



PySpark and Matrix Completion (Netflix Challenge) with MLlib

Vittorio Bisin

Python in Data Science

1. High level syntax and easy to learn
2. Interfaces easily with C/C++ allowing for high performance libraries
3. Extensive library of packages
 - A. `numpy/scipy/matplotlib`
 - Typical MATLAB functionality: fast array operations, scientific functions, plotting
 - B. `pandas`
 - similar to R's `data.frame`
 - C. `scikit-learn/statsmodels`
 - Implementation of machine learning algorithms
 - D. `nltk`
 - natural language processing

PySpark Architecture

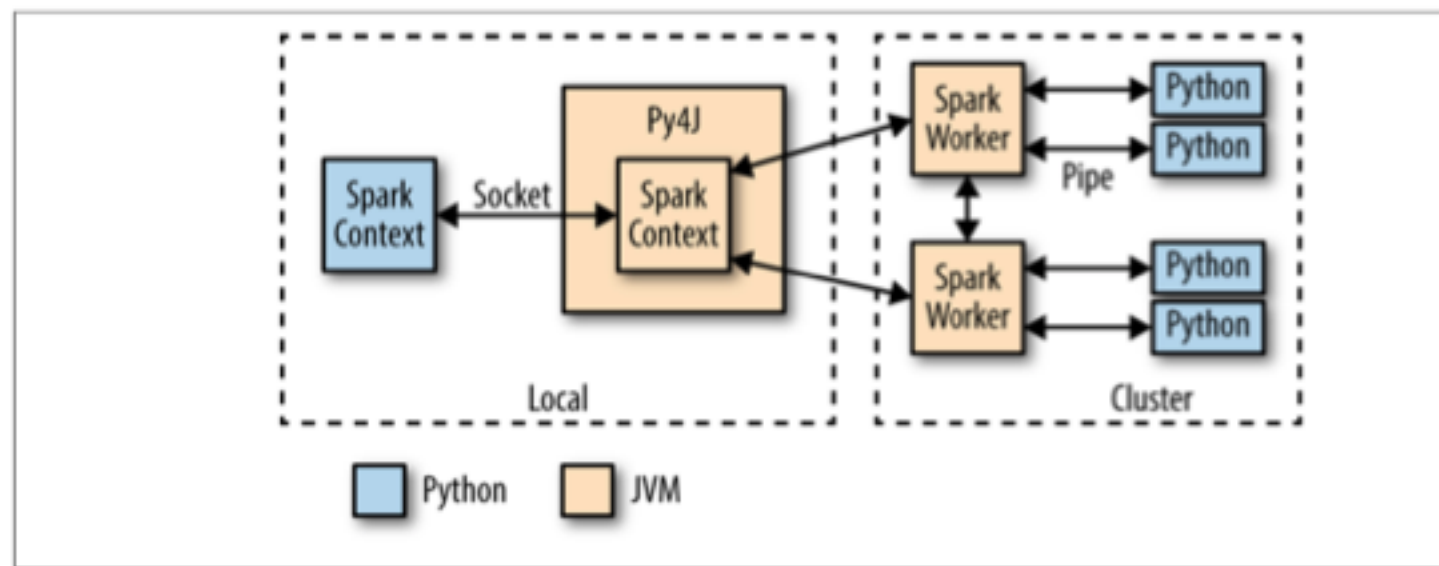


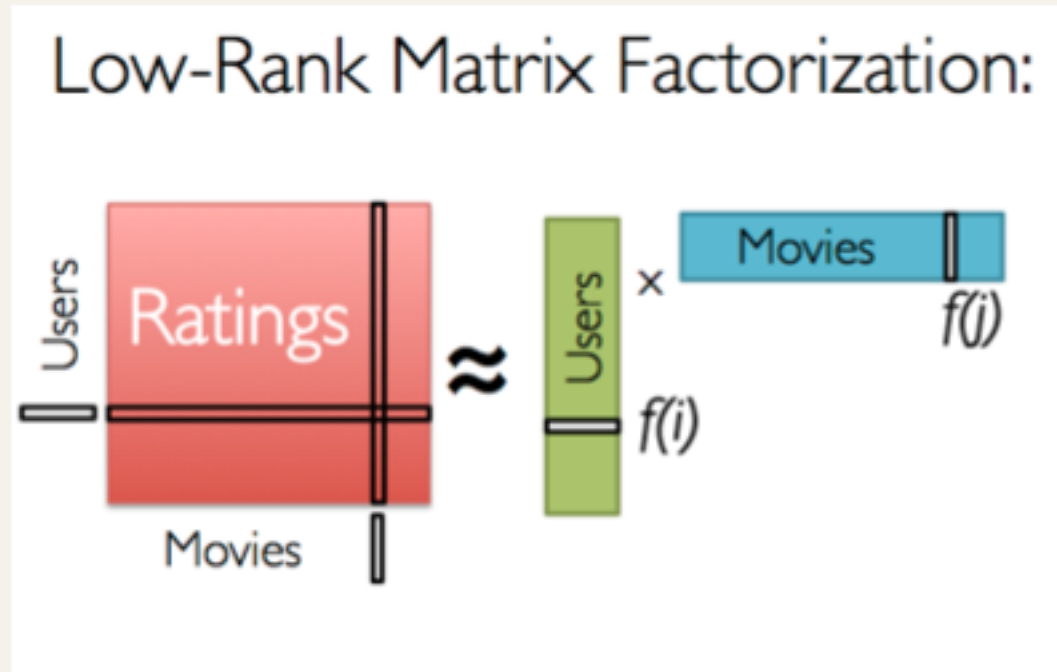
Figure 11-1. PySpark internal architecture

Figure and notes taken from Advanced Analytics with Spark: chapter 11

1. When PySpark's Python interpreter starts it also starts a JVM, and communicates with it using a socket.
2. The JVM is used as the Spark driver, loading a JavaSparkContext that communicates with the Spark executors across the cluster.
3. The Python API calls to the SparkContext object are translated into Java API calls to the JavaSparkContext
 - A. A Python RDD in the local PySpark client corresponds to a Python RDD object in the local JVM


Low Rank Matrix Factorization

1. Approximate high rank matrix with a low dimensional one (e.g. SVD)
 - A. Becomes necessary with training sets composed of 10 billion entries (see Netflix challenge on the next slide)
2. Normally used as a recommender system - where a user has a sparse representation and we want to predict its unknown entries



The Netflix Challenge

1. Collaborative filtering and matrix completion with user rating data.
 - A. \$1 million prize
2. Very sparse matrix since each user has not rated every possible movie.
 - B. 1% of matrix entries nonzero
 - C. Adapts well to low rank matrix factorization



Netflix Prize

Home Rules Leaderboard Updates

Leaderboard

Showing Team Scores: [Click Here to View All Scores](#)

Displaying Team: [All](#) > [Netflix](#)

Rank	Team Name	Best Team Score	% Improvement	Best Submit Time
Previous Title: RMSE = 0.8627 - Winning Team: Netflix's Proprietary Chase				
1	Netflix's Proprietary Chase	0.8607	10.00	2009-07-26 16:18:29
2	The Goodwin	0.8607	10.00	2009-07-26 16:18:29
3	Google Team	0.8602	9.80	2009-07-12 21:24:40
4	Opera Software and Microsoft Limited	0.8600	9.64	2009-07-15 11:12:51
5	VeriSign, Inc.	0.8591	9.21	2009-07-15 00:32:30
6	Proprietary Chase	0.8591	9.17	2009-06-24 12:09:56
7	Netflix's BigChase	0.8601	9.10	2009-05-13 09:10:28
8	Chase	0.8612	9.08	2009-07-24 17:18:41
9	Chase	0.8602	9.00	2009-07-12 15:11:51
10	BigChase	0.8603	8.97	2009-06-07 12:13:59
11	Opera Software	0.8623	8.97	2009-07-24 00:34:37
12	Netflix	0.8626	9.00	2009-07-26 17:19:11
Previous Title: RMSE = 0.8627 - Winning Team: Netflix in BigChase				
13	Intelligence	0.8642	9.27	2009-07-15 14:53:53
14	Chase	0.8643	9.28	2009-06-22 18:31:32
15	Chase	0.8601	9.10	2009-08-21 19:24:53
16	VeriSign, Inc.	0.8603	9.10	2009-07-15 15:53:54
17	Chase in a BigChase	0.8602	9.08	2009-08-24 15:02:54
18	Chase in a BigChase	0.8606	9.00	2009-08-07 17:19:17
19	Chase in a BigChase	0.8606	9.00	2009-07-28 18:03:54
20	Chase	0.8606	9.00	2009-08-21 19:24:53
Previous Title: RMSE = 0.8622 - Winning Team: BigChase				
Current status: RMSE = 0.8626				

There are currently 51027 comments on 47 626 teams from 180 different countries. We have received 4674 valid submissions from 5169 different teams. 3 submissions in the last 24 hours.

Questions about interpreting the leaderboard? Please read [this](#).

[illegible]

Methods 1

1. Original Matrix is NxM (N users and M movies)
2. Fix a $K \ll N, M$
 - A. K is the rank of the low dimensional approximation and the number of "concepts"
3. Let each user u be summarized as a k dimensional vector, each movie i by a k dimensional vector
4. Objective function is derived from minimizing the least squares error of the rating (LHS) and regularization (RHS)
 - B. Related to Tikhonov regularization/Ridge Regression

$$X = \begin{bmatrix} | & \cdots & | \\ x_1 & \cdots & x_n \\ | & \cdots & | \end{bmatrix}, Y = \begin{bmatrix} | & \cdots & | \\ y_1 & \cdots & y_m \\ | & \cdots & | \end{bmatrix}$$

$$\min_{X,Y} \sum_{r_{ui} \text{ observed}} (r_{ui} - x_u^T y_i)^2 + \lambda (\sum_u \|x_u\|^2 + \sum_i \|y_i\|^2)$$

Equations taken from "Distributed Algorithms and Optimization"
Stanford Spring 2015 lecture slides

$x_{1,1}$ = how much user 1 likes topic 1 (e.g. action movies)

$y_{1,1}$ = how much movie 1 is about topic 1 (e.g. how much action it contains)

Methods 2: Alternating Least Squares

“Coordinate Descent for matrices”

1. Unfortunately the objective function is not convex
 - A. The LHS is the problem
2. However it is convex w.r.t. X and Y separately
3. ALS \rightarrow minimize the objective function w.r.t X and Y separately, and repeat until convergence
4. Easily parallelizable (great with Spark)

$$\min_{X,Y} \sum_{r_{ui} \text{ observed}} (r_{ui} - x_u^T y_i)^2 + \lambda (\sum_u \|x_u\|^2 + \sum_i \|y_i\|^2)$$

My Demo

1. Used the MovieLens dataset
2. The best low dimensional matrix approximation was rank 4
3. Using the test data, the root mean squared error was 0.97
4. I used the model to predict my own movie recommendations

My Ratings:

Toy Story (1995) : 5

Casino (1995): 6

Ace Ventura: When Nature Calls (1995) 1

Babe (1995): 1

Il Postino (1994): 9

Taxi Driver (1976): 10

Show Girls (1995): 6

Clerks (1994): 8

Star Wars (1997): 6

Four Weddings and a Funeral (1994): 7

The Flintstones (1994): 3

Timecop (1994): 2

Pulp Fiction (1994): 10

The Godfather (1972): 10

The Usual Suspects (1995): 7

Results

1. Algorithm predicts my ratings for all movies in the dataset
2. Then recommends 25 movies with the highest predicted rating
 - A. Only selecting from movies that have been rated by other users a minimum of 25 times

Recommendations:

Sex: 9.6
Three Colors: Red (1994): 9.5
Brokeback Mountain (2005): 9.4
Manhattan (1979): 9.4
Network (1976): 9.3
Annie Hall (1977): 9.2
Chinatown (1974): 9.2
Deer Hunter (1978): 9.2
Adaption (2002): 9.1
Citizen Kane (1941): 9.1
Vertigo (1958): 9.1
2001: A Space Odyssey (1968): 9
Bound (1996): 9
Apocalypse Now (1979): 9
Midnight Cowboy (1969): 9
Harold and Maude (1971): 8.9
Rosemary's Baby (1968): 8.8
Dr. Strangelove or: How I Learned to Stop Worrying and
Love the Bomb (1964): 8.8
Psycho (1960): 8.8
All About Eve (1950): 8.8
Third Man: 8.8
Graduate: (8.8)
Fargo (1996): 8.8
Three Colors: Blue (1993): 8.7
Moonstruck (1987): 8.7