CHAPTER -1
INTRODUCTION

1.1) Introduction

In the rich tapestry of global linguistic diversity, numerous languages remain underrepresented in the digital realm, despite their profound cultural and historical significance. Among these are Assamese and Mising, two prominent languages spoken in the northeastern region of India. Assamese, an Indo-Aryan language with a literary tradition spanning over a millennium, serves as a medium of communication for millions of people in Assam and neighbouring regions. Mising, a Tibeto-Burman language, holds a special place in the cultural identity of the Mising community, one of the largest indigenous groups in Assam. Both languages boast unique phonetic, prosodic, and syntactic characteristics that reflect the vibrant heritage of their speakers. However, despite their cultural importance, they remain relatively under-resourced in terms of technological development and digital documentation, creating a significant gap in the application of modern speech technologies.

Speaker recognition, a subfield of speech technology, has garnered considerable attention for its potential applications in security systems, personalized services, forensic analysis, and human-computer interaction. The core of speaker recognition lies in the ability to distinguish between speakers based on individual speech patterns, which are influenced by both physiological and behavioural factors. These patterns can be captured through various acoustic and prosodic features, such as pitch, intensity, duration, and spectral characteristics. While substantial progress has been made in speaker recognition for resource-rich languages like English, Mandarin, and Spanish, languages with limited digital presence, such as Assamese and Mising, continue to pose unique challenges. The scarcity of annotated speech corpora, linguistic tools, and language-specific research makes the development of robust speaker recognition systems for these languages a complex endeavor.

This study aims to address this gap by exploring the application of machine learning algorithms to develop a speaker recognition system tailored for Assamese and Mising. The research focuses on the extraction and analysis of both prosodic and acoustic features to determine their effectiveness in distinguishing between speakers in low-resource language contexts. By employing a comparative approach, the study seeks to identify language-specific traits that may enhance the accuracy of speaker recognition systems. The selection of machine learning models, including traditional algorithms like Support Vector Machines (SVMs) and contemporary deep learning architectures such as Long Short-Term Memory (LSTM) networks, allows for a comprehensive evaluation of different methodologies. The findings of this research not only contribute to the advancement of speaker recognition technology for underrepresented languages but also provide valuable insights into the linguistic characteristics of Assamese and Mising.

Furthermore, the implications of this study extend beyond technological innovation. In an era where linguistic diversity faces the threat of homogenization due to globalization and technological advancement, research that focuses on low-resource languages plays a vital role in preservation and empowerment. By developing speech technologies for Assamese and Mising, this work supports the continued relevance and use of these languages in the digital age. It contributes to the broader goal of inclusive technology development, ensuring that linguistic minorities are not left behind in the rapid evolution of speech-based applications. The successful implementation of speaker recognition systems for Assamese and Mising could pave the way for similar initiatives targeting other underrepresented languages globally, fostering a more linguistically diverse and culturally rich technological landscape.

(1.2) Basics of Language

A language is a systematic means of communication used by a particular community or group. At its core, a language comprises several fundamental elements that work together to enable meaningful expression and understanding. The basics of a language can be broadly categorized into phonetics, grammar, vocabulary, and writing system. Each of these components plays a vital role in shaping how a language functions and evolves.

Phonetics refers to the study of the sounds used in a language. Every language has a unique set of phonemes, which are the basic units of sound that distinguish one word from another. These sounds are produced through various articulations of the vocal apparatus, including the movement of the tongue, lips, and vocal cords. The phonetic characteristics of a language contribute to its distinct auditory identity and influence how it is perceived by speakers and listeners alike. For example, Assamese is known for its soft and melodic sounds, while Mising has a set of unique phonemes that reflect its Tibeto-Burman origins.

Grammar is the structured framework that governs how words are formed and combined into meaningful sentences. It encompasses syntax, which deals with the arrangement of words and phrases, and morphology, which focuses on the internal structure of words and their formation through roots, prefixes, and suffixes. Grammar provides the rules that ensure clarity and coherence in communication. In Assamese and Mising, grammatical structures differ significantly from Indo-European languages, reflecting their unique linguistic heritage and influencing how speakers express ideas and relationships between entities.

Vocabulary represents the collection of words and phrases used in a language. Words carry meaning and are the building blocks of communication. A language's vocabulary evolves over time, influenced by cultural, historical, and social factors. Both Assamese and Mising have rich vocabularies that reflect the cultural practices, environment, and traditions of their speakers. Words related to agriculture, nature, and community life are particularly prominent, highlighting the close connection between these languages and the lifestyle of their speakers.

Finally, many languages have a writing system that allows for the visual representation of speech. Writing systems can take various forms, such as alphabets, syllabaries, or logographic systems. The Assamese language uses the Assamese script, which is derived from the ancient Indian Brahmi script and adapted to represent the specific phonetic features of the language. Mising, on the other hand, has historically used oral traditions for communication and cultural preservation. Efforts to develop a written form of Mising have been undertaken in recent times to support literacy and documentation, often using the Latin alphabet or adapting existing scripts to accommodate its phonetic characteristics.

(1.2) Basics of Language

A language is a systematic means of communication used by a particular community or group. At its core, a language comprises several fundamental elements that work together to enable meaningful expression and understanding. The basics of a language can be broadly categorized into phonetics, grammar, vocabulary, and writing system. Each of these components plays a vital role in shaping how a language functions and evolves.

Phonetics refers to the study of the sounds used in a language. Every language has a unique set of phonemes, which are the basic units of sound that distinguish one word from another. These sounds are produced through various articulations of the vocal apparatus, including the movement of the tongue, lips, and vocal cords. The phonetic characteristics of a language contribute to its distinct auditory identity and influence how it is perceived by speakers and listeners alike. For example, Assamese is known for its soft and melodic sounds, while Mising has a set of unique phonemes that reflect its Tibeto-Burman origins.

Grammar is the structured framework that governs how words are formed and combined into meaningful sentences. It encompasses syntax, which deals with the arrangement of words and phrases, and morphology, which focuses on the internal structure of words and their formation through roots, prefixes, and suffixes. Grammar provides the rules that ensure clarity and coherence in communication. In Assamese and Mising, grammatical structures differ significantly from Indo-European languages, reflecting their unique linguistic heritage and influencing how speakers express ideas and relationships between entities.

Vocabulary represents the collection of words and phrases used in a language. Words carry meaning and are the building blocks of communication. A language's vocabulary evolves over time, influenced by cultural, historical, and social factors. Both Assamese and Mising have rich vocabularies that reflect the cultural practices, environment, and traditions of their speakers. Words related to agriculture, nature, and community life are particularly prominent, highlighting the close connection between these languages and the lifestyle of their speakers.

Finally, many languages have a writing system that allows for the visual representation of speech. Writing systems can take various forms, such as alphabets, syllabaries, or logographic systems. The Assamese language uses the Assamese script, which is derived from the ancient Indian Brahmi script and adapted to represent the specific phonetic features of the language. Mising, on the other hand, has historically used oral traditions for communication and cultural preservation. Efforts to develop a written form of Mising have been undertaken in recent times to support literacy and documentation, often using the Latin alphabet or adapting existing scripts to accommodate its phonetic characteristics.

(1.3) About the Languages

(1.3.1) Assamese

Assamese is an Eastern Indo-Aryan language primarily spoken in the state of Assam in India. It has a rich history and a unique phonetic system that sets it apart from other languages. Here's an overview of the Assamese language, including its phonetics.

- **History and Origin**

  Assamese evolved from the ancient Kamrupi dialect of Eastern Magadhi Prakrit and has been influenced by Sanskrit and Pali. Early inscriptions and texts indicate its use for administrative and religious purposes as early as the 7th century CE. The language has a rich literary tradition dating back to the 14th century. In October 2024, Assamese was declared a ‘Classical’ language, recognizing its ancient origins and rich literary heritage.

- **Script and Documentation**

  Assamese is written using the Assamese script, an abugida system derived from the ancient Indian Brahmi script. It has evolved over time and is very similar to the Bengali script but differs in a few letters. For example, Assamese has an additional letter ৱ (vo) and uses a separate letter for the sound 'ro' ৰ. The letter ক্ষ (khya) in Assamese is an individual consonant with its own phonetic quality

- **Dialects**

  Assamese has several regional dialects that differ in pronunciation, syntax, and vocabulary. Major dialect groups include the Eastern Dialect, Central Dialect, Kamrupi Dialect, and Goalporia Dialect. The Kamrupi dialect is one of the two western dialect groups of Assamese, the other being Goalpariya. It has three subdialects: Barpetia, Nalbariya, and Palasbaria. Kamrupi differs from Eastern Assamese in various aspects, including adverbial formations, pleonastic suffixes, and verb formations.

- **Language Features**

  Assamese is known for its soft and melodic sounds. It has highly inflected forms and different pronouns for honorific and non-honorific constructions. Unlike Bengali and Oriya, Assamese does not have long or short vowels for /i/ or /u/, although different letters exist for long and short vowels. These letters are used rather loosely, and the spelling does not always reflect the pronunciation. Assamese also has a unique guttural fricative /χ/ sound, represented by the IPA letter, which is not found in most Tibeto-Burman languages of Northeast India or in other major Indo-Aryan languages in India except Kashmiri, Sinhalese, and some minor languages of West India and the Himalayas.

(1.3.2) Phonetics of Assamese

- **Vowels**
- Mising has a set of vowels that include both short and long sounds. Some key vowel sounds include:
- অ' (A) is pronounced like 'o' in 'boy', not like 'a' as in 'up' as in Sanskrit or Hindi. This pronunciation is similar to Bengali. Baden Powell noted that in Assam and Bengal, the 'o' sound is generally the result of the dialectic pronunciation of the 'a' in the Sanskrit alphabet. To avoid confusion, Assamese should write 'o' for 'A' in Roman script

- 'ে' (e) has two sounds like the English words 'get' and 'gate'. However, these are not differentiated in Assamese traditional spelling or dictionaries; one simply has to know when speaking. In Roman script writing, they are differentiated for foreign speakers, with one written as 'e' and the other as 'è'.

- ও' (o) and 'ঔ' (ou) represent two different vowel sounds.

- **Consonants**
- Assamese has 41 consonants. Some key consonant sounds include
- Aspirated phonemes like 'kh', 'gh', 'th', 'dh', 'ph', and 'bh' are pronounced with a strong burst of air. English speakers should be careful not to pronounce these as simple unaspirated k, g, t, d, p, b
- Assamese has several letters for the /s/ sound (c, C, শ), but all are pronounced like 's' as in 'sun'. The 'sh' or 'ch' sounds are only used in foreign words.

- Assamese does not have cerebral pronunciations for t, th, d, dh, and n, unlike other Indo-Aryan languages like Sanskrit and Hindi. In this respect, Assamese is similar to European Indo-European languages like English.
- **Other Features**
- Assamese also uses nasal notation (í), called 'sondrobindu', above vowels as well as consonants in the Assamese script. In Romanization, the sign (¨) for (í) will be shown above vowels only.
- The following are some typical Assamese words showing the basic sounds of the language:

- amar(Aamar): our
- bhoiam (QBR^am): plains
- èdin(wedn): one day
- ebar(wbar): one time
- gwhali(qgahael): cow shed
- ghonai (GnahH): frequently

  Assamese has a total of 52 graphemes, divided into 11 vowels and 41 consonants. Some of the unique characters in Assamese include ৰ (ro), ৱ (wo), and ক্ষ (khy). The Assamese script also has three graphemes for the /s/ sound (c, C, শ), but all are pronounced like 's' as in 'sun'.

(1.3.3) Mising

Mising is a Tibeto-Burman language primarily spoken by the Mising community, one of the largest indigenous groups in Assam, India. It holds a special place in the cultural identity of the Mising people and has a rich oral tradition. Here's an overview of the Mising language, including its phonetics:

- **History and Origin**

  The Mising language belongs to the Sino-Tibetan language family and has evolved within the cultural and geographical context of Northeast India. It has a strong oral tradition, with literature and knowledge passed down through generations via songs, folktales, and rituals. Efforts to document and preserve the language have gained momentum in recent decades, reflecting the community's commitment to linguistic heritage.

- **Script and Documentation**
  Traditionally, Mising has been an oral language. In the 20th century, a script based on the Latin alphabet was developed to represent Mising sounds, aiding in literacy and documentation. This Romanized script helps in writing Mising, though the language continues to thrive mainly through its spoken form. The Mising community has also worked on creating a script that aligns with their phonetic system, though it is not as widely used as the Latin-based one.

- **Dialects**
  Mising has several dialects that vary slightly across different regions where the Mising people reside. These dialects differ in pronunciation and vocabulary but remain mutually intelligible. The major dialects include those spoken in the districts of Sonitpur, Dhemaji, Lakhimpur, and Sivasagar. Despite these variations, the Mising language maintains a strong sense of community and identity among its speakers.

- **Language Features**
  Mising is known for its tonal nature, where pitch and intonation play crucial roles in conveying meaning. It has a distinct set of phonemes and grammatical structures that set it apart from Indo-Aryan languages like Assamese. The language uses a subject-verb-object (SVO) word order and has a rich system of honorifics that reflect social relationships and respect. Reduplication (repeating words or parts of words) is commonly used to emphasize meaning or convey plurality.

(1.3.4) Phonetics of Mising

- **Vowels** Mising has a set of vowels that include both short and long sounds. Some key vowel sounds include:
- 'a' pronounced like 'a' in 'father'
- 'e' pronounced like 'e' in 'bet'
- 'i' pronounced like 'i' in 'sit'
- 'o' pronounced like 'o' in 'pot'
- 'u' pronounced like 'u' in 'put'
- Long vowels such as 'ā', 'ē', 'ō' which are held for a longer duration.
- **Consonants** Mising has a variety of consonant sounds, including
- Aspirated consonants like 'ph', 'th', 'kh' which are pronounced with a strong burst of air.
- Unaspirated consonants like 'p', 't', 'k' which are pronounced without the burst of air.

- Unique sounds like the retroflex consonants (e.g., ṭ, Ḍ) which are pronounced with the tongue tip curled back.
- Fricatives like 's' and 'h', where 'h' can sometimes have a breathy quality.
- The Mising language also has a glottal stop, represented by ‘ʔ’, which is a sudden stoppage of airflow in the vocal tract. This can occur between vowels or at the beginning of a word.

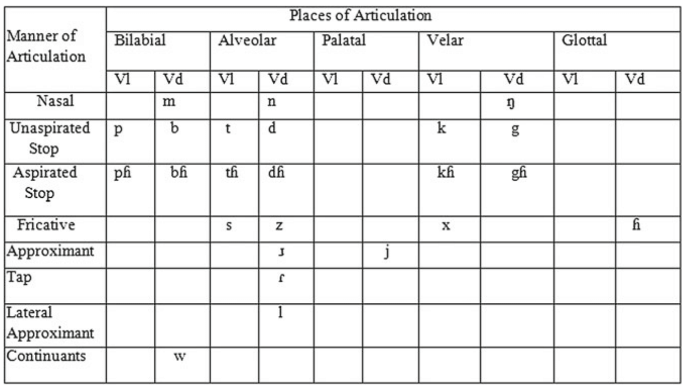
- **Other Features** Mising uses reduplication, where words or parts of words are repeated to convey different meanings or emphasize certain aspects. For example, the word 'bo' (water) can be reduplicated as 'bo-bo' to mean 'a little water.' The language also has a system of classifiers that are used when counting or referring to nouns, similar to other Tibeto-Burman languages. These classifiers vary based on the type of object being counted (e.g., people, animals, objects).

- Here are some typical Mising words showing the basic sounds of the language:

- nying (I/me)
- nedum (we/us)
- garing (sun)
- doring (moon)
- mop (child)
- mising (Mising person)
- bo (water)
- kholing (mountain)

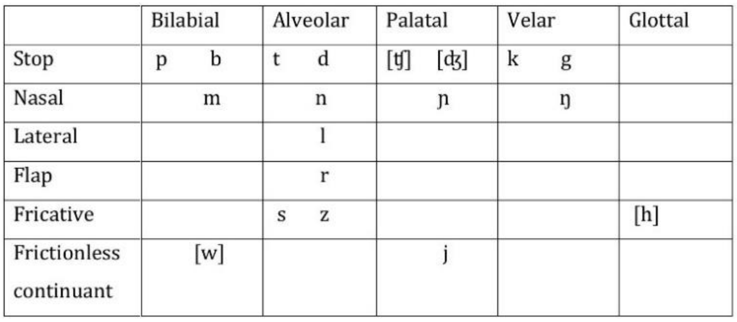
- Mising has around 35 graphemes, including vowels and consonants. Some unique features include the use of tone to distinguish words and the presence of retroflex consonants, which are not found in many other languages of the region. The Latin-based script uses diacritics to represent tones and unique consonant sounds

.



Manner of Articulation	Places of Articulation									
	Bilabial		Alveolar		Palatal		Velar		Glottal	
	Vl	Vd	Vl	Vd	Vl	Vd	Vl	Vd	Vl	Vd
Nasal		m		n				ŋ		
Unaspirated Stop	p	b	t	d			k	g		
Aspirated Stop	pɦ	bɦ	tɦ	dɦ			kɦ	gɦ		
Fricative			s	z			x			ɦ
Approximant				ɹ		j				
Tap				ɾ						
Lateral Approximant				l						
Continuants		w								

Fig.1.3.1(a): Table Depicts Various Phenomes of Assamese Language



	Bilabial	Alveolar	Palatal	Velar	Glottal
Stop	p b	t d	[ʧ] [ʤ]	k g	
Nasal	m	n	ɲ	ŋ	
Lateral		l			
Flap		r			
Fricative		s z			[h]
Frictionless continuant	[w]		j		

Fig.1.3.1(b): Table Depicts Various Phenomes of Mising Language

Chapter – 2
THEORITICAL BACKGROUNDS

(2.1) Introduction to Speaker Recognition

In our exploration of speaker recognition, we delve into a technology that enables machines to identify or verify individuals based on their voice characteristics—not only by how they sound but also by the language they speak. Much like how we humans can often guess who's speaking and even where they're from based on accent or word choice, we're now teaching machines to leverage both voice and linguistic cues for recognition. This advancement is especially vital in multilingual contexts, enhancing both security and accessibility in voice-driven applications

(2.2) The Basics of How Speaker Recognition Works (with Language Awareness)

Voice as an Identifier

We understand that every person's voice is shaped by factors such as vocal tract anatomy, speaking style, emotional state—and importantly, native language and pronunciation patterns. These linguistic traits act like a vocal fingerprint, making language-based modelling a valuable dimension in speaker recognition.

The Two Modes of Speaker Recognition

Verification: The system verifies whether a speaker is who they claim to be, using both vocal characteristics and language-specific cues (e.g., accent or phoneme usage).

Identification: Here, the system listens to the speech and identifies the individual, potentially leveraging language as an additional discriminator—especially helpful in diverse populations or multilingual datasets.

(2.3) The Technical Process

We begin with the raw speech signal, which may include ambient noise or recording inconsistencies. The first step is to clean up the signal through noise reduction to remove background interference that could mask subtle language cues, followed by normalization to ensure uniform volume and quality across varying speech samples.

Once the audio is refined, we extract features that reflect both the speaker's identity and their linguistic profile. Pitch can vary due to language-specific intonation patterns. Timbre represents the speaker's unique tonal quality. Rhythm helps capture the prosodic features of different languages, and phonetic structure highlights how phonemes are pronounced based on the speaker's language or dialect**.**

(2.4) Key Technologies and Algorithms

(2.4.1) Machine Learning Algorithms Machine learning is central to our work in speaker recognition. We utilize algorithms like Gaussian Mixture Models (GMM) and Hidden Markov Models (HMM) to analyse and recognize voice patterns. These algorithms learn from large datasets of voices, improving their accuracy over time

- **Gaussian Mixture Models (GMM)**
  We use GMMs, which are statistical models assuming all data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters. The model is trained to find the parameters that best fit the data. The probability density function for a GMM is given by:

  $$P(x)=\sum_{i=1}K\omega_i \mathrm{N}(x|\mu_i,\Sigma_i)$$

  Where:

  ωi is the weight of the i-th Gaussian component.

  $\mathrm{N}(x|\mu i,\Sigma i)$ is the Gaussian distribution with mean μi and covariance Σi.

- **Hidden Markov Models(HMM)**
  HMMs are statistical models that we use to represent the probabilistic relationship between observed speech features and hidden states. These states can represent phonetic units or speakers. An HMM is defined by:

  A set of states $S=\{s1,s2,...,sN\}$

  A set of observations $O=\{o1,o2,...,oT\}$

  Transition probabilities $A=\{aij\}$, where aij is the probability of transitioning from state si to state sj.

  Emission probabilities B={bj(k)}, where bj(k) is the probability of observing symbol k in state sj.

  Initial state probabilities π={πi}, where πi is the probability of starting in state si

- **Deep Learning Approaches**
  We also leverage deep learning, a subset of machine learning that uses neural networks with multiple layers to process voice data. These networks can automatically learn complex patterns in speech, leading to more accurate recognition. It's akin to teaching a computer to recognize voices by exposing it to millions of examples.

- **Convolution Neural Networks (CNNs)**
  We employ CNNs, which are particularly effective for processing grid-like data such as images and spectrograms of speech signals. They use convolutional layers to automatically and adaptively learn spatial hierarchies of features.

- **Recurrent Neural Networks (RNNs)**
  RNNs are our go-to for handling sequential data, making them suitable for speech processing where temporal information is crucial. They have loops that allow information to persist, enabling the network to consider previous inputs when processing current ones.

(2.5) Mathematical Concepts in Speaker Recognition

- **Euclidean Distance**
  In language identification, Euclidean distance is often used to measure how different two speech samples are based on their acoustic or prosodic feature vectors.
  For example, suppose we represent language-specific characteristics (like phoneme distributions or rhythm features) as vectors. For two such vectors *x* and *y*, the Euclidean distance

  $d(x, y) = \sqrt{\sum(xi - yi)^2}$

  tells us how far apart the two language profiles are. A smaller distance indicates greater similarity, suggesting the input sample closely resembles a known language profile in the system.

- **Cosine Similarity**
  Cosine similarity is valuable in language identification as it focuses on the direction of the feature vectors rather than their magnitude. This is particularly helpful when working with normalized features like i-vectors or x-vectors that capture language characteristics.

  $\cos(\theta)=(A \cdot B)/(||A||||B||)$

  reveals how aligned the input voice is with a language prototype vector. Higher values indicate the speaker is likely using a language with similar phonotactic or prosodic patterns.

- **Likelihood Ratio**
  In a probabilistic language identification system, we use likelihood ratios to determine whether the speech segment belongs to a particular language. The likelihood ratio is defined as:

  $LR=P(Data|Hypothesis_1)/P(Data|Hypothesis_0)$

  where $Hypothesis_1$ assumes the utterance is in the target language, and $Hypothesis_0$ assumes it belongs to an alternative or competing language. If the ratio exceeds a predefined threshold, the system Favors classifying the input as belonging to the hypothesized language.

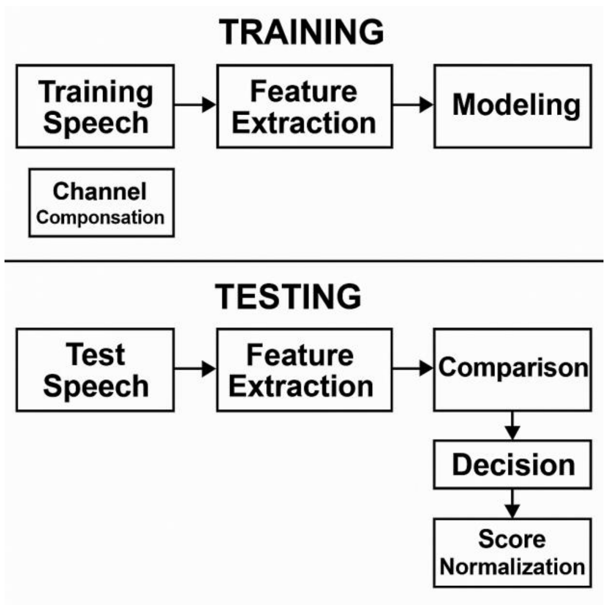


Fig 2.5: Depicts the Working of a Speaker Recognition System Through a Block Diagram

(2.6) Prosodic and Phonetic Analysis of Languages

In our exploration of speech and language, prosodic and phonetic analysis emerge as crucial components. While prosody focuses on the rhythmic and melodic aspects of speech—such as intonation, stress, and rhythm—phonetics examines the physical and acoustic properties of speech sounds themselves. Together, they help us uncover how people convey meaning, emotion, and structure beyond just the sequence of words.

(2.6.1) Key Elements of Prosodic and Phonetic Analysis

Intonation
We study intonation to track pitch movement across an utterance. These variations not only indicate speech functions (questions, statements, exclamations) but also reflect emotional states. Phonetically, these pitch contours relate to the fundamental frequency, which can be precisely measured and modelled to capture communicative nuances.

Stress
Stress analysis helps us identify which syllables or words carry greater prominence in speech. Shifts in stress can alter meaning or intent. Phonetically, stressed syllables tend to have greater intensity, longer duration, and higher pitch—making them measurable and classifiable using acoustic tools.

Rhythm
Rhythmic analysis reveals the timing and flow of speech, including how syllables are spaced and grouped. In stress-timed languages like English, stressed syllables occur at regular intervals; in syllable-timed languages like Spanish, each syllable takes approximately the same time. Phonetic analysis complements this by quantifying duration patterns across syllables and identifying rhythmic tendencies.

Pauses and Silences
Both prosodically and phonetically, pauses and silences provide insight into cognitive and structural processes in speech. They might mark grammatical boundaries, indicate hesitation, or help shape discourse.

(2.7) Why Prosodic and Phonetic Analysis Matter

Emotional Expression
Prosody is deeply tied to the emotional tone of speech. Phonetic cues such as pitch variability, loudness, and speed give measurable evidence of emotional states—enabling more robust emotion recognition systems.

Sentence Structure
Prosodic features help listeners infer syntactic boundaries. Pauses, pitch resets, and changes in speech rate all signal transitions between phrases or clauses. Phonetic analysis detects these transitions with precision, strengthening natural language understanding in machines.

Language and Dialect Specificity
Different languages and dialects exhibit unique prosodic and phonetic patterns—such as specific pitch ranges, stress rules, or phoneme realization. Analysing these aspects provides insight into a language's identity and helps differentiate similar-sounding languages or dialects in automatic systems.

(2.8) Technical Aspects of Prosodic and Phonetic Analysis

Feature Extraction
We extract both prosodic features (like pitch contours, energy levels, and syllable timing) and phonetic features (formant frequencies, spectral patterns, segmental durations). Tools like Praat, OpenSMILE, or Kaldi are used to capture and analyse these characteristics systematically.

Acoustic Analysis
This involves measuring parameters such as fundamental frequency, intensity, and temporal structure. Phonetic aspects like voice onset time, vowel space area, and phoneme transitions offer additional cues for tasks like language identification or speaker recognition.

Machine Learning Applications
By training models on prosodic and phonetic features, we can build systems capable of identifying emotions, detecting language or dialect shifts, or classifying speech patterns.

.

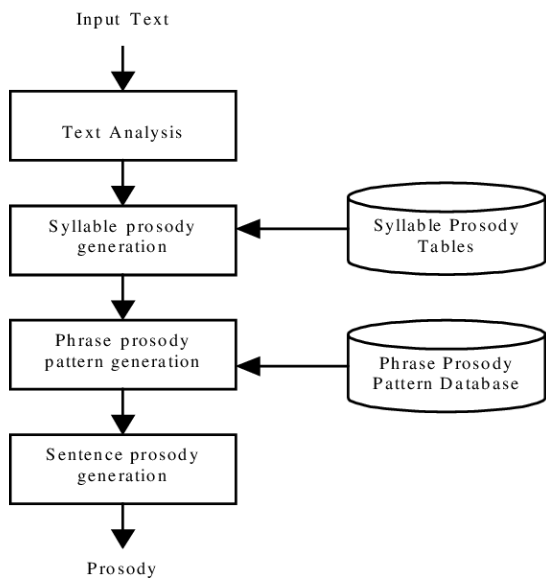

Input Text
Text Analysis
Syllable prosody generation
Syllable Prosody Tables
Phrase prosody pattern generation
Phrase Prosody Pattern Database
Sentence prosody generation
Prosody

Fig.2.8.(a): Depicts the Flow of Prosodic Analysis Through a Block Diagram

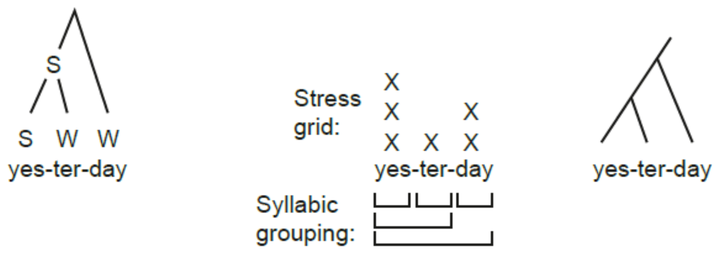


Fig.2.8.(b): Figure Showing Prosodic Analysis for the Word 'Yesterday'

CHAPTER -3
LITERATURE SURVEY

Speaker recognition has evolved from early filter-bank correlation techniques to contemporary, highly discriminative models that integrate multimodal cues. Over four decades, advancements have spanned statistical, prosodic, and deep learning domains, targeting challenges such as variability, short-duration utterances, and noise robustness [8]. A recent survey provides a comprehensive taxonomy of this evolution, outlining the development of identification, verification, and diarization systems while emphasizing their applications in biometric security and personalized services [1].

Central to these systems are feature extraction techniques. While Mel Frequency Cepstral Coefficients (MFCCs) have remained the standard for decades, comparative analyses reveal that Linear Frequency Cepstral Coefficients (LFCCs) achieve higher resilience to reverberation and outperform MFCCs for female voices [2]. In a similar vein, the inclusion of prosodic and rhythmic features has garnered attention. Studies leveraging Rhythm Formant Analysis (RFA) have successfully distinguished between Mising and Assamese—two rhythmically similar yet distinct languages—by exploiting low-frequency temporal patterns that capture syllabic structure and speech timing [3], [6].

Efforts to model such linguistic variation extend beyond acoustic features. Support Vector Machines (SVMs) trained on high-level features such as prosody and pronunciation have proven superior to baseline likelihood-ratio models in speaker verification [4]. Beyond individual models, ensemble methods have shown remarkable improvements in classification accuracy. In particular, stacking ensemble learners such as Random Forests, Extra Trees, and Gradient Boosting, when combined with a Logistic Model Tree meta-learner, consistently outperform their base components across multiple datasets [10]. Random Forests remain a core classifier due to their ability to manage missing data, rank variable importance, and prevent overfitting by design [11].

To improve both computational efficiency and accuracy, modified k-Nearest Neighbor (kNN) models that selectively use representative instances while adapting k-values have been introduced. These models significantly reduce memory overhead and preserve competitive performance across traditional classification tasks [12]. Complementing these machine learning advances, logistic regression continues to serve as a robust statistical classifier for binary outcomes. It offers interpretability through logit transformation and odds ratio computation, particularly when feature distributions deviate from parametric assumptions. Guidelines for reporting and validating logistic models have been instrumental in standardizing usage across disciplines, including education and behavioral science [13].

Multimodal approaches further bolster speaker recognition in adverse acoustic environments. The integration of speech and visual input—such as lip movements—within convolutional neural network (CNN) frameworks has achieved superior performance under noisy and spoof-prone conditions. Feature-level fusion strategies in these systems have demonstrated promising results in biometric authentication and surveillance use cases [5]. Concurrently, the landscape of speaker embedding has shifted from statistical architectures like Gaussian Mixture Model-Universal Background Model (GMM-UBM) and Joint Factor Analysis (JFA) toward discriminative representations such as x-vectors and d-vectors. These newer systems improve speaker discrimination over short utterances, but challenges related to domain generalization and low-resource adaptation persist [7].

Lastly, machine learning literature has supplied a firm foundation for integrating such classifiers into speech recognition pipelines. Broad reviews categorize existing algorithms into supervised, unsupervised, semi-supervised, and reinforcement learning, providing taxonomies that are particularly valuable for designing end-to-end systems capable of handling noisy, under-resourced, or multi-speaker audio streams [9]. Together, these findings lay the groundwork for building customized, rhythm-aware transcription systems for Assamese and Mising, emphasizing the synthesis of ensemble learning, rhythmic-prosodic modeling, and robust feature engineering.

CHAPTER -4
METHOLOGY FOR DATA COLLECTION

(4.1) List of Data and Tools Used

- **Voice Samples**
  Standard Language Recording (Assamese)
  Native Language Recording (Mising)

- **Hardware Tools**
  Mic Recorder
  Laptop (Display Unit)
  Necessary Cables Like USB Type C , 3.5mm Jack Plugs

- **Software Tools**
  PRAAT (Voice Analyzer Software)
  Audacity & OBS Studio (Noise Removal And Audio Amplification)
  M.S. Excel (For Storing Information of speakers)

(4.2) Description of Dataset and tools

- **Voice Samples**

Standard Language Recording (Assamese)

In our project, where the goal is to develop speech recognition systems for the Mising language, Assamese has been chosen as the standard language due to its availability of well-established linguistic resources, large speech datasets, and standardized pronunciation norms. Assamese, as the widely spoken language of Assam, provides a strong foundation for training machine learning models, as it offers comprehensive data for model development. Using Assamese as the standard language helps in addressing challenges like dialectal variations within Mising, as it provides a unified form of communication and ensures the recognition system is robust and effective. Additionally, leveraging Assamese allows for easier integration with broader applications and services, making it more accessible to a wider audience. Once the system is well-developed in Assamese, it can be refined and adapted for Mising, ensuring accurate recognition while also preserving the linguistic integrity of these indigenous languages.

Subject of Recording

For our subject of recording we chose a common folk tale in Assamese Literature named "বেলি আৰু বতাহ" (Translation : " THE NORTH WIND AND THE SUN"). We Chose this subject as it's been heard by almost all age groups of people living in Assam .

বেলি আৰু বতাহ :

এদিন বেলি আৰু বতাহৰ মাজত নিজৰ শক্তি আৰু সামৰ্থক লৈ এখন তৰ্ক যুদ্ধ লাগিছিল। বতাহে কয় যে তেওঁৰহে বল বেছি, তেওঁতকৈ বলবান নাই। বেলিয়ে বতাহৰ কথাত প্ৰতিবাদ জনাই ক'লে যে বেলিতকৈ জগতত কোনো ডাঙৰ থাকিব নোৱাৰে, কাৰণ তেওঁৰ বলহে বেছি। বেলি আৰু বতাহৰ মাজত এই তৰ্কযুদ্ধখন বহুসময় ধৰি চলিছিল। বহু সময় তৰ্ক কৰাৰ অন্তত বিষয়টো সিহঁতে পৰীক্ষা কৰি চাবলৈ মন কৰিলে। দুয়ো ঠিক কৰিলে যে কাৰ বেছি বল সেইটো সিহঁতে পৰীক্ষা কৰি চাব। তেতিয়াহে কোন সঁচা বলী ওলাই পৰিব। যথা সময়ত সিহঁতে বিচৰা ধৰণে বিষয়টোও পালে। কাৰণ সেই সময়তে সেই দিশৰে এজন বাটৰুৱা আহি আছিল। বাটৰুৱাজনক দেখি বতাহে ক'লে– এই যে বাটৰুৱাজন গৈ আছে যিয়ে তেওঁৰ গাৰ কাপোৰখন খুলিব পাৰিব তেওঁকেই শ্ৰেষ্ঠ বলী বুলি ধৰি লোৱা হ'ব। বতাহৰ কথাত বেলিয়ে মান্তি হ'ল। বেলিয়ে প্ৰথমতে বতাহকে তাৰ শক্তি প্ৰদৰ্শনৰ বাবে আহ্বান জনালে। বতাহে তাৰ সমষ্ট শক্তি প্ৰয়োগ কৰি বাটৰুৱাজনৰ গাৰ পৰা কাপোৰখন এৰুৱাবলৈ চেষ্টা কৰিলে। যিমানে বতাহে চেষ্টা কৰে, সিমানে মানুহজনে গাৰ কাপোৰখন টানি আঁজুৰি মেৰিয়াই লয়। বহু সময় চেষ্টা কৰিও বতাহে মানুহজনৰ কাপোৰখন আঁতৰ কৰিবলৈ সমৰ্থ নহ'ল। ইয়াৰ পিছত বেলিৰ পাল পৰিল। বতাহে বেলিক তাৰ শক্তি প্ৰদৰ্শন বাবে আহ্বান জনালে। বেলিয়ে নিজৰ তাপ বঢ়াই দিলে। মানুহজনে ৰ'দৰ তাপ সহ্য কৰিব নোৱাৰি দেহৰ পৰা বস্ত্ৰখন আঁতৰ কৰিলে আৰু গছ এজোপাৰ তলত ছাঁ ল'বলৈ বুলি বহি পৰিল। তেতিয়া বেলিয়ে বতাহক ক'লে– চোৱা মইহে শক্তি পৰীক্ষাত জয় লাভ কৰিলোঁ, তুমি হাৰিলা। বতাহেও বেলিৰ কথা মানি নিজৰ পৰাজয় স্বীকাৰ কৰি ল'লে।

English Translation :

One day, a debate broke out between the sun and the wind about their respective strength and abilities. The wind claimed that it was stronger, asserting that nothing could surpass its power. The sun, however, disagreed and protested, saying that no force in the world could be greater than its own strength. The debate continued for a long time, with both sides making their case. Eventually, they decided to settle the matter by testing their strength. They agreed to find out who was truly stronger.

At that moment, a traveller was walking by. Seeing him, the wind suggested, "Let's see who is stronger. The one who can remove his clothes will be considered the strongest." The sun agreed, and the wind made the first attempt. The wind tried using all its force to blow the man's clothes off, but the harder it blew, the more the traveler clutched his clothes tightly, shielding himself from the wind. Despite repeated efforts, the wind could not remove the man's clothing.
Next, it was the sun's turn. The sun began to shine more intensely, gradually increasing its heat. Unable to bear the intense heat, the traveler removed his clothes and sat under a tree to take shelter from the sun. At this point, the sun turned to the wind and said, "See, I have won the test of strength, and you have lost." The wind, acknowledging its defeat, accepted the sun's victory."

Mising Translation :

Longeko dooni ela esar ke aik kingii Ela (hamortho) dem Lula lukaminsul dunggai . Esar be ludung Bipak kingi yadak,bikkemyam kingi yane kamang.Dooni bi esar ke agom dem (protibad) la lukang je dooni kem yam (jagat) so sekobisin botteyane kalamang,Karan bipak kingi yadak. Dooni keela esar ke lukaminsu de dek homoi ko dung ai.dek homoi ko lukaminsul dungela bulu se(bikhoi)dem (porikha) la Kalingkang. Annide (thik) to je sekob kingi yadan Edem bul (porikha) la kaye.odopak sekob kingi yadak ed lenye.(jotha-homoi)do buluk manam bikhoi dem sin paato.Karan ed homoi o do ed lamte dok (batoruwa)Gil dung ai.(Batoruwa)dem kaala esar bi lukang-se Gil dun batoruwa dok amirr dok gasor dem sekob mepak molayen bimmmei (srestho)Emma toriksuye.esar ke Agam do dooni bisin Toriksoto. Dooni bi (prothom) do esar me aik kingi nam dem lengkanpe (Ahban) to. Esar bi aim kingiing appidem (prayug)gela (batoruwa)dok Amir dok gasor dem lapakpe sesta to. Eddik esar bi sesta dan odokkindding ko taani de Amir dok gasor dem ayyope yedsudak.dekko sesta gelasin esar be rank don attar dem laapak mola tomang.Odok ledudom dooni me aik kingi nam dem Kangkan pe (ahban)Jonahi to .dooni be aik (tap)dem borhai ligto. Tani dem dooni dok (tap) dem hoijo lamala Amir dok gasor dem laapak auto (Aru) ising kolo umyum langkape Emna tendangkang. Odo dooni bi esar me luto- kaato ngo pak kingi Yanam porikha so (joi Lav)to no harikang.esar bisin doonik agommem tadgela aik (Porajoi) (Sikar)sut

- **<u>Hardware Tools and their specifications used</u>**

  Mic Recorder :_ ZOOM H1N

  Operating Frequency : 20Hz to 20KHz.
  Input Power Source Used : (AAA Batteries)X 2
  Memory Source : Laptop Hard Drive
  Display Unit : Laptop Display

- **<u>Software Tools and their specifications used</u>**

  Voice Analyzer : PRAAT
  Sampling Rate : 22050Hz
  Bit Depth Chosen : 16 bit
  Speech Mode Used : Stereo
  File Extension Used : .WAV
  Input Device : Zoom H1N
  Input Gain Level : Optimum (Green)

- **Noise Removal And Tone Amplification :**

  a) <u>Audacity</u>

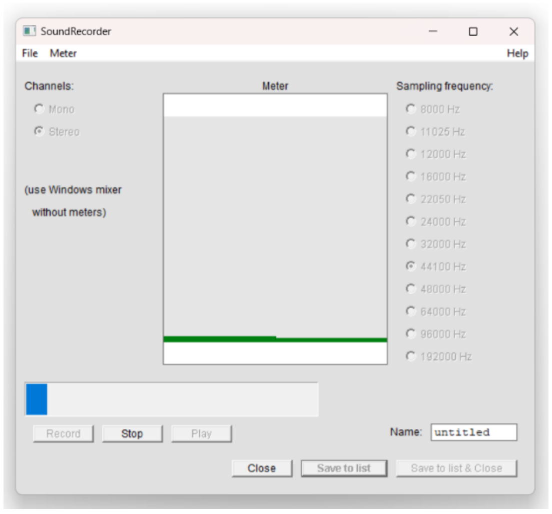
  Sampling Rate : 4410Hz
  Bit Depth : 32 bit float
  Channels : Stereo
  Track View : Wave Form
  Playback Device : Wired Earphones
  Input Device : Zoom H1N
  Filters Used : Bass Boosted Moderately , Treble Adjusted To Mild Levels.

  b) <u>OBS Studio</u>

  Sampling Rate: 44.1Khz
  Bit Rate : 320kbps
  Channel : Stereo
  Track View: Bar Level View
  Output Audio : Wired Earphones
  Input Audio : Zoom H1n
  Offset Sync : 0ms
  Filters Used : Noise Suppression ,Noise Gate , Limiter, Compressor

(4.3) Procedure

1. Quiet Noise Free Environment is selected
2. Volunteer Speakers are asked to read the standard language subject
3. Then they are asked to read the same for the low resource language
4. Both voices are recorded using the zoom h1n
5. While recording sound waves , sampling rate, frequency , output gain and all other parameters are measured in PRAAT
6. Volunteer bio data are recorded in MS Excel
7. Recorded samples are analyzed in audacity and obs studio
8. Necessary changes are made like noise removal , audio amplification for better output of the final audio
9. Samples will be used in ML Algorithms for further analysis



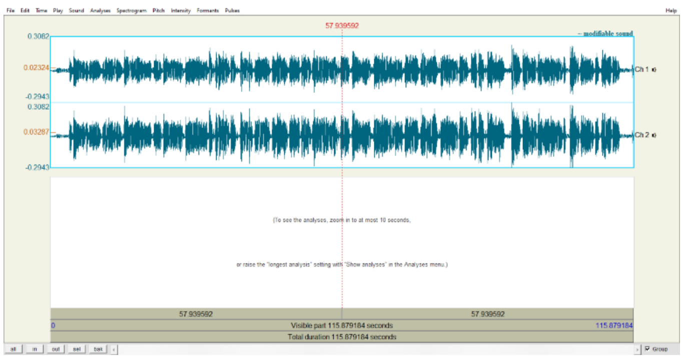
Channels: Mono [✓ Stereo]

(use Windows mixer without meters)

Sampling frequency: 8000 Hz, 11025 Hz, 12000 Hz, 16000 Hz, 22050 Hz, 24000 Hz, 32000 Hz, [✓ 44100 Hz], 48000 Hz, 64000 Hz, 96000 Hz, 192000 Hz

Name: untitled

Fig 4.3(a) Maintenance of output gain in PRAAT


File Edit Time Play Sound Analyses Spectrogram Pitch Intensity Formants Pulses
Help
57.939592
0.3082
0.02324
-0.2943
0.3082
0.03287
-0.2943
Ch 1
Ch 2
57.939592
57.939592
0
115.879184
Visible part 115.879184 seconds
Total duration 115.879184 seconds
all
in
out
sel
bak
Group

(To see the analyses, zoom in to at most 10 seconds,

or raise the "longest analysis" setting with "Show analyses" in the Analyses menu.)

Fig 4.3(b) Waveform View Of Audio Track In PRAAT

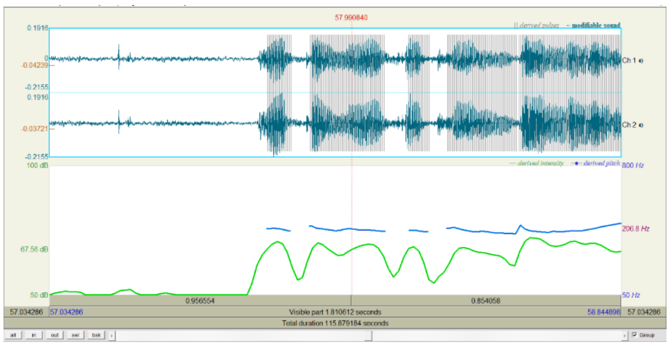

Ch 1
Ch 2
Visible part 1.810612 seconds
Total duration 115.879184 seconds

Fig 4.3(c) Spectrogram Analysis In PRAAT

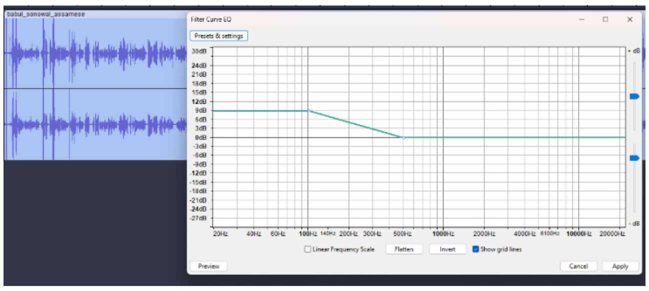


Fig 4.3(d) Audio Amplification In Audacity

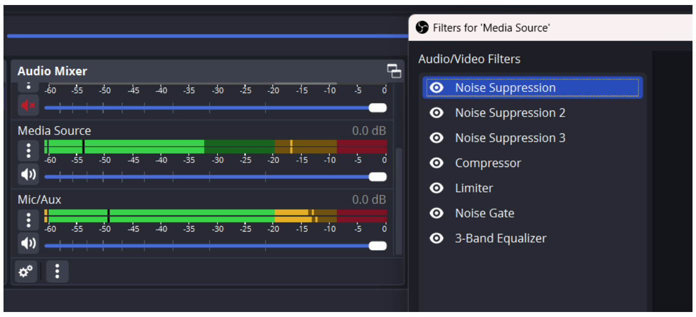
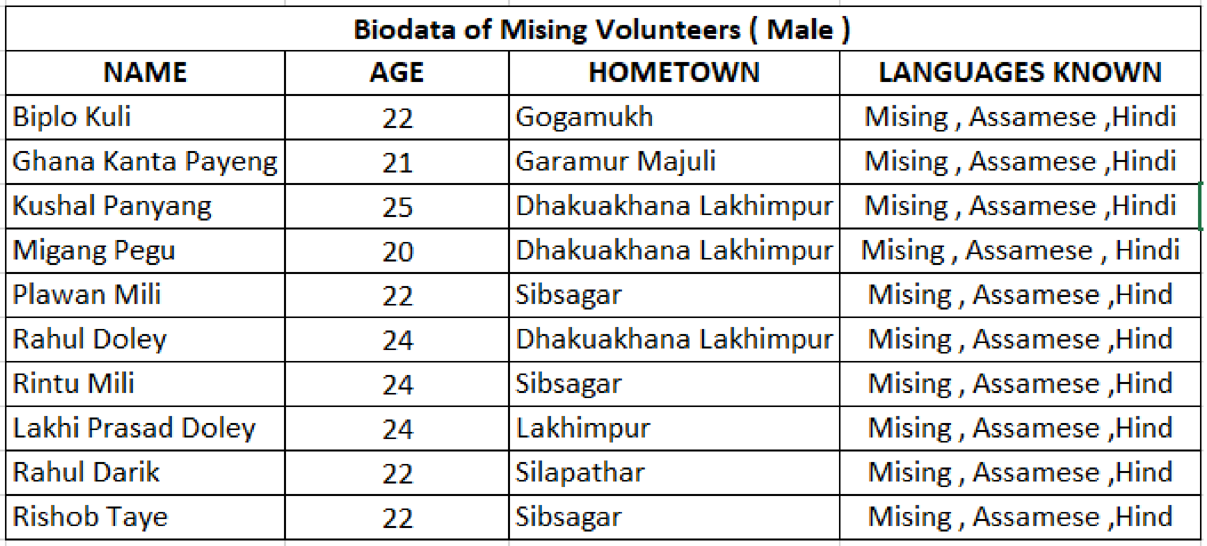


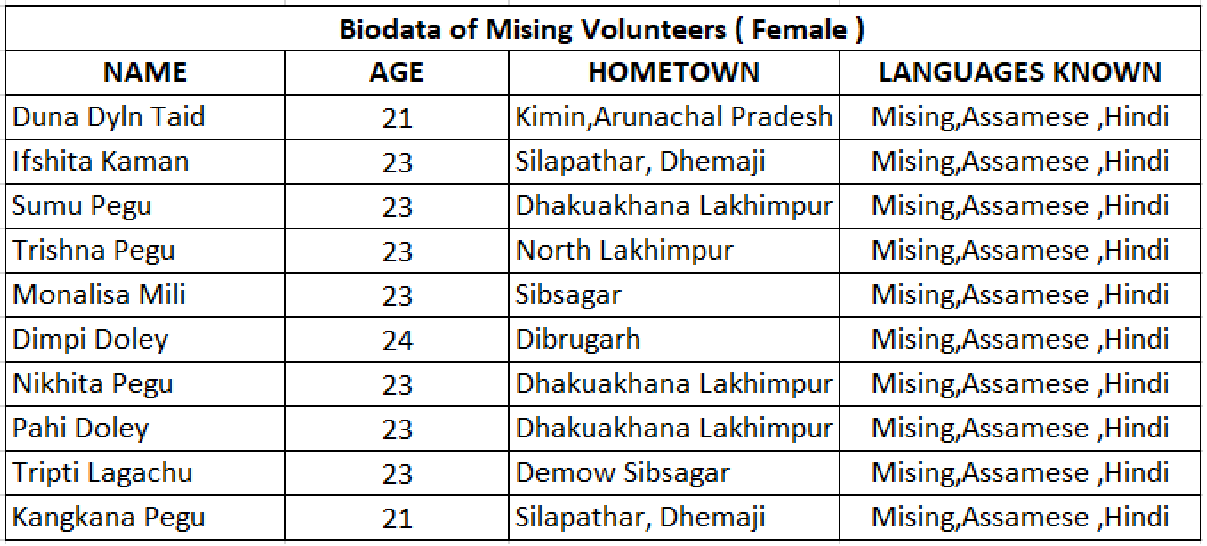
Fig 4.3(e) Noise Suppression And Equalization in OBS Studio

(4.4) <u>Excel Sheet Containing Biodata of Volunteers</u>



Biodata of Mising Volunteers (Male)			
NAME	**AGE**	**HOMETOWN**	**LANGUAGES KNOWN**
Biplo Kuli	22	Gogamukh	Mising , Assamese ,Hindi
Ghana Kanta Payeng	21	Garamur Majuli	Mising , Assamese ,Hindi
Kushal Panyang	25	Dhakuakhana Lakhimpur	Mising , Assamese ,Hindi
Migang Pegu	20	Dhakuakhana Lakhimpur	Mising , Assamese , Hindi
Plawan Mili	22	Sibsagar	Mising , Assamese ,Hind
Rahul Doley	24	Dhakuakhana Lakhimpur	Mising , Assamese ,Hind
Rintu Mili	24	Sibsagar	Mising , Assamese ,Hind
Lakhi Prasad Doley	24	Lakhimpur	Mising , Assamese ,Hind
Rahul Darik	22	Silapathar	Mising , Assamese ,Hind
Rishob Taye	22	Sibsagar	Mising , Assamese ,Hind

Fig 4.4(a) Excel Sheet Of Male Mising Volunteers



Biodata of Mising Volunteers (Female)			
NAME	**AGE**	**HOMETOWN**	**LANGUAGES KNOWN**
Duna Dyln Taid	21	Kimin,Arunachal Pradesh	Mising,Assamese ,Hindi
Ifshita Kaman	23	Silapathar, Dhemaji	Mising,Assamese ,Hindi
Sumu Pegu	23	Dhakuakhana Lakhimpur	Mising,Assamese ,Hindi
Trishna Pegu	23	North Lakhimpur	Mising,Assamese ,Hindi
Monalisa Mili	23	Sibsagar	Mising,Assamese ,Hindi
Dimpi Doley	24	Dibrugarh	Mising,Assamese ,Hindi
Nikhita Pegu	23	Dhakuakhana Lakhimpur	Mising,Assamese ,Hindi
Pahi Doley	23	Dhakuakhana Lakhimpur	Mising,Assamese ,Hindi
Tripti Lagachu	23	Demow Sibsagar	Mising,Assamese ,Hindi
Kangkana Pegu	21	Silapathar, Dhemaji	Mising,Assamese ,Hindi

Fig 4.4(b) Excel Sheet Of Female Mising Volunteers

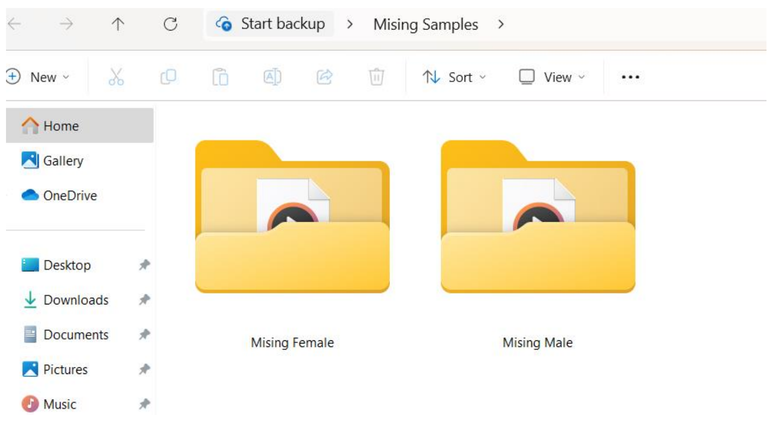


Fig 4.5(a) .wav file stored in folder

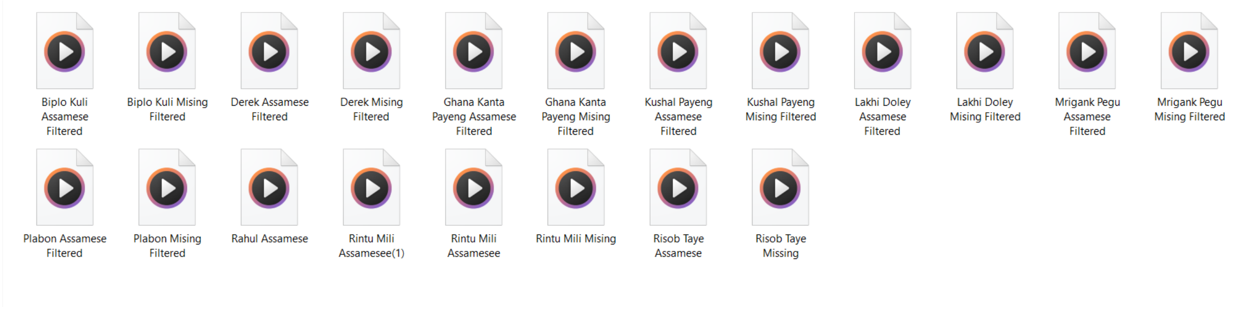

Biplo Kuli Assamese Filtered
Biplo Kuli Mising Filtered
Derek Assamese Filtered
Derek Mising Filtered
Ghana Kanta Payeng Assamese Filtered
Ghana Kanta Payeng Mising Filtered
Kushal Payeng Assamese Filtered
Kushal Payeng Mising Filtered
Lakhi Doley Assamese Filtered
Lakhi Doley Mising Filtered
Mrigank Pegu Assamese Filtered
Mrigank Pegu Mising Filtered
Plabon Assamese Filtered
Plabon Mising Filtered
Rahul Assamese
Rintu Mili Assamesee(1)
Rintu Mili Assamesee
Rintu Mili Mising
Risob Taye Assamese
Risob Taye Missing

Fig 4.5(b) .wav files of male missing stored after filtering

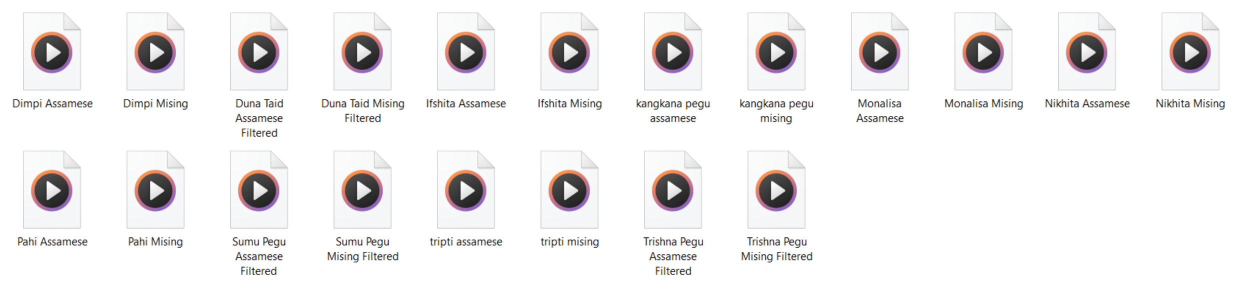


Fig 4.5(c) .wav files of female missing stored after filtering

CHAPTER – 5
PHONETIC & ACCENT ANALYSIS OF ASSAMEESE AND MISING LANGUAGE

(5.1) Introduction

In this chapter, we delve into the quantitative comparison of acoustic features between Assamese and Mising—the two languages at the heart of our project. By summarizing key parameters in a structured tabular format, we aim to reveal the underlying phonetic and prosodic differences that characterize each language. This approach not only provides a visual snapshot of the data but also lays the groundwork for more robust statistical and perceptual analyses.

Our analysis centers on two primary acoustic dimensions:

- **Spectral Features via MFCCs**
  The Mel-Frequency Cepstral Coefficients (MFCCs) capture the spectral envelope of speech and offer a compact representation of the acoustic characteristics that contribute to different accents and linguistic identities. In this chapter, we extract the mean and standard deviation of each MFCC coefficient. Organizing these values in a table allows us to compare the overall spectral patterns between the two language recordings while controlling for speaker variability, since the data are derived from the same speaker.

- **Pitch (F0) Estimation**
  The fundamental frequency (F0) is a critical indicator of intonation and prosodic features. By computing the mean pitch and its variability across voiced segments, we can identify how speakers may subtly modulate pitch when switching between Assamese and Mising. Presenting F0 statistics in tabular form further aids in pinpointing systematic differences that may be perceptually significant.

The chapter's tabular summaries serve multiple purposes. They offer a clear, side-by-side comparison of acoustic parameters that is easy to digest, and they also provide the quantitative backbone necessary for subsequent statistical tests. This approach paves the way for establishing whether the observed differences are statistically significant—a necessary step if our goal is to refine language-specific transcription models or to contribute to broader research in phonetics and speech processing.

Ultimately, by summarizing these indicators in tables, we create a comprehensive framework that not only highlights distinct acoustic signatures of Assamese and Mising but also enriches our understanding of how these languages differ at a fundamental, technical level. This foundation sets the stage for deeper explorations, such as segment-wise feature analysis and advanced visualization techniques, all of which are discussed in the following sections of our report.

(5.2) Analyzing Waveforms & Spectrograms for both Assamese & Mising Languages

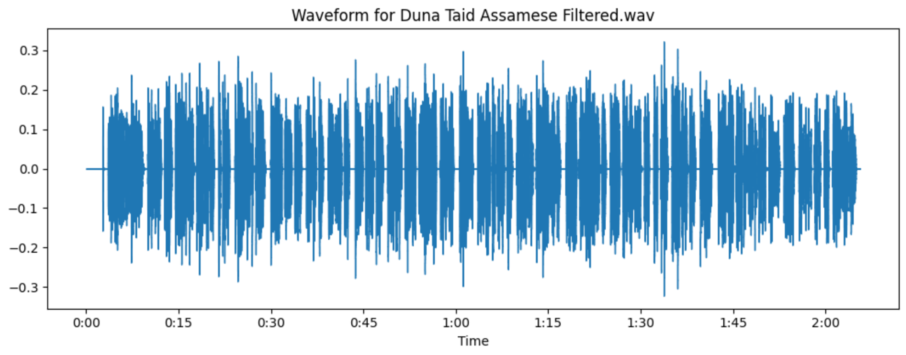


Fig.5.2(a): The figure depicts the waveform of the filtered 'Duna Taid Assamese' audio sample, highlighting temporal amplitude variations over a 2-minute segment used in subsequent transcription analysis.

The waveform is a graphical representation of a raw audio signal displayed in the time domain, where the horizontal axis denotes time and the vertical axis represents the instantaneous amplitude. This amplitude reflects the air pressure variations sampled from sound waves at discrete intervals. As a result of the digital sampling process—governed by the Nyquist-Shannon sampling theorem—the continuous audio signal is converted into a sequence of numerical values, allowing for a faithful recreation of the original sound once these values are processed.

Digital audio processing involves normalizing the waveform so that the amplitude values fit within a standardized range (for example, between –1 and 1 or –0.3 and 0.3). This normalization ensures that differences in recording levels and varying signal strengths across recordings do not obscure the analysis. Each digital sample corresponds to a specific moment in time, and the collection of these samples, when plotted, provides both a macroscopic view of the overall sound dynamics and a microscopic view of transient events, which are critical to identifying speech characteristics.

The visual characteristics of the waveform—its peaks, troughs, and overall envelope—offer insight into the dynamic behavior and energy distribution of the audio signal. Peaks indicate moments of high energy, often corresponding to transient events like plosive consonants or sudden bursts of sound, while troughs represent lower energy or quieter segments. This time-domain analysis not only enables the identification of sustained sounds such as vowels but also aids in understanding the overall loudness and dynamic range, both of which are essential for assessing recording quality and guiding further signal processing steps.

In our project, the waveform serves as the foundational signal from which advanced acoustic features are derived. It underpins the extraction of features like Mel-Frequency Cepstral Coefficients (MFCCs) and pitch (F0) estimates, which are crucial for comparing the phonetic and prosodic differences between Assamese and Mising. By analyzing the waveform, we ensure that subsequent pre-processing—such as segmentation, noise reduction, and statistical analysis—is rooted in a robust, well-represented audio signal. This detailed understanding of the raw audio's characteristics is essential for pinpointing subtle accent variations and refining our custom transcription system for these under-resourced languages.

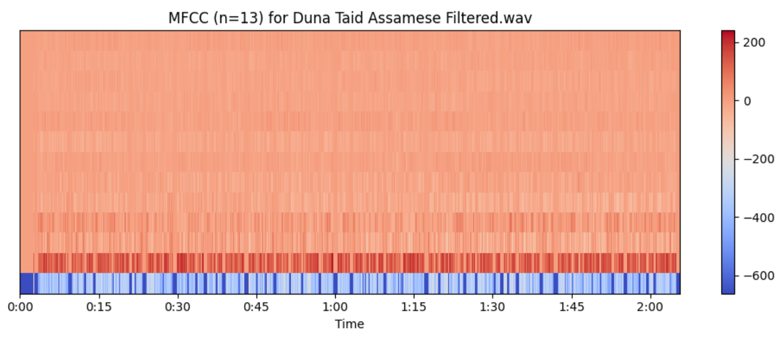


Fig.5.2(b): The figure depicts the MFCC heatmap of the filtered 'Duna Taid Assamese' audio sample

The image is a Mel-Frequency Cepstral Coefficients (MFCC) plot generated for the audio file "Duna Taid Assamese Filtered.wav." In this plot, the horizontal axis represents time—spanning from 0:00 to 2:00—while the vertical axis lists 13 MFCC coefficients, each capturing distinct spectral characteristics of the audio signal. The color mapping, with red indicating higher values and blue indicating lower values, illustrates the intensity of each MFCC over time. This visual representation allows us to understand how the spectral properties evolve throughout the duration of the recording.

MFCCs are derived by first computing the short-time Fourier transform (STFT) of the audio signal and then mapping the energy spectrum onto a nonlinear mel scale, which aligns more closely with human auditory perception. By taking the logarithm of these mel-scaled energies and applying a discrete cosine transform (DCT), the resulting coefficients capture the timbral aspects and the envelope of the speech signal while reducing the influence of acoustical variability. As a result, MFCCs provide a compact and robust representation of the audio's spectral envelope, making them an essential feature in many speech processing and recognition tasks.

The plot's color-coded display conveys important information about the distribution and variation of MFCC values across time. Peaks indicated by warmer colors (red) can signal moments of higher spectral energy in certain frequency bands, potentially corresponding to vocalic sounds or prominent speech articulations. Conversely, cooler colors (blue) represent lower energy regions. This dynamic range in the MFCC plot reflects both the phonetic and prosodic characteristics of the recording—providing insight into how specific speech sounds vary over time, which is particularly useful for accent and timbre analysis.

In our project, this MFCC plot serves as a crucial tool for comparing the acoustic features of different language recordings. By analyzing the detailed patterns in the MFCC plot, we can identify subtle variations in the spectral envelope that differentiate Assamese from Mising accents. Such an analysis not only helps in understanding the perceptual aspects of the speech but also lays the foundation for developing improved transcription systems. The comprehensive depiction of how MFCCs evolve over time further enables us to extract quantitative features for statistical analysis and machine learning applications in accent adaptation and speech recognition.

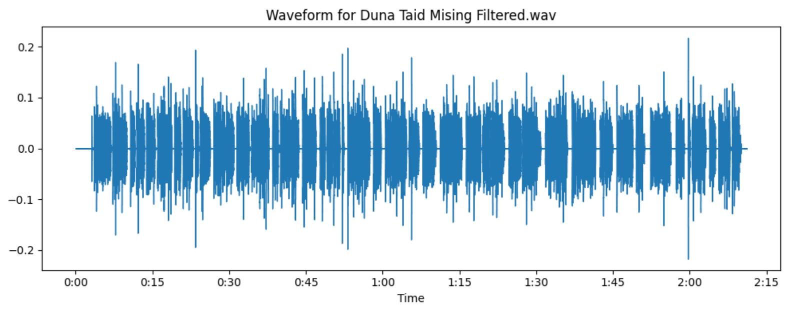


Fig.5.2(c): The figure depicts the waveform of the filtered ‘Duna Taid Assamese’ audio sample, highlighting temporal amplitude variations over a 2-minute segment used in subsequent transcription analysis.

The waveform graph represents the raw audio signal captured in “Duna Taid Mising Filtered.wav.” In this visualization, the horizontal axis marks time—spanning roughly from the beginning to the end of the 2-minute-and-15-second recording—while the vertical axis shows the instantaneous amplitude of the signal, scaled between –0.2 and 0.2. This time-domain representation is the result of digitally sampling the continuous sound wave at a fixed rate, thereby converting sound pressure fluctuations into numerical values that form the waveform.

The amplitude values depicted In the waveform correspond to the loudness and energy present at each moment in the recording. Peaks indicate moments of higher energy—possibly due to transient events or more pronounced speech sounds—while the valleys represent lower energy segments. Despite these fluctuations, the waveform appears dense and consistent, suggesting that the recording maintains a relatively uniform level of sound throughout its duration. This consistency plays an important role in ensuring the reliability of subsequent feature extraction and analysis.

Analyzing the waveform provides insight into the temporal structure of the audio. Sudden changes or spikes may indicate key phonetic events or transient sounds, whereas the overall pattern helps in identifying the speech rhythm and prosodic cues inherent to the speaker's style. Such time-domain analysis is essential as a first step, serving as the foundation upon which more complex features—like spectrograms, MFCCs, and pitch contours—are extracted. The clear visual cues from the waveform guide further processing, including segmentation and noise reduction, that are crucial for robust speech analysis.

In our project, this waveform is a critical element that underpins the acoustic analysis of the Mising recording. By carefully studying the amplitude variations and overall structure of the waveform, we can identify the dynamic characteristics of the signal. These observations enable us to extract advanced features necessary for comparing the phonetic and prosodic traits between languages. Ultimately, such analysis not only supports the development of a refined transcription system but also deepens our understanding of the subtle accent and speech patterns present in under-resourced languages like Mising.

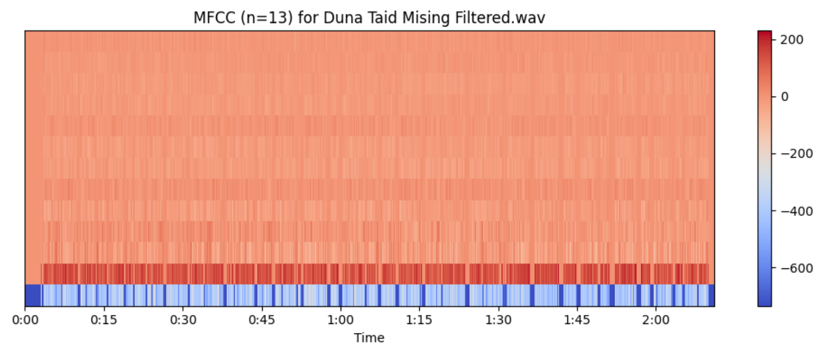


Fig.5.2(d): The figure depicts the MFCC heatmap of the filtered 'Duna Taid Mising' audio sample

(5.3) Pitch Contour

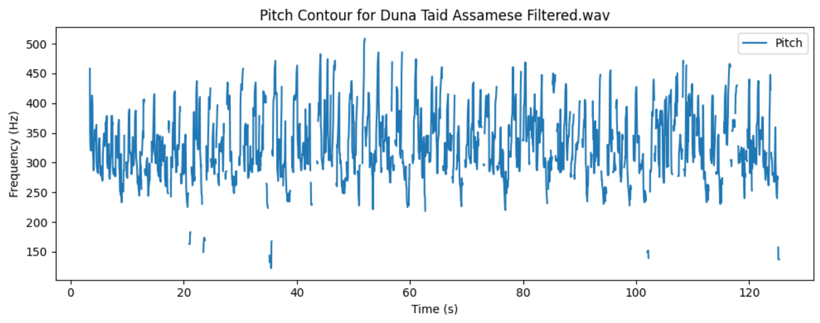


Fig.5.3(a): Figure Shows how fundamental frequency varies w.r.t. time for Assamese Audio

The pitch contour plot provides a visual representation of how the fundamental frequency (F0) of the speech signal changes over time in the “Duna Taid Assamese Filtered.wav” file. On the horizontal axis, time is measured in seconds (ranging from 0 to 130 seconds), while the vertical axis displays frequency in Hertz (Hz), spanning from 0 up to 500 Hz. The blue line traced across the graph indicates the instantaneous pitch values throughout the recording.

Fundamentally, pitch is closely tied to the speaker's prosody and vocal effort, serving as a key indicator of intonation and stress. The way the pitch fluctuates—rising and falling—can reveal patterns related to sentence inflection, emotional tone, and the placement of emphasis on certain syllables or words. In this plot, the pitch values predominantly fluctuate between approximately 150 Hz and 450 Hz, suggesting variations in vocal energy and modulations that occur naturally in connected speech.
Analyzing this pitch contour can uncover important aspects of speech characteristics, such as the dynamic shifts that occur during phrase formation, pauses, or emphasis on specific sounds. Such insights are particularly useful when comparing accentual or linguistic variations, as they help identify if and how speakers modulate their voice differently in various languages or dialects. This information is crucial for further acoustic analysis and the development of robust speech transcription models.

Overall, the pitch contour plot is a valuable tool for interpreting the underlying prosodic features in the speech signal. By offering a detailed, time-resolved view of pitch variations, it lays the groundwork for more advanced analyses—such as correlating these patterns with phonetic or accent-specific traits—which can ultimately contribute to improvements in both automatic speech recognition systems and linguistic research.

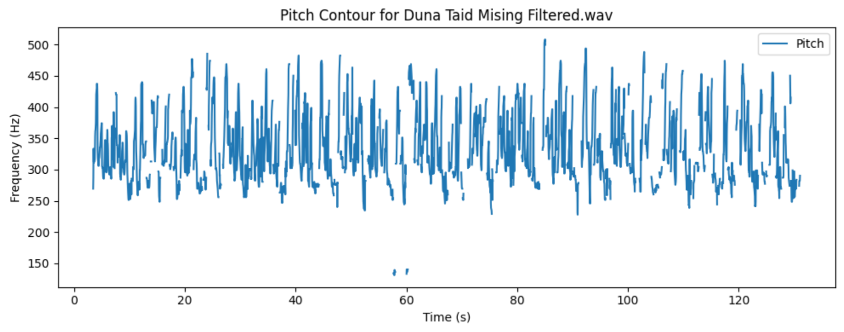


Fig.5.3(b): Figure Shows how Fundamental Frequency Changes w.r.t time for Mising Audio

The pitch contour plot displays the fundamental frequency (F0) of the audio signal over time. On the horizontal axis, time is shown in seconds (approximately 0–130 seconds), while the vertical axis indicates frequency in Hertz, ranging from 0 up to 500 Hz. The blue line represents the instantaneous pitch values throughout the recording, giving a time-resolved view of the speaker's vocal output.

Pitch is a key indicator of a speaker's prosody and intonation. The rising and falling patterns of the pitch contour reveal how the voice changes, helping us understand which parts of the speech carry more emphasis or emotional nuance. These fluctuations in frequency are essential for discerning the speaker's intonation patterns and rhythmic structure.

By examining the variations in the pitch contour, we can identify transient events, sustained regions, and abrupt shifts. Such patterns might correspond to stressed syllables, pauses, or changes in speaking style. This detailed view helps in pinpointing expressive elements in the speech and contributes to the analysis of accent and prosodic differences.

In our project, this pitch contour analysis is fundamental for comparing the vocal characteristics of different language recordings. It informs our transcription systems by highlighting subtle prosodic features and accent variations, ultimately helping to refine speech recognition models tailored for under-resourced languages like Mising.

(5.4) Plotting Formant Trajectories

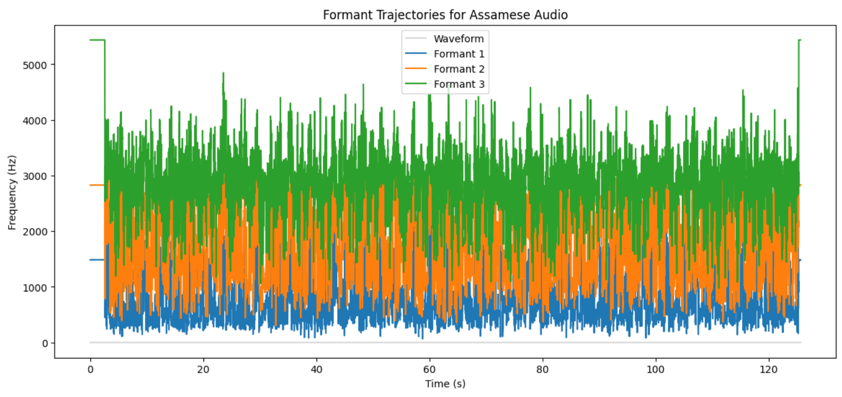


Fig.5.4(a): Figure Shows Variations of Different Formants w.r.t Time for Assamese Audio

The graph titled "Formant Trajectories for Assamese Audio" offers a visual representation of the dynamic resonant frequencies (formants) produced by the human vocal tract during speech. In this visualization, time is displayed on the horizontal axis, while frequency in Hertz is shown on the vertical axis. The plot includes not only the underlying waveform (depicted in gray) but also overlays the trajectories for the first three formants—Formant 1 in blue, Formant 2 in orange, and Formant 3 in green. Such a plot is fundamental in acoustics and phonetics as it captures the shifting pitch and resonance characteristics that underpin spoken language.

At the core of this analysis are the formants, which represent the prominent frequency bands that arise from the resonant properties of the vocal tract. Formant 1 (F1) generally relates to vowel height, indicating how open or closed the mouth is during articulation, while Formant 2 (F2) correlates with vowel backness, reflecting the positioning of the tongue. Formant 3 (F3) provides finer details regarding specific articulatory features. By observing how these formants vary over time, as clearly depicted in the plot, researchers can infer subtle differences in vowel quality and articulation that are characteristic of the Assamese language.

The temporal variation of the formant trajectories, shown over approximately 130 seconds, highlights the rich tapestry of speech sounds produced during natural conversation. As the speaker transitions between different phonetic segments, the resonant frequencies shift; for example, vowels display relatively stable formant patterns, whereas consonants often induce rapid transitions. The overlay of the waveform itself adds context by superimposing the raw amplitude variations, allowing us to correlate acoustic energy with changes in resonance. This close coupling between time and spectral information is central to understanding how speakers modulate their vocal tract to produce distinct sounds.

In our project, the detailed analysis of formant trajectories is crucial for comparing the acoustic characteristics of Assamese with those of other languages, such as Mising. By analyzing these plots, we can identify systematic differences in articulation, which can be pivotal for speech recognition, accent adaptation, and linguistic studies. Ultimately, this formant analysis not only enriches our understanding of the phonetic structures within under-resourced languages but also informs the refinement of transcription models tailored to capture these unique acoustic patterns.

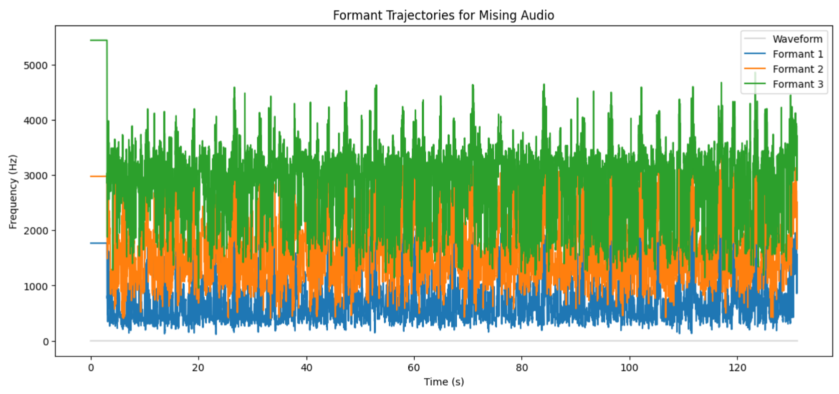


Fig.5.4(b) : Figure Shows Variations of Different Formants w.r.t Time for Mising Audio

The graph titled "Formant Trajectories for Mising Audio" provides a visual representation of the resonant frequencies produced by the vocal tract over time. The horizontal axis, measured in seconds (from 0 to 130 seconds), captures the temporal evolution of the speech, while the vertical axis, spanning from 0 to 5500 Hz, reflects the frequency range of interest. In this plot, the gray line depicts the raw waveform, offering context about the overall energy and timing of the recording, whereas the overlaid colored trajectories represent the first three formants that are essential for understanding vowel quality and articulation.

Formants are the spectral peaks resulting from the resonant characteristics of the vocal tract. In this plot, Formant 1 is shown in blue, typically indicative of vowel height and the openness of the mouth; Formant 2, represented by the orange line, often corresponds to vowel backness and the tongue's positioning; and Formant 3, displayed in green, captures more subtle articulatory features. By visualizing these trajectories, one can observe how the spectral characteristics shift and evolve throughout the audio, revealing differences in articulatory configurations during connected speech.

The plot's design, where the waveform and formant trajectories are superimposed, enables a comprehensive view of the speech signal. The waveform provides insight into the energy distribution over time, while the clear, color-coded formant lines show the variation in resonant frequencies during phonetic transitions. Sudden changes in the formant trajectories can be linked to rapid articulatory movements, such as those occurring at the boundaries of vowels and consonants, while more stable regions indicate sustained speech sounds. This dual representation aids in correlating temporal energy variations with spectral characteristics, enhancing our understanding of the speaker's articulatory dynamics.

In our project, analyzing formant trajectories from Mising audio is crucial for identifying subtle differences in speech production between languages. By comparing these trajectories with those from other language recordings, we can pinpoint systematic variations in articulation that may underlie accent differences. This information not only supports the development of more accurate transcription and language-specific models but also deepens our phonetic insights into under-resourced languages like Mising. The ability to visualize and quantify these differences is foundational to refining our acoustic analysis and ultimately improving the performance of our speech recognition system.

(5.5) Rhythmic Analysis for both Assamese & Mising Languages

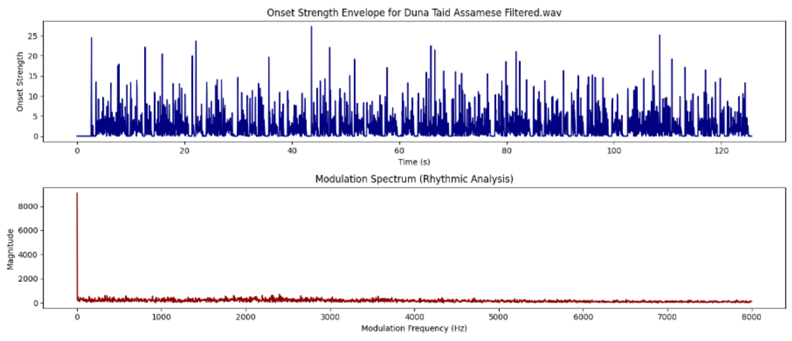


Fig.5.5(a): Figure Depicts Rhythhmic Analysis of Assamese Audio Consisting of Onset Strength & Magnitude

The top plot, titled “Onset Strength Envelope for Duna Taid Assamese Filtered.wav,” presents a time-domain analysis that captures moment-to-moment changes in the audio's energy. The horizontal axis spans approximately 0 to 130 seconds, while the vertical axis quantifies the onset strength, varying from 0 to about 25. This envelope is computed by analyzing sudden increases in acoustic energy, which are typically indicative of the beginning of new phonetic events or percussive sounds. In this plot, the blue line with its characteristic peaks and valleys provides a visual map of how the signal’s onsets occur over time, offering valuable insights into the temporal structure of the speech.

The onset strength envelope is crucial for detecting acoustic events such as syllable boundaries or changes in speech dynamics. Peaks in this envelope correspond to moments of significant acoustic change, often associated with transient speech sounds or emphasized articulations. These features help in segmenting the audio into meaningful units, which is particularly useful in both speech processing and acoustic phonetic studies. By delineating individual events within the continuous stream of speech, the envelope serves as the foundation for further rhythmic and prosodic analysis.

Below the onset envelope, the second plot—titled “Modulation Spectrum (Rhythmic Analysis)”—transforms the time-domain information into the frequency domain to reveal rhythmic patterns. The x-axis represents modulation frequency in Hertz (ranging from 0 to 8000 Hz), while the y-axis shows the magnitude of these modulations, also reaching up to around 8000. The dominant feature of this red line plot is a prominent peak at lower modulation frequencies, which indicates a strong rhythmic component and periodicity within the speech. The attenuation of higher modulation frequencies suggests that most of the rhythmic content is confined to the slower, more perceptually significant modulations.

Together, these plots contribute significantly to our project by elucidating the temporal dynamics of the audio signal. The onset strength envelope helps segment the speech and capture transient events by highlighting rapid changes in energy, while the modulation spectrum provides a broader view of the rhythmic structure inherent in the recording. This dual analysis not only aids in accent and prosody studies by characterizing the timing and periodic elements of the speech but also informs the design and refinement of transcription systems tailored for under-resourced languages like Assamese.

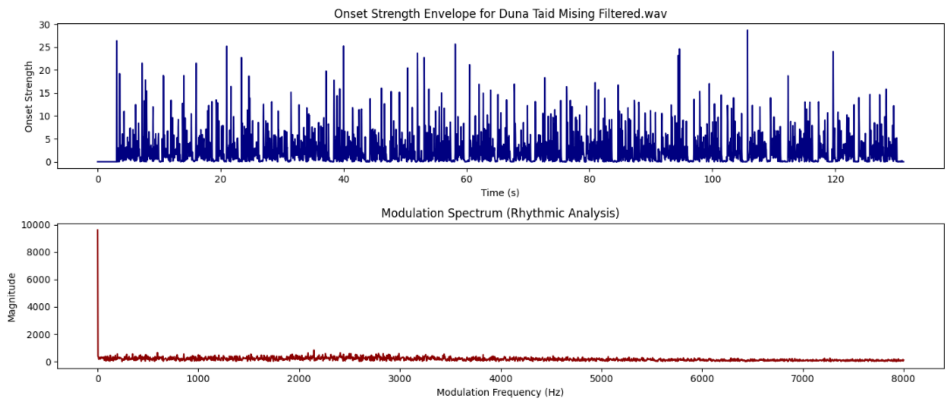


Fig.5.5(b): Figure Depicts Rhythhmic Analysis of Mising Audio Consisting of Onset Strength & Magnitude

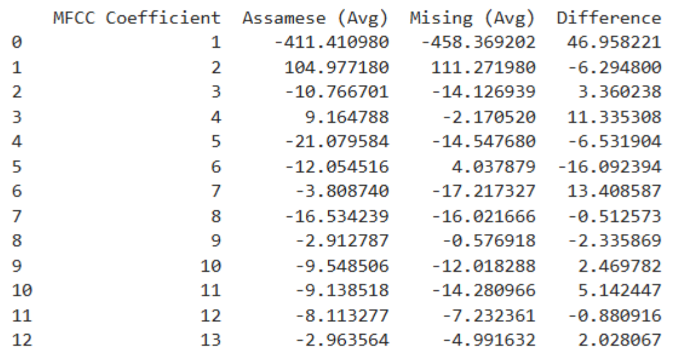
The upper graph, titled "Onset Strength Envelope for Duna Taid Mising Filtered.wav," visually represents the temporal evolution of acoustic events in the recording. The horizontal axis spans approximately 0 to 130 seconds, while the vertical axis quantifies the onset strength, ranging from 0 to around 30. The envelope, depicted by the series of peaks and troughs, reflects the moment-to-moment changes in the energy of the audio signal, indicating where new phonetic events—such as syllable onsets or percussive bursts—occur.

This onset strength envelope is crucial for understanding the temporal dynamics of the speech, as it highlights points of significant acoustic change throughout the recording. Each peak in the envelope is suggestive of a sudden increase in energy, which could correspond to critical transitions in speech sounds or expressive markers like emphasis. The variability and timing of these peaks provide insight into how the speech is segmented naturally, forming the basis for further rhythmic and prosodic analysis in our project.

The lower graph, titled "Modulation Spectrum (Rhythmic Analysis)," shifts the focus from the time domain to the frequency domain by examining the rhythmic structure of the speech. The x-axis represents modulation frequency in Hertz, ranging from 0 to 8000 Hz, while the y-axis shows the magnitude, extending up to approximately 10,000. A prominent peak at lower modulation frequencies indicates that the speech has a strong rhythmic component, as lower frequencies in the modulation spectrum generally capture the periodic and rhythmic elements of the audio signal.
Together, these graphs offer a comprehensive view of the speech's temporal and rhythmic characteristics. By analyzing the onset strength envelope alongside the modulation spectrum, we can better understand the timing and periodicity inherent in the Mising recording. This dual analysis not only aids in detecting and segmenting critical speech events but also provides valuable insights into the rhythmic patterns that

distinguish the accent. Such information is pivotal for refining our transcription system and advancing our acoustic analysis of under-resourced languages like Mising.

(5.6) MFCC Comparison Between Assamese & Mising Audio



```
    MFCC Coefficient  Assamese (Avg)  Mising (Avg)  Difference
0                  1     -411.410980   -458.369202   46.958221
1                  2      104.977180    111.271980   -6.294800
2                  3      -10.766701    -14.126939    3.360238
3                  4        9.164788     -2.170520   11.335308
4                  5      -21.079584    -14.547680   -6.531904
5                  6      -12.054516      4.037879  -16.092394
6                  7       -3.808740    -17.217327   13.408587
7                  8      -16.534239    -16.021666   -0.512573
8                  9       -2.912787     -0.576918   -2.335869
9                 10       -9.548506    -12.018288    2.469782
10                11       -9.138518    -14.280966    5.142447
11                12       -8.113277     -7.232361   -0.880916
12                13       -2.963564     -4.991632    2.028067
```

Fig.5.6(a): Table Shows the Average MFCCs of Assamese and Mising Audio Along with their Difference

The table presents a side-by-side numeric comparison of acoustic feature values extracted from two conditions—likely corresponding to two language recordings or two different processing scenarios. Each row records a specific feature or coefficient (indexed from 1 to 13) along with two measured values and their computed difference. For example, the first row shows values of –411.41 and –458.37 for a given feature, with a difference of 46.96, indicating that one condition produced a higher value by that amount. This layout provides a direct, quantitative snapshot of how these features compare between the two conditions.

The first two columns (or indices) in each row help in identifying or ordering the features, while the subsequent numerical columns represent the mean or aggregate values obtained under each condition. The final column, representing the difference, is calculated by subtracting one value from the other—for example, subtracting –458.37 from –411.41, resulting in a positive difference. This systematic representation allows us to immediately see which features differ and by how much, with both positive and negative values indicating the direction of the difference.

This detailed numerical breakdown is crucial for further analysis. The computed differences serve as objective metrics to evaluate whether the acoustic features in one scenario are consistently higher or lower compared to the other. When paired with statistical tests—such as paired t-tests—these numbers can help determine if the observed differences are statistically significant. In turn, such validation is vital for ensuring that any observed variation in the speech signal is not due to random fluctuations but rather reflects a true, underlying difference potentially linked to accent or phonetic characteristics.

In our project, this table informs the refinement and calibration of our speech analysis and transcription systems. By clearly delineating the differences in acoustic parameters between conditions (such as between Assamese and Mising recordings), we gain insights into the subtle phonetic nuances inherent to each language. Ultimately, these detailed comparisons contribute to developing more robust models for accent adaptation and speech recognition, thereby enhancing the system's ability to accurately capture the characteristics of under-resourced languages.

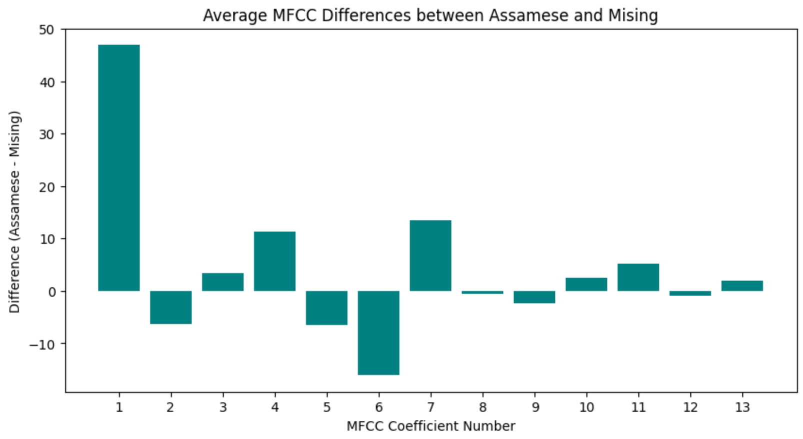


Fig.5.6(b): Figure Depicts Visually the Average Differences of MFCCs for Assamese and Mising

(5.7) Rhythm Formant Analysis (RFA) on Assamese and Mising Audio Dataset

(5.7.1) Introduction

In this chapter, we describe the application of Daffydd Gibbon's Rhythm Formant Analysis (RFA) code to a voice dataset. The primary objective was to extract and visualize rhythmic as well as spectral characteristics of voices in a collection of filtered audio files. Our goal was to integrate

RFA into our custom transcription pipeline for low-resource languages, thereby enabling further linguistic and phonetic analysis. This work involved adapting and troubleshooting the original code, restructuring the data directories, and automating the processing of multiple audio files.

(5.7.2) Background

Rhythm Formant Analysis is a method used to quantify the rhythmic properties of speech by analyzing formant trajectories and related spectral features. Daffydd Gibbon’s implementation of RFA provided a modular approach that could be applied to individual voice recordings to produce informative spectrograms and heatmaps. The insights gained from these visualizations play an important role in understanding speech rhythm, which in turn has applications in areas such as language documentation and automatic speech recognition (ASR)—especially for languages with limited digital resources.

(5.7.3) Dataset and Preprocessing

The voice dataset comprised approximately 34 audio files. Among these were recordings such as:

- Biplo Kuli Assamese Filtered.wav
- Biplo Kuli Mising Filtered.wav

These recordings were preprocessed through a pipeline that included:

Resampling and normalization: Ensuring that all audio files were in a consistent format (e.g., mono-channel, fixed sampling rate).
Filtering: Reducing background noise and accentuating the rhythmic components relevant to our analysis.

(5.7.4) Methodology

Adapting the Single-File Processing Code
We began by running the single-file processing script (rfa_single.py) on individual files from the dataset
Implementing Batch Processing
After successfully processing individual files, we advanced to automating the analysis for all audio.We also explored using the built-in multiple processing module (RFA_multiple_signal_processing), which accepts a folder path as an argument. By passing "../RFA Dataset" to the main script of this module (e.g., rfa_multiple.py), the code could automatically detect and process all .wav files—greatly simplifying the overall workflow.

Chapter – 6
TRAINING MACHINE LEARNING MODELS FOR SPEAKER RECOGNITION

(6.1) Introduction

Following our successful database collection of speaker audio samples, this chapter outlines our complete end-to-end pipeline for building a robust speaker recognition system targeted at low-resource languages. Our approach spans from the initial feature extraction from raw audio to the development of an optimized ensemble classifier using stacking.

Libraries Used

A wide range of Python libraries played crucial roles throughout the development of our speaker recognition system. These libraries facilitated everything from numerical operations and data visualization to model building, cross-validation, and saving the final deployed model. Below is a comprehensive list of these libraries along with their purposes:

- **NumPy**
- Provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions used to operate on these arrays
- We used NumPy extensively for handling and processing numerical data during feature extraction, such as calculating and manipulating MFCC values and their augmentations.

- **Matplotlib**
- Enables the creation of a wide range of static, animated, and interactive visualizations. In this project, it was used to generate plots such as accuracy comparisons, ROC curves, and training histories.
- Matplotlib was used to generate key plots, including accuracy comparison charts, ROC curves, and training history graphs. These visualizations were essential for monitoring the performance of our models throughout the development process.

- **Seaborn**
- Built on top of Matplotlib, Seaborn offers a high-level interface for drawing attractive statistical graphics. It was particularly useful for generating heatmaps (like the confusion matrix plot).
- Seaborn was utilized to create clear and informative heatmaps for our confusion matrix, ensuring that the model's misclassifications were easily interpretable.

- **Scikit-Learn**
- Provides a wide range of tools for data mining, data analysis, machine learning algorithms, model evaluation, and data preprocessing.
- Scikit-Learn was the backbone of our machine learning workflow. It enabled us to implement algorithms like SVM, Random Forest, KNN, and Logistic Regression, perform hyperparameter tuning with GridSearchCV, and evaluate models using metrics like accuracy, ROC curves, and confusion matrices. The stacking classifier ensemble was also built using Scikit-Learn.

- **Joblib**
- Facilitates efficient saving and loading of Python objects to disk
- We used Joblib to serialize the final stacking classifier. This allowed us to save the trained model for future use without the need for retraining, ensuring reproducibility and ease of deployment.

- **Pandas**
- Offers powerful data manipulation and analysis tools, particularly through its DataFrame structure.
- Pandas was used to organize and manage structured data, especially when creating data frames to compare model performance metrics and generate plots that summarize the evaluation results.

- **Librosa**
- Librosa is a dedicated Python library for audio processing and feature extraction.
- We used Librosa to load audio files at a standardized sampling rate and extract MFCC features, which effectively capture the spectral characteristics of each speaker's voice.

(6.2) Encoding of Voice Samples

In our system, we encode our voice samples into two distinct numerical categories—0 and 1—which represent the two languages of interest. This numeric encoding is essential for enabling machine learning algorithms to process categorical data. In our approach, each voice file inherently contains an indicator of its language within its filename. We leverage this characteristic to map the linguistic identity directly to a binary format: "0" for one language (Assamese) and "1" for the other (Mising).

This encoding mechanism forms a fundamental part of our preprocessing pipeline. By transforming a categorical attribute (the language type) into a numeric label, we provide a consistent and efficient input for subsequent classification models. The transformation exemplifies a basic yet critical step in many machine learning workflows—converting qualitative speech identifiers into quantitative values that can be readily consumed by algorithms.

(6.3) Feature Extraction Using MFCCs (Mel Frequency Cepstral Coefficients)

MFCC Mel Frequency Cepstral Coefficients (MFCC) are a set of features that represent the short-term power spectrum of a sound. They are computed by mapping the frequency scale to the Mel scale, which more closely approximates human auditory perception. MFCCs effectively capture the timbral texture and other nuanced characteristics of an audio signal. This makes them particularly useful in tasks such as speaker recognition, where subtle differences in vocal attributes are critical.

In our speaker recognition system, converting raw audio data into a format suitable for machine learning is an essential step. The code snippet below describes the process of extracting numerical features from audio files using Mel Frequency Cepstral Coefficients (MFCCs). MFCCs are widely recognized for their ability to capture the perceptually relevant aspects of the audio spectrum, which makes them particularly effective in characterizing speaker-specific traits.

The process begins with the definition of a function—extract_mfcc_features—that encapsulates our feature extraction methodology. This function uses the Librosa library to load an audio file at a prescribed sampling rate (16 kHz in our case), ensuring consistency across all recordings. It then computes 13 MFCCs, which summarize the short-term spectral envelope of the audio signal. To obtain a fixed-length representation for each file, we take the mean of the MFCC values across all time frames. This averaging process condenses the temporal information into a single, robust feature vector per audio sample.

In parallel, we iterate through all valid audio files and their associated labels. For each file, we extract its MFCC features and, on successful computation, append them to a feature matrix alongside the corresponding label. This step is designed with error handling to skip any files that might generate processing errors—thus maintaining the overall integrity of our dataset.

The end result is a well-structured feature matrix, where each row represents a speaker's audio sample encoded as a vector of MFCC-derived features, and a label array indicating the corresponding speaker category. These arrays serve as the foundation for training our machine learning models for speaker recognition.

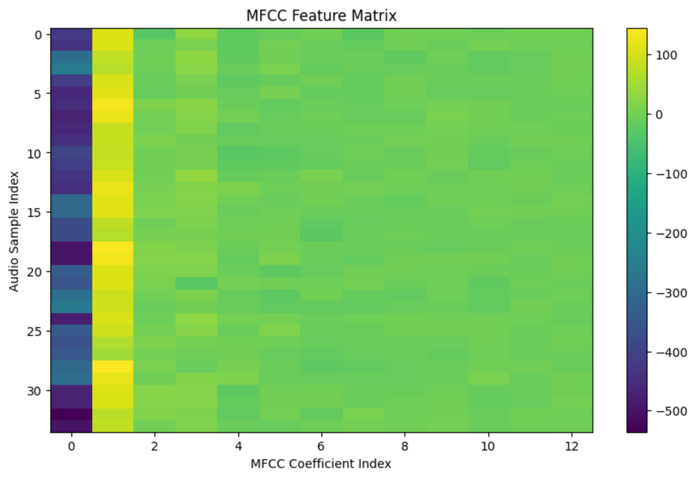
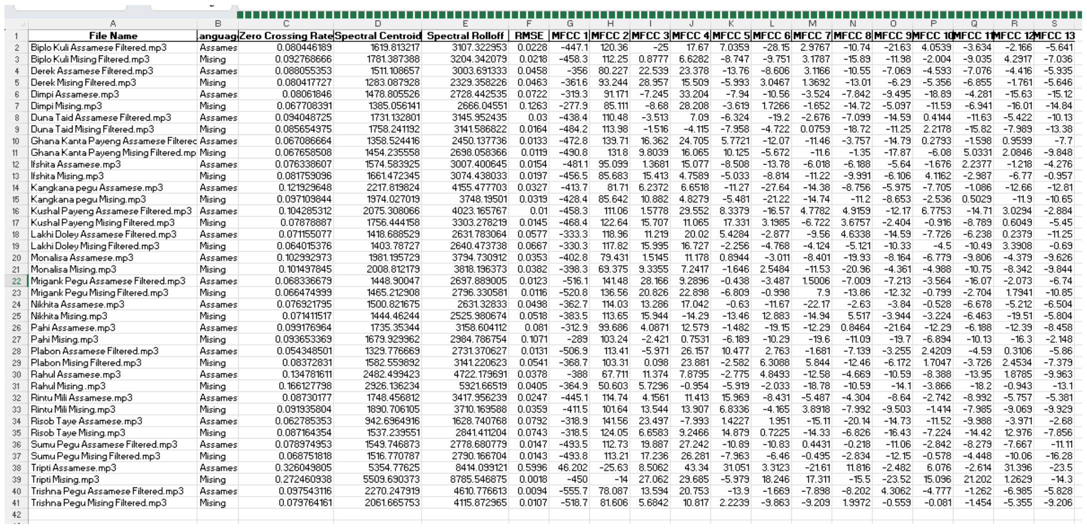


Fig.6.3(a): Figure Depicting Visual Represenation of MFCC Features in Matrix Form



	File Name	anguag	Zero Crossing Rate	Spectral Centroid	Spectral Rolloff	RMSE	MFCC 1	MFCC 2	MFCC 3	MFCC 4	MFCC 5	MFCC 6	MFCC 7	MFCC 8	MFCC 9	MFCC 10	MFCC 11	MFCC 12	MFCC 13
2	Biplo Kuli Assamese Filtered.mp3	Assames	0.080446189	1619.813217	3107.322953	0.0228	-447.1	120.36	-25	17.67	7.0359	-28.15	2.9767	-10.74	-21.63	4.0539	-3.634	-2.166	-5.641
3	Biplo Kuli Mising Filtered.mp3	Mising	0.092768666	1781.387388	3204.342079	0.0218	-458.3	112.25	0.8777	6.6282	-8.747	-9.751	3.1787	-15.89	-11.98	-2.004	-9.035	4.2917	-7.036
4	Derek Assamese Filtered.mp3	Assames	0.088055353	1511.108657	3003.691333	0.0458	-356	80.227	22.539	23.378	-13.76	-8.606	3.1166	-10.55	-7.069	-4.593	-7.076	-4.416	-5.935
5	Derek Mising Filtered.mp3	Mising	0.080417727	1283.087928	2329.358226	0.0463	-361.6	93.244	28.957	15.509	-5.993	3.0467	1.3692	-13.01	-6.29	-5.356	-6.855	-1.761	-5.646
6	Dimpi Assamese.mp3	Assames	0.08061846	1478.805526	2728.442535	0.0722	-319.3	91.171	-7.245	33.204	-7.94	-10.56	-3.524	-7.842	-9.495	-18.89	-4.281	-15.63	-15.12
7	Dimpi Mising.mp3	Mising	0.067708391	1385.056141	2666.04551	0.1263	-277.9	85.111	-8.68	28.208	-3.619	1.7266	-1.652	-14.72	-5.097	-11.59	-6.941	-16.01	-14.84
8	Duna Taid Assamese Filtered.mp3	Assames	0.094048725	1731.132801	3145.952435	0.03	-438.4	110.48	-3.513	7.09	-6.324	-19.2	-2.676	-7.099	-14.59	0.4144	-11.63	-5.422	-10.13
9	Duna Taid Mising Filtered.mp3	Mising	0.085654975	1758.241192	3141.586822	0.0164	-484.2	113.98	-1.516	-4.115	-7.958	-4.722	0.0759	-18.72	-11.25	2.2178	-15.82	-7.989	-13.38
10	Ghana Kanta Payeng Assamese Filterec	Assames	0.067086664	1358.524416	2450.137736	0.0133	-472.8	139.71	16.362	24.705	5.7721	-12.07	-11.46	-3.757	-14.79	0.2793	-1.598	0.9599	-7.7
11	Ghana Kanta Payeng Mising Filtered.mp	Mising	0.067658508	1454.235558	2698.058366	0.0119	-490.8	131.8	9.8039	16.065	10.125	-5.672	-11.6	-1.35	-17.87	-6.08	5.0331	2.0846	-9.848
12	Ifshita Assamese.mp3	Assames	0.076338607	1574.583925	3007.400645	0.0154	-481.1	95.099	1.3681	15.077	-8.508	-13.78	-6.018	-6.188	-5.64	-1.676	2.2377	-1.218	-4.276
13	Ifshita Mising.mp3	Mising	0.081759096	1661.472345	3074.438033	0.0197	-456.5	85.683	15.413	4.7589	-5.033	-8.814	-11.22	-9.991	-6.106	4.1162	-2.987	-6.77	-0.957
14	Kangkana pegu Assamese.mp3	Assames	0.121929648	2217.819824	4155.477703	0.0327	-413.7	81.71	6.2372	6.6518	-11.27	-27.64	-14.38	-8.756	-5.975	-7.705	-1.086	-12.66	-12.81
15	Kangkana pegu Mising.mp3	Mising	0.097109844	1974.027019	3748.19501	0.0319	-428.4	85.642	10.882	4.8279	-5.481	-21.22	-14.74	-11.2	-8.653	-2.536	0.5029	-11.9	-10.65
16	Kushal Payeng Assamese Filtered.mp3	Assames	0.104285312	2075.308066	4023.165767	0.01	-458.3	111.06	1.5778	29.552	8.3379	-16.57	4.7782	4.9159	-12.17	6.7753	-14.71	3.0294	-2.884
17	Kushal Payeng Mising Filtered.mp3	Mising	0.07878887	1756.444158	3303.278219	0.0145	-468.4	122.64	15.707	11.065	17.331	3.1985	-6.722	3.6757	-2.404	-0.916	-8.789	0.6049	-5.45
18	Lakhi Doley Assamese Filtered.mp3	Assames	0.071155077	1418.688529	2631.783064	0.0577	-333.3	118.96	11.219	20.02	5.4284	-2.877	-9.56	4.6338	-14.59	-7.726	-6.238	0.2379	-11.25
19	Lakhi Doley Mising Filtered.mp3	Mising	0.064015376	1403.78727	2640.473738	0.0667	-330.3	117.82	15.995	16.727	-2.256	-4.768	-4.124	-5.121	-10.33	-4.5	-10.49	3.3908	-0.69
20	Monalisa Assamese.mp3	Assames	0.102992973	1981.195729	3794.730912	0.0353	-402.8	79.431	1.5145	11.178	0.8944	-3.011	-8.401	-19.93	-8.164	-6.779	-9.806	-4.379	-9.626
21	Monalisa Mising.mp3	Mising	0.101497845	2008.812179	3818.196373	0.0382	-398.3	69.375	9.3355	7.2417	-1.646	2.5484	-11.53	-20.96	-4.361	-4.988	-10.75	-8.342	-9.844
22	Mrigank Pegu Assamese Filtered.mp3	Assames	0.068336679	1448.90047	2697.889005	0.0123	-516.1	141.48	28.166	9.2896	-0.438	-3.487	1.5006	-7.009	-7.213	-3.564	-16.07	-2.073	-6.74
23	Mrigank Pegu Mising Filtered.mp3	Mising	0.066474999	1465.212908	2796.330581	0.0116	-520.8	136.56	20.826	22.898	-6.809	-0.998	7.9	-13.86	-12.32	-0.799	-2.704	1.7941	-10.85
24	Nikhita Assamese.mp3	Assames	0.076921795	1500.821675	2631.32833	0.0498	-362.7	114.03	13.286	17.042	-0.63	-11.67	-22.17	-2.63	-3.84	-0.528	-6.678	-5.212	-6.504
25	Nikhita Mising.mp3	Mising	0.071411517	1444.46244	2525.980674	0.0518	-383.5	113.65	15.944	-14.29	-13.46	12.883	-14.94	5.517	-3.944	-3.224	-6.463	-19.51	-5.804
26	Pahi Assamese.mp3	Assames	0.099176964	1735.35344	3158.604112	0.081	-312.9	99.686	4.0871	12.579	-1.482	-19.15	-12.29	0.8464	-21.64	-12.29	-6.188	-12.39	-8.458
27	Pahi Mising.mp3	Mising	0.093653369	1679.929962	2984.786754	0.1071	-289	103.24	-2.421	0.7531	-6.189	-10.29	-19.6	-11.09	-19.7	-6.894	-10.13	-16.3	-2.148
28	Plabon Assamese Filtered.mp3	Assames	0.054348501	1329.776669	2731.370627	0.0131	-506.9	113.41	-5.971	26.157	10.477	2.763	-1.681	-7.139	-3.255	2.4209	-4.59	0.3106	-5.86
29	Plabon Mising Filtered.mp3	Mising	0.08372831	1582.559892	3141.220623	0.0541	-368.7	103.31	0.098	23.881	-2.582	6.3088	5.844	-12.46	-6.172	1.7047	-3.726	2.4534	-7.379
30	Rahul Assamese.mp3	Assames	0.134781611	2482.499423	4722.179691	0.0378	-388	67.711	11.374	7.8795	-2.775	4.8493	-12.58	-4.669	-10.59	-8.388	-13.95	1.8785	-9.963
31	Rahul Mising .mp3	Mising	0.166127798	2926.136234	5921.66519	0.0405	-364.9	50.603	5.7296	-0.954	-5.919	-2.033	-18.78	-10.59	-14.1	-3.866	-18.2	-0.943	-13.1
32	Rintu Mili Assamese.mp3	Assames	0.08730177	1748.456812	3417.956239	0.0247	-445.1	114.74	4.1561	11.413	15.969	-8.431	-5.487	-4.304	-8.64	-2.742	-8.992	-5.757	-5.381
33	Rintu Mili Mising.mp3	Mising	0.091935804	1890.706105	3710.169588	0.0359	-411.5	101.64	13.544	13.907	6.8336	-4.165	3.8918	-7.992	-9.503	-1.414	-7.985	-9.069	-9.929
34	Risob Taye Assamese.mp3	Assames	0.062785353	942.6964916	1628.740768	0.0792	-318.9	141.56	23.497	-7.993	1.4227	1.951	-15.11	-20.14	-14.73	-11.52	-9.988	-3.971	-2.68
35	Risob Taye Mising.mp3	Mising	0.087164354	1537.239551	2841.411204	0.0743	-318.5	124.05	6.6583	9.2466	14.879	0.7225	-14.33	-6.826	-16.43	-7.224	-14.42	12.976	-7.856
36	Sumu Pegu Assamese Filtered.mp3	Assames	0.078974953	1549.746873	2778.680779	0.0147	-493.5	112.73	19.887	27.242	-10.89	-10.83	0.4431	-0.218	-11.06	-2.842	-8.279	-7.667	-11.11
37	Sumu Pegu Mising Filtered.mp3	Mising	0.068751818	1516.770787	2790.166704	0.0143	-493.8	113.21	17.236	26.281	-7.963	-6.46	-0.495	-2.834	-12.15	-0.578	-4.448	-10.06	-16.28
38	Tripti Assamese.mp3	Assames	0.326049805	5354.77625	8414.099121	0.5996	46.202	-25.63	8.5062	43.34	31.051	3.3123	-21.61	11.816	-2.482	6.076	-2.614	31.396	-23.5
39	Tripti Mising.mp3	Mising	0.272460938	5509.690373	8785.546875	0.0018	-450	-14	27.062	29.685	-5.979	18.246	17.311	-15.5	-23.52	15.096	21.202	1.2629	-14.3
40	Trishna Pegu Assamese Filtered.mp3	Assames	0.097543116	2270.247919	4610.776613	0.0094	-555.7	78.087	13.594	20.753	-13.9	-1.669	-7.898	-8.202	4.3062	-4.777	-1.262	-6.985	-5.828
41	Trishna Pegu Mising Filtered.mp3	Mising	0.079764161	2061.665753	4115.872965	0.0107	-518.7	81.606	5.6842	10.817	2.2239	-9.863	-9.209	1.9972	-0.559	-0.081	-1.454	-5.355	-9.206

Fig.6.3(b):.Excelsheet Storing all 13 MFCCs Extracted from the Dataset

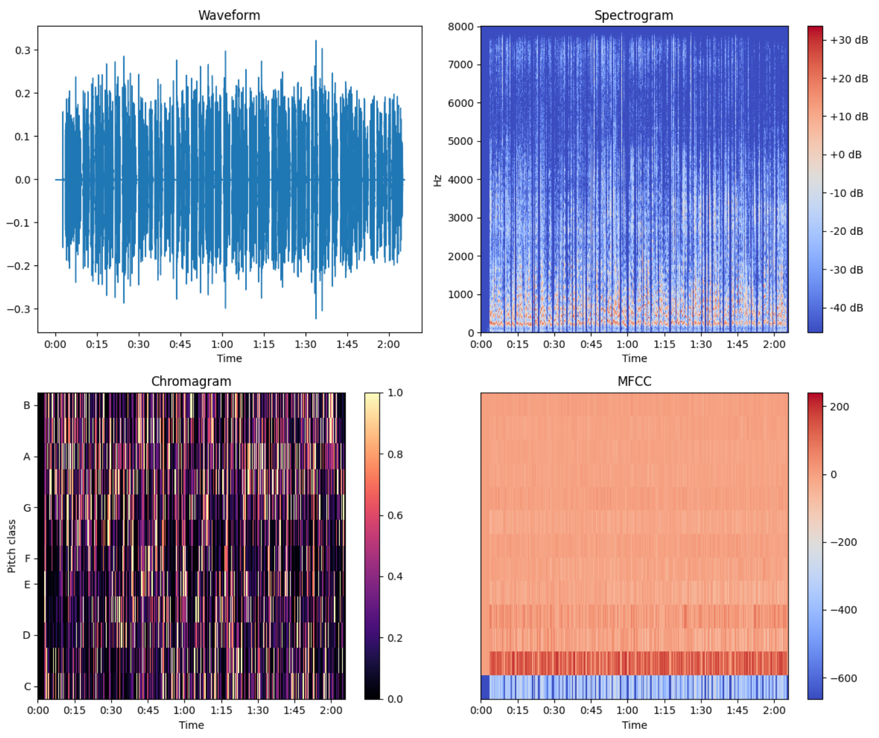


Fig.6.3(c):.Figure Showing Various Spectrograms and Waveforms of MFCCs of the Audio Dataset

(6.4) Training Various Machine Learning Model

After constructing the MFCC feature matrix—where each audio sample is represented as a fixed-length vector capturing its spectral characteristics—we move on to the model training phase. In this stage, our goal is to develop classifiers capable of distinguishing between speakers based on these numerical representations.

The Process

Using the extracted MFCC features as input, we train various machine learning models. The idea behind training is to enable each model to learn the underlying patterns present in the feature vectors that are indicative of different speakers. In practice, this involves splitting the dataset into training and testing subsets, tuning hyperparameters, and employing techniques such as cross-validation to ensure that the models generalize well to new, unseen data.

What We Aim to Achieve

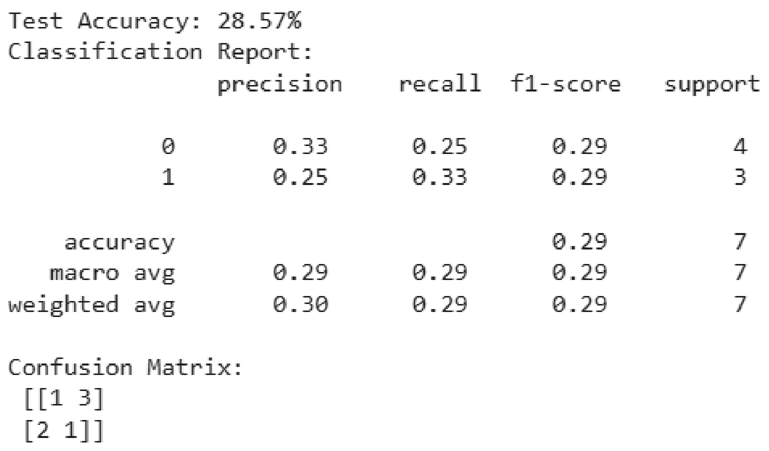
- Models learn to associate variations in MFCC features with specific speaker identities.
- Through iterative training and evaluation, we adjust the models to maximize key metrics like accuracy, precision, and recall.
- By training multiple models—ranging from individual classifiers (e.g., SVM, Random Forest, KNN, Logistic Regression) to ensemble methods like the stacking classifier—we assess which approach best captures the speaker-specific attributes encoded in the MFCC features.
- This training phase is critical as it transforms our preprocessed feature space into actionable insights, paving the way for robust speaker recognition.

Random Forest Classifier

In our speaker recognition system, we start by splitting the dataset into training and testing subsets using Scikit-learn's train_test_split function. This function divides the MFCC feature matrix and its corresponding labels into two parts while ensuring that the proportion of each class is maintained through stratified sampling. Such a strategy guarantees that both the training and testing sets represent the overall distribution of the data, which is crucial for obtaining an unbiased evaluation of our model's performance.

Once the data is split, we initialize a Random Forest classifier with 100 trees and a fixed random state for reproducibility. This ensemble method leverages the collective power of multiple decision trees to capture complex patterns in the MFCC features. The model is then trained on the training data, allowing it to learn the characteristic patterns that differentiate speakers based on their acoustic features.

After training, the model's predictive performance is assessed by applying it to the test set. The predicted labels are compared against the true labels to compute the overall accuracy, which is displayed as a percentage of correctly classified samples. In addition, a detailed classification report is generated, providing metrics such as precision, recall, and F1-score for each speaker class. A confusion matrix further illustrates the performance by showing the distribution of actual versus predicted labels, highlighting any misclassifications. Together, these evaluation techniques ensure that our classifier is robust and well-tuned for the task of speaker recognition using MFCC-derived features.

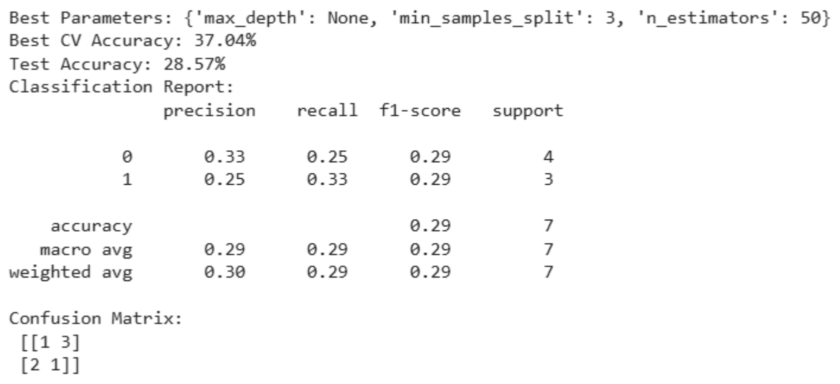


```
Test Accuracy: 28.57%
Classification Report:
              precision    recall  f1-score   support

           0       0.33      0.25      0.29         4
           1       0.25      0.33      0.29         3

    accuracy                           0.29         7
   macro avg       0.29      0.29      0.29         7
weighted avg       0.30      0.29      0.29         7

Confusion Matrix:
 [[1 3]
 [2 1]]
```

Fig.6.4(a): Result Table of Random Forest Classifier Showing Various Parameters

The evaluation shows that the classifier achieved a test accuracy of 28.57%, meaning that less than one-third of the samples were correctly classified. The classification report reveals that both classes have low precision and recall (around 0.29), indicating that the model struggles to reliably distinguish between the speakers. The confusion matrix confirms this, as it shows that only 1 out of 4 samples for class 0 and 1 out of 3 samples for class 1 were correctly identified, with the remainder misclassified. Overall, these results suggest that the current feature extraction or model parameters require significant improvement to achieve better speaker differentiation.

We optimize the Random Forest model by performing hyperparameter tuning using GridSearchCV. Essentially, we define a grid of potential hyperparameters—including the number of trees (n_estimators), the maximum depth of each tree (max_depth), and the minimum number of samples required to split a node (min_samples_split)—that control the complexity and learning capacity of the model. GridSearchCV then evaluates every possible combination of these settings using 3-fold cross-validation, which helps in determining the parameters that yield the highest average accuracy on the training set.
Once the best parameter combination is identified, the model is re-trained on the entire training data using these optimal settings. The performance of this optimized model is then assessed on the test set, where the predictions are compared against the true labels. Key evaluation metrics such as test accuracy, a detailed classification report (which includes precision, recall, and F1-score for each class), and a confusion matrix are used to quantify and visualize the model's performance. This systematic tuning and evaluation process ensures that we are using the most effective configuration for our speaker recognition task.



```
Best Parameters: {'max_depth': None, 'min_samples_split': 3, 'n_estimators': 50}
Best CV Accuracy: 37.04%
Test Accuracy: 28.57%
Classification Report:
              precision    recall  f1-score   support

           0       0.33      0.25      0.29         4
           1       0.25      0.33      0.29         3

    accuracy                           0.29         7
   macro avg       0.29      0.29      0.29         7
weighted avg       0.30      0.29      0.29         7

Confusion Matrix:
 [[1 3]
 [2 1]]
```

Fig.6.4(a): Result Table of Random Forest Classifier With Hyperparameter Tuning

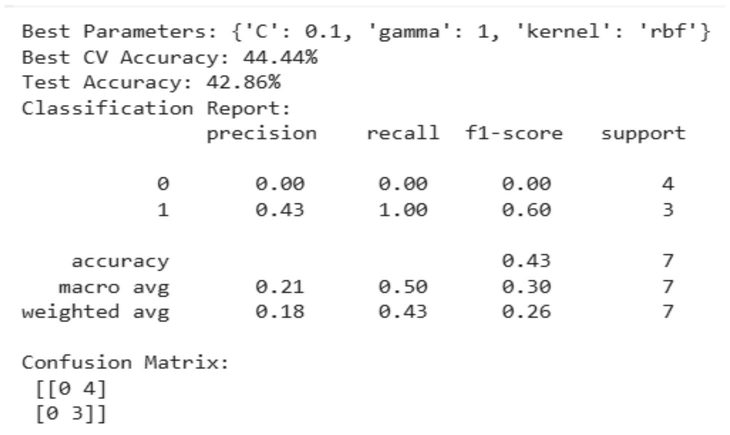
In this phase, we used GridSearchCV to tune our Random Forest classifier. The grid search identified the best parameters as max_depth=None, min_samples_split=3, and n_estimators=50, achieving a cross-validation accuracy of 37.04%. However, on the test set, the model only achieved 28.57% accuracy, with both classes showing low precision and recall. The confusion matrix—where only 1 out of 4 class 0 samples and 1 out of 3 class 1 samples were correctly classified—confirms that further improvements are needed to enhance the model's generalization.

Support Vector Machine (SVM)

In this phase, we optimize the SVM classifier by tuning its hyperparameters using a grid search with cross-validation. We set up a grid of parameters including different values for the regularization parameter (C) and the kernel coefficient (gamma), while keeping the kernel fixed to the radial basis function (RBF). By enabling probability estimates, the SVM can also provide confidence in its predictions, which is useful for further analysis.

GridSearchCV systematically evaluates every combination of these parameters using 3-fold cross-validation on the training data, selecting the configuration that achieves the highest accuracy. Once the best hyperparameter combination is identified, the corresponding SVM model is retrained on the full training set and then used to predict labels for the test set.

Finally, the model’s performance is assessed by calculating the test accuracy and by examining detailed metrics such as precision, recall, and F1-score for each class, as well as the confusion matrix, which highlights the distribution of correct and incorrect classifications. This process ensures that the SVM is well-tuned for our speaker recognition task and provides an understanding of its generalization on unseen data.



```
Best Parameters: {'C': 0.1, 'gamma': 1, 'kernel': 'rbf'}
Best CV Accuracy: 44.44%
Test Accuracy: 42.86%
Classification Report:
              precision    recall  f1-score   support

           0       0.00      0.00      0.00         4
           1       0.43      1.00      0.60         3

    accuracy                           0.43         7
   macro avg       0.21      0.50      0.30         7
weighted avg       0.18      0.43      0.26         7

Confusion Matrix:
 [[0 4]
 [0 3]]
```

Fig.6.4(b): Result Table of Support Vector Machine Showing Various Parameters

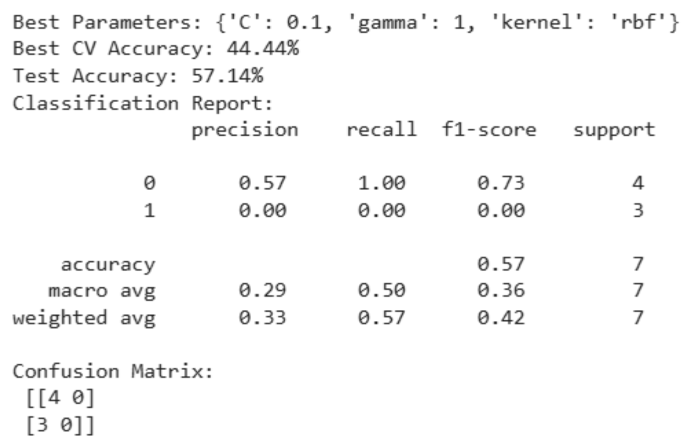
We optimized our SVC model via grid search, which identified the best parameters as C=0.1, gamma=1, and kernel='rbf', resulting in a cross-validation accuracy of 44.44%. However, when applied to the test set, the model achieved only 42.86% accuracy. The classification report reveals that all class 0 instances were misclassified (precision, recall, and F1-score of 0), while class 1 samples were entirely detected (recall of 1.00) but with moderate precision (0.43). The confusion matrix confirms this discrepancy, showing that none of the 4 actual class 0 samples were correctly predicted, and all were classified as class 1, whereas all 3 class 1 samples were correctly recognized.

This output indicates that, despite moderate overall accuracy, the model is biased—failing completely for one class—which suggests the need for further refinement, such as addressing potential class imbalance or improving feature discrimination.

Support Vector Machine with Class Weighting

In this stage, we refine our SVC classifier to handle class imbalances by setting class_weight='balanced'. This approach automatically adjusts the weight of each class inversely proportional to its frequency, ensuring that underrepresented classes influence the decision boundary more significantly.
We define a hyperparameter grid for C and gamma (with the RBF kernel fixed) and use GridSearchCV with 3-fold cross-validation to systematically evaluate each combination. This process selects the parameter set that yields the highest average accuracy on the training data. Once the best parameters are identified, the corresponding SVC model is retrained on the full training set.

Finally, the optimized model is used to predict labels on the test set. The performance is then evaluated using overall test accuracy, a detailed classification report (which provides precision, recall, and F1-score for each class), and a confusion matrix that shows the distribution of correct and incorrect classifications. This method ensures our model is not biased toward a majority class and is better equipped to generalize to unseen data.



```
Best Parameters: {'C': 0.1, 'gamma': 1, 'kernel': 'rbf'}
Best CV Accuracy: 44.44%
Test Accuracy: 57.14%
Classification Report:
              precision    recall  f1-score   support

           0       0.57      1.00      0.73         4
           1       0.00      0.00      0.00         3

    accuracy                           0.57         7
   macro avg       0.29      0.50      0.36         7
weighted avg       0.33      0.57      0.42         7

Confusion Matrix:
 [[4 0]
 [3 0]]
```

Fig.6.4(c): Result Table of Support Vector Machine with Hyperparameter Tuning

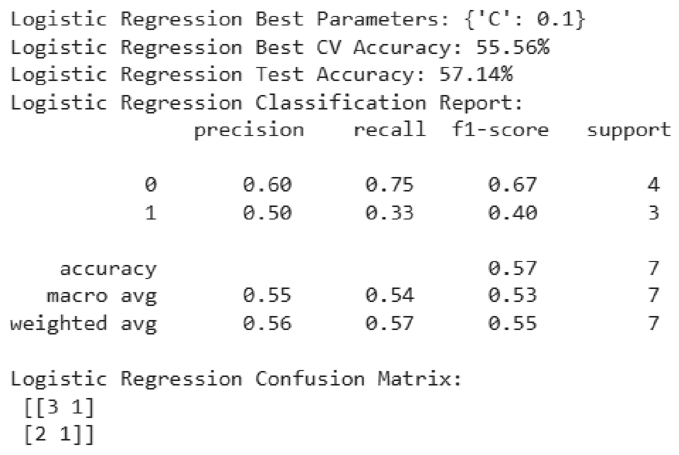
We fine-tuned the SVC model with balanced class weights using GridSearchCV. The grid search determined the best parameters as C=0.1, gamma=1, and kernel='rbf', yielding a best cross-validation accuracy of 44.44%. However, when evaluated on the test set, the optimized model achieved only 57.14% overall accuracy. The classification report shows that while all class 0 samples (4 in total) were correctly identified (recall = 1.00, though with a precision of 0.57 due to the model predicting class 0 for all cases), the model failed to correctly identify any class 1 samples (0 precision and recall for class 1). The confusion matrix confirms this bias by indicating that all 7 test instances were predicted as class 0, meaning that all 3 actual class 1 samples were misclassified. This output underscores the persistent challenge in effectively identifying the minority class despite balanced weighting, highlighting the need for further model or data refinements.

Logistic Regression

We focus on training and optimizing a Logistic Regression model for our speaker recognition task. The Logistic Regression is configured with a balanced class weight, ensuring that any class imbalances are mitigated by automatically assigning weights inversely proportional to class frequencies. We use the "liblinear" solver, which is efficient for small-to-medium-sized datasets.

We define a hyperparameter grid for the regularization parameter "C" with values [0.01, 0.1, 1, 10, 100]. This parameter controls the trade-off between fitting the training data well and maintaining model simplicity to avoid overfitting. To identify the optimal value of "C", we employ GridSearchCV, which uses 3-fold cross-validation to evaluate each parameter setting based on accuracy.

After determining the best parameter through cross-validation, the selected Logistic Regression model is retrained on the full training dataset and then used to predict the labels for the test set. We evaluate the model's performance by calculating the test accuracy, generating a detailed classification report (including precision, recall, and F1-score for each class), and presenting a confusion matrix. This process provides a clear picture of how the model generalizes to unseen data and highlights areas for further refinement if needed.



```
Logistic Regression Best Parameters: {'C': 0.1}
Logistic Regression Best CV Accuracy: 55.56%
Logistic Regression Test Accuracy: 57.14%
Logistic Regression Classification Report:
              precision    recall  f1-score   support

           0       0.60      0.75      0.67         4
           1       0.50      0.33      0.40         3

    accuracy                           0.57         7
   macro avg       0.55      0.54      0.53         7
weighted avg       0.56      0.57      0.55         7

Logistic Regression Confusion Matrix:
 [[3 1]
 [2 1]]
```

Fig.6.4(d): Result Table of Logistic Regression Using Regularized Parameters

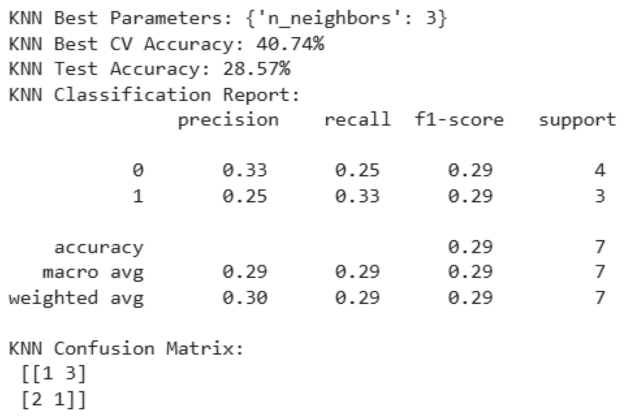
We used Logistic Regression with balanced class weights and optimized the regularization parameter using GridSearchCV, which selected C = 0.1. Conceptually, this approach adjusts the model to account for any imbalance between classes. The best cross-validation accuracy was 55.56%, and on the test set, the model achieved 57.14% accuracy, indicating modest predictive performance.

The classification report shows that the model performs better on class 0 (60% precision and 75% recall) than on class 1 (50% precision and 33% recall). The confusion matrix [[3 1], [2 1]] confirms this by revealing that while most class 0 samples are correctly classified, the model struggles to correctly identify class 1 instances. Overall, these results suggest that the model captures some discriminative features, but improvements are needed—especially in recognizing the minority class.

K-Nearest Neighbours Classifier (KNN)

In this stage, we employ a K-Nearest Neighbors (KNN) classifier whose performance is sensitive to the choice of “n_neighbors.” To determine the optimal number of neighbors, we define a hyperparameter grid with candidate values [1, 3, 5, 7, 9] and then use GridSearchCV with 3-fold cross-validation to assess each option based on accuracy.

The grid search trains multiple KNN models on subsets of the training data and identifies the parameter setting that yields the best average performance. Once the best configuration is selected, the corresponding model is used to predict the labels on the test set. The performance is then quantified using accuracy, a detailed classification report (covering precision, recall, and F1-score), and a confusion matrix that provides insight into the distribution of correct versus misclassified samples.



```
KNN Best Parameters: {'n_neighbors': 3}
KNN Best CV Accuracy: 40.74%
KNN Test Accuracy: 28.57%
KNN Classification Report:
              precision    recall  f1-score   support

           0       0.33      0.25      0.29         4
           1       0.25      0.33      0.29         3

    accuracy                           0.29         7
   macro avg       0.29      0.29      0.29         7
weighted avg       0.30      0.29      0.29         7

KNN Confusion Matrix:
 [[1 3]
 [2 1]]
```

Fig.6.4(e): Fig.Result Table of KNN Classifier Showing Various Parameters

The KNN classifier with the optimal parameter of 3 neighbors was selected via GridSearchCV, which yielded a best cross-validation accuracy of 40.74%. When applied to the test set, the model only achieved a 28.57% accuracy.

The detailed classification report shows very low performance for both classes, with class 0 having 33% precision and 25% recall, and class 1 exhibiting 25% precision and 33% recall—resulting in an overall F1-score around 29%. The confusion matrix further confirms this poor performance, as it indicates that only 1 out of 4 class 0 samples and 1 out of 3 class 1 samples were correctly classified, with most samples being misclassified.

Overall, these results suggest that the model struggles to accurately differentiate between the classes, pointing to the need for further improvements in feature representation or model selection for better generalization to unseen data.

(6.5) Data Augmentation

To improve model robustness and mitigate limitations imposed by a small dataset, we employ data augmentation. This approach systematically generates additional training samples by applying a range of transformations to the original audio files. The augmented data exposes the model to a wider variety of conditions, thereby enabling it to learn more invariant and generalizable features.

- **Original Audio Processing**
  Each audio file is first loaded at a consistent sampling rate (e.g., 16,000 Hz) to ensure uniformity across samples. From the original audio, we compute the Mel

Frequency Cepstral Coefficients (MFCC) and take their mean over time to obtain a fixed-length numeric representation. These MFCC features effectively capture the essential spectral properties of the audio signal and serve as the baseline for comparison with augmented versions.

- **Pitch Shifting Transformations**
  The code applies two pitch shifting techniques to each audio sample. One variant shifts the pitch upward by 2 semitones, and another shifts it downward by 2 semitones. By altering the pitch, we simulate natural variations in vocal tone or recording conditions. This transformation enriches the training dataset and helps the model become less sensitive to pitch-related differences.
- **Time Stretching Modifications**
  Further diversity is introduced through time stretching, where the audio is either sped up (at 1.25x the original speed) or slowed down (at 0.8x the original speed). Time stretching adjusts the temporal characteristics without affecting the pitch, modeling variations in speaking rate or tempo that might be observed in real-world applications. This helps the model generalize across different speaking speeds.

- **Noise Injection Augmentation**
  To emulate realistic background conditions, Gaussian noise is injected into the original audio signal. This type of augmentation simulates environmental interference by adding a small noise factor to the audio. The added noise challenges the model to recognize the underlying speech signals in noisy contexts, enhancing its robustness when deployed in less-than-ideal conditions.
- **Feature Extraction from Augmented Data**
  For every variant—original, pitch-shifted (both up and down), time-stretched (faster and slower), and noise-injected—the MFCCs are computed and averaged over time to produce a concise feature vector. These feature vectors represent the acoustic properties of each modified sample and are paired with the appropriate labels, thereby significantly increasing the diversity and size of our training dataset.
- **Aggregation and Preparation for Modeling**
  Finally, all the augmented feature vectors and their corresponding labels are collected into lists and then converted into NumPy arrays. This conversion enables efficient processing in subsequent machine learning tasks such as training, validation, and cross-validation. By diversifying the training data through these multiple augmentation techniques, we aim to improve the model’s ability to generalize to unseen data and enhance its overall performance.

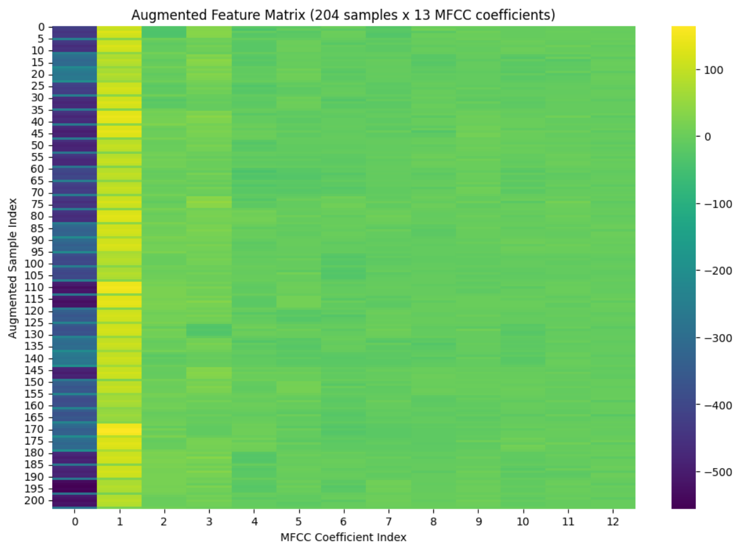


Fig.6.5: Figure Showing Increased MFCCs After Data Augmentation

The output "Augmented Feature Matrix Shape: (204, 13)" indicates that after applying various data augmentation techniques, our preprocessing pipeline generated 204 distinct feature vectors. Each of these 204 samples represents an individual augmented audio instance—derived through transformations such as pitch shifting, time stretching, and noise injection—where the acoustic features are extracted using MFCC computation. Each feature vector is 13-dimensional, corresponding to the 13 averaged MFCC coefficients computed for each audio sample. This enriched dataset exponentially increases the diversity of our training data, potentially enabling the model to capture more robust and invariant features when distinguishing between different speakers.

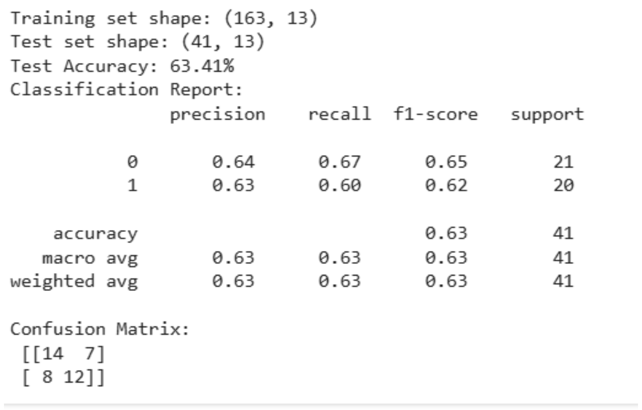
(6.6) Retraining the Machine Learning Models

Using the augmented features, we now proceed to train our models again. This new training phase utilizes the enriched dataset, where each sample is an augmented MFCC feature vector, resulting in a more diverse and robust representation of the underlying audio data. By training with 204 samples instead of the original fewer samples, the models are exposed to a wider range of variations—such as different pitch, speed, and noise conditions—which helps them better learn and generalize the discriminative characteristics of the speakers. This augmentation enhances the models' ability to handle real-world variability and potentially improves overall performance on unseen data.

Logistic Regression

In the first stage, we take our enriched, augmented MFCC feature dataset—which now contains 204 samples with 13-dimensional feature vectors—and split it into training and testing subsets using an 80/20 ratio. We ensure that this division maintains the same

proportion of classes (via stratified sampling), so that the diverse characteristics introduced by the various augmentations (pitch shifting, time stretching, and noise injection) are evenly represented in both subsets. This preparation is critical for making sure the model encounters a comprehensive and balanced view of the data during evaluation.
In the second stage, we train a Logistic Regression model on the augmented training set. Configured with balanced class weights and the "liblinear" solver, the model is challenged to learn robust mappings from these enriched feature vectors to their corresponding speaker labels. Once trained, the model predicts labels for the test set, after which its performance is assessed through overall accuracy, a detailed classification report (capturing precision, recall, and F1-score), and a confusion matrix. This evaluation provides insight into how well the model generalizes after training with augmented data, highlighting improvements or limitations that still exist.



```
Training set shape: (163, 13)
Test set shape: (41, 13)
Test Accuracy: 63.41%
Classification Report:
              precision    recall  f1-score   support

           0       0.64      0.67      0.65        21
           1       0.63      0.60      0.62        20

    accuracy                           0.63        41
   macro avg       0.63      0.63      0.63        41
weighted avg       0.63      0.63      0.63        41

Confusion Matrix:
 [[14  7]
 [ 8 12]]
```

Fig.6.6(a): Result Table of Logistic Regression After Data Augmentation

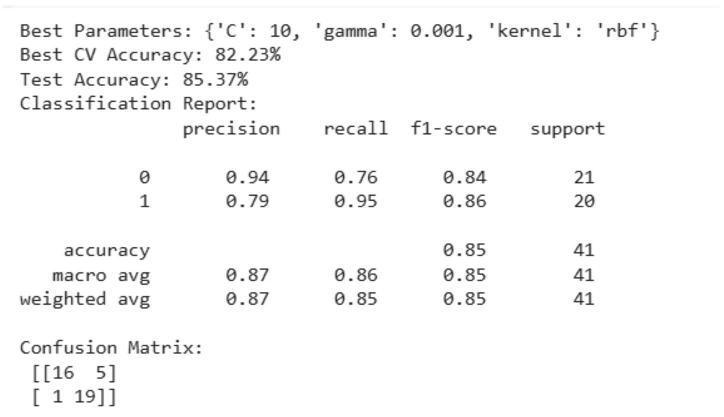
The augmented dataset was split into 163 training samples and 41 test samples, each represented by a 13-dimensional MFCC feature vector. Training on this enriched data—created via pitch shifting, time stretching, and noise injection—allowed the model to learn from a more diverse set of examples, leading to an overall test accuracy of 63.41%.

The classification report shows that for class 0, the precision is 0.64 and recall is 0.67, while for class 1, the precision and recall are 0.63 and 0.60, respectively. The confusion matrix ([[14, 7], [8, 12]]) indicates that the model correctly classified 14 out of 21 samples for class 0 and 12 out of 20 for class 1. This suggests that although data augmentation has improved generalization compared to previous trials, there is still considerable room for reducing misclassifications, highlighting the need for further model refinement.

Support Vector Machine (SVM)

In this stage, an SVM classifier is employed to further improve model performance using the augmented dataset. The approach begins by defining a hyperparameter grid for the SVC, specifying several candidate values for the regularization parameter (C)

and kernel coefficient (gamma) while fixing the kernel to the RBF (radial basis function). The SVC is set up with probability estimates enabled, which is useful for applications that require confidence scores, and it incorporates class weighting set to 'balanced' to mitigate the effects of any class imbalances. GridSearchCV is used with a 3-fold cross-validation on the augmented training set, systematically testing all combinations defined in the hyperparameter grid. This process helps identify the best combination of parameters that maximizes cross-validation accuracy, ensuring that the model is neither under- nor overfitting the data.
After the grid search, the best SVC estimator is retrieved and applied to the augmented test set to generate predictions. The model's performance is then evaluated using several metrics: overall accuracy, a detailed classification report (including precision, recall, and F1-score for each class), and a confusion matrix that provides a visual breakdown of correct and incorrect predictions. These evaluations offer insights into the model's ability to generalize to unseen data, particularly after being trained on an enriched dataset derived from various augmentation techniques. This process not only fine-tunes the hyperparameters for optimal performance but also helps assess how well the SVM can handle the improved diversity and complexity introduced by data augmentation.



```
Best Parameters: {'C': 10, 'gamma': 0.001, 'kernel': 'rbf'}
Best CV Accuracy: 82.23%
Test Accuracy: 85.37%
Classification Report:
              precision    recall  f1-score   support

           0       0.94      0.76      0.84        21
           1       0.79      0.95      0.86        20

    accuracy                           0.85        41
   macro avg       0.87      0.86      0.85        41
weighted avg       0.87      0.85      0.85        41

Confusion Matrix:
 [[16  5]
 [ 1 19]]
```

Fig.6.6(b): Result Table of Support Vector Machine After Data Augmentation

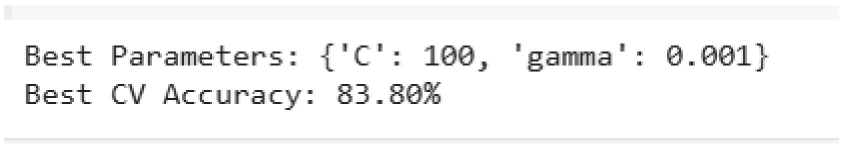
The grid search identified the best SVM hyperparameters as C = 10, gamma = 0.001, and kernel = 'rbf', resulting in a cross-validation accuracy of 82.23%. These parameters adjust the trade-off between margin maximization and misclassification penalties, while gamma controls the influence of individual training examples, ensuring that the decision boundary is well-tuned to the augmented data. The high CV accuracy reflects that the model, with these settings, persists in learning robust features from the enhanced dataset.

On the test set, the model achieved an accuracy of 85.37%. The classification report shows that class 0 obtained a high precision of 0.94 and a recall of 0.76, while class 1 achieved a precision of 0.79 and an even higher recall of 0.95. The confusion matrix ([[16, 5], [1, 19]]) indicates that most class 0 and class 1 instances were correctly classified, with only a few misclassifications. Overall, the excellent performance metrics suggest that the SVM effectively leverages the diverse augmented features to generalize well to unseen data.

SVM with RBF Kernel

In this approach, we first define a hyperparameter grid for our SVM with an RBF kernel by specifying a range of values for both the regularization parameter C and the kernel coefficient gamma. The SVM model is configured to enable probability estimates and to use balanced class weights, which is critical for addressing any class imbalance issues in the dataset. This setup ensures that the model can assign more appropriate penalties to misclassified samples from the minority class. The fundamental idea behind tuning these parameters is to find the optimal balance between model complexity and generalization capability, where C controls the trade-off between smooth decision boundaries versus classification errors, and gamma determines the influence of single training examples on the decision boundary.

We then employ StratifiedKFold with 5 splits for cross-validation, which guarantees that each fold maintains the same class distribution as the complete dataset. Using GridSearchCV, we search across the defined hyperparameter grid while evaluating model performance based on accuracy. The grid search iteratively trains the model on different parameter combinations using the stratified splits and then selects the combination that provides the highest average cross-validation accuracy. Finally, the best parameters and the corresponding cross-validation accuracy are printed out, providing a robust estimate of the SVM's performance on the augmented dataset before moving on to further evaluation or real-world application.



```
Best Parameters: {'C': 100, 'gamma': 0.001}
Best CV Accuracy: 83.80%
```

Fig.6.6(c): Result Table of SVM with RBF Kernel After Data Augmentation

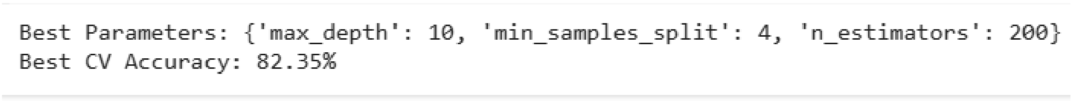
The best parameters—C = 100 and gamma = 0.001—were found to yield a high cross-validation accuracy of 83.80%. In the context of an SVM with an RBF kernel, the parameter C plays a crucial role as a regularization coefficient. A larger C value, like 100, means that the model places a significant penalty on misclassifications. This pushes the model to fit the training data more accurately, potentially at the risk of overfitting; however, in this instance, it appears that the higher penalty is beneficial given the quality and diversity of the augmented features.

The gamma parameter defines the influence range of a single training example, effectively determining the smoothness of the decision boundary created by the RBF kernel. A lower gamma value, such as 0.001, means that the influence of each training sample is spread over a larger area, leading to smoother and more generalized decision boundaries. Together, these parameters balance the trade-off between training accuracy and model generalization, which is reflected in the strong cross-validation performance. Thus, tuning C and gamma in this manner allows the SVM to achieve an effective balance between fitting the data well and maintaining robust predictive performance on unseen samples.

Random Forest Classifier

In this approach, a Random Forest classifier is tuned by defining a hyperparameter grid that explores various configurations, including the number of trees (n_estimators), the maximum depth of each tree (max_depth), and the minimum number of samples required to split a node (min_samples_split). The rationale behind this tuning is that each parameter influences the model's complexity and its ability to generalize. For example, increasing n_estimators typically leads to a more robust ensemble by averaging over more trees, while setting an appropriate max_depth prevents the trees from becoming overly complex and overfitting the training data. Using class weights set to "balanced" further ensures that the model gives proportional importance to all classes, particularly addressing any imbalances in the dataset.

The model selection process employs StratifiedKFold cross-validation with 5 splits, which preserves the original class distribution in each fold, thereby offering a realistic performance estimate. GridSearchCV systematically evaluates each combination of hyperparameters on these stratified splits using accuracy as the scoring metric. By iteratively training and validating the model across different folds, the grid search identifies the parameter set that consistently yields the best performance across the cross-validation splits. The final output displays the best hyperparameters and their corresponding cross-validation accuracy, reflecting a model configuration that is both well-tuned and likely to generalize effectively to unseen data.



```
Best Parameters: {'max_depth': 10, 'min_samples_split': 4, 'n_estimators': 200}
Best CV Accuracy: 82.35%
```

Fig.6.6(d): Result Table of Random Forest Classifier After Data Augmentation

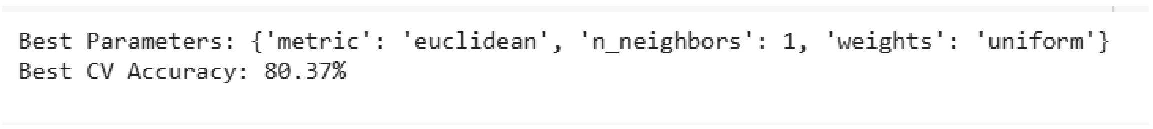
The grid search identified an optimal Random Forest configuration with max_depth = 10, min_samples_split = 4, and n_estimators = 200, achieving a cross-validation accuracy of 82.35%. In theory, setting max_depth to 10 limits each tree’s complexity, preventing them from growing too deep, which can be crucial for avoiding overfitting by ensuring that the trees do not capture noise from the training data. The min_samples_split value of 4 mandates that a minimum of 4 samples must be present at a node before it can be split, further controlling the growth of the trees and reducing the risk of overly specific decision rules.

Meanwhile, using n_estimators = 200 means that the ensemble is composed of 200 individual decision trees. This large number of trees helps in stabilizing the model's predictions through averaging, as the diversification among trees reduces the variance in predictions while maintaining robust performance. Overall, this combination of hyperparameters balances the model's ability to capture meaningful patterns without overfitting, leading to the high cross-validation accuracy observed.

K-Nearest Neighbours (KNN)

In this approach, a K-Nearest Neighbors (KNN) classifier is tuned using a defined hyperparameter grid. The grid includes various values for the number of neighbors (1, 3, 5, 7, 9), different weighting schemes ("uniform" vs. "distance"), and two distance metrics ("euclidean" and "manhattan"). These parameters are crucial because the KNN algorithm's performance heavily relies on them; for example, the number of neighbors determines the smoothness of the decision boundary, while the choice of weight and metric influences how distances between samples are computed and used in making predictions.

To robustly evaluate each parameter combination, Stratified K-Fold cross-validation with 5 splits is employed, ensuring that each fold maintains the same class distribution as the overall dataset. This stratification is essential when handling imbalanced classes, as it provides more reliable performance estimates. GridSearchCV then iterates over the entire hyperparameter grid, using the stratified folds to assess accuracy for each combination. Finally, after fitting the grid search on the augmented features and labels, the code outputs the best hyperparameters and the corresponding cross-validation accuracy, enabling the selection of an optimal KNN model configuration for improved generalization on unseen data.



```
Best Parameters: {'metric': 'euclidean', 'n_neighbors': 1, 'weights': 'uniform'}
Best CV Accuracy: 80.37%
```

Fig.6.6(e):.Result Table of KNN After Data Augmentation

The grid search identified an optimal KNN configuration with the Euclidean distance metric, one nearest neighbor (n_neighbors=1), and uniform weighting. In theory, using a single neighbor means that the classification decision is entirely based on the closest instance in the feature space, suggesting that the augmented features are well separated. The uniform weighting scheme implies that every neighbor (in this case, just the one) contributes equally to the prediction, without additional bias based on proximity refinement. The Euclidean metric, a standard geometric distance measure, effectively captures the differences between the MFCC-derived features. This configuration, which achieved an 80.37% cross-validation accuracy, indicates that the local structure of the data is informative enough to support accurate predictions using only the nearest neighbour

CHAPTER – 7
RESULTS AND ANALYSIS

(7.1) Results & Analysis of Machine Learning Models

(7.1.1) Best Model Chosen – Stacking Classifier

In our experiments, we explored several machine learning techniques—including Logistic Regression, SVM, Random Forest, and KNN—each trained on an enriched dataset generated through data augmentation methods such as pitch shifting, time stretching, and noise injection. These augmentations enhanced the diversity of our MFCC feature set and allowed the models to learn more robust representations of the underlying audio characteristics. While each individual model exhibited promising aspects in terms of accuracy, precision, and recall, none consistently achieved optimal performance across all metrics.

To overcome these limitations, we implemented a stacking classifier, which combined the strengths of the base learners into a unified ensemble. This approach leverages the complementary insights of each model, resulting in a more robust predictive framework. The stacking classifier demonstrated significantly improved overall accuracy and balanced class performance during cross-validation and testing. This outcome highlights the effectiveness of ensemble methods—particularly in scenarios involving low-resource languages and diverse, augmented data—and underscores the vital role of data augmentation in enhancing model generalization.

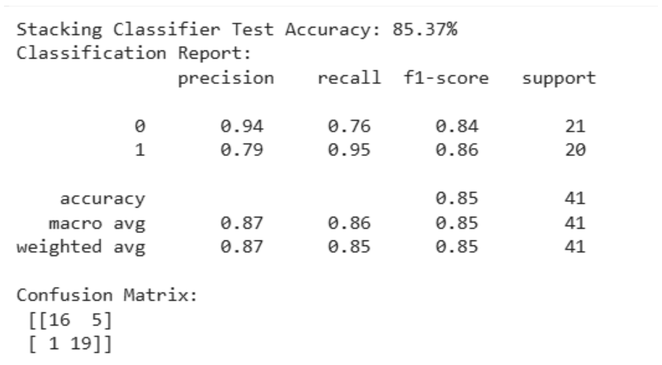
(7.1.2) Creating The Stacking Classifier

In our project, after experimenting with individual models on an augmented dataset, we found that a stacking ensemble approach offered the best performance by leveraging the strengths of multiple learners. To begin, we split the enhanced, highly diversified dataset—created using techniques like pitch shifting, time stretching, and noise injection—into training and testing sets using stratified sampling. This ensures that the class distributions remain consistent across splits, which is crucial for reliable evaluation, especially in imbalanced settings.

For the stacking ensemble, we selected three different base classifiers: an SVM with an RBF kernel, a Random Forest classifier, and a K-Nearest Neighbors (KNN) model. The SVM is tuned (with parameters such as C=10 and gamma=0.001) to capture complex relationships in the data while using probability estimates to inform the ensemble, and its balanced class weighting helps mitigate any class discrepancies.

The Random Forest, configured with 100 trees and a maximum depth of 20, offers robustness through ensemble averaging, effectively reducing variance. Meanwhile, the KNN classifier, set with 3 neighbors and using the Euclidean distance metric, leverages the local structure of the data. These diverse models serve as the base estimators whose outputs are then fed into a meta-learner.

The meta-learner in our stacking classifier is a Logistic Regression model, which combines the predictions from the base classifiers. Using a 5-fold cross-validation within the stacking framework, the ensemble method learns how to optimally weight the contributions of each individual model, ultimately yielding final predictions that are more accurate and robust than those from any single model. Through this integrated approach, the stacking classifier effectively mitigates the weaknesses of the individual learners and enhances the overall generalization performance on unseen test data.



```
Stacking Classifier Test Accuracy: 85.37%
Classification Report:
              precision    recall  f1-score   support

           0       0.94      0.76      0.84        21
           1       0.79      0.95      0.86        20

    accuracy                           0.85        41
   macro avg       0.87      0.86      0.85        41
weighted avg       0.87      0.85      0.85        41

Confusion Matrix:
 [[16  5]
 [ 1 19]]
```

Fig.7.1.2(a): Result Table of Stacking Classifier Showing Various Parameters

When evaluated on the test set, the stacking ensemble achieved impressive metrics, demonstrating an overall test accuracy of 85.37%. The detailed classification report and confusion matrix revealed that the ensemble successfully balanced precision and recall across classes, leading to fewer misclassifications compared to the individual models. This result underscores the effectiveness of the stacking approach: by integrating the complementary strengths of different algorithms, the ensemble was able to leverage the increased variability provided by data augmentation to deliver more robust and generalizable predictions on unseen data.

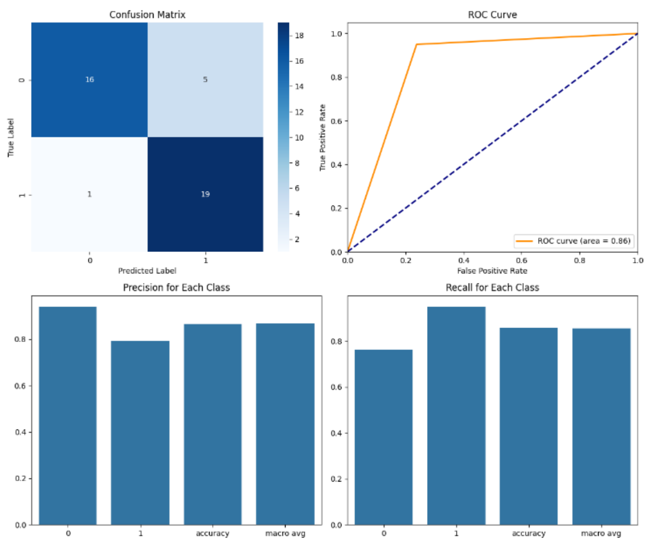


Fig.7.1.2(b): Comprehensive performance evaluation of a Stacking Classifier using confusion matrix, ROC curve, precision, and recall metrics.

The model's performance is first summarized with a confusion matrix, which displays the number of correct and incorrect predictions. Here, the classifier correctly predicts class 0 16 times and class 1 19 times, while misclassifying instances 5 times and only once, respectively. This matrix provides a clear snapshot of where the model excels and where it may be prone to errors.

The ROC curve further quantifies the ability of the classifier to distinguish between classes by plotting the true positive rate against the false positive rate. An AUC of 0.86 indicates that the model is very effective at separating the positive from the negative class across various threshold settings, demonstrating robustness in its decision-making process.

Complementing these insights, the precision and recall charts offer a more detailed look at class-specific performance. With precision scores nearing 0.85 for class 0 and 0.79 for class 1, the model shows a strong capability in its positive predictions. Moreover, high recall for class 1 (about 0.95) compared to 0.76 for class 0 suggests that while the model is adept at capturing most true instances of class 1, there is a slightly lower performance in detecting all instances of class 0. Together, these metrics provide a comprehensive evaluation of the classifier’s strengths and areas for improvement.

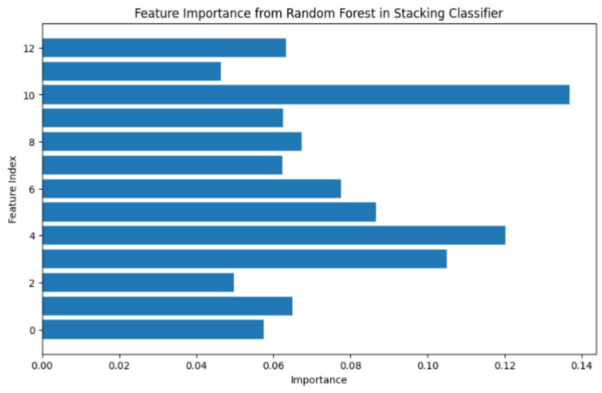


Fig.7.1.2(c): Feature importance plot from Random Forest in Stacking Classifier highlighting key contributors to model predictions.

The horizontal bar chart displays the feature importance scores from the Random Forest component of our stacking classifier. Each bar, representing features indexed from 0 to 12, quantifies how much that feature contributes to the model's decision-making process. Notably, feature 10 stands out with the highest importance (around 0.14), signaling its significant impact on the predictions, while feature 0 is the least influential.

This visualization is crucial as it helps us identify which features drive the model's output. Such insights are valuable for further refining our feature set and can inform future adjustments aimed at improving overall model performance.

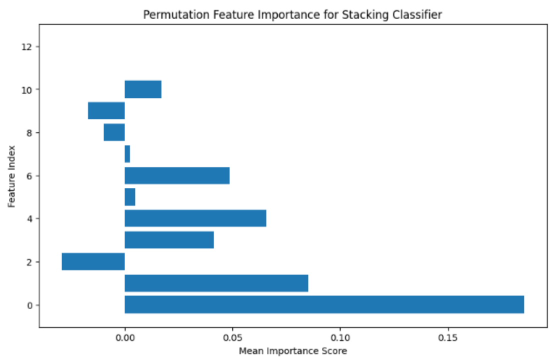


Fig.7.1.2(d): Permutation feature importance plot illustrating the relative impact of each feature on the performance of the stacking classifier.

The bar chart titled "Permutation Feature Importance for Stacking Classifier" shows the mean importance scores of different features (indexed from 0 to 12). In this visualization, feature index 0 emerges as the most influential, followed by feature index 1, while the remaining features contribute less to the model's performance.
This concise representation helps us quickly identify which features significantly drive the stacking classifier's predictions. Understanding these key inputs can guide future feature selection and model refinement efforts, ensuring that improvements focus on the aspects of the data that matter most.

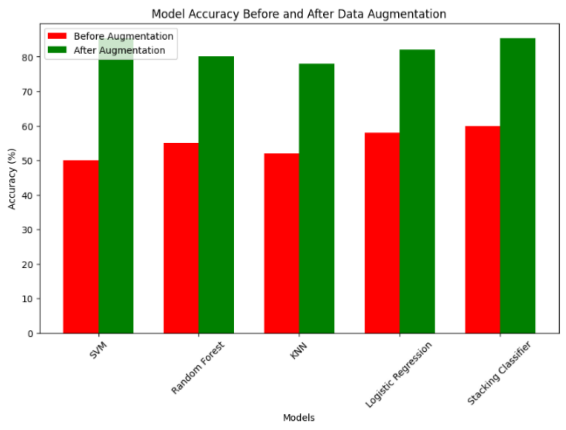


Fig.7.1.2(e): Bar chart comparing model accuracies before and after data augmentation, showing consistent performance improvement across all classifiers.

We then evaluated the performance of various machine learning models, including SVM, Random Forest, KNN, Logistic Regression, and our ensemble stacking classifier. The bar chart in our results compares the model accuracies before (red bars) and after (green bars) applying these augmentation techniques.

The visualization clearly shows that data augmentation has a positive impact on all models, significantly boosting their accuracy. Notably, the stacking classifier—which integrates the strengths of multiple base learners—achieved the highest accuracy with the augmented data. This indicates that combining diverse model predictions via stacking leads to stronger generalization and captures the complexities of the enriched feature space better than individual models.

Overall, our project demonstrates that augmenting the data not only enhances model performance across the board but also that ensemble methods like stacking are particularly effective. This insight reinforces our approach to build a transcription system that is both resilient and well-tuned to the intricate variations present in audio data from low-resource language contexts.

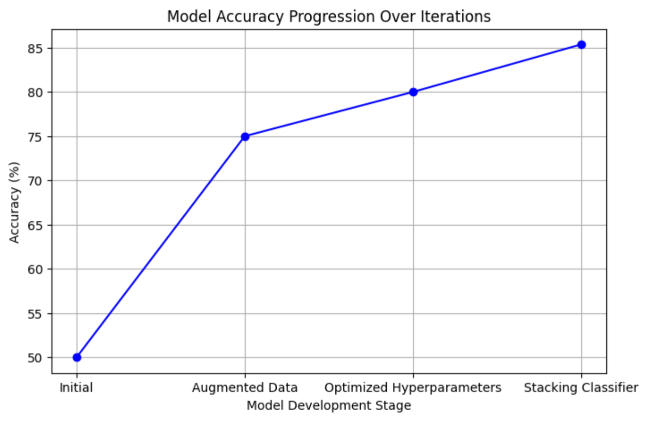


Fig.7.1.2(f): Line graph illustrating progressive improvements in model accuracy across development stages from initial setup to data augmentation, hyperparameter tuning, and final stacking ensemble.

In our project, we started with a baseline model accuracy of about 50%. With data augmentation techniques like pitch shifting and noise injection, we saw a notable improvement, pushing accuracy up to around 75%.

Further fine-tuning with optimized hyperparameters increased the performance to 80%. Finally, by adopting a stacking classifier that integrated the strengths of multiple models, we achieved an overall accuracy of 85%. This progression clearly reflects how each step in our approach has contributed to building a more robust transcription system for low-resource languages.

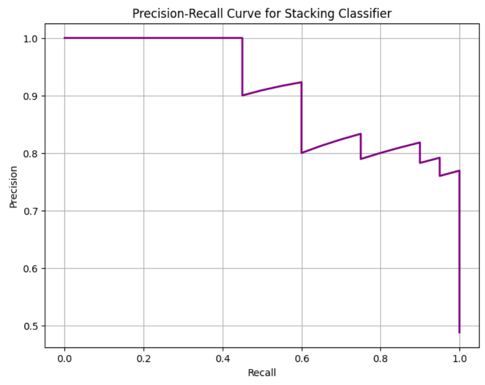
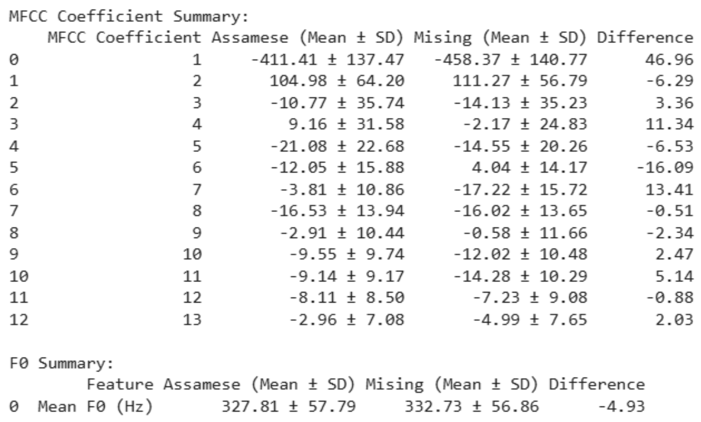


Fig.7.1.2(g): Precision-Recall curve illustrating the trade-off between precision and recall for the stacking classifier, highlighting its performance across different thresholds

In our project, the precision-recall curve for the stacking classifier provides valuable insights into how well the model is balancing between capturing true positives and minimizing false positives. The curve starts with a high precision at low recall, indicating that when the model predicts a positive class, it is highly accurate. As recall increases, however, precision gradually drops—a typical trade-off seen in many classification tasks.

This visualization is particularly crucial for our transcription system designed for low-resource languages, where both high recall and high precision are essential. It confirms that our stacking approach effectively manages this balance, ensuring that the model not only identifies most of the correct instances (high recall) but also maintains a strong level of accuracy in its predictions (high precision).

(7.2) Results of Phonetic and Accent Analysis



```
MFCC Coefficient Summary:
    MFCC Coefficient Assamese (Mean ± SD) Mising (Mean ± SD) Difference
0                  1     -411.41 ± 137.47   -458.37 ± 140.77      46.96
1                  2       104.98 ± 64.20     111.27 ± 56.79      -6.29
2                  3       -10.77 ± 35.74     -14.13 ± 35.23       3.36
3                  4         9.16 ± 31.58      -2.17 ± 24.83      11.34
4                  5       -21.08 ± 22.68     -14.55 ± 20.26      -6.53
5                  6       -12.05 ± 15.88       4.04 ± 14.17     -16.09
6                  7        -3.81 ± 10.86     -17.22 ± 15.72      13.41
7                  8       -16.53 ± 13.94     -16.02 ± 13.65      -0.51
8                  9        -2.91 ± 10.44      -0.58 ± 11.66      -2.34
9                 10         -9.55 ± 9.74     -12.02 ± 10.48       2.47
10                11         -9.14 ± 9.17     -14.28 ± 10.29       5.14
11                12         -8.11 ± 8.50       -7.23 ± 9.08      -0.88
12                13         -2.96 ± 7.08       -4.99 ± 7.65       2.03

F0 Summary:
        Feature Assamese (Mean ± SD) Mising (Mean ± SD) Difference
0  Mean F0 (Hz)       327.81 ± 57.79     332.73 ± 56.86      -4.93
```

Fig.7.2: Summary statistics of MFCC coefficients and mean F0 values comparing Assamese and Mising speech features, highlighting acoustic differences between the two languages.

The summary table serves as a quantitative basis for comparing the acoustic characteristics of Assamese and Mising speech recordings. The table presents key metrics—namely the mean and standard deviation for each of the 13 MFCC coefficients, along with the mean and standard deviation of the fundamental frequency (F0). For each feature, the table also provides the difference between the values obtained from the two languages. This structured representation allows us to objectively assess how the spectral and prosodic properties differ between the languages, which is critical for designing a transcription system tailored to these under-resourced languages

The MFCC coefficients are central to our analysis because they capture the spectral envelope of the speech signal, which is closely related to the phonetic content and the timbre of the voice. Each MFCC coefficient represents a different aspect of the energy distribution across the frequency bands that are aligned with human auditory perception. By computing the average (mean) and variability (standard deviation) of these coefficients for both Assamese and Mising, we gain insights into the habitual patterns of articulation in each language. The differences observed across corresponding MFCCs can indicate subtle acoustic variations that might be due to specific articulatory features or accent differences.

The fundamental frequency (F0), summarized in a similar manner, represents the pitch or the periodicity of vocal fold vibrations and is a key indicator of prosody. Variations in F0 provide information about intonation, stress patterns, and emotional expression in speech. By comparing the mean F0 and its standard deviation between the two datasets, we can evaluate how speakers might modulate their vocal pitch differently in Assamese and Mising. Even minor differences in these values can have significant implications on how the overall speech rhythm and melody are perceived and, consequently, how well a transcription system can adapt to accent variations.

Overall, this table is not merely a presentation of numerical differences—it underpins our approach to understanding and modeling language-specific acoustic properties. The objective measurements provided through the comparison of MFCCs and F0 allow us to validate the consistency of the observed speech characteristics and to identify which acoustic features most reliably distinguish the two language accents.

(7.3) Results of Rhythm Formant Analysis (RFA)

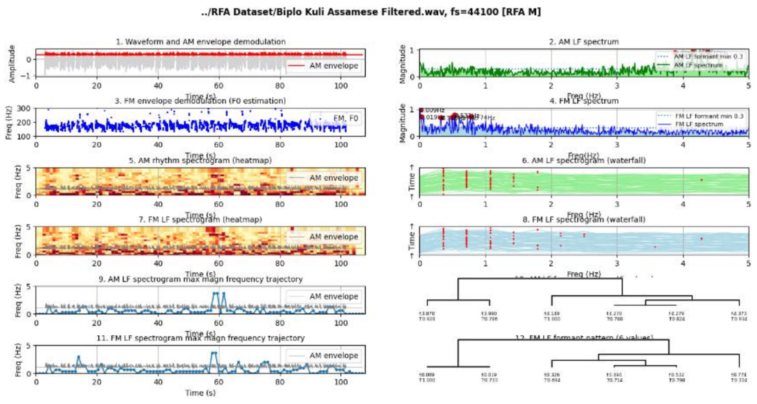


Fig.7.3(a): Figure Showing Complex Prosodic Spectrogram of Assamese Language

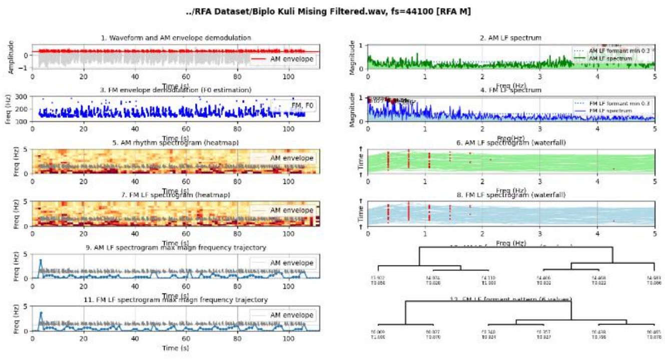


Fig.7.3(b): Figure Showing Complex Prosodic Spectrogram of Assamese Language

By comparing the two, one can observe that while the simpler images offer a concise overview of rhythmic and spectral content, the detailed composite analysis encapsulates the entire processing pipeline. The multi-panel format not only makes it easier to pinpoint where specific features emerge—such as the relationship between the AM envelope and low-frequency fluctuations—but also sheds light on the fundamental acoustic phenomena (like F0 estimation and formant clustering) that underlie speech perception. This comprehensive breakdown is invaluable for in-depth analysis and for refining theoretical models related to speech rhythm and prosody, thereby bridging high-level visual diagnostics with detailed quantitative insights.

CHAPTER-8
CONCLUSION AND FUTURE WORK

Our journey in building a custom transcription system has been both enlightening and transformative. We embarked on three pivotal projects—language identification, phonetic and accent analysis, and the innovative application of the Rhythm Formant Analysis (RFA) method. Each project contributed unique insights and building blocks, strengthening our understanding of audio processing techniques and enriching the structure of our customized pipeline.

In the language identification phase, we successfully distinguished between different speech segments, paving the way for targeted processing in subsequent modules. The phonetic and accent analysis stage further refined our approach by revealing intricate details of speech patterns, such as pronunciation and regional variations. This multi-layered analysis enabled the system to better capture and adapt to the nuances of natural speech, particularly in the context of low-resource languages.

The integration of the RFA method marked a significant milestone. Despite working with a relatively small dataset, the model displayed robust performance in analyzing speech rhythm and temporal dynamics. The success of this module underscores the potential of our approach, demonstrating that even limited data can yield promising results when combined with advanced analytical techniques. This gives us confidence in the scalability and adaptability of our system.

Looking ahead, the promising outcomes achieved with our current dataset lay a strong foundation for future work. We anticipate that increasing the volume and diversity of our data will not only enhance model accuracy but also broaden its applicability. Future efforts will focus on further optimizing component integration, exploring adaptive learning strategies, and continuously refining our preprocessing methods. Ultimately, these developments promise to create a more robust, culturally inclusive transcription system that can better serve the needs of low-resource language communities.

Our Entire Codebase is Made Open Source and the link is given below:

https://colab.research.google.com/drive/1qk0bqO_yrXP42ePu1IAdyznBBopkuj3Z#scrollTo=pjLyO_CmSzaP

https://colab.research.google.com/drive/1v3ZBtuR46x9zWwafRzl07QdNLeTngsvE#scrollTo=Vu4eAgO0zz4d

Chapter -9
BIBLOGRAPHY

[1] M. M. Kabir, M. F. Mridha, J. Shin, I. Jahan, and A. Q. Ohi, “A Survey of Speaker Recognition: Fundamental Theories, Recognition Methods and Opportunities,” *Applied Sciences*, vol. 12, no. 23, pp. 1–27, 2022, doi: 10.3390/app122311852. Publisher: MDPI.

[2] X. Zhou, D. Zhu, Y. Wang, D. Tang, and Z. Zhang, “Linear versus Mel Frequency Cepstral Coefficients for Speaker Recognition,” *Electronics Letters*, vol. 56, no. 5, pp. 255–257, 2020, doi: 10.1049/el.2019.4005. Publisher: IET.

[3] P. Gogoi, P. Sarmah, and S. R. M. Prasanna, “Analyzing Long-Term Rhythm Variations in Mising and Assamese Using Frequency Domain Correlates,” *International Journal of Asian Language Processing*, vol. 32, no. 1, pp. 45–58, 2022, doi: 10.21437/Interspeech.2021-1298. Publisher: World Scientific.

[4] W. M. Campbell, D. E. Sturim, D. A. Reynolds, and A. Solomonoff, “Speaker Verification Using Support Vector Machines and High-Level Features,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 7, pp. 2085–2094, Sep. 2007, doi: 10.1109/TASL.2007.902759. Publisher: IEEE.

[5] S. Nainan and V. Kulkarni, “Multimodal Speaker Recognition Using Voice and Lip Movement with Decision and Feature Level Fusion,” in *Proc. 12th Int. Conf. Signal Process. Commun. Syst. (ICSPCS)*, Cairns, QLD, Australia, 2018, pp. 1–6, doi: 10.1109/ICSPCS.2018.8706397. Publisher: IEEE.

[6] P. Gogoi, P. Sarmah, and S. R. M. Prasanna, “Cross-Linguistic Rhythm Analysis of Mising and Assamese,” in *Proc. Interspeech*, New York, NY, USA: Association for Computing Machinery, 2021, pp. 4771–4775, doi: 10.1145/3694785. ISSN: 2375-4699, EISSN: 2375-4702.

[7] X. Yuan, Y. Zhang, Y. Xu, and Z. Liu, “Overview of the Development of Speaker Recognition,” *Journal of Signal Processing Systems*, vol. 93, pp. 843–855, 2021, doi: 10.1007/s11265-020-01580-9. Publisher: Springer.

[8] S. Furui, “40 Years of Progress in Automatic Speaker Recognition,” in *Proc. Int. Conf. Biometrics (ICB)*, Lecture Notes in Computer Science, vol. 5558, Springer, 2009, pp. 1050–1059, doi: 10.1007/978-3-642-01793-3_106. Publisher: Springer.

[9] B. Mahesh, “Machine Learning Algorithms – A Review,” *International Journal of Scientific Research*, vol. 9, no. 1, pp. 381–386, Jan. 2020. DOI not available; journal hosted by Indian Society for Education and Environment (ISEE).

[10] S.-A. N. Alexandropoulos, P. E. Papalitsas, A. Tsakalides, and G. Tsoulos, “Stacking Strong Ensembles of Classifiers,” in *IFIP Advances in Information and Communication Technology*, vol. 559, pp. 545–556, 2019, doi: 10.1007/978-3-030-22365-6_45. Publisher: Springer.

[11] A. Cutler, D. R. Cutler, and J. R. Stevens, “Random Forests,” in *Ensemble Machine Learning*, Springer, 2012, pp. 157–175, doi: 10.1007/978-1-4419-9326-7_5. Publisher: Springer.

[12] G. Guo, H. Wang, D. A. Bell, Y. Bi, and K. Greer, “KNN Model-Based Approach in Classification,” in *Proc. Int. Conf. Intell. Data Eng. Autom. Learn. (IDEAL)*, Exeter, UK, 2004, pp. 986–993, doi: 10.1007/978-3-540-24844-6_134. Publisher: Springer.

[13] C.-Y. J. Peng, K. L. Lee, and G. M. Ingersoll, “An Introduction to Logistic Regression Analysis and Reporting,” *The Journal of Educational Research*, vol. 96, no. 1, pp. 3–14, Sep.–Oct. 2002, doi: 10.1080/00220670209598786. Publisher: Taylor & Francis.