IT469

Human Language Technologies

E-motions

Prepared by

Group No. : 6	
Leader Email: 438202774@student.ksu.edu.sa	
Student Name	**Student ID**
Reema Albadia	438202774
Muneera Alshunaifi	438202230
Raghad Alawad	438201513
Raghad Aloraini	438201731

Supervised by

Dr. Afnan Alsubaihin

Dr. Nora Almadi

ROLES AND RESPONSIBILITIES:

Student name	Role	Reviewed section	Correction made
Reema Albadia	*Coding and report*	*All members*	
Muneera Alshunaifi	*Coding and report*	*All members*	
Raghad Aloraini	*Coding and report*	*All members*	
Raghad Alawad	*Coding and report*	*All members*	

Table of Contents

List of Tables

List of Figures

1. Introduction

In the area of artificial intelligence, intelligent machines are being designed which are expected to perform tasks which can be performed by humans. We are making machines that are intelligent and smart just like human beings are. However, one thing machines cannot do is the ability to understand by itself, Also it cannot identify various emotions like humans. Despite so much progress in the field of machine learning and natural language processing, machines have still not become capable of detecting emotions clearly. Emotion Analysis is the broadly studied application of Natural Language Processing (NLP) and Machine learning. Basically, there is a lot of work has been done in the field of sentiment analysis. However, it is a difficult task to study the emotion of a human mind. Thus, we have taken this project to study about different types of emotions. [1]

In Natural language processing (NLP), the detection of human emotions in text is becoming highly important from an applicative point of view. Emotion is expressed as joy, sadness, anger, surprise, hate, fear and so on. Since there is not any standard emotion word hierarchy, focus is on the related research about emotion in cognitive psychology domain. In 2001, W. Gerrod Parrot wrote a book named "Emotions In Social Psychology", in which he explained the emotion system and formally classified the human emotions through an emotion hierarchy in six classes at primary level which are Love, Joy, Anger, Sadness, Fear and Surprise. Certain other words also fall in secondary and tertiary levels. [2] Moreover, In detecting text emotions we can face some challenges such as context-dependencies of emotions within text, another challenge in emotion detection is the lack of labelled emotions, also we have the challenge of ambiguity in the text.

We aim in this project to train a machine how to detect the emotions in text/sentence and then can predict the overall emotion for any given text/sentence. It will help in understanding the human computer interaction. The purpose of using text classification is to determine the subjective value of a text, i.e. does the content of the text refers to Anger emotion or Sadness emotion? Further, in this project the machine will be able to understand and detect the emotion of any textual input.

The steps we are going to follow in this project are as the following: First, we will acquire the data. Then we will do some exploratory data analysis (EDA) where we will explore the data and do simple visualization that may help us with useful insights. After that, We will do the cleaning and pre-processing step for the dataset to make it more precise, clean and error-free, continuing the steps with doing some feature engineering where we will extract some features from the dataset that can be presented numerically to apply it to the machine learning models, moving forward to the process of applying machine learning models, in this step we will use different models one of them is artificial recurrent neural network which consider as a deep learning model; we will use long-short term memory (LSTM), also we will use two of the traditional machine learning models specifically Naïve bayes and SVM. Finally, we will evaluate each model and then we will compare them to choose the best performing model.

Moreover, we are writing this report to gather the information that has been compiled as result for our project. We will describe the Data section and we will explain Text Cleaning and Preprocessing, after that we will illustrate the Feature engineering where we will analyze the indicators that served our classification model, then we will explain the Modeling development process, ending up with evaluating our models by outlining and comparing the evaluation measures.

2. Data

We are using an English language textual data in text format, the data we receive are three text files: train.txt, test.txt and val.txt. these three text files consist of an emoji-free sentences ending with semicolon that is followed by the emotion that is related to the given sentence. Each of the three files has its own usage; train.txt and val.txt are used for training the models, and the test.txt is used for testing the model. We have converted these three files into data frames consist of two columns: Text and Emotion, where Emotion column consist of the labels we are trying to predict.

Clearly, we are using this data to train the machine by giving textual data as input then the machine will be able to predict one of these six labels: joy, sadness, anger, fear, love and surprise.We applied morphological tool which is nltk for English words in order to separate the text into multiple words

To explore the data more, we have applied simple Exploratory Data Analysis a.k.a (EDA) and we have explored some characteristics of the data which are:

a. The Shape of train, test and validation data frames:

Train data:(16000 rows , 2 columns) **Test data:**(2000 rows, 2 columns)

Validation data: (2000 rows, 2 columns)

b. Count of each emotion in train data:

```
joy         5362

sadness     4666
anger       2159
fear        1937
love        1304
surprise     572
```

c. Count of each emotion in test data:

```
joy         695
sadness     581
anger       275
fear        224
love        159
surprise     66
```

d. Count of each emotion in validation data:

```
joy          704
sadness      550
anger        275
fear         212
love         178
surprise      81
```

e. Null values in the data set are equal to zero; which is great!

Also, we have plotted labels counts and we have concluded that Joy has the highest count overall emotions, and the lowest is Surprise. This insight indicate that the models may perform better for Joy emotion sentences than others but it will not be able to perform well for Surprise emotion sentences.
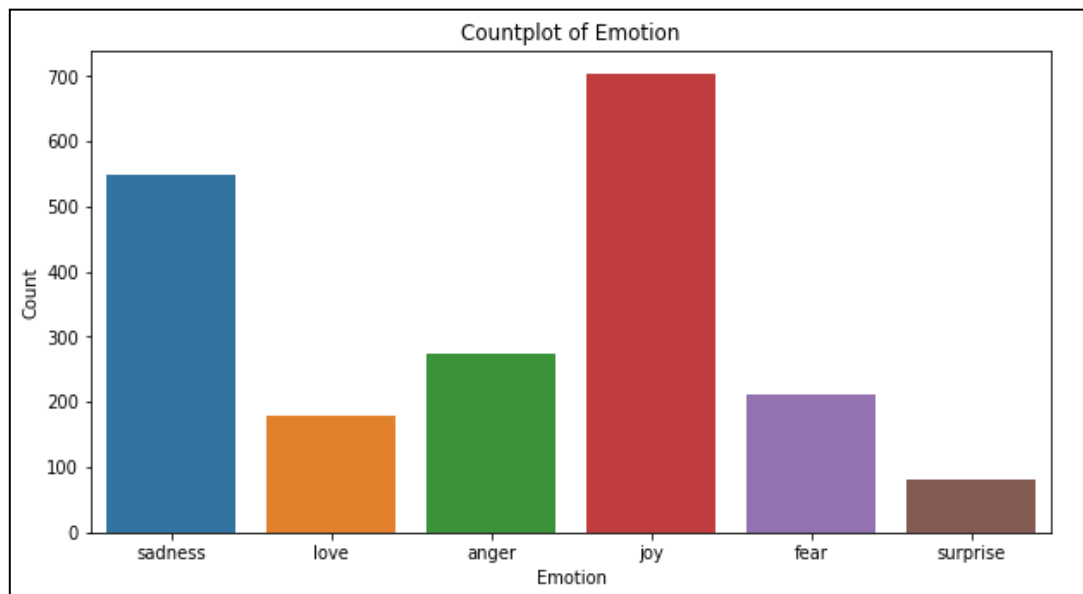

FIGURE 1 COUNT PLOT OF EMOTIONS

3. Cleaning and Preprocessing

In the text cleaning and preprocessing we aim to make the text clean precise and error-free, we have used the natural language toolkit as, it is a morphological tool and known as NLTK. First, we have defined two functions each is responsible for some cleaning and preprocessing steps. In the first function we used regular expression to remove URL's and repeated words.

The second function is responsible for stop words removal also converting the text to lowercase in addition to remove all tokens that is lesser than 3 characters and also punctuation removal even though because our corpus does not contain interesting punctuations for emotions such as Exclamations mark. Also, the function do strip multiple whitespaces and it strip the numeric, Finally the function do stemming for the text data.

These two functions will be applied for the Text column in the three data frames we have. The two functions are important to make the modeling process easier because it helps in getting rid of unhelpful parts in the text, Also it will help us bringing the text data into a form that is predictable and analyzable to detect the emotions. For example, Stemming is useful for dealing with sparsity issues as well as standardizing vocabulary, Also lowercasing can provide consistency of expected emotion prediction. [3]

4. Feature Engineering

We have extracted some features from the data that can be presented numerically to apply it to the machine learning models in the next step. We have Extracted these two features:

- Length of text in a record
- Adjectives in each text record

After extracting these features we figure out that it may not be useful for our problem since adjectives is not well-categorized and have illogical words that is not related to an adjective, also the length of text will not help us in detecting the emotions because there is no relationship with our indicator.

To continue, our indicator is Emotion feature, we have encoded the labels of the emotion column as the following:

Joy → 0 Fear → 1 Anger → 2 Sadness → 3 Love → 4 Surprise → 5

Then we have categorized these labels, after that we have applied two pre-trained model which are:

a. Word2Vec : It is used to produce word embeddings, It is shallow, two-layer neural networks that are trained to reconstruct linguistic contexts of words [4], it is useful for our LSTM model. First, we did padding to the data where each input will be having the same length and we have defined maximum numbers of words for our corpus and the input size for our model because it must be fixed size. After preparing input and target values we have created weighted matrix using genism Word2Vec. Genism provides Word2Vec class for working with Word2Vec model.

b. TF-IDF: First, we have transformed the corpus into numbers and that is known as Vectorization. The TF-IDF score can fed to Naïve Bayes and Support Vector Machine (SVM). It is greatly improving the results of the basics methods such as word counts. [5]

5. Modeling

We have used two pre-trained language models: Word2Vec and TF-IDF, which helped us in training our classifiers that is by converting our text data into numeric, so the machine will be able to learn. Actually the comparison between these two pre-trained language models diverse from each other; Word2Vec is usually used with deep learning and large corpus (e.g. LSTM), while TF-IDF is used usually for the traditional machine learning algorithms (e.g. Naïve Bayes and SVM). In this project we have applied three different machine learning models.

First, we have started by implementing an artificial recurrent neural network which consider as a deep-learning model which is: long-short term memory (LSTM) with GPU support, we have fit the model with 25 epochs and 120 batch size, and it have took approximately 10 minutes to run. Then we have plotted the train and validation accuracy plot, the plot summarized that the model could have stopped at epoch of 6 and attain a train accuracy of 0.65 and validation accuracy of 0.60, But as shown below the model is over-fitting our train dataset.
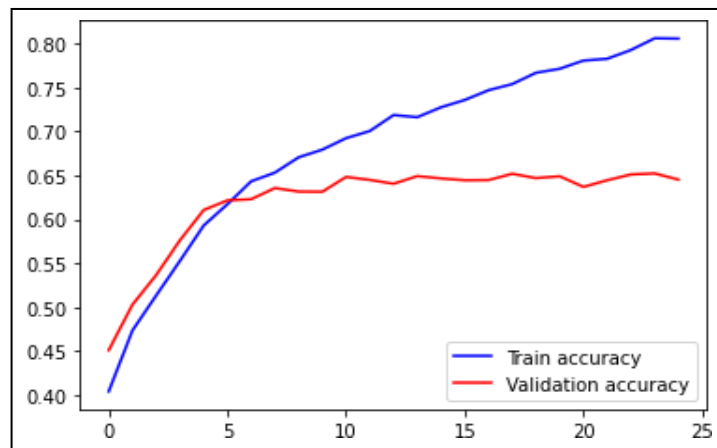

FIGURE 2 LSTM ACCURACY PLOT FOR CLEANED TEXT

Moreover, we have plotted train and validation Loss plot; In general, Loss is a summation of the errors made for each example in training or validation sets [6], so the lower the loss the better the model. But we are noticing from Loss plot, that loss does not turning to the lowest so fast. So we can insure our pre-analysis of the accuracy plot, Which is that our model is truly over-fitting the train dataset.
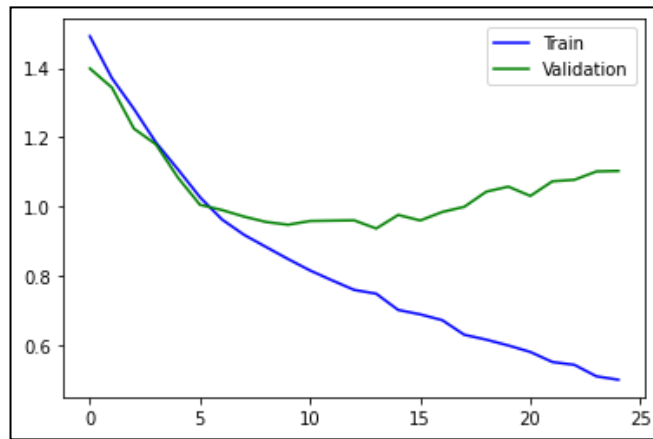
FIGURE 3 LSTM LOSS PLOT WITH CLEANED TEXT

Again, we have trained the model on uncleaned data. As it shows in figures below the loss plot and accuracy plot are as same as training the model on cleaned data, means it is still overfitting the train dataset.
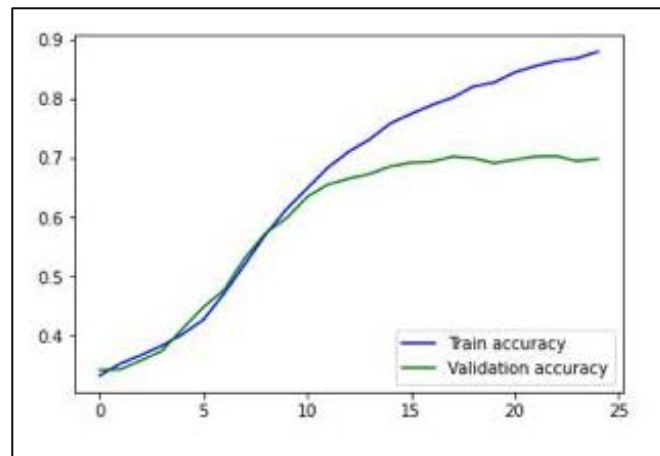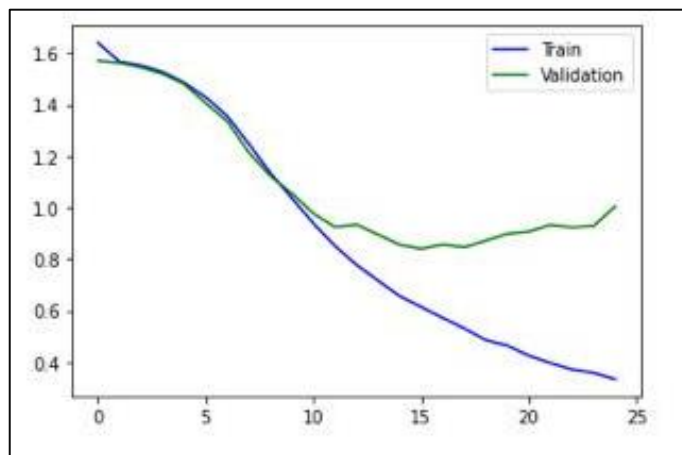

FIGURE 4 ACCURACY PLOT WITH UNCLEANED TEXT

FIGURE 5 LOSS PLOT WITH UNCLEANED TEXT

Second, We have applied Naïve Bayes with two pretrained models TF-IDF and Word2Vec. Naïve Bayes is a classification technique based on Bayes' Theorem with an assumption of independence among predictors, it assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature, basically we have applied it to our corpus because it is considered as the baseline classifier and it is the most straight forward and fastest classification algorithm. [7] Third, we have applied Support Vector Machine (SVM) with two pretrained models TF-IDF and Word2Vec; it is a supervised machine learning model that uses classification algorithms for two-group classification problems. After giving an SVM model sets of labeled training data for each category, it will be able to categorize the text. We have used it because it is effective in high dimensional space and also in cases where number of dimension is greater than number of samples. [8]

To sum up, we have trained these three models to our corpus, first time with cleaned data, and second time with uncleaned data, in order to know which model will perform well and in which data. For evaluation results, it will be described in details next part.

6. Evaluation Results

We have measured our models using Accuracy, Precision, Recall and F1-score. Here is the below their definitions and formulas. [9]

Accuracy: is the number of true probabilities over all the probabilities.

Formula: TP \ TP+TN + FP + FN

Precision: is the number of true positive(the predicted one that was matched with the actual one) over all the positive probabilities(All the predicted YES regardless to their actual value)

Formula: TP \ TP+ FP

Recall: is the number of true positive(predicted YES and the actual value is YES) over true positive and false negative(predicted NO and the actual value is YES).

Formula: TP \ TP+ FN

F1-Score: is combination between precision and recall.

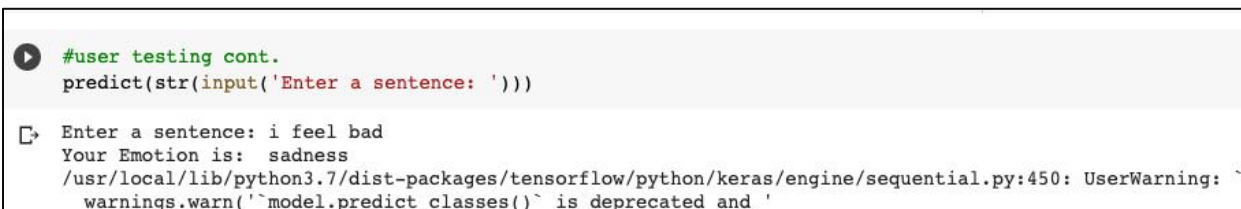**Formula:** 2X Precision Recall \ Precision + Recall

Also, we have used Confusion Matrix for comparison which is a metric that classify the data into two classes: Yes or No. While the inputs data belongs to predicted class and the output data belongs to actual class. [9]

Then we have compared the measures of each model with each other.

- **Evaluation Results for Cleaned Text data**

First, in LSTM model the weighted average of the precision was 0.66 and for the recall was 0.65, and for the f1-score was 0.65 while the accuracy was 0.64, as shown in below table the LSTM model can predict and perform better in label 0 (Joy) and label 3 (Fear) by resulting a range from 0.72 to 0.74 for Joy label in those three measures, the same with Fear label where it ranges approximately from 0.68 up to 0.76. The worst resulting label is Surprise where the result ranges approximately from 0.29 to 0.33 and that is maybe because our corpus does not contain a lot Surprise emotion sentences.

The figure below show the detection of given text by a user and it has detect the right emotion.

```
#user testing cont.
predict(str(input('Enter a sentence: ')))

Enter a sentence: i feel bad
Your Emotion is:  sadness
/usr/local/lib/python3.7/dist-packages/tensorflow/python/keras/engine/sequential.py:450: UserWarning: `
  warnings.warn('`model.predict_classes()` is deprecated and '
```

FIGURE 6 USER TESTING CLEAN TEXT

Second, in Naïve Bayes Classifier with pretrained model TF-ID, the weighted average of the precision was 0.73 and for the recallwas 0.69, and for the f1-score was 0.63 while the accuracy was 0.646, as shown in below table the Naïve Bayes Classifier is not able to predict and do any performance for Surprise label since it results 0 for the three evaluation measures, Also, it has weak performance for Fear label since it is result 1 in precision but 0.09 in recall and 0.16 in f1-score and that seems illogical, These illogical results may it is due the word boot-strap Aggregation aka bagging that Naïve classifier does; bagging is technique that is useful for both regression classification, it usually introduce a loss of interpretability of a model. [10] In other hands, Naïve Bayes is performing well for Love and Anger label. Regarding applying Naïve with pretrained model Word2Vec, it cannot be trained with Word2Vec since naïve is not able to accept any negative values from the word vectors. So from our insight we think this is a limitation in naïve bayes.

In SVM with pretrained model TF-IDF. the weighted average of the precision was 0.86 and for the recall was 0.85, and for the f1-score was 0.85 while the accuracy was also 0.85, as shown in below table SVM is performing well with almost all labels, maybe it is having slightly weak performance in Fear label because it is resulting 0.49 in the recall measure and also it may not perform well for Surprise label due to recall result which is 0.56. Moreover training the SVM with pretrained model Word2Vec has resulted a weighted average equals to 0.31 for f1-score and 0.4 for recall also 0.31 for precision, the accuracy was 0.39.

TABLE 1 EVALUATION MEASURES WITH CLEANED TEXT

Model	Accuracy	F1 Score	Precision		Recall	
LSTM	64%	56%	0 → joy	72%	1 → joy	74%
			1 → sadness	53%	2 → sadness	59%
			2 → anger	52%	3 → anger	66%
			3 → fear	76%	4 → fear	61%
			4 → love	47%	5 → love	56%
			5 → surprise	29%	6 → surprise	33%
SVM with Tf-idf	85%	85%	0 → joy	88%	1 → joy	79%
			1 → sadness	87%	2 → sadness	80%
			2 → anger	83%	3 → anger	96%
			3 → fear	93%	4 → fear	49%
			4 → love	87%	5 → love	92%
			5 → surprise	86%	6 → surprise	56%
Naïve Bayes with Tf-idf	69%	63%	0 → joy	92%	0 → joy	35%
			1 → sadness	91%	1 → sadness	34%
			2 → anger	65%	2 → anger	97%
			3 → fear	100%	3 → fear	9%
			4 → love	68%	4 → love	89%
			5 → surprise	0	5 → surprise	0
SVM with word2vec	39%	31%	0 → joy	0	0 → joy	0
			1 → sadness	33%	1 → sadness	0
			2 → anger	41%	2 → anger	72%
			3 → fear	0	3 → fear	0
			4 → love	38%	4 → love	50%
			5 → surprise	56%	5 → surprise	8%

- **Evaluation Results for Uncleaned Text data**

First, in LSTM model the weighted average of the precision was 0.39 and for the recall was 0.18 and for the f1-score was 0.2 while the accuracy was 0.18, as shown in below table the LSTM model can predict and perform better in label 0 (Joy) and label 2 (Anger) by resulting a range from 0.22 to 0.37 for Joy label in those three measures, the same with Anger label where it ranges approximately from 0.13 up to 0.56. The worst resulting label is Surprise where the result zeroes and that is maybe because our corpus does not contain a lot Surprise sentences. The figure below show the detection of given text by a user and it has detect the wrong emotion.
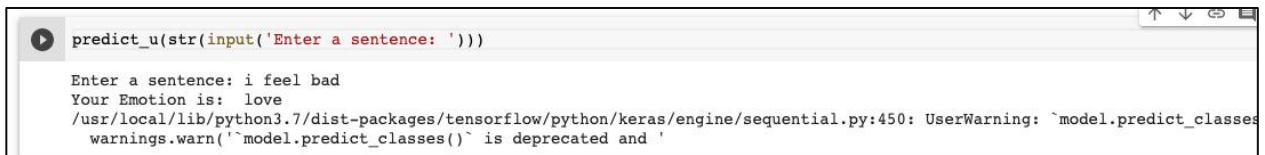
```
predict_u(str(input('Enter a sentence: ')))

Enter a sentence: i feel bad
Your Emotion is:  love
/usr/local/lib/python3.7/dist-packages/tensorflow/python/keras/engine/sequential.py:450: UserWarning: `model.predict_classes
  warnings.warn('`model.predict_classes()` is deprecated and '
```

FIGURE 7 USER TEST WITH UNCLEANED TEXT

Second, in Naïve Bayes Classifier with pretrained model TF-ID, the weighted average of the precision was 0.53 and for the recallwas 0.46, and for the f1-score was 0.39 while the accuracy was 0.46, still even with uncleaned data the Naïve Bayes Classifier is not able to predict and do any performance for Surprise label since it results 0 for the three evaluation measures.

In SVM with pretrained model TF-IDF. the weighted average of the precision was 0.88 and for the recall was 0.88, and for the f1-score was 0.87 while the accuracy was 0.88, as shown in below table SVM is performing well with almost all labels, even better than training it with cleaned Text data; It is resulting slightly higher performance.

Finally, we can conclude from comparison tables that SVM with pretrained model TF-IDF wins the best performance among all models for the uncleaned Text data since it gained the highest F1-score which was equal to 0.87. we think it may be due to the simplicity and purity of the original received data, also we think that cleaning and preprocessing causes loss of meaning of words that's why SVM is acting well with the uncleaned Text data

TABLE 2 EVALUATION MEASURES FOR UNCLEANED TEXT

Model	Accuracy	F1 Score	Precision		Recall	
LSTM	18%	12%	0 → joy	22%	0 → joy	37%
			1 → sadness	38%	1 → sadness	14%
			2 → anger	56%	2 → anger	13%
			3 → fear	4%	3 → fear	34%
			4 → love	6%	4 → love	5%
			5 → surprise	0	5 → surprise	0
SVM with Tf-idf	87%	87%	0 → joy	91%	0 → joy	83%
			1 → sadness	88%	1 → sadness	80%
			2 → anger	84%	2 → anger	97%
			3 → fear	91%	3 → fear	59%
			4 → love	90%	4 → love	93%
			5 → surprise	91%	5 → surprise	48%
Naïve Bayes with Tf-idf	46%	39%	0 → joy	73%	0 → joy	7%
			1 → sadness	82%	1 → sadness	10%
			2 → anger	47%	2 → anger	75%
			3 → fear	57%	3 → fear	5%
			4 → love	43%	4 → love	61%
			5 → surprise	0	5 → surprise	0

7. Limitations and Future Work

We have noticed from analyzing and training the data to different models that the data is limited in predicting Surprise sentences. Also we have noticed that the data does not contains emoji in which it consider an interesting and useful feature in any sentiment analysis and emotion detection problem, Further, after adding the adjective feature we noticed that it does not categorize the adjectives accurately in a sentence and that is may be due to a core problem in Textblob library, we have tried solving that issue either putting some conditions and trying other library but unfortunately we did not have any interesting results. In other hands, there is a limitation we faced during training Naïve Bayes classifier with pretrained model Word2Vec in which it does not work with Word2Vec, that is, because we think that naïve cannot actually trained and accept the negative values that comes from vectors of words.

Also, we have face difficulties in training Naïve bayes and SVM with Word2Vec on uncleaned Text data. It took so much time to run and we actually think that those classifiers does not required or need a word2vec as pre-trained language model.

As future work we can modify the data and add Emojies. Additionally, we can improve the prediction of Surprise label by adding more surprise sentences to the data. Moreover, we can try classification methods other than we have used in this project such as: Logistic regression and Decision Tree.

8. Conclusion

We have tried in this project to solve the problem of emotion detection, basically we have analyzed our data then clean it and doing preprocessing steps in order to indicate the features and know which are the interesting ones, we have extract from the data some features which are Length of Text record and we extracted the Adjectives for each Text record, these two features did not help us much in solving our problem since the adjectives feature is obtaining illogical categorization of adjectives in some Text records and the Length column does not have relationship with our Target column, also we have did some pre-trained language model: TF-IDF and Word2Vec which consider as an advanced feature engineering in order to apply it to the different three models; we have trained our data twice, one with cleaned Text data and other not cleaned Text data, we used deep learning model: LSTM, we have trained it using Word2Vec. Also we have trained the data using traditional machine learning algorithms which are: SVM and Naïve bayes, both are trained first using TF-IDF, second with Word2Vec. As a result SVM with TF-ID using uncleaned data is the best model to solve our problem among all models, as shown in the evaluation results part.

9. References

[1] https://github.com/Bhagya4347/Emotion-Analysis-On-Text/blob/master/Project_Report.pdf

[2] Emotion_Detection_from_Text.pdf

[3] https://www.kdnuggets.com/2019/04/text-preprocessing-nlp-machine-learning.html

[4] https://www.guru99.com/word-embedding-word2vec.html

[5] https://monkeylearn.com/blog/what-is-tf-idf/

[6] https://docs.paperspace.com/machine-learning/wiki/accuracy-and-loss

[7] https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/

[8] https://scikit-learn.org/stable/modules/svm.html

[9] https://www.dataschool.io/simple-guide-to-confusion-matrix-terminology/

[10] https://corporatefinanceinstitute.com/resources/knowledge/other/bagging-bootstrap-aggregation/