MUSIC RECOMMENDATION SYSTEM (MRS)

Project Leader: Yashi Kesarwani, Reg no: 0000197

Project Faculty Coordinator: Dr. Anirban Lakshman

Table of Contents

Overview

- Music is one of the most popular entertainment media in the digital era. Listening to music in the digital age is easier because of the features on the smartphone that can play music both offline and online.
- Nowadays, the availability of digital music is very abundant compared to the previous era, so to sort out all this digital music is very time consuming and causes information fatigue. Therefore, it is very useful to develop a music recommender system that can search music libraries automatically and suggest songs that are suitable for users.

Introduction

- A R**ecommender System**, or a R**ecommendation System** is a subclass of information filtering system that seeks to predict the "rating" or "preference" a user would give to an item. They are primarily used in commercial applications.
- Recommender systems aim to predict users’ interests and recommend product items that quite likely are interesting for them. They are among the most powerful machine learning systems that online retailers implement in order to drive sales.
- The Recommendation system which aims to predict rating or preference for music as per user's interests and likes is called **Music Recommendation System**.

Objective

The objective of this project is as follows:

- Examine the data, I am working with by performing initial Exploratory Data Analysis (EDA).
- Build a basic content recommender system using the concept of Term Frequency-Inverse Document Frequency (TF-IDF) for pattern matching.
- Build a couple versions of a basic collaborative recommender system using K Nearest Neighbors and Matrix Factorization algorithm.

Why Recommendation System ?

- Companies using recommender systems focus on increasing sales as a result of very personalized offers and an enhanced customer experience. Recommendations typically speed up searches and make it easier for users to access content they’re interested in and surprise them with offers they would have never searched for.
- What is more important is that companies are able to gain and retain customers by sending out emails with links to new offers that meet the recipients’ interests or suggestions of films and TV shows that suit their profiles.

Types of Recommendation System

- A **content-based recommender** recommends based on history of similar items purchased or interacted with in the past. One can attempt to recommend music that is perceptually similar to what the user has previously listened to, by measuring the similarity between audio signals.
- A **collaborative recommender** recommends based on usage trends of similar users. For instance, if Alice and Bob like movies X, Y and Z, and you like movies X and Y, you may be recommended movie Z.

Dataset Description

For Content based recommendation system-

- I used dataset **songdata.csv** which contains name, artist, and lyrics for 57650 songs in English. The data has been acquired from LyricsFreak through scraping.

For Collaborative filter recommendation system

- I am going to use the **Million Song Dataset**, a freely-available collection of audio features and metadata for a million contemporary popular music tracks.
- There are two files- first one contains user_ID, song_ID and listen_count and other contain song ID, title of that song, release, artist name and year and we merge these two files using song_ID.
- In the Dataset, we have- 2000000 observations , 9567 unique songs , 3375 unique artists , 76353 unique users

Methodology

For Content based Recommendation System -

A content-based recommendation algorithm has to perform the following two steps-

- First, extract features out of the content of the song descriptions to create an object representation.
- Second, define a similarity function among these object representations which mimics what human understands as an item-item similarity.

In my case, because I am working with text and words, **Term Frequency-Inverse Document Frequency (TF-IDF)** is used for the matching process.

Methodology

- The **Term frequency** of a word in the current document is solely the number of times it appears to the total number of words in a document.

  TF(word) = Number of times word appears/Total number of words

- **Inverse document frequency** of a term is the measure of how significant that term is in the whole corpus.

  IDF = log(Total number of documents/Number of documents containing the term)

- The algorithm finds the score for TF and IDF for each term in the document. After that, the product of TF and IDF of each word is obtained. This is called the TF-IDF Light of that term
- The higher the TF*IDF score, the stranger is that word in that context and thus, the more important the term.

Methodology

For Collaborative filtering Recommendation System

- Collaborative methods make a prediction on possible preferences using a matrix with ratings on different songs.
- **Interaction matrices** are based on the many entries that include a user-song pair as well as a value that represents the user's rating for that song.
- Doing some exploratory analysis, I discovered that a user listens to a mean number of 26 songs and a median of songs of 16. So, we will have a very sparse interaction matrix. So to make it eaisier, we will select only those users that have listened to atleast 16 songs.
- After a little more exploration, I discovered that a song is listened to by an average of 200 users, with minimum 48 and maximum 8277 users. To make it easier, we select only those songs which have been listened to by at least 200 users.

K Nearest Neighbors Algorithm

- The KNN algorithm is a **supervised**, **non-parametric**, **lazy-learning** method used for both classification and regression.
- This algorithm is based on feature similarity- I can start to see what it can be used for recommendation systems. It basically assumes that similar things are located near to each other. And here **distances** strike again to measure how similar or 'close' two points are.
- The data obtained from interaction matrix will be passed through a latent mapping algorithm, K-nearest neighbors, to determine cosine similarity amongst the user/artist relationships. This will help us determine which artists are most similar as in shortest distance apart within this latent mapping.

Matrix Factorization Algorithm

- **Matrix Factorization** is a powerful way to implement a recommendation system. The idea behind it is to represent users and items in a lower-dimensional latent space. This does not only solve the sparsity issue but also makes the method scalable.
- Matrix Factorization will tell us how much a user is related to a set of features as well as how much a song belongs to a particular set of latent features.
- Among the different matrix factorization techniques, I used the popular **singular value decomposition (SVD).**

Matrix Factorization Algorithm

The interaction matrix can be decomposed uniquely into three matrices; let’s called them **U**, **S**, and **V**.

In terms of my song recommender:

- - **U** is an *n* users x *r* user-latent feature matrix
- - **V** is an *m* songs x *r* song-latent feature matrix
- - **S** is an *r x r non-negative* diagonal matrix containing the singular values of the original matrix.

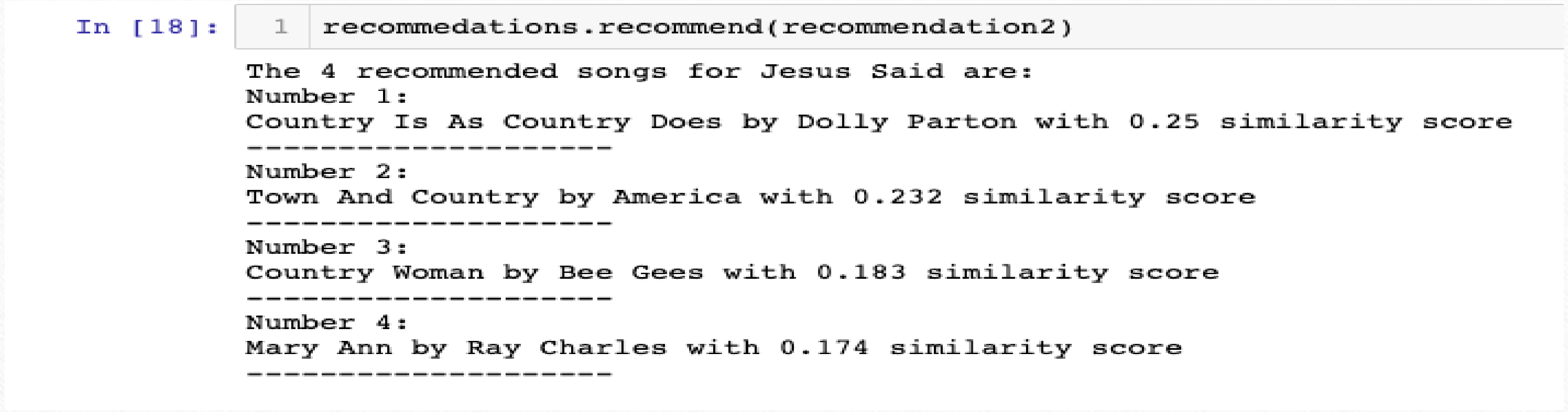
A singular value represents the importance of a specific feature in predicting a user preference. We will implement it using the built-in **SVD** function.

Matrix Factorization Algorithm

- The **GridSearchCV** class will compute accuracy metrics for the SVD algorithm on the combinations of parameters selected, over a cross-validation procedure.
- This method uses K-Fold as the cross-validation technique. After finding the best parameters for the model, I can create my final model. I can use the method fit to train the algorithm on the trainset, and then, the method test to return the predictions obtained from the test set.

Results

- **For content based recommendation system-**



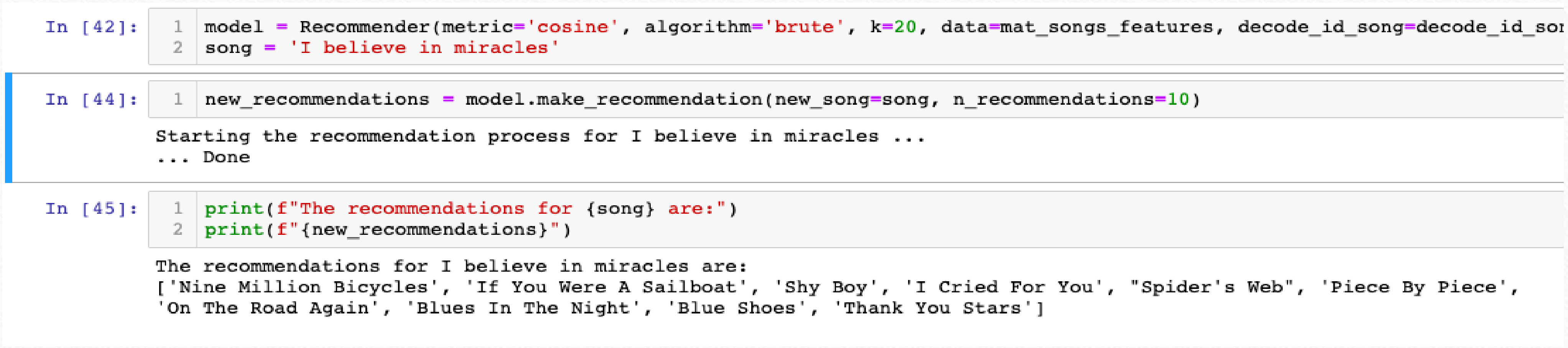
```
In [18]:  recommedations.recommend(recommendation2)

The 4 recommended songs for Jesus Said are:
Number 1:
Country Is As Country Does by Dolly Parton with 0.25 similarity score
--------------------
Number 2:
Town And Country by America with 0.232 similarity score
--------------------
Number 3:
Country Woman by Bee Gees with 0.183 similarity score
--------------------
Number 4:
Mary Ann by Ray Charles with 0.174 similarity score
--------------------
```

One of the biggest limitations of content-based recommendation systems is that the model only learns to recommend items of the same type that the user is already using but it lacks the surprise component of discovering something completely new.

Results

For Collaborative filtering based recommendation system-

- **KNN Algorithm-**



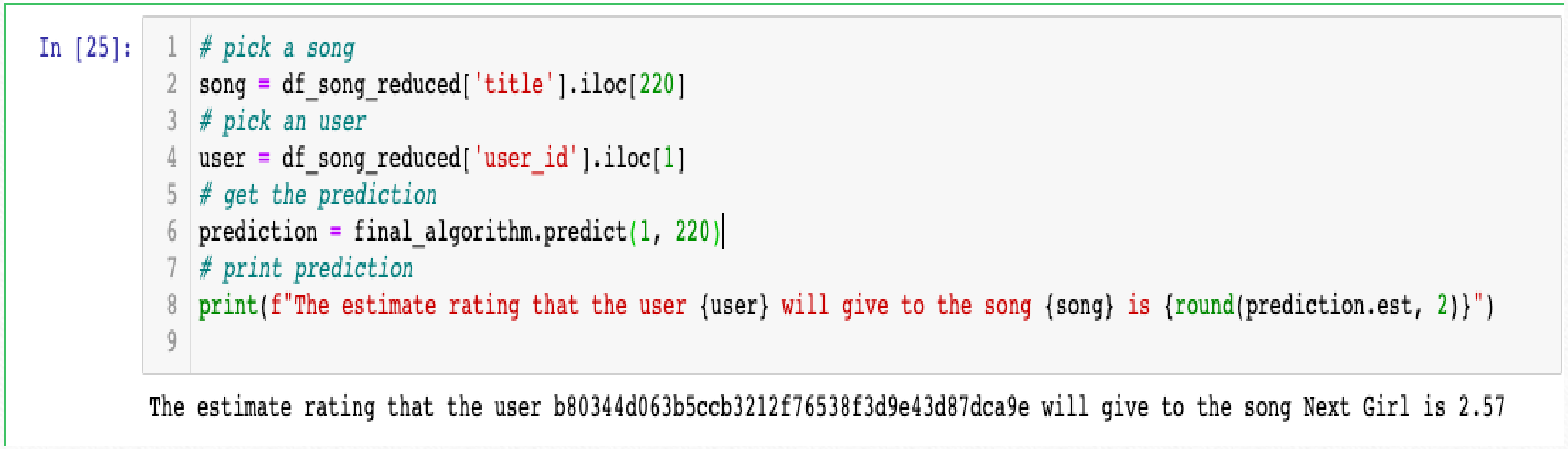
```
In [42]: model = Recommender(metric='cosine', algorithm='brute', k=20, data=mat_songs_features, decode_id_song=decode_id_sor
         song = 'I believe in miracles'

In [44]: new_recommendations = model.make_recommendation(new_song=song, n_recommendations=10)

         Starting the recommendation process for I believe in miracles ...
         ... Done

In [45]: print(f"The recommendations for {song} are:")
         print(f"{new_recommendations}")

         The recommendations for I believe in miracles are:
         ['Nine Million Bicycles', 'If You Were A Sailboat', 'Shy Boy', 'I Cried For You', "Spider's Web", 'Piece By Piece',
         'On The Road Again', 'Blues In The Night', 'Blue Shoes', 'Thank You Stars']
```

- There are some limitations to this method. It doesn't handle sparsity very well, and it's not computational efficient as we have more users or songs.

Results

- **Matrix Factorization Algorithm-**

The RMSE for the model is 2.18



```
In [25]:
# pick a song
song = df_song_reduced['title'].iloc[220]
# pick an user
user = df_song_reduced['user_id'].iloc[1]
# get the prediction
prediction = final_algorithm.predict(1, 220)
# print prediction
print(f"The estimate rating that the user {user} will give to the song {song} is {round(prediction.est, 2)}")

```

```
The estimate rating that the user b80344d063b5ccb3212f76538f3d9e43d87dca9e will give to the song Next Girl is 2.57
```

Conclusion

- I had a great learning experience while doing this project. I have learned about data mining and data cleaning while completing Exploratory Data Analysis (EDA).
- This is a project of my Artificial Intelligence course. I find it is very good as I got a chance to practice theories that I have learnt in the course, to do some implementation and try to get a better understanding of a real artificial intelligence problem: Music Recommender System.

Future Scope

- I made content based and collaborative based recommendation systems. Due to lack of time, I couldn't make hybrid models. Popularity based models are also good at recommendations. So, I’ll try to implement this also to predict top-N songs to the users which are most popular at a given time.
- There is a scope for incorporating the concept of multiple dimensions in music recommender system particularly. The traditional recommender system techniques are not very optimum. This leads a way towards further work and development in building an optimized recommender system which also takes into considerations the feedback and all other constraints related to recommendation.

References

- Kim, H. T., Kim, E., Lee, J. H., & Ahn, C. W. (2010, April). A recommender system based on genetic algorithm for music data. In 2010 2nd International Conference on Computer Engineering and Technology (Vol. 6, pp. V6-414). IEEE.
- S. Chang, A. Abdul, J. Chen and H. Liao, "A personalized music recommendation system using convolutional neural networks approach," 2018 IEEE International Conference on Applied System Invention (ICASI), Chiba, 2018, pp. 47-49.

Thank You!