Using AI to Detect Mental Health Risks in Tweets
By Max Ludwikowski

Introduction

Modern studies have indicated that a rise in rates of depression and anxiety, particularly among younger generations, can be attributed to a growing reliance on social media (Haidt). X, formerly known as Twitter, is one of the longest running and most popular social media platforms in the world, with over 368 million active users. An AI-powered textual analysis tool could scan tweets for distressing language and help X, and other platforms of a similar size, identify users who may be struggling with their mental health. With such a tool, X could guide users to helpful resources, therefore increasing the likelihood that they find treatment to improve their situation.

The goal of this project was to build, evaluate, and compare selected models, and, in turn, establish which would be best suited to solving the problem outlined above. The models were trained on pre-existing datasets containing tweets that were manually labeled in accordance with how indicative they were of mental health issues (see Data section for more details). Two models were constructed, one a K-Nearest Neighbors Semantic Search (KNN-SS) model and the second a single-layer perceptron. Both models were trained using text encoders (or vectorizers) to transform the text from the tweets to numerical scores or features. Ultimately, while both models proved reasonably accurate (72%-79% accuracy range for the KNN-SS and 82%-86% for the perceptron), the perceptron was determined to be the superior model in terms of accuracy and speed.

Methods

Given the multitude of techniques available for performing text classification, it was necessary to construct multiple models to evaluate differences in accuracy in speed. Both metrics are extremely important both ethically and logistically as such a tool would need to balance the sensitivity of identifying afflicted individuals while remaining functional when scaled to fit platforms of the size of X. Two models were constructed and compared against each other with these parameters in mind.

1. K-Nearest Neighbors Semantic Search

Our first model is a K-Nearest Neighbors Semantic Search (KNN-SS) built using the sentence transformer library from Hugging Face as well as the torch library from PyTorch. The term semantic search refers to techniques that analyze and search for text based on meaning. This can be differentiated from lexical search which tracks literal word or pattern matches. Semantic search can further be classified as symmetric and asymmetric. Symmetric semantic search algorithms are designed to match a query to an entry of similar size whereas an asymmetric search would typically take a shorter query and match it to a longer entry. Because tweets are limited to 280 characters, our model uses a symmetric semantic search.

Tweets are embedded in the “corpus” (training set) using a sentence transformer that maps the inputted text to 384-dimensional vector space. From there, our model takes each tweet

from the corpus and assigns it a cosine-similarity score for each test query. In practice, this process is measuring the similarity of a singular testing tweet to each individual tweet in the corpus. Our model then isolates the 5 tweets from the corpus with the highest similarity score; in other words, the 5 nearest neighbors. Lastly, the score *x* assigned to each neighbor is multiplied by a scalar *s* denoting its assigned label. These values are averaged to yield a final value for each test query. If that value exceeds a certain threshold *t*, the tweet is given a label *L* of 1 for "concerning," otherwise, the tweet is given a label of 0 for "benign."

Equation 1.1

$$L = \{0\ if\ \frac{1}{5}\sum_{5}^{i=1} x_i s \leq t;\ 1\ if\ \frac{1}{5}\sum_{5}^{i=1} x_i s > t$$

2. Perceptron

The second model constructed was a single-layer perceptron powered by the scikit-learn library. Scikit-learn's TfidfVectorizer function (see Equation 1.2) was used to convert the text from the tweets to numerical features. Per the documentation for scikit-learn, "Tf means term-frequency while tf–idf means term-frequency times inverse document-frequency." This function, in a similar manner to Google's PageRank algorithm, re-weights count features depending on how common a certain term *t* is. This works slightly differently from the sentence transformer used in the KNN-SS model in that smaller, frequently occurring words ("the", "a", "an", etc.) are given less weight due the lack of meaning they carry.

Equation 1.2

$$TFIDF(t,d) = TF(t,d) * IDF(t)$$

The crucial element here is the IDF term, which re-weights words given on how often they appear in other documents *d* (or in our case tweets) of the training set. The IDF can be computed as:

Equation 1.3

$$IDF(t) = log(\frac{1+n}{1+df(t)}) + 1$$

where *n* is the number of documents (or tweets) and *df(t)* is the number of documents with the specified term *t*.

The perceptron is then built by setting a number of training epochs (or training cycles) and a learning rate to use as a scalar for weight updates. Unlike the KNN-SS model which outputs scores on a continuous scale, the perceptron model simply uses binary classification to make its prediction.

Data

Data was collected from two sources, both prelabeled allowing for a smoother testing process.

A. Dataset 1

The first dataset contained over 2400 tweets including relevant mental health hashtags, with each being given one of three labels:

Table 1.1

Original Label	**New Label**	**Description**
0	0 or “benign”	A tweet meant to raise awareness for mental health or to offer help and advice
1	1 or “concerning”	A tweet containing one of the following: sarcastic statements, rants, and expressions of annoyance; general disturbances
2	1 or “concerning”	A tweet with dejected tone containing language indicating suicidal ideation or a case of depression

It was decided that the tweets labeled 0 would fall into the “benign” category and tweets labeled 1 and 2 would be joined together into a “concerning” category as manual analysis of the dataset showed that tweets with the label 1, while often describing fleeting moments of anger, nonetheless contained distressing language.

B. Dataset 2

The second dataset of over 1700 tweets, unlike the first, was not filtered to include mental health relevant hashtags. As such, the tweets in this dataset are more varied, providing good contrast with which to test our models.

Table 1.2

Original Label	**New Label**	**Description**
“Not Suicide Post”	0 or “benign”	Tweets that do not express

		any suicidal sentiments or intentions
“Potential Suicide Post”	1 or “concerning”	Tweets that exhibit indications of suicidal thoughts, feelings, or intentions.

Experiments

1. K-Nearest Neighbors Semantic Search

The first part of our experimentation process involved training and testing our KNN-SS model. Dataset 1, the larger of the two, was used to train the KNN-SS model while Dataset 2 formed the pool of testing tweets. The two datasets were separated into training and testing sets for two reasons. First, to maximize the model’s testing capabilities, the larger dataset was a better fit. Secondly, the tweets from Dataset 1 were filtered for the presence of mental health specific hashtags while the tweets in Dataset 2 were far more varied. Thus, we wanted to test if the KNN-SS model could adapt to this variety during testing to produce accurate results.

The tweets from Dataset 1 formed the corpus while 750 tweets, or queries, were selected at random from Dataset 2 to test the model. Upon determining the scores of the 5 nearest neighbors for each query tweet, a series of basic calculations was made to determine the predicted label. The score for each neighbor was multiplied by a scalar (-1 for “benign” tweets and 1 for “concerning” tweets), and the products were summed together and divided by 5 to yield a final average for each query tweet. If the average score was greater than or equal to threshold $t = 0.35$ (see Equation 1.1), the tweet was labeled “concerning,” and if less than 0.35, it was labeled “benign.” This threshold was found after extensive testing runs found that a value nearing 0.5 resulted in an exceeding number of false negatives (tweets labeled as “benign” that were in fact “concerning”). Accuracy for our K-Nearest Neighbors Semantic search model ranges between 72% and 79%.

2. Perceptron

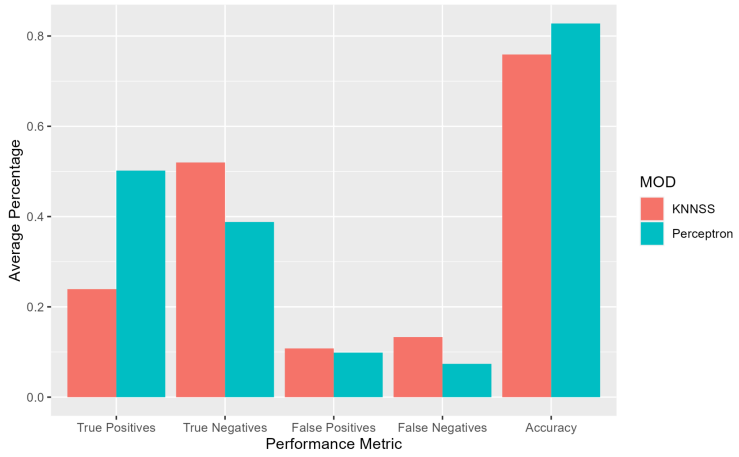
The second part of our experimentation consisted of training and testing our perceptron method in order to compare it against our existing KNN-SS model. Unlike in part one, Datasets 1 and 2 were combined to form a larger training set. This was done to determine if an increased variety in the dataset, even if on a smaller scale such as ours, would translate to increased accuracy.

The testing set was also increased to 25% of total tweets from our two datasets (roughly 1000 queries). The number of training epochs was set to 5000 and the learning rate was set to 0.1. These parameters were chosen in order to maximize speed and minimize volatility. When evaluating the model, a higher number of training epochs (in the order of tens of thousands) greatly decreased the speed of the program while a limited number had slight impacts on

accuracy. Furthermore, a larger learning rate, while not exceedingly detrimental to overall accuracy, often led to large differences between the number of false positives and false negatives. As such, a moderate value for the learning rate was deemed preferable. The perceptron model proved noticeably more accurate than the KNN-SS model with a range between 82% and 86%. This is partly explained by the TfidfVectorizer, which might be better suited for textual queries than the cosine-similarity score used for the K-Nearest Neighbor model. Additionally, the perceptron was considerably faster than the KNN-SS model. This is likely down to the fact that the KNN-SS model performs computations for every corpus tweet each time it is sent a test query. Conversely, the perceptron model is concerned only with binary classification which is not nearly as computationally intensive.

Results

Figure 1.1



Each of the two models was put through 50 test runs to evaluate accuracy, speed, and other advanced metrics. Both models showed considerable promise in terms of accuracy with the range for the KNN-SS and perceptron models averaging an accuracy of 76% and 83%, respectively. While these measurements are not sufficient for either model to be put into practice, they illustrate that creating a full-fledged tool is not beyond the realm of possibility. Measured also was the average percentage of true positives, true negatives, false positives, and false negatives across the 50 test runs. False positives refer to tweets mislabeled as “concerning” when they were in fact “benign” and vice versa for false negatives. The errors were evenly distributed between false positives and false negatives for both models, although this was not the case for the KNN-SS model prior to lowering the labeling threshold (see Equation 1.1).

Of note, however, was the difference between the two models in average true negative and true positive percentage. True positives here refer to tweets correctly labeled as “concerning” and true negatives to tweets correctly labeled as “benign.” The KNN-SS model, despite a lower overall accuracy, outperformed the perceptron model in average true negative percentage while the perceptron proved largely superior for average true positive percentage. While this could

simply be down to a small sample size, it is certainly a phenomenon to look into if further research is to take place.

Lastly, while both models impressed, it was determined that the superior accuracy and speed of the perceptron made it better suited for test classification in tweets given the scale of this project.

Conclusion

While we consider both of the KNN-SS and perceptron models to be successes considering the scale of this project, it is worth noting that rolling out such a tool on the platform would be an extremely sensitive endeavor as large numbers of both false positives and false negatives would cause a number of issues. False positives would disrupt user experience and, ultimately, discourage users from posting. Additionally, false negatives would undermine the goal of the tool altogether. In either case, X could face user and even legal backlash if the model is proven to be glaringly inaccurate.

As such, while the results from the two models (the perceptron in particular) were promising and exceeded expectations, further development would need to take place for a full-fledged tool to be released. A notable improvement would be to compile larger datasets. The datasets used for this project totaled around 4300 tweets, which while enough to form a basic training set, is not comprehensive enough to reflect the variety of tweets posted daily on X.

Works Cited

“6.2. Feature Extraction.” *Scikit*, scikit-learn.org/stable/modules/feature_extraction.html#text-feature-extraction. Accessed 24 Apr. 2024.

Haidt, Jonathan. *The Anxious Generation: How the Great Rewiring of Childhood is Causing an Epidemic of Mental Illness.* Allen Lane, 2024.

“Model Created Using AI and Tweets to Help Early Detection of Mental Disorders.” *The Economic Times*, 11 Apr. 2023, economictimes.indiatimes.com/tech/technology/model-created-using-ai-and-tweets-to-help-early-detection-of-mental-disorders/articleshow/99402817.cms?from=mdr.

"Number of Twitter Users Worldwide from 2019 to 2024." Statista, www.statista.com/statistics/282087/number-of-monthly-active-twitter-users/.

“Perceptron Algorithm for Classification Using Sklearn.” *GeeksforGeeks*, GeeksforGeeks, 11 Oct. 2023, www.geeksforgeeks.org/sklearn-classification-using-perceptron/.

“Semantic Search.” *Semantic Search - Sentence-Transformers Documentation*, sbert.net/examples/applications/semantic-search/README.html#python. Accessed 24 Apr. 2024.

“Sentence-Transformers/All-Minilm-L6-V2 · Hugging Face.” *Sentence-Transformers/All-MiniLM-L6-v2 · Hugging Face*, huggingface.co/sentence-transformers/all-MiniLM-L6-v2. Accessed 24 Apr. 2024.

“What Is Semantic Search?: A Comprehensive Semantic Search Guide.” *Elastic*, www.elastic.co/what-is/semantic-search. Accessed 23 Apr. 2024.