A Data-Driven Approach to Predicting Emotional Sentiment

Jackson La Vallee, Niraj Girase, Randal Fan, Tanuj Kenchannavar

Measuring emotional sentiment in the context of a sentence is a challenging task that encompasses multiple approaches to interpret and predict textual inputs accurately. Our project analyzes social media posts from common platforms, including Reddit and Twitter, and identifies key features from comments to predict the sentiment of each comment. Our guiding question is: How binary is emotional sentiment, and how confidently can we classify statements into one or more categories?

Our dataset[1] contains 53042 aggregate comments and their associated sentiment, which falls into one of seven categories[2]. Consider the statement: "I don't know what is going on in my life right now. It seems as though my mind is spinning with problems." The associated sentiment of this statement is Anxiety. It is not surprising that statements could easily fall under multiple categories. As this dataset is composed of words, visualization is complex. Later, after isolating features, we will apply Principal Component Analysis (PCA) to gain a preliminary visualization of our data.

Text classification has long been a central research focus in Natural Language Processing (NLP), with applications ranging from spam detection and sentiment analysis to mental health assessment and topic modeling. Given the domain-specific nature of emotional sentiment detection, careful attention must be given to preprocessing, feature representation, and the selection of robust machine learning algorithms.

Preprocessing is a crucial first step in any NLP pipeline, aimed at cleaning and normalizing raw text to make it more suitable for feature extraction and machine learning.

- **Text Normalization:** Converting text to lowercase and removing punctuation, numbers, and special characters reduces noise. As *Cambria and White* (2014) noted, normalization enhances lexical uniformity, facilitating the grouping of semantically identical terms (e.g., "Happy" and "happy").
- **Contraction Expansion:** Expanding contractions (e.g., "can't" to "cannot") improves semantic clarity, which is particularly critical in sentiment and emotion classification tasks where subtle linguistic differences can affect label assignment (*Ghosh and Veale*, 2017).
- **Tokenization and Stopword Removal:** Tokenization, often implemented via NLTK (*Bird et al.*, 2009), segments text into meaningful units. While removing common stopwords like "the" or "is" generally improves classification (*Manning et al.*, 2008), *Mohammad et al.* (2013) observed that retaining emotionally-relevant stopwords may sometimes be beneficial.
- **Handling Missing and Duplicate Data:** Ensuring data integrity by eliminating incomplete or duplicate entries is vital, as emphasized by *Han et al.* (2011), who noted that data quality substantially impacts model reliability.

Next, feature extraction transforms text into numerical vectors interpretable by machine learning models.

- **Bag-of-Words (BoW):** This approach treats each document as a multiset of words, discarding grammar and word order. *Harris* (1954) demonstrated that BoW can still capture significant information, particularly when frequency-based weighting is applied.
- **Term Frequency-Inverse Document Frequency (TF-IDF):** TF-IDF refines the BoW approach by down-weighting standard terms and emphasizing rare, informative ones (*Ramos*, 2003). Our project employed a TF-IDF vectorizer with parameters such as sublinear_tf=True, max_features=3000, and min_df=3, aligning with best practices for sparse document classification (*Sebastiani*, 2002).

Despite the rise of deep learning models like LSTMs and Transformers, classical machine learning algorithms remain highly effective for small to medium-sized datasets:

- **Support Vector Machines (SVMs):** SVMs are well-suited for high-dimensional data sets. *Joachims* (1998) showed that SVMs using kernel functions often outperform traditional classifiers. However, they are sensitive to class imbalance, as demonstrated by *Forman* (2003), who showed that performance can degrade without balancing techniques such as SMOTE or class weighting, which is consistent with our observed low precision for minority classes.
- **Random Forest:** Introduced by *Breiman* (2001), Random Forests reduce overfitting through ensemble learning but can struggle in sparse, high-dimensional text settings without careful feature engineering *(Fernández-Delgado et al.*, 2014).
- **Multinomial Naive Bayes (MNB):** Particularly effective for text classification, MNB assumes a multinomial distribution of word frequencies (*McCallum and Nigam*, 1998). However, it struggles with rare word distributions and overlapping class boundaries, as evidenced by our poor F1-scores for underrepresented emotional categories.
- **Stochastic Gradient Descent (SGD) Classifier:** Optimized through online learning, SGD is well-suited for large-scale, sparse data (*Zhang et al.*, 2004; Pedregosa et al., 2011). In our project, SGD achieved the highest accuracy and F1-scores, reflecting its ability to generalize effectively in high-dimensional environments.

Emotion detection datasets often exhibit significant class imbalance, such as in our case, with dominant classes such as "Normal" and "Depression" outnumbering others like "Stress" or "Personality Disorder." *He and Garcia* (2009) outlined methods to address this, including oversampling, undersampling, and cost-sensitive learning. However, aggressive undersampling can lead to information loss. Studies like *Poria et al.* (2017) stress the necessity of choosing evaluation metrics beyond accuracy, such as precision, recall, and F1-score.

Regarding performance metrics, relying solely on accuracy for evaluation can be misleading, particularly in imbalanced datasets. *Sokolova and Lapalme* (2009) advocate for macro-averaged and per-class precision, recall, and F1-scores for a more nuanced assessment.

Dataset, Pre-processing, and Feature Extraction

We worked with a textual dataset for a classification task. The dataset underwent several preprocessing steps to clean and prepare the text data for machine learning algorithms:

- **Drop missing values** and **duplicate rows** to ensure data integrity.
- **Text normalization:**
    - Converted all text to lowercase.
    - Removed special characters and punctuation using regular expressions.
    - Expanded contractions (e.g., “don’t” to “do not”) using the contractions library.
- **Tokenization** and **stopword removal:** Used the Natural Language Toolkit (NLTK) to tokenize the text and remove common stopwords that do not contribute to classification.
- Finally, any rows left **empty** after processing were removed.

After preprocessing, features were extracted using either bag-of-words or TF-IDF representations, respectively, depending on the algorithms applied. We experimented with four classical machine learning algorithms tailored for text classification. Each model was evaluated using cross-validation, and performance was assessed using metrics such as accuracy, precision, and recall. Some preliminary Exploratory Data Analysis (EDA) using PCA with three PCs allowed us to visualize the first 100 datapoints of each category. Clearly, they overlap:

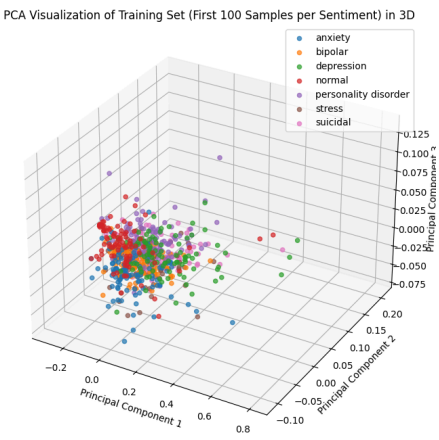

Figure: A scatterplot of 700 datapoints after applying a reduction via PCA

Model 1: Support Vector Machine (SVM)

- **Concept**: SVMs attempt to find the optimal hyperplane that separates classes in a high-dimensional space. The kernel trick enables the handling of non-linearly separable data.
- **Observation**: Although SVMs are effective in high-dimensional spaces, our model showed low precision. Dataset balancing was considered, but this would have required the removal of a large volume of data, resulting in significant information loss.

Model 2: Random Forest

- **Concept**: An ensemble learning technique that builds multiple decision trees on bootstrapped datasets and combines their predictions.
- **Observation**: Performance improved compared to SVM, but overall accuracy remained lower than expected. It handled the dataset size and types well, but lacked the edge in prediction quality.

Model 3: Multinomial Naive Bayes

- **Concept**: Particularly suited for text data, this model uses word frequency to calculate the likelihood of class membership using Bayes' Theorem.
- **Observation**: While efficient for sparse text data, it struggled with recognizing rare words, resulting in significantly low precision.

Model 4: Stochastic Gradient Descent (SGD) Classifier

- **Concept**: A linear classifier trained using SGD, which updates model parameters after each training instance, making it highly efficient for large, sparse datasets.
- **Observation**: This model outperformed others in accuracy, precision, and recall, making it the best performer for our dataset. Its compatibility with high-dimensional, sparse features made it especially effective for text classification.

Hyperparameter tuning and cross-validation were performed to optimize model performance and prevent overfitting, For the Support Vector Machine (SVM), the kernel type and regularization parameter 'C' were adjusted. In the Random Forest model, the number of estimators and the maximum tree depth were varied. For the Stochastic Gradient Descent (SGD) classifier, the regularization term (α) and loss function were tuned to achieve the best balance between training efficiency and predictive performance.

Results

To evaluate model performance, we split the preprocessed dataset into training and test sets using an 80/20 split. We transformed the text using a TF-IDF vectorizer with enhanced parameters (max_features=3000, sublinear_tf=True, min_df=3). This vectorization captured the importance of relevant terms while accounting for term frequency saturation and document frequency constraints. We evaluated three final machine learning classifiers: **Stochastic Gradient Descent (SGD) Classifier**, **Multinomial Naive Bayes**, and **Random Forest**. Their performance was assessed using accuracy, precision, recall, and F1-score. The classification reports below summarize the performance of each model across seven emotional sentiment categories.

SGD Classifier emerged as the best-performing model overall, achieving an **accuracy of 71.3%** on the test set. It demonstrated strong performance on high-frequency classes, such as *Normal* (precision: 0.78, recall: 0.96) and *Depression* (recall: 0.71). However, it struggled to recall lower-represented classes, such as *Personality Disorder* (recall: 0.10) and *Stress* (recall: 0.15), highlighting the impact of class imbalance.

Random Forest achieved an accuracy of **69.1%** and performed relatively better than Naive Bayes on minority classes, though it still faced challenges in distinguishing fine-grained sentiments. For example, it predicted *Personality Disorder* with 100% precision but only 22% recall. When used to classify an input sentence, it predicted "*Personality Disorder*" with 58% confidence, higher than other classes like *Depression* (25%) or *Suicidal* (8%).

Multinomial Naive Bayes yielded the lowest performance, with **an accuracy of 65.0%**. While effective in capturing broader class signals (*Normal*: F1-score 0.80), it was less robust in handling rare classes, resulting in low F1-scores for *Stress* (0.09) and *Personality Disorder* (0.11).

Classifier	Accuracy	Precision	Recall	F1-Score
SGD Classifier	0.713	0.699	0.650	0.636
Random Forest	0.691	0.706	0.691	0.677
Naive Bayes	0.650	0.699	0.650	0.636

Table: Comparing various metrics of the three different classifiers simulated

The results highlight the trade-offs between precision and recall for each model, underscoring the need for improved techniques for handling underrepresented classes. These performance gaps were also linked to preprocessing steps, particularly the unintentional removal of negation terms such as “not” or “don’t,” which are crucial for accurately interpreting sentiment.

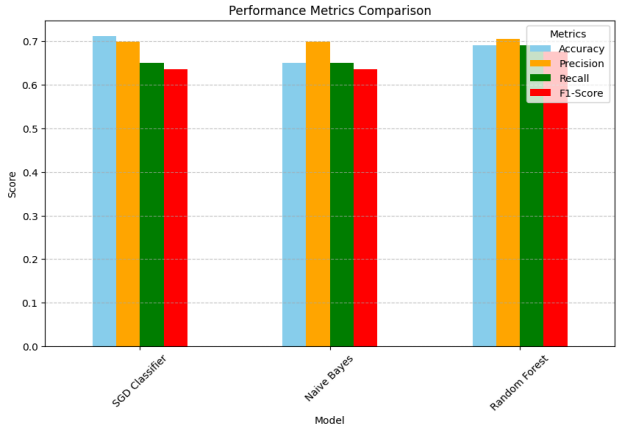

Figure: Performance metrics comparison across the three simulated models

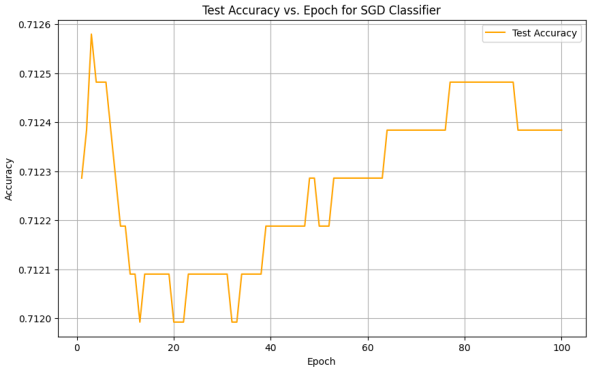

Figure: Accuracy with uneven weights

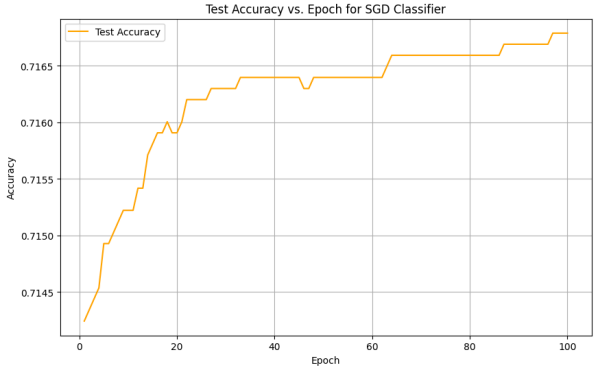

Figure: Accuracy with equal weights

Discussion

In this project, we investigated the prediction of emotional sentiment in social media comments using a dataset of over 53,000 posts from Reddit and Twitter. We trained and evaluated four classical classification models: Support Vector Machine (SVM), Random Forest, Multinomial Naive Bayes, and Stochastic Gradient Descent (SGD) Classifier. Among these, the SGD Classifier consistently outperformed the others in terms of precision, recall, and F1-score, particularly after addressing class imbalance through weight adjustments. The model demonstrated strong performance on high-frequency classes such as “Normal” and “Depression,” but struggled to accurately recall instances of underrepresented classes like “Personality Disorder” and “Bipolar.”

A major challenge encountered was the pronounced class imbalance within the dataset. Underrepresented categories (e.g., Personality Disorder) contributed to biased model predictions and reduced overall recall for minority classes. An additional issue arose during preprocessing: the removal of critical negation terms (e.g., “not,” “do not”), which are vital for sentiment interpretation. For instance, the sentences “I don’t have anxiety” and “I have anxiety” were treated identically, leading to misclassification. This highlighted the risk of semantic distortion during preprocessing, a key consideration in sensitive applications such as mental health detection, where misclassification could have serious consequences.

Unexpectedly, despite the theoretical suitability of SVM and Naive Bayes for text classification, both models underperformed relative to Random Forest and SGD, mainly due to the high sparsity and imbalance of the data. Random Forest achieved moderate success but still struggled with fine-grained differentiation among emotional sentiments. Several improvements could enhance model robustness. Incorporating explicit negation handling through custom stopword lists or negation tagging techniques (such as appending _NEG to words following a negation) would preserve critical semantic information. Furthermore, applying data augmentation methods can alleviate class imbalance and improve the performance of minority classes.

This project demonstrated the promise of machine learning in detecting emotional sentiment from social media data, suggesting important implications for early mental health interventions. It highlighted the complexity of sentiment classification, particularly where ethical considerations and nuanced language understanding are crucial. It also demonstrated how challenging it is to categorize sentiment in a strictly binary fashion.

Our team contributed equally throughout the project's completion. Randal focused on elements of tokenization and feature extraction, intending to design the ideal input to our models. Tanuj and Jackson focused on model development and setting up a driver for the modeling, intending to apply algorithms to transform the input. The focus of this part was mainly on implementation and optimization. Niraj delved deeply into the past literature on sentiment classification and proposed various models that should be tested against our problem, focusing on finding applicable and optimal algorithms.

References

[1] Sarkar, S. (2022). Sentiment Analysis for Mental Health [Data set]. Kaggle. https://www.kaggle.com/datasets/suchintikasarkar/sentiment-analysis-for-mental-health

[2] Note: The dataset includes seven sentiment categories: Normal, Depression, Suicidal, Anxiety, Stress, Bipolar, and Personality Disorder.

[3] Cambria, E., & White, B. (2014). Jumping NLP Curves: A Review of Natural Language Processing Research. IEEE Computational Intelligence Magazine.

[4] Ghosh, A., & Veale, T. (2016). Fracking Sarcasm Using Neural Network. Proceedings of the 7th Workshop on Computational Approaches to Subjectivity, Sentiment & Social Media Analysis.

[5] Bird, S., Klein, E., & Loper, E. (2009). Natural Language Processing with Python. O'Reilly Media.

[6] Manning, C. D., Raghavan, P., & Schütze, H. (2008). Introduction to Information Retrieval. Cambridge University Press.

[7] Mohammad, S. M., Dunne, C., & Dorr, B. (2013). Generating High-Coverage Semantic Orientation Lexicons from Overtly Marked Words and a Thesaurus. Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing.

[8] Han, J., Kamber, M., & Pei, J. (2011). Data Mining: Concepts and Techniques (3rd ed.). Morgan Kaufmann.

[9] Harris, Z. S. (1954). Distributional Structure. Word.

[10] Ramos, J. (2003). Using TF-IDF to Determine Word Relevance in Document Queries. Proceedings of the First Instructional Conference on Machine Learning.

[11] Sebastiani, F. (2002). Machine Learning in Automated Text Categorization. ACM Computing Surveys.

[12] Joachims, T. (1998). Text Categorization with Support Vector Machines: Learning with Many Relevant Features. Proceedings of the European Conference on Machine Learning.

[13] Forman, G. (2003). An Extensive Empirical Study of Feature Selection Metrics for Text Classification. Journal of Machine Learning Research.

[14] Breiman, L. (2001). Random Forests. Machine Learning.

[15] Fernández-Delgado, M., Cernadas, E., Barro, S., & Amorim, D. (2014). Do We Need Hundreds of Classifiers to Solve Real World Classification Problems? Journal of Machine Learning Research.

[16] McCallum, A., & Nigam, K. (1998). A Comparison of Event Models for Naive Bayes Text Classification. AAAI-98 Workshop on Learning for Text Categorization.

[17] Zhang, T. (2004). Solving Large Scale Linear Prediction Problems Using Stochastic Gradient Descent Algorithms. Proceedings of the Twenty-First International Conference on Machine Learning.

[18] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research.

[19] He, H., & Garcia, E. A. (2009). Learning from Imbalanced Data. IEEE Transactions on Knowledge and Data Engineering.

[20] Poria, S., Cambria, E., Hazarika, D., & Vij, P. (2017). A Deeper Look into Sarcastic Tweets Using Deep Convolutional Neural Networks. Proceedings of the 26th International Conference on World Wide Web Companion.

[21] Sokolova, M., & Lapalme, G. (2009). A Systematic Analysis of Performance Measures for Classification Tasks. Information Processing & Management.