009-06-17 12:47:27
10	BruceDengDaoCiYiYou	0.8641	9.18	2009-06-02 17:08:31
11	Ces	0.8642	9.17	2009-06-12 23:04:25
12	majia2	0.8642	9.17	2009-06-15 03:35:00
13	xiangliang	0.8642	9.17	2009-06-13 16:35:35
14	Feeds2	0.8647	9.11	2009-06-16 22:21:19
15	Just a guy in a garage	0.8650	9.08	2009-05-24 10:02:54
16	Team ESP	0.8653	9.05	2009-06-16 05:25:11
17	pengpengzhou	0.8654	9.04	2009-05-05 18:18:03
18	NewNetflixTeam	0.8657	9.01	2009-05-31 07:30:22
19	J Dennis Su	0.8658	9.00	2009-03-11 09:41:54
20	Vandelay Industries !	0.8658	9.00	2009-05-11 00:43:14

Recommender Systems



What do recommender system do?

- Recommender systems suggest items (information, products and services) that are of interest to the users based on customer demographic, features of the items, user preferences (e.g., rating or purchase history) etc.

- Ex. Book recommendation, movie recommendation, news recommendation

Advantages

- Effective Information management
  - Decreased information overload
- Better customer relationship management
  - Increased sales
  - Loyal customers
- Important in e-commerce sites
  - Effective Management of virtual customers
  - Personalized Service for the customers

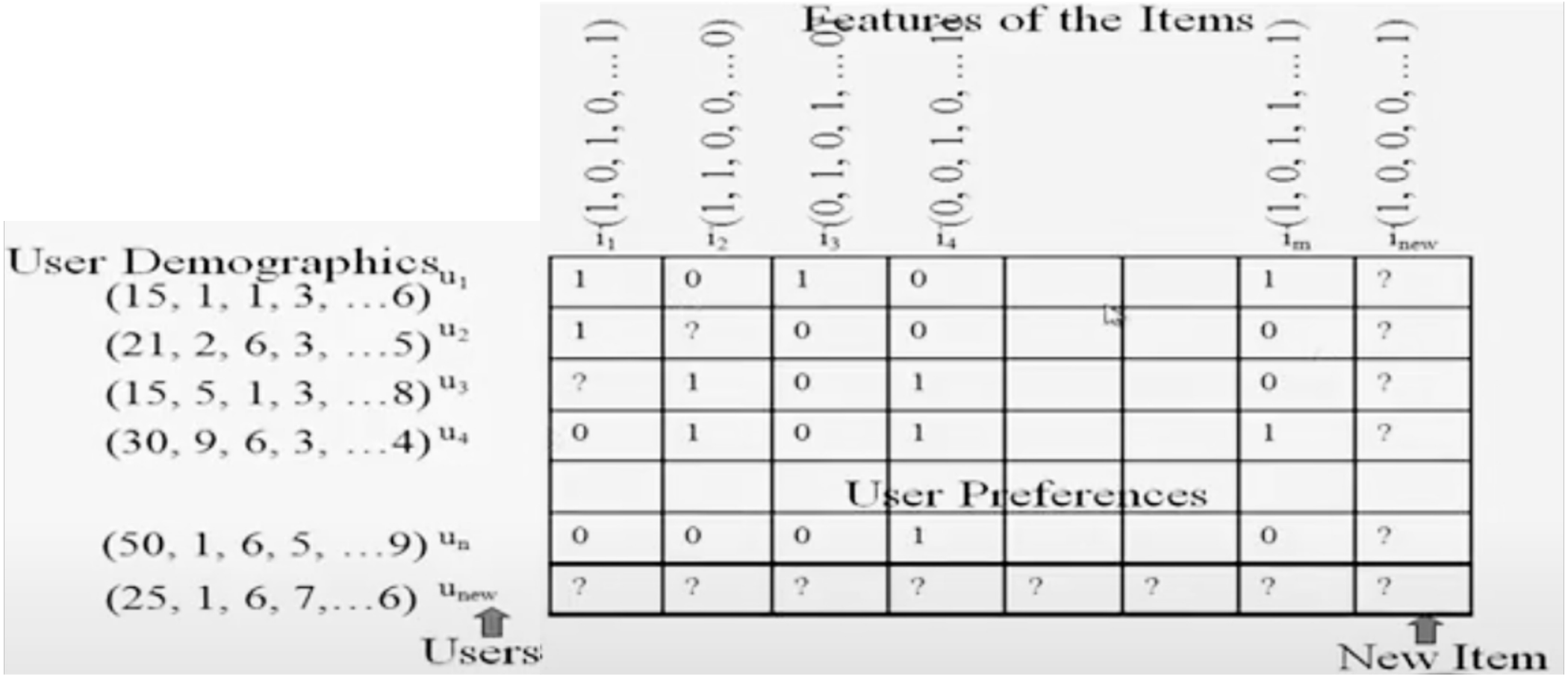
Some early research initiatives

- Social information filtering: algorithms for automating “word of mouth”
  - Recommending music albums
  - Upendra Shardanand and Pattie Maes
- GroupLens
  - Personalized recommendations for usenet news items
  - Paul Resnick and others
- CiteSeer
  - Recommendation for relevant articles
  - C. Lees Giles and others

Some early commercial implementations

- Double-click.com
  - Personalized banner ads
- Cdnow.com
  - Music albums
- Amazon.com
  - Books

Framework to understand Recommender Systems



Implicit vs explicit data

	Explicit Rating Data	**Implicit Rating Data**
Source	User questionnaire, Online survey, Offline survey, User review, etc.	User activities, System states and variables, Consumed time, etc.
Cost to collect	Expensive	Cheap

Elements in a typical recommendation scenario

Users U ($u_1, u_2 \ldots u_n$)

- Each user is associated with his demographic data
- Each user has a list of items I_{ui} ($I_{ui} \in I$) on which a user has expressed his preferences
- The preferences of user u_i on the item i_j (denoted as p_{ij}) is a subjective rating explicitly stated by the user or an implicit measure inferred from the purchase, navigation, browsing and searching pattern of the user.

Items I ($i_1, i_2 \ldots i_n$)

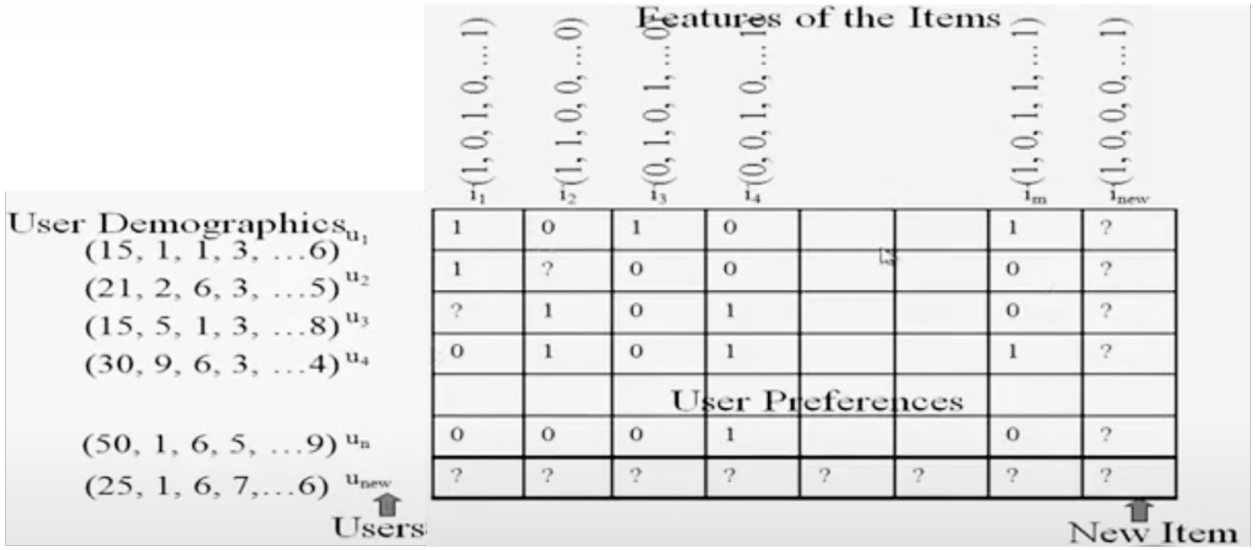
- Each item is associated with a set of features

Types of Recommendations Decisions

Prediction
- Predicting preferences for an item $i_j \notin I_{ua}$ for an active user u_a
- Can be further classified as personalized or non-personalized

Top-N recommendations
- Recommending a list of N items, $I_r \subset I$, that the active user u_a will like most. Recommended list must be on the items not already rated or chosen by u_a, that is $I_r \cap I_{ua} = \varnothing$
- Can be further classified as personalized or non-personalized

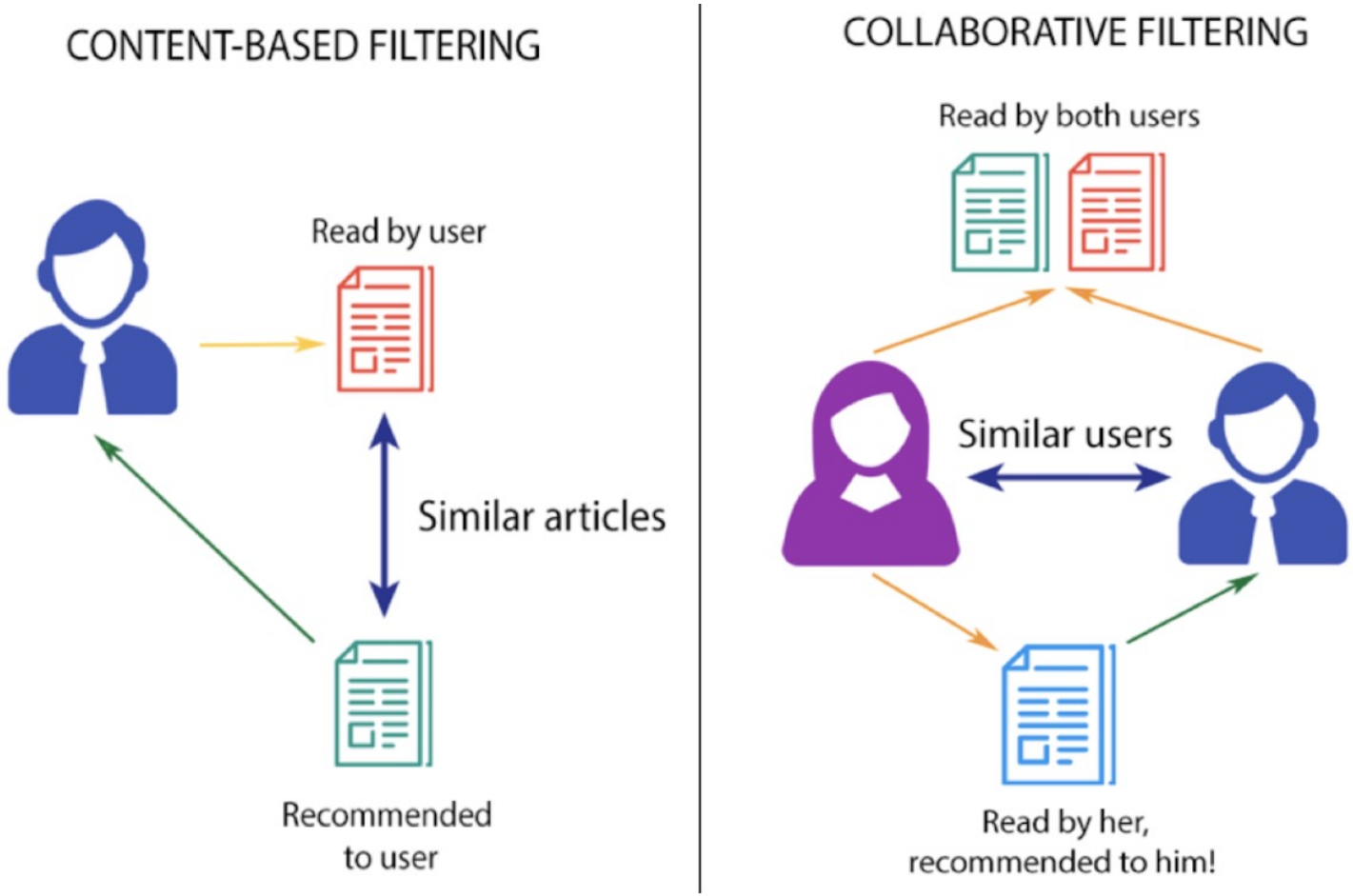
Top-M users
- Recommending a list of M users for a newly available item i_{new} who will value i_{new} most.



Types of Recommender Systems

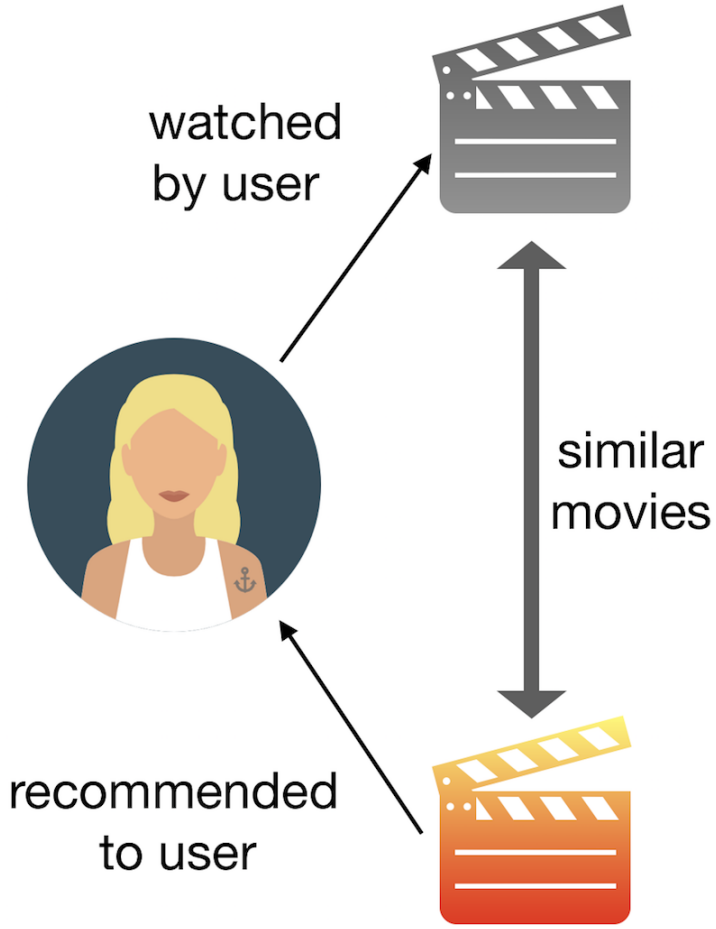
- Content based
- Collaborative based
- Hybrid
- Graph based
- Knowledge based
- Demographics based

Types of Recommender Systems



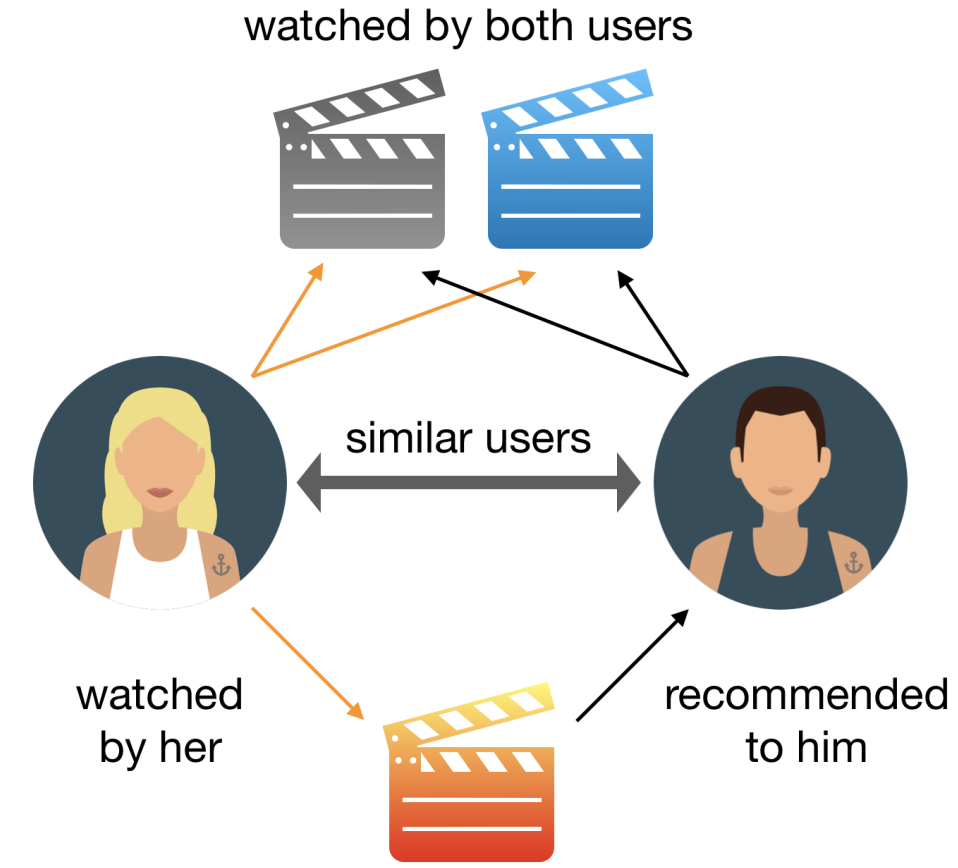
Content based recommendation system

- Content based information filtering
  - Degree of relevance to a particular user of an item determined by its content
- Information used
  - Features of items
  - Individual user preferences
- Type of recommendation decision
  - Prediction
  - Top-N recommendations
  - Top-M users
- Personalized



Collaborative based Recommender System

- User to user correlation based on taste
  - Social Information filtering
- Information used
  - User preferences
- Type of recommendation decision
  - Prediction
  - Top-N recommendations
- Personalized



Demographics based recommendation system

- User-to-user correlation-based demographics
- Information used
  - Individual user preferences
  - Features of the items
- Type of recommendation decision
  - Prediction
  - Top-N recommendations
  - Top-M users
- Personalized

Recommender Systems

S. No.	Techniques	Advantages	Disadvantages
1.	Collaborative filtering	Serendipitous and diverse recommendations, no item cold-start problem, no domain knowledge required	User cold-start problem, data sparsity, grey sheep problem, synonymy, shilling attack, scalability, quality based on ratings
2.	Content filtering	User-independent, transparent, new recommendations, no item cold-start	Overspecialization, user feedback is essential, user cold-start problem
3.	Hybrid	Makes system robust, improve performance	Knowledge of combining techniques for a particular domain
4.	Graph	Easy to find similar users and similar products	Ranking of results, modelling of graph
5.	Knowledge	No cold-start problem, no need of ratings	Need of knowledge acquisition, static recommendations
6.	Demographic	No need of ratings, improvement in recommendations with time	Privacy issue, low accuracy, lack of availability of demographic data, new user problem

Summary

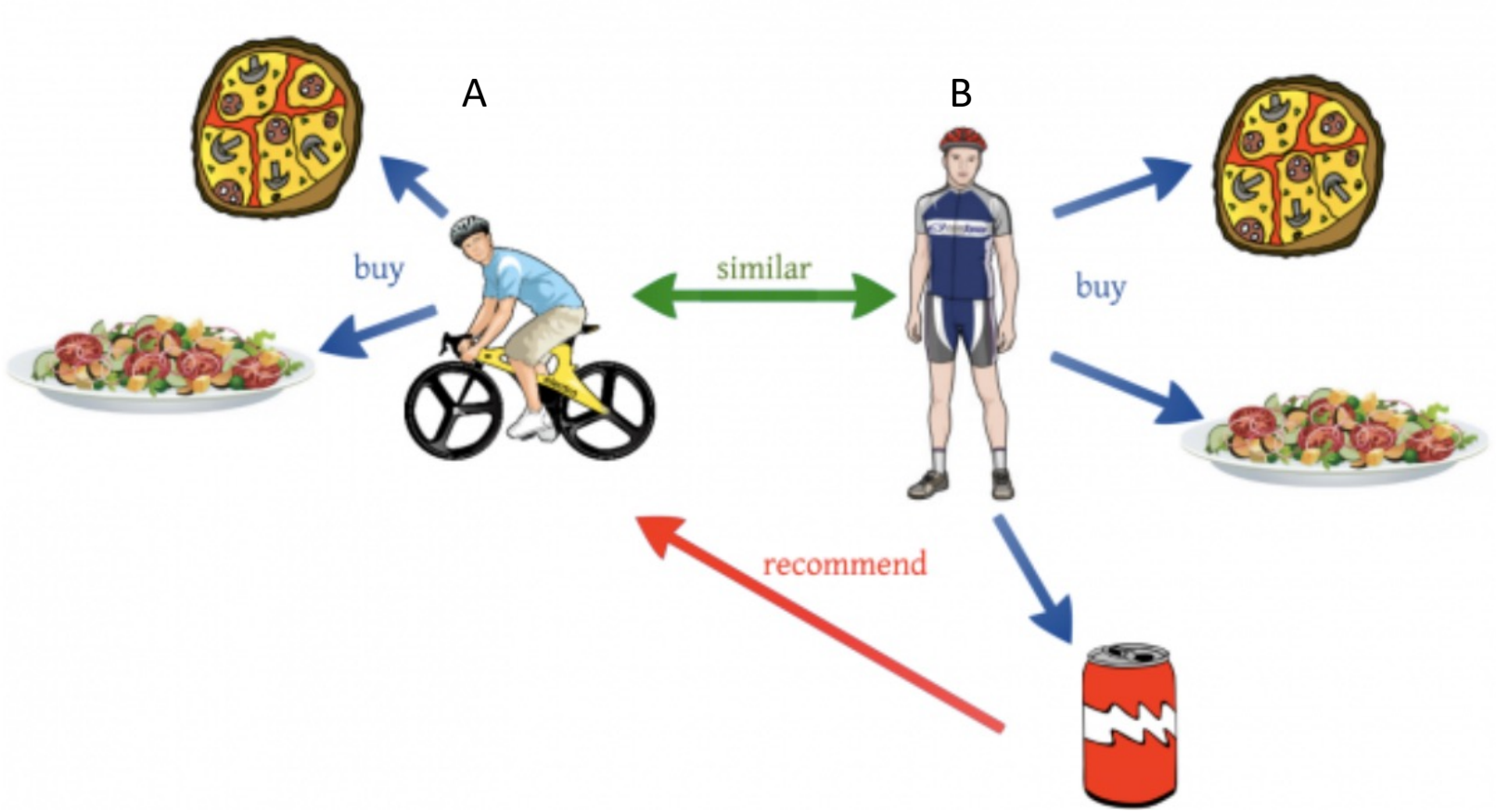
WE HAVE EXPLORED RECOMMENDER SYSTEMS

TYPES OF RECOMMENDER SYSTEMS

ADVANTAGES AND DISADVANTAGES OF RECOMMENDER SYSTEMS

Collaborative Filtering Approach

Collaborative Filtering

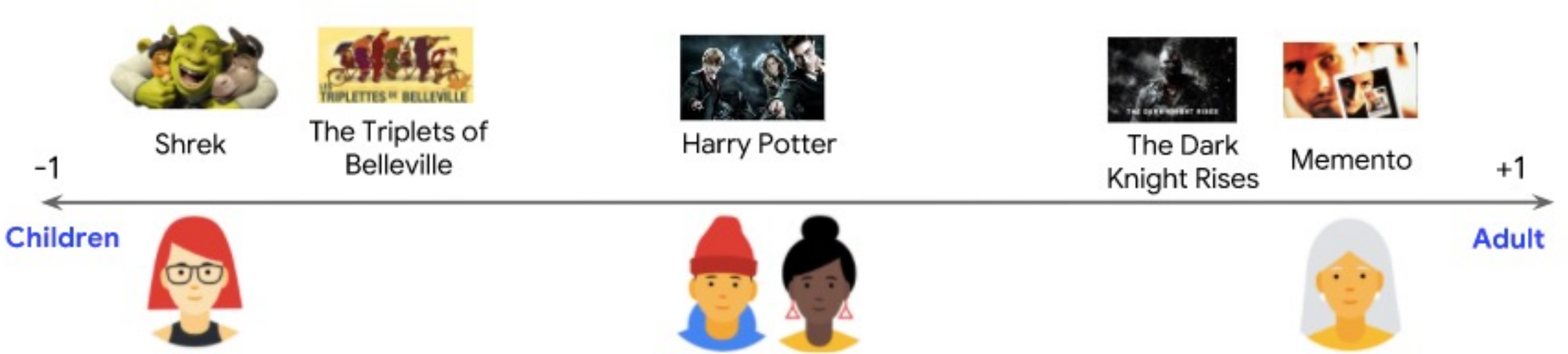


A Movie Recommendation Example

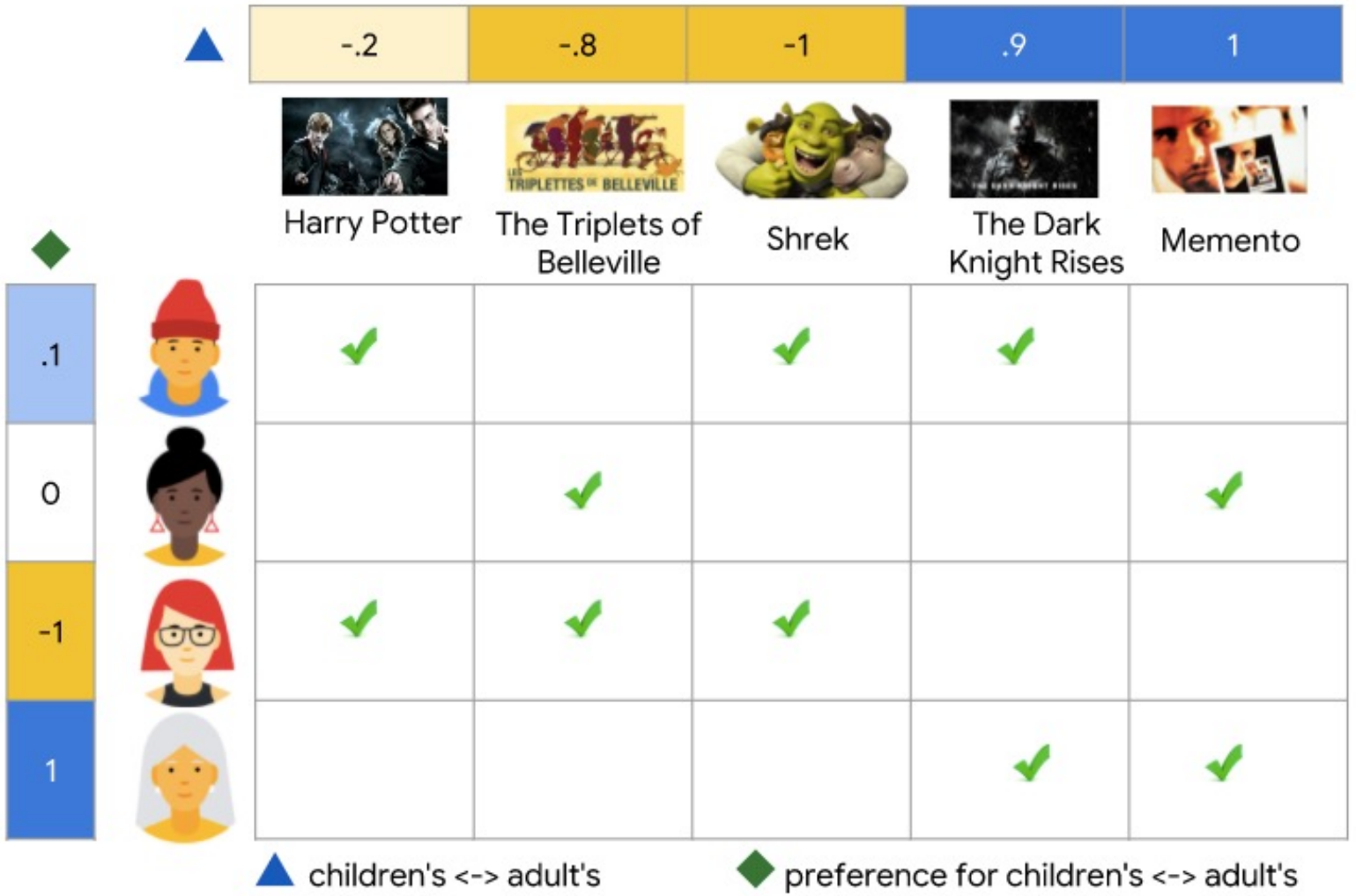
- Consider a movie recommendation system in which the training data consists of a feedback matrix in which:
  - Each row represents a user.
  - Each column represents an item (a movie).

Movie	Rating	Description
The Dark Knight Rises	PG-13	Batman endeavors to save Gotham City from nuclear annihilation in this sequel to The Dark Knight, set in the DC Comics universe.
Harry Potter and the Sorcerer's Stone	PG	A orphaned boy discovers he is a wizard and enrolls in Hogwarts School of Witchcraft and Wizardry, where he wages his first battle against the evil Lord Voldemort.
Shrek	PG	A lovable ogre and his donkey sidekick set off on a mission to rescue Princess Fiona, who is emprisoned in her castle by a dragon.
The Triplets of Belleville	PG-13	When professional cycler Champion is kidnapped during the Tour de France, his grandmother and overweight dog journey overseas to rescue him, with the help of a trio of elderly jazz singers.
Memento	R	An amnesiac desperately seeks to solve his wife's murder by tattooing clues onto his body.

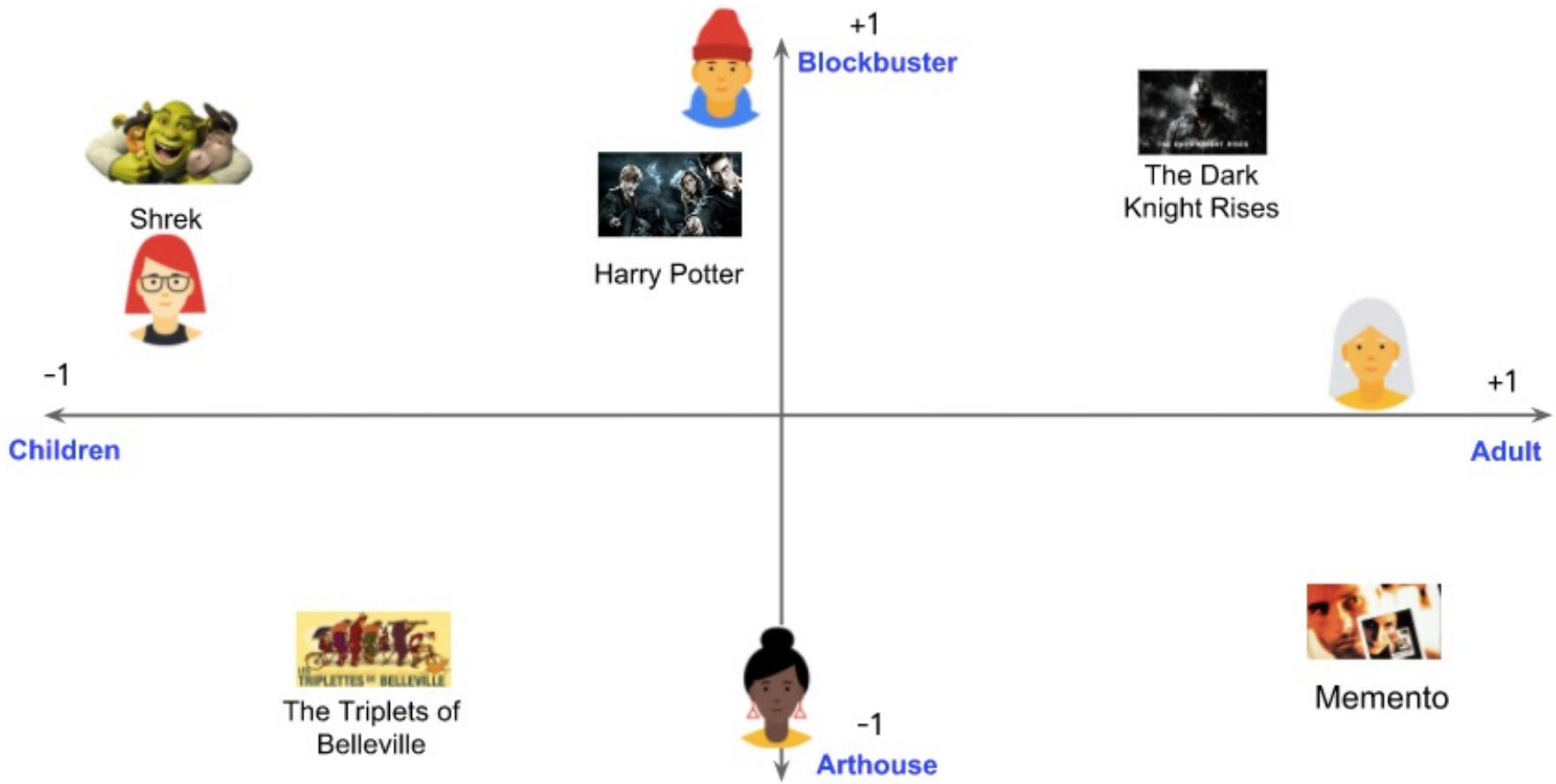
1D Embedding



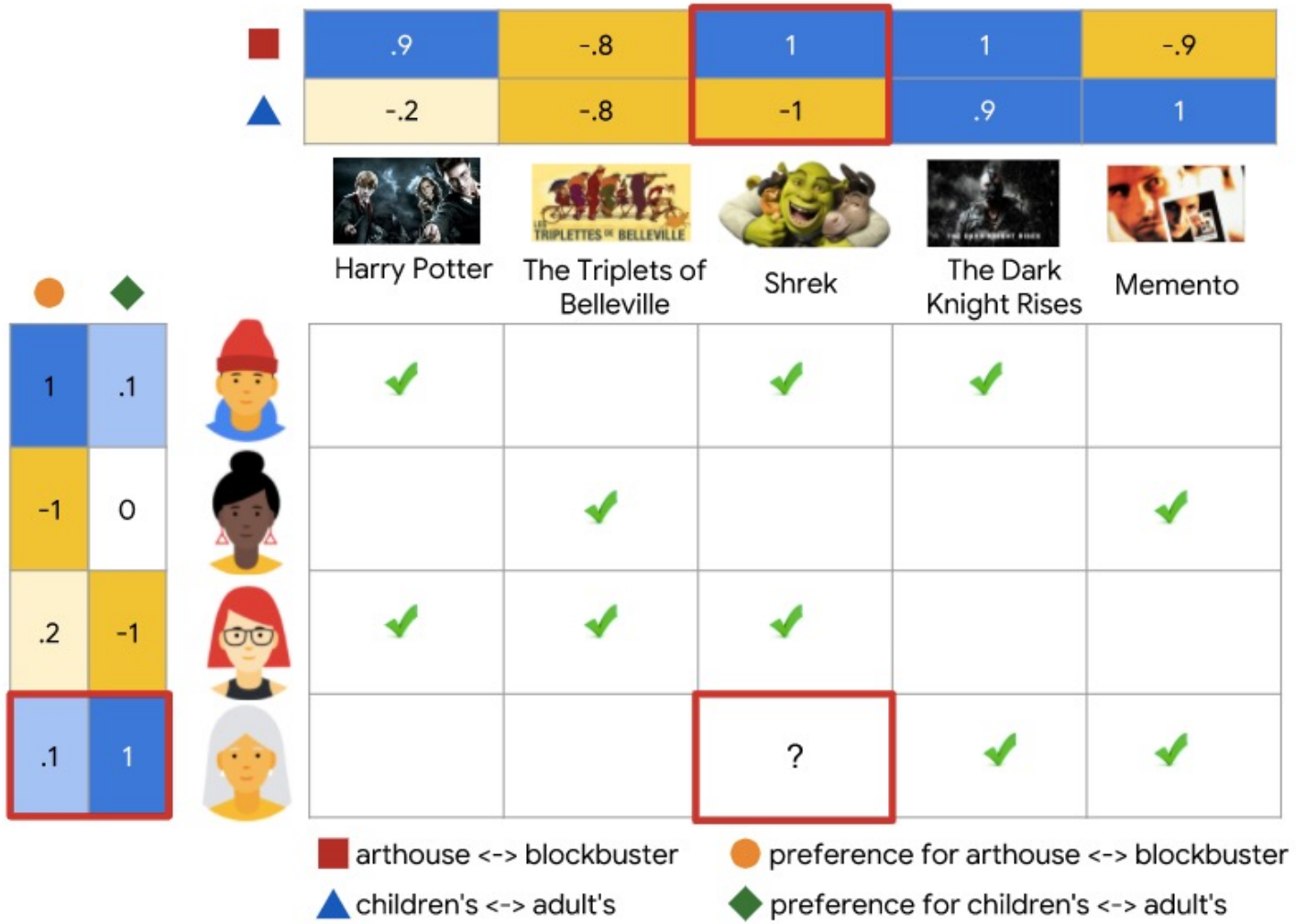
1D Embedding



2D Embedding



2D Embedding



Collaborative Filtering

Objective: Recommend Movies to User 3

	User 1	User 2	User 3
Movie 1	5	2	4
Movie 2	3	4	3
Movie 3	1	4	1
Movie 4	2		??
Movie 5	5	2	??

Approach 1: Find Similar Users and recommend movies that they like

Collaborative Filtering

Objective: Recommend Movies to User 3

	Movie 1	Movie 2	Movie 3	Movie 4	Movie 5
User 1	5	3	1	2	5
User 2	2	4	4		2
User 3	4	3	1	??	??

Approach 2: Find similar movies based on ratings given by other users.

Question

Given a set of ratings from past, recommend a movie for User 4

	User 1	User 2	User 3	User 4
Movie 1	5	2	4	1
Movie 2	3	4	3	5
Movie 3	1	4	1	4
Movie 4	2	1		
Movie 5	5	2	4	
Movie 6	3	5	3	

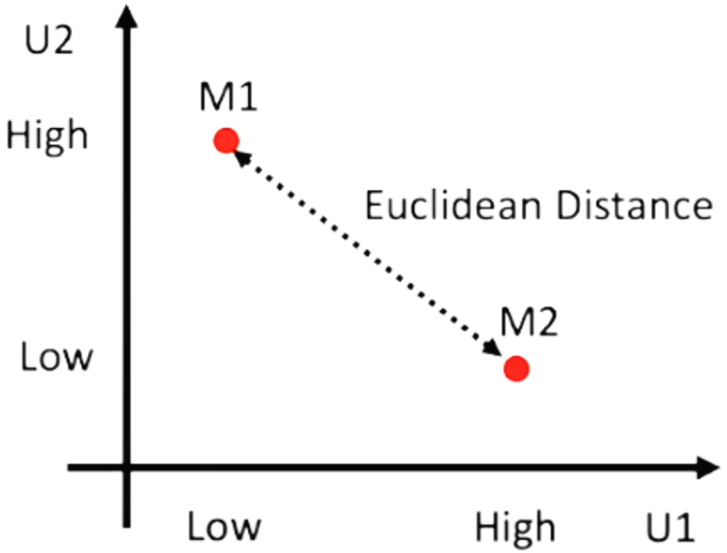
Question

Given a set of ratings from past, recommend a movie for User 4

	User 1	User 2	User 3	User 4
Movie 1	5	2	4	1
Movie 2	3	4	3	5
Movie 3	1	4	1	4
Movie 4	2	1		
Movie 5	5	2	4	
Movie 6	3	5	3	

Solution: Recommend Movie 6

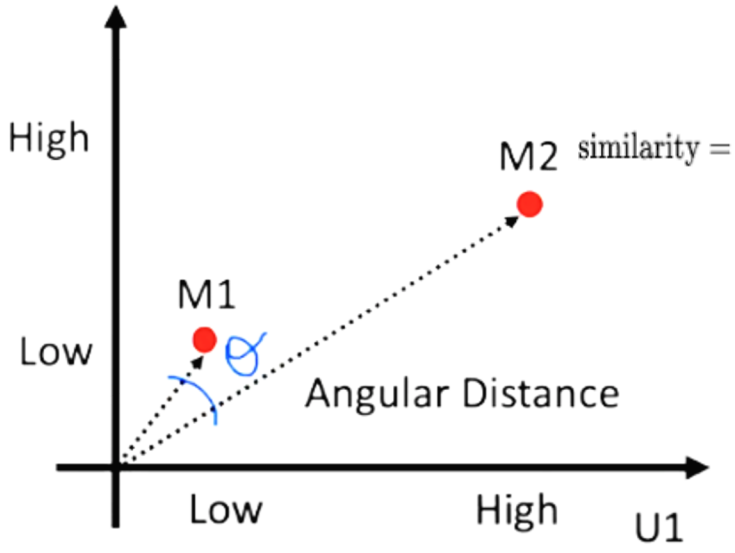
Quantifying the Similarity



Dissimilar Users

Option 1: Cosine Distance

Option 2: Pearson Correlation



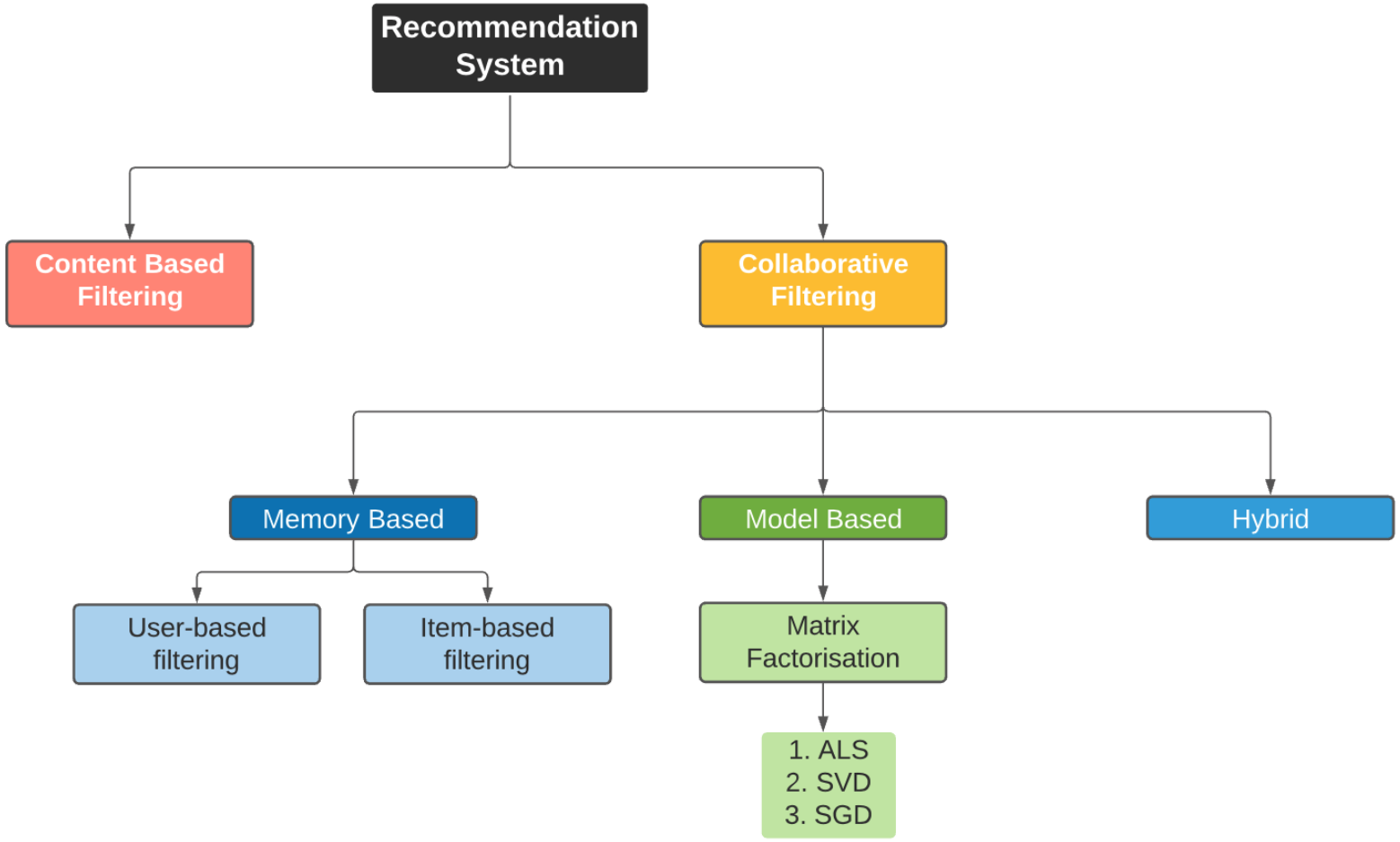
$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

Similar Users

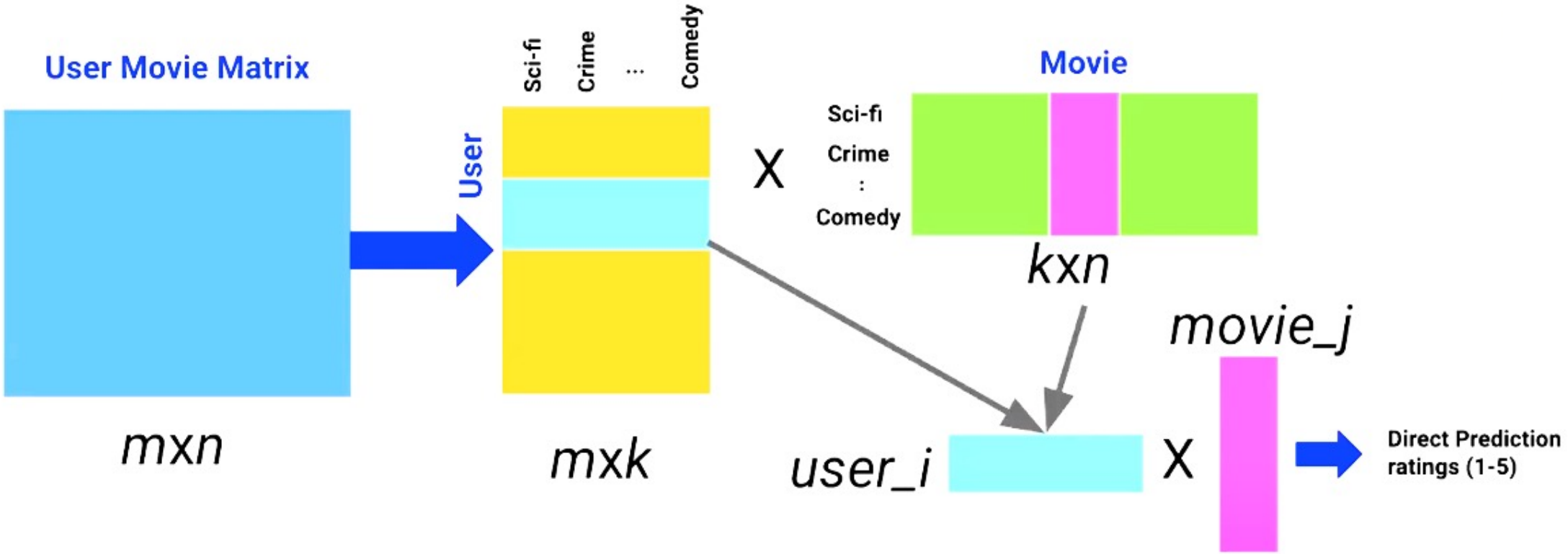
Steps Involved in Collaborative Filtering

- To build a system that can automatically recommend items to users based on the preferences of other users, the first step is to find similar users or items. The second step is to predict the ratings of the items that are not yet rated by the user. So, you will need the answers to the below questions:
  - How do you determine which users or items are similar to one another?
  - Given that you know which users are similar, how do you determine the rating that a user would give to an item based on the ratings of the similar users?
  - How do you measure the accuracy of the ratings you calculate?

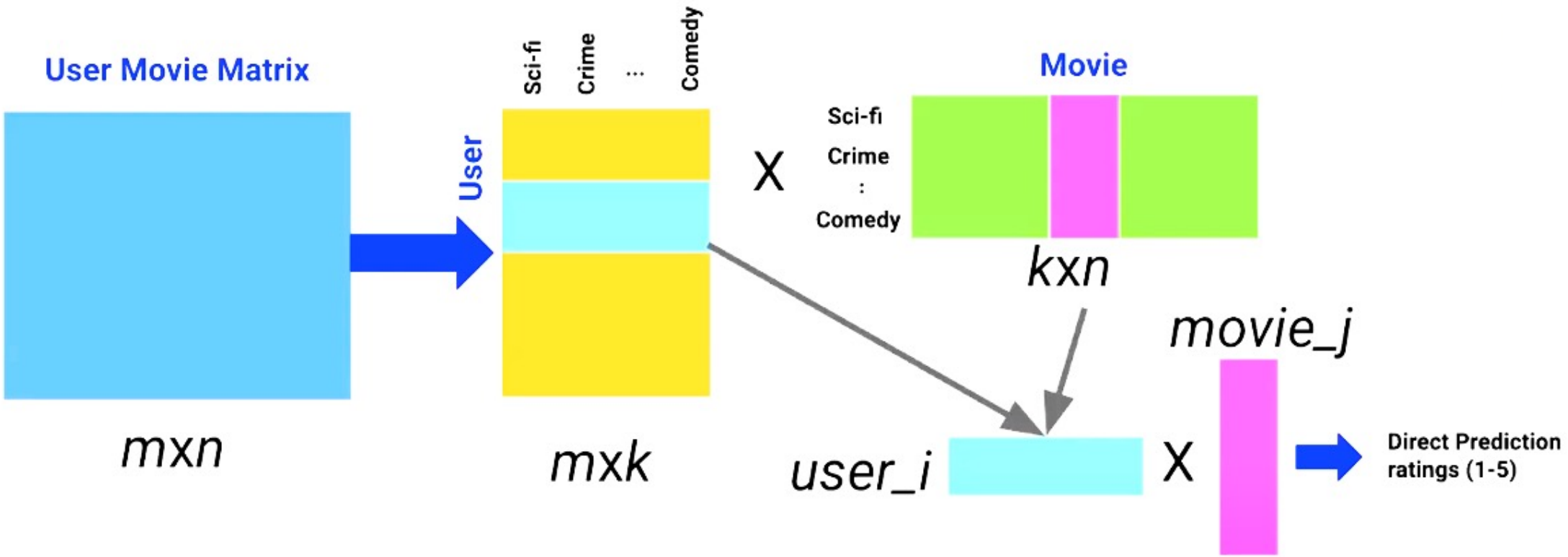
Collaborative Filtering



Model Based Collaborative Filtering

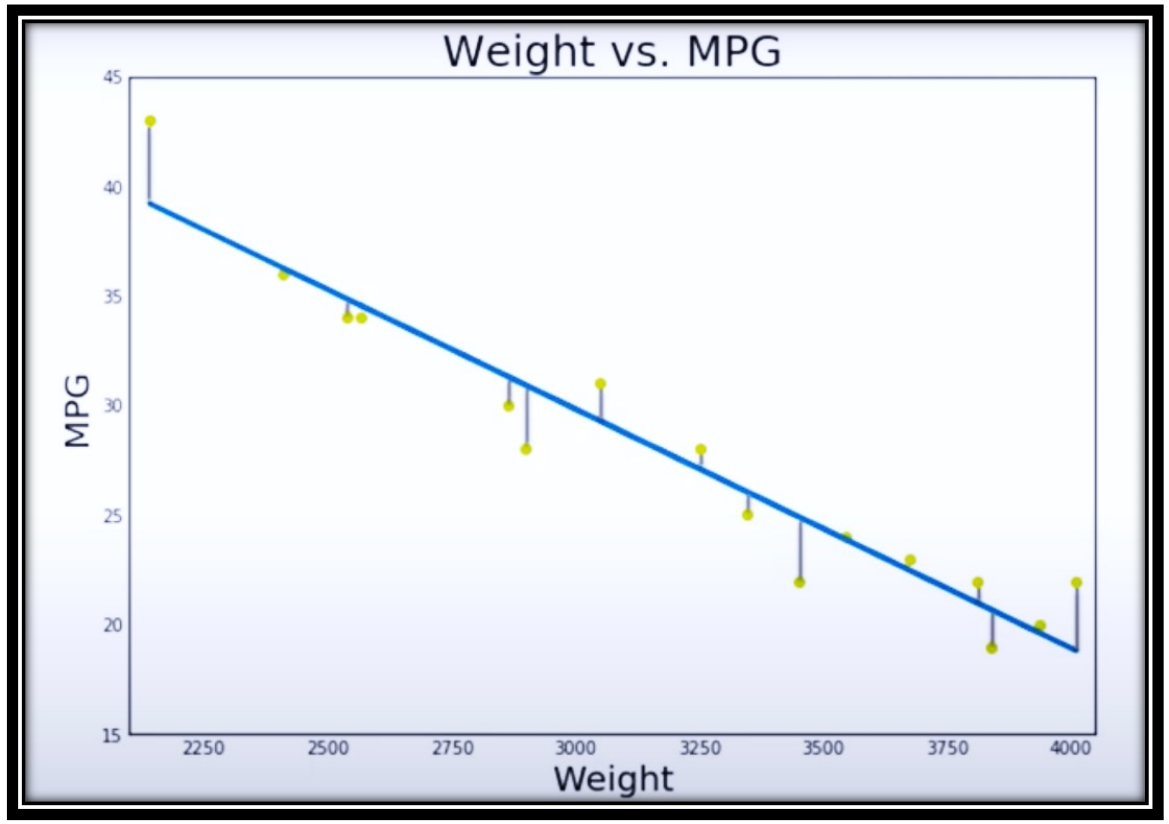


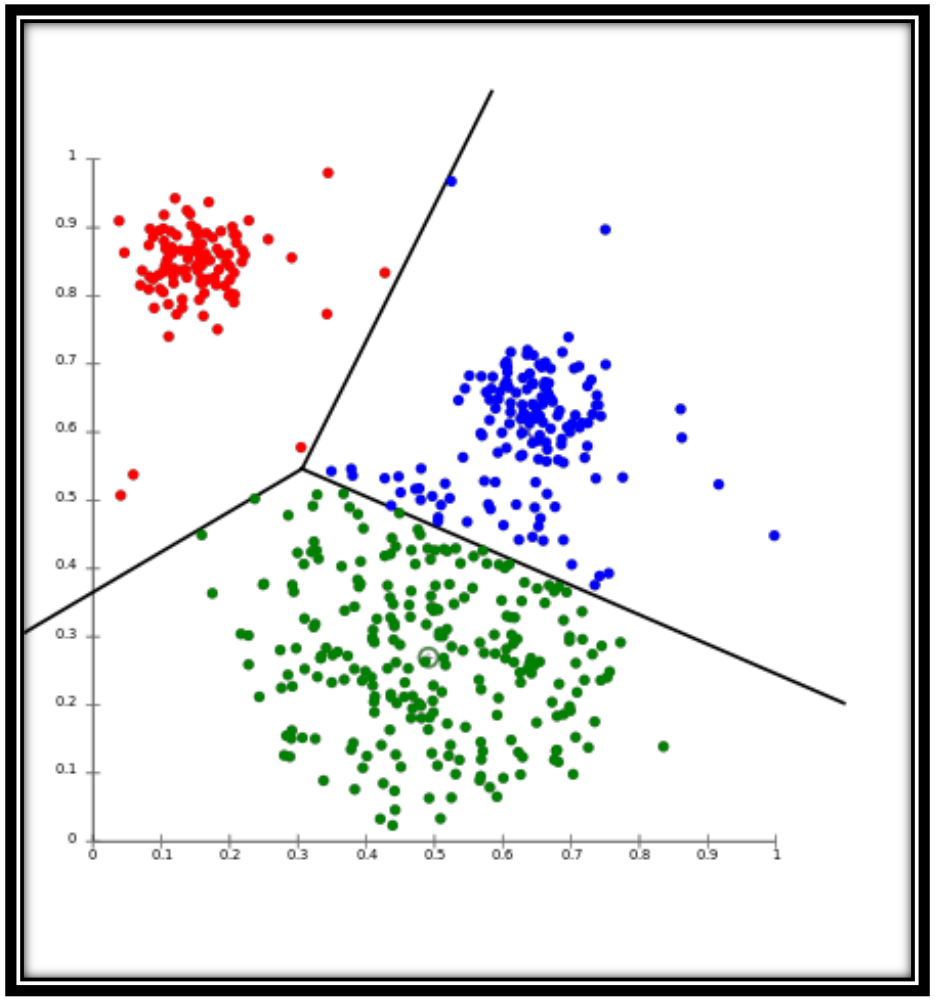
Matrix Factorization



Alternating Least Square (ALS)

Linear Regression and KMeans





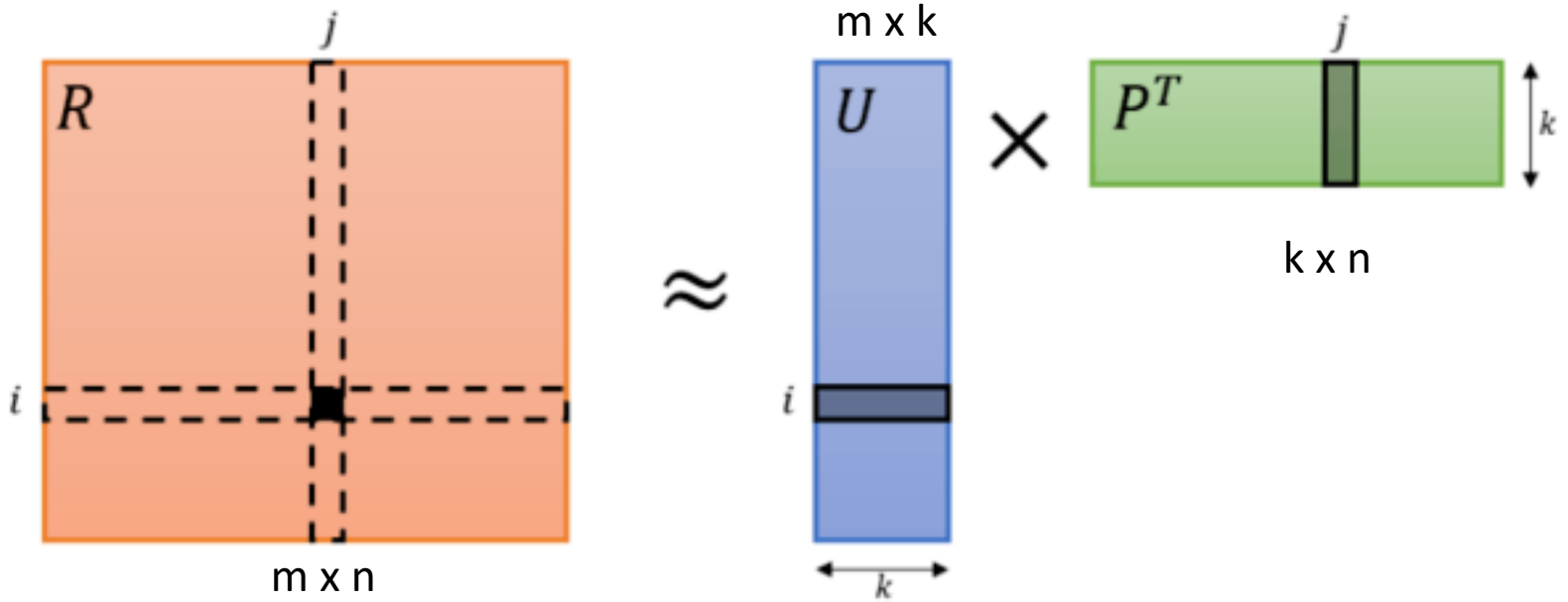
Alternating Least Square

	Movie 1	Movie 2	Movie ...	Movie N
User 1	1	BLANK	BLANK	3
User 2	BLANK	5	BLANK	3
User 3	BLANK	BLANK	1	BLANK
User 4	2	3	BLANK	BLANK
User 5	BLANK	BLANK	1	BLANK
User 6	4	BLANK	5	BLANK
User 7	BLANK	4	BLANK	BLANK
User ...	BLANK	3	BLANK	BLANK
User m	BLANK	BLANK	BLANK	4

ALS

	Movie 1	Movie 2	Movie ...	Movie N
User 1	1	4	2	3
User 2	1	5	3	3
User 3	2.5	2.8	1	3.5
User 4	2	3	2	3.5
User 5	2.5	2.8	1	3.1
User 6	4	1.2	5	1.4
User 7	1	4	2.5	3
User ...	2	3	2	3
User m	1	4	2	4

Alternating Least Square



$$Error_{ij} = \underbrace{\sum w_{ij}}_{\text{Completion Term}} \cdot \underbrace{(R_{ij} - u_i \times p_j^T)}_{\text{Cost Function}} + \underbrace{\lambda(\| U \|_2 + \| P \|_2)}_{\text{Regularization Term}}$$

Completion Term

where $w_{ij} \begin{cases} 1 & R_{ij} \text{ is known} \\ 0 & R_{ij} \text{ is unknown} \end{cases}$

Cost Function

Minimizes the difference between the product of our factor matrices and the original ratings matrix.

The matching solutions for u_i and p_j are:

$u_i = (P^T \times w_i \times P + \lambda I)^{-1} \times P^T \times w_i \times r_i$

$p_j = (U^T \times w_j \times U + \lambda I)^{-1} \times U^T \times w_j \times r_j$

Regularization Term

Prevents overfitting by applying a small amount to the error which requires more time/iterations to full minimize.

Alternating Least Square

Start with random U and P

Repeat:

- Fixing P, optimize U to minimize error on scores in R
- Fixing U, optimize P to minimize error on scores in R

Book Ratings

	userId	bookId	rating	time
0	1	1	4.0	964982703
1	1	3	4.0	964981247
2	1	6	4.0	964982224
3	1	47	5.0	964983815
4	1	50	5.0	964982931
5	1	70	3.0	964982400
6	1	101	5.0	964980868
7	1	110	4.0	964982176
8	1	151	5.0	964984041
9	1	157	5.0	964984100

ALS in Spark

```
from pyspark.ml.recommendation import ALS

from pyspark.ml.evaluation import RegressionEvaluator

from pyspark.ml.tuning import ParamGridBuilder, CrossValidator,
TrainValidationSplit
```

ALS in Spark

```
# Create train and test dataset, here we use 0.8 / 0.2
(training, test) = data.randomSplit([0.8, 0.2])


# Build the recommendation model using ALS on the
training data
als = ALS(userCol= “userId”, itemCol= “bookId”, ratingCol=
“rating”, coldStartStrategy = “drop”)
```

ALS Parameters

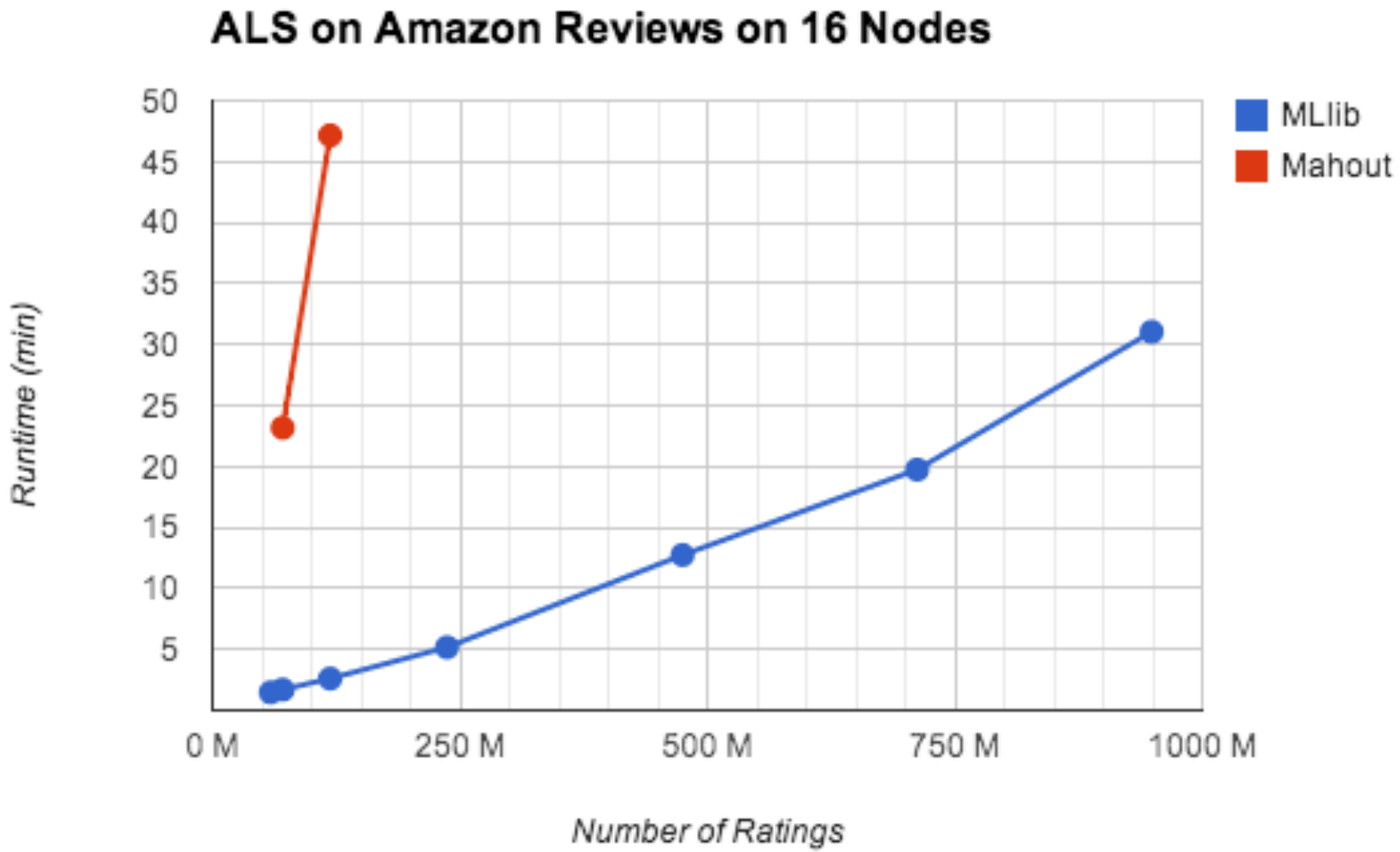
- *numBlocks* - defaults to 10
- *rank* - defaults to 10
- *maxIter* - defaults to 10
- *regParam* - defaults to 1.0
- *lambda* - defaults to 0.01
- *implicitPrefs* - defaults to false which means using *explicit feedback*.
- *alpha* - defaults to 1.0
- *nonnegative* - defaults to false

ALS in Spark

```
#Tune model using ParamGridBuilder
parameters=ParamGridBuilder()\
        .addGrid(als.rank,[--, --, --])\
        .addGrid(als.maxIter, [--, --, --])\
        .addGrid(als.regParam, [--, --, --])\
        .addGrid(als.numItemBlocks, [--, --, --])\
        .addGrid(als.numUserBlocks, [--, --, --])\
        .build()
#Use model evaluator as RMSE
eval = RegressionEvaluator(metricName= “rmse”,
                labelCol=  “rating”,predictionCol= “prediction”)
```

ALS in Spark

```
#Build train validation split
trainvs = TrainValidationSplit(
                    estimator=als,
                    estimatorParamMaps=parameters,
                    evaluator=eval)

#Build cross validator

cv = CrossValidator(estimator =als,
                    estimatorParamMaps =parameters,
                    evaluator =eval,
                    numFolds=3)


#Fit model to training data
model = trainvs.fit(training)


# Evaluate the model by computing the RMSE on the test data
predictions = model.transform(test)

rmse = eval.evaluate(predictions)
```


ALS on Amazon Reviews on 16 Nodes
50
45
40
35
30
25
20
15
10
5
Runtime (min)
0 M
250 M
500 M
750 M
1000 M
Number of Ratings
MLlib
Mahout

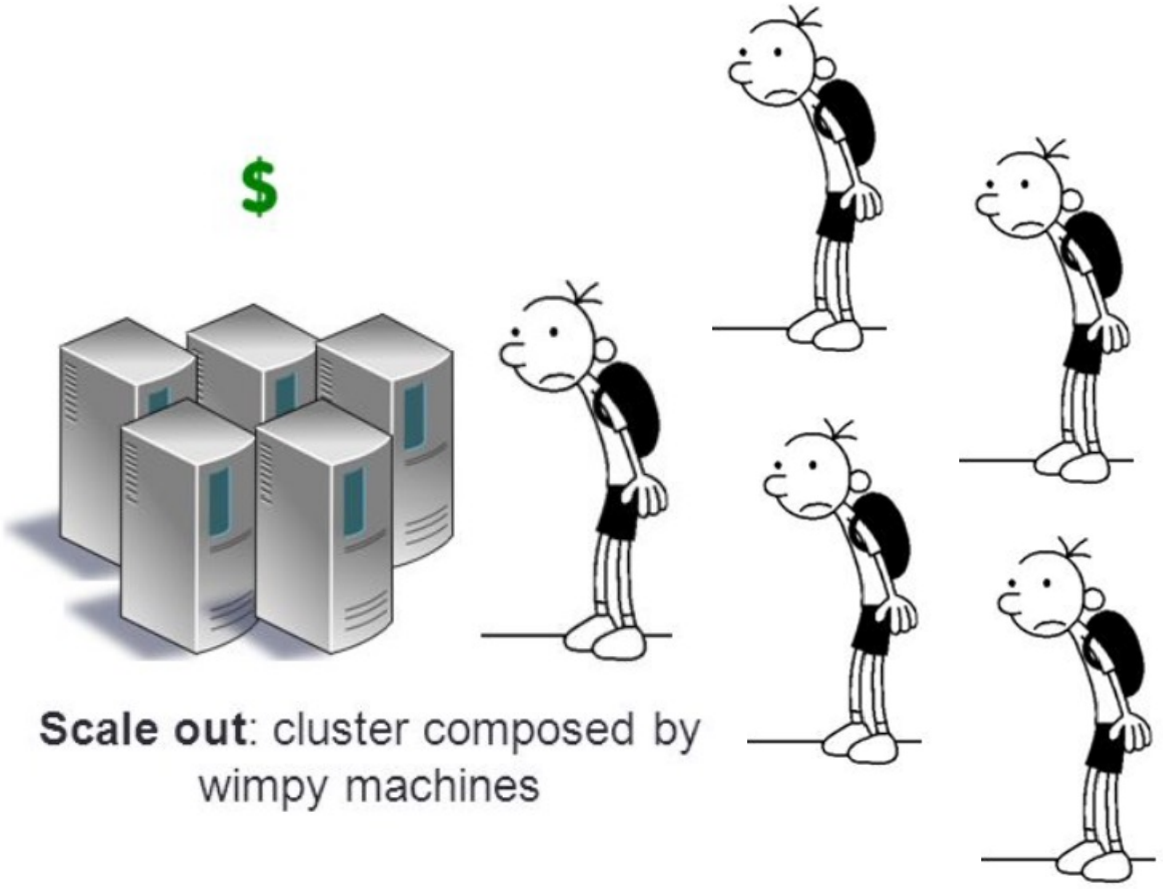
Examples

- **ML Tuning: model selection and hyperparameter tuning**
- https://spark.apache.org/docs/latest/ml-tuning.html#train-validation-split
- https://spark.apache.org/docs/2.2.0/ml-collaborative-filtering.html

Databricks

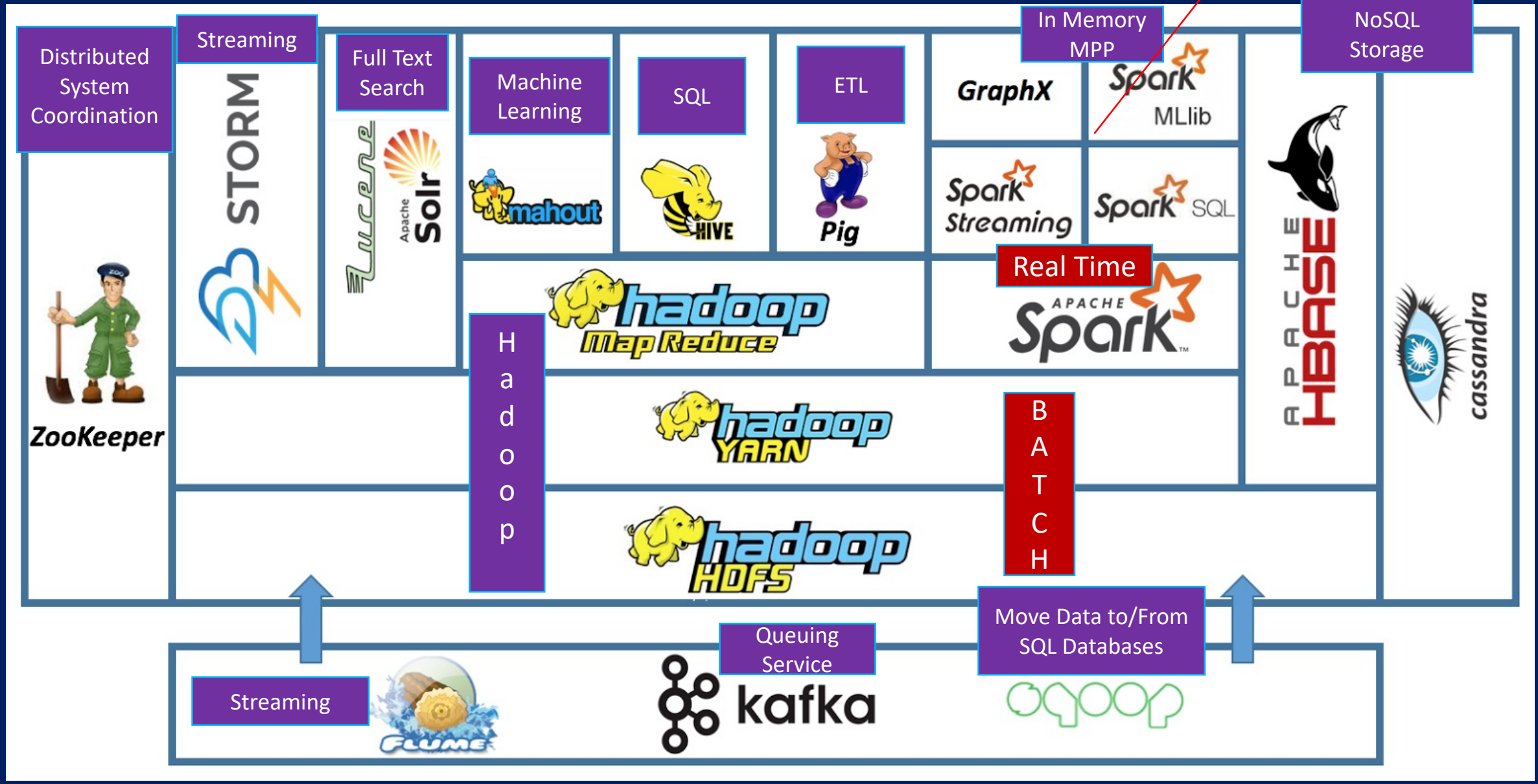
Scale up vs Scale out

Scale up: one machine with high hardware configuration



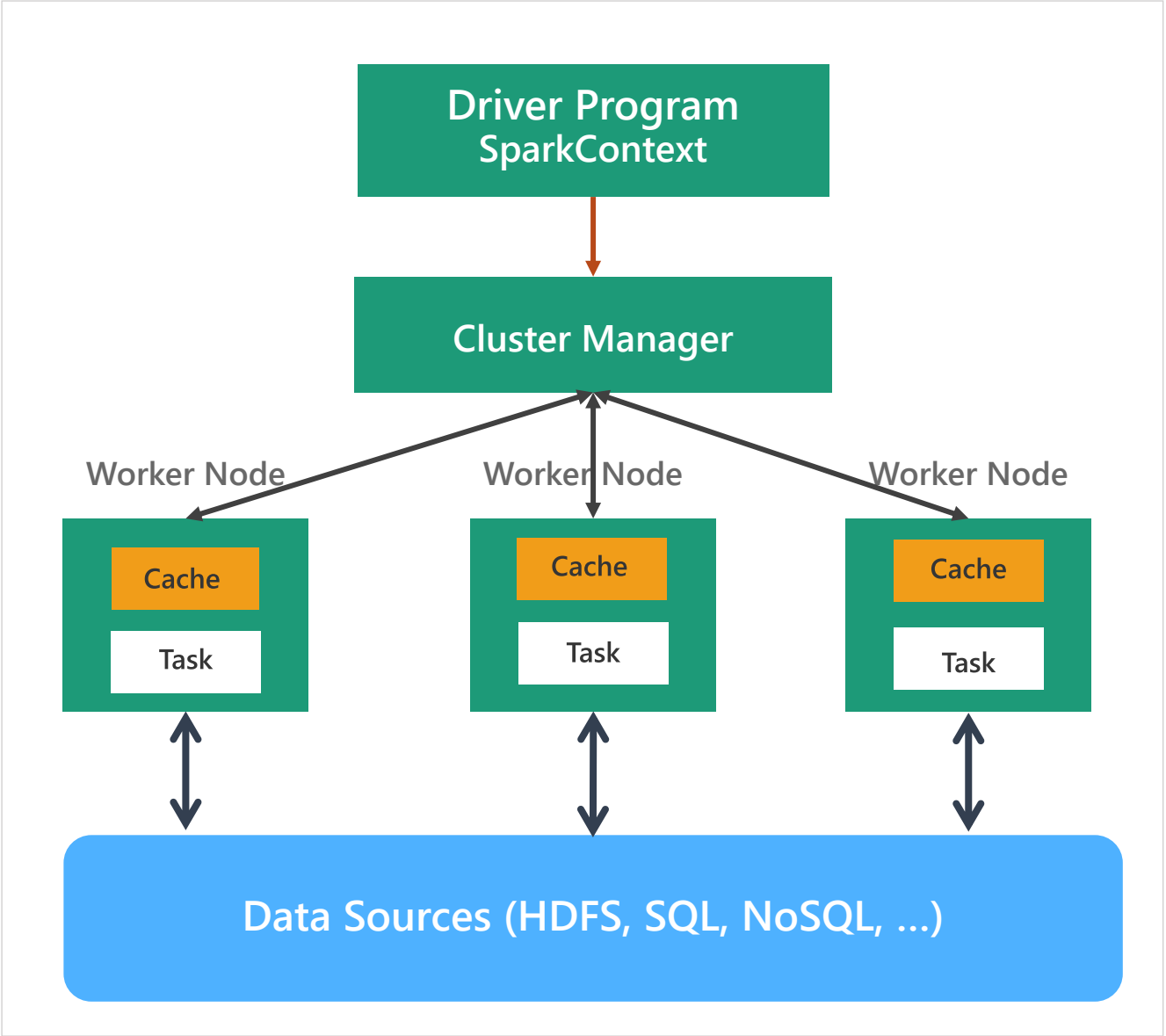
Scale out: cluster composed by wimpy machines

The Apache Hadoop Ecosystem



General Spark Cluster Architecture

- 'Driver' runs the user's 'main' function and executes the various parallel operations on the worker nodes.
- The results of the operations are collected by the driver
- The worker nodes read and write data from/to Data Sources including HDFS.
- Worker node also cache transformed data in memory as RDDs (Resilient Data Sets).
- Worker nodes and the Driver Node execute as VMs in public clouds (AWS, Google and Azure).

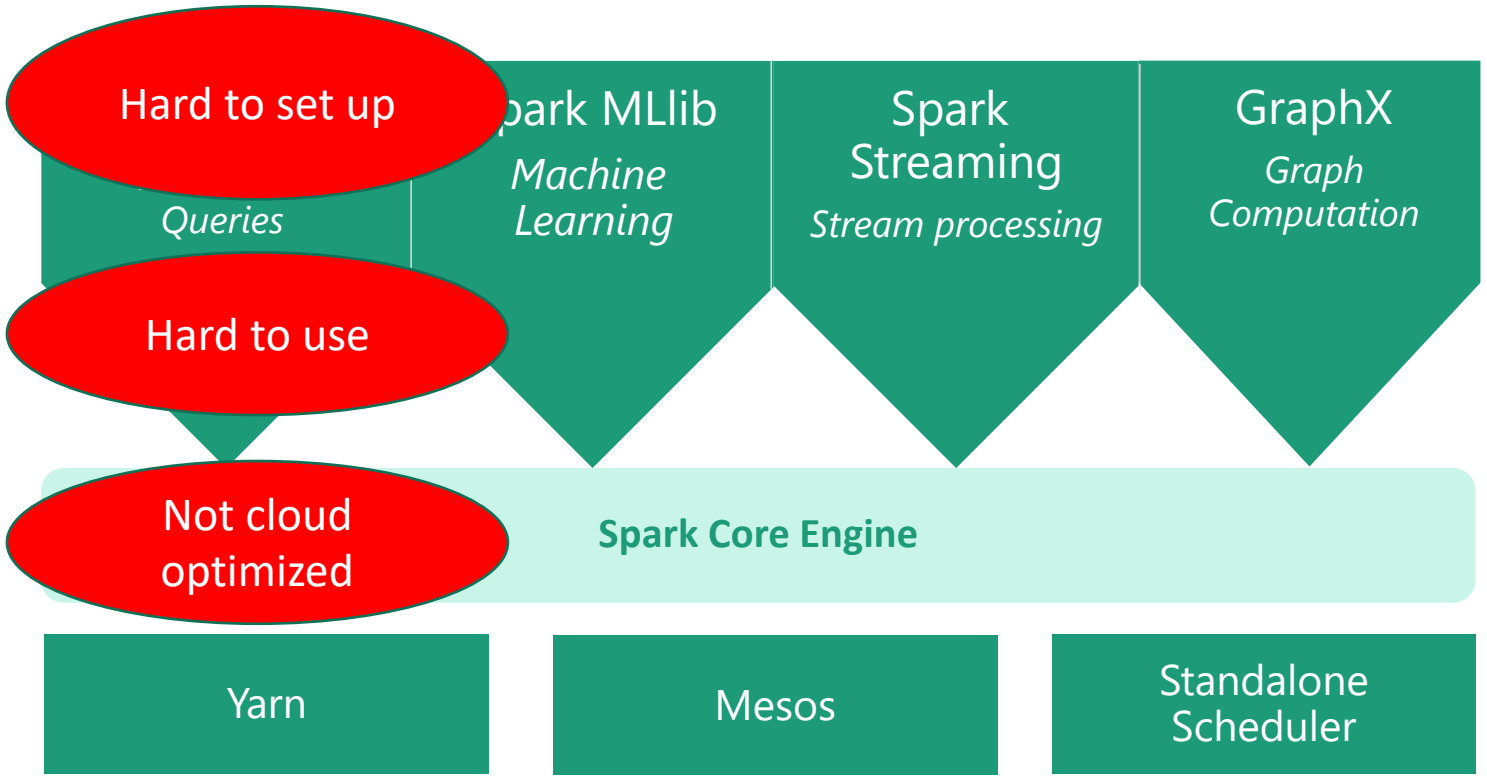


Apache Spark

A unified, open source, parallel, data processing framework for Big Data Analytics

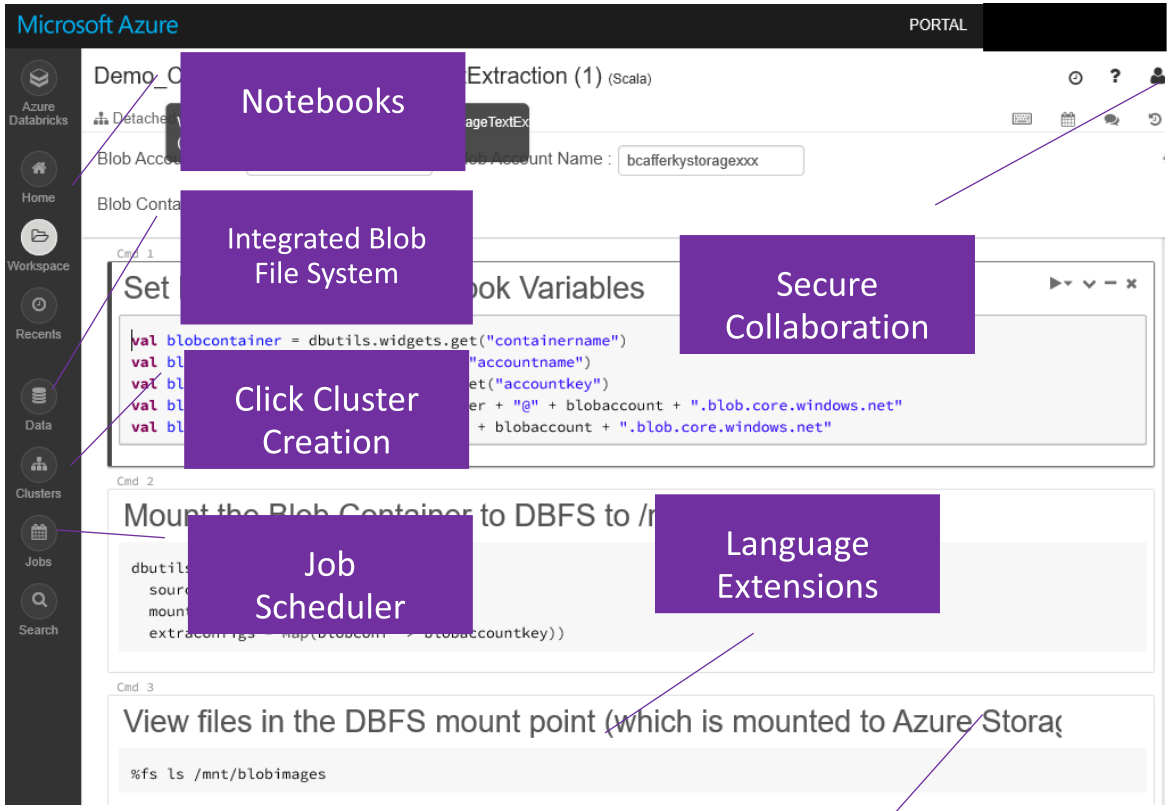
Spark Unifies:

- Batch Processing
- Interactive SQL
- Real-time processing
- Machine Learning
- Deep Learning
- Graph Processing



Why do I need Databricks?

Databricks



Notebooks

Integrated Blob File System

Click Cluster Creation

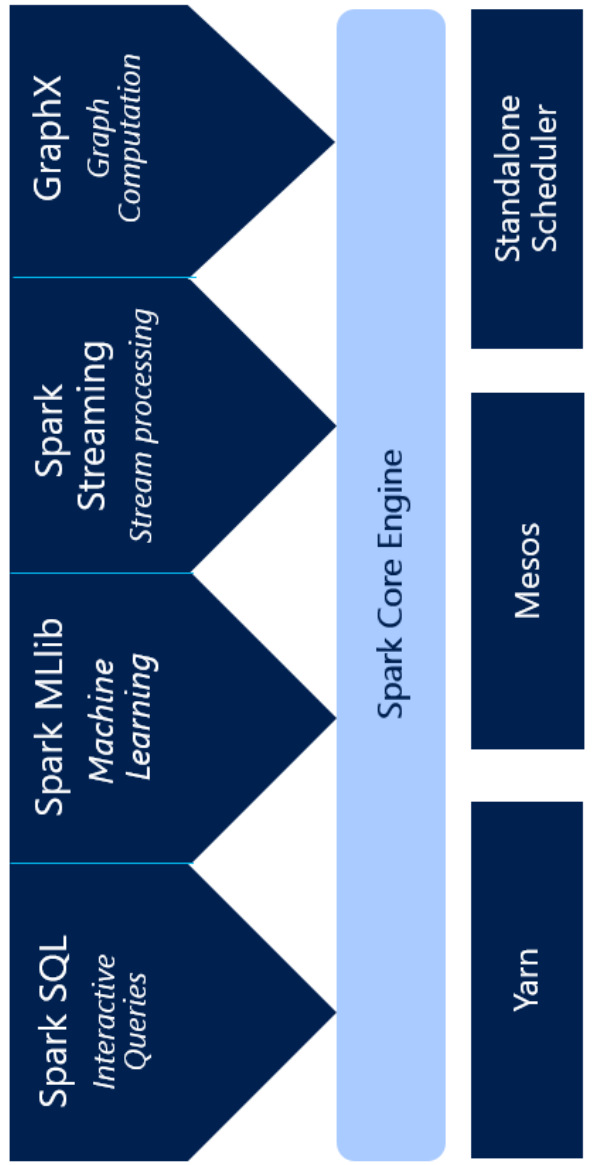
Job Scheduler

Secure Collaboration

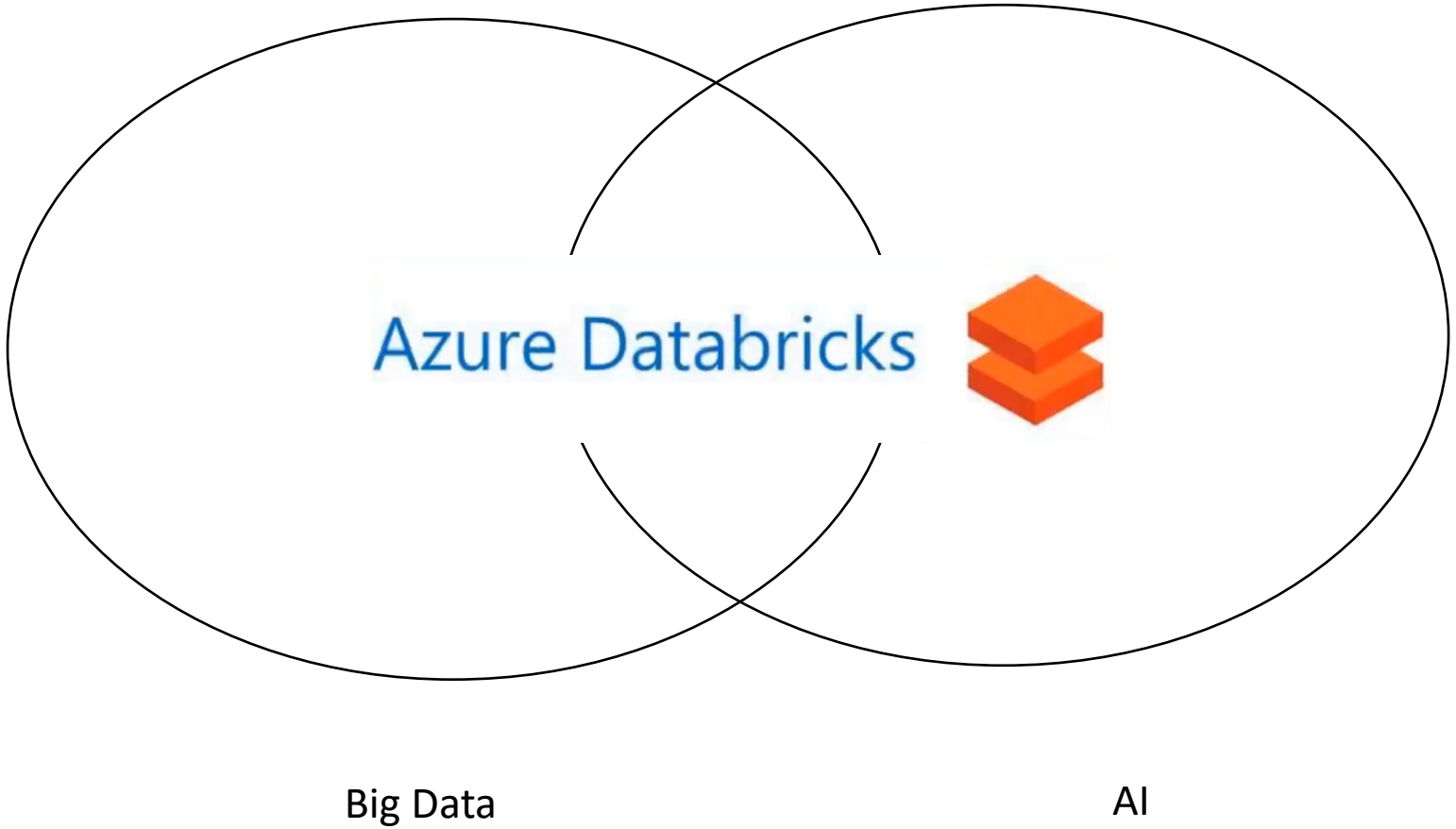
Language Extensions

Spark Optimizations

It all runs on Spark

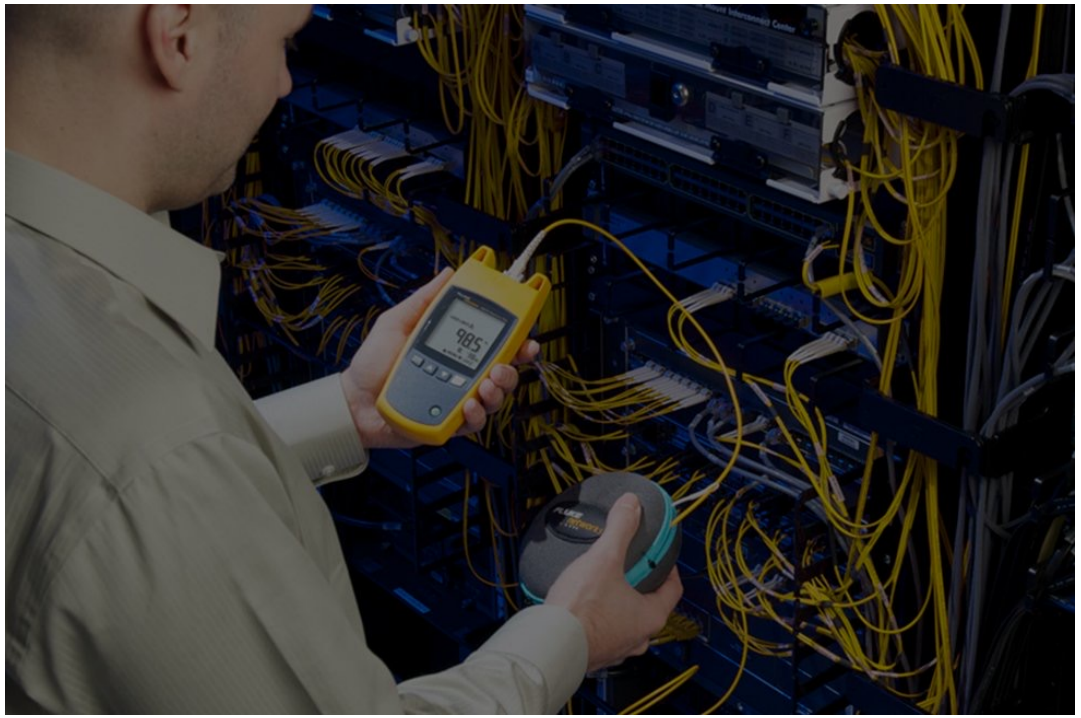


Cloud

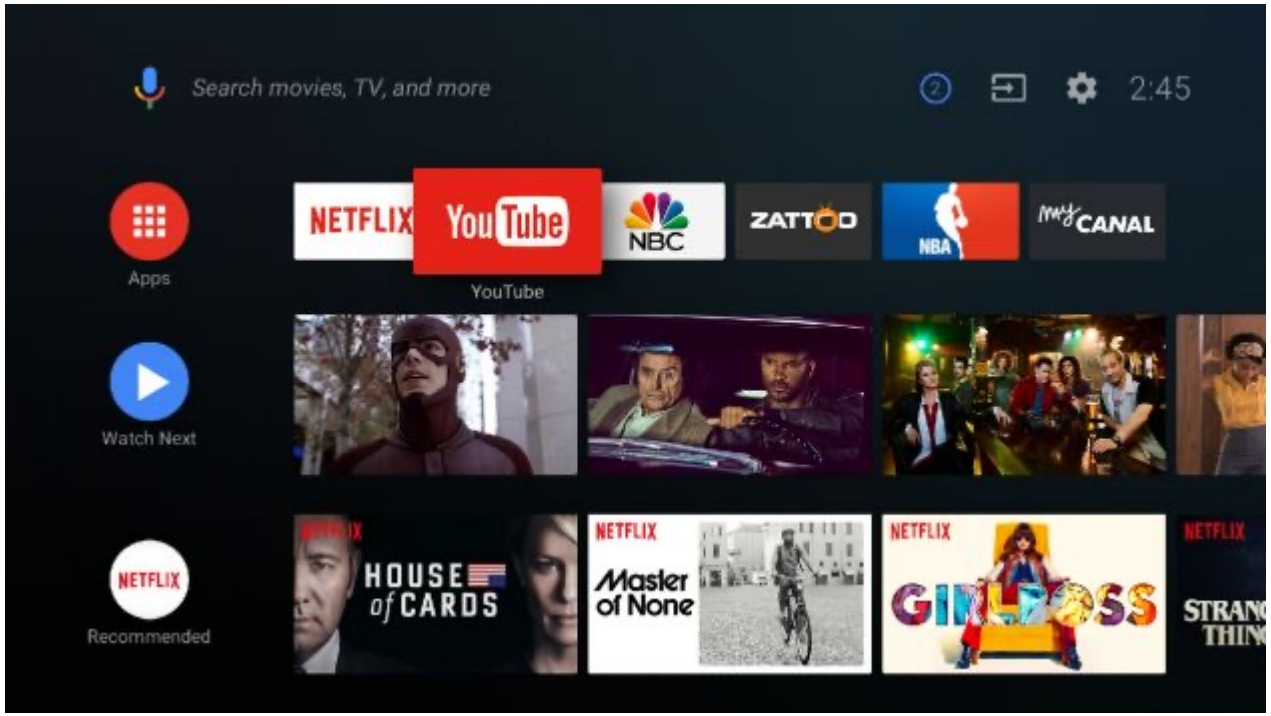


Comparison

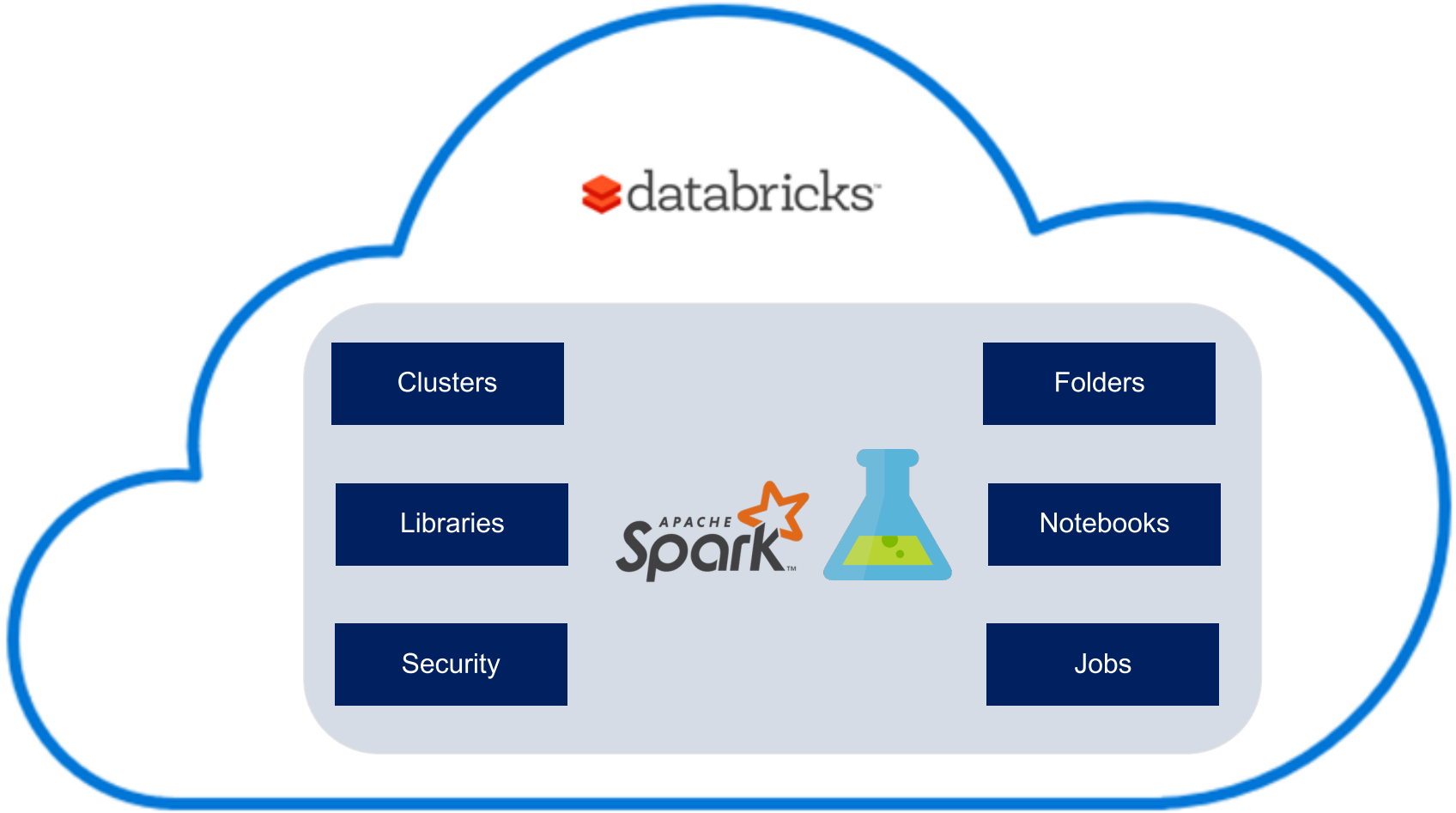
Apache Spark



Databricks



Databricks


databricks
Clusters
Folders
Libraries
APACHE Spark
Notebooks
Security
Jobs

Databricks

Creating a Cluster

Uploading Data

Workspace

Using Notebooks

Dashboards

Security

Cell Actions

Schedule

Revision History

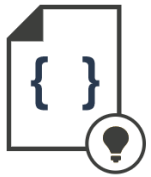
Comments

Libraries

Jobs

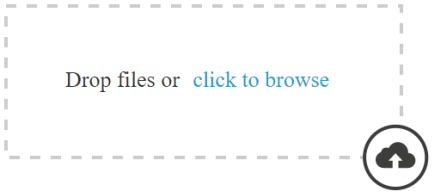
Upgrade

Welcome to databricks



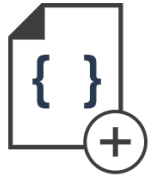
Explore the Quickstart Tutorial

Spin up a cluster, run queries on preloaded data, and display results in 5 minutes.



Import & Explore Data

Quickly import data, preview its schema, create a table, and query it in a notebook.



Create a Blank Notebook

Create a notebook to start querying, visualizing, and modeling your data.

Common Tasks

- New Notebook
- Create Table
- New Cluster
- New Job
- New MLflow Experiment
- Import Library
- Read Documentation

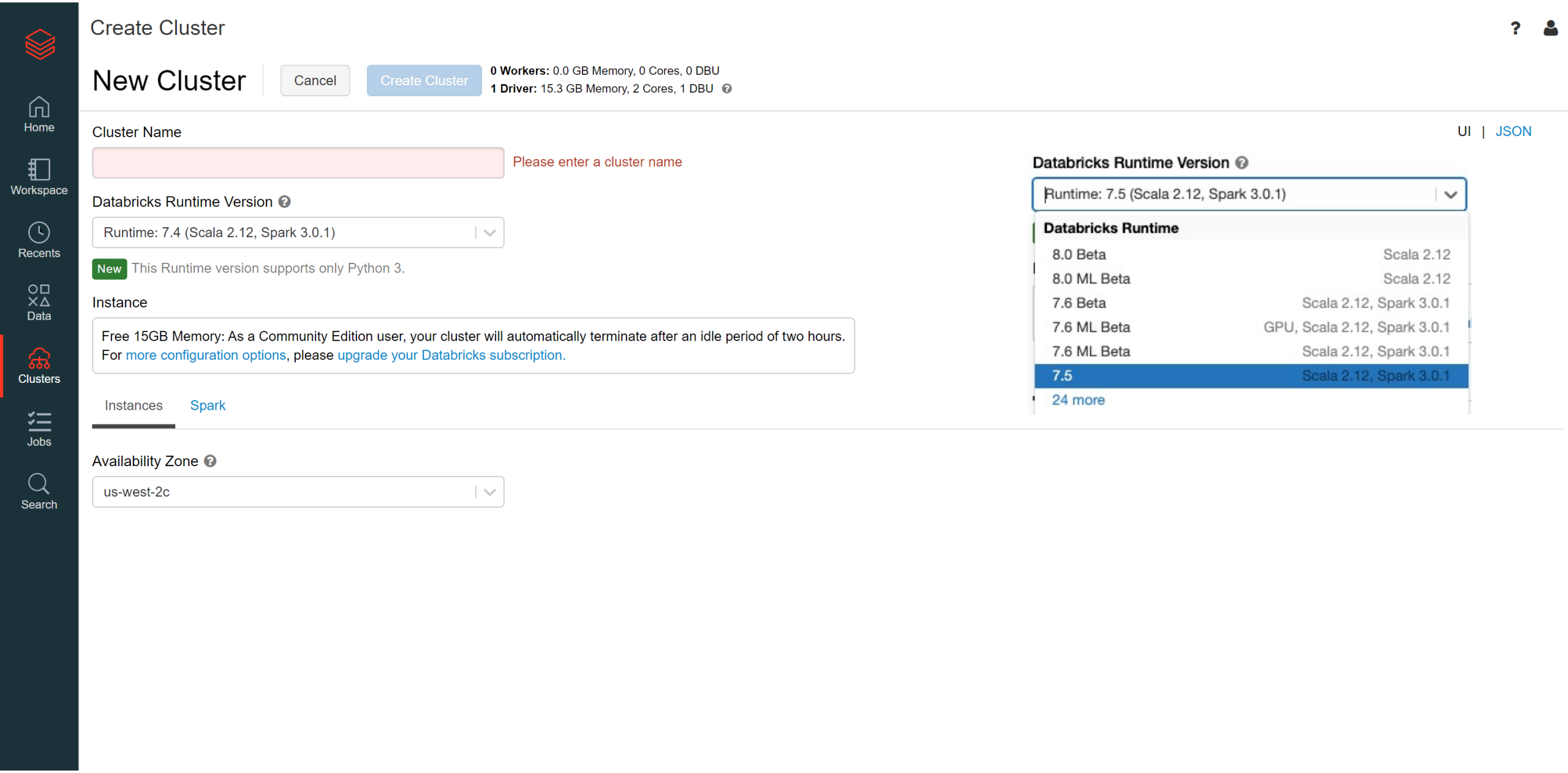
Recents

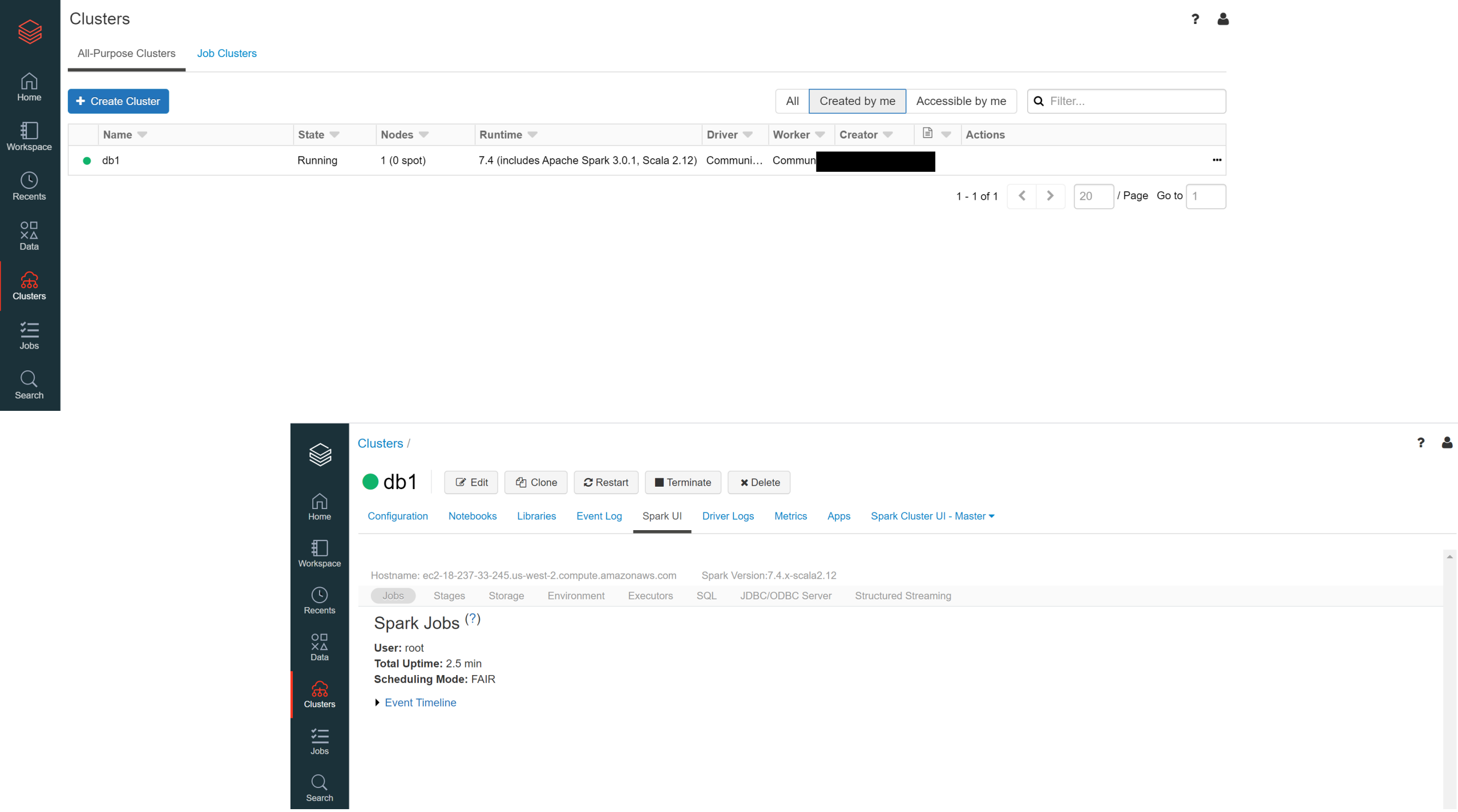
Recent files appear here as you work.

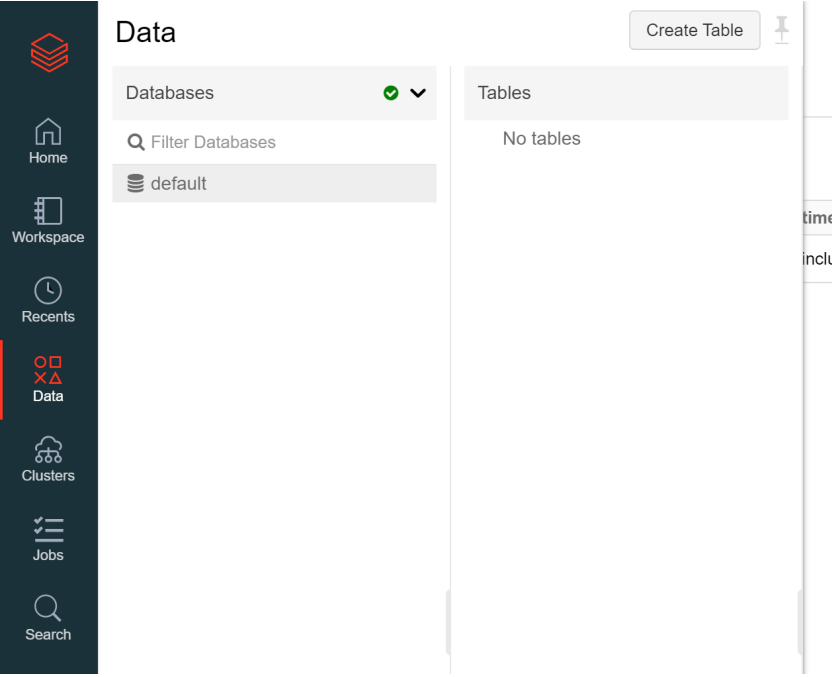
What's new in v3.34

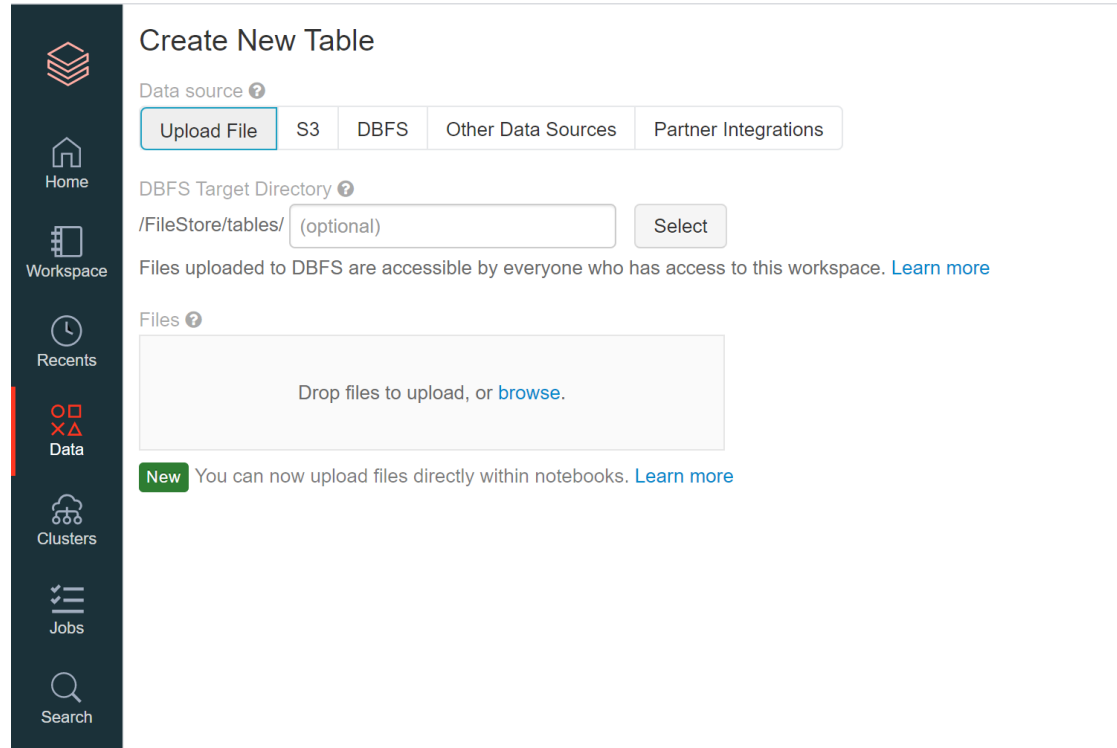
View latest release notes

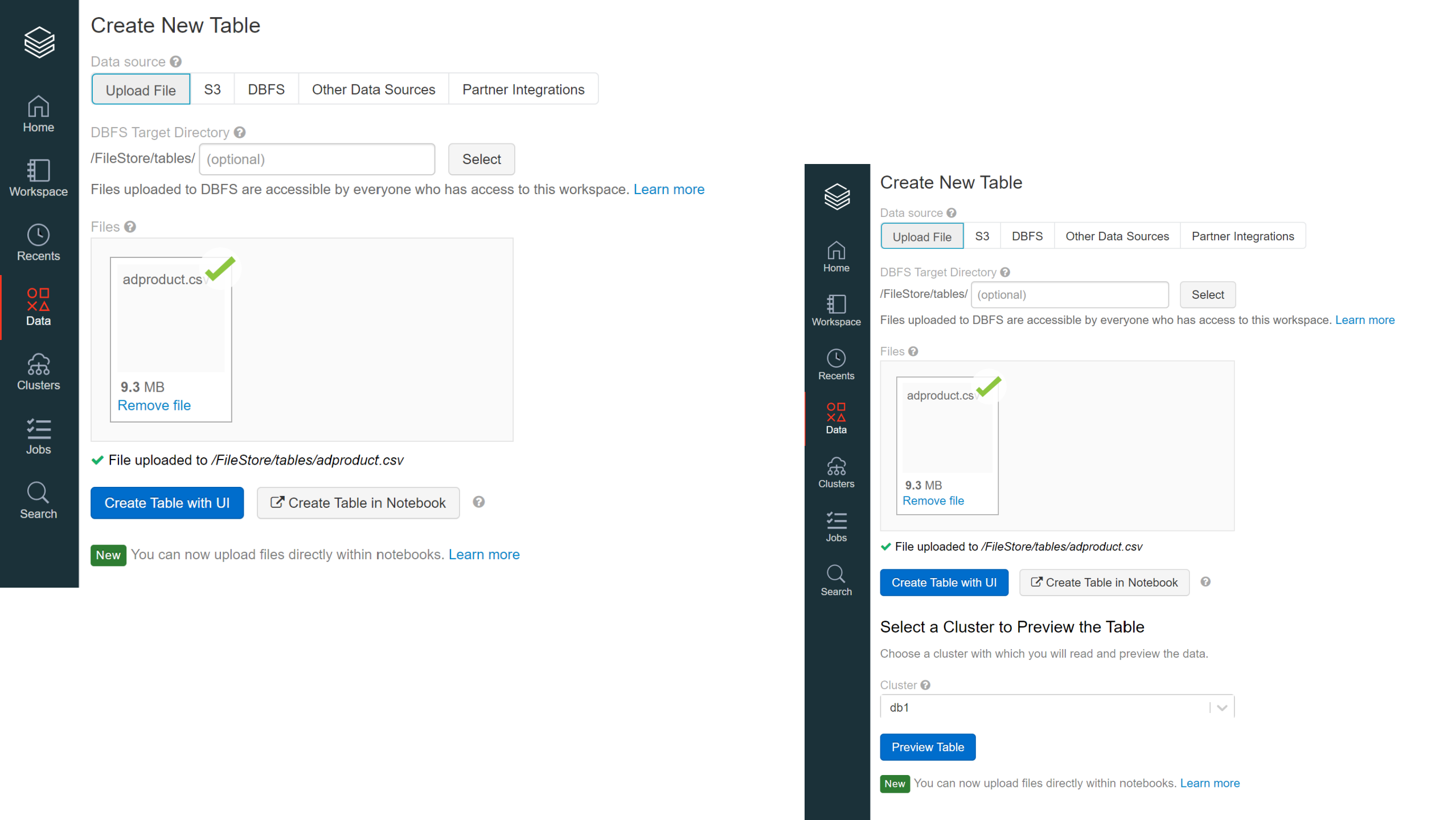
Home
Workspace
Recents
Data
Clusters
Jobs
Search


Create Cluster
New Cluster
Cancel
Create Cluster
0 Workers: 0.0 GB Memory, 0 Cores, 0 DBU
1 Driver: 15.3 GB Memory, 2 Cores, 1 DBU
Home
Workspace
Recents
Data
Clusters
Jobs
Search
Cluster Name
Please enter a cluster name
Databricks Runtime Version
Runtime: 7.4 (Scala 2.12, Spark 3.0.1)
New
This Runtime version supports only Python 3.
Instance
Free 15GB Memory: As a Community Edition user, your cluster will automatically terminate after an idle period of two hours.
For more configuration options, please upgrade your Databricks subscription.
Instances
Spark
Availability Zone
us-west-2c
UI | JSON
Databricks Runtime Version
Runtime: 7.5 (Scala 2.12, Spark 3.0.1)
Databricks Runtime
8.0 Beta Scala 2.12
8.0 ML Beta Scala 2.12
7.6 Beta Scala 2.12, Spark 3.0.1
7.6 ML Beta GPU, Scala 2.12, Spark 3.0.1
7.6 ML Beta Scala 2.12, Spark 3.0.1
7.5 Scala 2.12, Spark 3.0.1
24 more


Clusters
All-Purpose Clusters
Job Clusters
Create Cluster
All
Created by me
Accessible by me
Filter...
Name
State
Nodes
Runtime
Driver
Worker
Creator
Actions
db1
Running
1 (0 spot)
7.4 (includes Apache Spark 3.0.1, Scala 2.12)
Communi...
Commun
1 - 1 of 1
20
/ Page
Go to
1
Home
Workspace
Recents
Data
Clusters
Jobs
Search
Clusters /
db1
Edit
Clone
Restart
Terminate
Delete
Configuration
Notebooks
Libraries
Event Log
Spark UI
Driver Logs
Metrics
Apps
Spark Cluster UI - Master
Hostname: ec2-18-237-33-245.us-west-2.compute.amazonaws.com
Spark Version:7.4.x-scala2.12
Jobs
Stages
Storage
Environment
Executors
SQL
JDBC/ODBC Server
Structured Streaming
Spark Jobs (?)
User: root
Total Uptime: 2.5 min
Scheduling Mode: FAIR
Event Timeline


Data
Create Table
Databases
Filter Databases
default
Tables
No tables
Home
Workspace
Recents
Data
Clusters
Jobs
Search
All
Created by me
Accessible by me
Filter...
time
Driver
Worker
Creator
Actions
includes Apache Spark 3.0.1, Scala 2.12)
Communi…
Communi…
1 - 1 of 1
20
/ Page
Go to
1


Create New Table
Data source
Upload File
S3
DBFS
Other Data Sources
Partner Integrations
DBFS Target Directory
/FileStore/tables/
(optional)
Select
Files uploaded to DBFS are accessible by everyone who has access to this workspace. Learn more
Files
Drop files to upload, or browse.
New
You can now upload files directly within notebooks. Learn more
Home
Workspace
Recents
Data
Clusters
Jobs
Search


Create New Table
Data source
Upload File
S3
DBFS
Other Data Sources
Partner Integrations
DBFS Target Directory
/FileStore/tables/
(optional)
Select
Files uploaded to DBFS are accessible by everyone who has access to this workspace. Learn more
Files
adproduct.cs
9.3 MB
Remove file
File uploaded to /FileStore/tables/adproduct.csv
Create Table with UI
Create Table in Notebook
New
You can now upload files directly within notebooks. Learn more
Home
Workspace
Recents
Data
Clusters
Jobs
Search
Create New Table
Data source
Upload File
S3
DBFS
Other Data Sources
Partner Integrations
DBFS Target Directory
/FileStore/tables/
(optional)
Select
Files uploaded to DBFS are accessible by everyone who has access to this workspace. Learn more
Files
adproduct.cs
9.3 MB
Remove file
File uploaded to /FileStore/tables/adproduct.csv
Create Table with UI
Create Table in Notebook
Select a Cluster to Preview the Table
Choose a cluster with which you will read and preview the data.
Cluster
db1
Preview Table
New
You can now upload files directly within notebooks. Learn more
Home
Workspace
Recents
Data
Clusters
Jobs
Search

Create New Table

Remove file

File uploaded to */FileStore/tables/adproduct.csv*

Create Table with UI | Create Table in Notebook

Select a Cluster to Preview the Table

Choose a cluster with which you will read and preview the data.

Cluster

db1

Preview Table

Specify Table Attributes

Specify the Table Name, Database and Schema to add this to the data UI for other users to access

Table Name: adproduct_csv

Create in Database: default

File Type: CSV

Column Delimiter: ,

First row is header

Infer schema

Multi-line

Create Table

Create Table in Notebook

Table Preview

_c0	_c1	_c2	_c3	_c4	_c5
STRING	STRING	STRING	STRING	STRING	STRING
ProductKey	ProductAlternateKey	ProductSubcategoryKey	WeightUnitMeasureCode	SizeUnitMeasureCode	EnglishProductName
1	AR-5381	NULL	NULL	NULL	Adjustable Race
2	BA-8327	NULL	NULL	NULL	Bearing Ball
3	BE-2349	NULL	NULL	NULL	BB Ball Bearing
4	BE-2908	NULL	NULL	NULL	Headset Ball Bearings
5	BL-2036	NULL	NULL	NULL	Blade

New You can now upload files directly within notebooks. Learn more

Create New Table

Remove file

✔ File uploaded to */FileStore/tables/adproduct.csv*

Create Table with UI | Create Table in Notebook

Select a Cluster to Preview the Table

Choose a cluster with which you will read and preview the data.

Cluster: db1

Preview Table

Specify Table Attributes

Specify the Table Name, Database and Schema to add this to the data UI for other users to access

Table Name: adproduct_csv

Create in Database: default

File Type: CSV

Column Delimiter: ,

First row is header

Infer schema

Multi-line

Create Table

Create Table in Notebook

Table Preview

ProductKey	ProductAlternateKey	ProductSubcategoryKey	WeightUnitMeasureCode	SizeUnitMeasureCode	EnglishProductName
INT	STRING	STRING	STRING	STRING	STRING
1	AR-5381	NULL	NULL	NULL	Adjustable Race
2	BA-8327	NULL	NULL	NULL	Bearing Ball
3	BE-2349	NULL	NULL	NULL	BB Ball Bearing
4	BE-2908	NULL	NULL	NULL	Headset Ball Bearings
5	BL-2036	NULL	NULL	NULL	Blade
6	CA-5965	NULL	NULL	NULL	LL Crankarm

New You can now upload files directly within notebooks. Learn more

Home
Workspace
Recents
Data
Clusters
Jobs
Search

adproduct_csv

Refresh

db1

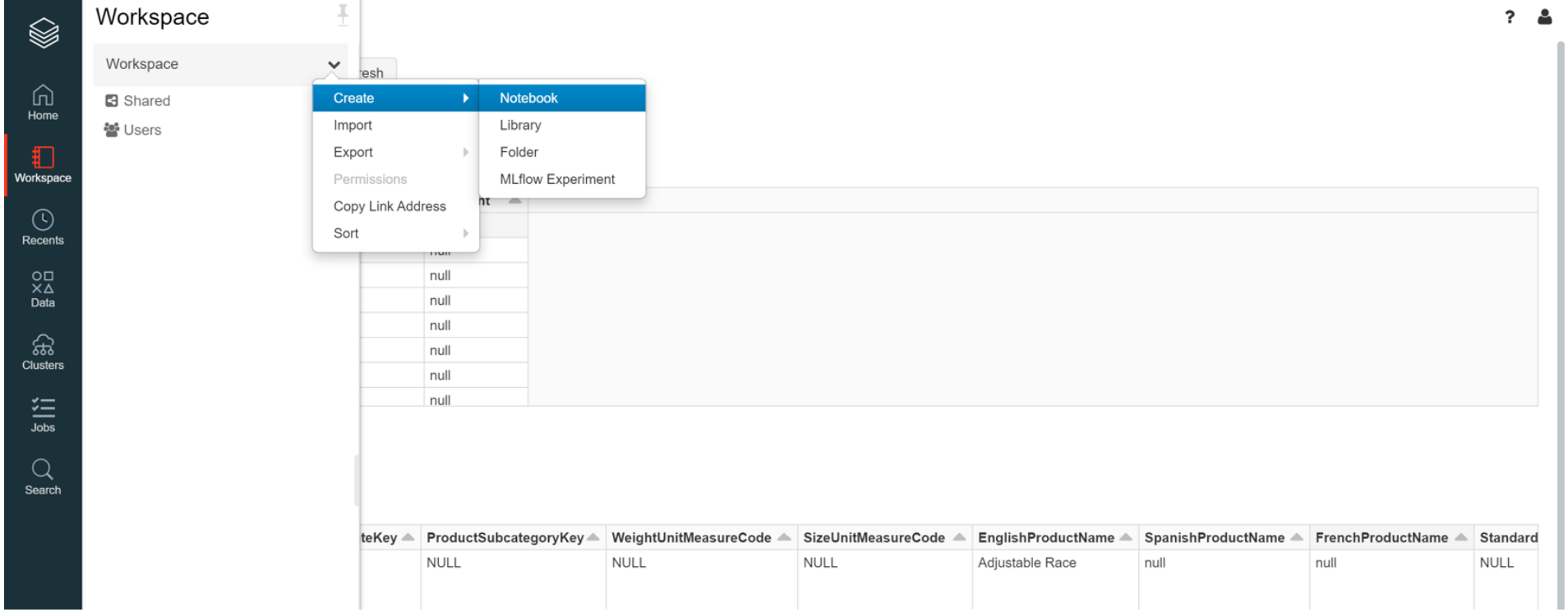
Schema:

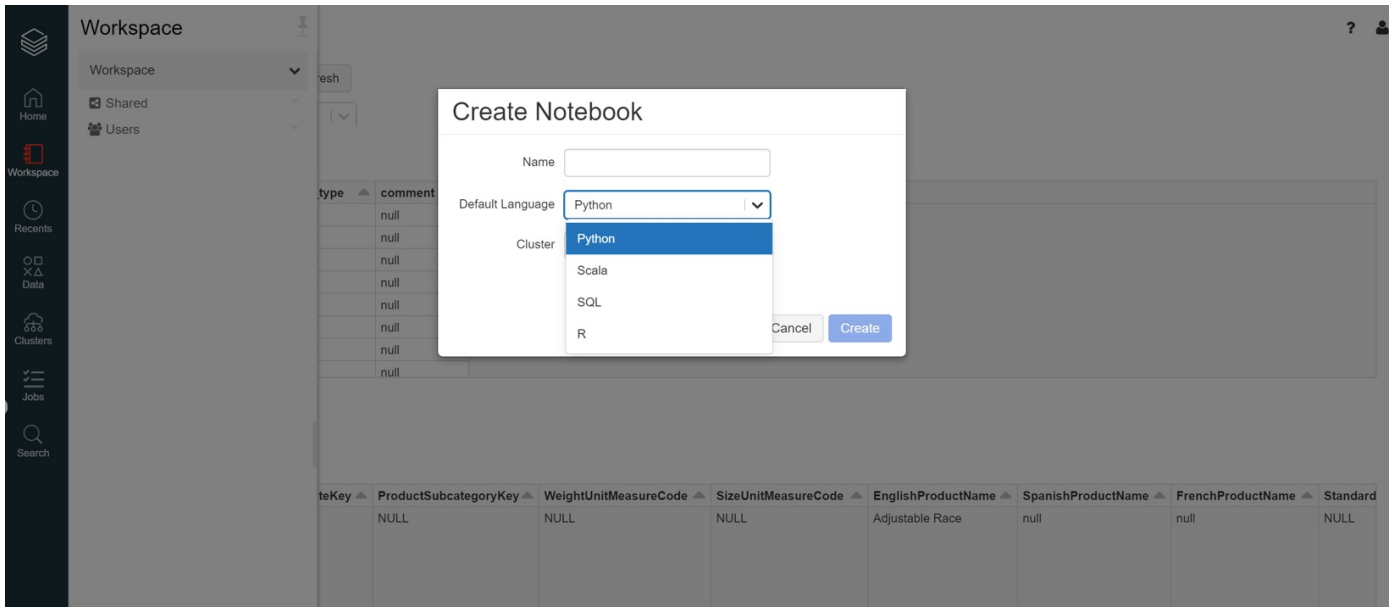
	col_name	data_type	comment
1	ProductKey	int	null
2	ProductAlternateKey	string	null
3	ProductSubcategoryKey	string	null
4	WeightUnitMeasureCode	string	null
5	SizeUnitMeasureCode	string	null
6	EnglishProductName	string	null
7	SpanishProductName	string	null
8	FrenchProductName	string	null

Showing all 36 rows.

Sample Data:

	ProductKey	ProductAlternateKey	ProductSubcategoryKey	WeightUnitMeasureCode	SizeUnitMeasureCode	EnglishProductName	SpanishProductName	FrenchProductName	Standard
1	1	AR-5381	NULL	NULL	NULL	Adjustable Race	null	null	NULL


Workspace
Workspace
Shared
Users
Home
Workspace
Recents
Data
Clusters
Jobs
Search
Create
Import
Export
Permissions
Copy Link Address
Sort
Notebook
Library
Folder
MLflow Experiment
null
null
null
null
null
null
null
teKey
ProductSubcategoryKey
WeightUnitMeasureCode
SizeUnitMeasureCode
EnglishProductName
SpanishProductName
FrenchProductName
Standard
NULL
NULL
NULL
Adjustable Race
null
null
NULL


Workspace
Create Notebook
Name
Default Language
Python
Cluster
Python
Scala
SQL
R
Cancel
Create

Cmd 1

```
1 select * from adproduct_csv limit 5
```

(1) Spark Jobs

	ProductKey	ProductAlternateKey	ProductSubcategoryKey	WeightUnitMeasureCode	SizeUnitMeasureCode	EnglishProductName	SpanishProductName	FrenchProductName	Sta
1	1	AR-5381	NULL	NULL	NULL	Adjustable Race	null	null	NU

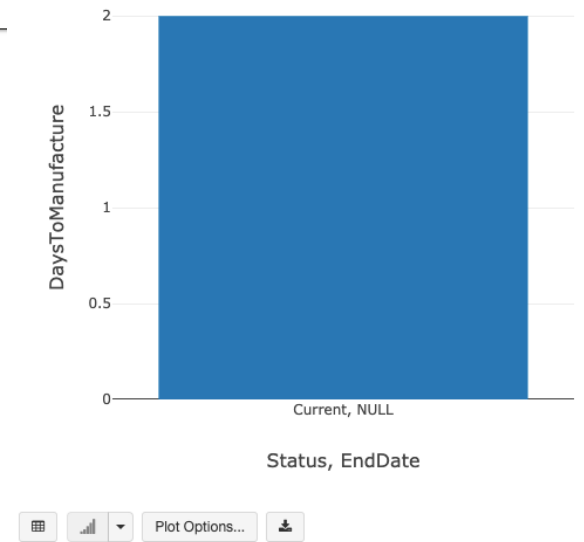
Showing all 5 rows.

Command took 0.35 seconds -- by [redacted] at 12/12/2020, 10:35:10 AM on db1

Shift+Enter to run shortcuts

1 select * from adproduct_csv limit 5

(1) Spark Jobs



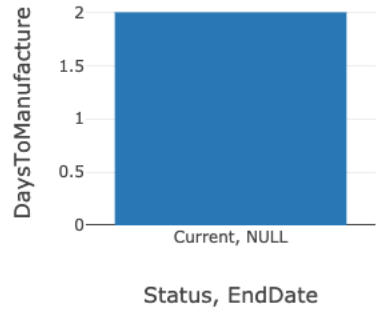
Customize Plot

All fields: ProductKey, ProductAlterna…, ProductSubcat…, WeightUnitMea…, SizeUnitMeasu…, EnglishProduct…, SpanishProduc…, FrenchProduct…, StandardCost, FinishedGoods…, Color

Keys: Status, EndDate

Series groupings:

Values: DaysToManufac…



- (•) Grouped
- ( ) Stacked
- ( ) 100% Stacked

Aggregation: SUM

Display type: Bar chart

[ ] Global color consistency

Cancel Apply

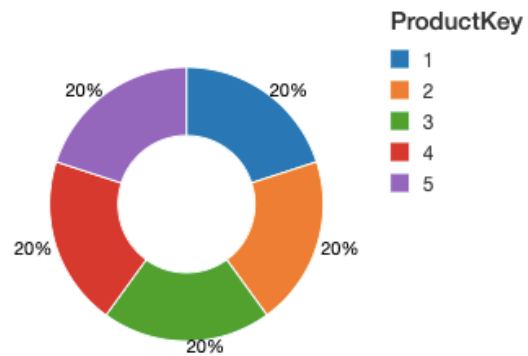
Customize Plot

All fields: ProductKey, ProductAlterna…, ProductSubcat…, WeightUnitMea…, SizeUnitMeasu…, EnglishProduct…, SpanishProduc…, FrenchProduct…, StandardCost, FinishedGoods…, Color

Keys: Produc…

Series groupings:

Values: DaysToManufac…

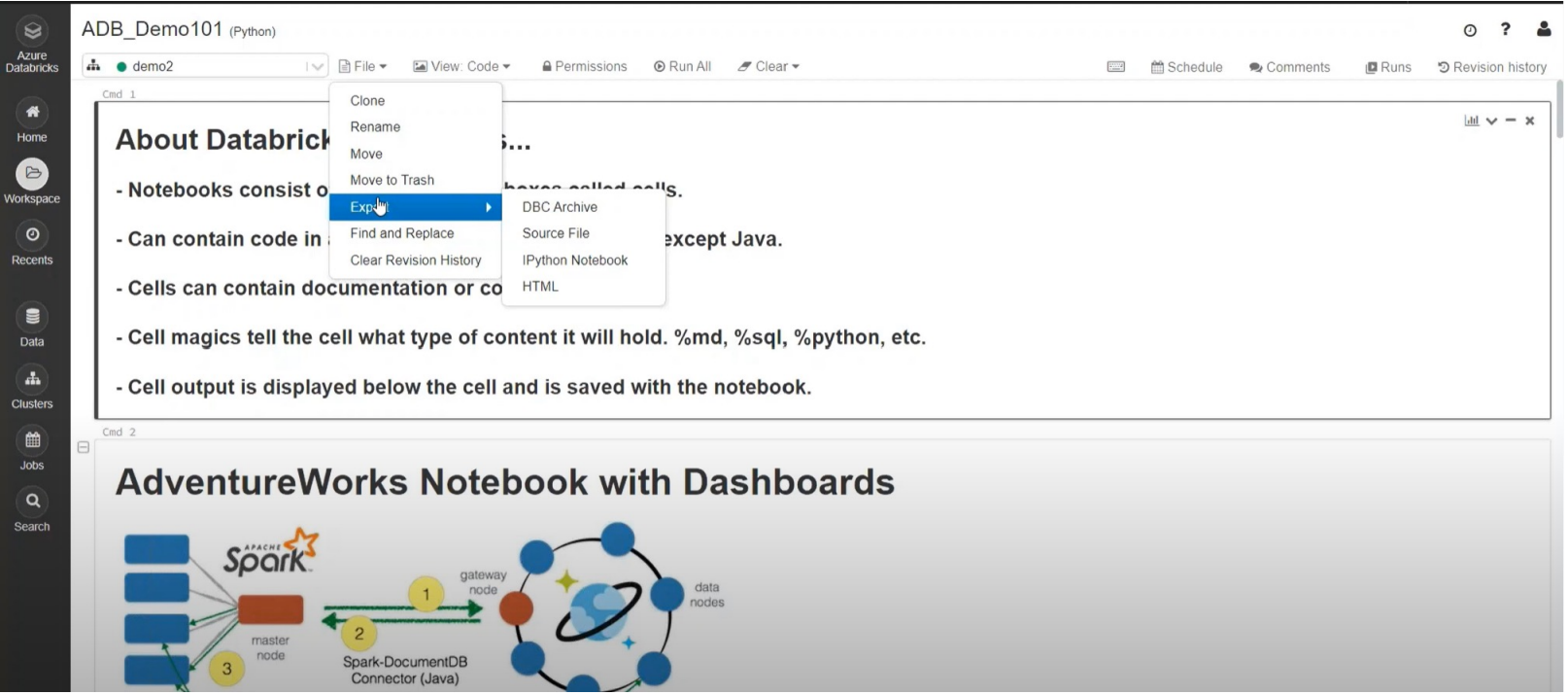


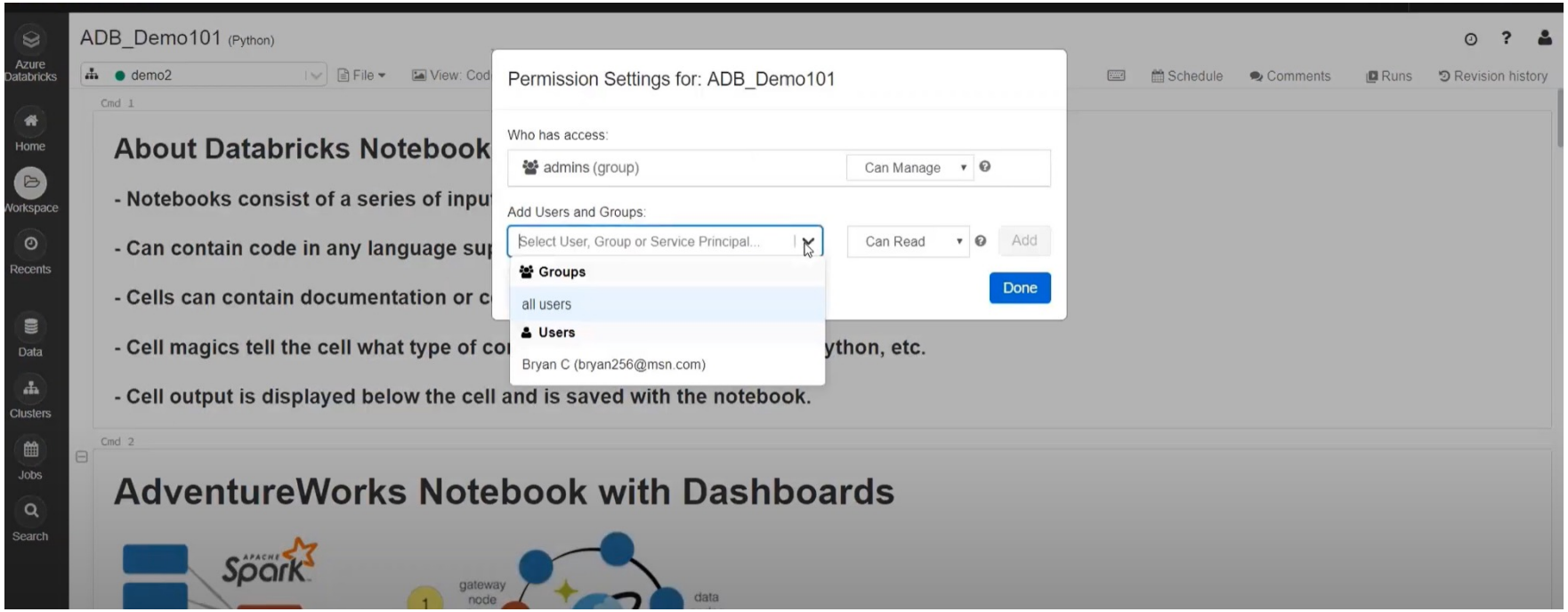
[x] Donut

Aggregation: COUNT

Display type: Pie chart

Cancel Apply


ADB_Demo101 (Python)
demo2
File
View: Code
Permissions
Run All
Clear
Schedule
Comments
Runs
Revision history
Clone
Rename
Move
Move to Trash
Export
Find and Replace
Clear Revision History
DBC Archive
Source File
IPython Notebook
HTML
Cmd 1
About Databrick...
- Notebooks consist o... boxes called cells.
- Can contain code in ... except Java.
- Cells can contain documentation or co...
- Cell magics tell the cell what type of content it will hold. %md, %sql, %python, etc.
- Cell output is displayed below the cell and is saved with the notebook.
Cmd 2
AdventureWorks Notebook with Dashboards
Spark
master node
gateway node
data nodes
Spark-DocumentDB Connector (Java)
Azure Databricks
Home
Workspace
Recents
Data
Clusters
Jobs
Search


ADB_Demo101 (Python)
demo2
File
View: Cod
Schedule
Comments
Runs
Revision history
Permission Settings for: ADB_Demo101
Who has access:
admins (group)
Can Manage
Add Users and Groups:
Select User, Group or Service Principal...
Can Read
Add
Groups
all users
Users
Bryan C (bryan256@msn.com)
Done
Cmd 1
About Databricks Notebook
- Notebooks consist of a series of inpu...
- Can contain code in any language su...
- Cells can contain documentation or c...
- Cell magics tell the cell what type of co... ython, etc.
- Cell output is displayed below the cell and is saved with the notebook.
Cmd 2
AdventureWorks Notebook with Dashboards
Spark
gateway node
data
Azure Databricks
Home
Workspace
Recents
Data
Clusters
Jobs
Search

ADB_Demo101 (Python)

demo2 | File | View: Code | Permissions | Run All | Clear | Schedule | Comments | Runs | Revision history

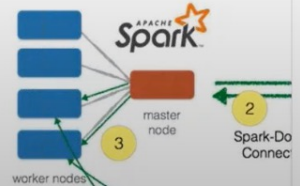
Cmd 1

About Databricks Not

- Notebooks consist of a serie
- Can contain code in any lang
- Cells can contain documenta
- Cell magics tell the cell what
- Cell output is displayed belo

Cmd 2

AdventureWorks



Edit mode

<Esc>	Switch to Command Mode
<Ctrl> <Alt> F	Find and Replace
<Shift> + <Enter>	Run command and move to next cell
<Alt> + <Enter>	Run command and insert new cell below
<Ctrl> + <Enter>	Run command
<Shift> + <Alt> + <Up>	Run all above commands (exclusive)
<Shift> + <Alt> + <Down>	Run all below commands (inclusive)
<Alt> + <Up> / <Down>	Move to previous/next cell
<Ctrl> + <Alt> + P	Insert a cell above
<Ctrl> + <Alt> + N	Insert a cell below
<Ctrl> + <Alt> + -	Split a cell at cursor
<Ctrl> + <Alt> + <Up>	Move a cell up
<Ctrl> + <Alt> + <Down>	Move a cell down
<Ctrl> + <Alt> + M	Toggle comments panel
<Ctrl> + <Alt> + C	Copy current cell
<Ctrl> + <Alt> + X	Cut current cell
<Ctrl> + <Alt> + V	Paste cell below
<Ctrl> + <Alt> + D	Delete current cell
<Up>	Move up or to previous cell
<Down>	Move down or to next cell
<Tab>	Autocomplete, indent selection
<Shift> + <Tab>	Unindent selection
<Ctrl> +] / [	Indent/Unindent selection

Command mode

<Enter>	Switch to Edit Mode
<Ctrl> <Alt> F	Find and Replace
<Shift> + <Enter>	Run command and move to next cell
<Ctrl> + <Enter>	Run command
<Shift> + <Alt> + <Up>	Run all above commands (exclusive)
<Shift> + <Alt> + <Down>	Run all below commands (inclusive)
D D	Delete current cell
<Shift> + D D	Delete current cell(skip prompt)
X	Cut current cell
C	Copy current cell
V	Paste cell below
<Shift> + V	Paste cell above
A	Insert a cell above
B	Insert a cell below
O	Toggle cell output
<Space>	Scroll down
<Shift> + <Space>	Scroll up
H	Toggle keyboard shortcuts menu
<Shift> + M	Merge with cell below
<Up> / P / K	Move to previous cell
<Down> / N / J	Move to next cell
<Ctrl> + <Click>	Select multiple cells
L	Toggle line numbers

ADB_Demo101 (Python)

demo2 | File | View: Code | Permissions | Run All | Clear | Schedule | Comments | Runs | Revision history

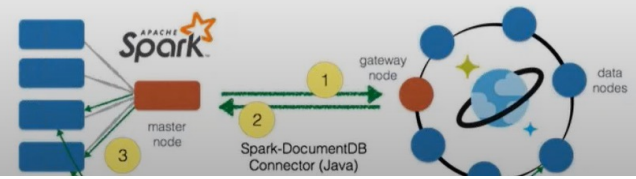
Cmd 1

About Databricks Notebooks...

- Notebooks consist of a series of input boxes called cells.
- Can contain code in any language supported by Spark except Java.
- Cells can contain documentation or code.
- Cell magics tell the cell what type of content it will hold. %md, %sql, %python, etc.
- Cell output is displayed below the cell and is saved with the notebook.

Cmd 2

AdventureWorks Notebook with Dashboards




ADB_Demo101 (Python)
demo2
File
View: Code
Permissions
Run All
Clear
Schedule
Comments
Runs
Revision history
Cmd 1
%md
About Databricks Notebooks...
- Notebooks consist of a series of input boxes called cells.
- Can contain code in any language supported by Spark except Java.
- Cells can contain documentation or code.
- Cell magics tell the cell what type of content it will hold. %md, %sql, %python, etc.
- Cell output is displayed below the cell and is saved with the notebook.
Cmd 2

Getting Started with Databricks

- Or Create a free Community Account on Databricks.com. This has the following limitations.
  - No Azure Resources Available
  - Only the Cluster Head Node is created, no workers.
  - 6 GB files.
  - Just for learning.

https://databricks.com/try-databricks

References

- Databricks Documentation
- https://docs.databricks.com/

- Apache Spark Documentation
- https://spark.apache.org/documentation.html