Refined Decision Tool Concept Note

With the development of information technology and the Internet, people have gradually entered the era of information overload from an era of lack of information. Information consumers want to easily find the content they are interested in, and information producers want to push their content to the most suitable place. How to distinguish target audience now turned into a problem.

The recommendation system establishes the relationship between the user and the item through different strategies, thereby displaying the items that the user may be interested in. It is based on a large amount of valid data. There are many kinds of algorithmic ideas behind it. The main classification includes four types: Item-Based Collaborative Filtering, User-Based Collaborative Filtering, Tag-based recommendation, use of social network data, etc.

Digital music has become the mainstream consumer content sought by many young people. How to help users quickly and accurately obtain music tracks that users are interested in the vast music library with the personalized recommendation of music has become increasingly important.

In China, there is a famous music application called NetEase Cloud Music. NetEase Cloud Music is positioned to build a user-centric mobile music community, using the features of "Music + Social" to help users better discover and share music. Netease Cloud Music is the first domestic music product based on a playlist. With the playlist as an entrance, users can obtain high-quality music according to their music tastes. There are many hot popular playlists which include users' music interest information.

1. Formalism

(1) The decision-maker's action set $\mathbb{A}$:

In this project, the decision maker is the music application. It needs to recommend appropriate songs for different users. If the music application could make an accurate recommendation, users will be more inclined to continue using this application. After accumulating a large number of users, the music application can make a profit by selling songs, membership fees, advertisements and so on.

- A: All songs in the music library
- a^*: Recommending songs based on users and songs.

(2) The state space of sample space $\mathbb{X}$ & description of the data generating process:

From the observation data, $\mathbb{X}$ contains various characteristics of song list.

- X_1: Name of song list. Unique values
- X_2: ID of song list. Unique values
- X_3: Count of subscriber for song list. Separate non-negative values
- X_4: Categories of song list. Separate values
- X_5: Name of song. Unique values
- X_6: ID of song. Unique values
- X_7: Name of artist. Unique values
- X_8: Popularity of song. Separate non-negative values
- $\cdots$

(3) The parameter space Θ:

As described above, the values of characteristics can be numerical and all of them are non-negative.

(4) The description of the decision-maker's prior beliefs:

- X_3: $x_3 \in [0, 1800000]$, which means the amount of subscribed count is between 0 and 1,800,000
- X_4: $x_4 \in [20, 40]$, which means there are about 20-40 different types of tags
- X_8: $x_8 \in [0, 100]$, which means the value of popularity is between 0 and 100

(5) The description of the process for getting better information

(A) Recommendation algorithm

Music is a typical area with rich long-tail items and strong user personalization needs, and the number and change of items are not as significant as the user. It is very suitable for item-based collaborative filtering algorithms. Most users who like music A also like music B. They consider the two musicians to be similar. The algorithm for calculating similarity is as follows:

$$w_{ij} = \frac{\sum_{u \in N(i) \cap N(j)} \frac{1}{\log(1+|N(u)|)}}{\sqrt{|N(i)||N(j)|}}$$

N(i) and N(j) denote users who like items i and j respectively, and N(u) denote user's favorite music collection

It should be noted that because of the wide consumption of popular music, and music has a great similarity with popular music. At the same time, the weight of each user's behavior is different. There is a connection between the music, which is not completely equivalent to the behavior of inactive users, so here is a certain punishment for popular music and active users. It is generally believed that unpopular music reflects interest more than popular music, and inactive users are more valuable than active users' consumption behavior.

After obtaining the music similarity, we can calculate the user u's interest in a music j by the following formula, and get an initial recommendation result, that is, the more similar to the music of interest in the user's music history, the more likely it is to get higher similarity.

$$p_{uj} = \sum_{i \in N(u) \cap S(j,K)} w_{ji} r_{ui}$$

S(j,K) is the set of K items that are most similar to item j, w_{ij} is the similarity of music j and i, and r_{ui} represents the user u's interest in music i

(B) Filter & Sort Module

After the initial results are obtained, further processing of the data is required in order to get a list of recommendations available. First of all, the recommendation system will filter according to certain established policies, such as poor quality music, music that users have prescribed behaviors (such as likes/downloads) within a certain period of time, music that the user shows obviously dislike, and so on.

After the filtering is completed, ranking the remaining results according to the existing ranking rules. Ranking rules need to consider similarity, user feedback (listen / download / like), novelty, diversity, and time diversity. And other factors, to obtain an equation containing multiple explanatory variables, the value calculated based on each song.

Time diversity refers to that the results obtained by users at different times or after generating new behaviors are not the same. To improve the time diversity of the recommendation system, we must start in two places. First, we must ensure the real-time nature of the recommendation system, and adjust the recommendation results in real-time to meet the user's recent needs when the user has new behavior. The second aspect is to ensure that the recommendation results change every day when the user has no new behavior.

(C) Normalization and Diversity

Due to the high similarity of music in some music categories (in general, the similarity of popular categories is relatively large), if not normalized, it may cause the recommended list to be a certain type of music. So in practice, you can consider all the songs of category A and B as an item, so the recommended result should be that the number of songs of category A and B is similar, and then choose 5 from category A and B respectively. The song can be selected in the above manner. Of course, in actual operation, the normalized granularity needs to be determined according to specific conditions.

(D) Cold start problem

Since the above-mentioned recommendation methods all rely on the user's historical data, there is a problem associated with this. For the first-time user and the music being recorded for the first time, how to make a recommendation?

(a) User cold start

- Use the social network account information. The music application could support users to use social network accounts in the registration and login interface. Import user information on social networks and publicly released information for data analysis. This may involve the analysis of demographic information (gender, occupation) and publicly published information to obtain a preliminary user portrait; meanwhile, it can also be based on social friendship chain recommendation.
- Ask users to describe their interests. Use the selected music tag to recommend music based on content attributes. The recommended music at the cold start should have the following characteristics:
    - Be representative and distinguishable, try not to recommend popular music that most people will like;
    - Popularity, users tend to be more familiar with music when expressing their interests for the first

time, and music that is too unpopular is not easy to evoke the user's willingness to express;
    - The diversity of startup item collections. In the initial recommendation, the startup item collections with the highest coverage should be provided to allow users to better feedback their interests;

(b) Music cold start

The newly recorded music cannot be recommended according to the above method because there is no accumulation of user usage data.

- Appropriate tilt on resources. During product operation, exposure opportunities can be left in some non-recommended locations, such as new song lists etc., thereby increasing the possibility of users' behavior on newly included music.
- Complemented by other recommended methods. According to the content information of the music, it is recommended to users who have liked music similar to its content; at the same time, the recommendation algorithm mentioned above can add item-related recommendations calculated using the content information of the items and collaborative filtering based on users In the actual operation of the recommendation system, different recommendation engines can be given different weights to obtain a mixed initial result, and then uniformly filter and sort.

(6) The relevant payoffs that could result from the action

For a music application, if it could make an accrurate recommendation, users are more likely to use this music application and the music application could make profits from advertisements, selling songs and membership fee.

The recommendation algorithm could be formulated as follows:

$$w_{ij} = \frac{\sum_{u \in N(i) \cap N(j)} \frac{1}{\log{(1+|N(u)|)}}}{\sqrt{|N(i)||N(j)|}}$$

N(i) and N(j) denote users who like items i and j respectively, and N(u) denote user's favorite music collection

$$p_{uj} = \sum_{i \in N(u) \cap S(j,K)} w_{ji} r_{ui}$$

S(j,K) is the set of K items that are most similar to item j, w_{ij} is the similarity of music j and i, and r_{ui} represents the user u's interest in music i

The payoff function could be formulated as follows:

$$Payoffs = f(x_1, x_2, \ldots) \times N_{users} \times (Profit_{advertisement} + Profit_{membership} + Profit_{song})$$

(7) The decision-maker's utility function or loss function

$$f(x_1, x_2, \ldots) = \begin{cases} 0, when\ making\ appropriate\ recommendation \\ 1, when\ making\ inappropriate\ recommendation \end{cases}$$

To judge whether it is an appropriate recommendation,

$$g(x_1, x_2, \ldots) = \begin{cases} 0, users\ skip\ in\ thirty\ seconds \\ 1, users\ listen\ more\ than\ thirty\ seconds \end{cases}$$

(8) The rule the decision-maker will use to choose a preferred action a^* from the menu $\mathbb{A}$ of possible actions

As described above:

- Recommendation algorithm
- Filter & Sort Module
- Normalization and Diversity
- User and Music Cold Start
- ...

2. Data

The original data are crawled from NetEase Cloud Music and stored as json. The data of song list and song could be described as follows:

(1) The format of song list


```
{
    "result": {
        "id": 111450065,
        "status": 0,
        "commentThreadId": "A_PL_0_111450065",
        "trackCount": 120,
        "updateTime": 1460164523907,
        "commentCount": 227,
        "ordered": true,
        "anonimous": false,
        "highQuality": false,
        "subscribers": [],
        "playCount": 687070,
        "trackNumberUpdateTime": 1460164523907,
        "createTime": 1443528317662,
        "name": "",
        "cloudTrackCount": 0,
        "shareCount": 149,
        "adType": 0,
        "trackUpdateTime": 1494134249465,
        "userId": 39256799,
        "coverImgId": 3359008023885470,
        "coverImgUrl": "http://p1.music.126.net/2ZFcuSJ6STR8WgzkIi2U-Q==/3359008023885470.jpg",
```

```
        "artists": null,
        "newImported": false,
        "subscribed": false,
        "privacy": 0,
        "specialType": 0,
        "description": "",
        "subscribedCount": 10882,
        "totalDuration": 0,
        "tags": []
        "creator": {
            "followed": false,
            "remarkName": null,
            "expertTags": [],
            "userId": 39256799,
            "authority": 0,
            "userType": 0,
            "gender": 1,
            "backgroundImgId": 3427177752524551,
            "city": 360600,
            "mutual": false,
            "avatarUrl":
"http://p1.music.126.net/TLRTrJpOM5lr68qJv1IyGQ==/1400777825738419.jpg",
            "avatarImgIdStr": "1400777825738419",
            "detailDescription": "",
            "province": 360000,
            "description": "",
            "birthday": 637516800000,
            "nickname": "",
            "vipType": 0,
            "avatarImgId": 1400777825738419,
            "defaultAvatar": false,
            "djStatus": 0,
            "accountStatus": 0,
            "backgroundImgIdStr": "3427177752524551",
            "backgroundUrl": "http://p1.music.126.net/LS96S_6VP9Hm7-T447-
X0g==/3427177752524551.jpg",
            "signature": "",
            "authStatus": 0}
        "tracks": [...]
    }
}
```


(2) The format of song


```
{
    "id": 29738501,
```

```
"name": "",
"duration": 174001,
"hearTime": 0,
"commentThreadId": "R_SO_4_29738501",
"score": 40,
"mvid": 0,
"hMusic": null,
"disc": "",
"fee": 0,
"no": 1,
"rtUrl": null,
"ringtone": null,
"rtUrls": [],
"rurl": null,
"status": 0,
"ftype": 0,
"mp3Url": "http://m2.music.126.net/vrVa20wHs8iIe0G8Oe7I9Q==/3222668581877701.mp3",
"audition": null,
"playedNum": 0,
"copyrightId": 0,
"rtype": 0,
"crbt": null,
"popularity": 40,
"dayPlays": 0,
"alias": [],
"copyFrom": "",
"position": 1,
"starred": false,,
"starredNum": 0
"bMusic": {
    "name": "",
    "extension": "mp3",
    "volumeDelta": 0.0553125,
    "sr": 44100,
    "dfsId": 3222668581877701,
    "playTime": 174001,
    "bitrate": 96000,
    "id": 52423394,
    "size": 2089713
},
"lMusic": {
    "name": "",
    "extension": "mp3",
    "volumeDelta": 0.0553125,
    "sr": 44100,
    "dfsId": 3222668581877701,
```

```
        "playTime": 174001,
        "bitrate": 96000,
        "id": 52423394,
        "size": 2089713
    },
    "mMusic": {
        "name": "",
        "extension": "mp3",
        "volumeDelta": -0.000265076,
        "sr": 44100,
        "dfsId": 3222668581877702,
        "playTime": 174001,
        "bitrate": 128000,
        "id": 52423395,
        "size": 2785510
    },
    "artists": [
        {
        "img1v1Url": "http://p1.music.126.net/6y-UleORITEDbvrOLV0Q8A==/5639395138885805.jpg",
        "name": "",
        "briefDesc": "",
        "albumSize": 0,
        "img1v1Id": 0,
        "musicSize": 0,
        "alias": [],
        "picId": 0,
        "picUrl": "http://p1.music.126.net/6y-UleORITEDbvrOLV0Q8A==/5639395138885805.jpg",
        "trans": "",
        "id": 122455
        }
    ],
    "album": {
        "id": 3054006,
        "status": 2,
        "type": null,
        "tags": "",
        "size": 69,
        "blurPicUrl": "http://p1.music.126.net/2XLMVZhzVZCOunaRCOQ7Bg==/3274345629219531.jpg",
        "copyrightId": 0,
        "name": "248",
        "companyId": 0,
        "songs": [],
        "description": "",
        "pic": 3274345629219531,
        "commentThreadId": "R_AL_3_3054006",
        "publishTime": 1388505600004,
```

```
        "briefDesc": "",
        "company": "",
        "picId": 3274345629219531,
        "alias": [],
        "picUrl": "http://p1.music.126.net/2XLMVZhzVZCOunaRCOQ7Bg==/3274345629219531.jpg",
        "artists": [
        {
            "img1v1Url": "http://p1.music.126.net/6y-
UleORITEDbvrOLV0Q8A==/5639395138885805.jpg",
            "name": "",
            "briefDesc": "",
            "albumSize": 0,
            "img1v1Id": 0,
            "musicSize": 0,
            "alias": [],
            "picId": 0,
            "picUrl": "http://p1.music.126.net/6y-UleORITEDbvrOLV0Q8A==/5639395138885805.jpg",
            "trans": "",
            "id": 122455
        }
        ],
        "artist": {
        "img1v1Url": "http://p1.music.126.net/6y-UleORITEDbvrOLV0Q8A==/5639395138885805.jpg",
        "name": "",
        "briefDesc": "",
        "albumSize": 0,
        "img1v1Id": 0,
        "musicSize": 0,
        "alias": [],
        "picId": 0,
        "picUrl": "http://p1.music.126.net/6y-UleORITEDbvrOLV0Q8A==/5639395138885805.jpg",
        "trans": "",
        "id": 0
        }
    }
}
```


(3) Import Data

The json file is about 16GB and hard to deal with, we could import the json file and save what we need in a csv file. We extract 8 characteristics: playlist_id, playlist_name, playlist_tags, playlist_subscribed_count, song_id, song_name, artists, and popularity.

```
import json
import pandas as pd
import time
```

```
def parse_song_list(song_list):
    data = json.loads(song_list)
    name = data['result']['name']
    tags = ",".join(data['result']['tags'])
    subscribed_count = data['result']['subscribedCount']
    playlist_id = data['result']['id']
    song_info = ''
    songs = data['result']['tracks']
    for song in songs:
        song_info += "|" + str(song['id']) + "," + song['name'] + ","+ song['artists'][0]
['name'] + "," + str(song['popularity'])
    return pd.DataFrame({'name': [name], 'tags': [tags], 'subscribedCount': [subscribed_count],
'playlist_id': [playlist_id],'tracks': [song_info]})


def parse_file(file_in, file_out):
    result = pd.DataFrame(columns=['name', 'tags', 'subscribedCount', 'playlist_id', 'tracks'])
    count = 0
    with open(file_in, 'r', encoding='utf-8') as f:
        for line in f:
            result = result.append(parse_song_list(line),ignore_index=True)
            count += 1
            if count % 500 == 0:
                print(time.strftime('%Y.%m.%d %H:%M:%S ',time.localtime(time.time()))+"Dealing
with Playlist No. " + str(count))
    result.to_csv(file_out,encoding='utf-8')


if __name__=="__main__":
    file_in = "/Users/chen/学习/项目资料/机器学习实战项目/第01课/playlist_detail_all.json"
    file_out = "/Users/chen/学习/项目资料/机器学习实战项目/第01课/163_music_playlist.csv"
    parse_file(file_in,file_out)
```

(4) Data Process

After getting useful information, we need to finish data process:

```
import pandas as pd


data_list_1 = pd.read_csv("/Users/chen/学习/项目资料/机器学习实战项目/第01
课/163_music_playlist.csv")
data_1 = data_list_1.drop('tracks', axis=1).join(data_list_1['tracks'].str.split('|',
expand=True).stack().reset_index(level=1, drop=True).rename('Songs'))
data_1 = data_1.dropna(subset=['Songs'],how='any')
```

```
data_1 = data_list_1.drop('Songs', axis=1).join(data_list_1['Songs'].str.split(',',
expand=True).stack().reset_index(level=1, drop=True))


data_list_2 = data_1
data_2 = data_list_2.dropna(subset=['Songs'],how='any')


data_list_3 = data_2

data_3 = data_list_3['Songs'].str.split(',',expand=True)
data_list_3['song_id']=data_3[0]
data_list_3['song_name']=data_3[1]
data_list_3['artists']=data_3[2]
data_list_3['popularity']=data_3[3]

data_list_3 = data_list_3.drop('Songs', axis=1)
data_list_3 = data_list_3.drop('Unnamed: 0', axis=1)
data_list_3 = data_list_3.drop('Unnamed: 0.1', axis=1)
data_list_3 = data_list_3.drop('Unnamed: 0.1.1', axis=1)

data_list_3.to_csv("/Users/chen/学习/项目资料/机器学习实战项目/第01
课/Processed_data_set.csv",encoding='utf-8')
```

And the processed data is shown as follows:

	Unnamed: 0	name	tags	subscribedCount	playlist_id	song_id	song_name	artists	popularity
0	0	☀清新夏日☀清新男声控|我想漂浮感受磁场	欧美,电子,另类/独立	11355	423245641	414691355	Lost (As I Am)	Superwalkers	80.0
1	1	☀清新夏日☀清新男声控|我想漂浮感受磁场	欧美,电子,另类/独立	11355	423245641	410802620	Next Escape	Viceroy	100.0
2	2	☀清新夏日☀清新男声控|我想漂浮感受磁场	欧美,电子,另类/独立	11355	423245641	419549837	Silhouette	Goldroom	60.0
3	3	☀清新夏日☀清新男声控|我想漂浮感受磁场	欧美,电子,另类/独立	11355	423245641	419485281	Feel My Love	Jake Quickenden	45.0
4	4	☀清新夏日☀清新男声控|我想漂浮感受磁场	欧美,电子,另类/独立	11355	423245641	412016420	Hit It	Teasley	65.0
5	5	☀清新夏日☀清新男声控|我想漂浮感受磁场	欧美,电子,另类/独立	11355	423245641	421160284	Catch U	DEAMN	85.0
6	6	☀清新夏日☀清新男声控|我想漂浮感受磁场	欧美,电子,另类/独立	11355	423245641	420513422	Breathe It In	Cyrus Thomas	30.0
7	7	☀清新夏日☀清新男声控|我想漂浮感受磁场	欧美,电子,另类/独立	11355	423245641	420500507	Tropical Suneo	Alex Parker	80.0
8	8	☀清新夏日☀清新男声控|我想漂浮感受磁场	欧美,电子,另类/独立	11355	423245641	421203274	New Age	Softer Still	55.0
9	9	☀清新夏日☀清新男声控|我想漂浮感受磁场	欧美,电子,另类/独立	11355	423245641	416933311	Moments	KEV	25.0