12-2020

Automation of Feature Selection and Generation of Optimal Feature Subsets for Beehive Audio Sample Classification

Aditya Bhouraskar
Utah State University

AUTOMATION OF FEATURE SELECTION AND GENERATION OF OPTIMAL FEATURE SUBSETS FOR BEEHIVE AUDIO SAMPLE CLASSIFICATION

by

Aditya Bhouraskar

A thesis submitted in partial fulfillment
of the requirements for the degree

of

MASTER OF SCIENCE

in

Computer Science

Approved:

Dr.Vladimir Kulyukin, Ph.D. Major Professor	Dr.Nicholas Flann, Ph.D. Committee Member
Dr.Curtis Dyreson, Ph.D. Committee Member	D. Richard Cutler, Ph.D. Interim Vice Provost for Graduate Studies

UTAH STATE UNIVERSITY
Logan, Utah

2020

ABSTRACT

Automation of Feature Selection and Generation of Optimal Feature Subsets for Beehive Audio Sample Classification

by

Aditya Bhouraskar, Master of Science

Utah State University, 2020

Major Professor: Dr.Vladimir Kulyukin, Ph.D.
Department: Computer Science

Features in Machine Learning (ML) are individual measurable property or characteristic of a phenomenon being observed. They act as an input in our system. In order to use a set of features for an ML model, the quality of the features plays an important role in problems related to pattern recognition, Classification and Regression. Feature Selection is one technique that helps achieve this goal of selecting the best features where one manually selects those features which contribute the most to the prediction model. The principal focus of this thesis is to demonstrate it is possible to automate feature selection techniques in an input dataset, as well as selecting the best features out of all possible combinations of feature subsets, which can work on par with Standard Machine Learning techniques and Deep Learning methods. In this automation process, we have mainly used three feature selection methods (Wrapper methods, Filter Methods and Embedded methods). These methods provide a number of possible selected feature subsets to different ML models, this process runs for all the datasets we are using in this research.

(64 pages)

PUBLIC ABSTRACT

Automation of Feature Selection and Generation of Optimal Feature Subsets for Beehive Audio Sample Classification

Aditya Bhouraskar

The last couple of decades have witnessed an abnormal phenomenon of reduction in the bee population, this is a serious matter of concern as three out of four crops available globally have honey bee as their sole pollinator causing significant economic losses and an unbalance in the ecosystem. There have been many theories about the cause of bee colony collapses such as parasites, pesticides and poor nutrition however conclusive evidence of this phenomenon is yet to be identified.

Human inspection of beehives requires precision. It takes an experienced beekeeper to determine the health of a hive by the sounds generated by the bees. If the sound indicates poor health, the beekeeper must then disrupt the hive to inspect and ascertain possible causes of poor health. This interferes with beehive activity, which can then threaten even further hive health. This work uses Feature Engineering and Machine Learning to develop techniques to monitor hive health. The thesis aims at building an automation technique for finding the best feature subsets using datasets containing different classes of audio sounds. Selecting good features forms the basis for machine learning models to further classify these audio samples. The purpose of finding the best features is to get a better audio classification which helps beekeepers know about the health of beehives and address problems such as bee immunity, effects of pesticides and environmental and nutritional stressors from remote locations.

ACKNOWLEDGMENTS

I would like to take this opportunity to thank my major professor Dr. Vladimir Kulyukin. His expertise and input were invaluable. He was prompt when responding to my inquiries when in-person meetings were not possible due to a global pandemic. He was an extraordinary mentor, from whom I learned life lessons I will carry forward in my professional career.

I would also like to express my sincere gratitude toward my research committee members Dr. Nicholas Flann and Dr. Curtis Dyreson, for providing appropriate support and feedback on various aspects of my research.

Finally, I appreciate the continuous support and encouragement of my beloved parents and my brother throughout the duration of my academic pursuits.

Aditya Bhouraskar

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACRONYMS

ML	machine learning
DL	deep learning
CCD	colony collapse disorder
NASS	National Agricultural Statistics Service
EBM	electronic beehive monitoring
RS	respiratory sounds
HMM	hidden Markov model
ICA	independent component analysis
DWT	discrete wavelet transform
SVM	support vector machine
DFT	discrete fourier transform
MFCC	mel frequency cepstral coefficients
DCT	discrete cosine transform
RF	random forest
KNN	k-nearest neighbors
RFE	recursive feature elimination
SFS	sequential feature selection
CNN	convolutional neural network

CHAPTER 1

INTRODUCTION

1.1 Background

Honeybees are the world's single most important species of pollinator, a key contributor to natural ecosystem functions. Some of the reasons bees are important are wild plant growth, food production, wildlife habitats and biodiversity. It takes more than soil, water and sunshine to make the world green. 30 percent of the world's crops and 90 percent of all plants require cross pollination to thrive [7]. Bees are responsible for pollinating about one-sixth of the flowering plant species worldwide [8]. Over the last two decades in the USA and Europe bee keepers have reported a decline in bee population by 50 %. The main symptom involved sudden loss of colonies' worker bee populations with few dead bees found near the hives, a puzzling phenomenon. The queen and capped brood were left behind. For hives to sustain themselves the presence of worker bees is of vital importance. Without them the hives eventually die. This phenomenon is known as colony collapse disorder (CCD) [9].

This disruptive behaviour is a problem that has been tormenting honey bees since 2006 is a syndrome specifically defined as a dead colony with no adult bees and with no dead bee bodies but with a live queen, and usually honey and immature bees, still present. The National Agricultural Statistics Service (NASS) reported 2.44 million honey-producing hives were in the United States in February 2008, down from 4.5 million in 1980, and 5.9 million in 1947 [10]. Some of the possible causes for this can be pesticides, mites, fungi, beekeeping practices (such as the use of antibiotics or long-distance transportation of beehives), malnutrition, poor quality queens, starvation and other pathogens. However, no single reason comes out to be the major cause of this issue, hence the beekeepers need to monitor the hives regularly as to take precautionary measures to prevent CCD. A human beehive inspection can be accurate but it needs a lot of research and stern precision to

collect the data without disturbing the bee colonies, that is where the use of Electronic Beehive Monitoring (EBM) systems can help in automating the collection of large amount of information without disrupting hives and bee colonies. The use of EBM systems have gained popularity over the years with some ongoing work and some future prospects as well.

Feature Engineering is a technique which aims at providing meaningful information from the predictive models. Any machine learning algorithms uses some input data to create outputs. This input data comprises of features which are usually in the form of structured columns with some specific characteristic providing details about the data. Feature Engineering has mainly two goals - preparing a proper input dataset and improving the performance of machine learning models.

Further the classification models can classify the data and give details about any anomalies present in it which could help bee keepers to keep track of honey bee hive's health and stress level surrounding it. Each kind of feature reveals some information about the health of the hives and analyzing them over a period of time can prove helpful to differentiate abnormal behaviours of bees. Thus selecting a set of feature which acts as an input and identifying which feature set makes the most difference is a valuable information in this field of research. Some related work has been described below

1.2 Related Work

In an approach to develop feature automation in Electronic Beehive monitoring, this thesis claims that with feature engineering and machine learning techniques a strong system can be developed for audio beehive monitoring to automate the identification of any anomalies in the beehives. In this research, we have automated several feature selection techniques to get the best feature subsets and compared their performance with Standard Machine Learning techniques and Deep Learning methods. This thesis build up on the work of Gupta [11], which extracts features (using PyAudio) from an audio sample collected from BeePi and classifies them using machine learning techniques. Extraction of features was a manual process and classifying each of them using machine learning methods was another manual process which seemed to be a monotonous approach and was limited to only few Au-

dio features. Automating this whole process of extracting all the features using the Python library PyAudioAnalysis [1] and then using three feature selection methods(wrapper, filter and embedded methods) helped in getting the best amongst all the extracted features and further extending the previous research. Some of the researchers have done considerable amount of work in the field of Audio Beehive Monitoring and feature engineering with sound classifications, their work has been discussed below:

Cecchi et al. [12] designed a Smart Sensor-Based Measurement System for advanced Bee Hive Monitoring to measure different parameters related to beehives such as hive weight, sounds emitted by the bees, temperature, humidity, and carbon di oxide inside the beehive, as well as weather conditions outside. The researchers have modularized the system and it is composed of two main modules named Bee board which is installed in each hive and Queen board which consists of RaspberryPi 3B equipped with several sensors to acquire weather parameters near the hives. The experimental results have shown various bee patterns such as average time period for a bee to move out of a beehive and its comparison in different weather seasons.

Ferrari et al. [13], proposes a system to detect Swarming events. Swarming is the natural means of reproduction of honey bee colonies. In the process of swarming the original single colony reproduces into two or more colonies with the queen bee leaving the primary swarm along with worker bees. The researchers have developed a method that enables the prediction of swarming so that the queen bee does not leave the hives. Three beehives were monitored and it was observed that an increase of the buzzing frequency at about 110 Hz with a rapid increase of energy peaks at 300 Hz was an indicative signal of swarming.

In an earlier work performed with the similar BUZZ1 and BUZZ2 datasets by Kulyukin et al. [14], several convolutional neural networks were designed and their performance was compared with some Standard Machine Learning methods such as logistic regression, k-nearest neighbors, support vector machines, and random forests. For BUZZ1 dataset, training and testing samples were separated from the validation samples by beehive and location while for the BUZZ2 dataset, training and testing samples were separated from

the validation samples by beehive, location, time, and bee race. The researchers observed that a shallower raw audio convolutional neural network with a custom layer performed on par with the four machine learning methods for BUZZ1. However the convolutional neural networks generalized better on the second, more challenging dataset, they took considerably more time to train than the machine learning methods.

In the field of Medical and Biological Engineering, the research of Xie et al. [15] proposes a new set of features using multi-scale Principal Component Analysis as a signal enhancement and feature extraction method to capture major variability of Fourier power spectra of the signal. The research is based on temporal characteristics of filtered narrow-band to classify respiratory sounds(RS) into normal and continuous adventitious type. The mean classification accuracy of 98.34% was resulted from 689 real RS recording segments shows the credibility of the current method.

One more research that deals with a similar approach of feature extraction is by Yong-Choon et al. [16]. This research deals with acoustic feature extraction of spectro-temporal sounds which leads to nonnegative features. Acoustic features are fed into a hidden Markov model (HMM) classifier. Experimental results confirm that the proposed feature extraction method improves the classification performance, especially in the presence of noise. The results were compared with an earlier Independent Component Analysis(ICA) based sound recognition system which was adopted in MPEG-7, which proved the validity of a high performance system.

In the research of Environmental sound recognition with Time-frequency audio features by Chu Selina and Narayanan Shrikanth [17] an approach is presented to recognize environmental sound for the understanding of scene or context surrounding an audio sensor. They have used a Matching Pursuit algorithm to obtain effective time frequency features which are intuitive and physically interpretable set of features. These features are further adopted to supplement the MFCC features to yield higher recognition accuracy for environmental sound.

An approach similar to the one applied in this research can be found in the research

of Ramalingam Thiruvengatanadhan and Dhanalakshmi P [18] which classifies speech and music audio sounds. Here a wavelet based feature extraction technique called Multi resolution analysis to extract the feature from the input signals is used which utilizes the Discrete Wavelet Transform (DWT) as the acoustic feature. ML method Support Vector Machine(SVM) is applied to classify audio into their classes namely Speech and Music by learning from the training data.

1.3 Current Work

Features are crucial in machine learning and are important for success of predictive problems, Feature Selection also known as variable selection is the process of selecting a subset of relevant features for use in model construction. The central idea behind a Feature selection technique is that some features are redundant or irrelevant which can be removed instead of using all the data, thus resulting in less amount of information lost.

It is to be noted that feature selection techniques should be distinguished from Feature Extraction. Feature Extraction creates new features from functions whereas feature selection selects a subset of those features. Extracting the most important features from a sample can have a significant impact on the performance of our classifier. In this study, firstly an audio sample is reduced to a set of 34 features which represent the full sample. Later, some feature selection techniques are applied, which reduces the current set of 34 features into a small subset of smaller number of features. The range of these subsets can vary from a single feature in a subset to 34 features. Finally, the credibility of these features were verified by using them with different ML classifier models to get the best classification performance.

CHAPTER 2

DATA COLLECTION

2.1 Data Collection from BeePi

BeePi is a multi sensor electronic beehive monitor and all the hardware associated with it fits in a standard Langstroth super box [2]. The hardware of this system consists of a raspberry pie computer, a miniature camera to collect video samples, an SD card is connected to the beePi where the system software and collected data resides, a waterproof temperature measuring sensor, an audio sensor which consists of multiple audio jacks and connect multiple microphones to collect audio samples from the hives, this setup collectively makes an Electronic beehive monitoring system (EBM) and can be seen in the figure 2.1.

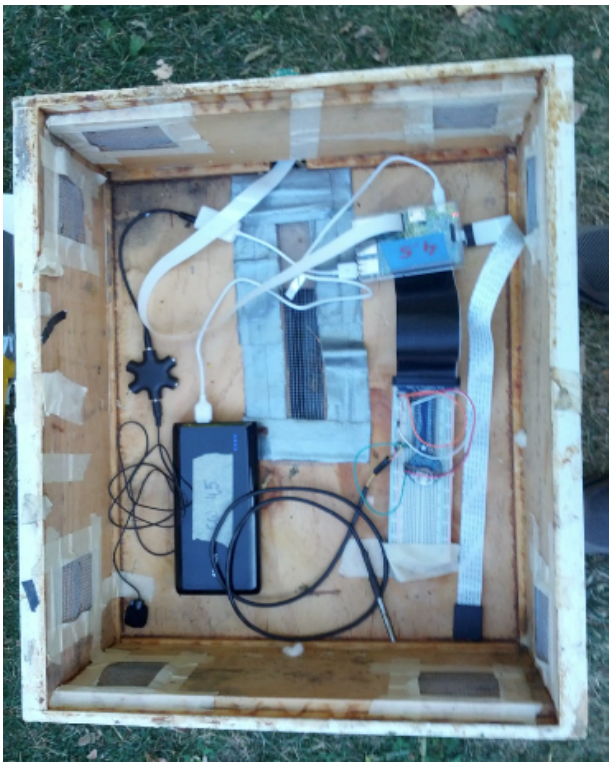

Fig. 2.1: BeePi hardware components placed inside Langstroth super [2]

In this work, we are focused on the audio sensor part of the EBM to work with audio samples. For this purpose, a six-way multiport 3.5 mm jack splitter is used with 3 microphones and each microphones are placed on different locations of the beehive, another port of the jack is connected to an audio adapter which in turn gets attached to a storage devise, this audio sensor setup can be seen in figure 2.2. EBM extracts critical information on colony behavior and health without invasive hive inspections, an essential feature of EBM is its reproducibility, this system can be replicated by other researchers and bee keepers, also the hardware used in this setup is not designed or made to order but taken from existing stock and supplies which makes it much cheaper as compared to other commercial equipments. The most important component pertaining to this research is the audio sensors which are the microphones which captures audio data in regular time intervals and attaches a timestamp to the file, all the audio files are in .wav format.

Fig. 2.2: Audio Sensors

Over the past few years several deployments have been made for BeePi in order to collect various forns of data such as videos, images, audio and temperature. These deploy-

ments took place at Logan, UT, USA in September 2014, followed by another in December 2014 and January 2015 in Garland, UT, USA. The most recent one was between May 2018 and July 2018 at Logan, UT, USA with four BeePi monitors placed in four new beehives. A very critical component of this setup is the audio sensor which saves a 30-second audio wav file every 15 minutes on a USB storage device connected to the Raspberry Pi [14]. A python script was used to chunk the 30-second audio sample into 2-second wav files with an overlap of 1-second. This resulted in 28 2-second wav files for each 30-second audio sample.

2.2 Audio data

The data collected was divided into 4 datasets namely BUZZ1, BUZZ2, BUZZ3 and BUZZ4. The former three datasets consists of three categories of audio samples i.e bee buzzing (Bee), cricket chirping (Cricket) and ambient noise (Noise). The sound of bees buzzing here is caused by the rapid movement of their wings. The rapid contraction of their wing flight muscles is what causes the high pitched whining (buzzing) sound, Crickets "chirp" sound occurs when they rub their wings or legs over each other, Ambient noise here refers to the random microphone clicks, human conversations, breeze, rain and relative silence (i.e. sounds which cannot be detected by humans). BUZZ4 dataset contains an additional audio category of "lawn mowing", this is the sound of the grass cutting machine.

The 2 second audio samples were heard meticulously to place it in one of the four non overlapping categories: Bee, Cricket, Noise, Lawn Mowing. The 2 second wav files were than read and stored in a numpy array in order to facilitate feature extraction from them by a Python library pyAudioAnalyis [1].

2.3 Datasets

We have used four different datasets in order to train, test and validate different machine learning models. These datasets are also publicly available. The content of these four datasets are described below:

- BUZZ1: contains a total of 10,260 2-second audio samples. [19]

- BUZZ2: contains a total of 12,914 2-second audio samples. [20]
- BUZZ3: contains a total of 15,254 2-second audio samples. [21]
- BUZZ4: contains a total of 18,594 2-second audio samples. [22]

Further the datasets have been divided into three categories i.e. the train data, the test data and the validation data which were separated from each other by beehive and location. The ratio of data divided between these categories is similar for all datasets. the distribution of the data among train, test and validation is represented in 2.1, 2.2, 2.3 and 2.4. The training, testing and validation data in all the datasets are different and comes from different hives.

Table 2.1: BUZZ1 sample distribution of 2-second audio samples.

	Bee	**Cricket**	**Noise**	**Total**
Train	2128	2128	2238	6494
Test	872	872	872	2616
Validate	300	500	350	1150
Total	3300	3500	3460	10260

Table 2.2: BUZZ2 sample distribution of 2-second audio samples.

	Bee	**Cricket**	**Noise**	**Total**
Train	2402	3000	2180	7582
Test	898	500	934	2332
Validate	1000	1000	1000	3000
Total	4300	4500	4114	12914

Table 2.3: BUZZ3 sample distribution of 2-second audio samples.

	Bee	**Cricket**	**Noise**	**Total**
Train	2880	3600	2520	9000
Test	1071	577	1098	2746
Validate	1170	1169	1169	3508
Total	5121	5346	4787	15254

Table 2.4: BUZZ4 sample distribution of 2-second audio samples.

	Bee	**Cricket**	**Noise**	**Lawn**	**Total**
Train	2880	3600	2520	2120	11120
Test	1071	577	1098	840	3586
Validate	1170	1169	1169	380	3888
Total	5121	5346	4787	3340	18594

CHAPTER 3

FEATURE ENGINEERING

3.1 Overview

Feature engineering is the process of transforming raw data into features that better represent the underlying problem to the predictive models, resulting in improved model accuracy on unseen data. When the goal is to get the best results from a predictive model, a lot depends on the algorithms used and getting the most out of the data for the algorithms to work with. The success of a machine learning algorithm depends on the way data is presented to them, this problem is solved by Feature Engineering. This chapter covers the sub topics in Feature Engineering i.e Feature Extraction and Feature Selection. Feature Extraction is the art of construction of new features, whereas feature selection works on forming the best feature subsets out of the available ones. The methods for extracting features and selecting them are discussed in detail in this chapter.

3.2 Feature Extraction

Feature Extraction is the process of dimensionality reduction by which an initial set of raw data is reduced to more manageable groups for processing. One of the characteristic of this input dataset is that it is too large to be processed and suspected to have redundancy and requires a lot of computing resources. These extracted features are expected to contain the relevant information from the input data, so that the desired task can be performed using this reduced representation of input data rather than using the large input dataset.

3.2.1 pyAudioAnalysis

The first step in this thesis work is Feature Extraction. Time, frequency and cepstral domain features of an audio signal have been extracted, we have done this using pyAudioAnalysis. pyAudioAnalysis is a Python library covering a wide range of audio analysis

tasks and by using this we can perform many tasks like extracting audio features and representations (example MFCCs, spectrogram and chromagram), Classify unknown sounds and apply dimensionality reduction to visualize audio data and content similarities. 3.1.

The time-domain features (features 1–3) are directly extracted from the raw signal samples. The frequency-domain features (features 4–34, apart from the MFCCs) are based on the magnitude of the Discrete Fourier Transform (DFT). Finally, the cepstral domain (e.g. used by the MFCCs) results after applying the Inverse DFT on the logarithmic spectrum [1]. The table 3.1 shows a list of extracted features from pyAudioAnalysis and a description of each feature is presented below it.

Table 3.1: 34 features extracted by pyAudioAnalysis [1].

Feature ID	**Feature Name**	**Description**
0	Zero Crossing Rate	The rate of sign-changes of the signal during the duration of a particular frame.
1	Energy	The sum of squares of the signal values, normalized by the respective frame length.
2	Entropy of Energy	The entropy of sub-frames' normalized energies. It can be interpreted as a measure of abrupt changes.
3	Spectral Centroid	The center of gravity of the spectrum.
4	Spectral Spread	The second central moment of the spectrum.
5	Spectral Entropy	Entropy of the normalized spectral energies for a set of sub-frames.
6	Spectral Flux	The squared difference between the normalized magnitudes of the spectra of the two successive frames.
7	Spectral Rolloff	The frequency below which 90% of the magnitude distribution of the spectrum is concentrated.
8-20	MFCCs	Mel Frequency Cepstral Coefficients form a cepstral representation where the frequency bands are not linear but distributed according to the mel-scale.
21-32	Chroma Vector	A 12-element representation of the spectral energy where the bins represent the 12 equal-tempered pitch classes of western-type music (semitone spacing).
33	Chroma Deviation	The standard deviation of the 12 chroma coefficients.

Zero Crossing Rate

In the context of speech recognition, the waveforms vary a lot in smoothness. For example, voiced speech sounds are more smooth than unvoiced ones. A simple way of measuring the smoothness of an audio signal is to calculate the number of zero crossings, the measurement of zero crossing rate can be identified by the number of time frame the amplitude of the audio signal has passed through a value of zero. The number of zero crossings are generally low for voiced speech and high for unvoiced speech. figure 3.1 shows an example of zero crossing for a broadband signal.

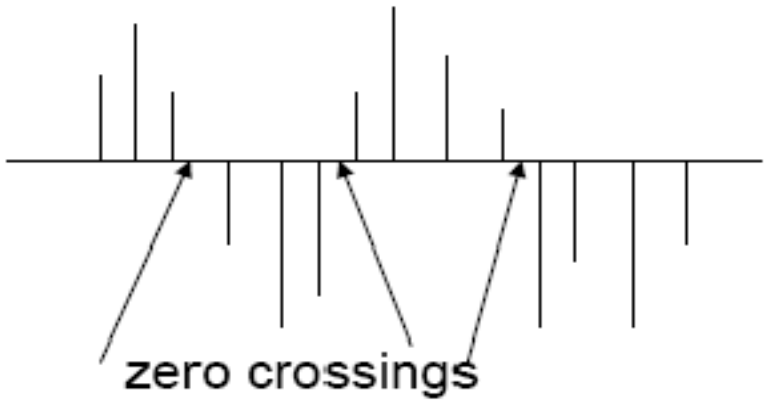

Fig. 3.1: Zero-crossing Rate [3].

zero-crossings rate can be defined as:

$$Zn = \sum_{m=-\infty}^{\infty} |sgn[x(m)] - sgn[x(m-1)]|w(n-m) \tag{3.1}$$

Energy

Another parameter for differentiating between voiced and unvoiced speech sound is calculating the energy of the audio signal. The amplitude of the speech signal changes with time, generally the amplitude of voiced speech is higher because of the frequent periodicity and unvoiced speech has a comparatively lower amplitude [3].

energy can be defined as :

$$Zn = \sum_{m=-\infty}^{\infty} [x(m)w(n-m)] \tag{3.2}$$

Entropy of Energy

The entropy of a signal is the measure of the amount of information it carries. The entropy of sub-frames' normalised energies. It can be interpreted as a measure of abrupt changes. The audio signal is first divided into small windows called short-term frames and then all 34 features are calculated for each frame. In order to compute entropy of energy, we first divide each short-term frame in K sub-frames of fixed duration. Then for each sub-frame,j, we compute its energy and divide it by the total energy of the short-term frame [23]. Energy indicates the loudness of an audio signal and can be calculated by the below equation:

$$E = \frac{1}{N} \sum_{n=1}^{N} (x(n)) \tag{3.3}$$

Here x(n)is the value of the sample (in time domain) and N is the total number of samples in the processing window (frame size). Thus, the energy of sub-frame,j, can be represented by.

$$e_j = \frac{E_{subFramej}}{E_{shortFramei}} \tag{3.4}$$

Spectral Centroid

Spectral Centroid is used to determine where does the centre of mass of the given spectrum lies. It is closely related to the brightness of a sound. Mathematically it is computed by taking the weighted average of of all the frequencies in the given signal which in turn are computed with the help of fourier transform by using the magnitudes as weights. Sometimes spectral centroid is used in reference with the median of the input because both of them are measures of central tendency. So at times both of them display similar behaviour

in some of the situations. Higher centroid values indicate much brighter textures having high rising frequencies [24]. It can be defined as:

$$centroid = \frac{\sum_{n=0}^{N-1} f(n)x(n)}{\sum_{n=0}^{N-1} x(n)} \tag{3.5}$$

Spectral Spread

This feature is used for transmission in telecommunication and radio signals. In this technique a signal such as acoustic signal generated with a particular bandwidth is spread in the frequency domain resulting in a signal with wider bandwidth. Higher the value of spectral spread, more distributed the spectrum is on both sides of the centroid whereas lower values implies that the spectrum is highly confined near the centroid [25]. Signals such as raw noise will have a higher spectral spread and a simple tonal sound which can be identified with the regularity of vibrations have a lower spectral spread.

Spectral Entropy

Spectral entropy is a quantitative analysis of the regularity or randomness of a power spectrum during a period of time for a given input audio. Mainly it is used to determine voiced and silence regions of speech. It is highly used in speech recognition because of its this discriminatory property for different frequency ranges [25]. For example a high frequency signal greater than 32 Hz can be separated from a low frequency signal less than 32 Hz.

The concept of Spectral Entropy is based on the Shannon Entropy [26]. The SE treats the audio signal's normalized power distribution in the frequency domain as a probability distribution, and calculates the Shannon entropy of it. Given a discrete random variable X, with possible outcomes x 1 , . . . , x n, which occur with probability P (x 1) , . . . , P (x n), the entropy of X can be defined as:

$$H(X) = -\sum_{i=1}^{n} \mathrm{P}(x_i) \log \mathrm{P}(x_i) \tag{3.6}$$

Spectral Flux

The spectral flux is defined as the squared difference between the normalized magnitudes of successive spectral distributions that correspond to successive signal frames [25]. A high value of spectral flux indicates a sudden change in spectral magnitudes, It has been suggested to be useful for the distinction of music and speech signals, since music has a higher rate of change [27]. It can be defined as:

$$F_r = \sum_{k=1}^{N/2}(|X_r[K]| - |X_{r-1}[K]|) \tag{3.7}$$

Spectral Rolloff

This feature determines the frequency in Hz below which a pre-defined percentage of the total spectral energy is concentrated. Spectral Rolloff is a measure of the amount of the right-skewedness of the power spectrum and can be referred as measurement of the critical frequency below which eighty five percent of magnitude distribution of the input is concentrated. Similar to that of spectral centroid it is a measure of spectral shape which yields values for frequencies in high ranges [23]. A spectral rolloff point can be calculated as:

$$SRP = f(N)\ where\ f(N) = (\frac{f(s)}{K})N \tag{3.8}$$

N fulfills the equation:

$$\sum_{k=0}^{N}(|X[K]| \leq TH \sum_{k=0}^{k-1}(|X[K]| \tag{3.9}$$

Here X(k) are the magnitude components, k is the frequency index and TH is a threshold between 0 and 1. A commonly used value for the threshold is 0.85.

MFCCs (8-20)

Mel Frequency Cepstral Coefficients (MFCC) are widely used in the applications of

audio recognition, they were introduced in 1980 by Davis and Mermelstein and they have been considered state of the art till today. MFCCs are the set of features which accurately describes the overall shape of a spectral envelope taking into account the nonlinear human perception of pitch, as described by the mel scale. A mel scale is a perceived scale of the pitch or the frequency at which a human listens. Since humans are good at refining any pitch change at a lower frequency than at high frequencies, implementing this scale gives us a closest approximation of what a human hears. The process of extracting MFCC features are explained in the figure

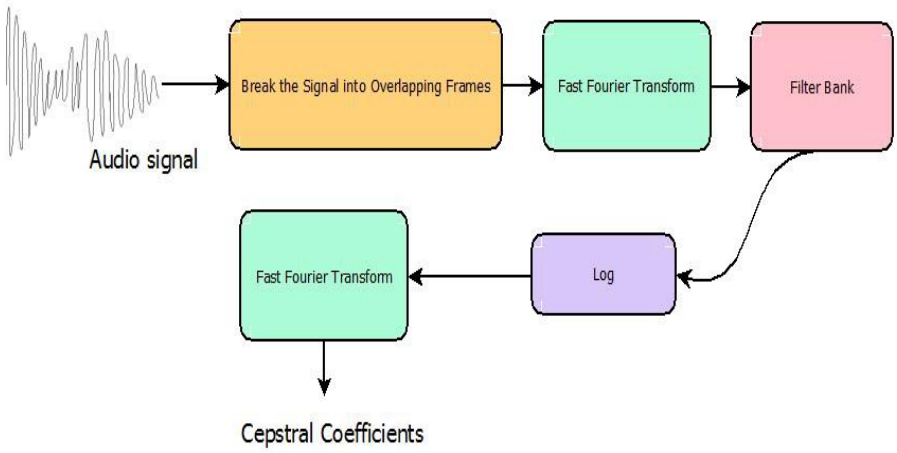

Fig. 3.2: MFCC features extraction process [4].

The cepstral coefficients are calculated from the mel-spectrum by taking the discrete cosine transform (DCT) of the logarithm of the mel-spectrum [23]. This calculation is given by:

$$c_i = \sum_{k=0}^{k-1} (logS_k).cos(\frac{i\pi}{K}(k - \frac{1}{2})) \tag{3.10}$$

Chroma Vector (21-32)

These features are also known as Chromagram, it relates to the 12 different pitch classes. A pitch class in music is a set of all pitches that are whole number of octaves, for example a

pitch class C can be all possible Cs in all octaves. These set of features are often considered for analyzing music whose pitches can be categorized into 12 categories. Distribution of energy is calculated for every Chroma vector and as a result we get an updated audio signal of twelve dimensional Chroma distribution vector. The chroma features can be computed by summing the log-frequency magnitude spectrum across octaves [28], it can be defined as:

$$C_f(b) = \sum_{z=0}^{Z-1} |X_{lf}(b + z\beta)| \tag{3.11}$$

In the above equation, X_{lf} is the log-frequency spectrum, z denotes the integer octave index, Z represents the number of octaves, b is the integer pitch class (chroma) index and β is the bins per octave.

3.3 Feature Selection

After extracting the features from pyAudioAnalysis python library [1], we are focused mainly on selecting the best subset of features which helps us in classifying our machine learning models and to remove less valuable features. A feature selection technique can be seen as a combination of operations which selects the subsets of features along with an evaluation matrix which evaluates the performance of those features. Feature Selection enables the machine learning algorithm to train faster, it reduces the complexity of a model and makes it easier to interpret, it also improves the accuracy of the model if given the right subset as input and it also reduces overfitting. The simplest algorithm would be to select each possible subset of features and calculate which subsets are most effective and results in minimum error rate. The Feature Selection technique can be divided into three categories - wrapper method, filter method and embedded method.

3.3.1 Wrapper Methods

Wrapper method uses a predictive model to score feature subset and train a model using them. Based on the deduction that we draw from the previous model we decide

to add or remove features from our subset. Wrapper method is also known as a greedy feature selection method as they focus on finding the features that results in giving the best performance model. The advantage of using wrapper method is they detect the interaction between variables and it minimizes the criterion of the learning model, thus it gives optimal subset of features which ensures lower error rate of the model. However, wrapper methods are computationally expensive and consumes more time as compared to other methods.

The wrapper method works by selecting a subset of features from the available possible feature subsets with the help of a search method. A Machine Learning algorithm is employed on the selected feature subset to evaluate the quality of these features based on the performance of the algorithm. This process is repeated with new subset of features and so on until the method evaluates the result for a specified number of features. [29]

3.3.2 Filter Methods

Filter methods does not rely on the performance of any machine learning algorithm, it selects features from a dataset independently and they are generally used as pre processing step. Features selected using filter methods can be used as an input to any machine learning model. Filter methods use statistical techniques to evaluate the relationship between each input variable and the target variable, and these scores are used as the basis to filter those input variables that will be used in the model. The advantages of using filter methods are that they are computationally inexpensive which could process abundance of features within a fraction of time and they are good at removing irrelevant, duplicate and redundant features.

3.3.3 Embedded methods

Embedded methods combine the qualities of filter and wrapper methods. It's implemented by algorithms that have their own built-in feature selection methods. Embedded methods complete the feature selection process within the construction of machine learning algorithms. In other words they perform the feature selection during the model training, this is why they are called Embedded methods. Embedded method solves the issues we en-

countered in both wrapper and filter methods. They take into consideration the interaction of features like wrapper method and they are faster like a filter method. The process of embedded method is as follows-

- Training a machine learning model.
- Deriving feature importance from this model and analyzing which features were most effective when making a prediction.
- Removing the less important features with the help of feature importance.

CHAPTER 4

FEATURE SELECTION AUTOMATION

4.1 Automation Tool

In this thesis, we are focused on automating the process of selecting the most optimal features out of all possible feature subsets from a feature selection method. In the earlier work presented by Gupta [11], the process was manually selecting a feature subset, checking its performance by providing it to a classification model. The classification model is fit with some hyperparameters and would be trained on a training dataset to obtain a training result, the same model would be used with a different smaller dataset to obtain a validation accuracy. Every step in this process requires manual intervention and the process needs to be repeated again in the same fashion with different features to verify the performance. This process resulted in a limited scope with limited feature sets and proved incapable to provide a longevity if the number of datasets increases in the future.

4.2 Automation Process

We have built a tool that automates the above process and explores the possibility of obtaining better performances with each subset of feature by working on each feature selection method. This work is carried out in the following steps:

- Features are extracted with the help of the Python library PyAudioAnalysis [1]. This open source python library extracts 34 features from a dataset which consists of .wav audio sample files of length 2 seconds. The datasets are prearranged in training, testing and validation data with all the data which were separated from each other from beehive and locations. The 2 seconds .wav file is read and stored into a numpy array using the `readAudioFile` function from the `audioBasicIO` module of pyAudioAnalysis. The features are than extracted from the numpy array in another numpy array

using `stFeatureExtraction` function of pyAudioAnalysis.

- The features are stored in a folder in the same directory of training data besides all the training data from which the features were extracted, features are placed similarly in validation and testing directories.

- The paths to training, testing and validation data of all the datasets are stored in arrays. We have developed a loop which runs on the number of datasets we are using, features are read using a read_features function, the function receives the path of extracted pyAudio features and returns a numpy array of train and test features.

- A nested loop which runs the number of times equivalent to the number of features extracted i.e 34. On every iteration different subsets of features are created ranging from a subset of size 1 to size 34.

- Individual Functions which creates ML classifier models are called from the loop. The function receives number of features, training and testing features stored in numpy arrays. A classification model(RF, KNN, SVM, LR) is trained and is fit using its fit method. After finalizing the model, we have saved it in a pickle file. Saving the model helps us by loading the model any time and use it to make predictions.

- Once the model is saved, We can predict the class for new data instances using our finalized classification model and the predict() function of scikit-learn. The same pickle file is loaded again to predict the confusion accuracy with the validation data to check the validity of our model.

The aim is to find most optimal features from our dataset which helps in classifying current instances and instances which could be later introduced in this work. The above process is repeated 34 times for each dataset and the resultant training and validation accuracies are stored in a spreadsheet. This automation process is explained in the figure 4.1 through block diagrams.

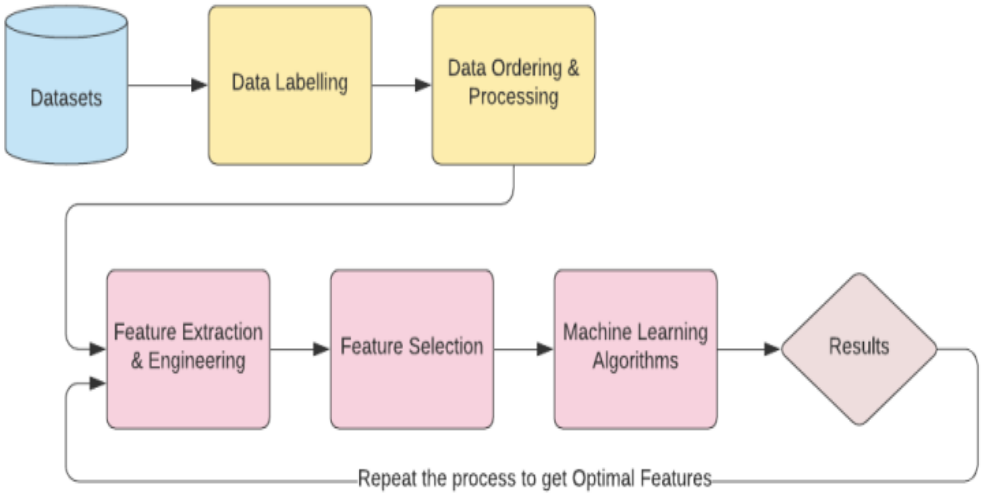

Fig. 4.1: Feature Selection Automation Process

4.3 ML Algorithms

In order to obtain effective features from our datasets, we have used some Machine Learning algorithms to check selected feature's performance. Machine Learning is a concept where machines are trained to learn, some machines learn on its own. A machine can be compared with a basic nature of humans that it improves automatically through experiences and patterns, these patterns are found within data. In case of an ML classifier to detect or classify patterns in the input data, it requires careful feature engineering that can determine optimal feature vectors for classification and regression problems. In this research we have made use of Classification kind of Supervised learning and made a distinction amongst multiple categories of data types i.e Bee buzzing, Cricket Chirping, Ambient noise and Lawn mowing sound.

4.3.1 Random Forest

Random Forest is an ML method for classification which uses multiple learning algorithm as an ensemble to obtain better predictive performance. Random forest operates by making huge number of decision trees during training time, it outputs the classification result as the mode of the result of all individual trees whereas the mean of individual trees for regression problem. As the name suggests, Random Forest adds randomness to the model by searching for the best feature among a random subset of features rather than taking all features into consideration to split a tree node. [30]

4.3.2 KNN

K-Nearest Neighbors (KNN) is a type of supervised machine learning model used for regression and classification problems which performs instance-based learning wherein learning is accomplished by comparing new problem instances with the ones already seen in training. In KNN, a function is approximated locally and relies on distances for classification by assuming that similar things are near to each other in close proximity. The KNN algorithm works in a way that after loading the data we initialize the value of K, the euclidean distance is calculated between the test data and training data, the distances are than sorted in ascending order. The first K closest neighbor's labels are picked from the sorted order and output is stored as a mean of the labels if its a regression problem or the mode of the K-labels if it is a classification problem. Higher the value of K, more stable our algorithm will be. [31]

4.3.3 Logistic Regression

Logistic Regression is a machine learning model which is used in classifying the probability of a certain event existing such as pass/fail or dead/alive. Logistic Regression is named for a function which works at the core of this method also called as logistic function or sigmoid function. Mathematically, logistic regression is designed to work on binary data which has possible two outputs and represented by an indicator variable where the two values are labelled "0" or "1". However for outputs with more than two values, we use

Multinomial Logistic Regression. This helps in generalizing Logistic Regression to a classification technique where problem instances needs to be categorized in more than two classes.

4.3.4 SVM

Support Vector Machines(SVM) algorithm was invented by Vladimir Vapnik is represented as the organization of problem instances mapped in a space and similar classes of problem instances mapped together in space separated by a visible gap from other points, newer examples are than mapped into the same space where similar data points are mapped and predicted based on the side of the gap on which they fall [32]. The main focus of the SVM algorithm is to create an N-dimensional hyperplane where N stands for number of features that distinctly classifies the data points. The objective is to find a plane with maximum distances between the data points of classes so that the future data points can be classified with more ease and confidence. The dimension of the hyperplane depends on the number of features, if the feature is one than the hyperplane can be imagined as a line or if the number of features are 2 than it can be a 2-dimensional plane. SVM works on a linear function just like the sigmoid function in Logistic Regression, depending upon the output of the function we decide the data point belongs to which class.

CHAPTER 5

EXPERIMENTS AND ANALYSIS

5.1 Methods used and their Parameters

In order to capture the most optimal features we have ran our automation on three feature selection methods - `Wrapper` methods, `Filter` methods and `Embedded` methods. The top performing features were selected on many criterion such as removing irrelevant variables, methods searching for well performing subsets of features, algorithms that performs automatic feature selection during training and difference of a model's classification accuracy affected by adding or removing certain features. This chapter presents the performance of features identified in our datasets (BUZZ1, BUZZ2, BUZZ3 and BUZZ4) using different feature selection techniques which are discussed ahead.

5.1.1 Recursive Feature Elimination

Recursive Feature Elimination (RFE) can be implemented by scikit-learn Python machine learning library. RFE is a wrapper-based feature selection technique which lets different ML algorithms to be wrapped under RFE which helps in selecting features. RFE works by eliminating weakest features until the specified number of features are selected, this can be achieved by fitting a given machine learning model, ranking the features and discarding the weakest (least-important) of them. This process is repeated recursively by fitting machine learning model every time and discarding some features till we reach a desired number. Here the feature importance is calculated by model's `coeff_` or `feature_importances_` attribute [33]. An RFE object can be created by using the feature_selection module of sklearn. The hyperparameters in an RFE technique are

- **estimator** takes an object as an input of a supervised learning model with a fit method that provides information about feature importance either through a coef_attribute

or through a feature_importances_attribute.

- **n_features_to_select** takes an integer value as input which defines the number of features to select.
- **step** defines the number of integer to remove in each iteration, default value is 1.

5.1.2 Sequential Feature Selection

Sequential Feature Selection(SFS) algorithms are a type of greedy search algorithms which are used to reduce a d-dimensional feature space into k-dimensions where k < d [34]. The goal of SFS is to automatically find that subset of features which is most important for the given problem space. the main goals of SFS is to reduce the computationally extensive load from scanning through all possible subset of features and reduce the error rate in our model by removing irrelevant and redundant features. There are two types of SFS techniques:

Sequential Forward Selection

The Sequential Forward Selection is also known as Heuristic search method. In this, we initialize the algorithm with an empty set which is our feature subset of size k where k = 0 in the beginning [34]. In the first step, the best single feature is selected using some criterion function and is inserted into the set. Then pairs of features are formed using one of the remaining feature that maximizes our criterion function and added to our feature subset, further another feature is added to our feature subset and this process is continued until a predefined number of features are selected. In this thesis, we have used this kind of feature selection technique.

Sequential Forward Selection Pseudo Code

1. Create an empty set: $Y_k = \{\emptyset\}$, $k = 0$.
2. Select best remaining feature:
   $x^+ = arg\ max_{x^+ \in Y_k}[J(Y_k + x^+)]$
3. If $J((Y_k + x^+) > J((Y_k)$
   a. Update $Y_{k+1} = Y_k + x^+$
   b. $k = k + 1$
   c. Go back to step 2.

Fig. 5.1: Pseudo Code for Sequential Forward Selection [5]

5.1.3 Relief Algorithm

Relief Algorithm was developed by Kira and Rendell in 1992 is a type of filter method which applies a feature score to each feature which can be used to separate top scoring features for feature selection [35]. Relief feature scoring is based on the identification of feature value differences between nearest neighbor instance pairs. If a feature value difference is observed in a neighboring instance pair with the same class (a 'hit'), the feature score decreases. Alternatively, if a feature value difference is observed in a neighboring instance pair with different class values (a 'miss'), the feature score increases. There are three types of Relief-based feature selection algorithms (RBA):

- **Basic Relief Algorithm** is used for classification problem with two classes.
- **ReliefF** is an extension of relief and is used for multi class problems. In this thesis, We have used ReliefF algorithm.
- **RReliefF** is similar to ReliefF algorithm and used for regression problems rather than classification.

ReliefF Algorithm

The main objective of this RBA algorithm is to estimate the quality of attributes on the basis of how well the attribute can distinguish between instances that are near to each other. In this algorithm, all the attribute weights are initially set to zero. We select a random instance Ri and find its two nearest neighbors - one of the same class which creates a hit and one from a different class which creates a miss [6]. The weight updation for a feature is reduced if Ri and H have different values and they belong to same class whereas the weight is reduces if instances Ri and M have different values and they belong to different class. This is explained better in the figure 5.2

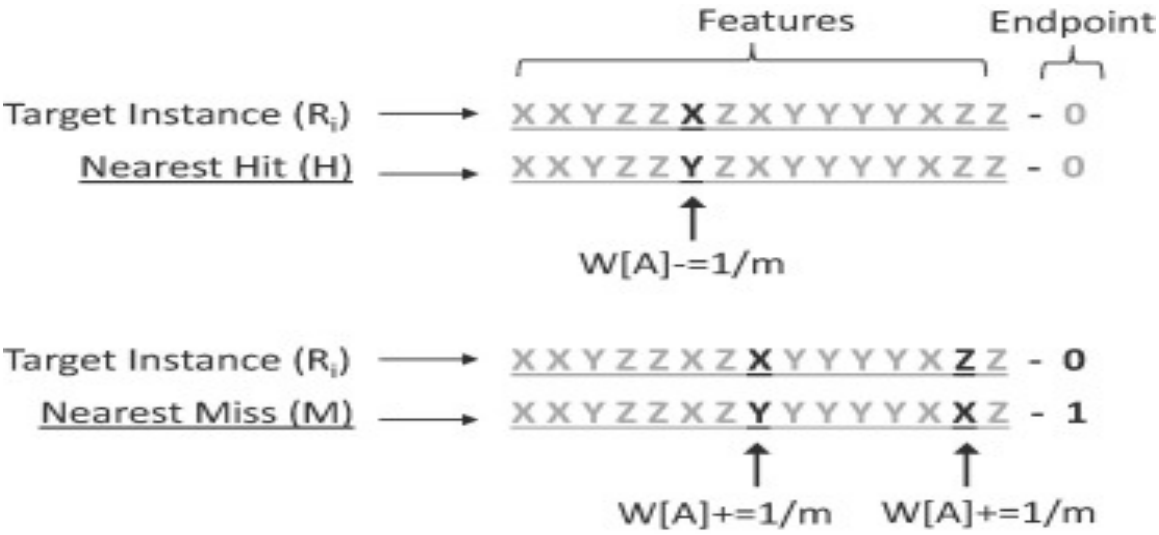

Fig. 5.2: Feature weight updation in ReliefF method [6]

ReleifF feature selection is applied using ReliefF class of scikit-rebate [36]. The parameters for the ReliefF algorithm are as follows:

- **n_features_to_select** is the number of best features to retain after the feature selection process. In this research work, the number of features are 34 which are extracted from pyAudioAnalysis, the feature scores are in descending order with the best features as the features with highest score.

- **n_neighbors** is the number of features to consider while assigning score to each features, more neighbors results in more accurate scores. We have used the default value of n_neighbor which is 100.

- **n_jobs** specifies the maximum number of concurrently running CPUs. If it is set to 1 or 2 then it uses 1 or 2 cores, we have set the value to -1 which uses all cores.

5.1.4 Random Forest Feature Importance

Random Forest Feature Importance is a Embedded type of feature selection method. We have implemented this method from RandomForestClassifier class from the ensemble module of sklearn [37]. Important features are calculated in random forest by selecting some random samples from given dataset and constructing a decision tree for each sample and get feature score prediction from each sample, a voting is been performed to score features and the feature with highest votes becomes the most important feature. We have used the default value of number of trees in this model i.e n_estimators = 100, a max_features value of log2. The table 5.7 presents the 10 most optimal features selected by this Embedded technique in all 4 datasets we have used.

5.2 Results

We have performed our experiments on 4 different feature selection methods i.e Recursive Feature Elimination(RFE), Sequential Forward Selection(SFS), ReliefF Feature selection and Random Forest Feature Importance method. In order to find the optimal feature subsets, the performance of these feature selection techniques on our datasets have been discussed below.

5.2.1 Performance on RFE

Table 5.1: RFE Results.

Dataset	Model	#features	Indices of Optimal Features
Buzz1	**Random Forest**	**5**	[5, 10, 13, 14, 15]
	Logistic Regression	**11**	[0, 3, 5, 12, 13, 14, 15, 19, 23, 29, 33]
	SVM	**10**	[5, 10, 11, 12, 13, 14, 15, 19, 20, 31]
Buzz2	**Random Forest**	**12**	[0, 3, 4, 5, 7, 9, 10, 13, 14, 15, 30, 32]
	Logistic Regression	**13**	[1, 3, 5, 7, 12, 13, 14, 15, 19, 20, 23, 26, 29]
	SVM	**9**	[5, 11, 12, 13, 14, 15, 19, 20, 31]
Buzz3	**Random Forest**	**19**	[0, 3, 5, 7, 9, 10, 12, 13, 14, 15, 16, 17, 19, 20, 21, 23, 25, 27, 30]
	Logistic Regression	**16**	[0, 1, 5, 7, 10, 12, 13, 14, 15, 16, 17, 18, 19, 20, 23, 26]
	SVM	**12**	[1, 5, 9, 10, 12, 13, 14, 16, 17, 18, 19, 20]
Buzz4	**Random Forest**	**14**	[0, 3, 4, 5, 9, 10, 11, 12, 13, 14, 15, 16, 17, 23]
	Logistic Regression	**16**	[0, 1, 5, 6, 7, 10, 13, 14, 16, 17, 18, 19, 20, 23, 26, 33]
	SVM	**11**	[3, 4, 5, 9, 11, 12, 13, 14, 18, 19, 20]

The highest classification accuracies have been achieved for each ML model and for each dataset with the number of features mentioned in table 5.1, hence making them the most optimal features. The criterion to select these features is explained in the table with

an example of 5 optimal features involved in BUZZ1 with Random Forest method. As it is clear that when one feature with index 15 was selected, the validation accuracy was 66.26 %, another feature with index 13 was selected which helped improve the classification to 69.30 %, RFE continued adding another feature and so on until the highest validation accuracy was achieved.

Table 5.2: Individual Feature Performance

Dataset	**Model**	**Indices of features**	**Testing Accuracy**	**Validation Accuracy**
Buzz1	**Random Forest**	[15]	76.64%	66.26%
		[13, 15]	92.66%	69.30%
		[5, 13, 15]	98.08%	92%
		[5, 13, 14, 15]	98.58%	99.02%
		[5, 10, 13, 14, 15]	99.82%	99.08%

As we can see from the 5.3, the best results were obtained on Buzz1 dataset. Random Forest with a subset of just 5 features out of the 34 extracted from pyAudioAnalysis gives a validation accuracy of 99%. Logistic Regression proved to be the most efficient in BUZZ2 dataset with 94.23 % and 13 features. For dataset BUZZ3 and BUZZ4, Random Forest acts as an effective wrapper with a valid accuracy of 90.88 % and 84.65 % and 11 and 15 features respectively.

Table 5.3: RFE Results.

Dataset	**Model**	**# features**	**Testing Accuracy**	**Validation Accuracy**
Buzz1	**Random Forest**	**5**	99.82%	99.08%
	Logistic Regression	**11**	98.95%	98%
	SVM	**10**	99.2%	98.8%
Buzz2	**Random Forest**	**12**	98.67%	78.13%
	Logistic Regression	**13**	90.65%	94.23%
	SVM	**9**	98.11%	80.63%
Buzz3	**Random Forest**	**19**	67.9%	85.03%
	Logistic Regression	**16**	78.47%	94.49%
	SVM	**12**	80.69%	95.23%
Buzz4	**Random Forest**	**14**	70.44%	90.15%
	Logistic Regression	**16**	68.65%	92.64%
	SVM	**11**	71.75%	95.65%

The graphs in figure 5.3, 5.4, 5.5 and 5.6 shows the most optimal selected features on all the datasets with the y-axis pointing at the classification accuracy and x-axis representing the number of features. The indices of those features are mentioned on the peaks of the graph. RFE can be only applied to models which exposes `coeff_` or `feature_importances_` attribute and hence machine learning algorithms such as K-nearest neighbor and Support Vector Machines (other than linear kernels) could not be used here.

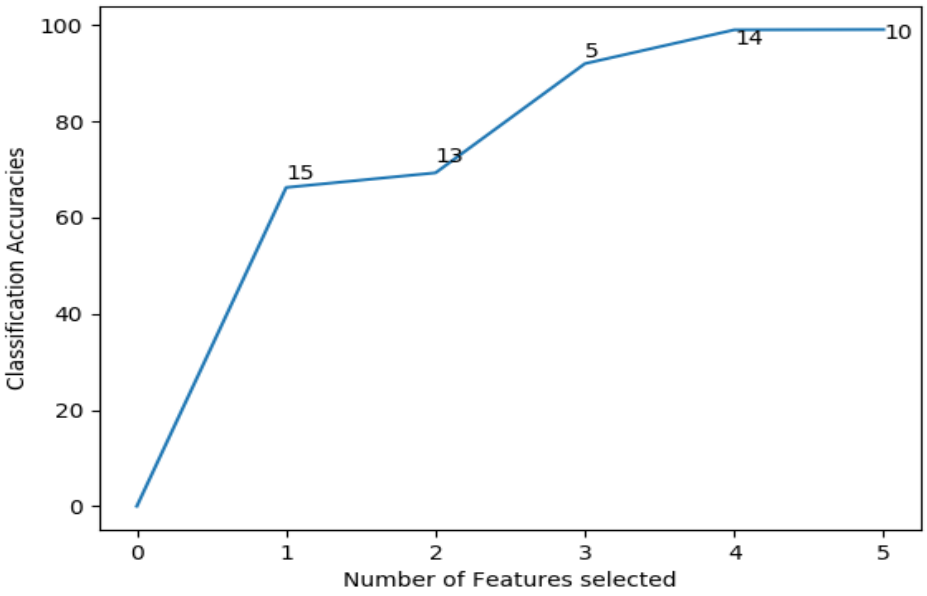

Fig. 5.3: Optimal Features selected by RFE in BUZZ1 dataset

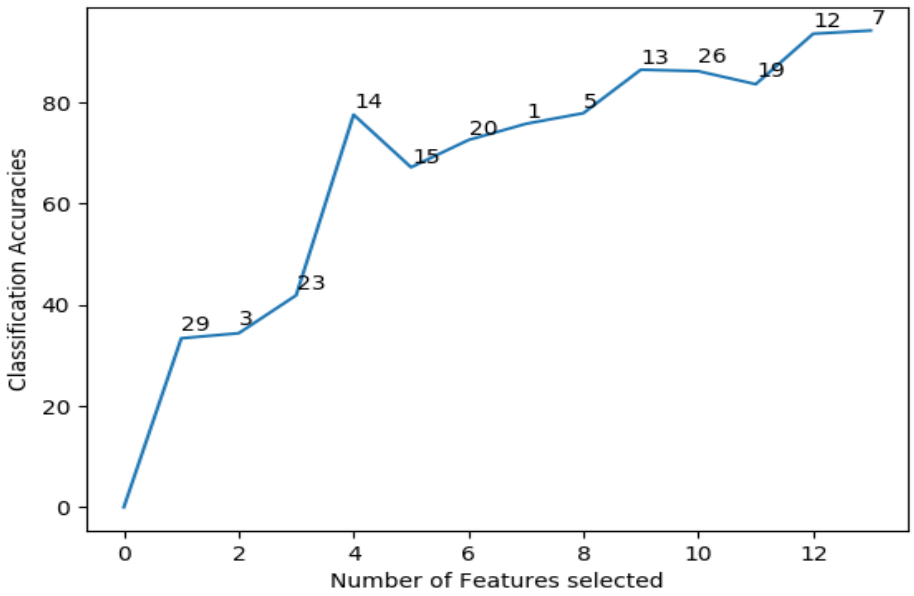

Fig. 5.4: Optimal Features selected by RFE in BUZZ2 dataset

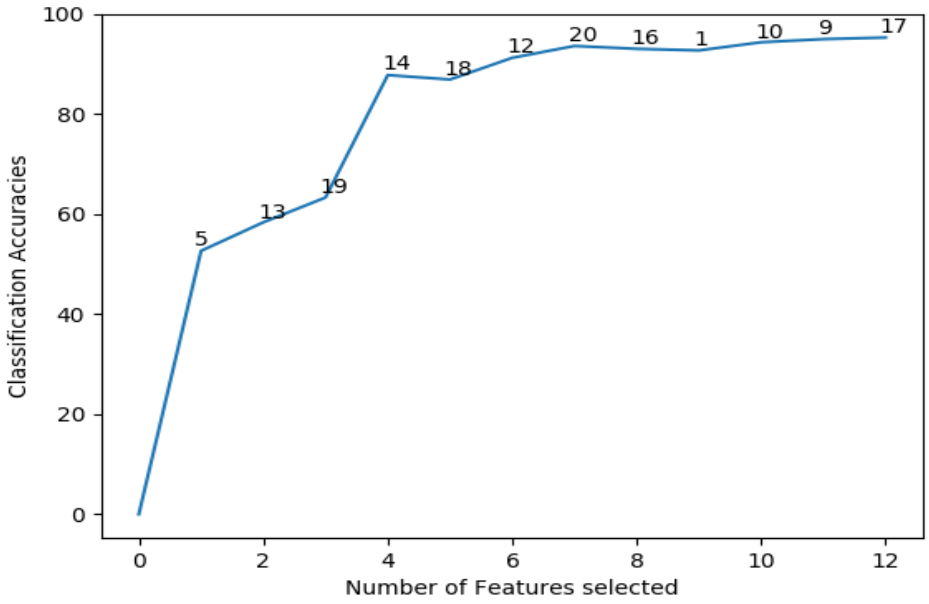

Fig. 5.5: Optimal Features selected by RFE in BUZZ3 dataset

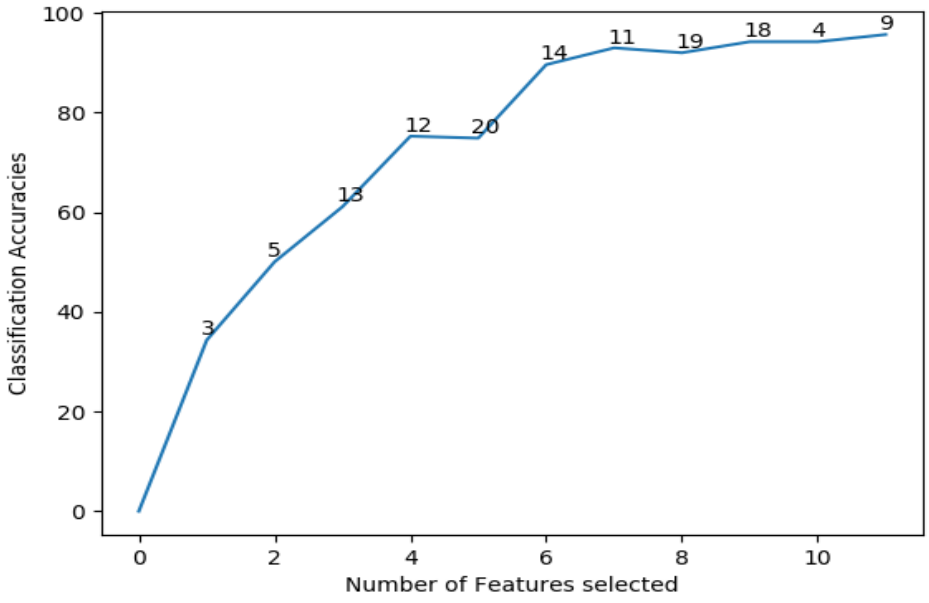

Fig. 5.6: Optimal Features selected by RFE in BUZZ4 dataset

5.2.2 Performance on SFS

Table 5.4: SFS Results.

Dataset	Model	features	Indices of Optimal Features
Buzz1	**Random Forest**	**13**	[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]
	Logistic Regression	**11**	[0, 3, 6, 9, 13, 14, 15, 18, 20, 22, 28]
	SVM	**5**	[5, 10, 13, 14, 15]
	KNN	**11**	[0, 2, 3, 5, 12, 13, 14, 15, 16, 18, 20]
Buzz2	**Random Forest**	**19**	[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14,15, 16, 32, 33]
	Logistic Regression	**18**	[1, 3, 5, 6, 8, 10, 12, 14, 15, 17, 18, 19, 20,22, 23, 24, 27, 33]
	SVM	**14**	[2, 5, 9, 10, 11, 12, 13, 14, 15, 17, 18, 19, 20, 31]
	KNN	**7**	[5, 12, 13, 14, 15, 18, 20]
Buzz3	**Random Forest**	**19**	[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 32, 33]
	Logistic Regression	**9**	[0, 1, 5, 10, 13, 14, 16, 20, 29]
	SVM	**13**	[2, 3, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]
	KNN	**6**	[5, 10, 12, 13, 14, 16]
Buzz4	**Random Forest**	**13**	[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]
	Logistic Regression	**10**	[1, 3, 5, 10, 11, 12, 13, 14, 18, 19]
	SVM	**16**	[3, 4, 7, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18,19, 20, 31]
	KNN	**10**	[5, 9, 10, 11, 12, 13, 14, 15, 16, 18]

The feature subset giving us the highest classification for each ML model and each dataset has been noted in table 5.1. The highest classification accuracies achieved in all the datasets with the most optimal features selected by SFS is mentioned in table 5.5. With just 5 features in BUZZ1 dataset SVM was able to give a classification accuracy of 98.86 %, KNN in BUZZ2 gave 92.5 % with 12 features, SVM performed best in BUZZ3 with 13 features and a classification accuracy of 91.24 % and BUZZ4 with 14 features performed best with a classification accuracy of 84.85 % on validation data.

Table 5.5: Highest SFS Classification in all four datasets.

Dataset	**Model**	**# features**	**Testing Accuracy**	**Validation Accuracy**
Buzz1	**SVM**	**5**	98.12%	98.86%
Buzz2	**KNN**	**7**	96.91%	95.43%
Buzz3	**KNN**	**6**	83.21%	97.54%
Buzz4	**Logistic Regression**	**10**	67.81%	95.08%

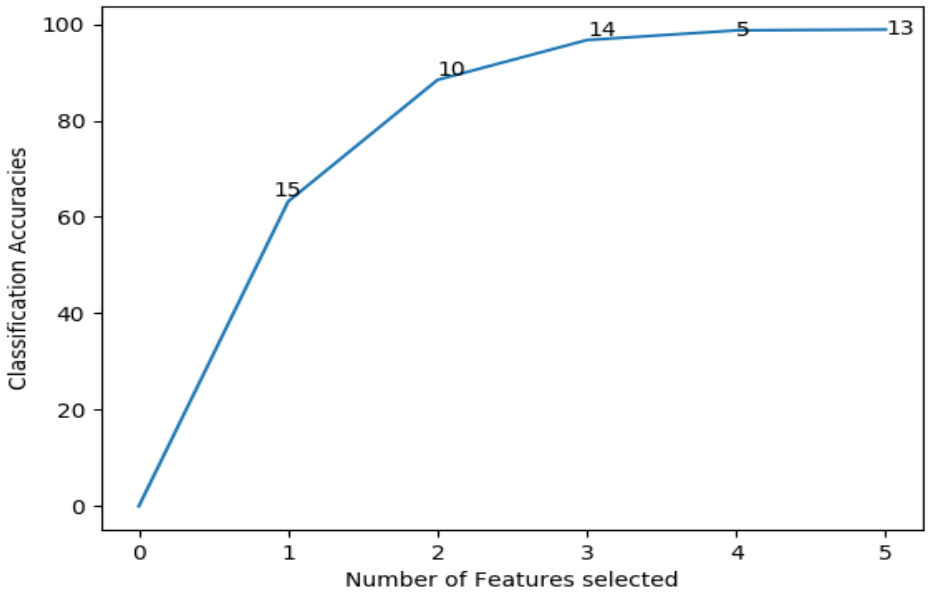

Fig. 5.7: Optimal Features selected by SFS in BUZZ1 dataset

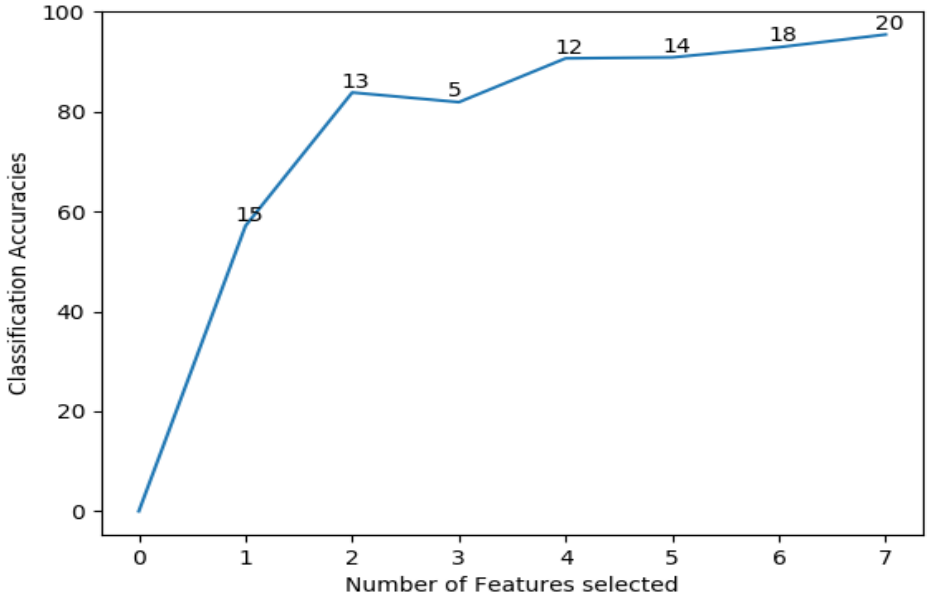

Fig. 5.8: Optimal Features selected by SFS in BUZZ2 dataset

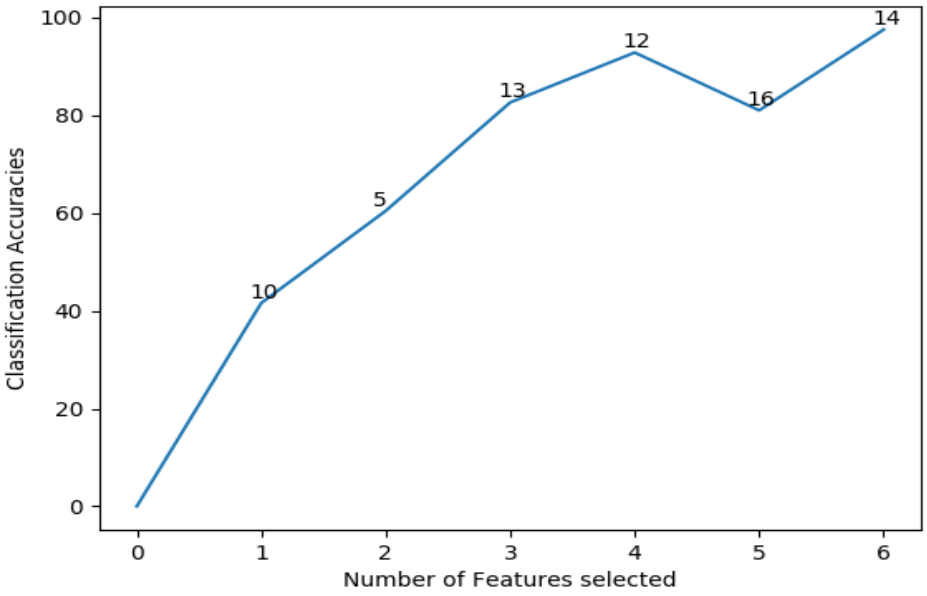

Fig. 5.9: Optimal Features selected by SFS in BUZZ3 dataset

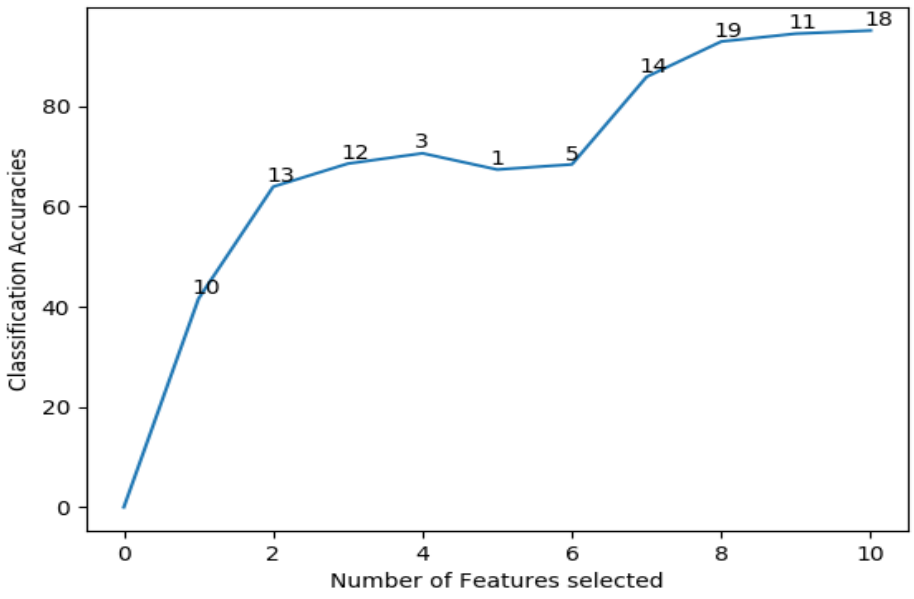

Fig. 5.10: Optimal Features selected by SFS in BUZZ4 dataset

5.2.3 Performance on ReliefF Algorithm

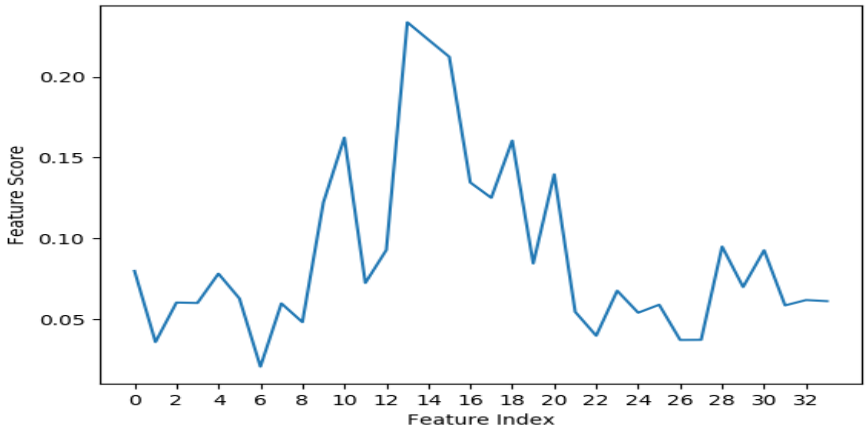

Fig. 5.11: ReliefF feature score for Buzz1

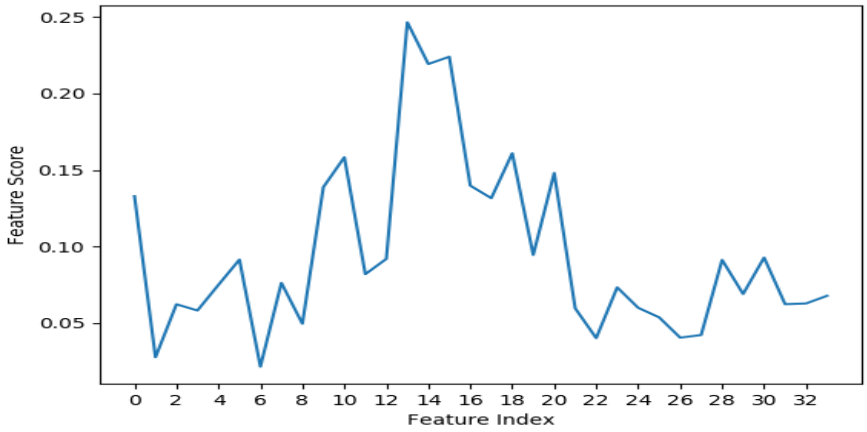

Fig. 5.12: ReliefF feature score for Buzz2

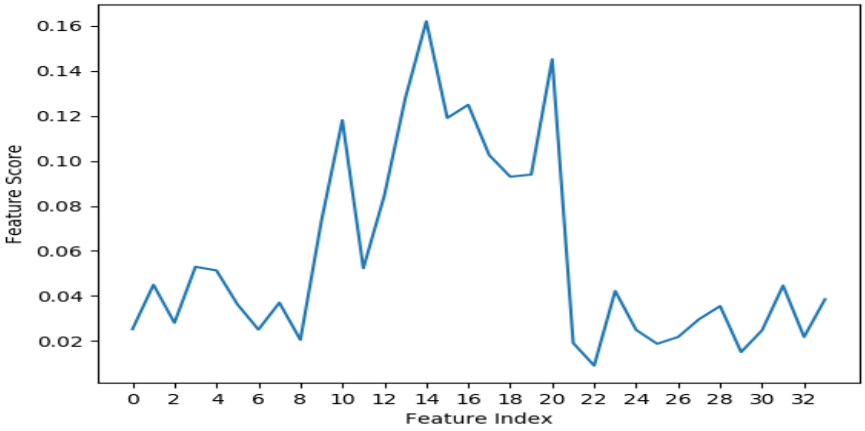

Fig. 5.13: ReliefF feature score for Buzz3

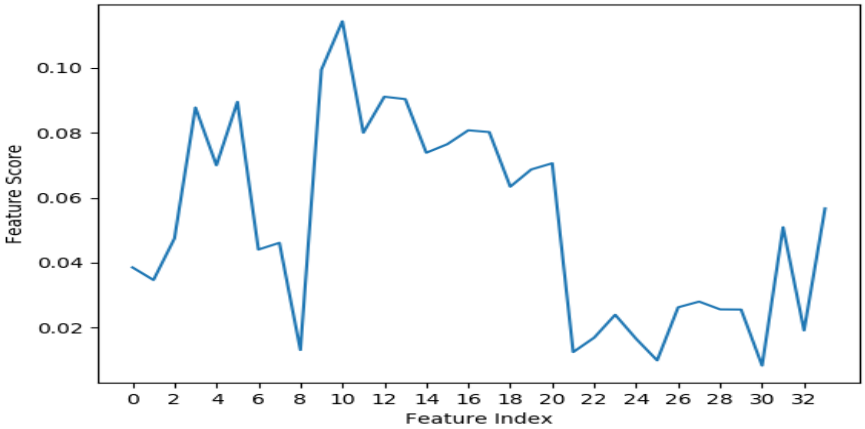

Fig. 5.14: ReliefF feature score for Buzz4

We have ran the ReliefF algorithm for all four of our dataset, the results are shown in figure 5.11, 5.12, 5.13 and 5.14 If we look at the top 10 features from all four datasets in table 5.6, the majority of features selected are MFCCs. For dataset Buzz1, Buzz2 and Buzz4 9 out of 10 features are MFCC whereas in Buzz3 all 10 features are MFCC.

Table 5.6: Best features selected from ReliefF feature selection

Dataset	**Indices of 10 best features**	**MFCCs selected**
BUZZ1	[13, 14, 15, 10, 18, 20, 16, 17, 9, 2]	9
BUZZ2	[13, 15, 14, 18, 10, 20, 16, 9, 0, 17]	9
BUZZ3	[10 ,12 ,16 ,17 ,13 ,5 ,9 ,15 ,14 ,20]	9
BUZZ4	[10 ,9 ,12 ,13 ,5 ,3 ,16 ,17 ,11 ,15]	8

5.2.4 Performance on Random Forest Feature Importance

Table 5.7: Best features selected from Random Forest Feature Importance

Dataset	**Indices of 10 most optimal features**
BUZZ1	[13, 14, 15, 9, 30, 10, 5, 7, 32, 0]
BUZZ2	[13, 14, 15, 30, 9, 7, 10, 5, 0, 3]
BUZZ3	[5, 10, 13, 9, 0, 12, 14, 7, 23, 16]
BUZZ4	[10, 5, 9, 13, 3, 12, 0, 11, 4, 16]

The feature importance graph for all the datasets along with the scores for each feature can be seen in 5.15, 5.16, 5.17 and 5.18.

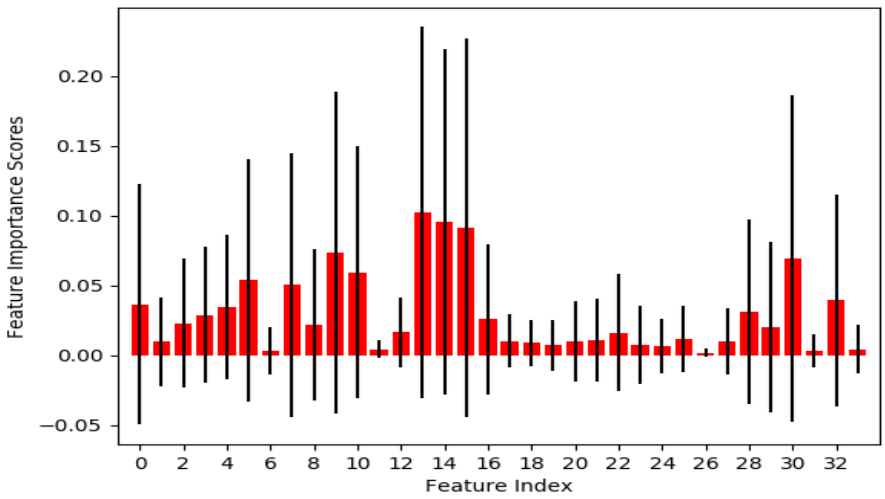

Fig. 5.15: RF feature score for Buzz1

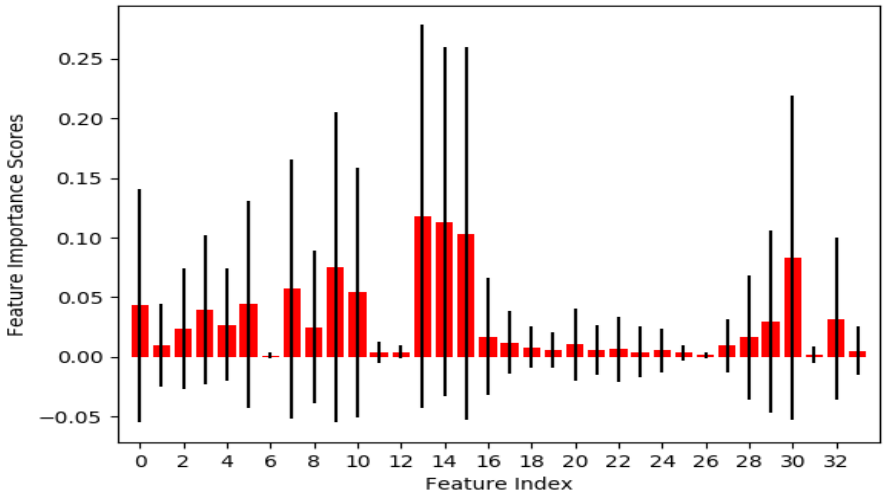

Fig. 5.16: RF feature score for Buzz2

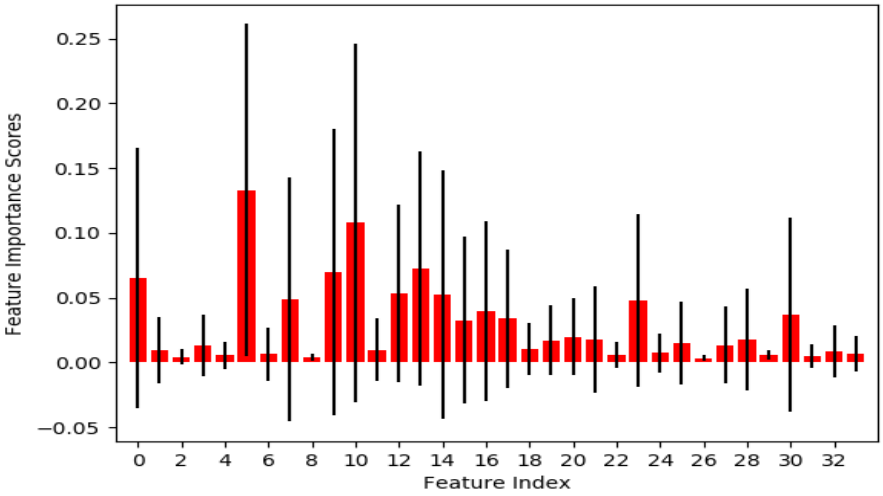

Fig. 5.17: RF feature score for Buzz3

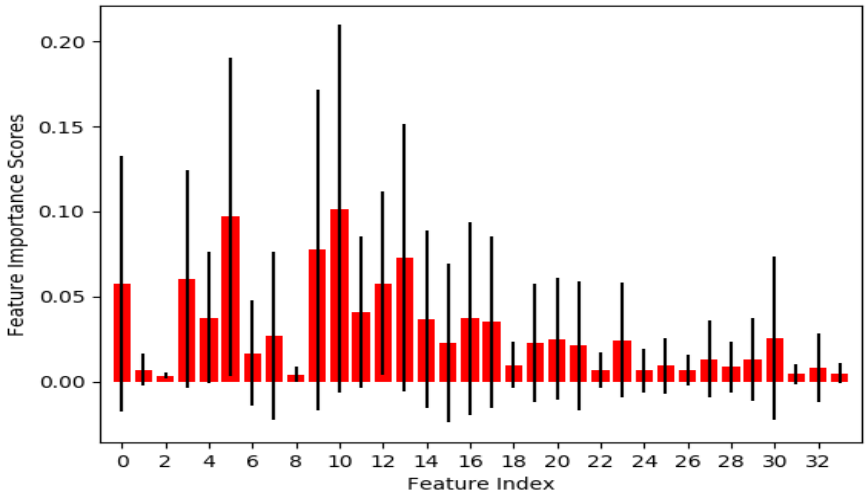

Fig. 5.18: RF feature score for Buzz4

5.2.5 Final Classification

We have found that after the best features were identified, the ML models that gave us

the best results on the validation data of all 4 datasets were - for BUZZ1 dataset Random Forest with RFE feature selection method was the best performing model with a 99.08 % classification, For BUZZ2 and BUZZ3 datasets, KNN was observed to have performed best with 7 and 6 features selected by SFS giving a classification accuracy of 95.43 % and 97.54 %. On BUZZ4 dataset, SVM with 11 features selected from RFE classified 95.65 % of the data. Table 5.8 shows the final classification accuracies received on all our datasets on testing and validation data along with the combination feature selection method and the ML model used.

Table 5.8: Final classification on all four datasets.

Dataset	**ML Model**	**Feature Selection Method**	**Testing Accuracy**	**Validation Accuracy**
BUZZ1	**Random Forest**	**RFE**	99.82%	99.08%
BUZZ2	**KNN**	**SFS**	96.91%	95.43%
BUZZ3	**KNN**	**SFS**	83.21%	97.54%
BUZZ4	**SVM**	**RFE**	71.75%	95.65%

5.3 ESC 50 Dataset

In this thesis work, we have used an external dataset ESC-50: Dataset for Environmental Sound to perform the same experiments we did with our datasets. This dataset was available to us from Piczak et al. open source github account [38]. The ESC-50 dataset is a labeled collection of 2000 environmental audio recordings suitable for methods of environmental sound classification. The dataset consists of 5-second-long recordings organized into 50 semantical classes with 40 examples per class loosely arranged into 5 major categories - Animals, Natural soundscapes water sounds, Human non-speech sounds, Interior/domestic sounds and Exterior/urban noises.

The dataset has been prearranged into 5 folds for comparable cross-validation, making sure that fragments from the same original source file are contained in a single fold. 80% of the total data was used for training our ML models, while the remaining was used for validation purpose. The feature extraction was carried out in the same way as we did earlier with the pyAudioAnalysis python library, 34 features were extracted. Piczak [39] designed a convolutional neural network for classifying the ESC-50 dataset and was able to achieve an accuracy of 64.5%. We have used two of our wrapper methods - Sequential Forward Selection (SFS) and Recursive Feature Elimination (RFE) with this dataset to present a strong set of features which could strengthen the classification accuracy.

5.3.1 Results on RFE

We ran our RFE feature selection automation on ESC50 dataset. The performance can be noted from the table 5.9 as the best result of 40.41 % was achieved with the Random Forest with 29 features whereas Logistic Regression and SVM performed with a below average accuracies of 29.16 % with 30 features and 30.2 % with 17 features respectively.

Table 5.9: Results of RFE on ESC50 dataset

Model	**Features**	**Indices of Optimal Features**	**Testing Accuracy**
Random Forest	29	[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 32, 33]	40.41%
Logistic Regression	30	[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 23, 24, 25, 27, 28, 29, 30, 31]	29.16%
SVM (Linear Kernel)	17	[2, 3, 5, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20]	30.2%

5.3.2 Results on SFS

The same style of experiment was performed with another wrapper, the Sequential Forward Selection (SFS) method performed nearly same as the RFE method, The results can be seen in table 5.10. Random Forest method performed the best with 39.79 %, Logistic Regression with 28.54 %, SVM with 30.41 % and KNN performed poorly with 20.41 % classification accuracy.

Table 5.10: Results of SFS on ESC50 dataset

Model	**Features**	**Indices of Optimal Features**	**Testing Accuracy**
Random Forest	32	[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 32, 33]	39.79%
Logistic Regression	29	[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 25, 26, 29, 32, 33]	29.16%
SVM	28	[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 24, 25, 27, 32, 33]	30.41%
KNN	29	[0, 1, 2, 3, 4, 6, 7, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33]	20.41%

It can be observed from the feature selection result that almost all the extracted features were required to receive highest classification. The results states that Piczak's convolutional neural network (CNN) with a classification accuracy of 64.5 % performed better than the

standard Machine Learning models with feature engineering. One of the reasons for an ineffective result is that the feature selection process learns the importance of a feature by continuous sampling of each data file. The ESC50 dataset is a limited data source with less amount of data available with very little number of 40 audio clips belonging to each 50 sound classes.

5.4 Comparison with DL and ML methods

We compared our most optimal results for BUZZ1, BUZZ2 and BUZZ3 datasets with the standard performance of convolutional neural network RawConvNet designed by Kulyukin et al. [14] and standard Machine Learning results obtained by Gupta [11] on the same datasets, since the results on BUZZ4 datasets are not available with these two techniques. The comparison among the validation accuracies of Convolutional Neural Network, Standard Machine Learning method and feature automation is presented in the table 5.11.

Table 5.11: Feature Automation comparison with DL and ML methods

Dataset	**Deep Learning**	**Machine Learning**	**Feature Automation**
BUZZ1	95.21 %	98.43 %	99.08%
BUZZ2	96.53 %	95.33 %	95.43%
BUZZ3	96.97 %	97.91 %	97.54%

CHAPTER 6

CONCLUSION AND FUTURE WORK

For Beehive health monitoring, this thesis concludes that it is possible to automate features in audio feature selection techniques to receive best feature subsets for audio classification which helps the bee keepers to know about the health of bee hives. Combining Feature Engineering and Machine Learning we were able to rectify the best feature subsets which performed on par with Standard Machine Learning techniques and Deep Learning methods. In the comparison with the ML and DL performances, the automation on feature selection outperformed the results of Convolutional Neural Network and ML model in BUZZ1 dataset achieving a classification of 99.08 %. On BUZZ2 dataset, the performance was on par with the DL result (i.e 96.53 %) with a slightly lower accuracy of 95.43 % whereas it performed better for a larger dataset BUZZ3 with an accuracy of 97.54 % on validation data.

We can conclude that SFS (Sequential Forward Selection) proved to be the best feature selection method with a highest classification accuracy of more than 95 % in all four datasets. In the overall analysis of features, we had extracted 34 features from the Python library of pyAudioAnalysis and explored a variety of feature selection techniques. Some of the features which stands out for achieving most optimal results are - feature index 5 which is a spectral entropy feature and proved effective with different audio frequency ranges, some of the MFCCs (Mel Frequency Cepstral Coefficients) in the range of index 8 to 20 proved to be the most effective in this research are feature indices 13, 14 and 15. These are the most commonly used features which were picked by almost all the feature selection techniques to become the part of optimal feature subset.

It can be observed from the proportionality between the classification accuracy and number of features selected in the feature selection graph presented in RFE and SFS, not all the features present in the feature subset helps in enhancing the classification, some

features can be identified to be reducing the classification accuracy. Therefore, in the future work for this research we propose to design a system to pick only those features which are proving to be the most effective for a given classification problem and use them as an input for different classification models.

REFERENCES

[1] T. Giannakopoulos, "pyaudioanalysis: An open-source python library for audio signal analysis," *PLOS ONE*, vol. 10, no. 12, pp. 1–17, 12 2015. [Online]. Available: https://doi.org/10.1371/journal.pone.0144610

[2] V. Kulyukin and S. K. Reka, "A computer vision algorithm for omnidirectional bee counting at langstroth beehive entrances." Int'l Conf. IP, Comp. Vision, and Pattern Recognition, 2016.

[3] G. BachuR., "1 separation of voiced and unvoiced using zero crossing rate and energy of the speech signal," 2008.

[4] J. Jogy. "How I Understood: What features to consider while training audio files?". [Online]. Available: https://towardsdatascience.com/how-i-understood-what-features-to-consider-while-training-audio-files-eedfb6e9002b

[5] A. Smith, O. Mendoza-Schrock, S. Kangas, M. Dierking, and A. Shaw, "An end-to-end vechicle classification pipeline using vibrometry data," vol. 9079, 06 2014, p. 90790O.

[6] R. J. Urbanowicz, M. Meeker, W. LaCava, R. S. Olson, and J. H. Moore, "Relief-Based Feature Selection: Introduction and Review," *arXiv e-prints*, p. arXiv:1711.08421, Nov 2017.

[7] C. Fibre. "6 Ridiculously Easy Ways to Help the Environment". [Online]. Available: http://www.carolinafibre.com/6-ridiculously-easy-ways-to-help-the-environment/

[8] H. Kimball. "Bees and Pollination: How Important is It?". [Online]. Available: https://preservationofhoneybees.org/essays/2016-4h-essays/item/21-hadley-kimball

[9] U. S. E. P. Agency. "Pollinator Protection". [Online]. Available: https://www.epa.gov/pollinator-protection

[10] "Colony collapse disorder". [Online]. Available: https://en.wikipedia.org/wiki/Colony_collapse_disorder

[11] Gupta C., "Feature selection and analysis for standard machine learning classification of audio beehive samples," Master's thesis, Utah State University, Logan, UT, 2019. [Online]. Available: https://digitalcommons.usu.edu/etd/7564/

[12] Cecchi, S.; Spinsante, S.; Terenzi, A.; Orcioni, S, "A smart sensor-based measurement system for advanced bee hive monitoring," *MDPI*, 2020. [Online]. Available: https://doi.org/10.3390/s20092726

[13] S. Ferrari, M. Silva, M. Guarino, and D. Berckmans, "Monitoring of swarming sounds in bee hives for early detection of the swarming period," *Comput. Electron. Agric.*, vol. 64, no. 1, p. 72–77, Nov. 2008. [Online]. Available: https://doi.org/10.1016/j.compag.2008.05.010

[14] V. Kulyukin, S. Mukherjee, and P. Amlathe, "Toward audio beehive monitoring: Deep learning vs. standard machine learning in classifying beehive audio samples," *Applied Sciences*, vol. 8, p. 1573, 09 2018.

[15] S. Xie, F. Jin, S. Krishnan, and F. Sattar, "Signal feature extraction by multi-scale pca and its application to respiratory sound classification," *Medical Biological Engineering Computing*, vol. 50, pp. 759 – 768, 07 2012. [Online]. Available: https://doi.org/10.1007/s11517-012-0903-y

[16] Y.-C. Cho and S. Choi, "Nonnegative features of spectro-temporal sounds for classification," *Pattern Recogn. Lett.*, vol. 26, no. 9, p. 1327–1336, Jul. 2005. [Online]. Available: https://doi.org/10.1016/j.patrec.2004.11.026

[17] S. Chu, S. Narayanan, and C. . J. Kuo, "Environmental sound recognition with time–frequency audio features," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 17, no. 6, pp. 1142–1158, 2009.

[18] T. Ramalingam and P. Dhanalakshmi, "Speech/music classification using wavelet based feature extraction techniques," *Journal of Computer Science*, vol. 10, no. 1, pp. 34–44, Nov. 2013. [Online]. Available: https://thescipub.com/abstract/jcssp.2014.34.44

[19] V. Kulyukin and A. Bhouraskar. Dataset buzz1 of bee buzzing, cricket chirping and ambient noise audio samples. [Online]. Available: https://usu.box.com/s/d2nyv2bw8ehs7n1ruife88qh5b4sjbnd

[20] ——. Dataset buzz2 of bee buzzing, cricket chirping and ambient noise audio samples. [Online]. Available: https://usu.box.com/s/dqpwhkb5llzif013h8w0kpr3xd5ppcne

[21] ——. Dataset buzz3 of bee buzzing, cricket chirping and ambient noise audio samples. [Online]. Available: https://usu.box.com/s/lzidbyulugsowqdstpewkjfqk3xklhdx

[22] ——. Dataset buzz4 of bee buzzing, cricket chirping, ambient noise and lawn mowing audio samples. [Online]. Available: https://usu.box.com/s/6luscd50q0udp778z7phrq5aut4mmkz2

[23] G. T. Abreha, "An environmental audio-based context recognition system using smartphones," August 2014. [Online]. Available: http://essay.utwente.nl/66444/

[24] "Spectral Centroid". [Online]. Available: https://en.wikipedia.org/wiki/Spectral_centroid

[25] A. Lahiri. "RAudioAnalysis". [Online]. Available: https://github.com/a1shadows/raudioanalysis

[26] MathWorks. "Spectral Entropy". [Online]. Available: https://www.mathworks.com/help/signal/ref/pentropy.html#mw_2fe5e61a-bb74-4d74-a10d-1233063eaee0

[27] A. Al-Shoshan, "Speech and music classification and separation: A review," *Eng. Sci*, vol. 19, pp. 95–133, 01 2006.

[28] J. P. Bello, "Chroma and tonality," *MPATE-GE 2623 Music Information Retrieval*. [Online]. Available: http://www.nyu.edu/classes/bello/MIR_files/tonality.pdf

[29] Y. Charfaoui. "Hands-on with Feature Engineering Techniques: Advanced Methods". [Online]. Available: https://heartbeat.fritz.ai/hands-on-with-feature-engineering-advanced-methods-in-python-for-machine-learning-e05bf12da06a

[30] N. Donges. A complete guide to the random forest algorithm. [Online]. Available: https://builtin.com/data-science/random-forest-algorithm

[31] O. Harrison. Machine learning basics with the k-nearest neighbors algorithm. [Online]. Available: https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761

[32] "Support-vector machine". [Online]. Available: https://en.wikipedia.org/wiki/Support-vector_machine

[33] "sklearn feature selection RFE". [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.RFE.html

[34] "Sequential Feature Selector". [Online]. Available: http://rasbt.github.io/mlxtend/user_guide/feature_selection/SequentialFeatureSelector/

[35] K. Kira and L. A. Rendell, "The feature selection problem: Traditional methods and a new algorithm," in *Proceedings of the Tenth National Conference on Artificial Intelligence*, ser. AAAI'92. AAAI Press, 1992, pp. 129–134. [Online]. Available: http://dl.acm.org/citation.cfm?id=1867135.1867155

[36] "Benchmarking relief-based feature selection methods for bioinformatics data mining". [Online]. Available: https://doi.org/10.1016/j.jbi.2018.07.015

[37] "sklearn ensemble RandomForestClassifier". [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[38] K. J. Piczak, "ESC: Dataset for Environmental Sound Classification," in *Proceedings of the 23rd Annual ACM Conference on Multimedia*. ACM Press, pp. 1015–1018. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2733373.2806390

[39] ——, "Environmental sound classification with convolutional neural networks," *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1–6, 2015.