🎃 Funk SVD Algorithm for Recommender Systems 👻

Farhad Vadiee

Centre for the Science of Learning & Technology (SLATE)

October 31, 2024, YoungAI

Introduction to Recommender Systems

- Recommender systems are algorithms that suggest items to users based on their preferences.

Introduction to Recommender Systems

- Recommender systems are algorithms that suggest items to users based on their preferences.
- Commonly used in platforms like Netflix, Amazon, and Spotify.

Introduction to Recommender Systems

- Recommender systems are algorithms that suggest items to users based on their preferences.
- Commonly used in platforms like Netflix, Amazon, and Spotify.
- Types of recommender systems:
  - **Content-Based Filtering**: Recommends items similar to those the user has liked in the past.
  - **Collaborative Filtering**: Recommends items based on the preferences of similar users.
  - **Hybrid Methods**: Combines multiple recommendation techniques to improve accuracy.

💀 Collaborative Filtering

- Collaborative Filtering is one of the most popular techniques for building recommender systems.
- **User-User Collaborative Filtering**: Finds similar users to make recommendations.
- **Item-Item Collaborative Filtering**: Finds similar items based on user interactions.
- Challenges:
  - **Cold Start Problem**: Difficult to recommend items to new users or recommend new items.
  - **Sparsity**: User-item matrices are often sparse, with many missing ratings.

Introduction to Funk SVD

- Funk SVD is a matrix factorization technique used in recommendation systems.
- It decomposes the user-item interaction matrix into latent factors.
- Developed as part of the Netflix Prize competition.

Matrix Factorization

- Goal: Represent the user-item matrix R as a product of two lower-dimensional matrices.
- User latent factor matrix: $P \in \mathbb{R}^{m \times k}$.
- Item latent factor matrix: $Q \in \mathbb{R}^{n \times k}$.
- $$R \simeq PQ^T$$

Matrix Factorization

- Goal: Represent the user-item matrix R as a product of two lower-dimensional matrices.
- User latent factor matrix: $P \in \mathbb{R}^{m \times k}$.
- Item latent factor matrix: $Q \in \mathbb{R}^{n \times k}$.
- $$R \simeq PQ^T$$
- Predicted rating:
$$r_{ui} = P_u^T Q_i$$

Visualizing User-Item Matrix R

- Consider the following user-item rating matrix R:

$$R = \begin{bmatrix} 5 & ? & 3 \\ ? & 4 & ? \\ 1 & ? & 2 \end{bmatrix}$$

- Goal: Decompose R into two latent matrices P and Q.

$$P = \begin{bmatrix} p_{11} & p_{12} \\ p_{21} & p_{22} \\ p_{31} & p_{32} \end{bmatrix}, \quad Q^T = \begin{bmatrix} q_{11} & q_{12} & q_{13} \\ q_{21} & q_{22} & q_{23} \end{bmatrix}$$

💪 Training Process

- Define a loss function
- Initialize latent factor matrices P and Q with small random values.
- Use an optimizer to minimize the loss function
- Hyperparameters:

Loss Function

- The objective is to minimize the loss function:

$$L = \sum_{(u,i)\in \text{train}} (r_{ui} - P_u^T Q_i)^2 + \lambda(\|P_u\|^2 + \|Q_i\|^2)$$

- The first term represents the squared error between the actual rating r_{ui} and the predicted rating $P_u^T Q_i$.
- The second term is a regularization term to prevent overfitting.
- λ: Regularization parameter.

🔁 SGD Update Step Visualization

- Stochastic Gradient Descent (SGD) is an iterative optimization algorithm used to minimize a loss function. In each iteration, it updates parameters based on the gradient of the loss function with respect to those parameters.
- Example: Updating latent factors for user 1 and item 3.
- Predicted rating:
$$\hat{r}_{13} = P_1^T Q_3$$
- Error:
$$e_{13} = r_{13} - \hat{r}_{13}$$

Updating Latent Factors

- Stochastic Gradient Descent (SGD) Update:

$$P_1 \leftarrow P_1 + \eta \cdot (e_{13} \cdot Q_3 - \lambda \cdot P_1)$$
$$Q_3 \leftarrow Q_3 + \eta \cdot (e_{13} \cdot P_1 - \lambda \cdot Q_3)$$

- η: Learning rate.
- λ: Regularization term.

💪 Training Process

- Initialize latent factor matrices P and Q with small random values:
$$P \sim \mathcal{N}(0, \frac{1}{k}), \quad Q \sim \mathcal{N}(0, \frac{1}{k})$$
- Use Stochastic Gradient Descent (SGD) to minimize the error over n epochs.
- Hyperparameters:
  - Learning rate: $\eta = 0.005$
  - Regularization term: $\lambda = 0.2$
  - Number of latent factors: $k = 20$
- Training and test split: 80% training, 20

🏆 Advantages of Funk SVD Compared to Other ML Models

- **Scalability**: Funk SVD can handle very large datasets efficiently, making it suitable for recommendation systems like Netflix.
- **Latent Factor Discovery**: It captures latent features of users and items, such as genres or preferences, which are not explicitly available.
- **Sparsity Handling**: Unlike traditional machine learning models, Funk SVD is well-suited for sparse user-item matrices, where many ratings are missing.
- **Personalization**: Provides personalized recommendations by learning user-specific and item-specific latent factors, which many standard ML models struggle with.
- **Interpretability**: The latent factors can be interpreted to understand user preferences and item attributes, providing insights beyond simple predictions.

Summary

- Funk SVD uses matrix factorization to predict user-item interactions.
- Optimizes latent factors using SGD and regularization.
- Provides accurate recommendations by capturing hidden user preferences.

Questions?

Any Questions?