CHAPTER 3
THEORETICAL BACKGROUND

This chapter describes the background theory of preprocessing audio signals, feature extraction from speech and gender classification techniques. Detailed descriptions of algorithms used in the proposed system are discussed. The classification task involves the implementation of various classifiers for gender identification task. The classifiers are chosen mainly based on the non-linear nature of data. An Artificial Neural Network (ANNs), ensemble method random forest (RFs), Logistic Regression (LR), Support Vector Machine (SVM) and Gaussian Naive Bayes (GNB) are used to develop the classification model.

3.1. Preprocessing

The speech recordings of children consist of many silence and unvoiced regions. Audio trimming is done to remove silence duration and unvoiced regions from start and end of each audio file. Trimming the audio to remove silence helps to reduce the length of the dataset, and it also helps in removing redundant data. The portion of the signal with silence is trimmed as it does not contain any useful information. The same procedure is done at the end of the audio signal. Librosa's functions are applied for preprocessing and librosa is a python package for music and audio analysis. It provides the building blocks necessary to create music information retrieval systems.

An audio signal is represented as a one-dimensional Numpy array, denoted as y throughout librosa. Typically the signal y is accompanied by the sampling rate (denoted sr) which denotes the frequency (in Hz) at which values of y are sampled. The duration of a signal can then be computed by dividing the number of samples by the sampling rate. By default, when loading stereo audio files, the librosa.load() function down mixes to mono by averaging left and right channels, and then resamples the monophonic signal to the default rate sr=2250 Hz.

The librosa.effects.trim function is used for trimming silence and unvoiced

region. This function includes important parameters: y, top_db, ref, frame_length, hop_length. Y is input audio signal and this can be mono or stereo. Top_db is threshold value in decibels below reference to consider as silence and default is 60db. Ref is the reference power and by default, it uses np.max and compares to the peak power in the signal. Frame_length is the number of samples per analysis frame. Hop_length is the number of samples between analysis frames. Most audio analysis methods operate not at the native sampling rate of the signal, but over small frames of the signal which are spaced by a hop length (in samples). The default frame and hop lengths are set to 2048 and 512 samples, respectively. After applying this function, the trimmed signal without silence and unvoiced speech is obtained.

3.2. Feature Extraction from Children's Speech

Classifying the gender of a person based on their voice is a challenging problem in speech recognition. There are numerous machine learning, deep learning models to classify the person is male or female based on speech. Deep learning models are more appropriate for unstructured data such as audio, video and images. Deep learning models perform better results when the data is large. Many researches have made for gender identification in adults using various classification approaches and feature combinations. Feature extraction from a given signal is the most significant phase in gender identification. For efficacy of recognition mechanism robust features are required. The main part of article is the feature extraction mechanism with the detailed process for the MFCC features. There are many techniques for the feature extraction but the advantage of using MFCC technique as a method for extracting features is giving robust and concise features that produces with high accuracy in the recognition process and provide with effective results during classification mechanism.

The most widely used features for speech recognition are the acoustic features namely MFCC. In sound processing, the mel-frequency cepstrum (MFC) is a representation of the short-term power spectrum of a sound. The power spectrum describes the distribution of power into frequency components composing that signal. MFCCs are coefficients that collectively make up an MFC. The reason for MFCC being most commonly used for extracting features is that it is most nearest to the actual human auditory speech perception. This method is considered to be the best available approximation of human ear. MFCC technique takes frequency domain as

its standard base and thus it approximates the human system response more closely than any other system. It is based on the short term analysis, and thus from each frame of speech signal a MFCC vector is computed. MFCC feature extraction method is less complex in implementation and more effective and robust under various conditions. It is a standard method for feature extraction in speech recognition. Figure 3.1 shows the detail process of MFCC feature extraction of audio files [72].

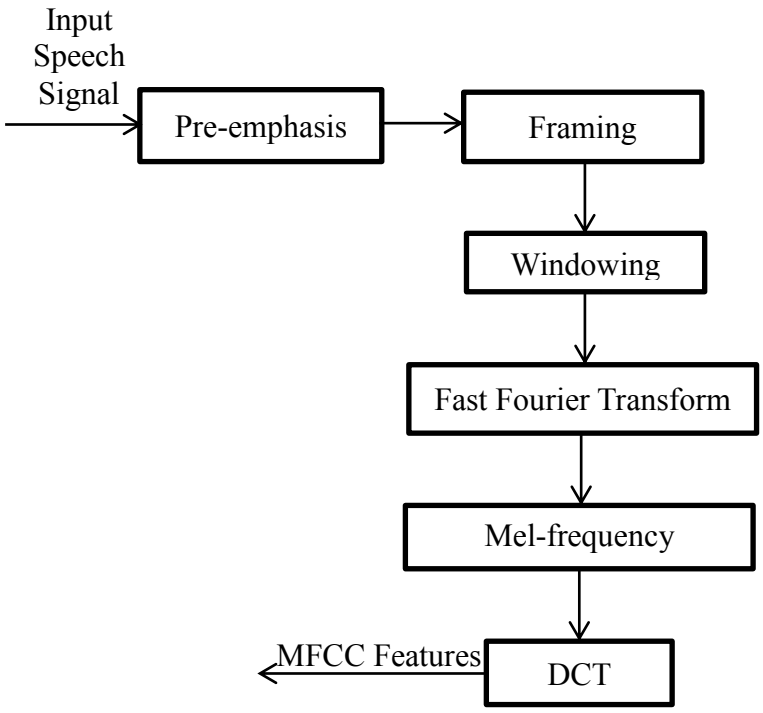

Figure 3.1. Steps for Computing MFCCs
Source: [72]

3.2.1. Pre-emphasis

Typically, prior to some process, such as transmission over cable, or recording to phonograph record or tape, the input frequency range which is susceptible as noise is boosted. This is referred to as "pre-emphasis". Pre-emphasis is a way of compensating for the rapid decaying spectrum of speech. Due to the structure of human speech production system, damping occurs in high-frequency regions. Especially, higher frequencies will have less energy compared to the lower ones, thus, getting poor results with the prediction model. To cope with this, a high pass filter is applied on the signal in order to enhance these components. The filter increases the

energy of higher frequency signal and decreases the energy of lower frequency signal and obtains a much evenly distributed spectrum. This is called the pre-emphasizing step [4]. Widely used pre-emphasis filter is given as,

$$Y(n) = x(n) - \alpha * x(n-1), \alpha \approx (0.95 – 0.97) \quad \text{Equation 3.1}$$

Where, Y (n) = pre-emphasis signal, x (n) = input signal and α= filter coefficient.

3.2.2. Framing

Like in all voice analysis ways, also MFCC methodology is applied on the short parts where voice has stationary acoustic features. Framing is the process of blocking the speech signal into short portion of n samples known as frames in the time domain [73]. These frames are generally selected as 20-30 milliseconds, a shift of 10-15 milliseconds along the signal. If the signal is framed into 25ms, the frame length for a 16 kHz signal is 0.025*16000 = 400 samples. If the frame step is 10ms (160 samples), that permits some overlap to the frames, the earliest 400 sample frame starts at sample 0, the next 400 sample frame starts at sample 160 etc., until the ending of the speech file is reached.

3.2.3. Windowing

When frequency content of a signal is computed, errors can and do arise when taking a limited-duration snapshot of a signal that actually lasts for a longer time. Windowing is a way to reduce these errors, though it cannot eliminate them completely. After framing step, every individual frame is windowed using window function. Each of the above frames is multiplied with a window function to minimize signal discontinuities at the beginning and at the end of each frame [73]. This step makes the end of each frame connects smoothly with the beginning of the next. MFCC uses hamming window and the equation is as follows:

$$Y(n) = x(n) \,.\, W(n) \quad \text{Equation 3.2}$$

Where, n = number of samples in each frame, Y (n) = output signal, x (n) = input signal and W (n) = Hamming window. The Hamming window stops just shy of zero, meaning that the signal will still have a slight discontinuity. Windowing functions act

on raw data to reduce the effects of the leakage that occurs during an FFT of the data. Hamming window has the following form:

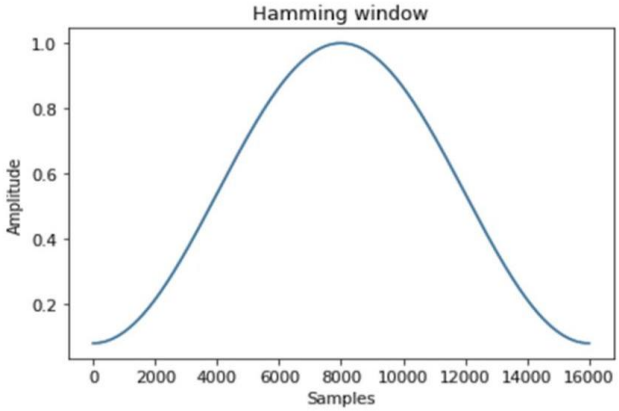

Figure 3.2. Hamming Window

Source: [73]

3.2.4. Fast Fourier Transform

Fast Fourier Transform (FFT) is a process of changing time domain into frequency domain. N-point FFT is applied on each frame to calculate the frequency spectrum, that is also called Short-Time Fourier-Transform (STFT), where N is typically 256 or 512. The FFT size must be longer than the frame length. If N is smaller than the length of the input, the input is removed. If it is larger, the input is filled with zeros. After that, compute the power spectrum (periodogram) using the following Equation 3.4 [73]:

$$P = \frac{|FFT(x_i)|^2}{N} \qquad \text{Equation 3.3}$$

Where, P= power spectrum (periodogram), x_i = the i^{th} frame of signal x.

3.2.5. Mel-frequency Warping

The mel-scale correlates perceived frequency, or pitch, of a pure tone to its actual measured frequency. Humans are much better at recognizing small changes in pitch at low frequencies than they are at high frequencies. Incorporating this scale make our features match more closely what humans hear. To convert the achieved amplitude spectrum into mel-scale, a filter set placed linearly with respect to mel-scale is used. This set consists of triangle band pass filters that are overlapping 50% . Generally, the value of filter coefficient is selected between 20 and 40.Each filter in

the filter set is triangular starting at the first point, reaches its peak at the second point, then returns to zero at the 3rd point. The second filter will start at the 2nd point, reach its max at the 3rd, then be zero at the 4th etc. At this stage, mel filter bank energies is obtained by multiplying power spectrum of the signal with mel filter bank. After this stage, take the log of each of the energies from the above step and get log filter bank energies [73]. Figure 3.3 describes a mel filter bank containing 20 filters. This filter bank starts at 0Hz and ends at 7000Hz.

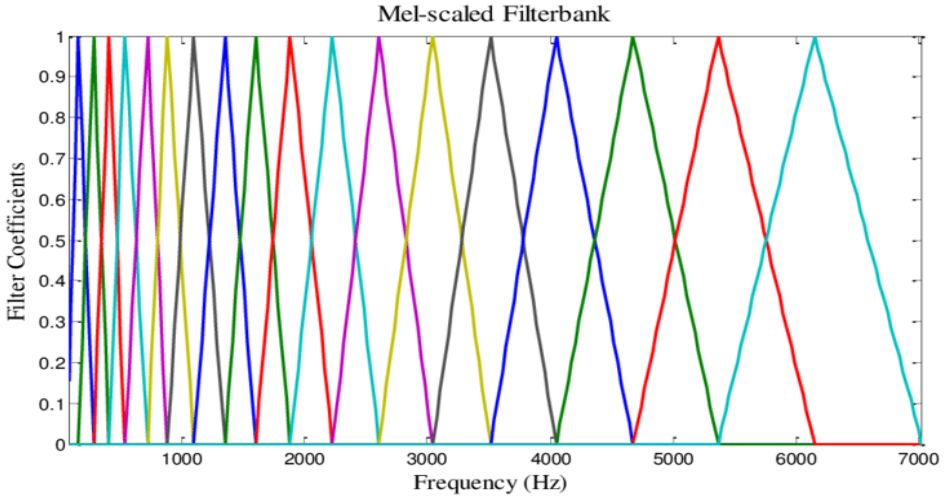

Figure 3.3. Mel Filter Bank

Source: [73]

3.2.6. Discrete Cosine Transform

Discrete Cosine Transform (DCT) is applied on the log energy E_k obtained from the triangular band pass filters to have L mel-scale cepstral coefficients.

$$C_m = \sum_{k=1}^{N} \cos[m*(k-0.5)*\pi/N] * E_k, \; m=1,2\ldots L \qquad \text{Equation 3.4}$$

Where, C_m = coefficient values, N = number of triangular band pass filters, L = number of mel-scale cepstral coefficient, E_k = log energy. DCT transforms the frequency domain into a time domain. Typically, for ASR, the resulting cepstral coefficients 2-13 are retained and the rest are discarded. The resulting features (12 numbers for each frame) are called MFCCs [73].

3.3. Artificial Neural Networks

Artificial Neural Networks (ANNs) have been around since the 1940s but never worked efficiently, but recently they have become extremely powerful and are

one of the most popular machine learning models because of its results which no other model can come close to. Neural networks are very powerful when using massive datasets. An ANN is an information processing model that is inspired by the way biological nervous systems, such as the brain, process information. In simpler terms it is a simple mathematical model of the brain which is used to process nonlinear relationships between inputs and outputs in parallel like a human brain does every second [74].

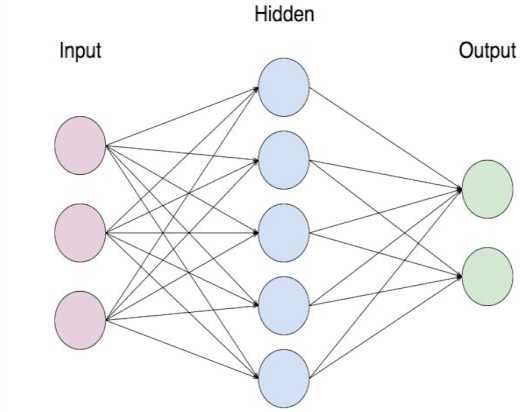

Figure 3.4. Simple Example of a Neural Network
Source: [74]

Information flows through a neural network in two different ways. When the model is learning (being trained) or operating normally (after being trained either being used or tested), patterns of information from the dataset are being fed into the network via the input neurons, which trigger the layers of hidden neurons, and these in turn arrive at the output neurons. This is called a feedforward network. Not all neurons “fire” all the time. Each neuron receives inputs from the neurons to its left, and the inputs are multiplied by the weights of the connections they travel along. Every neuron adds up all the inputs it receives in this way and (this is the simplest neural network) if the sum is more than a certain threshold value, the neuron “fires” and triggers the neurons it’s connected to (the neurons on its right). Feedforward neural network is considered for the experimentation. The feedforward neural network was the first and simplest type of artificial neural network devised. In this network, the information moves in only one direction, forward, from the input nodes, through the hidden nodes (if any) and to the output nodes. There are no cycles or loops in the network. The number of hidden neurons are set equal to the the mean of the neurons

in the input and output layers. From the analysis activation function is set to ’sigmoid’ [75].

Deep feedforward networks or also known multilayer perceptrons are the foundation of most deep learning models. Networks like convolutional neural network (CNNs) and recurrent neural network (RNNs) are some special cases of feedforward networks. These networks are mostly used for supervised machine learning tasks where the target function is already known. The main goal of a feedforward network is to approximate some function f*. For example, a regression function y = f *(x) maps an input x to a value y. A feedforward network defines a mapping y = f (x; θ) and learns the value of the parameters θ that result in the best function approximation. The reason these networks are called feedforward is that **t**he flow of information takes place in the forward direction, as x is used to calculate some intermediate function in the hidden layer which in turn is used to calculate y. In this, if feedback is added from the last hidden layer to the first hidden layer it would represent a recurrent neural network.

For designing any feed-forward neural network there are some things that will be needed to decide, most of the networks require some ingredients, some of which are the same for designing machine learning algorithms.

3.3.1. Optimizer

An Optimizer or optimization algorithm is used to minimize the cost function, this updates the values of the weights and biases after every training cycle or epoch until the cost function reaches the global optimum. Optimization algorithms are of two types: First Order Optimization Algorithms and Second Order Optimization Algorithms.

First order optimization algorithms minimize or maximize a cost function using its gradient values with respect to the parameters. The first order derivative tells whether the function is decreasing or increasing at a particular point, in short, it gives the line which is tangent to the surface.

Second order optimization algorithms use second order derivatives to minimize the cost function and are also called Hessian. Since the second derivative is costly to compute, the second order is not used much. The second order derivative tells whether the first derivative is increasing or decreasing which hints at the

function’s curvature. Second Order Derivative provide with a quadratic surface which touches the curvature of the Error Surface.

3.3.2. Cost Function

Cost function at any point of training shows the difference between approximations made by the model and the actual target value trying to reach and is always single-valued, as its job is to evaluate how a network as a whole is. Just like machine learning algorithms, feedforward networks are also trained using gradients based learning, in such learning method an algorithm like stochastic gradient descent is used to minimize the cost function.

The whole training process is heavily dependent on the choice of cost function; the choice of the cost function is more or less the same for other parametric models. A cost function is mostly of form C(W, B, Sr, Er) where W is the weights of the neural network, B is the biases of the network, Sr is the input of a single training sample, and Er is the desired output of that training sample. Some cost functions are: quadratic cost function, cross-entropy cost function, exponential cost function and Hellinger distance cost function.

3.3.3. Output Units

Output units are those units which are present in the output layer, and their job is to give us the desired output or prediction to finish the task that the neural network must perform. Choice of the output units is tightly coupled with the choice of the cost function. Any unit which can be used as a hidden unit in the neural network can also be used as the output unit. Choice of output units are: linear units, sigmoid units, softmax units, hidden units and rectified linear units [76].

3.3.3.1. Linear units

Simplest kind of output units are linear output units which are used for Gaussian output distributions, and these units are based on an affine transformation which offers non linearity to the output layer. For linear layers, maximizing the log-likelihood is equivalent to minimizing the mean squared errors, the maximum likelihood makes it easier to lean the covariance of the Gaussian distribution. The advantage of these linear units is that they do not saturate, ie their gradient always

remains constant and never approaches zero, there these units pose no difficulty for gradient-based optimization algorithms [76]. Given h features, a layer of linear outputs produces a vector given in the Equation 3.6.

$$y=W^T h+b \qquad \text{Equation 3.5}$$

For linear layers maximizing the log-likelihood is equivalent to minimizing the mean squared errors, the maximum likelihood makes it easier to lean the covariance of the Gaussian distribution. The advantage of these linear units is that they do not saturate, ie their gradient always remains constant and never approaches zero, there these units pose no difficulty for gradient-based optimization algorithms [76].

3.3.3.2. Sigmoid units

$$S(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{e^x+1} \qquad \text{Equation 3.6}$$

For solving a binary classification problem, combining sigmoid output units is maximum likelihood. A Sigmoid output unit has 2 components one is it uses a linear layer to compute z = w*h+b and then it uses activation function to convert z into probability [76] . When other loss functions are used, such as mean squared error, the loss can saturate anytime, i.e. the gradients can shrink too small to be useful for learning. For this reason, maximum likelihood is preferred [76]. Sigmoid unit function is shown in Equation 3.7.

3.3.3.3. Softmax units

In mathematics, the softmax function, also known as softargmax or normalized exponential function, is a function that takes as input a vector of *K* real numbers, and normalizes it into a probability distribution consisting of *K* probabilities proportional to the exponentials of the input numbers. The softmax function is often used in the final layer of a neural network-based classifier. Softmax units are mainly used for Multitudinous output distributions. These units are used for a probability distribution over a discrete variable with n possible values, and hence this can also be seen as a

generalization of the sigmoid function which represents the probability distribution over a binary variable. Softmax function is defined by:

$$\text{softmax}\ (z)_i = \frac{\exp(z_i)}{\sum_j \exp(z_j)} \qquad \text{Equation 3.7}$$

Like sigmoid function, softmax function can also saturate, ie the gradients can shrink too small to be useful for learning. In the case of softmax, as it has multiple output units, the units can only saturate when the differences between input values become extreme. These units are governed the by winner take all principle as the total probability is always 1 and cannot exceed, its value of one out gets closer to 1 it is sure the value of outputs from other output units will near to 0 [76].

3.3.3.4. Hidden units

Selecting the type of hidden unit is also active research and no particular unit can guarantee that it will outperform all others in every problem but we still have some units which are default choices in the beginning, For example, rectified linear units or popularly known as Relu is mostly used, this is due to intuitive reasons rather than experimental, in reality, It is usually impossible to predict in advance which will work best. Selecting a hidden unit involves trial and error, intuiting that a kind of hidden unit may work well and then testing [76].

3.3.3.5. Rectified linear units

The rectified linear activation function is a piecewise linear function that will output the input directly if is positive, otherwise, it will output zero. It has become the default activation function for many types of neural networks because a model that uses it is easier to train and often achieves better performance. These functions use the activation function defined by g(z):

$$g\ (z) = \max\{0, z\} \qquad \text{Equation 3.8}$$

Relus are easy to optimize as they are similar to linear units, the only difference between them is that the output 0 for half of their domains. The reason they are so famous is that they always have a constant large gradient whenever the unit is active.

the gradient the direction is far more useful for learning than it would be with activation functions that introduce second-order effects. ReLU has a drawback that they cannot learn via gradient-based methods for which their activation is zero [76].

3.4. Random Forest

Random Forest (RF), like its name implies, consists of a large number of individual decision trees that operate as an ensemble. Ensemble is a data mining technique composed of number of individual classifiers to classify the data to generate new instances of data. The ensemble learning method is used for classification and regression of data. RF is the most popular ensemble technique that can use both for classification and regression kinds of problems. Each individual tree in the RF spits out a class prediction and the class with the most votes becomes our model's prediction. RF uses decision trees as base classifier.

3.4.1. Basic Decision Tree Concept

Decision trees are a non-parametric supervised learning method used for classification and regression. Decision trees learn from data to approximate a sine curve with a set of if-then-else decision rules. The deeper the tree, the more complex the decision rules and the fitter the model.

Decision tree builds classification or regression models in the form of a tree structure. It breaks down a data set into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes. A decision node has two or more branches. Leaf node represents a classification or decision. The topmost decision node in a tree which corresponds to the best predictor called root node. Decision trees can handle both categorical and numerical data [77].

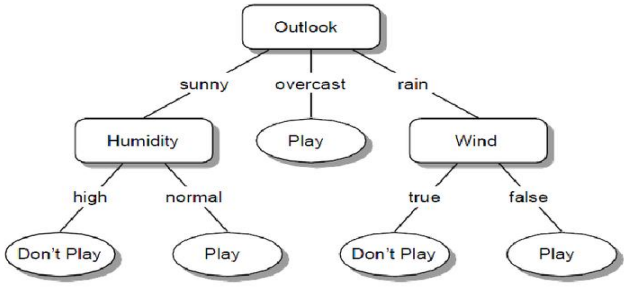

Figure 3.5. Simple Decision Tree

Source: [77]

3.4.2. How RF Classifier Works for Classification

An ensemble consists of number of trained classifiers whose predictors are combined to classify new instances. RF is a classifier consisting of collection of tree-structured classifiers where independent random vectors are distributed identically and each tree cast a unit vote for the most popular class at input x. A random vector is generated which is independent of the past random vectors with same distribution and a tree is generated by using the training test [77]. The pseudocode for random forest algorithm can split into two stages.

1. Random forest creation pseudocode.
2. Pseudocode to perform prediction from the created random forest classifier.

3.4.2.1. RF creation pseudocode

1. Randomly select “k” features from total “m” features, Where k << m.
2. Among the “k” features, calculate the node “d” using the best split point.
3. Split the node into daughter nodes using the best split.
4. Repeat 1 to 3 steps until “l” number of nodes has been reached.
5. Build forest by repeating steps 1 to 4 for “n” number times to create “n” number of trees.

The beginning of RF algorithm starts with randomly selecting “k” features out of total “m” features. In the next stage, the randomly selected “k” features are used to find the root node by using the best split approach. The next stage, the daughter nodes will be calculated using the same best split approach. Will the first 3 stages until forming the tree with a root node and having the target as the leaf node. Finally, repeating 1 to 4 stages creates “n” randomly created trees. This randomly created trees forms the random forest.

3.4.2.2. RF prediction pseudocode

There are two stages in RF algorithm, one is random forest creation, the other is to make a prediction from the RF classifier created in the first stage. To perform prediction, RF algorithm uses the below pseudocode.

1. Take the test features and use the rules of each randomly created decision tree to predict the outcome and stores the predicted outcome (target).
2. Calculate the votes for each predicted target.

3. Consider the high voted predicted target as the final prediction from the random forest algorithm.

To perform prediction, using the trained random forest algorithm the test features need to pass through the rules of each randomly created trees. Suppose let's say 100 random decision trees are formed to form the random forest. Each random forest will predict different target (outcome) for the same test feature. Then by considering each predicted target votes will be calculated. Suppose the 100 random decision trees are prediction some 3 unique targets x, y, z then the votes of x is nothing but out of 100 random decision tree how many trees prediction is x. Likewise for other 2 targets (y, z). If x is getting high votes. Let's say out of 100 random decision tree 60 trees are predicting the target will be x. Then the final random forest returns the x as the predicted target. This concept of voting is known as majority voting [78].

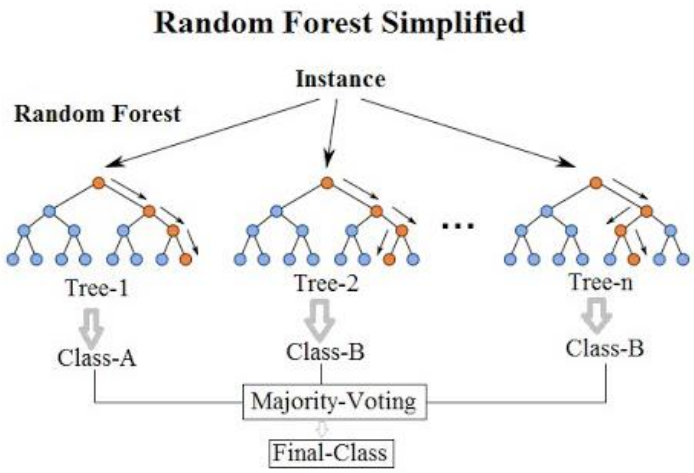

Figure 3.6. Random Forest Simple Explanation

Source: [78]

3.5. Logistic Regression

Logistic Regression (LR) called the logistic model or logit model, analyzes the relationship between multiple independent variables and a categorical dependent variable, and estimates the probability of occurrence of an event such as pass/fail, win/lose, alive/dead or healthy/sick by fitting data to a logistic curve. There are two models of LR, binary LR and multinomial LR. Binary logistic regression is typically

used when the dependent variable is dichotomous and the independent variables are either continuous or categorical. When the dependent variable is not dichotomous and is comprised of more than two categories, a multinomial logistic regression can be employed.

As an illustrative example, consideration of how coronary heart disease (CHD) can be predicted by the level of serum cholesterol. The probability of CHD increases with the serum cholesterol level. However, the relationship between CHD and serum cholesterol is nonlinear and the probability of CHD changes very little at the low or high extremes of serum cholesterol. This pattern is typical because probabilities cannot lie outside the range from 0 to 1. The relationship can be described as an ‘S’-shaped curve. The logistic model is popular because the logistic function, on which the logistic regression model is based, provides estimates in the range 0 to 1 and an appealing S-shaped description of the combined effect of several risk factors on the risk for an event.

Mathematically, a binary logistic model has a dependent variable with two possible values, such as pass/fail which is represented by an indicator variable, where the two values are labeled "0" and "1". In the logistic model, the log-odds (the logarithm of the odds) for the value labeled "1" is a linear combination of one or more independent variables ("predictors"); the independent variables can each be a binary variable (two classes, coded by an indicator variable) or a continuous variable (any real value). The corresponding probability of the value labeled "1" can vary between 0 (certainly the value "0") and 1 (certainly the value "1"), hence the labeling; the function that converts log-odds to probability is the logistic function. The unit of measurement for the log-odds scale is called a logit, from logistic unit, hence the alternative names. Analogous models with a different sigmoid function instead of the logistic function can also be used, such as the probit model; the defining characteristic of the logistic model is that increasing one of the independent variables multiplicatively scales the odds of the given outcome at a constant rate, with each independent variable having its own parameter; for a binary dependent variable this generalizes the odds ratio [79].

3.5.1. Odds

Odds of an event are the ratio of the probability that an event will occur to the probability that it will not occur. If the probability of an event occurring is p, the

probability of the event not occurring is (1-p). Then the corresponding odds are a value given by Equation 3.10. Since logistic regression calculates the probability of an event occurring over the probability of an event not occurring, the impact of independent variables is usually explained in terms of odds.

$$\text{odds of } \{\text{Event}\} = \frac{p}{1-p} \qquad \text{Equation 3.9}$$

With logistic regression the mean of the response variable p in terms of an explanatory variable x is modeled relating p and x through the equation $p=\alpha+\beta x$. Unfortunately, this is not a good model because extreme values of x will give values of $\alpha+\beta x$ that does not fall between 0 and 1. The logistic regression solution to this problem is to transform the odds using the natural logarithm. With logistic regression the natural log odds can be modeled as a linear function of the explanatory variable

$$\text{logit}\,(y) = \ln(\text{odds}) = \ln\left(\frac{p}{1-p}\right) = \alpha+\beta x \qquad \text{Equation 3.10}$$

where p is the probability of interested outcome and x is the explanatory variable. The parameters of the logistic regression are α and β. This is the simple logistic model [80]. Taking the antilog of equation (1) on both sides, one can derive an equation for the prediction of the probability of the occurrence of interested outcome as

$$\begin{aligned} p &= P(Y = \text{interested outcome} / X = x, \text{a specific value}) \\ &= \frac{e^{\alpha+\beta x}}{1+e^{\alpha+\beta x}} = \frac{1}{1+e^{-(\alpha+\beta x)}} \end{aligned} \qquad \text{Equation 3.11}$$

3.5.2. Odds Ratio

An odds ratio (OR) is a measure of association between an exposure and an outcome. The OR represents the odds that an outcome will occur given a particular exposure , compared to the odds of the outcome occurring in the absence of that exposure. OR is a comparative measure of two odds relative to different events. The OR is defined as the ratio of the odds of A in the presence of B and the odds of A in

the absence of B. For two events A and B, the corresponding odds of A occurring relative to B occurring is

$$\text{odds ratio } \{A \text{ vs } B\} = \frac{\text{odds } \{A\}}{\text{odds } \{B\}} = \frac{P_A/(1-P_A)}{P_B/(1-P_B)} \qquad \text{Equation 3.12}$$

When a logistic regression is calculated, the regression coefficient (b_1) is the estimated increase in the logged odds of the outcome per unit increase in the value of the independent variable. In other words, the exponential function of the regression coefficient (e^{b1}) is the OR associated with a one unit increase in the independent variable. The OR can also be used to determine whether a particular exposure is a risk factor for a particular outcome, and to compare the magnitude of various risk factors for that outcome. OR=1 indicates exposure does not affect odds of outcome. OR>1 indicates exposure associated with higher odds of outcome. OR<1 indicates exposure associated with lower odds of outcome. For example, the variable smoking is coded as 0 (=no smoking) and 1 (=smoking), and the odds ratio for this variable is 3.2. Then, the odds for a positive outcome in smoking cases are 3.2 times higher than in non-smoking cases. LR is one way to generalize the OR beyond two binary variables [80]. Suppose we have a binary response variable Y and a binary predictor variable X, and in addition we have other predictor variables $Z_1 \ldots Z_k$ that may or may not be binary. If multiple LR is used to regress Y on X, $Z_1 \ldots Z_k$, then the estimated coefficient β_x for X is related to a conditional OR. Specifically, at the population level,

$$e^{\beta_x} = \frac{P(Y=1|X=1,Z_{1\ldots},Z_k)/P(Y=0|X=1,Z_{1\ldots},Z_k)}{P(Y=1|X=0,Z_{1\ldots},Z_k)/P(Y=0|X=0,Z_{1\ldots},Z_k)} \qquad \text{Equation 3.13}$$

$e^{\beta x}$ is an estimate of this conditional odds ratio. The interpretation of $e^{\beta x}$ is as an estimate of the OR between Y and X when the values of Z1,..., Z_k are held fixed [80].

3.5.3. The Logistic Curve

Logistic regression is a method for fitting a regression curve, y=f(x), when y consists of binary coded (0, 1--failure, success) data. When the response is a binary (dichotomous) variable and x is numerical, logistic regression fits a logistic curve to the relationship between x and y. Logistic curve is an S-shaped or sigmoid curve, often used to model population growth [81]. A logistic curve starts with slow, linear

growth, followed by exponential growth, which then slows again to a stable rate. A simple logistic function is defined by the formula

$$y = \frac{e^x}{1 + e^x} = \frac{1}{1 + e^{-x}} \qquad \text{Equation 3.14}$$

which is graphed in Figure 3.7.

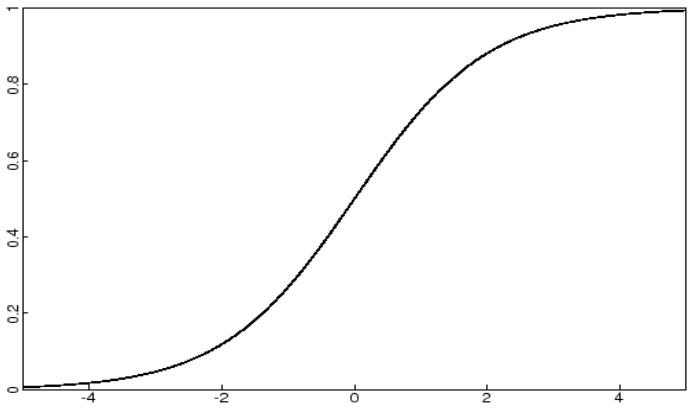

Figure 3.7. Graph of Logistic Curve where α=0 and β=1

Source: [81]

3.6. Support Vector Machines

In machine learning, support-vector machines (SVMs, also support-vector networks) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on the side of the gap on which they fall. In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.

When data are unlabeled, supervised learning is not possible, and an unsupervised learning approach is required, which attempts to find natural clustering of the data to groups, and then map new data to these formed groups. The support-

vector clustering[82]algorithm, created by Hava Siegelmann and Vladimir Vapnik, applies the statistics of support vectors, developed in the support vector machines algorithm, to categorize unlabeled data, and is one of the most widely used clustering algorithms in industrial applications [82].

3.6.1. Linear Classification

Before considering N-dimensional hyperplanes, let's look at a simple 2-dimensional example. Assume a classification is performed on data that has a categorical target variable with two categories. It can also be assumed that assumed that there are two predictor variables with continuous values. The data points are plotted using the value of one predictor on the X axis and the other on the Y axis as shown below. One category of the target variable is represented by rectangles while the other category is represented by ovals. In this idealized example, the cases with one category are in the lower left corner and the cases with the other category are in the upper right corner; the cases are completely separated. The SVM analysis attempts to find a 1-dimensional hyperplane (i.e. a line) that separates the cases based on their target categories. There are an infinite number of possible lines. The question is which line is better, and how to define the optimal line. The dashed lines drawn parallel to the separating line mark the distance between the dividing line and the closest vectors to the line. The distance between the dashed lines is called the margin [83]. The vectors (points) that constrain the width of the margin are the support vectors. The following figure illustrates this.

An SVM analysis finds the line (or, in general, hyperplane) that is oriented so that the margin between the support vectors is maximized. Intuitively an SVM projects data points into a higher dimensional space, specified by a kernel function, and computes a maximum-margin hyperplane decision surface that separates the two classes. In the figure above, the line in the right panel is superior to the line in the left panel. If all analyses consisted of two-category target variables with two predictor variables, and the cluster of points could be divided by a straight line, life would be easy. Unfortunately, this is not generally the case, so SVM must deal with (a) more than two predictor variables, (b) separating the points with non-linear curves, (c) handling the cases where clusters cannot be completely separated, and (d) handling classifications with more than two categories [84].

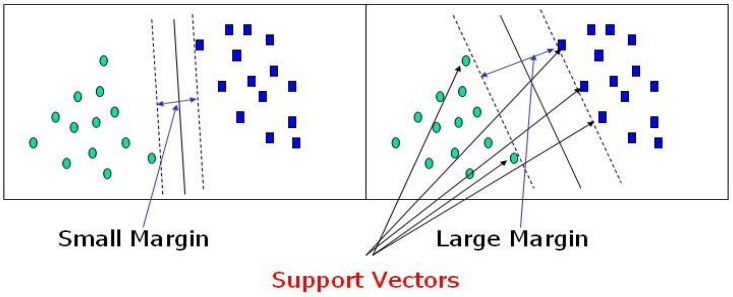

Figure 3.8. Two Dimensional Mapping

Source: [83]

3.6.2. Non Linear Classification

Non linear kernel function fits the maximum-margin hyperplane in a transformed feature space. The simplest way to divide two groups is with a straight line, flat plane or an N-dimensional hyperplane. But if the points are separated by a nonlinear region, a nonlinear dividing line is needed. Rather than fitting nonlinear curves to the data, SVM handles this by using a kernel function to map the data into a different space where a hyperplane can be used to do the separation. The kernel function may transform the data into a higher dimensional space to make it possible to perform the separation [83].

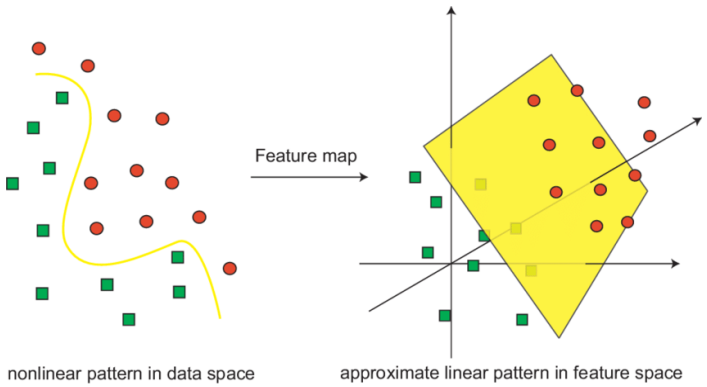

Figure 3.9. Higher Dimensional Mapping

Source: [83]

3.6.3. Kernels in SVM

The kernel function may transform the data into a higher dimensional space to make it possible to perform the separation. Kernel functions are a class of algorithms for pattern analysis or recognition, whose best known element is SVM. Training vectors xi are mapped into a higher (may be infinite) dimensional space by the

function Φ. Then SVM finds a linear separating hyperplane with the maximal margin in this higher dimension space .C > 0 is the penalty parameter of the error term. Furthermore, K(xi , xj) ≡ Φ(xi)T Φ(xj) is called the kernel function. There are many kernel functions in SVM, so how to select a good kernel function is also a research issue [85]. The choice of a Kernel depends on the problem at hand because it depends on what kind of model needs to build. A Polynomial kernel, for example, allows to model feature conjunctions up to the order of the polynomial. Radial basis functions allows to pick out circles in contrast with the Linear kernel, which allows only to pick out lines (or hyperplanes). Many kernel mapping functions can be used probably an infinite number. Normalized Polynomial, RBF, linear, Sigmoid, GussianRBF and String Kernels etc can be used based on application requirement. But a few kernel functions have been found to work well in for a wide variety of applications. The default and recommended kernel function is the Radial Basis Function (RBF) [84].

3.7. Naive Bayes Classifier

In machine learning, naive Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naive) independence assumptions between the features. Naïve Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem. Maximum-likelihood training can be done by evaluating a closed-form expression, which takes linear time, rather than by expensive iterative approximation as used for many other types of classifiers.

In the statistics and computer science literature, naive Bayes models are known under a variety of names, including simple Bayes and independence Bayes. All these names reference the use of Bayes' theorem in the classifier's decision rule, but naïve Bayes is not (necessarily) a Bayesian method [85]. The Naive Bayes algorithm affords fast, highly scalable model building and scoring. It scales linearly with the number of predictors and rows. Naive Bayes can be used for both binary and multiclass classification problems [85].

3.7.1. Bayes’ Theorem

Bayes Theorem is named after Thomas Bayes (1701–1761), who first introduced Bayes’ theorem which then later get developed further by Pierre Simon

Laplace, who published the modern equation of the Bayes Theorem in 1812. In general Bayes Theorem describes the probability of an event, based on prior knowledge of conditions be related of conditions to the event. So it basically fits perfectly for machine learning, because that is exactly what machine learning does: making predictions for the future based on prior experience [86]. Mathematically the Bayes theorem can be written as following:

$$P(A|B)=\frac{P(B|A)P(A)}{P(B)} \qquad \text{Equation 3.15}$$

Let's break the equation down:

1. A and B are events,
2. P(A) and P(B) (P(B) not 0) are the probabilities of the event independent from each other,
3. P(A|B) is the probability A under the condition B,
4. Equivalent with P(B|A), it is the probability of observing event B given that event A is true.

3.7.2. Gaussian Naive Bayes

A Gaussian Naive Bayes classifier (GNB) is a probabilistic machine learning model that's used for classification task. The crux of the classifier is based on the Bayes theorem. A GNB algorithm is a special type of NB algorithm. It's specifically used when the features have continuous values. It's also assumed that all the features are following a gaussian distribution i.e, normal distribution. Primarily NB is a linear classifier, which is a supervised machine learning method and works as a probabilistic classifier as well. GNB is the simplest and the most popular one. When handling real-time data with continuous distribution, NB classifier considers that the big data is generated through a Gaussian process with normal distribution. In general the Naive Bayes classifier is not linear, but if the likelihood factors p(xi|c) are from exponential families, the Naive Bayes classifier corresponds to a linear classifier in a particular feature space. GNB is an algorithm having a probabilistic approach. It involves prior and posterior probability calculation of the classes in the dataset and the test data divided into a class respectively. Mathematical Formula for Prior Probability can be seen in Equation 3.17.

$$\text{Prior Probability (c)} = \frac{\text{No.of instances of class c}}{\text{Total No. of instances in the dataset}} \quad \text{Equation 3.16}$$

Prior probabilities of all the classes are calculated using the same formula.

$$\text{Posterior Probability}(x|c) = P(x_1|c) * P(x_2|c) * \ldots * P(x_n|c) \quad \text{Equation 3.17}$$

Mathematical Formula for Posterior Probability of a test data x given class c which is the product of the conditional probabilities of all the features of the test data given class c in Equation 3.18.

$$P(x_i|c) = \frac{1}{\sqrt{2*\pi*\text{sigma}^2_{xi,c}}} * \exp\left(-\frac{(x_i - \text{mean}_{xi,c})^2}{2*\text{sigma}^2_{xi,c}}\right) \quad \text{Equation 3.18}$$

Mathematical Expression for obtaining the conditional probabilities of a test feature given a class and x_i is a test data feature, c is a class and sigma² is the associated Sample Variance in Equation 3.19. Finally, the conditional probability of each class given an instance (test instance) is calculated using Bayes Theorem.

Mathematical Expression of conditional probability of class ci given test data x. Equation 3.20 is repeated for all the classes and the class showing the highest probability is ultimately declared the predicted result [87].

$$P(c_i|x) = \frac{P(x|c_i) * P(c_i)}{\sum_j P(x|c_j) * P(c_j)} \quad \text{Equation 3.19}$$

3.8. Performance Metrics

Performance metrics are used to measure the behavior, activities, and performance of a business. This should be in the form of data that measures required data within a range, allowing a basis to be formed supporting the achievement of overall business goals. The performance metrics chosen to evaluate machine learning models are very important. Choice of metrics influences how the performance of machine learning algorithms is measured and compared. Metrics used for evaluation of children gender classification are precision, recall, F1-score and support.

3.8.1. Confusion Matrix

The confusion matrix is a two by two table that contains four outcomes produced by a binary classifier. Various measures, such as error-rate, accuracy, specificity, sensitivity, and precision, are derived from the confusion matrix. Moreover, several advanced measures, such as ROC and precision-recall, are based on them. The confusion matrix table has two dimensions ("Actual" and "Predicted"), and sets of "classes" in both dimensions. Actual classifications are columns and predicted ones are rows. The Confusion matrix in itself is not a performance measure as such, but almost all of the performance metrics are based on confusion matrix and the numbers inside it.

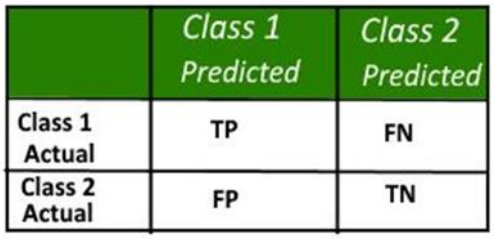

Figure 3.10. Sample Confusion Matrix

1. True Positives (TP): True positives are the cases when the actual class of the data point was 1(True) and the predicted is also 1(True),
2. True Negatives (TN): True negatives are the cases when the actual class of the data point was 0(False) and the predicted is also 0(False),
3. False Positives (FP): False positives are the cases when the actual class of the data point was 0(False) and the predicted is 1(True). False is because the model has predicted incorrectly and positive because the class predicted was a positive one,
4. False Negatives (FN): False negatives are the cases when the actual class of the data point was 1(True) and the predicted is 0(False). False is because the model has predicted incorrectly and negative because the class predicted was a negative one.

Accuracy in classification problems is the number of correct predictions made by the model over all kinds predictions made. Classification accuracy can be calculated based on confusion matrix as shown in Equation 3.21.

$$\text{Accuracy}= \frac{TP+TN}{TP+FP+FN+TN} \qquad \text{Equation 3.20}$$

3.8.2. Precision

Precision is the amount of positive predictions that were correct. Precision is the number of correct positive results divided by the number of positive results predicted by the classifier. The best precision is 1.0, whereas the worst is 0. Accuracy refers to the closeness of a measured value to a standard or known value. Precision refers to the closeness of two or more measurements to each other.

$$\text{Precision}= \frac{TP}{TP+FP} \qquad \text{Equation 3.21}$$

3.8.3. Recall

Recall is the opposite of precision and it measures false negatives against true positives. It refers to the percentage of total relevant results correctly classified by the algorithm. In other words, recall is the number of correct positive results divided by the number of all relevant samples (all samples that should have been identified as positive).

$$\text{Recall}= \frac{TP}{TP+FN} \qquad \text{Equation 3.22}$$

3.8.4. F1 Score

F1 score is the Harmonic Mean between precision and recall. The range for F1 score is [0, 1]. It shows how precise the classifier is (how many instances it classifies correctly), as well as how robust it is (it does not miss a significant number of instances).

$$F1=2* \frac{1}{\frac{1}{\text{Precision}}+\frac{1}{\text{Recall}}} \qquad \text{Equation 3.23}$$

The F1score is often used in the field of information retrieval for measuring search, document classification, and query classification performance. The F-score is also used in machine learning and used in the natural language processing literature .The greater the F1 score, the better the performance of the model is. Mathematically, it can be expressed as Equation 3.24.

3.9. Summary

In this chapter, most widely used feature extraction algorithm, MFCC is described. Work flow of MFCC features extraction is discussed in details. For children gender classification, five classification methods namely RF, ANN, SVM, LR and GNB are used. Background theory, working strategy, usage and advantages and limitations of each learning algorithm are discussed. In the next chapter, system design and detailed work flow of the proposed children gender classification system will be discussed.