Building Recommendation Systems in Spark ML

Janani Ravi
CO-FOUNDER, LOONYCORN
www.loonycorn.com

Overview

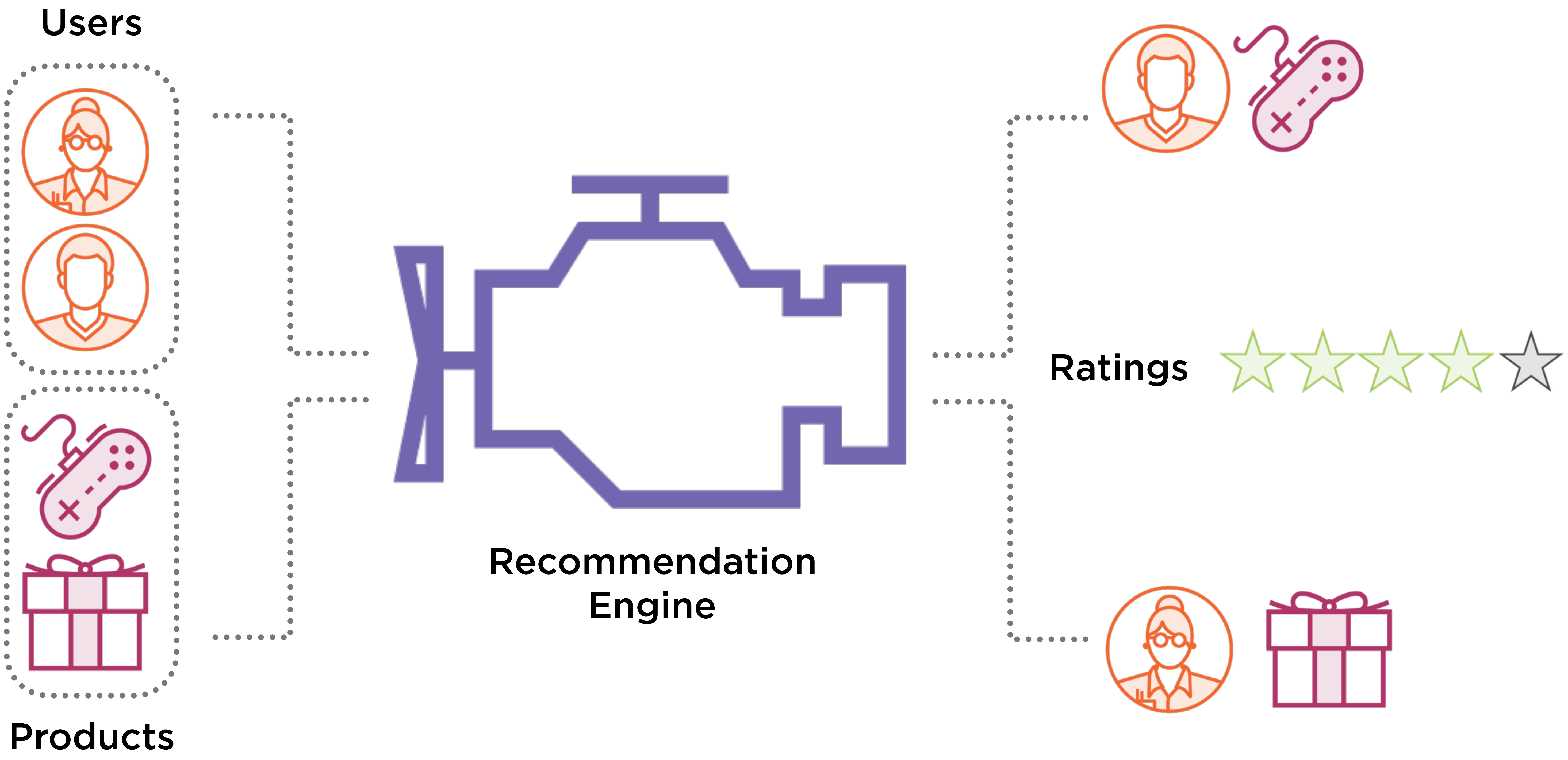
Collaborative filtering algorithms are used to make product recommendations to users

Spark offers estimators which use the ALS method to implement collaborative filtering

Abstracts you from the math involved in ALS

Recommendations can be based on explicit and implicit ratings

Recommendation Systems



Users

Products

Recommendation Engine

Ratings

Approaches to Recommendations

Content-based

Estimate rating using this user and this product alone

Collaborative

Employ information about other users, products too

Hybrid

Combine both content-based and collaborative filtering

Approaches to Recommendations

Content-based

Estimate rating using this user and this product alone

Collaborative

Employ information about other users, products too

Hybrid

Combine both content-based and collaborative filtering

Content-based Filtering

Individual Users

Products

Personalized Recommendations

Views

Personalized Recommendations

Purchases

Content-based Filtering

Match product description to user profile

Two significant drawbacks

- Requires accurate, rich product metadata
- Hard to extend across product types

Approaches to Recommendations

Content-based

Estimate rating using this user and this product alone

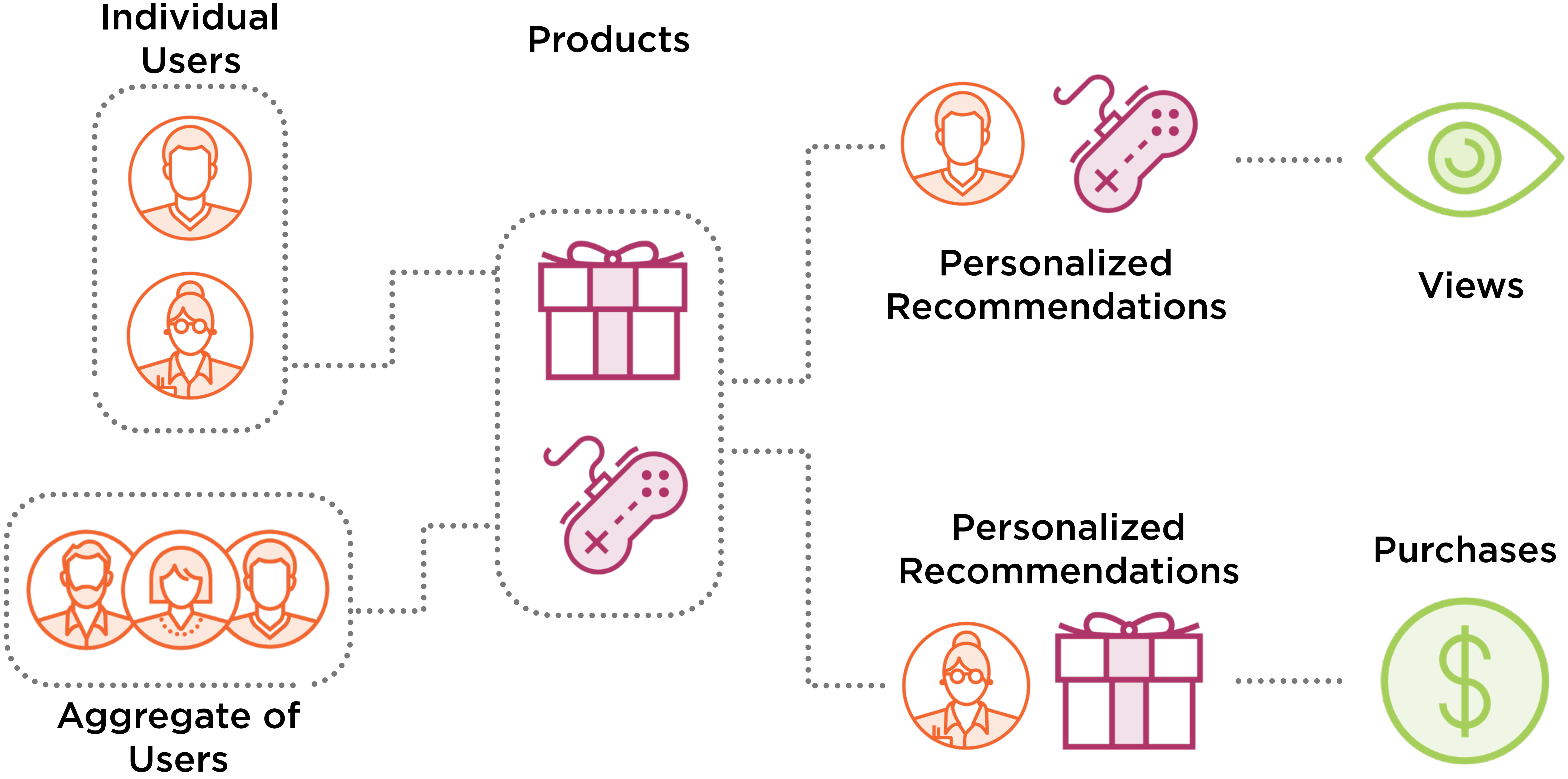
Collaborative

Employ information about other users, products too

Hybrid

Combine both content-based and collaborative filtering

Collaborative Filtering



Individual Users

Aggregate of Users

Products

Personalized Recommendations

Views

Personalized Recommendations

Purchases

Collaborative Filtering

Individual Users

Products

Personalized Recommendations

Views

Personalized Recommendations

Purchases

Aggregate of Users

Collaborative Filtering

Collaborative Filtering

Users who agreed in the past will agree in the future, and that they will like similar kinds of items as they liked in the past

Collaborative Filtering

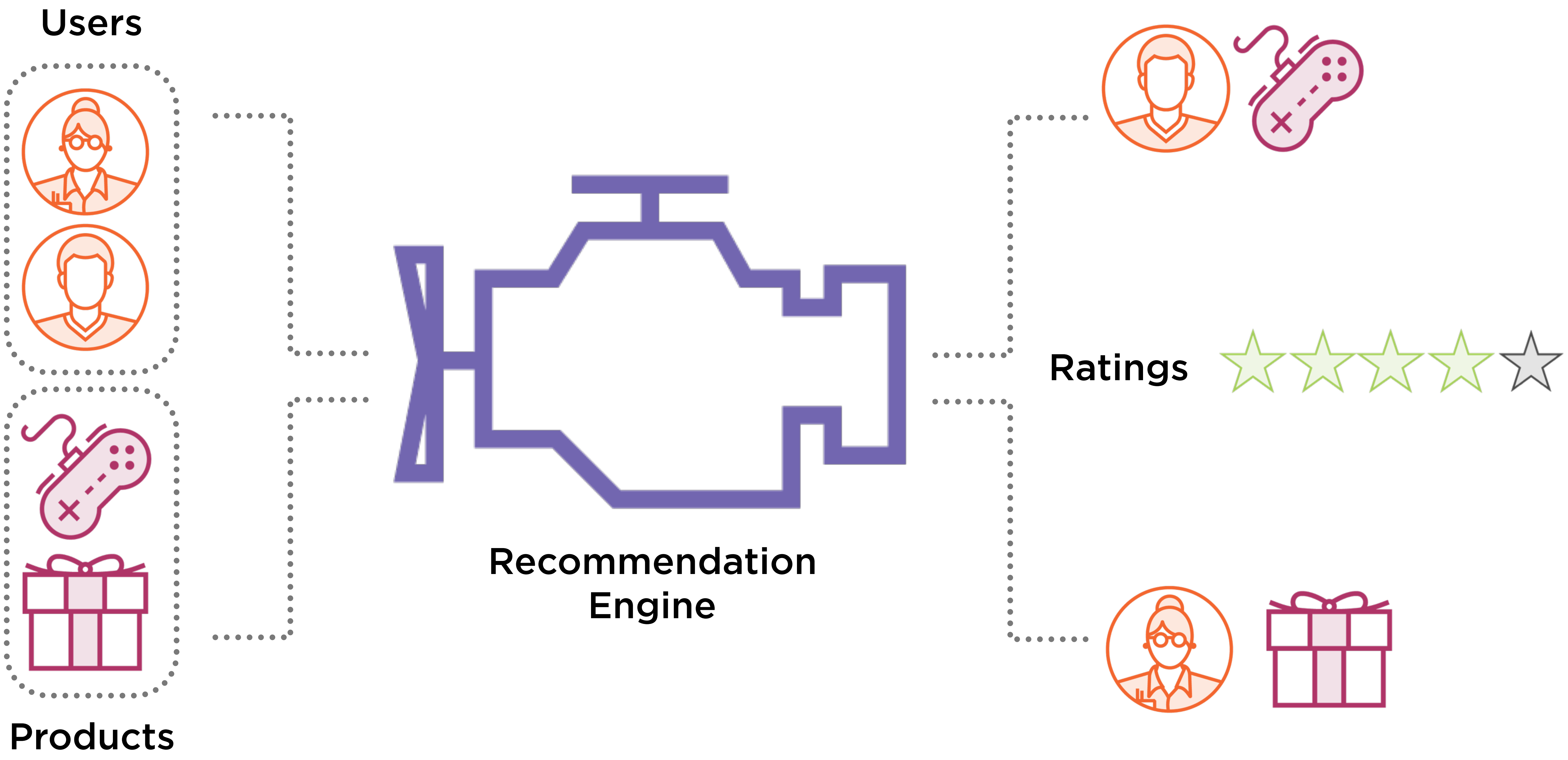
Users who agreed in the past will agree in the future, and that they will like similar kinds of items as they liked in the past

Collaborative Filtering

Users who agreed in the past will agree in the future, and that they will like similar kinds of items as they liked in the past

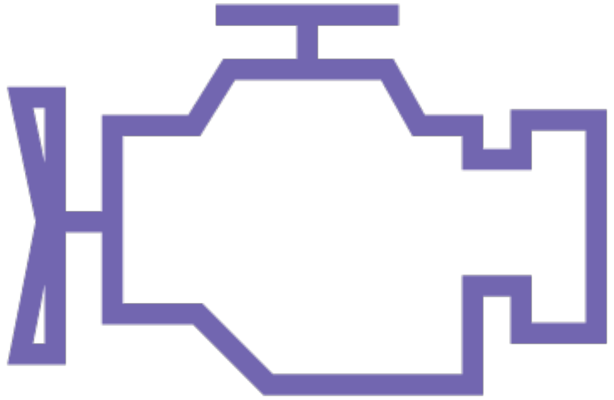
“People who buy X also buy Y”

Recommendation Systems



Estimate how a user would rate every product

Recommend the products to the user which have the highest estimated ratings



Ratings

Desired output of Recommendation Engine

- Ratings Matrix: score for each combination of user and product
- Number of rows = Number of users (n_u)
- Number of columns = Number of products (n_p)

Ratings Matrix

$$\begin{bmatrix} r_{11} & r_{12} & r_{13} & & r_{1n_p} \\ r_{21} & r_{22} & r_{23} & & r_{2n_p} \\ r_{31} & r_{32} & r_{33} & \cdots & r_{3n_p} \\ \cdots & \cdots & \cdots & r_{ij} & \cdots \\ r_{n_u1} & r_{n_u2} & r_{n_u3} & & r_{n_un_p} \end{bmatrix}$$

n_u rows

n_p columns

Each element predicts how much a particular user will like a particular product

Ratings Matrix

$$\begin{array}{c|ccccc} & \text{Product 1} & \text{Product 2} & \text{Product 3} & & \text{Product } n_p \\ \text{User 1} & r_{11} & r_{12} & r_{13} & & r_{1n_p} \\ & r_{21} & r_{22} & r_{23} & & r_{2n_p} \\ & r_{31} & r_{32} & r_{33} & \cdots & r_{3n_p} \\ & \cdots & \cdots & \cdots & r_{ij} & \cdots \\ & r_{n_u1} & r_{n_u2} & r_{n_u3} & & r_{n_un_p} \end{array}$$

Each row represents the preferences of 1 user for different products

Ratings Matrix

$$\begin{array}{c|ccccc} & \text{Product 1} & \text{Product 2} & \text{Product 3} & & \text{Product } n_p \\ & r_{11} & r_{12} & r_{13} & & r_{1n_p} \\ \text{User 2} & r_{21} & r_{22} & r_{23} & & r_{2n_p} \\ & r_{31} & r_{32} & r_{33} & \cdots & r_{3n_p} \\ & \cdots & \cdots & \cdots & r_{ij} & \cdots \\ & r_{n_u1} & r_{n_u2} & r_{n_u3} & & r_{n_un_p} \end{array}$$

Each row represents the preferences of 1 user for different products

Ratings Matrix

$$\begin{array}{cccccc} & \text{Product 1} & \text{Product 2} & \text{Product 3} & & \text{Product } n_p \\ & r_{11} & r_{12} & r_{13} & & r_{1n_p} \\ & r_{21} & r_{22} & r_{23} & & r_{2n_p} \\ & r_{31} & r_{32} & r_{33} & \cdots & r_{3n_p} \\ & \cdots & \cdots & \cdots & r_{ij} & \cdots \\ \text{User } n_u & \boxed{r_{n_u1}} & \boxed{r_{n_u2}} & \boxed{r_{n_u3}} & & \boxed{r_{n_un_p}} \end{array}$$

Each row represents the preferences of 1 user for different products

Ratings Matrix

$$\begin{bmatrix} r_{11} & r_{12} & r_{13} & & r_{1n_p} \\ r_{21} & r_{22} & r_{23} & & r_{2n_p} \\ r_{31} & r_{32} & r_{33} & \dots & r_{3n_p} \\ \dots & \dots & \dots & r_{ij} & \dots \\ r_{n_u1} & r_{n_u2} & r_{n_u3} & & r_{n_un_p} \end{bmatrix}$$

n_u rows

n_p columns

Each column represents the preference for a single product across all users

Ratings Matrix

$$\begin{array}{l|ccccc|}
 & \text{Product 1} & & & & \\
\text{User 1} & r_{11} & r_{12} & r_{13} & & r_{1n_p} \\
\text{User 2} & r_{21} & r_{22} & r_{23} & & r_{2n_p} \\
 & r_{31} & r_{32} & r_{33} & \cdots & r_{3n_p} \\
 & \cdots & \cdots & \cdots & r_{ij} & \cdots \\
\text{User } n_u & r_{n_u1} & r_{n_u2} & r_{n_u3} & & r_{n_un_p}
\end{array}$$

Each column represents the preference for a single product across all users

Ratings Matrix

$$\begin{array}{c|ccccc} & & \text{Product 2} & & & \\ \hline \text{User 1} & r_{11} & r_{12} & r_{13} & & r_{1n_p} \\ \text{User 2} & r_{21} & r_{22} & r_{23} & & r_{2n_p} \\ & r_{31} & r_{32} & r_{33} & \cdots & r_{3n_p} \\ & \cdots & \cdots & \cdots & r_{ij} & \cdots \\ \text{User } n_u & r_{n_u1} & r_{n_u2} & r_{n_u3} & & r_{n_un_p} \end{array}$$

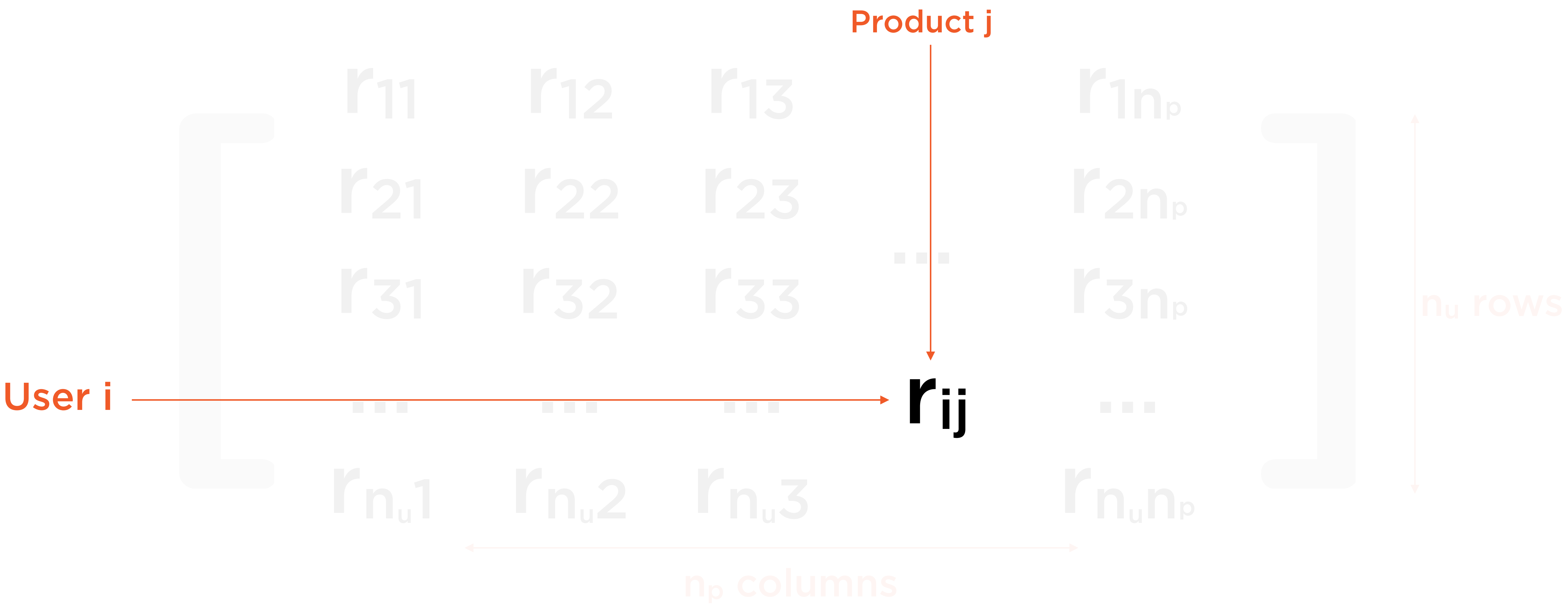
Each column represents the preference for a single product across all users

Ratings Matrix

$$\begin{array}{lccccc} & & & & & \text{Product } n_p \\ \text{User 1} & r_{11} & r_{12} & r_{13} & & r_{1n_p} \\ \text{User 2} & r_{21} & r_{22} & r_{23} & & r_{2n_p} \\ & r_{31} & r_{32} & r_{33} & \cdots & r_{3n_p} \\ & \cdots & \cdots & \cdots & r_{ij} & \cdots \\ \text{User } n_u & r_{n_u1} & r_{n_u2} & r_{n_u3} & & r_{n_un_p} \end{array}$$

Each column represents the preference for a single product across all users

Ratings Matrix



Product j

$$\begin{bmatrix} r_{11} & r_{12} & r_{13} & & r_{1n_p} \\ r_{21} & r_{22} & r_{23} & & r_{2n_p} \\ r_{31} & r_{32} & r_{33} & \cdots & r_{3n_p} \\ \vdots & \vdots & \vdots & r_{ij} & \cdots \\ r_{n_u1} & r_{n_u2} & r_{n_u3} & & r_{n_un_p} \end{bmatrix}$$

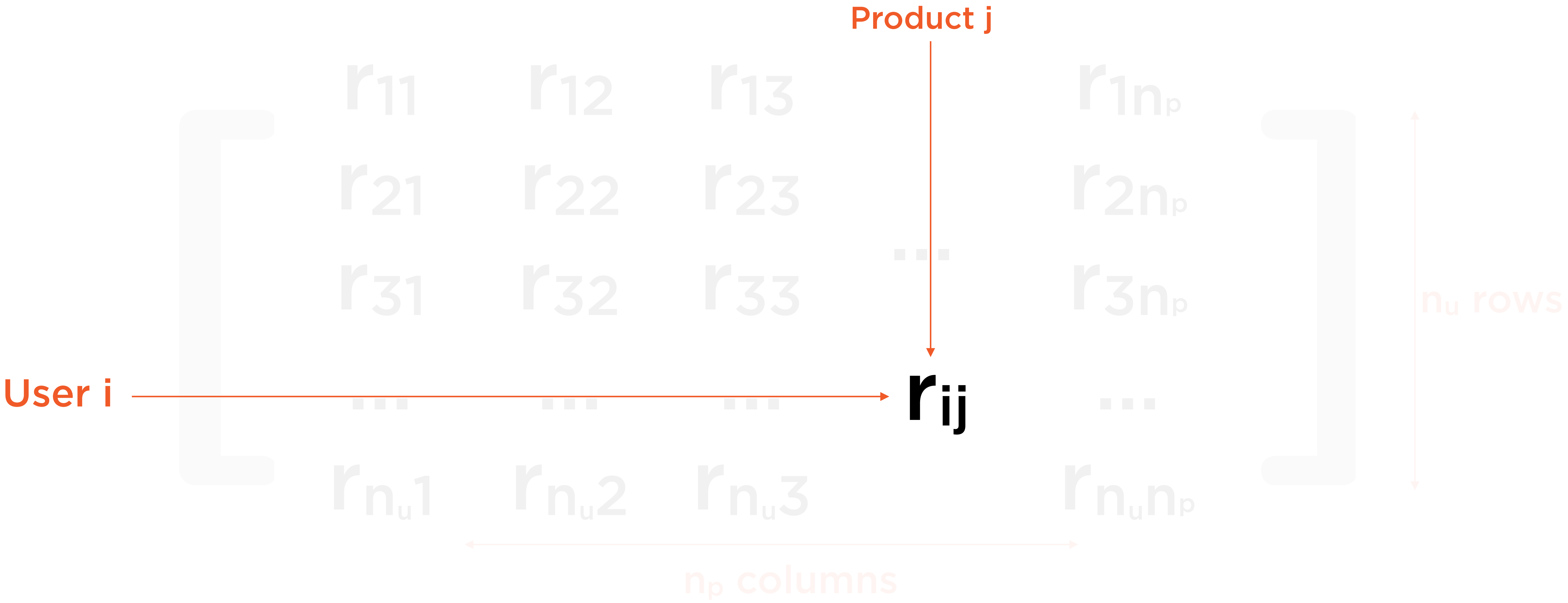
User i

n_u rows

n_p columns

Consider the rating of user i for product j

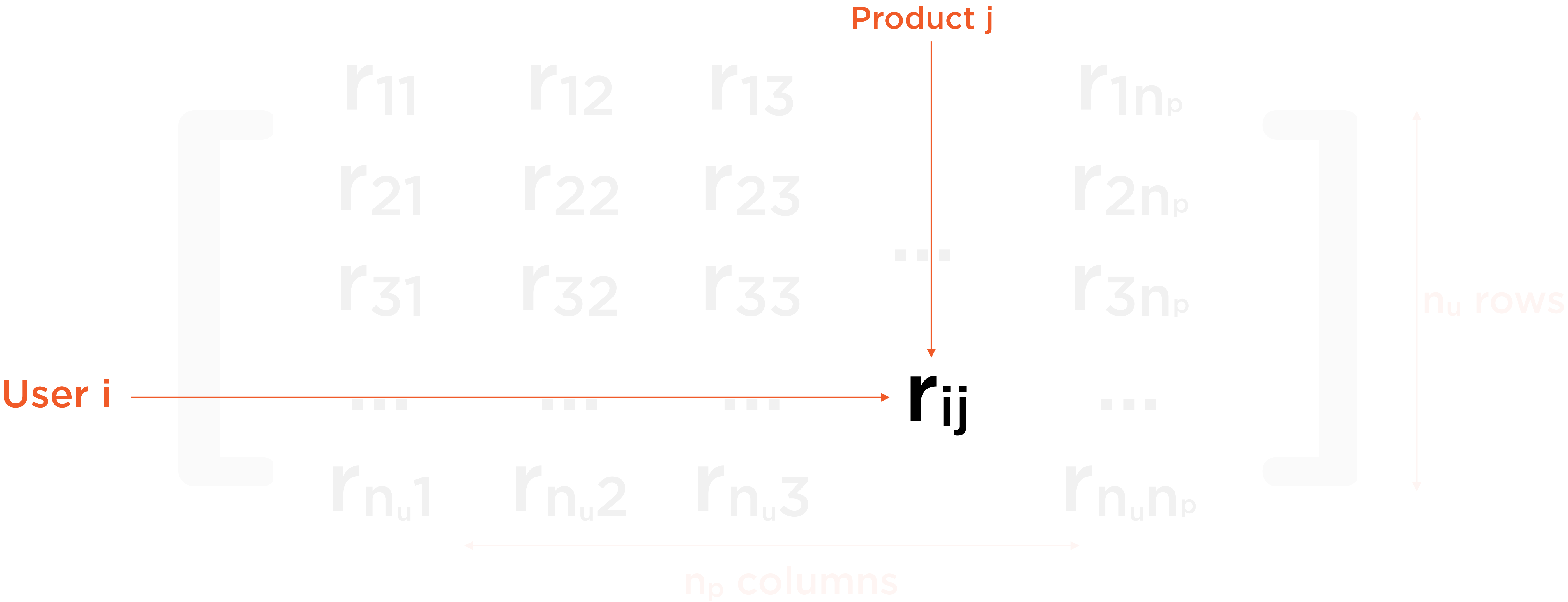
Ratings Matrix



Very rarely, this user might actually have rated this product (e.g. by adding a rating + review)

★★★★☆

Ratings Matrix



Product j

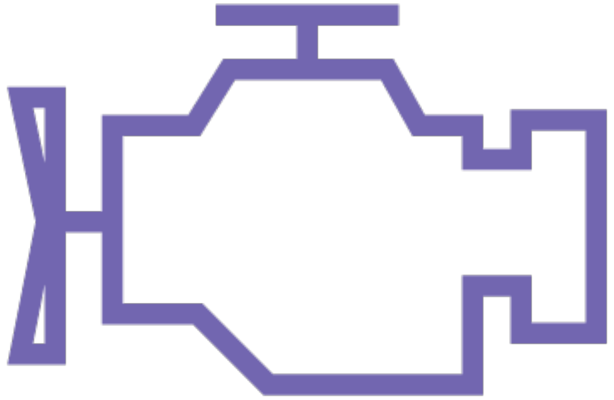
$$\left[\begin{array}{ccccc} r_{11} & r_{12} & r_{13} & & r_{1n_p} \\ r_{21} & r_{22} & r_{23} & & r_{2n_p} \\ r_{31} & r_{32} & r_{33} & \cdots & r_{3n_p} \\ \cdots & \cdots & \cdots & r_{ij} & \cdots \\ r_{n_u1} & r_{n_u2} & r_{n_u3} & & r_{n_un_p} \end{array}\right]$$

User i

n_u rows

n_p columns

But usually, this value is initially missing and must be **estimated**



Estimating Ratings Matrix

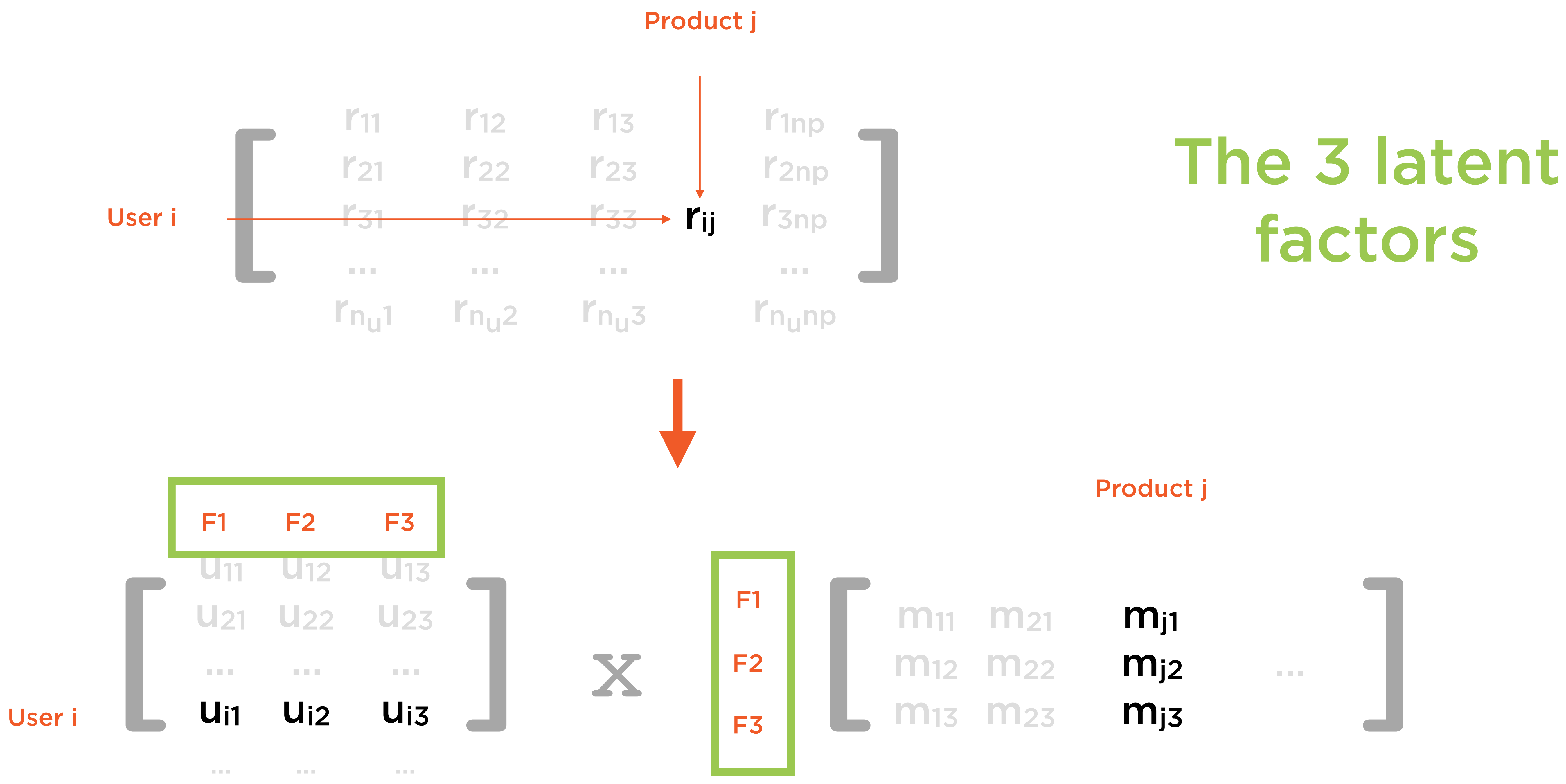
What if we could identify hidden factors that define this value?

This is a common technique called latent factor analysis

Pick a number of latent factors, say 3

$$n_f = 3$$

Ratings Matrix

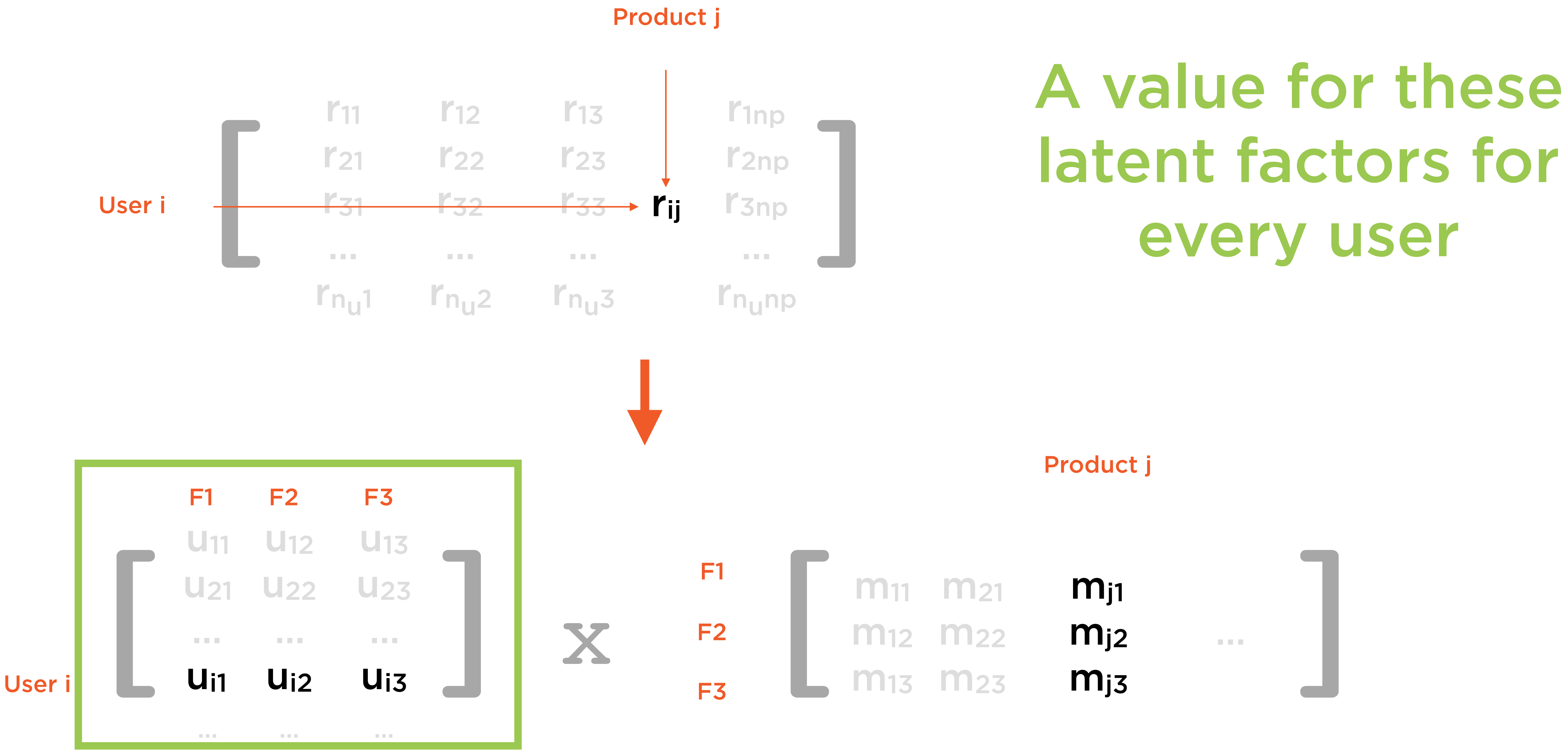


$$\begin{array}{c|ccccc} & & & & \text{Product } j & \\ & r_{11} & r_{12} & r_{13} & & r_{1np} \\ & r_{21} & r_{22} & r_{23} & & r_{2np} \\ \text{User } i & r_{31} & r_{32} & r_{33} & r_{ij} & r_{3np} \\ & \cdots & \cdots & \cdots & & \cdots \\ & r_{n_u1} & r_{n_u2} & r_{n_u3} & & r_{n_unp} \end{array}$$

The 3 latent factors

$$\begin{array}{c|ccc} & F1 & F2 & F3 \\ & u_{11} & u_{12} & u_{13} \\ & u_{21} & u_{22} & u_{23} \\ & \cdots & \cdots & \cdots \\ \text{User } i & u_{i1} & u_{i2} & u_{i3} \\ & \cdots & \cdots & \cdots \end{array} \times \begin{array}{c|cccc} & & & \text{Product } j & \\ F1 & m_{11} & m_{21} & m_{j1} & \\ F2 & m_{12} & m_{22} & m_{j2} & \cdots \\ F3 & m_{13} & m_{23} & m_{j3} & \end{array}$$

Ratings Matrix

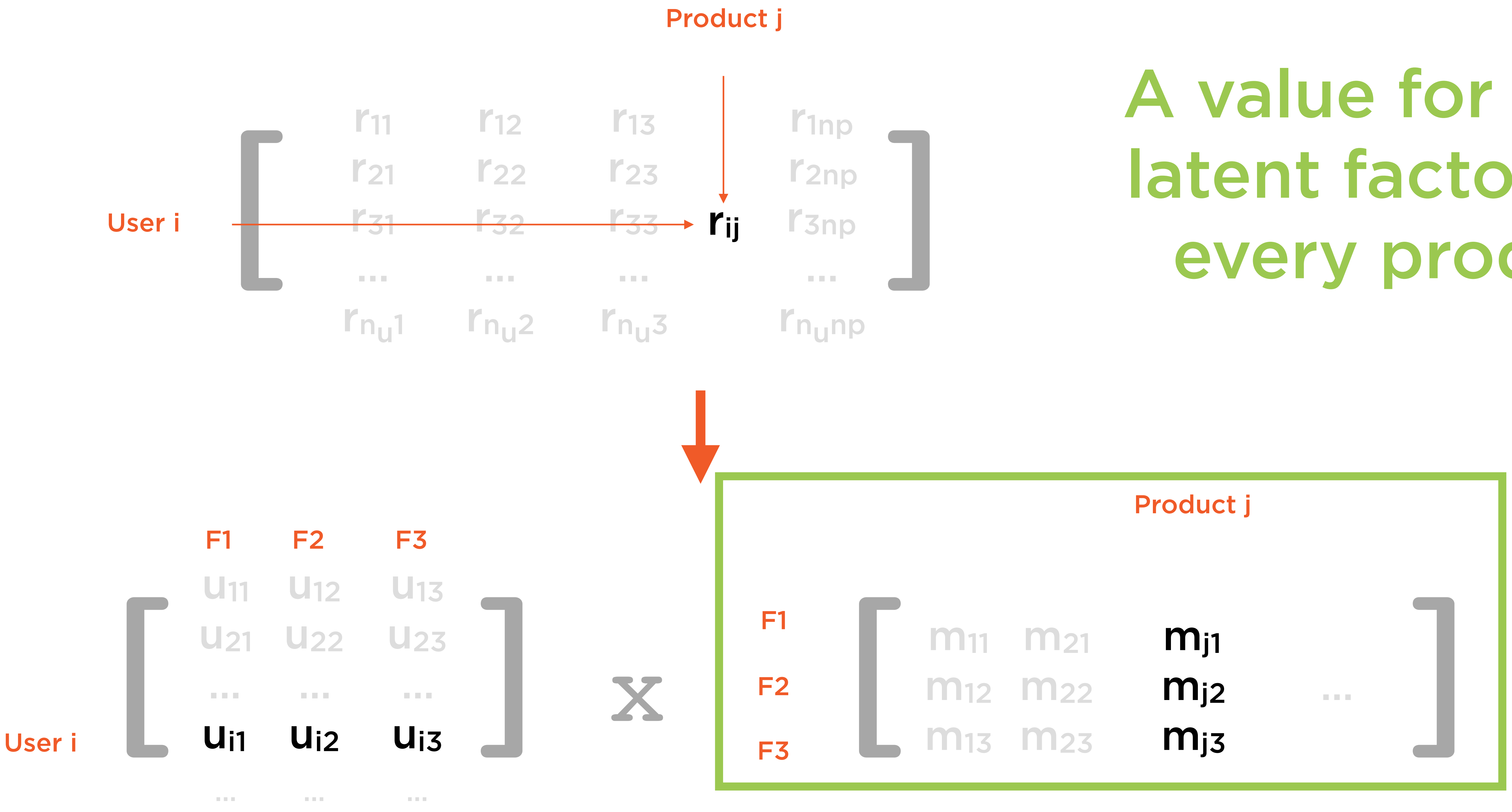


$$\begin{array}{cccccc} & & & & \text{Product } j & \\ & r_{11} & r_{12} & r_{13} & & r_{1np} \\ & r_{21} & r_{22} & r_{23} & & r_{2np} \\ \text{User } i & r_{31} & r_{32} & r_{33} & r_{ij} & r_{3np} \\ & \ldots & \ldots & \ldots & & \ldots \\ & r_{n_u1} & r_{n_u2} & r_{n_u3} & & r_{n_unp} \end{array}$$

$$\begin{array}{cccc} & F1 & F2 & F3 \\ & u_{11} & u_{12} & u_{13} \\ & u_{21} & u_{22} & u_{23} \\ & \ldots & \ldots & \ldots \\ \text{User } i & u_{i1} & u_{i2} & u_{i3} \\ & \ldots & \ldots & \ldots \end{array} \quad \times \quad \begin{array}{ccccc} & & & \text{Product } j & \\ F1 & m_{11} & m_{21} & m_{j1} & \\ F2 & m_{12} & m_{22} & m_{j2} & \ldots \\ F3 & m_{13} & m_{23} & m_{j3} & \end{array}$$

A value for these latent factors for every user

Ratings Matrix

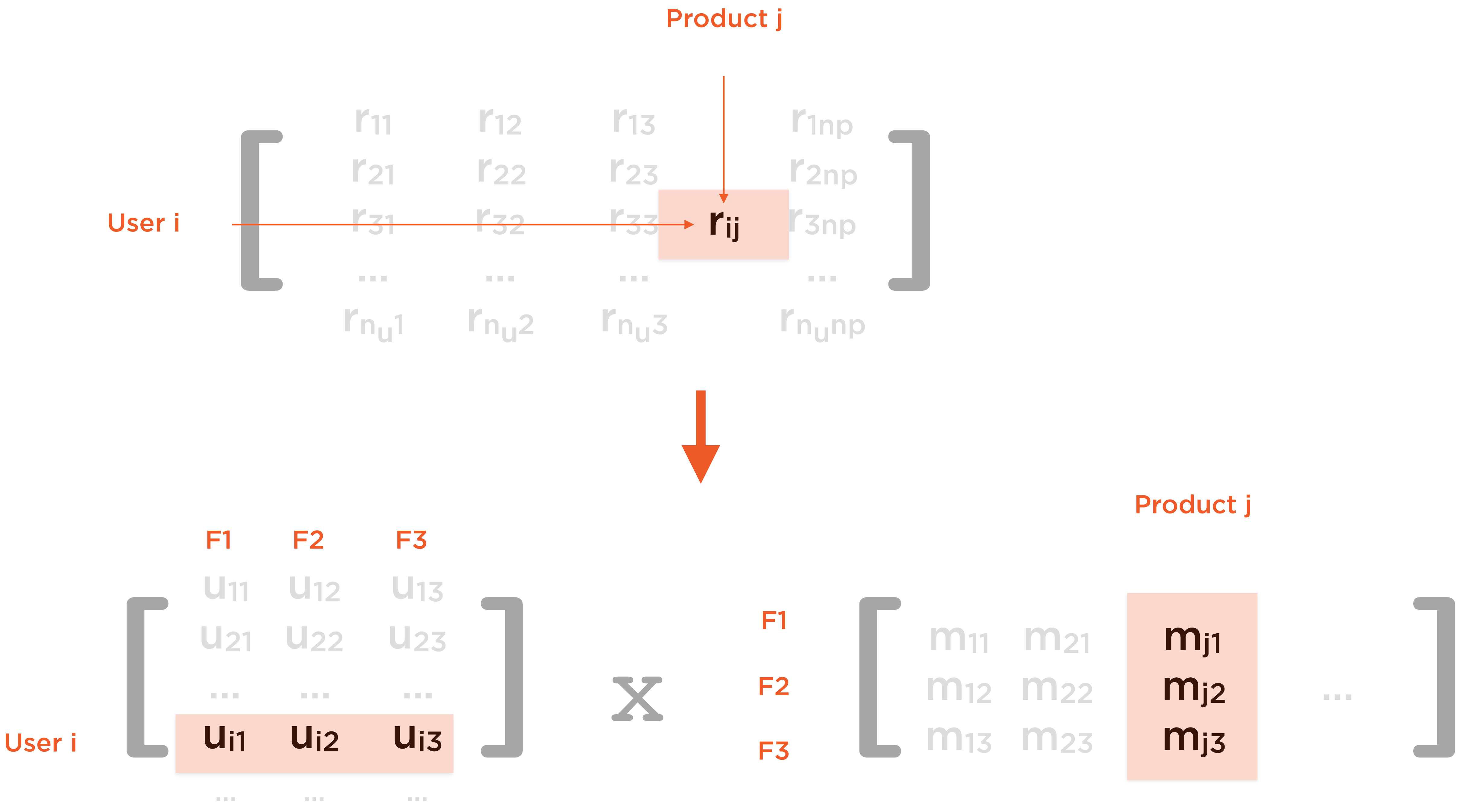


$$\begin{array}{c|ccccc}
 & & & & \text{Product } j & \\
 & r_{11} & r_{12} & r_{13} & & r_{1np} \\
 & r_{21} & r_{22} & r_{23} & & r_{2np} \\
\text{User } i & r_{31} & r_{32} & r_{33} & r_{ij} & r_{3np} \\
 & \cdots & \cdots & \cdots & & \cdots \\
 & r_{n_u1} & r_{n_u2} & r_{n_u3} & & r_{n_unp}
\end{array}$$

$$\begin{array}{c|ccc}
 & F1 & F2 & F3 \\
 & u_{11} & u_{12} & u_{13} \\
 & u_{21} & u_{22} & u_{23} \\
 & \cdots & \cdots & \cdots \\
\text{User } i & u_{i1} & u_{i2} & u_{i3} \\
 & \cdots & \cdots & \cdots
\end{array}
\times
\begin{array}{c|cccc}
 & & & \text{Product } j & \\
F1 & m_{11} & m_{21} & m_{j1} & \\
F2 & m_{12} & m_{22} & m_{j2} & \cdots \\
F3 & m_{13} & m_{23} & m_{j3} &
\end{array}$$

A value for these latent factors for every product

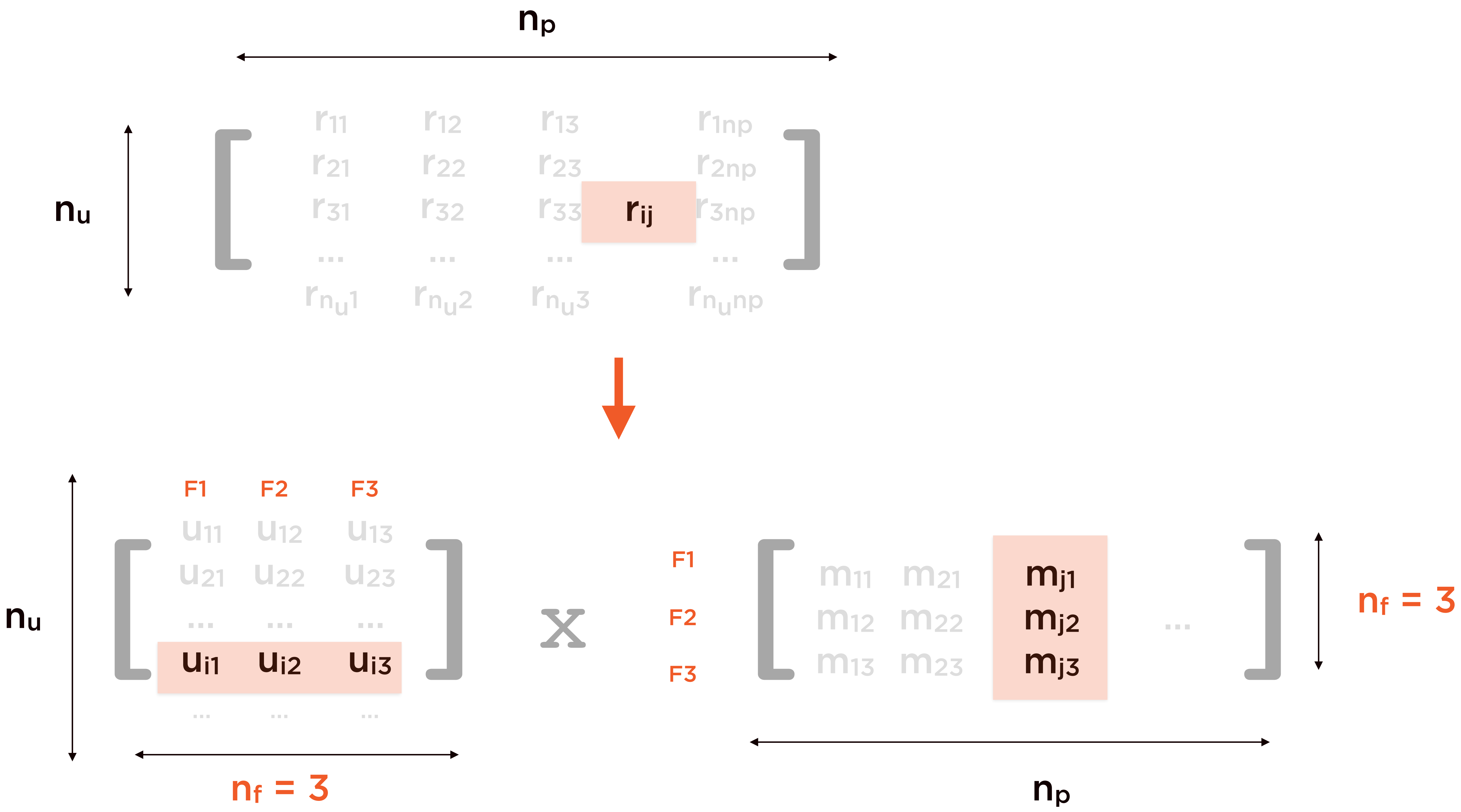
Ratings Matrix



$$\begin{array}{c|ccccc|} & & & & \text{Product j} & \\ & r_{11} & r_{12} & r_{13} & & r_{1np} \\ & r_{21} & r_{22} & r_{23} & & r_{2np} \\ \text{User i} & r_{31} & r_{32} & r_{33} & r_{ij} & r_{3np} \\ & \dots & \dots & \dots & & \dots \\ & r_{n_u1} & r_{n_u2} & r_{n_u3} & & r_{n_unp} \end{array}$$

$$\begin{array}{c|ccc|} & F1 & F2 & F3 \\ & u_{11} & u_{12} & u_{13} \\ & u_{21} & u_{22} & u_{23} \\ & \dots & \dots & \dots \\ \text{User i} & u_{i1} & u_{i2} & u_{i3} \\ & \dots & \dots & \dots \end{array} \quad x \quad \begin{array}{c|cccc|} & & & \text{Product j} & \\ F1 & m_{11} & m_{21} & m_{j1} & \\ F2 & m_{12} & m_{22} & m_{j2} & \dots \\ F3 & m_{13} & m_{23} & m_{j3} & \end{array}$$

Ratings Matrix



n_p

$$n_u \quad \left[\begin{array}{ccccc} r_{11} & r_{12} & r_{13} & & r_{1np} \\ r_{21} & r_{22} & r_{23} & & r_{2np} \\ r_{31} & r_{32} & r_{33} & r_{ij} & r_{3np} \\ \cdots & \cdots & \cdots & & \cdots \\ r_{n_u1} & r_{n_u2} & r_{n_u3} & & r_{n_unp} \end{array}\right]$$

$$n_u \quad \begin{array}{ccc} F1 & F2 & F3 \\ u_{11} & u_{12} & u_{13} \\ u_{21} & u_{22} & u_{23} \\ \cdots & \cdots & \cdots \\ u_{i1} & u_{i2} & u_{i3} \\ \cdots & \cdots & \cdots \end{array} \quad \times \quad \begin{array}{c|cccc} F1 & m_{11} & m_{21} & m_{j1} & \\ F2 & m_{12} & m_{22} & m_{j2} & \cdots \\ F3 & m_{13} & m_{23} & m_{j3} & \end{array} \quad n_f = 3$$

$n_f = 3$

n_p

Matrix Factorization

$$1 \updownarrow \underbrace{r_{ij}}_{1} = \underbrace{\begin{bmatrix} m_{j1} \\ m_{j2} \\ m_{j3} \end{bmatrix}}_{1} \updownarrow n_f = 3 \quad \underbrace{\begin{bmatrix} u_{i1} & u_{i2} & u_{i3} \end{bmatrix}}_{n_f = 3} \updownarrow 1$$

Each entry in the user-rating matrix can be expressed as a matrix product

Matrix Factorization

$$1 \updownarrow \underset{\overset{\longleftrightarrow}{1}}{r_{ij}} = \underset{\overset{\longleftrightarrow}{1}}{\begin{bmatrix} m_{j1} \\ m_{j2} \\ m_{j3} \end{bmatrix}} \updownarrow n_f = 3 \; \underset{\overset{\longleftrightarrow}{n_f = 3}}{\begin{bmatrix} u_{i1} & u_{i2} & u_{i3} \end{bmatrix}} \updownarrow 1$$

If we generalize this we get a system of linear equations to be solved

Matrix Factorization

$$1 \; \underbrace{r_{ij}}_{1} \; = \; \underbrace{\begin{bmatrix} m_{j1} \\ m_{j2} \\ m_{j3} \end{bmatrix}}_{1} \; n_f = 3 \; \underbrace{\begin{bmatrix} u_{i1} & u_{i2} & u_{i3} \end{bmatrix}}_{n_f = 3} \; 1$$

Solving all of them simultaneously would allow us to estimate the entire matrix R

Matrix Factorization

$$n_u \; \underset{n_p}{R} \quad = \quad \left[\begin{array}{ccccc} r_{11} & r_{12} & r_{13} & & r_{1n_p} \\ r_{21} & r_{22} & r_{23} & & r_{2n_p} \\ r_{31} & r_{32} & r_{33} & \cdots & r_{3n_p} \\ \cdots & \cdots & \cdots & r_{ij} & \cdots \\ r_{n_u1} & r_{n_u2} & r_{n_u3} & & r_{n_un_p} \end{array}\right] \; n_u \text{ rows}$$

$$n_p \text{ columns}$$

Express this matrix as the product of two matrices, U and M

$$R = U \times M$$

n_u rows,	n_u rows,	n_f rows,
n_p columns	n_f columns	n_p columns

n_f is a hyperparameter

Estimating Rating Matrix

$$R = U \times M$$

n_u rows,
n_p columns

n_u rows,
n_f columns

n_f rows,
n_p columns

n_f is a hyperparameter

“rank”

“Number of latent factors”

“Dimensionality of feature space”

Estimating Rating Matrix

$$R = U \times M$$

n_u rows,	n_u rows,	n_f rows,
n_p columns	n_f columns	n_p columns

If R were available…

…Many matrix techniques to find U,M

e.g. Singular Value Decomposition

(Used in PCA)

$$R = U \times M$$

n_u rows, n_p columns	n_u rows, n_f columns	n_f rows, n_p columns

Estimating Rating Matrix

But R is not available and needs to be estimated

Use Alternating-Least-Squares (ALS)

Standard numerical algorithm

Alternating Least Squares (ALS)

Minimize

$$\sum_{i,\,j} (r_{ij} - u_i m_j)^2$$

To find

U, M

The value of U and M define the “best” rating matrix

$R = U \times M$

```
Step 1: Initialize M


Step 2:
  Fix M, solve to find U


Step 3:
  Fix U, solve to find M

Step 4:
  If stopping criterion not met
     Repeat Steps 2 and 3
```

- Assign average rating for that product as first row
  - Small random numbers for other rows
- Solve to minimize squared errors
- Solve to minimize squared errors
- Stop if RMSE on training data lower than some threshold

Estimating Rating Matrix

$$R = U \times M$$

n_u rows,
n_p columns

n_u rows,
n_f columns

n_f rows,
n_p columns

Each element of U, M is a free parameter

The number of free parameters is very large

Likely to lead to overfitting

Add regularization to penalize large parameters

ALS-WR

$$R = U \; x \; M$$

n_u rows, n_p columns	n_u rows, n_f columns	n_f rows, n_p columns

Alternating-Least-Squares (ALS)

Weighted Regularization (WR)

ALS-WR

Minimize

$$\sum_{i,j} (r_{ij} - u_i m_j)^2 + \lambda \left(\sum_{i} n_u^i u_i^2 + \sum_{j} n_m^j m_j^2 \right)$$

To find

U, M

λ is a hyperparameter that penalizes complex models

ALS-WR

Minimize

$$\sum_{i,j} (r_{ij} - u_i m_j)^2 + \lambda \left(\sum_{i} n_u{}^i u_i{}^2 + \sum_{j} n_m{}^j m_j{}^2 \right)$$

To find

U, M

λ is a hyperparameter that penalizes complex models

Difficulties in Estimation

Sparsity

Most initial entries will be missing and need estimation

Cold Start

New users or products with no history

Computational Intensity

Millions of users, millions of products

Difficulties in Estimation

Sparsity

Most initial entries will be missing and need estimation

Cold Start

New users or products with no history

Computational Intensity

Millions of users, millions of products

Choice of Rating Measure

Sparsity

Most initial entries will be missing and need estimation

Smart choice of rating measure can help mitigate

Two types

- explicit
- implicit

Spark supports both

Choice of Rating Measure

Implicit

Number of times viewed, length of time listened...

Explicit

Reviews or ratings manually entered, explicit actions performed...

Explicit and Implicit Feedback

Explicit Feedback	Implicit Feedback
Star ratings, thumbs-up/down buttons clicked	Browsing history, mouse movements, time spent listening/viewing
Far less available	Easily available and plentiful
Easy to use directly in recommendation algorithms	Hard to use directly in recommendation algorithms for several reasons
Explicit ratings express preference between items	Implicit ratings do not express preferences between items
Can be used to order preferences	Can only use to model confidence we have in a particular observation

Challenges of Implicit Feedback

No negative feedback

Inherently noisy (viewing to buy a gift?)

Unlike explicit feedback - no preference or order

However implicit feedback does give us confidence in data items

Spark's Approach to Implicit Feedback

Treat elements of R as explicit feedback

But do so in a smart, efficient way

Allows use of implicit feedback without compromising correctness

Basically - model R as strength of observations

Spark's Approach to Implicit Feedback

"Collaborative Filtering for Implicit Feedback Datasets"

- Yifan Hu, Yehuda Koren, Chris Volinsky (2008)

Spark's Approach to Implicit Feedback

Optimization: ALS-WR can be used

- RMSE fine for use here

Evaluation: RMSE will not suffice

- RMSE won't work (no negative feedback)
- Use rank measure instead
- Math is complex and out-of-scope

Difficulties in Estimation

Sparsity

Most initial entries will be missing and need estimation

Cold Start

New users or products with no history

Computational Intensity

Millions of users, millions of products

Cold Start

During Validation

Encounter user or product not in the training dataset

In Production

Entirely new products or users enter the system

Cold Start

New users or products with no history

Cold Start Strategies

Different strategies needed for these two cases

- In production: Spark assigns NaN (default)
- In cross-validation: Spark will “drop” row

“nan” and “drop” are two permissible strategies

Difficulties in Estimation

Sparsity

Most initial entries will be missing and need estimation

Cold Start

New users or products with no history

Computational Intensity

Millions of users, millions of products

Spark's Approach to Scalability

"Collaborative Filtering for Implicit Feedback Datasets"

- Yifan Hu, Yehuda Koren, Chris Volinsky (2008)

This algorithm not only solves for implicit feedback…

…it also scales linearly with dataset size!

Demo

Recommendation systems using explicit ratings in spark.ml

Demo

Recommendation systems using implicit ratings in spark.ml

Overview

Collaborative filtering algorithms are used to make product recommendations to users

Spark offers estimators which use the ALS method to implement collaborative filtering

Recommendations can be based on explicit and implicit ratings

Related Courses - Spark

Handling Fast Data with Apache Spark SQL and Streaming

- Programming in Spark 2 using Scala

Getting Started with Stream Processing with Spark Streaming

- Stream processing with Spark 1.x in Python

Related Courses - ML

Understanding the Foundations of TensorFlow

- Introduction to TensorFlow to build NNs

TensorFlow: Getting Started