Validation of Scoring Criteria and Scoring Thresholds within GWEEK's Speech Intelligence Analytics Levels

Industrial Team Project

Abhijeet Thakur, Ibad Ur Rahman, Martin Skauen, Rahul Sengupta

A report on Industrial Team Project

The University Of Sheffield.

Computer Science
The University Of Sheffield
United Kingdom
May 12,2019

;

Abstract

Speech Classification and Analysis were the core purpose of this project. The two main objectives of this project were identifying the existence of any relationship between GWEEK score and Vocabulary, and classifying between Read and Planned Speech. As a result of the first task, it was found that Vocabulary did not have a major impact on the GWEEK score. For the second task, Deep Learning, as well as classical Machine Learning approaches were applied on raw audio data as well as extracted text data from the audio. The classical machine learning methods on extracted text fetched better results than the Deep Learning approach on raw audio, which implies that extracted features given to model work better than raw features like audio.

Acknowledgements

First of all, we would like to appreciate the support from GWEEK Team for making us understand the objectives and helping us in setting up the program and scripts for the API. The GWEEK team was quick in giving replies to our emails and this project would not have been possible without their support. We would like to give special thanks to our supervisor Professor Thomas Hain who guided us in the methodologies which helped us in our project, as all four of us did not have any experience with speech technology. He helped us a lot in understanding the background knowledge required for this project. Lastly, we would like to thank The University of Sheffield for believing in us and granting the opportunity to work on this amazing project.

Acronyms

Abbreviations	Full Form
RMSE	Root Mean Square Error
JSON	Javascript Object Notation
ASR	Automatic Speech Recognition
CNN	Convolotional Neural Network
MFCC	Mel-Frequency Cepstral Coefficient
NLP	Natural Language Processing
TF-IDF	Term Frequency Inverse Document Frequency
SVD	Singular Value Decomposition
API	Application Programming Interface
PCA	Principal Component Analysis
SIL	Silence

Contents

List of Figures

List of Tables

1 Introduction

1.1 A Broad Understanding

Speech Processing is the study of speech signals and the methods involved in the process. It is the analysis of a human speech that uses Digital Signal Processing. There are many aspects of speech processing: speech synthesis, voice or speech recognition, voice analysis, speaker recognition, speech coding and compression, speech enhancement, etc. Initially, speech processing and recognition were mainly focused on understanding a few simple phonetic elements such as vowels until three researchers at Bell Labs, namely Stephen. Balashek, R. Biddulph, and K. H. Davis developed a system that could figure out the digits spoken by a human, in 1952.[1] By the early 2000s, neural networks and deep learning came in to use as the primary Speech Processing techniques. Today, speech processing is growing exponentially in popularity.

1.2 Relevance

Speech is a natural interface for many programs that don't run on computers, which are increasingly becoming more common. Some applications are:

- **Artificial Intelligence:** Voice Assistants like Alexa, Siri, etc use Natural Language Processing to help humans do day-to-day tasks like web-searching, navigation, playing music, reading news, etc. Moreover, robots are increasingly being employed in roles once performed by humans, including in conversation and interface.
- **Helping the Visually- and Hearing-Impaired:** On-screen Readers, that convert text to speech, are used by many visually-impaired people. On the other hand, converting audio into text is a critical communication tool for people with hearing impairment.
- **Compression:** Transcription, an aspect of Speech Processing, is a very important procedure where speech is given as input and text is received as an output. It has found its use in several business firms, legal and medical purposes. The most glaring benefit that comes with it is data compression: a text file containing the same information as an audio file is much smaller compared to the original audio file.

Apart from the points mentioned above, there are many more applications of Speech Processing one of which is the field of research in this paper.

1.3 What GWEEK Does

Often, many people face the dilemma of public-speaking, most of it comes from the lack of confidence. The main objective of public-speaking is to get one's message across, and for that, it is imperative to have good communication skills. It involves many aspects like pitch, tonality, intonation, vocabulary, etc. In summary, what GWEEK does is to analyse a user's speech and give him/her feedback on the things that need to be approved. In addition, a score called GWEEK Score, which is marked out of 0-100 is provided. In this way, users have the option to compare themselves with other talented users worldwide and eventually help themselves in various scenarios in life, both public and private. Above all, GWEEK[2] is like a personal speech coach. There are four levels of learning (Speech Intelligence), which give personalised feedback to the user. They are:

1. **Si1 - Audio Capture:** In this level, the focus is on para-linguistic and verbal behaviours, where the user is expected to be clear and confident.
2. **Si2 - Video Capture:** In this level, para-linguistic and non-verbal behaviours are checked where the user is expected to look more natural.
3. **Si3 - Video Capture:** In this level, semantic complexity plays importance where the user is expected to be more impactful.
4. **Si4 - Video Capture:** In this level, the focus is on discourse structure where the user is expected to make a lot of sense while speaking.

The step-by-step progress of a user over time is reflected by the GWEEK Score to help the user achieve the learning criteria at each level. Interesting as it is, a user can compare his/her abilities with global leaders, analysed at the same Si Level of the user.

1.4 Background

Through the years there has been a lot of research involving speech processing by using it as a medium to help people with speaking disorders. Jiang, Haihua et al.[3] released a paper in 2015, connecting certain acoustic features in speech with depression, which is considered the most widespread mental disorder. They compared three types of speech in the analysis; interview, picture description and reading. The results showed that classification accuracy (depression/not depression) amongst men was proven to be significantly higher in picture description (vs reading and interviews). In 2016, Duc Le et al.[4] developed a method for automatically assessing intelligibility amongst people suffering from aphasia. The method assesses three types of intelligibility aspects; prosody, fluidity and clarity. Their work included an application designed for therapeutic purposes due to the lack of human expertise in the field.

As shown, analysis of speech (typically prosodic features in speech) can be vital for many people's health, communication abilities etc. However, there are other research topics not related to speaking disorders, that are still very much relevant to us when working alongside GWEEK. ASR has improved drastically in recent years, and all the important IT companies have their own software, and in many languages too. Perhaps the most common problem is recognising word by word, but there are additional challenges faced in speech technology. One is classification. KJ Piczak et al.[5] did research on sound classification with respect to common sounds like animal noise, carpenter equipment etc. In conclusion, it is stated that deep learning approaches have the ability to outperform approaches where features are extracted manually. Their research states that convolutional neural networks work well even when the data is limited and the augmentation is simple. However, to get the most out of a CNN a considerably large amount of data is required. The training time and computational costs are the weaknesses of this approach.

A way of approaching speech, is by modelling the use of pauses, like in this journal [6] by Igras-Cybulska, Magdalena, et al. Typically, the way of quantifying pause-use is by measuring the duration and frequency. Pauses in speech can broadly be discriminated into three groups; silent, filled, and breath pause. A filled pause is when a speaker makes use of a pause to think about what to say next but then fills the pause with meaningless sound[7]. Frequent, lengthy use of filled pauses in, let's say an interview or speech will obviously lead to a decreased fluency and efficiency. It might give the listener the impression that the speaker is not well prepared. On the other hand, if the speaker is not given time to pause, he or she might not formulate the information understandably. Additionally, it will be harder for listeners to digest the input if the speaker speaks too fast[7].

Pauses, therefore, play a very arguable role in communication between a speaker and his/her audience. The team mentioned early in the previous paragraph [6] considered pause as a prosodic feature to model the Polish language. The main aim of the project was to research if pause features could improve existing speaker recognition algorithms. That is, in a data set where there are multiple speakers, each adding multiple recordings; will the system recognise each speaker on a new test set? In general, these algorithms apply pitch, energy, MFCC or segmental information. A side task which was extremely relevant to this project was that the team ran classification algorithms to classify read and spontaneous speech on the same recordings. At best, an accuracy of 75% was obtained when classifying strictly using pause features fed to an extreme gradient boosting algorithm. It was concluded that pause as a classification feature like this is more decisive than in the task of recognising speakers[6].

In order to approach Vocabulary in a way that was significant to our research, we had to look at this case study[**?**] by Mirjana Kovac which investigated the frequency and distribution of speech errors, as well as the influence of the task type on their rate. 101 engineering students in Croatia took part in the study where a recorded speech sample in the English language lasting for approximately ten hours was transcribed, whereby more than three and a half thousand speech errors were recorded. It was found that Morphological errors were dominant due to a significantly frequent omission of articles. The distribution of different subcategories of lexical errors pointed to a relatively low frequency of unintended switches to their native languages, indicating that the participants were able to separate the two languages during lexical access. Statistical testing of the influence of the task type on speech errors displayed that the retelling of the chronological order of events resulted in a significantly higher rate of syntactic errors if compared to other tasks. The rate of lexical and phonological errors depended on the frequency of use, that is, less frequent words were more susceptible to lexical errors than high-frequency words.

On the other hand, it is assumed that speakers who have English as their native language would not be vulnerable to the above errors, and as a result, their lexical range would be much higher than the people who have English as their second language. Moreover, people who hardly communicate in English in their day-to-day lives would display the highest number of speech errors. As a result, it was decided that it was in the best interest to choose recordings of the top two categories of English language speakers so that the Vocabulary or lexical usage had a wide range.

1.5 Task Definition

Keeping the previous researches in mind, our aim was to enhance the GWEEK App capabilities in a way that the end-user gets to have a more detailed analysis by incorporating some new aspects in the analysis criteria. In this paper, the primary focus consists of two pertinent questions:

1. **Can Vocabulary be used in a predictive model to predict GWEEK score?**
2. **Is it possible to classify Planned and Read speech, given that some users have Read text into the app?**

It was agreed upon that for Si1 level analysis purposes, the team at GWEEK would give us the required *.wav* audio files along with their JSON equivalents.

1.6 Sections

The report is split up in the following way:

Methods contains our early challenges and planning of the project, methods and algorithms for both task 1 and 2 as well as data collection and changes in the process. **Results** contains all the results of the analysis. **Discussion** is the part where the methods are set up against each other for comparison with a basis in the result, along with an interpretation of the results. Finally, the **Conclusion** section contains the conclusion that is drawn from this project work, which is followed by some suggestions for future implementations that can be done on this, under **Future Work**.

2 Methods

2.1 Initial Challenges

Before proceeding with the methods, there were some questions that came to our mind. The primary focus was on these three questions:
1. How is the GWEEK Score generated?
2. What is the end-to-end journey?
3. How should Vocabulary, Planned Speech and Read Speech be defined?

As per GWEEK, to answer the first question, for all Si analytic levels, specific features are extracted from each 1-2 minute audio/video recording. These become 'input' data points which enter a function containing an algorithm to compute the score.

The end-to-end journey: a speaker speaks for 60-120 seconds ⇀ the audio file gets uploaded to the GWEEK server ⇀ its features are extracted by AUTO-GWEEK ASR/NLP layers ⇀ inputs (extracted features) enter the GWEEK Score algorithm ⇀ GWEEK Score and personal feedback are generated ⇀ they are returned to the learner's app.

The answer to the third question was a bit tricky. In general terms, the word, vocabulary, refers to the body of words in a language, but on the other hand, making our program understand the nuances of English language was a bit of a task. Additionally, each user is given a topic to give the recording on, a topic which could limit the vocabulary of that user, and possibly affect the GWEEK score. However, our task was not topic detection. The decision was that vocabulary should only extend up to the lexical variety of the speech in 1) Each audio file 2) All audio files combined. Coming to defining the two aforementioned categories of speeches, we decided that we would bracket our Planned Speech to any speech given by a user that was not Read. Planned Speech data would include spontaneous speech, prepared speech, extemporaneous speech, entertaining speech, informative speech, a demonstrative speech, persuasive speech, oratorical speech, special occasion speech, motivational speech, debate speech and all other kinds of speeches that involved a user not reading from any material. On the other hand, just as the name suggests, Read Speech would be the speeches given by different users (one at a time), by taking reading out loud any text material including taking help from promoters.

Figure 1: Graphic Visualisation of GWEEK Users' Vocabulary

2.2 Data Cleaning

Quite early on, 2 of the team members were allocated the job of data cleaning. This was quite a comprehensive job as it consisted of downloading, converting, and external files from an external source. The purpose of this task was to access labelled data (Read/spontaneous speech) as the data given by GWEEK was unlabelled. This was a hindrance to train our algorithm. However, once labelled data was resourced and cleaned, we could feed it to an algorithm (which will be talked more about later) and then test on the unlabelled GWEEK data.

2.3 Task 1: Vocabulary

As for the first task, the initial approach was an idea of statistical analysis, including some basic data exploration and correlation plots. A subset of 492 JSON files was used. The distribution of GWEEK score, a correlation between what the team defined as features describing vocabulary (the features are described below), and correlation plots between these features and the target variable GWEEK score were considered important. However, a more extensive analysis was suggested; Build 2 classification models and 1 regression model for predicting GWEEK score using 3 vocabulary based features. The 2 classifiers are of the formats 1) Binary and 2) Multiclass. It was challenging to determine the thresholds for this task. The primary idea was to use the subset of 492 to set the thresholds, assuming this would represent the full data. As the JSON file generated from GWEEK was proprietary, the whole process had to be reverse-engineered to get to the core of the calculation of the GWEEK Score.

For both Binary and Multiclass, we calculated 3 variables. That was Feature1: Count of distinct words given by a user (per second), Feature2: Fraction of unique words per all words, Feature3: Count of words given by each user, but not by any other user. These features would be referenced to like the same throughout the report.

As the results were not satisfactory, new ideas were thought of. The next approach involved TF-IDF, which is a popular pre-processing technique in NLP. A Sci-kit Learn library was used to describe each word in the subset into a vector used as a feature for the predictive model. Each user, therefore, gets a TF-IDF score from each word, in terms of how many times the word is mentioned by the user and how wide-spRead the word is across the population. A reference and further explanation are provided here [9]. The output was a very sparse matrix. An attempt to apply Truncated SVD was used to deal with that sparsity. The process involved the reduction of a number of columns from the total number of unique words in the whole set

to 5. The data was normalised before fed into the SVD. More information about Sklearn's SVD library can be found here[10].

2.4 Algorithms

To utilise the modelling architecture, it was decided to try out more various algorithms for classification. Sci-kit learn has a logistic regression module. It is a statistical method for predicting Binary and Multiclass classification problems. Basically, it is a special case of linear regression where the target variable acts as categorical in nature as it uses a log-odds as the dependent variable. Furthermore, estimation in logistic regression is done through maximum Likelihood[11]. Moreover, there is Naive Bayes, which is a relatively simple and efficient algorithm used for classification. It simplifies the metrics calculation by calculating the probabilities of the attribute for a given class independent of the values for other attributes[12]. Random Forest Classifier is a meta estimator that fits the numerous number of a decision tree on various sample size from a data set, then average all of them to gradually improve and get the best accuracy with controlled over-fitting. We are using "Random Forest Classifier" for the classification because it uses group classifier instead of single to predict the target[13].

Gradient Boosting focuses consecutively to reduce error with each model until it gets the best model. Gradient Boosting uses weak learner to predict outcomes. This weak model we predict loss function which helps us to reduce error[14]. It uses function space for the optimisation, rather than in parameter space, which makes the use of custom loss function quite easy. It focuses on different examples step by step which helps it in learning, how to deal with unbalanced data[15], so it is suitable for our data. The evaluation step also needs to be emphasised. Due to the imbalance in GWEEK scores, computing accuracies might not give a very interpretable result if the objective is to pick the best algorithm. For this reason, we have also computed F1-score wherever it is possible. F1-score is a weighted average of precision and recall and is explained in detail here [16].

2.5 Task 2: Data Collection

Moving on to the second task, which was a classification of Read vs spontaneous speech, the initial idea was to feed labelled data to a deep learning algorithm. The data given by GWEEK was possible of sufficient size to train a neural network, however, the data was not labelled. Our plan included downloading TED talks to represent Planned (spontaneous) speech and audiobooks data to represent Read speech. The collection of data then consisted of 11 GB of Read speech and around 18 GB of Planned speech. After the aforementioned data cleaning process was completed, we were left with 3300 2-minute recordings of Read speech, and around 6000 2-minute recordings of Planned speech. All recording was in *.wav* format.

2.6 Task 2: Neural Network Approach

There are many methods used to classify audio. One of which is directly re-sampling the raw audio files and feeding it to a classifier, another can transform the audio using Fourier Transform before feeding it. Amongst all methods, the one giving that seems to give the best outcome for other similar datasets is the MFCC approach.

2.6.1 MFCC (Mel Frequency Cepstral Coefficient)

This approach is used for many audio classification tasks and since in almost all classification tasks, the most important thing to give to neural network model is the features that can in our case distinguish between Read speech and Planned speech. So since the audio signal is complex and can change in every bit hence the signal is framed to shorter frames, then the power spectrum of each frame is calculated, then the periodogram estimate of the power spectrum is calculated, after that the Mel filter-bank is applied and with a couple of more steps we finally get a matrix.

Once the MFCC was chosen as the primary approach we later agreed to try feeding the images to a Convolutional Neural Network, after a discussion with our supervisor. In fact, there are numerous advantages given such an approach. CNN is due to the presence of parameter sharing and pooling operations computationally efficient compared to its successors. CNN also holds the ability to learn prominent features without human knowledge[17]. What features it was going to learn given TED talks and audio-books, however, was impossible to say before feeding the data. If the CNN actually classified TED talks and audio books instead of actually classifying Planned speech vs Read speech, then the task was not solved. We had to be cautious in that manner. Perhaps a CNN was not helpful in this case.

2.6.2 Reasons for CNN failure

The reason why CNN failed to classify audio was that different people have a different way of saying specific words. One person saying 'um' will be different from another person saying the same word. The main feature for classifying speech as Read vs Planned was the pauses and the number of filler words ('um' and 'uh') in the speech. Hence, CNN was not able to pick (or learn) this because of the variations of accents.

2.7 Task 2: Pause Feature Approach

Anyhow, the results were not satisfactory as the desired accuracy was not achieved. So another approach was to be applied, hence instead of looking at the raw audio files the extracted features from those files can be helpful, the idea of building a Binary classifier from pause information seems too helpful. Using CNN seemed to be complex and it takes more time to train and plus it is prone to over-fit on. A pause classifier was theoretically much simpler to implement. As we had learned, through one of the papers, the role of pauses could play a huge impact in classifying the two categories of speeches, Planned Speech and Read Speech. But before thinking of implementing a Binary Classifier, there was another hurdle. Even though the TED talks data (Planned Speech) and the audio-books data (Read Speech) were there, their JSON equivalents to extract the desired pause-gaps were still missing, and it was needed for training.

The two groups of labelled *.wav* files had to be converted to JSON data and for that, we had more options. One option was to look up an open-source library performing speech recognition. However, the data given by GWEEK that was already annotated, and it was out of the question to use to different speech recognition software for training and testing data as the GWEEK API gave a much better representation of audio in a JSON file. Additionally, GWEEK's JSON format was very practical for this task as it will be mentioned in the next 2 paragraphs. The GWEEK team provided us with their speech recognition API so that file uploading to their server became possible. We had 9000 audio files with the label of Planned or Read and the GWEEK API is to be used for getting the respective JSON files which contain the features extracted from them. Since the API is on the production the files had to be sent with a 2-minute gap on weekends and 5 minutes gap on working days between each Upload and Pick-up in order to prevent the GWEEK server from crashing and clogging with API requests. We spent a lot of time resolving authentication issues and getting the required access.

Once initiation of the file uploads took place, we could access the JSON outputs from GWEEK API. Once the JSON responses started coming in, they were written to files and saved in to the two groups of Planned Speech and Read Speech respectively.

Due to the 5-minute gap restriction between each request on the working day and 2 minutes gap restriction between each request on weekends we were able to get the respective JSON files (features) for 804 files (TED talk and audio books combined).

In the JSON files, the frequency and duration of Pauses based on the tag, 'SIL', were extracted. 'SIL' marks a long silent pause in the recording. However, in order to achieve high accuracy of classification, using pauses, we assumed we had to extract a few more features so that our algorithm could be trained and efficiently distinguish Planned Speech from Read Speech. It was acknowledged from [6] that filled pauses were significant. From the data exploration early on it was an easy task to observe filler words. GWEEK's speech recognition recognises "uh" and "um" when a person is guilty of a hesitation in the recording.

By incorporating filler word features, similar to the existing SIL-features, it was thought that the algorithm would show a higher accuracy in determining the two categories of speeches. It looked promising at first, due to the frequent use of filled pauses by few TED talkers. However, while listening to example files the team realised that SIL(s) in the TED data also differ from the audio books. A person reading from an audio-book would tend to have proper pauses, with much more emphasis on the proper usage of punctuation marks. This was not the case for most of the TED speakers, as pauses occurred irregularly and were often filled with applause and laughter from the audience. Combining all features of silent pauses and filled pauses would hopefully play an important role in distinguishing speech types.

By writing a python script, the frequency and duration of the two filler words along with pauses were extracted, after rejecting the outliers (JSON files with GWEEK score less than 5). Once the pause features were retrieved and structured in a data frame, they were fed into a classification algorithm, similar to what we did in the first task. Using gradient boosting was thought of as a good starting point, due to our awareness of the implementation, and other researchers' success with the same approach[6]. In case of sparsity in the input data, the SVD pre-processing step[10] was also applied before giving it as an input to algorithms, for possibly improving the results. Since the libraries for Naive Bayes, Random Forest and Logistic Regression were already imported for task 1, they were also used in task 2. What then followed was left was a comparison with the CNN in terms of interpretability, results and computational cost. Finally, a very important part of the project was

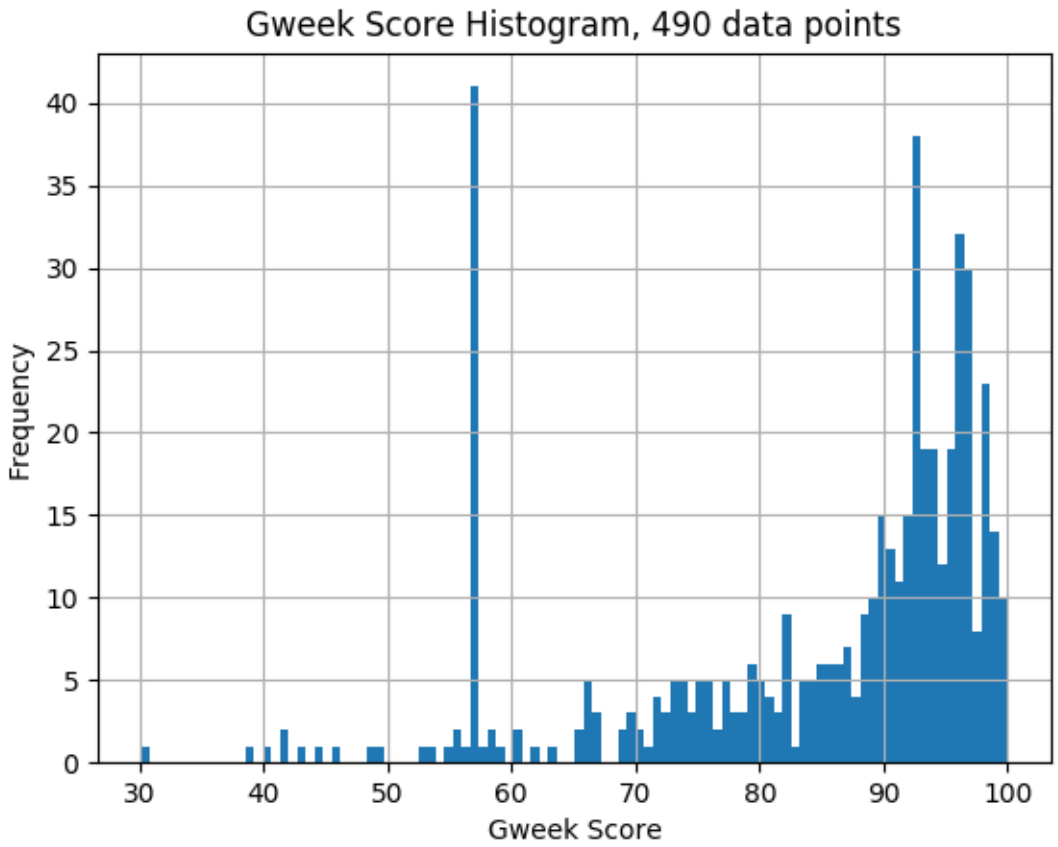


Figure 2: A Histogram of a Subset of Data

to give an estimate of approximately what fraction of GWEEK users that has actually Read material and uploaded it. This was possible after training.

2.8 Task 2: Principal Component Analysis

To visualise the external data (from TED and audio books) and the GWEEK data we used Principal Component Analysis to analyse that if the features we have selected differentiates between the Read and Planned speech. Principal Component Analysis uses eigenvectors to reduce to dimensions of the features while keeping the variance in the data.

3 Results

3.1 Task 1, subset

Some data exploration results are presented in the following subsection. A subset of 492 JSON files provided by GWEEK was used. The histogram shows how the majority of the users have scored 80+. Oddly enough, the most frequent score with this representation was roughly 57. Such an uneven distribution made it difficult to predict.

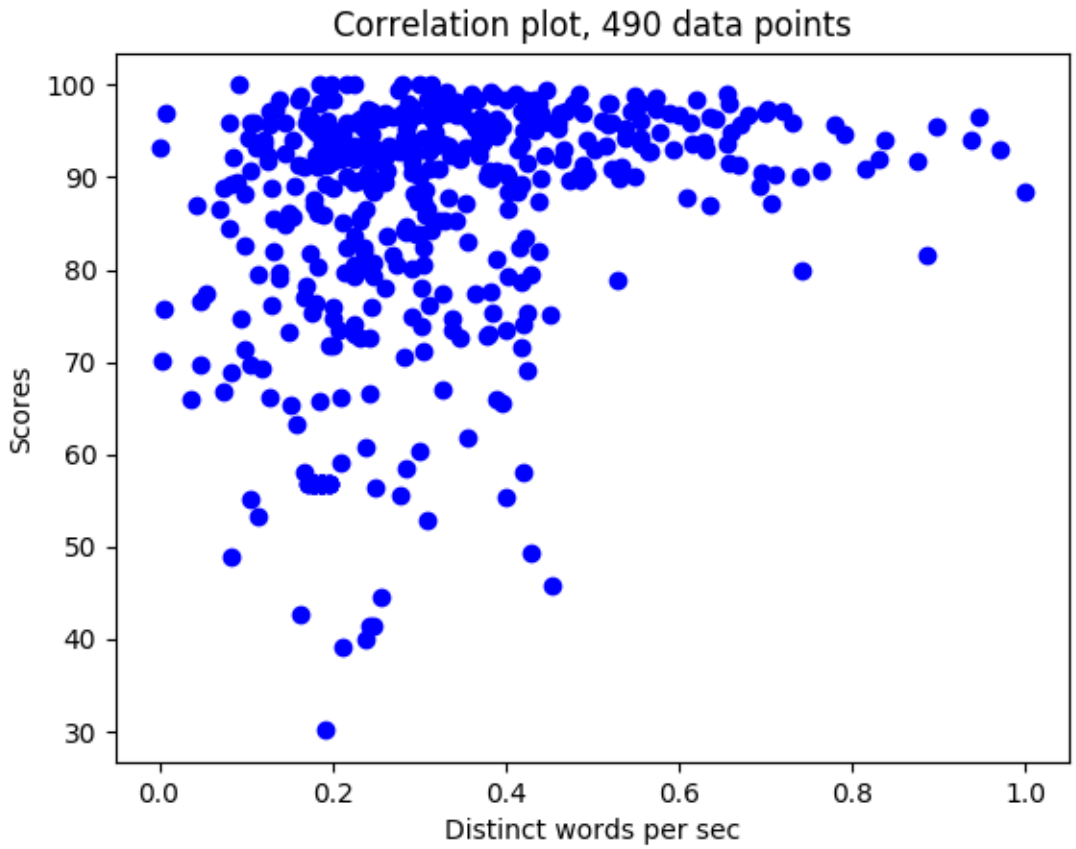


Figure 3: Correlation Plot 1

The graph shows the correlation between Feature1 and GWEEK score. It seems to be a positive correlation in this case. However, it is not enough to make a conclusion about the behaviour of GWEEK score. This is why the decision was made to include more variables.

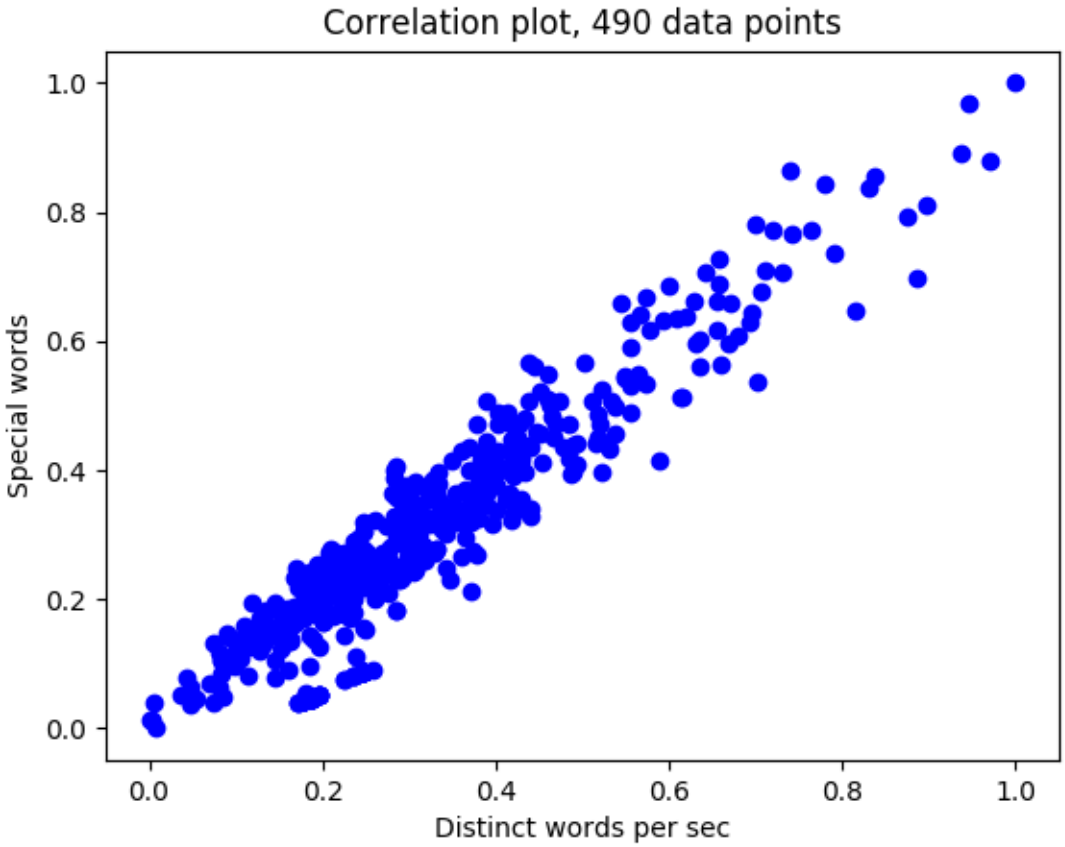


Figure 4: Correlation Plot 2

The above plot describes the addition of such a variable. Unfortunately, the graph shows the massive dependency between Feature1 and Feature3. These variables were assumed to be independent in a regression model.

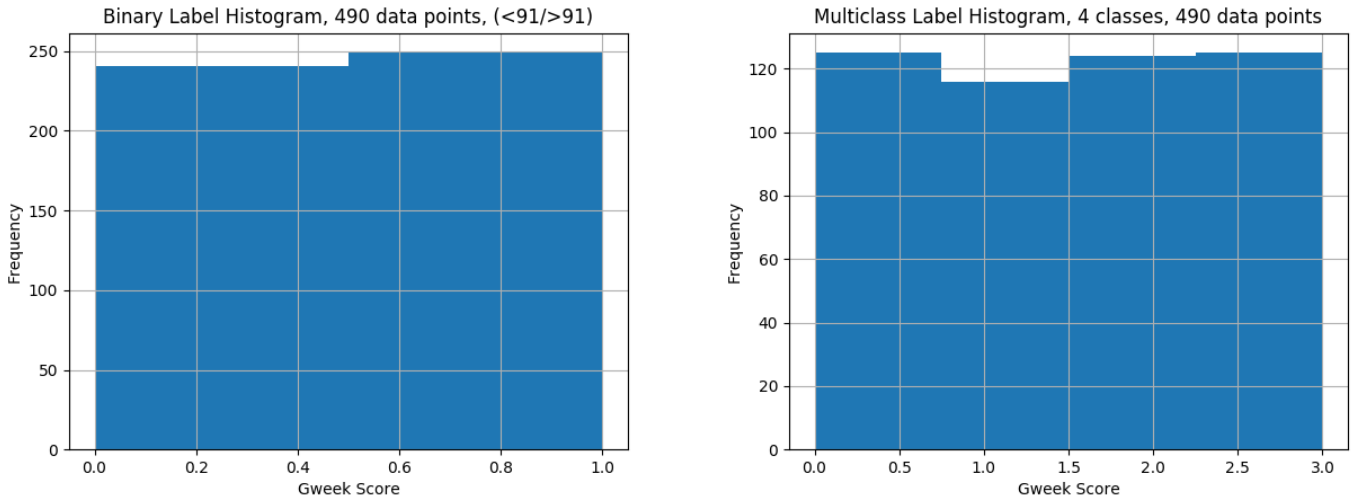


Figure 5: Binary and Multiclass Histograms

As mentioned, the histogram on the left has been used to test out different thresholds for classifying GWEEK score, when training on more data. The histogram on the right shows the same data points assigned with 4 labels, 0-3. Balanced data is essential for making conclusions about any predictive classifier. The thresholds that were used were the following:

1. 0 to 79
2. Greater than 79 to 91
3. Greater than 91 to 95.5
4. Greater than 95.5 to 100

Once the thresholds were set, we could classify all users using a larger data set.

3.2 Task 1, Large sample

Next, a data set of 3230 files were used to compute accuracy, F1-score and RMSE using 5 different algorithms; Logistic Regression, Random Forest, Naive Bayes, Gradient Boost and Linear Regression. The results are given in the next 3 tables.

Problem Class	Algorithm	Feature1-Feature3 Accuracy/F1	TF-IDF Accuracy/F1	TF-IDF, SVD Accuracy/F1
Binary Classification	Logistic	0.704 / 0.661	0.684 / 0.715	0.534 / 0.642
	Random Forest	0.619 / 0.579	0.641 / 0.590	0.599 / 0.551
	Naive Bayes	0.669 / 0.567	0.596 / 0.590	0.550 / 0.088
	Gradient Boost	0.695 / 0.579	0.676 / 0.590	0.579 / 0.551

Table 1: Accuracy and F1 for Binary Classification

Problem Class	Algorithm	Feature1-Feature3 Accuracy/F1	TF-IDF Accuracy/F1	TF-IDF, SVD Accuracy/F1
Multiclass Classification	Logistic	0.379 / 0.483	0.443 / 0.442	0.344 / 0.444
	Random Forest	0.401/ 0.408	0.402 / 0.402	0.365 / 0.367
	Naive Bayes	0.398 / 0.472	0.356 / 0.356	0.293 / 0.402
	Gradient Boost	0.416 / 0.408	0.443 / 0.402	0.398 / 0.367

Table 2: Accuracy and F1 for Multiclass Classification

Problem Class	Algorithm RMSE	Feature1-Feature3 RMSE	TF-IDF RMSE	TF-IDF, SVD RMSE
Regression	Linear Regression	13.69	11.91	13.04

Table 3: RMSE

3.3 Task 2, CNN

All audio files were converted into images using MFCC[18] with a common dimension of 20 x 2000. Once creating an image for all 9300 data points they were given as input to the Convolutional neural network. Because of the complexity of the network and size of data the CNN took approximately 10 hours to pre-process and train. By choosing an 80-20% split, we obtained an accuracy of 65% accuracy on the validation set. The accuracy increased from 45% to 65% and then stayed like this for the rest of the run. This couldn't be considered as a feasible result as 65% of accuracy approximately implies predicting on random as we had binary data.

3.4 Task 2, Pause, 794 data points

Obviously, 65% was not satisfactory when there was a 50% chance of predicting at random. However, moving over to the pause approach resulted in improvement immediately. The program ran on 794 files, whereas 418 were TED talks. 92.4% of all filled pauses occurred in the TED talks. The plot shows an example of independent variables passed to the model.

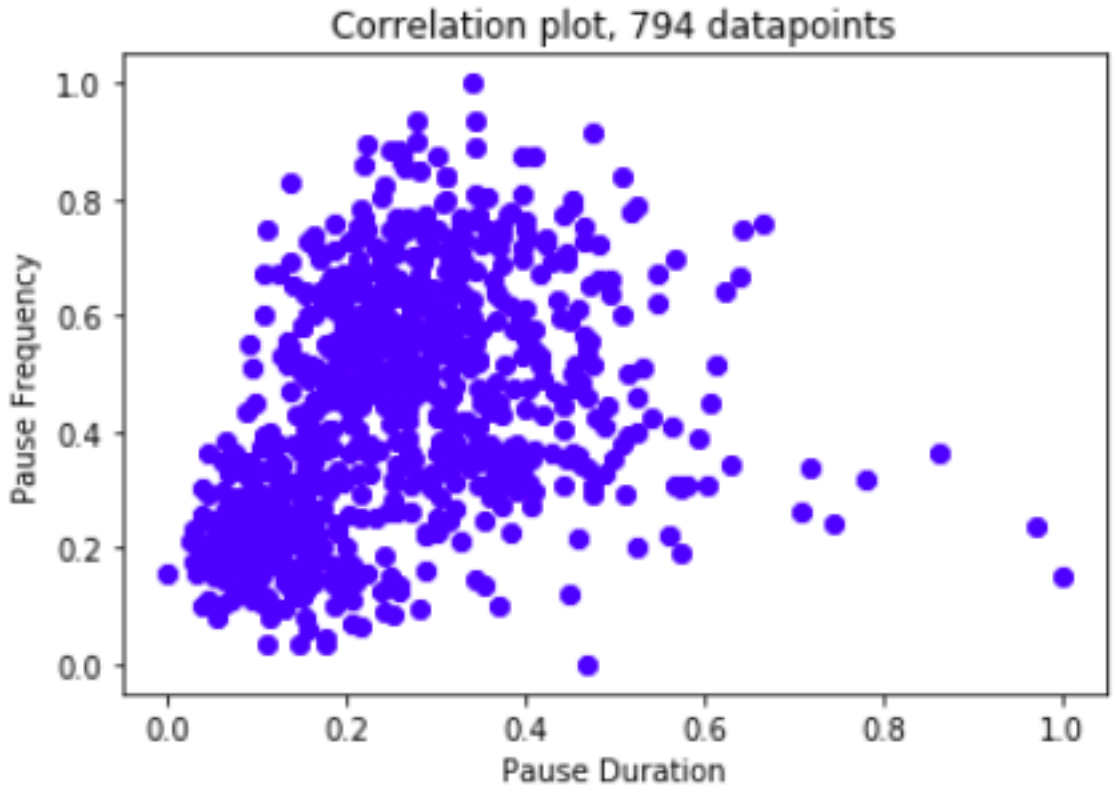


Figure 6: Correlation Plot 3

The results from training samples from TED talks and audio books are given in the table below.

Problem Class	Algorithm	Neural Network Accuracy	Pause Accuracy / F1	Pause, SVD Accuracy / F1
Binary Classification	CNN	0.65	-	-
	Logistic	-	0.78/0.79	0.78/0.79
	Random Forest	-	0.81/0.81	0.79/0.80
	Naive Bayes	-	0.71/0.64	0.78/0.76
	Gradient Boost	-	0.82/0.81	0.81/0.80

Table 4: Accuracy and F1-score for TED and Audio Books

The results make it relatively clear that pause features are more successful in terms of classifying TED talks and Audio-Books. It also states that Random Forest and Gradient Boost is preferable in this particular task with accuracy roughly around 80%. A large sample of 3230 files from GWEEK was tested on the given models. The testing produced the following:

- Logistic Regression predicts 36.8% of the sample as Read
- Random Forest predicts 50.1% of the sample as Read
- Naive Bayes predicts 3.3% of the sample as Read
- Gradient Boost predicts 56.4% of the sample as Read

Since in terms of accuracy the Random forest and Gradient Boost had the highest but we could infer that the tree-based algorithm has overfitted over here, this can be justified with the fact that even though the GWEEK dataset is unlabelled but we know for sure that very less amount of data points in the GWEEK data are Read speech. In this case, the Random Forest predicts almost 50% of the GWEEK data as Read speech which was not the true case, same was the case with Logistic Regression and Gradient Boosting.
However, Naive Bayes maybe have 2% less accuracy than Random Forest in the test set but Naive Bayes predicts the whole GWEEK in a more realistic way as compared to Tree Based classifiers and Logistic Regression. Hence it would be correct to say that Naive Bayes is the perfect algorithm for this classification tasks as it does not over-fits the training data.

3.5 Task 2: Principal Component Analysis

The results of Principal Component Analysis on the TED talks and the audio books data can be seen below.

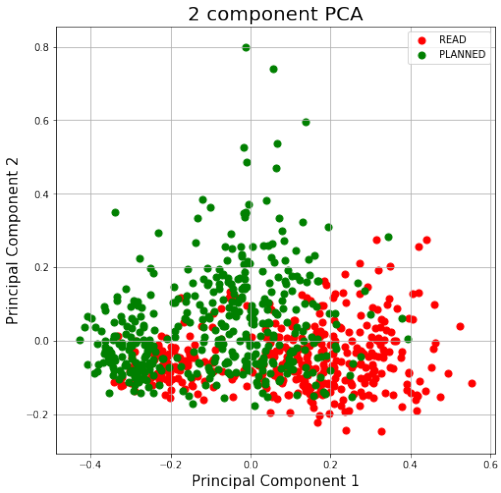
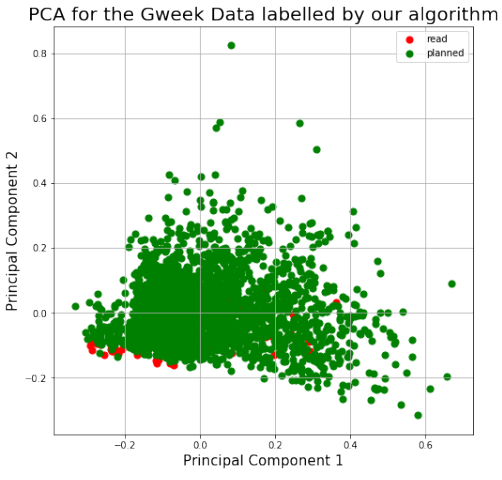


Figure 7: PCA on TED and AudioBooks

The results of Principal Component Analysis on the GWEEK provided data, which means that the GWEEK data was passed through Naive Bayes classifier which was trained on training data and the labels created by Naive Bayes Classifier were used to colour the points with respect to the predicted labels.

As we can easily observe from below image that the read speech as predicted by the model is on the bottom of the graph. Thus, it implies that most of the data that fall under Read speech will be on the bottom left of the PCA graph; hence it validates our algorithm.



[h]

Figure 8: PCA of Read Speech and Planned Speech

4 Discussion

4.1 Task 1

From the tables in section 4.2 comes the result that TF-IDF without SVD is the preferable approach of the 3. That can be seen from both the accuracy (0.684/0.443) and F1 score (0.715/0.442) for Logistic Regression as well as the RMSE (11.91). However, these are not satisfactory results. The vocabulary features can at max be inferred as slightly related to GWEEK scores, not completely related. This can be justified from the fact that by using different algorithms of machine learning, both probabilistic and Decision tree based, none were giving satisfactory results. In fact, many F1-scores were directly bad.

The hypothesis was that if the Machine learning algorithms could be trained on the vocabulary features and predict with good results then that infer that there is a relation between vocabulary and GWEEK scores. The algorithms couldn't predict very well, but that does not infer independence completely. The definition we had of vocabulary might not have been suitable. Additionally, since the metadata was not available about the topic of recording and the questions asked without recording, hence the perfect definition of vocabulary cannot be inferred directly from the audio.

Also, it is likely that the thresholds set in the data exploration didn't fully distribute the whole set, as the GWEEK scores given to us were unbalanced initially. These are factors that might lead to a drop in accuracy.

4.2 Task 2

As stated above, CNN didn't work because the algorithm was not able to recognise the filler words spoken by different people say, because of their different accents, different way of speaking and pronouncing words. The result from task 2 is a good example of how the increased complexity of a model does not necessarily lead to improved results. Pause features are definitely a good way to go when classifying types of speech. One might say that the people contributing to the audio are trained Readers and therefore rarely make mistakes. If they ever make one, it is likely they would start the recording over again. On the contrary, TED talkers are also performing very professional and formal. By choosing any other non-Read speech database, the chances of increased occurrences and duration of pauses in that data are relatively high. Still, the simple models predicted better than expected.
Logistic Regression, Naive Bayes, Random Forest Classifier and Gradient Boosting Algorithm were applied to the data. Similarly, all these algorithms were also applied after applying SVD (Singular Value Decomposition) on data.

- **Logistic vs Random Forest:**

  As the data-set only contains two labels, 'Read' and 'Planned', the instinct was to go for Logistic Regression first as it is a Binary Classification algorithm which tries to find a linear separation of classes for the given variables. We used Random Forest as it provides the best feature in the result. The implementation helped us in discovering important relation between features. It was a go-to method since it provides a good balance between precision and over-fitting. From the results, we can observe that the Accuracy and the F1-scores using both the algorithms on the test data were almost similar, with Random Forest having a slight edge over Logistic Regression. However, this does not necessarily mean that it's a better algorithm compared to Logistic Regression. This is because when tested on the unlabelled GWEEK data, Random Forest was giving out the accuracy of around 50%, which is far from the truth. On the other hand, Logistic Regression gives a better prediction for the unlabelled GWEEK data.

- **Naive Bayes vs Logistic Regression:**

  Both Naive Bayes and Logistic Regression gave 78% accuracy approximately, however, the F1-score for Logistic Regression was much better than that of Naive Bayes. But, interestingly, it was observed that Naive Bayes predicted that only 3.3% of the unlabelled GWEEK data was Read which is extremely close to what was communicated to us by the GWEEK team. Naive Bayes bases its assumptions on the fact that the features are conditionally independent. Nevertheless, actual data-sets are never perfectly independent, but they can be pretty close. In short, Naive Bayes has a higher bias but lower variance compared to logistic regression. Even though both Naive Bayes and Logistic regression are linear classifiers, Logistic Regression makes a prediction for the probability using a direct functional form whereas, Naive Bayes, given the results, figures out how the data was generated[19].

In reality, people are likely to present information very differently from the training recordings, whether they are reading or talking somewhat freely. For this reason, and the fact the data from GWEEK is unlabelled and remarkably unbalanced, Naive Bayes predicted the expected results on the GWEEK data. Hence, it is legitimate to imply that the Naive Bayes algorithm learned the underlying function to classify the audio as Read vs Planned speech for the GWEEK data using the external TED Talks and audio books data.

5 Conclusion

The vocabulary based features chosen were unable to predict GWEEK scores with good Accuracy and F1-score. With a different definition of vocabulary and more balanced data, the outcome might have been better (or worse). However, the team concludes a slight relation as Logistic Regression predicts with roughly 70% accuracy and F1-score. The classification of Read and Planned speech gave good accuracy and it is Ready to be put on production. For audio classification, the simpler approach tends to work better than using the complex Convolutional Neural Networks as complex model failed to pick the underlying pattern of silence words which can be easily picked from the JSON file for each audio file provided by the GWEEK through their propriety API.

6 Future Work

Real-time audio analysis can be done with the help of the JSON files that are created by the GWEEK API. It is computationally feasible to analyse the text files instead of audio files, and an engine can be created to do the analysis on the audio files which will be helpful in analysing the user behaviour. Secondly, real-time prediction of audio to be Planned or Read speech can be given to user based on speech, For example, if the user is reading from a book the application will automatically identify it and prompt the user to speak spontaneously instead of reading from a source. Thirdly, splitting the data into topics based on what the users are asked to speak on, can help to build a more concentrated definition of vocabulary. Perhaps, letting users choose their own topic will help distinguishing between who has a good/bad vocabulary and then compare it with GWEEK score.

References

[1] Juang, B.-H.; Rabiner, L.R. (2006), "Speech Recognition, Automatic: History", Encyclopedia of Language Linguistics, Elsevier, pp. 806–819, doi:10.1016/b0-08-044854-2/00906-8, ISBN 9780080448541

[2] Gweek. (2019). Gweek – Improve Communication Skills With Speech Training. [online] Available at: https://www.gweekspeech.com/ [Accessed 2 May 2019].

[3] Jiang, Haihua, et al. "Investigation of different speech types and emotions for detecting depression using different classifiers." Speech Communication 90 (2017): 39-46.

[4] Le, Duc, et al. "Automatic assessment of speech intelligibility for individuals with aphasia." IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP) 24.11 (2016): 2187-2199.

[5] Piczak, Karol J. "Environmental sound classification with convolutional neural networks." 2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP). IEEE, 2015.nternational Workshop on Machine Learning for Signal Processing (MLSP). IEEE, 2015.

[6] Igras-Cybulska, Magdalena, et al. "Structure of pauses in speech in the context of speaker verification and classification of speech type." EURASIP Journal on Audio, Speech, and Music Processing 2016.1 (2016): 18.

[7] Oliveira, Miguel. "The role of pause occurrence and pause duration in the signaling of narrative structure." International Conference for Natural Language Processing in Portugal. Springer, Berlin, Heidelberg, 2002.

[8] Kovac, M. (201). Speech Errors in English as Foreign Language: A Case Study of Engineering Students in Croatia. [online] Bib.irb.hr. Available at: https://bib.irb.hr/prikazi-rad?rad=976961 [Accessed 8 May 2019].

[9] "sklearn.feature_extraction.text.TfidfVectorizer — scikit-learn 0.20.3 documentation", Scikit-learn.org, 2018. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html. [Accessed: 09- May- 2019].

[10] Scikit-learn.org. (2019). sklearn.decomposition.TruncatedSVD — scikit-learn 0.21.0 documentation. [online] Available at: https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.TruncatedSVD.html [Accessed 13 May 2019].

[11] Avinash Navlani (2018) Understanding Logistic Regression in Python, Available at: https://www.datacamp.com/community/tutorials/understanding-logistic-regression-python (Accessed: 9th May 2019).

[12] Avinash Navlani (2018) Naive Bayes Classification using Scikit-learn, Available at: https://www.datacamp.com/community/tutorials/naive-bayes-scikit-learn (Accessed: 9th May 2019).

[13] Avinash Navlani (2018) Understanding Random Forests Classifiers in Python, Available at: https://www.datacamp.com/community/tutorials/random-forests-classifier-python (Accessed: 9th May 2019).

[14] J. Brownlee, "A Gentle Introduction to the Gradient Boosting Algorithm for Machine Learning", Machine Learning Mastery, 2019. [Online]. Available: https://machinelearningmastery.com/gentle-introduction-gradient-boosting-algorithm-machine-learning/. [Accessed: 09- May- 2019].

[15] Abolfazl Ravanshad (2018) Gradient Boosting vs Random Forest, Available at: https://medium.com/@aravanshad/gradient-boosting-versus-random-forest-cfa3fa8f0d80 (Accessed: 9th May 2019).

[16] "sklearn.metrics.f1_score — scikit-learn 0.20.3 documentation", Scikit-learn.org, 2018. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html. [Accessed: 09- May- 2019].

[17] Arden Dertat (2018) Applied Deep Learning - Part 4: Convolutional Neural Networks, Available at: https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2 (Accessed: 9th May 2019).

[18] Practicalcryptography.com. (2019). Practical Cryptography. [online] Available at: http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/ [Accessed 12 May 2019].

[19] [3]"Naive Bayes vs Logistic Regression", Medium, 2019. [Online]. Available: https://medium.com/@sangha_deb/naive-bayes-vs-logistic-regression-a319b07a5d4c (Accessed: 9th May 2019).

[20] Sas.com, 2019. [Online]. Available: https://www.sas.com/content/dam/sas/support/en/sas-global-forum-proceedings/2018/1857-2018.pdf.

[21] Openslr.org. (2019). openslr.org. [online] Available at: http://www.openslr.org/51/ [Accessed 13 May 2019].

[22] Openslr.org. (2019). openslr.org. [online] Available at: http://www.openslr.org/12/ [Accessed 13 May 2019].