Tweet Emotion Analysis

Applied Machine Learning

Rohan Mallick 19115103
Aviral Garg 19115036

May 2022

IIT Roorkee
Instructor: Prof. G.N. Pillai

Abstract

Emotions reflect different user perspectives towards actions and events; therefore, they are expressed in dynamic linguistic forms. Emotion Detection is similar to Sentiment Analysis but also quite different. Emotion Analysis is used to detect different types of feelings from the expression of text, such as anger, love, fear, happiness, sadness, and surprise (emotions) whereas Sentiment Analysis is used to detect only positive, neutral, or negative feelings (sentiments) from text. Studying how to understand emotions can be a tough task, so technology is used to do the job. The main purpose of this work is to introduce a way of recognizing emotions from the text. Thus, the Twitter Emotions dataset is analyzed in detail in this work, and a multi-class classification model is developed to differentiate the six wide varieties of emotions with post-analysis details. The LSTM classification model has the highest accuracy results. There exists a probable future scope to extend similar techniques to sentiments in unsupervised settings.

1. Introduction

Detecting a person's emotional state by analyzing a text document written by him may appear difficult, but it is necessary because textual expressions result not only from the direct use of emotional words, but also from the interpretation of the meaning of concepts and the interaction of concepts described in the text. In human-computer interaction, analyzing the emotion in the text plays a crucial role. Joy, sadness, anger, surprise, love, and fear are the main emotions expressed by humans as studied in psychology. The state of being emotional is often aligned with making conscious arousal of feelings subjectively or with influence from the environment; thus, emotions are derived from the personal experiences of individuals and as well as their interactions with their surroundings. Emotions play vital roles in people's lives and overall makeup. They provide observers with information regarding the current state and well-being of an individual. The field of Emotional Detection can also be applied in a wide variety of applications such as emotion analysis of suicide notes, multimedia tagging and detecting interesting or unwanted behaviours in a conversation. While detecting emotions from voice, speech, images, videos and other multimodal methods have a huge knowledge base, there is a great sparsity in text research. One of the reasons can be attributed to the fact that unlike multimodal methods, texts may not portray direct signals to the emotions. Also, the challenge of detecting emotions from short messages, emojis, emoticons and grammatical errors could be crippling coupled with the ongoing evolution of new words as a result of language dynamics.

1.1 Related Work

There has been extensive work on sentiment analysis using deep learning but comparatively less in the field of emotion analysis. Several works use handcrafted features and statistics-based approaches to train emotion recognition models (Blitzer et al., 2007; Becker et al., 2017; Saravia et al., 2016a). Some of these studies rely on affect lexicons, such as LIWC (Pennebaker et al., 2007) and WordNet Affect (Strapparava et al., 2004), to extract emotion features from a text-based corpus. A recent study trained emotion detection systems built on emoticons and hashtag features (Volkova and Bachrach, 2016). Handcrafted features are useful for emotion recognition but are usually constrained by manually created resources.

There are several effective datasets, such as SemEval-2017 Task 4 (Rosenthal et al., 2017) and the Olympic games dataset (Sintsova et al., 2013). However, these datasets are limited by quantity. In emotion recognition studies, Plutchik's wheel of emotions (Plutchik, 2001) or Ekman's six basic emotions (Ekman, 1992) are commonly adopted to define emotion categories (Mohammad, 2012; Suttles and Ide, 2013).

Recent emotion recognition systems employ representation learning for automatic feature extraction (Poria et al., 2016; Savigny and Purwarianti, 2017; Abdul-Mageed and Ungar, 2017). In general, a combination of word embeddings (Mikolov et al., 2013) and a convolutional neural network (CNN) performs well for sentence classification tasks (Kim, 2014; Zhang et al., 2015). These models learn features that tend to have high coverage, high adaptability, require minimum supervision, and capture contextual information to some extent.

1.2 Applications

1.2.1 E-commerce

Users usually publish personal comments on products purchased on e-commerce platforms. Using this source of product comments, real-time emotional analysis is conducted to obtain useful emotional and behavioral consumer features (consumer feedbacks), enabling the prediction of trend changes in consumer preferences.

1.2.2 Public opinion analysis

Public opinion analysis is used to objectively reflect the state of public opinion on the internet by collecting and sorting out people's attitudes as well as discovering relevant opinion tendencies.

1.2.3 Big search

Big search can understand user search intentions on a semantic level while also perceiving user needs according to their spatio-temporal location, emotional state, and historical preferences.

1.2.4 Healthcare

According to text recorded from social networks on psychological counseling, the emotional state information of patients suspected to be depressed was analyzed and screened using emotion analysis technology. Yang et al. used the disease analysis interview corpus from the University of Southern California to analyze the patients with poor mental health (e.g Those diagnosed with anxiety or PTSD) for depression.

2. Methods

2.1 Requirements

All the required libraries required in the dataset analysis are imported. Some of them include sklearn, nltk, TensorFlow, matplotlib, etc. We use sklearn for Machine Learning models and Tensorflow for Deep Learning models.

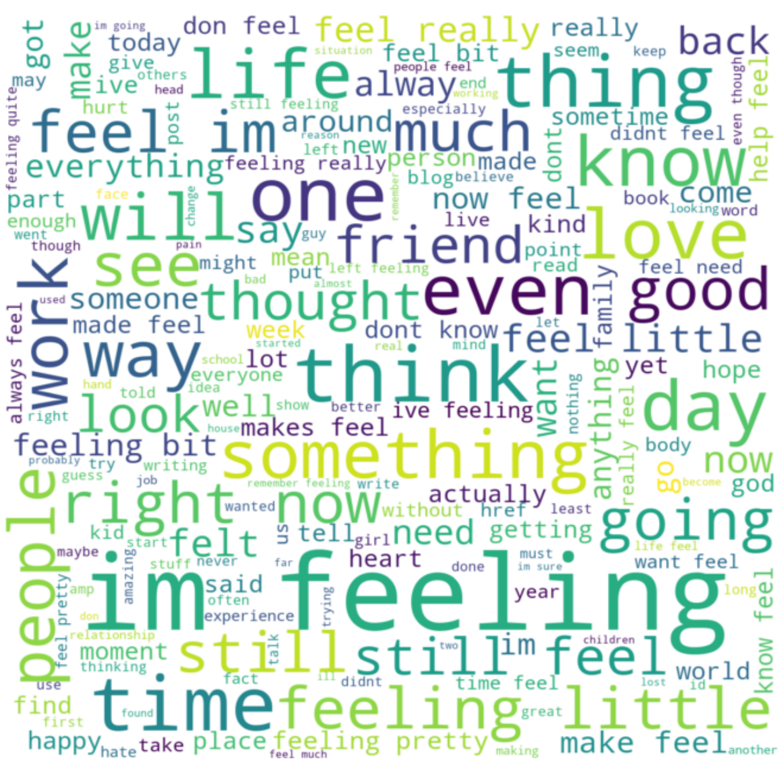
2.2 Preparing the dataset

After importing the necessary libraries,

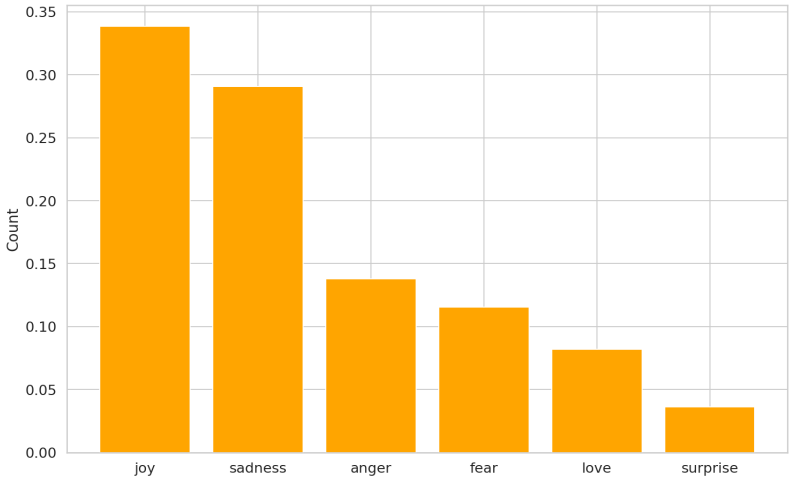
- The data is imported from the dropbox using the pickle directory.
- Then the dataset is saved in both NumPy array and pandas data frame format.
- Data is then cleaned with default settings such as lemmatization, stemming, lower, no punctuation, and no digit.
- Then, the data is tokenized and converted to embedding either in TFIDF or Word2Vec.
- Then, the data is normalized both along batch and feature dimensions.
- Finally, the dataset is split for model training and testing with a test split of 0.1.

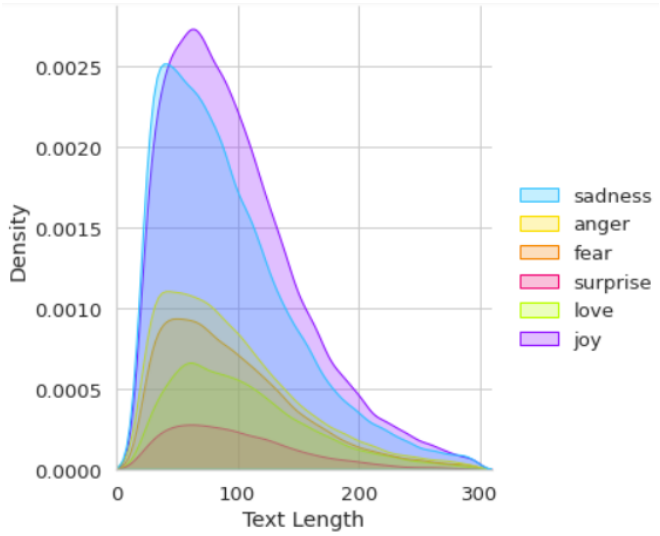
2.3 Dataset Analysis

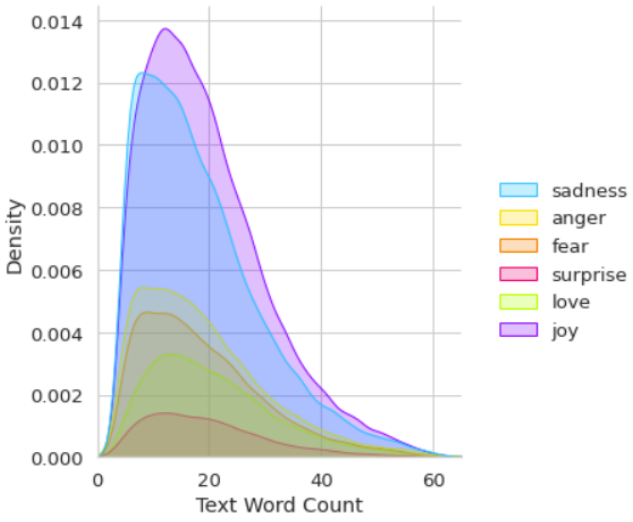
- We use the CARER: tweet emotion dataset (Saravia et al., 2018). It contains 400k text-emotion pairs and six emotions i.e. joy, sadness, anger, surprise, love, and fear.
- Using WordCloud: It is a data visualization technique used to draw plots representing the text data. The size of each word in the plot indicates the frequency or one can say the importance of that word in our analysis. They are used to analyze the way people react on social media sites and how they review stuff on shopping websites.



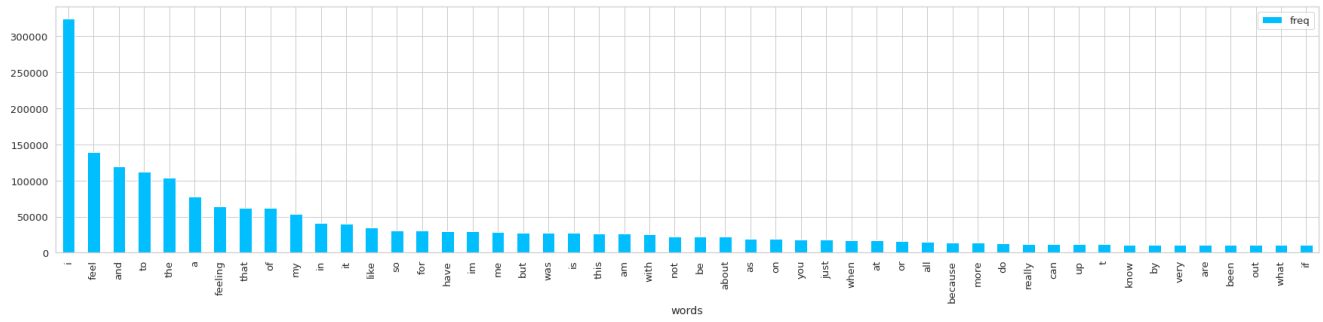
- Word, Label, and Character count: The number of words and character count and their densities is plotted. The number of tweets related to a specific emotion is also analyzed in the histogram given below.







- Frequency of words: The most frequent words in our tweets without any cleaning or normalization.



2.4 Cleaning the dataset

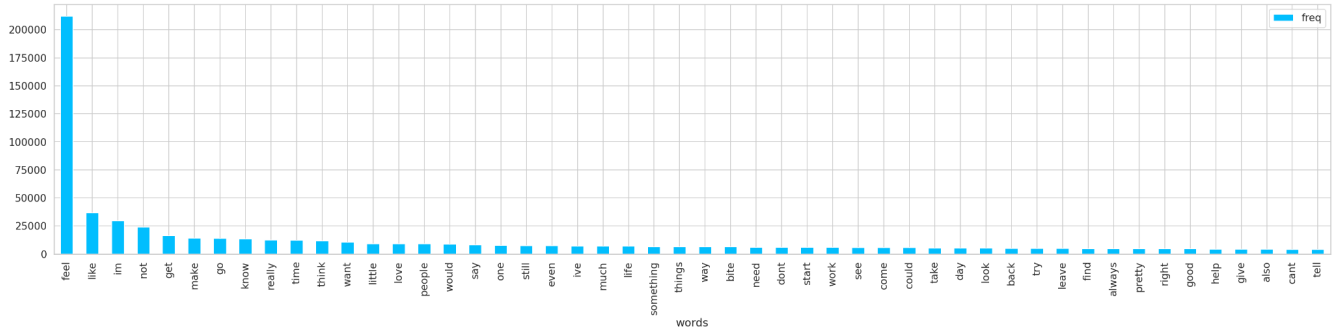
The data is first cleaned using the regex library's replace function. The data is cleaned and all special characters, numbers, and URLs are removed. After that, all the words in uppercase are converted into lower case letters using tokenization. Then, we remove stopwords, punctuation, and words with length less than 2. All the words are then joined and a new word cloud is formed. After this, we stemming and lemmatization techniques to further clean the data. Stemming uses the stem of the word to analyze the data and lemmatization uses the context in which the word is used. Lemmatization is generally more accurate than stemming. Dictionaries are lemma lists, not stem lists. In stemming, the beginning of a word or the ending of a word is cut and the stem is obtained. That is why it has some limitations. Whereas, lemmatization takes into consideration the morphological analysis of words.

Before lemmatization

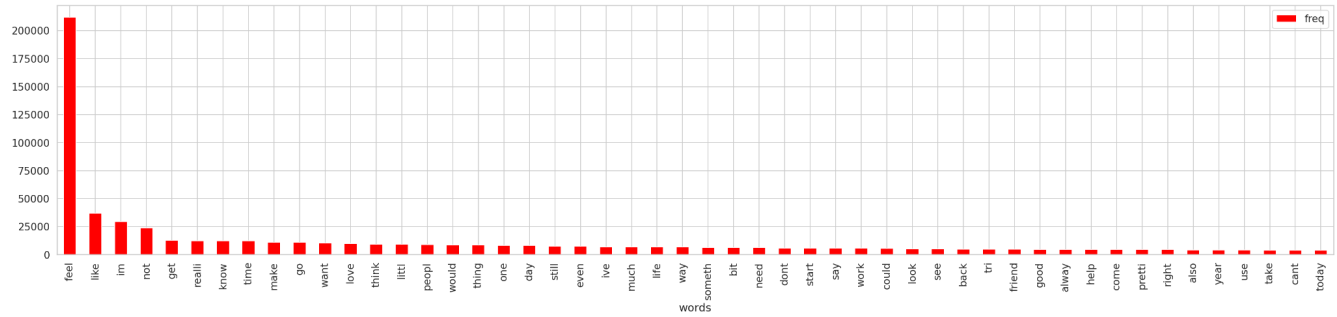
After lemmatization

2.5 Frequency Distribution of Top 50 Words

- **Using LEMM:**



- **Using STEM:**



2.6 Embedding and Tokenization

- **TFIDF**

  Term Frequency Inverse Document Frequency is defined as the calculation of the relevance of a word in a corpus to a text. The value increases proportionally to the number of times the word appears in the text but is inversely related to the word frequency in the corpus. We use TFIDF to convert sentences into sentence vectors.

- **Word2Vec**

  Word2Vec creates word embeddings vectors of the words. Word embeddings eventually help in establishing the association of a word with another similar meaning word through the created vectors. Similar meaning words are closer in space, indicating their semantic similarity.

2.7 Data Visualization

For data visualization, we use a few feature transformation techniques to reduce the feature dimensions.

- **Principal components analysis:**

  TFIDF features: The variances retained by the first three principal components are Principal component 1: 1.915396 %, Principal component 2: 1.460563 %, and Principal component 3: 1.388614 %. 400 out of 500 dimensions explain 90% of the variance. 135 out of 500 dimensions explain 50% of the variance.

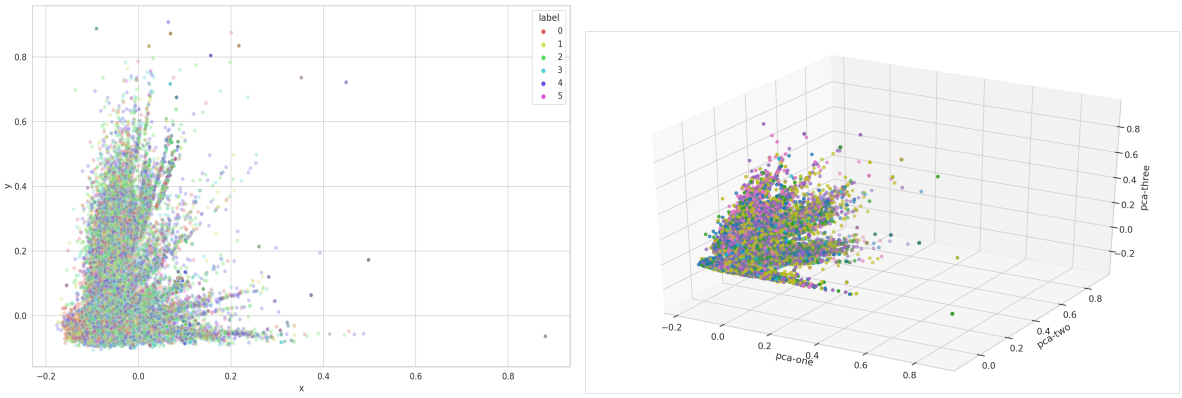
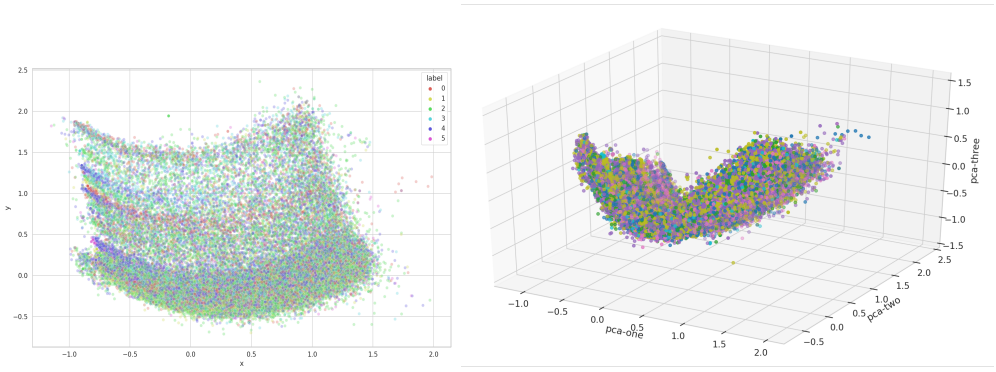


Figure: Dimensionality reduction using PCA with n=2, 3 (TFIDF: above & word2vec: below)



- **t-SNE:**

  T-distributed Stochastic Neighbor Embedding is a non-linear dimensionality reduction technique used for visualization of high-dimensional datasets. We use this method to reduce the dimensionality to two.

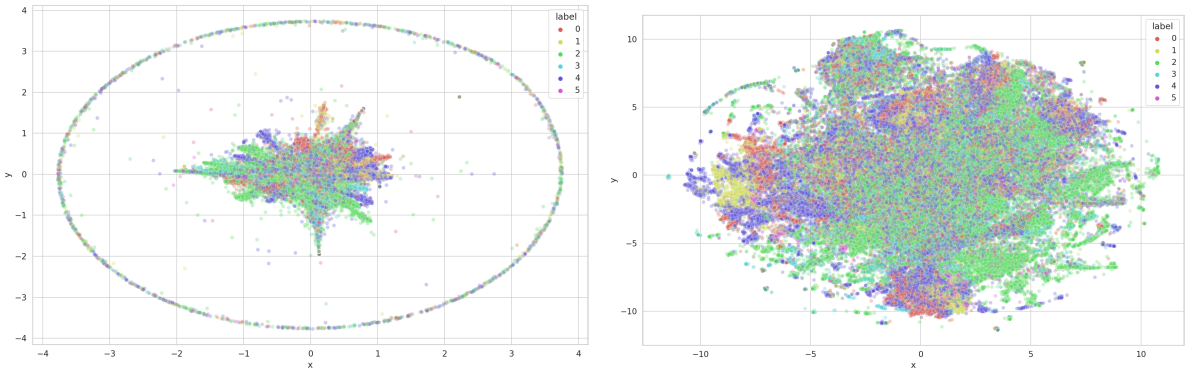


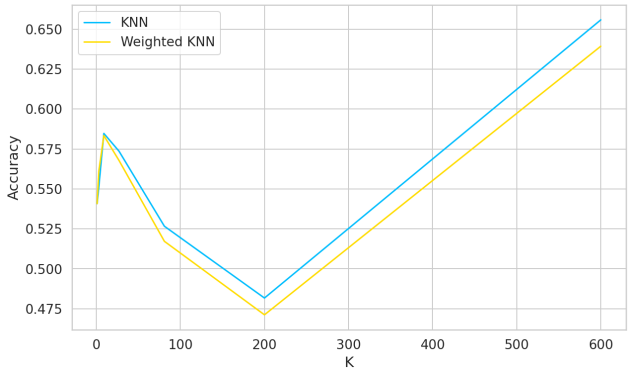
Figure: Dimensionality reduction using t-SNE with 2 features (TFIDF: left & word2vec: right)

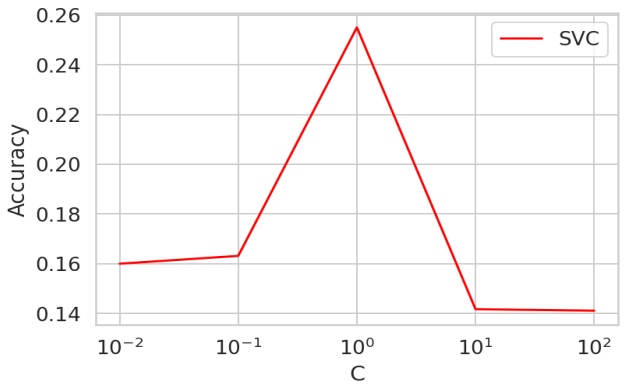
2.8 Developing classification model

We experiment with multiple classifiers including Machine learning algorithms and deep learning algorithms. For Machine learning algorithms like Logistic Regression, ANN, SVM, KNN, Decision Tree, Random Forest, Ada Boost, XGBoost, Naive Bayes (Non-temporal models) we use the Tf-Idf feature encoder. And for temporal models like RNN, LSTM, and 1D CNN we use word2vec embeddings which are trained on the training dataset itself.

2.9 Ablations

We do an ablation on K of K Nearest Neighbor and Weighted K Nearest Neighbor also C of SVC over validation accuracy. We find k=27, C = 1 to be optimum respectively.





2.10 Post Evaluation

We use the validation and test set to calculate the following metrics for each model.

- **Confusion Matrix**

  It is a table of predicted vs actual number of instances of each class. Here we use the logarithm of the normalized number of instances. (along true label)

- **ROC curve**

  A graph showing true positive rate vs false positive rate achieved on different values of probability thresholds. It is a tradeoff between sensitivity (tpr) and specificity (1-fpr). Helps to choose an optimal cutoff (threshold). An ideal model must have 100% tpr and 0% fpr.

- **Precision-Recall curve**

  A graph showing precision vs recall achieved on different values of probability thresholds. It is a tradeoff between precision and recall (tpr). Helps to choose an optimal cutoff/threshold. An ideal model must have 100% precision and 100% recall.

- **Accuracy**

  Fraction of predictions which models got right. Balanced accuracy isn't affected by the number of instances of each class.

- **F1 score**

  It combines precision and recall into a single score i.e. it is the harmonic mean of both the precision and Recall.

- **Precision**

  The proportion of positive predictions that was actually correct.

- **Recall**

  The proportion of positive labels that were predicted correctly. Also called the True Positive Rate.

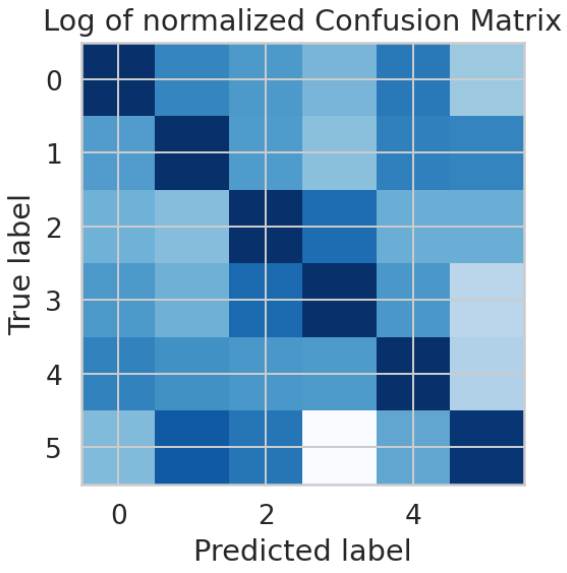


Fig: Sample Confusion Matrix

3. Analysis

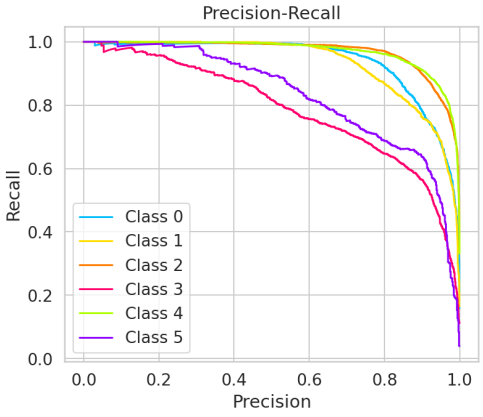
Classifier used	Embedding	No. of features	Hyperparameters	Accuracy %	Macro F1%
SVC	TFIDF	500	Kernel: RBF, C=1	35	9
Random Forest	TFIDF	500	Trees = 100, Max-Depth = 7	35	12
Decision Tree	TFIDF	500	Max-Depth = 15	39	26
Naive Bayes	TFIDF	500		51	52
K Nearest Neighbors	TFIDF	500	K = 40	56	51
AdaBoost	TFIDF	500	Trees = 100 LR = 1.0	60	58
XGBoost	TFIDF	500	Trees = 100	67	64
Logistic Regression	TFIDF	500		68	64
Neural Network	TFIDF	500	500->100->6 LR = 0.001	69	66
RNN	Word2Vec	35 (40 words)	RNN (100) -> 6	79	68
LSTM	**Word2Vec**	**35 (40 words)**	**LSTM (100) -> 6**	**86**	**82**
1D CNN	Word2Vec	35 (40 words)	CNN (100) -> Pool (2) -> CNN (25) -> Pool(2) -> 20 -> 6	87	71

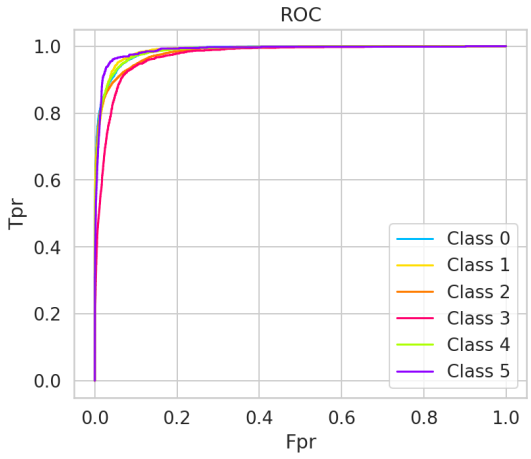
Table: Scores for all models

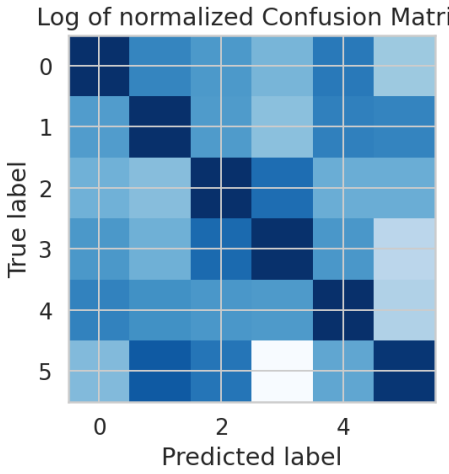
We find out that LSTM and Logistic Regression (in ML) have the best classification performance wrt the validation Macro f1 score.

- **LSTM & RNN**

LSTM and RNN are generalized neural networks with memory and recurrence.
We find out that LSTM performs very well in most of the classes. With slight decrement in metrics for classes with low number of instances (such as sadness, surprise). But anyway, LSTM learns well in all the classes. RNN has the same characteristics but with slightly lower metrics due to lower complexity.







	precision	recall	f1-score	support
anger	0.86	0.85	0.86	2732
fear	0.88	0.81	0.84	2278
joy	0.89	0.92	0.90	6770
love	0.74	0.71	0.72	1685
sadness	0.91	0.92	0.91	5832
surprise	0.72	0.76	0.73	703
accuracy			0.87	20000
macro avg	0.83	0.83	0.83	20000
weighted avg	0.87	0.87	0.87	20000

roc_auc = 0.9852271455476398

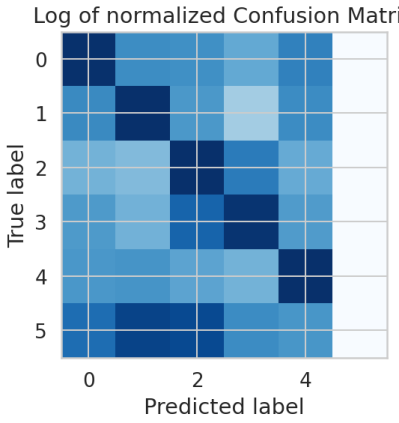
Fig: Metrics for LSTM

- **1D CNN**

1D CNN is a convolutional network which convolutes in the temporal dimension.
We find out that 1D CNN performs very well in most of the classes. But even though it has high accuracy, it is not able to classify some of the classes with low number of instances (such as surprise) i.e. it could not learn the structure of these classes. We can see this mishap clearly in the precision-recall curve. It is best to use balanced accuracy (normalized wrt to class size) to get the real picture.

	precision	recall	f1-score	support
anger	0.87	0.87	0.87	2763
fear	0.74	0.91	0.82	2198
joy	0.88	0.93	0.90	6883
love	0.77	0.71	0.74	1600
sadness	0.93	0.92	0.92	5825
surprise	0.00	0.00	0.00	731
accuracy			0.86	20000
macro avg	0.70	0.72	0.71	20000
weighted avg	0.84	0.86	0.85	20000

roc_auc = 0.9812852965019133



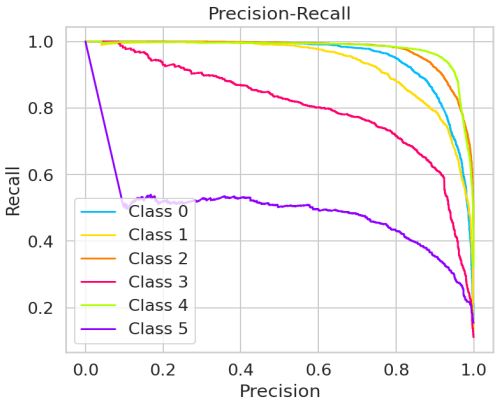
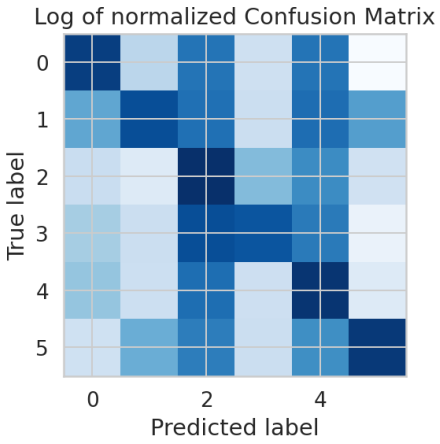
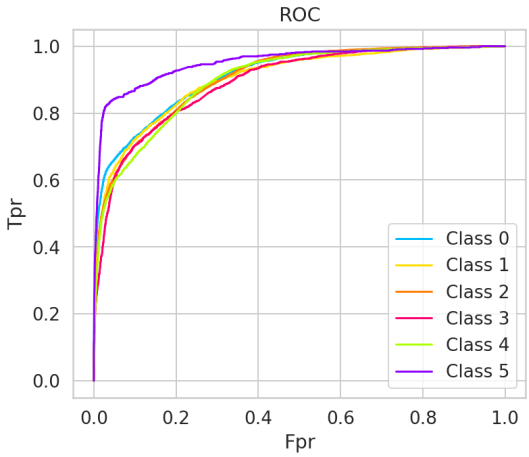


Fig: Metrics for 1D CNN

- **Logistic Regression**

Logistic regression models data with a logistic model (sigmoid and a perceptron).
We find out that Logistic Regression performs very well in all of the classes. Every class has a similar metric. But it is miles apart from the temporal deep learning approaches. (LSTM, RNN).
A reason why TFIDF models don't perform at par could be the lack of temporal structure of the sentence and thus losing significant information.





	precision	recall	f1-score	support
anger	0.80	0.60	0.68	2712
fear	0.83	0.43	0.57	2289
joy	0.65	0.82	0.73	6819
love	0.63	0.37	0.47	1642
sadness	0.66	0.73	0.70	5780
surprise	0.64	0.67	0.66	758
accuracy			0.68	20000
macro avg	0.70	0.60	0.63	20000
weighted avg	0.69	0.68	0.67	20000

roc_auc = 0.9103629148340286

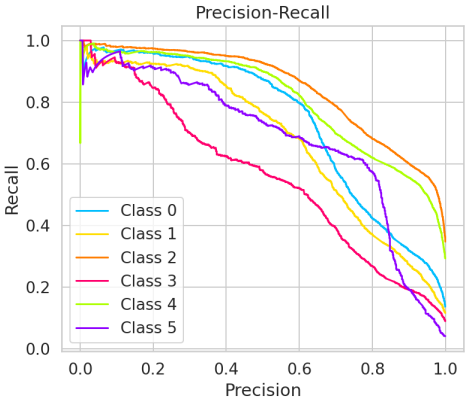


Fig: Metrics for Logistic Regression

- **Multi Layer Perceptron (ANN)**

  MLP is a connection of multiple perceptrons in multiple layers.
  We find out that ANN performs best in all of the classes. Every class has a similar metric. But it is miles apart from the temporal approaches. (LSTM, RNN)

- **Ada Boost & XGBoost**

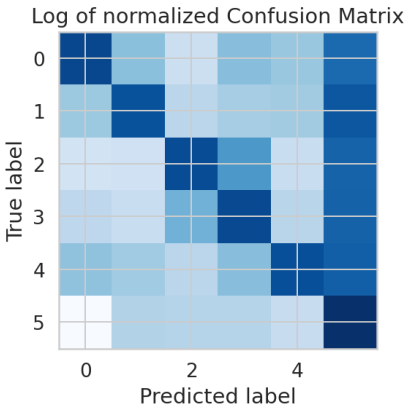
  Boosting models involve training sequential classifiers with adaptive or gradient boosting.
  Boosting models such as AdaBoost and XGBoost perform fairly well.

- **Naive Bayes & K Nearest Neighbor**

  Naive Bayes and KNN perform fairly well. But due to naivety and simple nature they fail to achieve higher metrics.

	precision	recall	f1-score	support
anger	0.77	0.57	0.65	2712
fear	0.69	0.46	0.55	2289
joy	0.91	0.51	0.65	6819
love	0.41	0.54	0.47	1642
sadness	0.89	0.49	0.63	5780
surprise	0.09	0.90	0.17	758
accuracy			0.52	20000
macro avg	0.63	0.58	0.52	20000
weighted avg	0.79	0.52	0.60	20000

roc_auc = 0.8603228322996842



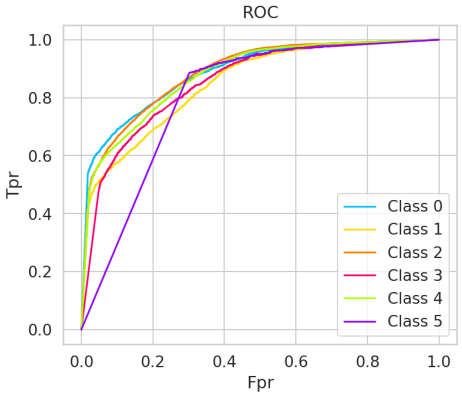


Fig: Metrics for Naive Bayes

- **Random Forest & Decision Trees**

  Random forest is a bagged ensemble of Decision Trees. Decision Trees are tree-like structures with a decision at each node.
  Simple Tree structures like Decision Trees and Random Forest fail to perform i.e. learn the structure of data. All metrics hover around naive dummy scores.
  A reason why trees and SVM fail could be that there are exponential permutations of words and they would take a long time to learn. Whereas, linear (additive) models such as logistic regression, ensembles etc already have the inductive bias of linearity and can easily learn the additive nature of word-word relationship in a sentence.

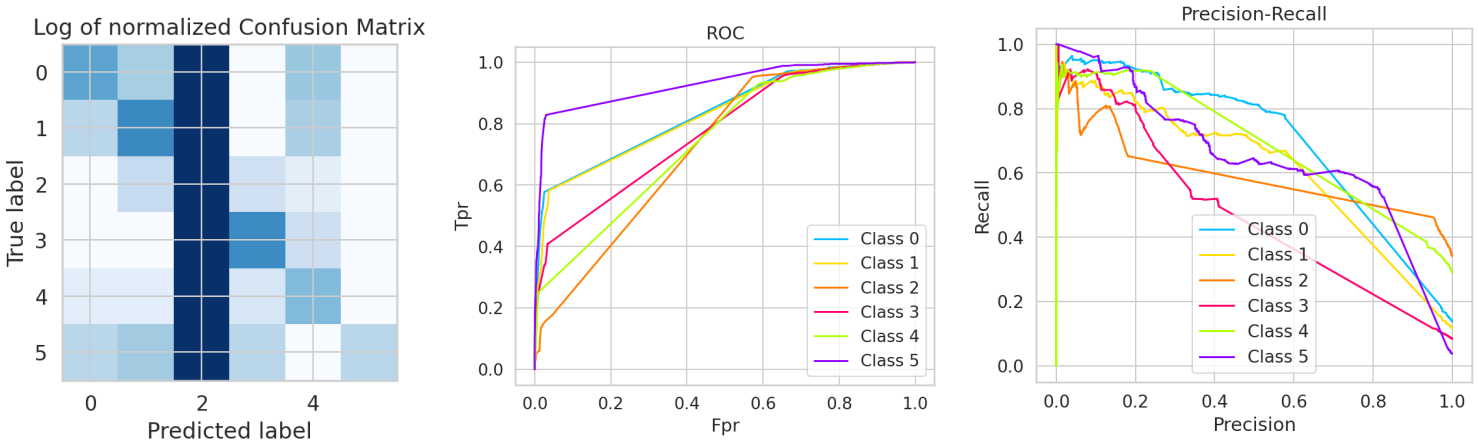


Fig: Metrics for Random Forest

- **Support Vector Classifier**

  SVM algorithm finds a hyperplane and related support vectors which maximize the margin.
  SVC fails to perform and learn the structure of data. All metrics are no more than dummy heuristics.

- **General Observation**

  We find that most misclassifications are classified as classes with max number of instances. Hence, this class has the max recall.

4. Conclusion and Future scope

Thus, we have successfully developed a classifier for tweet emotional analysis. The tweets have been classified into basic emotion categories using various techniques and models. We find out various insights into the dataset and the merits and shortcomings of various models. Finally, we found out that LSTM and Logistic Regression have the best validation Macro f1 score.

There exists a possible scope of expanding our knowledge and techniques and further expanding our project by analyzing the sentiments of tweets and further connecting sentiments with emotions and improving the accuracy of our model. Further, the system can also be extended to consider various emojis and gifs and stickers, and images in analyzing the sentiments and emotions of the user.

5. References

[1] Saravia, Elvis & Liu, Hsien-Chi & Huang, Yen-Hao & Wu, Junlin & Chen, Yi-Shin. (2018). CARER: Contextualized Affect Representations for Emotion Recognition. 3687-3697. 10.18653/v1/D18-1404.

[2] Peng, S., Cao, L., Zhou, Y., Ouyang, Z., Yang, A., Li, X., Jia, W., & Yu, S. (2021). A survey on Deep Learning for textual emotion analysis in social networks. *Digital Communications and Networks*. https://doi.org/10.1016/j.dcan.2021.10.003

[3] Y, C.-Y., Chew-Yean Yam Follow, Yam, C.-Y., 4, A. T. D., T, A., 16, J. P. D., & P, J. (2020, March 15). *Emotion detection and recognition from text using Deep Learning*. CSE Developer Blog. Retrieved May 12, 2022, from https://devblogs.microsoft.com/cse/2015/11/29/emotion-detection-and-recognition-from-text-using-deep-learning/

[4] Hochreiter, Sepp & Schmidhuber, Jürgen. (1997). Long Short-term Memory. Neural computation. 9. 1735-80. 10.1162/neco.1997.9.8.1735.

[5] Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, *20*(3), 273–297.

[6] van der Maaten, Laurens & Hinton, Geoffrey. (2008). Visualizing data using t-SNE. Journal of Machine Learning Research. 9. 2579-2605.