SPARK+AI SUMMIT EUROPE

Building an Implicit Recommendation Engine

Sophie Watson @sophwats sophie@redhat.com

#SAISDS12

SPARK+AI
SUMMIT EUROPE

Building an Implicit Recommendation Engine

Sophie Watson @sophwats sophie@redhat.com

#SAISDS12

NETFLIX
hulu
movies

goodreads
books

TESCO
food

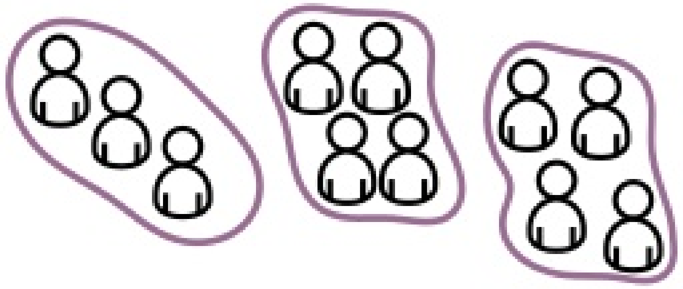
Expedia®
hotels

music

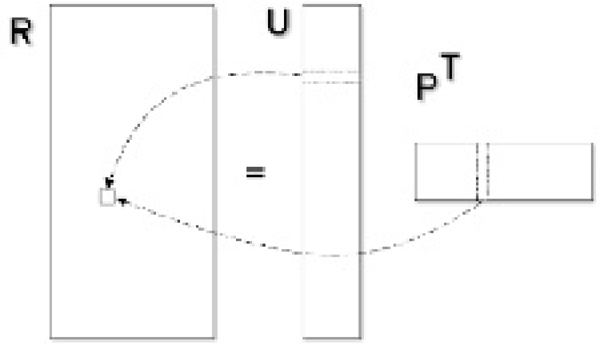
people to follow

Outline

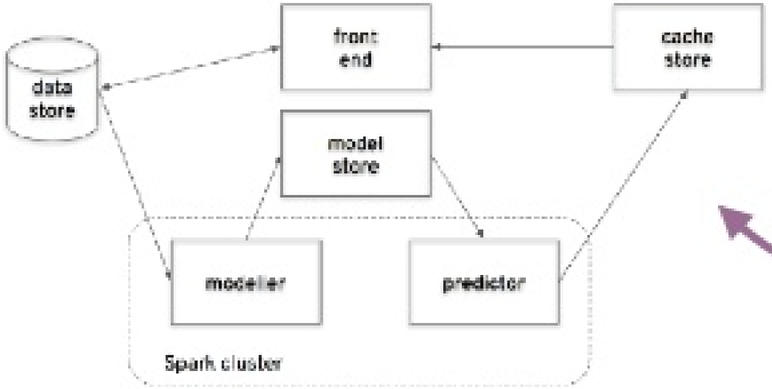
Collaborative Filtering



Alternating Least Squares



APACHE Spark



data store

front end

cache store

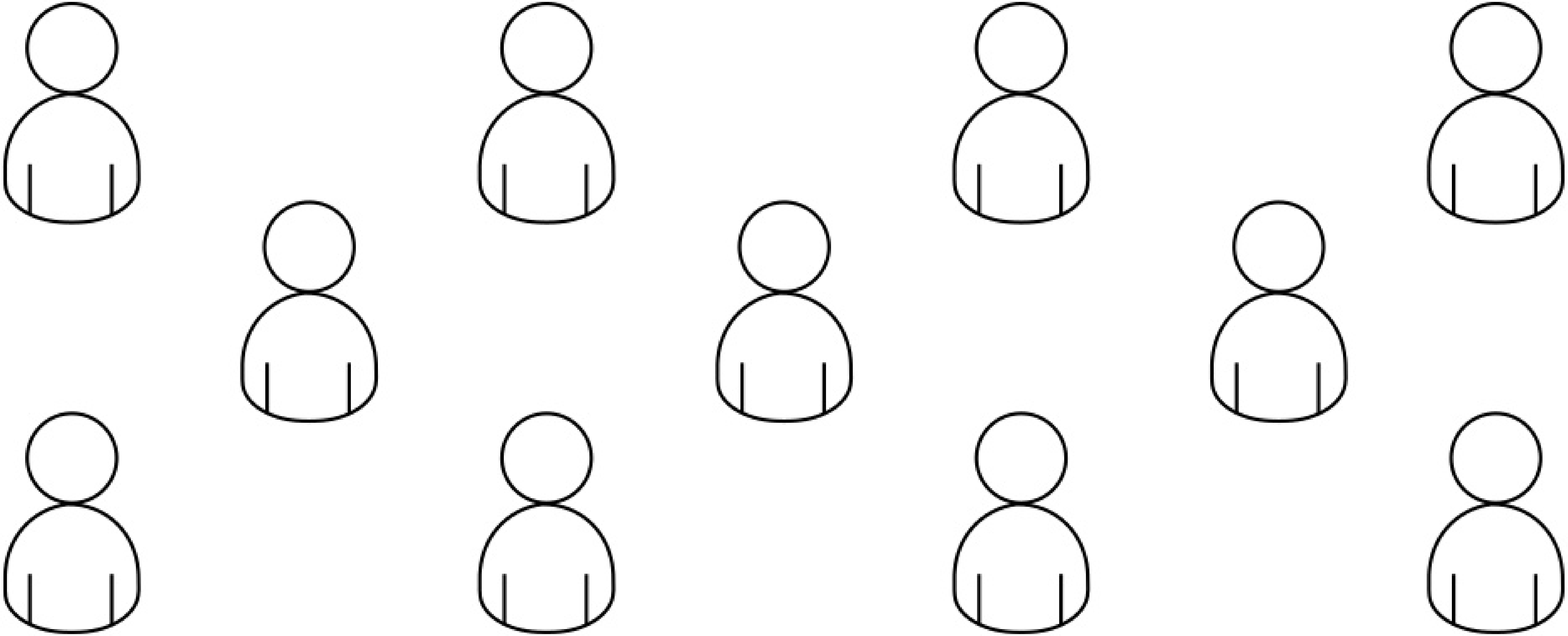
model store

modeller

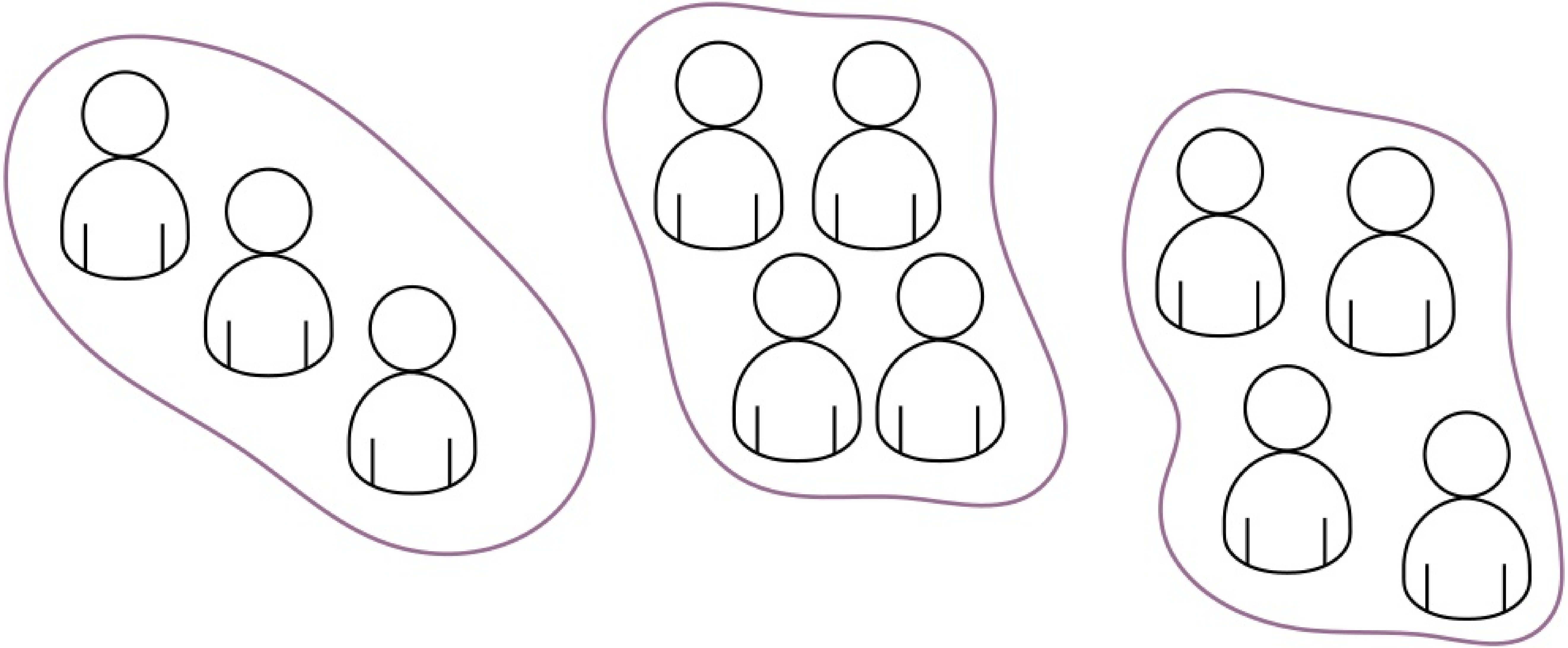
predictor

Spark cluster

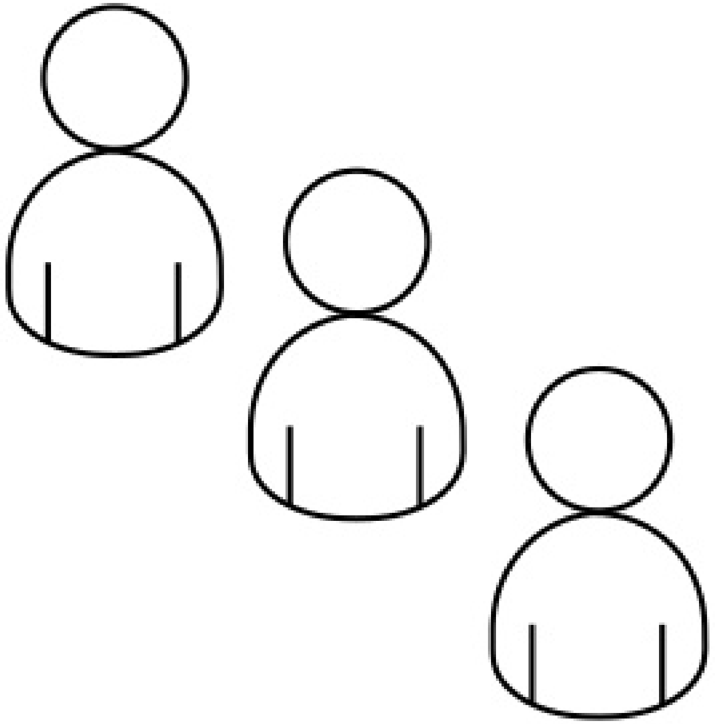
Collaborative Filtering



Collaborative Filtering



Collaborative Filtering



Collaborative Filtering

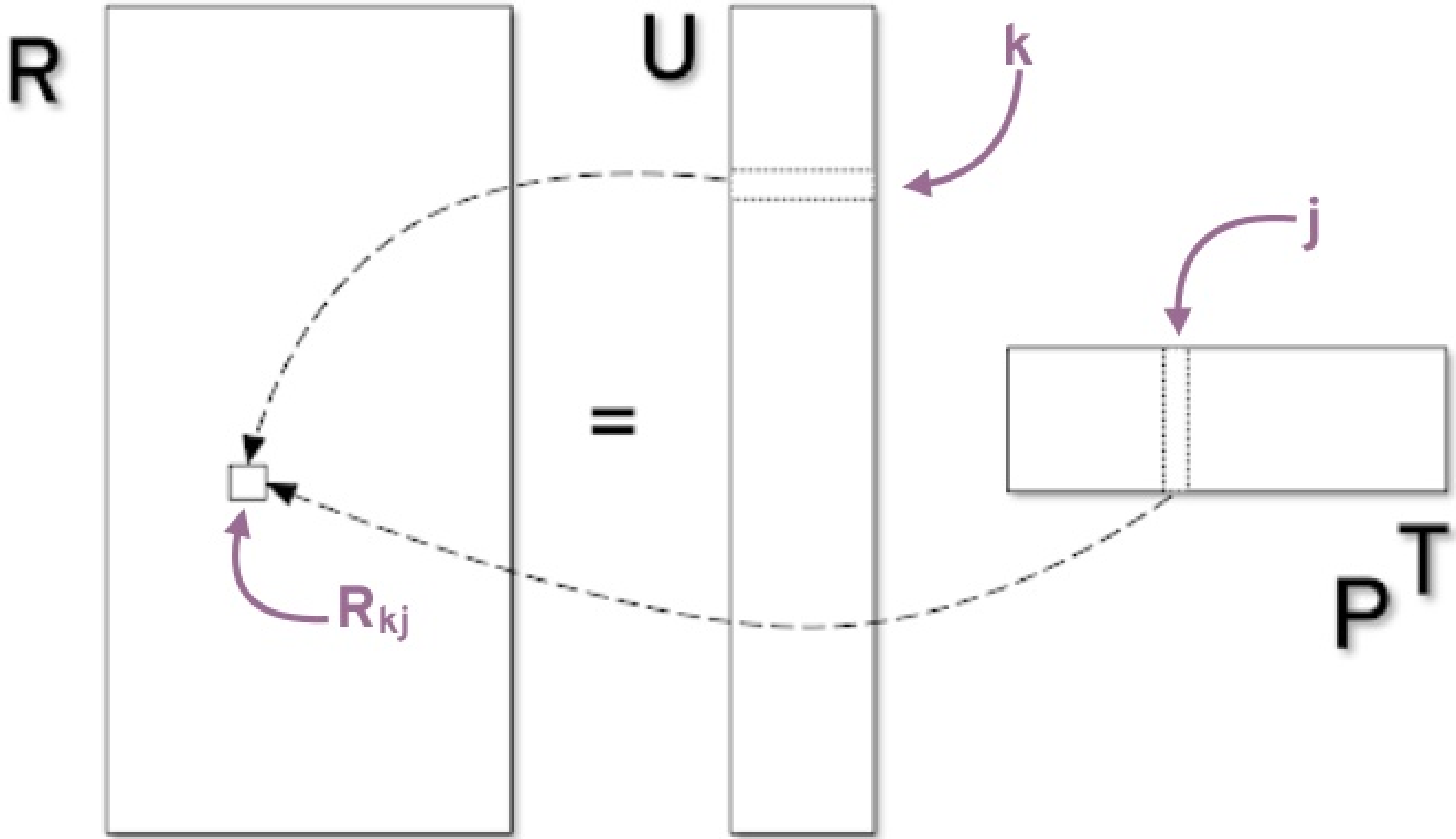
	1	2	3
Product A	5☆	5☆	5☆
Product B	3☆	3☆	3☆
Product C	5☆	3☆	?☆

Collaborative Filtering

	1	2	3
Product A	5☆	5☆	5☆
Product B	3☆	3☆	3☆
Product C	5☆	3☆	4☆

Alternating Least Squares

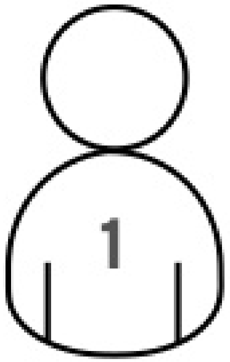
$$R = \begin{array}{cc} \begin{matrix} \text{user 1} & \text{user 2} & \text{user 3} & \cdots & \text{user N} \end{matrix} & \\ \begin{bmatrix} 1 & 4.5 & ? & \cdots & 3 \\ ? & 3 & 3 & \cdots & 4 \\ 5 & 3 & ? & \cdots & ? \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 2 & 4 & 1 & \cdots & ? \end{bmatrix} & \begin{matrix} \text{product 1} \\ \text{product 2} \\ \text{product 3} \\ \vdots \\ \text{product M} \end{matrix} \end{array}$$


R
U
k
j
=
R_{kj}
P^T

Alternating Least Squares

$$R = \begin{array}{cc} \begin{array}{ccccc} \text{user 1} & \text{user 2} & \text{user 3} & \cdots & \text{user N} \end{array} & \\ \left[\begin{array}{ccccc} 1 & 4.5 & 3.8 & \cdots & 3 \\ 3.2 & 3 & 3 & \cdots & 4 \\ 5 & 3 & 3.4 & \cdots & 3.1 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 2 & 4 & 1 & \cdots & 2.7 \end{array}\right] & \begin{array}{c} \text{product 1} \\ \text{product 2} \\ \text{product 3} \\ \vdots \\ \text{product M} \end{array} \end{array}$$

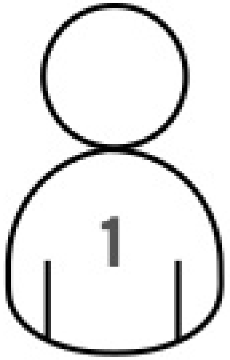
Implicit Data



Song A

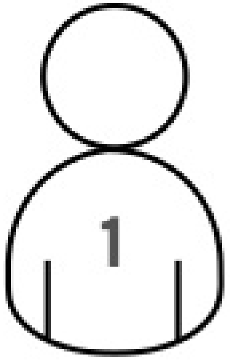
1 play

Implicit Data



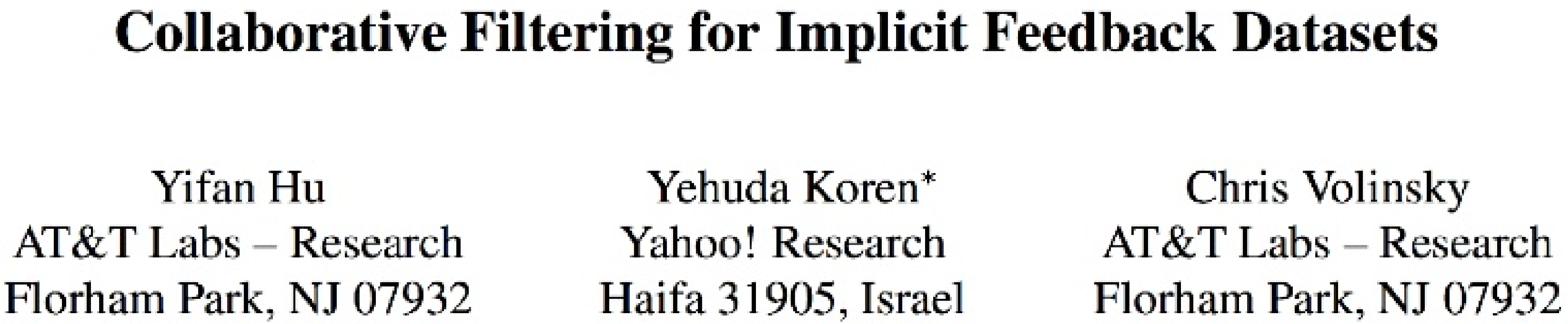
Song A	1 play
Song B	**0 plays**

Implicit Data



Song A	1 play
Song B	0 plays
Song C	**100 plays**

Implicit Alternating Least Squares



Collaborative Filtering for Implicit Feedback Datasets

Yifan Hu
AT&T Labs – Research
Florham Park, NJ 07932

Yehuda Koren*
Yahoo! Research
Haifa 31905, Israel

Chris Volinsky
AT&T Labs – Research
Florham Park, NJ 07932

Implicit Alternating Least Squares

The aim:

$p_{ui} \in (0,1)$ **preference**

Implicit Alternating Least Squares

$p_{ui} \in (0,1)$ **preference**

The recorded data:

$r_{ui} \in \mathbb{R}$ **recording**

Implicit Alternating Least Squares

$p_{ui} \in (0,1)$ **preference**

$r_{ui} \in \mathbb{R}$ **recording**

The recorded data:

$$p_{ui} = \begin{cases} 1 & \text{if} \quad r_{ui} > 0 \\ 0 & \text{if} \quad r_{ui} = 0 \end{cases}$$

Implicit Alternating Least Squares

Confidence:

$$C_{ui} = 1 + \alpha r_{ui}$$

$p_{ui} \in (0,1)$ **preference**

$r_{ui} \in \mathbb{R}$ **recording**

$$p_{ui} = \begin{cases} 1 & \text{if } r_{ui} > 0 \\ 0 & \text{if } r_{ui} = 0 \end{cases}$$

Implicit Alternating Least Squares

$p_{ui} \in (0,1)$ **preference**

$r_{ui} \in \mathbb{R}$ **recording**

$$p_{ui} = \begin{cases} 1 & \text{if } r_{ui} > 0 \\ 0 & \text{if } r_{ui} = 0 \end{cases}$$

Confidence:

$$C_{ui} = 1 + \alpha r_{ui}$$

tuning parameter

Implicit Alternating Least Squares

$p_{ui} \in (0,1)$ **preference**

$r_{ui} \in \mathbb{R}$ **recording**

Minimisation:

$$C_{ui}\left(p_{ui} - U_u X_i^T\right)$$

$$p_{ui} = \begin{cases} 1 & \text{if } r_{ui} > 0 \\ 0 & \text{if } r_{ui} = 0 \end{cases}$$

user vector

Item vector

What does Spark offer?

```python
from pyspark.mllib.recommendation import ALS
```

Data

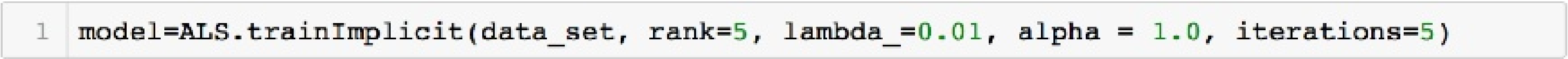
Lastfm dataset

17 million recordings / 360k users / 200k artists

```
['00000c289a1829a808ac09c00daf10bc3c4e223b',
 '8bfac288-ccc5-448d-9573-c33ea2aa5c30',
 'red',
 'hot',
 'chili',
 'peppers',
 '691'],
```

Building the model

(user id, product id, recording)

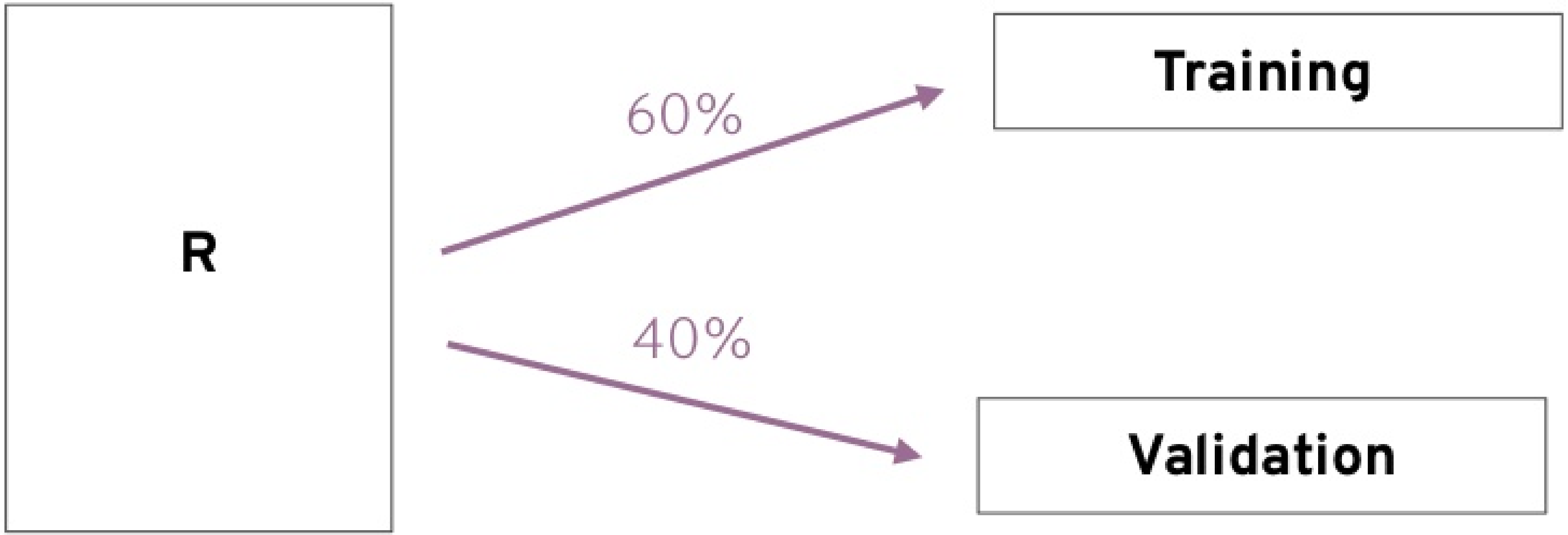


```
model=ALS.trainImplicit(data_set, rank=5, lambda_=0.01, alpha = 1.0, iterations=5)
```

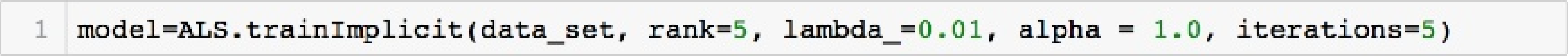
tuning parameters

Tuning Parameters

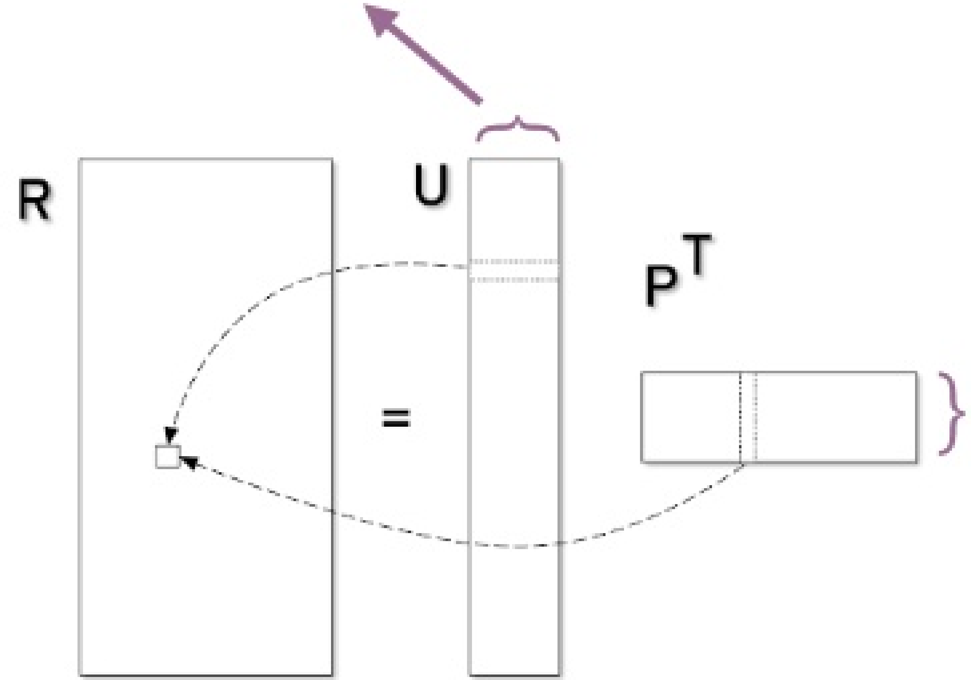
```
model=ALS.trainImplicit(data_set, rank=5, lambda_=0.01, alpha = 1.0, iterations=5)
```



Rank

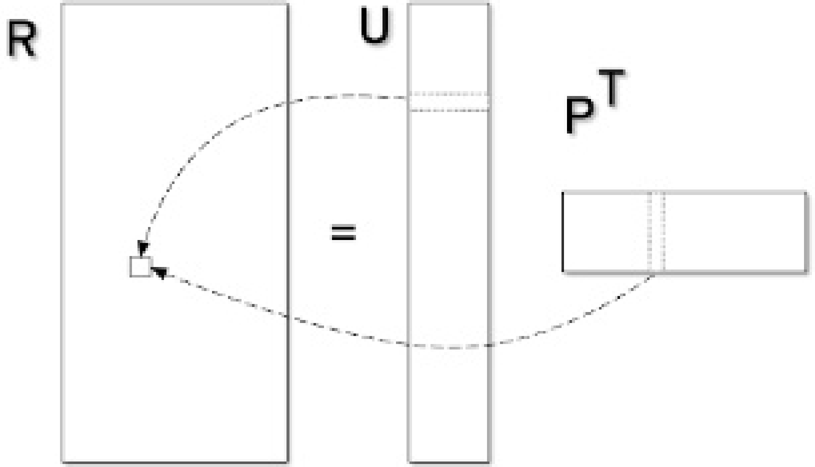


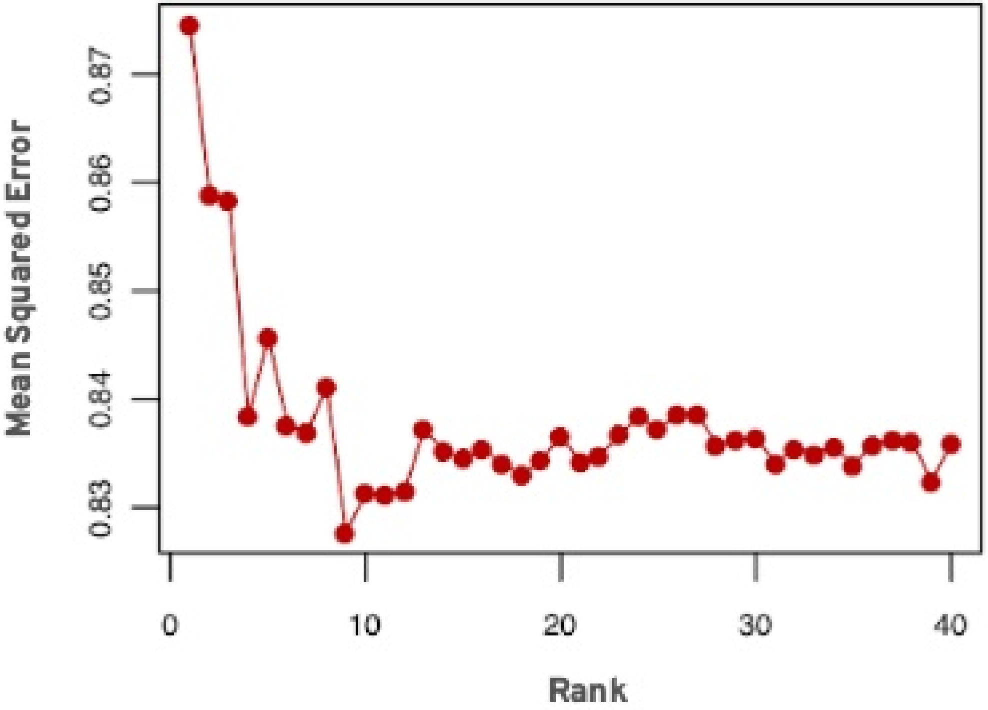
```
model=ALS.trainImplicit(data_set, rank=5, lambda_=0.01, alpha = 1.0, iterations=5)
```



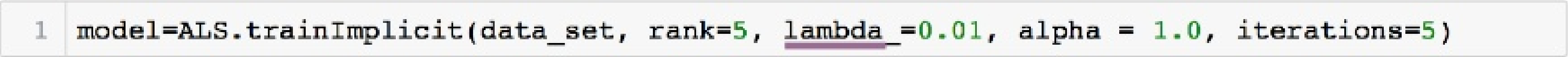
Rank

```
model=ALS.trainImplicit(data_set, rank=5, lambda_=0.01, alpha = 1.0, iterations=5)
```





Lambda

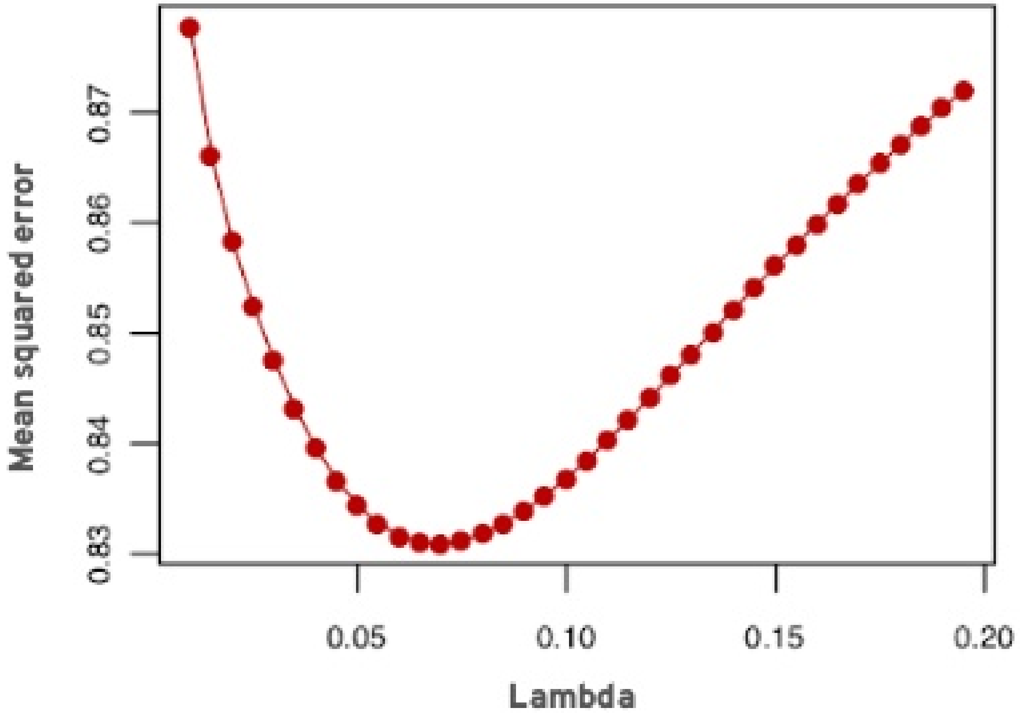


```
model=ALS.trainImplicit(data_set, rank=5, lambda_=0.01, alpha = 1.0, iterations=5)
```

Minimisation:

$$C_{ui}\left(p_{ui} - U_u X_i^T\right)$$

$$+ \lambda \left(\| U \| + \| X \| \right)$$



Alpha

```
model=ALS.trainImplicit(data_set, rank=5, lambda_=0.01, alpha = 1.0, iterations=5)
```

$r_{ui} \in \mathbb{R}$ **recording**

$$C_{ui} = 1 + \alpha r_{ui}$$

Alpha

```
model=ALS.trainImplicit(data_set, rank=5, lambda_=0.01, alpha = 1.0, iterations=5)
```

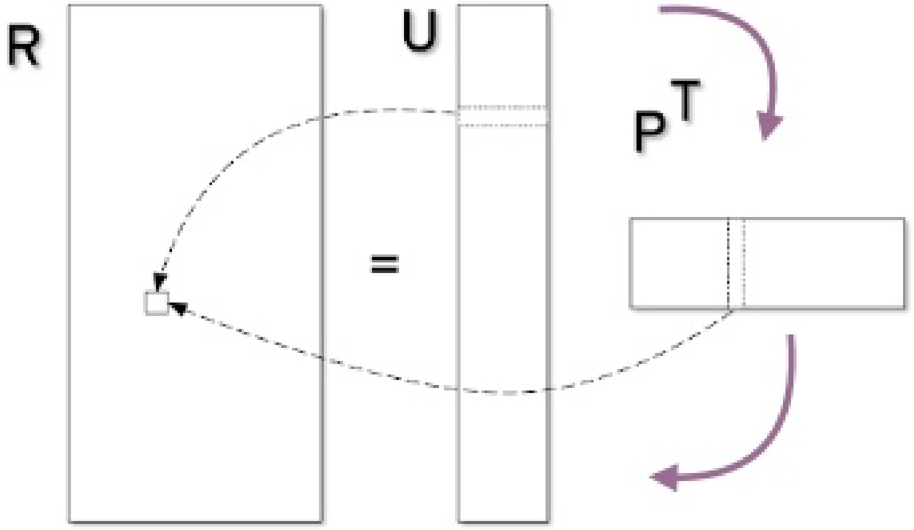
$r_{ui} \in \mathbb{R}$ recording

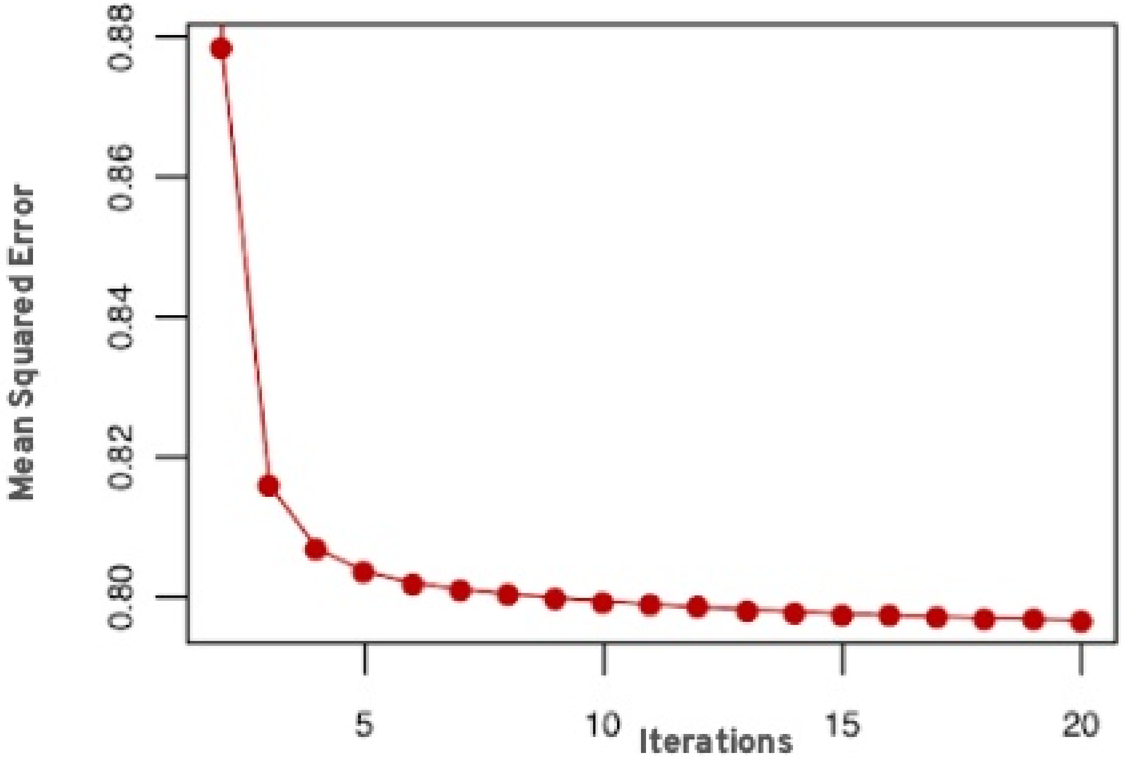
$$C_{ui} = 1 + \alpha r_{ui}$$

relates to scale of recording

Iterations

```
model=ALS.trainImplicit(data_set, rank=5, lambda_=0.01, alpha = 1.0, iterations=5)
```



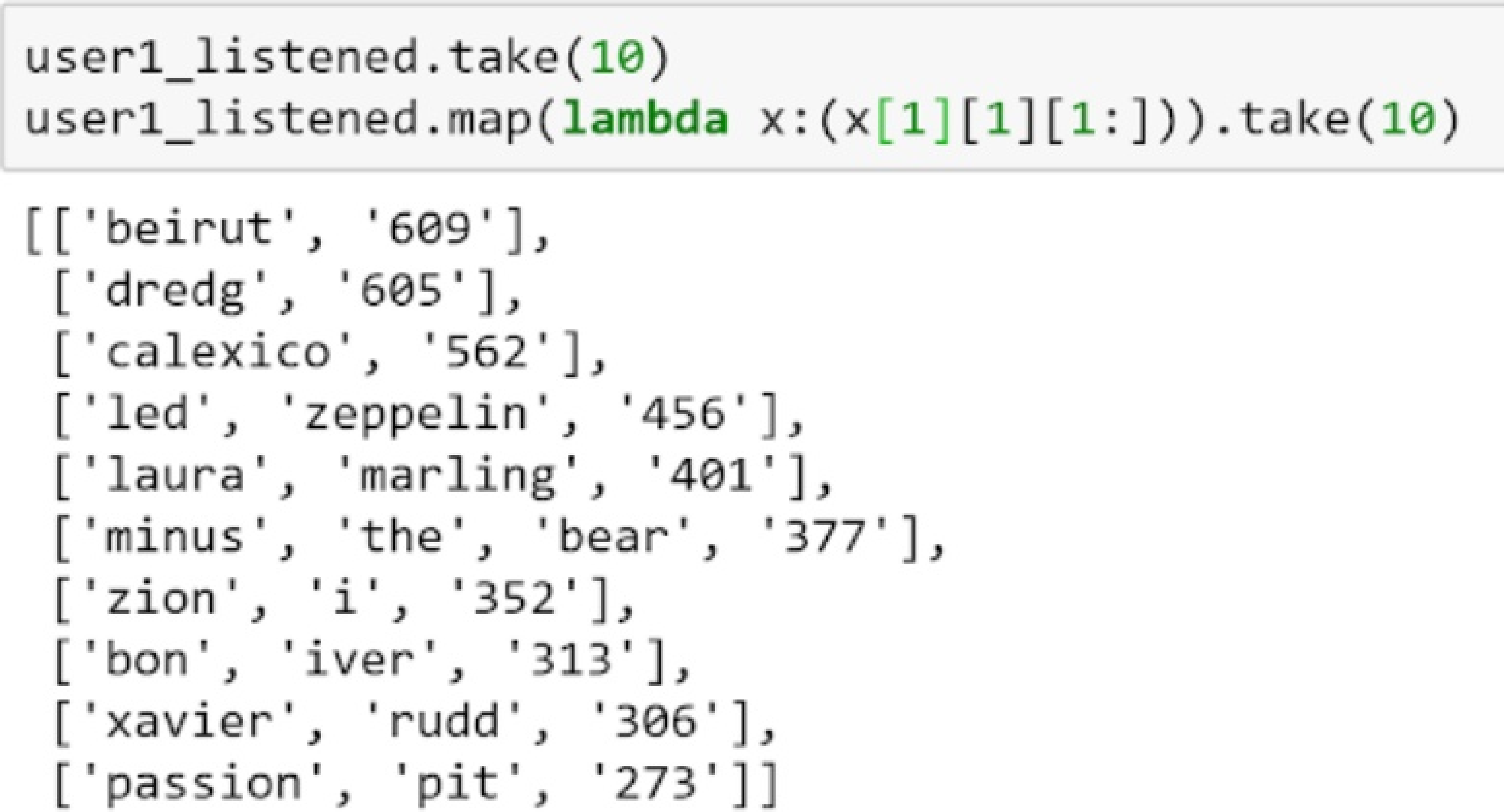


Making Predictions

```
model=ALS.trainImplicit(data_set, rank=5, lambda_=0.01, alpha = 1.0, iterations=5)
```

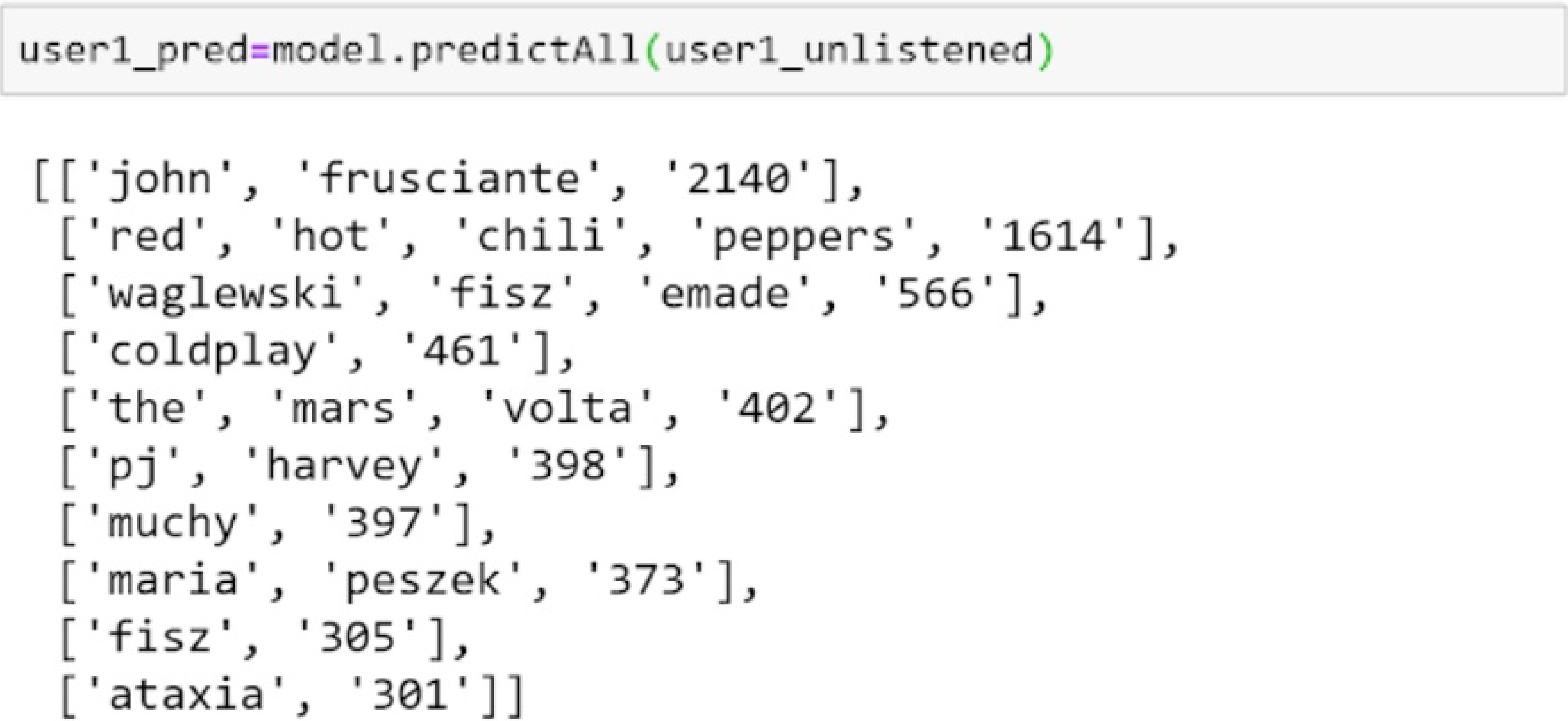
```
predictions = model.predictAll(zero_listens)
```

(user id, item id)



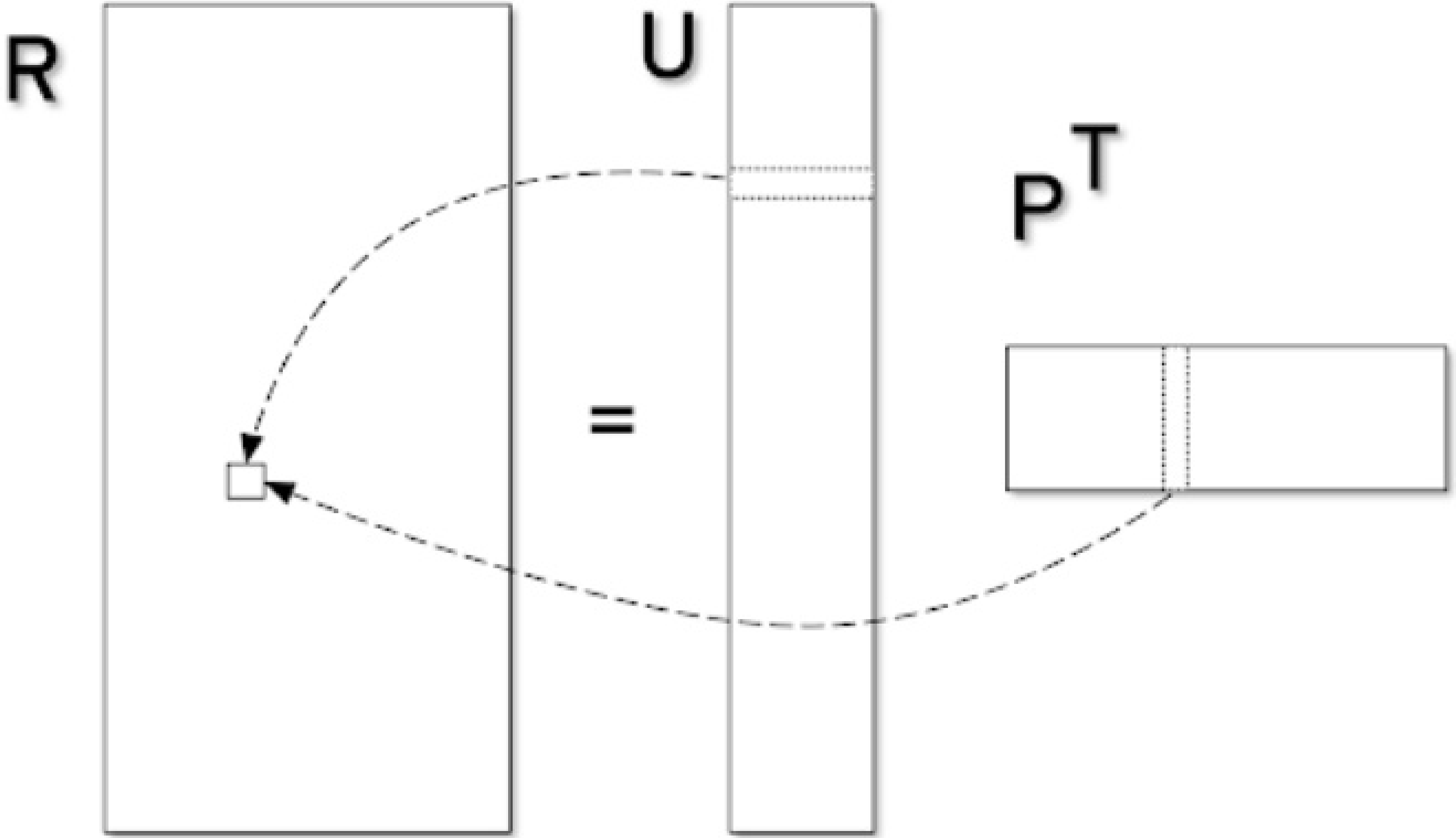
```
user1_listened.take(10)
user1_listened.map(lambda x:(x[1][1][1:])).take(10)
```

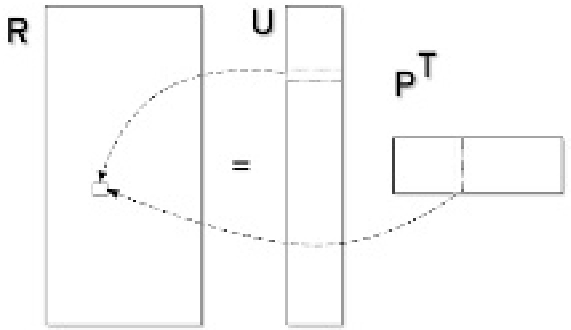
```
[['beirut', '609'],
 ['dredg', '605'],
 ['calexico', '562'],
 ['led', 'zeppelin', '456'],
 ['laura', 'marling', '401'],
 ['minus', 'the', 'bear', '377'],
 ['zion', 'i', '352'],
 ['bon', 'iver', '313'],
 ['xavier', 'rudd', '306'],
 ['passion', 'pit', '273']]
```

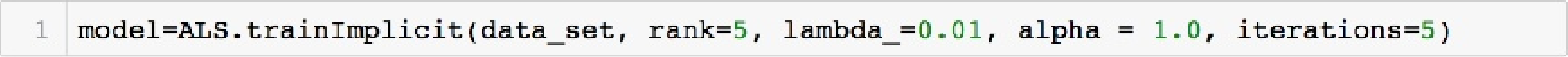


```
user1_pred=model.predictAll(user1_unlistened)
```

```
[['john', 'frusciante', '2140'],
 ['red', 'hot', 'chili', 'peppers', '1614'],
 ['waglewski', 'fisz', 'emade', '566'],
 ['coldplay', '461'],
 ['the', 'mars', 'volta', '402'],
 ['pj', 'harvey', '398'],
 ['muchy', '397'],
 ['maria', 'peszek', '373'],
 ['fisz', '305'],
 ['ataxia', '301']]
```

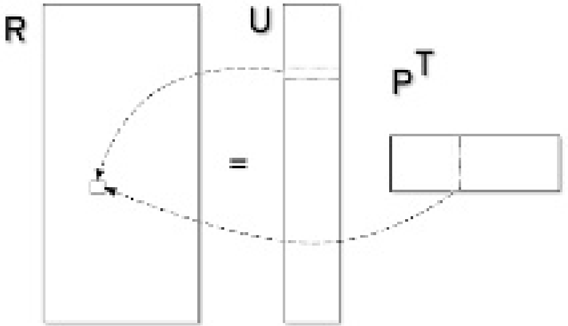

R
=
U
P^T

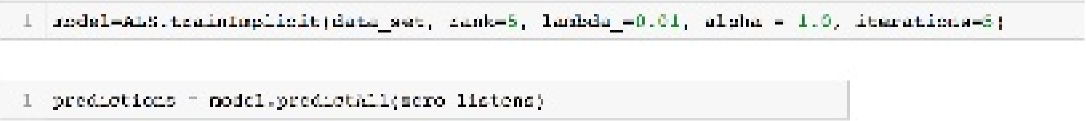




```
model=ALS.trainImplicit(data_set, rank=5, lambda_=0.01, alpha = 1.0, iterations=5)
```

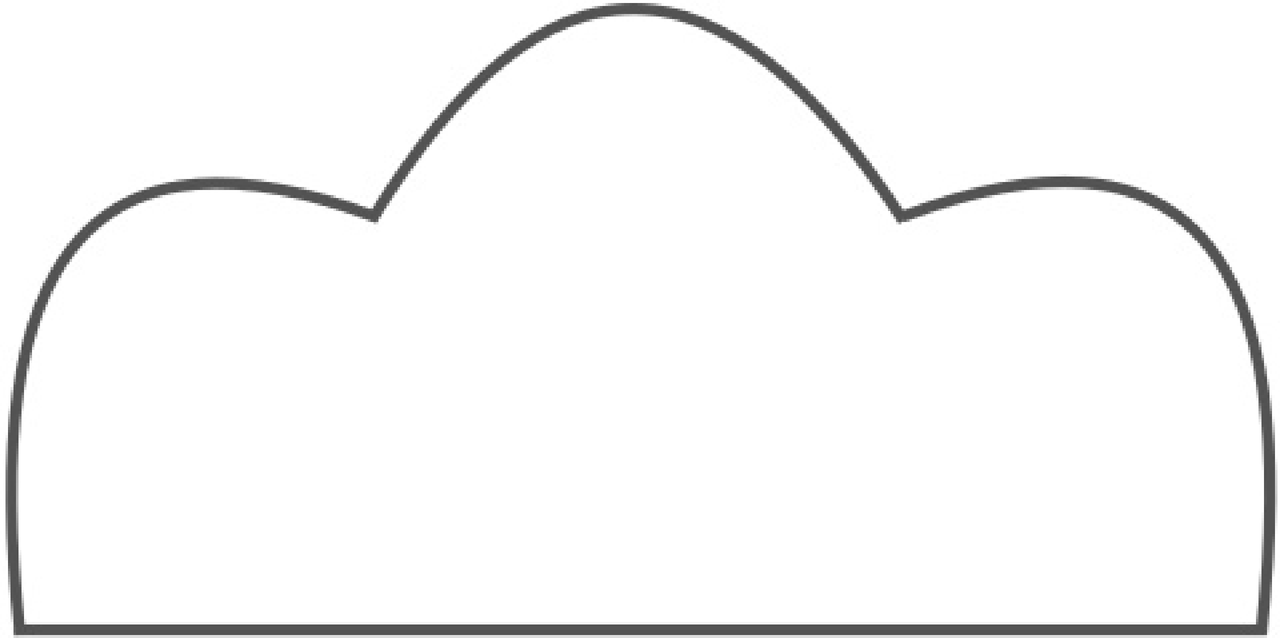
```
predictions = model.predictAll(zero_listens)
```



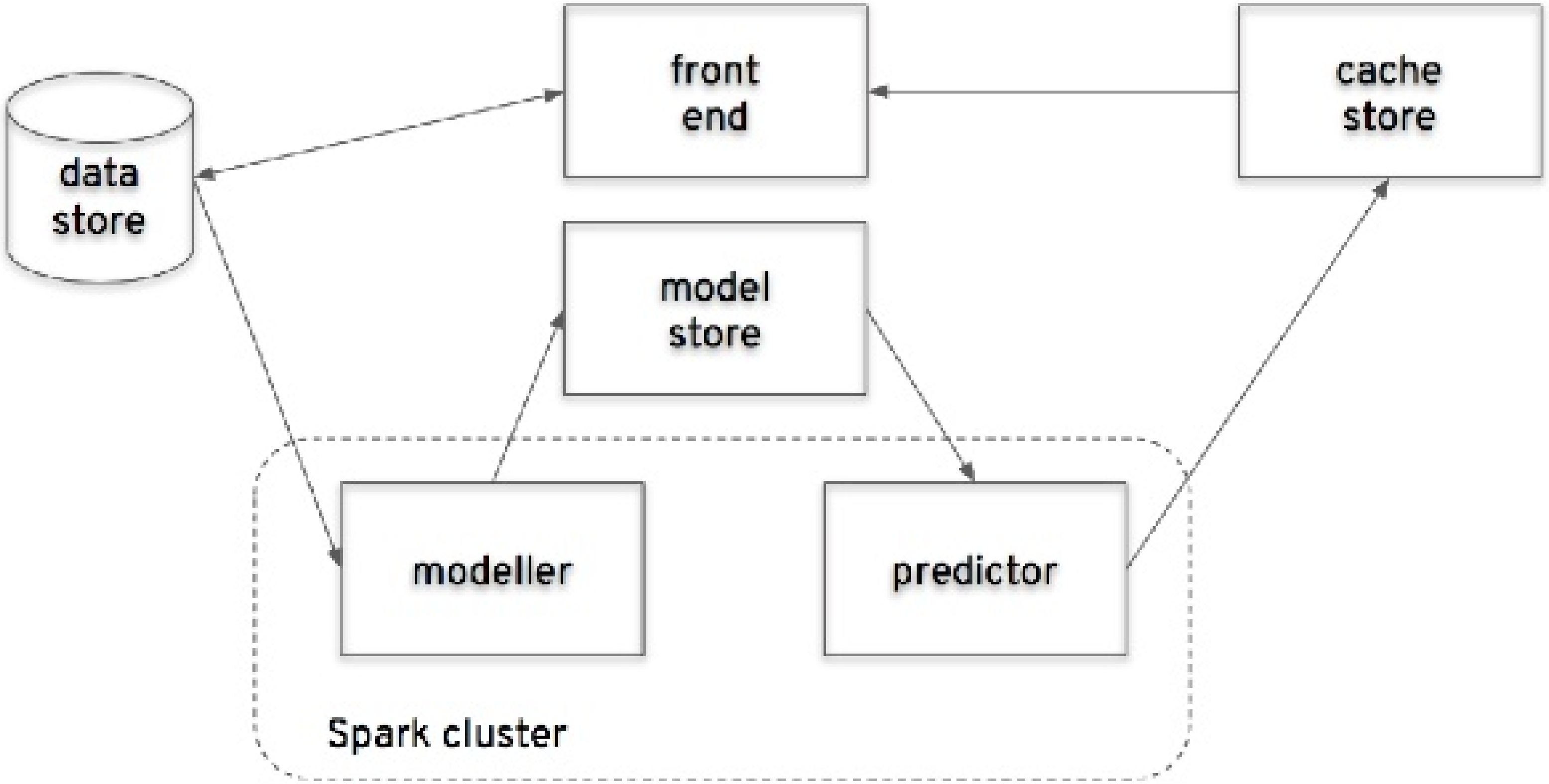


```
model=ALS.trainImplicit(data_set, rank=5, lambda_=0.01, alpha = 1.0, iterations=5)
```

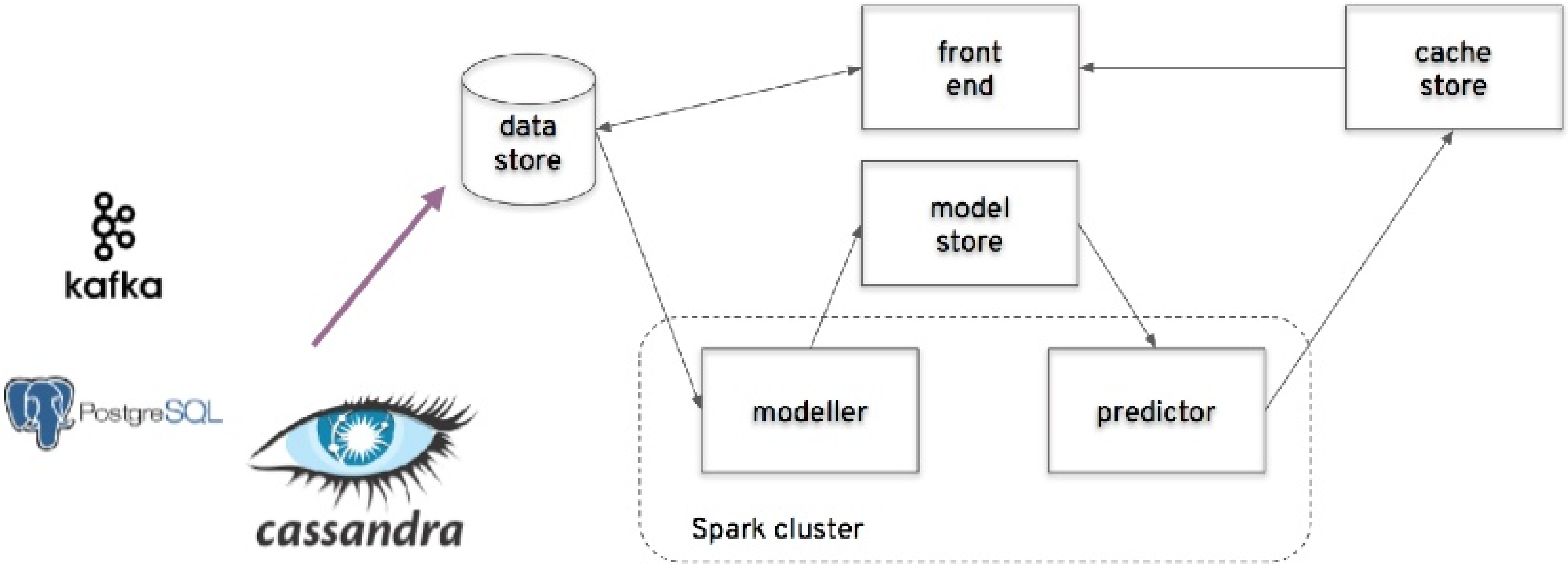
```
predictions = model.predictAll(zero_listens)
```



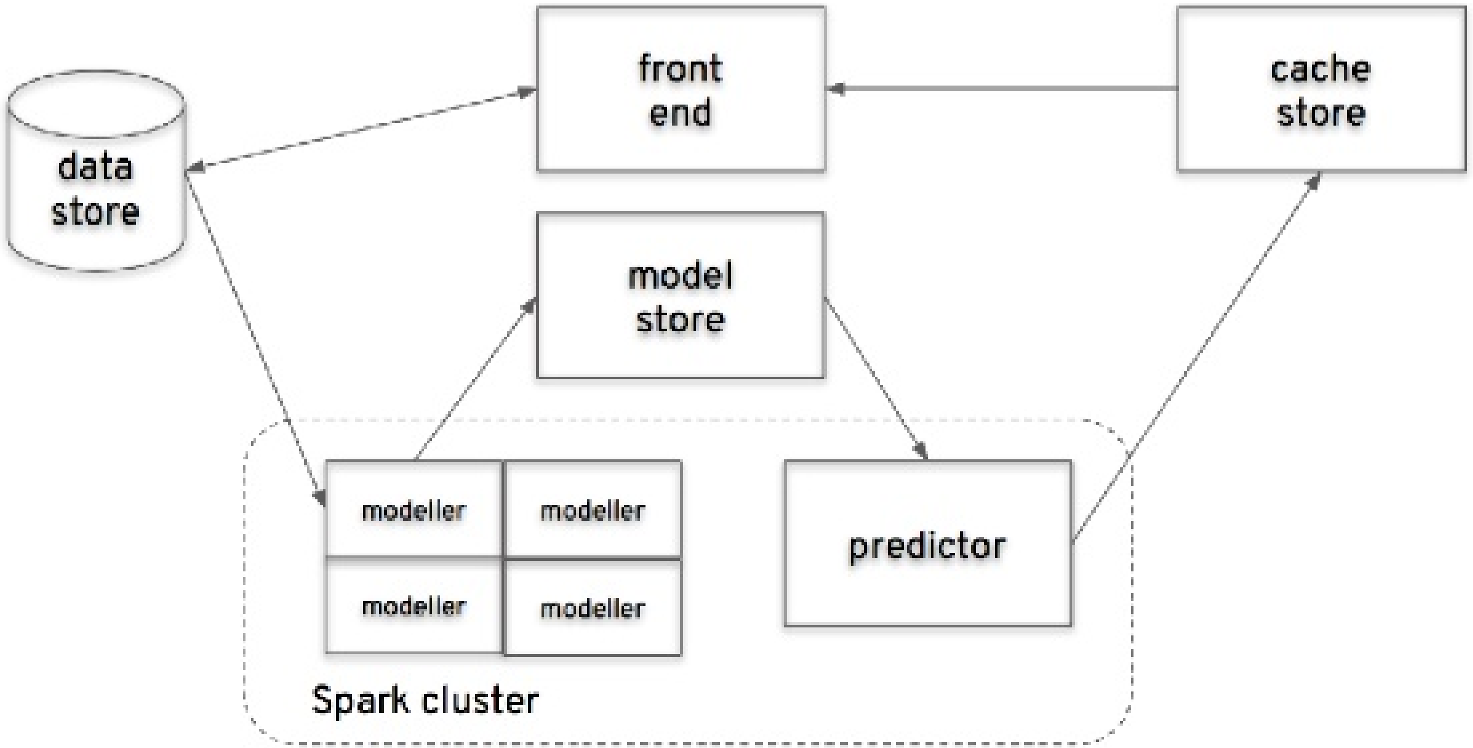
Microservices



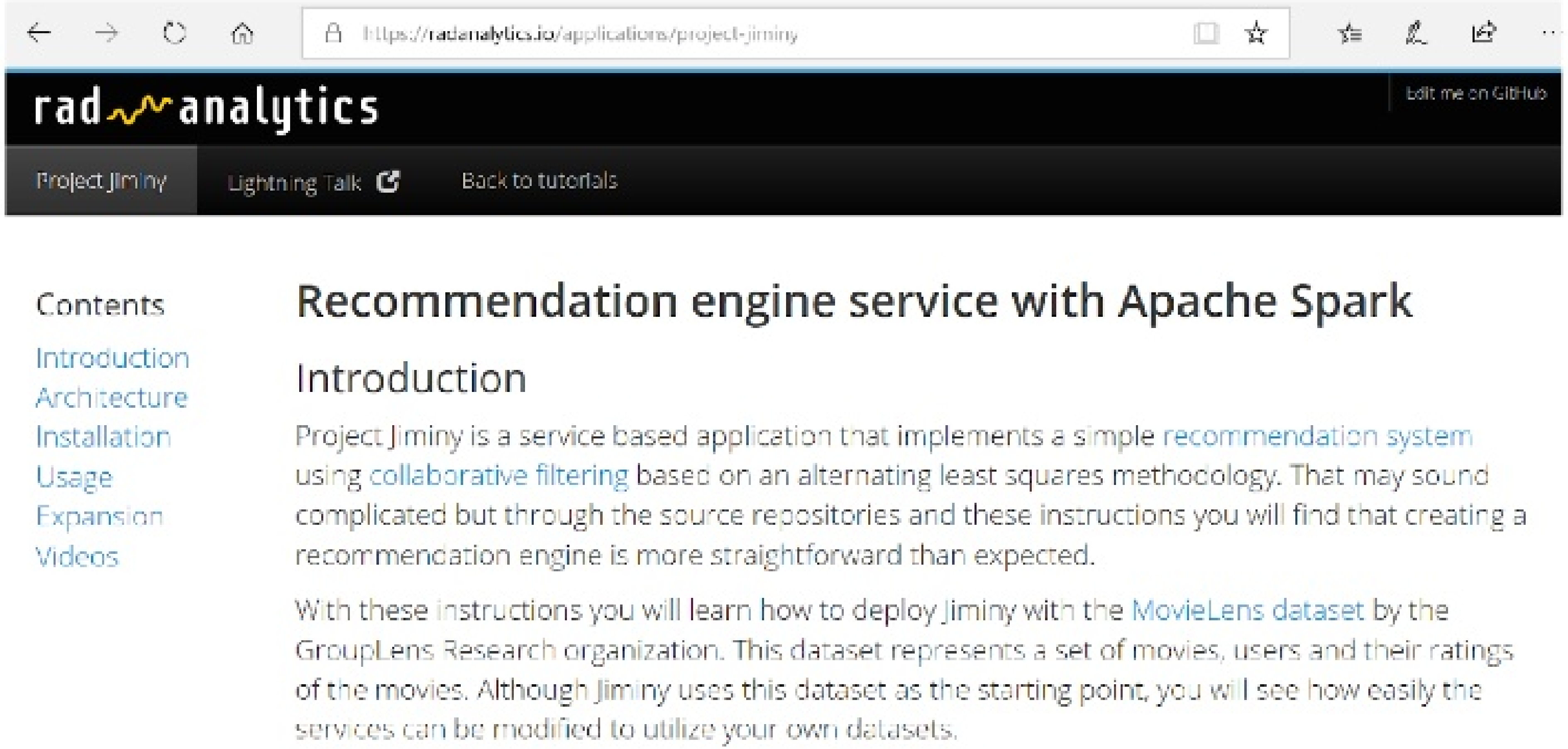
Microservices



Microservices



radanalytics.io



Recommendation engine service with Apache Spark

Introduction

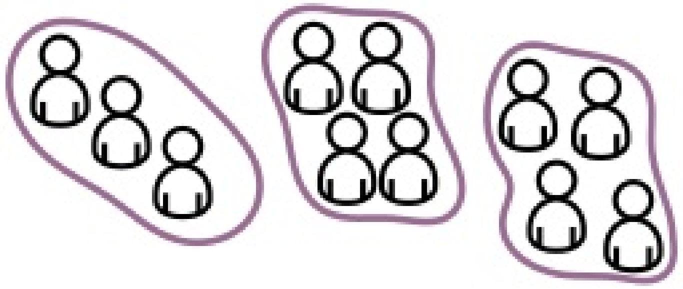
Project Jiminy is a service based application that implements a simple recommendation system using collaborative filtering based on an alternating least squares methocology. That may sound complicated but through the source repositories and these instructions you will find that creating a recommendation engine is more straightforward than expected.

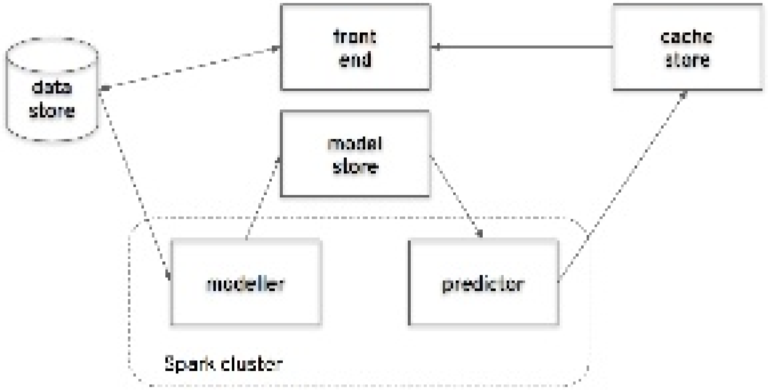
With these instructions you will learn how to deploy Jiminy with the MovieLens dataset by the GroupLens Research organization. This dataset represents a set of movies, users and their ratings of the movies. Although Jiminy uses this dataset as the starting point, you will see how easily the services can be modified to utilize your own datasets.

@sophwats

sophie@redhat.com

Collaborative Filtering




front end

cache store

data store

model store

modeller

predictor

Spark cluster

Alternating Least Squares