Risk Prediction Sentiment Analysis
CS 4624 Multimedia, Hypertext, and Information Access
Final Report
May 14, 2021

Virginia Tech, Blacksburg VA 24061
Instructor: Edward A. Fox
Client: Christian Johnson
Team: Jhanavi Ghadia, Sai Gurrapu, Tarun Singh, Sai Krishna Yeshala
Table of Tables

Table 1: Fields included in the worker survey - provided by the client 10
Table 2: Data Set I provided by the client 10
Table 3: Data Set II provided by the client 11

Table of Figures

Figure 1: Speech to Text Flow Diagram 6
Figure 2: Data Cleaning Process 8
Figure 3: Project Services Flowchart 9
Figure 4: Database schema for the web application 12
Figure 5: Microphone Access Code 13
Figure 6: Google Web Speech API Usage 13
Figure 7: NLTK sentiment analysis example 14
Figure 8: Example code for sentiment analysis with NLTK 15
Figure 9: Example code for the classifier models with Sci-Kit 16
Figure 10: Example of part of speech tagged sentence structure 17
Figure 11: Parts of speech code 17
Figure 12: Django Application Flowchart 18
Figure 13: PostgreSQL for database system 18
Figure 14: Speech to text testing 19
Figure 15: Sentiment analysis and upload feature testing 20
Figure 16: Workflows and their corresponding system solutions 22
Figure 17: Workflows and their corresponding system solutions. 22
Figure 18: Web Application - About Page 23
Figure 19: Web Application - Upload Report 23
Figure 20: Web Application - Questionnaire Survey 24
Figure 21: Web Application - Success Screen 24
Figure 22: Web Application - Survey Results 25
Figure 23: User Interface - Administrator Home Page 26
Figure 24: User Interface - Upload report 27
Figure 25: User Interface - Sentiment Analysis results 27
Figure 26: User Interface - Employee home page 28
Figure 27: User Interface - Employee survey page 28
1. Executive Summary

The risk sentiment analysis tool for workers and workplaces aims to help judge and find out the safety culture and risk levels of a workplace. Our program takes the narrative reports of the safety and risk conditions from the employees, passes that on to our sentiment analysis program, and returns the sentiment values (positive/negative/neutral/mixed) to the users. These values can be referred to by the workplace owners or other employees so they have an estimate of the safety conditions at a particular place. Our goal is to accumulate all of the information provided by the concerned/satisfied employees, apply proper sentiment analysis to that, and have a reliable output for reference.

Our client, Christian Johnson, has provided us with raw data which consists of employee information, their area of concern, severity of their concern, date an incident occurred, exact location of the incident, and their explanatory description. Our deliverable includes a functioning API running through a web interface that allows the users to upload the data, process it, and obtain the sentiment values which will give them a clear idea about the situation. They will also be able to input an audio clip; our speech to text conversion algorithm will convert their words into a text input and output the results. Most of our project work involves data cleaning (getting rid of irrelevant entries and cleaning the data accordingly), sentiment analysis using the Python Sentiment Analysis API, developing a web interface (which will provide a platform to the users to upload the narrative reports or individual entries), and showing the output results.
2. Introduction

2.1 Objective
For any individual trying to seek employment and looking at various workplaces, it is very important for them to be aware of the risk conditions and safety levels at the place they are thinking about. It might be difficult for them to find current or past employees to get their feedback and suggestions. Through our project, we can take in the detailed narrative reports as well as individual feedback entries, and after performing sentiment analysis on that, enable them to see the output. The output consists of sentiment values and will be able to ensure a better understanding of the safety levels. We also provide the flexibility to our users to provide input in the form of audio, so our speech to text conversion algorithm will convert the words in the audio to text and serve that as an input.

2.2 Client
Our client, Christian Johnson, is Safety Health & Environmental (SHE) Global Director at AstraZeneca in Maryland. He serves interests in AI/ML sentiment analysis approaches for assessing risk and pharma & biopharma industry SHE challenges. This project serves as a proof of concept for his work interests.

2.3 Project Background
For every individual, it is very important to be completely aware of the workplace situation of a company before joining. But there’s not any reliable source where the risk concerns and safety issues are properly addressed and provide clear ideas regarding past incidents and the solutions to the employee’s concerns. This project is able to take in detailed narrative information and descriptions of the employee’s feedback. For the scope of our project, we focused on the data set, provided by our client, which consists of the details of the injury, days between hire and the injury, several descriptive categories of the injuries/accident, and the comment from the user. The idea behind this project was from our client, Christian Johnson. As a team, we brainstormed different ideas about various tools and methodologies, and came up with the project plan.

2.4 Organization of Report
Our report starts with the summary and the background of the project. Following that we have the objectives and client information and project background, which show the motivation of the project. In Section 3, we have detailed information about speech to text conversion, sentiment analysis, user interface, and data cleaning processes. In Section 4 we discuss the design and the details of our web interface, which is the main deliverable for our project. Next we have an in-depth description of the speech to text implementation, different tools used for web interface and sentiment analysis such as the VADER tool, and Clarabridge. Following the implementation is a brief user manual on how to set up and use the application. Lastly, we conclude with lessons learned and challenges encountered.
3. Requirements

The project had crucial components that were required to be integrated. The following components were requirements for the Risk Prediction Sentiment project: Speech to Text, User Interface, Sentiment Analysis, and Data Cleaning.

3.1 Speech to Text

It is often easier to naturally speak instead of typing text. The Speech to Text service makes it effortless for workers to send their workplace incidents and experiences onto the web application. As shown in Figure 1, the speech of the workers will be automatically recognized and after natural processing of the speech in real-time, the output will be the transcription of that speech. The workers must enable their microphone on their device. We support smartphones, personal computers, or any other device that has a microphone. Since we’re accessing the device’s hardware through our web application, a security notification will pop up that workers must accept. By accepting the notification, they allow our application to access and record the data from the microphone. In real-time, our application service will record the microphone data and convert the spoken words into text. Essentially, our input for the speech-to-text service is audio data and the output is an accurate transcription of the audio. Once we have the transcription it is available inside our web application for many use cases. We can perform data analysis, sentiment analysis, visualization, etc.

Figure 1: Speech to Text Flow Diagram. Adapted from [10].

3.2 User Interface

The user interface is the core requirement of this tool for the workers and the managers of a company to be able to interact with the sentiment analysis backend by providing input data from the worker survey. The interface has been carefully designed in collaboration with the client to meet the requirements of the workers and the managers using the tool. The two branches
involved in this tool address the two separate workflows we designed for the workers and the managers. The manager’s branch of the application involves a results page to illustrate the sentiment analysis results for them to be able to better judge the workplace safety conditions. The worker’s branch does not involve a results page but provides them with an easy-to-use interface to record their feedback on the safety conditions in the form of a survey or an audio recording. Their responses are ultimately stored in the database for long-term analysis.

3.3 Sentiment Analysis

Sentiment Analysis is a natural language processing technique typically used to determine whether data is positive, negative, or neutral. It is often used by businesses to detect sentiment in social data and to understand customers [2]. This allows businesses to identify and assess critical issues in real-time. In our case, we are using sentiment analysis to assess the risk associated with reports in businesses. The main requirement for our application is assessing the sentiment of these business reports and identifying risks that are present in any of these institutions. The data will be entered into some sort of Excel format, and then be processed and assessed for its sentiment. This will then be displayed to the manager or whoever is running the scans for sentiment analysis.

There are many different types of sentiment analysis techniques, but for our purposes, we are using an automatic approach to sentiment analysis. This means our system doesn’t rely on manually crafted rules, but we model it instead as a classification problem where we implement a machine learning classifier. In the training process, our model learns to associate a particular input to a corresponding output based on the test samples used for training. In the prediction process, the feature extractor is used to transform unseen text inputs into feature vectors. The feature vectors are then fed into the model which generates predicted tags such as a percentage of positive, negative, or neutral.

3.4 Data Cleaning

We have the raw data of the narrative description of the user’s entries which consists of the date of hire, date the incident happened, type of hazard, activities performed, level of risk, damages, etc. Figure 2 explains how the raw data will be cleaned using machine learning algorithms and we can get the most accurate data for sentiment analysis. For our data cleaning part, we made sure that there aren’t any irrelevant entries, which can create problems with our sentiment analysis part.
Figure 2: Data Cleaning Process. Adapted from [3].
4. Design

4.1 Overview

The client played a major role in helping us with the general design of the project. We have worked with him extensively to come up with the workflows. The image in Figure 3 reflects the overall workflow of the project. The following subsections illustrate the datasets provided by the client for us to work with and perform sentiment analysis on. We used these datasets to design the database schema for the web application for long-term storage of the worker feedback responses.

![Project Services Flowchart](image)

Figure 3: Project Services Flowchart

4.2 Data Set

<table>
<thead>
<tr>
<th>Field Title</th>
<th>Input</th>
<th>Variable Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anonymous Report</td>
<td></td>
<td>Single choice</td>
</tr>
<tr>
<td>Name</td>
<td>Speech to text or text-only entry</td>
<td>Open text</td>
</tr>
<tr>
<td>Site Name</td>
<td></td>
<td>Single choice</td>
</tr>
<tr>
<td>Submit Date</td>
<td>Organization</td>
<td>Site</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------------</td>
<td>---------</td>
</tr>
<tr>
<td>3-Jul-19</td>
<td>Global Operations</td>
<td>Site D</td>
</tr>
<tr>
<td>18-Sep-18</td>
<td>Global Operations</td>
<td>Site B</td>
</tr>
<tr>
<td>4-Oct-18</td>
<td>Global Operations</td>
<td>Site B</td>
</tr>
<tr>
<td>4-Oct-18</td>
<td>Global Operations</td>
<td>Site B</td>
</tr>
<tr>
<td>4-Oct-18</td>
<td>Global Operations</td>
<td>Site B</td>
</tr>
<tr>
<td>16-Oct-18</td>
<td>Global Operations</td>
<td>Site B</td>
</tr>
<tr>
<td>22-Oct-18</td>
<td>Global Operations</td>
<td>Site C</td>
</tr>
</tbody>
</table>

Table 1: Fields included in the worker survey - provided by the client

Table 2: Data Set I provided by the client. Activities Observed column blurred to preserve privacy.
<table>
<thead>
<tr>
<th>Feedback Given</th>
<th>At-Risk Behavior or Condition</th>
<th>Barriers</th>
<th>Escalated to Concern Reporting</th>
<th>Form Name</th>
<th>Action Item</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>No</td>
<td>No</td>
<td>General</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Asked Wetmill/Feedhouse team if machine guarding audits have been completed. Not aware of a recent audit.</td>
<td>Yes</td>
<td>• Procedures/Instructions/ Work Standards</td>
<td>No</td>
<td>Work Task</td>
<td>121 (Closed)</td>
</tr>
<tr>
<td>Ttalking to employees about their immediate workspace and the ergonomic impact of monitor position.</td>
<td>Yes</td>
<td>• Complacency</td>
<td>No</td>
<td>General</td>
<td>None</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>• Job Planning</td>
<td>No</td>
<td>General</td>
<td>None</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>• Engineering</td>
<td>No</td>
<td>General</td>
<td>None</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>• Inspection/Auditing</td>
<td>No</td>
<td>Work Task</td>
<td>None</td>
</tr>
<tr>
<td>Great job and clean up</td>
<td>Yes</td>
<td>• Job Planning</td>
<td>No</td>
<td>Work Task</td>
<td>None</td>
</tr>
</tbody>
</table>

Table: Three Data Set II provided by the client

The form of the data provided by the client is shown in Table 1. The actual data we worked on is shown in Table 2 and Table 3. It consists of separate columns for the user’s information, the incident information, and the user’s feedback about the incident. It also shows if the process/investigation about the incident is still ongoing or not, since based on that, the sentiment analysis reports can change.

4.3 Data Storage

We are using the PostgreSQL database management system to store the data collected through the survey responses recorded by the workers [9]. As the design suggests, the application will store the survey responses from the workers, and additional data provided by the managers through the CSV files. As per the dataset and the survey fields provided by the client, we came up with a basic database schema in our first iteration to use for this application to store the required fields for long-term sentiment analysis; see Figure 4.
Figure 4: Database schema for the web application
5. Implementation

5.1 Speech to Text Implementation

Our implementation for this service leverages Google’s Web Speech API framework for high fidelity output [4]. It is a simple and easy-to-use Javascript API that we can use on web applications to enable speech recognition and synthesis. The speech recognition feature converts speech into text transcription and the speech synthesis enables the user to control and interact with the on-screen web components. The important aspect of using Google’s Web Speech API is that it allows continuous dictation which means in real-time the user can see their speech transcription as they are speaking. Although this comes with 3-4 seconds of delay, this is negligible for the application performance. The Google Web Speech API only performs the Natural Language Processing aspect of the Speech to Text workflow. To pass our audio or microphone data into the Web Speech API and get the transcription output, our implementation requires the use of a Python wrapper library called SpeechRecognition [10]. This enables us to easily transmit our microphone audio data into the API without writing any driver-level or system code to access the microphone.

The Python code in Figure 5 demonstrates how we can access the microphone and capture the audio. It also listens to the ambient noise and makes sure that our speech recognition is accurate and only picks up the intended spoken words.

```
Python
>>> import speech_recognition as sr

>>> r = sr.Recognizer()
>>> m = sr.Microphone()
>>> recognize_speech_from_mic(r, m)
{'success': True, 'error': None, 'transcription': 'hello'}
>>> # Your output will vary depending on what you say
```

Figure 5: Microphone Access Code

We instantiate our Google Web Speech API using the code in Figure 6. Once the API is ready our microphone audio is transmitted to the API and our transcription is generated in real-time as the worker is speaking.

```
Python
>>> r.recognize_google(audio)
'hello'
```

Figure 6: Google Web Speech API Usage
5.2 Sentiment Analysis Implementation

In our initial implementation of the project, we used the pre-trained Python sentiment analysis libraries rather than training our model, for simplicity’s sake [5]. The library we decided to was is the Python NLTK library which is the natural language toolkit that Python provides already. Finally, we decided to use a rule-based approach to the sentiment analysis by applying different NLP techniques including, tokenization, part of speech tagging, and parsing.

5.2.1 VADER Tool

NLTK already has a pre-trained sentiment analyzer called VADER (Valence Aware Dictionary and sEntiment Reasoner) [5]. The initial implementation of VADER can be seen in Figure 7 where we can see the initial implementation of the sentiment polarity scores. Some issues are evident, however, when using a pre-trained sentiment analysis model. VADER is best suited for the language used in social media such as short sentences with some slang and abbreviations. It is less accurate when rating longer, structured sentences but provides us with a good launching point to assess the data we are working with. In Figure 7 we can see a basic implementation of using the pre-trained VADER model.

![Figure 7: NLTK sentiment analysis example](image)

After using the initial features provided by VADER, we can create a more accurate model by modifying the code to ignore words or values we don’t want to assess or giving more weight to certain characteristics based on the data we are given. By implementing the code snippet in Figure 8, we can ignore any unwanted words or stop words and get more accurate sentiment results for our desired data. The code will remove any unwanted words that have been tagged.
Our next step in analyzing the sentiment is implementing our classifier or using a sentiment analysis tool that is more suited for customer or risk prediction data. Scikit-learn is a free software machine learning library for the Python programming language that we used. It features various classification, regression, and clustering techniques that can be implemented in our sentiment analysis solution. This allows us to run the different classifiers that Scikit-learn provides and see the various accuracies of each classifier, which can be seen in Figure 9.
5.2.3 Sentiment Analysis Structure

Before writing code, it is important to determine the structure of the sentences we will be taking in as input. This is a key decision as it has an impact on what algorithms we will be using in our sentiment analysis. For our project, we decided to go with a structure where a text is defined as a list of sentences, a sentence is a list of tokens, and each token is a tuple of three elements, a word form (the exact word in the text), a word lemma (a generalized version of the word), and a list of associated tags. This allows for us to have a sentence structure that is ready for advanced sentiment processing without being too complex. After we determine the structure of the sentiment analysis, we can tag the words in the sentence with parts of speech tags and categorize the words while removing any unnecessary words as is shown in Figure 10.
The NLTK library provides functionality for recognizing parts of speech so by using the functions from the NLTK library we can tag the sentences. In Figure 11 the code for the parts of speech tagging is shown where each token has a form, a lemma, and a list of tags.

```python
class POSTagger(object):
    def __init__(self):
        pass

    def pos_tag(self, sentences):
        input format: list of lists of words
        e.g.: [['this', 'is', 'a', 'sentence'], ['this', 'is', 'another', 'one']]
        output format: list of lists of tagged tokens. Each tagged token has a
        form, a lemma, and a list of tags
        e.g.: [[('this', 'this', ['DT']), ('is', 'be', ['VB']), ('a', 'a', ['DT']), ('sentence', 'sentence', ['NN'])],
               [('this', 'this', ['DT']), ('is', 'be', ['VB']), ('another', 'another', ['DT']), ('one', 'one', ['CD'])]]

        pos = [nltk.pos_tag(sentence) for sentence in sentences]
        # adapt format
        pos = [[[word, word, [postag]] for (word, postag) in sentence] for sentence in pos]
        return pos
```

Figure 11. Parts of speech code
5.3 Django web application

Django is a Python-based open-source web framework to build full-stack web applications. Our decision to choose this application has been based on some of the advantages of Django such as an excellent online community to resolve issues during app development, and the scalability provided by this framework [7]. Moreover, the speech-to-text conversion and the sentiment analysis aspects of the project are being implemented in Python. Django follows a Model-View-Template architecture as shown in Figure 12. The database models are defined to meet the requirements of the data being stored and the logical connections between the models. The views and the templates related to the front end of the application where the web pages are rendered based on the URLs that the users are directed to.

![Django Application Flowchart](image)

*Figure 12: Django Application Flowchart. Adapted from [8].*

5.4 PostgreSQL

PostgreSQL is a powerful, open-source relational database system that has proven to be successful in handling heavy data workloads. We have decided to use this system to store the information from the data that is being recorded by user surveys and audio recordings. Django is compatible with PostgreSQL and provides several data types that work with it well [9].

![PostgreSQL](image)

*Figure 13: PostgreSQL for database system [9]*
6. Testing

One of the major tasks in this project was to ensure that the various individual components involved are integrated into the web application and are working as expected.

Testing the speech to text component:

![Risk Prediction Sentiment Analysis](image)

*Figure 14: Speech to text testing*

The red dot on the chrome tab in the screen dump in Figure 14 indicates that the browser is successfully accessing the microphone input. The input field right above the submit button correctly indicates the intended feedback spoken through the microphone.
Testing the upload report functionality and sentiment analysis code:

Figure 15: Sentiment analysis and upload feature testing

The report uploaded to the upload page as shown in Figure 15 has been successfully recognized on the system and the sentiment analysis is being performed on the section which consists of the feedback provided by the users.
7. Developer’s Manual

This section provides a detailed description of how we broke down our problem into multiple tasks and subtasks to integrate all the components and develop the web application. The details for accessing the code base and installing the application are not provided in this section as a result of a collective decision taken by our team and the client to upload the codebase to a private repository.

7.1 Methodology

To support the goal of examining the feedback provided by a worker at a workplace of any sort, the system needs to support the tasks of:

1) Allowing the users to provide input data to perform the sentiment analysis.
   - This task is divided into two different paths with one applying to the input from the managers at the company and the other to the workers at the company.
   - For the first path, the system needs to allow the manager to upload a CSV file containing the feedback from the workers on the safety conditions of the workplace over a period of time.
   - The second path requires the workers to either fill out a written survey regarding the safety conditions or perform the live audio recording of the feedback, both of which will be stored in the database.

2) Perform data cleaning on the uploaded file/ survey or recorded feedback from the worker.
   - In this task, the system is required to eliminate noisy data from the provided input to ensure that the sentiment analysis is performed efficiently.

3) Extracting NLP/text-based measures and performing sentiment analysis.
   - This task extracts the required measures from the provided data and performs sentiment analysis on the data to produce the results for the managers so they can judge the safety conditions at their workplace.

7.2 Workflows

Figure 16 is an illustration of the workflows that demonstrate the system solutions and their correspondence to the Goals. Each goal is a requirement of the project and includes how that requirement was fulfilled and developed for this project.
The flowchart illustrated in the section in Figure 17 lays out the backend functionality and the processes followed by the web application. It gives a high-level overview of what the application does behind the scenes.

Figure 16: Workflows and their corresponding system solutions.

Figure 17: Workflows and their corresponding system solutions.
7.3 Wireframes

Figure 18: Web Application - About Page

Figure 19: Web Application - Upload Report
Figure 20: Web Application - Questionnaire Survey

Figure 21: Web Application - Success Screen
Figure 22: Web Application - Survey Results

This section gives a brief overview of how the web application operates and interacts, according to the approved design. The wireframe walkthrough as shown in Figure 18 shows the worker and the manager workflow combined in the same interface.

The User Interface side of the application is divided into two branches at a high level: the administrator branch, and the employee branch. For our prototype, to give a better idea of the branches, we used a drop-down menu to choose the type of view to work with the application. The employee view allows the manager to upload the report of worker feedback responses received over a period of time. This branch also provides the option to view the results generated by the sentiment analysis on the uploaded report.

The worker branch is limited in functionality and allows the workers to quickly upload their feedback into the system through the survey page. To make their lives easier, we have also integrated the functionality of uploading their feedback using their voice.

8.1 Web Application User Interface:

As shown in Figure 23, the landing page of the application is the home page for the administrator view. This page consists of a short description of the tool that we have implemented and what technologies it runs in the background.
Figure 24: User Interface - Upload report

Figure 24 is the upload report page consisting of an option for the administrator to upload a worker survey report that consists of the worker survey feedback over a period of time. The upload button confirms the upload of the document and also begins performing sentiment analysis on the uploaded reported feedback.

Figure 25: User Interface - Sentiment Analysis results
Figure 25 shows the results page which is generated when the sentiment analysis on the worker feedback is completed. It consists of a graph that represents the change of positive, neutral, and negative sentiment results over a year.

**Figure 25: Risk Prediction Sentiment Analysis**

Figure 26: User Interface - Employee home page

Figure 26 shows the employee home page consists of the same information present on the administrator home page currently. This gives a brief overview of the tool and how it functions.

**Figure 26: User Interface - Employee home page**

Figure 27: User Interface - Employee survey page

Figure 27: User Interface - Employee survey page
Figure 27 shows the employee survey page consists of a form that allows the workers to provide feedback on the safety conditions of the workplace on a day-to-day basis. This form includes a feature for the workers to use their voice to record their feedback and the sentiment analysis is performed on the text that is converted from voice to text.
9. Work Completed

Our team worked to successfully build a working web application with a working UI for both employees who will be entering data, and a separate UI for the administrator who will be running analyses on the data collected. The User Interface for the application was developed by Sai Krishna Yeshala who also integrated the speech to text and sentiment analysis into the application, and added the data visualization graph for the sentiment analysis results. In the employee section of the application, we achieved our stretch goal of applying a speech-to-text API for users to enter their reports by voice which would then be analyzed. The speech-to-text development and integration was completed by Sai Gurrapu. The main part of our application was successfully able to process data provided by an “admin user” and send it to be analyzed in the back-end by the sentiment analysis algorithm we developed. The data cleaning code was developed by Jhanavi Ghadia who also was the main point of contact to our client. Our sentiment analysis algorithm was a rule-based algorithm that analyzed the reports given and successfully identified a majority of the positive and negative polarity data points given to it recording accuracy of 77% from our tests on 100 test cases, that were judged manually by Tarun Singh who did the research and development for the sentiment analysis algorithm. We were unable to train an AI model to analyze the sentiment due to a lack of data which in the future could improve the sentiment analysis accuracy. The completed web application is not being hosted on a public URL and the repository has been passed over to the client. After the completion of our web application, we were able to present our application to a team from Gensuite, the company that was able to provide data to our client for this project.
10. Lessons Learned

10.1 Timeline and Schedule
Our initial timeline plans involved a series of milestones over the semester spanning two-week increments where we met with the client and reassessed the situation.

February
- Week 1-2:
  - Review resources available in the CS department
  - Research on sentiment analysis tools and facial expression analysis
- Week 3-4:
  - Get access to data
  - Research different tools to use for sentiment analysis
  - Research and determine the different components

March
- Week 1-2:
  - Develop a Python parser for the given reports
  - Develop a sentiment analysis API to analyze data
  - Develop wireframes for the web application
  - Voice to Text Transcription

April
- Week 1-2:
  - Develop Web app
  - Integrate all the components into the Web App
- Week 3-4:
  - Create Documentation
  - Submit final report to Client

May
- Week 1-2:
  - Final presentation to Gensuite
  - Shared the entire project and report with the client

10.2 Challenges
This was a new experience for all of us and there were many challenges that we faced throughout the development.

10.2.1 Gathering sufficient data
Our client provided us with a basic data schema that we would be working with early on in the project. However, there was a disconnect between the data we would be working with during development and the final data that users would enter and work with. We weren’t sure how the final data would be formatted and this led to some confusion during development as we had to
develop a parser to take in the data and the data cleaning section will clean all the data and pass the most accurate data to the sentiment analysis.

10.2.2 Learning to use different toolkits
Prior to this project, none of the group members had much experience with sentiment analysis and voice-to-text recognition. After a lot of research about sentiment analysis tools and data cleaning libraries, we finally came up with code that would achieve our desired results. There was a large learning curve to using these types of toolkits such as sci-kit learn and the Python NLTK toolkits and overcoming this learning curve was a large part of the work for this project.

10.2.3 Transitioning to virtual collaboration environments
With the COVID-19 outbreak, all of the work for this project was done virtually, which was still relatively new for all of us. Learning how to work together in a virtual environment where we couldn’t directly meet with our client or our group members was a new experience for all of us and had its challenges that we had to figure out. This led to some communication issues as well as difficulty in collaborating in virtual coding environments. We found a solution to collaborate through Zoom video conferencing and Google Drive.

10.2.4 COVID-19
One of our group members contracted the coronavirus during the project which slowed the workflow and led to some scheduling conflicts throughout the project.

10.3 Overall Lessons
We learned that our application should cater to the needs of the workers. Worker safety and environmental conditions must be addressed properly. Our Risk Sentiment Analysis project aims to address the inefficiencies in the process through an intuitive web application that enables workers to share their workplace incidents and reports using their natural voice. Our application performs automatic sentiment analysis on that data to determine how safe the workplace environment is. Moreover, managers can perform data analysis on the received data, see the sentiment score of the workplace, and consider the sentiment of each worker’s response.

10.4 Future Work
The concept of developing a tool that analyzes risk prediction sentiment was Christian’s idea and we were only trying to put his vision into reality through our project. The vision for the project is that Christian can use our work to help companies and allow them to use our platform to assess the risk of their company work based on the data entered by employees. We developed this application with that larger vision in mind, so rather than just having an application that assessed the sentiment of given data, we built a complete platform that could be used by company employees and managers to get a holistic understanding of the application.
For the future, assuming a training loop helps improve sentiment precision, it would be beneficial to obtain more industry data and involve some human administrators to review the results and train the AI to higher levels of precision to reduce false positives and false negatives. The next step would be to deploy a single entry application at one or more pilot sites and have real workers enter their narratives over time and then make user interface improvements to allow that code to be linked to a company’s other data systems. Finally, we can organize and perform a data science project to prove whether safety ‘sentiment’ is correlated with other company performance measures. For example, if a company’s safety sentiment is growing more positive, would they experience improved performance in safety, like lower rates of injury.
11. Acknowledgements

The group would like to acknowledge our client Christian Johnson who was helpful in every way and provided us with all of the necessary guidance and data promptly. Christian was very knowledgeable in his field and provided us with guidance from a non-software development perspective, which allowed us to add improvements and make changes to the project based on his vision. We would also like to acknowledge our professor, Dr. Edward Fox, for his assistance throughout the project and for providing guidance whenever we had questioned about the implementation of the course.
12. References


