Large Scale Recommender Systems in Spark

Kirk Hunter, Mrunmayee H. Bhagwat, Swetha Reddy

Data & Tools

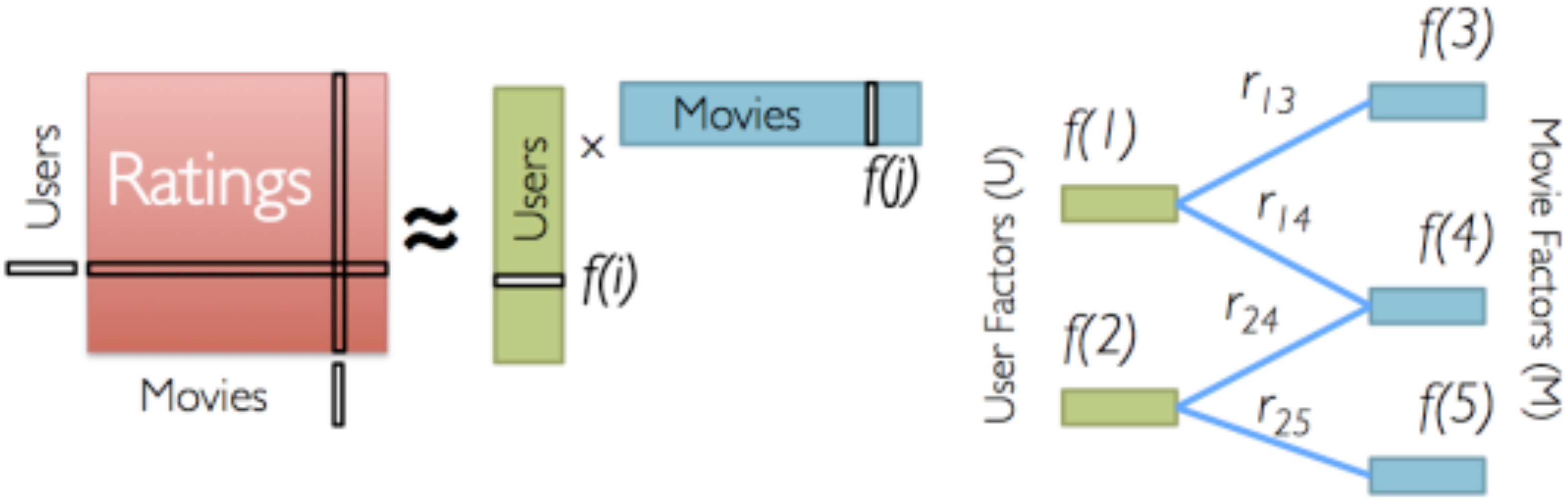
- Yahoo! Music Ratings (user id, song id, rating)
- Train: 700 million ratings, 1.8 million users, 136K songs
- Test: 18 million ratings, 1.8 million users, 136K songs
- Stored data in S3
- 5 node cluster running Spark on AWS EMR

Methods

- Alternating Least Squares (ALS)
- Locality Sensitive Hashing (LSH)

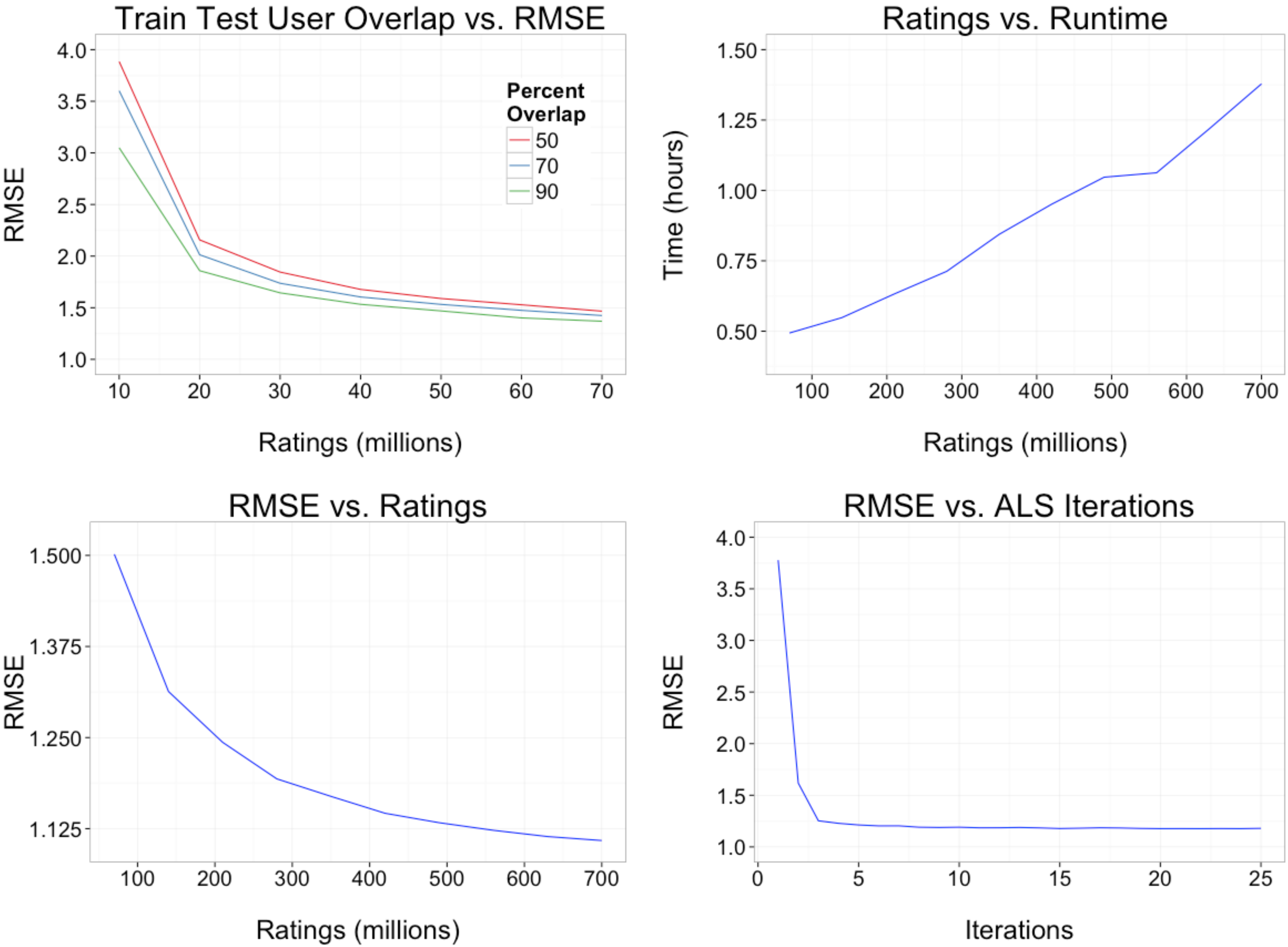
Alternating Least Squares (ALS)

Low-Rank Matrix Factorization:

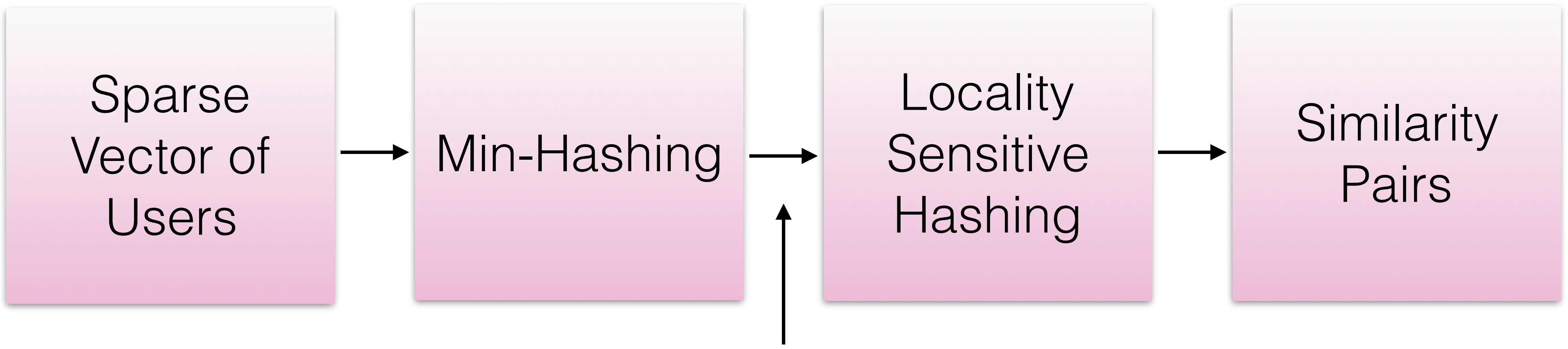


Iterate:

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


Train Test User Overlap vs. RMSE
RMSE
4.0
3.5
3.0
2.5
2.0
1.5
1.0
Percent Overlap
50
70
90
10
20
30
40
50
60
70
Ratings (millions)
Ratings vs. Runtime
Time (hours)
1.50
1.25
1.00
0.75
0.50
100
200
300
400
500
600
700
Ratings (millions)
RMSE vs. Ratings
RMSE
1.500
1.375
1.250
1.125
100
200
300
400
500
600
700
Ratings (millions)
RMSE vs. ALS Iterations
RMSE
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0
5
10
15
20
25
Iterations

Locality Sensitive Hashing (LSH)



Sparse Vector of Users

Min-Hashing

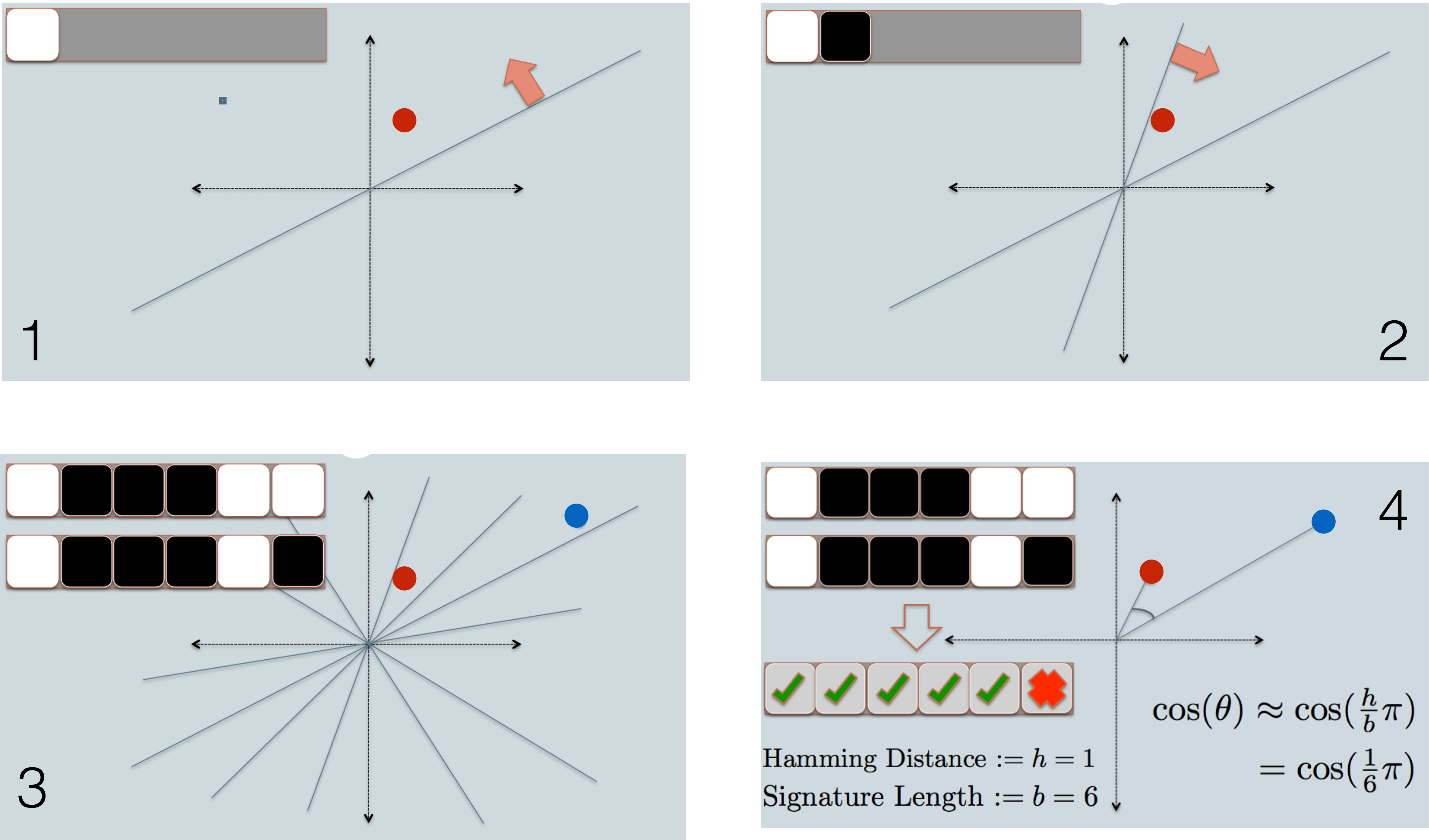
Locality Sensitive Hashing

Similarity Pairs

Signatures

Short integer vectors that
represent sets, and
reflect their similarity

LSH - Cosine Similarity

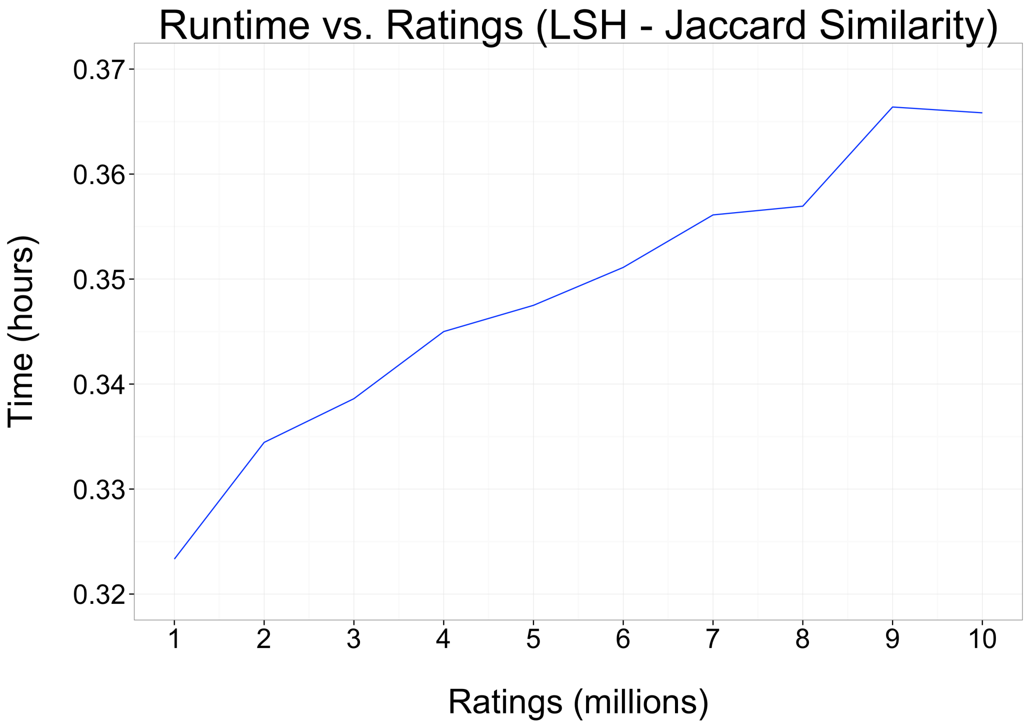


Hamming Distance $:= h = 1$

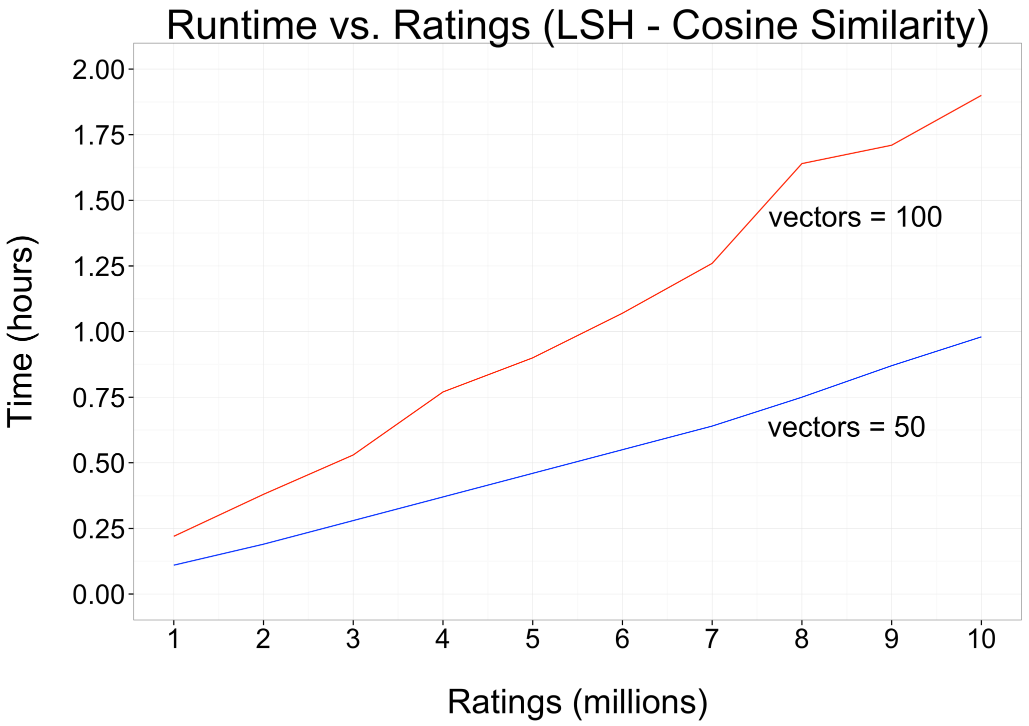
Signature Length $:= b = 6$

$$\cos(\theta) \approx \cos(\tfrac{h}{b}\pi)$$
$$= \cos(\tfrac{1}{6}\pi)$$

LSH Results & Output



Runtime vs. Ratings (LSH - Jaccard Similarity)



Runtime vs. Ratings (LSH - Cosine Similarity)

Summary

- Simple collaborative filtering doesn’t scale
- Went from $O(n^2)$ to $O(n)$ with ALS and LSH
- LSH currently not available in Spark’s MLlib

Thank You!