P34: Build a classification model and recommendation model using Spotify data

Machine Learning 2021

Final Project Presentation

14th Dec. 2021
MEET THE TEAM!!

Neha Sharma
Junior Researcher/Phd, Computational Network Science Group

Muhammad Awais Ali
Junior Researcher/Phd, Process Mining Research Group,

Farooq Ayoub Dar
Junior Researcher/Phd, Pervasive and Distributed Systems Group
1) **Build a classifier:** Use different machine learning algorithms to build a classifier for our given data and predict whether a user will like a song or not.

1) **Build a recommendation system:** Develop a recommender system based on songs features to create suggestions based on the similarities of the songs.

*Motivation:* Use machine learning to gain insight for the things I use daily, such as music. Personal opinion(Spotify recommendations lack diversity).*
ROADMAP:

DATA Scraping
Using Spotipy Web API to collect user’s data such as liked songs playlist and disliked songs playlist.

EDA
Exploratory data analysis to understand the data better.

Feature engineering
Based on EDA, identify important features

Classification model
Logistic regression, Decision Tree, Random Forest, Boosting methods, (default and hyperparameter tuned models)

Recommender Model
Content Based Filtering using song features.
EXPLORATORY DATA ANALYSIS & FEATURE ENGINEERING:

**DATA Scraping**
Using Spotipy Web API to collect user’s data such as liked songs playlist and disliked songs playlist.

**EDA**
Exploratory data analysis to understand the data better.

**Feature engineering**
Based on EDA, identify important features

**Classification model**
Logistic regression, Decision Tree, Random Forest, Boosting methods,(default and hyperparameter tuned models)

**Recommender Model**
Content Based Filtering using song features.
EXPLORATORY DATA ANALYSIS & FEATURE ENGINEERING(I):

TARGET[0,1]: Describes if a user likes (1) or does not like (0) a given song

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 11038 entries, 0 to 13880
Data columns (total 14 columns):
#    Column    Non-Null Count   Dtype
---  ------     ---------------   -----  
0    danceability  11038 non-null   float64
1     energy    11038 non-null   float64
2       key    11038 non-null    int64
3   loudness    11038 non-null   float64
4       mode    11038 non-null    int64
5  speechiness    11038 non-null   float64
6  acousticsness    11038 non-null   float64
7  instrumentalness  11038 non-null   float64
8      liveness    11038 non-null   float64
9    valence    11038 non-null   float64
10     tempo    11038 non-null   float64
11 duration_ms     11038 non-null    int64
12  time_signature    11038 non-null    int64
13       target    11038 non-null    int64
dtypes: float64(9), int64(5)
memory usage: 1.3 MB
```
EXPLORATORY DATA ANALYSIS & FEATURE ENGINEERING(II):

Categorical Features:
1. ‘key’: Integers map to pitches using standard Pitch Class notation
2. ‘mode’: Major is represented by 1 and minor is 0.
3. ‘time_signature’: how many beats are in each bar

Continuous Features:
1. ‘danceability’: A value of 0.0 is least danceable and 1.0 is most danceable.
2. ‘energy’: 0.0 to 1.0 and represents a perceptual measure of intensity and activity
3. ‘loudness’: range between -60 and 0 db
4. ‘speechiness’: detects the presence of spoken words in a track
5. ‘acousticness’: A confidence measure from 0.0 to 1.0 of whether the track is acoustic
6. ‘instrumentalness’: Predicts whether a track contains no vocals
7. ‘liveness’: Detects the presence of an audience in the recording
8. ‘valence’ : 0.0 to 1.0 describing the musical positiveness conveyed by a track
9. ‘tempo’: the speed or pace of a given piece and derives directly from the average beat duration
10. ‘duration_ms’
• **Features Correlation:** linear correlation between features with Pearson correlation coefficient. Linear correlation can be positive or negative.

- Features Correlation:
  - valence and danceability
  - valence and energy
  - acousticness and loudness
  - acousticness and energy
  - loudness and energy

---

---
Principal Component Analysis:
- Variance ratio: ([0.20824602, 0.09807221])
- Both liked and disliked classes are not distinguishable. There is no clear boundary between these classes.

Mann Whitney test and get p-value to identify which features have different distributions when comparing like and not_like songs.

<table>
<thead>
<tr>
<th>feature</th>
<th>pvalue</th>
</tr>
</thead>
<tbody>
<tr>
<td>energy</td>
<td>6.573271e-160</td>
</tr>
<tr>
<td>speechiness</td>
<td>2.951094e-75</td>
</tr>
<tr>
<td>acousticness</td>
<td>9.733286e-62</td>
</tr>
<tr>
<td>danceability</td>
<td>4.551270e-49</td>
</tr>
<tr>
<td>duration_ms</td>
<td>5.010833e-44</td>
</tr>
<tr>
<td>valence</td>
<td>7.742325e-41</td>
</tr>
<tr>
<td>loudness</td>
<td>1.646322e-37</td>
</tr>
</tbody>
</table>
CLASSIFICATION MODELS:

DATA Scraping
Using Spotipy Web API to collect user’s data such as liked songs playlist and disliked songs playlist.

EDA
Exploratory data analysis to understand the data better.

Feature engineering
Based on EDA, identify important features

Classification model
Logistic regression, Decision Tree, Random Forest, Boosting methods, (default and hyperparameter tuned models)

Recommender Model
Content Based Filtering using song features.
## CLASSIFICATION MODELS:

<table>
<thead>
<tr>
<th>Classifiers</th>
<th>Mean F1-Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logistic Regression</td>
<td>0.33</td>
</tr>
<tr>
<td>Decision Tree</td>
<td>0.62</td>
</tr>
<tr>
<td>Random Forest</td>
<td>0.66</td>
</tr>
<tr>
<td>AdaBoost</td>
<td>0.65</td>
</tr>
<tr>
<td>XGBoost</td>
<td>0.63</td>
</tr>
</tbody>
</table>
CLASSIFICATION MODELS(I): HYPERPARAMETER TUNING:

- XGBoost is performing best compare to other models after parameter tuning

<table>
<thead>
<tr>
<th>Classifiers</th>
<th>Best Parameters</th>
<th>Mean F1-Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Forest</td>
<td>max_depth: 7, min_samples_leaf: 1, min_samples_split: 5, n_estimators: 300</td>
<td>0.67</td>
</tr>
<tr>
<td>AdaBoost</td>
<td>max_depth: 4, min_samples_leaf: 10, learning_rate: 0.1, n_estimators: 50</td>
<td>0.66</td>
</tr>
<tr>
<td>XGBoost</td>
<td>n_estimators: 223, min_child_weight: 5, max_depth: 4, learning_rate: 0.05</td>
<td>0.70</td>
</tr>
</tbody>
</table>
RECOMMENDER MODEL:

DATA Scraping
Using Spotipy Web API to collect user’s data such as liked songs playlist and disliked songs playlist.

EDA
Exploratory data analysis to understand the data better.

Feature engineering
Based on EDA, identify important features

Classification model
Logistic regression, Decision Tree, Random Forest, Boosting methods,(default and hyperparameter tuned models)

Recommender Model
Content Based Filtering using song features.
RECOMMENDER MODEL:

- Content based Recommendation
- K-Nearest Neighbours
- Distance Metric: Manhattan
  - Is consistently more preferable than the euclidean distance for higher dimensional data
RECOMMENDER MODEL:

my_favorite = 'Imagine Dragons'
make_recommendation(
    model_knn=model_knn,
    data=song_user_mat_sparse,
    fav_song=my_favorite,
    mapper=song_to_idx,
    n_recommendations=5)

You have input song: Imagine Dragons

Recommendation system start to make inference
......

Recommendations for Imagine Dragons:
1: The Beatles:The Night Before - Remastered, with distance of 0.1383057958348184
2: Hanif Shaikh Vijay Lama:Bekarar, with distance of 0.13539280948938714
3: AURORA:The River, with distance of 0.1308975713061168
4: Ellie Goulding:Burn, with distance of 0.1280178036917885
5: Various Artists:Meri Maa, with distance of 0.12607322857661585
Thank you for listening

Spotify