Predicting Music Genres From Song Attributes

Gwen Delos Santos

April 11 2024

1 Introduction

Music encompasses various elements like instrumentation, key, tempo, and more, each contributing to create a specific sound and appeal. However, when defining the genre of a song, certain key attributes stand out. Understanding these defining features could enable songwriters to target specific audiences effectively and help listeners discover music that aligns with their preferences. Furthermore, music genres form the basis for developing music recommendation systems, refining advertising tactics, and enhancing overall user experiences.

I aim to explore the underlying characteristics that define different music genres and how accurately we can classify songs into these genres.

2 Problem Setup/Formulation

I found two datasets on Kaggle, both derived from Spotify, containing similar features. However, each dataset includes some unique features not present in the other. My goal is to utilize both datasets to gather as much data as possible. Below are sample data points from each dataset:

Table 1: Prediction of Music Genre (Part 1)

instance_id	artist_name	track_name
89706	Kesha	Cannibal
90236	Santana	I'm Feeling You
47457	Ludwig van Beethoven	Symphony No. 9 in D Minor, Op. 125 "Choral": II. Molto vivace

Table 2: Prediction of Music Genre (Part 2)

popularity	acousticness	danceability	duration_ms	energy	instrumentalness	key
56	0.0078	0.71	194253	0.68	0.0165	C#
40	0.0715	0.571	270973	0.9	1.98e-06	F
42	0.958	0.459	819347	0.148	0.747	D

Table 3: Prediction of Music Genre (Part 3)

liveness	loudness	mode	speechiness	tempo	obtained_date	valence	music_genre
0.0493	-4.676	Major	0.106	129.971	4-Apr	0.571	Rap
0.118	-4.975	Minor	0.0499	120.026	4-Apr	0.794	Blues
0.0365	-24.798	Minor	0.056	119.797	4-Apr	0.253	Classical

Table 4: Spotify Tracks genre (Part 1)

Unnamed: 0	track_id	artists	album_name
8	0IktbUcnAGrvD03AWnz3Q8	Jason Mraz;Colbie Caillat	We Sing. We Dance. We Steal Things.
72695	1qz8RLkJJLeSBZoE1haOwH	For Today	Breaker
91099	12cG3BUpk6PzLAzWK0mryO	Red Hot Chili Peppers	Alternative Rock Mixtape

Table 5: Spotify Tracks genre (Part 2)

track_name	popularity	duration_ms	explicit	danceability	energy	key	loudness	mode
Lucky	74	189613	FALSE	0.625	0.414	0	-8.7	1
White Flag	25	213973	FALSE	0.314	0.935	10	-4.153	0
By the Way	2	215666	FALSE	0.441	0.986	0	-2.466	1

Table 6: Spotify Tracks genre (Part 3)

speechiness	acousticness	instrumentalness	liveness	valence	tempo	time_signature	track_genre
0.0369	0.294	0	0.151	0.669	130.088	4	acoustic
0.161	0.00341	0	0.407	0.294	133.657	4	metalcore
0.116	0.038	0.000781	0.0998	0.184	123.974	4	rock

To integrate both datasets, I first identified matching column names to unify the features. Additionally, I standardized the target labels—renaming 'music_genre' or 'track_genre' to simple 'genre' for consistency across datasets. I ensured uniformity in genre names by converting them to lowercase labels. Similarly, I standardized the different features such as the representation of musical keys, converting them all to integers. Moreover, I opted to remove rows with null values, ensuring that only complete data points were included in the training process. Descriptions of the finalized features: (with reference to the Spotify Dataset Descriptions)

Liveness: A value ranging from 0 to 1, representing the probability of a track being performed live.

Key: Utilizing Pitch class notation, a value ranging from 0 to 11, where each integer corresponds to a specific pitch class. For instance, C is represented by 0, C#/Db by 1, and D by 2.

Speechiness: A value between 0 and 1 indicating the prevalence of spoken words in a track. Higher values signify a greater presence of pure vocals.

Acousticness: A confidence score ranging from 0 to 1, reflecting the likelihood that a data point is acoustic.

Popularity: A scale from 0 to 100, with 100 representing the highest level of popularity. It is calculated based on the number of plays the song receives.

Energy: A value between 0 and 1, where a higher score indicates heightened energy within the track.

Loudness: Measured in decibels (dB), indicating the volume level of the track.

Duration_ms: Measured in milliseconds, denoting the duration of the track.

Instrumentalness: A value ranging from 0 to 1, with higher values suggesting a greater likelihood of the track being instrumental (i.e., containing no vocals).

Mode: Represented by 0 or 1, indicating whether the track follows a minor (0) or major (1) scale.

Valence: A value between 0 and 1, reflecting the overall positivity of a track.

Tempo: Measured in beats per minute (BPM), indicating the speed or pace of the track.

Danceability: A score between 0 and 1, determined by factors such as tempo, rhythm, and stability, representing how suitable the track is for dancing.

Some sample features in the new dataset:

Table 7: Sample Features (Part 1)

liveness	key	speechiness	acousticness	genre	popularity	energy	loudness	duration_ms
0.203	2	0.044	0.198	emo	47	0.438	-8.77	190240
0.16	7	0.121	0.0261	ska	35	0.861	-3.267	159120
0.363	0	0.0652	0.0117	alternative	51	0.757	-8.065	246200

Table 8: Sample Features (Part 2)

instrumentalness	mode	valence	tempo	Danceability
0	0	0.108	111.027	0.84
1.11e-05	1	0.858	180.132	0.528
0.14	1	0.311	89.694	0.288

3 Methodology

I compared three different models: Decision Trees, K-Nearest Neighbors (KNN) with k=3, and a Feedforward Neural Network (FNN). For the Decision Trees and KNN models, I utilized the scikit-learn library, while I implemented the FNN using PyTorch. To ensure consistency in model training, I preprocessed the dataset by shuffling the rows and splitting it into 66% for training and 33% for testing. I chose not to allocate a separate validation set since scikit-learn models train in one go rather than in epochs, ensuring training consistency across all models. For hyperparameter tuning within the FNN, I experimented with various configurations, including activation functions, learning rates, epochs, and layer sizes, to optimize model performance.

To establish baseline comparisons, I evaluated two simple approaches: random chance, where accuracy is 1/the total number of classes, and majority class classification, where I compared the accuracy of classifying the most frequent class observed in the training set. Similarly, for the top-K accuracies, 'random chance' in this case is the accuracy of predicting the top-K most frequent classes observed in the training set and the top k class classifier is the accuracy of classifying the top k classes within the training set.

Furthermore, I explored different variations of the dataset. Initially containing 115 different classifications, I merged some classes to reduce it to 34 classes. This reduction aimed to simplify the task for the models by reducing the number of classes and potential noise in the data. Additionally, I experimented with feature selection, removing potentially irrelevant features from the dataset.

The original features included danceability, tempo, duration_ms, mode, acousticness, popularity, energy, liveness, valence, key, speechiness, genre, instrumentalness, and loudness. After feature selection, I retained danceability, tempo, mode, energy, valence, key, genre, instrumentalness, and loudness. In this way, I would be able to see if my own intuition on music features show differences from using all the available features.

4 Evaluation

Success was evaluated through overall accuracy and top-k accuracy, where k equals to roughly 5% of all the classes, i.e., the dataset with 115 classes would have $k = 6$.

- **Accuracy:** Measures the proportion of correctly classified instances out of the total number of instances in the dataset. This provides a straightforward indication of how well the models were able to precisely identify the correct genre for each piece of music.
- **Top-k Accuracy:** Extends the evaluation beyond a pinpoint metric by considering whether the correct genre appears within the top-k predictions made by the models.

Accuracy provides a clear measure of a model's overall performance, allowing for easy comparison between different models and datasets. Secondly, by incorporating top-k accuracy, the evaluation becomes more nuanced, capturing cases where the correct genre may not be the top prediction but is still within the realm of likelihood according to the model. A high accuracy would prove a great success as the models would be able to distinguish between the few provided features. However, by having the top k accuracies, we allow for a margin of error since some genres may be similar to others.

5 Results

Overall, the performance of each model fell short compared to the baseline metrics of random chance and majority class classification. While each model outperformed random chance, their performance was comparable to that of the majority class classifier, especially evident in the case of the Feedforward Neural Network (FNN) where the accuracies were nearly identical for each dataset.

When comparing the models against each other, the decision trees consistently demonstrated the highest accuracy levels, occasionally even surpassing the baseline comparisons. Decision trees exhibited a strong fit to the training data, potentially indicating overfitting, but their testing accuracies remained competitive, suggesting that pruning techniques could further enhance their performance.

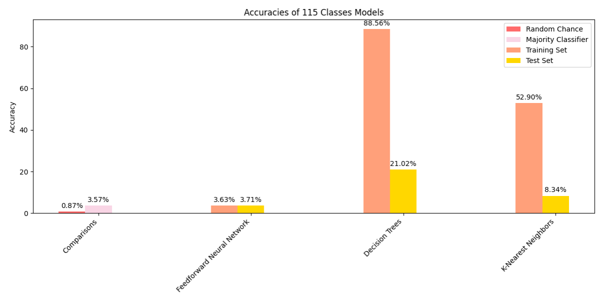
In contrast, the FNN struggled to generalize to new data points, with training and testing accuracies converging closely. This indicates neither overfitting nor underfitting but rather suggests that the FNN requires further optimization or may not be the most suitable model choice for this classification task.

Lastly, the performance of the K-Nearest Neighbors (KNN) algorithm varied depending on the dataset configurations. It performed best with fewer classes, fewer features, and when predicting within the top k classes. This may be attributed to the reduction of feature dimensions leading to more similar neighbors, thereby mitigating potential sources of prediction variance.

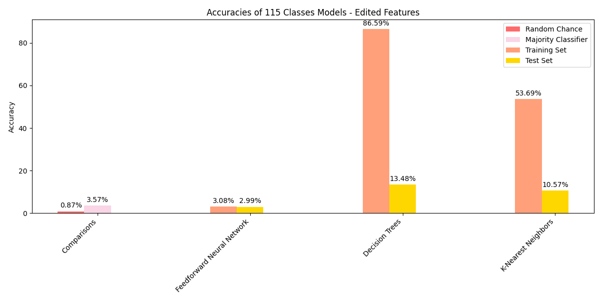
When comparing various dataset configurations, I had hoped that reducing the number of classes would lead to more distinct changes in accuracies. However, the observed differences were not substantial. While accuracies did improve for each model, they did so in proportion to the baseline accuracies, resulting in minimal overall variation. This lack of significant change could stem from discrepancies between my personal genre interpretations and the features used, potentially introducing noise into the data.

Similarly, when limiting the features, I aimed to see whether my subjective understanding of music classification could impact model performance. Once again, this adjustment failed to yield distinct changes beyond minor fluctuations.

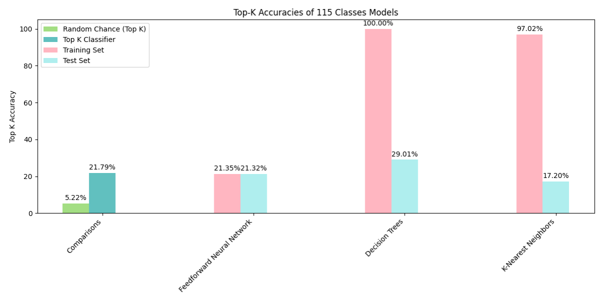
The minimal shifts in performance across the different dataset configurations have several implications. Firstly, the reliance on personal interpretations highlights the challenge of accurately predicting genres, even for humans. Additionally, issues may have arisen from initial data cleaning processes, suggesting the potential benefits of analyzing individual datasets before combining them. Moreover, the datasets utilized may not fully capture all aspects of music, such as instrumentation, time period, and rhythm. Furthermore, since the data originated from Spotify's algorithms for predicting likelihoods and probabilities, its ability to accurately reflect song components is not guaranteed. For future endeavors in genre classification, exploring additional features or alternative datasets could prove valuable.



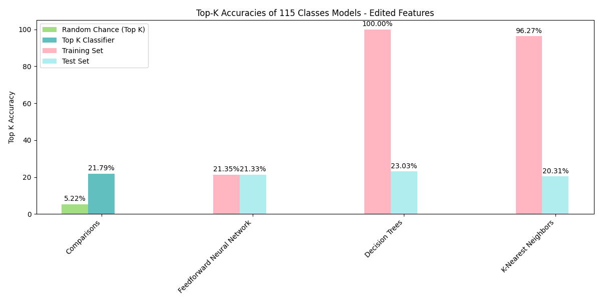
(a) Accuracies of 115 Classes Models



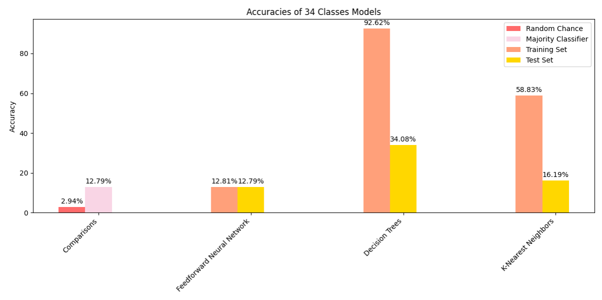
(b) Accuracies of 115 Classes Models - Edited Features



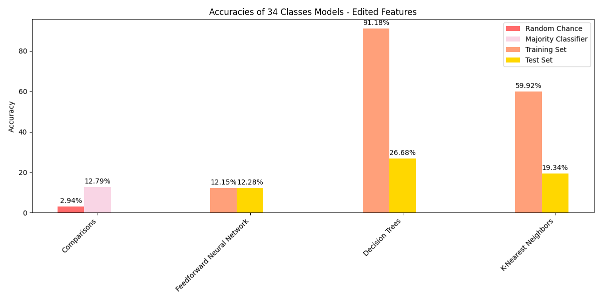
(c) Top-K Accuracies of 115 Classes Models



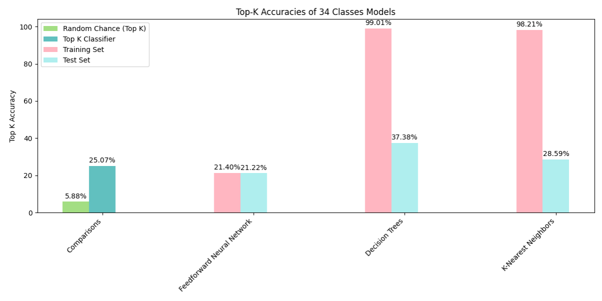
(d) Top-K Accuracies of 115 Classes Models - Edited Features



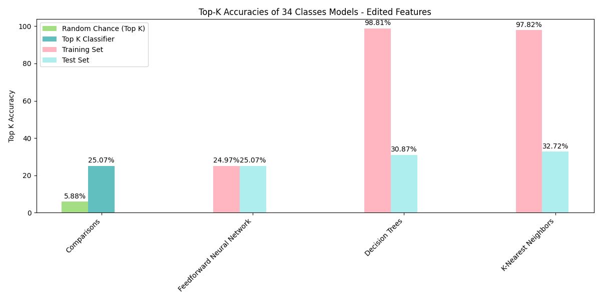
(e) Accuracies of 34 Classes Models



(f) Accuracies of 34 Classes Models - Edited Features



(g) Top-K Accuracies of 34 Classes Models



(h) Top-K Accuracies of 34 Classes Models - Edited Features

Figure 1: Comparing accuracies and top-k accuracies of different class models

References

- **Decision Trees Documentation:**
  https://scikit-learn.org/stable/modules/tree.html
- **Data Cleaning with Pandas:**
  https://saturncloud.io/blog/how-to-replace-multiple-values-in-one-column-using-pandas/
  (For data cleaning)
- **Reverse Label Encoding in scikit-learn:**
  https://stackoverflow.com/questions/58217005/how-to-reverse-label-encoder-from-sklearn-for-multipl
  (For how to reverse the label encoding)
- **FNN Implementation from CMPUT 328 Assignment 2:**
  (For FNN)
- **FNN Implementation from CMPUT 466 Assignment 2:**
  (For FNN)
- **Bar Graph Plotting Guide:**
  https://stackoverflow.com/questions/14270391/how-to-plot-multiple-bars-grouped
  (How to plot bar graphs)
- **Documentation for K-Nearest Neighbors Classifier:**
  https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.
  html
- **Difference between predict() and predict_proba() in scikit-learn:**
  https://stackoverflow.com/questions/61184906/difference-between-predict-vs-predict-proba-in-scikit
- **Prediction of Music Genre Dataset:**
  Vicsuperman. (2022, April). Prediction of Music Genre, Version 1. Retrieved January 25, 2024, from https://www.kaggle.com/datasets/vicsuperman/prediction-of-music-genre/data.
- **Spotify Tracks Genre Dataset:**
  The Devastator. Spotify Tracks Genre Dataset. Version 1. Retrieved January 25, 2024 from https://www.kaggle.com/datasets/thedevastator/spotify-tracks-genre-dataset.