MUSICBOX RECOMMENDATION SYSTEM

JIAHUI TAO

AGENDA

- Data Loading
- Data Cleaning
- Data Exploration
- Model Evaluation
- Benchmarks
- ALS Explicit Model
- ALS Implicit Model

DATA LOADING

- Assign schema for 2 data frames: **play** and **download**
- Load source data into Spark (Databricks) and compile into data frames
- Extract date from the file name and create a new feature as "date"
- Check the size of original data
  - Play: 164,081,952 rows
  - Download: 7,746,631 rows

Play
uid
device
song_id
song_type
song_name
singer
play_time
song_length
paid_flag
date

Download
uid
device
song_id
song_name
singer
paid_flag
date

DATA CLEANING- INITIAL PROCESSING

- Trim each column to remove extra space
- Cast “play_time”, “song_length” to int
- Cast “date” to date
- Remove rows where “uid” or “play_time” is null
- Remove rows where “song_name” and “singer” are in one column (data was not separated properly)
- Process “uid”
  - substring from misformatted uid’s
  - define a function to trim the special characters from substring
  - cast “uid” to int

DATA CLEANING- REINDEX SONG ID'S

Issue: mis-mapping from song_name to song_id

- **play** has 1,679,872 song_id's, 1,760,049 song_name's and 2,255,855 song_name&singer
- 32,754 song_id's have more than 1 song_name (1.95% of total unique sid's) with 67,150,874 rows (40.9% of all rows)
- 265,937 song_name&singer with >1 song_id, which is 11.8% of total song_name&singer

Solution: reindex song_name & singer

- select song_name, and singer from **play** and **download**
- remove rows where song_name or singer is null or ""
- identify a song by song_name and singer, and input with new sid
- save the song_name, singer, sid as "sid.parquet" for future reference
- join play and download table with 'sid' table respectively

sid	play_sid	down_sid
2,309,630	162,451,873	6,592,790

DATA CLEANING-SONG_LENGTH CONSISTENCY

- The "song_length" for each song is not consistent.
- Use the most frequent song_length as the song_length for each sid
- Songs with song_length only equal to 0 are removed
- 2,146,170 songs remains

```
%sql
select sid, song_length
from (
     select sid, song_length,
            rank() over (partition by sid order by count(*) desc) as rank
     from play_sid
     where song_length>0
     group by sid, song_length
        )r
where r.rank=1
order by sid
```

DATA EXPLORATION

- identify robots
  - there are 54 days in records
  - 99% users have records of 1843 or less, which is 34 songs a day
  - 99.5% users have records of 2489 or less, which is 46 songs a day
  - 99.9% users have records of 4678 or less, which is 87 songs a day
  - IQR=139-8=131 139+131*1.5=335.5
  - I decided to remove uid's with 2489 or more records (seen as robots)
- after removing the robots
  - play_filtered: 112,836,361 rows; 797,170 users; 1,913,909 songs
  - down_filtered: 4,920,514 rows; 214,947 users; 535,303 songs

MODEL EVALUATION

- Root Mean Squared Error (RMSE): Used for explicit models
- Expected Percentile Ranking (EPR): Evaluate how well the recommendation model rank the songs
  - From backup paper for pyspark ALS implicit model, "Collaborative Filtering for Implicit Feedback Datasets"

$$EPR = \frac{\sum_{u,i} r_{ui} \times rank_{ui}}{\sum_{u,i} r_{ui}}$$

r_{ui} is the actual rating of user u for song i in the test set

$rank_{ui}$ is the percentile ranking of song i in the recommended order for user u; the most recommended is 0%, and the least recommended is 100%

uid	sid	rating	prediction	rank
5000	1046	7	0.98	0%
5000	3022	4	0.86	16.7%
5000	4056	6	0.77	33.3%
5000	3200	3	0.68	50%
5000	4433	0	0.55	66.7%
5000	1002	2	0.43	83.3%
5000	5097	1	0.20	100%

$$EPR = \frac{7 \times 0\% + 4 \times 16.7\% + 6 \times 33.3\% + 3 \times 50\% + 0 \times 66.7\% + 2 \times 83.3\% + 1 \times 100\%}{7 + 4 + 6 + 3 + 0 + 2 + 1}$$

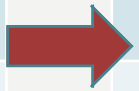
$$= 29.70\%$$

BENCHMARKS

- Popularity-based Model
  - Order songs by the number of users listened (popularity)
  - Recommend to all users the same list of songs
  - A robust model

- Best EPR possible
  - Assume all the songs recommended by the model is ordered as they actually listened (from most frequent to least frequent)

uid	sid	rating	prediction	rank
5000	1046	7	0.98	0%
5000	3022	4	0.86	16.7%
5000	4056	6	0.77	33.3%
5000	3200	3	0.68	50%
5000	4433	0	0.55	66.7%
5000	1002	2	0.43	83.3%
5000	5097	1	0.20	100%



uid	sid	rating	rank
5000	1046	7	0%
5000	3022	4	33.3%
5000	4056	6	16.7%
5000	3200	3	50%
5000	4433	0	100%
5000	1002	2	66.7%
5000	5097	1	83.3%

$$EPR = 26.09\%$$

EXPLICIT MODEL

Data Processing:

- Use all data available
- If play_time/song_length >= threshold, count as 1
  - compared threshold = 60% and 80%; decided to use 80%
- Scoring: bin the play count to 4 levels
  - 0-dislike count=0-1 0-50%
  - 1-listened count=2-3 50%-90%
  - 2-like count=4-6 90%-97.5%
  - 3-very like count>6 97.5%-100%
- Download: score +1, but capped by 3
- Randomly split all data by 8:2 to train and test

EXPLICIT MODEL - TUNING

from pyspark.ml.recommendation import ALS

Parameters

- **maxIter:** the maximum number of iterations to run.
- **rank:** the number of latent factors in the model.
- **regParam:** the regularization parameter in ALS.
- **nonnegative:** whether or not to use nonnegative constraints for least squares.
- **implicitPrefs:** whether to use the explicit feedback ALS variant or one adapted for implicit feedback data.
- **coldStartStrategy**: set to 'drop' to ensure we don't get NaN evaluation metrics

Benchmarks: Popularity Model EPR: 43.51% Best EPR: 7.07%

maxIter	rank	regParam	nonnegative	implicitPrefs	coldStart Strategy	RMSE	EPR
7	50	0.08	False	False	drop	0.6535	34.39%

IMPLICIT MODEL

- Transferring the raw observations (r_{ui}) into two separate magnitudes with distinct interpretations: preferences (p_{ui}) and confidence levels (c_{ui}).

$$p_{ui} = \begin{cases} 1 & r_{ui} > 0 \\ 0 & r_{ui} = 0 \end{cases} \qquad c_{ui} = 1 + \alpha \log(1 + r_{ui}/\varepsilon)$$

- Use ALS to find the user matrix and item matrix which minimize

$$\sum_{u,i} c_{ui}(p_{ui} - x_u^T \cdot y_i)^2 + \lambda(\sum_u \|x_u\|^2 + \sum_i \|y_i\|^2)$$

- The predicted preference of user u for song i is

$$\widehat{p_{ui}} = x_u^T \cdot y_i$$

IMPLICIT MODEL

I. week 13-17 as training data, week 18 as testing data

- Benchmarks: Popularity Model EPR: 45.99% Best EPR: 13.55%

maxIter	rank	regParam	alpha	nonnegative	implicitPrefs	coldStart Strategy	EPR
7	50	0.1	100	False	True	drop	43.59%

II. Use all data available, random split by 80%:20%

- Benchmarks: Popularity Model EPR: 43.32% Best EPR: 12.92%

maxIter	rank	regParam	alpha	nonnegative	implicitPrefs	coldStart Strategy	EPR
7	50	0.1	100	False	True	drop	33.04%

THANK YOU !