Machine Learning-Based Music Genre Classification on Spotify Data

Ayush Saha (20BDS0273), Prakhar (20BCE2113), Vatsal Khushalani (20BDS0144) , Ayush Agarwal (20BBS0193)

Introduction

Overview

“Machine Learning-Based Music Genre Classification on Spotify Data" aims to develop a system that automatically classifies music into different genres using machine learning techniques. The project utilizes a dataset obtained from Spotify, which contains various audio features such as tempo, energy, danceability, and acoustics, among others.

The objective of this project is to train a machine learning model using this dataset to accurately predict the genre of a given song. The model will learn patterns and relationships between the audio features and the corresponding genre labels present in the dataset. By analyzing the audio characteristics of songs, the model will be able to make predictions on new, unseen songs and assign them to specific genres.

The project will involve several steps, including data preprocessing, feature extraction, model training, and evaluation. Various machine learning algorithms such as decision trees, random forests, or neural networks can be used for genre classification. The performance of the model will be assessed using metrics such as accuracy, precision, recall, and F1 score.

By successfully developing a reliable genre classification model, this project can have several applications. It can be used by music streaming platforms like Spotify to enhance music recommendations and personalized playlists for users. It can also aid in music metadata tagging, music indexing, and genre-based music analysis. Additionally, it can serve as a foundation for further research in music information retrieval and contribute to advancements in music classification algorithms.

Purpose

Music genre classification categorizes music into distinct styles, aiding organization, discovery, recommendations, marketing, research, and understanding of cultural and historical context. It facilitates navigation, personalized recommendations, and targeted promotion while providing a framework for the analysis and exploration of diverse musical expressions.

Literature Survey

Existing Solution

1. **Music Genre Classification - Rajeeva Shreedhara Bhat ,Rohit B. R. ,Mamatha K. R.**
   This paper discusses the application of machine learning, specifically

Convolutional Neural Networks (CNN), to categorize music genres. The authors address the challenge of organizing large music databases and the evolving nature of genres, arguing manual classification is tedious. The CNN model is trained using the GTZAN dataset with Mel Spectrum representations of each track, achieved using Python's libROSA package. The findings demonstrate the feasibility of automated music genre classification, paving the way for larger datasets and diverse track formats. Future work includes updating the model to accommodate genre evolution and incorporating mood-based music recommendation systems.

2. **Music Genre Classification using Machine Learning Techniques - Hareesh Bahuleyan**
   This research study investigates machine learning (ML) applications for music genre classification using the AudioSet dataset. Two methods are proposed: one involves generating a MEL spectrogram of the audio signal, treating it as an image, and using a convolutional neural network (CNN) for classification; the second extracts time and frequency domain features from the audio signals, then applies traditional ML classifiers (Logistic Regression, Random Forests, Gradient Boosting, SVMs) for classification. Results show CNNs outperformed feature-engineered models, and ensemble models were beneficial. Future studies could focus on pre-processing noisy data to improve model performance.

Proposed Solution

Our proposed solution aims to streamline the song identification process by allowing users to input a Spotify link. This approach enhances user interaction by eliminating the need for manual entry of individual audio features and metadata values. Through our web app, we utilize the Spotify API to gather the audio features of the song, which are then utilized by our model to predict the genre of the song.

Hardware Requirements

1. **CPU**: A modern multi-core CPU (e.g., Intel Core i5 or higher) is typically sufficient for running classification models, especially for smaller datasets and simpler models.

2. **GPU**: For more computationally intensive models, especially those based on deep learning techniques, having a compatible GPU can significantly speed up the training and inference process. NVIDIA GPUs, such as the GeForce RTX series or Tesla GPUs, are commonly used for deep learning tasks.

3. **RAM**: Sufficient RAM is crucial for handling large datasets. The amount of RAM required depends on the size of the dataset and the model's memory requirements. Generally, 8 GB or more is recommended, but for

larger datasets or complex models, 16 GB or higher is preferable.

4. **Storage**: Adequate storage space is necessary for storing the dataset, trained models, and any intermediate results. Solid-state drives (SSDs) are preferred over traditional hard disk drives (HDDs) for faster data access.

5. **Network**: If the classification model involves utilizing cloud-based services or training on distributed systems, a reliable and high-speed internet connection is required.

Additional Considerations: Depending on the project requirements, additional hardware components like specialized hardware accelerators (e.g., Tensor Processing Units - TPUs) or memory-optimized systems may be necessary.

Software Requirements

1. **Programming Language**: Python

2. **Machine Learning Libraries**: scikit-learn, TensorFlow, or PyTorch

3. **Data Processing Libraries**: pandas, NumPy

4. **Text or Audio Processing Libraries**: Spotipy (for audio processing in music classification projects)

5. **Development Environment**: PyCharm, Jupyter Notebook, or Visual Studio Code.

6. **Flask**: A lightweight web framework for building web applications in Python

Additional Libraries: Matplotlib or Plotly (for visualization), scikit-learn's metrics module (for model evaluation), and any other specific libraries you may need for your project.

Experimental Investigations

While creating the machine learning model to predict the genre of the song, several experimental investigations were conducted to improve the model's performance.

Feature Selection: All different feature sets derived from Spotify data were explored, which included audio features (acoustics, danceability, energy, instrumentality, tone, etc.), and metadata (artist name, album name, release year, song name, etc.). The lyrics data of the song was not included. Experiments with different combinations of features were conducted to determine which subset provided the most predictive power.

Feature Engineering: Create new features based on knowledge or hypotheses about things that might influence the genre prediction. Transformations were applied to the existing data. This involved EDA, encoding categorical data and scaling the data.

Model Selection: Experiments with different machine learning algorithms suitable for classification tasks were conducted. These included decision trees, random forests, support vector machines, and neural networks. Their performance was compared

in terms of accuracy, precision, recall, and F1-score to identify the most effective model for your task. Random Forest was the selected model with the best scores.

Hyperparameter Tuning: Fine-tuning of parameters of random forest algorithm to optimize its performance and find the best configuration for your model.

Handling Class Imbalance: Analysis of the distribution of genres in your dataset and address of any class imbalance issues was conducted using techniques like oversampling, undersampling, and using class weights during training to help ensure your model does not favor dominant genres and performs well across all genres.

Advantages

Organization and Discovery: Music genre classification helps organize and categorize vast amounts of music, making it easier for users to navigate and discover new music based on their preferences. It enables efficient searching and browsing through specific genres, allowing users to explore and expand their musical horizons.

Personalized Recommendations: Genre classification forms the foundation for accurate music recommendations. By understanding a user's preferred genre or genres, recommendation systems can suggest similar music that aligns with their tastes. This enhances the user experience by introducing them to new artists and songs they are likely to enjoy.

Targeted Marketing and Promotion: Genres are utilized as marketing tools in the music industry. By classifying music into genres, record labels, artists, and streaming platforms can effectively target specific audience segments. This enables personalized marketing and promotion strategies, ensuring that music reaches the intended audience and increasing the likelihood of engagement and consumption.

Cultural and Historical Understanding: Genres reflect the cultural and historical context of music. Through genre classification, we gain insights into the evolution of musical styles, the influence of different eras and regions, and the development of subcultures. It helps us appreciate and understand the rich tapestry of musical heritage across time and geography.

Music Research and Analysis: Music genre classification facilitates research and analysis in various fields such as musicology, sociology, and cultural studies. Researchers can study specific genres, analyze their characteristics, explore interconnections between genres, and delve into the cultural, social, and artistic significance of different musical styles.

Enhanced User Experience: Genre classification enhances the user experience by providing a structure and framework for organizing and exploring music. Users can easily identify and access the type of music they are in the mood for, leading to a more enjoyable and tailored listening experience.

Music Industry Insights: Genre classification provides valuable insights to the music industry regarding consumer preferences, trends, and market demands. This information assists in decision-making processes related to artist signings, content curation, playlist creation, and strategic planning.

Disadvantages

Subjectivity and Ambiguity: Music genre classification is inherently subjective and can be ambiguous. Determining the genre of a particular song can be challenging, as many songs incorporate elements from multiple genres or defy traditional genre boundaries. Disagreements can arise due to personal preferences and cultural perspectives, leading to inconsistencies in genre classification.

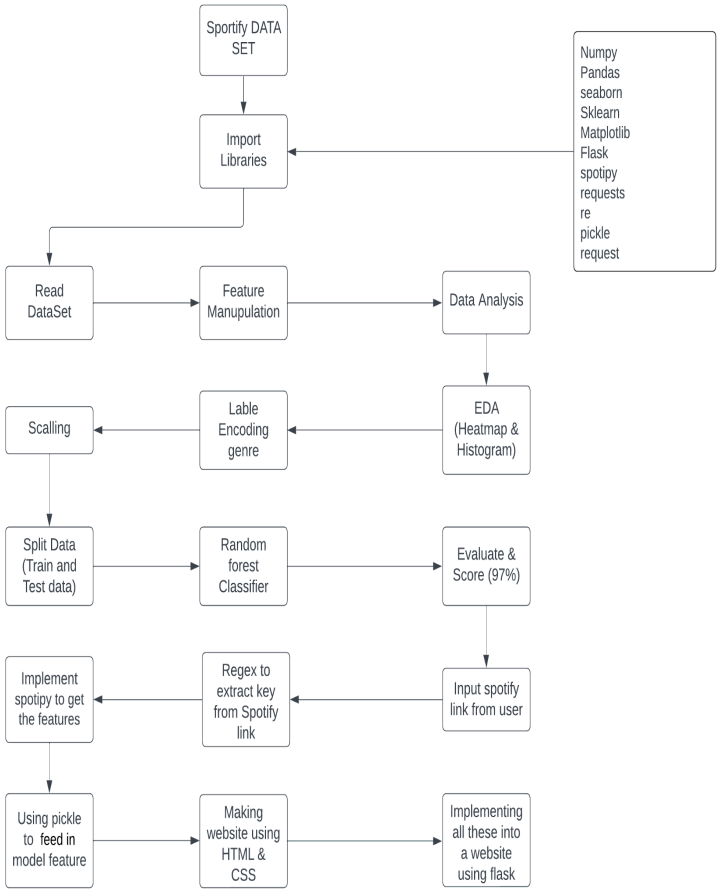
Genre Limitations: Classifying music into genres can lead to oversimplification and overlook the complexity and diversity within artists and songs. It may not capture the nuances and unique characteristics of individual compositions, potentially reducing appreciation for unconventional or boundary-pushing music that does not neatly fit into predefined genres.

Evolving and Hybrid Genres: Music is constantly evolving, and new genres emerge as artists experiment and fuse different styles. Traditional genre classifications may struggle to keep up with these changes and accurately capture the characteristics of emerging or hybrid genres. This can result in misclassification or the creation of new subgenres to accommodate evolving trends.

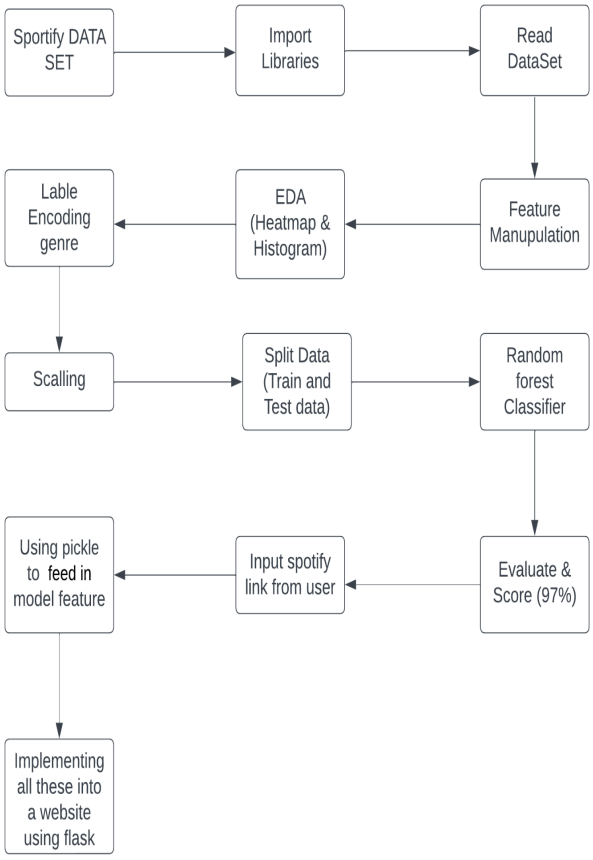
Stereotyping and Bias: Genre classification can perpetuate stereotypes and biases. Certain genres may be associated with specific demographics or cultural backgrounds, leading to assumptions or generalizations. This can reinforce stereotypes and limit opportunities for artists who do not conform to the perceived expectations of their assigned genre.

Over-Reliance on Genre Labels: Relying heavily on genre labels can lead to a shallow understanding of music. Listeners may overlook individual artistic expression and focus solely on genre categorizations, potentially missing out on diverse and innovative musical experiences. It can also contribute to a narrow-minded approach to music, where listeners may dismiss certain genres without exploring their unique qualities.

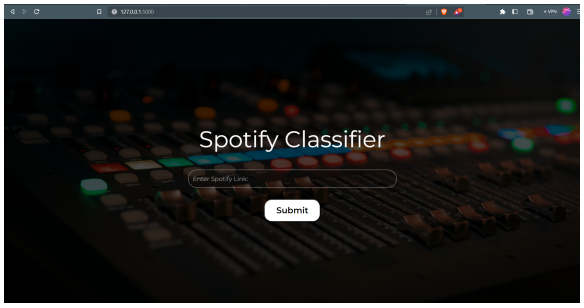
BlockDiagram:

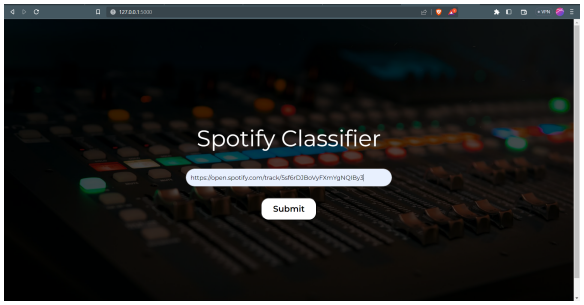


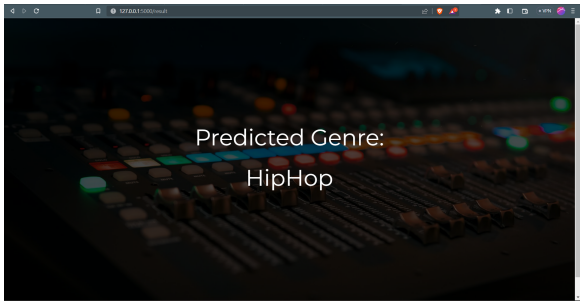
Flowchart:



Result







Applications

1. Music Recommendation System
2. Music Streaming Platform
3. Music Marketing and Promotions
4. Music Analysis and Research
5. Music Licensing and Copyright
6. Radio and Playlist Curation
7. Music Education and Teaching

Conclusion

Our project's ultimate goal is to enable users to input the Spotify URL of any song and receive a genre prediction. To achieve this, we utilize the Spotify API to gather audio features and metadata for the given song. The Flask web environment is utilized to create a browser interface for gene prediction. Currently, our model achieves a 96.99% accuracy on the training data and 81.21% accuracy on the test data. At present, our model can predict the genre of a song from a dataset of audio features encompassing 15 different musical genres.

Future Scope

In order to expand the project's potential, it can incorporate additional data sources like lyrics databases or music reviews. This integration aims to enhance the dataset and potentially improve the accuracy of genre prediction. The merging and integration of these external sources with the existing Spotify data will require experimentation. Furthermore, the web app will be updated to include audio detection capabilities, enabling it to identify songs from audio recorded through the device's microphone. This functionality can be achieved by integrating external APIs into the Flask app. Additionally, further experimentation with various algorithms is necessary to identify any potential algorithms that can outperform the current model in terms of accuracy.

Bibliography

1. https://www.researchgate.net/publication/324218667_Music_Genre_Classification_using_Machine_Learning_Techniques
2. https://www.internationaljournalssrg.org/IJCMS/2020/Volume7-Issue1/IJCMS-V7I1P102.pdf
3. https://towardsdatascience.com/spotify-api-audio-features-5d8bcbd780b2
4. https://stackoverflow.com/questions/71803776/spotify-audio-feautures-in-python
5. https://developer.spotify.com/
6. https://www.kaggle.com/datasets/mrmorj/dataset-of-songs-in-spotify?select=genres_v2.csv
7. https://pythonhow.com/python-tutorial/flask/Adding-CSS-styling-to-your-website/#:~:text=CSS%20stylesheets%20are%20considered%20static,and%20should%20be%20named%20static
8. https://thinkinfi.com/flask-adding-html-and-css

Appendix

Source Code

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn import preprocessing
import pandas as pd
df= pd.read_csv('C://Users//Ayush//Desktop//genres_v2.csv')
df.head()
df=   df.drop(['id','uri',   'track_href',   'analysis_url',   'song_name',
'Unnamed: 0', 'title' ,'type'], axis=1)
df['duration_min'] = df['duration_ms']/60000
df=df.drop(['duration_ms'], axis=1)
df.head()
df.describe()
df.isnull().sum()
df = df.drop_duplicates()
df["genre"].value_counts()
print('There  are  {df.shape[0]}  rows  and  {df.shape[1]}  columns  in  the
dataset.\n')
import matplotlib.pyplot as plt
numeric = df._get_numeric_data()
genre = df['genre']
num_hist = numeric.hist(layout=(3,5), figsize = (20,10))
plt.show()
grouped_genre = df.groupby('genre')
for col in numeric.columns:
    fig,ax = plt.subplots()

    for i, d in grouped_genre:
        d[col].hist(alpha = 0.4, ax=ax, label = i, figsize = (10,4))
        ax.set_title(col)

    ax.legend()
    plt.show()
from sklearn.preprocessing import LabelEncoder
df['genre'] = LabelEncoder().fit_transform(df['genre'])
df['genre'].value_counts()
```

```
import seaborn as sns
df['genre'] = LabelEncoder().fit_transform(df['genre'])
corr = df.corr()
sns.heatmap(corr)

X = df.drop('genre', axis=1)
y = df['genre']
X.shape
y.shape
from sklearn.preprocessing import StandardScaler
X= StandardScaler().fit_transform(X)
from imblearn.over_sampling import SMOTE

smote = SMOTE()
X, y = smote.fit_resample(X, y)
X.shape
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42, shuffle=True)
from sklearn import metrics
from sklearn.ensemble import RandomForestClassifier
rfc= RandomForestClassifier()
rfc.fit(X_train, y_train)
y_pred= rfc.predict(X_test)
print("Accuracy                        of                        RandomForest
Classifier:",metrics.accuracy_score(y_test, y_pred))
import pickle
pickle.dump(rfc, open('model.pkl', 'wb'))
pickled_model = pickle.load(open('rfc.pkl', 'rb'))
pickled_model.predict(X_test)
print(rfc.score(X_train, y_train))
print(rfc.score(X_test, y_test))
import spotipy
from spotipy.oauth2 import SpotifyClientCredentials
import requests
import re
client_id = '42941b21de9046c8b880147deda15f1a'
client_secret = 'ff1daeadc5b94461b376e7599e628eaf'
redirect_uri = 'http://localhost:8000/callback'

# Create a client credentials manager
```

```
client_credentials_manager = SpotifyClientCredentials(client_id=client_id,
client_secret=client_secret)

# Create a Spotipy client
sp                                                                    =
spotipy.Spotify(client_credentials_manager=client_credentials_manager)

data = {
    'url': 'https://audd.tech/example.mp3',
    'return': 'spotify',
    'apitoken': 'test'
}
files = {
    'file': open('C://Users//Ayush//Desktop//rap1.mp3' , 'rb'),
}
result = requests.post('https://api.audd.io/', data=data, files=files)
x = result.text
track_id = x.split("spotify:track:")[-1][:-4]
# Get track details using the track ID
track = sp.track(track_id)
track_name = track['name']
artist_name = track['artists'][0]['name']
album_name = track['album']['name']
duration_ms = track['duration_ms']

# print('Track:', track_name)
# print('Artist:', artist_name)
# get audio features
song_name_artist = track_name + artist_name
song = sp.search(song_name_artist, limit=5)
audio_features = sp.audio_features(song["tracks"]["items"][0]["id"])[0]
audio_features
from sklearn.preprocessing import LabelEncoder as le
from sklearn.model_selection import RandomizedSearchCV
import numpy as np
from collections import defaultdict

arr=[]
for i in audio_features.keys():
  if(i=='type'):
```

```
    break
  arr.append(audio_features[i])
arr.append(audio_features['time_signature'])
arr.append(audio_features['duration_ms']/60000)

results= ['Dark Trap', 'Emo', 'HipHop', 'Pop', 'Rap', 'RnB', 'Trap Metal',
'Underground Rap', 'DnB', 'HardStyle', 'PsyTrance', 'TechHouse', 'Techno',
'Trance', 'Trap']

out= rfc.predict([arr])
print(results[out[0]])

@app.route("/")
def hello_world():
    return render_template('index.html')

@app.route("/result", methods = ['GET','POST'])
def result():
    values = request.form.to_dict()
    pred = songsearch(values['id'])
    return render_template('result.html', pred = pred)

if __name__ == '__main__':
   app.run()
```