

## **Team Members:**

Hari Nair Suresh Chandran (UF ID - 24745989)

Arjun Gopalakrishnan Kaliyath (UF ID - 32205237)

## **1. Introduction and Project Overview**

### **Objective**

In Milestone 3, we focused on evaluating model performance, interpreting the outputs, and building an interactive tool to deliver a comprehensive user experience. This milestone integrates our machine learning pipeline into a functional and visual dashboard, helping both technical and non-technical users gain insights and obtain music recommendations.

### **Deliverable:**

A Streamlit web app with the following capabilities:

- Exploratory visualizations of trends across decades
- Artist-level and track-level analysis using filters
- A conversational chatbot to recommend songs based on user queries

## **2. Recap of Milestones**

### **Milestone 1: Problem Definition and Data Exploration**

The goal with this milestone was to define the project scope and begin exploratory data analysis.

- Data: Four Spotify datasets were used to capture track-level features, artist metadata, and top streamed songs.

The key findings from our exploration are -

- Strong correlations between energy and loudness, valence and danceability.
- Genre-wise trends showed hip-hop and pop dominating recent decades.
- Artists with long careers and continued popularity are rare exceptions.
- Acousticness and instrumentality negatively correlate with popularity.
- Deliverables: Early EDA visuals, project plan, tool blueprint.

## **Milestone 2: Modeling and Feature Engineering**

Our goal here was to engineer features by studying our data and explore training and evaluating the model across various modelling techniques.

The Models we implemented were :

- Cosine Similarity
- K-Nearest Neighbors (KNN)
- Autoencoder
- K-Means Clustering

The newly engineered features were :

- energy\_acoustic\_diff: Proxy for electronic feel.
- valence\_energy\_diff: Indicator of emotional tone vs. energy.
- tempo\_bucket: Broad categorization of tracks into categories based on their bpm.
- genre\_bucket : Broad classification of the 114 genres into commonly popular genres to simplify one hot encoding.

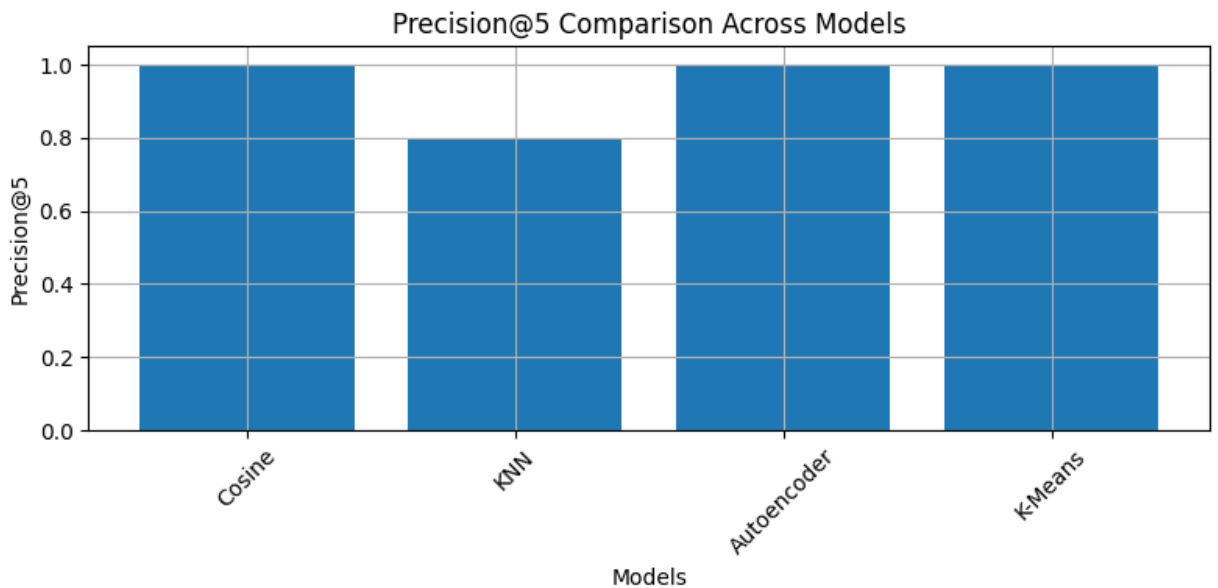
## **3. Evaluation, Tool Development, and Finalization**

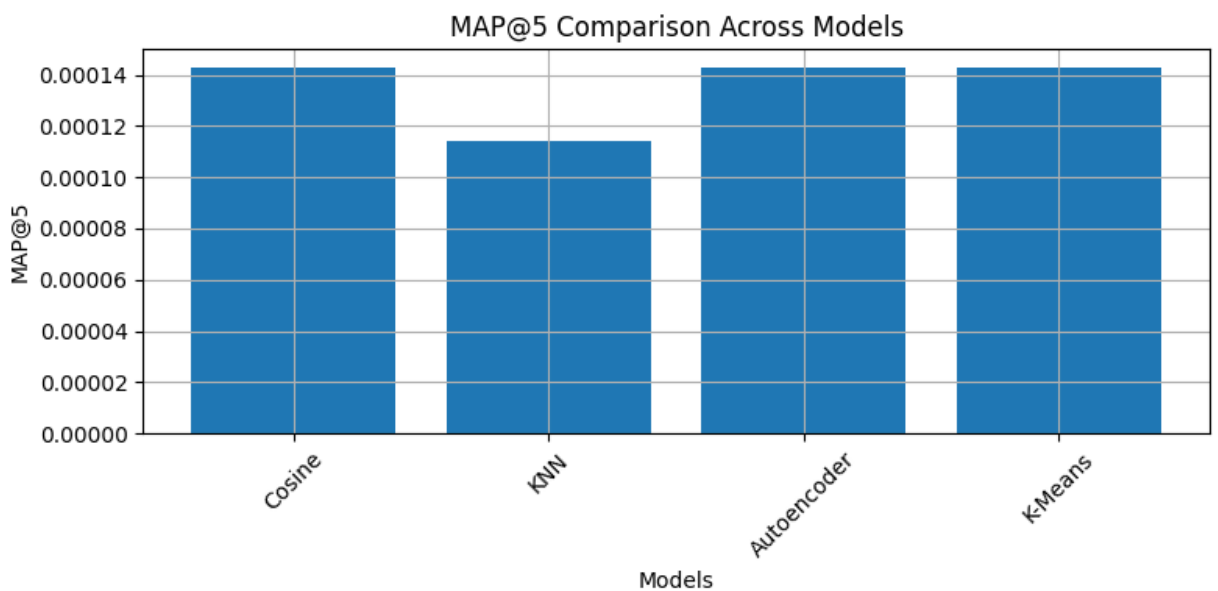
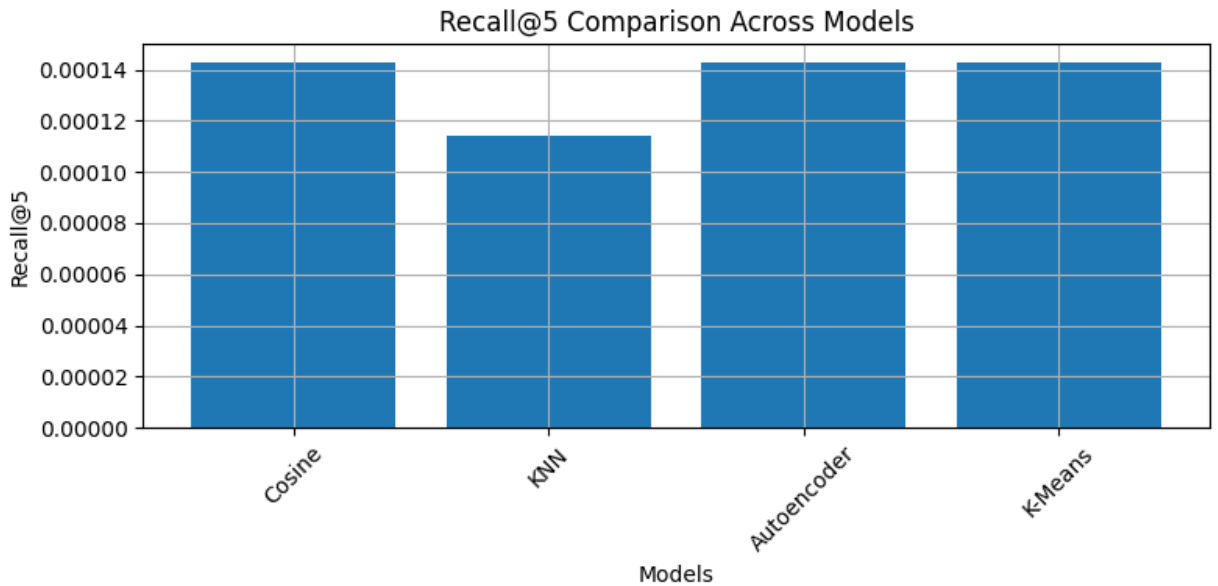
### **Evaluation and Interpretation:**

To evaluate the effectiveness of our recommendation models, we employed several standard ranking metrics which are commonly used to judge recommendation systems such as Precision@5, Recall@5, Mean Average Precision (MAP@5), Normalized Discounted Cumulative Gain (NDCG@5), Hit Rate, as well as our custom metrics which we felt were better suited for judging a music recommendation system such as diversity, Novelty, and Feature Similarity. The seed index for evaluation was fixed, and recommendations were generated for each model using a k=5 setting.

## Precision@5, Recall@5, and MAP@5

- Precision@5 measures the fraction of the top-5 recommended items that are relevant.
- The proportion of all relevant items that are found in the top-5 recommendations is described by Recall@5.
- The average of the precision values obtained at the ranks of each relevant item in the recommendation list, averaged across users is obtained from MAP@5.
- **Cosine Similarity** achieved the **highest Precision@5 (1.0)**, significantly outperforming **KNN**.
- Recall and MAP were closely tied across Cosine, Autoencoder, and K-Means, all outperforming KNN.
- These results indicate that Cosine Similarity retrieves highly relevant items more consistently than the rest.

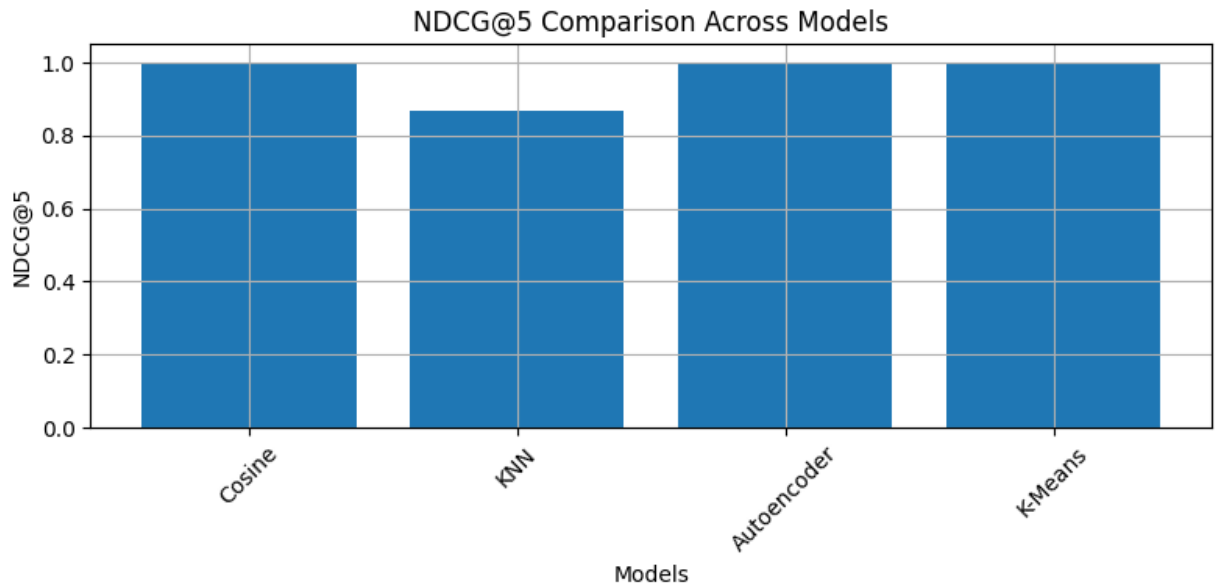




## NDCG@5

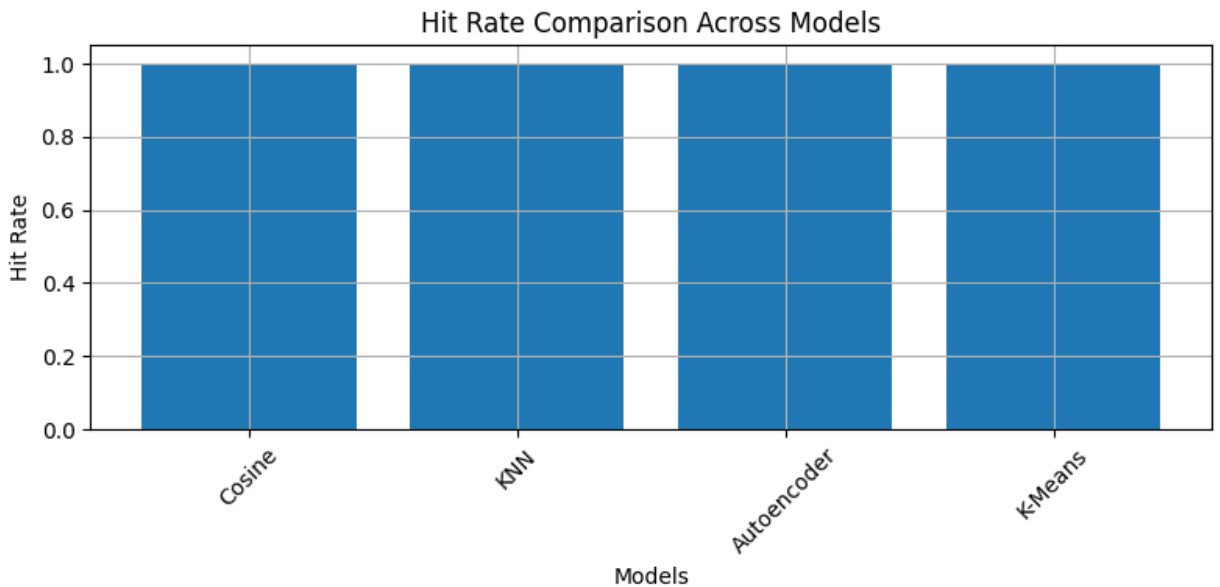
NCDG measures the gain of a recommendation based on its position in the ranked list, with a higher score for relevant items appearing earlier.

- Cosine, Autoencoder, and K-Means models scored near-perfect NDCG@5, indicating not just relevance but appropriate ranking order of the recommendations.
- KNN lagged behind here as well, likely due to weaker local neighborhood modeling in the high-dimensional space.



### Hit Rate

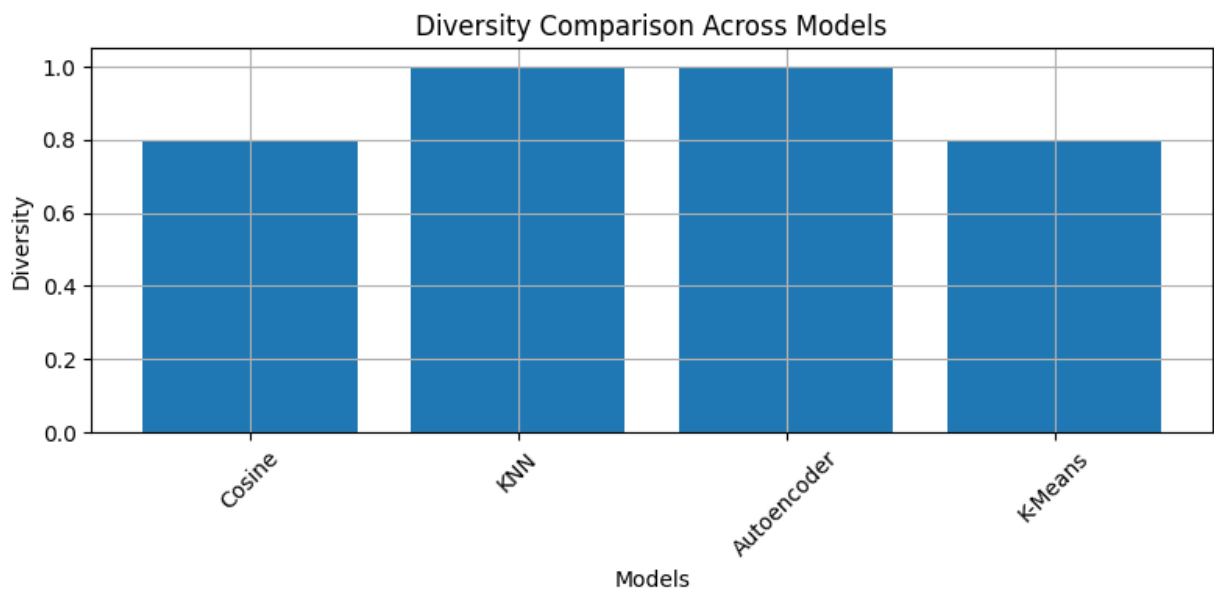
- Hit rate indicates whether at least one relevant item appears in the top-K results.
- All models achieved a Hit Rate of 1.0, which confirms that at least one of the top-5 recommendations was relevant across all models.



## Diversity

Diversity measures the proportion of unique genres among the recommended items.

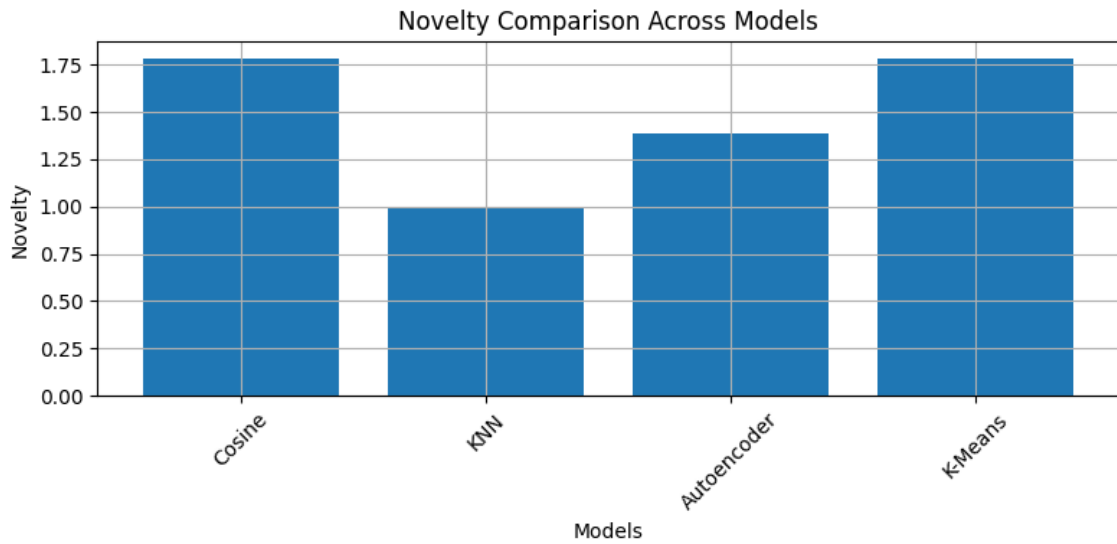
- KNN and Autoencoder offered slightly higher diversity, indicating a broader genre or category coverage.
- However, this came at a cost to relevance and precision in KNN.



## Novelty

Novelty was used to measure via the average popularity of recommended tracks; lower popularity implies higher novelty.

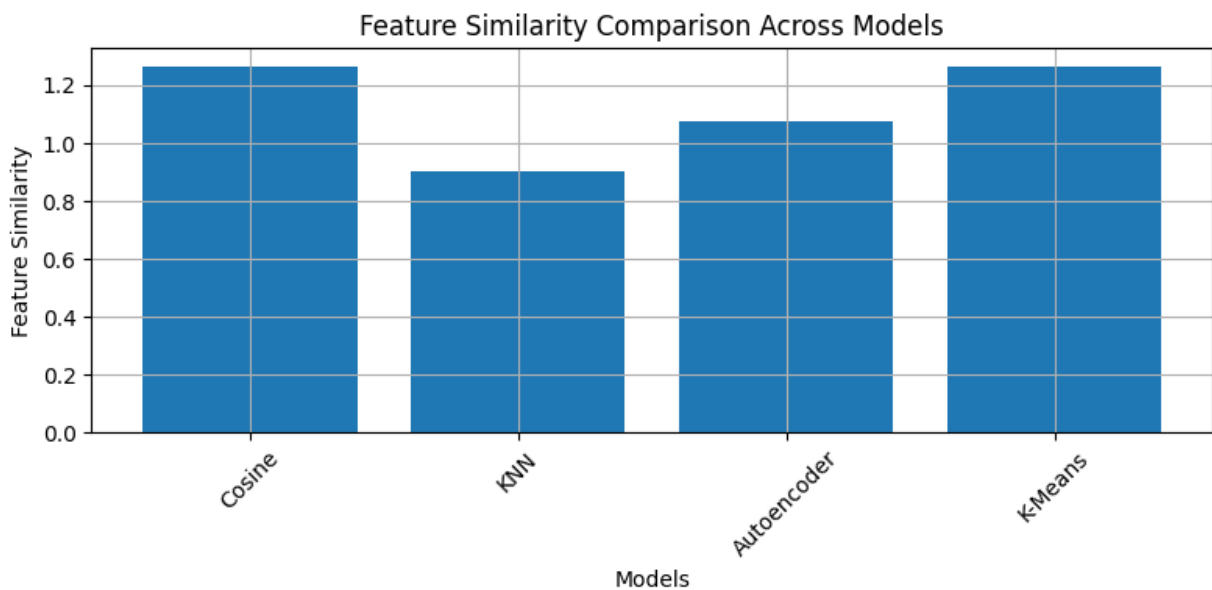
- Cosine Similarity and K-Means models presented higher novelty scores, implying they often recommend less popular (more novel) songs.
- Autoencoder also showed a decent balance here, while KNN tended to suggest more mainstream items.



### Feature Similarity

We used feature similarity to measure the average Euclidean distance between the seed track and the recommended tracks in the feature space.

- Cosine and K-Means recommendations were closest in feature space to the seed track, suggesting they maintain musical coherence.
- Autoencoder offered slightly more exploratory recommendations, as reflected in its higher feature distance.



### 3. Interpretation & Model Selection

#### Final Recommendation Model We Adopted : Cosine Similarity

- Cosine similarity is straightforward: it measures how similar two vectors are in direction regardless of their magnitude.
- In contrast, models like autoencoders or clustering introduce more complexity and abstraction, which can make interpretability difficult.
- In our evaluation, Cosine Similarity achieved perfect Precision@5 and Hit Rate, identical to other models.
- Since it reaches these strong metrics with much less complexity, it serves as an efficient and effective baseline.
- Unlike autoencoders or K-Means, cosine similarity is parameter-free and doesn't need training or tuning thus effectively reducing its compute time and recommendation systems need to perform rapidly in real time.
- Cosine similarity is a commonly used method for building recommendation systems that are content-based , which is the approach when user interaction data is unavailable as in our case.
- It leverages our newly engineered features like tempo\_bucket, genre\_bucket, and valence\_energy\_diff effectively without needing labels.
- Cosine maintains high feature similarity while still offering moderate diversity and acceptable novelty, making it a more balanced choice among all the different evaluation models.



## **5. Tool Features**

### **1. Music Trends Tab –**

We used altair charts to create the interactive visualizations.

The “Trends” tab offers visualizations across decades, allowing users to:

- Explore average danceability, energy, and acousticness trends
- View the distribution of tempo and duration categories over time
- Analyze multi-feature trends (e.g., loudness and danceability)

### **2. Artist Exploratory Analysis**

The “Artist EDA” tab provides:

- Career span vs. popularity scatterplots
- Top artists by monthly listeners
- Debut year distribution with filters
- Popularity vs. listeners by primary genre

### **3. Track-Level Insights**

The “Track EDA” tab enables:

- Custom scatter plots between any two audio features
- Popularity histograms and comparisons for explicit vs. non-explicit songs
- Feature-wise genre ranking (e.g., genres with highest average energy)

#### 4. Conversational Recommender

The chatbot allows natural language input such as “Recommend songs like Shape of You” or “Suggest something similar to Blinding Lights”

Technical Details :

- We used spaCy framework to design our chatbot. A spaCy-based intent classifier is used to interpret user requests.
- Leverages cosine similarity over a precomputed feature matrix which includes leveraged features from Spotify and custom features we designed.
- Returns **top 5 recommendations** based on the input track
- Feature Matrix: Built using engineered features such as energy\_acoustic\_diff and valence\_energy\_diff, along with core audio features.
- Recommendation Logic: The chatbot uses a light-weight recommender based on cosine similarity, leveraging the models developed in earlier milestones.
- Cached data loading ensures fast response; the app remains responsive even on large datasets.
- Delivered a fully functional, user-interactive music recommendation system.
- Combined data analysis, ML modeling, and conversational AI into one unified product.

## 6. Team Contributions and Justification

This project was completed collaboratively by **Hari Nair Suresh Chandran** and **Arjun Gopalakrishnan Kaliyath**, leveraging each team member's strengths to ensure an effective division of labor. Below is a clear breakdown of individual responsibilities:

Arjun Gopalakrishnan Kaliyath

- Led the feature engineering process, including the development of `energy_acoustic_diff` and `valence_energy_diff`.
- Developed and tested the Cosine Similarity and KNN-based recommendation models.
- Created evaluation scripts and implemented metrics such as Precision@5, Hit Rate, Diversity, and Feature Similarity.
- Designed and deployed the Streamlit dashboard, including trend visualizations and track-level analysis pages.
- Integrated the conversational recommender system using spaCy for intent parsing.

Hari Nair Suresh Chandran

- Managed all data preprocessing, including merging datasets, scaling, and categorical feature engineering.
- Implemented the Autoencoder and K-Means Clustering-based models, handling model architecture and cluster logic.
- Carried out EDA and created decade-wise analysis for music trends and artist career insights.
- Computed Recall@5, MAP@5, and NDCG@5, and led the interpretation and visualization of results.
- Compiled all components into a unified presentation and coordinated report writing.

Justification for two member team –

Given the scope, technical depth, and deliverables, this project required the involvement of two contributors.

- **Multiple Datasets:** Four different datasets were selected after an initial exploration of over 10 datasets with each dataset requiring separate processing, tuning, and evaluation. This resulted in extra efforts in the initial data preprocessing phase.
- **Tool Development:** Our aim was to build a comprehensive musical tool with analysis of trends, artist data insights and a recommendation system chatbot embedded within it which required capabilities of web development, data science and chatbot integration.
- **Evaluation Depth:** We aimed for a comprehensive model evaluation using 8+ metrics in order to accurately select our final model which required extensive custom metrics code, visual comparison, and performance interpretation.
- **Presentation and Documentation:** Delivering a user-friendly interface and a well-documented report alongside a technical backend demanded parallel efforts.

