# Recommender Systems Lecture —3

## Matrix Factorization in Collaborative Filtering

Matrix Factorization is a technique used in collaborative filtering to predict user preferences for items (e.g., movies, products) by decomposing the user-item interaction matrix into lower-dimensional matrices representing latent user and item features.

**Equation for Matrix Factorization:**

Given a user-item ratings matrix $(R)$ of dimensions $(n \times m)$, where $(n)$ is the number of users and $(m)$ is the number of items, Matrix Factorization aims to find two lower-dimensional matrices, $(P)$ (of dimensions $(n \times k)$)and $(Q)$ (of dimensions $(k \times m)$), such that their product approximates $(R)$:

- $[R \approx P \times Q^T]$
- 
- Where:
    - $(R)$ is the original user-item ratings matrix.
    - $(P)$ is the user matrix, representing users in a $(k)$-dimensional latent feature space.
    - $(Q)$ is the item matrix, also in a $(k)$-dimensional latent space, and $(Q^T)$ denotes the transpose of $(Q)$.
    - $(k)$ is the number of latent features.

**How It Works:**

- **Latent Features:** Identifies underlying attributes that determine how a user rates an item.
- **User and Item Matrices (P and Q):** Represent users and items in terms of latent features. The dot product of P and Q approximates the original user-item ratings matrix.
- **Sparse Matrix Handling:** Predicts missing ratings by filling in the sparse matrix with the dot product of P and Q, making recommendations for unseen items.

**Process:**

1. **Initialization:** Start with random values for P and Q matrices.
2. **Gradient Descent:** Iteratively adjust P and Q to minimize the difference between actual ratings and the product of P and Q, aiming for a local minimum.

**Collaborative Filtering vs. Content-Based Systems:**

- **Domain Knowledge:** CF doesn't require domain knowledge, as it learns from user behavior.
- **Serendipity:** CF can suggest unexpected items a user may like based on similarities with other users.
- **Context-Independence:** Works without needing specific item features.

**Advantages:**

- **Automatic Feature Learning**: Discovers user and item features without explicit programming.
- **Discover New Interests:** Can recommend items a user has not explicitly expressed interest in.
- **No Need for Item Metadata:** Operates purely on user-item interaction data.

**Disadvantages:**

- **Cold Start Problem:** Struggles to recommend for new users or items without sufficient data.
- **Computational Cost:** The process of learning P and Q matrices can be resource-intensive.
- **Unique Taste Challenge:** This may not perform well for users with very unique preferences.

Matrix Factorization has become a fundamental approach in building sophisticated recommender systems, enabling personalized suggestions by uncovering the complex patterns in user-item interactions.

## Principal Component Analysis (PCA)

PCA is essentially a form of Matrix Factorization (MF) tailored for dimensionality reduction and feature extraction. It decomposes the data matrix into components that capture the most variance or information about the data.

**Covariance Matrix Calculation:**

- Calculated using $S = \frac{1}{n} X^T X$, where $X$ is the standardized data matrix. The covariance matrix, $S$, is square (dimensions $d \times d$) and symmetric.

**Eigen Decomposition**:

- **Eigen Decomposition:** Before PCA, this method was used to decompose matrices.
- **Decomposition of Covariance Matrix**: $S = Q \Lambda Q^T$
  - $Q$: Matrix with columns as $d$ eigenvectors.
  - $\Lambda$: Diagonal matrix with eigenvalues.
  - $Q^T$: Transpose of $Q$, rows as eigenvectors' transpose.

- **Properties:** The singular values in $\Lambda$ satisfy the condition $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_d$.

**PCA vs. MF:**
- **PCA:** A specialized type of MF focused on decomposing the covariance matrix to find the principal components (eigenvectors) that maximize variance.
- **Constraints in PCA:**
  - **Orthogonality:** Eigenvectors (columns of $Q$) are perpendicular to each other.
  - **Diagonalization:** $\Lambda$ is a diagonal matrix with eigenvalues.

## Singular Value Decomposition (SVD)

SVD is a powerful matrix decomposition technique that breaks down a data matrix into three distinct matrices, revealing the underlying structure of the data in a way that's optimal for many applications, including dimensionality reduction and feature extraction.

**SVD Formulation:**
- Decomposes the data matrix $X$ into $X = U\Sigma V^T$, where:
  - $U$: Contains left singular vectors (eigenvectors of $XX^T$).
  - $\Sigma$: A diagonal matrix with singular values (square roots of eigenvalues from $U$ or $V$).
  - $V^T$: Contains right singular vectors (eigenvectors of $X^TX$), transposed.

**Characteristics:**
- **Dimensions:** $\Sigma$ is rectangular, aligning with the condition $n > d$.
- **Eigenvectors vs. Singular Vectors:** Eigenvectors are for square matrices, while singular vectors extend to rectangular matrices.
- **Relation to PCA:** $\Sigma$ in SVD corresponds to the eigenvalues in PCA, making SVD and PCA closely related.

**Finding Singular Values:**
- Singular values ($\sigma$) are derived from the eigenvalues ($\lambda$) of either $XX^T$ or $X^TX$, with the relationship $\sigma = \sqrt{\lambda}$.
- The singular values in $\Sigma$ are ordered by magnitude.

**Importance of SVD:**
- **Versatility:** Applicable to any matrix, not limited to square or symmetric matrices, unlike PCA.
- **Truncated SVD:** A variation that selects a subset of singular values, useful for reducing dimensionality while preserving as much information as possible.
- **Feature Engineering:** Can uncover latent features that capture the most significant structure in the data.

**Applications**:

- **Recommender Systems:** Helps in capturing preferences and similarities among users and items.
- **Natural Language Processing (NLP):** For semantic analysis and topic modeling.
- **Image Compression:** Reduces storage without significantly compromising quality.

SVD is integral to understanding data structures and patterns, offering a mathematical foundation for many modern machine learning and data processing tasks. Its ability to decompose any matrix, regardless of shape, and reveal its singular values makes it indispensable for sophisticated data analysis and system design.