

**Project Report**

**On**

**Movie Recommendation System using ML**



Submitted in partial fulfillment for the award of

Post Graduate Diploma in Big Data Analytics (PG-DBDA)

From Know-IT(Pune)

**Guided by:**

**Anay Tamhankar**

**Prasad Deshmukh**

**Submitted By:**

Aditya Chauhan (230343025001)

Prajwal Ratnakar Sirsat (230343025039)

Shantanu Arun Kolekar (230343025043)

Soham Nilesh Aher (230343025047)



**CERTIFICATE**

**TO WHOMSOEVER IT MAY CONCERN**

**This is to certify that**

Aditya Chauhan (230343025001)

Prajwal Ratnakar Sirsat (230343025039)

Shantanu Arun Kolekar (230343025043)

Soham Nilesh Aher (230343025047)

**Have successfully completed their project on**

**Movie Recommendation System using ML**

**Under the guidance of Anay Tamhankar Sir and Prasad Deshmukh Sir**



# ACKNOWLEDGEMENT

This project **“Music Recommendation System”** was a great learning experience for us and we are submitting this work to CDAC Know-IT (Pune).

We all are very glad to mention the name of **Anay Tamhankar Sir** and **Prasad Deshmukh Sir** for their valuable guidance to work on this project. Their guidance and support helped us to overcome various obstacles and intricacies during the course of project work.

We are highly grateful to **Mr. Vaibhav Inamdar** Manager (Know-IT), C-DAC, for his guidance and support whenever necessary while doing this course Post Graduate Diploma in Big Data Analytics (PG-DBDA) through C-DAC ACTS, Pune.

Our most heartfelt thanks goes to **Mrs. Bakul Joshi** (Course Coordinator, PG-DBDA) who gave all the required support and kind coordination to provide all the necessities like required hardware, internet facility and extra Lab hours to complete the project and throughout the course up to the last day here in C-DAC Know-IT, Pune.

**From:**

Aditya Chauhan (230343025001)

Prajwal Ratnakar Sirsat (230343025039)

Shantanu Arun Kolekar (230343025043)

Soham Nilesh Aher (230343025047)



**TABLE OF CONTENTS**

#### Abstract

1. **INTRODUCTION**

#### SYSTEM REQUIREMENTS

* 1. Software Requirements
  2. Hardware Requirements

#### FUNCTIONAL REQUIREMENTS

1. **SYSTEM ARCHITECTURE**

#### METHODOLOGY

1. **MACHINE LEARNING ALGORITHMS**

#### DATA VISUALIZATION AND REPRESENTATION

1. **CONCLUSION**
2. **FUTURE SCOPE**

**References**



## 

## ABSTRACT

This project aims to develop an advanced music recommendation system that enhances user engagement and satisfaction. The project involves aggregating music data from diverse sources, including APIs like Spotify and Last.fm. Leveraging cutting-edge data processing techniques, such as ETL in Apache Spark, the system will process and harmonize the data efficiently.

Through the implementation of sophisticated machine learning algorithms, including collaborative filtering, content-based filtering, and deep learning, the project seeks to uncover intricate music consumption patterns. The Apache Spark platform will facilitate the analysis of large-scale music datasets, enabling the extraction of meaningful insights.

In addition to algorithmic prowess, the project will emphasize intuitive data visualization techniques. Through visually appealing dashboards and interactive interfaces, users will gain an insightful and enjoyable way to explore and discover music.

This music recommendation system empowers users to effortlessly discover music hotspots within their preferred genres and moods. The project aims to revolutionize music discovery, fostering a more personalized, engaging, and user-centric music consumption experience.



## 1.INTRODUCTION

In this rapidly evolving digital landscape, the conventional approach to music discovery often falls short of capturing users' diverse preferences. To address this, our project takes a bold step by focusing on the lyrical essence of songs—a facet often neglected in existing recommendation systems. By leveraging Apache Spark's distributed computing capabilities and the prowess of machine learning algorithms, we aim to craft a recommendation system that understands and caters to the intricate emotional and thematic nuances encapsulated within song lyrics.

Our project employs this distributed computing framework to process and analyze vast quantities of lyrics efficiently. Through natural language processing techniques, we extract meaningful features from the lyrics that encapsulate themes, emotions, and stylistic elements.

Harnessing the potential of content-based filtering, our system shifts the paradigm of music recommendations. By understanding the textual essence of songs, it tailors recommendations to users lyrical preferences, transcending conventional genre-based suggestions. Whether it's songs evoking nostalgia or lyrics that resonate with specific moods, our system aspires to connect users with music on a profound level.

In conclusion, our project envisions a new era of music recommendation—one that is driven by the depth and resonance of song lyrics. By synergizing Spark's computational prowess and machine learning's predictive capabilities, we aim to create a musical voyage that transcends musical boundaries and speaks to the soul of each listener.

#### Datasets and Features:

#### The data used in this project was collected from www.kaggle.com the main goal of the analysis is to built an accurate and robust recommendation model to recommend the music based on the lyrical content of the music. This research uses vectorization, cosine similarity.



## 2.SYSTEM REQUIREMENTS

#### Software Requirements:

 Python 3

 Apache Spark

 Tableau

 Jupyter Notebook

 Google Colab

 **OS – Windows**

#### Hardware Requirements:

* Platform – Windows
* RAM – 8 GB of RAM,
* Peripheral Devices – Mouse, Keyboard, Monitor
* A network connection for data recovering over network.



## 3.FUNCTIONAL REQUIREMENTS

#### Python 3:

 Python is a general purpose and high-level programming language.

 It is use for developing desktop GUI applications, websites and web applications.

 Python allows to focus on core functionality of the application by taking care of common programming tasks.

 Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, Small Talk, and Unix shell and other scripting languages.

#### Apache Spark:

 What is Spark: Apache Spark is an open-source distributed computing system designed for processing large volumes of data.

 Key Features: Spark provides a number of key features that make it well-suited for processing big data, including in-memory processing, support for various data sources and formats, fault- tolerance, and scalability.

 Spark also provides a range of APIs, including SQL, streaming, machine learning, and graph processing, making it a versatile platform for a wide range of use cases.



#### Tableau:

 Data visualization is the graphical representation of information and data.

 It helps create interactive elements like charts, graphs, and maps, data visualization tools provide an accessible way to see and understand trends, outliers, and patterns in data.

 Tableau is widely used for Business Intelligence but is not limited to it.

 It helps create interactive graphs and charts in the form of dashboards and worksheets to gain business insights.

 All of this is made possible with gestures as simple as drag and drop.



#### Data Cleaning Process:

#### 

#### 

**Fig: Data Cleaning Process**

Certainly! Data cleaning is a crucial step in preparing data for analysis in a Music Recommendation System using Lyrics Features. Here's an explanation of the data cleaning process in the context of such a system:

**1.Data Collection:**

Gathering Lyrics Data: Collect lyrics data from various sources like APIs (e.g., Genius, Musixmatch), web scraping, or datasets. Ensure that the data includes relevant details such as song titles, artists, and lyrics.

**2.Data Cleaning Process:**

Handling Missing Values: Check for missing values in song titles, artists, or lyrics. Decide whether to omit incomplete records or use imputation techniques to fill missing values.

**I.Removing Duplicates:** Identify and remove duplicate song entries to prevent redundancy in the dataset.

**II.Removing Non-Lyric Text:** Some lyrics data may include non-lyrical content like tags or annotations. Remove such content to focus solely on the lyrical text.

**3.Text Preprocessing:**

**i.Lowercasing:** Convert all lyrics to lowercase to ensure uniformity in the dataset.

**ii.Tokenization:** Split lyrics into individual words or tokens for analysis.

**iii.Removing Punctuation:** Strip punctuation marks that don't contribute to lyrical analysis.



**iv.Removing Stopwords:** Remove common stopwords (e.g., "the," "and") that appear frequently but provide little insight into lyrical content.

**v.Stemming/Lemmatization:** Reduce words to their base or root form to group related words together (e.g., "running" and "ran" become "run").

**4.Handling Special Characters and Numbers:** Decide whether to remove or retain special characters, numbers, and digits based on their relevance to lyrical analysis.

**5.Handling Non-English Text:** If the dataset contains non-English lyrics, decide whether to remove them or apply language translation techniques if relevant.

**6.Dealing with Irrelevant Content:** Remove non-lyrical content such as adlibs, instrumental sections, or repetition markers if they don't contribute to the lyrical analysis.

**7.Finalizing Lyrics Features:** Create a consolidated, clean dataset with song titles, artists, and preprocessed lyrics. This dataset will serve as the foundation for analysis and modeling.

**Benefits:**

**i.Enhanced Accuracy:** Clean data ensures accurate analysis and better model performance.

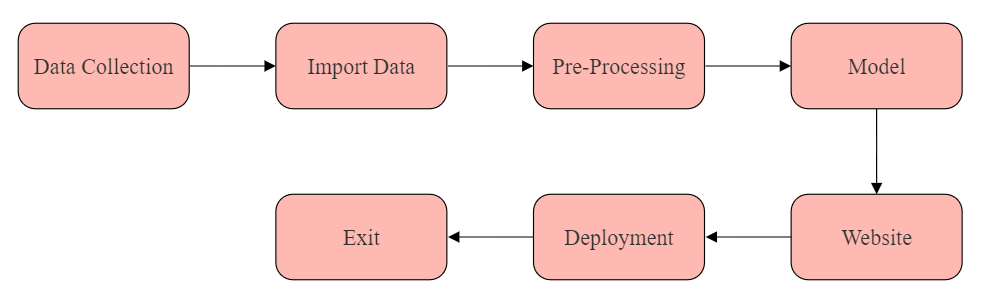
**ii.Focused Analysis:** Preprocessed lyrics enable meaningful insights into lyrical themes and sentiments.

**iii.Efficient Processing:** Clean data reduces errors and improves the efficiency of downstream tasks.

**Conclusion:**

Data cleaning is an essential step in the preparation of lyrics data for a Music Recommendation System. By refining the dataset and preprocessing the lyrics, we ensure that the subsequent analysis and modeling are based on accurate and relevant information, leading to more effective recommendations and user engagement.

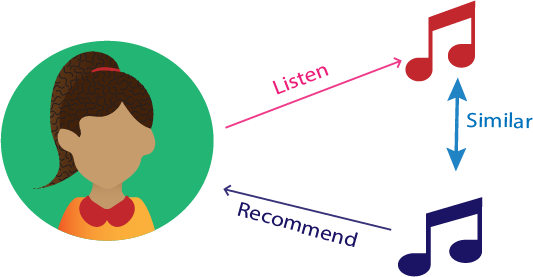
## 4.SYSTEM ARCHITECTURE



**Fig: System Architecture of Music Recommendation System**



# 5.METHODOLOGY

****

**Fig: Process of Recommending Music**

Content-based filtering is a recommendation system technique that generates personalized

recommendations for users based on the content or attributes of items and the preferences of the

users. It focuses on analyzing the characteristics of items and matching them to a user's preferences.

This approach is particularly useful when historical user interaction data is limited or when users have specific, well-defined preferences.



# 6.MACHINE LEARNING ALGORITHMS

Machine learning is the research that explores the development of algorithms that can learn from data

and provide predictions based on it. Works that study flight systems are increasing the usage of

machine learning methods. The methods which we used include TFIDF Vectorizer, Cosine\_Similarity, Content-Based Filtering.

Machine learning in the context of a Music Recommendation System leverages algorithms to learn from lyrics data and deliver personalized song suggestions. This approach is transforming the way users discover music.

In this project we use various machine learning algorithms which are as follows:

## 1.TF-IDF Vectorizer:

TF-IDF (Term Frequency-Inverse Document Frequency) Vectorizer is a powerful tool in Natural Language Processing (NLP) that transforms a collection of text documents into a numerical representation. It captures the significance of words in documents while considering their frequency and rarity across the entire corpus.

**Process:**

* **Term Frequency (TF):** It calculates the frequency of a word within a document. Words appearing frequently in a document receive higher TF values.
* **Inverse Document Frequency (IDF):** It quantifies the rarity of a word across the entire collection of documents. Words that appear in many documents have lower IDF values.
* **TF-IDF Score:** The product of TF and IDF scores assigns higher weight to words that are important within a specific document but not overly common across all documents

### 

### Advantage:

### Dimensionality Reduction: TF-IDF converts text data into a numerical matrix, reducing the dimensionality of the data, making it suitable for machine learning algorithms.

### Contextual Significance: By considering both the frequency within a document and rarity across the corpus, TF-IDF captures the contextual significance of words.

### Focus on Content: TF-IDF helps identify keywords or terms that distinguish documents, which is valuable in information retrieval and analysis.

### Application:

### In a Music Recommendation System using lyrics, TF-IDF can be applied to extract important words from song lyrics. It identifies words that define the lyrical essence of songs, such as themes, emotions, and style. By quantifying their significance using TF-IDF scores, the system gains insights into what makes each song unique.

### Conclusion:

TF-IDF Vectorizer is an essential NLP technique that transforms text data into a structured numerical format while capturing the contextual importance of words. In a Music Recommendation System, applying TF-IDF to lyrics enables the system to understand lyrical themes, emotions, and styles, enriching the music discovery experience for users.



## 2.Cosine\_Similarity:

Cosine Similarity is a mathematical measure used to assess the similarity between two non-zero vectors in a multi-dimensional space. In the context of text analysis, it quantifies the similarity between documents or pieces of text, irrespective of their length or magnitude.

**Process:**

* **Vectors and Angles:** In a vector space, each document or text can be represented as a vector where each dimension corresponds to a term's frequency or weight.
* **Cosine of the Angle:** Cosine Similarity calculates the cosine of the angle between two vectors. A smaller angle indicates a higher similarity, while a larger angle indicates dissimilarity.

**Benefits and Applications:**

1. **Scale Invariance:** Cosine Similarity is not affected by the magnitude of vectors, making it suitable for comparing documents of varying lengths.
2. **Directional Measure:** It focuses on the orientation of vectors rather than their magnitude, capturing the semantic similarity between texts.
3. **Text Analysis:** Cosine Similarity finds applications in text analysis tasks such as document clustering, recommendation systems, and information retrieval.

**Cosine Similarity in Music Recommendation:**

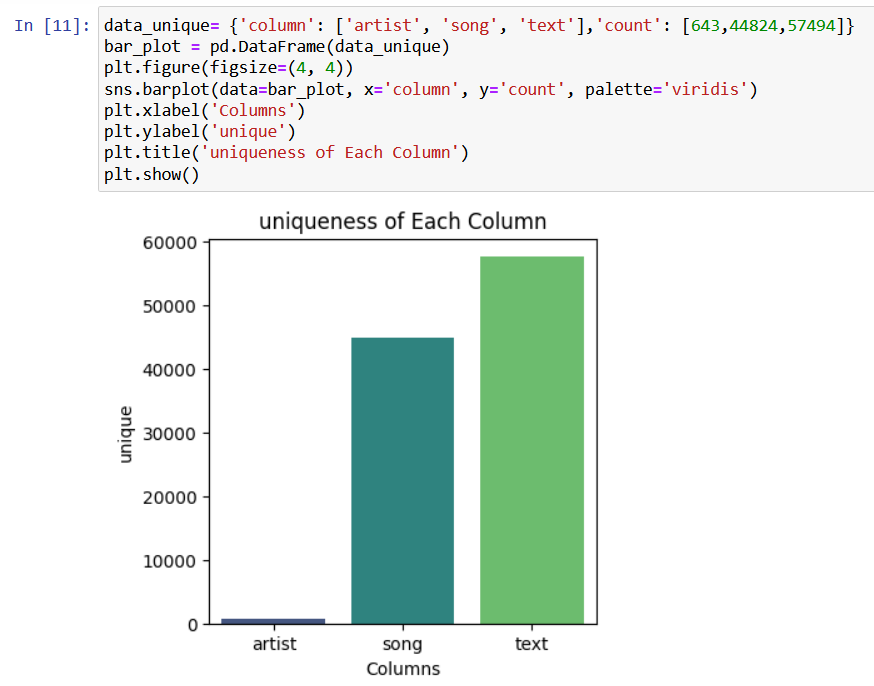
In a Music Recommendation System using lyrics, Cosine Similarity can be applied to quantify the lyrical similarity between songs. By representing songs as vectors based on their word frequencies, the system can calculate the cosine similarity to identify how closely the themes, emotions, and lyrical essence align.

**Conclusion:**

Cosine Similarity is a fundamental mathematical tool that facilitates the comparison of text documents or vectors in various domains. In a Music Recommendation System, applying Cosine Similarity to lyrics enables the system to measure lyrical resonance and make more accurate song suggestions based on shared themes, emotions, and content.



## 7.DATA VISUALIZATION AND REPRESENTATION





# 8.CONCLUSION

## Conclusion:

Music Recommendation System on the songdata dataset has shown promising results in recommending music with having similar lyrical content with different artist. The Music Recommendation System using Lyrics presents an innovative paradigm for music discovery. By delving into the lyrical essence of songs, the system offers users a more profound and personalized music journey. Through the analysis of lyrical themes, emotions, and sentiment, users receive recommendations that resonate with their innermost preferences and emotional states.

This system bridges the gap between musical preferences and lyrical content, enabling users to explore music that aligns with their moods and interests. As technology advances, the future holds exciting possibilities for incorporating sentiment analysis, emotion classification, and multilingual support, ensuring that recommendations become even more tailored and engaging.

Ultimately, the Music Recommendation System using Lyrics is a testament to the evolving relationship between technology and music. It enriches the connection between users and their favorite songs, providing a platform for musical exploration that goes beyond genres and beats, diving into the poetic heart of every melody.

.

# 9.FUTURE SCOPE



## Future Scope:

## In future music recommendation system on the songdata dataset can be further improved by incorporating additional data sources, such as :-

**1.Lyrics Sentiment Analysis:** By incorporating sentiment analysis, the system can understand users' emotions better, leading to recommendations aligned with their moods.

**2.Lyrics Emotion Classification:** Implementing emotion classification enhances the music experience by suggesting songs that resonate with users' emotional states, offering an immersive journey.

**3.Multilingual Support:** Extending the system's capabilities to recommend songs in diverse languages caters to global audiences and broadens musical horizons.

**4.User-Generated Content Integration:** Allowing users to contribute their own interpretations of lyrics fosters collaboration and adds a personal touch to music discovery.

**5.Exploration through Lyrical Themes:** Enabling users to explore songs based on specific themes encourages deeper engagement with favorite concepts.

**6.Incorporating User Context:** By leveraging real-time context like location and activities, the system provides recommendations that align with users' current situations.



# References

**Research Papers:**

1. Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems (TOIS), 22(1), 5-53.
2. McFee, B., Bertin-Mahieux, T., & Ellis, D. P. (2012). The Million Song Dataset and the Echo Nest API. Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR).

**Books:**

1. Ricci, F., Rokach, L., & Shapira, B. (2015). Introduction to Recommender Systems Handbook. Springer.
2. Online Articles and Blogs:
3. "Building a Music Recommendation System with Python," Towards Data Science. Available at: <https://towardsdatascience.com/building-a-music-recommendation-system-with-python-7d5a234b42f3>
4. "How Does Spotify Know You So Well?" Spotify Engineering Blog. Available at: <https://engineering.atspotify.com/2014/08/25/how-does-spotify-know-you-so-well/>

**Online Courses and Tutorials:**

1. **Coursera:** "Machine Learning" by Andrew Ng. Available at: <https://www.coursera.org/learn/machine-learning>
2. **Udacity:** "Deep Learning Nanodegree." Available at: <https://www.udacity.com/course/deep-learning-nanodegree--nd101>
3. **Technical Documentation and APIs:**
4. Spotify for Developers: Accessing song data and creating playlists. Available at: <https://developer.spotify.com/documentation/web-api/>