CERTIFICATE

ACKNOWLEDGEMENT

Summer training has an important role in exposing the real-life situation in an industry. It was great experience for me to work on the training at "**AI-SHALA Pvt Ltd"** through which I could learn how to work in a professional environment.

Now, I would like to thank the people who guided me and have been a constant source of inspiration throughout the tenure of my summer training. I am sincerely grateful To **MR. Anil Sharma and MS. Ayushi Pandey** who rendered me his valuable assistance, constant encouragement and able guidance which made this training possible.

I also wish my deep sense of gratitude to all the faculty members whose guidance and encouragement made my training successfully.

Prashant Kumar

07815002821

B Tech (ECE)

ABSTRACT

Recommendation System belongs to the class of Information Retrieval, Data Mining and Machine Learning. Recommender systems play a major role in today's ecommerce industry. Recommender systems recommend items to users such as books, movies, videos, electronic products and many other products in general. Recommender systems help the users to get personalized recommendations, helps users to take correct decisions in their online transactions, increase sales and redefine the users web browsing experience, retain the customers, enhance their shopping experience. Information overload problem is solved by search engines, but they do not provide personalization of data. Recommendation engines provide personalization. There are different type of recommender systems such as content-based, collaborative filtering, hybrid recommender system, demographic and keyword based recommender system. Variety of algorithms are used by various researchers in each type of recommendation system. Lot of work has been done on this topic, still it is a very favourite topic among data scientists. It also comes under the domain of data Science.

TABLE OF CONTENTS

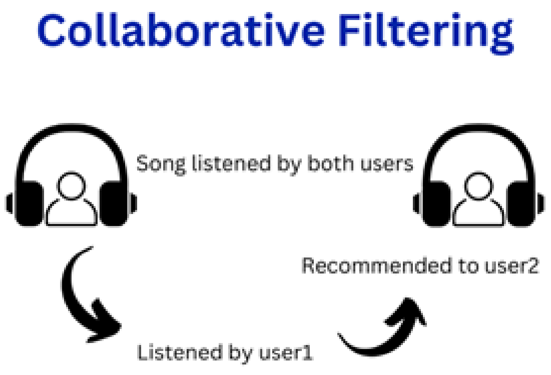
CHAPTER – 1

INTRODUCTION

The first suggestion system was created in 1979. Elaine Rich defined her Grundy library system as follows: it is used to offer books to users after a brief interview in which the user is requested to fill in his first and last name, and then Grundy asks them to define themselves in a few key terms in order to discover their preferences and classify them as a "stereotype." Grundy provides an initial suggestion by providing a summary of the book after the data has been recorded. If the user is unhappy with the option, Grundy asks questions to figure out which part of the book it made a mistake on and then proposes a fresh one. Recommendation systems, which first appeared in the 1990s, have advanced significantly in recent years, particularly with the introduction of Machine Learning and networks. On the one hand, the expanding use of today's digital world, which is characterized by a wealth of data, has enabled us to collect massive user databases. On the other hand, when computing power increased, it became possible to handle these data, particularly using Machine Learning, when human skills were no longer capable of conducting a thorough examination of such a large amount of data. Unlike search engines, which get queries with specific information about what the user wants, a recommendation system does not receive a direct request from the user, but instead must provide them fresh options based on their past behaviors. E-commerce sites that want to sell as many commodities or services as possible to customers (travel, books, etc.) must swiftly recommend appropriate commodities. The purpose of services that provide streaming music and movies is to keep people on their platform for as long as possible. The recurring theme is that appropriate recommendations are required. Recent advancements in this industry have been significant, and these tips are advantageous to both businesses looking to maximize earnings and customers who are no longer overwhelmed by the quantity of options available. Making decisions is therefore made simple, and a good tip saves a lot of time. The Recommender System is a software application and algorithm that provides suggestions for items that a user is most interested in. Recommendations are used in a variety of real-world situations, such as deciding what products to buy, listening to music, or reading the latest news. On the other side, there has been a shift in recorded commodity music, particularly after Apple acquired Beats Music in 2014 8. The music 7 industry's economic model has recently shifted from commodity sales to subscriptions and streaming. In comparison to prior eras, the availability of digital music is now abundant due to the new business model in the music industry. As a result, the importance of a music recommender system for music suppliers cannot be overstated. It is foreseeing.Collaborative filtering makes suggestions based on the collaborative power of the available evaluation by users. It is assumed that if people rate music things similarly or behave similarly, they would rate other music items similarly as well. The sparse evaluation matrix is the major issue in collaborative filtering methods since most users only see a tiny portion of all music libraries, hence most assessments are not decided. Content-based filtering, on the other hand, makes suggestions based on the characteristics of the music pieces. We will see if we can get better recommendations by using real-time data, such as a user's heart rate and the time of day, when making recommendations in this project. The recommendations will be made by a system that employs several machine learning techniques and is accessible via a mobile application. The system uses a smart watch to recognise the user’s heart rate in order to give recommendations of songs according to what kind of music is usually associated with that heart rate and time of day for that specific user. For instance, if a user is out running, the user’s heart rate is probably higher than normal. Many music firms, such as Amazon Music, Wynk Music, and Gaana.com, now use recommender algorithms, and the old technique of selling music has shifted to a cloudbased one. All of their music resources are now available in the cloud, and customers may listen to tracks directly from there. However, the problem is that the cloud system has a large amount of music. As a result, we must categorize all of the songs based on various genres, artists' regions, age groups, and languages, with the primary purpose of categorizing these songs according to the user's preferences. Because users demand a good return on their time and money, we can attract a large number of clients by offering a variety of valuable services that they are interested in. We're using a variety of machine learning methods as well as data mining techniques for this project. We tested a number of algorithms and compared the results to determine the most effective algorithm for our model.

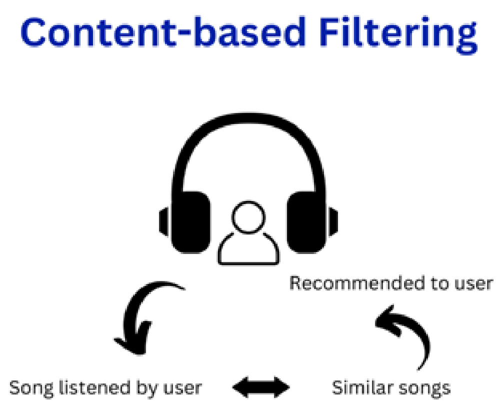
There are various types of methods by which we can establish a recommendation system they are :

1) **Collaborative filtering :**



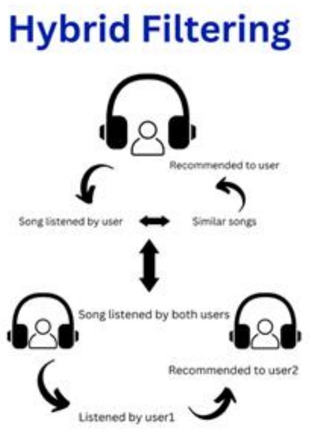
This kind of recommendation is based on an examination of both the behavior of the listeners and the behavior of all other platform users. The basic premise is that other users' opinions may be utilized to make a credible prediction about another user's preferences for an item that they have not yet rated: a user is given recommendations based on other users who share their tastes. Indeed, for years, we've asked our friends, family, and coworkers for recommendations when it came to music, restaurants, movies, and other entertainment. This mechanism is the one that is being attempted to be replicated here. This strategy (based on stars offered by other users) was pioneered by Netflix, but it is now widely used, notably for Spotify's Discover Weekly. Collaborative filtering makes suggestions based on the collaborative power of the available evaluation by users. It is assumed that if various users rate music things similarly or have similar behaviour, they would rate other music items similarly. The sparse evaluation matrix is the major issue in collaborative filtering methods since most users only see a tiny portion of all music libraries and hence most assessments are not decided. When a new user is added to the system it will not initially have enough ratings for the system to find sufficiently similar users and thus the accuracy of the predictions will be limited. Another example of cold start is when a new item is added to the system. The 9 item will not have any ratings so it won't be recommended to any users. There are many ways so solve this problem, asking new user for initial ratings or recommending the most popular items while gathering more information are two approaches to overcome the new user cold start problem. An approach to solve the new item problem is to implement content-based filtering in a hybrid approach which will be discussed in the coming section. For example, if user X and user Y have rated a large number of products similarly, their relationship will be strong, and when user X offers a fresh high rating on an item that user Y has not yet reviewed, the system will suggest that item to user Y. A neighbourhood is a collection of people who share similar tastes, and predictions and suggestions may be generated by looking at the neighbor's rankings and user history. Pearson's correlation coefficient might be used to determine how similar two users are. The fundamental assumption here is that the opinions of other users can be used to provide a reasonable prediction of another user's preferences for an item that they have not yet rated: a user is given recommendations based on users with whom they share the same tastes with. Indeed, during years, in order to choose music, restaurants, movies, etc.. We have been asking our friends, family, and colleagues to recommend something they liked. And it is this mechanism that is attempted to be reproduced here. Netflix was a pioneer of this method (based on stars given by other users) but it is now widely used, including for Spotify's Discover Weekly. Collaborative filtering makes suggestions based on the collaborative power of the available evaluation by users. Implementing content-based filtering in a hybrid method to handle the new item problem is one solution, which will be described in the next section. The cold start problem for new users also applies to content-based filtering, because a new user will have no songs to filter and recommend. As stated in the scope, our system does not rely on collaborative filtering because we do not have the necessary user base to support it. Deep neural networks are the only implementation that uses collaborative filtering, and even then, its use is limited. We had planned to use collaborative filtering in conjunction with content-based filtering and real-time data, but this was not feasible. It is assumed that if people rate music things similarly or act in similar ways, they would rate other music items similarly. Because most users only encounter a tiny portion of all music libraries, the sparse evaluation matrix is the fundamental issue in collaborative filtering approaches. As a result, most assessments are not decided.

2) <u>**Content based filtering :**</u>



The investigation of the content of the items candidates for suggestion is what contentbased recommendation is all about. This method attempts to deduce the user's tastes in order to suggest goods that are similar in content to those they have previously enjoyed. This method does not require listener feedback; it is only based on sound similarity, which is calculated using information taken from previously heard songs. The characteristics of each item are used in content-based filtering to locate items that are comparable. We may propose an item based on how similar it is to all other things in the dataset by assigning a score to how similar each item is. We utilise the properties (loudness, pace, etc.) of each song in a Spotify playlist to calculate the average score of the entire playlist. Then we suggest a song that has a comparable score to the playlist but isn't on it. Collaboration filtering has concerns with new items, while content-based filtering does not, as will be addressed in the following section. However, one disadvantage of content-based filtering is that it will only propose songs that are similar to what you currently listen to and will not suggest music that are different. For a new user who has only listened to a few songs, the suggestions will be based exclusively on these few listenings, making them incredibly predictable and disappointing. However, if the extracted information is diverse enough and based on a variety of characteristics, the system may be able to make connections between songs that do not sound similar to the human ear. A two-stage technique is used in content-based approaches to extract traditional audio content elements and forecast user preferences. In order to identify audio perceptual similarity, several studies have focused on extracting and comparing acoustic variables like as timbre and rhythm. Personalized suggestions in the form of item rankings will arise from both collaborative and content-based screening. Because it compares the similarity of characteristics on audio signals, our method may be classified as contentbased music recommendation. We provide music suggestions based on how similar the user's opinions of their favourite music were when they first heard it. To locate related music, the project uses content-based filtering in addition to user id and real-time parameters. 11 The difficulty of collecting data does not apply to our project because Spotify offers a wide range of information associated with a track, which will be utilised as the material for comparing songs in our system. However, due to a lack of user data, we will only be able to use three of the features: volume, mode, and tempo. It's impossible to say whether these characteristics are the best for describing a recording, while pace and loudness have been linked to heart rate. The theory behind content-based filtering is that if a person is interested in one item, they are also likely to be interested in similar products. As a result, content-based filtering employs labels and characteristics to organise and filter objects. Different qualities can be retrieved depending on the object category, and items with comparable attributes are grouped together. Only the target user's history is required this method. The content of the items in the user's history is compared to the content of other domain items, and the domain items with the highest resemblance to the ones in the user's past are recommended. When proposing news articles, content-based filtering is a popular use. The similarity between the elements is the basis for this strategy. It's a matter of extracting features to best describe the music in order to evaluate similarities. The Machine Learning algorithms then suggest the item that is the most similar to the ones that the user already likes. A two-stage technique is used in content-based approaches to extract traditional audio content elements and forecast user preferences. In order to identify audio perceptual similarity, several studies have focused on extracting and comparing acoustic variables like as timbre and rhythm. Personalized suggestions in the form of item rankings will arise from both collaborative and content-based screening. Because it compares the similarity of characteristics on audio signals, our method may be classified as contentbased music recommendation. We provide music suggestions based on how similar the user's opinions of their favourite music were when they first heard it. The main advantage of this approach is that an unknown music is just as likely to be recommended as a currently popular one, or even a timeless one. This allows new artists with a few" views" to be brought up as well. Moreover, the problem of the cold start and in particular of the new items is thus avoided: when new items are introduced into the system, they can be recommended directly, without requiring integration time as is the case for recommendation systems based on a collaborative filtering approach.

3) **Hybrid filtering :**



It is also feasible to develop a hybrid recommendation system by combining the preceding complementing strategies. It can also be based on less well-known techniques like locationbased recommendations. This strategy can help with cold start and sparsity issues. Several 13 implementations may be put up, the first of which combines the recommendation systems into one. A hybrid method also allows for the evaluation of more criteria for each proposal. Demographic filtering, a sort of collaborative filtering that puts people in the same demographic together, can also be applied into such a system. Our recommender system uses both content-based and real-time data to filter results, making it a hybrid method. As previously said, we would have preferred to take a collaborative approach to address some of the difficulties with content-based filtering. Personalized suggestions in the form of item rankings will arise from both collaborative and content-based screening. Because it compares the similarity of characteristics on audio signals, our method may be classified as content-based music recommendation. Using realtime data does not solve the difficulties with content-based filtering; rather, it is meant to improve the suggestions by customising them depending on the user's present condition. It's also feasible to maintain many systems distinct and give weights to them, as well as the option to switch between them at anytime. Finally, outputs from one system may be extracted and utilised as input for a subsequent system.

CHAPTER – 2

TECHNOLOGY USED

Machine Learning :

Machine learning (ML) is a branch of artificial intelligence (AI) that enables computers to “self-learn” from training data and improve over time, without being explicitly programmed. Machine learning algorithms are able to detect patterns in data and learn from them, in order to make their own predictions. In short, machine learning algorithms and models learn through experience.

In traditional programming, a computer engineer writes a series of directions that instruct a computer how to transform input data into a desired output. Instructions are mostly based on an IF-THEN structure: when certain conditions are met, the program executes a specific action.

Machine learning, on the other hand, is an automated process that enables machines to solve problems with little or no human input, and take actions based on past observations.

While artificial intelligence and machine learning are often used interchangeably, they are two different concepts. AI is the broader concept – machines making decisions, learning new skills, and solving problems in a similar way to humans – whereas machine learning is a subset of AI that enables intelligent systems to autonomously learn new things from data.

Instead of programming machine learning algorithms to perform tasks, you can feed them examples of labeled data (known as training data), which helps them make calculations, process data, and identify patterns automatically.

Put simply, Google’s Chief Decision Scientist describes machine learning as a fancy labeling machine. After teaching machines to label things like apples and pears, by showing them examples of fruit, eventually they will start labeling apples and pears without any help – provided they have learned from appropriate and accurate training examples.

Machine learning can be put to work on massive amounts of data and can perform much more accurately than humans. It can help you save time and money on tasks and analyses, like solving customer pain points to improve customer satisfaction, support ticket automation, and data mining from internal sources and all over the internet.

Python :

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics developed by Guido van Rossum. It was originally released in 1991. Designed to be easy as well as fun, the name "Python" is a nod to the British comedy group Monty Python. Python has a reputation as a beginner-friendly language, replacing Java as the most widely used introductory language because it handles much of the complexity for the user, allowing beginners to focus on fully grasping programming concepts rather than minute details. Python is used for server-side web development, software development, mathematics, and system scripting, and is popular for Rapid Application Development and as a scripting or glue language to tie existing components because of its high-level, built-in data structures, dynamic typing, and dynamic binding. Program maintenance costs are reduced with Python due to the easily learned syntax and emphasis on readability. Additionally, Python's support of modules and packages facilitates modular programs and reuse of code. Python is an open source community language, so numerous independent programmers are continually building libraries and functionality for it.

Jupyter Notebook :

The Jupyter Notebook is an open source web application that you can use to create and share documents that contain live code, equations, visualizations, and text. Jupyter Notebook is maintained by the people at Project Jupyter. Jupyter Notebooks are a spin-off project from the IPython project, which used to have an IPython Notebook project itself. The name, Jupyter, comes from the core supported programming languages that it supports: Julia, Python, and R. Jupyter ships

CHAPTER - 3

LITERATURE SURVEY

We were awestruck with Spotify's recommendation engine. We always wondered how Spotify manages to recommend that perfect song, playlist or even that 'daily mix' . We now have more technology than ever before to ensure that if you're the smallest, strangest musician in the world, doing something that only 20 people in the world will dig, we can now find those 20 people and connect the dots between 20 the artist and listeners. This has been the motivation for this project to use various machine learning techniques and to develop a music recommendation engine similar to that of Spotify, which takes music listening experience to another level. Music Recommendation Systems. Recommender systems help consumers deal with the problem of information overload by providing them with individualised, unique content and service suggestions. Various methods for developing recommendation systems have recently been created, including collaborative filtering, content-based filtering, and hybrid filtering. The collaborative filtering approach is the most developed and widely used. Collaborative filtering suggests things by locating other users who have similar tastes to the current user and using their recommendations. Collaborative recommender systems have been used in a variety of settings. problems – new items and new users. The first problem refers to the new items that are meant to be recommended, but the information that is associated with them,. The common characteristics in these systems are constant when using users' preferences compared with users' context (location, mood, weather, etc.). For instance, in the library when people are sitting there maybe they need quiet and melodious music to listen according to the environment where they are in. Last.fm, All music, Spotify, Pandora and Shazam are commercial music recommendation systems which are considered to be excellent systems by focusing on the music already played in order to help the users to find more music. Users are able to connect to a web-based music streaming service to access the recommendations. All the tracks that are played on this stream are recommended. It is Based on songs or artists which users either upload from your iTunes playlists or add as favourites on the site where users start managing their library of music with tags and keep tracking of the music the friends who listening to and getting multiple recommendations per song played. Additionally, this app filters recommendations by decade, genre, and popularity, as well as builds fabulous playlists (Song et al., 2012).t has been found that CF generally gives better recommendations than CB. However, this is only true if there is usage data available, such as the ratings given to previous tracks. If this is not the case, then it will not prove accurate results and, consequently, suffer from the Cold-Start problem, which includes two categories of problems – new items and new users. The first problem refers to the new items that are meant to be recommended, but the information that is associated with them,. Meanwhile, many researchers have used social media (Twitter & Facebook) to identify user's mood (tension, depression, anger, vigor, fatigue, confusion) and also identify user's personality (openness, conscientiousness, extraversion, 21 agreeableness, neuroticism) where these are very important factors which influence on user's music taste (Wang et al., 2014; Roberts et al., 2012; Pandarachalil et al., 2015; Ross et al., 2009; Bachrach et al., 2012; Back et al., 2010) and also contextual features (location & event) can lead to different emotional effects due to objective features of the situation or subjective perceptions of the listeners (Scherer et al., 2001). Music lyrics are also considered to be one of emotional presentation because they include some kinds of implicit thinking, thus we can fully understand emotions and their associated thinking in each song (Nunes and Jannach, 2017; Tintarev and Masthoff, 2008). Cano et al. (2017) mentioned that there is a strong relation between the user mood and listening to the music. The people may want to listen to music which has the same mood of them when they are in specific mood and in contrast the people want to listen to different kind of music which encourage them to enhance their mood and this thing depend on the psychological studies and therefore, the author produced a contextual mood-based music recommender system which is able to regulate the driver's mood and also try to put the driver in a positive mood when driving because listening to the music while driving has always been one of the most favourite activities carried out by people. Finally, similarly, active learning approaches suffer from various limitations.

CHAPTER – 4

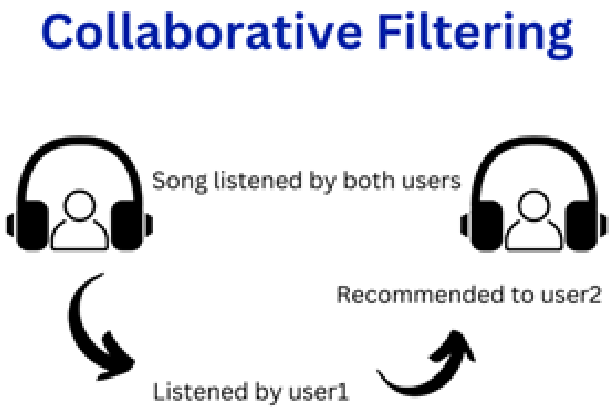
APPROACH

There are various types of recommender system with various approaches and some of them which we have used in our project are :

1. Collaborative Filtering :

Collaborative filtering involves collecting information from many users and then making predictions based on some similarity measures between users and between items. This can be classified into user-based and itembased models. Figure 1: Item Based In item-based model, it is assumed that songs that are often listened together by some users tend to be similar and are more likely to be listened together in future also by some other user. According to user based similarity model, users who have similar listening histories, i.e., have listened to the same songs in the past tend to have similar interests and will probably listen to the same songs in future too. We need some similarity measure to compare between two songs or between two users. Cosine similarity weighs each of the users equally which is usually not the case. User should be weighted less if he has shown interests to many variety of items (it shows that either she does not discern between songs based on their quality, or just likes to explore). Likewise, user is weighted more if listens to very limited set of songs. The similarity measure, wij = P(i j), also has drawbacks that some songs which are listened more by users have higher similarity values not because they are similar and listened together but because they are more popular. We have used conditional probability based model of similarity[2] between users and between items:
Different values of α were tested to finally come with a good similarity measure. Then, for each new user u and song i, user-based scoring function is calculated as h U ui = P vui f(wuv)[2] Similarly, item-based scoring function is h I ui = P jIi f(wij)[2] Locality of scoring function is also necessary to emphasize items that are more similar. We have used exponential function to determine locality. f(w) = w q , qN[2] This determines how individual scoring components affects the overall scoring between two items. The similar things are emphasized more while less similar ones contribution drop down to zero. After computing user-based and item-based lists, we used stochastic aggregation

to combine them. This is done by randomly choosing one of them according to their probability distribution and then recommending top scored items from them. When the song history of a user is too small to utilize the user-based recommendation algorithm, we can offer recommendations based on song similarity, which yields better results when number Figure 3: Matrix M of songs is smaller than that of users. We got the best results for stochastic aggregation of item-based model with values of q and α as 3 and 0.15, and of user-based model with values 0.3 and 5 for α and q respectively, giving overall mAP 0.08212. (See details about mAP in the Sec 3.5) This method does not include any personalization. Moreover, majority of songs have too few listeners so they are least likely to be recommended. But still, we as well as Kaggle winner got best results from this method among many others. Here, we have not used play count information in final result as they did not give good result because similarity model is biased to a few songs played multiple times and calculation noise was generated by a few very popular songs.



The filtering uses a **user-item interaction matrix**. This matrix defines the interaction between the user to the item and it can be shown by the rating given by the users. In case there’s no explicit rating, the parameters like user searched for that item or clicked on that, etc are considered.

a) <u>Item – Based collaborative filtering :</u>

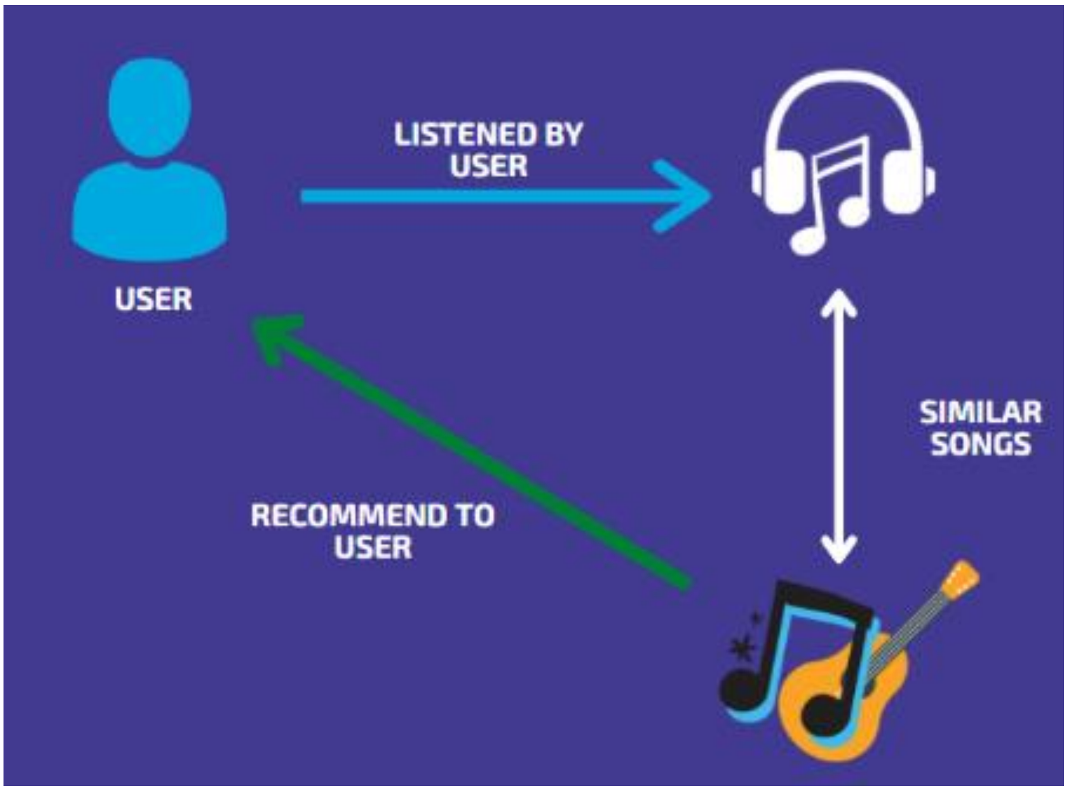
Item-item collaborative filtering, or item-based, or item-to-item, is a form of collaborative filtering for recommender systems based on the similarity between items calculated using people's ratings of those items. Item-item collaborative filtering was invented and used by Amazon.com in 1998. It was first published in an academic conference in 2001.

Earlier collaborative filtering systems based on rating similarity between users (known as user-user collaborative filtering) had several problems:

- systems performed poorly when they had many items but comparatively few ratings
- computing similarities between all pairs of users was expensive
- user profiles changed quickly and the entire system model had to be recomputed

Item-item models resolve these problems in systems that have more users than items. Item-item models use rating distributions *per item*, not *per user*. With more users than items, each item tends to have more ratings than each user, so an item's average rating usually doesn't change quickly. This leads to more stable rating distributions in the model, so the model doesn't have to be rebuilt as often. When users consume and then rate an item, that item's similar items are picked from the existing system model and added to the user's recommendations.

Item-based collaborative filtering is the recommendation system to use the similarity between items using the ratings by users. In this article, I explain its basic concept and practice how to make the item-based collaborative filtering using Python.



ADVANTAGES OF COLLABORATIVE FILTERING :

- **No domain knowledge necessary**

  We don't need domain knowledge because the embeddings are automatically learned.

- **Serendipity**

  The model can help users discover new interests. In isolation, the ML system may not know the user is interested in a given item, but the model might still recommend it because similar users are interested in that item.

- **Great starting point**

  To some extent, the system needs only the feedback matrix to train a matrix factorization model. In particular, the system doesn't need contextual features. In practice, this can be used as one of multiple candidate generators.

A. DISADVANTAGES OF COLLABORATIVE FILTERING :

- **Cannot handle fresh items**

  The prediction of the model for a given (user, item) pair is the dot product of the corresponding embeddings. So, if an item is not seen during training, the system can't create an embedding for it and can't query the model with this item. This issue is often called the **cold-start problem**.

- **Hard to include side features for query/item**

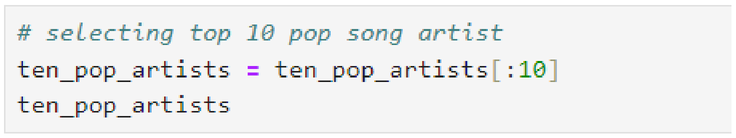
  Side features are any features beyond the query or item ID. For movie recommendations, the side features might include country or age. Including available side features improves the quality of the model. Although it may not be easy to include side features in WALS, a generalization of WALS makes this possible.

CHAPTER – 5

RESULT

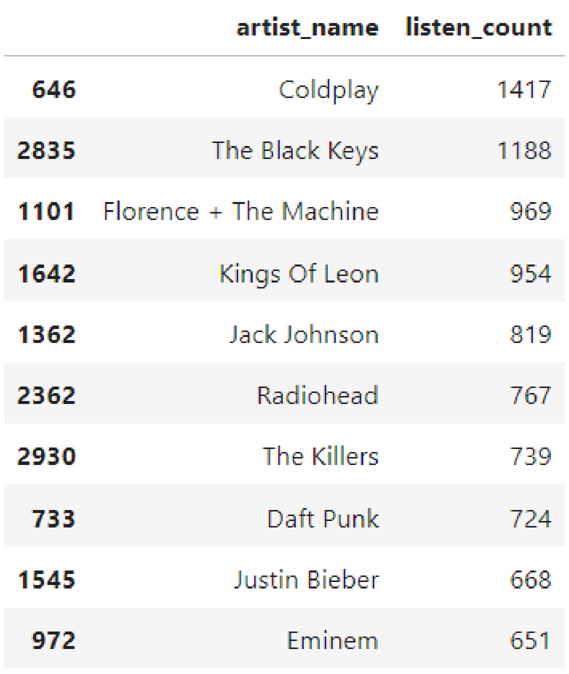
1. Selecting top 10 artist from the dataset

Command :



```
# selecting top 10 pop song artist
ten_pop_artists = ten_pop_artists[:10]
ten_pop_artists
```

Output :



	artist_name	listen_count
646	Coldplay	1417
2835	The Black Keys	1188
1101	Florence + The Machine	969
1642	Kings Of Leon	954
1362	Jack Johnson	819
2362	Radiohead	767
2930	The Killers	739
733	Daft Punk	724
1545	Justin Bieber	668
972	Eminem	651

2. Giving song history of particular user

Command :

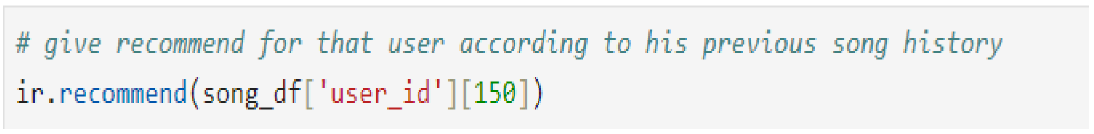
```
user_items = ir.get_user_items(song_df['user_id'][150])

# display user songs history
for user_item in user_items:
    print(user_item)
```

Output :

```
Harder Better Faster Stronger-Daft Punk
Jumping Jack Flash-The Rolling Stones
Aerodynamic-Daft Punk
You Know What You Are?-Nine Inch Nails
Indo Silver Club-Daft Punk
Steam Machine-Daft Punk
High Life-Daft Punk
D.A.N.C.E. [Radio Edit]-Justice
Emotion-Daft Punk
Meanwhile_ Rick James...-Cake
Face To Face-Daft Punk
Digital Love-Daft Punk
Fresh-Daft Punk
Sad Songs And Waltzes-Cake
Especially In Michigan (Album Version)-Red Hot Chili Peppers
Rock'n Roll-Daft Punk
Nightvision-Daft Punk
The Brainwasher-Daft Punk
C'mon Girl (Album Version)-Red Hot Chili Peppers
It's Tricky-RUN-DMC
Around The World (Radio Edit)-Daft Punk
Strip My Mind (Album Version)-Red Hot Chili Peppers
Top Down-Swizz Beatz
The Real Slim Shady-Eminem
I Could Have Lied (Album Version)-Red Hot Chili Peppers
Opera Singer-Cake
The Prime Time Of Your Life-Daft Punk
I'm Back-Eminem
Sinisten tähtien alla-J. Karjalainen & Mustat Lasit
One More Time (Short Radio Edit)-Daft Punk
Da Funk-Daft Punk
Crescendolls-Daft Punk
Angie (1993 Digital Remaster)-The Rolling Stones
Too Long-Daft Punk
Superhereos-Daft Punk
Breaking The Girl (Album Version)-Red Hot Chili Peppers
Technologic-Daft Punk
```

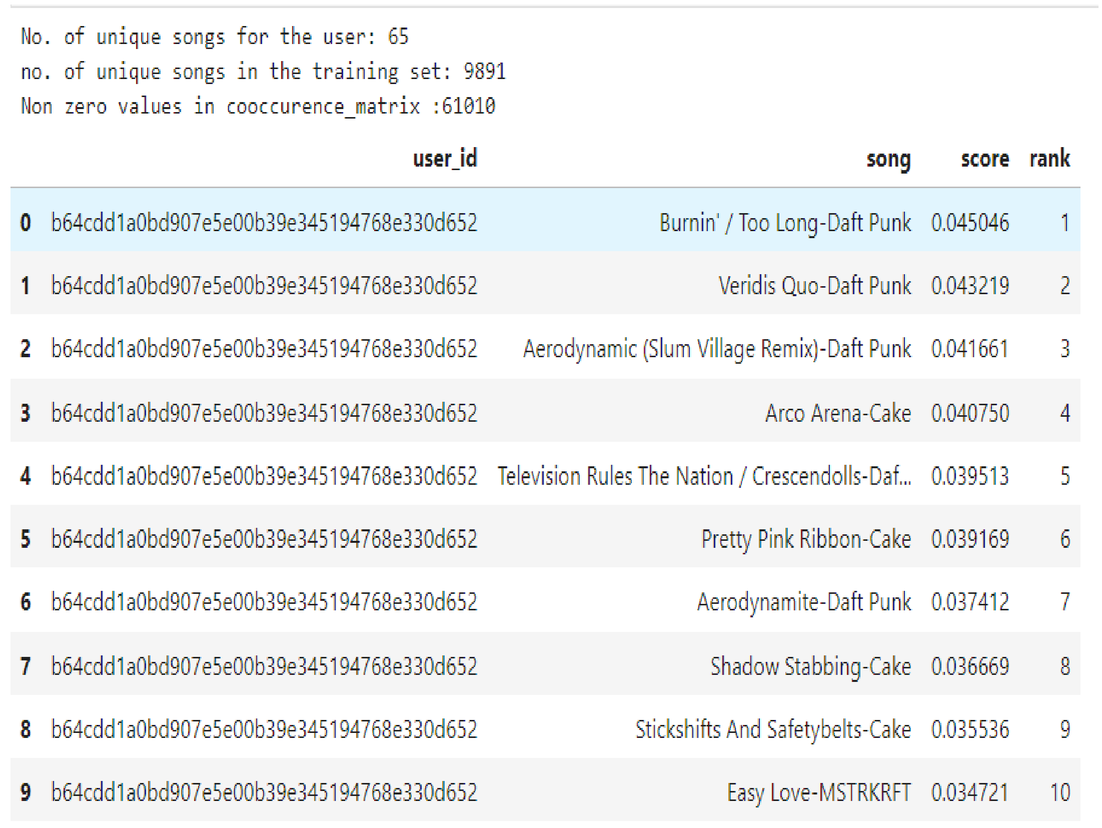
3. <u>Recommend song on the basis of user’s previous history</u>

<u>Command :</u>



```python
# give recommend for that user according to his previous song history
ir.recommend(song_df['user_id'][150])
```

<u>Output :</u>

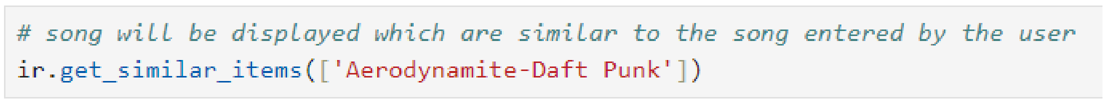


```
No. of unique songs for the user: 65
no. of unique songs in the training set: 9891
Non zero values in cooccurence_matrix :61010
```

	user_id	song	score	rank
0	b64cdd1a0bd907e5e00b39e345194768e330d652	Burnin' / Too Long-Daft Punk	0.045046	1
1	b64cdd1a0bd907e5e00b39e345194768e330d652	Veridis Quo-Daft Punk	0.043219	2
2	b64cdd1a0bd907e5e00b39e345194768e330d652	Aerodynamic (Slum Village Remix)-Daft Punk	0.041661	3
3	b64cdd1a0bd907e5e00b39e345194768e330d652	Arco Arena-Cake	0.040750	4
4	b64cdd1a0bd907e5e00b39e345194768e330d652	Television Rules The Nation / Crescendolls-Daf...	0.039513	5
5	b64cdd1a0bd907e5e00b39e345194768e330d652	Pretty Pink Ribbon-Cake	0.039169	6
6	b64cdd1a0bd907e5e00b39e345194768e330d652	Aerodynamite-Daft Punk	0.037412	7
7	b64cdd1a0bd907e5e00b39e345194768e330d652	Shadow Stabbing-Cake	0.036669	8
8	b64cdd1a0bd907e5e00b39e345194768e330d652	Stickshifts And Safetybelts-Cake	0.035536	9
9	b64cdd1a0bd907e5e00b39e345194768e330d652	Easy Love-MSTRKRFT	0.034721	10

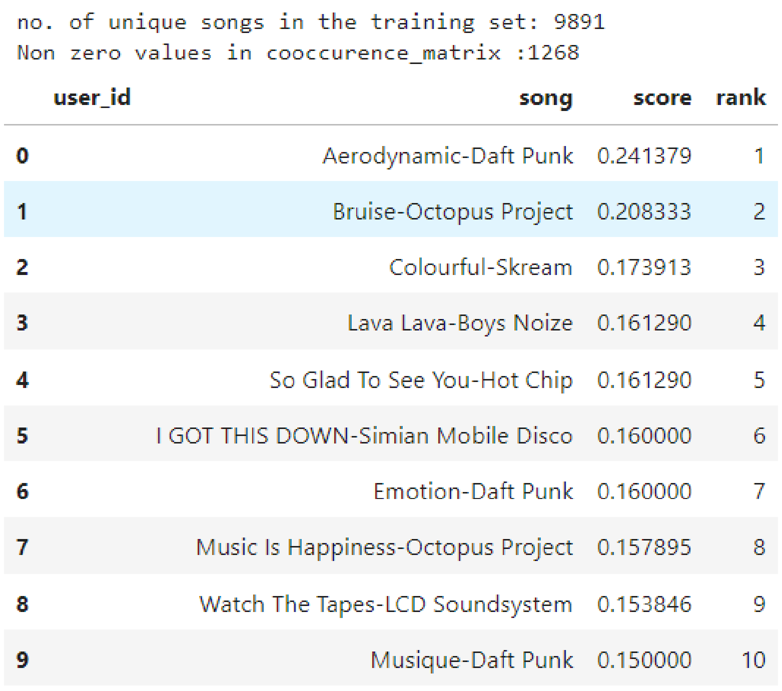
4. Recommend songs on the user’s input

Command :



```
# song will be displayed which are similar to the song entered by the user
ir.get_similar_items(['Aerodynamite-Daft Punk'])
```

Output :



```
no. of unique songs in the training set: 9891
Non zero values in cooccurence_matrix :1268
```

	user_id	song	score	rank
0		Aerodynamic-Daft Punk	0.241379	1
1		Bruise-Octopus Project	0.208333	2
2		Colourful-Skream	0.173913	3
3		Lava Lava-Boys Noize	0.161290	4
4		So Glad To See You-Hot Chip	0.161290	5
5		I GOT THIS DOWN-Simian Mobile Disco	0.160000	6
6		Emotion-Daft Punk	0.160000	7
7		Music Is Happiness-Octopus Project	0.157895	8
8		Watch The Tapes-LCD Soundsystem	0.153846	9
9		Musique-Daft Punk	0.150000	10

CHAPTER – 6

CONCLUSION

This is a project of our Artificial Intelligence course. We find it is very good as we got a chance to practice theories that we have learnt in the course, to do some implementation and to try to get a better understanding of a real artificial intelligence problem: Music Recommender Sys4 tem. There are many different approaches to this problem and we get to know some algorithms in detail and especially the four models that weve explained in the paper. By manipulating the dataset, changing the learning set and testing set, changing some parameters of the problem and analyzing the result, we earn a lot practicing skills. Weve faced a lot of problem in dealing with this huge dataset, how to explore it in a better way and we also had difficulties in some programming details. However, with lot of efforts, we have overcame all of these. The best part of this project is the teamwork. Both of us come from different countries and thus have different cultures and ways of working. We took a bit of time to get to know each other, to adjust ourselves and to perform like a team. We become much more efficient by the time the team spirit is formed and we also enjoy more. We both find this project a nice experience and all the effort put is worthy. We have learnt a lot from this project. In terms of research, we still have a lot to do to make our studies a better one. Music Recommender System is such a wide, open and complicated subject that we can take some initiatives and do a lot more tests in future. We also got to realize that building a recommender system is not a trivial task. The fact that its large scale dataset makes it difficult in many aspects. Firstly, recommending 500 correct songs out of 380 million for different users is not an easy task to get a high precision. Thats why we didnt get any result better than 10 %. Even the Kaggle winner has only got 17 %. Secondly, the metadata includes huge information and when exploring it, it is difficult to extract relevant features for song. Thirdly, technically speaking, processing such a huge dataset is memory and CPU intensive. All these difficulties due to the data and to the system itself make it more challenging and also more attractive. We hope that we will get other opportunities in the future to work in the domain of artificial intelligence. We are certain that we can do a better job.

FUTURE WORKS :

• Run the algorithms on a distributed system, like Hadoop or Condor, to parallelize the computation, decrease the runtime and leverage distributed memory to run the complete MSD.
• Combine different methods and learn the weightage for each method according to the dataset
• Automatically generate relevant features
• Develop more recommendation algorithms based on different data (e.g. the how the user is feeling, social recommendation, etc)

REFERENCES

1. Francesco Ricci and Lior Rokach and Bracha Shapira, Introduction to Recommender Systems Handbook, Recommender Systems Handbook, Springer, 2011, pp. 1–35

2. ^ Jump up to:[a] [b] Terveen, Loren; Hill, Will (2001). "Beyond Recommender Systems: Helping People Help Each Other" (PDF). Addison-Wesley. p. 6. Retrieved 16 January 2012.

3. ^ An integrated approach to TV & VOD Recommendations Archived 6 June 2012 at the Wayback Machine

4. ^ John S. Breese, David Heckerman, and Carl Kadie, Empirical Analysis of Predictive Algorithms for Collaborative Filtering, 1998 Archived 19 October 2013 at the Wayback Machine

5. ^ Xiaoyuan Su, Taghi M. Khoshgoftaar, A survey of collaborative filtering techniques, Advances in Artificial Intelligence archive, 2009.

6. ^ Jump up to:[a] [b] [c] Recommender Systems – The Textbook | Charu C. Aggarwal | Springer. Springer. 2016. ISBN 9783319296579.

7. ^ Ghazanfar, Mustansar Ali; Prügel-Bennett, Adam; Szedmak, Sandor (2012). "Kernel-Mapping Recommender system algorithms". Information Sciences. **208**: 81–104. CiteSeerX 10.1.1.701.7729. doi:10.1016/j.ins.2012.04.012. S2CID 20328670.

8. ^ Das, Abhinandan S.; Datar, Mayur; Garg, Ashutosh; Rajaram, Shyam (2007). "Google news personalization". Proceedings of the 16th international conference on World Wide Web – WWW '07. p. 271.

9. Mehta, Bhaskar; Hofmann, Thomas; Nejdl, Wolfgang (19 October 2007). Proceedings of the 2007 ACM conference on Recommender systems – Rec Sys '07. Portal.acm.org. p. 49. CiteSeerX 10.1.1.695.1712.