Analyzing Suicidal Text Using Natural Language Processing

Cassandra Barton
Utah State University

Follow this and additional works at: https://digitalcommons.usu.edu/gradreports

Part of the Applied Mathematics Commons, and the Other Mathematics Commons

Recommended Citation
https://digitalcommons.usu.edu/gradreports/1640
Analyzing Suicidal Text Using Natural Language Processing

Cassandra DeNise Barton
Advisor: Dr. Luis Gordillo
Department of Mathematics and Statistics, Utah State University
April 26, 2022

Abstract: Using Natural Language Processing (NLP), we are able to analyze text from suicidal individuals. This can be done using a variety of methods. I analyzed a dataset of a girl named Victoria that died by suicide. I used a machine learning method to train a different dataset and tested it on her diary entries to classify her text into two categories: suicidal vs non-suicidal. I used topic modeling to find out unique topics in each subset. I also found a pattern in her diary entries. NLP allows us to help individuals that are suicidal and their family members and close friends.

Key Words: Natural Language Processing, TF-IDF, Singular Value Decomposition, Machine Learning, Topic Modeling.

1 Introduction

According to a study done by The Harris Poll, 93% of Americans believe that suicide can be prevented [8]. For something that the vast majority of citizens believe can be prevented, why is there a death by suicide every 11 minutes [4]? What are we to do about it?

The motivation for this research came primarily from my desire to make more of a direct impact on those around me. This could not have hit closer to home than when my cousin took her own life. Using the skills I acquired in the two semesters immediately preceding her death, I used various methods to detect suicidal sentiment in text with the hope that one day it will be used, not as a post-death study exercise, but as a prevention weapon to combat this cruel mental disease. What I wanted to know was this: Is there a way to anticipate suicidal events before they happen?

Many papers have been published about using Natural Language Processing (NLP) to research suicidal text. In a study done by Gema Castillo-Sanchez, it was found that 50% of research articles used various types of data mining techniques to research suicidal text, including but not limited to, Linguistic Inquiry, Word Count, Latent Dirichlet Analysis (LDA), Latent Semantic Analysis (LSA) and Word2Vec [2]. However, the vast majority of these studies used data found from social media platforms from distinct individuals, often hundreds. John Pestian asked 66 individuals, 33 suicidal and 33 non-suicidal to write suicide notes. He then classified each of those notes as suicidal or not with 70% accuracy [7]. Glen Coppersmith published a paper in which he analyzed suicidal notes from 186 social media users [3]. Andrea Fernandes used over 500 individuals to classify their text as suicidal or non-suicidal using both rule-based programming and machine learning [5]. There are many additional research papers that have been published using similar methods.

However, these research papers were not analyzed on a personal level, largely because of the lack of availability to diaries. It is hard enough to find a diary to analyze. It is even harder to find a diary to analyze of someone who is suicidal. If one is available, the owner has to be extremely vulnerable to give it up as it often reopens fragile and sensitive wounds. In the case of my cousin, my aunt and uncle are still coping with what has happened. For this reason, I searched elsewhere and was lucky enough to find one dataset of a girl named Victoria.

Using machine learning techniques, I classified each of her diary entries as ”suicidal” or ”non-suicidal”.

1
But, I did not just want to classify entries, I wanted to find a pattern in her suicidal entries and see if I could figure out why. After many futile attempts of doing so (TextBlob, Syntatic Dependencies, Word Count, Word2Vec) I found a training dataset to classify text in her diary. I then used topic modeling on those two subsets to determine what the main topics/words were that she was writing about/used when she was "suicidal" versus when she was "non-suicidal". This was all done using Python. The training model I used classified text with 70% accuracy.

2 Analysis

2.1 Datasets

Victoria’s dataset was found on Kaggle [10]. It contains 62 rows, each being an entry in her diary. There are 4 columns:

- **vic_detail**: States what kind of entry
- **journ_entry**: Her journal entry
- **stage**: Numbered 0-6. Her family attached a number to each entry to show the regression of her mental health
- **notes**: Any notes the family has for each entry

Following is what the dataframe structure looks like for Victoria’s diary:

![Figure 1: Dataframe Structure of Victoria’s Diary](image1)

The training dataset I found also came from Kaggle [6]. It is a suicide detection dataset that has 232,074 rows. Each entry came from "SuicideWatch" and "depression" subreddits of the Reddit platform between 2008-2021. There are 2 columns:

- **text**: text entry
- **class**: Classification of each text entry as "suicide" or "non-suicide".

Following is what the dataframe structure looks like for the training dataset:

![Figure 2: Dataframe Structure of Training Dataset](image2)

2.2 Method

The preferred method of text classification up until the 1990’s was to hard-code conditions. Machine Learning was frowned upon because it was not seen as "robust" as hard-coding [1]. In my simple text classifications, I tried doing this and it became very time-consuming. The 1990’s is when Machine Learning started to gain traction because it saved so much time provided you had a dataset to train your model. This was the approach I used to classify my text. I took 80% of the data and trained it. I then used that model to test on the remaining 20% to see how good of a model it would be. It was 70% accurate.

I then used this model to test on the diary of my great-grandmother [9]. She had no history of suicide. I wanted to compare the output of Alice’s diary to that of Victoria’s to make sure the model was working correctly. I created a scatterplot where each of her suicidal comments has an output of 1. If it was non-suicidal, it returned an output of 0.

![Figure 3: Suicide Text Classification Plot - Alice](image3)

We can see in Figure 3 that all entries with the ex-
ception of three were all non-suicidal. I was actually surprised that the model returned any suicidal entries at all because I have read her diary in its entirety and never remembered any indications of suicide. I read those diary entries to see what they talked about. The first was about a flood that scared her to death. The second was about her brother that accidentally chopped the other brother’s toe off while cutting wood. She said he was horrified. The third was about the first time she ever drove a car and how she crashed it in a ditch. She grew up on a farm where accidents were not uncommon. Her lifestyle was different than the majority of people’s today.

I then used the model to predict on Victoria’s diary.

I finally found a pattern! I found that Victoria’s suicidal diary entries occurred in clusters. She had around 3-5 suicidal comments that all occur within around 7 entries. It then seems she is “fine” for another 7 entries or so before she has another suicidal diary entry.

I researched the three most common Topic Modeling methods: Latent Semantic Analysis (LSA), Latent Dirichlet Allocation (LDA), and Term Frequency, Inverse Document Frequency (TF-IDF). Any Topic Model requires the same three basic assumptions:

1. Each document contains a variety of topics
2. Each topic contains a collection of words
3. Documents with similar topics will contain the same words and patterns

This is how they differ:

1. **LSA**: This is the most traditional method. It uses an idea of a distributional hypothesis. This means that words and expressions that occur in similar pieces of text will have similar meanings. It is the most easily implemented.

2. **LDA**: Considered to be the Baysian version of LSA. It is better at generalizing topics.

3. **TF-IDF**: This method gives each word a “score” based on how frequently it appears uniquely in each document. This method is most useful when you are trying to figure out the differences in each document.

After researching the most common methods used in Topic Modeling, I decided to use TF-IDF because I wanted to know what the unique topics/words were between her suicidal entries and her non-suicidal entries.

After counting how many words appear in each document, our matrix will have a structure similar to the following:

```
<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>the</th>
<th>be</th>
<th>am</th>
<th>in</th>
<th>of</th>
<th>on</th>
<th>to</th>
<th>for</th>
<th>with</th>
<th>are</th>
<th>has</th>
<th>have</th>
</tr>
</thead>
<tbody>
<tr>
<td>Doc1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Doc2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Doc3</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Doc4</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
```

This causes problems though because the more documents you include, the greater number of terms it will have, and the size of this matrix can become too large to deal with. This is where Singular Value Decomposition (SVD) is used.

### 2.3 Singular Value Decomposition

Before we can use (SVD), we need to preprocess the text. This is done by tokenization (putting each word in a list), lemmatization (changing each conjugation of a word back to its root form so that we avoid analyzing words that mean the same thing), removing punctuation and stopwords removal (removing unnecessary words such as “the, and, in, etc”. After
we preprocess the text, we then use Singular Value Decomposition. This allows us to break up a large matrix, often with hundreds or thousands of unique words (columns), into three separate matrices.

\[ M = U \cdot \sum \cdot V \]  

(1)

\( M \) is our original matrix similar to that in Figure 5. It is a \( m \times n \) matrix with \( m \) files/documents and \( n \) terms. But, there are only so many important topics. So, we know that \( n \) contains a certain number of important topics, \( r \). \( U \) is a \( m \times r \) matrix and is called the document-topic matrix. It has the following structure:

![Figure 6: Structure of the matrix U](image)

But, there is not a way for us to determine what each of these topics represent. That is where the matrix \( V \) becomes useful. It is the term-topic matrix that is \( r \times n \). It allows us to determine what each of the topics may be. It has the following structure:

![Figure 7: Structure of the matrix V](image)

For each of the topics, we can order each of the rows from descending value. The topics contains the rows with the highest values. \( \sum \) is a diagonal matrix that is \( r \times r \) that allows us to complete the matrix multiplication. We are now ready to do TF-IDF.

### 2.4 TF-IDF

In TF-IDF, if a word occurs many times in a document, we give it a high score. However, if given many documents, and the word occurs frequently across all of them, this does not reveal very much information. Thus, we give that word a low score.

As an example, if we are given a book about horses and each chapter covers a different horse breed, TF-IDF will not return the word "horse". It will return words such as "thoroughbred", "quarter", "percheron", "appaloosa", etc. Because those words appear uniquely in each of their respective chapters (documents). The formula is the following:

\[ w_{i,j} = tf_{i,j} \cdot \log \left( \frac{N}{df_i} \right) \]  

(2)

where \( tf_{i,j} \) is the number of times a word \( i \) appears in a document \( j \), \( df_i \) is the number of documents that contain \( i \), and \( N \) is the total number of documents. We provide an example. Suppose we have the following two documents with their respective term frequencies:

![Figure 8: Term Counts for each Document](image)

If we want to find the TF-IDF of the word "this", we have the following:

\[ tf('this', d_1) = \frac{1}{5} = .2 \]

\[ tf('this', d_2) = \frac{1}{7} \approx .14 \]
\[ idf('this', D) = \log(\frac{2}{2}) = 0 \]

\[ tfidf('this', d_1, D) = 0.2 \cdot 0 = 0 \]

\[ tfidf('this', d_2, D) = 0.14 \cdot 0 = 0 \]

Because the word "this" appears roughly the same number of times in both documents, under TF-IDF, it does not give us any useful information in the differences between the two. Thus, we give that word a score of 0 for both documents. We try the word "example":

\[ tf('example', d_1) = \frac{0}{5} = 0 \]

\[ tf('example', d_2) = \frac{3}{7} \approx 0.429 \]

\[ idf('example', D) = \log(\frac{2}{1}) = 0.301 \]

\[ tfidf('example', d_1, D) = 0 \cdot 0.301 = 0 \]

\[ tfidf('example', d_2, D) = 0.429 \cdot 0.301 \approx 0.129 \]

Thus, we give the word "example" a score of 0 in Document 1 and a score of 0.129 in Document 2. This tells us that Document 2 used "example" more than Document 1.

2.5 Results

After performing SVD and TF-IDF, we obtained the following results for Victoria’s diary.

We see in Figure 8 that for Victoria’s suicidal comments, her top topics/words were “like, thing, know, want, one, people would, time, etc.”. Assuming her writing would not vary while we analyze her diary, we would tell her family that when she uses those words in her writing, she is in a suicidal state. In this way, the family can be aware without nagging her and wondering if she is “okay” or not. From Figure 4, we would also tell her family that these suicidal entries occur in a cluster of about 7 and then her suicidal entries will not return until roughly 7 entries later.

3 Conclusion

We were able to create a model that would predict when suicidal events would happen. Natural Language Processing is extremely useful when researching ways to help those that are suicidal. Medical professionals are not the only ones that are able to help. Mathematicians can do it as well. Using machine learning I was able to train a model to test on each of her entries and classify them as suicidal or non-suicidal. I then used topic modeling to find words she used when she was suicidal versus when she was not. If I were to explain my conclusions to her family, I would tell them when she uses certain words or topics in her writing, she is in a suicidal state and that it will be that way for about a week before she comes out of it. Families should be supportive at every stage, but this helps them be aware of the process.
4 Future Work

Future work that could be done to improve this project would be to create a more accurate training model to classify the text with higher accuracy. The model I used predicted with 70% accuracy. This would allow TF-IDF to pick out unique words better because the entries would be classified correctly.

I hope in the future to be able to develop some kind of app that can send notifications to family members if a loved one is in a suicidal slump. The majority of teens spend a lot of time on the internet and social media so I would develop it so it took in what they are reading on the internet or texts they write to others. This would be conditional on the users allowing access to this kind of information. With every little step, we can make progress towards what 92% of Americans believe is preventable, suicide.

References

[1] Alan Blair. Lec 6: Agent Types - Artificial Intelligence, Alan Blair, UNSW. YouTube Lectures. 2018. URL: https://www.youtube.com/watch?v=QnEWab0hj0&list=PLkdgwLLVhz12NLKDFBU1Pvg~T0aMk017&index=7.


