A Project Report

on

Speech Emotion Recognition System Using Classification Algorithm in Machine Learning

Submitted in partial fulfillment of the
requirement for the award of the degree of

Bachelor of Technology
in
Computer Science and Engineering

GALGOTIAS UNIVERSITY
(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

Under The Supervision of

Mr. Arjun KP

ASSIS. PROFESSOR

Submitted By

SHAMBHAVEE SINGH (18SCSE1010264)

KUMAR SHUBHAM (18SCSE1010349)

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA
INDIA
May 2022

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA

CANDIDATE'S DECLARATION

I/We hereby certify that the work which is being presented in the project, entitled **"SPEECH EMOTION RECOGNITION SYSTEM USING CLASSIFICATION ALGORITHM IN MACHINE LEARNING ."** in partial fulfillment of the requirements for the award of the **Bachelor of Technology in Computer Science and Engineering** submitted in the **School of Computing Science and Engineering** of Galgotias University, Greater Noida, is an original work carried out during the period of **JANUARY-2022 to MAY-2022**, under the supervision of **Mr. Arjun KP, Assistant Professor, Department of Computer Science and Engineering**, of School of Computing Science and Engineering , Galgotias University, Greater Noida

The matter presented in the thesis/project/dissertation has not been submitted by me/us for the award of any other degree of this or any other places.

KUMAR SHUBHAM

18SCSE1010349

SHAMBHAVEE SINGH

18SCSE1010264

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Mr. ARJUN KP

ASSOCIATE PROFESSOR

CERTIFICATE

The Final Thesis/Project/ Dissertation Viva-Voce examination of **18SCSE1010349- KUMAR SHUBHAM, 18SCSE1010264- SHAMBHAVEE SINGH** has been held on_____and his/her work is recommended for the award of **BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING**.

Signature of Examiner **Signature of Supervisor**

Signature of Project Coordinator **Signature of Dean**

Date:

Place:

Table Of Contents

Abstract

Speech Emotion Recognition, contracted as SER, is the demonstration of endeavoring to perceive human feeling and full of feeling states from speech. This is gaining by the way that voice regularly reflects hidden feeling through tone and pitch. This is additionally the marvel that creatures like caninesand ponies utilize to have the option to comprehend human feeling. If we ever noticed, call centers employees never talk in the same manner, their way of pitching to the customers changes with customers. Now, this does happen with common people too, but how is this relevant to call centers? Here is the answer, the employees recognize customers' emotions from speech, so they can improve their service and convert more people. In this way, they are using speech emotion recognition. Speech Emotion Recognition is extreme since feelings are abstract and clarifying sound is testing.

List Of Figures

CHAPTER-1

Introduction

1.1 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). Aiming to develop and improve HCI methods is of paramount importance because HCI is the front-end of AI which millions of user experience. Some of the existing HCI methods involve communication through touch, movement, hand gestures, voice and facial gestures. 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 comprehendthe human's speech percept in order to accurately pick up the commands given to it. This fieldof 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 aspart of implementing a Speech Emotion Detection system.

The mode of speech is the quickest and the most characteristic strategy for correspondence between people. This reality has persuaded researchers to consider speech as a quick and effective strategy for communication amid humans as well as machines. Conversely, this necessitates the machine ought to have adequate insight to perceive human voices. Since, the late fifties, there has been enormous research over recognition of speech modes, which alludes to the development in changing over the human discourse into a grouping of words. However, in spite of an extraordinary advancement made in recognition of speech, we are

still a long way from having a characteristic association among man and machine because the machine does notcomprehend the emotional condition of the speaker.

This has presented a moderately recent research area, in particular, speech emotion recognition, which is characterized as extricating the emotional condition of a speaker from his or her discourse. It is expected that this sort of recognition can be utilized to extricate valuable semantics from speech, and thus, enhances the execution of systems for recognition of speech.

Human-Robot interface has been established hastily in a modern era where recognition in the emotion of speech playsa vital role. Such kinds of interacting robots, i.e., emotional robots have been extensively industrialized and were applied to a series of areas. One of the chief significant capabilities of emotional robots is recognition of speech which mainly comprises of emotion recognition, facial expression recognition as well as recognition of body language emotion.

Detection of the emotional condition is a significant feature of human-machine interface researches. The attributes utilized in recognition of emotion were consequent from alterations in facial mimics as well as speech indications. Emotion remains as a physiological response that ensues in situations like sadness, fear or happiness. Similarly, generation of speech is a physiological process where, thevariation in state of emotion will be imitated in speech as well as face. This transformation in speech will also be assign of identity, mental status and physical health of a person. Speech emotion recognition has been formulated as a pattern recognition problem that involves feature extraction and emotion classification/regression. Extracting meaningful and informative sets of features has attracted the emotion recognition community. The majority of advanced researches on computerized speech emotion acknowledgment focus on the paralinguistic channel of speech, with the extrication of acoustics descriptors, mainly, identified with prosody as well as spectral highlights. With the idea of perceiving the emotional condition of the speaker, recognizingparalinguistic features which don't rely upon the speaker or the lexical substance should be separated from speech. In general, there are two sorts of data in speech. They are linguistic data and paralinguistic data. The former always alludes to the context or the importance of speech while the latter comes to mean certain messages, for example, the feeling contained within the discourse.

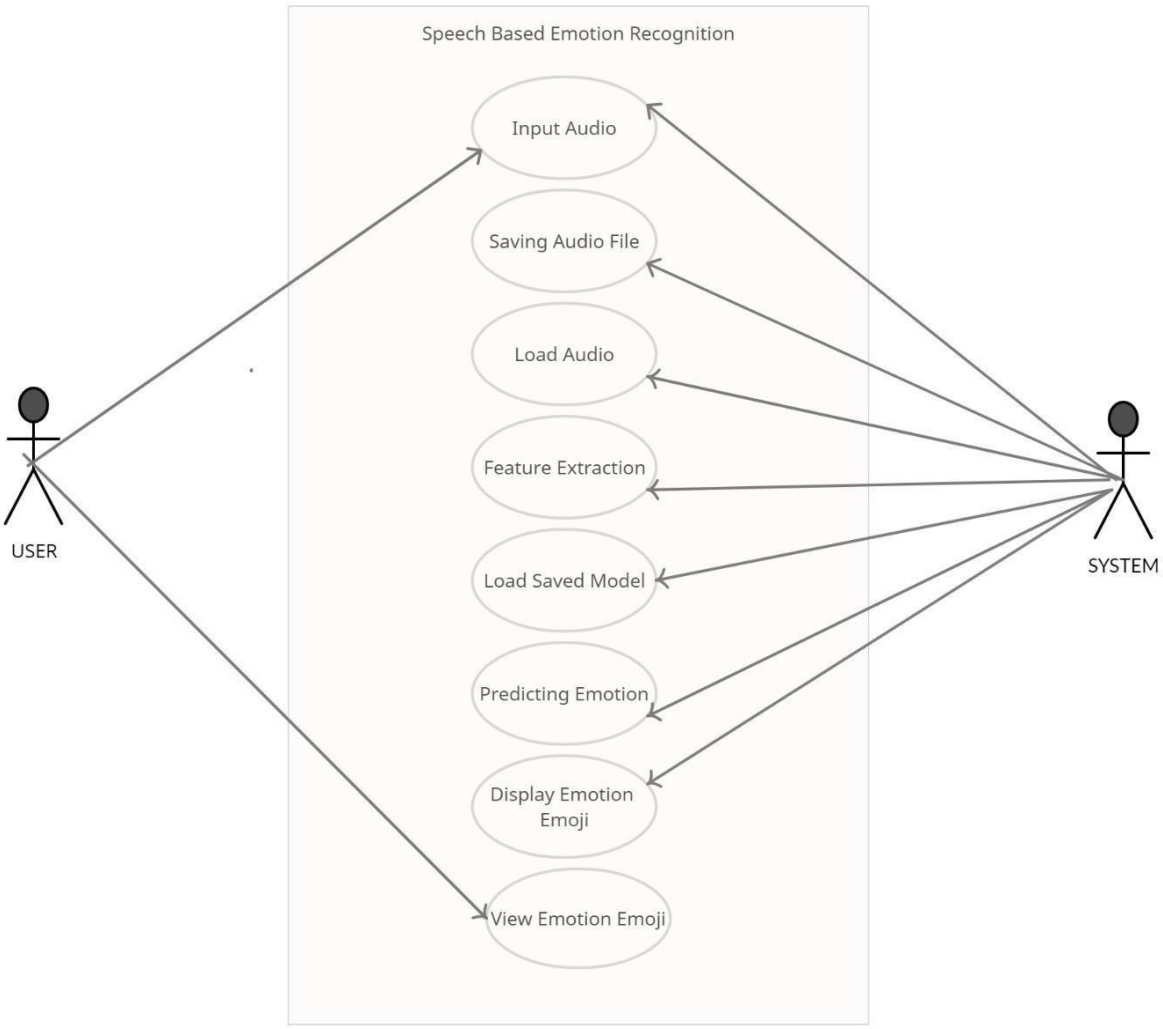


Fig1. Use Case Diagram

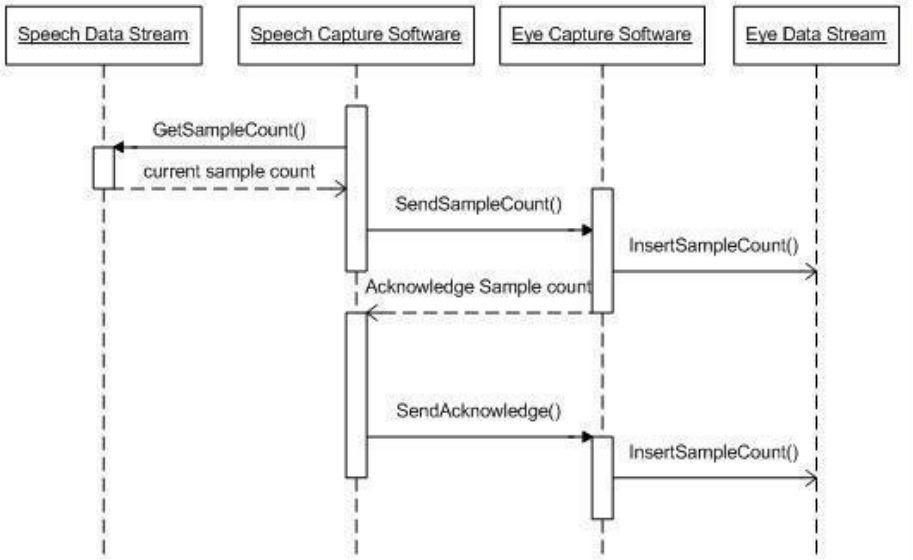


Fig 2. Sequence Diagram

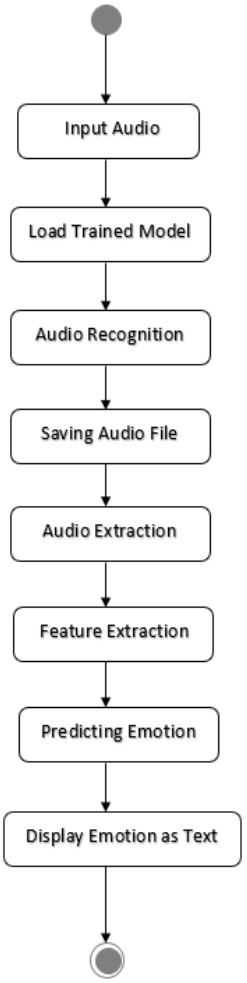


Fig3. Activity Diagram

1.2 Problem Formulation

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. 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. 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 founding lie detecting systems, criminal department analysis, and in humanoids.

1.2.1 Tools and Technology Used

- TENSORFLOW: It is an end-to-end open-source platform for machine learning, having comprehensive, flexible ecosystem of tools, libraries and community resources.
- LIBROSA: It is a python package for music and audio analysis. It provides the building blocks necessary to create music information retrieval systems.
- NUMPY: It is a Python library used for working with arrays, also has functions for working in domain of linear algebra, Fourier transform, and matrices.
- SOUNDFILE: It is an audio library based on LIBSNDFILE, CFFI and NumPy. It is used for reading and writing many different sampled sound file formats.
- SKLEARN: It is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python.
- PYAUDIO: It provides Python bindings for PORTAUDIO, the cross-platform audio I/O library. With PYAUDIO, you can easily use Python to play and record audio on a variety of platforms.
- TQDM: TQDM is a Python library that allows you to output a smart progress bar by wrapping around any iterable. A **TQDM** progress bar not only shows you how much time has elapsed, but also shows the estimated time remaining for the iterable.
- MATPLOTLIB: Matplotlib is a cross-platform, data visualization and graphical plotting library for Python and its numerical extension NumPy. As such, it offers a viable open source alternative to MATLAB. Developers can also use matplotlib's APIs (Application Programming Interfaces) to embed plots in GUI applications. A Python matplotlib script is structured so that a few lines of code are all that is required in most instances to generate a visual data plot. The matplotlib scripting layer overlays two APIs:
  - The pyplot API is a hierarchy of Python code objects topped by *matplotlib.pyplot*
  - An OO (Object-Oriented) API collection of objects that can be assembled with greater flexibility than pyplot. This API provides direct access to Matplotlib's backend layers.

CHAPTER 2
Literature Survey

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.

Over the last years, an excessive investigation has been completed to recognize emotions by using speech statistics. Cao et al proposed a ranking SVM method for synthesize information about emotion recognition to solve the problem of binary classification. This ranking method, instruct SVM algorithms for particular emotions, treating data from every utterer as a distinct query then mixed all predictions from rankers to apply multi-class prediction. Ranking SVM achieves two advantages, first, for training and testing steps in speaker- independent it obtains speaker specific data. Second, it considers the intuition that each speaker may express mixed of emotion to recognize the dominant emotion. Ranking approaches achieves substantial gain in terms of accuracy compare to conventional SVM in two public datasets of acted emotional speech, Berlin and LDC. In both acted data and the spontaneous data, which comprises neutral intense emotional utterances, ranking-based SVM achieved higher

accuracy in recognizing emotional utterances than conventional SVM methods. Unweight average (UA) or Balance accuracy achieved 44.4%.

Chen aimed to improve speech emotion recognition in speaker-independent with three level speech emotion recognition method. This method classify different emotions from coarse to fine then select appropriate feature by using Fisher rate. The output of Fisher rate is an input parameter for multi- level SVM based classifier. Furthermore principal component analysis (PCA) and artificial neural network (ANN) are employed to reduce the dimensionality and classification of four comparative experiments, respectively. Four comparative experiments include Fisher + SVM, PCA + SVM, Fisher + ANN and PCA + ANN. Consequence indicates in dimension reduction Fisher is better than PCA and for classification, SVM is more expansible than ANN for emotion recognition in speaker independent is. The recognition rates for three level are 86.5%, 68.5% and 50.2% separately in Beihang university database of emotional speech (BHUDES). Nweetal proposed a new system for emotion classification of utterance signals. The system employed a short time log frequency power coefficients (LFPC) and discrete HMM to characterize the speech signals and classifier respectively. This method classified the emotion into six differentcategories then used the private dataset to train and test the new system. In order to evaluate the performance of the proposed method, LFPC is compared with theMel-frequency Cepstral coefficients (MFCC) and linear prediction Cepstral coefficients (LPCC). Result demonstrate the average and best classification accuracy achieved 78% and 96% respectively. Furthermore, results expose that LFPC is a better option as feature for emotion classification than the standard features.

Wu et al proposed a new modulation spectral features (MSFs) human speech emotion recognition. Appropriate feature extracted from an auditory-inspired long-term Spectro temporal by utilizing a modulation filter bank and an auditory filter bank for speech decomposition. This method obtained acoustic frequency and temporal modulation frequency components for convey important data which is missing from traditional short-term spectral features. For classification process, SVM with radial basis function (RBF) are adopted. Berlin and Veraam Mittag (VAM) are employed to evaluate MSFs. In experimental result, the MSFs display capable performance in comparison with MFCC and perceptual linear prediction coefficients (PLPC). When MSFs utilized argument prosodic features, there is a considerable improvement in performance of recognition. Furthermore overall

recognition rate of 91.6% is achieved for classification. Rong et al presented an ensemble random forest to trees (ERF Trees) method with a high number of features for emotion recognition without referring any language or linguistic information remains an unclosed problem. This method is applied on small size of data with high number of features. In order to evaluate the proposed method an experiment results on a Chinese emotional speech dataset designates, this method achieved improvement on emotion recognition rate. Furthermore, ERF Trees performs better than popular dimension reduction methods such as PCA and multi-dimensional scaling (MDS) and recently developed ISO Map. The best accuracy with 16 features for female dataset achieved the maximum correct rate of 82.54%, while the worst is only 16% on 84 features with natural data set.

Wu et al proposed a fusion-based method for speech emotion recognition by employing multiple classifier and acoustic-prosodic (AP) features and semantic labels (SLs). In this fusion method, first AP features are extracted then three different types of base-level classifier include GMMs, SVMs, MLP and Meta decision tree (MDT) are used. The maximum entropy model (MaxEnt) in the semantic labels method are applied. MaxEnt modeled the association between emotion association rules (EARs) and emotion states in emotion recognition. In the final state to define the emotion recognition outcome, the integrated consequence from the SL-based and AS-based are utilized. The experimental result on private dataset shows the performance based on MDT archives 80%, SL- based recognition archives 80.92, and mixture of AP and SL archives 83.55%.

Narayanan proposed domain-specific emotion recognition by utilizing speech signals from call center application. Detecting negative and non-negative emotion (e.g. anger and happy) are the main focus of this research. Different types of information include acoustic, lexical, and discourse are used for emotion recognition. In addition, information-theoretic contents of emotional salience are presented to obtain data at emotion information at the language level. Both k-NN and linear discriminant classifier are used to work with different types of features. Experimental result confirms that the best results are achieved by combination of acoustic and language data. Outcomes demonstrates by combining three information sources instead of one source, classification accuracy increases by 40.7% for males and 36.4% for females. Compare to pervious work improvement range in accuracy is from 1.4% to 6.75% for male and 0.75% to 3.96% for female.

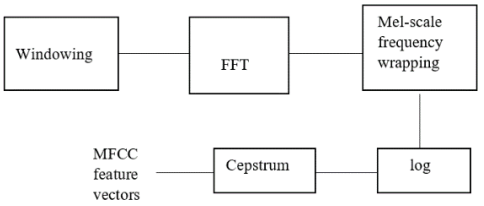
2.1 Important Terminology

Features of speech are of two types:

1. Qualitative (vocal: harsh, tense, breathy).
2. Spectral (LPC, MFCC, LFPC).

   Spectral features like MFCC, pitch and carry emotional information because of their smallest p- values.

   Mel Frequency Cepstrum Coefficient (MFCC) is widely used for speech and speech emotion recognition models for their high accuracy [7]. Mel frequency cepstrum is an illustration of short-term power spectrum of sound. In MFCC high frequency resolution with minimum noise can be achieved in low frequency region compared to high frequency region[7]. Linear prediction cepstrum coefficient (LPCC) is most effective representation of speech signal provides the information about the properties of particular signal of any individual person and due to accordance of different emotions this channel characteristic will get change, so emotions in speech can be extracted by using these features.



There are different types of classifiers that can be used to detect the emotions from human speech once the features are extracted. The different classifiers are described below:

- **SVM:** The Support Vector Machines (SVM) is an ML (Machine Learning) algorithm works on group problems. It can be used for both Linear and non- linear type of classification. In linear type of classification, two support vectors are taken and a hyper plane is drawn which is equidistant to the support vectors. This hyper plane is also called as decision boundary. Anything that falls above the hyperplane is regarded as one group and anything that falls below hyper plane is considered as another group. For non-linear classification of data, we take another plane namely z-plane and z plane represent the equation of a circle. Through z-plane which is parallel to x-plane we create a hyper plane which separates one group from another.

- **RF:** The Random Forest classifier (RF)is a group tree-based learning algorithm. The Random Forest Classifier is a group of decision trees from arbitrarily chosen subset of preparing set. It totals the votes from various choice trees to choose the last class of the test object. It is one of the most efficient learning calculations accessible. For some, databases, it delivers a profoundly exact result. It runs

effectively on huge databases. It can deal with a huge number of information without variable attenuation. It gives assessments of what factors that are significant in the characterization. It produces an inside fair gauge of the generalization error as the forest building advances.

- **MLP:** Multiple Layer Perception (MLP) has three layers namely input layer, hidden layer and output layer. The input layer is used for giving inputs. After allocating inputs they are matched to the neurons in hidden layer. The hidden layer is used to increase the accuracy of prediction. The more number of hidden layers the more accurate the model will be and then the hidden layer is mapped to the output which gives the prediction.

- **KERAS:** KERAS is a high-level neural network API which runs on tensorflow. It has two major models namely sequential model and functional model. Sequential model works likea linear stack of layers. It is used for building simple classification network and encoder decoder models. In this model we treat each layer as a object which leads to next layer. Functional model is a multi- input and multi-output model. Each complex node in this model gets divided into two or more branches. This is a model with shared layers and gives accurate results to 95% of the cases.

- **GNB:** Gaussian Navie Byes classifier is a powerful algorithm for predictive learning. It runs on Bayes theorem. GNB is used for classification and assumes feature for a normal distribution.

- **KNN ALGORITHM:** K-Nearest Neighbor is a simple algorithm which is used for prediction purpose. Here K=N, if k=n it chooses the n nearest neighbors and votes them. The neighbor that has the maximum number of votes is taken as the predicted value. It is an effective classifier if the training data is large and it is also robust to nosier databases. The only drawback of this classifier is that for different values of k we get different predictions.

2.2 RELATED WORKS

This section deals with the previously made works related with speech emotion recognition (SER), Pattern Recognition Neural Network (PRNN), K – Nearest Neighbour (KNN), Mel- Frequency Cepstrum Coefficients (MFCC) and Gray Level Co-occurrence Matrix (GLCM).

AnSERtechnique based on an enhanced brain emotional learning (BEL) model, which is stimulated by emotional handling mechanism of limbic structure in the brain was proposed.

However, BEL technique had its drawbacks and to overcome that, Genetic Algorithm (GA) was employed for updating the weights of BEL. The proposed technique had obtained a maximum average recognition accuracy of 90.28% in case of speaker-dependent speech emotion recognition while the highest average accuracy of 64.60% was obtained in case of speaker- independent speech emotion recognition.

In the same year, Sparse Hierarchical Coding (SC) approach was presented for emotion recognition systems. The proposed system comprised of motivated perceptual features (FPH) which resulted in a better and improved prediction of valence and arousal values compared to that of using only prosodic features (F200). Further, in the second stage of this technique, the proposed feature set was enhanced through an unsupervised feature learning method to automatically mine the non-linear relationship among the emotional speech data.

Similarly, a 1D and one 2D CNN LSTM networks, comprising off our LFLBs and one LSTM layer were proposed, which were built to learn both the local and global emotion-related features. These designed networks along with the support of CNN and LSTM were utilized in recognizing a speaker's emotional state. The outcomes of the two structures were verified over two standard databases. The consequences depicted that the two intended CNN LSTM networks might learn unique features as well as model high-level data of emotional information. When associated with other standard feature illustrations and techniques, 2D CNN LSTM network correspondingly possesses the edge over average accuracy.

An innovative category of a feature associated with prominence was proposed in 2018. This system, in conjunction with customary acoustic attributes, were utilized in classifying seven characteristic is similar emotional conditions. At that point, a consistency evaluation algorithm is introduced to approve the dependability of annotation data of this database. The results uncover that the explanation consistency on conspicuousness achieves over 60% on an average. At last, the proposed noticeable quality highlights were approved on CDESD through speaker-dependent as well as speaker-independent experiments with four regularly utilized classifiers.

In 2017, a novel approach was presented for empirically discovering feed-forward as well as recurrent neural network designs besides their modifications. The experiments conducted have revealed the methodology of how these architectural networks and their variants could beemployed for paralinguistic speech recognition, chiefly about emotion recognition. The system of Convolutional Neural Networks (ConvNets) displayed better results compared with other techniques.

The speech emotion recognition techniques were surveyed based on their features, classification

schemes, and databases which were considered to be the important aspects of design for an SER system. Numerous features that were utilized in

characterizing dissimilar emotions, the classification practices utilized in earlier investigations, as well as the significant design principles of emotional communication lists, were studied. Majority of classifiers used in previous researches were HMM, GMM, ANN, and SVM while it is also noted that the majority of the existing classification techniques do not model the temporal structure of the training data.

Various deep neural networks that are intended for acoustic modeling in speech recognition were discussed by Hintonet al., in 2012. It was expressed that, as an alternative to evaluate the fit of most of the Hidden Markov Model (HMM) for dealing with the temporal variability of speech and Gaussian mixture models (GMMs) is using a feed-forward neural network. Such neural networks were used as they take several frames of coefficients as input and produces posterior probabilities over HMM states as output. One of the major disadvantages of DNN is that it is much harder to train over a large cluster of machines with massive datasets.

Similarly, in 2013, the spiking neural networks for pattern recognition by introducing a new classof eSNN called dynamic eSNN which utilizes both rank-order learning and dynamic synapses tolearn in a fast, online mode was discussed by Kasabov, et.al. A new technique called deSSNwas also introduced that utilizes rank-order (RO) learning and SDSP spike-time learning in an unsupervised, supervised, or semi-supervised modes.

A comprehensive mean distance-based K-Nearest neighbor (KNN) classifier through multi-comprehensive mean expenses as well as the structured, comprehensive mean expanse which were based on representatives of comprehensive mean was proposed in 2019. In the proposed method, multi-local mean vectors of the given query sample in each class were calculated by adopting its class-specific K-Nearest neighbors. By the obtained k local mean vectors per class, the k generalized mean spaces were determined and were utilized in designing the definite nested comprehensive mean expanse.

A new ensemble technique for K-Nearest neighbor algorithm was also proposed in 2019. Here, a multimodal perturbation- based ensemble algorithm called Reduced Random Subspace- based Bagging (RRSB) was proposed which generates accurate but diverse component classifiers to improve the performance of ensemble classification. With the measured different k values, the proposed system of RRSB appears to be robust than other techniques. Also, the testing time of RRSB was much lesser than GA and PSO-based algorithms.

A novel version of K-Nearest neighbor algorithm was presented which not only uses similarity but also dependency in case of classification. A novel technique called dependent Neural

Network (d-NN). The advantages of this system are that the similarity of the query to these samples was weighed by angle going between the query and each of the samples and the query is labeled according to its nearest dependent neighbors that are determined by a joint function.

The outcomes have revealed that the accuracy and cost for computation of d-NN were much superior when compared to other popular machine learning techniques.

Kim (2010) and Dabas & Kumar (2014) reviewed some of the well-known methods used in numerous stages of a pattern recognition system using ANN as well as compared and identified their potential applications in the current scenario. The chief features of neural networks are that they possess the capability in learning compound nonlinear input-output associations, utilize successive techniques of training in addition to adapting themselves with data. The outcomes have revealed that the accuracy level of forecasting by the present dataset was much better as the application of ANN in every pattern recognition case provided better operation.

segmented in too long sequences for easy accessing of the proposed algorithm. This acts as the emotional speech database which is the primary requirement for any recognition network.

The emotional database comprised of six basic emotional classes such as,

- Angry
- Happy
- Sad
- Neutral
- Surprise
- Fear

DATA FLOW DIAGRAM

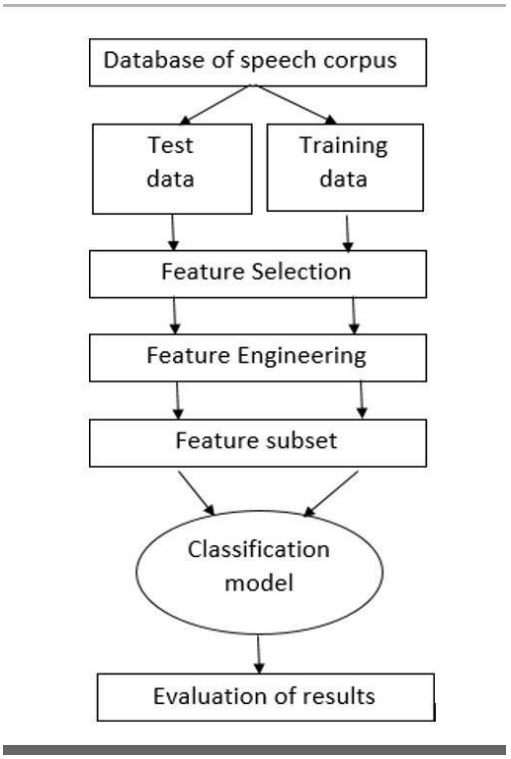


Fig 4. Data Flow Diagram

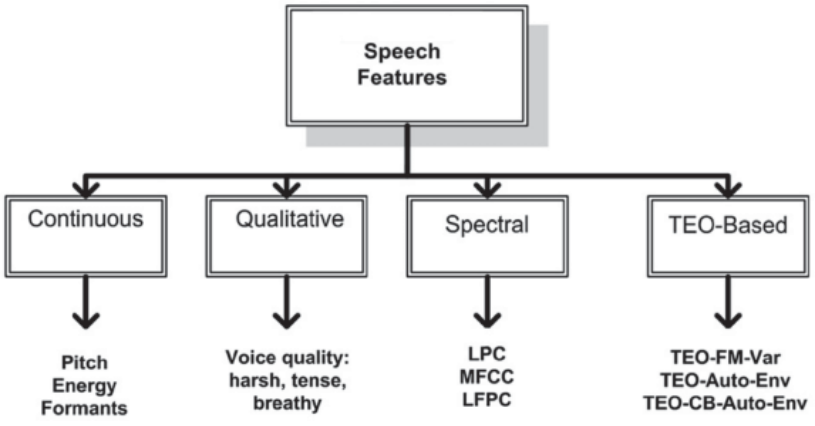


Fig 5. Feature Categories

CHAPTER 3
MODULES

1. LIBROSA:

Librosa is a Python package for music and audio analysis. Librosa is basically used when we work with audio data like in music generation (using LSTM's), Automatic Speech Recognition.

It provides the building blocks necessary to create the music information retrieval systems. Librosa helps to visualize the audio signals and also do the feature extractions in it using different signal processing techniques.

- ***Installation***

Librosa can be installed from the conda-forge channel. Open the Anaconda prompt and write:

conda install -c conda-forge librosa

```
C:\Users\Nikhil>conda install -c conda-forge librosa
Collecting package metadata (current_repodata.json): done
Solving environment: failed with initial frozen solve. Retrying with flexible solve.
Solving environment: failed with repodata from current_repodata.json, will retry with next repodata source.
Collecting package metadata (repodata.json): done
Solving environment: |
Warning: 4 possible package resolutions (only showing differing packages):
  - anaconda/win-64::ca-certificates-2019.8.28-0, anaconda/win-64::openssl-1.1.1d-he774522_2
  - anaconda/win-64::ca-certificates-2019.8.28-0, defaults/win-64::openssl-1.1.1d-he774522_2
  - anaconda/win-64::openssl-1.1.1d-he774522_2, defaults/win-64::ca-certificates-2019.8.28-0
  - defaults/win-64::ca-certificates-2019.8.28-0, defaults/win-64::openssl-1.1.1d-he774522done

## Package Plan ##

  environment location: C:\Users\Nikhil\Anaconda3

  added / updated specs:
    - librosa


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    audioread-2.1.8            |          py37_1          32 KB  conda-forge
    certifi-2019.9.11          |          py37_0         147 KB  conda-forge
    conda-4.8.2                |          py37_0         3.0 MB  conda-forge
    librosa-0.6.3              |            py_0         1.5 MB  conda-forge
    resampy-0.2.2              |            py_0         332 KB  conda-forge
```

2. SOUNDFILE:

SoundFile is an audio library based on libsndfile, CFFI and NumPy. SoundFile can read and write sound files. File reading/writing is supported through libsndfile, which is a free, cross-platform, open-source (LGPL) library for reading and writing many different sampled sound file formats that runs on many platforms including Windows, OS X, and Unix. It is accessed through CFFI, which is a foreign function interface for Python calling C code. CFFI is supported for CPython 2.6+, 3.x and PyPy 2.0+. SoundFile represents audio data as NumPy arrays.

3. PYAUDIO:

PyAudio provides Python bindings for PortAudio, the cross-platform audio I/O library. With PyAudio, you can easily use Python to play and record audio on a variety of platforms. PyAudio is inspired by:

1. pyPortAudio/fastaudio: Python bindings for PortAudio v18 API.2.tkSnack: cross-platform

sound toolkit for Tcl/Tk and Python.

To use PyAudio, first instantiate PyAudio using **pyaudio.PyAudio()** which sets up the portaudio system.

To record or play audio, open a stream on the desired device with the desired audio parameters using **pyaudio.PyAudio.open()**. This sets up a **pyaudio.Stream** to play or record audio.

Play audio by writing audio data to the stream using **pyaudio.Stream.write()**, or read audio data from the stream using **pyaudio.Stream.read()**.

Note that in "blocking mode", each **pyaudio.Stream.write()** or **pyaudio.Stream.read()** blocks until all the given/requested frames have been played/recorded. Alternatively, to generate audio data on the fly or immediately process recorded audio data, use the "callback mode" outlined below.

Use **pyaudio.Stream.stop_stream()** to pause playing/recording, and **pyaudio. Stream.close()** to terminate the stream. Finally, terminate the portaudio session using **pyaudio.PyAudio.terminate()**

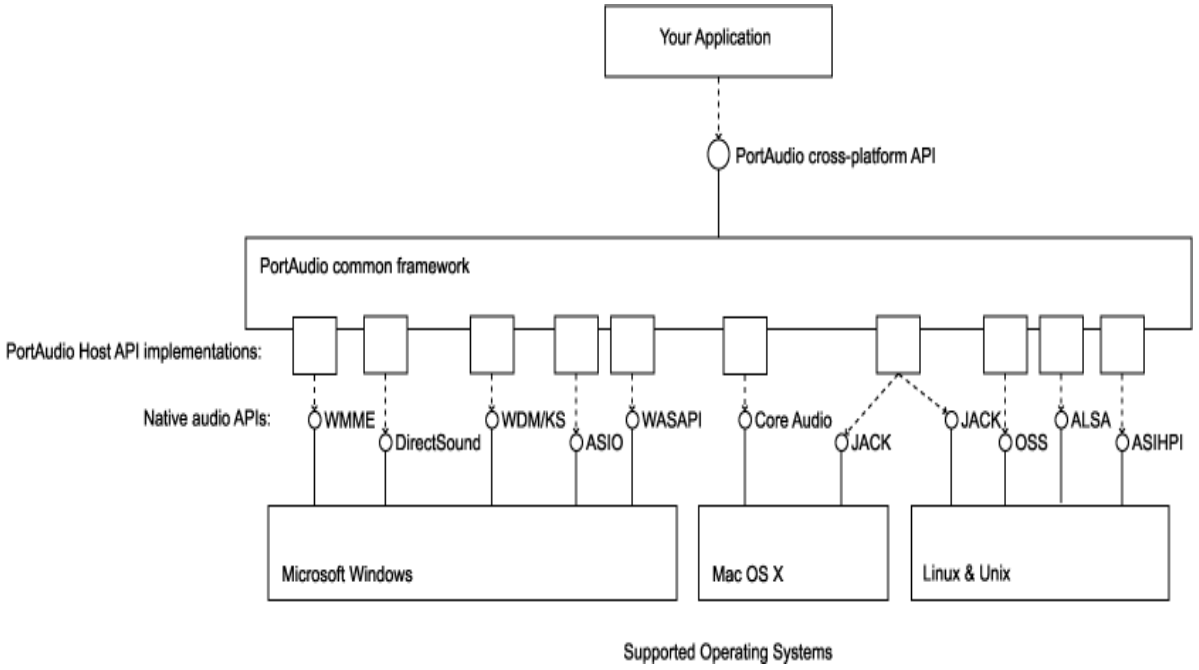
Installing PyAudio

pip install PyAudio-0.2.11-cp37-cp37m-win_amd64.whl

Importing PyAudio

import pyaudio

Exploring Audio Input and Output Devices



4. NUMPY:

NumPy is a Python library used for working with arrays.It also has functions for working in domain of linear algebra, fourier transform, and matrices.NumPy was created in 2005 by Travis Oliphant. It is an open source project and you can use it freely.NumPy stands for Numerical Python.

NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.

At the core of the NumPy package, is the *ndarray* object. This encapsulates *n*-dimensional arrays of homogeneous data types, with many operations being performed in compiled code for performance. There are several important differences between NumPy arrays and the standard Python sequences:

- NumPy arrays have a fixed size at creation, unlike Python lists (which can grow dynamically). Changing the size of an *ndarray* will create a new array and delete the original.

- The elements in a NumPy array are all required to be of the same data type, and thus will be the same size in memory. The exception: one can have arrays of (Python, including NumPy) objects, thereby allowing for arrays of different sized elements.

- NumPy arrays facilitate advanced mathematical and other types of operations on large numbers of data. Typically, such operations are executed more efficiently and with less code than is possible using Python's built-in sequences.

- A growing plethora of scientific and mathematical Python-based packages are using NumPy arrays; though these typically support Python-sequence input, they convert such input to NumPy arrays prior to processing, and they often output NumPy arrays. In other words, in order to efficiently use much (perhaps even most) of today's scientific/mathematical Python-based software, just knowing how to use Python's built-in sequence types is insufficient - one also needs to know how to use NumPy arrays.

Why is NumPy Fast?

Vectorization describes the absence of any explicit looping, indexing, etc., in the code - these things are taking place, of course, just "behind the scenes" in optimized, pre-compiled C code. Vectorized code has many advantages, among which are:

- vectorized code is more concise and easier to read
- fewer lines of code generally means fewer bugs
- the code more closely resembles standard mathematical notation (making it easier, typically,to correctly code mathematical constructs)
- vectorization results in more "Pythonic" code. Without vectorization, our code would be littered with inefficient and difficult to read for loops.

Broadcasting is the term used to describe the implicit element-by-element behavior of operations; generally speaking, in NumPy all operations, not just arithmetic operations, but logical, bit-wise, functional, etc., behave in this implicit element-by-element fashion, i.e., they broadcast. Moreover, in the example above, a and b could be multidimensional arrays of the same shape, or a scalar and an array, or even two arrays of with different shapes, provided that the smaller array is "expandable" to the shape of the larger in such a way that the resulting broadcast is unambiguous. For detailed "rules" of broadcasting see *basics.broadcasting*.

5. TENSORFLOW:

TensorFlow is Google Brain's second-generation system. Version 1.0.0 was released on February 11, 2017. While the reference implementation runs on single devices, TensorFlow can run on multiple CPUs and GPUs (with optional CUDA and SYCL extensions for general-purpose computing on graphics processing units).[17] TensorFlow is available on 64-bit Linux, macOS, Windows, and mobile computing platforms including Android and iOS. Its flexible architecture allows for the easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs), and from desktops to clusters of servers to mobile and edge devices. TensorFlow computations are expressed as stateful dataflow graphs. The name TensorFlow derives from the operations that such neural networks perform on multidimensional data arrays, which are referred to as *tensors*. During the Google I/O Conference in June 2016, Jeff Dean stated that 1,500 repositories on GitHub mentioned TensorFlow, of which only 5 were from Google.

- **Features:**

Auto Differentiation

Auto-Differentiation is the process of automatically calculating the gradient vector of a model with respect to each of its parameters. With this feature, TensorFlow can automatically compute the gradients for the parameters in a model, which is useful to algorithms such as backpropagation which require gradients to optimize performance. To do so, the framework must keep track of the order of operations done to the input Tensors in a model, and then compute the gradients with respect to the appropriate parameters.

Eager execution

TensorFlow includes an “eager execution” mode, which means that operations are evaluated immediately as opposed to being added to a computational graph which is executed later. Code executed eagerly can be examined step-by step-through a debugger, since data is augmented at each line of code rather than later in a computational graph. This execution paradigm is considered to be easier to debug because of its step-by-step transparency.

Distribute

In both eager and graph executions, TensorFlow provides an API for distributing computation across multiple devices with various distribution strategies. This distributed computing can often speed up the execution of training and evaluating of TensorFlow models and is a common practice in the field of AI.

Losses

To train and assess models, TensorFlow provides a set of loss functions (also known as cost functions). Some popular examples include mean squared error (MSE) and binary cross

entropy (BCE). These loss functions compute the “error” or “difference” between a model’s output and the expected output (more broadly, the difference between two tensors). For different datasets and models, different losses are used to prioritize certain aspects of performance.

Metrics

In order to assess the performance of machine learning models, TensorFlow gives API access to commonly used metrics. Examples include various accuracy metrics (binary, categorical, sparse categorical) along with other metrics such as Precision, Recall, and Intersection-over-Union (IOU).

TF.nn

TensorFlow.nn is a module for executing primitive neural network operations on models.[38] Some of these operations include variations of convolutions (1/2/3D, Atrous, depthwise), activation functions (Softmax, RELU, GELU, Sigmoid, etc.) and their variations, and other Tensor operations (max-pooling, bias-add, etc.).

Optimizers

TensorFlow offers a set of optimizers for training neural networks, including ADAM, ADAGRAD, and Stochastic Gradient Descent (SGD). When training a model, different optimizers offer different modes of parameter tuning, often affecting a model’s convergence and performance.

6. TQDM:

TQDM is a Python library that allows you to output a smart progress bar by wrapping around any iterable. A **TQDM** progress bar not only shows you how much time has elapsed, but also shows the estimated time remaining for the iterable.

Installing and importing TQDM

Since **tqdm** is part of the Python Package Index (PyPI), it can be installed using the **pip install tqdm** command.

7. MATPLOTLIB:

Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It was introduced by John Hunter in the year 2002. One of the greatest benefits

of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc.

Installation:

Windows, Linux and macOS distributions have matplotlib and most of its dependencies as wheel packages. Run the following command to install matplotlib package

python -mpip install -U matplotlib

Importing matplotlib:

from matplotlib import pyplot as plt

or

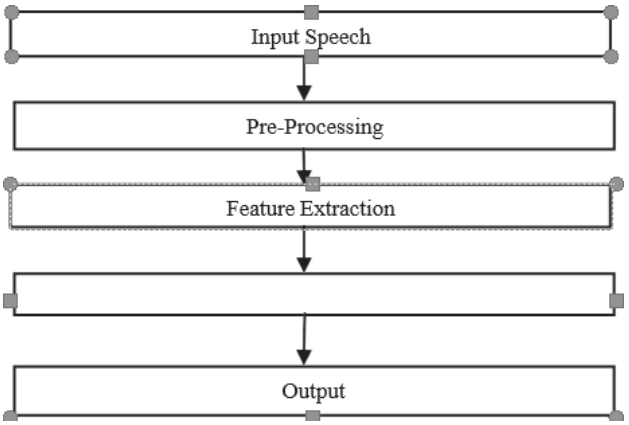
import matplotlib.pyplot as plt

Basic plots in Matplotlib:

Matplotlib comes with a wide variety of plots. Plots helps to understand trends, patterns, and to make correlations. They're typically instruments for reasoning about quantitative information. For example, Line Plot, Bar Plot, Histogram etc.

CHAPTER 4
METHODOLOGY

In any speech emotion recognition methodology, the outcomes were greatly influenced by pre-processing, feature extraction as well as classification. To obtain better results, methodology involving wiener filter, the cascaded system of PRNN and K-NN and a hybrid system of MFCC and GLCM were utilized in all these three phases respectively. The Wiener filter was used, as this type of filter controls the output error and is easy in construction. In case of a cascaded system of PRNN and K-NN, the PRNN recognizes the pattern of the signal produced through the emotional waves. Also, the K-NN approach determines the more likely conceivable nearest pattern regarding the signal. Similarly, a hybrid system of MFCC and GLCM was made as the power spectrum of sound produced by MFCC was greatly used by GLCM to produce a better grayscale matrix for classification.



B. Input Speech

The input or source to this algorithm is a speech which is to be recognized. In this section, the raw waveform has been segmented in too long sequences for easy accessing of the proposed algorithm. This acts as the emotional speech database which is the primary requirement for any recognition network.

The emotional database comprised of six basic emotional classes such as,

- Angry
- Happy
- Sad
- Neutral
- Surprise
- Fear

These samples from the emotional database are then used as input sources. Emotional Prosody

Speech as well as Transcripts were established by Linguistic Data Consortium that comprises of audio recordings along with corresponding transcripts, collected over an eight-month period in 2000-2001and were intended for supporting research in emotional prosody. There are 30 data files with 15 recordings in sphere format and their transcripts.

The sphere files were encoded in two-channel interleaved 16- bit PCM, high-byte-first (big- endian) format, for a total of 2,912,067,980 bytes (2777 Mbytes) or nine hours of sphere data. The utterances were recorded directly into WAVES+ data files, on two channels with a sampling rate of 22.05K. The two microphones used were a stand-mounted boom ShureSN94 and a headset Seinnheiser HMD 410.

The original session recordings were provided in their entirety, including informal chit-chat and discussion between each emotion category elicitation task. Time alignment was limited to utterances within the formal elicitation tasks and miscellaneous regions have been marked as such.

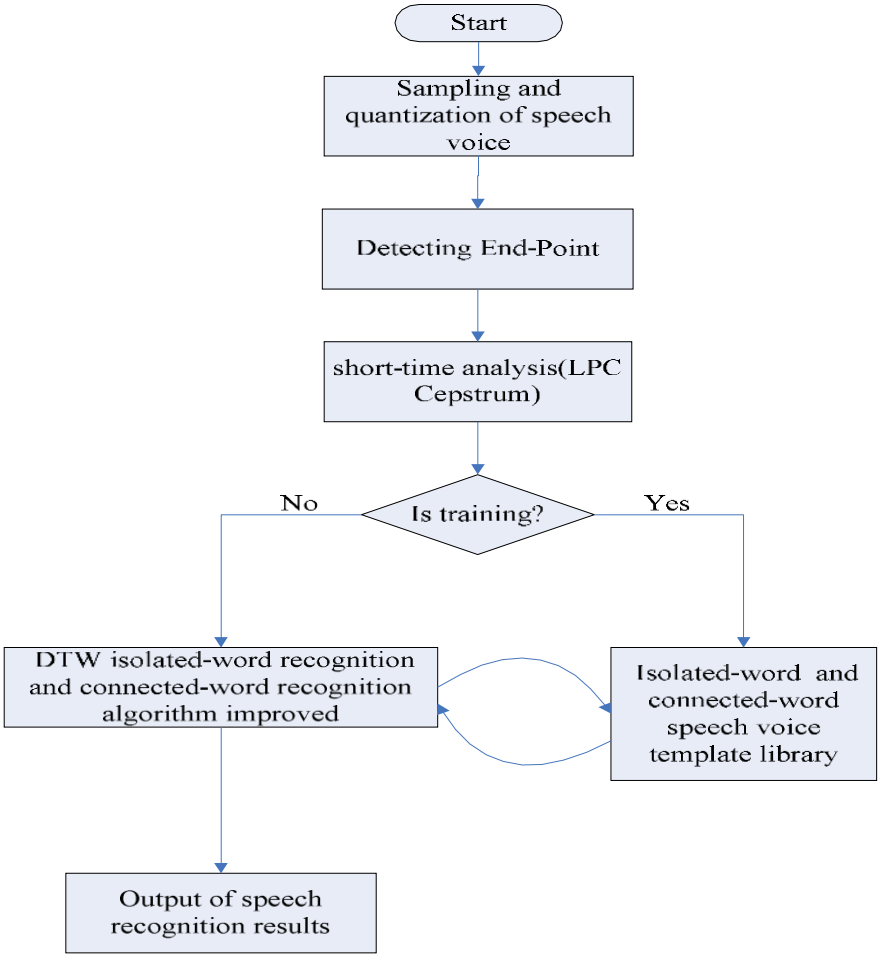
C. Pre-Processing

The signals obtained from the input source are first pre- processed to make it more compatible, noise-free and suitable for feature extraction. In such systems utilizing arbitrarily spoken phrases, always there exists a likelihood that spoken term is generally preceded as well as succeeded by silence. Removal of silence is the primary stage of pre-processing.

In this proposed system, the pre-processing is carried out using a Wiener filter. Wiener filter provides the minimum mean squared error regarding coefficient vector, and hence this filter is an optimum filter to use in pre-processing of speech recognition. It is used to reduce the noise from speech signal. The general equation of Wiener filter for minimization of mean squared errorcan be expressed as follows.

D. Software design

The figure shows the Simplified system flowchart, and the flowchart is explained in detail as follows.



The sampling frequency of speech voice is 8KHZ, and the quantization precision is 8bit. Small sampling frequency and small quantization precision are picked because embedded CPU usuallyrun slowly and cheap A/D chip work mostly at 8bit. After sampling and quantization, End- Point-Detection algorithm is invoked. The product of zero-crossing rate and average energyis used to detect End-Point, and the changeable windows length is used to get short-time speechvoice data. The initial window length is set at 50ms and the window displacement is set at 20ms.The window length is set at 10ms and the window displacement is set at 10ms after the End- Point is found. The window length is set at 20ms and the window displacement is set at 5ms after the End-Point is confirmed. The Hamming windows function is used to reduce margin abnormality. After the End-Point-Detection is finished, the Durbin algorithm is used to carry out LPC Cepstrum analysis which gets speech voice characters while rectangular window is sliding. After the speech voice sequence is got, the word data translated from the voice sequenceis written into the template library if training is selected, and DTW algorithm is used to match pattern between template and voice sequence in order to get real speech word and emotion.

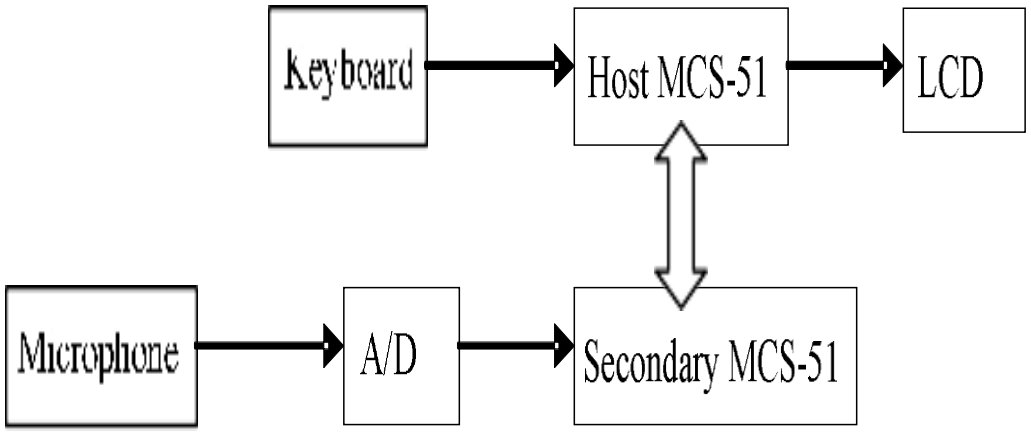
E. Speech voice input interface design of embeded system

The figure 5 shows the embedded system hardware chart and is explained in detail as follows.

MCS-51 series 8031 single-chip microprocessor is used as CPU of the system, and it contains memory and input/output interface also. The 8031 is a popular and one of the cheapest single-chip micro-processor which is suitable to make quality system at low cost.

ADC0816 is used as ADC with an 8-bit analog-to-digital converter, 16-channel multiplexer and microprocessor compatible control logic.

The figure shows main circuit chart of speech emotion recognition interface. The 8031interrupt service routine includes codes which implement all steps of speech emotion recognition. A write instruction is used to startup ADC. The ADC0816 sends EOC to INT1 of the 8031 to invoke the interrupt service routine. CLK of the ADC0816 is 1/6 of the 8031master clock frequency.



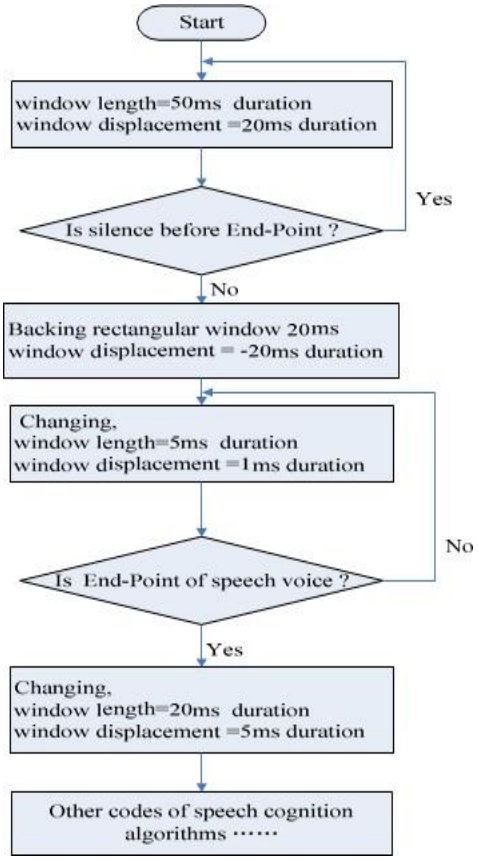
F. Improvement of End-Point-Detection algorithm

End-Point-Detection algorithm is used to measure the noise power and to automatically initiate recording of a spoken word. The End-Point-Detection sensitivity is one of main disadvantages in DTWalgorithm.

There are two methods to solve this problem. One is on DTW algorithm itself such as reducing sensitivity of End- Point-Detection with broadening End-Point, and the other is beyond DTW algorithm such as being accurate on End-Point- Detection. We study on rectangular window and window length,and propose an algorithm being more accurate on End-Point-Detection.

While End-Point of acoustic voice is detected, short-time average energy or zero-crossing rate multiplying average signal magnitude is often used. End-Point-Detection of acoustic voices more accurate obviously if window length is small, while speech recognition system will run slowly and there are some recognition errors produced by which short-time noise pulses are

sometime treated as speech voice. Speech recognition system will run quickly and short-time noise pulses is skipped if window length is large, while End-Point-Detection is inaccurate and the inaccuracy is deadly because of sensitivity of End-Point-Detection for DTW algorithm. These disadvantages are avoided if window length is medium, but these advantages would disappear. We propose an approach detecting accurately End-Point as well as avoiding the disadvantage by reason of too small window length. We call the approach "End-Point-Detectionalgorithm with change able window length" (see Figure 1). The algorithm is applied to the speechrecognition device developed by us and has an expected effective.



4.1 Implementation of Code:

Prerequisites

- import librosa
- import soundfile
- import os, glob, pickle
- import numpy as np
- from sklearn.model_selection import train_test_split

- from sklearn.neural_network import MLPClassifier
- from sklearn.metrics import accuracy_score

2. Define a function extract_feature to extract the mfcc, chroma, and mel features from a sound file. This function takes 4 parameters- the file name and three Boolean parameters for the three features:

- **mfcc:** Mel Frequency Cepstral Coefficient a sound
- **chroma:** Pertains to the 12 different pitch classes
- **mel:** Mel Spectrogram Frequency

Open the sound file with soundfile.SoundFile using with-as so it's automatically closed once we're done. Read from it and call it X. Also, get the sample rate. If chroma is True, get the Short-Time Fourier Transform of X.

, represents the short-term power spectrum of

```
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. An overview of data

Let result be an empty numpy array. Now, for each feature of the three, if it exists, make a call to the corresponding function from librosa.feature (eg- librosa.feature.mfcc for mfcc), and get the mean value. Call the function hstack() from numpy with result and the feature value, and store this in result. hstack() stacks arrays in sequence horizontally (in a columnar fashion). Then, return the result

#DataFlair - Extract features (mfcc, chroma, mel) from a sound filedef extract_feature(file_name,

```
mfcc, chroma, mel):
  with soundfile.SoundFile(file_name) as sound_file:
      X = sound_file.read(dtype="float32")
      sample_rate=sound_file.samplerate
      if chroma:
          stft=np.abs(librosa.stft(X))
      result=np.array([])
      if mfcc:
```

```
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. 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).

```
In [12]: import seaborn as sns
         sns.boxplot(x="36", y="2", data=data)

Out[12]: <matplotlib.axes._subplots.AxesSubplot at 0x5efd3e34a8>
```

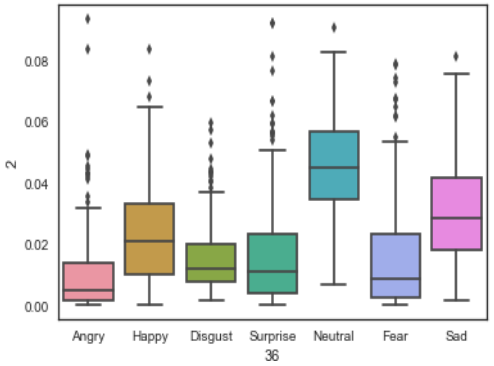


Fig. Box plot analysis of feature values

```
 mfccs=np.mean(librosa.feature.mfcc(y=X, sr=sample_rate, n_mfcc=40).T, axis=0)
 result=np.hstack((result, mfccs))
      if chroma:
         chroma=np.mean(librosa.feature.chroma_stft(S=stft, sr=sample_rate).T,axis=0)
         result=np.hstack((result, chroma))
if mel:
         mel=np.mean(librosa.feature.melspectrogram(X, sr=sample_rate).T,axis=0)
         result=np.hstack((result, mel))
return result
```

```
[2]: #DataFlair - Extract features (mfcc, chroma, mel) from a sound file
     def extract_feature(file_name, mfcc, chroma, mel):
         with soundfile.SoundFile(file_name) as sound_file:
             X = sound_file.read(dtype="float32")
             sample_rate=sound_file.samplerate
             if chroma:
                 stft=np.abs(librosa.stft(X))
             result=np.array([])
             if mfcc:
                 mfccs=np.mean(librosa.feature.mfcc(y=X, sr=sample_rate, n_mfcc=40).T, axis=0)
                 result=np.hstack((result, mfccs))
             if chroma:
                 chroma=np.mean(librosa.feature.chroma_stft(S=stft, sr=sample_rate).T,axis=0)
                 result=np.hstack((result, chroma))
             if mel:
                 mel=np.mean(librosa.feature.melspectrogram(X, sr=sample_rate).T,axis=0)
                 result=np.hstack((result, mel))
         return result
```

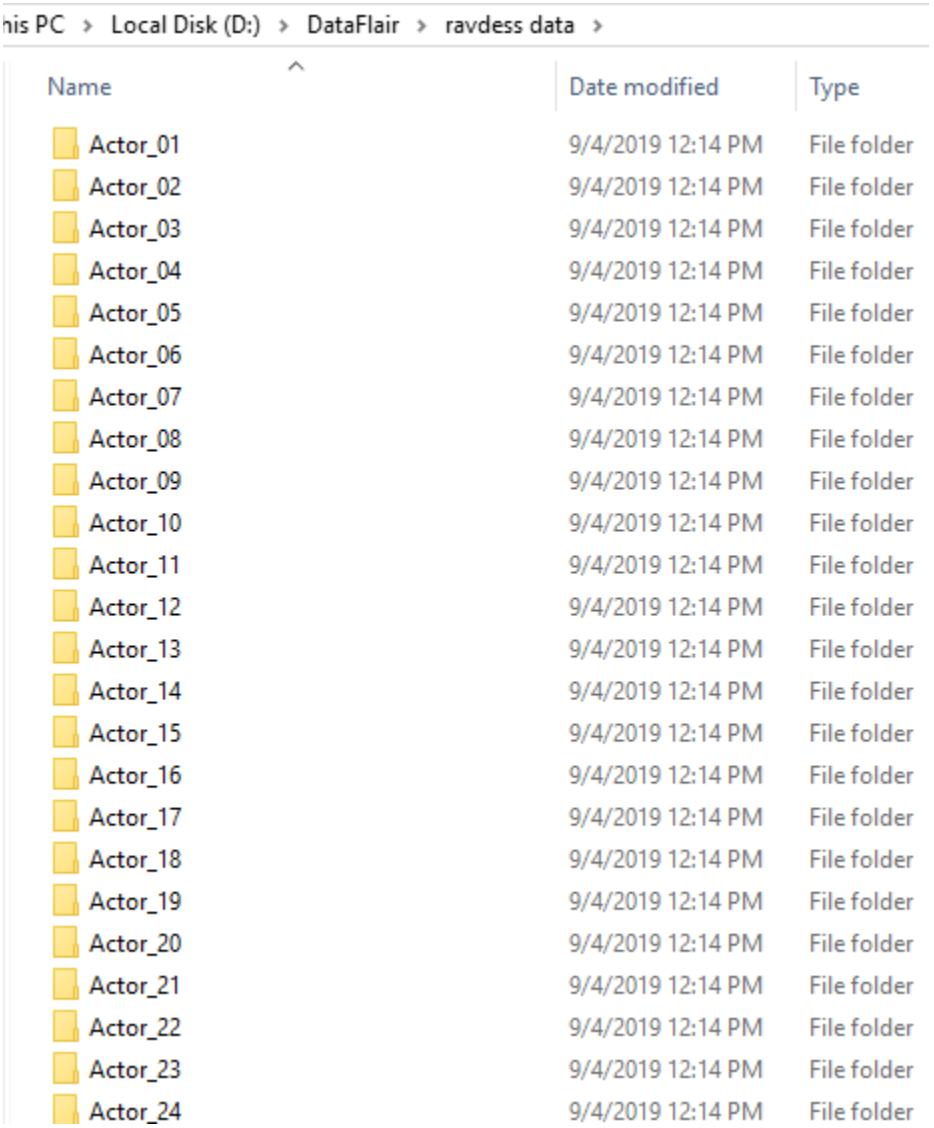
3. Now, let's define a ***dictionary*** to hold numbers and the emotions available in the RAVDESS dataset, and a list to hold those we want- calm, happy, fearful, disgust.

```
4. #DataFlair - Emotions in the RAVDESS dataset
5. emotions={
6. '01':'neutral',
7. '02':'calm', 8.
   '03':'happy',
9. '04':'sad', 10.
'05':'angry',
11. '06':'fearful',
12. '07':'disgust',
13. '08':'surprised'
14. }
15.
16. #DataFlair - Emotions to observe
17. observed_emotions=['calm', 'happy', 'fearful', 'disgust']
```

```
[3]: #DataFlair - Emotions in the RAVDESS dataset
     emotions={
         '01':'neutral',
         '02':'calm',
         '03':'happy',
         '04':'sad',
         '05':'angry',
         '06':'fearful',
         '07':'disgust',
         '08':'surprised'
     }

     #DataFlair - Emotions to observe
     observed_emotions=['calm', 'happy', 'fearful', 'disgust']
```

4. Now, let's load the data with a function load_data() – this takes in the relative size of the test set as parameter. x and y are empty lists; we'll use the glob() function from the glob module to get all the pathnames for the sound files in our dataset. The pattern we use for this is: "D:\\DataFlair\\ravdess data\\Actor_**.wav". This is because our dataset looks like this:



Chapter 5
Algorithms

5.1 LOGISTIC REGRESSION

Logistic Regression is a supervised classification algorithm which produces probability values of data belonging to different classes. 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”. 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).

5.2 NAIVE BAYES

Naive Bayes classifier is based on Bayes theorem, which determines the probability of an event based on a prior probability of events. 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. 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.

5.3 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 X n2 samples) and O(n samples X n3 samples).

5.4 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. 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.

5.5 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)).

5.6 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

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.

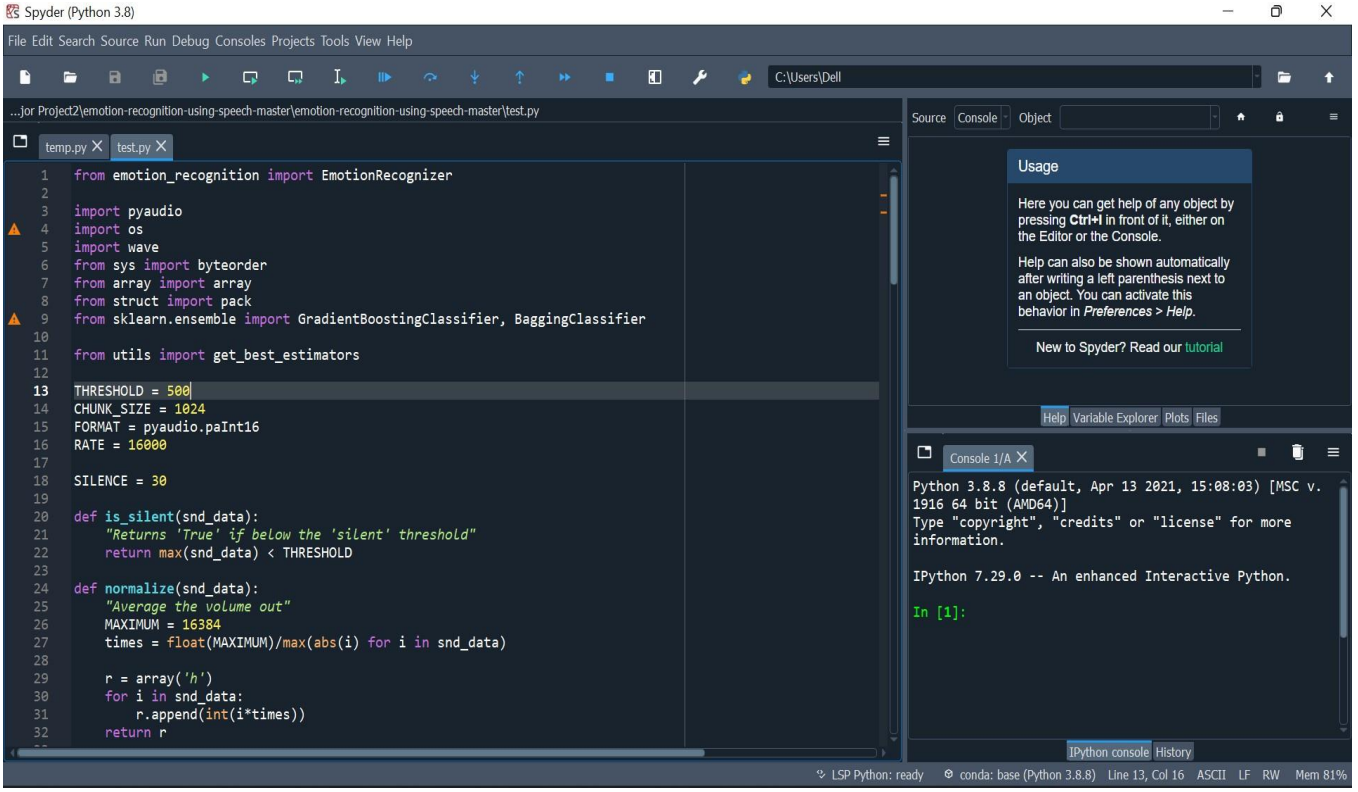
5.7 GRADIENT BOOSTING TREE

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.

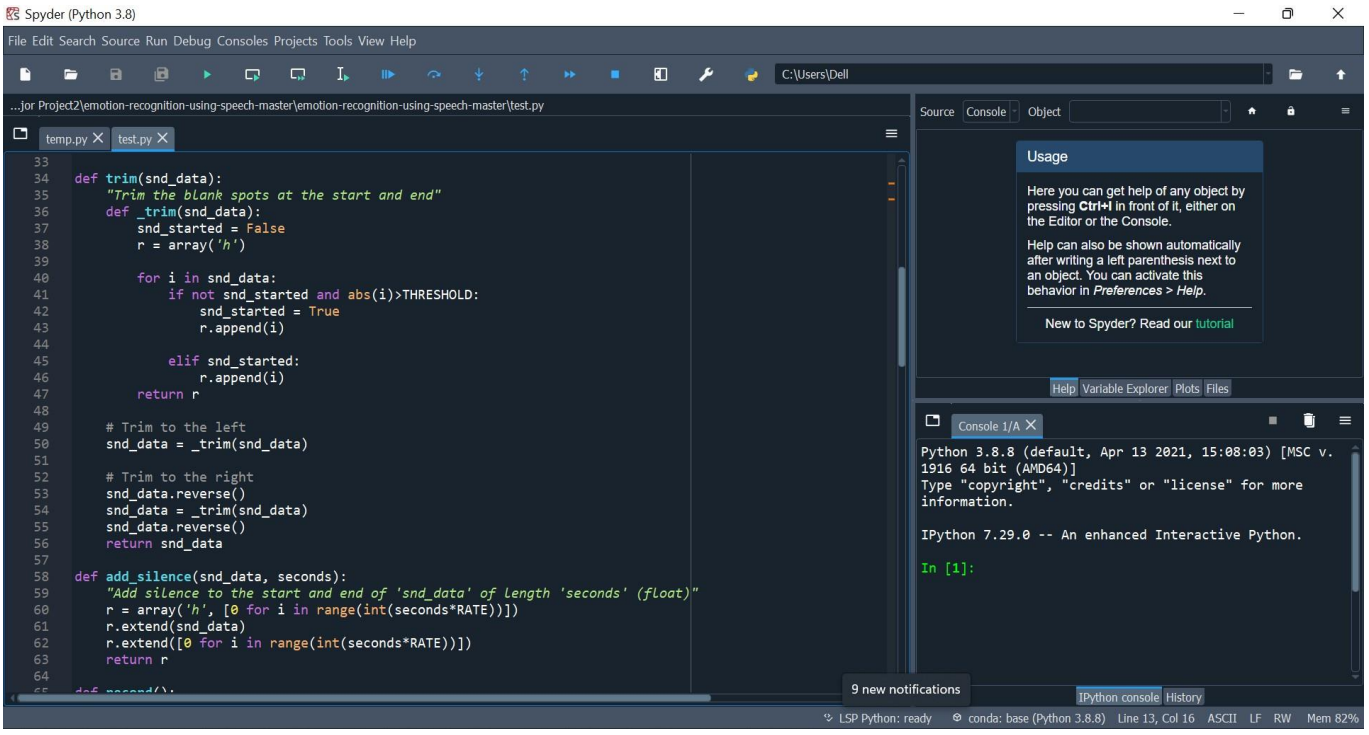
- Optimizing the loss function It is a mathematical function representing errors occurring in a model. The logarithmic loss function is best suited for classification problems. However, one may define their own loss function.
- Using a weak learner for the predictions A learning algorithm such as decision 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.
- 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.

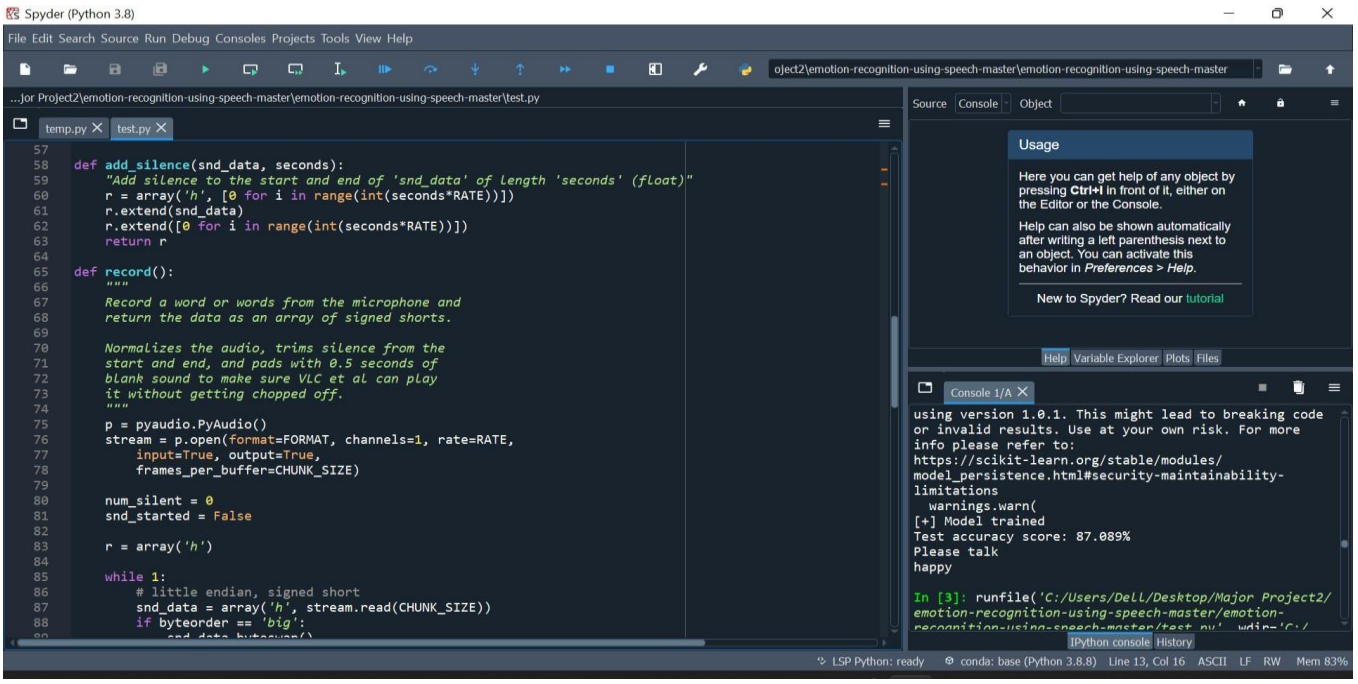
CHAPTER 6
RESULTS



Result 1



Result 2



Result 3

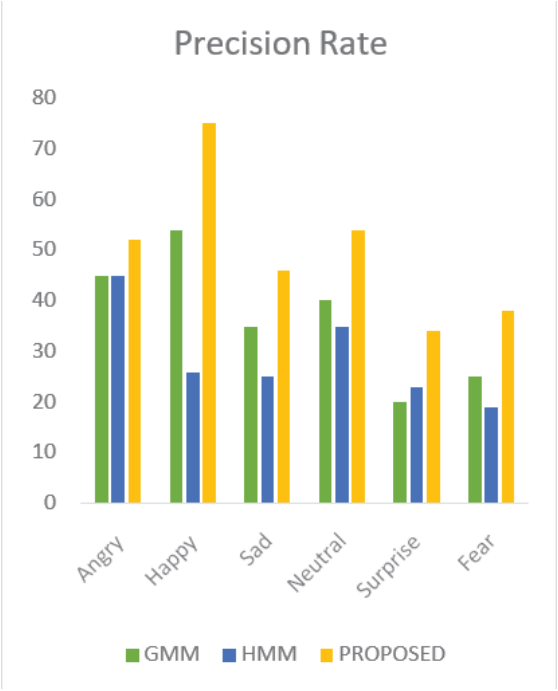


Fig. Accuracy Finding from various technique

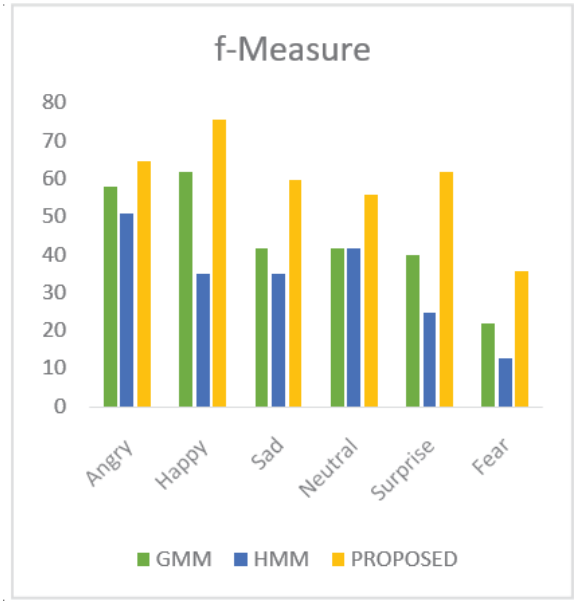
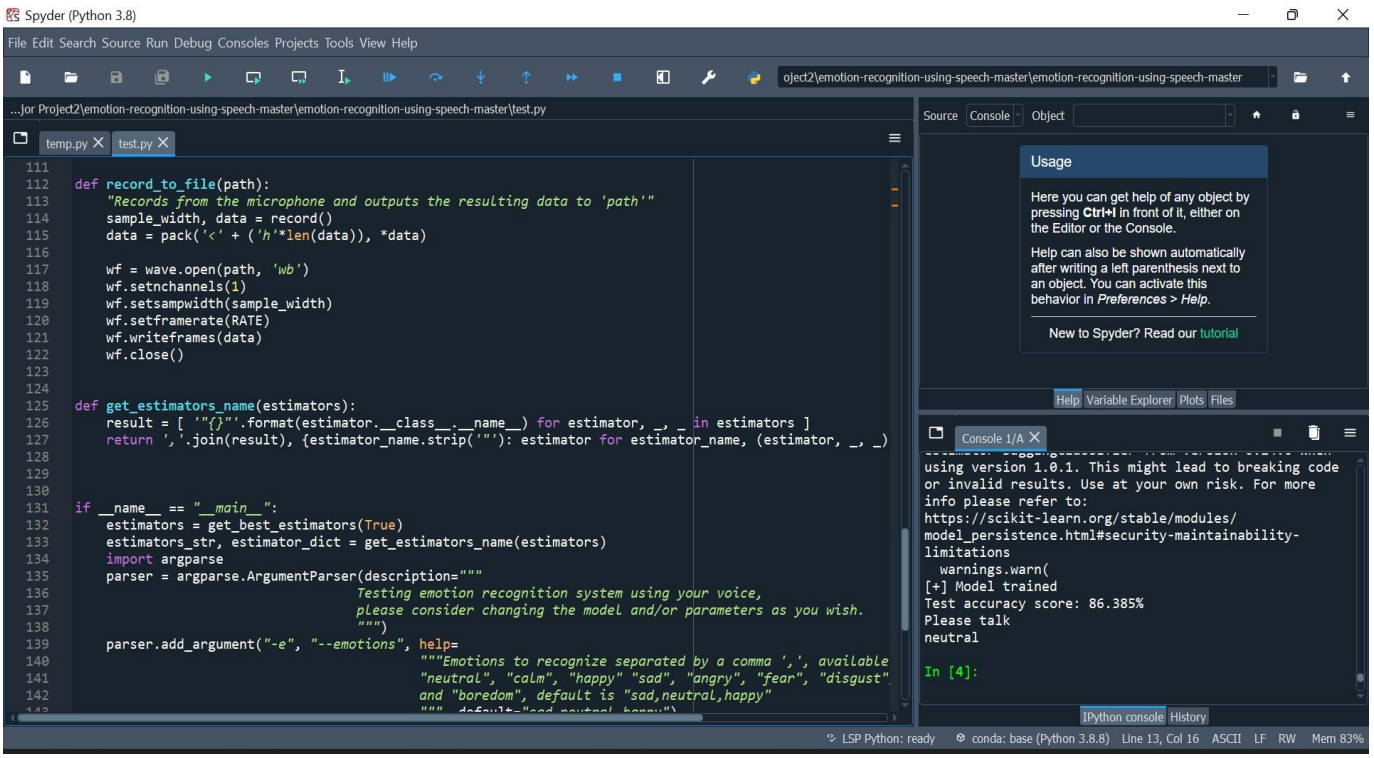


Fig. Measures from various techniques

The above Fig. denotes the f-measured values obtained through various techniques used in this system. It can be noted that the maximum f-measure value was obtained for happy and the minimum value was obtained in case of fear.

Thus, from the above figures, it can be noted that the proposed system appears to be a better network than the previously used techniques of GMM and HMM.



Result 4

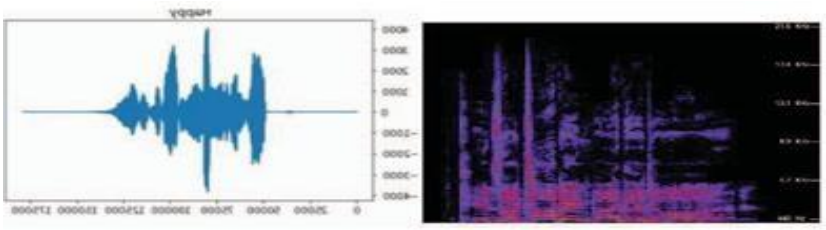


Fig. Spectral sample for Happy Emotion

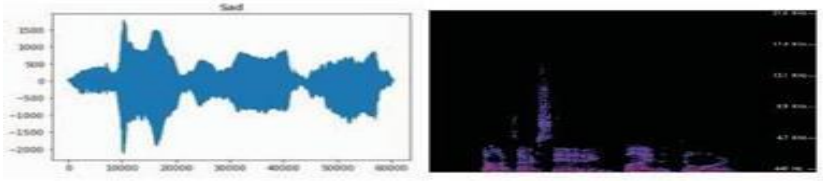


Fig. Spectral sample for Sad Emotion

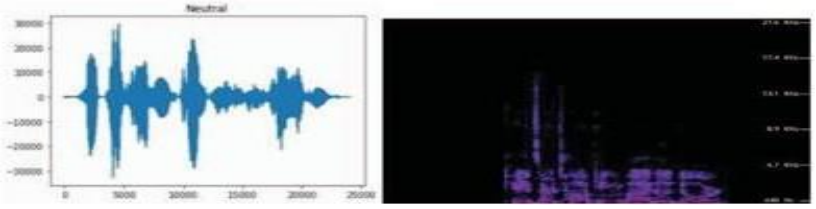


Fig. Spectral sample for Neutral Emotion.

CONCLUSION

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 fortherapy 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.

In this Python mini project, we learned to recognize emotions from speech. We used an MLP Classifier for this and made use of the sound file library to read the sound file, and the librosa library to extract features from it. As you'll see, the model delivered an accuracy of 84.4%. That's good enough for us yet.

REFERENCES:

1. Özseven, T. (2018). Investigation of the effect of spectrogram images and different texture analysis methods on speech emotion recognition. Applied Acoustics, 142, 70-77.
2. Yang, B., & Lugger, M. (2010). Emotion recognition from speech signals using new harmony features. signal processing, 90(5), 1415- 1423.
3. Schuller, B., Batliner, A., Steidl, S., & Seppi, D. (2011). Recognising realistic emotions and affect in speech: State of the art and lessons learnt from the first challenge. Speech Communication, 53(9-10), 1062-1087.
4. Anagnostopoulos, C. N., Iliou, T., & Giannoukos, I. (2015). Features and classifiers for emotion recognition from speech: a survey from 2000 to 2011. Artificial IntelligenceReview, 43(2), 155-177.
5. Anagnostopoulos, C. N., Iliou, T., & Giannoukos, I. (2015). Features and classifiers for emotion recognition from speech: a survey from 2000 to 2011. Artificial IntelligenceReview, 43(2), 155-177.
6. Guidi, A., Vanello, N., Bertschy, G., Gentili, C., Landini, L., &Scilingo,
7. P. (2015). Automatic analysis of speech F0 contour for the characterization of mood changesin bipolar patients. Biomedical Signal Processing and Control, 17, 29-37.
8. Liu, Z. T., Xie, Q., Wu, M., Cao, W. H., Mei, Y., & Mao, J. W. (2018).
9. Speech emotion recognition based on an improved brain emotion learning model. Neurocomputing.
10. Torres-Boza, D., Oveneke, M. C., Wang, F., Jiang, D., Verhelst, W., &Sahli, H. (2018). Hierarchical sparse coding framework for speech emotion recognition. Speech Communication, 99, 80-89.
11. Zhao, J., Mao, X., & Chen, L. (2019). Speech emotion recognition using deep 1D & 2D CNN LSTM networks. Biomedical Signal Processing and Control, 47, 312-323.
12. Jing, S., Mao, X., & Chen, L. (2018). Prominence features: Effective emotional features for speech emotion recognition. Digital Signal Processing, 72, 216-231.
13. Fayek, H. M., Lech, M., & Cavedon, L. (2017). Evaluating deep learning architectures for Speech Emotion Recognition. Neural Networks, 92, 60-68.
14. El Ayadi, M., Kamel, M. S., & Karray, F. (2011). Survey on speech emotion recognition: Features, classification schemes, and databases. Pattern Recognition, 44(3), 572-587.
15. Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A. R., Jaitly, N., ... & Kingsbury, B. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared viewsof four research groups. IEEE Signal processing magazine, 29(6), 82-97.
16. Kasabov, N., Dhoble, K., Nuntalid, N., & Indiveri, G. (2013). Dynamic evolving spiking neural networks for on-line spatio-and spectro- temporal pattern recognition. Neural

Networks, 41, 188-201.

17. Gou, J., Ma, H., Ou, W., Zeng, S., Rao, Y., & Yang, H. (2019). A
18. generalized mean distance-based K-Nearest neighbor classifier. Expert Systems with Applications, 115, 356-372.
19. Zhang, Y., Cao, G., Wang, B., & Li, X. (2019). A novel ensemble method for K-Nearest neighbor. Pattern Recognition, 85, 13-25.
20. Ertuğrul, Ö. F., & Tağluk, M. E. (2017). A novel version of k nearest neighbor: Dependent nearest neighbor. Applied Soft Computing, 55, 480-490.
21. Kim, T. H. (2010, June). Pattern recognition using artificial neural network: a review. In International Conference on Information Security and Assurance (pp. 138-148). Springer, Berlin, Heidelberg
22. Dabas, P., & Kumar, U. (2014). Pattern Recognition using Artificial Neural Network. International Journal of Computer Applications Technology and Research, 3(6), 358-360.
23. Lima, Í. A., Alencar, M. S., Lopes, W. T., & Madeiro, F. (2015, June). Evaluation of optimal and sub-optimal speech noise reduction wiener filters. In Telecommunications (IWT), 2015 International Workshop on (pp. 1-5). IEEE.
24. Basu, S., Chakraborty, J., Bag, A., & Aftabuddin, M. (2017, March). A review on emotion recognition using speech. In Inventive Communication and Computational Technologies (ICICCT), 2017 International Conference on (pp. 109-114). IEEE.
25. Lanjewar, R. B., & Chaudhari, D. S. (2013). Speech emotion recognition: a review. International Journal of Innovative Technology and Exploring Engineering (IJITEE), 2(4), 68-71.
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-neighboursalgorithm-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-learning3c0fe15b6674.
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-

machinelearning/.

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/.