Spring 2018

SPEECH EMOTION DETECTION USING MACHINE LEARNING TECHNIQUES

Neethu Sundarprasad
San Jose State University

SPEECH EMOTION DETECTION USING MACHINE LEARNING TECHNIQUES

A Thesis

Presented to

The Faculty of the Department of Computer Science

San José State University

In Partial Fulfillment

Of the Requirements for the Degree

Master of Science

by

Neethu Sundarprasad

May, 2018

The Designated Thesis Committee Approves the Thesis Titled

SPEECH EMOTION DETECTION

by

Neethu Sundarprasad

APPROVED FOR THE DEPARTMENTS OF COMPUTER SCIENCE

SAN JOSÉ STATE UNIVERSITY

May 2018

Dr. Robert Chun	Department of Computer Science
Dr. Sami Khuri	Department of Computer Science
Senthil Arumugam	Cisco Systems

ABSTRACT

Communication is the key to express one's thoughts and ideas clearly. Amongst all forms of communication, speech is the most preferred and powerful form of communications in human. The era of the Internet of Things (IoT) is rapidly advancing in bringing more intelligent systems available for everyday use. These applications range from simple wearables and widgets to complex self-driving vehicles and automated systems employed in various fields. Intelligent applications are interactive and require minimum user effort to function, and mostly function on voice-based input. This creates the necessity for these computer applications to completely comprehend human speech. A speech percept can reveal information about the speaker including gender, age, language, and emotion. Several existing speech recognition systems used in IoT applications are integrated with an emotion detection system in order to analyze the emotional state of the speaker. The performance of the emotion detection system can greatly influence the overall performance of the IoT application in many ways and can provide many advantages over the functionalities of these applications. This research presents a speech emotion detection system with improvements over an existing system in terms of data, feature selection, and methodology that aims at classifying speech percepts based on emotions, more accurately.

ACKNOWLEDGMENTS

I would like to express my gratitude towards my advisor Dr. Robert Chun for guiding me through the extensive research process that became this Master's thesis. Furthermore, I would like to thank Dr. Natalia Khuri for exposing me to this topic. I would also like to thank my committee members Dr. Sami Khuri and Senthil Arumugam for their suggestions and support. Lastly, I would like to thank my family and friends for their encouragement and belief in me.

TABLE OF CONTENTS

CHAPTER

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

For several years now, the growth in the field of Artificial Intelligence (AI) has been accelerated. AI, which was once a subject understood by computer scientists only, has now reached the house of a common man in the form of intelligent systems. The advancements of AI have engendered to several technologies involving Human-Computer Interaction (HCI) [1]. Aiming to develop and improve HCI methods is of paramount importance because HCI is the front-end of AI which millions of users experience. Some of the existing HCI methods involve communication through touch, movement, hand gestures, voice and facial gestures [1]. Among the different methods, the voice-based intelligent devices are gaining popularity in a wide range of applications. In a voice-based system, a computer agent is required to completely comprehend the human's speech percept in order to accurately pick up the commands given to it. This field of study is termed as Speech Processing and consists of three components:

- Speaker Identification
- Speech Recognition
- Speech Emotion Detection

Speech Emotion Detection is challenging to implement among the other components due to its complexity. Furthermore, the definition of an intelligent computer system requires the system to mimic human behavior. A striking nature unique to humans is the ability to alter conversations based on the emotional state of the speaker and the listener. Speech emotion detection can be built as a classification problem solved using several machine learning algorithms. This project discusses in detail the various methods and experiments carried out as part of implementing a Speech Emotion Detection system.

1.1 IMPORTANCE

Communication is the key to express oneself. Humans use most part of their body and voice to effectively communicate. Hand gestures, body language, and the tone and temperament are all collectively used to express one's feeling. Though the verbal part of the communication varies by languages practiced across the globe, the non-verbal part of communication is the expression of feeling which is most likely common among all. Therefore, any advanced technology developed to produce a social environment experience also covers understanding emotional context in speech.

Improvements in the field of emotion detection positively impact a multitude of applications. Some of the research areas that benefit from automating the emotion detection technique include psychology, psychiatry, and neuroscience. These departments of cognitive sciences rely on human interaction, where the subject of study is put through a series of questions and situations, and based on their reactions and responses, several inferences are made. A potential drawback occurs as few people are classified introverts and hesitate to communicate. Therefore, replacing the traditional procedures with a computer-based detection system can benefit the study. Similarly, the practical applications of the speech-based emotion detection are many. Smart home appliances and assistants (Examples: Amazon Alexa [2] and Google Home [3]) are ubiquitous these days. Additionally, customer care-based call centers often have an automated voice control which might not please most of their angry customers. Redirecting such calls to a human attendant will improve the service. Other applications include eLearning, online tutoring, investigation, personal assistant (Example: Apple Siri [4] and Samsung S Voice [5]) etc. A very recent application could be seen in self-driving cars. These vehicles heavily depend on voice-based controlling. An unlikely situation, such as anxiety, can cause the passenger to utter unclear sentences. In these situations, understanding the emotional content expressed becomes of prime importance.

1.2 MOTIVATION

Identifying the emotion expressed in a speech percept has several use cases in the modern day applications. Human-Computer Interaction (HCI) is a field of research that studies interactive applications between humans and computers [1]. For an effective HCI application, it is necessary for the computer system to understand more than just words. On the other hand, the field of Internet of Things (IoT) is rapidly growing. Many real word IoT applications that are used on a daily basis such as Amazon Alexa, Google Home and Mycroft function on voice-based inputs. The role of voice in IoT applications is pivotal. The study in a recent article foresees that by 2022, about 12% of all IoT applications would fully function based on voice commands only [6]. These voice interactions could be mono-directional or bi-directional, and in both cases, it is highly important to comprehend the speech signal. Further, there are Artificial Intelligence (AI) and Natural Language Processing (NLP) based applications that use functions of IoT and HCI to create complex systems. Self-driving cars are one such application that controls many of its functions using voice-based commands. Identifying the emotional state of the user comes with a great advantage in this application. Considering emergency situations in which the user may be unable to clearly provide a voice command, the emotion expressed through the user's tone of voice can be used to turn on certain emergency features of the vehicle. A much simpler application of speech emotion detection can be seen in call centers, in which automated voice calls can be efficiently transferred to customer service agents for further discussion. Other applications of using a speech emotion detection system can be found in lie detecting systems, criminal department analysis, and in humanoids.

CHAPTER 2

EXISTING SYSTEMS

2.1 METHODOLOGY

The speech emotion detection system is implemented as a Machine Learning (ML) model. The steps of implementation are comparable to any other ML project, with additional fine-tuning procedures to make the model function better. The flowchart represents a pictorial overview of the process (see Figure 1). The first step is data collection, which is of prime importance. The model being developed will learn from the data provided to it and all the decisions and results that a developed model will produce is guided by the data. The second step, called feature engineering, is a collection of several machine learning tasks that are executed over the collected data. These procedures address the several data representation and data quality issues. The third step is often considered the core of an ML project where an algorithmic based model is developed. This model uses an ML algorithm to learn about the data and train itself to respond to any new data it is exposed to. The final step is to evaluate the functioning of the built model. Very often, developers repeat the steps of developing a model and evaluating it to compare the performance of different algorithms. Comparison results help to choose the appropriate ML algorithm most relevant to the problem.

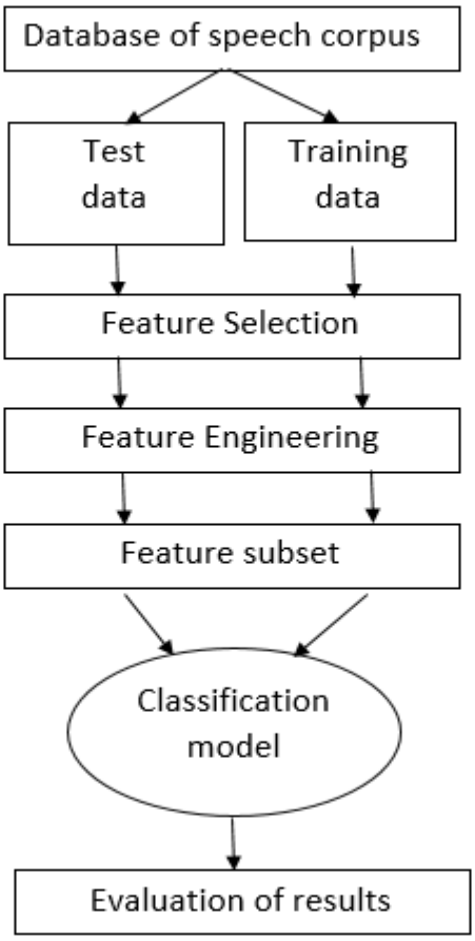

Fig.1 Flow of implementation

2.2 RANKING SVM APPROACH

Cao et al. [7] proposed a system that considered that the emotion expressed by humans are mostly a result of mixed feeling. Therefore, they suggested an improvement over the SVM algorithm that would consider mixed signals and choose the most dominant one. For this purpose, a ranking SVM algorithm was chosen. The ranking SVM takes all predictions from individual binary classification SVM classifiers also called as rankers, and applies it to the final multi-class problem. Using the ranking SVM algorithm, an accuracy of 44.40% was achieved in their system.

2.3 DIMENSIONALITY REDUCTION METHOD

Chen et al. [8] developed a system that had improvements in the pre-processing stage. Two pre-processing techniques, namely Fisher and Principle Component Analysis (PCA), were used in combination with two classifier algorithms, namely SVM and ANN. They carried out four experiments, each with a different combination of pre-processing and the classifier algorithm. The first experiment used Fisher method to select features for a multi-level SVM classifier (Fisher + SVM). The second experiment was to reduce feature dimensionality using Principle Component Analysis (PCA) for the SVM classifier (PCA + SVM). The third experiment used the Fisher technique over the ANN model (Fisher + ANN). Finally, PCA was applied before classification using ANN (PCA + ANN). From these experiments, two important conclusions were made. Firstly, dimensionality reduction improves the performance of the system. Secondly, SVM classifier algorithm classifies better than the ANN algorithm in the case of emotion detection. The winning experiment had an accuracy of 86.50% using Fisher for dimensionality reduction and SVM for classification.

2.4 LPC COEFFICIENT APPROACH

In the Nwe et al. [9] system, a subset of features, similar to the Mel Frequency Cepstral Coefficients (MFCC), was used. They used the Log Frequency Power Coefficients (LFPC) over a Hidden Markov Model (HMM) to classify emotions in speech. Their work is not publically available, as they used a dataset privately available to them. However, they claim that using the LFPC coefficients over the MFFCC coefficients shows a significant improvement in terms of the accuracy of the model. The average classification accuracy in their model is 78% and the best accuracy is even higher 96%.

2.5 EXTENDING THE FEATURE SPACE

Rong et al. [10] proposed an innovative way to improve the accuracy of existing models. Traditionally, computer scientists were using various pre-processing techniques to reduce the number of features. Contrastingly, this new system increased the number of features used for classification. They claimed to have performed classification over a small dataset containing audio percepts in the Chinese language, but do not disclose the features that they used. However, they also mentioned that none of their features are language-dependent. Using a high number of features over an ensemble random forest algorithm (ERFTrees), they achieved an accuracy of 82.54%.

2.6 DOMAIN SPECIFIC CLASSIFICATION

Narayanan [11], in his work, proposes a system that uses a more real-world dataset. For his work, data was collected from a call center and he performed a binary classification with only two distinct emotions, namely happy and angry. The research used numerous features including acoustic, lexical and other language-based features over the KNN algorithm. Moreover, this research was conducted specifically for the call center domain and was evaluated across male and female customers. The accuracy values showed improvements of 40.70% and 36.40% in male and female customers, respectively.

CHAPTER 3

DATASET

Two datasets created in the English language, namely the Toronto Emotional Speech Set (TESS) and the emotional dataset from Knowledge Extraction based on Evolutionary Learning (KEEL), contain a more diverse and realistic audio. The descriptions of the dataset are as follows.

3.1 TORONTO EMOTIONAL SPEECH SET (TESS)

The researchers from the Department of Psychology at the University of Toronto have created a speech emotion based dataset in 2010, in the English language [12]. The database contains 2800 sound files of speech utterances in seven basic emotional categories, namely: Happy, Sad, Angry, Surprise, Fear, Disgust and Neutral. It is an acted recording, where actors from two age groups of Old (64-year-old) and Young (26-year-old) had performed the dictation.

A few qualities of this dataset which makes it good for this project are:

- The size of the dataset is large enough for the model to be trained effectively. The more exposure to data given to a model helps it to perform better.
- All basic emotional categories of data are present. A combination of these emotions can be used for further research like Sarcasm and Depression detection.
- Data is collected from two different age groups which will improve the classification.
- The audio files are mono signals, which ensures an error-free conversion with most of the programming libraries.

3.2 KNOWLEDGE EXTRACTION BASED ON EVOLUTIONARY LEARNING (KEEL)

KEEL is an online dataset repository contributed by machine learning researchers worldwide [13]. The emotion for speech dataset contains 72 features extracted for each of the 593 sound files. The data are labeled across six emotions, namely: Happy, Sad, Angry, Surprise, Fear and Neutral. The repository also offers data to be downloaded in 10 or 5 folds for the purpose of training and testing.

A few qualities of this dataset which makes it good for this project are:

- Data is represented as features directly, which saves conversion time and procedures.
- All basic emotional categories of data are present. A combination of these emotions can be used for further research like Sarcasm and Depression detection.

CHAPTER 4

FEATURE EXTRACTION

4.1 THE PROCESS

Speech is a varying sound signal. Humans are capable of making modifications to the sound signal using their vocal tract, tongue, and teeth to pronounce the phoneme. The features are a way to quantify data. A better representation of the speech signals to get the most information from the speech is through extracting features common among speech signals. Some characteristics of good features include [14]:

- The features should be independent of each other. Most features in the feature vector are correlated to each other. Therefore it is crucial to select a subset of features that are individual and independent of each other.
- The features should be informative to the context. Only those features that are more descriptive about the emotional content are to be selected for further analysis.
- The features should be consistent across all data samples. Features that are unique and specific to certain data samples should be avoided.
- The values of the features should be processed. The initial feature selection process can result in a raw feature vector that is unmanageable. The process of Feature Engineering will remove any outliers, missing values, and null values.

The features in a speech percept that is relevant to the emotional content can be grouped into two main categories:

1. Prosodic features
2. Phonetic features.

The prosodic features are the energy, pitch, tempo, loudness, formant, and intensity. The phonetic features are mostly related to the pronunciation of the words based on the language. Therefore for the purpose of emotion detection, the analysis is performed on the prosodic features or a combination of them. Mostly the pitch and loudness are the features that are very relevant to the emotional content.

4.2 LIST OF FEATURES

See Table 1 for the features that were extracted for each frame of the audio signal, along with their definitions [15].

Table.1 List of features present in an audio signal

Feature ID	Feature Name	Description
1	Zero Crossing Rate	*"The rate at which the signal changes its sign."*
2	Energy	*"The sum of the signal values squared and normalized using frame length."*
3	Entropy of Energy	*"The value of the change in energy."*
4	Spectral Centroid	*"The value at the center of the spectrum."*
5	Spectral Spread	*"The value of the bandwidth in the spectrum."*
6	Spectral Entropy	*"The value of the change in the spectral energy."*
7	Spectral Flux	*"The square of the difference between the spectral energies of consecutive frames."*
8	Spectral Rolloff	*"The value of the frequency under which 90% of the spectral distribution occurs."*
9-21	MFCCs	*"Mel Frequency Cepstral Coefficient values of the frequency bands distributed in the Mel-scale."*
22-33	Chroma Vector	*"The 12 values representing the energy belonging to each pitch class."*
34	Chroma Deviation	*"The value of the standard deviation of the Chroma vectors."*

4.3 Mel Frequency Cepstrum Coefficients (MFCC) FEATURES

A subset of features that are used for speech emotion detection is grouped under a category called the Mel Frequency Cepstrum Coefficients (MFCC) [16]. It can be explained as follows:

- The word Mel represents the scale used in Frequency vs Pitch measurement (see Figure 2) [16]. The value measured in frequency scale can be converted into Mel scale using the formula $\mathbf{m = 2595 \log_{10} (1 + (f/700))}$
- The word Cepstrum represents the Fourier Transform of the log spectrum of the speech signal.

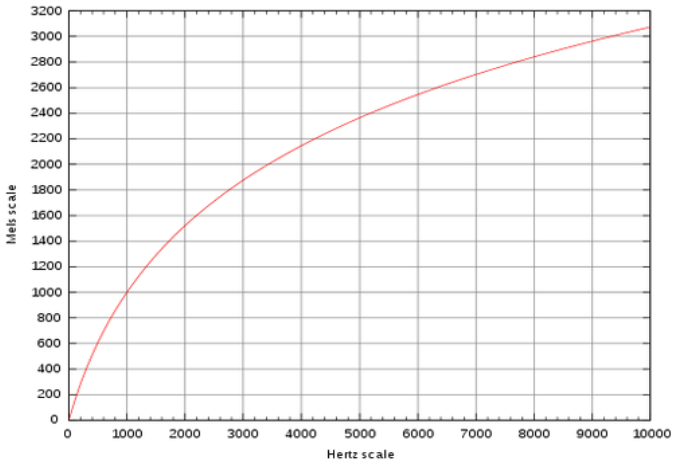

Fig.2 The Mel scale

Image Source: Practicalcryptography.com. (2018). Practical Cryptography. [online] Available at: http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/.

4.3.1 COEFFICIENT COMPUTATION

Below is the mathematical approach to compute the MFCC features from a speech signal [16]:

1. The first step is to frame the audio signal. The method of frame blocking discussed earlier is used to split the audio signals into frames of an optimal length of 20ms to 30ms, with 50% overlap.

2. The next step is mathematical. In this step, for each frame of the signal, the power spectrum is computed. The power spectrum, also known as Periodogram, identifies the frequencies present in each frame (see Figure 3) [16]. In order to select a particular band of frequencies, the value at each frame is multiplied by a Hamming window value. Mathematically, the periodogram is the squared value of the modulus of the Discrete Fourier Transform (DFT).

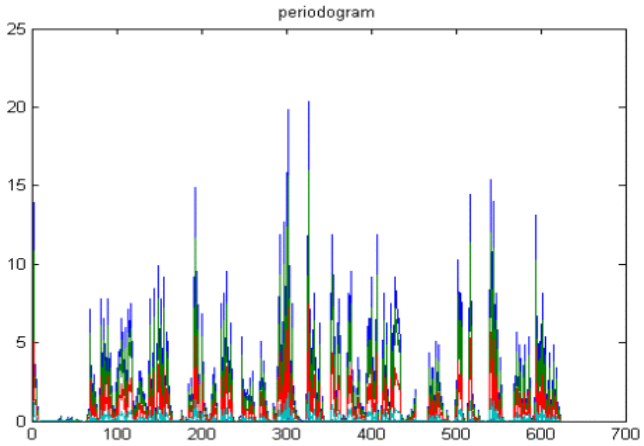

Fig.3 Periodogram

Image Source: Practicalcryptography.com. (2018). Practical Cryptography. [online] Available at: http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/.

3. Next, the power spectra obtained can contain many closely spaced frequencies. These variations in the frequencies make it difficult to obtain the energy values present in the signal. Thus, to scale the values, a filter named Mel Filterbank is applied to the power spectrum. The Mel Filterbank is a collection of triangular filters in the frequency domain. Nearing 0Hz, the frequencies are narrow to each other; further higher, the frequencies become wider (see Figure 4) [16]. The product of the power spectrum values and the Mel Filterbank values provides the energy in each frame. However, since overlapping frames are used in the analysis, the energy values obtained for the individual frames would be correlating with each other.

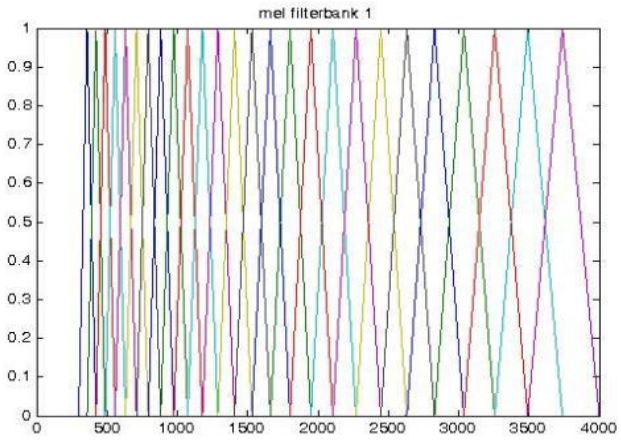

Fig.4 The Mel filterbank

Image Source: Practicalcryptography.com. (2018). Practical Cryptography. [online] Available at: http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/.

4. The final step is to de-correlate the energies. For this purpose, the Discrete Cosine Transform (DCT) function is used. DCT outputs a list of coefficient values corresponding to the pitch and energy values obtained so far. The lower level coefficients (the first 12 to 13 coefficients) of each frame represents steady changes in the pitch and energy values, and therefore they are better for analysis. These lower level coefficients are called the Mel Frequency cepstral coefficients.

4.4 FRAME BLOCKING

The frame blocking method is used to analyze sound signals. It is the process of dividing the sound signal into blocks known as frames and performing analysis over each block, rather than the signal at large. It is preferred to analyze individual frames because audio signals are stable within short time intervals. Several acoustic features can be interpreted from a single frame. In order to ensure the time-varying characteristics of the signal are measured accurately, some part of the neighboring frames is also analyzed at every step to identify any subtle changes in the sound signal. This value is often termed as frame

overlap, indicating the amount of overlap to include from the neighboring frames. The steps of frame blocking are as follows [17]:

1. Set *frame size* value to an appropriate number.

   Each frame should not be too small or too large, as this would mislead the time-varying characteristics of the features. Standard framing window size is 20ms to 30ms for audio signal processing.

2. Set *frame overlap* value.

   If the overlap value is too large, more duration from the neighboring frames will need to be analyzed at each analysis step. This will increase the computation and hence is not recommended. Ideally, ½ or 1/3 of frame overlap is suggested.

3. Perform analysis on each frame

   Each frame is a unit of computation of the sound signal. Feature extraction of frames will quantify the acoustic features of the audio signal.

4.5 SILENCE REMOVAL

An audio signal at the time of recording can accommodate silent regions where no utterances had been made. Such silent regions of the audio signal do not provide any useful information regarding the emotion expressed, and can be removed. A semi-supervised learning approach is used for silence detection in audio signals. In this approach, a model is initially trained with sample audio signals in order to be able to distinguish between high and low energy features [15]. Later, a percentage of high and low energy frames are used as endpoints to detect the regions of actual audio in the signal. Finally, applying the trained model over the entire signal will provide silence-free audio segments [15]. Threshold values such as frame size, frame step, and sampling frequency are tunable in order to smooth the output signals.

CHAPTER 5

DATA PREPARATION

5.1 DATA QUALITY ISSUES

Data must be cleaned to perform any meaningful analysis. As a next step, the dataset thus collected had to be inspected for its quality. Some of the data quality issues addressed for this experimentation includes:

1. Missing value analysis
2. Outlier identification
3. Null value handling
4. Invalid data
5. Duplicate data

5.1.1 MISSING VALUE ANALYSIS

Due to several influencing factors, a few or more data rows can contain no values for specific features. These values are termed ‘missing’ from the dataset. A large number of missing values can provide insights into the data. For example, if a particular feature has most of its values missing for all data rows, then it can be inferred that the feature is likely uncommon and can be removed from the dataset. Contrastingly, a small number of missing values can represent data entry error. Analyzing and amending individual features over missing values will improve and fix the quality of the dataset. Some of the methods to handle missing values include; if the number of missing values is large then the corresponding data rows or features can be removed, whereas if very few values are missing for a feature it can be imputed which means replacing with the mean or most frequent value of the feature.

5.1.2 OUTLIER IDENTIFICATION

Outlier values are also considered as data modifiers because often the prediction algorithm used will be misled by the outlier values. Outliers also alter the statistics of the overall data such as mean, variance and standard deviation. The proportion of the outliers amongst the whole dataset can be used to made decisions on how to handle them. If the outliers lie within a small range of difference and contribute to a very small proportion, then no fixes will be required. Some methods to handle large proportions of outliers include, replacing outlier values with boundary, or mean, or median, or mode values.

5.1.3 NULL VALUE HANDLING

A common error that can occur in a dataset is the null value error. It is when the words ‘null’ or ‘NA’ is used in place of missing values as fillers. Null values are mostly treated and handled in a fashion similar to missing values.

5.1.4 INVALID DATA

A dataset can have values irrelevant to the data type, such as symbols and special characters. These values, despite being meaningless, can cause errors during processing. Depending on the amount of invalid data present, it can either be removed or imputed.

5.1.5 DUPLICATE DATA

Few features might be a duplicate of each other, with different names or units of measurement. Such features increase the dimensionality of the data with no further significance. Removal of duplicate features is highly recommended.

5.2 NORMALIZATION AND STANDARDIZATION

Different characteristics of the audio signal, represented by its features, are computed on different units or scales. Rescaling the values to a uniform range will ensure accurate calculations are made. Many algorithms use distance metrics for their computation. Therefore it is necessary that all the values in the dataset are normalized. Two approaches are commonly used for the purpose of rescaling, namely Normalization and Standardization. Normalization alters all numeric values to lie in the range 0 to 1. For this purpose, all outliers in the data must be eliminated prior to normalizing the data. The formula for normalization is given as

$$x_{new} = (x - x_{min}) / (x_{max} - x_{min})$$

In the formula, x represents the data [18].

Standardization transforms the data to have a mean value of zero and a variance of one. However, standardizing the data provides more insights into the data than normalization. The formula for normalization is given as

$$x_{new} = (x - \mu) / \sigma$$

In the formula, x represents the data [18].

5.3 PEARSON CORRELATION COEFFICIENT

Correlation is the liner association that exists between each pair of features in the dataset and is used to identify the features that are highly associated or correlated with the decision attribute [19]. The Pearson correlation coefficient, *r* is a value that denotes the strength of the correlation. The value of *r* ranges between -1 to +1 through 0, where the negative values denote a lesser correlation between

variables and the positive values denote greater correlation. A value of 0 denotes that there is no correlation between the variables (see Figure 5) [19].

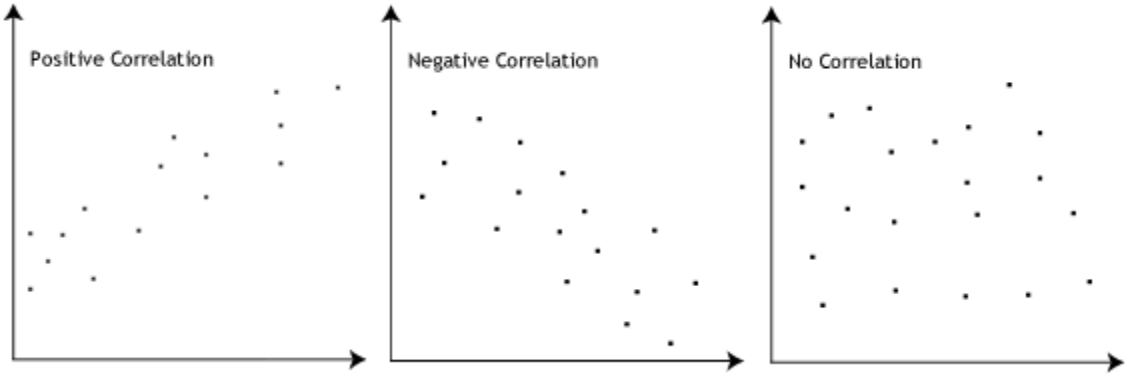

Fig.5 Pearson correlation coefficient

Image Source: Statistics.laerd.com. (2018). Pearson Product-Moment Correlation - When you should run this test, the range of values the coefficient can take and how to measure strength of association.. [online] Available at: https://statistics.laerd.com/statistical-guides/pearson-correlation-coefficient-statistical-guide.php.

The strength of the association can be determined using the value of the Pearson coefficient *r*. However, the strength of the association also depends on the type of variables under measurement. The Person method of correlation computation can be used on all numeric data irrespective of whether they have been scaled or not. Additionally, this approach treats all variables equally and does not consider any proposed dependence between the variables. The following guidelines have been proposed to determine the strength of correlation (see Figure 6) [19].

	Coefficient, *r*	
Strength of Association	Positive	Negative
Small	.1 to .3	-0.1 to -0.3
Medium	.3 to .5	-0.3 to -0.5
Large	.5 to 1.0	-0.5 to -1.0

Fig.6 Pearson correlation coefficient guidelines

Image Source: Statistics.laerd.com. (2018). Pearson Product-Moment Correlation - When you should run this test, the range of values the coefficient can take and how to measure strength of association.. [online] Available at: https://statistics.laerd.com/statistical-guides/pearson-correlation-coefficient-statistical-guide.php.

5.4 CLUSTERING

5.4.1 K MEANS CLUSTERING ALGORITHM

K-Means is an unsupervised clustering algorithm that will form 'k' groups within the data based on feature similarity. It is an iterative process by which data are iteratively grouped based on the similarity between their features. Clustering the data into groups provides more insights on the distribution of the training data available, and also easily helps classify any unknown (new) data. The algorithm is a two-step iterative process in which, based on distance metrics between the data points and centroids of the cluster, groups of similar data are created. The steps of the algorithm are [20]:

Initially, random values of the centroid(s) are assumed.

1. Data Assignment

   Based on the value of a distance metric (For example, Euclidean distance and Manhattan distance), data points are assigned to the closest neighboring centroids.

2. Centroid re-computation

   Centroids or mean value of all data points are calculated and updated at each step following the data assignment step.

Steps 1 and 2 are iteratively performed until the groups are distinctively classified. There are two conditions that ensure accuracy in clustering, namely: the inter-cluster distance and intra-cluster difference. The distance between the centroids of each cluster should be larger, ensuring that each group is well-separated from each other showing distinct differences. Additionally, the distance between each point within the cluster should be smaller, ensuring the similarity between data points within the group. By analyzing the final value of each centroid, the characteristics of the data belonging to the cluster can be quantitatively explained.

5.4.1.1 THE ELBOW METHOD - CHOOSING K-VALUE

The ‘k’ in k-means denote the number of clusters the data needed to be grouped into. Traditionally, the algorithm is repeated over different values of k and the results are compared by the average within cluster distance to the centroid. Alternatively, the elbow method can be used to depict an optimal value of k [20]. In the elbow method, the average within cluster distance to the centroid value is plotted against different values of k and the point of the curve where the distance sharply bends is the optimal value of k (see Figure 7) [20].

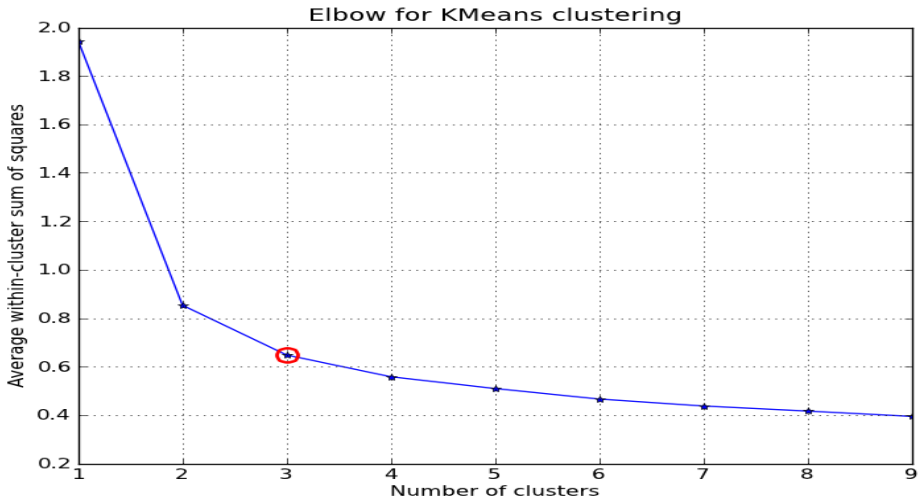

Fig.7 Elbow method for K means clustering

Image Source: Trevino, A. (2018). Introduction to K-means Clustering. [online] Datascience.com. Available at: https://www.datascience.com/blog/k-means-clustering.

5.5 PRINCIPAL COMPONENT ANALYSIS

In some cases, most or all of the features might have an impact on the decision making. However, a high dimensional dataset with a large feature space could potentially slow the performance of the system in terms of space and time complexity. Choosing the right set of features for analysis can be challenging, as it requires high-level domain knowledge. A solution to this problem is the Principle Component Analysis (PCA) technique for dimensionality reduction. PCA is an approach to bring out the principal components or the important aspects of the data. By using this method, the original feature space of the data is transformed into a new set of features while retaining the variation present in the data [21].

The technique of PCA analyses the variance of the data by measuring the covariance between the features. This is done mathematically using the concept of eigenvalues and eigenvectors. Eigenvalues are

numbers denoting the value of variance in each dimension of the data, and the eigenvector is the dimension with the highest eigenvalue. This eigenvector is the principal component of the dataset [22]. Given below are the implementation steps of PCA [21].

5.5.1 DATA NORMALIZATION

PCA works with the numerical values of the dataset to compute the variance, hence it is necessary that the values are scaled and normalized. All normalized data variables will have a mean value of 0.

5.5.2 COVARIANCE MATRIX COMPUTATION

An NxN covariance matrix is computed, where N is the number of features in the dataset. The elements of the covariance matrix represent the variance between each of the features in the dataset.

5.5.3 EIGENVALUES AND EIGENVECTOR COMPUTATION

Eigenvalues and eigenvectors are computed using the covariance matrix. This computation is purely mathematical and many programming libraries have built-in functions for this calculation. At the end of the computation, N eigenvalues for an N-dimensional dataset is obtained. The eigenvalues thus obtained are the components of PCA [22].

5.5.4 CHOOSING COMPONENTS

The eigenvector component with the largest eigenvalue is the 1st Principal component, containing the most information about the dataset. Sorting the eigenvalues in decreasing order can give the list of principal components with the amount of variance needed. Depending on how much

information is needed, programmers can choose the top P number of components needed for further analysis.

5.5.5 FORMING PRINCIPLE COMPONENTS

A new dataset is created using the principal components selected for analysis. Mathematically, left multiplication of the transposed feature vector with the scaled original dataset will produce the new dataset.

New Dataset = $(\text{Feature Vector})^T$ X $(\text{Scaled Data})^T$

5.5.6 SCREE PLOT

Scree plot is a way to select the optimal number of components to be selected such that enough information is being retained form the raw dataset. It is a curve plot having the information maintained and the number of components on the different axis. The elbow point of the curve indicates the optimal value of components to be used for further analysis.

CHAPTER 6

ALGORITHMS

6.1 LOGISTIC REGRESSION

Logistic Regression is a supervised classification algorithm which produces probability values of data belonging to different classes [23]. There are three types of Logistic Regression algorithms, namely Binary class, Multi-class and Ordinal class logistic algorithms depending on the type of target class. The Wikipedia definition states that "*Logistic regression computes the relationship between the target (dependent) variable and one or more independent variables using the estimated probability values through a logistic function"* [24]. The logistic function, also known as a sigmoid function, maps predicted values to probability values. The procedure of a multiclass logistic regression algorithm is as follows:

1. For an N class problem, divide into N pairs of binary class problems.
2. For each binary class problem
    2.1 For each observation of a binary class problem
        2.1.1 Compute probability values of the observation belonging to a class
3. Make the final prediction by computing the maximum probability value amongst all classes

The time complexity of the algorithm is in the order of the number of data samples, represented as

$O(n_{samples})$.

6.2 NAÏVE BAYES

Naïve Bayes classifier is based on Bayes theorem, which determines the probability of an event based on a prior probability of events [26]. Bayes theorem is used to compute prior probability values.

This classifier algorithm assumes feature independence. No correlation between the features is considered. The algorithm is said to be Naïve because it treats all the features to independently contribute to deciding the target class. The steps of a simple Naïve Bayes algorithm is as follows [25]:

1. Create a frequency table for all features individually. Tag the frequency of each entry against the target class.
2. Create a likelihood table by computing probability values for each entry in the frequency table.
3. Calculate posterior probability for each target class using the Bayes theorem.
4. Declare the target class with the highest posterior probability value as the predicted outcome.

The time complexity of the algorithm is in the order of the number of data samples, represented as O ($n_{samples}$). There are three types of Naïve Bayes algorithm, namely: the Gaussian Naïve Bayes (GNB) which is applicable with features following a normal distribution, the Multinomial Naïve Bayes (MNB) which is most suited to use when the number of times the outcome occurs is to be computed, and the Bernoulli Naïve Bayes (BNB) for a dataset with binary features.

6.3 SUPPORT VECTOR MACHINES

Support Vector Machines (SVM) are a supervised algorithm that works for both classification and regression problems. Support vectors are coordinate points in space, formed using the attributes of a data point. Briefly, for an N-dimensional dataset, each data point is plotted on an N-dimensional space using all its feature vector values as a coordinate point [27]. Classification between the classes is performed by finding a hyperplane in space that clearly separates the distinct classes. SVM works best for high

dimensional data. The important aspect of implementing SVM algorithm is finding the hyperplane. Two conditions are to be met in the order given while choosing the right hyperplane.

1. The hyperplane should classify the classes most accurately
2. The margin distance from the hyperplane to the nearest data point must be maximized.

For a low dimensional dataset, the method of *kernel trick* in SVM introduces additional features to transform the dataset to a high dimensional space and thereby make identifying the hyperplane achievable.

The linear solver based SVM is a better implementation of SVM in terms of time complexity. The complexity scales between $O(n_{samples} \times n^2_{samples})$ and $O(n_{samples} \times n^3_{samples})$.

6.4 K-NEAREST NEIGHBOR

K-Nearest Neighbor (KNN) is the simplest classification algorithm. The approach is to plot all data points on space, and with any new sample, observe its k nearest points on space and make a decision based on majority voting. Thus, KNN algorithm involves no training and it takes the least calculation time when implemented with an optimal value of k. The steps of KNN algorithm is as follows [28]:

1. For a given instance, find its distance from all other data points. Use an appropriate distance metric based on the problem instance.
2. Sort the computed distances in increasing order. Depending on the value of k, observe the nearest k points.
3. Identify the majority class amongst the k points, and declare it as the predicted class.

Choosing an optimal value of k is a challenge in this approach. Most often, the process is repeated for a number of different trials of k. The evaluation scores are then observed using a graph to find the optimal value of k.

There is no training in the model of KNN and hence there is no training time complexity value. While testing, the number of nearest samples to be looked up for decides the complexity of the algorithm and is controlled by the value of k.

6.5 DECISION TREE

The Decision tree is an approach where the entire dataset is visualized as a tree, as the classification problem is loosely translated as finding the path from the root to leaf based on several decision conditions at each sub-tree level. Each feature in the dataset is treated as a decision node at its sub-tree level. The initial tree is designed using the values of the training sample. For a new data point, its values are tracked on the built tree from root to leaf where the leaf node represents the target or predicted class. There are two types of decision trees based on the type of the target class, namely [29]:

1. Binary variable decision tree – for binary class problems
2. Continuous variable decision tree – for continuous value problem

The Decision tree is a very simple algorithm to implement, as it requires less data preparation. It is also widely used in data exploration. However, decision trees are very susceptible to noises in the dataset. This might lead to the problem of overfitting.

The time complexity of decision tree depends on the height of the tree controlled by the number of data samples, and by the number of features used for the split. It can be represented as $O(n_{samples}\, n_{features} \log(n_{samples}))$.

6.6 RANDOM FOREST

Random forest is a supervised classification similar to decision trees. While the root node and splitting features in the decision tree are based on the Gini and Information gain values, the random forest algorithm does it in a random fashion. The random forest is a collection of decision trees; therefore, a large number of trees gives better results. Overfitting is a potential drawback of random forests, but increasing the number of trees can reduce overfitting. Random forest also has several advantages like its capability to handle missing values and classify multi-class categorical variables. The steps of building a random forest classifier are as follows [30]:

1. Select a subset of features from the dataset.
2. From the selected subset of features, using the best split method, pick a node.
3. Continue the best split method to form child nodes from the subset of features.
4. Repeat the steps until all nodes are used as split.
5. Iteratively create n number of trees using steps 1 -4 to form a forest.

The time complexity of Random Forest is higher by the factor of the number of trees used for building the forest.

6.7 GRADIENT BOOSTING TREE

Gradient boosting is a class of algorithms which is used over other regular algorithms (for example Gradient boosting over decision trees algorithm) in order to improve the performance of the regular algorithm. The improvement is created over three steps [31]:

1. Optimizing the loss function

   It is a mathematical function representing errors occurring in a model. The logarithmic loss function is best suited for classification problems. However, one may define their own loss function.

2. Using a weak learner for the predictions

   A learning algorithm such as decision tree is a greedy approach, as it decides its splitting attribute at each step using the best split method. Hence this algorithm is usually a good choice to be used with gradient boosting.

3. An additive model that minimizes the loss function by adding more weak learner

   Gradient descent algorithmic models use weak learner algorithms as a sub-model, in this case, it is a Decision Tree algorithm. Conventionally, the gradient descent procedure is used to reduce the weights of the parameters used in the weak learner. After calculating the loss induced by the model, at each step more trees are added to the model to reduce the loss or errors.

CHAPTER 7

EVALUATION

7.1 EVALUATION METRICS

The most important characteristic of machine learning models is its ability to improve. Once the model is built, even before testing the model on real data, machine learning experts evaluate the performance of the model. Evaluation metrics reveal important model parameters and provides numeric scores that will help judge the functioning of the model. The most important metric needed to evaluate the model is the confusion matrix (see Figure 8) [32].

The structure of a confusion matrix is against the actual and predicted positive and negative classes, and contains four values which are used to compute other metrics. The true positive represents the correct predictions made in the positive class, and the true negatives represent the correct predictions made in the negative class. The false positives and false negatives are the observations wrongly predicted for their respective classes.

	Predicted class		
Actual Class		Class = Yes	Class = No
	Class = Yes	True Positive	False Negative
	Class = No	False Positive	True Negative

Fig.8 Confusion Matrix

Image Source: Exsilio Blog. (2018). Accuracy, Precision, Recall & F1 Score: Interpretation of Performance Measures - Exsilio Blog. [online] Available at: http://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/.

Four important metrics can be derived using the values in the confusion matrix, namely [31]:

7.1.1 ACCURACY

It is the ratio of the observations predicted correctly to the total number of observations. Accuracy works best for datasets with an equal class distribution, and hence it is not always a good measure to evaluate the model. Accuracy can be computed as

Accuracy = (True Positives + True Negatives) / (True Positives + False positives+True Negatives + False Negatives)

7.1.2 PRECISION

It is the ratio of the positive observations predicted correctly to the total positive observations predicted. Higher the value of precision, better and more accurate the model actually is. Precision can also work with an uneven class distribution. It can be computed as

Precision = True Positives / (True positives + False positives)

7.1.3 RECALL OR SENSITIVITY

It is the ratio of the positive observations predicted correctly to the total positive observations. A recall score of 50% and more reveals a good performing model. Recall can also work with an uneven class distribution. It can be computed as

Recall = True positives / (True positives + False negatives)

7.1.4 F1 score

It is the weighted average value of precision and recall. The F1 score is the best metric for uneven class distribution. F1 can be computed as

F1= 2 * (Recall * Precision) / (Recall + Precision)

CHAPTER 8

IMPLEMENTATION

8.1 DATA COLLECTION

The first step in implementing the Speech Emotion Recognition system is to collect audio samples under different emotional categories which can be used to train the model. The audio samples are usually wav or mp3 files and publically available for download. The following steps are explained relative to the experiments performed on the TESS dataset.

8.2 PYTHON LIBRARY

The next step after data collection was to represent these audio files numerically, in order to perform further analysis on them. This step is called feature extraction, where quantitative values for different features of the audio is obtained. The pyAudioAnalysis library was used for this purpose [15]. This python library provides functions for short-term feature extraction, with tunable windowing parameters such as frame size and frame step. At the end of this step, each audio file was represented as a row in a CSV file with 34 columns representing the different features. Each feature will have a range of values for one audio file obtained over the various frames in that audio signal. The python library pyAudioAnalysis is an open Python library that provides a wide range of audio-related functionalities focusing on feature extraction, classification, segmentation, and visualization issues. The library depends on several other libraries which are:

- Numpy
- Matplotlib
- Scipy

- Sklearn
- Hmmlearn
- Simplejson
- eyeD3
- pydub

8.3 DATA VISUALIZATION

Visualizing the data gives more understanding of the problem and the type of solution to be built. The distribution of classes, the number of instances under each category, the spread of the data, the correlation between the features and clustering are a few methods to visualize the data. Python and R provide statistical functions for data visualization.

8.3.1 FEATURE ANALYSIS

Primarily, the number of rows and columns and a preview of the data is viewed (see Figure 9.A).

```
In [3]: data = pd.read_csv('data.csv') #reading the csv data

In [4]: data.shape
Out[4]: (2399, 36)

In [5]: data.head(n=3) #preview of the data
Out[5]:
```

	1	2	3	4	5	6	7	8	9	10	...	27	28	29	30	31
0	0.237025	0.001949	2.180585	0.342264	0.210608	1.761245	0.001733	0.412815	-26.535934	0.319416	...	0.000223	0.001806	0.000748	0.000370	0.005889
1	0.255306	0.019663	2.964614	0.369318	0.233425	1.677484	0.009996	0.505462	-26.642530	1.078969	...	0.002802	0.008537	0.009402	0.003874	0.002074
2	0.262975	0.004484	1.195476	0.356051	0.253092	1.473366	0.003087	0.467647	-27.314425	0.848172	...	0.002532	0.001380	0.000552	0.000348	0.000545

3 rows × 36 columns

Fig.9.A. An overview of data

Next, the number of examples under each category are counted (see Figure 9.B).

```
In [6]: data['36'].value_counts()
Out[6]: Surprise    402
        Happy       400
        Disgust     400
        Neutral     400
        Sad         399
        Fear        200
        Angry       198
        Name: 36, dtype: int64
```

Fig.9.B. Overview of data grouped by categories

Distribution of the data on each of its feature can be visualized using box plots and violin plots or using pair plots. The Seaborn package in Python provides methods to draw such plots. Shown here is the data distribution of the feature 'Energy' among the different categories (see Figure 10.A and Figure 10.B).

```
In [12]: import seaborn as sns
         sns.boxplot(x="36", y="2", data=data)
Out[12]: <matplotlib.axes._subplots.AxesSubplot at 0x5efd3e34a8>
```

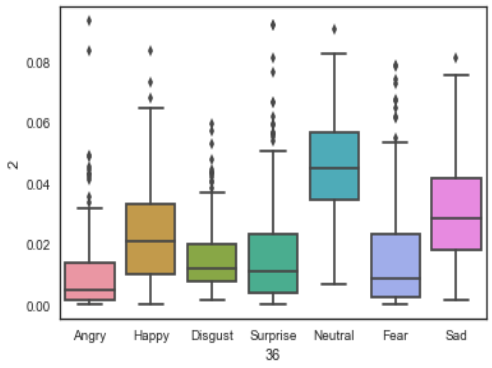

Fig.10.A. Box plot analysis of feature values

```
In [13]: sns.violinplot(x="36", y="2", data=data)
Out[13]: <matplotlib.axes._subplots.AxesSubplot at 0x5efd4fb358>
```

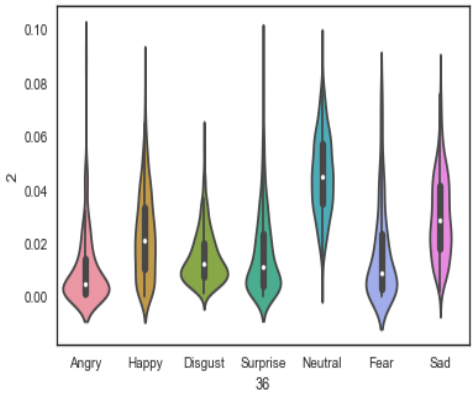

Fig.10.B. Violin plot analysis of feature values

The statistical language R provides several functions to effectively understand the statistics of the data. For each feature, the statistical values were visualized and it was observed that the raw data was not standardized (see Figure 11).

```
       X1                X2                X3              X4                X5              X6                X7                X8                X9
 Min.   :0.01898   Min.   :0.0002479   Min.   :0.8198   Min.   :0.1076   Min.   :0.1338   Min.   :0.01319   Min.   :0.0001811   Min.   :0.02143   Min.   :-32.96
 1st Qu.:0.15546   1st Qu.:0.0080144   1st Qu.:2.6473   1st Qu.:0.2931   1st Qu.:0.1840   1st Qu.:1.04099   1st Qu.:0.0035809   1st Qu.:0.31828   1st Qu.:-27.33
 Median :0.22599   Median :0.0198240   Median :2.8741   Median :0.3289   Median :0.2000   Median :1.33882   Median :0.0061594   Median :0.39727   Median :-26.32
 Mean   :0.23878   Mean   :0.0237834   Mean   :2.7632   Mean   :0.3455   Mean   :0.2023   Mean   :1.31511   Mean   :0.0073466   Mean   :0.40830   Mean   :-26.52
 3rd Qu.:0.29751   3rd Qu.:0.0363465   3rd Qu.:2.9984   3rd Qu.:0.4005   3rd Qu.:0.2174   3rd Qu.:1.53587   3rd Qu.:0.0096574   3rd Qu.:0.48298   3rd Qu.:-25.54
 Max.   :0.69626   Max.   :0.0933426   Max.   :3.2281   Max.   :0.7304   Max.   :0.3118   Max.   :3.00326   Max.   :0.0509154   Max.   :0.96534   Max.   :-22.13

      X10               X11               X12               X13               X14               X15               X16               X17
 Min.   :-1.122   Min.   :-2.1393   Min.   :-1.25896   Min.   :-1.4527   Min.   :-1.03556   Min.   :-1.04332   Min.   :-1.20529   Min.   :-1.3340107
 1st Qu.: 1.026   1st Qu.:-0.7804   1st Qu.:-0.21547   1st Qu.:-0.6456   1st Qu.:-0.32735   1st Qu.:-0.15881   1st Qu.:-0.17463   1st Qu.:-0.1659003
 Median : 1.498   Median :-0.3394   Median : 0.07098   Median :-0.4172   Median :-0.07785   Median : 0.04823   Median :-0.03496   Median :-0.0004108
 Mean   : 1.495   Mean   :-0.3457   Mean   : 0.04115   Mean   :-0.4169   Mean   :-0.04215   Mean   : 0.01938   Mean   :-0.06578   Mean   :-0.0518579
 3rd Qu.: 2.010   3rd Qu.: 0.1162   3rd Qu.: 0.32359   3rd Qu.:-0.1893   3rd Qu.: 0.19766   3rd Qu.: 0.20017   3rd Qu.: 0.08734   3rd Qu.: 0.1305859
 Max.   : 3.780   Max.   : 1.3581   Max.   : 1.12542   Max.   : 0.5616   Max.   : 1.04685   Max.   : 0.94772   Max.   : 0.49288   Max.   : 0.7206110

      X18               X19                X20               X21               X22               X23               X24               X25
 Min.   :-0.96893   Min.   :-0.8357281   Min.   :-0.79690   Min.   :-0.85185   Min.   :0.0000211   Min.   :0.0000271   Min.   :0.0000623   Min.   :0.0000744
 1st Qu.:-0.30044   1st Qu.:-0.1358281   1st Qu.:-0.29167   1st Qu.:-0.33274   1st Qu.:0.0005862   1st Qu.:0.0004490   1st Qu.:0.0009469   1st Qu.:0.0005527
 Median :-0.15212   Median :-0.0002672   Median :-0.09269   Median :-0.17381   Median :0.0015330   Median :0.0010544   Median :0.0023683   Median :0.0012137
 Mean   :-0.17123   Mean   :-0.0049830   Mean   :-0.06890   Mean   :-0.13697   Mean   :0.0050965   Mean   :0.0033127   Mean   :0.0047547   Mean   :0.0034065
 3rd Qu.:-0.02544   3rd Qu.: 0.1328477   3rd Qu.: 0.11949   3rd Qu.: 0.03094   3rd Qu.:0.0048227   3rd Qu.:0.0035430   3rd Qu.:0.0056950   3rd Qu.:0.0035870
 Max.   : 0.51956   Max.   : 0.9343624   Max.   : 1.03844   Max.   : 0.74888   Max.   :0.1042071   Max.   :0.1137400   Max.   :0.0840600   Max.   :0.0827991

      X26               X27               X28               X29               X30               X31               X32               X33
 Min.   :0.0000979   Min.   :0.0000397   Min.   :0.0000409   Min.   :0.0000350   Min.   :0.0000362   Min.   :0.0000343   Min.   :0.0000418   Min.   :0.0000305
 1st Qu.:0.0009131   1st Qu.:0.0003715   1st Qu.:0.0008550   1st Qu.:0.0007248   1st Qu.:0.0005415   1st Qu.:0.0008563   1st Qu.:0.0033102   1st Qu.:0.0007767
 Median :0.0020207   Median :0.0010257   Median :0.0022048   Median :0.0018806   Median :0.0016713   Median :0.0032559   Median :0.0096335   Median :0.0020540
 Mean   :0.0058273   Mean   :0.0049895   Mean   :0.0046888   Mean   :0.0051188   Mean   :0.0067453   Mean   :0.0124376   Mean   :0.0161145   Mean   :0.0082052
 3rd Qu.:0.0052759   3rd Qu.:0.0038781   3rd Qu.:0.0054777   3rd Qu.:0.0051923   3rd Qu.:0.0064854   3rd Qu.:0.0126654   3rd Qu.:0.0225707   3rd Qu.:0.0073837
 Max.   :0.1346614   Max.   :0.1143216   Max.   :0.1299185   Max.   :0.1319882   Max.   :0.1165654   Max.   :0.1797250   Max.   :0.1867128   Max.   :0.1422646

      X34              X35            X36
 Min.   :0.0002408   Old  :1399   Angry   :198
 1st Qu.:0.0062759   Young:1000   Disgust :400
 Median :0.0105646                Fear    :200
 Mean   :0.0128518                Happy   :400
 3rd Qu.:0.0169517                Neutral :400
```

Fig.11 Summary of data-before standardization

8.3.2 CORRELATION

The pearson correlation coefficient values were analyzed which provided insights about the features that correlate positively and negatively with the target class. In this observation, no features had a negative correlation with the target class (see Figure 12).

```
> flattenCorrMatrix(round(res2$r,2), round(res2$P,2))
    row column   cor    p
1    X1     X2 -0.19 0.00
2    X1     X3 -0.05 0.03
3    X2     X3  0.37 0.00
4    X1     X4  0.88 0.00
5    X2     X4 -0.19 0.00
6    X3     X4 -0.05 0.03
7    X1     X5 -0.10 0.00
8    X2     X5 -0.16 0.00
9    X3     X5 -0.41 0.00
10   X4     X5  0.10 0.00
11   X1     X6  0.51 0.00
12   X2     X6 -0.19 0.00
13   X3     X6 -0.02 0.24
14   X4     X6  0.72 0.00
15   X5     X6  0.25 0.00
16   X1     X7 -0.36 0.00
17   X2     X7  0.37 0.00
18   X3     X7  0.33 0.00
19   X4     X7 -0.39 0.00
20   X5     X7 -0.15 0.00
21   X6     X7 -0.49 0.00
22   X1     X8  0.80 0.00
23   X2     X8 -0.16 0.00
24   X3     X8  0.04 0.07
25   X4     X8  0.93 0.00
26   X5     X8  0.07 0.00
27   X6     X8  0.85 0.00
28   X7     X8 -0.42 0.00
29   X1     X9 -0.45 0.00
30   X2     X9  0.48 0.00
31   X3     X9  0.05 0.03
32   X4     X9 -0.62 0.00
33   X5     X9 -0.07 0.00
34   X6     X9 -0.38 0.00
35   X7     X9  0.05 0.01
36   X8     X9 -0.54 0.00
472  X7    X32  0.18 0.00
473  X8    X32 -0.31 0.00
474  X9    X32 -0.03 0.09
475 X10    X32  0.22 0.00
476 X11    X32  0.10 0.00
477 X12    X32  0.11 0.00
478 X13    X32  0.03 0.12
479 X14    X32 -0.22 0.00
480 X15    X32  0.10 0.00
481 X16    X32  0.06 0.00
482 X17    X32  0.29 0.00
483 X18    X32  0.23 0.00
484 X19    X32  0.07 0.00
485 X20    X32  0.01 0.65
486 X21    X32 -0.32 0.00
487 X22    X32 -0.03 0.15
488 X23    X32 -0.03 0.16
489 X24    X32 -0.02 0.38
490 X25    X32 -0.10 0.00
491 X26    X32 -0.13 0.00
492 X27    X32 -0.11 0.00
493 X28    X32  0.03 0.14
494 X29    X32  0.04 0.04
495 X30    X32  0.07 0.00
496 X31    X32  0.12 0.00
497  X1    X33  0.00 0.98
498  X2    X33  0.12 0.00
499  X3    X33  0.07 0.00
500  X4    X33 -0.02 0.34
501  X5    X33 -0.06 0.00
502  X6    X33 -0.17 0.00
503  X7    X33  0.35 0.00
504  X8    X33 -0.07 0.00
505  X9    X33 -0.08 0.00
506 X10    X33  0.19 0.00
507 X11    X33  0.17 0.00
508 X12    X33  0.16 0.00
509 X13    X33  0.09 0.00
```

Fig.12 Correlation values of data

8.3.3 CLUSTERING

Clustering the data provided a deeper understanding of the features. The k-means clustering was performed iteratively for various values of k and evaluated against the sum of squares metric. For k values less than 7, the cluster grouping was imbalanced, and for values greater than 10, the clusters were becoming too spread out. The elbow method of plotting reveals a sharp turn at k=7 which produces balanced clusters (see Figure 13). Interestingly there are 7 categories of emotions tagged in the dataset, hence making 7 distinct cluster groups of the data shows a clear separation with the feature values among the groups.

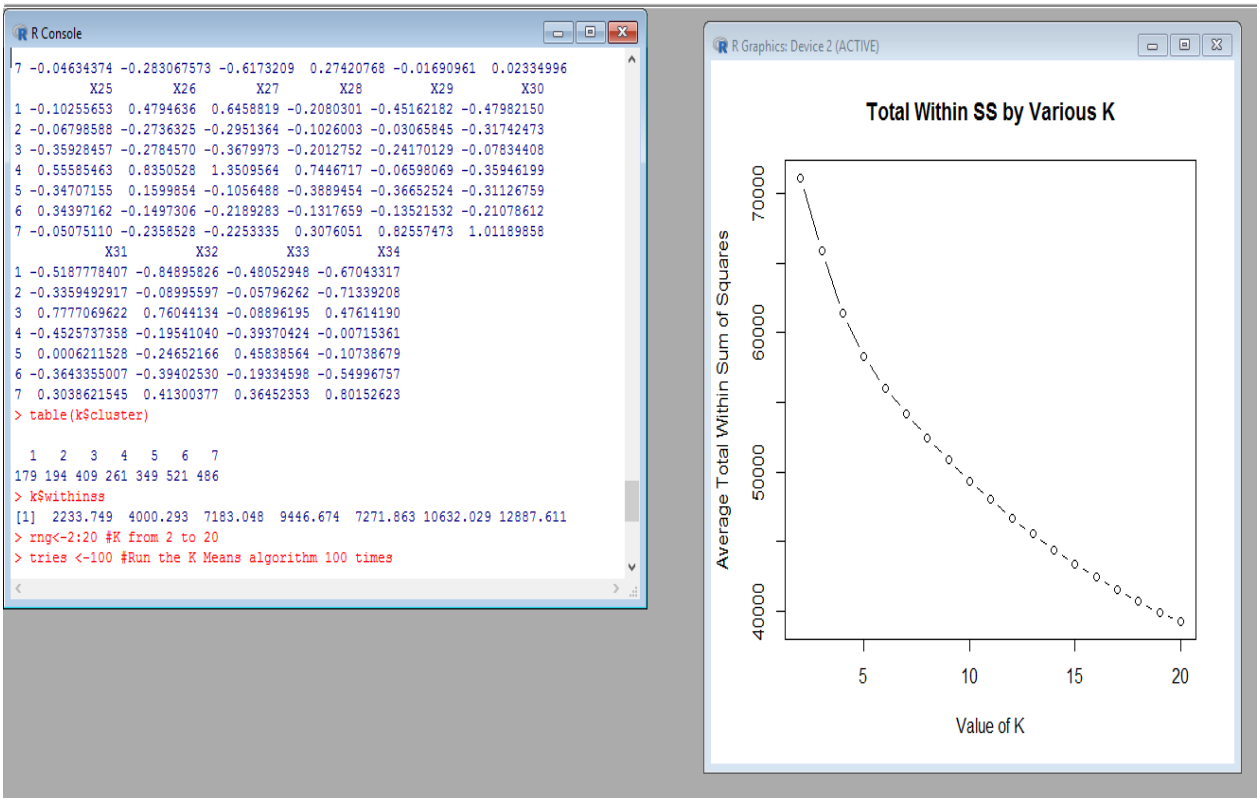

Fig.13 K-Means clustering of data and the elbow method

8.4 DATA PREPARATION

After analyzing the data through various visualizations, the next step is to prepare the data for processing. The steps of data preparation include fixing quality issues, standardization, and normalization. First, the data is checked for quality issues such as missing values (see Figure 14), outliers (see Figure 15), invalid data and duplicate data. There were no missing values, invalid or duplicate values in the dataset.

```
> sapply(data, function(x) sum(is.na(x)))
 X1  X2  X3  X4  X5  X6  X7  X8  X9 X10 X11 X12 X13 X14 X15 X16 X17 X18 X19 X20 X21 X22 X23 X24 X25 X26 X27 X28 X29 X30 X31 X32 X33 X34 X35 X36
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
>
```

Fig.14 Missing value analysis

With the outlier analysis, for each feature, the proportion of the outliers are viewed along with the changes in the mean value of the feature with and without the outliers (see Figure 17). This insight will help decide if the outliers are actual outliers or if they contribute to decision making.

```
> outlierKD(data,data$X20)
Outliers identified: 39 from 2399 observations
Proportion (%) of outliers: 1.62567736556899
Mean of the outliers: 0.850866036692308
Mean without removing outliers: -0.0689005157190496
Mean if we remove outliers: -0.0841000477292373
```

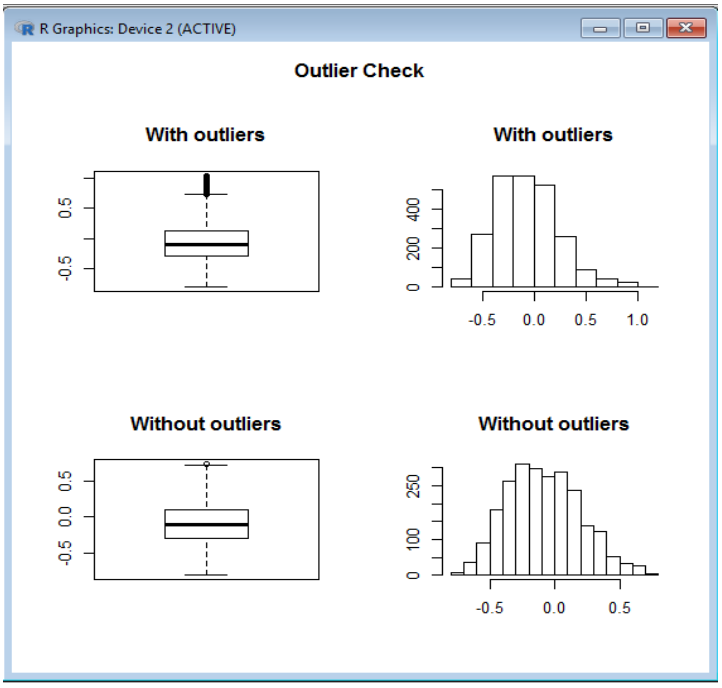

Fig.15 Outlier analysis

Next, normalization was performed on the data as the raw data was recorded on a different scale. After standardization, all feature values now are in the range 0 to 1 (see Figure 16).

```
> normalize<-function(x){
+           return ((x-min(x))/ (max(x)-min(x)))}
> data_norm <- as.data.frame(lapply(data[1:34],normalize)
+ )
> summary(data_norm)
       X1                X2                X3                X4                X5                X6                X7                X8                X9
 Min.   :0.0000   Min.   :0.00000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.00000   Min.   :0.0000   Min.   :0.0000
 1st Qu.:0.2015   1st Qu.:0.08343   1st Qu.:0.7589   1st Qu.:0.2977   1st Qu.:0.2819   1st Qu.:0.3437   1st Qu.:0.06701   1st Qu.:0.3145   1st Qu.:0.5197
 Median :0.3057   Median :0.21028   Median :0.8530   Median :0.3553   Median :0.3721   Median :0.4433   Median :0.11784   Median :0.3982   Median :0.6129
 Mean   :0.3245   Mean   :0.25281   Mean   :0.8070   Mean   :0.3819   Mean   :0.3850   Mean   :0.4354   Mean   :0.14124   Mean   :0.4099   Mean   :0.5952
 3rd Qu.:0.4112   3rd Qu.:0.38776   3rd Qu.:0.9046   3rd Qu.:0.4703   3rd Qu.:0.4697   3rd Qu.:0.5092   3rd Qu.:0.18678   3rd Qu.:0.4890   3rd Qu.:0.6855
 Max.   :1.0000   Max.   :1.00000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.00000   Max.   :1.0000   Max.   :1.0000
      X10               X11               X12               X13               X14               X15               X16               X17               X18
 Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
 1st Qu.:0.4382   1st Qu.:0.3886   1st Qu.:0.4376   1st Qu.:0.4007   1st Qu.:0.3401   1st Qu.:0.4442   1st Qu.:0.6069   1st Qu.:0.5685   1st Qu.:0.4491
 Median :0.5344   Median :0.5146   Median :0.5578   Median :0.5141   Median :0.4599   Median :0.5482   Median :0.6892   Median :0.6491   Median :0.5487
 Mean   :0.5338   Mean   :0.5128   Mean   :0.5453   Mean   :0.5142   Mean   :0.4771   Mean   :0.5337   Mean   :0.6710   Mean   :0.6240   Mean   :0.5359
 3rd Qu.:0.6389   3rd Qu.:0.6449   3rd Qu.:0.6637   3rd Qu.:0.6272   3rd Qu.:0.5922   3rd Qu.:0.6245   3rd Qu.:0.7612   3rd Qu.:0.7128   3rd Qu.:0.6339
 Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
      X19               X20               X21               X22               X23               X24               X25               X26               X27
 Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.000000   Min.   :0.000000   Min.   :0.00000   Min.   :0.000000   Min.   :0.000000   Min.   :0.000000
 1st Qu.:0.3954   1st Qu.:0.2753   1st Qu.:0.3243   1st Qu.:0.005424   1st Qu.:0.003710   1st Qu.:0.01053   1st Qu.:0.005781   1st Qu.:0.006058   1st Qu.:0.002903
 Median :0.4720   Median :0.3837   Median :0.4236   Median :0.014512   Median :0.009035   Median :0.02745   Median :0.013772   Median :0.014289   Median :0.008627
 Mean   :0.4693   Mean   :0.3967   Mean   :0.4466   Mean   :0.048715   Mean   :0.028893   Mean   :0.05586   Mean   :0.040280   Mean   :0.042578   Mean   :0.043312
 3rd Qu.:0.5472   3rd Qu.:0.4993   3rd Qu.:0.5515   3rd Qu.:0.046087   3rd Qu.:0.030919   3rd Qu.:0.06706   3rd Qu.:0.042461   3rd Qu.:0.038480   3rd Qu.:0.033587
 Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.000000   Max.   :1.000000   Max.   :1.00000   Max.   :1.000000   Max.   :1.000000   Max.   :1.000000
      X28               X29               X30               X31               X32               X33               X34
 Min.   :0.000000   Min.   :0.000000   Min.   :0.000000   Min.   :0.000000   Min.   :0.00000   Min.   :0.000000   Min.   :0.0000
 1st Qu.:0.006268   1st Qu.:0.005228   1st Qu.:0.004336   1st Qu.:0.004574   1st Qu.:0.01751   1st Qu.:0.005246   1st Qu.:0.1072
 Median :0.016661   Median :0.013987   Median :0.014032   Median :0.017929   Median :0.05138   Median :0.014227   Median :0.1834
 Mean   :0.035786   Mean   :0.038527   Mean   :0.057575   Mean   :0.069026   Mean   :0.08610   Mean   :0.057473   Mean   :0.2240
 3rd Qu.:0.041861   3rd Qu.:0.039084   3rd Qu.:0.055344   3rd Qu.:0.070294   3rd Qu.:0.12069   3rd Qu.:0.051698   3rd Qu.:0.2969
 Max.   :1.000000   Max.   :1.000000   Max.   :1.000000   Max.   :1.000000   Max.   :1.00000   Max.   :1.000000   Max.   :1.0000
>
```

Fig.16 Summary of data-after standardization

8.5 FEATURE ENGINEERING

Feature engineering is the process of transforming, reducing or constructing features for the dataset. As mentioned earlier in the raw data, each feature has multiple values for each frame of the audio signal. By the frame blocking and windowing techniques, the frame size and frame overlap values can be tuned to obtain accurate values of the audio signal. Further, using the averaging technique, average values of different features for the audio signals are obtained. Now the transformed data contains 34 discrete values representing each audio signal (see Figure 17).

```
'data.frame':   2399 obs. of  36 variables:
 $ X1 : num  0.237 0.255 0.263 0.257 0.36 ...
 $ X2 : num  0.00195 0.01966 0.00448 0.01052 0.00177 ...
 $ X3 : num  2.18 2.96 1.2 2.7 1.99 ...
 $ X4 : num  0.342 0.369 0.356 0.319 0.446 ...
 $ X5 : num  0.211 0.233 0.253 0.161 0.212 ...
 $ X6 : num  1.76 1.68 1.47 1.13 1.02 ...
 $ X7 : num  0.00173 0.01 0.00309 0.00481 0.00627 ...
 $ X8 : num  0.413 0.505 0.468 0.371 0.467 ...
 $ X9 : num  -26.5 -26.6 -27.3 -25.5 -30.9 ...
 $ X10: num  0.319 1.079 0.848 1.525 1.515 ...
 $ X11: num  -0.601 -0.449 -0.815 -1.599 0.199 ...
 $ X12: num  0.377 0.233 -0.805 0.542 0.113 ...
 $ X13: num  -0.1659 -0.5333 -0.1893 -0.3023 -0.0479 ...
 $ X14: num  -0.0775 -0.146 0.086 -0.1791 0.0789 ...
 $ X15: num  -0.4021 0.2564 0.0718 0.1876 -0.1029 ...
 $ X16: num  0.07605 0.00145 -0.06518 -0.06518 -0.11702 ...
 $ X17: num  -0.0312 -0.132 -0.1864 0.0749 0.0602 ...
 $ X18: num  -0.3779 -0.3208 -0.45 0.0143 -0.0165 ...
 $ X19: num  -0.481 -0.471 -0.128 -0.092 -0.199 ...
 $ X20: num  -0.131 -0.397 0.222 -0.232 -0.229 ...
 $ X21: num  -0.1184 -0.3074 0.0971 0.0465 -0.3394 ...
 $ X22: num  0.00306 0.00244 0.00119 0.00199 0.0038 ...
 $ X23: num  0.00401 0.00739 0.00116 0.00255 0.0024 ...
 $ X24: num  0.01949 0.01463 0.00858 0.00723 0.00249 ...
 $ X25: num  0.001811 0.00111 0.017956 0.004755 0.000928 ...
 $ X26: num  0.000783 0.001008 0.010507 0.001274 0.000897 ...
 $ X27: num  0.000223 0.002802 0.002532 0.000879 0.000628 ...
 $ X28: num  0.00181 0.00854 0.00138 0.0012 0.00118 ...
 $ X29: num  0.000748 0.009402 0.000552 0.005429 0.003076 ...
 $ X30: num  0.00037 0.003874 0.000348 0.005733 0.000591 ...
 $ X31: num  0.005889 0.002074 0.000545 0.005029 0.017033 ...
 $ X32: num  0.01402 0.01521 0.00413 0.0017 0.05498 ...
 $ X33: num  0.001227 0.001349 0.000677 0.00557 0.029241 ...
 $ X34: num  0.0077 0.00864 0.00621 0.00363 0.01612 ...
```

Fig.17 Extracted features and their data types

Reducing the number of features is a crucial decision to take. Considering features to be removed is generally based on subject knowledge and hence can affect the performance of the system. Next, a series of experiments are performed with this prepared dataset in order to analyze the important features.

CHAPTER 9

EXPERIMENTS

9.1 APPROACH 1 – WITH ALL EXTRACTED FEATURES

For this experiment, all the 34 features were considered. The data represented 2399 audio files. The steps of implementation are listed briefly (see Figure 18).

1. Using K-fold cross-validation method, the dataset is split into training and validation sets for the purpose of testing the model.
2. A classifier model is built using one of the classification algorithms and its parameters are observed. At this stage, tuning the values of parameters is optional.
3. The model is trained using the training data.
4. The trained model is evaluated using the validation set and the accuracy score is computed.
5. The model is tested and evaluation metrics such as precision, recall, and F1 scores are computed.

```
from sklearn.model_selection import cross_val_score
from sklearn.metrics import accuracy_score, f1_score, precision_score, recall_score, classification_report, confusion_matrix
from sklearn.svm import SVC

#splitting the data into 70% training and 30% testing
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=0)

#creating the SVM model
clf_svm = SVC(kernel='linear', C=1)

#training the model
clf_svm.fit(X_train, y_train)

#Accuracy score for the model
scores = cross_val_score(clf_svm, x, y, cv=5)
print ("SVM")
print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))

#Testing the model
y_pred = clf_svm.predict(X_test)

#Evaluation metrics - Precision, Recall and F1 scores
print("F1:",round(f1_score(y_test, y_pred, average="macro"),2))
print("Precision:",round(precision_score(y_test, y_pred, average="macro"),2))
print("Recall:",round(recall_score(y_test, y_pred, average="macro"),2))
```

```
SVM
Accuracy: 0.84 (+/- 0.03)
F1: 0.83
Precision: 0.83
Recall: 0.84
```

Fig.18 Implementation

The same experiment was repeated with various classification algorithms and the results were compared (see Table 2). The results had higher accuracy scores than expected. There was an average performance score of 75%. Besides the accuracy, the F1 score is the arithmetic mean of Precision and Recall, thereby making it a good metric for comparing the models. SVM has a good F1 score of 83%, making it a winning algorithm for this approach. It can also be observed that the tree-based algorithms have an improved performance with enhancements. The decision tree classifier has a score of 70%+, improved with the Random forest classifier with a score of 75%+, and further improved using the Gradient Decent classifier with a score of 77%+. The accuracy score of Logistic regression is around 80%. However, the other metrics

have an average score of 70%. The Naïve Bayes classifier has a score of 74%+ and KNN classifier has a score of 80%+.

Table.2 Results of approach-1 implementation

ALGORITHM	ACCURACY	PRECISION	RECALL	F1 SCORE
SVM	**84% (+/- 0.03)**	**83%**	**84%**	**83%**
Decision Tree	**74% (+/- 0.03)**	**71%**	**72%**	**71%**
KNN	**82% (+/- 0.05)**	**80%**	**80%**	**80%**
Logistic Regression	**80% (+/- 0.03)**	**72%**	**69%**	**70%**
Random Forest	**78% (+/- 0.04)**	**77%**	**75%**	**76%**
Gaussian Naïve Bayes	**76% (+/- 0.05)**	**74%**	**74%**	**74%**
Gradient Boosting Trees	**80% (+/- 0.03)**	**77%**	**77%**	**77%**

9.2 APPROACH 2 – WITH MFCC COEFFICIENTS

Among all the features retrieved from the audio signal, several researchers suggest that the MFCC values alone closely relate to the emotional tone of the audio. Studies suggest that using the MFCC features can reduce the dimensionality of the training set, and thereby take less computation time. This experiment repeats the same procedure as the previous but using only the MFCC values which are 13 dimensional.

Experiments similar to the first approach were carried out and the results are tabulated (see Table 3). The results of the second approach had scores lower than the first approach. The average accuracy of the models built using the second approach is 72%. KNN is the winning algorithm for this approach with a leading score of 80%+. Similar to the previous approach, there were improvements in the Decision Tree classifier with enhancements, with scores improving from 65% to 70%.

Table.3 Results of approach-2 implementation

ALGORITHM	ACCURACY	PRECISION	RECALL	F1 SCORE
SVM	**79% (+/- 0.03)**	**78%**	**77%**	**77%**
Decision Tree	**68% (+/- 0.03)**	**65%**	**65%**	**65%**
KNN	**84% (+/- 0.05)**	**80%**	**80%**	**80%**
Logistic Regression	**80% (+/- 0.03)**	**78%**	**77%**	**77%**
Random Forest	**71% (+/- 0.04)**	**70%**	**69%**	**70%**
Gaussian Naïve Bayes	**75% (+/- 0.05)**	**73%**	**73%**	**72%**
Gradient Boosting Trees	**73% (+/- 0.03)**	**72%**	**69%**	**70%**

9.3 APPROACH 3 – USING PCA FOR DECOMPOSITION

From the previous experiments, it can be learned that reducing the dimensions by cutting off the features, will reduce the performance of the model. This is because by cutting off features most of the information from the original dataset is not retained. Principal Component Analysis (PCA) is a

dimensionality reduction methodology in which the variation from the raw dataset can be specified to be retained. PCA has a pre-requisite that the data should be standardized before performing PCA on it. For this experiment, a scree plot was plotted (see Figure 19) to determine the optimal number of components to be retained. The elbow point was at 25 component with 95% information being retained. The data now is 25 dimensional. The steps of model building and evaluation was carried out next (see Figure 20).

1. The data is split into training and validation sets.
2. All the numeric values of the data are standardized in order to maintain a uniform distribution.
3. PCA is performed on the data with 95% information retain value, and the respective number of components are observed.
4. The data is transformed to contain the principal components.
5. The model is built and evaluated with the new data.

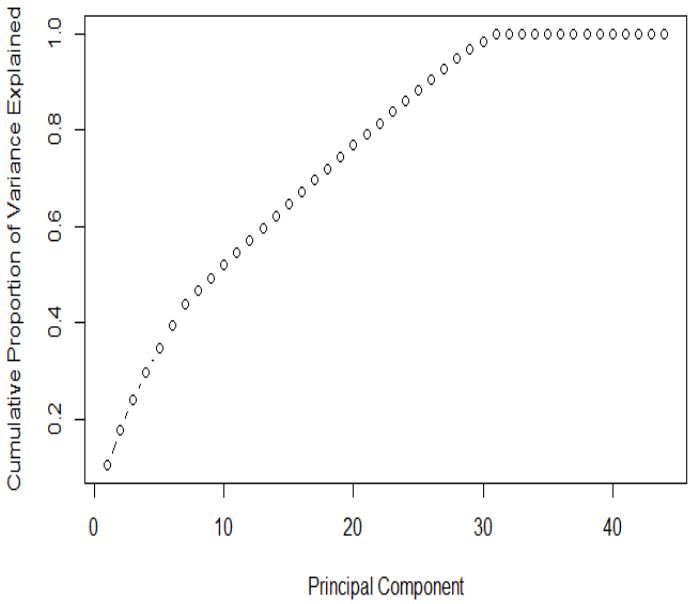

Fig.19 Scree plot

```
train_img, test_img, train_lbl, test_lbl = train_test_split( x, y, test_size=1/7.0, random_state=0)
```

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
# Fit on training set only.
scaler.fit(train_img)
# Apply transform to both the training set and the test set.
train_img = scaler.transform(train_img)
test_img = scaler.transform(test_img)
```

```
from sklearn.decomposition import PCA
# Make an instance of the Model
pca = PCA(.95)
```

```
pca.fit(train_img)
```

```
PCA(copy=True, iterated_power='auto', n_components=0.95, random_state=None,
  svd_solver='auto', tol=0.0, whiten=False)
```

```
print(pca.n_components_ )
```

```
25
```

```
train_img = pca.transform(train_img)
test_img = pca.transform(test_img)
```

Fig.20 PCA implementation

The implementation results of the third approach are tabulated (see Table 4) and the performance of the different classifiers are compared. The average score of this approach is 77%, which is an improvement over the other two approaches. The winning algorithm, similar to the first approach is SVM with a high score of 90%+. KNN classifier had scores closer to the winning algorithm of 87%. The Decision tree classifier improved from 68% to 71% using the Random Forest classifier, and to 75% with Gradient boosting trees. The logistic regression has an average score of 87%. Naïve Bayes has a score of 75%.

Table.4 Results of approach-3 implementation

ALGORITHM	ACCURACY	PRECISION	RECALL	F1 SCORE
SVM	**92% (+/- 0.03)**	**90%**	**90%**	**90%**
Decision Tree	**72% (+/- 0.03)**	**69%**	**68%**	**68%**
KNN	**89% (+/- 0.05)**	**88%**	**87%**	**87%**
Logistic Regression	**87% (+/- 0.04)**	**87%**	**86%**	**86%**
Random Forest	**74% (+/- 0.04)**	**74%**	**71%**	**72%**
Gaussian Naïve Bayes	**77% (+/- 0.05)**	**75%**	**75%**	**75%**
Gradient Boosting Trees	**81% (+/- 0.03)**	**76%**	**75%**	**75%**

9.4 COMPARISION OF THE RESULTS

The various algorithms performed differently with each approach of the implementation. Since accuracy is not always a good measure for evaluating the model, the F1 scores can be used for comparison. On comparing the F1 scores from the results of each approach (see Table 5 and Figure 21), led to useful conclusions. SVM, KNN, and Logistic Regression classifiers have a low performance in the second approach and an improved performance in the third approach, as compared to the first approach. The tree-based classifiers, such as Decision Tree, Random Forest, and Gradient Boosted trees performed the best for the first approach and has a low performance with the other approaches. The KNN classifier had a constant performance score for all the three approaches.

Table.5 Comparison of the results

ALGORITHM	APPROACH 1	APPROACH 2	APPROACH 3
SVM	**83%**	**77%**	**90%**
Decision Tree	**71%**	**65%**	**68%**
KNN	**80%**	**80%**	**87%**
Logistic Regression	**70%**	**77%**	**86%**
Random Forest	**76%**	**69%**	**72%**
Gaussian Naïve Bayes	**74%**	**73%**	**75%**
Gradient Boosting Trees	**77%**	**69%**	**75%**

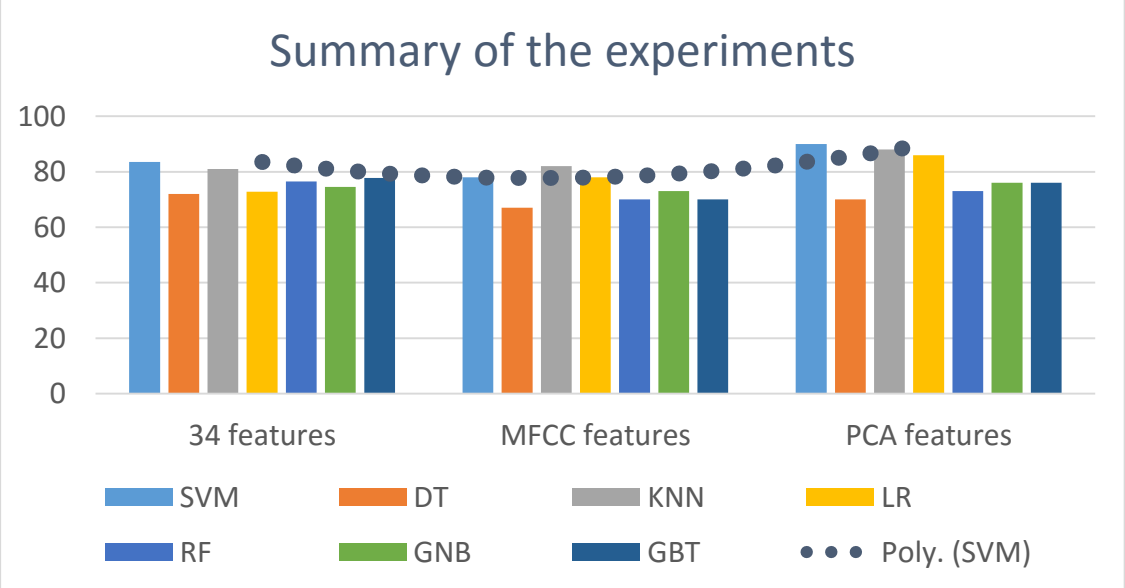

Fig.21 Summary of the experiments

9.5 IMPLEMENTATION ON THE KEEL DATASET

The third approach was found performing better than the other two approaches. Therefore the same implementation procedure was run on the KEEL dataset and the results are tabulated (see Table 6). The performance scores are lower than the TESS dataset, due to the smaller size of the training data in KEEL. The highest score of the KEEL dataset is by the SVM classifier with an average value of 65%, making it the winning algorithm. The performance of the different models is similar to the results of the TESS dataset.

Table.6 Results of KEEL dataset

ALGORITHM	ACCURACY	PRECISION	RECALL	F1 SCORE
SVM	**67% (+/- 0.03)**	**65%**	**66%**	**65%**
Decision Tree	**49% (+/- 0.03)**	**48%**	**49%**	**49%**
KNN	**55% (+/- 0.05)**	**55%**	**56%**	**55%**
Logistic Regression	**48% (+/- 0.04)**	**48%**	**48%**	**48%**
Random Forest	**51% (+/- 0.04)**	**50%**	**50%**	**50%**
Gaussian Naïve Bayes	**43% (+/- 0.05)**	**43%**	**42%**	**43%**
Gradient Boosting Trees	**51% (+/- 0.03)**	**50%**	**51%**	**50%**

CHAPTER 10

RESULTS

Several observations and conclusions can be derived from the results of the implementation. There is an overall improvement in the performance scores between the different approaches. The implementation following the first approach has a fair performance with a high score of 83% using the SVM algorithm, and the second approach worked well using the KNN algorithm for a high score of 80%, and the third approach had a 90% score using the SVM algorithm, which is the highest among all three approaches. The following observations can be made from the results:

Observation 1: Upon comparing of the results of the first and third approach, the SVM and KNN algorithm had improvements in the different performance metric scores. However, the Decision Tree, Random Forest and, Gradient Boosting Trees had diminishing scores. The Bayesian algorithm performed constantly between these approaches.

The improved scores of the SVM and KNN algorithm can be attributed to the dimensionality reduction used in the third approach. Reducing the dimensionality of the data increases the ratio of the size of the dataset to the number of dimensions, which reduces the bias of the classifier towards any particular class. Contrastingly, the performance of the tree-based algorithms improves with a larger feature set. This is because, the depth of the decision tree increases with adding more features, and thereby help making more accurate decisions. The Bayesian principle of the Naïve Bayes algorithm works on the prior probability value calculated for each data in the training set, which remains unchanged among the two approaches, and hence the scores remain unchanged.

Observation 2: The overall performance of the second approach was lower than the other two approaches.

This is because of the selective feature approach fails to contain most of the information from the speech signal, and can be concluded that using only the MFCC values alone cannot be a good measure to classify the emotional content of speech.

Observation 3: The classification report of the first approach (see Figure 22) shows that the misclassification is higher for the emotions Happy and Surprise.

This bias is due to the common properties of the features in these two categories. The dimensionality reduction step used in the third approach has greatly minimized this bias (see Figure 23).

On comparison to the baseline system by Chen et al. [8], the proposed system has improvements in the accuracy score (see Figure 24). The third experiment is the winning approach for the proposed methodology.

```
Classification Report
              precision    recall  f1-score   support

       Angry       0.80      0.81      0.80        48
     Disgust       0.80      0.94      0.86       112
        Fear       0.80      0.88      0.84        68
       Happy       0.89      0.81      0.85       135
     Neutral       0.88      0.91      0.89       118
         Sad       0.85      0.81      0.83       114
    Surprise       0.77      0.69      0.73       125

avg / total       0.83      0.83      0.83       720

Confusion Matrix
[[ 39   0   6   1   0   1   1]
 [  1 105   0   0   2   2   2]
 [  5   1  60   1   0   0   1]
 [  1   5   4 110   0   0  15]
 [  0   2   0   0 107   7   2]
 [  0   8   0   0   9  92   5]
 [  3  10   5  11   4   6  86]]
```

Fig.22 Classification report for approach-1 results

```
Classification Report
              precision    recall  f1-score   support

       Angry       0.92      0.92      0.92        24
     Disgust       0.88      0.94      0.91        52
        Fear       0.89      0.91      0.90        34
       Happy       0.98      0.94      0.96        66
     Neutral       0.95      0.98      0.96        53
         Sad       0.90      0.88      0.89        49
    Surprise       0.94      0.89      0.91        65

avg / total       0.93      0.92      0.92       343

Confusion Matrix
[[22  0  1  0  0  1  0]
 [ 0 49  1  0  0  1  1]
 [ 2  0 31  0  0  1  0]
 [ 0  1  1 62  0  0  2]
 [ 0  0  0  0 52  0  1]
 [ 0  3  1  0  2 43  0]
 [ 0  3  0  1  1  2 58]]
```

Fig.23 Classification report for approach-3 results

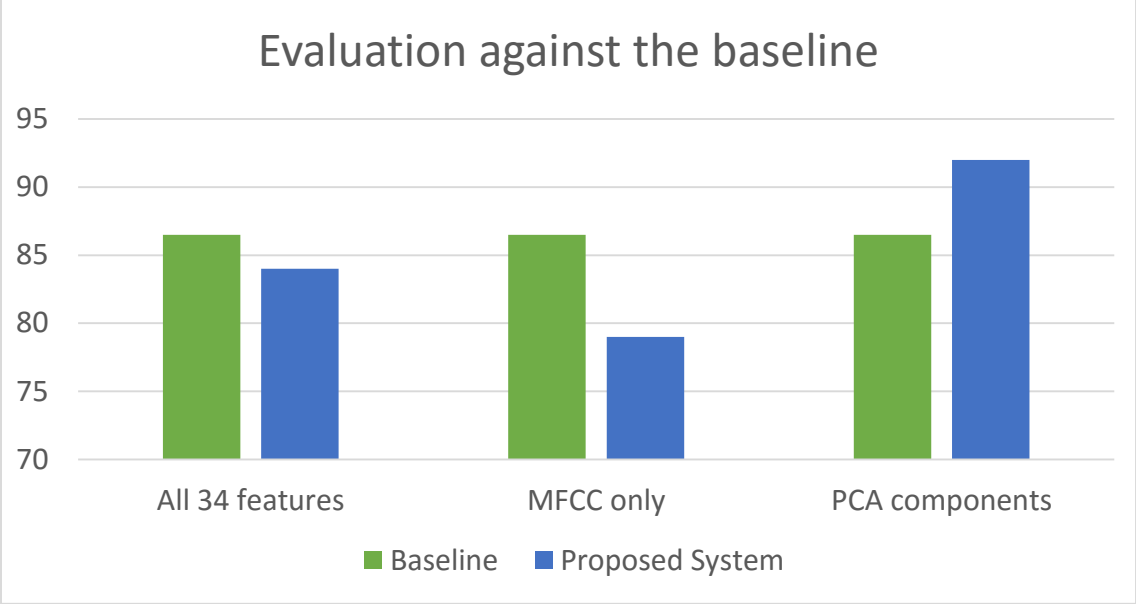

Fig.24 Evaluation against the baseline

CHAPTER 11

CONCLUSION AND FUTURE WORK

The emerging growth and development in the field of AI and machine learning have led to the new era of automation. Most of these automated devices work based on voice commands from the user. Many advantages can be built over the existing systems if besides recognizing the words, the machines could comprehend the emotion of the speaker (user). Some applications of a speech emotion detection system are computer-based tutorial applications, automated call center conversations, a diagnostic tool used for therapy and automatic translation system.

In this thesis, the steps of building a speech emotion detection system were discussed in detail and some experiments were carried out to understand the impact of each step. Initially, the limited number of publically available speech database made it challenging to implement a well-trained model. Next, several novel approaches to feature extraction had been proposed in the earlier works, and selecting the best approach included performing many experiments. Finally, the classifier selection involved learning about the strength and weakness of each classifying algorithm with respect to emotion recognition. At the end of the experimentation, it can be concluded that an integrated feature space will produce a better recognition rate when compared to a single feature.

For future advancements, the proposed project can be further modeled in terms of efficiency, accuracy, and usability. Additional to the emotions, the model can be extended to recognize feelings such as depression and mood changes. Such systems can be used by therapists to monitor the mood swings of the patients. A challenging product of creating machines with emotion is to incorporate a sarcasm detection system. Sarcasm detection is a more complex problem of emotion detection since sarcasm cannot be easily identified using only the words or tone of the speaker. A sentiment detection using

vocabulary, can be integrated with speech emotion detection to identify a possible sarcasm. Therefore, in the future, there would emerge many applications of a speech-based emotion recognition system.

References

[1] Soegaard, M. and Friis Dam, R. (2013). The Encyclopedia of Human-Computer Interaction. 2nd ed.

[2] Developer.amazon.com. (2018). Amazon Alexa. [online] Available at: https://developer.amazon.com/alexa

[3] Store.google.com. (2018). Google Home Tips & Tricks – Google Store. [online] Available at: https://store.google.com/product/google_home_learn

[4] Apple. (2018). iOS - Siri. [online] Available at: https://www.apple.com/ios/siri/

[5] The Official Samsung Galaxy Site. (2018). What is S Voice?. [online] Available at: http://www.samsung.com/global/galaxy/what-is/s-voice/ [Accessed 2 May 2018].

[6] Gartner.com. (2018). Gartner Says 8.4 Billion Connected. [online] Available at: https://www.gartner.com/newsroom/id/3598917.

[7] H. Cao, R. Verma, and A. Nenkova, “Speaker-sensitive emotion recognition via ranking: Studies on acted and spontaneous speech,” Comput. Speech Lang., vol. 28, no. 1, pp. 186–202, Jan. 2015.

[8] L. Chen, X. Mao, Y. Xue, and L. L. Cheng, “Speech emotion recognition: Features and classification models,” Digit. Signal Process., vol. 22, no. 6, pp. 1154–1160, Dec. 2012.

[9] T. L. Nwe, S. W. Foo, and L. C. De Silva, “Speech emotion recognition using hidden Markov models,” Speech Commun., vol. 41, no. 4, pp. 603–623, Nov. 2003.

[10] J. Rong, G. Li, and Y.-P. P. Chen, “Acoustic feature selection for automatic emotion recognition from speech,” Inf. Process. Manag., vol. 45, no. 3, pp. 315–328, May 2009.

[11] S. S. Narayanan, “Toward detecting emotions in spoken dialogs,” IEEE Trans. Speech Audio Process., vol. 13, no. 2, pp. 293–303, Mar. 2005.

[12] Dupuis, K. and Pichora-Fuller, M. (2010). [Collection] University of Toronto, Psychology Department, Toronto emotional speech set (TESS). Toronto.

[13] J. Alcalá-Fdez, A. Fernandez, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera. KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework. Journal of Multiple-Valued Logic and Soft Computing 17:2-3 (2011) 255-287.

[14] S, Khalid, T, Khalil and S, Nasreen. (2014). 2014 Science and Information Conference, A survey of feature selection and feature extraction techniques in machine learning. PP.372-378.

[15] Giannakopoulos, T. (2018). pyAudioAnalysis. [online] GitHub. Available at: https://github.com/tyiannak/pyAudioAnalysis.

[16] Practicalcryptography.com. (2018). Practical Cryptography. [online] Available at: http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/.

[17] Dsp.stackexchange.com. (2018). Framing an audio signal. [online] Available at: https://dsp.stackexchange.com/questions/27243/framing-an-audio-signal.

[18] Dataminingblog.com. (2018). Standardization vs. normalization | Data Mining Blog - www.dataminingblog.com. [online] Available at: http://www.dataminingblog.com/standardization-vs-normalization/.

[19] Statistics.laerd.com. (2018). Pearson Product-Moment Correlation - When you should run this test, the range of values the coefficient can take and how to measure strength of association.. [online] Available at: https://statistics.laerd.com/statistical-guides/pearson-correlation-coefficient-statistical-guide.php.

[20] Trevino, A. (2018). Introduction to K-means Clustering. [online] Datascience.com. Available at: https://www.datascience.com/blog/k-means-clustering.

[21] DeZyre. (2018). Principal Component Analysis Tutorial. [online] Available at: https://www.dezyre.com/data-science-in-python-tutorial/principal-component-analysis-tutorial.

[22] George Dallas. (2018). Principal Component Analysis 4 Dummies: Eigenvectors, Eigenvalues and Dimension Reduction. [online] Available at: https://georgemdallas.wordpress.com/2013/10/30/principal-component-analysis-4-dummies-eigenvectors-eigenvalues-and-dimension-reduction/.

[23] Analytics Vidhya. (2018). Simple Guide to Logistic Regression in R. [online] Available at: https://www.analyticsvidhya.com/blog/2015/11/beginners-guide-on-logistic-regression-in-r/.

[24] En.wikipedia.org. (2018). Logistic regression. [online] Available at: https://en.wikipedia.org/wiki/Logistic_regression.

[25] Ray, S. (2018). 6 Easy Steps to Learn Naive Bayes Algorithm (with code in Python). [online] Analytics Vidhya. Available at: https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/.

[26] En.wikipedia.org. (2018). Bayes' theorem. [online] Available at: https://en.wikipedia.org/wiki/Bayes%27_theorem.

[27] Ray, S. (2018). Understanding Support Vector Machine algorithm from examples (along with code). [online] Analytics Vidhya. Available at: https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/.

[28] Srivastava, T. (2018). Introduction to KNN, K-Nearest Neighbors : Simplified. [online] Analytics Vidhya. Available at: https://www.analyticsvidhya.com/blog/2014/10/introduction-k-neighbours-algorithm-clustering/.

[29] Ray, S. (2018). Decision Tree | Predictive Analytics. [online] Analytics Vidhya. Available at: https://www.analyticsvidhya.com/blog/2015/01/decision-tree-simplified/2/.

[30] (2018). How Random Forest Algorithm Works in Machine Learning. [online] Available at: https://medium.com/@Synced/how-random-forest-algorithm-works-in-machine-learning-3c0fe15b6674.

[31] Brownlee, J. (2018). A Gentle Introduction to the Gradient Boosting Algorithm for Machine Learning - Machine Learning Mastery. [online] Machine Learning Mastery. Available at: https://machinelearningmastery.com/gentle-introduction-gradient-boosting-algorithm-machine-learning/.

[32] Exsilio Blog. (2018). Accuracy, Precision, Recall & F1 Score: Interpretation of Performance Measures - Exsilio Blog. [online] Available at: http://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/.