Task1

References:

https://surprise.readthedocs.io/en/stable/getting_started.html

https://analyticsindiamag.com/singular-value-decomposition-svd-application-recommender-system/

https://blog.csdn.net/Luqiang_Shi/article/details/87885680

SVD, or singular value decomposition is used as the algorithm of the recommender system. It is classical method from linear algebra is getting popular in the field of data science and machine learning. This popularity is because of its application in developing recommender systems. There are a lot of online user-centric applications such as video players, music players, e-commerce applications, etc., where users are recommended with further items to engage with.

SVD is a matrix factorisation technique, which reduces the number of features of a dataset by reducing the space dimension from N-dimension to K-dimension (where K<N). In the context of the recommender system, the SVD is used as a collaborative filtering technique. It uses a matrix structure where each row represents a user, and each column represents an item. The elements of this matrix are the ratings that are given to items by users.

The factorisation of this matrix is done by the singular value decomposition. It finds factors of matrices from the factorisation of a high-level (user-item-rating) matrix. The singular value decomposition is a method of decomposing a matrix into three other matrices as given below:

$$A = USV^T$$

Where A is a m x n utility matrix, U is a m x r orthogonal left singular matrix, which represents the relationship between users and latent factors, S is a r x r diagonal matrix, which describes the strength of each latent factor and V is a r x n diagonal right singular matrix, which indicates the similarity between items and latent factors.

Let each item be represented by a vector X_i and each user is represented by a vector y_u. The expected rating by a user on an item P_{ui} can be given as:

$$P_{ui} = x_i^T \cdot y_u$$

The X_i and y_u can be obtained in a manner that the square error difference between their dot product and the expected rating in the user-item matrix is minimum. It can be expressed as:

$$\text{Min } (x, y) \sum_{(u,i)\in K} \left(r_{ui} - x_i^T y_u\right)^2$$

In order to let the model generalize well and not overfit the training data, a regularization term is added as a penalty to the above formula:

$$\text{Min } (x, y) \sum_{(u,i)\in K} \left(r_{ui} - x_i^T y_u\right)^2 + \lambda\left(\|x_i\|^2 + \|y_u\|^2\right)$$

In order to reduce the error between the value predicted by the model and the actual value, the algorithm uses a bias term. Let for a user-item pair (u, i), μ is the average rating of all items, bi is the average rating of item i minus μ and bu is the average rating given by user u minus μ, the final equation after adding the regularization term and bias can be given as:

$$\text{Min } (x, y, b_i, b_u) \sum_{(u,i)=K} \left(r_{ui} - x_i^T y_u - \mu - b_i - b_u\right)^2 \cdot x_i\| \quad ^2 + \|y_u\ \|^2 + b_i^2 + b_u^2\Big)$$

Task2

The collaborative filtering algorithm need to process the whole user-item-rating matrix, which is extremely sparse in practice. Lots of computation resources and time are needed. Similar to PCA, the SVD algorithm can reduce the dimension of the matrix by keeping the main component of the matrix using the covariance matrix. Suppose the diagonal matrix of SVD is Δ, the matrix after dimension reduction is

$$\text{NewData} = \left(\text{Data}^{T} \times U \times \Delta^{-1}\right)^{T}$$

Where U is the left singular matrix.

Therefore, the advantage of SVD is it can process sparse matrix more efficient, and user can decide how much information to preserve any time. And it can reduce the noise, eigher. It is more like a practical implementation of collaborative filtering.