Fraud Detection: Using Artificial Intelligence to Identify Suspicious Persons Over the Phone

An Interactive Qualifying Project Report:
submitted to the Faculty
of the

WORCESTER POLYTECHNIC INSTITUTE
in partial fulfillment of the requirements for the
Degree of Bachelor of Science by

Daniel Aksman
Zackary Fitzgibbon
Alec Kneedler

In collaboration with:

Alex Malkhasov
Arseniy Pozdniakov

and

Professor Svetlana Nikitina, Advisor
Professor Anton Losev, Advisor
Professor Mikhail Koroteyev, Advisor

This report represents the work of one or more WPI undergraduate students submitted to the faculty as evidence of completion of a degree requirement. WPI routinely publishes these reports on its web site without editorial or peer review.

Abstract

Banks in Russia have experienced a rise in client complaints regarding fraudulent monetary transfers made under the influence of social engineering. In collaboration with students at the Financial University in Moscow, the team used artificial intelligence to identify a speaker's emotional state using acoustic markers. Three models were developed and tested with eleven datasets to enhance accurate identification. Based on the results, the authors provide recommendations on the most salient vocal features and classification models to aid with fraud detection.

Acknowledgements

Financial University Sponsors:

Professor Anton Losev, *Advisor*
Professor Mikhail Koroteyev, *Advisor*

Financial University Student collaborators:

George Pankov
Alex Malkhasov
Arseniy Pozdniakov
David Akzholbaev
Karina Musina
Maxim Korotkov
Olga Skiba
Konstantin Antonov

Worcester Polytechnic Institute:

Svetlana Nikitina, *Advisor/Associate Teaching Professor of English*
John F Zeugner, *Professor Emeritus*
Lori Steckervetz, *Library Consultant*
Benjamin Cross, *Electrical Engineering Student*

We are very grateful for your assistance on this project. Thank you!

Best,

Alec, Dan, and Zack

Table of Contents

Table of Figures

Executive Summary

The term cybercrime traditionally inspires thoughts of a lone wolf carefully picking through layers of security with complicated programs and algorithms. However, the vast majority of cybercrime now relies on common deception. In 2018, 98% of all cybercrimes included social engineering in some form (PurpleSec, 2020). Typical methods include downloading malware onto a victim's device under false pretexts via email attachments or pop-up advertisements with the goal of extracting personal information for use in the subversion of security protocols. This approach of fooling the system rather than finding and exploiting weaknesses is easier to implement on a larger scale. Once the victim's personal data has been acquired, the scammer can gain access to accounts and more sensitive information which enables fraudulent activity. Frequent social engineering objectives include gaining access to personal information.

In Russia, there is currently no method in place capable of automatically detecting fraudulent or fraud-influenced behavior over the phone. Recent rises in social engineering-based complaints to Russian banks have highlighted the problems present in the current manual system. Currently, banks utilize phone operators to screen incoming phone calls from clients, and then rely on their personal judgement to determine whether the caller is attempting fraud or influenced by a fraudulent third party. This method is not very reliable for detecting fraud, as phone operators often make mistakes, and mistakes can cause the financial institution to lose valuable time and money. If a caller asks to take money out of his or her account and transfer it into a contraband account, it becomes the bank's fault when the caller returns to retrieve their stolen money. It is also common for Russian banks to refer a caller to a psychology expert for further questioning if the operator suspects a threat of fraud. A more reliable system for detection would decrease unnecessary caller questioning, saving the time of the expert psychologists and time and money of the bank who hired them. Such a system would utilize specific vocal characteristics of callers to analyze their emotional state, and then determine if that emotional state matches that of a fraudulent caller.

In collaboration with students and professors at the Financial University under the Government of the Russian Federation, the team worked to address the above problems. A group from Worcester Polytechnic Institute conducted research into speech emotion recognition. Concurrently, students from the Financial University developed a system for client identification using vocal signatures. Side by side, the two teams tackled the issue of emotion classification and made progress towards the creation of a fraud detection system.

The development of a speech emotion recognition system has four main objectives. The first to be addressed was the process of data collection, followed by feature selection, preprocessing methods, and AI architecture determination. Throughout the development process detailed below, all four of these sections evolved as new results proposed novel solutions and opportunities for research.

As the training and validation of any machine learning algorithm is dependent on data, the first milestone set by the team was to acquire or create a sizable dataset. It was established that

the best data for this application would be sound bites (specifically .WAV form) of subjects uttering short phrases exhibiting various emotions.

Once sufficient data had been collected to allow early testing, the team shifted to processing and extracting features. Preprocessing steps such as denoising the files were hypothesized to improve detection rates by reducing the impacts of variables deemed unimportant to the system. This was also important in reducing the variance between files sourced from different databases. As addressed in chapter 2, acoustic features are qualitative descriptors of sound, represented as numeric values. They are stored either in an array or as a single value for each frame inside the audio sample. Some common acoustic features include pitch, amplitude, and waveform (sine, saw, square, random etc.). To improve system performance, other acoustic features were extracted and studied. After extraction, the features that showed the highest amount of correlation between certain emotions and/or between other features were selected for model building. The features extracted from each file were:

- Chroma Standard Deviation
- Chroma Mean
- MEL Standard Deviation
- MEL Mean
- Spectral Centroid
- MFCC Standard Deviation
- MFCC Mean
- MFCC 1st Derivative
- MFCC 2nd Derivative
- Root Mean Square Mean
- Spectral Rolloff Mean
- Spectral Rolloff Range
- Zero Crossing Rate
- Speaking Rate
- Jitter

From previous research, three AI architectures were selected for analysis. As new data was acquired and vocal features were altered, the analytical models were assessed for accuracy and efficiency. Initially the team hypothesized that the mesh neural network would provide the best results. The random forest, and logistical regression models were also evaluated. The results are shown below in table 1.

Dataset	RF	LR	MLP
RAVDESS with 6 features	35.24%	28.64%	25.1%
All databases with 6 features	39.94%	23.38%	37.41%
All databases with 13 features	42.72%	31.86%	36.79%
13 features normalized	22.24%	24.45%	32.59%
13 features log base 2	42.96%	34.76%	40.86%
13 features inverse	42.90%	32.61%	38.66%
all databases with 15 features	42.76%	31.52%	37.73%
15 features log base 2	43.82%	34.31%	39.57%
15 features inverse	43.45%	33.82%	38.15%
15 features log base e	43.20%	35.07%	39.71%
10 features	43.11%	29.49%	36.87%

Table 1 Model accuracy by dataset

While evaluating the results of the project, the team developed a set of recommendations for future research. Specifically, next steps are suggested regarding databases, acoustic features, and artificial intelligence models. The team believes that system performance could be improved by incorporating the EESDB, TESS, SEMAINE, and DES databases into the training dataset. Evening out emotional distribution in the dataset may also prove valuable in increasing the accuracy of the artificial intelligence models. Regarding acoustic features, the team recommends expanding the feature-set to include new inputs such as wavelet packets, bispectrality features, and/or bicoherence features. To process these increased volumes of data and features, more complicated machine learning models will be required. A neural network could perform well as they excel in applications involving complex data.

This project is only a steppingstone towards the overall goal of detecting the influence of social engineering, but it represents progress towards a novel solution. Ideally, the models could have been trained directly on data representing socially engineered bank patrons and normal transactions. However, due to the absence of a preexisting social engineering audio database and the sensitivity of the information required, securing such data was impossible. Based on the team's literary review, emotional analysis is often the most effective way to discern a socially engineered person. Typically, effected bank patrons exhibit a combination of anger, fear, stress, and anxiety more often than those making routine transactions. For this reason, the team believes that identifying individuals in distress will provide a strong basis for a social engineering detection system.

While this project does not fully address the issue currently facing banks worldwide, the team hopes this rudimentary system may contribute to the final goal of creating an effective AI solution to reduce the risk of fraudulent transaction.

Chapter 1: Introduction

The term cybercrime traditionally inspires thoughts of a lone wolf carefully picking through layers of security with complicated programs and algorithms. However, the vast majority of cybercrime now relies on common deception. In 2018, 98% of all cybercrimes included social engineering in some form (PurpleSec, 2020). Typical methods include downloading malware onto a victim's device under false pretexts via email attachments or pop-up advertisements with the goal of extracting personal information for use in the subversion of security protocols. This approach of fooling the system rather than finding and exploiting weaknesses is easier to implement on a larger scale. Once the victim's personal data has been acquired, the scammer can gain access to accounts and more sensitive information which enables fraudulent activity. Frequent social engineering objectives include gaining access to bank accounts or medical records.

There is currently no good method to automatically detect fraudulent or fraud-influenced behavior over the phone in Russia. Over the last decade, there have been an increasing number of social engineering influenced fraud cases perpetrated on Russian banks, and the current method to detect these cases is not automatic and not very successful. Currently, banks utilize phone operators to screen incoming phone calls from clients, and then rely on their personal judgement to determine whether the caller is attempting fraud or influenced by a fraudulent third party. This method is not very reliable for detecting fraud, as phone operators often make mistakes, and mistakes can cause the financial institution to lose valuable time and money. If a caller asks to take money out of his or her account and transfer it into a contraband account, it becomes the bank's fault when the caller returns to retrieve their stolen money. It is also common for Russian banks to refer a caller to a psychology expert for further questioning if the operator suspects a threat of fraud. A more reliable system for detection would decrease useless caller questioning, saving the time of the expert psychologists and time and money of the bank who hired them. Such a system would utilize specific vocal characteristics of callers to analyze their emotional state.

In collaboration with students and professors at the Financial University under the Government of the Russian Federation, this team assembled to address the above problems. A group from Worcester Polytechnic Institute conducted research into speech emotion recognition. Concurrently, students from the Financial University developed a system for client identification using vocal signatures. Side by side, the two teams tackled the issue of emotion classification and made progress towards the creation of a fraud detection system.

Image 1 The Financial University under The Russian Federation

The goal of this interactive qualifying project is to improve the ability of Russian banks to detect fraud. To complete this goal, the team must accomplish the following objectives:

- Acquire data from online databases for use in training and validation
- Identify key vocal features to extract and analyze
- Implement preprocessing measures to reduce noise
- Explore Artificial Intelligence architectures
- Develop a working emotion classification algorithm prototype

To achieve these goals, the team first researched current technologies relating to identifying individuals by voice. The team also investigated vocal features exhibiting high correlations to emotional states. In coordination with The Financial University under The Russian Federation, four artificial intelligences were developed, one for client identification and three to detect emotions to aid in the detection of socially engineered patrons.

Chapter 2: Background and Literary Review

2.1: Roadmap

The following chapter contains the background detailing the impact of social engineering and recent attempts to detect it. The importance of recent developments in the fields of sociology and artificial intelligence in addressing this issue cannot be overstated. The principle topic of defining and categorizing social engineering will first be addressed, followed by its implications on the modern world. Once the concept of social engineering has been established, existing forms of detection, both autonomous and manual, will be explored. Finally, the technology behind many artificial intelligence (AI) powered approaches to detection will be introduced and the near future of social engineering practices will be discussed.

2.2: Social Engineering

Social engineering is using deception to persuade a person to disclose private information, or to provide unauthorized entry to a network or system (Social Engineering, 2020). This process of unauthorized access is known as "piggybacking" (Kubasek et. al, 2017). Clever deception is what separates social engineering from traditional attacks in which the criminal targets the source directly. In social engineering, the criminal deceives another person who has access to the source network to get them around a security barrier. For example, if a criminal wanted to procure unauthorized access to a computer system, they would deceive a worker into unknowingly providing the password to get around the security (Laribee, 2006). Common types of social engineering methods are phone scams, phishing, pop-ups, and pretexting (Salahdine, 2019).

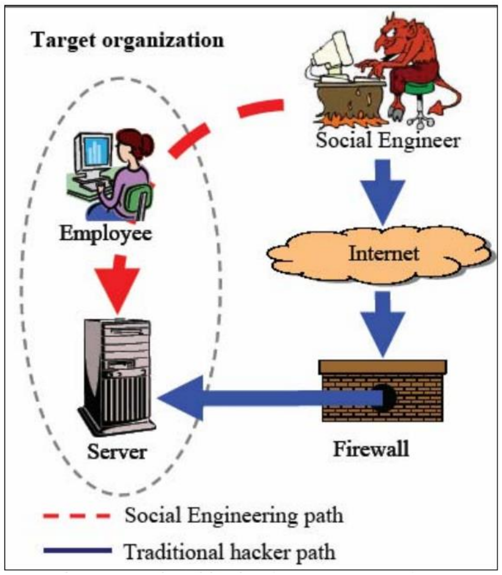


Figure 1 The paths taken by a social engineer and a traditional hacker (Laribee, 2006)

A phone scam involves the assailant coaxing a victim into giving up personal details, such as credit card information. These scams usually start with the attacker buying information that was illegally obtained through a traditional attack. Once the criminal has the victim's information, they tailor a story for their chosen target. In one case, a group of scammers bought the personal information of 1,800 high school students in China. Next, they randomly selected students to call and falsely inform them they had received a scholarship. The hook for their scam was that the scholarship was only redeemable for one day. If someone was fooled by phone scam and contacted the fake financial office that was mentioned in the phone call, they instructed the victim to deposit their tuition money into a bank account and then they would receive the scholarship. Once the money was in the account, the criminals withdrew the victim's money and disappeared. This was so successful because the victim needed scholarship money in order to attend college, and the criminals claimed to be from an accredited scholarship organization. The criminals were able to create a sense of trust and urgency by saying they were from a real organization and stating that 'Today is the last day!'. This advanced manipulation caused the victim to act irrationally and fail to double check their actions (Zheng, 2019).

A phishing attack, unlike a phone scam, is a broad attack, it is not aimed at a specific victim (Zheng, 2019). Phishing is normally done through an email sent to a large group of people with a hyperlink that goes to a website that is used to steal information. For example, the email would state that you had won a free iPhone, and all you had to do is enter your address and credit card information and it would be shipped to your house. The website would take that information and store it on a database for the phisher to use at a later point. Analogous to a phone scam, the victim is deceived into giving up personal info (Salahdine, 2019).

A pop-up window is a broad attack like phishing. There are two main types: those that steals the user's information and those that install viruses onto the user's computer. A common implementation of the first prompts the user to reenter their login information when visiting an infected site. These are most frequently employed on websites holding valuable information such as banking pages. The other common pop-up attack features a warning that the user's device has been infected with a malware and prompts them to download free anti-virus software to resolve the issue. Unsurprisingly, installing the recommended software infects the computer with a virus. In both cases, detail is the cornerstone of a successful attack. The criminal will make the pop-up look official and include as many small details as they can. The more detail that is included the more people that will fall for the attack (Salahdine, 2019). Most of the scam's detail contributes to the author's fabricated authority. The establishment of the fictional scenario or persona used by the attacker is known as pretexting.

Pretexting is present in almost all forms of social engineering. In phone scams, phishing, and pop-up windows pretexting is the main deception the victims fall for. This tool is used to create a false but compelling persona or scenario that will peak a person's interest and ultimately deceive them (Salahdine, 2019). In the phone scam example, the pretext was that the victim received the fake scholarship. In the phishing example, it was the free iPhone. A typical pop-up window would establish the pretext that the user had been logged out of their bank's website. In all these cases, the user was presented with a falsehood which compelled them to act in a way which would give the orchestrator personal information.

2.2.1: The Breadth of the Problem

Social engineering is a global threat. Any individual with internet access on any device can be targeted. In the United Kingdom, there was an increase of social engineering in 2009. Most of these attacks consisted of fraudulent criminals impersonating a bank client and deceiving a customer service representative to forward phone calls to the criminal's phone. Next, they use the information gained to log into the victim's bank account.

If the bank calls the victim with questions about suspicious transactions, the attacker can answer as the victim's phone is no longer getting the calls. The same method has been used in the United States but due to differences in laws they were not as successful (Sausner, 2009). In 2018, 49% of attacks on Russian banks used some sort of social engineering where only 36% of attacks were hacking attacks. One survey found that 75% of employees at Russian banks click on links found in phishing emails. Another 25% of employees have entered their credentials into fake authentication forms (Shmyrova, 2019).

2.2.2: The Impact of Social Engineering

Unlike most forms of criminal theft, social engineering utilizes forms of manipulation to exploit individuals. In 2004, the U.S Department of Justice predicted that one in three people will be affected by social engineering at least once in their lifetime (Workman, 2007) Even though people are aware of this threat, the majority of people will do nothing about it. Financial loss is the primary concern of social engineering attacks. In the UK, it was reported in 2009 that 1.2 billion British pounds were stolen as a result of identity theft, and another 22 million pounds were lost as a result of phishing email scams. There are also numerous recorded cases of people being tricked into giving up personal banking information, such as credit or debit card numbers.

Due to our gullible nature and false sense of security, we often fall victim to scams, fraud, information extortion, or false payments through uncertain withdrawals. Recently, cybersecurity technology has been pushing the boundary on what it means to be safe online. Computer programs like Norton can easily detect and diagnose malicious viruses or other forms of hacking that are designed to steal people's private information. Banks have strong cyber security that will usually keep your bank account safe from intruders. However, as information security has evolved, so have methods to bypass it.

Social engineers seemed to have realized that a simple way to extort someone for their personal (or company) information is by simply asking them. This could be through emails, in person, or over the phone. It is well known that the more interpersonal the interaction is, the more the receiver is willing to trust the pretense of the interaction. Due to this, various methods of detecting fraudulent requests have been implemented by workplaces. The most common method of detecting an unauthorized request is through personal training. Although this helps, it is still limited to an individual's intellect and experience (Sinigoj, 2004).

With the many distractions of the 21st century, social engineering attacks have become possible and very easy to fall prey to due to the constant bombardment of information the average person receives. Multiple automatic methods of social engineering detection have been developed to combat the issue without relying on human judgement. This problem has led to the study of machine learning, which has been applied in various forms and has evolved greatly over the past few decades.

2.3: The History of Social Engineering Detection Systems

One of the oldest forms of automated social engineering detection arose from a digital forensics conference proceeding authored by Michael Hoeschele in 2005 called “Detecting Social Engineering”. In the proceeding, the author described a system – Social Engineering Defense Architecture (SEDA) – which is designed to detect fraudulent behavior through phone lines and log data for future use. This system uses “text-independent voice signature authentication” to detect social engineering. The system creates a digital profile consisting of various data points recorded from the call of the attacker. Some of these identifying data points include the name used by the attacker, position or status, time recorded, type of request (money transfer, password, certain personal access, etc.), and most importantly his or her “voice signature”. A voice signature is a computer’s way of remembering and recalling a person's voice through sound and intonation (see bottom for in depth discussion of voice signatures).

The SEDA complies a database of various names and links them with voice signatures to prevent social engineers from calling the same victim multiple times, or at least to recognize if someone is trying to extort information. The SEDA uses a decision tree, which is a very common form of social engineering detection in recent advances, to track a call step by step. Tests the SEDA tree would run include checking to see if the call is from inside or outside the company, if the caller’s supposed knowledge of the business or client matches real information about the business or client, direct requests for a usernames and passwords, or if the caller’s proclaimed identity is in the benchmark database of fraudulent callers. Although this system is somewhat primitive in the technology used to detect social engineering, it still proved extremely useful because it created a working database of call logs that can be used for future forensic analysis or social engineering detection systems. The next system that will be discussed, which was developed in 2009, used the same benchmark data created by Hoeschele’s SEDA in 2005 (Hoeschele, 2005).

In 2009, Sandouka et al. developed a system for detecting social engineering by simulating a neural network with a computer program. They used machine learning technology to simulate a brain that uses a series of algorithms to determine if a request is fraudulent. Sandouka used an existing database of benchmark data obtained from the previously discussed study done by Michael Hoeschele. The machine takes benchmark data, extracts important features from the data, sends the features to the neural network for learning, and provides a decision on whether the data signifies a threatening call. This system is designed to work similarly to the human brain, and it can detect patterns in speech according to the process depicted below in figure 2. According to the authors, a neural network-based SE detection system would enable “fast, real-time and automatic” SE detection and would allow for objective decision making, as opposed to human subjective decision making. In their conclusion, the authors demonstrated that the addition of input variables assisted in the success of their device; however, they mentioned that the data used was comprised of fake scenarios and it would have been much more useful if they had real life interactions between callers and victims.

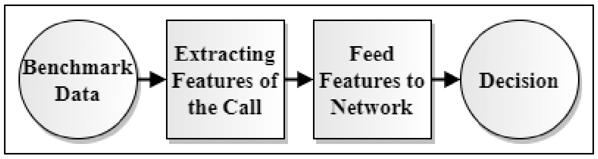


Figure 2 Functional diagram of the neural network (Sandouka, 2009)

A more recent social engineering detection system is the Social Engineering Attack Detection Model 2, or SEADM2. This system was designed by Francois Mouton and his research team (2017). The SEADM2 works by prompting the application user with a series of yes or no questions in a decision tree to obtain a result of whether the caller is attempting an attack.

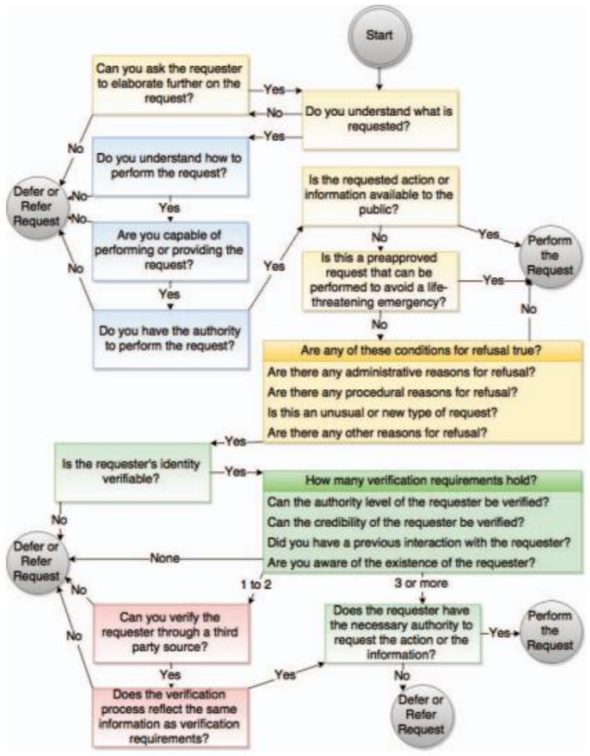

Start
Do you understand what is requested?
Can you ask the requester to elaborate further on the request?
Yes
No
Yes
No
Do you understand how to perform the request?
Defer or Refer Request
No
No
Yes
Are you capable of performing or providing the request?
No
Yes
Do you have the authority to perform the request?
Yes
Is the requested action or information available to the public?
Yes
Perform the Request
No
Is this a preapproved request that can be performed to avoid a life-threatening emergency?
Yes
No
No
Are any of these conditions for refusal true?
Are there any administrative reasons for refusal?
Are there any procedural reasons for refusal?
Is this an unusual or new type of request?
Are there any other reasons for refusal?
Yes
Is the requester's identity verifiable?
Yes
No
How many verification requirements hold?
Can the authority level of the requester be verified?
Can the credibility of the requester be verified?
Did you have a previous interaction with the requester?
Are you aware of the existence of the requester?
None
Defer or Refer Request
1 to 2
3 or more
No
Can you verify the requester through a third party source?
No
Yes
Does the verification process reflect the same information as verification requirements?
Yes
Does the requester have the necessary authority to request the action or the information?
Yes
Perform the Request
No
Defer or Refer Request

Figure 3 The SEADM2 prompt flowchart (Mouton, 2017)

The benefit of this system is that it can be used to detect both verbal and textual forms of social engineering, meaning that it can combat both calls and phishing. Although the level of technology seems to be a step back when compared to the SEDA or neural network systems, as it consists of a single flowchart, its subject testing showed a high level of success. The authors stated that this system was tested on 20 subjects and significantly reduced the success of social engineering attacks on everyday people (Mouton, 2017). However, the system did not significantly reduce false positives, meaning that it did not help people detect innocent requests as well as it helped to detect harmful requests.

A salient focus area of Social Engineering detection systems is automatic recognition of deception through verbal analysis. However, due to the incredibly complex nature of the human language, accomplishing this task proves to be very difficult. In 2010, Victor Raskin of Purdue

University released a study pertaining to a new application of Natural Language Processing (NLP) and Ontological Semantics (ONSE) in the field of Information Assurance. The goal of the Semantic Forensic NLP is to accurately model a human faculty, which refers to a subject making a judgement. This main goal can be divided into two parts. Firstly, the model would be able to detect if deception is occurring, and secondly, the model would be able to determine the truth of the nature of the interaction.

Raskin states that while both tasks are important, the focus lies in the latter, for it would give more insight into incoming attacks and their nature, and therefore provide information that could be used in the future to further prevent attacks. He gives an example for both cases on what a successful detection would look like. For the first case, success would simply entail detecting a lie, such as calling out a declaration of the day being Tuesday on a Saturday. In this case, the machine would know that the day is Saturday and not Tuesday. For the second case, success is detecting the truth of a situation by omitting falsehoods. This is an important case in deception because most social engineering attempting deception will avoid telling full-fledged lies and focus more on giving ambiguous scenarios. To accomplish these goals, it is important to first understand how people detect lies. We detect lies by comparing comprehended information to our existing knowledge of that subject. Therefore, any system that can process language should be backed with a database (lexicon) of truth's that can be compared to incoming information. Raskin calls this database a “Fact Repository” (FR) (Raskin, 2010). A FR can grow and change according to new input information that it is given. This ties us back to the use of “benchmark data” by Sandouka and Hoeschele, implying that a commonality among these new systems is the use of previously acquired user data.

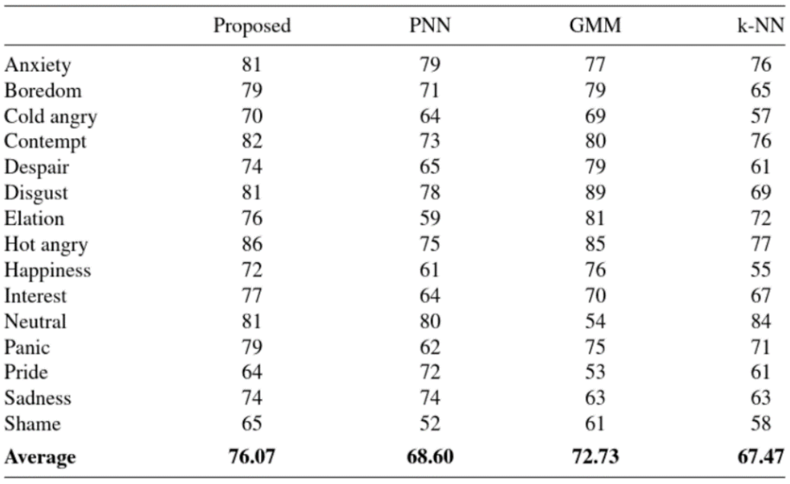
The main problem with all these recent social engineering detection models is that they lack evidence of providing any useful results concerning their effectiveness. This is perhaps due to a lack of implementation in industry, or due to a lack of having enough test trials. The system that holds the most promise for providing a good solution to our social engineering problem is the neural network system. The neural network system shows the most promise because according to theory, the use of a neural network will allow for automatic pattern recognition and therefore for streamlined emotion detection (since emotions are just different patterns of behavior). One issue that arises from using the neural network is that immense numbers of test data are required for the training of a neural network. The more training a neural network is given, the more successful it will be in detecting meaningful patterns. However, it is difficult to obtain data when it comes to the banking industry because operator-client interactions often contain personal client data such as bank account information. Obviously, this information will not be released to the public (Losev, 2020).

Given the dearth of training data, what other algorithmic approaches to social engineering detection are viable? One possible next step is to use a psycho-emotional state detection device, as suggested by Anton Losev of The Financial University in Moscow. Such a device would work by analyzing the speech of a client calling the bank, and in turn providing the bank operator on the other side of the call with important insight into the caller's emotional or mental state. Currently, this technology is purely theoretical, and would most likely need to be supplemented by existing science in the field of emotion recognition through speech, such as provided by Valery Petrushin. Another possible step would be to use a more advanced neural network that requires less training data. Such a system is described by Ling Cen from the Institute of InfoComm Research at Singapore.

In February of 2015, the current president of Illuminated Numerati, Valery A. Petrushin, published a patent in the U.S titled "Detecting emotion in voice signals in a call center". This patent lays out the technology of a computer monitoring system that can extract and analyze specific acoustic features from the speech of a customer/client during a phone call. In the analysis phase, the system passes the various extracted features of the call through algorithms and then compares the output with existing outputs in a database connected to the system. This database contains previously found outputs linked with specific emotional states, such as happy, sad, angry, nervous, neutral, and many more. The main purpose of this comparison, according to Petrushin, is to give call center operators real-time insight into the emotional state of their caller. This insight would then allow the operator to make a better judgement as to how to act towards the caller. Although Petrushin describes a few scenarios, one scenario that stood out was the operator vs. Fraudulent caller scenario. In this scenario, the operator would benefit by knowing that the caller is attempting to commit fraudulent activity because then the operator would obviously end the call or perhaps try to expose the fraudulent caller in order to avoid extortion. However, one might think that people themselves would be the best emotion detectors and therefore eliminate the need for AI detection of emotion. While this is the case for Petrushin's system, (Mean detection rate for people: 71.9% success rate, for system: 63.5% success rate), it just shows that there is room for improvement in the subject. One source of error that Petrushin noted is the fact that the emotion database he used in the system was created from a survey of only 43 people. Perhaps a larger survey of people or the use of a larger existing database would increase the accuracy. Also, a survey of more authentic sounding speech could increase the accuracy of results, such as analyzing people undergoing authentic emotions instead of analyzing people who just acted them out. Authentic emotion-influenced speech, especially nervousness-affected speech (which is a key emotion in perpetrators of fraud) is discussed by author Petri Laukka in his journal entry: In a Nervous Voice: Acoustic Analysis and Perception of Anxiety in Social Phobics' Speech. This topic is addressed in the later section, "Comprehending the issue of non-authentic emotionally-influenced speech".

Outside of the results of his system, Petrushin offered good insight into the nature of the emotions of perpetrators of fraud. One key emotion exhibited by perpetrators, according to Petrushin and others such as Eric van Dijk of Leiden University (Dijk, 2018), is nervousness, or stress. A nervous person will often have a slight wavering in his or her tone of voice, and to the system that wavering would be represented in the form of deviations in the fundamental frequency of subject's voice tone. Other vocal features that change under the influence of stress include speaking rate, volume, voice tremor, change in spacing between syllables, and a change in the fundamental pitch or frequency of the voice. Petrushin labels these features as conscious changes, meaning speakers actively use brainpower to change the way they sound (in the previously mentioned ways) under stress. However, when a certain point of stress is reached, other features in the voice begin to change subconsciously because of natural biological changes in the human's facial and bodily anatomy. One of these changes is a general decline in vocal quality, which is a direct result of natural tightening in the vocal cords. Although it may be difficult to hear this change audibly, it's clearly visible when looking closely at recordings of the voice. People can often experience this phenomenon when speaking for extended periods of time under false pretense or with uncertainty (such as a during a presentation with a big crowd). The methods of detecting these vocal features through the processing sound signals is discussed in the later section "Extraction of Vocal Features Through Sound Processing"

Also, in 2015, a method for detecting emotion through speech, titled *Maximum A Posteriori Based Fusion Method for Emotion Recognition,* was dissected in chapter 9 of Amit Konar's textbook on emotion recognition (Konar, 2015). The author of this chapter, Ling Cen, compares his "new" system to the architecture of existing speech emotion recognition systems, and discusses the applications of such systems in industry. Cen specifically points out that his and other systems can recognize fear, nervousness, and stress in a person's voice, and these emotions have commonly been linked to suspicious or abnormal behavior prevalent in criminals. The detection of fear, nervousness, and stress in a bank client (or any caller), could in turn be useful for the detection of lying or suspicious behavior coming from that client. The main proposition that Cen argues helps accomplish more accurate detection is the combination of various existing methods into one. He first provides accuracy levels of many different emotion recognition systems on their own. The following table provides a summary of the test results of various systems. According to the data, the proposed system performed the best.



	Proposed	PNN	GMM	k-NN
Anxiety	81	79	77	76
Boredom	79	71	79	65
Cold angry	70	64	69	57
Contempt	82	73	80	76
Despair	74	65	79	61
Disgust	81	78	89	69
Elation	76	59	81	72
Hot angry	86	75	85	77
Happiness	72	61	76	55
Interest	77	64	70	67
Neutral	81	80	54	84
Panic	79	62	75	71
Pride	64	72	53	61
Sadness	74	74	63	63
Shame	65	52	61	58
Average	**76.07**	**68.60**	**72.73**	**67.47**

Table 2 Recognition accuracies (%) of models in speaker-dependent training mode

It is also important to note that the proposed scheme performed the best in the detection of anxiety, panic, and neutral, which are all emotions important for the detection of Social Engineering activity. Along with providing evidence of the success of his scheme, the author also goes in depth into the mathematical and computer science of the scheme. Various headings in the chapter include: "Acoustic Feature Extraction for Emotion Recognition", "Proposed Map-Based Fusion Method", "Base Classifiers", "MAP-based Fusion", "Addressing Small Dataset Training Problem", "Training and Testing Procedure", "Training", "Testing", "Experiment", and "Conclusion" (Konar, 2015).

2.4 Current Companies That Detect Emotions in Callers

Cogito is one extant program capable of detecting emotions in both customers and employees, which is used by MetLife Insurance among other insurance agencies and major credit card companies in the United States. Cogito monitors a telephone call and notifies the call center employee of the speakers' moods (Simonite, 2018). There are two ways the program gives/generates its notifications: the first is a constant gauge of how well the call was going, green for a balanced conversation, yellow and orange when one side is too slow or too quick to respond, and red when the call need instant direct attention. The second employs with pop up messages. For example, when the empathy prompt is displayed that is the employee's cue to say something reassuring to make the caller feel more welcome (Bercovici, 2017). The primary focus Cogito listens to is the tone of the caller's voice and how it changes over the duration of the call; however, it also listens to the pace and pattern. Humans can notice changes in tone and pace but are less reliable than an automated system (Simonite, 2018). The roots of the company trace back to 1999, when Sandy Pentland proved the presence of non-verbal signals in communication between humans. Later, in 2007, Joshua Feast joined with Pentland to work on a Defense Advanced Research Projects Agency (DARPA) funded project to detect psychological states in PTSD patients. In 2012 they jointly created Cogito by generalizing the DARPA project for consumer application, focusing on health care companies (Cogito).

Another US company capable of detecting emotion through speech is Nemesysco, who specialize in voice analysis technology. They have developed and patented the "Layered Voice Analysis" technique, using a combination of vocal measurements, parameters and algorithms found in novel and older studies of voice analysis. The information derived from the conglomerated research put them on the forefront of the field. Nemesysco technologies are modular, with applications ranging from employee recruitment to call center customer service to insurance rate decision assistance. When combined with a polygraph Nemesysco can improve lie detection results drastically, above and beyond what a polygraph alone can provide. While it can provide offline analysis of prerecorded interviews, the system is also capable of real-time psycho-emotional analysis of speech (Nemesysco, 2016).

2.5: Examining the Issue of Inauthentic Emotions in Speech

Anxiety is not a simple emotion to detect. Unlike anger, happiness, or sadness, anxiety is a combination of various emotions experienced at the same time. The emotions that comprise anxiety include fear, uncertainty, distress, apprehension, and worry (Laukka, 2008). This means that capturing authentic anxiety is difficult because replicating multiple different emotions at once is nearly impossible. When people act out emotions, their biologically-emotional state plays an immense role in the emotion they attempt to portray, especially in their voice, and therefore they are hindered in their ability to portray inauthentic complicated emotions like the ones that comprise anxiety. Capturing authentic anxiety is important in fraud detection AI because fraudsters will most likely be in somewhat of an authentic anxious emotional state. Although Laukka does not capture the speech of fraudsters, she captures the speech of people with anxiety disorders, and since these people are naturally in a state of anxiety they can speak with a naturally anxious voice. This proves useful for fraud detection because there are not many existing sound files of fraudsters speaking in action. This study, done with the acoustic information from anxious subjects, provides the analysis of soundscapes of naturally anxious people (Laukka, 2008). A further distinction that the author notes is a difference among two types of anxiety. State Anxiety occurs when a person experiences a threatening situation, and trait anxiety occurs when a person has a lingering fear of dealing with stressful situations. This study focuses more on state anxiety and therefore is more useful as an application to fraud detection because fraudsters will typically experience state anxiety. Although it's very difficult to obtain a large database of speech utterances with real emotions, it's important to point out that this would be the best type of data to train an emotion detection model on.

2.6: Extracting Vocal Features Through Sound Processing

The human body can produce two different kinds of sounds through speech. The first sound is a result of vibrations in the vocal cords, ranging from 100 to 300 hertz. The second sound is result of vibrations occurring throughout the cavities of the head, resulting in much lower frequencies. These latter variations are called formant frequencies, and in a sound profile these frequencies appear as amplitude modulations, or waves. The first type of sound appears in the form of amplitude spikes with rapid decays. There is also a third form of vibrational frequency in the voice and it relates to the second type of sound in the voice, having to do with formant frequencies (Petrushin, 2015). To extract these frequencies from voice, Petrushin used specific recording and sound processing hardware, but currently, digital processing techniques are favorable, for digital sound software is easier to work and obtain. In the chapter Acoustic Feature Extraction for Emotion Recognition (Konar, 2015), chapter author Ling Cen describes the method he used for acoustic feature extraction. To prepare audio samples, Cen used three pre-processing techniques: Pre-emphasis, Framing and Windowing. These techniques were used on three widely known short-time cepstral features: Linear prediction-based cepstral coefficients, Perceptual linear prediction cepstral coefficients, and Mel-frequency cepstral coefficients. The third of these cepstral coefficients, MFCC, is a commonly referenced acoustic feature and is worth pursuing further (See later section: Mel-Frequency Cepstral Coefficients). Currently, python libraries, such as libROSA, provide functions that can extract these cepstral features, and other acoustic features from input .wav files. Another commonly cited vocal feature is the rate of pauses within speech. Pause rate is formally known as disfluency in the field, and this feature has been examined as a key factor of anxiety ridden speech by Petrushin (2015), Laukka (2008), and Alimuradov (2019). In his 2019 study of speech signal segmentation, Alan Alimuradov used two extractable acoustic features to examine disfluency in speech: Short-time energy and Zero-crossing rate. He also analyzed the statistical properties of background noise and one-dimensional Mahalanobis distance. This study did not provide a means for detecting the emotional state of a person through analyzing pauses, but it did provide a means for capturing the pauses within speech for further emotional state research.

According to multiple sources, such as Alimuradov (2019), Batliner (2013), and Cen (2015), Mel-frequency cepstral coefficients are among the most popular of acoustic features used in automatic speech recognition and emotion extraction from speech. The Mel frequency cestrum is defined as a representation of the linear cosine transformation function of a short-term log power spectrum from an acoustic signal. This scaling, which produces a Mel-Scale, transforms the frequencies of a spoken sound into frequencies that are more closely related to those heard by the human ear. Therefore, when the various frequency statistics are concentrated into coefficients, they can represent characteristics of the human voice that we can understand. MFCC's are concentrated coefficients from a speech signal that each represent a single acoustic feature within a signal. For example, the first coefficient, 0, represents the loudness, or energy of the signal at a specific point in time. This array of acoustic features can then be combined into a single mean value, for each frame of audio, and used as an emotion classifier. Other statistical variations of the values can also be taken, such as range, standard deviation, and median functions. This feature allows for accurate classification of emotions through speech because it carries a wide variety of information about emotions.

2.6.1: Noise Reduction

Noise reduction is a key element of audio preprocessing which is applied in many fields. In situations requiring automatic feature extraction it is of the utmost importance. Oftentimes, the algorithm operates in the frequency domain, rather than the time domain. To this end, a Fourier transformation may be employed to extract the pure tones' spectrum of a sound file. By comparing the spectrums of an ambient noise segment and a noisy file, it is possible to dampen the volume of specific frequencies present in the noisy audio segment.

Any Fourier-based noise reduction algorithm will create some artifacts as a byproduct of imperfect discrimination. In this application, discrimination refers to deciding whether an analog signal is above or below a decision threshold. This is central to the fields of digital data processing and information theory. In general, these artifacts are quieter than the original noise. On certain occasions, they may stand out more than they did in noisy file (Audacity, 2015).

2.6.2: Shimmer, Jitter, and Speaking Rate

Shimmer and jitter are acoustic features commonly used to measure perturbation in audio signals. Therefore, these features are useful in the analysis of normalcy in a speaker's voice. While jitter refers to the instability of frequency in voice, shimmer measures the instability of amplitude, with both being measured in percent. According to Linda M. Carrol, a senior voice scientist at The Children's Hospital of Philadelphia, the normal ranges of these two features lie between 0.5% and 0.7% (Carroll, 2011). The ranges themselves were not employed in model development, rather these numeric values were used as new variables for the random forest's classifier. Out of the two features, the team decided jitter was the most promising.

According to VirtualSpeech, speaking rate refers to the measure of how quickly someone speaks. This is easily obtained by dividing the time of speech by the number of words said per minute (Barnard, 2018). Speaking rate does not directly correlate with specific emotions and varies wildly between speakers. However, it is indicative of vocal energy. For example, if someone is very angry or very excited, they will have much more energy in their voice than if they were sad, bored, or neutral. More vocal and body energy produces a higher speaking rate. People who are nervous will talk faster to quickly flush out the information they are trying to convey, decreasing their time on the hot seat (Barnard, 2018). With these presumptions in mind, the team decided to analyze speaking rate between utterances under different emotions. To calculate the speaking rate, an automatic method was employed to divide the overall time of the speaking period by the number of silences it contains.

2.7: Current Social Engineering Detection Methods in Russia

The Association of Russian Banks is worried about the risks due to social engineering as well as other internet related crime. According to the SWIFT business forum, they are looking to the advancements that other banks are making in "data analysis and tracking users' behavioral patterns" as a possible solution to the social engineering problem (SWIFT, 2019). Currently, however, the banks have a simple, low-tech procedure for preventing socially engineered people from making withdrawals. When a client intending to make a transaction calls the bank or meets with a teller at the bank, it is the teller's responsibility to determine whether the transaction may be fraudulent. If this person detects fraud, they refer them to be evaluated by a psychologist who determines whether there is social engineering involved. Then the psychologist will help the person overcome the socially engineered influence and prevent the fraudulent transaction from taking place.

This system needs to be updated because it is both slow and often unsuccessful. Because the tellers need to be checking for fraud while doing the transaction, they are significantly slowed down which causes increased wait times. Services of expert psychologists tend to be costly to banks. Also, because the tellers are human, they are unable to have a high enough success rate at detecting and preventing the fraudulent transactions. Another possible problem with having a human detect fraud, is that the same social engineering tactics used on a client can be used on the tellers, leaving the bank vulnerable. Perhaps removing the fallible, human aspect from the analysis will increase speed and success.

Banks from all over the world are aiming to tackle the problem of fraud; many have already developed technological solutions. A common approach is to develop an Artificial Intelligence which can analyze a client's banking history to test if a pending transaction is typical. This data-based solution is much less direct than a psychology-based solution and, for the specific problem of social engineering, would need to include a psychological aspect to diffuse the situation. The problem with transaction history analysis is that, unlike Russia's current solution, often this system would generate a false alert for the bank. This could cause further delays and therefore would not be a good solution. Thus, flagging suspicious calls using artificial intelligence would be a better approach.

Russian Telemedical Companies are also being affected by social engineering attacks. These incursions are more direct than the typical methods used against the banks; often, the social engineer will call the company directly, pretending to be a client, to obtain prescription drugs. This is difficult for the telemedical companies to detect because over the phone it can be challenging to confirm someone's identity. Currently, many companies with phone security concerns use Personal Identification Numbers (PIN) to confirm the identity of a client. The PIN system is not perfectly reliable because there are ways for the social engineer to get this number from the client. AI could also be used to solve this problem using voice recognition to identify the client's vocal signature, and to confirm their identity. This security measure is growing in popularity and in reliability as technology improves. Relying on vocal signatures over human judgment, with a sophisticated enough system, could be more reliable than the PIN system.

A voice signature is very similar to a written one as it provides a unique and unobtrusive way to identify an individual. Voice signatures are becoming more common due to their higher accuracy in personal identification than regular written signatures. While traditional signatures can be forged or copied with modest training or technology, voice signatures are nearly impossible to forge as no two voices are the same. Voice signatures are convenient because they can be checked over the phone, even during a conversation. This also reduces the need for prior security steps during phone transactions or interactions with sensitive information. TechTarget, an information technology definition website, stated that national health service such as Medicare (ACA) and Medicaid offer clients an option to use voice signatures for added account security. The fact that these national healthcare institutions use voice signatures for client account security shows great promise in voice signature application for other organizations (TechTarget, 2017). The greatest problem with voice signature technology, however, is that it currently remains a trade secret amongst the organizations that use it and is therefore difficult to research or develop privately.

2.8: Introduction to Classification Algorithms

The algorithms discussed below aim to solve two distinct categories of problem: classification and regression. Setting aside terminological variance, the fundamental difference between the two is that the classification model produces a discrete output value representative of a class whereas the regression model returns a real number. Beyond this difference, the problems are very similar. Both have inputs (called "predictors" in a classification problem or "independent variables" in regression applications) and outputs ("labels" and "dependent variables" in kind). The inputs are features by which the data is broken down and sorted.

An exemplary regressive problem may involve predicting the odds of winning a chess match given the current state of a game. In this case the independent variables may include the number of pieces remaining on each side, or the number of safe places for the king to move. The dependent variable calculated could be the probability of one player's success which would be returned as a number between zero and one. If the players were less concerned with the odds and wished only to predict who was currently winning a classification algorithm could be used. This model could still use the same criteria as its predictors, but the return label would be a one or a zero (Harrison, 2019).

The K-Nearest-Neighbor (KNN) algorithm can be used to solve both classification and regression problems using proximity-based logic. The theory behind this model is simple to understand and implement. In short, when the query data is plotted alongside the training data the label of the query data is probably like the labels of the nearest point in the plot.

To use a KNN, the data must first be loaded. Afterwards the value of K is assigned. K represents the number of values near the query point which will be taken into consideration. It is important to choose a K value which suits the application. Lower values will improve precision but reduce stability whereas high values will have the opposite effect. In classifying problems an odd K value is utilized as it prevents ties between label results. Once the K value has been initiated, the function loops through all the example points in the training dataset. For each point the program calculates the distance from the current example to the query example and adds the index of the current example to a heap with the distance as the key. Once all points have been evaluated, K entries are popped from the top of the heap. If the KNN is being used to solve a regression problem, the program will then return the mean of the selected dependent variables. If it is a classification problem, it will return the mode of the selected labels (Harrison, 2019).

The KNN is a simple and versatile tool which is well suited for small tasks. In more complicated endeavors however the repetitive calculations take their toll on execution time making this system unideal for problems involving several inputs or large amounts of data.

2.9: Introduction to Random Forests

One of the most effective classification models in use today is the random forest. These systems consist of a plethora of unique decision trees which individually classify the input and vote to determine the output class. A decision tree is a classification architecture which sorts an input into its proper "class" based on a branching structure of decisions regarding the data "features". The strength of a random forest lies in the diversity of the trees of which it consists. By creating trees whose results have low correlation, the system becomes more resilient to individual errors.

To create a successful random forest two criteria must be met. The first is that all the features selected must have at least some predictive ability. Fulfilling this requirement can be accomplished using an already trained system to calculate the usefulness of various features. The second criteria requires that there be low correlation between the predictions of the trees. If the trees behaved similarly, there would be commonality between their errors and the system would become less stable.

Bootstrap agitation and feature randomness are two methods used to create diverse trees. Bootstrap agitation trains each tree on a random selection of the input data, possibly including repeated data. Feature randomness assigns each tree a random subset of features to use. While multiple trees may share some features, no two trees will be the same (You, 2019).

The generalization error for forests converges to a limit as the number of trees in the forest increases. Using a random selection of features for each split in the trees yields error rates favorable to Adaptive Boosting which are more robust with respect to noise (Freund and Schapire, 1996). Internal estimates monitor error, strength, and correlation - used to show the response to increasing the number of features used in the splitting - as well as variable importance.

When creating a random forest, each tree is characterized by a unique vector. To create the kth tree a random vector Θ_k is generated, independent of the past random vectors $\Theta_1, \ldots, \Theta_{k-1}$ but with the same distribution. The tree is then grown using the training data set and Θ_k, resulting in a classifier $h_{(x,\Theta k)}$ where x is an input vector. In random split selection Θ consists of several independent random integers between 1 and K. The nature and dimensionality of Θ depends on its use in tree construction.

2.10: Introduction to Neural Networks

The term neural network stems from the decision-making structure of this machine learning architecture. Discrete nodes interpret inputs differently, sending their results to others in a highly entangled web like the human brain. Neural networks are an example of deep machine learning, where the algorithm is teaching itself to solve the problem with limited operator input. While this family of AI algorithms is ideal for intricate applications, a common criticism is the lack of transparency in the decision-making process.

As the name suggests, neural networks are architecturally based on the human nervous system. To understand the functionality of a neural network necessitates a cursory knowledge of its biological analogue. Nervous systems are capable of highly efficient and complex decision making thanks to a very simple building block: the neuron. As shown below, neurons are comprised of three primary components. The dendrites function as inputs to the soma (cell body). Dendrites extend to other neurons or exterior sources of input and conduct electrical synapses to the soma. The cell body is responsible for determining the appropriate output which is then transmitted through the axon to other neurons or muscle tissue. As a unit the neuron functions as a complicated logic gate, processing on average the input from 6,000 other neurons to decide whether to send an electrical impulse to other cells through its axon. The electrical signals which carry information between axons and dendrites are called synapses. Notably, neurons at a synaptic junction never come into contact. Fluids called neurotransmitter substances conduct the synapses between neurons. The resistivity of the neurotransmitter substances at a junction determine the synaptic weight (the relative strength of the input to the receiving neuron) of the transmission. To simplify the operation of the soma, it functions much like a summer, which will trigger a synapse if a certain threshold voltage is met (da Silva et al, 2017).

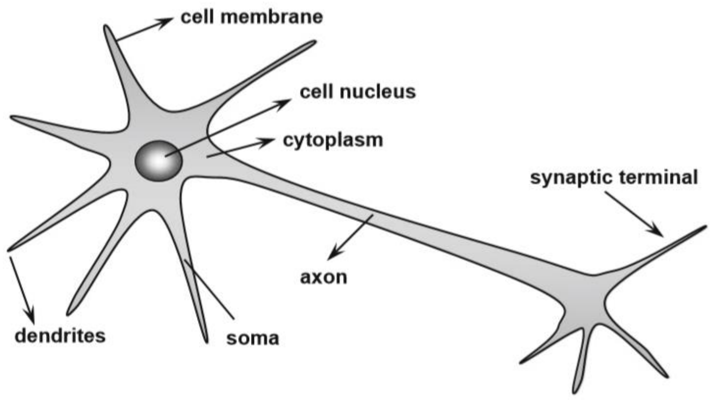


Figure 4 Structure of the organic neuron (Silva et al 2017)

Artificial neurons are designed to mimic evolution's solution for a learning network. As illustrated below, a typical implementation has four components designed to mimic their biological counterparts. The inputs (x1, x2, … xn) are first scaled by a synaptic weight (w1, w2 … wn). This is intended to affect the artificial synapses the same way variable neurotransmitter substances weight their biological counterpart. The summer block then measures the aggregate weighted input and subtracts the activation threshold (theta) to produce the activation potential (u). In simple applications, the activation function (g(u)) is a heavy-side step function, outputting a logical high if the activation potential is positive, indicating that the activation threshold has been met. In this configuration the artificial neuron functions identically to its natural analogue. Other common activation functions include the bipolar step function, symmetric ramp function, logistic function, hyperbolic tangent function, Gaussian function, and linear function. The output signal (y) is then sent to additional neurons or externally flagged if important.

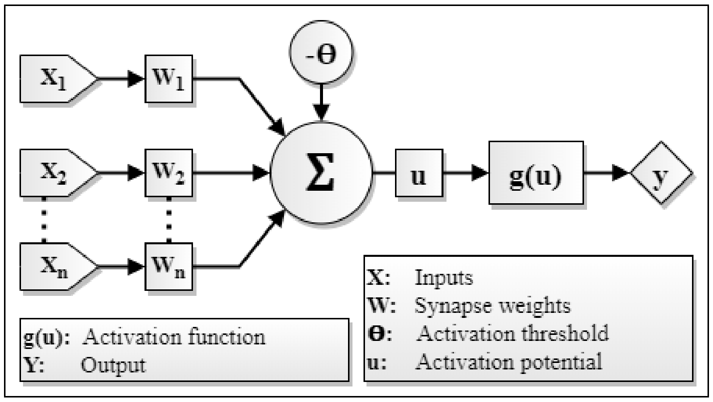


Figure 5 Structure of the artificial neuron

2.10.1: Strengths of a Neural Network

Many features set the neural network apart as a candidate for machine learning. Crucially for this project, there are extant training methods for initial construction of the network from datasets. The trained algorithm can then additionally adapt over time autonomously as it gains experience. This is most instituted with flexible synapse weights which can drastically affect neuron performance.

The outputs of a trained artificial neural network can vary based on the application. One invaluable characteristic is the system's ability to generalize solution criteria. In the intended application, this would allow the AI to identify novel phone conversations as atypical by recognizing general trends in the data. Organizing said data is another strength of this AI architecture. Once trained, the AI can identify multiple groups of interest in input data. Broad applications could separate typical and atypical conversations, while more advanced ones may be able to identify more specific subgroups in either case.

System integrity is an important factor to consider in any field. The multimodal structure of neural networks provides many unique benefits on that front. Learned information guiding neuron synapses is stored in several connected nodes which provides information security in the event some nodes are lost. The multitude of connections can also provide a more stable system. As more nodes are added the system becomes more accurate and fault resistant. In the event of partial nodal corruption, the remaining neurons are usually capable of overcoming the error and self-correcting over time.

Rapid prototyping was a necessity given the short development window available to the team. Fortunately, the neural network is comprised of simple components. When compared to other machine learning techniques the function of artificial neurons is much easier to prototype.

2.11: Audio Data Collection

The successful training and implementation of any learning algorithm is contingent upon the collection large amounts of quality data. There are three methods of gathering audio data of emotions: recording people speaking a sentence with different emotions, extracting audio from existing media such as movies, and using databases that have already been created.

The most consistent and reliable method of data acquisition is to record different people speaking a set sentence with emotion. While highly replicable, the results of this process are often contingent on the acting ability of those recorded. To circumvent this problem only trained or semi trained actors could be interviewed and recorded, however this limits the amount of data that can be obtained. Alternatively, the desired emotion could naturally be coaxed out of the speaker. For example, a subject could be asked to recite their sentences to a crowd to produce a nervous response. To get an angry response an interviewer could falsely accuse the person of an act. A sad or happy response could be induced if the speaker was first asked to read a sad or happy story.

The second method to gather audio data is to extract the audio from different forms of extant media such as movies, news reports, documentaries, where the speaker is exhibiting authentic emotion. There are two problems with this method: efficiency and legality. To effectively train an artificial intelligence thousands of data points are needed. In this case each data point is an audio file. The time needed to find and acquire the piece of media, watch it to pinpoint the sentence to extract, and finally to extract the sentence and mark what emotion it is spoken with is highly inefficient when there are other ways to acquire data. The legality of using published and possibly copyrighted material is also a concern. Despite these issues, the data reaped from this method is excellent for training. The mix of people speaking with authentic emotion and professionally trained actors speaking with emotion provide stellar results.

Finally, it is possible to use preexisting databases which gathered their data using one of the above methods. This approach is the easiest and one of the more effective. These sources often provide ready to use data which has been vetted for quality already. Additionally, many of these databases have been used in previous research on emotion detection systems.

2.12: Growing Concerns and Future Leads

Robocalls are becoming as much of a social engineering threat as in person calls. The term “robocall” refers to automatically generated voice calls. Although most people simply ignore robocalls due to their spam-like nature, sometimes robocalls can seem legitimate if they claim to be coming from a source familiar to the receiver. According to CNN Business, around 25 million Americans have been victims of fraud through robocalls, and around half of calls originating from unknown numbers are robocalls. This is a concerning statistic, only worsened by the fact that 90% of robocalls will claim to know the target. While most people can currently identify robocalls, recent technological advances have showed increasing capability in mimicking a natural human voice. Realistic sounding scam calls may soon be automatically customized for and used against large groups of people. This further indicates the need for systems capable of detecting fraudulent calls as any call could potentially be fraudulent (Wadhwa, 2018). Emotional analysis systems could provide insight into how future algorithms may identify human versus robotic callers.

Chapter 3: Methodology

3.1: Introduction to Methods

Within the development of a speech emotion recognition system lie four main objectives. The first to be addressed was the process of data collection, followed by feature selection, preprocessing methods, and AI architecture determination. Throughout the development process detailed below, all four of these sections evolved as new results proposed novel solutions and opportunities for research. For a system to function properly, all the aforementioned sections must be addressed.

As the training and validation of any machine learning algorithm is dependent on data, the first milestone set by the team was to acquire or create a sizable dataset. It was established that the best data for this application would be sound bites (specifically .WAV form) of subjects uttering short phrases exhibiting various emotions.

Once sufficient data had been collected to allow early testing, the team shifted to processing and extracting features. Preprocessing steps such as denoising the files was hypothesized to improve detection rates by reducing the impacts of variables deemed unimportant to the system. This was also important in reducing the variance between files sourced from different databases. As addressed in chapter 2, acoustic features are qualitative descriptors of sound, represented as numeric values. They are stored either in an array or as a single value for each frame inside the audio sample. Some common acoustic features include pitch, amplitude, and waveform (sine, saw, square, random etc.). To improve system performance, other acoustic features were extracted and studied. After extraction, the features that showed the highest amount of correlation between certain emotions and/or between other features were selected for model building.

From previous research detailed above, three AI architectures were selected for analysis. As new data was acquired and vocal features were altered, the analytical models were assessed for accuracy and efficiency. Initially the team hypothesized that the mesh neural network would provide the best results. The random forest and logistical regression models were also tested as detailed below.

3.2: Data Collection

The first problem facing the team was that of data collection. The acquisition of quality data is an integral part of training an artificial intelligence. More data can often yield better results; however, bad data can disproportionately impact the training process, producing poor results. Therefore, the goal of data collection is to gather a wealth of data while trying to avoid/purge bad data. Due to limitations on time and budget, the team decided that the most practical method would be to utilize existing databases. Table 3, below, contains the languages and sizes of the datasets sourced.

Name	Language	Number of Files
RAVDESS	English	1440
Emo-DB	German	535
SAVEE	English	480
CREMA-D	English	7442

Table 3 Databases sourced for training data

3.2.1: Databases Sourced

The first database to which the team gained access was the Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS). This database included audio, audiovisual and video exclusive files of 24 professionally trained actors speaking and singing with emotion. Each recording was rated by 10 individuals pulled at random from a group of 247. The ratings assessed the emotional accuracy of the recordings to filter out poor performances. The included 1,440 audio-only speech files were used in this project. The emotions represented in this database were neutral, calm, happy, sad, angry, fearful, disgust, and surprised (Livingston SR, 2018).

The Berlin Speech Database (Emo-DB) was incorporated next into the team's training data. This database contained speech from five female and five male actors simulating emotions. For each emotion, the actor or actress recorded five long sentences and five short sentences. In total, about 800 sentences were recorded, although only 535 sentences were kept due to quality issues. To test the quality of the data, 20 individuals listened to the audio files, categorized the emotion that was being portrayed, and rated the naturalness. All files with a recognition rate greater than 80% and naturalness greater than 60% were kept. The emotions represented in this database were neutral, angry, fearful, boredom, happy, sad, and disgust (Burkhardt, 2005).

The third database used in training was the Surrey Audio-Visual Expressed Emotion (SAVEE) database. This database included recordings of four male researchers from the University of Surrey. Researchers recorded 15 sentences exhibiting anger, disgust, fear, happiness, sadness, and surprise. Each also recorded 30 neutral sentences to serve as a baseline. This resulted in 120 recordings per speaker or 480 total recordings (Jackson, 2015).

The final databased sourced for this project was the Crowd-Sourced Emotional Multimodal Actors Dataset (CREMA-D). This database included recordings from 91 actors from various backgrounds. To improve quality, 2,443 crowd sourced raters labeled emotions and intensity values for each recording. Any data rated below the threshold was dropped leaving a total of 7,442 recordings. The emotions represented in this database were neutral, happy, sad, angry, fearful, and disgust (H. Cao).

3.2.2: Acoustic Feature Extraction and Processing

A large amount of information regarding a person's emotional state is conveyed through his or her voice. Although most adults naturally conceal their emotions, often presenting a neutral or happy visage, signs can leak through in the form of subconscious vocal cues. The vocal cords of a nervous speaker, for instance, will naturally tighten, resulting in a higher pitched voice. These cues may be difficult for a listener to detect; a properly trained algorithm, however, may be able to use them to interpret emotions. From previous literature, it was hypothesized that a combination of these various acoustic features could be analyzed to determine the emotional state of a speaker.

The first step towards determining which features were to be included in the final model was to extract all features of interest from the data files for testing. To accomplish this, the team used the python sound and music processing library libROSA to extract a variety of options. The table below shows a list of features which were extracted using libROSA and used for research.

Feature	Description
chroma_stft	Chromatogram from a waveform
melspectrogram	Mel-scale spectrogram
mfcc	Mel-frequency cepstral coefficients
rms	Root mean square of the energy inside each frame
spectral_rolloff	Roll off frequency
spectral_centroid	Centroid frequency
zero_crossing_rate	Zero frequency occurrences (time-domain)

Table 4 Audio features extracted for characterization

Each of the features in table 4 provided a set of numeric values which were then synthesized into columns inside of a Comma Separated Variable (CSV) data frame. Although some of these features had been utilized before as evidenced by their frequent appearances in literature on the topic, some had not been. The team hypothesized that additional features not previously researched could provide insight into the speaker's emotional state.

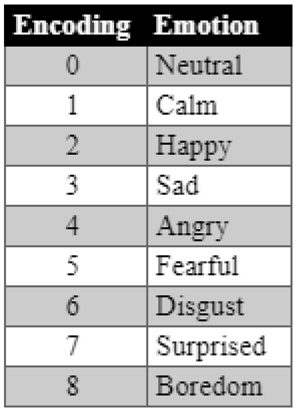
3.2.3: Data Processing

Once the features had been extracted, the data was prepared for use by the three models. To accelerate the training process, the data was stored in CSV files using the Panda Python3 library. Using simple file types such as CSV allowed the artificial intelligence to read the data more easily and reduced memory required, enabling faster collaboration. The team then experimented with using different functions to scale the data. Specifically, the normalizing, inverse, logarithm base two, and the natural logarithm functions were implemented. These were used to change the range of values produced by each feature. It was hypothesized that when data was of the same order of magnitude, the artificial intelligence will treat each feature more equally than if the data was of different orders of magnitude.

The normalizing function uses the formula $(1 - x_{min})/(x_{max} - x_{min})$ to scale the values between zero and one. The inverse function was implemented as $1/(x + 1)$. The denominator of $(x + 1)$ prevented division by zero, as some input values were expected to fall within $-1 < x \leq 0$. The team used two logarithmic functions: $\log_2(|x| + 1)$ and $\ln(|x| + 1)$. The absolute value of the input was taken, and one was added to remain within the domain of the log function.

3.3: Dataset Creation

The team created eleven different data sets to test the effects of data volume, feature selection, processing techniques, and model design on performance. The first dataset that was created using only the RAVDESS database and six features. At this stage, the team was testing chroma standard deviation, chroma mean, MEL standard deviation, MEL mean, MFCC standard deviation and MFCC mean. The second dataset incorporated the remaining three databases to test the effects of imbalanced class representation and data volume on model accuracy. The third data set added seven features, including the spectral centroid, MFCC first derivative, MFCC second derivative, root mean square, spectral rolloff mean, spectral rolloff range and zero crossing rate.



Encoding	Emotion
0	Neutral
1	Calm
2	Happy
3	Sad
4	Angry
5	Fearful
6	Disgust
7	Surprised
8	Boredom

Table 5 Emotion codes

The next three data sets retained the established features and databases, but employed three different scaling techniques: normalization, logarithm base two, and inverse. The following four datasets included two new features: speaking rate and jitters. Sets eight through ten employed three scaling techniques: logarithm base two, natural logarithm, and inverse. Dataset seven was used as a baseline for unscaled accuracy. Normalization was omitted from this round of testing as it had already proven ineffective. The final dataset created included ten features. Features exhibiting high correlation were omitted to reduce redundancy. Five features were removed, including the MEL standard deviation, root mean square, spectral rolloff range, zero crossing rate and speaking rate.

3.4: Feature Data Analysis

To create a consistent test environment, each dataset was tested on the same three classification models. The team developed random forest, linear regression, and multi-layered perceptron (mesh neural network) models. These were chosen for their simplicity and prevalence in previous literature. It was hypothesized that simpler algorithms would outperform more sophisticated ones when presented with a simple dataset. The random forest model was created with 100 trees. Each tree was grown with a maximum node depth of 31 to prevent latency in testing. The linear regression model was designed to emulate a multinomial regression as this architecture can differentiate between several classes. The multi-layered perceptron neural network model was created with an alpha value of 0.01, ten hidden layers of 50 neurons each and an adaptive learning rate. The code for the three models can be found at appendix C.

To streamline the models' handling of data, each emotion was assigned an integer. The emotion codes are shown in table 5. Next, the team created graphs to study correlations between features and emotions These included correlation matrices, distributions of features, and t-Distributed Stochastic Neighbor Embedding (TSNE) plots. A correlation matrix shows the correlation between the different features. The distribution of features graph is a set of histograms, one for each feature, that shows the value of each feature according to the emotion. A TSNE plot displays high dimensional data on a 2-dimensional plot to enhance pattern visualization.

Chapter 4: Findings

4.1: Introduction to Findings

The results and recommendations produced by the methods detailed above fall into four categories: those regarding the databases used throughout the development, the audio features evaluated, the impacts of scaling data, and the efficacy of different AI models. These results were derived from four tests as follows. The first three tests included testing for feature correlation, feature value distribution between certain emotions over sample datasets, and TSNE plotting. The fourth test, for accuracy of emotion detection using different classification models, was the main goal of the report. Therefore, the subsequent tests were designed to determine the best inputs for the team's model.

4.2: Databases

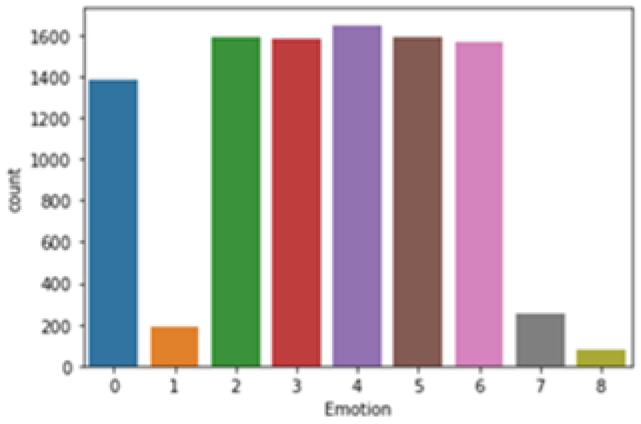
Machine learning is reliant on data for training and validation. As the project progressed the team expanded the training data set using files acquired from four online databases. To test the effect additional data had on the accuracy of the models the team created two datasets, including the same features, to isolate the effect of the data. The first dataset included only the RAVDESS database and the second dataset included all four of the databases. As shown in table 10 below, the accuracy of the random forest model increased by 4.7%, while that of the linear regression model decreased by 5.5%. The multi-layer perceptron model also saw gains in accuracy of 12.31%. On average, the increased dataset size resulted in an accuracy increase of 3.84% which validated the hypothesis that adding data would improve performance.

To further improve accuracy, additional databases could be used to further increase the supply of training data available to the AIs. In acquiring the four databases used in these experiments, the team also discovered other promising options. Four of note to future studies are the Elderly Emotion Speech Database (EESDB), the Toronto Emotional Speech Set (TESS), the SEMAINE database, and the Danish Emotional Speech Database (DES). These databases are already set up for emotion extraction and have been used in multiple other research studies on detecting emotion.

The Voxceleb and IEMOCAP databases also contain a wealth of emotional speech data but were deemed unusable for the purposes of this project. The Voxceleb database was promising with over one million utterances; however, they are not paired with emotions. For this reason, only an unsupervised learning method could be used with this dataset. The IEMOCAP database does provide emotional tags for its data. In this database, psychological tools were used to elicit real emotions from the test subjects being recorded. Each sound file featured multiple speakers and multiple emotions, unlike the other databases which included one speaker exhibiting one emotion per file. Unfortunately, extracting single speaker utterances would have required a time-prohibitive level of preprocessing work. For this reason, the team decided to omit this database.

4.2.1: Regarding Emotion Selection

Figure seven below contains the emotional makeup of the current training dataset. The imbalance between common and underrepresented classes results from differences between emotions included in the source database. The team believes the relatively low accuracy of the classification models may be partially attributed to this discrepancy in representation.



Encoding	Emotion
0	Neutral
1	Calm
2	Happy
3	Sad
4	Angry
5	Fearful
6	Disgust
7	Surprised
8	Boredom

Figure 7 Emotion Distribution of the second dataset

Currently, the models detect more emotions than necessary for this application. They are designed to classify data into nine different categories: neutral, calm, happy, sad, angry, fearful, disgust, surprise, and boredom. Due to variance in emotions included in the available databases, some common classes, such as happy, angry, and sad were overrepresented in the training datasets. This discrepancy in representation is perhaps the greatest reason for a lower accuracy of detection within the random forest's classification model (43.82%). The team has hypothesized that to increase accuracy, each output classification of the model must be evenly represented in the training data.

The total number of outputs could also be decreased. Distinguishing between similar emotions such as calm and boredom is difficult and unnecessary for a social engineering application. The inclusion of such minor and functionally unimportant distinctions between classes could greatly reduce the model's performance. Furthermore, the model would be more likely to select the correct class if it had fewer outputs to choose from. Since the sole purpose of this speech emotion recognition architecture is to detect socially engineered patrons, limiting the scope to only detect angry, fearful, happy, and calm may be an acceptable concession in exchange for superior accuracy. These distinctions would theoretically separate out likely cases of social engineering based on the team's literary review of common social engineering signs. A calm, happy, or bored client is likely unimpaired, while an angry, fearful, or distressed patron may not be in a suitable state of mind to make a transaction. It may even be possible to further reduce the output goals of classification into "check - good", or "uncheck-bad" classification.

4.3: Acoustic Feature Findings

4.3.1: Test 1 – Identifying Feature Correlation

A series of heat maps and pair-plots were created to evaluate the correlation between acoustic features. The aim was to identify features with high correlation, as they could either be combined, or one of the features from the pair could be removed. Having highly correlated features as inputs in a classification model has been shown to decrease the accuracy of the model. Below are the heatmaps and pair-plots used for datasets containing six features, thirteen features, and fifteen features.

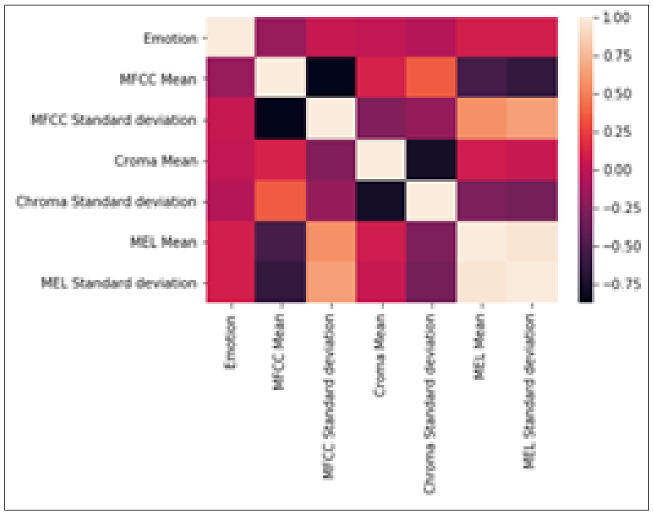


Figure 6 Heatmap of the initial six features used

In this heatmap (figure 8), lighter colors represent higher correlation while darker colors represent lower correlation. Most of the feature pairs had a darker square, meaning that most of the features were separate and uncorrelated. However, the MEL mean & the MEL standard deviation pair had a very light square (the lightest of all), meaning that those two features have a high correlation and were therefore combined into one feature. Some other lighter squares include the pair of MFCC standard deviation and the MEL standard deviation, the pair of MEL mean & MFCC standard deviation, and the pair of MFCC mean & Chroma standard deviation. Although there was limited correlation between these, they were left as separate features because the correlation was not high enough. Below (table 6) is a summary table of high correlation features displaying which features were removed due to high correlation:

Features	MEL mean	MEL standard deviation
MEL mean	Same feature	Highly correlated
MEL standard deviation	Highly correlated	Same feature

Table 6 Correlation of 6 features

These results indicate a potential redundancy between the MEL mean feature and the MEL standard deviation feature. For this reason, removing one may improve performance. This is especially true for the random forest model which is dependent on uncorrelated input features.

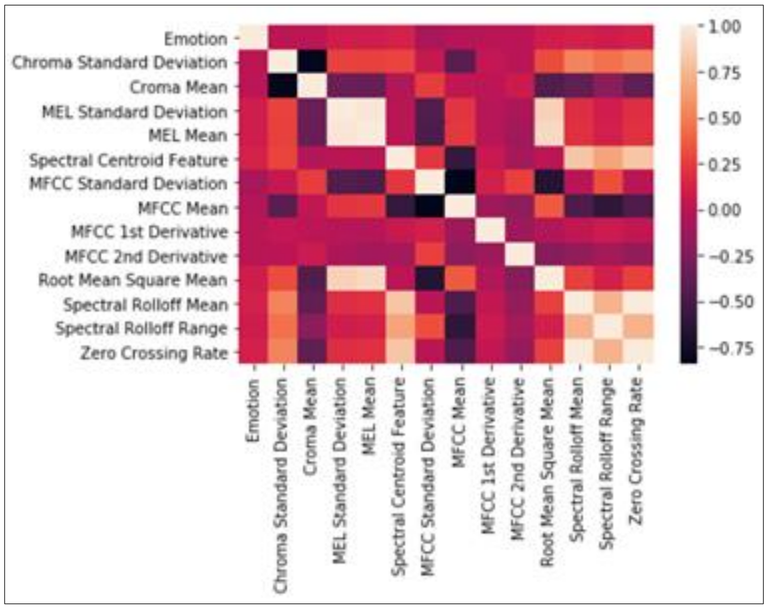


Figure 7 Heatmap of the thirteen features used in the second phase

Figure 9 above shows the previous six features, along with seven new features extracted, starting with the Spectral Centroid (in the italics). According to this heatmap, the following pairs showed high correlation: MEL standard deviation & Root Mean Squares mean, MEL mean & Root Mean Squares mean, Spectral Rolloff Mean & Spectral Centroid, Spectral Rolloff range & Spectral Centroid, Zero Crossing Rate & Spectral Centroid, Zero Crossing Rate & Spectral Rolloff mean, Zero Crossing Rate & Spectral Rolloff range, Spectral Rolloff range & Spectral Rolloff mean. Table 7, below, is a summary table of high correlation features displaying which features were removed due to high correlation:

Feature	MEL mean	MEL std	RMSM	SRM	ZCR	SC
MEL mean	Same feature	Highly cor.	Highly cor.	-	-	-
MEL std	Highly cor.	Same feature	Highly cor.	-	-	-
RMSM	Highly cor.	Highly cor.	Same feature	-	-	-
SRM	-	-	-	Same feature	Highly cor.	Highly cor.
ZCR	-	-	-	Highly cor.	Same feature	Highly cor.
SC	-	-	-	Highly cor.	Highly cor.	Same feature

Table 7 Correlation of 13 features

These results indicate a potential redundancy between the MEL mean feature, the MEL standard deviation feature, and the Root Mean Squares. To reduce input correlation, two of the three may be removed. It is also safe to remove two of the following three: Spectral Rolloff mean, Zero Crossing Rate, or Spectral Centroid.

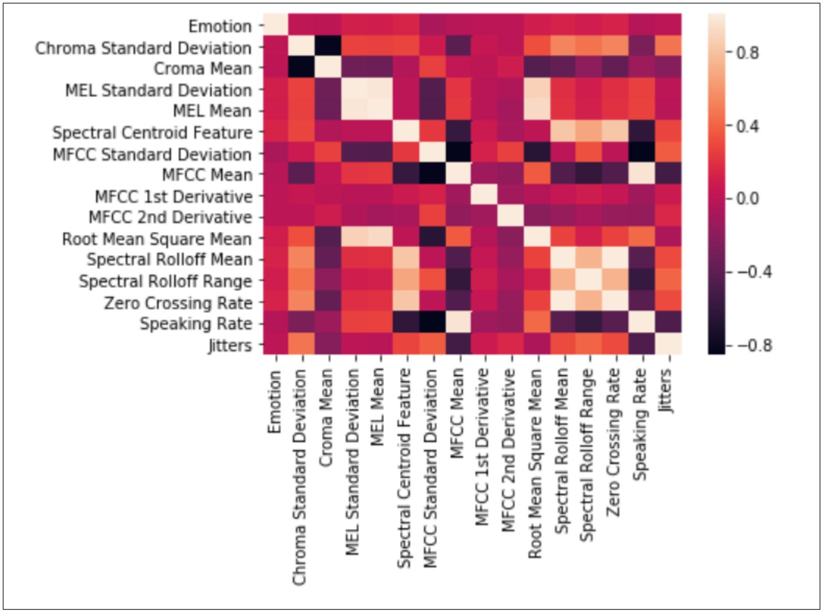


Figure 8 Heatmap of the fifteen features used in the second phase

The Jitters feature had no correlation with the other features, so it was safe to keep. However, speaking rate had a very high correlation with MFCC mean. This suggests that it is best to remove either the MFCC mean or Speaking Rate.

The pair-plot below (figure 11) depicts all fifteen features showing scatterplots of correlation. Column feature is x-axis, row feature is y-axis, each point is a different sample. The following table (table 8) shows the numbers and letters with their corresponding features.

Feature	Row	Column
Chroma Standard Deviation	1	A
Chroma Mean	2	B
MEL Standard Deviation	3	C
MEL Mean	4	D
Spectral Centroid	5	E
MFCC Standard Deviation	6	F
MFCC Mean	7	G
MFCC 1st Derivative	8	H
MFCC 2nd Derivative	9	I
Root Mean Square Mean	10	J
Spectral Rolloff Mean	11	K
Spectral Rolloff Range	12	L
Zero Crossing Rate	13	M
Speaking Rate	14	N
Jitters	15	O

Table 8 Pair-plot key

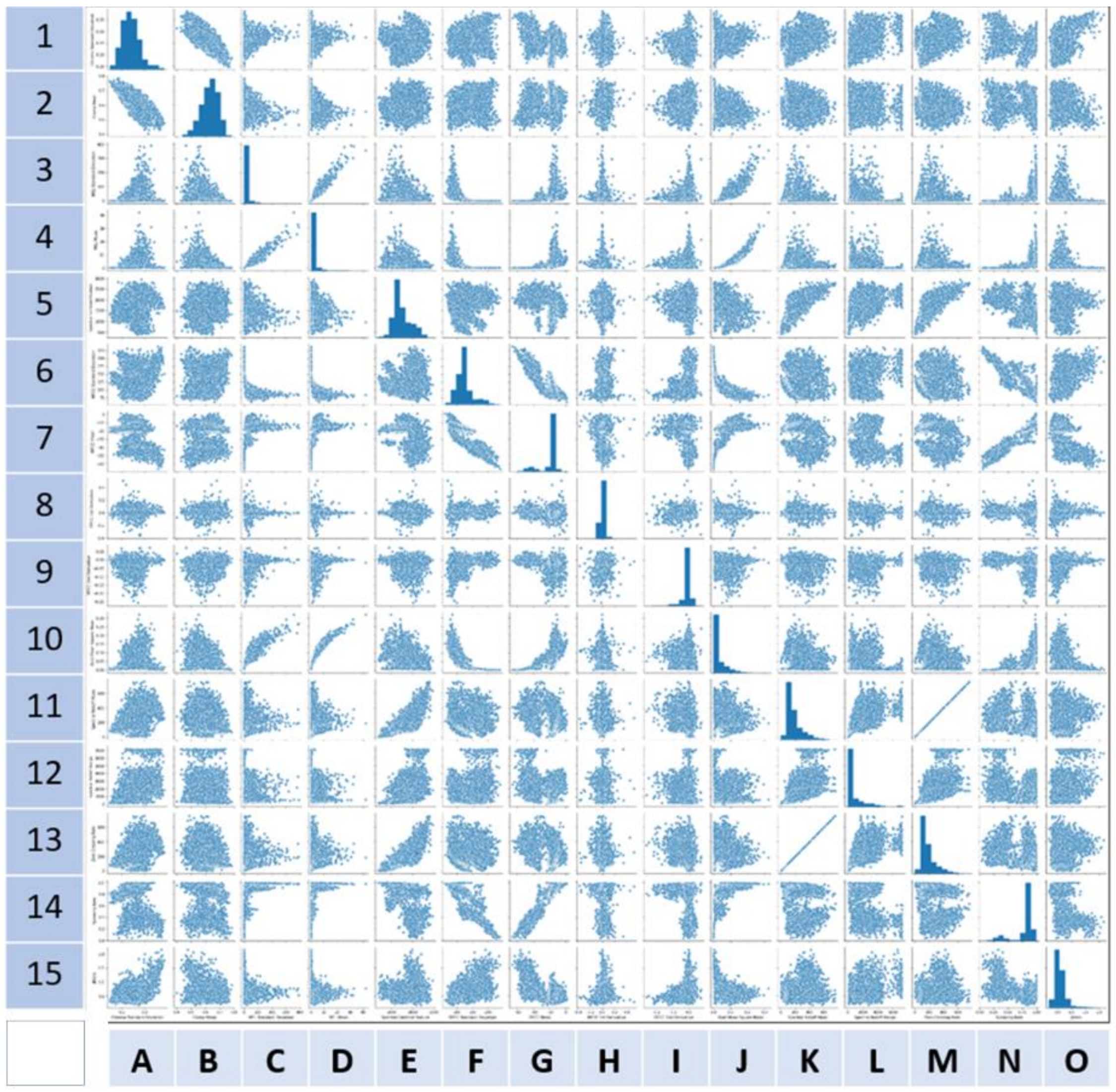


Figure 9 Pair-plot of all fifteen features showing scatterplots of correlation

Some of the plots closely resembled a linear regression. These plots (similar to what was observed in the heat maps), suggested that the two features being compared were too similar and could therefore either be synthesized, or paired down. Exemplary plot pairs include 13K & 11M; 3D & 4C; and 6G & 7F. Another observation was that some contain one variable (feature) which remained stagnant over a certain value while the other variable fluctuated heavily over its axis. This meant that the ranges of the two variables being compared were vastly different. While one range may have fallen between 0 and 1, the other range could have fallen between 200 and 600. These occurrences explained why one of the features could appear stagnant while the other fluctuated, given the scale of the x-axis and y-axis remained constant. Examples of these plots included 8(A-O), H (1-15), and 9(A-O).

4.3.2: Test 2 – Emotional Feature Distribution

The second test examined the distribution of feature values for different emotion classes. Each color in the histogram plots (figures 12-19) below represents an emotion. Emotion encoding follows the same pattern outlined in the Methodology chapter. For reference, see table 9.

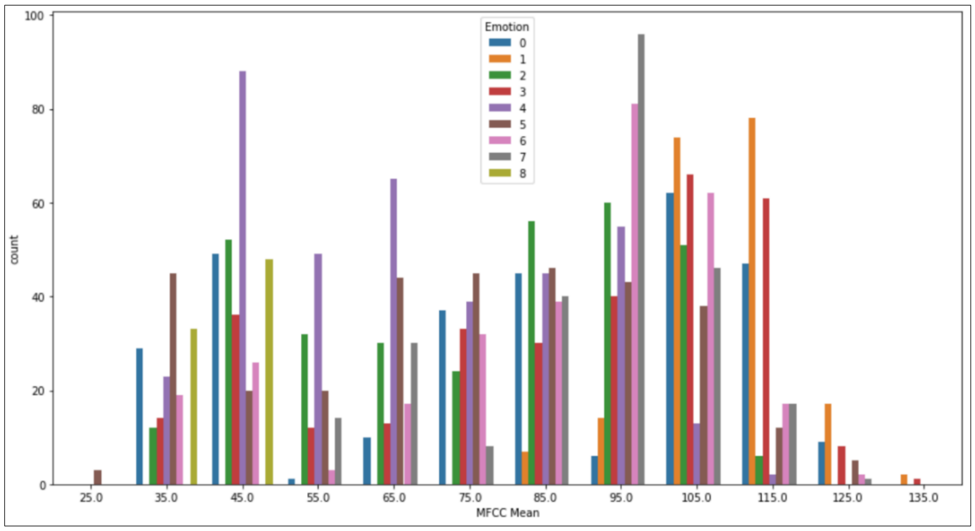


Figure 10 RAVDES MFCC mean

Encoding	Emotion
0	Neutral
1	Calm
2	Happy
3	Sad
4	Angry
5	Fearful
6	Disgust
7	Surprised
8	Boredom

Table 9 Emotion Codes

For the MFCC mean graph (above) the team had hypothesized a normal distribution of the different emotions. The discovery that each emotion contained multiple peaks made classification more challenging.

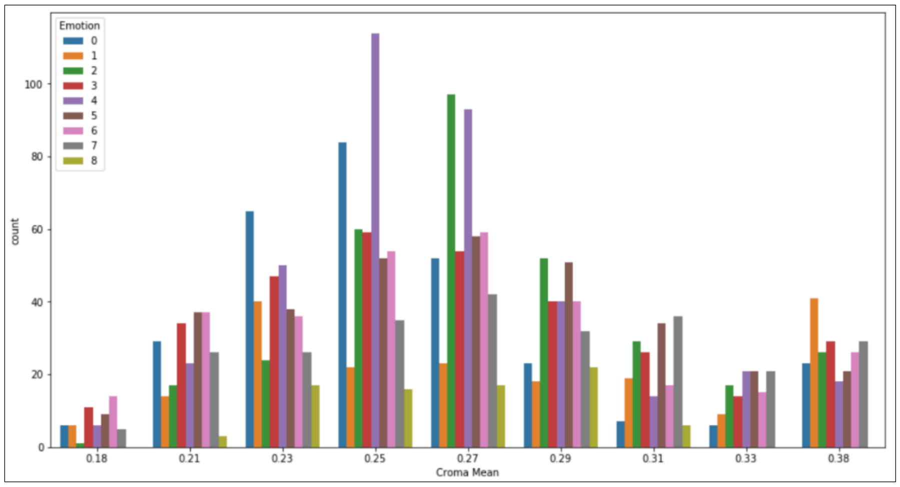


Figure 11 RAVDESS Chroma mean

The Chroma mean graph (above) follows a normal distribution with a slight deviation at x=0.38. It is believed that this deviance could be resolved with additional training data. The MFCC mean (figure 14), Chroma standard deviation (figure 15), and MFCC 1st Derivative (figure 16) graphs all followed a normal distribution with minimal deviation. These features allowed for easier classification and were especially helpful for the random forest model.

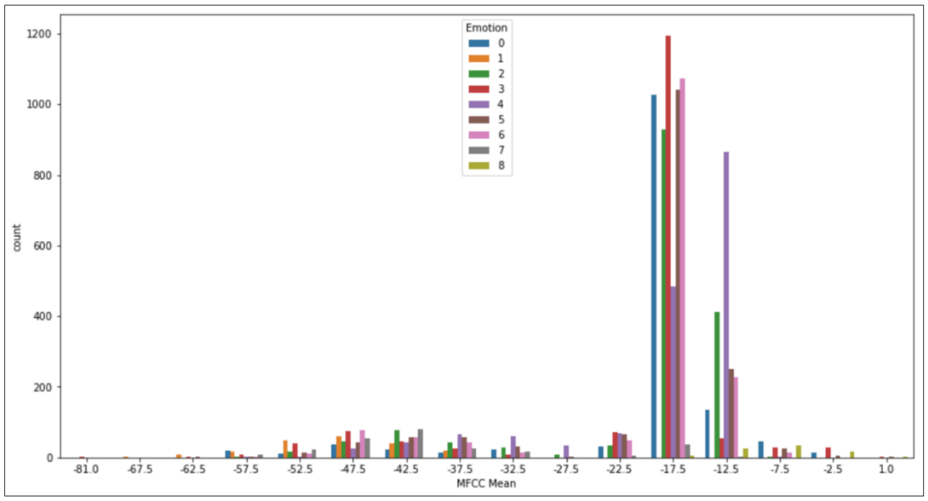


Figure 12 Full dataset MFCC mean

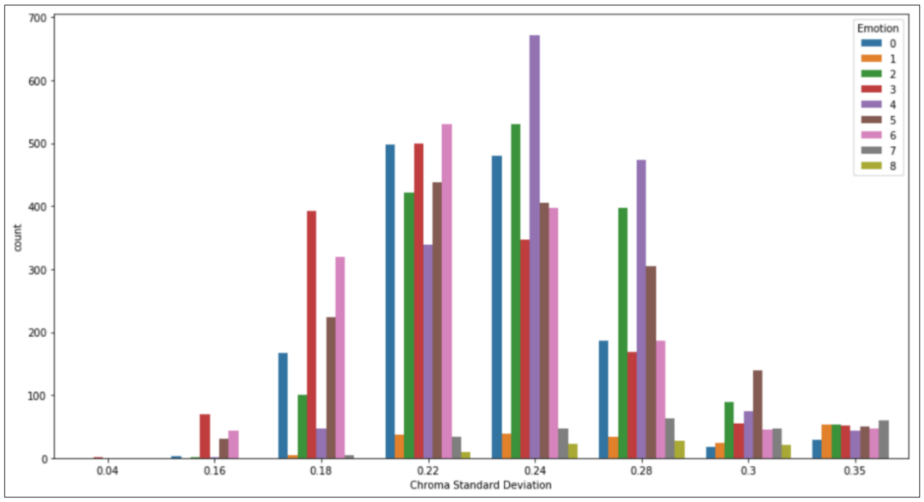


Figure 13 Full dataset Chroma standard deviation

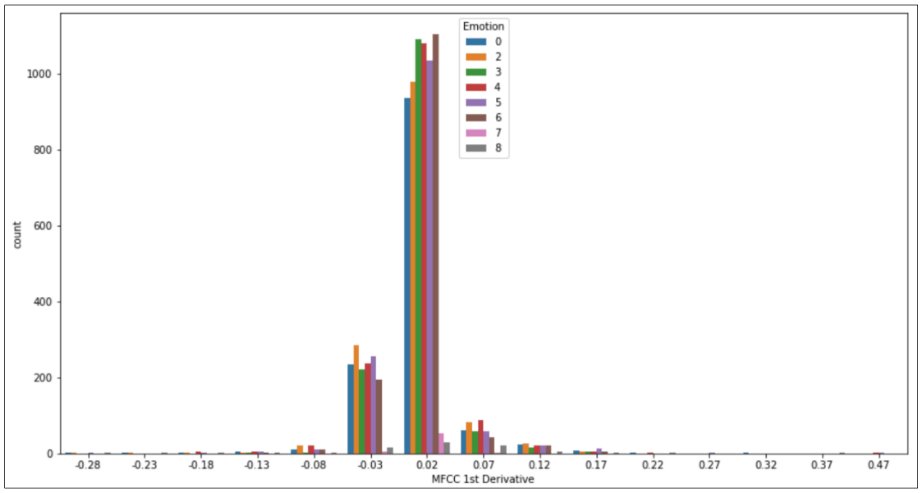


Figure 14 Full dataset MFCC 1st derivative

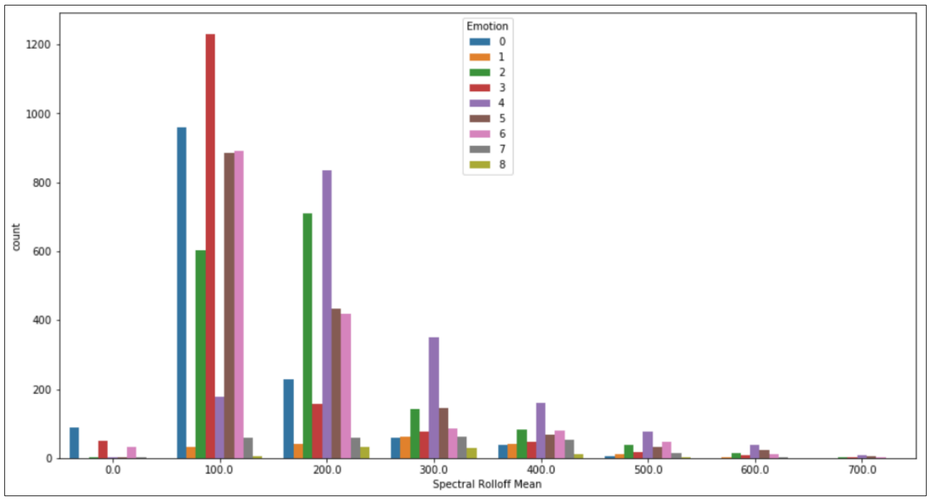


Figure 15 Full dataset spectral rolloff mean

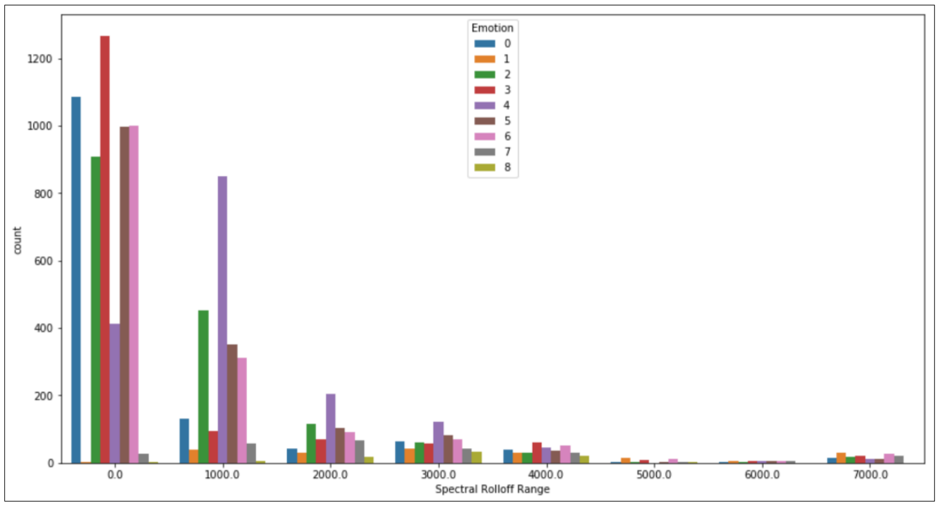


Figure 16 Full dataset spectral rolloff range

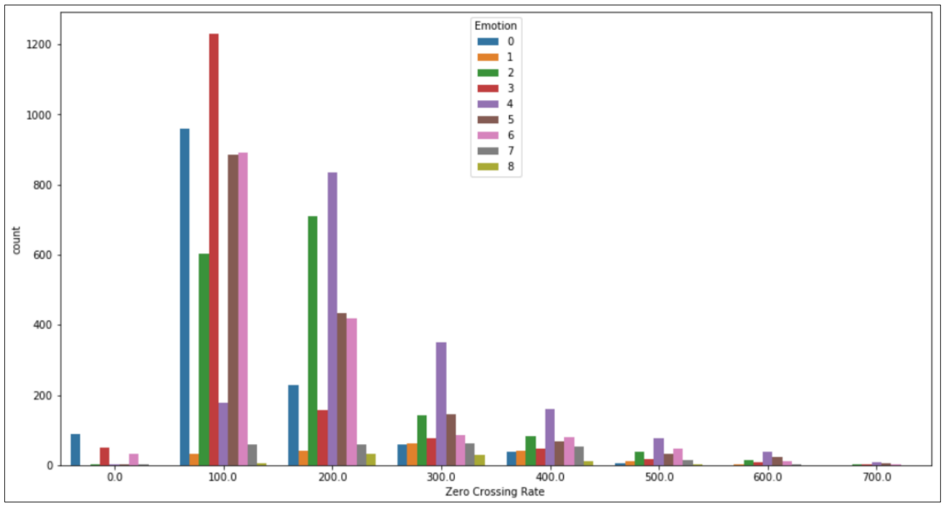


Figure 17 Full dataset zero crossing rate

The spectral rolloff mean (figure 17), spectral rolloff range (figure 18), and zero crossing rate (figure 19) all followed a lognormal distribution. This was not as useful to the random forest model, but it worked well for the multi-layer perceptron model.

4.3.3: Discussion of Features and Emotions

As briefly explained above, the feature selection process can be further improved. Some features, such as MFCC mean and Speaking Rate, followed a linear regression when plotted against one another. It is probable that having both as inputs in the AI classification model could confuse it. By removing redundant features such as these, the random forest model especially, could benefit. The final two experiments run with these models explored the impact of removing five possibly redundant variables. The logistical regression and the multi-layer perceptron models were less accurate with the limited feature set; however, the random forest had an increase of 0.35%. It is probable that if a scaling function was applied to this dataset, the random forest would outperform the current most accurate model.

In addition to removing redundant features, the system could potentially benefit from the addition of novel ones. If the features exhibit low correlation to each other and have some predictive ability, the models will benefit. The team investigated the fundamental frequency feature as a possible input, but had trouble coming up with an algorithm to extract it correctly. Future research could pursue this further.

4.4: Scaling

The team used four different methods of scaling features to increase the accuracy of the artificial intelligence models. First, the data was normalized between zero and one. This method performed very poorly, with all three models decreasing in accuracy and the random forest model suffering a 50% drop in performance. Next, the data was inverted. This method yielded small increases for the logistical regression and the multi-layer perceptron models, but a negligible increase for the random forest model. The third method was putting the data through the logarithm base 2 function. This method had an increase for both the logistical regression model and the multi-layer perceptron model, but the random forest model stayed constant with only a negligible increase. The last model was putting the data through the natural logarithm function. This method saw negligible differences from the logarithm base two method.

In the future, the feature manipulation model can still be greatly improved. The scales (ranges) feature values differ greatly. For example, while the MFCC 1st derivatives lie between 0 to 2, the Spectral Centroids often lie between 1000 and 4000. Mapping these values to same scale could potentially improve system accuracy. The random forest especially stands to benefit as it relies on evenly scaled inputs for proper output placement.

Along with proper scaling, other pre-processing methods such as filtering should be further investigated. Filtering is important in audio processing because it can be used to remove unwanted frequencies or sounds from audio files. Manual denoising of audio files is time intensive and therefore functionally impossible for large scale datasets. Automatic filtering is both commonly used and highly promising for this application. The team has hypothesized that performance could be greatly improved by implementing such a program.

4.5: AI Models

The team used each dataset to train three artificial intelligence models: a random forest, a linear regression algorithm, and a multi-layer perceptron neural network. Throughout all but the normalized datasets, the random forest model achieved the highest accuracy while the logistical regression model had the lowest (see table 10). The multi-layer perceptron fell in between but significantly outperformed the linear regression model. The failure of the logistical regression model is attributable to the data provided. For a logistical regression model to work with high accuracy, the data must fall into groups. As evidenced by the figures below, there was little grouping present. When just the RAVDESS database was used, there was little grouping compared to when all the databases and the set of 15 features were used. In contrast to logistical regression, random forest and neural networks such as a multi-layer perceptron do not need data that falls into distinct groups to produce highly accurate results.

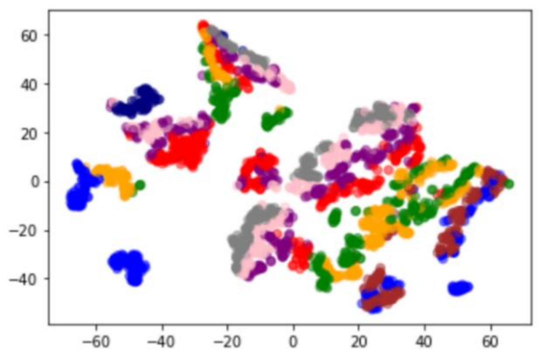


Figure 18 TSNE of dataset 1

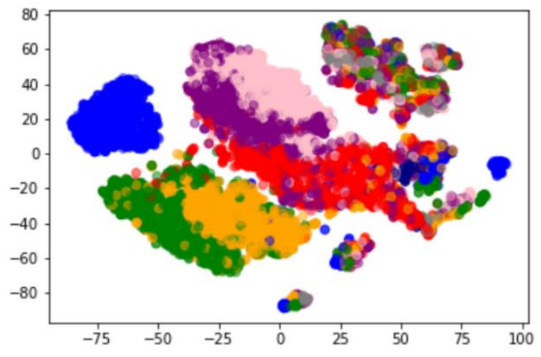


Figure 22 TSNE of dataset 2 (13 features)

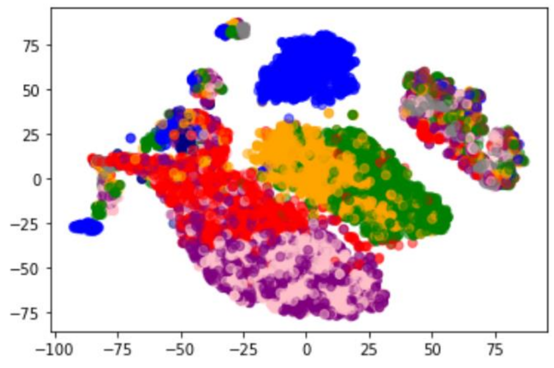


Figure 23 TSNE of dataset 3 (15 features)

Dataset	RF	LR	MLP
RAVDESS with 6 features	35.24%	28.64%	25.1%
All databases with 6 features	39.94%	23.38%	37.41%
All databases with 13 features	42.72%	31.86%	36.79%
13 features normalized	22.24%	24.45%	32.59%
13 features log base 2	42.96%	34.76%	40.86%
13 features inverse	42.90%	32.61%	38.66%
All databases with 15 features	42.76%	31.52%	37.73%
15 features log base 2	43.82%	34.31%	39.57%
15 features inverse	43.45%	33.82%	38.15%
15 features log base e	43.20%	35.07%	39.71%
10 features	43.11%	29.49%	36.87%

Table 10 Model accuracy by dataset

The team created and experimented with three relatively simple artificial intelligence models, documented below in the appendices. In the future, more complex systems may be required to implement some of the recommended next steps including increases to data volume and feature count. Three promising models which could be useful are the mesh neural network, recurrent neural network, and the support vector machine. These recommendations are based on the data above in table 10. As the amount of data and acoustic features increased, the multi-layer perceptron approached the accuracy of the random forest. The team hypothesized that a more complicated neural network architecture may rival, or even surpass, the performance of a random forest when presented with sufficient data and features.

Chapter 5: Conclusions

The purpose of this project was to research the reasons and methods behind the creation of an automatic speech emotion recognition system and to demonstrate these steps by creating a functional prototype. First, the team found public benchmark data of emotion utterances. The useful acoustic features were then extracted from these benchmark datasets and converted into usable files. The quality of each acoustic feature was then evaluated experimentally. Finally, the best acoustic features were used as input variables in three different artificial intelligence classification models. The highest detection accuracy achieved was using the log base two transformation of all fifteen features, extracted from all the benchmark datasets, and plugged into the random forest. This produced a 43.82% detection rate. While this rate may be lower than the team had initially hoped to achieve, it represents great progress towards a solution. The low detection rate could be attributed to several factors, likely including the input features and emotion classes used, and the feature processing employed.

While evaluating the results of the project, the team derived a set of recommendations for future research. Specifically, next steps are suggested regarding databases, acoustic features, and artificial intelligence models. The team believes that system performance could be improved by incorporating the EESDB, TESS, SEMAINE, and DES databases into the training dataset. Evening out the distribution of emotions in the dataset may also prove valuable in increasing the accuracy of the artificial intelligence models. Regarding acoustic features, the team recommends expanding the feature-set to include new inputs such as wavelet packets (Wang, 2020), bispectrality features, and/or bicoherence features (Yogesh, 2017). To process these increased volumes of data and features, more complicated machine learning models will be required. A neural network could perform well as they excel in applications involving complex data.

This project is only a steppingstone towards the overall goal of detecting people under the influence of social engineering, but it represents progress towards a novel solution. Ideally, the models could have been trained directly on data representing socially engineered bank patrons and normal transactions. However, due to the absence of a preexisting social engineering audio database and the sensitivity of the information required, securing such data was impossible. Based on the team's literary review, emotional analysis is often the most effective way to discern a socially engineered person. Typically, effected bank patrons exhibit a combination of anger, fear, stress, and anxiety more than those making routine transactions. For this reason, the team believes that identifying individuals in distress will provide a strong basis for social engineering detection.

In Russia, there is currently no system in place capable of automatically detecting fraudulent or fraud-influenced behavior over the phone. Recent rises in social engineering-based complaints to Russian banks have highlighted the problems present in the current manual system. With this rudimentary emotional classification system, the team hopes to contribute to the final goal of creating an effective AI solution to reduce the risk of fraudulent transactions.

Bibliography

A. K. Alimuradov, A. Y. Tychkov, & P. P. Churakov. (2019). "A novel approach to speech signal segmentation based on empirical mode decomposition to assess human psycho-emotional state,". *3rd School on Dynamics of Complex Networks and their Application in Intellectual Robotics,* , 9-12.

Akçay, M. B., & Oğuz, K. (2020). Speech emotion recognition: Emotional models, databases, features, preprocessing methods, supporting modalities, and classifiers. *Speech Communication, 116,* 56-76. doi:10.1016/j.specom.2019.12.001

Alan K. Alimuradov, Alexander Yu. Tychkov, & Pyotr P. Churakov. (2019). Increasing detection efficiency of psycho-emotional disorders based on adaptive decomposition and cepstral analysis of speech signals. Retrieved from https://ieeexplore.ieee.org/document/8708761/authors#authors

Amit Konar, & Aruna Chakraborty. (2015). Maximum a posteriori based fusion method for speech emotion recognition. *Emotion recognition : A pattern analysis approach* (pp. 216-236) John Wiley & Sons, Incorporated.

Anton Batliner, & Bjö Schuller. (2013). Acoustic features. *Computational paralinguistics : Emotion, affect and personality in speech and language processing* (pp. 186-216). Germany: John Wiley & Sons, Incorporated.

Audacity [computer software]

Barnard, D. (2018). Average speaking rate and words per minute. Retrieved from https://virtualspeech.com/blog/average-speaking-rate-words-per-minute

Bercovici, J. (2017). Why the secret to making customer service more human isn't human at all. *Inc.Com,* Retrieved from https://www.inc.com/magazine/201707/jeff-bercovici/cogito-ai-reads-nonverbal-cues.html

Bidgoli, M., & Grossklags, J. (2017). *"Hello. this is the IRS calling.": A case study on scams, extortion, impersonation, and phone spoofing* IEEE. doi:10.1109/ECRIME.2017.7945055

Breiman, L. (2001). Random forests. Retrieved from https://www.stat.berkeley.edu/~breiman/randomforest2001.pdf

Buchanan, B. G. (2005). A (very) brief history of artificial intelligence. *AI Magazine, 26*(4), 53. Retrieved from https://search.proquest.com/docview/208132026

Burkhardt, F., Paeschke, A., Rolfes, M., Sendlmeier, W., & Weiss, B. (2005). *A database of german emotional speech*

Cao, H. (2019, July 15,). Perspectives: I'm developing AI that can read emotions. it's not as creepy as it sounds. *Cnn* Retrieved from https://www.cnn.com/2019/07/15/perspectives/ai-reads-emotions/index.html

Carrol, L. (2011). *The voice lab: Is it just numbers?* np.

Cheng, J., Reddy, S., Saraswat, V., & Lapata, M. (2019). Learning an executable neural semantic parser. *Computational Linguistics, 45*(1), 59-94. doi:10.1162/coli_a_00342

Christin Kirchhübel, & David M. Howard. (2013). Detecting suspicious behaviour using speech: Acoustic correlates of deceptive speech – an exploratory investigation. *Applied Ergonomics, 44*(5), 694-702. Retrieved from https://www.sciencedirect.com/science/article/pii/S0003687012000993#!

Cogito.Our story. Retrieved from https://www.cogitocorp.com/company/

Cole, M. R. (2018). *Hands-on neural network programming with C# : Add powerful neural network capabilities to your C# enterprise applications*. Birmingham, UK: Packt Publishing Ltd.

da Silva, I. N., Hernane Spatti, D., Andrade Flauzino, R., Liboni, L. H. B., & dos Reis Alves, Silas Franco. (2017). *Artificial neural networks A practical course* (1st ed.). Cham: Springer International Publishing. doi:10.1007/978-3-319-43162-8

Del Pozo, I., Iturralde, M., & Restrepo, F. (2018). *Social engineering: Application of psychology to information security* IEEE. doi:10.1109/W-FiCloud.2018.00023

Don't miss the following research. Retrieved from http://nemesysco.com/research

Ellison, D. (2018). Fraud detection using autoencoders in keras with a TensorFlow backend. Retrieved from https://blogs.oracle.com/datascience/fraud-detection-using-autoencoders-in-keras-with-a-tensorflow-backend

Ershov, V. F.The association of russian banks (ARB) and banking community: The practice and prospects of cooperation . *European Researcher, 85*(10-2), 1843-1857. Retrieved from http://www.erjournal.ru/journals_n/1414324366.pdf

Eyben, F. W. (2017). *openEAR - introducing the munich open-source emotion and affect recognition toolkit* IEEE. Retrieved from https://www.openaire.eu/search/publication?articleId=od_______518::4873107633c1f38b058c0c997b0b4f2c

Eyben, F., Wollmer, M., & Schuller, B. (2009/10/12). OpenEAR - introducing the munich open-source emotion and affect recognition toolkit. Paper presented at the *Affective Computing and Intelligent Interaction and Workshops,*

Greitzer, F. L., Strozer, J. R., Cohen, S., Moore, A. P., Mundie, D., & Cowley, J. (2014). *Analysis of unintentional insider threats deriving from social engineering exploits* IEEE. doi:10.1109/SPW.2014.39

H. Cao, D. G. Cooper, M. K. Keutmann, R. C. Gur, A. Nenkova, & R. Verma. (2014). *CREMA-D: Crowd-sourced emotional multimodal actors dataset* doi:10.1109/TAFFC.2014.2336244

Harrison, O. (2019). Machine learning basics with the K-nearest neighbors algorithm. Retrieved from https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761

Hirschberg, J., Beˇnuˇ s, ˇS., Brenier, J. M., Enos, F., Friedman, S., Gilman, S., Girand, C., Graciarena, M., Kathol, A., Michaelis, L., Pellom, B., Shriberg, E., and Stolcke, A. (2005). Distinguishing deceptive from non-deceptive speech. Retrieved from https://www.sri.com/wp-content/uploads/pdf/distinguishing_deceptive_from_non-deceptive_speech.pdf

Hoeschele, M. (2005). *Detecting social engineering* (174th ed.) International Federation for Information Processing.

Hoeschele, M., & Rogers, M. (2005). Detecting social engineering. Paper presented at the 67-77.

Jackson, P., & Haq, S. (2015). Surrey audio-visual expressed emotion (SAVEE) database. Retrieved from http://kahlan.eps.surrey.ac.uk/savee/

Kattan, A., Abdullah, R., & Geem, Z. W. (2011). *Artificial neural network training and software implementation techniques*. Hauppauge, N.Y: Nova Science Publishers.

Kim, P. (2017). *MATLAB deep learning with machine learning, neural networks and artificial intelligence* (1st ed.). Berkeley, CA: Apress. doi:10.1007/978-1-4842-2845-6

Konar, A., & Chakraborty, A. (2015). *Emotion recognition : A pattern analysis approach*. Somerset: John Wiley & Sons, Incorporated. Retrieved from http://ebookcentral.proquest.com.ezpxy-web-p-u01.wpi.edu/lib/wpi/detail.action?docID=1889216

Laribee, L., & Naval Postgraduate School, Monterey Ca. (2006). *Development of methodical social engineering taxonomy project*

Laukka, P., Linnman, C., Åhs, F., Pissiota, A., Frans, Ö, Faria, V., . . . Furmark, T. (2008). In a nervous voice: Acoustic analysis and perception of anxiety in social phobics' speech. *Journal of Nonverbal Behavior, 32,* 195-214. Retrieved from 10.1007/s10919-008-0055-9

Livingstone, S. R., & Russo, F. A. (2018). *The ryerson audio-visual database of emotional speech and song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in north american english* Public Library of Science. Retrieved from https://doi.org/10.1371/journal.pone.0196391

M. Murugappan, S. M. (2013). Human emotion recognition through short time electroencephalogram (EEG) signals using fast fourier transform (FFT). *2013 IEEE 9th International Colloquium on Signal Processing and its Applications,* , 289-294.

Margaret Rouse. (2017). Voice signature. Retrieved from https://whatis.techtarget.com/definition/voice-signature

Moons, B., Bankman, D., & Verhelst, M. (2019). *Embedded deep learning algorithms, architectures and circuits for always-on neural network processing* (1st ed.). Cham: Springer International Publishing. doi:10.1007/978-3-319-99223-5

Mouton, F., Nottingham, A., Leenen, L., & Venter, H. S. (2017). *Underlying finite state machine for the social engineering attack detection model* IEEE. doi:10.1109/ISSA.2017.8251781

Nemesysco. (2016, -01-12T12:58:13+00:00). The LVA technology. Retrieved from http://nemesysco.com/speech-analysis-technology

Oxford, E. D."Social engineering, n.". Retrieved from https://www.oed.com/view/Entry/272695?redirectedFrom=social+engineering

Patan, K. (2019). *Robust and fault-tolerant control neural-network-based solutions* (1st ed.). Cham: Springer International Publishing. doi:10.1007/978-3-030-11869-3

Polamuri, S. (2017, -05-22T19:08:23+00:00). How the random forest algorithm works in machine learning. Retrieved from https://dataaspirant.com/2017/05/22/random-forest-algorithm-machine-learing/

Proofpoint. (2016). *The human factor.* ().Proofpoint.

Raskin, V., Taylor, J. M., & Hempelmann, C. F. (September 2010). Ontological semantic technology for detecting insider threat and social engineering. Paper presented at the *2010 Nspw,* 115-128.

Salahdine, F., & Kaabouch, N. (2019). Social engineering attacks: A survey. *Future Internet, 11*(4) doi:10.3390/fi11040089

Sandouka, H., Cullen, A. J., & Mann, I. (2009). *Social engineering detection using neural networks* IEEE. doi:10.1109/CW.2009.59

Sandouka, H., Cullen, A. J., & Mann, I. (Sep 2009). Social engineering detection using neural networks. Paper presented at the 273-278. doi:10.1109/CW.2009.59 Retrieved from https://ieeexplore.ieee.org/document/5279574

Sausner, R. (2009). SECURITY: Out-of-band authentication gets outfoxed: Social engineering blamed as UK banks work to adapt to a sophisticated new fraud threat that begins with telecoms but leads to consumer bank accounts being drained of funds. *Bank Technology News, 22*(12), 1-n/a. Retrieved from http://ezproxy.wpi.edu/login?url=https://search.proquest.com/docview/208172815?accountid=29120

Shmyrova, V. (2019, July 8,). Three quarters of russian banks vulnerable to cyber attacks. *CNews.Ru* Retrieved from https://safe.cnews.ru/news/top/2019-07-08_tri_chetverti_rossijskih_bankov_uyazvimy_dlya_kiberatak

Shravani, M., R. Narayanan, D. Geyasruti, & S. Lalitha. (2015). Emotion detection using MFCC and cepstrum features. *Procedia Computer Science, 70,* 29-35. Retrieved from https://www.sciencedirect.com/science/article/pii/S1877050915031841#!

Simonite, T. (2018). This call may be monitored for tone and emotion. *Wired,* Retrieved from https://www.wired.com/story/this-call-may-be-monitored-for-tone-and-emotion/

Solanas, A., & Martínez-Ballesté, A. (2010). *Advances in artificial intelligence for privacy protection and security*. Hackensack, N.J: World Scientific.

Sun, L., Zou, B., Fu, S., Chen, J., & Wang, F. (2019). Speech emotion recognition based on DNN-decision tree SVM model. *Speech Communication, 115*, 29-37. doi:10.1016/j.specom.2019.10.004

Tarun Wadhwa. (2018). Why robocalls are about to get more dangerous. *CNN Business,*

Tetri, P., & Vuorinen, J. (2013). Dissecting social engineering. *Behaviour & Information Technology, 23*(10), 1014-1023.

Wang, K. (2020). Wavelet packet analysis for speaker-independent emotion recognition. *Neurocomputing,*

Wiriyathammabhum, P., Summers-Stay, D., Fermüller, C., & Aloimonos, Y. (2017). Computer vision and natural language processing. *ACM Computing Surveys (CSUR), 49*(4), 1-44. doi:10.1145/3009906

Xiangyu, L., Qiuyang, L., & Chandel, S. (2017). *Social engineering and insider threats* IEEE. doi:10.1109/CyberC.2017.91

Yan Zhou, Heming Zhao, Xinyu Pan, & Li Shang. (2015). Deception detecting from speech signal using relevance vector machine and non-linear dynamics features. *Neurocomputing, 151*, 1042-1052. Retrieved from https://www.sciencedirect.com/science/article/pii/S0925231214013435#!

Yiu, T. (2019). Understanding random forest.

Yogesh, C. K. (2017). Hybrid BBO PSO and higher order spectral features for emotion and stress recognition from natural speech. *Applied Soft Computing,* Retrieved from http://dx.doi.org/10.1016/j.asoc.2017.03.013

Zheng, K., Wu, T., Wang, X., Wu, B., & Wu, C. (2019). A session and dialogue-based social engineering framework. *IEEE Access, 7*, 67781-67794. doi:10.1109/ACCESS.2019.2919150

Appendix A: Authorship

Daniel Aksman mainly focused on feature extraction and selection research. He participated in the writing of much of the background and literature review section and assisted heavily with overall structuring and editing of the paper. Jointly working with Alex Malkhasov and Zack Fitzgibbon, he contributed to the methodology, findings, and background sections about AI architecture testing while leading the front of writing for the background, introduction, methodology, and findings sections about feature extraction and selection. Daniel also contributed heavily to the collection of data sections of the paper, which mainly consist of the methods we used to obtain our speech emotion datasets.

Zackary Fitzgibbon focused on the development of the different datasets used to train the AI models. He also worked with Alex Malkhasov to create and train said models on these datasets. He contributed to the paper in the findings and methodology sections about the databases, datasets, and AI models used. Zackary also contributed to the background with sections on social engineering and companies that use emotion detecting software along with lending a hand to other sections of the report where it was needed.

Alex Malkhasov was the head of data science research and development. He created multiple models to test for correlation between various acoustic features within speech signals. Alex coded a variety of different tests to offer the reader a visual representation of the data trends, the different emotion labels, and between different features themselves (including the pair plots and heatmaps). Arseniy Pozdniakov contributed greatly to the random forests background section, as well as the coding behind the speaking rate and jitter acoustic features.

Alec Kneedler served as the lead editor of the report and contributed to team organization and management. Contributions to the research body included expertise classification algorithm architecture with a focus on neural networks. Primary additions to the report included the introduction, executive summary, and conclusion. The most intensive editing was the synthesis of the methodology and findings sections.

Appendix B: Python code to create CSV file

```python
import librosa
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
import pandas as pd
import datetime
import parselmouth
import statistics
from scipy.signal import argrelextrema
from scipy.io import wavfile

# Extract features from a sound file
def extract_feature(file_name):
    y, sr = librosa.load(file_name)
    hop_length = 512
    result = []

    # get standard deviation and mean of chroma
    stft = np.abs(librosa.stft(y))
    chroma_std = np.std(librosa.feature.chroma_stft(S=stft, sr=sr))
    result.append(chroma_std)
    chroma_mean = np.mean(librosa.feature.chroma_stft(S=stft, sr=sr))
    result.append(chroma_mean)

    # get standard deviation and mean of mel
    mel_std = np.std(librosa.feature.melspectrogram(y, sr=sr))
    result.append(mel_std)
    mel_mean = np.mean(librosa.feature.melspectrogram(y, sr=sr))
    result.append(mel_mean)


    # Spectral Centroid Feature ('cent' is an array of values, 'cent_mean' is the mean of the array)
    cent = librosa.feature.spectral_centroid(y=y, sr=sr)
    cent_mean = np.mean(cent)
    result.append(cent_mean)
    # MFCC, delta, and delta second order
    # Compute MFCC features from the raw signal
    mfcc = librosa.feature.mfcc(y=y, sr=sr, hop_length=hop_length, n_mfcc=13)

    # get standard deviation and mean of mfcc
    mfcc_std = np.std(mfcc)
    result.append(mfcc_std)

    mfcc_mean = np.mean(mfcc)
    result.append(mfcc_mean)

    # And the first-order differences
    mfcc_delta = np.mean(librosa.feature.delta(mfcc))
    result.append(mfcc_delta)

    # And second-order differences
    mfcc_delta2 = np.mean(librosa.feature.delta(mfcc, order=2))
    result.append(mfcc_delta2)

    # RMS feature and mean
    rms = librosa.feature.rms(y=y)
    rms_mean = np.mean(rms)
    result.append(rms_mean)

    # Spectral Rolloff Mean
    rolloff = librosa.feature.spectral_rolloff(y=y, sr=sr,
roll_percent=0.1)
    rolloff_mean = np.mean(rolloff)
    result.append(rolloff_mean)

    #Spectral Rolloff Range
    rolloff_range = np.ptp(rolloff)
    result.append(rolloff_range)

    # Zero Crossing Rate
    ZCR = librosa.feature.zero_crossing_rate(y)
    result.append(rolloff_mean)

    # speaking rate of the file
    samplerate, data = wavfile.read(file_name)
    data_voiced = []
    for i in data:
        if abs(i) >= 50:
            data_voiced.append(i)
    x = np.arange(len(data))/float(samplerate)
    y = np.arange(len(data_voiced))/float(samplerate)
    result.append(y[len(y)-1] / x[len(x)-1])

    # Jitters
    snd = parselmouth.Sound(file_name)
    intensity = snd.to_intensity()
    narr = np.array(intensity.values[0])
    lst_temp = list(narr)
    minimas = argrelextrema(narr, np.less)
    lst = list(minimas[0])
    period_borders = []
    for i in range(0,len(lst) - 1):
        period_borders.append(intensity.xs()[lst[i]])
    periods = []
    # got error that period_borders[0] doesnt exist for a file
    if(period_borders[0]):
        periods.append(period_borders[0])
        for i in range(1,len(period_borders)):
            periods.append(period_borders[i] - period_borders[i-1])
        jitters = []
        for i in range(1,len(periods) - 1):
            jitters.append(abs(periods[i-1] - periods[i]) /
(sum(periods) / len(periods)))
        #gets standard deviation of jitters
        result.append(statistics.stdev(jitters))

    return result

# possible emotions
emotions = {
    #RAVDESS emotions
    '01': 'neutral',
    '02': 'calm',
    '03': 'happy',
    '04': 'sad',
    '05': 'angry',
    '06': 'fearful',
    '07': 'disgust',
    '08': 'surprised',
    #EMO-DB emotions
    'W': 'angry',
    'L': 'boredom',
    'E': 'disgust',
    'A': 'fearful',
    'F': 'happy',
    'T': 'sad',
    'N': 'neutral',
    #SAVEE emotions
    'a': 'angry',
    'd': 'disgust',
    'f': 'fearful',
    'h': 'happy',
    'n': 'neutral',
    'sa': 'sad',
    'su': 'surprised',
    #CREMA-D emotions
    'ANG': 'angry',
    'FEA': 'fearful',
    'DIS': 'disgust',
    'HAP': 'happy',
    'SAD': 'sad',
    'NEU': 'neutral'
}

# Load the data and extract features for each sound file
def load_data():
    file_dics = []
    # get data from RAVDESS dataset
    for file in glob.glob("....\\data\\English Data\\Actor_*\\*.wav"):
        dir_name = os.path.dirname(file)
        file_name = os.path.basename(file)
        emotion = emotions[file_name.split("-")[2]]
        results = extract_feature(dir_name + '\\' + file_name)
        data = {'File Name': file_name, 'Emotion': emotion,
            'Chroma Standard Deviation': results[0],
            'Croma Mean': results[1],
            'MEL Standard Deviation': results[2],
            'MEL Mean': results[3],
            'Spectral Centroid Feature': results[4],
            'MFCC Standard Deviation': results[5],
            'MFCC Mean': results[6],
            'MFCC 1st Derivative': results[7],
            'MFCC 2nd Derivative': results[8],
                'Root Mean Square Mean': results[9],
                'Spectral Rolloff Mean': results[10],
                'Spectral Rolloff Range': results[11],
                'Zero Crossing Rate': results[12],
                'Speaking Rate': results[13],
                'Jitters': results[14]
            }
            file_dics.append(data)

    # get data from EMO-DB
    for file in glob.glob("....\\data\\German Data\\wav\\*.wav"):
        dir_name = os.path.dirname(file)
        file_name = os.path.basename(file)
        emotion = emotions[file_name[5]]
        results = extract_feature(dir_name + '\\' + file_name)
        data = {'File Name': file_name, 'Emotion': emotion,
            'Chroma Standard Deviation': results[0],
            'Croma Mean': results[1],
            'MEL Standard Deviation': results[2],
            'MEL Mean': results[3],
            'Spectral Centroid Feature': results[4],
            'MFCC Standard Deviation': results[5],
            'MFCC Mean': results[6],
            'MFCC 1st Derivative': results[7],
            'MFCC 2nd Derivative': results[8],
            'Root Mean Square Mean': results[9],
            'Spectral Rolloff Mean': results[10],
            'Spectral Rolloff Range': results[11],
            'Zero Crossing Rate': results[12],
            'Speaking Rate': results[13],
            'Jitters': results[14]
        }
        file_dics.append(data)

    # get data from SAVEE dataset
    for file in glob.glob("....\\data\\British Data\\*\\*.wav"):
        dir_name = os.path.dirname(file)
        file_name = os.path.basename(file)
        e = file_name[0]
        if e == 's':
            e+= file_name[1]
        emotion = emotions[e]
        results = extract_feature(dir_name + '\\' + file_name)
        data = {'File Name': file_name, 'Emotion': emotion,
            'Chroma Standard Deviation': results[0],
            'Croma Mean': results[1],
            'MEL Standard Deviation': results[2],
            'MEL Mean': results[3],
            'Spectral Centroid Feature': results[4],
            'MFCC Standard Deviation': results[5],
            'MFCC Mean': results[6],
            'MFCC 1st Derivative': results[7],
            'MFCC 2nd Derivative': results[8],
            'Root Mean Square Mean': results[9],
            'Spectral Rolloff Mean': results[10],
            'Spectral Rolloff Range': results[11],
            'Zero Crossing Rate': results[12],
                'Speaking Rate': results[13],
                'Jitters': results[14]
            }
            file_dics.append(data)

            # get data from CREMA-D dataset
        for file in glob.glob("....\\data\\AudioWAV\\*.wav"):
            dir_name = os.path.dirname(file)
            file_name = os.path.basename(file)
            split = file_name.split('_')
            emotion = emotions[split[2]]
            results = extract_feature(dir_name + '\\' + file_name)
            data = {'File Name': file_name, 'Emotion': emotion,
                'Chroma Standard Deviation': results[0],
                'Croma Mean': results[1],
                'MEL Standard Deviation': results[2],
                'MEL Mean': results[3],
                'Spectral Centroid Feature': results[4],
                'MFCC Standard Deviation': results[5],
                'MFCC Mean': results[6],
                'MFCC 1st Derivative': results[7],
                'MFCC 2nd Derivative': results[8],
                'Root Mean Square Mean': results[9],
                'Spectral Rolloff Mean': results[10],
                'Spectral Rolloff Range': results[11],
                'Zero Crossing Rate': results[12],
                'Speaking Rate': results[13],
                'Jitters': results[14]
            }
            file_dics.append(data)

        return file_dics

file_dics = load_data()

data_set = pd.DataFrame(file_dics)
data_set.to_csv ('....\\data\\Emotion Data.csv', index = False,
   header=True)
```

Appendix C: Python file to train AI models

```python
import librosa
import soundfile
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.neural_network import MLPClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
import pandas as pd
emotions = {'neutral': 0,
            'calm': 1,
            'happy': 2,
            'sad': 3,
            'angry': 4,
            'fearful': 5,
            'disgust': 6,
            'surprised': 7,
            'boredom': 8}

def load_data():
    data = pd.read_csv("....\\data\\Emotion Data.csv")
    data["Emotion"] = data["Emotion"].map(emotions)
    names = data["File Name"]
    y = data["Emotion"]
    data = data.drop("Emotion", axis = 1)
    data = data.drop("File Name", axis = 1)
    return train_test_split(data, y, test_size = 0.3)

print("Iteration: ", x )
# Split the dataset
x_train, x_test, y_train, y_test = load_data()

# Initialize the Model to be train
# Models used are at lines 43, 45, and 47
model =LogisticRegression(random_state = 11, verbose = 3, n_jobs = -1)

# MLPClassifier(alpha=0.01, batch_size=256, epsilon=1e-08, hidden_layer_sizes=(50,50,50,50,50,50,50,50,50,50), learning_rate='adaptive',max_iter=250, solver = 'adam')
# LogisticRegression(random_state = 11, verbose = 3, n_jobs = -1)
# RandomForestClassifier(oob_score = True, n_jobs = -1, random_state = 11, max_depth = 31, verbose = 3)

# Train the model
model.fit(x_train, y_train)

# Predict for the test set
y_pred = model.predict(x_test)

# Calculate the accuracy of our model
accuracy = accuracy_score(y_true=y_test, y_pred=y_pred)
# Print the accuracy
print("Accuracy: {:.2f}%".format(accuracy))
```

Appendix D: python file to normalize data

```python
import csv
import numpy as np
import pandas as pd

## Initialize Arrays
File_Name = []
Emotion = []

## Chroma
 Chr_SD = []
 Chr_M = []

 ## MEL
 MEL_SD = []
 MEL_M = []

 ## Spectral centroid feature
 SCF = []

 ## MFCC
 MFCC_SD = []
 MFCC_M = []
 MFCC_1D = []
 MFCC_2D = []

 ## Root Mean Square
 RMS_M = []

 ## Spectral rolloff
 SR_M = []
 SR_R = []

 ## Zero crossing rate
 ZCR = []

 ## Index through the rows of the csv file and add values to the matrices
 with open('....\\data\\Emotion Data.csv', 'r') as f:
     reader = csv.reader(f, delimiter=',')
     for row in reader:
         File_Name.append(row[0])
         Emotion.append(row[1])
         Chr_SD.append(row[2])
         Chr_M.append(row[3])
         MEL_SD.append(row[4])
         MEL_M.append(row[5])
         SCF.append(row[6])
         MFCC_SD.append(row[7])
         MFCC_M.append(row[8])
         MFCC_1D.append(row[9])
         MFCC_2D.append(row[10])
         RMS_M.append(row[11])
         SR_M.append(row[12])
```

```
            SR_R.append(row[13])
            ZCR.append(row[14])

    File_Name.pop(0)
    Emotion.pop(0)
    Chr_SD.pop(0)
    Chr_M.pop(0)
    MEL_SD.pop(0)
    MEL_M.pop(0)
    SCF.pop(0)
    MFCC_SD.pop(0)
    MFCC_M.pop(0)
    MFCC_1D.pop(0)
    MFCC_2D.pop(0)
    RMS_M.pop(0)
    SR_M.pop(0)
    SR_R.pop(0)
    ZCR.pop(0)

    File_Name = np.array(File_Name)
    Emotion = np.array(Emotion)
    Chr_SD = np.array(Chr_SD).astype(np.float)
    Chr_M = np.array(Chr_M).astype(np.float)
    MEL_SD = np.array(MEL_SD).astype(np.float)
    MEL_M = np.array(MEL_M).astype(np.float)
    SCF = np.array(SCF).astype(np.float)
    MFCC_SD = np.array(MFCC_SD).astype(np.float)
    MFCC_M = np.array(MFCC_M).astype(np.float)
    MFCC_1D = np.array(MFCC_1D).astype(np.float)
    MFCC_2D = np.array(MFCC_2D).astype(np.float)
    RMS_M = np.array(RMS_M).astype(np.float)
    SR_M = np.array(SR_M).astype(np.float)
    SR_R = np.array(SR_R).astype(np.float)
    ZCR = np.array(ZCR).astype(np.float)


    Chr_SD = np.interp(MEL_SD, (MEL_SD.min(), MEL_SD.max()), (0,1))
    Chr_M = np.interp(MEL_SD, (MEL_SD.min(), MEL_SD.max()), (0,1))
    MEL_SD = np.interp(MEL_SD, (MEL_SD.min(), MEL_SD.max()), (0,1))
    MEL_M = np.interp(MEL_SD, (MEL_SD.min(), MEL_SD.max()), (0,1))
    SCF = np.interp(MEL_SD, (MEL_SD.min(), MEL_SD.max()), (0,1))
    MFCC_SD = np.interp(MEL_SD, (MEL_SD.min(), MEL_SD.max()), (0,1))
    MFCC_M = np.interp(MEL_SD, (MEL_SD.min(), MEL_SD.max()), (0,1))
    MFCC_1D = np.interp(MEL_SD, (MEL_SD.min(), MEL_SD.max()), (0,1))
    MFCC_2D = np.interp(MEL_SD, (MEL_SD.min(), MEL_SD.max()), (0,1))
    RMS_M = np.interp(MEL_SD, (MEL_SD.min(), MEL_SD.max()), (0,1))
    SR_M = np.interp(MEL_SD, (MEL_SD.min(), MEL_SD.max()), (0,1))
    SR_R = np.interp(MEL_SD, (MEL_SD.min(), MEL_SD.max()), (0,1))
    ZCR = np.interp(MEL_SD, (MEL_SD.min(), MEL_SD.max()), (0,1))

    data = []
    data.append(File_Name)
    data.append(Emotion)
    data.append(Chr_SD)
    data.append(Chr_M)
    data.append(MEL_SD)
    data.append(MEL_M)
```

```
data.append(SCF)
data.append(MFCC_SD)
data.append(MFCC_M)
data.append(MFCC_1D)
data.append(MFCC_2D)
data.append(RMS_M)
data.append(SR_M)
data.append(SR_R)
data.append(ZCR)

data_set = pd.DataFrame(data)
data_set.to_csv ('....\\data\\Emotion Data Normalized.csv', index = False, header=True)
```

Appendix E: Python file to get graphs of data

```python
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.manifold import TSNE
from sklearn.preprocessing import StandardScaler

data = pd.read_csv("..\\data\\Emotion Data.csv")

 # map emotions to number
 emotions = {'neutral': 0,
             'calm': 1,
             'happy': 2,
             'sad': 3,
             'angry': 4,
             'fearful': 5,
             'disgust': 6,
             'surprised': 7,
             'boredom': 8}
 data["Emotion"] = data["Emotion"].map(emotions)

 # display heatmap of data correlation
 sns.heatmap(data.corr());

 #display pairplot of data correlation
 %config InlineBackend.figure_format = 'png'
 sns.pairplot(data[data.columns[1:]]);

 names = data["File Name"]
 data = data.drop("File Name", axis = 1)

 # get TSNE plot
 tsne = TSNE(n_components = 2, random_state = 11)
 tsne_repr = tsne.fit_transform(StandardScaler().fit_transform(data))
 plt.scatter(tsne_repr[:, 0], tsne_repr[:, 1],c=data['Emotion'].map({0:
"blue", 1: "brown", 2: "orange", 3: "green", 4: "red",5: "purple", 6:
"pink", 7: "grey", 8: "navy"}), alpha=.5);

 ## creating new dataset object not to add changes to object "data"
 df = data.copy()

 # get feature distribution of MFCC Mean
 np.sort(df["MFCC Mean"].unique())
 df.loc[df["MFCC Mean"] < -80, "MFCC Mean"] = -81
 df.loc[df["MFCC Mean"] >= 0, "MFCC Mean"] = 1

 for value in range(-80, 0, 5):
     df.loc[((df["MFCC Mean"] >= value) & (df["MFCC Mean"] < value +
5)), "MFCC Mean"] = value + 2.5

 plt.figure(figsize = (15, 8))
 sns.countplot(data = df, x = "MFCC Mean", hue = "Emotion");

```

```
## creating new dataset object not to add changes to object "data"
df = data.copy()

# get feature distribution of Chroma Standard Deviation
np.sort(df["Chroma Standard Deviation"].unique())
df.loc[df["Chroma Standard Deviation"] < 0.05, "Chroma Standard Deviation"] = 0.04
df.loc[df["Chroma Standard Deviation"] >= 0.32, "Chroma Standard Deviation"] = 0.35

for value in range(5, 32, 3):
    df.loc[((df["Chroma Standard Deviation"] >= value * 0.01) & (df["Chroma Standard Deviation"] < value * 0.01 + 0.03)), "Chroma Standard Deviation"] = round(value * 0.01 + 0.015, 2)

plt.figure(figsize = (15, 8))
sns.countplot(data = df, x = "Chroma Standard Deviation", hue = "Emotion");
## creating new dataset object not to add changes to object "data”
df = data.copy()

# get feature distribution of Chroma Mean
np.sort(df["Croma Mean"].unique())
df.loc[df["Croma Mean"] < 0.45, "Croma Mean"] = 0.41
df.loc[df["Croma Mean"] >= 0.7, "Croma Mean"] = 0.74
value = -0.19

while value < 0.7:
    df.loc[((df["Croma Mean"] >= value) & (df["Croma Mean"] < value + 0.05)), "Croma Mean"] = round(value + 0.04, 2)
    value += 0.05

np.sort(df["Croma Mean"].unique())
plt.figure(figsize = (15, 8))
sns.countplot(data = df, x = "Croma Mean", hue = "Emotion");
```