Matrix Factorization for Movie Recommendation Systems

Nipun Batra

IIT Gandhinagar

August 30, 2025

Today's Learning Journey

- **The Problem**: Why do we need recommendation systems?

Today's Learning Journey

- **The Problem**: Why do we need recommendation systems?
- **Matrix View**: How ratings become a mathematical problem

Today's Learning Journey

- **The Problem**: Why do we need recommendation systems?
- **Matrix View**: How ratings become a mathematical problem
- **Key Insight**: Matrix factorization as the solution

Today's Learning Journey

- **The Problem**: Why do we need recommendation systems?
- **Matrix View**: How ratings become a mathematical problem
- **Key Insight**: Matrix factorization as the solution
- **Step-by-Step**: Building intuition with examples

Today's Learning Journey

- **The Problem**: Why do we need recommendation systems?
- **Matrix View**: How ratings become a mathematical problem
- **Key Insight**: Matrix factorization as the solution
- **Step-by-Step**: Building intuition with examples
- **Algorithms**: ALS vs Gradient Descent

Today's Learning Journey

- **The Problem**: Why do we need recommendation systems?
- **Matrix View**: How ratings become a mathematical problem
- **Key Insight**: Matrix factorization as the solution
- **Step-by-Step**: Building intuition with examples
- **Algorithms**: ALS vs Gradient Descent
- **Practice**: Hands-on understanding

Problem Setup

The Movie Recommendation Challenge

Real-World Scenario:

- Netflix: 200M+ users, 15K+ titles

The Movie Recommendation Challenge

Real-World Scenario:

- Netflix: 200M+ users, 15K+ titles
- Amazon: 300M+ users, millions of products

The Movie Recommendation Challenge

Real-World Scenario:

- Netflix: 200M+ users, 15K+ titles
- Amazon: 300M+ users, millions of products
- Spotify: 400M+ users, 70M+ songs

The Movie Recommendation Challenge

Real-World Scenario:

- Netflix: 200M+ users, 15K+ titles
- Amazon: 300M+ users, millions of products
- Spotify: 400M+ users, 70M+ songs
- Most ratings are missing!

The Movie Recommendation Challenge

Real-World Scenario:

- Netflix: 200M+ users, 15K+ titles
- Amazon: 300M+ users, millions of products
- Spotify: 400M+ users, 70M+ songs
- Most ratings are missing!

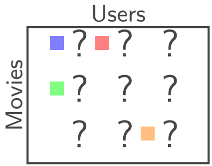
The Movie Recommendation Challenge

Real-World Scenario:

- Netflix: 200M+ users, 15K+ titles
- Amazon: 300M+ users, millions of products
- Spotify: 400M+ users, 70M+ songs
- Most ratings are missing!

Think About It:

- You've rated 100 movies out of 15,000



Sparse Rating Matrix

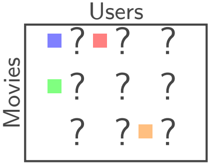
The Movie Recommendation Challenge

Real-World Scenario:

- Netflix: 200M+ users, 15K+ titles
- Amazon: 300M+ users, millions of products
- Spotify: 400M+ users, 70M+ songs
- Most ratings are missing!

Think About It:

- You've rated 100 movies out of 15,000
- Your friend has similar but different tastes



Sparse Rating Matrix

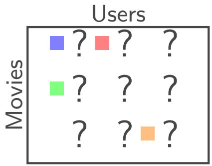
The Movie Recommendation Challenge

Real-World Scenario:

- Netflix: 200M+ users, 15K+ titles
- Amazon: 300M+ users, millions of products
- Spotify: 400M+ users, 70M+ songs
- Most ratings are missing!

Think About It:

- You've rated 100 movies out of 15,000
- Your friend has similar but different tastes
- How do we predict what



Sparse Rating Matrix

Pop Quiz 1: Understanding the Scale

Quick Question

If Netflix has 200 million users and 15,000 movies, how many possible ratings exist?

Pop Quiz 1: Understanding the Scale

Quick Question

If Netflix has 200 million users and 15,000 movies, how many possible ratings exist?

Answer: $200 \times 10^6 \times 15 \times 10^3 = 3 \times 10^{12}$ possible ratings!

Pop Quiz 1: Understanding the Scale

Quick Question

If Netflix has 200 million users and 15,000 movies, how many possible ratings exist?

Answer: $200 \times 10^6 \times 15 \times 10^3 = 3 \times 10^{12}$ possible ratings! But typical users rate only 20-100 movies. What percentage of the matrix is filled?

Pop Quiz 1: Understanding the Scale

Quick Question

If Netflix has 200 million users and 15,000 movies, how many possible ratings exist?

Answer: $200 \times 10^6 \times 15 \times 10^3 = 3 \times 10^{12}$ possible ratings!
But typical users rate only 20-100 movies. What percentage of the matrix is filled?
Answer: $\frac{100}{15000} = 0.67\%$ - extremely sparse!

Mathematical Problem Setup

The Rating Matrix $\mathbf{A} \in \mathbb{R}^{N \times M}$:

Mathematical Problem Setup

The Rating Matrix $\mathbf{A} \in \mathbb{R}^{N \times M}$:

$$\mathbf{A} = \begin{bmatrix} a_{11} & ? & a_{13} & ? & \cdots \\ ? & a_{22} & ? & a_{24} & \cdots \\ a_{31} & ? & ? & a_{34} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

Mathematical Problem Setup

The Rating Matrix $\mathbf{A} \in \mathbb{R}^{N \times M}$:

$$\mathbf{A} = \begin{bmatrix} a_{11} & ? & a_{13} & ? & \cdots \\ ? & a_{22} & ? & a_{24} & \cdots \\ a_{31} & ? & ? & a_{34} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

- **Rows**: Users $u_1, u_2, \ldots, u_N$

Mathematical Problem Setup

The Rating Matrix $\mathbf{A} \in \mathbb{R}^{N \times M}$:

$$\mathbf{A} = \begin{bmatrix} a_{11} & ? & a_{13} & ? & \cdots \\ ? & a_{22} & ? & a_{24} & \cdots \\ a_{31} & ? & ? & a_{34} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

- **Rows**: Users $u_1, u_2, \ldots, u_N$
- **Columns**: Movies $m_1, m_2, \ldots, m_M$

Mathematical Problem Setup

The Rating Matrix $\mathbf{A} \in \mathbb{R}^{N \times M}$:

$$\mathbf{A} = \begin{bmatrix} a_{11} & ? & a_{13} & ? & \cdots \\ ? & a_{22} & ? & a_{24} & \cdots \\ a_{31} & ? & ? & a_{34} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

- **Rows**: Users $u_1, u_2, \ldots, u_N$
- **Columns**: Movies $m_1, m_2, \ldots, m_M$
- **Entries**: $a_{ij} \in \{1, 2, 3, 4, 5\}$ (when observed)

Mathematical Problem Setup

The Rating Matrix $\mathbf{A} \in \mathbb{R}^{N \times M}$:

$$\mathbf{A} = \begin{bmatrix} a_{11} & ? & a_{13} & ? & \cdots \\ ? & a_{22} & ? & a_{24} & \cdots \\ a_{31} & ? & ? & a_{34} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

- **Rows**: Users $u_1, u_2, \ldots, u_N$
- **Columns**: Movies $m_1, m_2, \ldots, m_M$
- **Entries**: $a_{ij} \in \{1, 2, 3, 4, 5\}$ (when observed)
- **Challenge**: Predict missing entries $?$

Mathematical Problem Setup

The Rating Matrix $\mathbf{A} \in \mathbb{R}^{N \times M}$:

$$\mathbf{A} = \begin{bmatrix} a_{11} & ? & a_{13} & ? & \cdots \\ ? & a_{22} & ? & a_{24} & \cdots \\ a_{31} & ? & ? & a_{34} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

- **Rows**: Users $u_1, u_2, \ldots, u_N$
- **Columns**: Movies $m_1, m_2, \ldots, m_M$
- **Entries**: $a_{ij} \in \{1, 2, 3, 4, 5\}$ (when observed)
- **Challenge**: Predict missing entries ?
- **Notation**: $\Omega = \{(i, j) : a_{ij} \text{ is observed}\}$

Concrete Example: Our Movie Dataset

Let's work with a small, concrete example:

Concrete Example: Our Movie Dataset

Let's work with a small, concrete example:

User	Sholay	Swades	Batman	Interstellar	Shawshank
Alice	5	4	2	3	2
Bob	?	5	1	4	?
Carol	4	?	1	5	?

Concrete Example: Our Movie Dataset

Let's work with a small, concrete example:

User	Sholay	Swades	Batman	Interstellar	Shawshank
Alice	5	4	2	3	2
Bob	?	5	1	4	?
Carol	4	?	1	5	?

Observations:

- Alice loves Bollywood films (Sholay, Swades)

Concrete Example: Our Movie Dataset

Let's work with a small, concrete example:

User	**Sholay**	**Swades**	**Batman**	**Interstellar**	**Shawshank**
Alice	5	4	2	3	2
Bob	?	5	1	4	?
Carol	4	?	1	5	?

Observations:

- Alice loves Bollywood films (Sholay, Swades)
- Carol enjoys Sci-Fi (Interstellar)

Concrete Example: Our Movie Dataset

Let's work with a small, concrete example:

User	Sholay	Swades	Batman	Interstellar	Shawshank
Alice	5	4	2	3	2
Bob	?	5	1	4	?
Carol	4	?	1	5	?

Observations:

- Alice loves Bollywood films (Sholay, Swades)
- Carol enjoys Sci-Fi (Interstellar)
- Can we predict Bob's rating for Sholay?

Concrete Example: Our Movie Dataset

Let's work with a small, concrete example:

User	Sholay	Swades	Batman	Interstellar	Shawshank
Alice	5	4	2	3	2
Bob	?	5	1	4	?
Carol	4	?	1	5	?

Observations:

- Alice loves Bollywood films (Sholay, Swades)
- Carol enjoys Sci-Fi (Interstellar)
- Can we predict Bob's rating for Sholay?
- Can we predict Carol's rating for Swades?

Key Insight: Latent Features

Before We Dive In: A Simple Question

Why do you like the movies you like?

Before We Dive In: A Simple Question

Why do you like the movies you like?

Maybe because of:

- Genre (Action, Romance, Comedy)

Before We Dive In: A Simple Question

Why do you like the movies you like?

Maybe because of:

- Genre (Action, Romance, Comedy)
- Star cast (Shah Rukh Khan, Tom Cruise)

Before We Dive In: A Simple Question

Why do you like the movies you like?

Maybe because of:

- Genre (Action, Romance, Comedy)
- Star cast (Shah Rukh Khan, Tom Cruise)
- Director (Christopher Nolan, Rajkumar Hirani)

Before We Dive In: A Simple Question

Why do you like the movies you like?

Maybe because of:

- Genre (Action, Romance, Comedy)
- Star cast (Shah Rukh Khan, Tom Cruise)
- Director (Christopher Nolan, Rajkumar Hirani)
- Language (Hindi, English, Tamil)

Before We Dive In: A Simple Question

Why do you like the movies you like?

Maybe because of:

- Genre (Action, Romance, Comedy)
- Star cast (Shah Rukh Khan, Tom Cruise)
- Director (Christopher Nolan, Rajkumar Hirani)
- Language (Hindi, English, Tamil)
- Era (90s classics, modern CGI)

Key Insight:

- Your taste = combination of preferences
- Movie appeal = combination of features
- But we don't know these explicitly!

The Core Insight: Hidden Patterns

Hypothesis: User preferences and movie characteristics can be captured by a small number of **latent features**.

The Core Insight: Hidden Patterns

Hypothesis: User preferences and movie characteristics can be captured by a small number of **latent features**.

Intuition: Think of latent features as "hidden DNA" of movies and users!

The Core Insight: Hidden Patterns

Hypothesis: User preferences and movie characteristics can be captured by a small number of **latent features**.
Intuition: Think of latent features as "hidden DNA" of movies and users!

For Movies:

- Bollywood vs Hollywood

The Core Insight: Hidden Patterns

Hypothesis: User preferences and movie characteristics can be captured by a small number of **latent features**.
Intuition: Think of latent features as "hidden DNA" of movies and users!

For Movies:

- Bollywood vs Hollywood
- Action vs Drama

The Core Insight: Hidden Patterns

Hypothesis: User preferences and movie characteristics can be captured by a small number of **latent features**.
Intuition: Think of latent features as "hidden DNA" of movies and users!

For Movies:

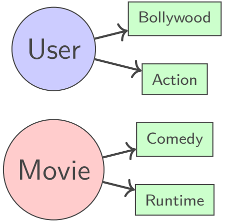
- Bollywood vs Hollywood
- Action vs Drama
- Comedy vs Serious

The Core Insight: Hidden Patterns

Hypothesis: User preferences and movie characteristics can be captured by a small number of **latent features**.
Intuition: Think of latent features as "hidden DNA" of movies and users!

For Movies:

- Bollywood vs Hollywood
- Action vs Drama
- Comedy vs Serious
- Runtime (Short vs Long)



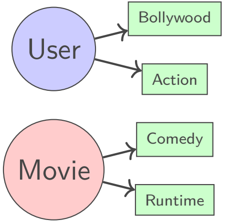
Latent Features

The Core Insight: Hidden Patterns

Hypothesis: User preferences and movie characteristics can be captured by a small number of **latent features**.
Intuition: Think of latent features as "hidden DNA" of movies and users!

For Movies:

- Bollywood vs Hollywood
- Action vs Drama
- Comedy vs Serious
- Runtime (Short vs Long)
- Year (Classic vs Modern)



Latent Features

Step 1: Define Movie Features Explicitly

Let's manually define features for our 5 movies:

Step 1: Define Movie Features Explicitly

Let's manually define features for our 5 movies:

Movie	**Bollywood**	**Sci-Fi**	**Drama**
Sholay	0.95	0.10	0.85
Swades	1.00	0.20	0.90
Batman	0.05	0.80	0.30
Interstellar	0.05	0.95	0.70
Shawshank	0.05	0.15	0.95

Step 1: Define Movie Features Explicitly

Let's manually define features for our 5 movies:

Movie	**Bollywood**	**Sci-Fi**	**Drama**
Sholay	0.95	0.10	0.85
Swades	1.00	0.20	0.90
Batman	0.05	0.80	0.30
Interstellar	0.05	0.95	0.70
Shawshank	0.05	0.15	0.95

Movie Feature Matrix $\mathbf{H} \in \mathbb{R}^{3\times 5}$:

$$\mathbf{H} = \begin{bmatrix} 0.95 & 1.00 & 0.05 & 0.05 & 0.05 \\ 0.10 & 0.20 & 0.80 & 0.95 & 0.15 \\ 0.85 & 0.90 & 0.30 & 0.70 & 0.95 \end{bmatrix}$$

Step 2: What About User Preferences?

User Feature Matrix $\mathbf{W} \in \mathbb{R}^{3\times 3}$ represents user preferences:

Step 2: What About User Preferences?

User Feature Matrix $\mathbf{W} \in \mathbb{R}^{3\times 3}$ represents user preferences:

$$\mathbf{W} = \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{bmatrix}$$

Step 2: What About User Preferences?

User Feature Matrix $\mathbf{W} \in \mathbb{R}^{3\times 3}$ represents user preferences:

$$\mathbf{W} = \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{bmatrix}$$

Where row i represents user i's affinity for:

- w_{i1}: Bollywood preference

Step 2: What About User Preferences?

User Feature Matrix $\mathbf{W} \in \mathbb{R}^{3\times 3}$ represents user preferences:

$$\mathbf{W} = \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{bmatrix}$$

Where row i represents user i's affinity for:

- w_{i1}: Bollywood preference
- w_{i2}: Sci-Fi preference

Step 2: What About User Preferences?

User Feature Matrix $\mathbf{W} \in \mathbb{R}^{3\times 3}$ represents user preferences:

$$\mathbf{W} = \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{bmatrix}$$

Where row i represents user i's affinity for:

- w_{i1}: Bollywood preference
- w_{i2}: Sci-Fi preference
- w_{i3}: Drama preference

Step 2: What About User Preferences?

User Feature Matrix $\mathbf{W} \in \mathbb{R}^{3\times 3}$ represents user preferences:

$$\mathbf{W} = \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{bmatrix}$$

Where row i represents user i's affinity for:

- w_{i1}: Bollywood preference
- w_{i2}: Sci-Fi preference
- w_{i3}: Drama preference

Step 2: What About User Preferences?

User Feature Matrix $\mathbf{W} \in \mathbb{R}^{3\times 3}$ represents user preferences:

$$\mathbf{W} = \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{bmatrix}$$

Where row i represents user i's affinity for:

- w_{i1}: Bollywood preference
- w_{i2}: Sci-Fi preference
- w_{i3}: Drama preference

Key Question: How do we learn these w_{ij} values from observed ratings?

Step 3: The Matrix Factorization Idea

Core Hypothesis: Rating = User preferences · Movie features

Step 3: The Matrix Factorization Idea

Core Hypothesis: Rating = User preferences · Movie features

$$\boxed{a_{ij} \approx \mathbf{w}_i^T \mathbf{h}_j = \sum_{k=1}^{r} w_{ik} h_{kj}}$$

Step 3: The Matrix Factorization Idea

Core Hypothesis: Rating = User preferences · Movie features

$$\boxed{a_{ij} \approx \mathbf{w}_i^T \mathbf{h}_j = \sum_{k=1}^{r} w_{ik} h_{kj}}$$

In Matrix Form:

$$\boxed{\mathbf{A} \approx \mathbf{W}\mathbf{H}}$$

Step 3: The Matrix Factorization Idea

Core Hypothesis: Rating = User preferences · Movie features

$$\boxed{a_{ij} \approx \mathbf{w}_i^T \mathbf{h}_j = \sum_{k=1}^{r} w_{ik} h_{kj}}$$

In Matrix Form:

$$\boxed{\mathbf{A} \approx \mathbf{W}\mathbf{H}}$$

$$\mathbf{A}_{3\times5} = \begin{bmatrix} 5 & 4 & 2 & 3 & 2 \\ ? & 5 & 1 & 4 & ? \\ 4 & ? & 1 & 5 & ? \end{bmatrix} \approx$$

$$\begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{bmatrix} \begin{bmatrix} 0.95 & 1.00 & 0.05 & 0.05 & 0.05 \\ 0.10 & 0.20 & 0.80 & 0.95 & 0.15 \\ 0.85 & 0.90 & 0.30 & 0.70 & 0.95 \end{bmatrix} =$$

$$\mathbf{W}_{3\times3}\mathbf{H}_{3\times5}$$

Step 4: Understanding the Calculation

Let's think step by step...

Step 4: Understanding the Calculation

Let's think step by step...

Alice's Profile:

- How much does she like Bollywood? w_{11}

Step 4: Understanding the Calculation

Let's think step by step...

Alice's Profile:

- How much does she like Bollywood? w_{11}
- How much does she like Action? w_{12}

Step 4: Understanding the Calculation

Let's think step by step...

Alice's Profile:

- How much does she like Bollywood? w_{11}
- How much does she like Action? w_{12}
- How much does she like Comedy? w_{13}

Sholay's DNA:

- Bollywood-ness: 0.95 (very high!)
- Action-ness: 0.10 (low)
- Comedy-ness: 0.85 (high)

Step 4: Understanding the Calculation

Let's think step by step...

Alice's Profile:

- How much does she like Bollywood? w_{11}
- How much does she like Action? w_{12}
- How much does she like Comedy? w_{13}

Sholay's DNA:

- Bollywood-ness: 0.95 (very high!)
- Action-ness: 0.10 (low)
- Comedy-ness: 0.85 (high)

Step 4: Understanding the Calculation

Let's think step by step...

Alice's Profile:

- How much does she like Bollywood? w_{11}
- How much does she like Action? w_{12}
- How much does she like Comedy? w_{13}

Sholay's DNA:

- Bollywood-ness: 0.95 (very high!)
- Action-ness: 0.10 (low)
- Comedy-ness: 0.85 (high)

The Magic Formula:

$$\text{Alice's rating} = \text{Alice's preferences} \cdot \text{Sholay's features}$$

Step 4: Detailed Calculation Example

Let's compute Alice's predicted rating for Sholay:

Step 4: Detailed Calculation Example

Let's compute Alice's predicted rating for Sholay:
Alice's preferences: $\mathbf{w}_1 = [w_{11}, w_{12}, w_{13}]$ **Sholay's features:** $\mathbf{h}_1 = [0.95, 0.10, 0.85]^T$

Step 4: Detailed Calculation Example

Let's compute Alice's predicted rating for Sholay:
Alice's preferences: $\mathbf{w}_1 = [w_{11}, w_{12}, w_{13}]$ **Sholay's features:** $\mathbf{h}_1 = [0.95, 0.10, 0.85]^T$

$$\hat{a}_{11} = \mathbf{w}_1^T \mathbf{h}_1 \quad (1)$$

$$= w_{11} \cdot 0.95 + w_{12} \cdot 0.10 + w_{13} \cdot 0.85 \quad (2)$$

Step 4: Detailed Calculation Example

Let's compute Alice's predicted rating for Sholay:
Alice's preferences: $\mathbf{w}_1 = [w_{11}, w_{12}, w_{13}]$ **Sholay's features:** $\mathbf{h}_1 = [0.95, 0.10, 0.85]^T$

$$\hat{a}_{11} = \mathbf{w}_1^T \mathbf{h}_1 \tag{1}$$

$$= w_{11} \cdot 0.95 + w_{12} \cdot 0.10 + w_{13} \cdot 0.85 \tag{2}$$

Goal: Find w_{11}, w_{12}, w_{13} such that $\hat{a}_{11} \approx 5$ (Alice's actual rating)

Pop Quiz 2: Matrix Dimensions

Dimension Check

If we have N users, M movies, and r latent features:

Pop Quiz 2: Matrix Dimensions

Dimension Check

If we have N users, M movies, and r latent features:

1. What are the dimensions of $\mathbf{A}$?

Pop Quiz 2: Matrix Dimensions

Dimension Check

If we have N users, M movies, and r latent features:

1. What are the dimensions of $\mathbf{A}$?
2. What are the dimensions of $\mathbf{W}$?

Pop Quiz 2: Matrix Dimensions

Dimension Check

If we have N users, M movies, and r latent features:

1. What are the dimensions of $\mathbf{A}$?
2. What are the dimensions of $\mathbf{W}$?
3. What are the dimensions of $\mathbf{H}$?

Pop Quiz 2: Matrix Dimensions

Dimension Check

If we have N users, M movies, and r latent features:

1. What are the dimensions of $\mathbf{A}$?
2. What are the dimensions of $\mathbf{W}$?
3. What are the dimensions of $\mathbf{H}$?
4. How many parameters do we need to learn?

Pop Quiz 2: Matrix Dimensions

Dimension Check

If we have N users, M movies, and r latent features:

1. What are the dimensions of $\mathbf{A}$?
2. What are the dimensions of $\mathbf{W}$?
3. What are the dimensions of $\mathbf{H}$?
4. How many parameters do we need to learn?

Pop Quiz 2: Matrix Dimensions

Dimension Check

If we have N users, M movies, and r latent features:

1. What are the dimensions of $\mathbf{A}$?
2. What are the dimensions of $\mathbf{W}$?
3. What are the dimensions of $\mathbf{H}$?
4. How many parameters do we need to learn?

Answers:

1. $\mathbf{A} \in \mathbb{R}^{N \times M}$
2. $\mathbf{W} \in \mathbb{R}^{N \times r}$
3. $\mathbf{H} \in \mathbb{R}^{r \times M}$
4. Total parameters: $Nr + rM = r(N + M)$

Pop Quiz 2: Matrix Dimensions

Dimension Check

If we have N users, M movies, and r latent features:

1. What are the dimensions of $\mathbf{A}$?
2. What are the dimensions of $\mathbf{W}$?
3. What are the dimensions of $\mathbf{H}$?
4. How many parameters do we need to learn?

Answers:

1. $\mathbf{A} \in \mathbb{R}^{N \times M}$
2. $\mathbf{W} \in \mathbb{R}^{N \times r}$
3. $\mathbf{H} \in \mathbb{R}^{r \times M}$
4. Total parameters: $Nr + rM = r(N + M)$

Key Insight: If $r \ll \min(N, M)$, we have huge parameter

Learning the Factorization

The Optimization Problem

Objective: Minimize prediction error on observed ratings only

The Optimization Problem

Objective: Minimize prediction error on observed ratings only

$$\boxed{\text{minimize}_{\mathbf{W},\mathbf{H}} \sum_{(i,j)\in\Omega} (a_{ij} - \mathbf{w}_i^T \mathbf{h}_j)^2}$$

The Optimization Problem

Objective: Minimize prediction error on observed ratings only

$$\boxed{\text{minimize}_{\mathbf{W},\mathbf{H}} \sum_{(i,j)\in\Omega} (a_{ij} - \mathbf{w}_i^T \mathbf{h}_j)^2}$$

In Matrix Notation:

$$\boxed{\text{minimize}_{\mathbf{W},\mathbf{H}} \, \|P_\Omega(\mathbf{A} - \mathbf{WH})\|_F^2}$$

The Optimization Problem

Objective: Minimize prediction error on observed ratings only

$$\boxed{\text{minimize}_{\mathbf{W},\mathbf{H}} \sum_{(i,j)\in\Omega} (a_{ij} - \mathbf{w}_i^T \mathbf{h}_j)^2}$$

In Matrix Notation:

$$\boxed{\text{minimize}_{\mathbf{W},\mathbf{H}} \, \|P_\Omega(\mathbf{A} - \mathbf{W}\mathbf{H})\|_F^2}$$

Where:

- $P_\Omega(\cdot)$: projection onto observed entries

The Optimization Problem

Objective: Minimize prediction error on observed ratings only

$$\boxed{\text{minimize}_{\mathbf{W},\mathbf{H}} \sum_{(i,j)\in\Omega} (a_{ij} - \mathbf{w}_i^T \mathbf{h}_j)^2}$$

In Matrix Notation:

$$\boxed{\text{minimize}_{\mathbf{W},\mathbf{H}} \, \|P_\Omega(\mathbf{A} - \mathbf{W}\mathbf{H})\|_F^2}$$

Where:

- $P_\Omega(\cdot)$: projection onto observed entries
- $\|\cdot\|_F$: Frobenius norm

The Optimization Problem

Objective: Minimize prediction error on observed ratings only

$$\boxed{\text{minimize}_{\mathbf{W},\mathbf{H}} \sum_{(i,j)\in\Omega} (a_{ij} - \mathbf{w}_i^T \mathbf{h}_j)^2}$$

In Matrix Notation:

$$\boxed{\text{minimize}_{\mathbf{W},\mathbf{H}} \, \|P_\Omega(\mathbf{A} - \mathbf{W}\mathbf{H})\|_F^2}$$

Where:

- $P_\Omega(\cdot)$: projection onto observed entries
- $\|\cdot\|_F$: Frobenius norm
- Ω: set of observed (i, j) pairs

Why This is Challenging

Problem Characteristics:

- **Non-convex:** Multiple local minima exist

Why This is Challenging

Problem Characteristics:

- **Non-convex:** Multiple local minima exist
- **Bilinear:** Linear in $\mathbf{W}$ when $\mathbf{H}$ fixed, and vice versa

Why This is Challenging

Problem Characteristics:

- **Non-convex:** Multiple local minima exist
- **Bilinear:** Linear in $\mathbf{W}$ when $\mathbf{H}$ fixed, and vice versa
- **Large-scale:** Millions of users and items

Why This is Challenging

Problem Characteristics:

- **Non-convex:** Multiple local minima exist
- **Bilinear:** Linear in $\mathbf{W}$ when $\mathbf{H}$ fixed, and vice versa
- **Large-scale:** Millions of users and items
- **Sparse:** Only 0.1-1% of entries observed

Why This is Challenging

Problem Characteristics:

- **Non-convex:** Multiple local minima exist
- **Bilinear:** Linear in $\mathbf{W}$ when $\mathbf{H}$ fixed, and vice versa
- **Large-scale:** Millions of users and items
- **Sparse:** Only 0.1-1% of entries observed

Why This is Challenging

Problem Characteristics:

- **Non-convex:** Multiple local minima exist
- **Bilinear:** Linear in $\mathbf{W}$ when $\mathbf{H}$ fixed, and vice versa
- **Large-scale:** Millions of users and items
- **Sparse:** Only 0.1-1% of entries observed

Key Insight: While non-convex jointly, it's convex in each matrix individually!

Algorithm 1: Alternating Least Squares (ALS)

ALS: The Big Picture

Alternating Least Squares Strategy:

ALS: The Big Picture

Alternating Least Squares Strategy:

1. **Initialize:** $\mathbf{W}^{(0)}$ and $\mathbf{H}^{(0)}$ randomly

ALS: The Big Picture

Alternating Least Squares Strategy:

1. **Initialize:** $\mathbf{W}^{(0)}$ and $\mathbf{H}^{(0)}$ randomly
2. **Repeat until convergence:**

ALS: The Big Picture

Alternating Least Squares Strategy:

1. **Initialize:** $\mathbf{W}^{(0)}$ and $\mathbf{H}^{(0)}$ randomly
2. **Repeat until convergence:**
    1) **Fix $\mathbf{H}$, solve for $\mathbf{W}$:** Each row independently

ALS: The Big Picture

Alternating Least Squares Strategy:

1. **Initialize:** $\mathbf{W}^{(0)}$ and $\mathbf{H}^{(0)}$ randomly
2. **Repeat until convergence:**
   1) **Fix $\mathbf{H}$, solve for $\mathbf{W}$:** Each row independently
   2) **Fix $\mathbf{W}$, solve for $\mathbf{H}$:** Each column independently

ALS: The Big Picture

Alternating Least Squares Strategy:

1. **Initialize:** $\mathbf{W}^{(0)}$ and $\mathbf{H}^{(0)}$ randomly
2. **Repeat until convergence:**
   1) **Fix $\mathbf{H}$, solve for $\mathbf{W}$:** Each row independently
   2) **Fix $\mathbf{W}$, solve for $\mathbf{H}$:** Each column independently
3. Each subproblem is a standard least squares problem!

ALS: The Big Picture

Alternating Least Squares Strategy:

1. **Initialize:** $\mathbf{W}^{(0)}$ and $\mathbf{H}^{(0)}$ randomly
2. **Repeat until convergence:**
    1) **Fix $\mathbf{H}$, solve for $\mathbf{W}$:** Each row independently
    2) **Fix $\mathbf{W}$, solve for $\mathbf{H}$:** Each column independently
3. Each subproblem is a standard least squares problem!

ALS: The Big Picture

Alternating Least Squares Strategy:

1. **Initialize:** $\mathbf{W}^{(0)}$ and $\mathbf{H}^{(0)}$ randomly
2. **Repeat until convergence:**
    1) **Fix $\mathbf{H}$, solve for $\mathbf{W}$:** Each row independently
    2) **Fix $\mathbf{W}$, solve for $\mathbf{H}$:** Each column independently
3. Each subproblem is a standard least squares problem!

Fix $\mathbf{W}$ → Solve $\mathbf{H}$ → Fix $\mathbf{H}$ → Solve $\mathbf{W}$

ALS Step 1: Updating User Features

Fix $\mathbb{H}$, solve for each user i independently:

ALS Step 1: Updating User Features

Fix $\mathbf{H}$, solve for each user i independently:

$$\text{minimize}_{\mathbf{w}_i} \sum_{j:(i,j)\in\Omega} (a_{ij} - \mathbf{w}_i^T \mathbf{h}_j)^2$$

ALS Step 1: Updating User Features

Fix $\mathbf{H}$, solve for each user i independently:

$$\text{minimize}_{\mathbf{w}_i} \sum_{j:(i,j)\in\Omega} (a_{ij} - \mathbf{w}_i^T \mathbf{h}_j)^2$$

Matrix Form for User i: Let $\Omega_i = \{j : (i,j) \in \Omega\}$ (movies rated by user i)

ALS Step 1: Updating User Features

Fix $\mathbf{H}$, solve for each user i independently:

$$\text{minimize}_{\mathbf{w}_i} \sum_{j:(i,j)\in\Omega} (a_{ij} - \mathbf{w}_i^T \mathbf{h}_j)^2$$

Matrix Form for User i: Let $\Omega_i = \{j : (i,j) \in \Omega\}$ (movies rated by user i)

$$\mathbf{y}_i = [a_{i,j_1}, a_{i,j_2}, \ldots, a_{i,j_{|\Omega_i|}}]^T \tag{3}$$

$$\mathbf{X}_i = [\mathbf{h}_{j_1}, \mathbf{h}_{j_2}, \ldots, \mathbf{h}_{j_{|\Omega_i|}}]^T \tag{4}$$

ALS Step 1: Updating User Features

Fix $\mathbf{H}$, solve for each user i independently:

$$\text{minimize}_{\mathbf{w}_i} \sum_{j:(i,j)\in\Omega} (a_{ij} - \mathbf{w}_i^T \mathbf{h}_j)^2$$

Matrix Form for User i: Let $\Omega_i = \{j : (i,j) \in \Omega\}$ (movies rated by user i)

$$\mathbf{y}_i = [a_{i,j_1}, a_{i,j_2}, \ldots, a_{i,j_{|\Omega_i|}}]^T \tag{3}$$

$$\mathbf{X}_i = [\mathbf{h}_{j_1}, \mathbf{h}_{j_2}, \ldots, \mathbf{h}_{j_{|\Omega_i|}}]^T \tag{4}$$

Least Squares Solution:

$$\boxed{\mathbf{w}_i^* = (\mathbf{X}_i^T \mathbf{X}_i)^{-1} \mathbf{X}_i^T \mathbf{y}_i}$$

ALS Step 1: Concrete Example

Update Alice's preferences ($\mathbf{w}_1$):
Alice rated: Sholay(5), Swades(4), Batman(2), Interstellar(3), Shawshank(2)

ALS Step 1: Concrete Example

Update Alice's preferences ($\mathbf{w}_1$):
Alice rated: Sholay(5), Swades(4), Batman(2), Interstellar(3), Shawshank(2)

$$\mathbf{y}_1 = \begin{bmatrix} 5 \\ 4 \\ 2 \\ 3 \\ 2 \end{bmatrix} \tag{5}$$

$$\mathbf{X}_1 = \begin{bmatrix} 0.95 & 0.10 & 0.85 \\ 1.00 & 0.20 & 0.90 \\ 0.05 & 0.80 & 0.30 \\ 0.05 & 0.95 & 0.70 \\ 0.05 & 0.15 & 0.95 \end{bmatrix} \tag{6}$$

ALS Step 1: Concrete Example

Update Alice's preferences ($\mathbf{w}_1$):
Alice rated: Sholay(5), Swades(4), Batman(2), Interstellar(3), Shawshank(2)

$$\mathbf{y}_1 = \begin{bmatrix} 5 \\ 4 \\ 2 \\ 3 \\ 2 \end{bmatrix} \tag{5}$$

$$\mathbf{X}_1 = \begin{bmatrix} 0.95 & 0.10 & 0.85 \\ 1.00 & 0.20 & 0.90 \\ 0.05 & 0.80 & 0.30 \\ 0.05 & 0.95 & 0.70 \\ 0.05 & 0.15 & 0.95 \end{bmatrix} \tag{6}$$

Solution: $\mathbf{w}_1^* = (\mathbf{X}_1^T \mathbf{X}_1)^{-1} \mathbf{X}_1^T \mathbf{y}_1$
This gives us Alice's feature preferences!

ALS Step 2: Updating Movie Features

Fix $\mathbf{W}$, solve for each movie j independently:

ALS Step 2: Updating Movie Features

Fix $\mathbf{W}$, solve for each movie j independently:

$$\text{minimize}_{\mathbf{h}_j} \sum_{i:(i,j)\in\Omega} (a_{ij} - \mathbf{w}_i^T \mathbf{h}_j)^2$$

ALS Step 2: Updating Movie Features

Fix $\mathbf{W}$, solve for each movie j independently:

$$\text{minimize}_{\mathbf{h}_j} \sum_{i:(i,j)\in\Omega} (a_{ij} - \mathbf{w}_i^T \mathbf{h}_j)^2$$

Matrix Form for Movie j: Let $\Omega_j = \{i : (i,j) \in \Omega\}$ (users who rated movie j)

ALS Step 2: Updating Movie Features

Fix $\mathbf{W}$, solve for each movie j independently:

$$\text{minimize}_{\mathbf{h}_j} \sum_{i:(i,j)\in\Omega} (a_{ij} - \mathbf{w}_i^T \mathbf{h}_j)^2$$

Matrix Form for Movie j: Let $\Omega_j = \{i : (i,j) \in \Omega\}$ (users who rated movie j)

$$\mathbf{y}_j = [a_{i_1,j}, a_{i_2,j}, \ldots, a_{i_{|\Omega_j|},j}]^T \quad (7)$$

$$\mathbf{X}_j = [\mathbf{w}_{i_1}, \mathbf{w}_{i_2}, \ldots, \mathbf{w}_{i_{|\Omega_j|}}]^T \quad (8)$$

ALS Step 2: Updating Movie Features

Fix $\mathbf{W}$, solve for each movie j independently:

$$\text{minimize}_{\mathbf{h}_j} \sum_{i:(i,j)\in\Omega} (a_{ij} - \mathbf{w}_i^T \mathbf{h}_j)^2$$

Matrix Form for Movie j: Let $\Omega_j = \{i : (i,j) \in \Omega\}$ (users who rated movie j)

$$\mathbf{y}_j = [a_{i_1,j}, a_{i_2,j}, \ldots, a_{i_{|\Omega_j|},j}]^T \tag{7}$$

$$\mathbf{X}_j = [\mathbf{w}_{i_1}, \mathbf{w}_{i_2}, \ldots, \mathbf{w}_{i_{|\Omega_j|}}]^T \tag{8}$$

Least Squares Solution:

$$\boxed{\mathbf{h}_j^* = (\mathbf{X}_j^T \mathbf{X}_j)^{-1} \mathbf{X}_j^T \mathbf{y}_j}$$

ALS: Complete Algorithm

Algorithm 1: [

H] **Input:** Rating matrix $\mathbf{A}$, rank r, max iterations T

1. **Initialize:** $\mathbf{W}^{(0)} \in \mathbb{R}^{N \times r}$, $\mathbf{H}^{(0)} \in \mathbb{R}^{r \times M}$ randomly

Output: $\mathbf{W}^{(T)}, \mathbf{H}^{(T)}$

ALS: Complete Algorithm

Algorithm 2: [

H] **Input:** Rating matrix $\mathbf{A}$, rank r, max iterations T

1. **Initialize:** $\mathbf{W}^{(0)} \in \mathbb{R}^{N \times r}$, $\mathbf{H}^{(0)} \in \mathbb{R}^{r \times M}$ randomly
2. **For** $t = 1, 2, \ldots, T$:

Output: $\mathbf{W}^{(T)}, \mathbf{H}^{(T)}$

ALS: Complete Algorithm

Algorithm 3: [

H] **Input:** Rating matrix $\mathbf{A}$, rank r, max iterations T

1. **Initialize:** $\mathbf{W}^{(0)} \in \mathbb{R}^{N\times r}$, $\mathbf{H}^{(0)} \in \mathbb{R}^{r\times M}$ randomly
2. **For** $t = 1, 2, \ldots, T$:
   1) **Update Users:** For each user $i = 1, \ldots, N$:

$$\mathbf{w}_i^{(t)} = (\mathbf{X}_i^T \mathbf{X}_i)^{-1} \mathbf{X}_i^T \mathbf{y}_i$$

Output: $\mathbf{W}^{(T)}, \mathbf{H}^{(T)}$

ALS: Complete Algorithm

Algorithm 4: [

H] **Input:** Rating matrix $\mathbf{A}$, rank r, max iterations T

1. **Initialize:** $\mathbf{W}^{(0)} \in \mathbb{R}^{N\times r}$, $\mathbf{H}^{(0)} \in \mathbb{R}^{r\times M}$ randomly
2. **For** $t = 1, 2, \ldots, T$:
   1) **Update Users:** For each user $i = 1, \ldots, N$:

   $$\mathbf{w}_i^{(t)} = (\mathbf{X}_i^T \mathbf{X}_i)^{-1} \mathbf{X}_i^T \mathbf{y}_i$$

   2) **Update Movies:** For each movie $j = 1, \ldots, M$:

   $$\mathbf{h}_j^{(t)} = (\mathbf{X}_j^T \mathbf{X}_j)^{-1} \mathbf{X}_j^T \mathbf{y}_j$$

Output: $\mathbf{W}^{(T)}$, $\mathbf{H}^{(T)}$

ALS: Complete Algorithm

Algorithm 5: [

H] **Input:** Rating matrix $\mathbf{A}$, rank r, max iterations T

1. **Initialize:** $\mathbf{W}^{(0)} \in \mathbb{R}^{N \times r}$, $\mathbf{H}^{(0)} \in \mathbb{R}^{r \times M}$ randomly
2. **For** $t = 1, 2, \ldots, T$:
   1) **Update Users:** For each user $i = 1, \ldots, N$:

   $$\mathbf{w}_i^{(t)} = (\mathbf{X}_i^T \mathbf{X}_i)^{-1} \mathbf{X}_i^T \mathbf{y}_i$$

   2) **Update Movies:** For each movie $j = 1, \ldots, M$:

   $$\mathbf{h}_j^{(t)} = (\mathbf{X}_j^T \mathbf{X}_j)^{-1} \mathbf{X}_j^T \mathbf{y}_j$$

3. **Check Convergence:** Stop if $\|\mathbf{W}^{(t)}\mathbf{H}^{(t)} - \mathbf{W}^{(t-1)}\mathbf{H}^{(t-1)}\|_F < \epsilon$

Output: $\mathbf{W}^{(T)}$, $\mathbf{H}^{(T)}$

Algorithm 2: Gradient Descent

Gradient Descent Approach

Simultaneous Updates: Update both $\mathbf{W}$ and $\mathbf{H}$ together

Gradient Descent Approach

Simultaneous Updates: Update both $\mathbf{W}$ and $\mathbf{H}$ together
Objective Function:

$$L(\mathbf{W}, \mathbf{H}) = \sum_{(i,j)\in\Omega} (a_{ij} - \mathbf{w}_i^T \mathbf{h}_j)^2$$

Gradient Descent Approach

Simultaneous Updates: Update both $\mathbf{W}$ and $\mathbf{H}$ together
Objective Function:

$$L(\mathbf{W}, \mathbf{H}) = \sum_{(i,j)\in\Omega} (a_{ij} - \mathbf{w}_i^T \mathbf{h}_j)^2$$

Gradients:

$$\frac{\partial L}{\partial \mathbf{w}_i} = -2 \sum_{j:(i,j)\in\Omega} (a_{ij} - \mathbf{w}_i^T \mathbf{h}_j)\mathbf{h}_j \tag{9}$$

$$\frac{\partial L}{\partial \mathbf{h}_j} = -2 \sum_{i:(i,j)\in\Omega} (a_{ij} - \mathbf{w}_i^T \mathbf{h}_j)\mathbf{w}_i \tag{10}$$

SGD: Learning Like a Human

Imagine you're learning someone's taste in movies...

SGD: Learning Like a Human

Imagine you're learning someone's taste in movies...

Your Process:

1. Make a guess about their rating

SGD: Learning Like a Human

Imagine you're learning someone's taste in movies...

Your Process:

1. Make a guess about their rating
2. See their actual rating

SGD: Learning Like a Human

Imagine you're learning someone's taste in movies...

Your Process:

1. Make a guess about their rating
2. See their actual rating
3. Adjust your understanding

SGD: Learning Like a Human

Imagine you're learning someone's taste in movies...

Your Process:

1. Make a guess about their rating
2. See their actual rating
3. Adjust your understanding
4. Repeat for next movie

SGD does exactly this!

- One rating at a time
- Small adjustments
- Gradually improves

Stochastic Gradient Descent (SGD)

For each observed rating $(i, j) \in \Omega$:

Stochastic Gradient Descent (SGD)

For each observed rating $(i, j) \in \Omega$**:**

1. **Predict:** $\hat{a}_{ij} = \mathbf{w}_i^T \mathbf{h}_j$

Stochastic Gradient Descent (SGD)

For each observed rating $(i, j) \in \Omega$:

1. **Predict:** $\hat{a}_{ij} = \mathbf{w}_i^T \mathbf{h}_j$
2. **Compute Error:** $e_{ij} = a_{ij} - \hat{a}_{ij}$

Stochastic Gradient Descent (SGD)

For each observed rating $(i, j) \in \Omega$:

1. **Predict:** $\hat{a}_{ij} = \mathbf{w}_i^T \mathbf{h}_j$
2. **Compute Error:** $e_{ij} = a_{ij} - \hat{a}_{ij}$
3. **Update:**

$$\mathbf{w}_i \leftarrow \mathbf{w}_i + \alpha \cdot e_{ij} \cdot \mathbf{h}_j \tag{11}$$

$$\mathbf{h}_j \leftarrow \mathbf{h}_j + \alpha \cdot e_{ij} \cdot \mathbf{w}_i \tag{12}$$

Stochastic Gradient Descent (SGD)

For each observed rating $(i, j) \in \Omega$:

1. **Predict:** $\hat{a}_{ij} = \mathbf{w}_i^T \mathbf{h}_j$
2. **Compute Error:** $e_{ij} = a_{ij} - \hat{a}_{ij}$
3. **Update:**

$$\mathbf{w}_i \leftarrow \mathbf{w}_i + \alpha \cdot e_{ij} \cdot \mathbf{h}_j \tag{11}$$

$$\mathbf{h}_j \leftarrow \mathbf{h}_j + \alpha \cdot e_{ij} \cdot \mathbf{w}_i \tag{12}$$

Stochastic Gradient Descent (SGD)

For each observed rating $(i, j) \in \Omega$:

1. **Predict:** $\hat{a}_{ij} = \mathbf{w}_i^T \mathbf{h}_j$
2. **Compute Error:** $e_{ij} = a_{ij} - \hat{a}_{ij}$
3. **Update:**

$$\mathbf{w}_i \leftarrow \mathbf{w}_i + \alpha \cdot e_{ij} \cdot \mathbf{h}_j \tag{11}$$

$$\mathbf{h}_j \leftarrow \mathbf{h}_j + \alpha \cdot e_{ij} \cdot \mathbf{w}_i \tag{12}$$

Intuition:

- If $e_{ij} > 0$: Predicted rating too low $\rightarrow$ Increase similarity

Stochastic Gradient Descent (SGD)

For each observed rating $(i, j) \in \Omega$:

1. **Predict:** $\hat{a}_{ij} = \mathbf{w}_i^T \mathbf{h}_j$
2. **Compute Error:** $e_{ij} = a_{ij} - \hat{a}_{ij}$
3. **Update:**

$$\mathbf{w}_i \leftarrow \mathbf{w}_i + \alpha \cdot e_{ij} \cdot \mathbf{h}_j \tag{11}$$

$$\mathbf{h}_j \leftarrow \mathbf{h}_j + \alpha \cdot e_{ij} \cdot \mathbf{w}_i \tag{12}$$

Intuition:

- If $e_{ij} > 0$: Predicted rating too low $\rightarrow$ Increase similarity
- If $e_{ij} < 0$: Predicted rating too high $\rightarrow$ Decrease similarity

Stochastic Gradient Descent (SGD)

For each observed rating $(i, j) \in \Omega$:

1. **Predict:** $\hat{a}_{ij} = \mathbf{w}_i^T \mathbf{h}_j$
2. **Compute Error:** $e_{ij} = a_{ij} - \hat{a}_{ij}$
3. **Update:**

$$\mathbf{w}_i \leftarrow \mathbf{w}_i + \alpha \cdot e_{ij} \cdot \mathbf{h}_j \tag{11}$$

$$\mathbf{h}_j \leftarrow \mathbf{h}_j + \alpha \cdot e_{ij} \cdot \mathbf{w}_i \tag{12}$$

Intuition:

- If $e_{ij} > 0$: Predicted rating too low $\rightarrow$ Increase similarity
- If $e_{ij} < 0$: Predicted rating too high $\rightarrow$ Decrease similarity
- Learning rate α controls step size

SGD: Step-by-Step Example

Example: Alice rates Sholay as 5, but we predict 3.2

SGD: Step-by-Step Example

Example: Alice rates Sholay as 5, but we predict 3.2

$$\text{Current: } \mathbf{w}_1 = [0.4, 0.2, 0.3], \quad \mathbf{h}_1 = [0.95, 0.10, 0.85] \tag{13}$$

$$\text{Prediction: } \hat{a}_{11} = 0.4 \times 0.95 + 0.2 \times 0.10 + 0.3 \times 0.85 = 0.655 \tag{14}$$

$$\text{Error: } e_{11} = 5 - 0.655 = 4.345 \tag{15}$$

SGD: Step-by-Step Example

Example: Alice rates Sholay as 5, but we predict 3.2

$$\text{Current: } \mathbf{w}_1 = [0.4, 0.2, 0.3], \quad \mathbf{h}_1 = [0.95, 0.10, 0.85] \tag{13}$$

$$\text{Prediction: } \hat{a}_{11} = 0.4 \times 0.95 + 0.2 \times 0.10 + 0.3 \times 0.85 = 0.655 \tag{14}$$

$$\text{Error: } e_{11} = 5 - 0.655 = 4.345 \tag{15}$$

Updates with $\alpha = 0.01$:

$$\mathbf{w}_1 \leftarrow [0.4, 0.2, 0.3] + 0.01 \times 4.345 \times [0.95, 0.10, 0.85] \tag{16}$$

$$= [0.4413, 0.2043, 0.3369] \tag{17}$$

$$\mathbf{h}_1 \leftarrow [0.95, 0.10, 0.85] + 0.01 \times 4.345 \times [0.4, 0.2, 0.3] \tag{18}$$

$$= [0.9674, 0.1087, 0.8631] \tag{19}$$

Pop Quiz 3: SGD Understanding

Quick Check

A user gives a rating of 2 to a movie, but our model predicts 4.5.

Pop Quiz 3: SGD Understanding

Quick Check

A user gives a rating of 2 to a movie, but our model predicts 4.5.

1. What is the error e_{ij}?

Pop Quiz 3: SGD Understanding

Quick Check

A user gives a rating of 2 to a movie, but our model predicts 4.5.

1. What is the error e_{ij}?
2. Should we increase or decrease the user-movie similarity?

Pop Quiz 3: SGD Understanding

Quick Check

A user gives a rating of 2 to a movie, but our model predicts 4.5.

1. What is the error e_{ij}?
2. Should we increase or decrease the user-movie similarity?
3. If $\alpha = 0.1$, $\mathbf{w}_i = [0.8, 0.3]$, $\mathbf{h}_j = [0.6, 0.9]$, what are the updates?

Pop Quiz 3: SGD Understanding

Quick Check

A user gives a rating of 2 to a movie, but our model predicts 4.5.

1. What is the error e_{ij}?
2. Should we increase or decrease the user-movie similarity?
3. If $\alpha = 0.1$, $\mathbf{w}_i = [0.8, 0.3]$, $\mathbf{h}_j = [0.6, 0.9]$, what are the updates?

Pop Quiz 3: SGD Understanding

Quick Check

A user gives a rating of 2 to a movie, but our model predicts 4.5.

1. What is the error e_{ij}?
2. Should we increase or decrease the user-movie similarity?
3. If $\alpha = 0.1$, $\mathbf{w}_i = [0.8, 0.3]$, $\mathbf{h}_j = [0.6, 0.9]$, what are the updates?

Answers:

1. $e_{ij} = 2 - 4.5 = -2.5$
2. Decrease similarity (negative error)
3. $\mathbf{w}_i \leftarrow [0.8, 0.3] + 0.1 \times (-2.5) \times [0.6, 0.9] = [0.65, 0.075]$
4. $\mathbf{h}_j \leftarrow [0.6, 0.9] + 0.1 \times (-2.5) \times [0.8, 0.3] = [0.4, 0.825]$

Algorithm Comparison and Practical Considerations

ALS vs SGD: Head-to-Head Comparison

Aspect	**ALS**	**SGD**
Updates	Alternating	Simultaneous
Convergence	Faster, more stable	Slower, can oscillate
Parallelization	Excellent	Limited
Memory	Higher	Lower
Implementation	Complex	Simple
Hyperparameters	Few (rank r)	Many (α, schedule)
Scalability	Very good	Good

ALS vs SGD: Head-to-Head Comparison

Aspect	**ALS**	**SGD**
Updates	Alternating	Simultaneous
Convergence	Faster, more stable	Slower, can oscillate
Parallelization	Excellent	Limited
Memory	Higher	Lower
Implementation	Complex	Simple
Hyperparameters	Few (rank r)	Many (α, schedule)
Scalability	Very good	Good

When to Use Which?

- **ALS:** Large-scale, production systems (Spark, distributed)

ALS vs SGD: Head-to-Head Comparison

Aspect	**ALS**	**SGD**
Updates	Alternating	Simultaneous
Convergence	Faster, more stable	Slower, can oscillate
Parallelization	Excellent	Limited
Memory	Higher	Lower
Implementation	Complex	Simple
Hyperparameters	Few (rank r)	Many (α, schedule)
Scalability	Very good	Good

When to Use Which?

- **ALS:** Large-scale, production systems (Spark, distributed)
- **SGD:** Online learning, real-time updates, research

Advanced Practical Considerations

Regularization: Prevent overfitting

$$\text{minimize}_{\mathbf{W},\mathbf{H}} \sum_{(i,j)\in\Omega} (a_{ij} - \mathbf{w}_i^T \mathbf{h}_j)^2 + \lambda(\|\mathbf{W}\|_F^2 + \|\mathbf{H}\|_F^2)$$

Advanced Practical Considerations

Regularization: Prevent overfitting

$$\text{minimize}_{\mathbf{W},\mathbf{H}} \sum_{(i,j)\in\Omega} (a_{ij} - \mathbf{w}_i^T \mathbf{h}_j)^2 + \lambda(\|\mathbf{W}\|_F^2 + \|\mathbf{H}\|_F^2)$$

Bias Terms: Account for global, user, and item biases

$$\hat{a}_{ij} = \mu + b_i + b_j + \mathbf{w}_i^T \mathbf{h}_j$$

Advanced Practical Considerations

Regularization: Prevent overfitting

$$\text{minimize}_{\mathbf{W},\mathbf{H}} \sum_{(i,j)\in\Omega} (a_{ij} - \mathbf{w}_i^T \mathbf{h}_j)^2 + \lambda(\|\mathbf{W}\|_F^2 + \|\mathbf{H}\|_F^2)$$

Bias Terms: Account for global, user, and item biases

$$\hat{a}_{ij} = \mu + b_i + b_j + \mathbf{w}_i^T \mathbf{h}_j$$

Implicit Feedback: Binary observations (clicks, views)

$$\text{Confidence: } c_{ij} = 1 + \alpha \cdot \text{frequency}_{ij}$$

Advanced Practical Considerations

Regularization: Prevent overfitting

$$\text{minimize}_{\mathbf{W},\mathbf{H}} \sum_{(i,j)\in\Omega} (a_{ij} - \mathbf{w}_i^T \mathbf{h}_j)^2 + \lambda(\|\mathbf{W}\|_F^2 + \|\mathbf{H}\|_F^2)$$

Bias Terms: Account for global, user, and item biases

$$\hat{a}_{ij} = \mu + b_i + b_j + \mathbf{w}_i^T \mathbf{h}_j$$

Implicit Feedback: Binary observations (clicks, views)

$$\text{Confidence: } c_{ij} = 1 + \alpha \cdot \text{frequency}_{ij}$$

Cold Start Problem: New users/items with no ratings

- Content-based features

Advanced Practical Considerations

Regularization: Prevent overfitting

$$\text{minimize}_{\mathbf{W},\mathbf{H}} \sum_{(i,j)\in\Omega} (a_{ij} - \mathbf{w}_i^T \mathbf{h}_j)^2 + \lambda(\|\mathbf{W}\|_F^2 + \|\mathbf{H}\|_F^2)$$

Bias Terms: Account for global, user, and item biases

$$\hat{a}_{ij} = \mu + b_i + b_j + \mathbf{w}_i^T \mathbf{h}_j$$

Implicit Feedback: Binary observations (clicks, views)

$$\text{Confidence: } c_{ij} = 1 + \alpha \cdot \text{frequency}_{ij}$$

Cold Start Problem: New users/items with no ratings

- Content-based features
- Demographic information

Advanced Practical Considerations

Regularization: Prevent overfitting

$$\text{minimize}_{\mathbf{W},\mathbf{H}} \sum_{(i,j)\in\Omega} (a_{ij} - \mathbf{w}_i^T \mathbf{h}_j)^2 + \lambda(\|\mathbf{W}\|_F^2 + \|\mathbf{H}\|_F^2)$$

Bias Terms: Account for global, user, and item biases

$$\hat{a}_{ij} = \mu + b_i + b_j + \mathbf{w}_i^T \mathbf{h}_j$$

Implicit Feedback: Binary observations (clicks, views)

$$\text{Confidence: } c_{ij} = 1 + \alpha \cdot \text{frequency}_{ij}$$

Cold Start Problem: New users/items with no ratings

- Content-based features
- Demographic information
- Hybrid approaches

Hands-On Understanding

Let's Build Intuition: Small Example

Our 3×3 rating matrix:

$$\mathbf{A} = \begin{bmatrix} 5 & ? & 2 \\ 4 & 4 & ? \\ ? & 5 & 1 \end{bmatrix}$$

Let's Build Intuition: Small Example

Our 3×3 rating matrix:

$$\mathbf{A} = \begin{bmatrix} 5 & ? & 2 \\ 4 & 4 & ? \\ ? & 5 & 1 \end{bmatrix}$$

Goal: Find $\mathbf{W} \in \mathbb{R}^{3\times 2}$ and $\mathbf{H} \in \mathbb{R}^{2\times 3}$ such that:

$$\mathbf{A} \approx \mathbf{W}\mathbf{H} = \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \\ w_{31} & w_{32} \end{bmatrix} \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \end{bmatrix}$$

Let's Build Intuition: Small Example

Our 3×3 rating matrix:

$$\mathbf{A} = \begin{bmatrix} 5 & ? & 2 \\ 4 & 4 & ? \\ ? & 5 & 1 \end{bmatrix}$$

Goal: Find $\mathbf{W} \in \mathbb{R}^{3\times 2}$ and $\mathbf{H} \in \mathbb{R}^{2\times 3}$ such that:

$$\mathbf{A} \approx \mathbf{WH} = \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \\ w_{31} & w_{32} \end{bmatrix} \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \end{bmatrix}$$

Constraint: Only minimize error on observed entries!

Step-by-Step ALS Solution

Iteration 1: Initialize randomly

$$\mathbf{W}^{(0)} = \begin{bmatrix} 0.5 & 0.3 \\ 0.4 & 0.6 \\ 0.2 & 0.8 \end{bmatrix}, \quad \mathbf{H}^{(0)} = \begin{bmatrix} 1.0 & 0.5 & 0.2 \\ 0.3 & 1.2 & 0.8 \end{bmatrix}$$

Step-by-Step ALS Solution

Iteration 1: Initialize randomly

$$\mathbf{W}^{(0)} = \begin{bmatrix} 0.5 & 0.3 \\ 0.4 & 0.6 \\ 0.2 & 0.8 \end{bmatrix}, \quad \mathbf{H}^{(0)} = \begin{bmatrix} 1.0 & 0.5 & 0.2 \\ 0.3 & 1.2 & 0.8 \end{bmatrix}$$

Update User 1: Only use observed ratings (positions 1,3)

$$\mathbf{y}_1 = [5, 2]^T \tag{20}$$

$$\mathbf{X}_1 = \begin{bmatrix} 1.0 & 0.3 \\ 0.2 & 0.8 \end{bmatrix} \text{ (columns 1,3 of } \mathbf{H}^{(0)\,T}\text{)} \tag{21}$$

Step-by-Step ALS Solution

Iteration 1: Initialize randomly

$$\mathbf{W}^{(0)} = \begin{bmatrix} 0.5 & 0.3 \\ 0.4 & 0.6 \\ 0.2 & 0.8 \end{bmatrix}, \quad \mathbf{H}^{(0)} = \begin{bmatrix} 1.0 & 0.5 & 0.2 \\ 0.3 & 1.2 & 0.8 \end{bmatrix}$$

Update User 1: Only use observed ratings (positions 1,3)

$$\mathbf{y}_1 = [5, 2]^T \tag{20}$$

$$\mathbf{X}_1 = \begin{bmatrix} 1.0 & 0.3 \\ 0.2 & 0.8 \end{bmatrix} \text{ (columns 1,3 of } \mathbf{H}^{(0)\,T}\text{)} \tag{21}$$

Solve: $\mathbf{w}_1^{(1)} = (\mathbf{X}_1^T \mathbf{X}_1)^{-1} \mathbf{X}_1^T \mathbf{y}_1$
Continue for all users and movies...

Pop Quiz 4: Final Challenge

Master Check

You're Netflix's lead ML engineer. You have:

- 200M users, 15K movies
- 20B ratings (0.67% filled)
- Need real-time recommendations
- New users/movies arrive daily

Pop Quiz 4: Final Challenge

Master Check

You're Netflix's lead ML engineer. You have:

- 200M users, 15K movies
- 20B ratings (0.67% filled)
- Need real-time recommendations
- New users/movies arrive daily

Design your recommendation system:

1. Which algorithm: ALS or SGD? Why?

Pop Quiz 4: Final Challenge

Master Check

You're Netflix's lead ML engineer. You have:

- 200M users, 15K movies
- 20B ratings (0.67% filled)
- Need real-time recommendations
- New users/movies arrive daily

Design your recommendation system:

1. Which algorithm: ALS or SGD? Why?
2. What rank r would you choose?

Pop Quiz 4: Final Challenge

Master Check

You're Netflix's lead ML engineer. You have:

- 200M users, 15K movies
- 20B ratings (0.67% filled)
- Need real-time recommendations
- New users/movies arrive daily

Design your recommendation system:

1. Which algorithm: ALS or SGD? Why?
2. What rank r would you choose?
3. How to handle new users?

Pop Quiz 4: Final Challenge

Master Check

You're Netflix's lead ML engineer. You have:

- 200M users, 15K movies
- 20B ratings (0.67% filled)
- Need real-time recommendations
- New users/movies arrive daily

Design your recommendation system:

1. Which algorithm: ALS or SGD? Why?
2. What rank r would you choose?
3. How to handle new users?
4. How to handle the scale?

Pop Quiz 4: Final Challenge

Master Check

You're Netflix's lead ML engineer. You have:

- 200M users, 15K movies
- 20B ratings (0.67% filled)
- Need real-time recommendations
- New users/movies arrive daily

Design your recommendation system:

1. Which algorithm: ALS or SGD? Why?
2. What rank r would you choose?
3. How to handle new users?
4. How to handle the scale?

Pop Quiz 4: Final Challenge

Master Check

You're Netflix's lead ML engineer. You have:

- 200M users, 15K movies
- 20B ratings (0.67% filled)
- Need real-time recommendations
- New users/movies arrive daily

Design your recommendation system:

1. Which algorithm: ALS or SGD? Why?
2. What rank r would you choose?
3. How to handle new users?
4. How to handle the scale?

Suggested Solution:

Summary and Key Takeaways

Key Insights Summary

1. **Sparsity** $\Rightarrow$ **Factorization**: Sparse rating matrices can be approximated by low-rank factorizations

Key Insights Summary

1. **Sparsity** $\Rightarrow$ **Factorization**: Sparse rating matrices can be approximated by low-rank factorizations
2. **Latent Features**: Users and items are characterized by latent factors (not manually defined!)

Key Insights Summary

1. **Sparsity $\Rightarrow$ Factorization**: Sparse rating matrices can be approximated by low-rank factorizations
2. **Latent Features**: Users and items are characterized by latent factors (not manually defined!)
3. **Bilinear Problem**: Non-convex jointly, but convex individually $\rightarrow$ Alternating optimization works well

Key Insights Summary

1. **Sparsity $\Rightarrow$ Factorization**: Sparse rating matrices can be approximated by low-rank factorizations
2. **Latent Features**: Users and items are characterized by latent factors (not manually defined!)
3. **Bilinear Problem**: Non-convex jointly, but convex individually $\rightarrow$ Alternating optimization works well
4. **Scale Matters**: Algorithm choice depends on data size and computational constraints

Key Insights Summary

1. **Sparsity $\Rightarrow$ Factorization**: Sparse rating matrices can be approximated by low-rank factorizations
2. **Latent Features**: Users and items are characterized by latent factors (not manually defined!)
3. **Bilinear Problem**: Non-convex jointly, but convex individually $\rightarrow$ Alternating optimization works well
4. **Scale Matters**: Algorithm choice depends on data size and computational constraints
5. **Real-World Complexity**: Regularization, bias terms, cold start, implicit feedback all matter

Key Insights Summary

1. **Sparsity $\Rightarrow$ Factorization**: Sparse rating matrices can be approximated by low-rank factorizations
2. **Latent Features**: Users and items are characterized by latent factors (not manually defined!)
3. **Bilinear Problem**: Non-convex jointly, but convex individually $\rightarrow$ Alternating optimization works well
4. **Scale Matters**: Algorithm choice depends on data size and computational constraints
5. **Real-World Complexity**: Regularization, bias terms, cold start, implicit feedback all matter

Key Insights Summary

1. **Sparsity $\Rightarrow$ Factorization**: Sparse rating matrices can be approximated by low-rank factorizations
2. **Latent Features**: Users and items are characterized by latent factors (not manually defined!)
3. **Bilinear Problem**: Non-convex jointly, but convex individually $\rightarrow$ Alternating optimization works well
4. **Scale Matters**: Algorithm choice depends on data size and computational constraints
5. **Real-World Complexity**: Regularization, bias terms, cold start, implicit feedback all matter

The Mathematical Beauty:

$$\boxed{\text{Collaborative Filtering} = \text{Matrix Factorization} = \text{Dimensionality Reduct}}$$

Extensions and Advanced Topics

Beyond Basic Matrix Factorization:

Extensions and Advanced Topics

Beyond Basic Matrix Factorization:

- **Non-negative Matrix Factorization (NMF)**: Interpretable factors

Extensions and Advanced Topics

Beyond Basic Matrix Factorization:

- **Non-negative Matrix Factorization (NMF)**: Interpretable factors
- **Deep Matrix Factorization**: Neural networks for non-linear patterns

Extensions and Advanced Topics

Beyond Basic Matrix Factorization:

- **Non-negative Matrix Factorization (NMF)**: Interpretable factors
- **Deep Matrix Factorization**: Neural networks for non-linear patterns
- **Factorization Machines**: Handle multi-way interactions

Extensions and Advanced Topics

Beyond Basic Matrix Factorization:

- **Non-negative Matrix Factorization (NMF)**: Interpretable factors
- **Deep Matrix Factorization**: Neural networks for non-linear patterns
- **Factorization Machines**: Handle multi-way interactions
- **Variational Autoencoders**: Probabilistic approach to recommendations

Extensions and Advanced Topics

Beyond Basic Matrix Factorization:

- **Non-negative Matrix Factorization (NMF)**: Interpretable factors
- **Deep Matrix Factorization**: Neural networks for non-linear patterns
- **Factorization Machines**: Handle multi-way interactions
- **Variational Autoencoders**: Probabilistic approach to recommendations
- **Graph Neural Networks**: Leverage user-item interaction graphs

Extensions and Advanced Topics

Beyond Basic Matrix Factorization:

- **Non-negative Matrix Factorization (NMF)**: Interpretable factors
- **Deep Matrix Factorization**: Neural networks for non-linear patterns
- **Factorization Machines**: Handle multi-way interactions
- **Variational Autoencoders**: Probabilistic approach to recommendations
- **Graph Neural Networks**: Leverage user-item interaction graphs
- **Multi-armed Bandits**: Exploration vs exploitation in recommendations

Extensions and Advanced Topics

Beyond Basic Matrix Factorization:

- **Non-negative Matrix Factorization (NMF)**: Interpretable factors
- **Deep Matrix Factorization**: Neural networks for non-linear patterns
- **Factorization Machines**: Handle multi-way interactions
- **Variational Autoencoders**: Probabilistic approach to recommendations
- **Graph Neural Networks**: Leverage user-item interaction graphs
- **Multi-armed Bandits**: Exploration vs exploitation in recommendations

Extensions and Advanced Topics

Beyond Basic Matrix Factorization:

- **Non-negative Matrix Factorization (NMF)**: Interpretable factors
- **Deep Matrix Factorization**: Neural networks for non-linear patterns
- **Factorization Machines**: Handle multi-way interactions
- **Variational Autoencoders**: Probabilistic approach to recommendations
- **Graph Neural Networks**: Leverage user-item interaction graphs
- **Multi-armed Bandits**: Exploration vs exploitation in recommendations

Applications Beyond Movies:

- E-commerce (Amazon, eBay)

Extensions and Advanced Topics

Beyond Basic Matrix Factorization:

- **Non-negative Matrix Factorization (NMF)**: Interpretable factors
- **Deep Matrix Factorization**: Neural networks for non-linear patterns
- **Factorization Machines**: Handle multi-way interactions
- **Variational Autoencoders**: Probabilistic approach to recommendations
- **Graph Neural Networks**: Leverage user-item interaction graphs
- **Multi-armed Bandits**: Exploration vs exploitation in recommendations

Applications Beyond Movies:

- E-commerce (Amazon, eBay)
- Music streaming (Spotify, Apple Music)

Extensions and Advanced Topics

Beyond Basic Matrix Factorization:

- **Non-negative Matrix Factorization (NMF)**: Interpretable factors
- **Deep Matrix Factorization**: Neural networks for non-linear patterns
- **Factorization Machines**: Handle multi-way interactions
- **Variational Autoencoders**: Probabilistic approach to recommendations
- **Graph Neural Networks**: Leverage user-item interaction graphs
- **Multi-armed Bandits**: Exploration vs exploitation in recommendations

Applications Beyond Movies:

- E-commerce (Amazon, eBay)
- Music streaming (Spotify, Apple Music)
- Social media (Facebook, LinkedIn)

Extensions and Advanced Topics

Beyond Basic Matrix Factorization:

- **Non-negative Matrix Factorization (NMF)**: Interpretable factors
- **Deep Matrix Factorization**: Neural networks for non-linear patterns
- **Factorization Machines**: Handle multi-way interactions
- **Variational Autoencoders**: Probabilistic approach to recommendations
- **Graph Neural Networks**: Leverage user-item interaction graphs
- **Multi-armed Bandits**: Exploration vs exploitation in recommendations

Applications Beyond Movies:

- E-commerce (Amazon, eBay)
- Music streaming (Spotify, Apple Music)
- Social media (Facebook, LinkedIn)

Final Pop Quiz: Comprehensive Understanding

Mastery Test

True or False? Explain your reasoning:

Final Pop Quiz: Comprehensive Understanding

Mastery Test

True or False? Explain your reasoning:

1. Matrix factorization can only work with explicit ratings

Final Pop Quiz: Comprehensive Understanding

Mastery Test

True or False? Explain your reasoning:

1. Matrix factorization can only work with explicit ratings
2. ALS always converges to the global optimum

Final Pop Quiz: Comprehensive Understanding

Mastery Test

True or False? Explain your reasoning:

1. Matrix factorization can only work with explicit ratings
2. ALS always converges to the global optimum
3. A rank-1 factorization means all users have identical preferences

Final Pop Quiz: Comprehensive Understanding

Mastery Test

True or False? Explain your reasoning:

1. Matrix factorization can only work with explicit ratings
2. ALS always converges to the global optimum
3. A rank-1 factorization means all users have identical preferences
4. Adding regularization always improves recommendations

Final Pop Quiz: Comprehensive Understanding

Mastery Test

True or False? Explain your reasoning:

1. Matrix factorization can only work with explicit ratings
2. ALS always converges to the global optimum
3. A rank-1 factorization means all users have identical preferences
4. Adding regularization always improves recommendations
5. SGD is better than ALS for all applications

Final Pop Quiz: Comprehensive Understanding

Mastery Test

True or False? Explain your reasoning:

1. Matrix factorization can only work with explicit ratings
2. ALS always converges to the global optimum
3. A rank-1 factorization means all users have identical preferences
4. Adding regularization always improves recommendations
5. SGD is better than ALS for all applications

Final Pop Quiz: Comprehensive Understanding

Mastery Test

True or False? Explain your reasoning:

1. Matrix factorization can only work with explicit ratings
2. ALS always converges to the global optimum
3. A rank-1 factorization means all users have identical preferences
4. Adding regularization always improves recommendations
5. SGD is better than ALS for all applications

Answers:

1. **False** - Works with implicit feedback too (clicks, views)
2. **False** - Converges to local optimum (problem is