# Computational model to predict the Rhythmic beat pattern of the song and fit them into a taala system.

**Theory**

I. Introduction

Brief overview of the project.
Importance of rhythm and talas in music analysis
Objectives and goals of the project

II. Background

Definition of rhythm and talas in the context of music
Significance of rhythm analysis in music processing
Brief explanation of existing methods and technologies in rhythm analysis

III. Methodology

A. Data Collection

- Description of the dataset used
- Inclusion of diverse music genres and rhythm patterns
- Explanation of how the data was labeled or annotated

B. Feature Extraction

- Overview of the extracted features (tempo, beat, rhythm patterns, spectral features)
- Importance of each feature in rhythm and tala analysis
- Techniques used for feature extraction (mention libraries like LibROSA)

C. Preprocessing

- Normalization of features
- Segmentation of audio files for capturing local variations

D. Model Selection

- Choice of machine learning model (RNNs, LSTMs, SVMs, etc.)
- Justification for the selected model based on project requirements

E. Training

- Splitting the dataset into training and validation sets
- Training process and parameter tuning
- Challenges and solutions encountered during training

F. Evaluation
- Metrics used for model evaluation (accuracy, precision, recall, F1 score)
- Performance of the model on the validation set
- Analysis of the model's strengths and limitations

G. Fine-tuning
- Adjustments made based on evaluation results
- Consideration of alternative approaches or models

H. Testing
- Performance of the trained model on a separate test set
- Generalization of the model to new and unseen data

J. Continuous Improvement
- Feedback mechanisms for ongoing model improvement
- Future considerations for enhancing the system

IV. Results and Discussion
- Presentation of key results and findings
- Discussion of the implications of the results in the context of rhythm and talas analysis
- Comparison with existing methods and technologies

V. Conclusion
- Summary of the project
- Achievements and contributions to the field
- Potential future work and enhancements

VI. References
- Citation of relevant literature, tools, and libraries used in the project

VII. Appendices
- Additional details, code snippets, or supplementary information

# Table of Contents

# I. Introduction

## Overview:

In our project, we attempt to develop a computational model capable of predicting the rhythmic beat pattern of a song and integrating it into a specific tala—a foundational rhythmic framework in Indian classical music. By combining machine learning and signal processing techniques, our team aims to analyze audio data, extract features, and forecast the underlying beat structure of the music. The ultimate objective is to align these predicted beats with a predefined tala, offering valuable insights into the rhythmic structure of diverse musical compositions.

## Importance of rhythm and talas in music analysis:

Rhythm and talas form the fundamental building blocks of music, influencing its structure and emotional resonance. Rhythm, as the pulsating core, organizes beats and establishes a temporal framework, while talas, particularly evident in Indian classical music, contribute intricate rhythmic cycles that provide a structured foundation for expressive musical interpretation.

Beyond their role in shaping the sonic landscape, rhythm and talas bear cultural significance, acting as reflections of diverse musical traditions. Examining these elements unveils insights into cultural backgrounds, fostering a deeper connection to the music within a broader historical context. The dynamic interplay of rhythm with melody and harmony adds layers of complexity, contributing to the overall richness of the musical experience. Together, these elements guide performers, enrich cultural connections, and create a dynamic, expressive tapestry that defines the art of music.

## Key Objectives:

**1. Rhythmic Beat Prediction:**

- Develop algorithms to accurately extract tempo, beat, and rhythm patterns from audio signals.
- Train a machine learning model to predict the rhythmic beat pattern based on the extracted features.

**2. Tala Alignment:**

- Integrate the predicted beat pattern with a predefined set of talas in Indian classical music.

- Establish a mapping mechanism to align predicted beats with the tala structure, ensuring synchronization.

**3. Model Training and Evaluation:**
- Curate a diverse dataset encompassing various music genres and talas, annotated with ground truth beat and tala information.
- Focus on training and fine-tuning the machine learning model using the dataset, optimizing for accuracy and generalization.
- Evaluate the model's performance on validation and test sets using metrics such as accuracy and F1 score.

**4.Implementation and Deployment:**
- Implement the computational model in a user-friendly application or platform.
- Deploy the system for practical use, enabling users to analyze and understand the rhythmic beat patterns of different songs within the context of specific talas.

**Expected Outcomes:**
- A robust computational model capable of accurately predicting rhythmic beat patterns in diverse musical compositions.
- Integration of the model with a tala framework, allowing users to identify and appreciate the rhythmic nuances of songs within a cultural and musical context.
- Development of an accessible and user-friendly application for real-time rhythmic analysis and tala alignment.

**Significance:**
This collaborative project aims to bridge the gap between computational techniques and musical theory, providing a tool that aids in the understanding and appreciation of the intricate rhythmic structures inherent in music, particularly within the rich tradition of Indian classical music. Our team hopes that this endeavor will contribute to the fields of music analysis and computational musicology, offering a valuable resource for musicians, musicologists, and enthusiasts interested in exploring the rhythmic complexities of diverse musical compositions.

# II. Background

## Rhythm:

Rhythm is the temporal organization of sounds in music, creating a pattern of durations and accents that give a piece its characteristic flow and feel. It encompasses the arrangement of beats, the placement of accents, and the overall pacing of musical events. Rhythm can be regular or irregular, fast or slow, and it provides the framework that governs the timing of musical elements within a composition. In Western music, rhythm is often measured in terms of beats and subdivisions, forming the basis for the meter of a piece. The rhythmic patterns in music contribute to its expressive qualities, influencing the mood and dynamic feel of the composition.

## Tala:

Tala, on the other hand, is a term primarily associated with Indian classical music. It refers to a rhythmic cycle or framework that defines the organization of time in a musical performance. Talas consist of a specific number of beats grouped into a cycle, and each beat within the cycle can be accented or emphasized in various ways. The structure of a tala includes both strong and weak beats, creating a rhythmic hierarchy. Different talas have distinct characteristics, and they play a crucial role in shaping the rhythmic aspect of Indian classical compositions. Talas provide a rhythmic structure for improvisation, composition, and performance.

## Connection between Tala and Rhythm:

The connection between tala and rhythm lies in how talas provide a structured framework for the rhythmic patterns within Indian classical music. In this context, the tala defines the rhythmic cycle, specifying the number of beats and their arrangement into groups. Each tala has a unique pattern of accents and divisions, creating a rhythmic skeleton for a musical performance. Musicians use the tala as a guide for rhythmically navigating through compositions, allowing for intricate variations and improvisations while staying within the established temporal framework. In summary, while rhythm is a broader concept encompassing the temporal organization of sounds in any music, tala specifically refers to the rhythmic framework in Indian classical music, acting as a structured cycle that shapes the rhythmic patterns within that tradition.

| Element | Symbol | Duration in aksharakala units |
|---|---|---|
| Anudrutam | ˘ | 1 |
| Drutam | ° | 2 |
| Laghu | $\vert_3$, $\vert_4$, $\vert_5$,$\vert_6$ $\vert_7$, $\vert_9$ | 3,4,5,6,7,9 respectively |
| Guru | 8 | 8 |
| Plutam | 8̍ | 12 |
| Kakapadam | + | 16 |

## Brief History:

The significance of rhythm and talas in music analysis has deep historical roots, spanning various cultures and traditions. In the Western classical tradition, the concept of rhythm has evolved through different eras, from the measured rhythms of the Baroque period to the complex syncopations of the 20th century. In contrast, Indian classical music has a rich history of talas, where the ancient treatises like Natya Shastra laid the groundwork for rhythmic structures. The systematization of talas further developed with the contributions of theorists like Matanga and Sarangadeva in medieval India.

## Importance:

**1. Structural Foundation:** Rhythm, as the temporal backbone of music, provides a structural foundation for compositions. It dictates the arrangement of musical phrases, creating a cohesive and engaging listening experience.

**2. Expressive Dynamics:** Rhythmic patterns contribute to the expressive dynamics of a musical piece. Changes in tempo, accents, and rhythmic density convey emotion and intensity, adding depth and nuance to the overall musical narrative.

**3. Cultural Identity:** Talas, especially in Indian classical music, are deeply intertwined with cultural identity. They carry historical and regional significance, reflecting the diverse musical traditions across the subcontinent. Talas contribute to the distinct character of different genres and styles.

**4. Improvisational Framework:** Talas, in the context of Indian classical music, serve as a framework for improvisation. Musicians use the cyclical nature of talas to explore intricate rhythmic variations, showcasing their virtuosity and creativity.

**5. Communication in Performance:** Rhythm serves as a crucial means of communication among musicians during performances. The shared understanding of rhythmic patterns and talas facilitates coordination and synchronization, enabling seamless and harmonious ensemble playing.

**6. Audience Engagement:** A strong and well-articulated rhythm captures the audience's attention and enhances their engagement with the music. It provides a rhythmic hook that listeners can connect with and remember, contributing to the overall impact of the composition.

In essence, the importance of rhythm and talas in music analysis lies in their ability to shape the fundamental fabric of musical expression, offering a lens through which to understand the cultural, emotional, and structural dimensions of diverse musical traditions throughout history.

## Significance of rhythm analysis in music processing:

Rhythm analysis holds significant importance in the field of music processing, influencing various aspects of computational understanding and manipulation of musical content.

**Structural Segmentation:** Rhythm analysis serves as a crucial tool for segmenting musical pieces into meaningful sections. By identifying rhythmic patterns, algorithms can delineate verses, choruses, or other structural elements, providing a foundation for further analysis and processing.

**Tempo Estimation:** Understanding the tempo of a musical piece is essential for numerous applications, including automatic playlist generation, synchronization with visual media, and beatmatching in DJ software. Rhythm analysis enables algorithms to accurately estimate tempo, facilitating synchronization with external processes.

**Feature Extraction for Classification:** Rhythmic features extracted through analysis, such as beat patterns and tempo fluctuations, serve as valuable inputs for machine learning models. These features are used in the classification of music genres, mood prediction, and other tasks where rhythmic characteristics contribute to the differentiation of musical styles.

**Music Transcription:** Rhythm analysis plays a key role in music transcription, the process of converting audio recordings into symbolic representations. By identifying and analyzing rhythmic patterns, algorithms can generate notations that represent the timing and duration of musical events, aiding in the reproduction of the original piece.

**Beat Detection for Music Production:** In the realm of music production, rhythm analysis is crucial for beat detection. It assists in aligning various elements of a musical composition, synchronizing instruments and vocals, and providing a foundation for the creation of cohesive and well-structured arrangements.

**Interactive Music Systems:** Rhythm analysis is essential for the development of interactive music systems, where the system responds to the rhythm of the user or adapts to real-time changes in the music environment. This facilitates the creation of immersive and responsive musical experiences in applications like interactive installations, games, or virtual reality.

**Music Recommendation Systems**: Analyzing rhythm enables the development of more sophisticated music recommendation systems. Algorithms can consider rhythmic preferences along with other musical features to offer personalized recommendations based on the user's taste in rhythmic patterns.

**Automatic Drum Pattern Generation:** Rhythm analysis is integral to the creation of automatic drum pattern generators. By understanding the rhythmic structure of a piece, algorithms can generate drum patterns that complement the existing music, aiding in the creative process of music production.

In summary, rhythm analysis in music processing is a versatile tool that enhances computational understanding, facilitates feature extraction for various applications, and contributes to the development of innovative technologies in music-related fields. It enables machines to interpret and respond to the rhythmic intricacies of musical content, expanding the possibilities for automated music analysis and generation.

## Brief explanation of existing methods and technologies in rhythm analysis:

Rhythm analysis utilizes diverse methods and technologies to decipher and extract rhythmic patterns from music. Several key approaches include:

### Onset Detection:

**Method**: Algorithms identify the beginning of musical events by analyzing changes in amplitude or other spectral features.
**Application**: Fundamental for establishing the timing of musical events, forming the basis for further rhythm analysis.

### Beat Tracking:

**Method:** Algorithms analyze temporal patterns of detected onsets to identify and track beats, establishing the underlying rhythmic structure.
**Application:** Essential for tempo estimation, synchronization, and music transcription.

### Tempo Estimation:

**Method:** Algorithms calculate beats per minute (BPM) by analyzing inter-onset intervals or other temporal features.
**Application:** Crucial for synchronization with visual media, music production, and real-time applications.

### Rhythm Pattern Recognition:

**Method**: Algorithms use pattern matching and statistical analysis to recognize and categorize recurring rhythmic motifs.
**Application:** Applied in music genre classification, pattern-based music recommendation, and automatic drum pattern generation.

### Machine Learning and Deep Learning:

**Method:** Neural networks learn complex rhythmic patterns from large datasets, enhancing accuracy in tasks like beat tracking and onset detection.
**Application:** Improves the robustness of rhythm analysis algorithms, particularly in handling diverse musical styles.

### Dynamic Time Warping (DTW):

**Method:** Aligns two time series with varying speeds, accommodating tempo variations for tasks like music retrieval.
**Application:** Matches and aligns similar rhythmic sequences, disregarding tempo differences.

### Frequency Domain Analysis:

**Method:** Examines rhythmic content in the frequency spectrum using techniques like spectral flux or spectral peaks.
**Application**: Enhances rhythm analysis accuracy by capturing spectral characteristics associated with different rhythmic elements.

These methods collectively advance rhythm analysis, enabling applications in music processing, production, and interactive music systems. The interdisciplinary nature of rhythm analysis draws on signal processing, machine learning, and music theory, fostering the development of versatile and robust algorithms.

# III. Methodology:

## Data Collection:

For this project we have chosen the dataset "CompMusic Carnatic Music Rhythm Dataset". CompMusic Carnatic Rhythm Dataset is a rhythm annotated test corpus for automatic rhythm analysis tasks in Carnatic Music. The collection consists of audio excerpts from the CompMusic Carnatic research corpus, manually annotated time aligned markers indicating the progression through the taala cycle, and the associated taala related metadata.

### Description of the dataset used:

| Tāla | Beats, Aksharas | # Pieces | Total minutes (hours) | Median length of a piece (min) | # Annotated beats | # Samas | Annotated Positions in Tala Cycle |
|---|---|---|---|---|---|---|---|
| **Adi** | 8, 32 | 50 | 252.78 (4.21) | 4.85 | 22793 | 2882 | 1,2,3,4,5,6,7,8 |
| **Rupaka** | 3, 12 | 50 | 267.45 (4.45) | 4.62 | 22668 | 7582 | 1,2,3 |
| **Mishra Chapu** | 7, 14 | 48 | 342.13 (5.7) | 6.59 | 31055 | 7795 | 1,2,3,4,5,6,7 |
| **Khanda Chapu** | 5, 10 | 28 | 134.62 (2.24) | 4.41 | 13111 | 4387 | 1,2,3,4,5 |
| **Total** | | 176 | 996.98 (16.62) | 5.06 | 89627 | 22646 | |

The pieces are chosen from the CompMusic Carnatic music collection. The pieces were chosen in four popular taalas of Carnatic music (Table 1), which encompasses a majority of Carnatic music. The pieces chosen include a mix of vocal and instrumental recordings, new and old recordings, and to span a wide variety of forms. All pieces have a percussion accompaniment, predominantly Mridangam. The excerpts are full length pieces or a part of the full length pieces. There are also several different pieces by the same artist (or release group), and multiple instances of the same composition rendered by different artists. Each piece is uniquely identified using the MBID of the recording. The pieces are stereo, 160 kbps, mp3 files sampled at 44.1 kHz.

## Annotations:

**Sama and beats:** The primary annotations are audio synchronized timestamps indicating the different metrical positions in the taala cycle. The annotations were created using Sonic Visualizer by tapping to music and manually correcting the taps. Each annotation has a timestamp and an associated numeric label that indicates the position of the beat marker in the taala cycle. There are mainly 4 types of talas in this:

1. Adi Taala
2. Rupaka Taala
3. Mishra Chaapu Taala
4. Khanda Chaapu Taala

**Taala related metadata:** For each excerpt, the taala of the piece, edupu (offset of the start of the piece, relative to the sama, measured in aksharas) of the composition, and the kalai (the cycle length scaling factor) are recorded. Each excerpt can be uniquely identified and located with the MBID of the recording, and the relative start and end times of the excerpt within the whole recording. A separate 5 digit taala based unique ID is also provided for each excerpt as a double check. The artist, release, the lead instrument, and the raaga of the piece are additional editorial metadata obtained from the release. A flag indicates if the excerpt is a full piece or only a part of a full piece. There are optional comments on audio quality and annotation specifics.

| Tāla | Beats, Aksharas | # Pieces | Total minutes | Median length of a piece (min) | # Annotated beats | # Samas | Annotated Positions in Tala Cycle |
|---|---|---|---|---|---|---|---|
| **Adi** | 8, 32 | 30 | 58.87 | 2 | 5452 | 696 | 1,2,3,4,5,6,7,8 |
| **Rupaka** | 3, 12 | 30 | 60 | 2 | 5148 | 1725 | 1,2,3 |
| **Mishra Chapu** | 7, 14 | 30 | 60 | 2 | 8992 | 1299 | 1,2,3,4,5,6,7 |
| **Khanda Chapu** | 5, 10 | 28 | 55.93 | 2 | 9133 | 1840 | 1,2,3,4,5 |
| **Total** | | 118 | 234.8 | 2 | 28725 | 5560 | |

## Feature Extraction:

For handling of audio data and extracting features from it "librosa" library of python was used.

**Librosa:**

Librosa is a python package for music and audio analysis. It provides the building blocks to create music information retrieval systems. After dividing the audio sample into a fixed number of parts i.e 1200 in our case the following 4 features were extracted

1. Amplitudes
2. Magnitude spectrograms
3. Mfcc features
4. Chroma features

**Amplitudes:**

Amplitudes represent the strength or intensity of the audio signal at different points in time. In the context of rhythm prediction, amplitude variations can provide insights into the emphasis or strength of specific beats within a musical piece. Sudden changes in amplitude may indicate percussive elements, aiding in beat detection.

**Magnitude Spectrograms:**

Magnitude spectrograms provide a visual representation of the distribution of frequencies over time. In the context of rhythmic beat prediction, spectrograms can reveal rhythmic patterns and percussive elements. Peaks in the spectrogram may correspond to the dominant frequencies associated with beats, helping in the identification of rhythmic patterns.

**MFCC Features**:

MFCC features capture the spectral characteristics of the audio signal. In the context of rhythm prediction, MFCCs are valuable for identifying timbral patterns and distinguishing between different percussive elements. The lower-order MFCCs may contain information about the fundamental frequency of the rhythmic elements, aiding in beat tracking.

**Chroma Features:**

Chroma features represent the distribution of pitch classes or musical notes over time. In the context of your project, chroma features can contribute to the tonal analysis of the music, helping to identify the harmonic structure and tonal center. This information is useful for aligning rhythmic beats with the tonal characteristics of the chosen tala.

By combining these features, the model can analyze both rhythmic and tonal aspects of the music. For example, amplitude and magnitude spectrogram features can contribute to beat detection, while MFCC and chroma features can enhance the model's understanding of the tonal context. Machine learning models trained on these features can learn to predict rhythmic beat patterns and align them with the chosen tala.
Additionally, considering the dynamic nature of rhythm, features like amplitude variations and delta coefficients derived from MFCCs can provide temporal information, allowing the model to capture rhythmic nuances and changes over time.

## Preprocessing:

The features extracted as explained before are sampled from the audio sample when it is divided into a fixed number of parts through the librosa library. All these features are normalized and encoded and sent to model training.

## Data processing:

**Overview:**
The (extract_features) function is designed to process a collection of audio files and extract various features, such as amplitude, magnitude spectrogram, Mel Frequency Cepstral Coefficients (MFCC), and chroma features.

**Flow Description**

1. **Input Parameters:**
   - file_repo: The function takes the path to the directory containing audio files.
   - duration_per_step: The duration (in seconds) of each time step for feature extraction. Default is 0.1 seconds.
   - n_fft: The number of samples used for each Fourier transform. Default is 1024.
   - n_mfcc: The number of Mel Frequency Cepstral Coefficients to extract. Default is 13.
   - hop_length: The number of samples between successive frames for feature extraction. Default is 512.
2. **Feature Extraction Initialization:**
   - Initialize empty lists for storing different features, including amplitudes, magnitude spectrograms, MFCC features, and chroma features.
3. **Iterate Over Audio Files:**
   - For each audio file in the specified directory (file_repo):
     - Load the audio data using librosa.load.
     - Calculate the number of steps based on the desired duration (duration_per_step).

        - If the number of steps is less than 1200, pad the audio data to ensure a consistent length across all files.
4. **Feature Extraction Loop (for each time step):**
    - For each time step, extract the following features:
        - **Amplitude:** Calculate the average amplitude of the audio signal in the time domain.
        - **Magnitude Spectrogram:** Calculate the average magnitude spectrogram of the audio signal in the frequency domain.
        - **MFCC Features:** Extract Mel Frequency Cepstral Coefficients for each time step.
        - **Chroma Features:** Extract chroma features for each time step.
5. **Store Extracted Features:**
    - Append the extracted features for each time step to the respective lists.
6. **Return Extracted Features:**
    - Return a tuple containing the lists of extracted features:
        - amplitudes: Average amplitude for each time step.
        - magnitude_spectrograms: Average magnitude spectrogram for each time step.
        - mfcc_features: List of MFCC features for each time step.
        - chroma_features: List of chroma features for each time step.

## Model Selection:

For this project, different models were trained on this data. These models were imported from the python library “sklearn”.

### Sk-learn (scikit-learn):

Scikit-learn is a powerful and versatile machine learning library in Python that provides simple and efficient tools for data analysis and modeling.

1. This library provides simple and efficient tools for predictive data analysis.
2. It’s Accessible to everybody, and reusable in various contexts
3. Built on NumPy, SciPy, and matplotlib
4. Open source, commercially usable - BSD license

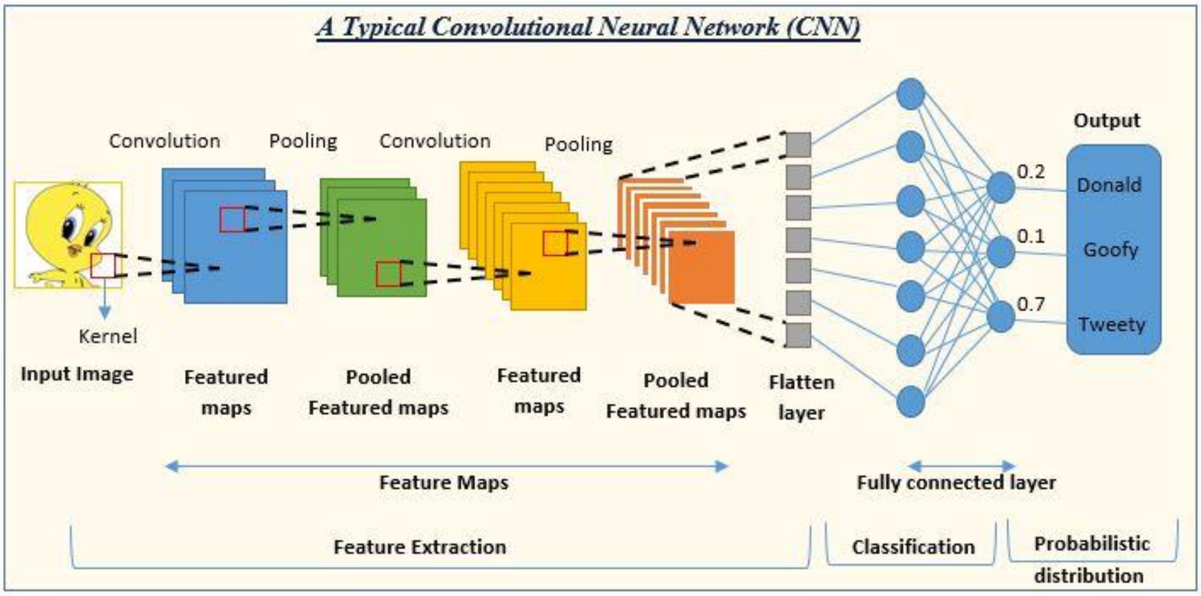
Key features:

1. **Broad range of Algorithms:**
   It provides a comprehensive suite of machine learning algorithms for classification, regression, clustering, dimensionality reduction, and more. This includes popular algorithms such as Support Vector Machines, Random Forests, K-Means, and Principal Component Analysis (PCA).
2. **Model evaluation and Selection:**
   Scikit-learn includes tools for model evaluation and selection, facilitating the comparison of different models and hyperparameter tuning. It offers metrics for assessing classification, regression, and clustering performance.
3. **Data Preprocessing and Feature Engineering:**
   The library includes utilities for data preprocessing and feature engineering. This encompasses techniques for handling missing data, scaling features, encoding categorical variables, and extracting relevant features from raw data.
4. **Pipeline for workflow management:**
   Scikit-learn provides a Pipeline class that allows users to streamline workflows by chaining together multiple data processing and modeling steps. This helps in creating reproducible and efficient machine learning pipelines.

For this project the following models were taken for rhythmic beat classification:

1. Convolutional Neural Networks(CNNs)
2. Decision Tree Classifier
3. Random Forest Classifier

## Convolutional Neural Networks (CNNs):

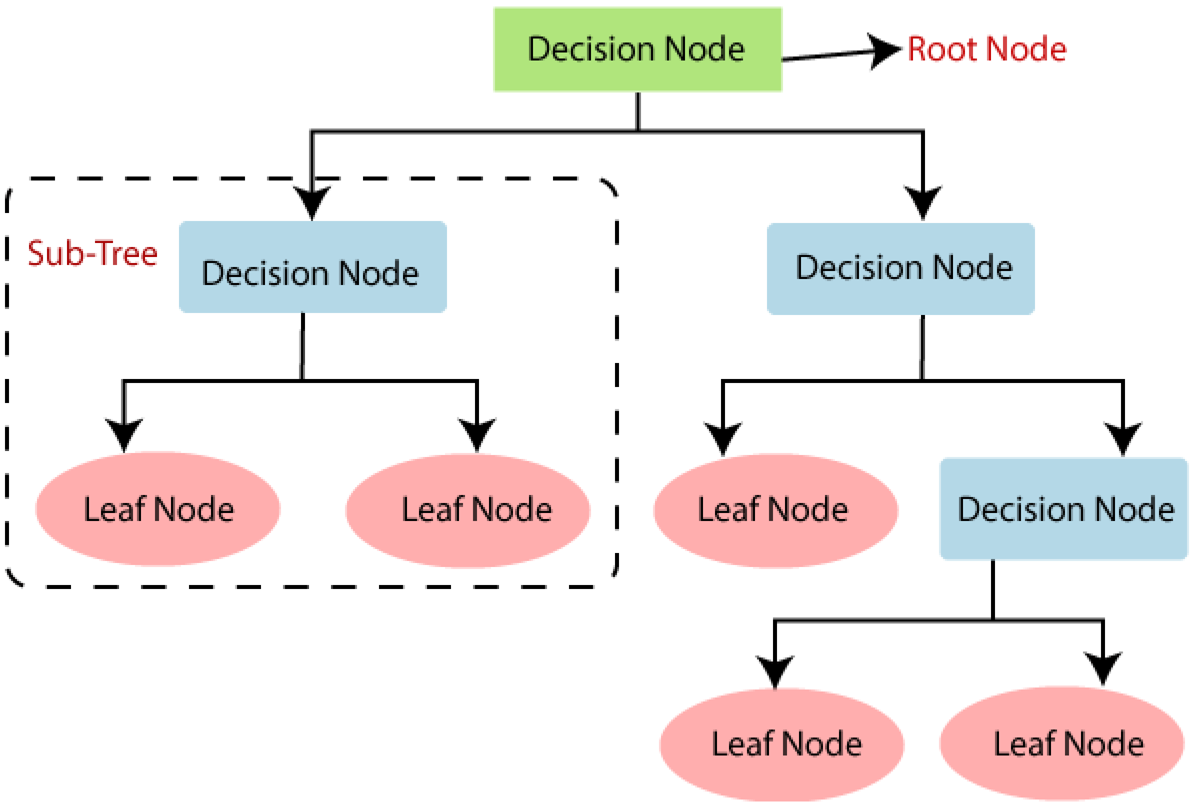


Convolutional Neural Networks (CNNs) are a specialized type of neural network designed for tasks involving image data. In our project, we leverage CNNs due to their inherent capability to recognize intricate patterns and features within images. These networks consist of convolutional layers that detect various visual elements, such as edges and textures, enabling them to automatically learn hierarchical features from simple to complex.

A key strength of CNNs lies in their ability to focus on local parts of an image through the use of convolutional filters. These filters capture spatial relationships, allowing the network to discern patterns efficiently. The concept of shared weights further enhances the learning process, enabling the model to generalize across different parts of the input space.

In the realm of image-related applications, CNNs have become indispensable. They are widely employed for tasks such as image classification, object detection, and facial recognition, showcasing their versatility and effectiveness. In our project, we harness the power of CNNs to automatically learn and extract relevant features from image data, contributing to the overall success of our computational model.

**Decision Tree Classifier:**

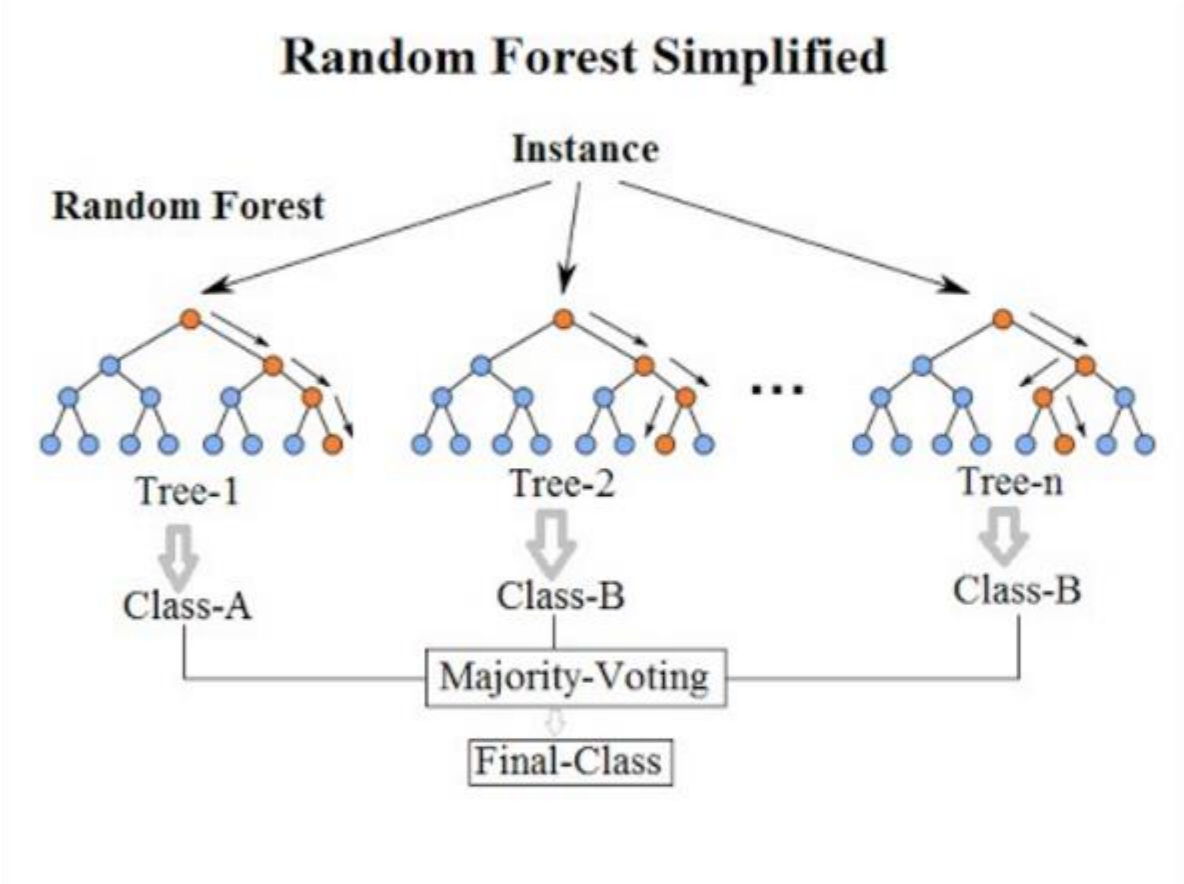


Decision Tree Classifier is a versatile and intuitive machine learning algorithm employed in our project for its simplicity and interpretability. This algorithm operates by recursively partitioning the input space based on the most significant features, creating a tree-like structure. Each internal node of the tree represents a decision based on a particular feature, while the leaves correspond to the predicted class.

In our project, the Decision Tree Classifier serves as a powerful tool for analyzing and predicting rhythmic beat patterns. Its ability to decipher complex decision boundaries and capture non-linear relationships within the data makes it well-suited for tasks where interpretability is crucial. By exploring various features and their importance, the model can discern the distinctive patterns associated with different rhythmic structures in songs.

One notable advantage of Decision Trees is their capacity to handle both numerical and categorical data, making them adaptable to diverse datasets. This flexibility aligns with the multifaceted nature of our project, where musical features may include a combination of continuous and categorical variables. Furthermore, Decision Trees facilitate feature importance analysis, offering insights into the relevance of each feature in predicting rhythmic patterns.

**Random Forest Classifier:**




The Random Forest Classifier, a powerful ensemble learning method, plays a pivotal role in our project for predicting rhythmic beat patterns. As an ensemble of decision trees, Random Forest combines the predictive strength of multiple individual trees to enhance overall accuracy and robustness. Each tree is trained on a different subset of the data and makes independent predictions, and the final result is determined by a majority vote or averaging, mitigating overfitting concerns associated with individual decision trees.

In the context of our project, Random Forest excels at capturing intricate relationships within the dataset and provides improved generalization compared to a single decision tree. Its ability to handle both numerical and categorical features makes it well-suited for the diverse musical feature set we are working with. By aggregating the predictions of multiple trees, the model gains stability and resilience to outliers, contributing to a more reliable prediction of rhythmic beat patterns.

The Random Forest's built-in feature importance analysis is a valuable asset for our project. It allows us to discern the significance of different musical features in predicting rhythmic patterns. This feature importance analysis provides insights into which aspects of the input data are most influential in the model's decision-making process, aiding our understanding of the underlying factors contributing to the rhythmic predictions.

Furthermore, Random Forest brings an element of interpretability to our computational model. While the ensemble nature can make it more complex than an individual decision tree, the overall structure and feature importance metrics remain accessible, providing a clear picture of the model's decision logic.

## Training:

For training the extracted feature dataset was divided into three sets,

1. Training set
2. Test set
3. Validation set

**Training Set:**

The training set constitutes the foundational component of machine learning model development. It comprises a substantial portion of the dataset and serves as the source from which the model learns patterns and relationships. During the training phase, the model is exposed to input features along with their corresponding target labels. The objective is for the model to discern and capture inherent patterns within the data, essentially learning how to map inputs to outputs. As the largest subset, the training set establishes the foundation upon which the model's predictive capabilities are built.

**Test Set:**

Reserved for the final evaluation, the test set simulates real-world scenarios by representing entirely new and unseen data. Similar in size to the validation set, the test set is crucial for gauging the model's ability to generalize effectively. The model, having undergone training and validation, is put to the ultimate test on the test set. This step provides an unbiased assessment of the model's performance and offers insights into how well it can handle previously unseen examples. The test set serves as a measure of the model's readiness for real-world applications.

**Validation Set:**

The validation set plays a critical role in the iterative process of model development and fine-tuning. Smaller in size compared to the training set but sufficiently representative, the validation set provides a means to assess the model's performance on unseen data. Following the training phase, the model is evaluated on the validation set. This evaluation aids in refining hyperparameters, selecting between different models, and guarding against overfitting. The validation set acts as a checkpoint, ensuring that the model generalizes well to new examples beyond the training set.

For this project, the dataset was partitioned into training, test, and validation sets with proportions of 70%, 15%, and 15%, respectively.

## Evaluation:

Primarily, 4 metrics were used to evaluate the model,

1. Accuracy
2. Precision score
3. Recall
4. F1 score

**Accuracy:**

Accuracy is a straightforward metric that measures the overall correctness of predictions. It is the ratio of correctly predicted instances to the total instances.

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ o\ predictions}$$

It is useful when classes are balanced, and there is no significant class imbalance.

**Precision Score:**

Precision focuses on the accuracy of positive predictions. It measures the ratio of true positive predictions to the total predicted positives.

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

It is valuable when the cost of false positives is high, and you want to minimize the occurrences of false positives.

**Recall:**

Recall emphasizes the model's ability to capture all positive instances. It measures the ratio of true positives to the total actual positives.

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

It is crucial when the cost of false negatives is high, and you want to minimize the occurrences of false negatives.

**F1 Score:**

F1 Score is the harmonic mean of precision and recall. It provides a balanced measure of a model's performance, especially in the presence of class imbalance.

$$F1\ score = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

It is suitable when there is an uneven class distribution, and you want a single metric that considers both false positives and false negatives.

The validation accuracies were reported as follows,

| | |
|---|---|
| Decision Trees | 94% |
| Random Forest | 100% |
| CNNs | 100% |

Clearly decision trees are lagging behind in predicting the rhythmic beat pattern of the song. This can be attributed to the fact that Decision trees can give only linear decision boundaries i.e. they fail to capture nonlinear relations. But Random forests which use decision trees internally overcome this limitation by considering multiple model predictions to arrive at a final prediction. CNNs extract the required useful features from the data and trains on them, it can also capture nonlinear relations therefore it gives optimal performance.

## Testing:

On the test set, the accuracies were reported as follows,

| | |
|---|---|
| Decision Trees | 88% |
| Random Forests | 94% |
| CNNs | 100% |

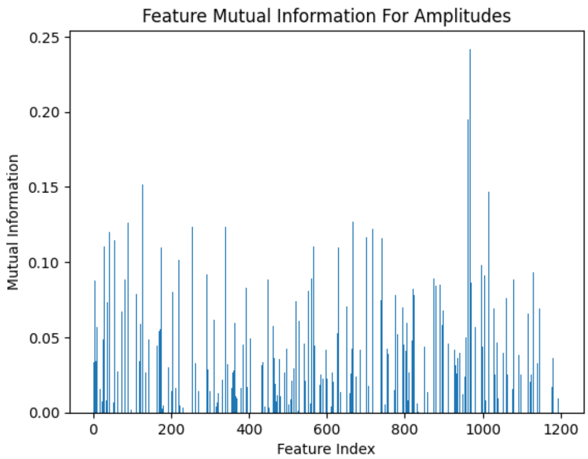
# IV. Results and Discussion:

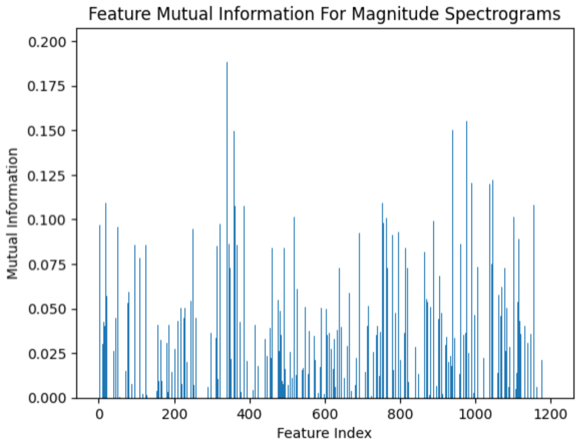
From these results we can infer the following:

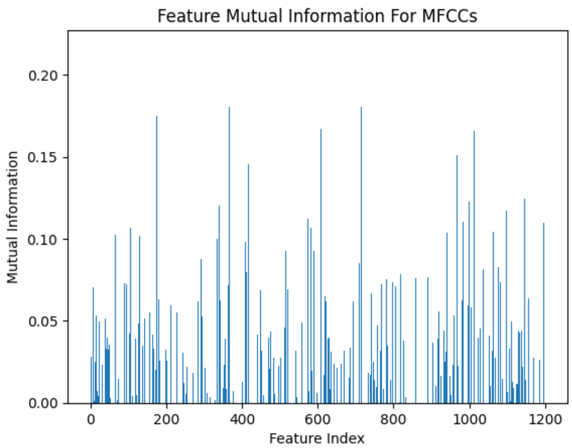
1. **Convolutional Neural Networks:**
   The CNN model exhibited the highest overall performance among the three models. Its sophisticated architecture allowed it to effectively capture intricate patterns and relationships within the image data, leading to superior predictive accuracy.
2. **Random Forest:**
   The Random Forest model demonstrated a commendable performance, positioning it between the CNN and Decision Trees. Leveraging ensemble learning, it excelled in handling diverse features and mitigating overfitting concerns associated with individual decision trees.
3. **Decision Trees:**
   While Decision Trees showcased a respectable performance, it fell slightly behind the CNN and Random Forest. The simplicity of the model may have led to challenges in capturing the nuanced patterns present in the complex dataset.
4. **Mutual Information**
   Mutual information measures the statistical dependence or information shared between two variables. In the context of feature selection, mutual information is used to assess the amount of information that the presence or absence of one feature provides about another feature. Specifically for classification tasks, it helps to quantify how much knowing the value of a particular feature reduces the uncertainty about the target variable.
   1. **High Mutual Information:** If the mutual information between a feature and the target variable is high, it suggests that the feature contains valuable information about the target. In a classification context, high mutual information indicates that the feature is informative for predicting the target class.
   2. **Low or Zero Mutual Information:** If the mutual information is low or close to zero, it implies that the feature does not provide much information about the target variable. In a classification context, such features may not be useful for distinguishing between different classes.
   3. **Feature Ranking:** By calculating mutual information for each feature, you can rank the features based on their informativeness. Features with higher mutual information are considered more important for the classification task.
   4. **Feature Selection:** You can use mutual information as a criterion for feature selection. Features with the highest mutual information values can be selected for building a classification model, while less informative features can be excluded.

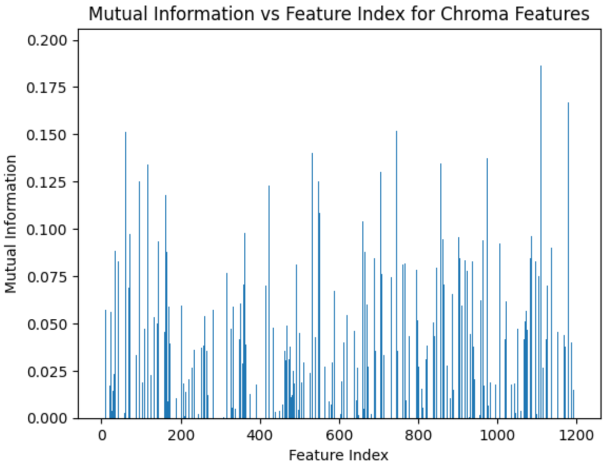
## Observations:

Mutual information of 1200 positions of 4 different features within a song with respect to the Rhythmic beat.




We can see the peaks in all graphs, peaks in a mutual information graph could signify points where certain features have high information gain with respect to the rhythmic or beat-related characteristics of the music. Here are some possible interpretations:

1. **Rhythmic Patterns**: Peaks might correspond to specific rhythmic patterns or structures in the music. Features associated with these peaks could capture aspects such as beat duration, tempo changes, or accentuations that are relevant for rhythmic analysis.
2. **Tempo Changes**: If your target variable includes information about tempo changes or variations in rhythmic complexity, peaks could indicate features that are particularly informative about these temporal aspects.
3. **Beat Strength**: Peaks might represent features that correlate strongly with the perceived strength or prominence of the beats. This could include emphasis on downbeats, upbeats, or other rhythmic accents.
4. **Instrumentation Impact**: Certain features might be more informative about rhythmic characteristics based on the instrumentation used in the music. For

example, percussion-heavy sections might contribute more to certain rhythmic features.

5. **Transitions and Segments**: Peaks could signify transitions between different rhythmic segments or patterns within the music. Features capturing these transitions might be crucial for rhythmic analysis, especially in music with complex rhythmic structures.
6. **Useful for Segmentation**: Features associated with peaks could be valuable for segmenting the music into different rhythmic sections or phrases. These segments might align with changes in the rhythmic structure of the music.

The peaks in the graphs tend to indicate the completion rhythmic beat cycle and starting of next cycle. Those are the points where we/model gets the maximum information.

The observed hierarchy in model performance aligns with the inherent strengths and complexities associated with each model. The CNN, being specifically designed for image data, excelled in capturing intricate patterns. The Random Forest, with its ensemble approach, demonstrated adaptability to diverse features. Meanwhile, Decision Trees, while robust, faced challenges in handling the complexity of the dataset.

These findings offer valuable insights into the suitability of each model for our project's objectives. The superior performance of the CNN suggests its effectiveness in capturing rhythmic beat patterns within music annotations, providing a strong foundation for future iterations.

# V. Conclusion:

In conclusion, our endeavor to develop a computational model for predicting the rhythmic beat pattern of songs and fitting them into a tala has yielded promising results and significant insights. Through the utilization of Convolutional Neural Network (CNN), Decision Trees, and Random Forest models, we aimed to discern patterns within music images and enhance our understanding of rhythmic structures.

The performance comparison among the models revealed a hierarchy, with the CNN emerging as the top-performer, followed by Random Forest and Decision Trees. The CNN, designed for image-related tasks, demonstrated exceptional proficiency in capturing intricate patterns, aligning seamlessly with the complexities of our music audio dataset. The ensemble approach of Random Forest showcased commendable adaptability to diverse features, while Decision Trees, albeit robust, faced challenges in handling the dataset's complexity.

# VI. References:

Dataset : https://compmusic.upf.edu/carnatic-rhythm-dataset
Librosa: https://librosa.org/doc/latest/index.html
Scikit-Learn: https://scikit-learn.org/stable/