Valentina Gonzalez Bohorquez
Varun Krishnan
954:577 STAT Software
Team #10

Final Project
Project Report
Dec. 16, 2021

Text Emotion Classification

I. Introduction

Natural language processing (NLP) employs computational and linguistics techniques to aid computers understand, and sometimes generate human languages in the form of texts and speech. Sentiment analysis, a branch of NLP, analyzes an individual's opinions, sentiments, attitudes, and emotions towards objects or topics. The role of sentiment analysis is to detect and extract all sentiment entities and their attributes in the context of a document, such as a tweet. The purpose of this project was to identify the best emotion classification model for a set of tweets from Twitter. This multinomial classification problem employed baseline models: Naive Bayes and a Simple Neural Network. Additionally, it employed an upgraded Neural Network, a Neural Network with Glove Embeddings, and a Bidirectional Encoder Representations from Transformers (BERT) model. However, the model predicted to perform the best is the DistilBERT architecture. Varun worked on the baseline approaches, Naive Bayes and the Neural Networks and Valentina worked on the data cleaning/visualizations and the main approach, the BERT model.

II. Related Work

Emotions play vital roles in human existence by reflecting our current state and well-being. In recent years, the rise of social media has greatly contributed to online reviews and ratings, whose ideals are based on an individual's emotions and self-expression. For instance, a customer's review encodes their emotions regarding the purchase of a particular product, which in return provides insight into the decision-making process and ensures business growth [3]. According to "Mining Opinions, Sentiments, and Emotions," sentiment text is divided into five sections: object, attribute of the object, sentiment meaning to the object, holder of sentiment, and time of sentiment expression [2]. This structure can be based on a product, service, topic, person, organization, or event. The connection established in this structure is described as a set of feature values that have different layers, and each of these layers has their own attribute values [1]. Furthermore, the sentiment meaning of an object refers to the sentiments of an entity in a certain

attribute [3]. These can be expressed as positive, negative, neutral, or contain different intensities of such attitudes.

Sentiment classification analyzes the overall emotional bias of a document. Individuals may have different sentiments regarding distinct matters, which can reveal various emotions that differ from one entity to another for the same object. As a result of this, a multisentiment, or multiclassification, problem is created, where there may be one sentiment for one entity, and another sentiment for another, within an analysis of the same object, or topic [3]. Dividing human sentiments into several classes, such as joy, anger, fear, love, sadness, and surprise, is referred to as emotion recognition, or emotion detection. This branch of sentiment analysis represents the extraction and analysis of emotions. Many different methodologies can be employed to interpret emotion detection, such as lexicon based, machine learning based, and deep learning based. Each methodology has its own set of benefits and drawbacks. For the purpose of this project, machine learning algorithms Naive Bayes, several Neural Networks, and BERT model architectures will be assessed.

III. Datasets

The dataset is publicly available on Kaggle. The authors of the dataset constructed a set of hashtags to collect English tweets from a Twitter API belonging to six emotions: anger, fear, joy, love, sadness, and surprise. The dataset consists of 20,000 tweets, and their corresponding labels. The dataset was already split into train, validation, and test sets. The training dataset has 16,000 tweets, meanwhile the validation and test datasets each have 2,000 tweets.

IV. Data Analytic Strategy

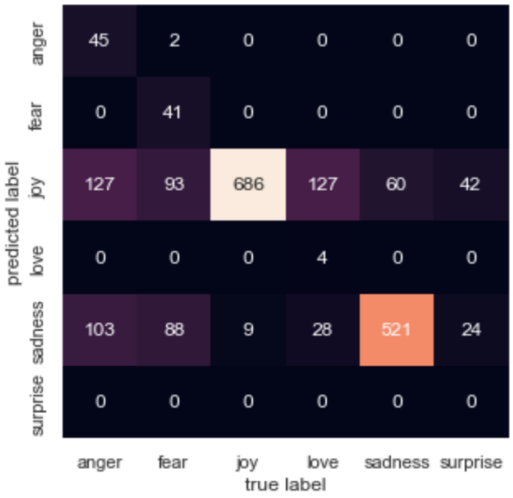
Several machine learning models were built and evaluated on the test dataset. Our goal was to determine which model performed the best and provided the highest accuracy percentage. Our baseline models included Naive Bayes and a simple Neural Network architecture. The other mothers built were an Upgraded Neural Network, a Neural Network with Glove Embeddings, and a Bidirectional Encoder Representations from Transformers (BERT) model. We built the baseline models to get an idea of how well they performed on the dataset, and then moved on to more complex models and compared the results.

V. Results

Before we started experimenting with different models, we first had to clean the data. The dataset was already formatted, which made it easier for cleaning. Every line contained a full

tweet with its associated emotion separated by a semicolon character. We wrote a function to read and process the data files. However, since the data files were already split into two different files for testing and training, there was no need to split the dataset into subsets. Next, we used the TFIDFVectorizer and MultinomialNaiveBayes modules from scikit-learn to train a Naive Bayes model on the training dataset. Using the model we created predictions for the test set. We used the seaborn package to plot a confusion matrix graphic that looked like this:

Confusion Matrix: Naive Bayes

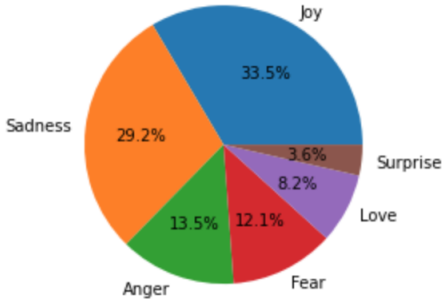


The confusion matrix shows that Naive Bayes overall did not perform very well. We can see that it predicted tweets with the sadness emotion with 86% accuracy, and it mis-predicted the other 14% as joy. Although the Naive Bayes model predicted the joy emotion with 98% accuracy, tweets involving other emotions (anger, fear, love and sadness) were mis-predicted as joy. Evidently, the model seems to favor the emotion joy, as most of the tweets in the test set were labeled with that emotion. On the other hand, the model performed very poorly with predicting anger, fear and surprise. For the anger emotion, the model was only correct 16% of the time, and it seemed to favor joy and sadness for tweets that were actually supposed to invoke anger. Furthermore, the same outcome occurred with the fear emotion, our Naive Bayes model was only 18% accurate, mis-labeling such tweets with emotions of joy and sadness. The model performed the worst when predicting surprise at 0% accuracy. However, surprise had the lowest sample size in the test data with only 66 tweets.

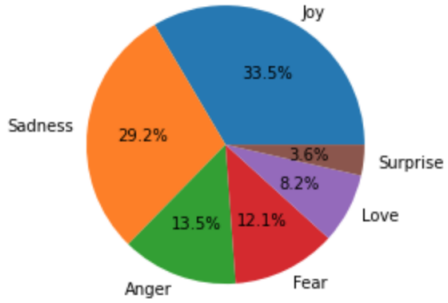
The model's predictions had higher accuracy with emotions of joy and sadness. In fact, we can see that there is a clear relationship between sample size and prediction accuracy, since

these emotions had the highest sample counts (>500). Anger, fear, and love performed much worse and had under 200 sample counts in the test set. However, the emotion of surprise performed the worst out of all emotions. Evidently, the greater the sample size of an emotion, the better prediction it provided for that specific emotion. This trend also correlates to the sample counts of each emotion in the training dataset. There were ~2000 tweets classified as anger, ~2000 for fear, ~5000 for joy, ~1000 for love, ~4500 for sadness and ~500 for surprise. In fact, the class imbalance greatly impacted the prediction accuracy of the Naive Bayes model. The pie charts below display the class distribution in our train and test sets.

Class Distribution: Train Dataset



Class Distribution: Test Dataset



We suspect that there is some overfitting for the joy emotion. With a greater sample size, the model became 'too good' at predicting joy with a 98% accuracy, but it mis-classified many other tweets as joy. Additionally, we calculated the micro-averaged F1 score and the macro-averaged F1 score for our Naive Bayes result.

```
print("Microaveraged F1 score for category predictions " + str(micro_cat))
print("Macroaveraged F1 score for category predictions " + str(macro_cat))
```

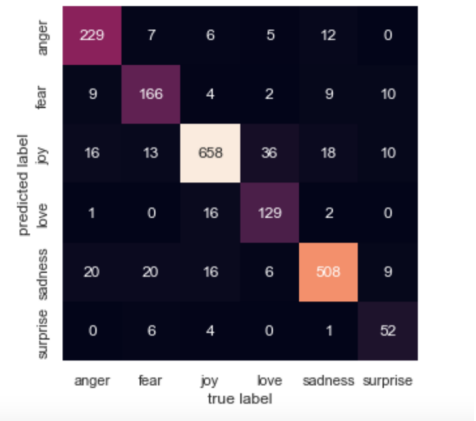
```
Microaveraged F1 score for category predictions 0.6485
Macroaveraged F1 score for category predictions 0.35955253966635575
```

Our macro-averaged F1 score is much lower than our micro-averaged score. Since our experiment is a multi-classification problem with differing data counts over our classes, the macro-averaged F1 score holds less importance than the micro-averaged score. The reason being that the macro-average F1 score will take the average of the scores across all classes, which implies that each class is balanced with every other class. However, this is not the case for our dataset, since some of our classes had many data points (>5000), while some had very little data

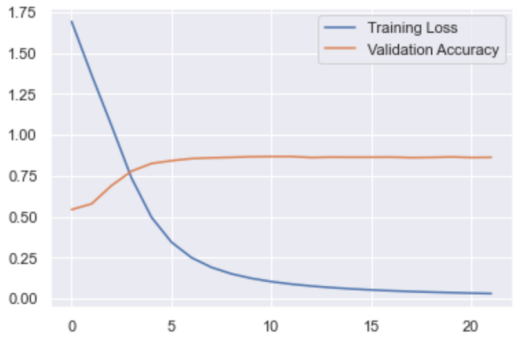
points (<500). Overall, Naive Bayes was a good baseline model, as it provided a good accuracy for some of the emotions in the dataset.

Our second baseline model was a Simple Neural Network. For this architecture, our inputs were TF-IDF vectors of our words and our output was a chosen class of 6 possible outcomes. Our model had one hidden layer with fifty units, and it used the relu activation function. The training loss vs. validation accuracy and a confusion matrix of our results is shown below.

Confusion Matrix: Simple Neural Network



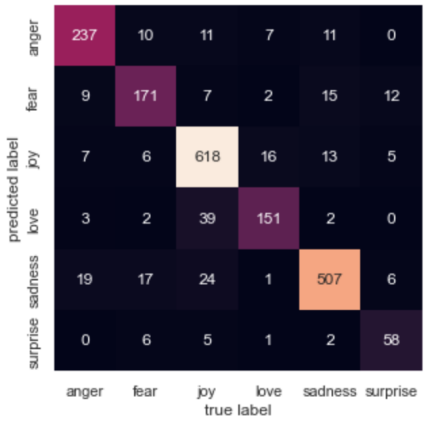
Training Loss VS. Validation Accuracy: Simple Neural Network



Surprisingly, the simple neural network performed quite well. This model had a micro-averaged F1 score of 0.87 and a macro-averaged score of 0.83. Our validation accuracy reached its max around epoch 5 (5th iteration) of training, and it did not improve from there. However, the training loss continually decreased through the iterations. This simple Neural Network had a training accuracy of 98% and a validation accuracy of 87%, which resulted in a training error of around 2% and a validation error of around 13%. As a result of this, our model had a higher variance and some overfitting. Ideally, we wanted our validation accuracy to be higher before we can conclude that there is low variance. However to address the overfitting, we tried a different, more complex Neural Network architecture, where we added a regularization parameter.

The Upgraded Neural Network model had four hidden layers with 100 units each. We kept the relu activation function, and added a regularization parameter to help with the overfitting from the previous Neural Network model. The results for our upgraded neural network are shown below.

Confusion Matrix: Upgraded Neural Network

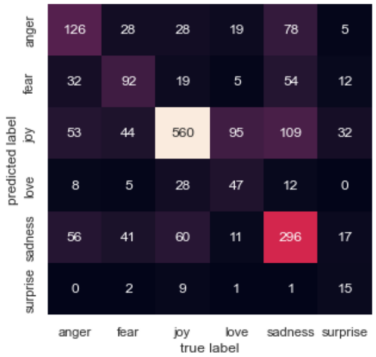


Training Loss VS. Validation Accuracy: Upgraded Neural Network

Expanding the architecture gave us very little change in the results. The training accuracy and validation accuracy were only slightly better (98.4% & 87.1%) compared to the results from the first model. The confusion matrix shows that there is still some possible overfitting of the model towards the joy and sadness emotion as many of the mis-classified sentences are being predicted as joy or sadness. Although this model did not give us the level of accuracy we were looking for, it was still a good experiment to see if we could improve our results from it. The previous two neural network models were using TF-IDF vectors of words as inputs so for our last neural network model we kept the same architecture but applied Glove word embeddings and kept an out of vocabulary dictionary using random vectors for unknown words.

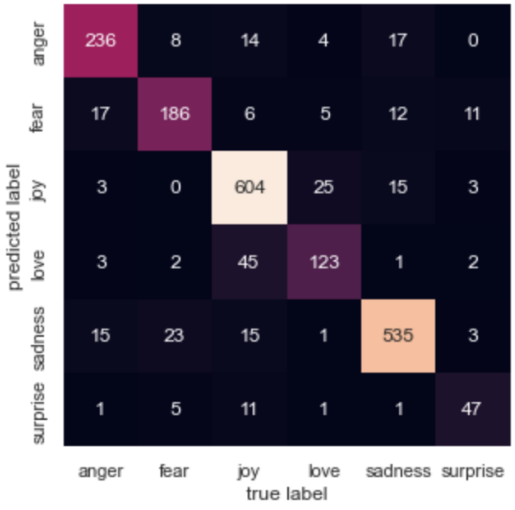
For our third Neural Network, we used the same number of layers, units and the regularization term from the Upgraded Neural Network. However, the training and validation accuracy for the Neural Network With Glove embeddings decreased significantly. Our results are shown below.

Confusion Matrix: Neural Network With Glove



Our validation accuracy was only 56.8% with word embeddings. One of reasons we suspected that caused the much lower accuracy was that the word embeddings may not have been a good fit for the words in our dataset. The sentences in our dataset were originally tweets from Twitter and a lot of the language and shorthand slang commonly used on Twitter would not exist in a dictionary or normal word embedding set. We believe a set of word embeddings focused on social media language that includes "Twitter-speak" would better fit the vocabulary in our dataset. After applying the word embeddings to our training set, we checked the size of our Out-Of-Vocabulary (OOV) dictionary and it had ~1100 entries. Our training dataset had 15,212 unique words, and the OOV dictionary contained about 8% of our training vocabulary. We inspected the values in the OOV dictionary and saw a lot of common misspelled words, 2+ words that were combined into one big word, many texting/social media abbreviations such as 'idk' and 'nvm' and a few words in different languages were used. Since the Upgraded Neural Network had the best performance out of all the Neural Networks we employed, we applied that model to predict the test dataset. The test performance results are shown below.

Confusion Matrix: Upgraded Neural Network

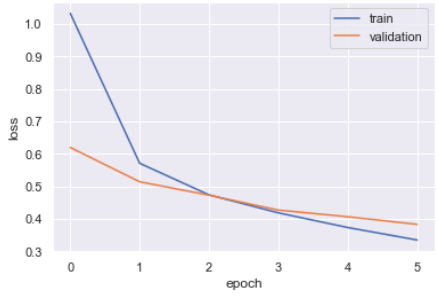


The test set results were similar to the validation set scores, since the Upgraded Neural Network had an 86.5% accuracy rate and the macro-averaged F1 score was 82%. Overall, the neural network models gave very good results and the high accuracy compared to the Naive Bayes architecture.

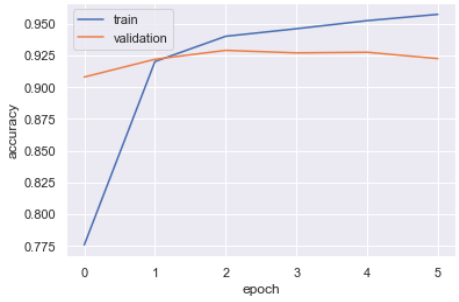
The final model we tried on the dataset was the DistilBERT architecture. Since our dataset was already split into train and test, we just had to upload the separate datasets and define our X_train, y_train, X_test, and y_test. The emotions (anger, fear, joy, love, sadness, and surprise) were our y, or target, values and we had to convert these categorical values into dummy/indicator values. We used DistilBertTokenizerFast to tokenize the text, and set the maximum number of words to tokenize to 218. We utilized the self-defined function *batch_encode* to encode text data in batches. With this function, we created the X_train_ids, X_train_attention, X_test_ids, and X_test_attention.

When defining the model architecture, we set the dropout and attention dropout to 0.2. Next, we created a self-defined function called *build_model,* and set the learning rate to 5s-5 and random state to 42. Using this function, we created the DistilBERT model. We trained the model by setting the classification layer weights EPOCHS = 6 and BATCH_SIZE = 64. Per every epoch, the loss and validation loss decreased, and the accuracy and validation accuracy increased. Then, we tuned and applied the model to the test set; FT_EPOCHS = 6 and BATCH_SIZE = 6. Overall, the accuracy of the DistilBERT model on the test dataset was 92.3%, which is the best accuracy out of all the models we trained. We plotted the training loss vs. validation loss, and training loss vs. validation accuracy as displayed below.

Training Loss VS. Validation Loss: DistilBERT



Training Loss VS. Validation Accuracy: DistilBERT



VI. Conclusion

This text emotion recognition project employed baseline models: Naive Bayes and a Simple Neural Network, as well as an upgraded Neural Network, a Neural Network with Glove Embeddings, and a Bidirectional Encoder Representations from Transformers (BERT) model.

For starters, the Naive Bayes overall did not perform very well, since our dataset had imbalanced classes, which resulted in predicting higher accuracy for emotions with a greater sample size like joy and sadness. On the other hand, the Simple Neural Network, had one hidden layer with fifty units, and it used the relu activation function. This model had a validation accuracy of 87%, but it had higher variance and some overfitting. The Upgraded Neural Network model had four hidden layers with 100 units each. We kept the relu activation function, and added a regularization parameter. However, the validation accuracy was only slightly better at 87.1%.

For the Neural Network With Glove embeddings, our validation accuracy decreased significantly because the Out-Of-Vocabulary (OOV) dictionary had ~1100 entries, about 8% of our training vocabulary. Since the Upgraded Neural Network had the best performance out of all the Neural Networks we employed, we applied that model to predict the test dataset, which resulted in 86.5% accuracy. In conclusion, the DistilBERT model provided best results compared to Naive Bayes and the Neural Networks, with a 92.3% accuracy on the test data. Overall, we believe emotion classification is very possible to accomplish with highly accurate results, however having a good training set is necessary. Our results could be improved with a larger training set, and with a balanced amount of classes. Moreover, a word embeddings set for social media and shorthand texting language would have performed better, since all of our data was collected from Twitter. This project was a great learning experience for us because we learned how to apply NLP techniques, which we learned throughout this semester, to a problem that has real-life applications in today's day and age.

VII. References

[1] B. Liu, *Sentiment analysis and subjectivity,* in Handbook of Natural Language Processing, N. Indurkhya and F. J. Damerau, Eds., 2nd ed. Boca Raton, FL, USA: CRC Press, 2010. https://www.cs.uic.edu/~liub/FBS/NLP-handbook-sentiment-analysis.pdf

[2] B. Liu, Sentiment Analysis: *Mining Opinions, Sentiments, and Emotions*, 1st ed. New York, NY, USA: Cambridge Univ. Press, 2015. https://www.cs.uic.edu/~liub/FBS/SentimentAnalysis-and-OpinionMining.pdf

[3] Zhang, S., Zhang D., Zhong H., Wang G., *A Multiclassification Model of Sentiment for E-Commerce Reviews,* Volume 8, 2020.

https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9226475

[4] *Emotion Dataset for NLP*.

https://www.kaggle.com/praveengovi/emotions-dataset-for-nlp