Faculty of Engineering & Technology – Electrical & Computer Engineering Department

First Semester 2022 – 2023

Artificial Intelligence – ENCS3340

Tweet Emotion Detection Project (Machine Learning)

Prepared By:

Ali Mohammad – 1190502

Ahmaide AlAwawdah – 1190823

Instructor:

Mr. Aziz Qaroush

Section: 1

Date: February 2023

Abstract

The main objective of this project is to classify Arabic tweets to be either positive or negative tweets, by analyzing their featured emotions in order to feed and train the system with its needed data to make it a predicting system with high classification accuracy, which will also be tested.

The used programing language for this project is Python, using version 3.9, with the use of some useful libraries such as nltk, ar_correction, pandas, sklearn, and numpy.

❖ Table of Content

1.Problem Specification 1

2.Project Stages........ 2

2.1. Data Gathering 2

2.2. Data Preprocessing 2

2.3. Features Extraction 5

2.4. Training & Testing 6

2.5. Testing Calculations 7

2.6. The Deployment........ 7

3.Running The Program 8

3.1. Data Preprocessing Results 8

3.2. Feature Extraction Results 8

3.3. Classifications Models Results 13

4.Conclusion........ 20

5.References 21

❖ Table of Figures

Figure 1-1: Tweet Classification 1

Figure 2-1: Emotion Detection System Diagram 2

Figure 2-2: Data Preprocessing 3

Figure 2-3: Stemming Process [2] 4

Figure 2-4: Feature Extraction Diagram [3] 5

Figure 2-5: Training and Testing Diagram 6

Figure 3-1: Data Processing Outcome 8

Figure 3-2: Emoji Counting Difference 9

Figure 3-3: Hashtag Counting Difference 9

Figure 3-4: Sentence Counting Difference 10

Figure 3-5: Character Counting Difference 10

Figure 3-6: Word Counting Difference 10

Figure 3-7: Hashtag Percentage Difference 11

Figure 3-8: Emoji Percentage Difference 11

Figure 3-9: Average words in a sentence Difference 11

Figure 3-10: Average Characters in a Word Difference 12

Figure 3-11: Stop Words Counting Difference 12

Figure 3-12: Random Forest Confusion Matrix 13

Figure 3-13: Random Forest True Positive Graph 13

Figure 3-14: Random Forest False Negative Graph 14

Figure 3-15: Random Forest Precision-Recall Graph 14

1. Problem Specification

In this project the problem is specified to be the classification of Arabic tweets, where a tweet can be classified as either positive or negative tweet judging by its content (text and emojis).

The content should display the needed information that can describe the emotions in this tweet in order for the system to detect it and decide the situation of the tweet (positive or negative).

Figure 1-1: Tweet Classification

First the tweets content is full of unnecessary data that may damage the tweet's classification, and for that the tweets will need some processing in order to get rid of these unnecessary contents to give the tweet the ability to contribute in training the system or to be ability to be classified.

In order to determine whether if a tweets content is either positive or negative, a set of features must be defined, where these features depend on the tweet's emojis and text content as the system measures the statistics of these features for a good amount of both positive and negative training tweets, so when they're measured for a given tweet the system shall decide if its positive or negative.

So, in order to make a complete classification system, two sets of data (tweets) will be needed with each tweet being classified as if it is actually positive or negative, where the first set will train the system to classify the tweets and the second set will test the work flow of the system after it was trained with the first set.

2. Project Stages

For this project as shown in figure 2-1, the system is divided into multiple parts where each part plays a part that takes the system to its needed outcomes.

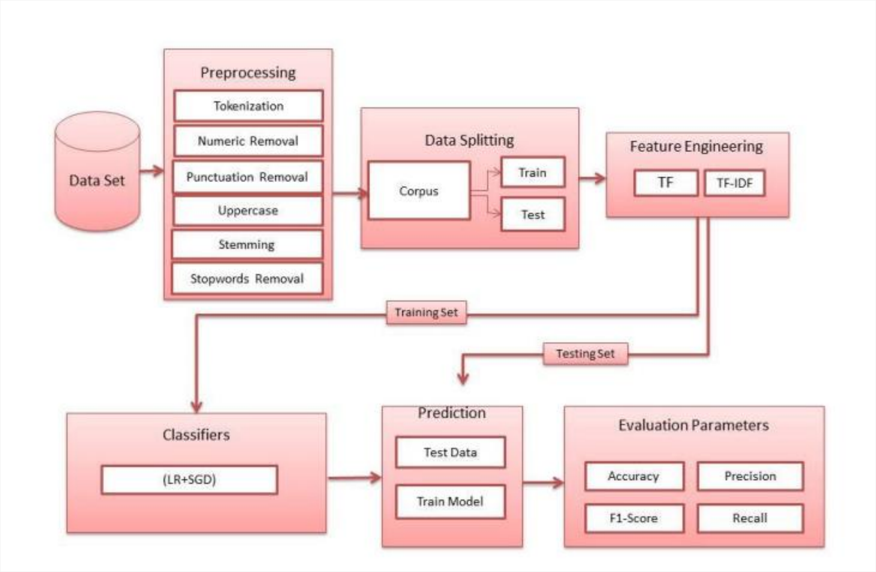


Figure 2-1: Emotion Detection System Diagram

2.1. Data Gathering

In order to build a system that detects the emotions in Arabic tweets tweet it, first it needs to be fed by given gathered tweets which with their classifications being given too, in order to train the system with a percentage of tweets for each class, and then tested with the remaining tweets.

In this project there two data files are inserted where one of them will contain the gathered positive tweets, and the other will have the negative tweets.

2.2. Data Preprocessing

After gathering the data, before any of the training or testing processes, that data will include many components that needs to be cleared out in order to have clean data that can be processed so it can have better clarifying of its features, and for that the data will need to be preprocessed (cleaned) using the nltk library in python, some features are calculated before the text being fully processed (will be detailed) , figure 2-2 below shows an example of an Arabic text preprocessing.

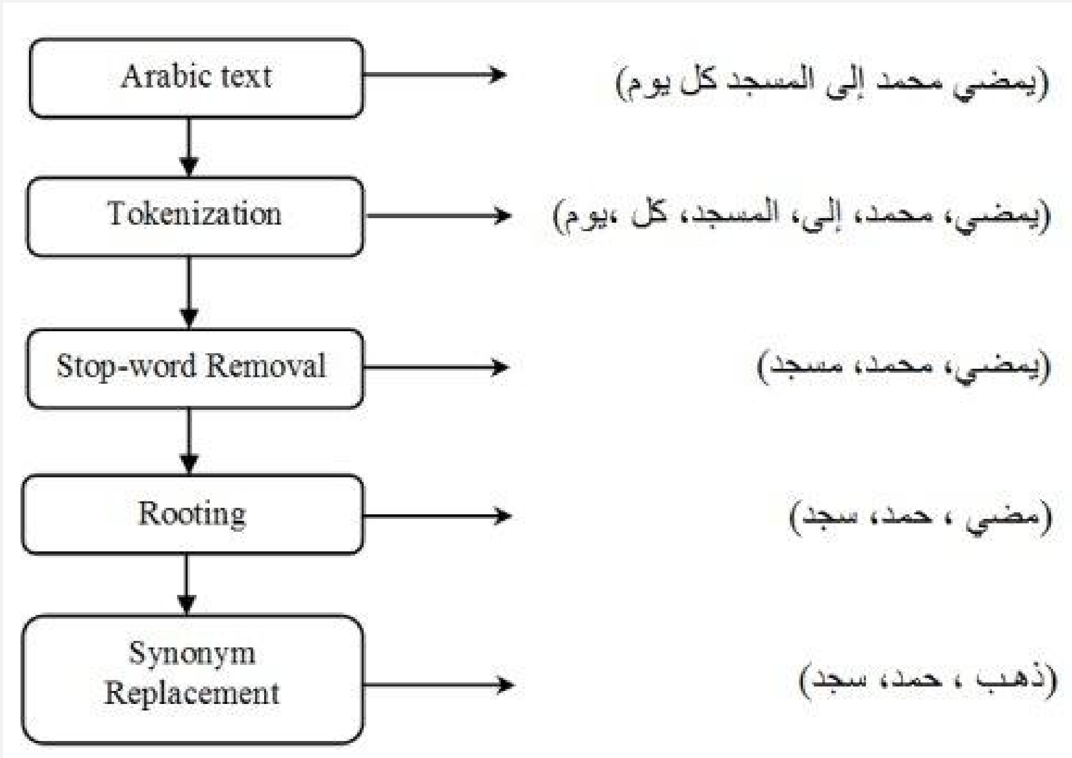
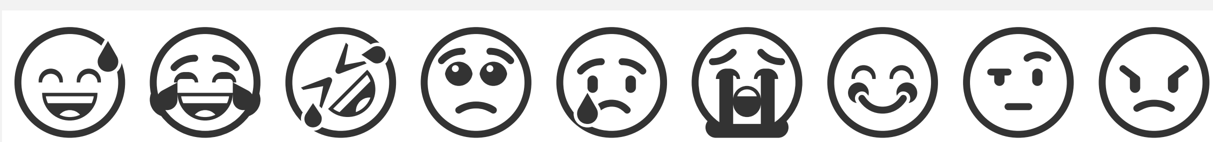


Figure 2-2: Data Preprocessing

Here are the operations that are done to the data in order for it to be preprocessed:

2.2.1. Emojis Extraction

As said before these Arabic tweets can contain emojis that can be useful in the tweet's emotion detection, emojis such as:



These emojis will be extracted from the text in order to decide their features separately.

2.2.2. Removing Hashtags

Hashtags such as (#ديوان_الشعر، #مساء_الخير_للجميع، #الاتحاد) won't help in the emotion detection process yet it might damage it, which makes them useless, so they will be extracted from the text, yet before the removal the number of hashtags in each tweet is calculated in order to be used as a feature.

2.2.3. Removing Characters other Than Arabic Letters

Any character in the tweet's content such as other language character (English, Spanish, etc.) or any numeric character is removed from the text, as it won't give any detail about the emotions in the text.

2.2.4. Remove Vocalization

As known Arabic words can have some vocalization on its letters such as (بَ، بٍ، بْ، بً، بُ، بٌ، بِ) these vocalizations are not useful in coding as they will make the strings characters more complicated so they will be removed.[1]

2.2.5. Give each Arabic letter one Shape

As known in the Arabic letters, a letter can have more than a state or shape such as "ا" can be any of the following (أ، ا، إ، آ) so these kinds of letters should be formalized to have only one shape or one state.

2.2.6. Removing Consecutive Characters

Consecutive characters in the tweet’s texts are also remove.

2.2.7. Removing Stop words

Stop words are also a part of blocking the emotion detection process so the stop words are removed from the text after setting up that the stop words that are being searched for are Arabic stop words, stop words such as (اكثر ،الذي، من، على، في، إن، أي، بعد، لولا، نحن، يا).

2.2.8. Tokenization

The tokenization process is simply separating the text words from each other, as it turns the string into a list of its content, which will make the feature extraction process easier.

2.2.9. Stemming

Stemming is a process that gives the original meaning of each word with removing its extra, this can help in extracting the similarities between the same classification tweets, as shown in figure 2-3.

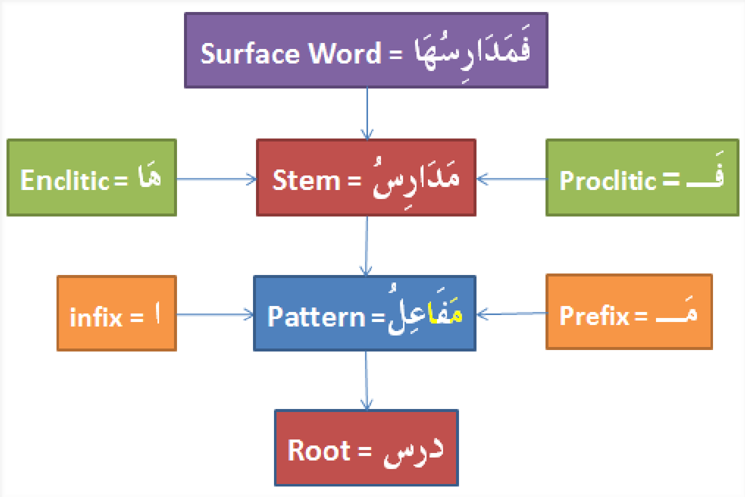


Figure 2-3: Stemming Process [2]

2.3. Features Extraction

Each tweet has its own contained features, these features can play a big role it detecting the tweets emotions, in order for it to be classified, features can depend on the both the tweet's text content, and the content emojis too, as each one of them can produce its own features.

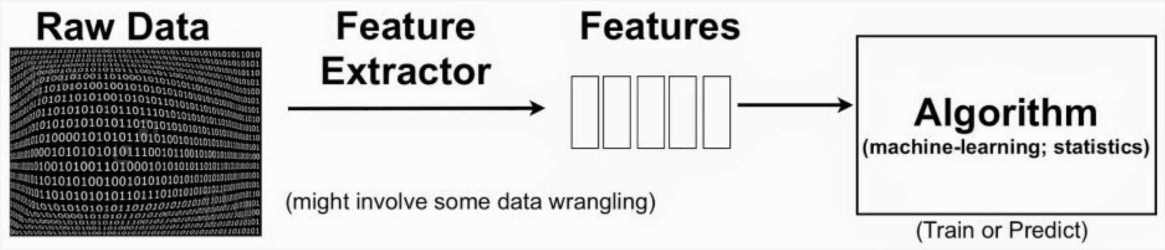


Figure 2-4: Feature Extraction Diagram [3]

2.3.1. Manually Extracted Features

Some features were extracted manually in order to increase the performance of the models, since the automated extractions only measures the frequency of the words, the following features were included in this machine learning system manually:

1. Emoji Count: The number of the included emojis in a tweet can be a feature that is used in

2. Hashtag Count: Same as the emojis the number of the included hashtags in the tweet can also be a feature that is used in classification.

3. Sentence Count: The deviation of a tweet to sentences can depend on the meaning inside the that tweet which can also play a role in the tweet's classification.

4. Chars Count: This feature calculates the number of Arabic alphabetical characters in the tweet.

5. Words Count: This feature calculates the number of Arabic alphabetical words in the tweet.

6. Hashtags Percentage: The number of hashtags can tell a different meaning by having different tweets lengths, so the number of hashtags to the number of the words in the tweet can help reducing this problem, so it is used here as a feature too.

7. Emoji Percentage: Same as the hashtags the percentage of the included emojis to the number of words in a tweet can play in this process which can make it a feature.

8. Average Sentence Length: The average length of the sentence as the summation of the number of words in each sentence to the number of sentences in a tweet is used as a feature.

9. Average Word Length: The average length of the words as the summation of the Arabic characters in each word to the number of words in a tweet is used as a feature.

10. Stop Words Count: The number of the Arabic stop words in the tweet (it is calculated before the tweet is preprocessed since the preprocessing removes the stop words), as the number of stop words such as (اكثر، الذي، من، على، في، إن، أي، بعد، لولا، نحن، يا) can relate to the tweet's emotions.

2.3.2. Automated Features

The term frequency-inverse document frequency (TF-IDF) can be used in order to calculate the frequency of the emojis and Arabic words that are included in the tweets in order to extract a number of features for the tweet that can play a big role in its classification, where it can be explained in the following formulas: [4]

$$TF(t,d) = \frac{number\ of\ times\ t\ appears\ in\ d}{total\ number\ of\ terms\ in\ d}$$

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

$$TF - IDF(t,d) = TF(t,d)\ x\ IDF(t)$$

2.4. Training & Testing

Now that the features are set, the given data set should be distributed as the system shall be trained on 75% of the given tweets (both positive and negative) by their given features, and trained on the remaining 25% by classifying them using the learnt method from the training data in order to evaluate the outcome classifications of the tested data and compare them with the original ones, figure 2-5 shows the diagram of the two processes. [5]

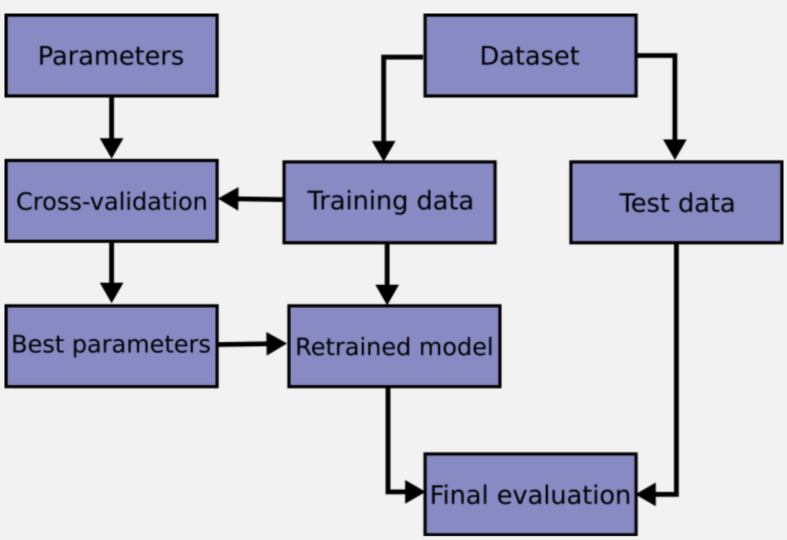


Figure 2-5: Training and Testing Diagram

2.5. Testing Calculations

For this project three classifiers are being used:

1. **Random Forest:** Which is an implementation of a decision tree.
2. **Naïve Bayes:** Using probability.
3. **Multi-Layer Perception:** Neural Network Classifier.

Each classifiers performance is measured by the following:

1. **Accuracy:** using the following formula
$$\frac{\boldsymbol{Correct\ Classifications}}{\boldsymbol{Total\ Cases}}$$
2. **Sensitivity (Recall):** using the following formula
$$\frac{\boldsymbol{True\ Positive}}{\boldsymbol{Classified\ Positive}}$$
3. **Precession:** using the following formula
$$\frac{\boldsymbol{True\ Positive}}{\boldsymbol{Actual\ Positive}}$$
4. **F_1 -Score:** using the following formula
$$\frac{\boldsymbol{2\ pr}}{\boldsymbol{p+r}}$$
5. **ROC AUC:** which is the area under the receiver operating characteristics.

In order to avoid overfitting, the average score using 5-fold cross validation is implemented, as it divides the data for training and testing 5 times, with different distribution of the testing and training data each time yet with the distribution percentage remains the same.

2.6. The Deployment

A set of unclassified Arabic tweets can be given to the system, after the system has learned how to classify tweets, the system will take those tweets process them and classify them by their features using the previous three classifiers, where each classifier has its own results in determining whether the tweets are positive or negative.

3. Running The Program

3.1. Data Preprocessing Results

The table below shows the out come of the preprocessing of some tweets with the Emoji extraction.

Table 3-1: Tweets Preprocessing

Tweet	Tweet's Preprocessing text	Emojis
أحببته حتى أقنعني إن ما فات العمر كان إنتظار له 🙈	احب حتي قنع ان فات عمر نظر	['🙈']
لم يبدو ان دجلة اعتادت على التهام اجساد ابنائها من سبايكر للعبارة ..👆👆💔 ما سمعته .. ان البشر يعطش فيشرب الماء ..	بدو ان دجل عاد علي تهم جسد من يكر عبر سمع ان بشر عطش شرب ماء	['👆👆💔']
لو بيدي أرجع ساعتي وين أرجع؟ إليا صدفه؟ والله أبقى أفر بيها لما ترجع بشر ماعرفه! وأنهي العلاقة من العرق.. قبل الدمع	بيد رجع سعت وين رجل الا هدفه وله ابق افر بيه رجع بشر اعرف ونه علق عرق دمع	[]
انت يمكن الي ناسي احداث البصرة قبل شهور او متغافل عنها نقتلوا متظاهرين بدم بارد منو قتلهم 😕	انت يمكن الي نسي حدث بصر شهر او غافل عنه قتل ظاهر بدم برد منو قتل	['😕']
من يخاف فليس منا.. تصبحون على 📌 خير #النصر_الاتحاد	يخف فلس منا صبح علي خير	['📌']

3.2. Feature Extraction Results

The csv that includes the processed data with the chosen features will be as in figure 3-1.

Figure 3-1: Data Processing Outcome

Each features effect is calculated and plotted as a histogram where the it gives the average value of each feature for both classifications, as its all shown below.

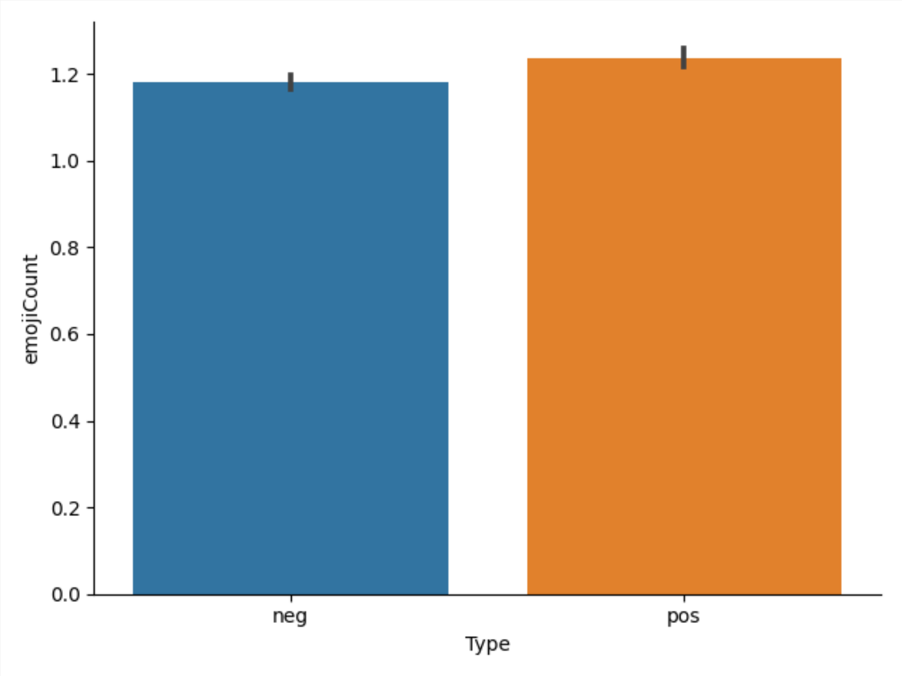


Figure 3-2: Emoji Counting Difference

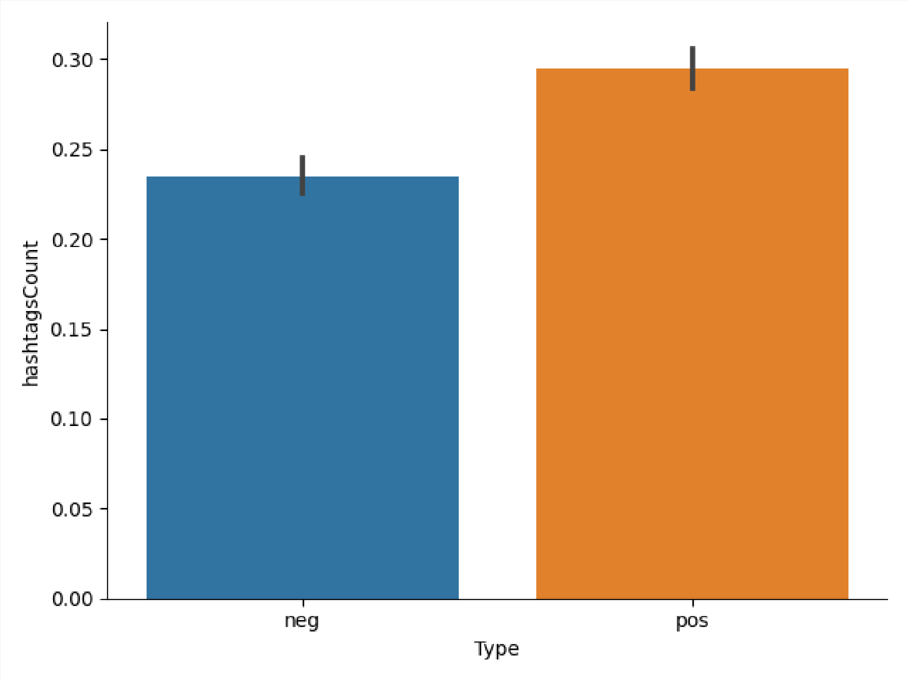


Figure 3-3: Hashtag Counting Difference

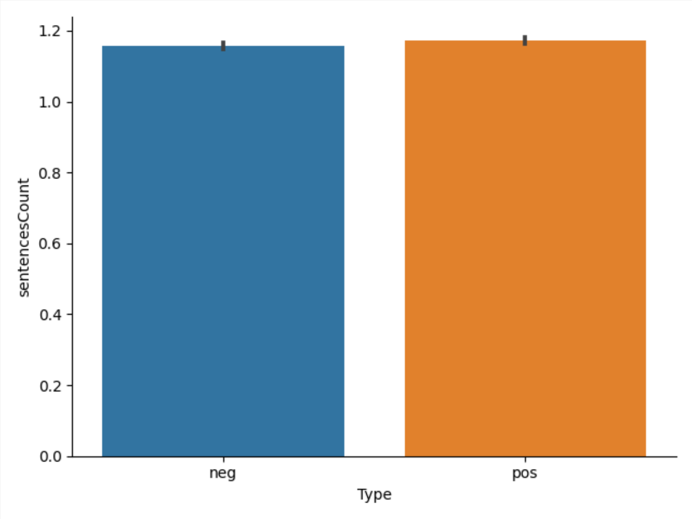


Figure 3-4: Sentence Counting Difference

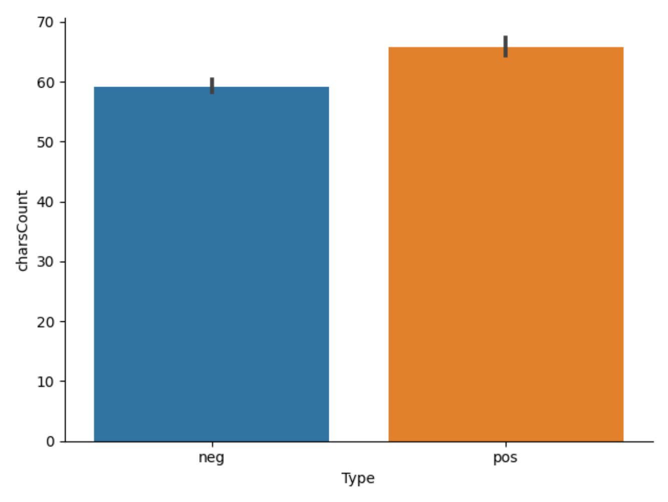


Figure 3-5: Character Counting Difference

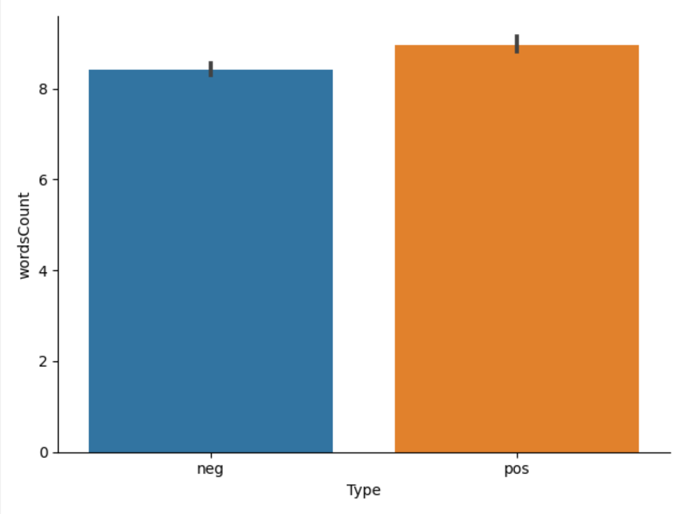


Figure 3-6: Word Counting Difference

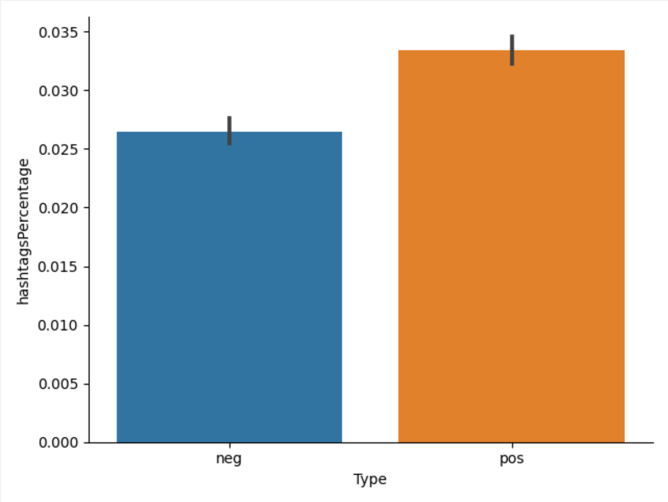


Figure 3-7: Hashtag Percentage Difference

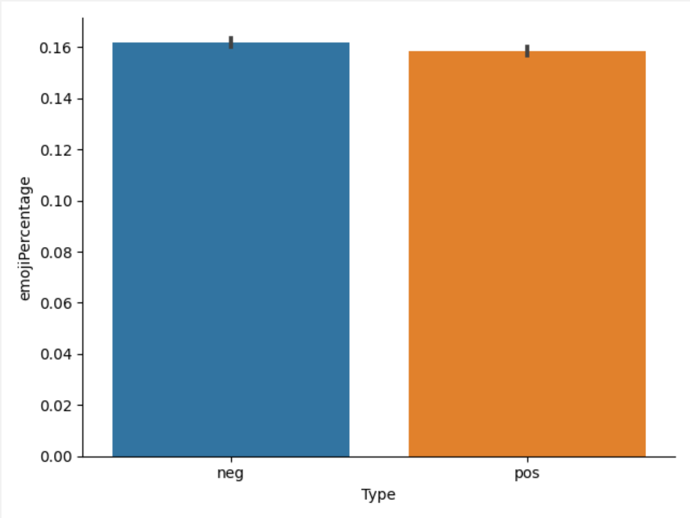


Figure 3-8: Emoji Percentage Difference

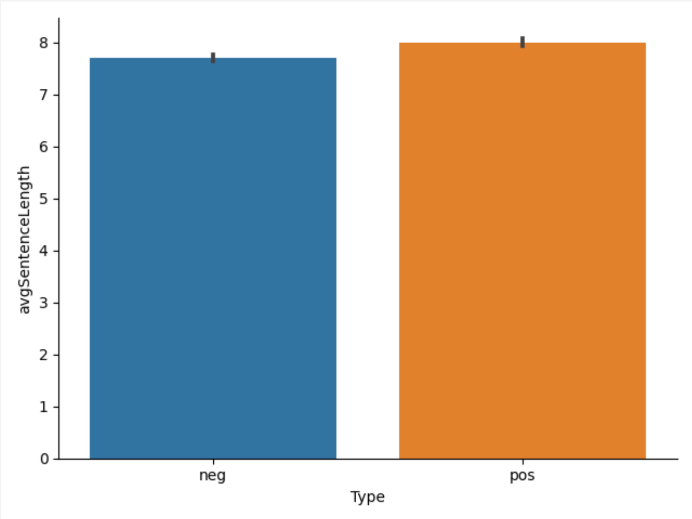


Figure 3-9: Average words in a sentence Difference

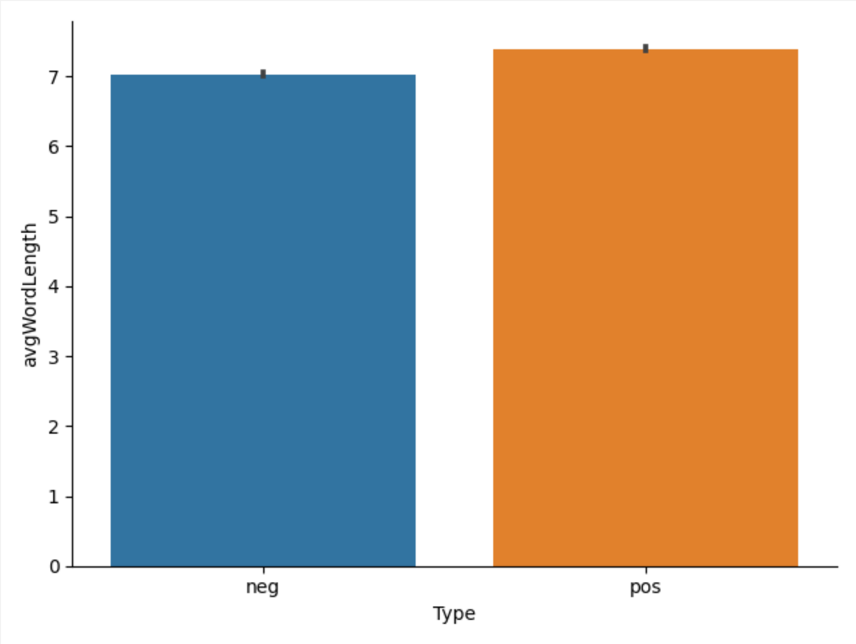


Figure 3-10: Average Characters in a Word Difference

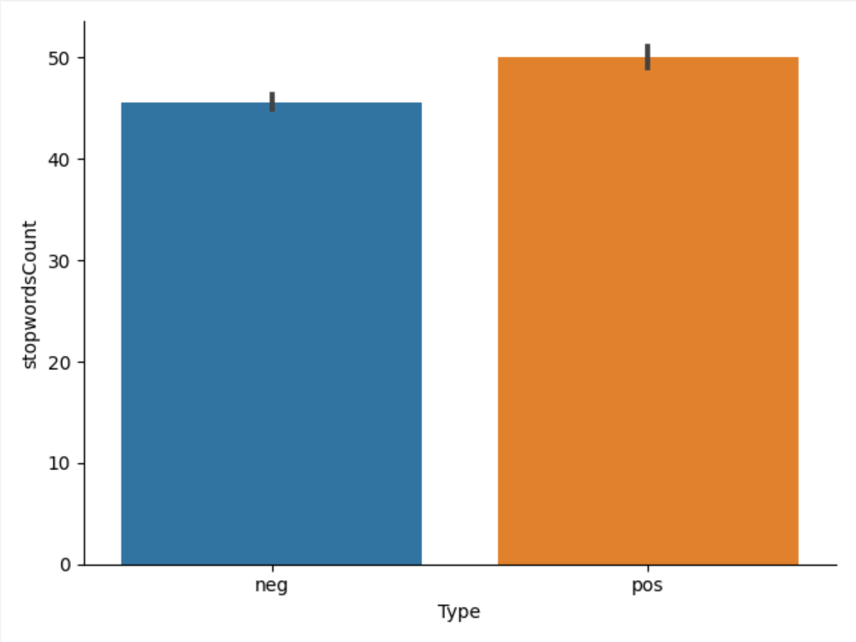


Figure 3-11: Stop Words Counting Difference

3.3. Classifications Models Results

Each classifier model has its confusion matrix, and precession, recall, accuracy, and F1-rate, adding on the ROC AUC and the scores using 5-fold cross validation.

3.3.1. Random Forest

The Confusion matrix for the random forest classifications is displayed in figure 3-12, with the true positive rate graph in figure 3-13, false negative rate graph in figure 3-14, and the precision recall graph in figure 3-15.

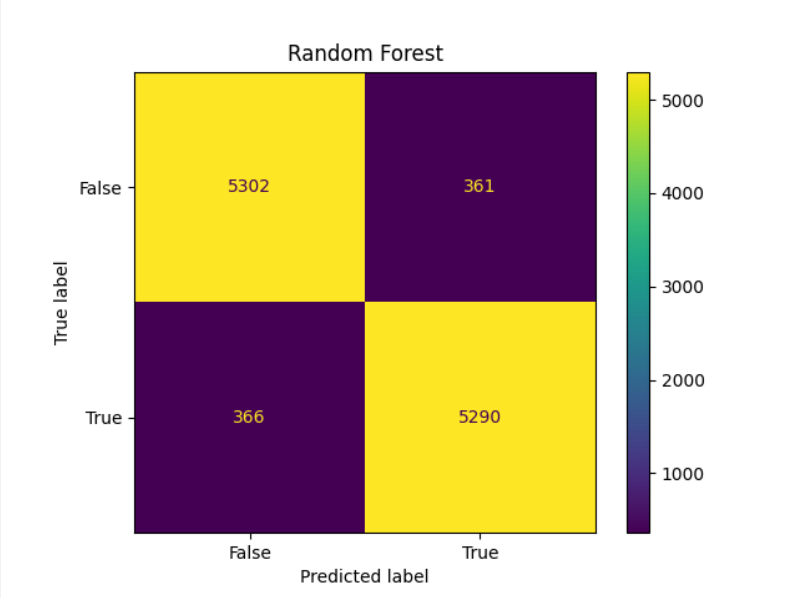


Figure 3-12: Random Forest Confusion Matrix

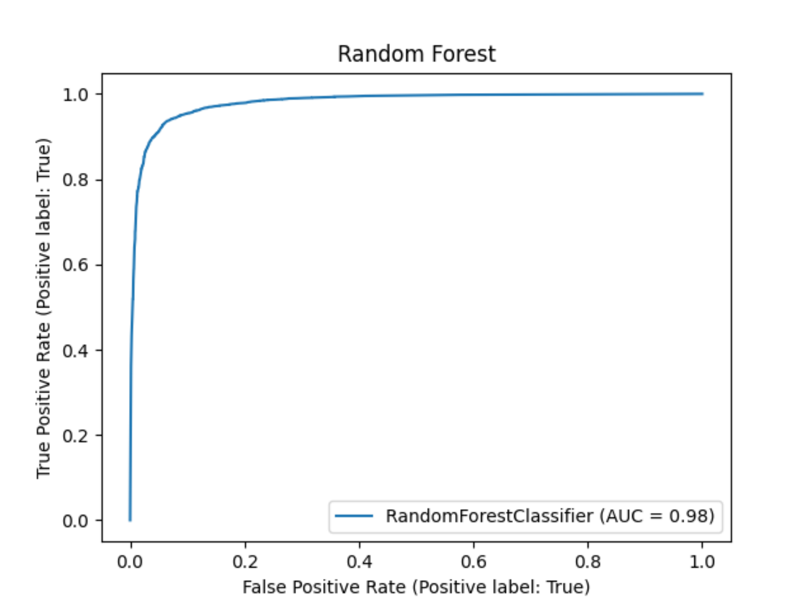


Figure 3-13: Random Forest True Positive Graph

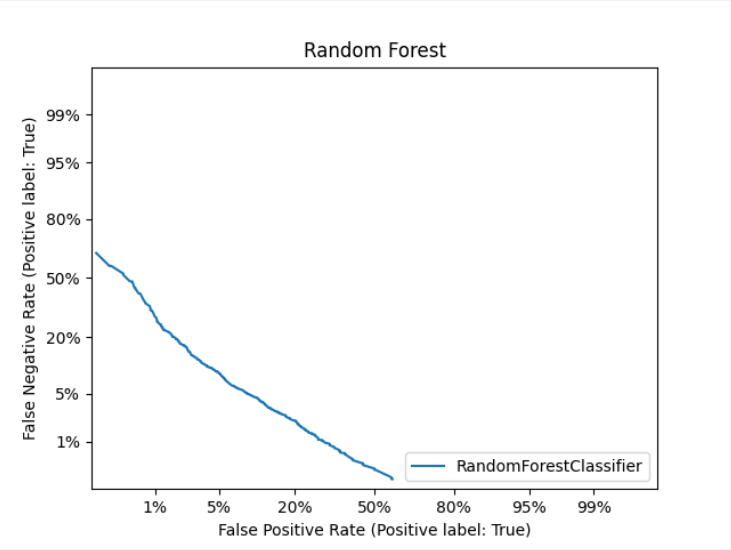


Figure 3-14: Random Forest False Negative Graph

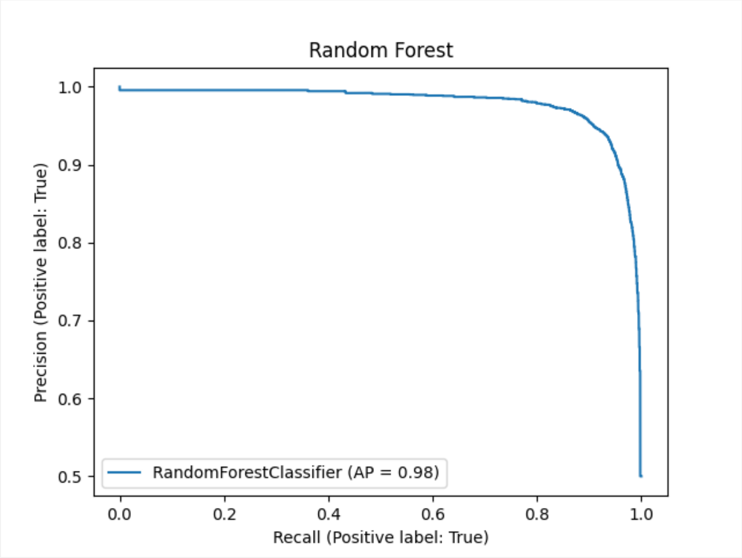


Figure 3-15: Random Forest Precision-Recall Graph

The Performance measurements for the random forest were as follows:

Sensitivity (recall) score: 0.9352899575671852
precision score: 0.9361175013271987
f1 score: 0.9357035464756346
accuracy score: 0.9357717112819154
ROC AUC: 0.9799541385169519
Scores using 5-fold cross validation: [0.9326339 0.93175041 0.93705135 0.93219216 0.92932082]
Average score using 5-fold cross validation: 0.933

3.3.2. Bernoulli Naïve Bayes

The Confusion matrix for the Naïve Bayes classifications is displayed in figure 3-16, with the true positive rate graph in figure 3-17, false negative rate graph in figure 3-18, and the precision recall graph in figure 3-19.

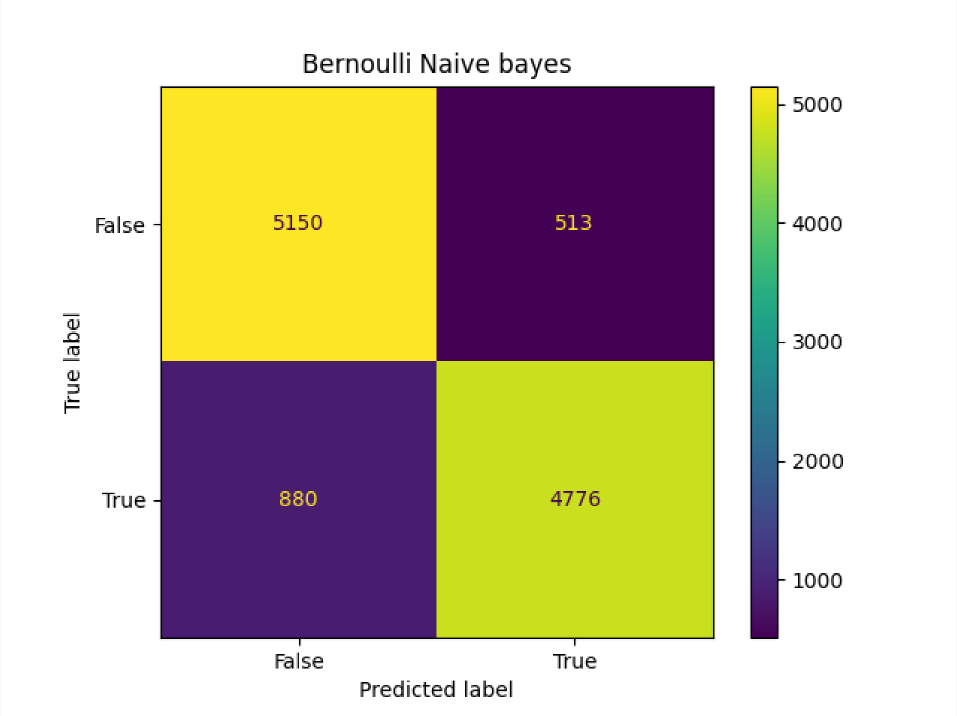


Figure 3-16: Naïve Bayes Confusion Matrix

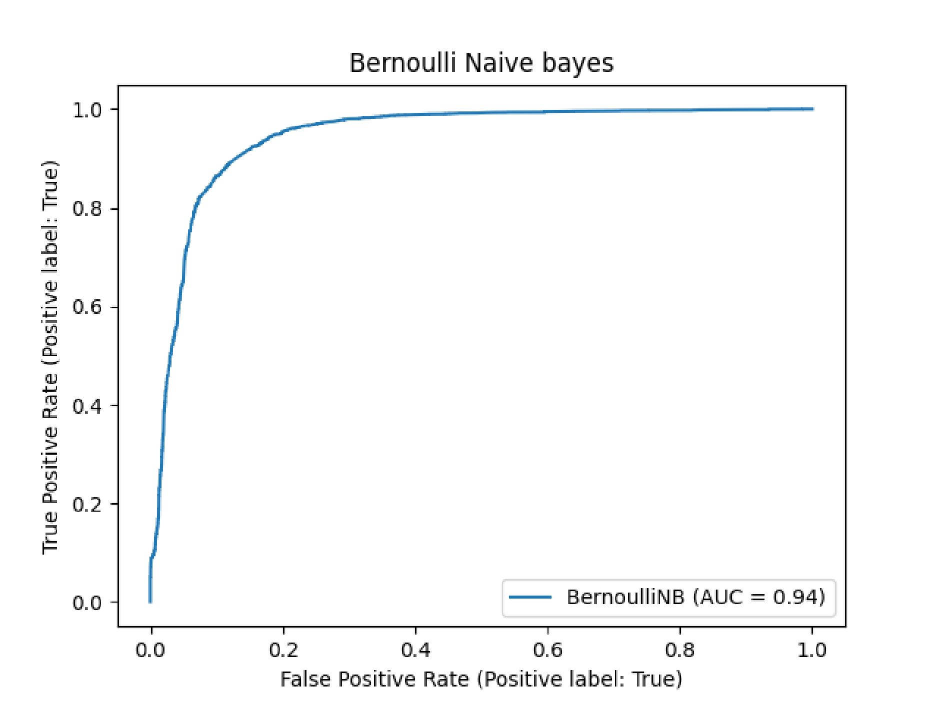


Figure 3-17: Naïve Bayes True Positive Graph

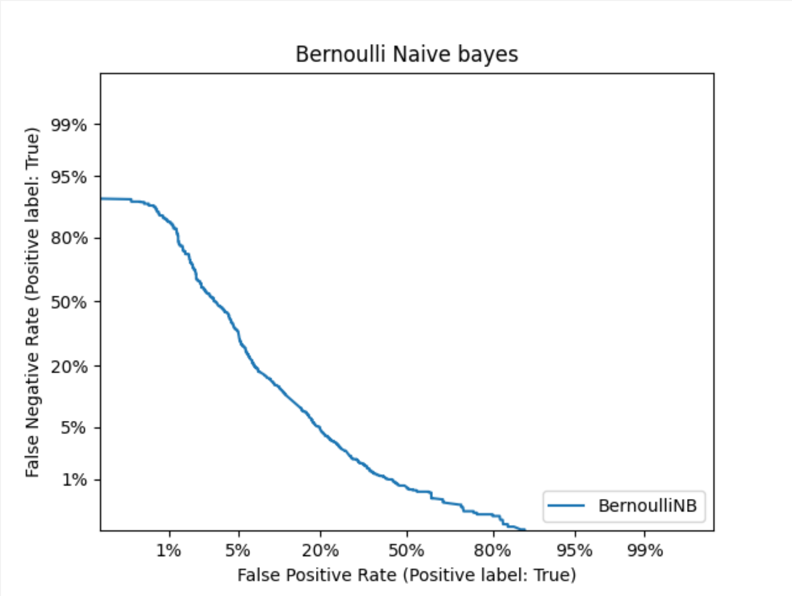


Figure 3-18: Naïve Bayes False Negative Graph

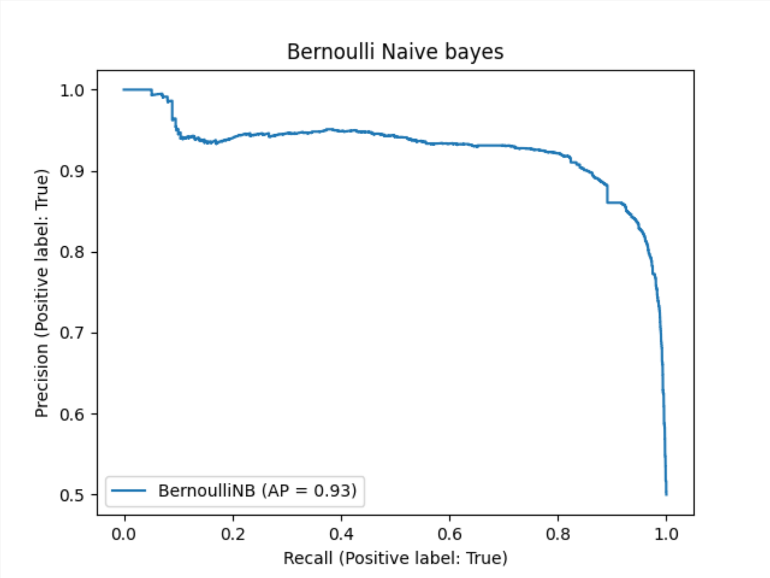


Figure 3-19: Naïve Bayes Precision-Recall Graph

The Performance measurements for naïve Bayes were as follows:

Sensitivity (recall) score: 0.8444130127298444
precision score: 0.9030062393647192
f1 score: 0.8727272727272726
accuracy score: 0.8769325912183055
ROC AUC: 0.9440795183804347
Scores using 5-fold cross validation: [0.87818885 0.8725566 0.8753175 0.87410271 0.87553838]
Average score using 5-fold cross validation: 0.875

3.3.3. Multi-Layer Perceptron

The Confusion matrix for the Multi-Layer Perceptron classifications is displayed in figure 3-20, with the true positive rate graph in figure 3-21, false negative rate graph in figure 3-22.

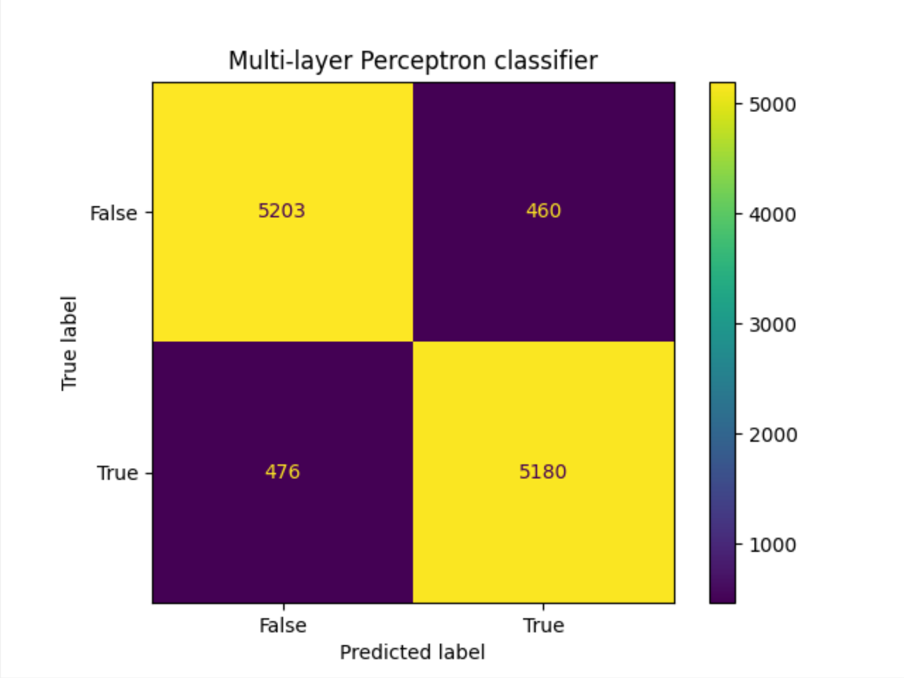


Figure 3-20: Multi-Layer Perceptron Confusion Matrix

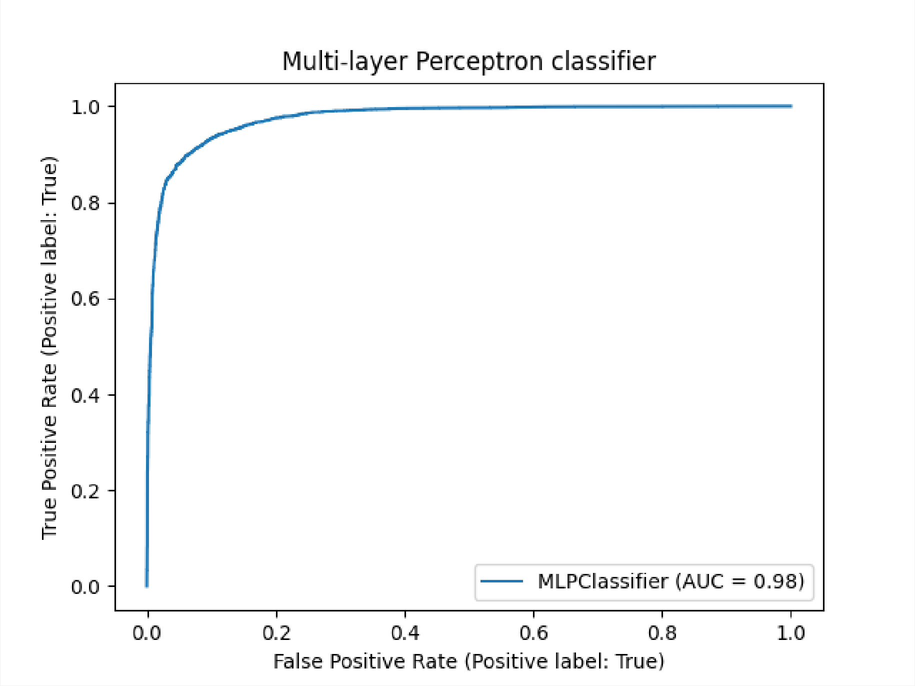


Figure 3-21: Multi-Layer Perceptron True Positive Graph

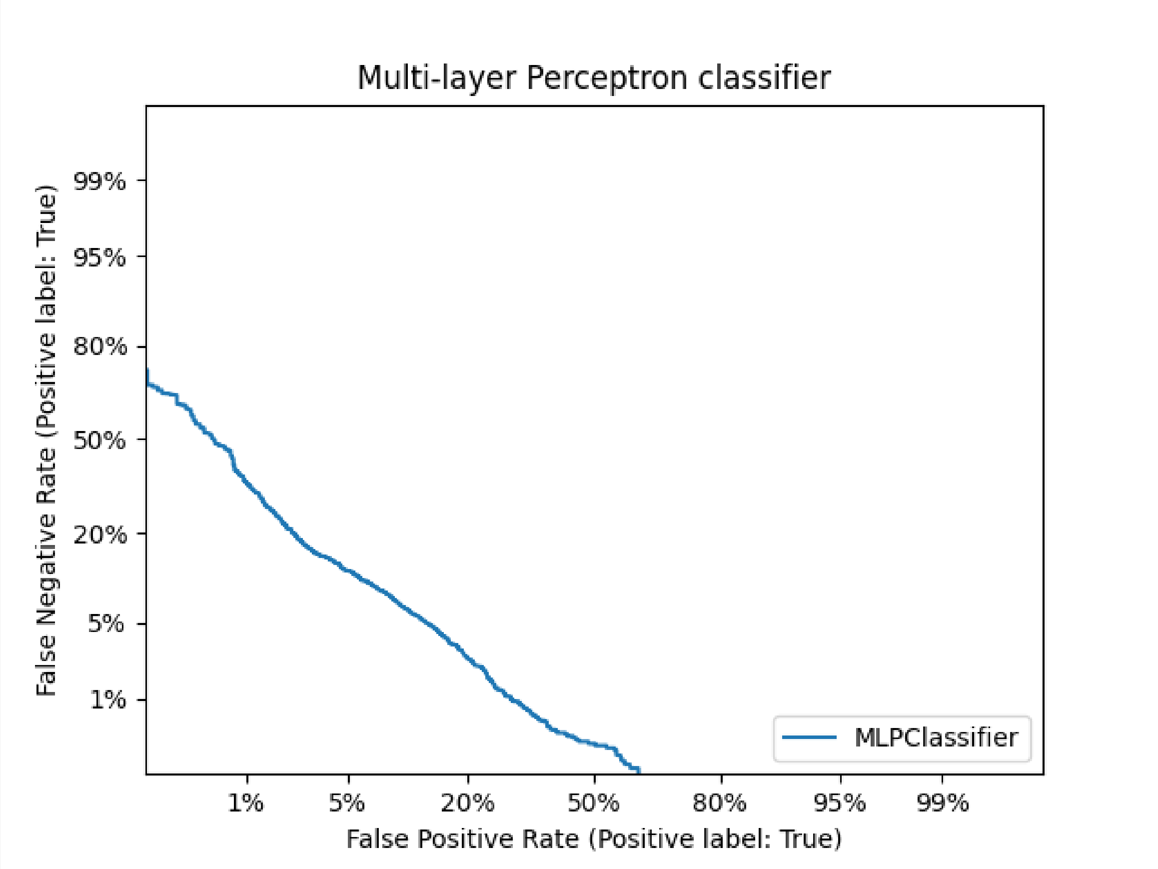


Figure 3-21: Multi-Layer Perceptron False Negative Graph

The Performance measurements for the Multi-Layer Perceptron were as follows:

Sensitivity (recall) score: 0.9158415841584159
precision score: 0.9184397163120568
f1 score: 0.9171388101983002
accuracy score: 0.917307182613305
ROC AUC: 0.9751695976338131
Scores using 5-fold cross validation: [0.92269464 0.92203203 0.91728327 0.91109884 0.91540585]
Average score using 5-fold cross validation: 0.918

3.3.4. Comparing Between Classifiers

First it is noticed that the accuracy's highest value was less than 94%, and that's because most of the features are extracted from the emojis and few emojis were extracted from the text. In order to get better performance of the model, useful features must be extracted from the text.

The performance difference in the three classifications can be found in table 3-2 below.

Table 3-2: Classifications Performance Comparison

Classification	Accuracy	Precision	Recall	F1	ROC AUC	Avg using 5-fold cross validation
Random Forest	0.936	0.936	0.935	0.936	0.98	0.933
Naïve Bayes	0..878	0.903	0.844	0.873	0.944	0.875
Multi-Layer Perceptron	0.917	0.918	0.916	0.917	0.975	0.918

As seen in table 3-2 the performance measurements for the random forest were very close, same thing for the Multi-layer perceptron, yet the Naïve Bayes classification gave 90% precision rate, yet 84% recall rate.

From the table it is obvious that the random forest has resulted in the best performance of all the three classifications as it gave the best precision, recall, and ROC AUC, where the Multi-layer Perceptron came in second, and Naïve Bayes came in last.

4. Conclusion

In conclusion after working on this project, a better understanding of machine learning mechanism is achieved, as it showed how computers take the smallest details in the data set and converts them in short time into classifications and decisions.

this project also gives a better understanding of the importance and the needs of its stages, where data preprocessing was implemented in order to give the computer a clear data that can result in giving more accurate classifications, feature extraction shows what are the important and needed things in the data for its classification, training is what teaches the system to become familiar with the data, and testing tells the situation of the entire systems quality.

Machine learning and natural language processing are currently playing a big role in the world today as most likely everything depends on them in either (banks, governments, medicine, social medic, etc..), as they made life easier and saved peoples time in data gathering and making decisions, as the computer does things in a way faster time.

5. References

[1]. Arabic words vocalization – arabDict.com:

https://www.arabdict.com/en/english-arabic

Accessed on February 15th 2023, at 12:01 AM

[2]. Stemming – researchgate.net:

https://www.researchgate.net/publication/320607773_Arabic_information_retrieval_Stemming_or_lemmatization Accessed on February 15th 2023, at 12:40 AM

[3]. Feature Extraction – medium.com:

https://medium.com/analytics-vidhya/feature-extraction-and-challenges-a1e4f3f4cb53

Accessed on February 17th 2023, at 1:40 AM

[4]. TF-IDF –kinder-chen.medium.com:

https://kinder-chen.medium.com/introduction-to-natural-language-processing-tf-idf-1507e907c19 Accessed on February 17th 2023, at 2:27 AM

[5]. Training and Testing – researchgate.net:

https://www.researchgate.net/figure/Training-and-testing-our-machine-learning-approach_fig2_318132501 Accessed on February 17th 2023, at 3:14 AM