# Exploring how emojis and emoticons affect sentiment analysis

Serghei Socolovschi Angel Igareta
serghei@kth.se | angel@igareta.com

February 12, 2021

# Abstract

In the 21st century, communication has shifted due to the spread of new technologies and social networks. In order to highlight their emotions, people often resort to pictograms, such as emoticons and emojis. With the increase in the number of messages exchanged on the Internet, the interest in understanding the emotions of users also rises, leading to the development of Natural Language Processing (NLP) techniques that help to analyze the sentiment polarity of a text. However, there is no agreement regarding the inclusion of pictograms in the analysis. The existing research does not provide any clear and definitive results regarding the impact of emojis and emoticons in the sentiment analysis, due to either the datasets or methodology chosen.

This research aims to investigate the impact of emojis and emoticons in sentiment analysis. The experiments were conducted using a public dataset of American airlines Twitter reviews [1], which were manually labeled into three categories: positive, neutral, and negative. After thorough preprocessing and feature extraction stages, several versions of training and test sets were created, such as keeping or removing the pictograms, transforming them into words, etc.

The data was used to train a collection of machine learning techniques, such as Support Vector Machine (SVM), Multinominal Naive Bayes (MNB), K-Nearest Neighbors (KNN), Decision Tree (DT), and Randon Forest (RF). Finally, the trained models were evaluated according to performance metrics (accuracy, f1-score, AUC, and time) to define the best configuration.

The results proved that pictograms could be used to improve sentiment analysis. Out of the five machine learning classifiers tested, three of them (SVM, MNB, and KNN) had a significant improvement when adding emoticons, while regarding the inclusion of emojis only KNN obtained better results.

**Keywords:** sentiment analysis, emojis, emoticons, machine learning, classifiers, tweet analysis

# Contents

# Chapter 1

# Introduction

## 1.1 Background

Sentiment analysis is the field of Natural Language Processing (NLP) that studies the polarity of the text. Several types of classification can be performed: the analysis of subjectivity, which seeks to understand if the text is subjective or objective, the analysis of polarities that aims to understand if the sentences invoke a positive, negative or neutral emotional reaction, or a multi-class analysis that maps the text with a particular emotion (for example, joy, fear or sadness). It is considered to be a powerful tool to predict the emotional response of users by analyzing their posts. The sentiment analysis can be applied to study the brand reputation, stock markets, and the public reaction to world events.

With the growth of micro-blogging social media platforms, such as Twitter, the researchers can get access to a massive amount of data to analyze. Emojis and emoticons represent a strong clue when it comes to the analysis of the texts. Most of the time, they are intensifying the conveyed sentiment. However, in certain instances, they can allude to hidden meanings and irony. Hence, it might be interesting to include them in sentiment analysis to obtain a better understanding of the text.

Machine Learning classification is one of the approaches used in sentiment analysis. It is done by splitting the text into units, extracting the features, and labeling the input dataset. It is important to carefully separate the data into training, validation, and test sets in order to avoid over-fitting. The quality of the final model relies on multiple factors, such as data preprocessing, feature extraction, and validation techniques. Machine learning techniques were proven to operate better in the domain in which they were trained [2]. Some of the most common techniques are:

- Support Vector Machine (SVM) is a classification algorithm that plots the data points in the space in such a way that the points belonging to different categories are divided and have the maximum margin from the separation plane. The new points are classified depending on which side of the separation plane they fall. In this research, the 'kernel trick' will be used to plot inputs into high-dimensional feature spaces.
- Naive Bayes (NB) is a probabilistic classifier that proved to be particularly efficient for sentiment analysis. The probability of input belonging to a certain class can be computed using the Bayes' Theorem. The Naive assumption considers all the features in the input to be independent, highly simplifying the computation of the probability. There are different implementations of the algorithm. In this study, the Multinominal Naive Bayes will be used.
- K-Nearest Neighbors (KNN) is a non-parametric algorithm used for classification. After receiving the input, the algorithm finds its K-nearest neighbors and assigns the label that matches the majority of the neighbors.
- Decision Tree (DT) is a graphical tool used for decision making. In a machine learning scenario, it could be applied to classification tasks, using the features to partition the dataset.
- Random Forest (RF) is an ensemble learning method that operates by constructing multiple trees and outputting the class that is voted by the individual trees. Compared to the DT model, RF tends to overfit less over the training dataset.

### 1.1.1 Emoticons and Emojis

Emoticons are sequences of text symbols meant to represent facial expressions and emotions, for example ':)' or ':(', making its first appearance on the Internet in the year 1982 [3]. Emojis can be considered as a further evolution of emoticons, which were first introduced in Japan in 1999. By maintaining all the expressions present in the emoticon lexicon, emojis added new concepts, such as natural elements, food, and sport into the vocabulary. Both of these symbol representations became popular in text-based communication through the last years and led to the implementation of emoji keyboards on the major mobile operating systems, Apple iOS and Android.

## 1.2 Theoretical Framework

For this research, the literature review was focused mostly on sentiment analysis studies that were using machine learning classifiers. In [4], it was proved that emoticons, and especially keywords related to particular emotions (love, joy, sadness, and anger), could improve sentiment analysis. In the experiments, a corpus of 750.000 automatically labeled Greek forum messages was processed and used to train several ML classifiers (e.g., MNB, Bernoulli Naive Bayes (BNB), Logistic Regression (LR), SVM, and KNN), reaching a 77% and 93% of accuracy in average, using emoticons and keywords as features respectively.

A similar study was performed by LeCompte and Chen [5], where the researchers were trying to classify 54.000 tweets into seven broad emotional classes (sad, angry, happy, scared, thankful, surprised, and love), labeling the data using a method proposed by Wang et al. [6]. Once the data was processed, several training sets were generated, including or excluding emojis from the analysis, or by using unigram or bigram features. The sets were used to train MNB and SVM models. The best results were achieved by an MNB model using unigram features (63.5% accuracy in case the emojis were ignored and 64.2% when emojis were included in the training phase).

In another study [7], the emojis were confronted with the slang words used on Chinese social networks. A dataset composed of 3000 posts, labeled in "humorous" and "non-humorous" classes, was used to train a wide range of classification algorithms, including also ML techniques, such as LR, SVM, NB, KNN, RF, and DT. The best performance was achieved by SVM, and emojis were leading to better results than slang features (f1-score of 73.77% and 72.61% respectively), however, the best results were achieved by combining both types of features (74.74%).

Another group of researchers [8] studied how sarcasm can be detected using sentiment analysis techniques. A collection of 2000 tweets, labeled into two classes ("sarcastic" and "non-sarcastic"), was used to train DT, RF, Gradient Boosting, Adaboost, LR, and NB classifiers. The best results were achieved by using the Adaboost ensemble technique, reaching 81,7% accuracy, when emojis and slang were labeled with more common words, and 80,2% when emojis and slang were left untouched.

The impact of the emoticons in sentiment analysis was also analyzed by Palsson and Szersze [9], using a dataset of 4200 manually labeled tweets. Among ML techniques, BNB and SVM were chosen. The models were trained using two types of training sets, one in which emoticons were untouched and another one where they were substituted with a word label. The best performance among ML techniques was achieved by SVM with an f1-score of 65,5%, in case the emoticons were transformed into words.

## 1.3 Problem Discussion

The main outcome of the literature review was that the research on the influence of emoticons and emojis in sentiment analysis has been approached very differently by researchers. There are well-structured studies, such as [4][7], which convey interesting results, however, they were using Greek and Chinese datasets. The main issues in the studies performed on other languages were different linguistic structure, cultural bias and, occasionally, different use of emojis. The multi-lingual sentiment analysis was defined as one of the next steps in the research.

Besides that, most of the studies exploited different classifiers to evaluate the performance of their methods. Furthermore, many research were either addressing other types of sentiment analysis, such as a multi-class emotional labeling [4][5] or a more niche one regarding humor [7] or sarcasm [8], resulting in fewer studies that consider the same type of classification used in this study.

Moreover, the pictogram treatment was varying from study to study. The comparison was created either by eliminating them or by transforming them into words. Lastly, all the previous works were considering either emoticons or emojis. Even though one might expect that there might be similarities in using one or the other type of symbols, there was no study found that would compare both elements.

As the result, it was difficult to find a common ground between the findings of the presented papers, showing that there are several topics that could be studied more in detail.

In this study, a series of experiments will evaluate the influence of different types of pictograms used in online communication. The research will include the training of a wide spectrum of ML classifiers, following the best practices described in the analyzed papers and adapting them to the research question. The report will present a broader review of classifiers and their performances on the preestablished task, showing which kind of pictogram improves the sentiment score of the text and in what measure.

### 1.3.1 Research Question

The main question the research is attempting to answer is the following:

***How does the inclusion of pictograms, such as emoticons and emojis, affect the sentiment analysis performed by machine learning classification algorithms?***

# Chapter 2

# Research Methodology

## 2.1 Choice of Research Method

The *empirical method* will be used for accomplishing the research tasks, as the research will pursue the study of previous knowledge in sentiment analysis and analyze the outcomes of the experiments [10].

This study will attempt to give an answer to the research question, using quantitative methods and publicly available datasets. After the preprocessing of the textual data, different versions of the dataset will be created, by omitting or transforming the emojis. Next, the datasets will be used to train a set of models. In addition to training, the tuning of hyper-parameters will be done to select the best configuration for each algorithm. As the last step, the models will make predictions on the test set, and, based on the performance metrics, it is expected to not only show the impact of pictograms in sentiment analysis, but also which classifiers perform better the task.

## 2.2 Dataset Selection

Due to the nature of this research, the following requirements must be met for the dataset to be adequate:

- It needs to include a large amount of text containing emoticons or emojis. This is very important as the research aims to see how these pictograms affect the sentiment of the text.
- It must contain a numeric or binary sentiment label for each text in the dataset. Also, this label should be annotated manually rather than automatically using a sentiment analysis technique or based on a score, as these data may not be reliable.

Most of the public sentiment analysis datasets found during the research can be divided into two categories:

1. Datasets have been automatically annotated using ML techniques. As stated above, this data might not be reliable as they are using sentiment analysis techniques to calculate the polarity of the sentence. Some examples are IMDB dataset [11] or Sentiment140 dataset [12].
2. Datasets containing reviews of products or services. Instead of including the polarity of the text, they include the user's score on a scale of 1 to 5. Similar to the previous case, this data might contain a large number of outliers that give positive feedback but not the highest score. Some examples are Amazon Fine Food Reviews [13] or 515K Hotel Reviews Data in Europe [14].

After thorough research, a suitable dataset was found that met the presented requirements. The dataset is called *'Twitter US Airline Sentiment'* [1] and it analyzes how travelers in February of 2015 expressed their feelings of each major U.S. airline on Twitter. The text was manually labeled by contributors into positive, negative, and neutral tweets.

The dataset contains a total of 14.640 tweets, which is a relatively small size compared to other popular sentiment analysis datasets. This might be due to the difficulty of manually labeling the polarity of the text. Also, about 5% of the tweets include either an emoticon or emoji, which is an adequate density for the purpose of the research.

Of the 15 columns in the original dataset, only 3 were selected for the research. By making this selection the ethical problems concerning the data are to be avoided since sensitive columns such as the location of the tweet and the name of the user will not be used. The selected columns are:

- Text: Tweet response to a major U.S. airline on Twitter, starting with a mention of the airline.
- Airline Sentiment: Indicates the sentiment polarity of the corresponding tweet.
- Airline Sentiment Confidence.

## 2.3 Dataset Preprocessing

To use the dataset as input for the machine learning techniques and improve its quality, the subsequent preprocessing tasks were performed.

### 2.3.1 Handling Duplicates and Null Values

The dataset did not contain null values in any of the selected columns but there was the presence of duplicate values, specifically 213 rows were duplicated. These were common responses to airlines such as 'thank you!' or 'thanks'. The duplicated rows were removed.

### 2.3.2 Data Distribution Normalization

As previously stated, only a 5% of the tweets contain a pictogram. Hence, to generate a more uniform dataset and avoid that models consider these elements as outliers, the original distribution was transformed to contain the same quantity of tweets with a graphical representation as those without. The resulting dataset contains 1320 tweets, where a 50% of them contain either an emoticon or an emoji.

### 2.3.3 Data Cleaning

To perform NLP for sentiment analysis and achieve good results, it is necessary to clean up the input data and then decompose the text into a format understandable to ML techniques. In this case, the data are tweets, which usually contain elements such as mentions, URLs, hashtags, emoticons, or emojis. However, instead of making an initial decision on whether to remove or to keep a certain element, an early hyper-tuning step will be performed. This section will present all the data processing done and which of these elements will be used for the early hyper-tuning stage.

The preprocessing steps performed to the dataset are:

- Spacing: Sequential spaces or newlines in a tweet are replaced by a single space.
- Punctuation: As punctuation can make the text difficult to understand, it is completely removed except for the punctuation that forms the emoticons, which was kept.
- URLs: These elements are removed from the original dataset as they are not related to sentiment.
- Accents from the text: Usually in the text, accented characters/letters can appear, which would be seen by the model as different characters, for example: 'é' and 'e'. These need to be standardized into ASCII characters.
- Expand contractions: Contractions are shortened versions of words. It is common to find them in tweets as the language style is usually informal. Each contraction was transformed to its expanded form to help with text standardization.
- Lemmatization: This approach allows to extract the root forms of the words in the text, thus generating more occurrences of the same meaning of the word, which assists in the standardization of the text. Lemmatization was selected instead of stemming because speed is not a major concern in this case and the result is more representative of the type of text being used.
- Sentiment Confidence: Texts with sentiment confidence below a 60% were removed to avoid misclassification.

The preprocessing tasks to be considered as hyper-parameters in an early hyper-tuning stage are:

- Emojis and emoticons: The core value of the research, the main hyper-tuning will be done by processing the dataset, eliminating or maintaining each of these elements.

- Textual representation of emojis and emoticons: Instead of using the graphical representation of either an emoticon or emoji, it is also interesting to study the effect of the textual representation. For instance, if the input text contains ':)', the transformed text will contain 'happy_face_or_smiley' instead.
- Mentions and hashtags: Include or remove these elements to study their effect.
- Lowercase: Transform or not every character in the dataset to lowercase. It was chosen as a hyper-parameter as negative sentiment could be represented using uppercase text.
- Stop words: Include or remove words with little significance in the text. Typically, these can be articles, conjunctions, prepositions, and so on. Some examples of stop words are a, an, the, and the like.
- Sentiment representation: In sentiment analysis, the polarity of the text can be represented through a binary form where 0 is negative and 1 is positive or through a numeric form adding a neutral figure. During the hyper-tuning, sentiment in both forms will be tested.

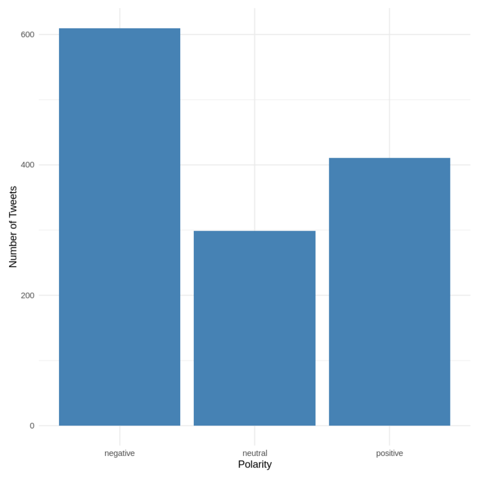
### 2.3.4 Data Preparation

As a final step, once the data has been preprocessed, it is necessary to convert each set of words to a vector to provide it as input for the machine learning models. In order to do so, two vectorizers will be used in the hyper tuning:

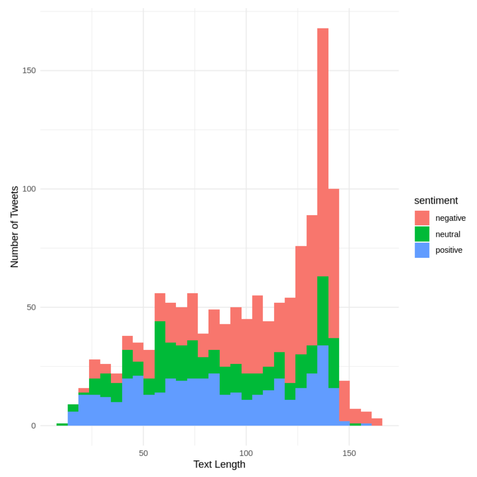
- Count: This will transform the text in our dataset into a bag of words model, which will include the occurrence of each word within the document. This approach can be expanded to use bi-grams and n-grams, that analyze the presence of longer sequences of words in the text. In this case, both unigrams and bigrams were used.
- Tf-idf (term frequency times the inverse document frequency): Here, a numerical statistic is used to reflect how important a word is to a document in a collection or corpus. It aims to reduce the impact of very frequently occurring tokens that might be less informative than the characteristics that occur in a small fraction of the corpus.

### 2.3.5 Data Preprocessing Results

After the preprocessing, the distribution of the tweet polarity can be observed in figure 2.1a, which represents how the sentiment is distributed among the tweets in the dataset. It is possible to observe that they are relatively balanced, although a considerable amount of them are negative. In addition, figure 2.1b shows how the sentiment is distributed attending to the tweet length, where the longest tweets usually correspond with the negative class.

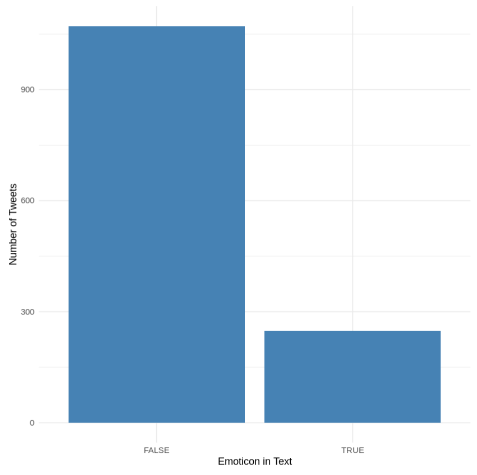


(a) Tweet Sentiment Distribution

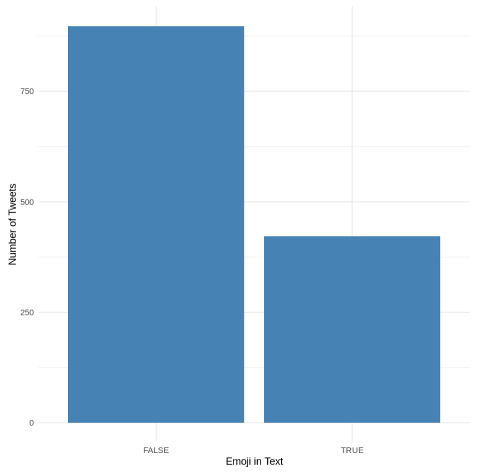


(b) Tweet Length Distribution

The final distribution for emojis and emoticons can be appreciated in the following figure, where out of 1320 rows, a 18.89% of them contain emoticons and a 31.7% contain emojis.



(a) Emoticon Distribution



(b) Emoji Distribution

## 2.4 Application of Research Method

In order to see the impact of pictograms in sentiment analysis, different classification techniques will be used, already presented in the introduction. All of them were implemented using the Machine Learning library sklearn, which offers different classification algorithms [15]. The next list summarizes the parameters presented in the previous section with their possible values, that will be used for generating combinations to find the best results for the task of sentiment analysis and how models are affected by the different features.

1. Sentiment type: ['binary_sentiment', 'numeric_sentiment']
2. Include mentions: [True, False]
3. Include hashtags: [True, False]
4. Include stop words: [True, False]
5. Transform text to lowercase: [True, False]
6. Emoji included: [True, False]
7. Emoticon included: [True, False]
8. Transform emojis into text: [True, False]
9. Transform emoticons into text: [True, False]
10. Vectorizer: ['tf-idf', 'count']
11. Model: ['svm', 'mnb', 'knn', 'dt', 'rf']
    - SVM: Regularization: [1, 3] — Kernel: ['linear', 'rbf']
    - MNB: Alpha: [1,3] — Fit prior: [True, False]
    - KNN: Number of neighbors: [8, 13] — Weights: ['uniform', 'distance']
    - DT: Max depth: [10, 20] — Criterion: ['gini', 'entropy'] — Max features: ['sqrt', 'log2']
    - RF: Number of trees: [10, 20, 50] — Max depth: [10, 20] — Criterion: ['gini', 'entropy'] — Max features: ['sqrt', 'log2']

The combinations possible with these parameters are about $5 * 2^{10} = 5120$, not including the custom hyper-parameters for each model, which on average would multiply the number of combinations by $2^{\{3,4\}}$. Since this is a significant number of combinations that would demand expensive computational resources, the hyper-tuning was divided into two steps. The first one referred to as 'A-priori Parameter Selection', involves testing all the previous combinations using only one model. From these results, the best values of the parameters in the position 1 to 5 on the list will be used for the general hyper-tuning, which pursues the research goal of studying how emojis and emoticons affect the sentiment polarity of texts. The dataset was divided into an 80-20 train-test split for the first hyper-tuning while a k-fold cross-validation with 3 folds was used in the general one, aiming to have more representative and unbiased results.

The hyper-tuning was implemented using TensorBoard, the TensorFlow visualization toolkit which provides to track and visualize the selected metrics and draw scatter-plots to see how the combinations perform.

### 2.4.1 Metrics

The statistics being collected per each execution are:

- Accuracy: Metric used to measure how many observations were correctly classified, including both positive and negative classes. In imbalanced problems, it could be misleading as one of the classes would have few occurrences and that is not represented in the formula.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

- F1 Score: Metric used to keep a balance between Precision and Recall. It is used when the label follows an uneven distribution, especially in binary classes.

$$F1Score = \frac{2 * Precision * Recall}{Precision + Recall}$$

- Area Under the Curve (AUC): Metric that indicates the tradeoff between true positive rate (TPR) and false-positive rate (FPR). The higher the TPR is and the lowest the FPR, the better is the result.

- Execution time: Training and testing time.

### 2.4.2 A-priori Parameter Selection

This section presents the results using Multinomial Naive Bayes algorithm with $2^{10}$ combinations about the pre-processing of the data and $2^2$ combinations regarding model hyper-parameters, resulting in a total of $2^{12} = 4096$ combinations. The metrics selected to compare these results were accuracy and F1-score. Below are the decisions for each parameter, which are based on the graphs in the appendix.

- Sentiment Type: On average, the use of the label sentiment binary performed significantly better both in accuracy and f1-score so it will be used for the general hyper-tuning.

- Include mentions: The results when including the mentions in the preprocessed dataset show slightly better results. On the other side, if the implemented models were to be used with another dataset, probably mentions should be removed, to improve generalization. However, because of the nature of the selected dataset, (responses mentioning specific airlines) the addition of mentions improves the results.

- Include hashtags: The inclusion of hashtags did not show any improvement in the results so they will be removed from the text.

- Include stop words: No clear distinction in the results can be appreciated with the addition of stop words, therefore they will be removed, as proposed by most of the researched literature, to reduce the amount of text that will be processed by the models.

- Transform text to lowercase: Since there is no significant difference between the metrics when transforming the text to lowercase, the text will be converted to the lower case because the best results were achieved with it and it is a commonly used approach in NLP.

# Chapter 3

# Results and Analysis

## 3.1 General Hyper-tuning

By fixing the parameters from the previous section, the combinations left are $5 * 2^5 = 160$ multiplied by the hyper-parameters per model, which are around $2^{\{3,4\}}$, resulting in approximately 1280 to 2560 total combinations. Since this does not demand extreme computation resources, all of the combinations were included in the same hyper-parameter tuning session allowing us to have an overview of how all the models perform altogether.

This section will begin by presenting the individual result per machine learning classifier and finish with an overview that groups them all. The charts obtained from the individual results can be found in the appendix.

### 3.1.1 Support Vector Machine

Regarding the selected hyper-parameters, the results showed that the use of a 'linear' or 'rbf' kernel does not have a strong difference and that the parameter '2' for regularization performs better. In respect of the pictogram influence over the results, the inclusion of emoticons showed slightly better results than not including it, obtaining approximately 0.1% more of accuracy. On the other side, the results behaved similarly when including or removing emojis.

| **emoticon_included** | **emoji_included** | **svm_kernel** | **svm_regularization** | **accuracy** | **fl -score** | **auc** | **time** |
|---|---|---|---|---|---|---|---|
| true | true | linear | 2.0000 | 0.82613 | 0.82606 | 0.82467 | 0.80735 |
| true | false | rbf | 2.0000 | 0.82535 | 0.82547 | 0.81838 | 0.88668 |
| true | true | linear | 2.0000 | 0.82458 | 0.82437 | 0.82429 | 0.81278 |
| true | true | rbf | 2.0000 | 0.82458 | 0.82452 | 0.82059 | 0.89181 |
| true | true | linear | 1.0000 | 0.82304 | 0.82300 | 0.82481 | 0.83394 |

Table 3.1: SVM - Top 5 combinations

### 3.1.2 Multinomial Naive Bayes

The results indicated that the use of a higher alpha performed slightly better and also when learning the prior probabilities. MNB is one of the few models where there is a clear improvement by keeping the emoticons in the text. Because of that, the parameter that transformed the emoticons into their textual representation was also analyzed to see if the transformation changed the results, but it showed no correlation. In addition, including or removing emojis did not have any significant effect on the result.

| emoticon_included | emoji_included | mnb_fit_prior | mnb_alpha | accuracy | f1-score | auc | time |
|---|---|---|---|---|---|---|---|
| true | true | false | 2.0 | 0.82303 | 0.82260 | 0.82007 | 0.0060273 |
| true | false | true | 2.0 | 0.82303 | 0.82306 | 0.82266 | 0.0061757 |
| true | true | false | 2.0 | 0.82149 | 0.82119 | 0.81915 | 0.0074159 |
| true | false | false | 2.0 | 0.82072 | 0.82000 | 0.81675 | 0.0084571 |
| true | false | false | 2.0 | 0.81994 | 0.81948 | 0.81682 | 0.0058525 |

Table 3.2: MNB - Top 5 combinations

### 3.1.3 K-Nearest Neighbors

In general, the use of 'uniform' weights obtained marginally better results when the number of neighbors was not low (more than 8), otherwise 'distance' weights performed better. Concerning pictograms, KNN is the only trained algorithm that obtained better results when both emoticons and emojis were included instead of removed. In addition, transforming both pictograms into text resulted in worse metrics.

| emoticon_included | emoji_included | knn_weights | knn_n_neighbors | accuracy | fl-score | auc | time |
|---|---|---|---|---|---|---|---|
| true | true | uniform | 8.0000 | 0.78748 | 0.78665 | 0.78335 | 0.053573 |
| true | true | uniform | 12.000 | 0.78516 | 0.78247 | 0.77787 | 0.054796 |
| true | true | uniform | 12.000 | 0.78439 | 0.78189 | 0.77742 | 0.050366 |
| true | true | uniform | 10.000 | 0.78130 | 0.77941 | 0.77520 | 0.053341 |
| true | true | uniform | 8.0000 | 0.77821 | 0.77740 | 0.77429 | 0.051730 |

Table 3.3: KNN - Top 5 combinations

### 3.1.4 Decision Tree

According to the results, the different measures of impurity of partitions did not significantly affect the metrics. However, the results improved when the depth of the decision tree was higher, and when the square root of the total features was used for the maximum number of features per tree. The use of either emoticons or emojis did not show any correlation with accuracy or f1-score.

| emoticon_included | emoji_included | dt_criterion | dt_max_depth | dt_max-features | accuracy | fl -score | auc | time |
|---|---|---|---|---|---|---|---|---|
| true | false | entropy | 20.000 | sqrt | 0.67390 | 0.66265 | 0.65963 | 0.0095996 |
| true | true | gini | 20.000 | sqrt | 0.67385 | 0.65854 | 0.65956 | 0.011702 |
| true | true | gini | 20.000 | sqrt | 0.67157 | 0.6621 5 | 0.67489 | 0.013388 |
| true | true | gini | 20.000 | sqrt | 0.66845 | 0.65197 | 0.65445 | 0.010276 |
| true | true | gini | 20.000 | sqrt | 0.66771 | 0.65032 | 0.65002 | 0.012759 |

Table 3.4: DT - Top 5 combinations

### 3.1.5 Random Forest

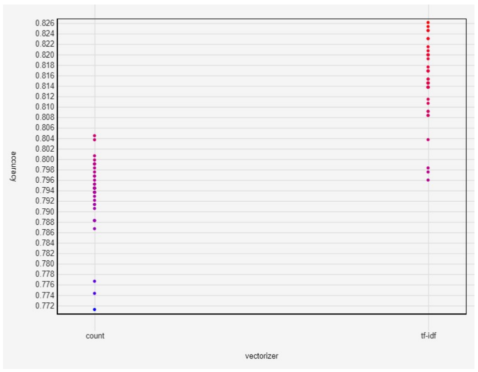
In this model, increasing the maximum depth of the individual trees to 20 obtained better results, as in the DT model. Also, the accuracy decreased with the higher the number of trees was after 10, probably because of the lack of features to assign to the individual trees. Finally, with respect to the influence of the pictograms in the results, despite most of the top 5 combinations include them, the scatter plots did not show a strong correlation.

| emoticon_included | emoji_included | rf_max_depth | rf_max-features | rf_n_trees | accuracy | fl -score | auc | time |
|---|---|---|---|---|---|---|---|---|
| true | true | 20.000 | sqrt | 50.000 | 0.78592 | 0.78432 | 0.78019 | 0.25576 |
| true | true | 20.000 | sqrt | 50.000 | 0.78440 | 0.78186 | 0.77672 | 0.22786 |
| true | false | 20.000 | sqrt | 20.000 | 0.78285 | 0.78094 | 0.77648 | 0.097444 |
| true | true | 20.000 | sqrt | 50.000 | 0.78129 | 0.77892 | 0.77441 | 0.23931 |
| true | false | 20.000 | sqrt | 50.000 | 0.77820 | 0.77365 | 0.76804 | 0.22944 |

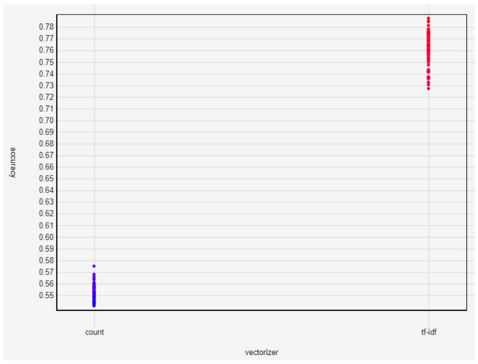
Table 3.5: RF - Top 5 combinations

## 3.2 Conclusions

After analyzing the models individually, it is interesting to have a general view of how these models performed and compare them against each other. With respect of the different vectorizers used, two different clusters of models were found. The MNB, DT and RF models performed similarly in terms of precision and recall. On the other side, KNN and SVM models tf-idf obtained significantly better results.

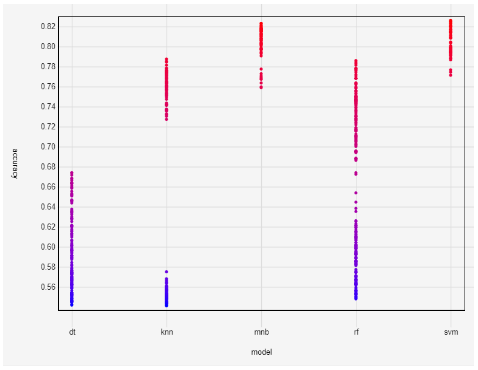


(a) Accuracy per vectorizer in SVM

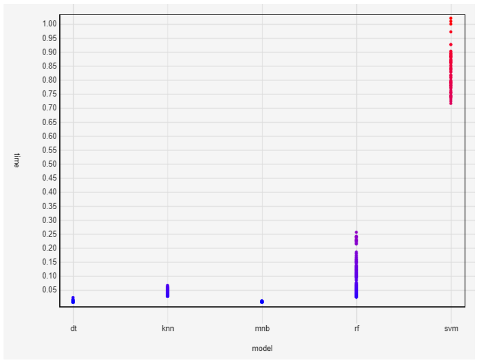


(b) Accuracy per vectorizer in KNN

As a general overview regarding the performance of each model, in terms of accuracy and f1-score, both MNB and SVM have similar results, around 80% in both metrics. However, in terms of speed, MNB was around 140 times faster than SVM. Hence in the trade-off accuracy-time, the model that performed better was MNB.



(a) Accuracy per Model

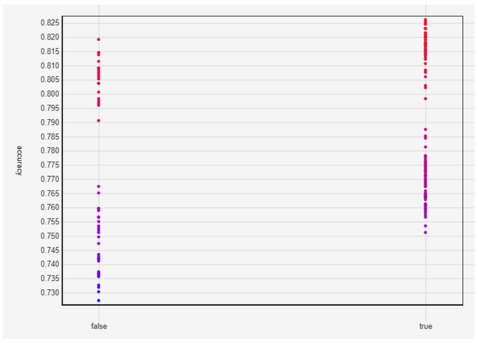


(b) Time per Model

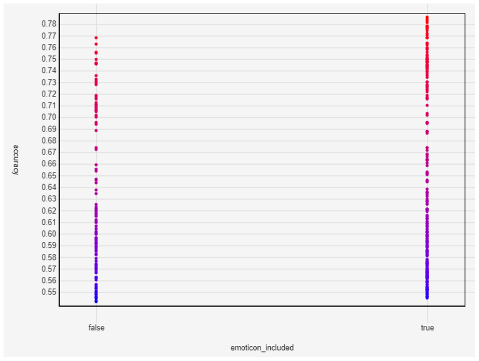
| model | vectorizer | emoticon_included | emoticon_text | emoji_included | emoji_text | accuracy | f1-score | auc | time |
|---|---|---|---|---|---|---|---|---|---|
| svm | tf-idf | true | false | true | false | 0.82613 | 0.82606 | 0.82467 | 0.80735 |
| svm | tf-idf | true | false | false | false | 0.82535 | 0.82547 | 0.81838 | 0.88668 |
| svm | tf-idf | true | true | true | false | 0.82458 | 0.82437 | 0.82429 | 0.81278 |
| svm | tf-idf | true | true | true | false | 0.82458 | 0.82452 | 0.82059 | 0.89181 |
| svm | tf-idf | true | true | true | false | 0.82304 | 0.82300 | 0.82481 | 0.83394 |
| svm | tf-idf | true | false | true | false | 0.82303 | 0.82311 | 0.82319 | 0.82848 |
| mnb | tf-idf | true | false | true | false | 0.82303 | 0.82260 | 0.82007 | 0.0060273 |
| mnb | count | true | true | false | false | 0.82303 | 0.82306 | 0.82266 | 0.0061757 |
| svm | tf-idf | true | true | false | false | 0.82149 | 0.82147 | 0.82370 | 0.86882 |
| mnb | tf-idf | true | true | true | false | 0.82149 | 0.82119 | 0.81915 | 0.0074159 |

Table 3.6: Top 10 combinations

Finally, regarding the research question of how the pictograms affected the models, the addition of emoticons gave better results in the SVM, MNB and KNN models (figure 3.3a), while it did not show significant variation in the DT and RF models (figure 3.3b). On the other hand, the inclusion of emojis only showed a significant improvement in the KNN model, around 0.2%, while in all the other models it obtained worse results. In addition, with respect to the representation of these pictograms, in all the models, most of the best results were obtained using a graphic rather than a textual representation.



(a) Emoticon Effect in Accuracy - SVM, MNB and KNN models (excluding results from count vectorizer)



(b) Emoticon Effect in Accuracy - DT and RF models

# Chapter 4

# Discussion

The results of the experimentation demonstrated that pictograms could be used to improve sentiment analysis. Out of the five machine learning classifiers tested, three of them (SVM, MNB, and KNN) had a significant improvement when adding emoticons, while regarding the inclusion of emojis only KNN obtained better results. Hence, emoticons can be used to better reflect the sentiment of a text, perhaps because nowadays a lesser variety of them is used and emojis can have different meanings, including also the use of irony or sarcasm.

Regarding the preprocessing techniques, the results indicate that some models perform better using the tf-idf vectorizer as input (SVM, KNN), while others have similar results using the counter vectorizer. In respect of the label used to classify the texts, the binary encoding (where neutral texts were considered as positive) worked significantly better in comparison with the numerical encoding. The experiments also showed how the input data can be transformed to obtain better results in tweets and which machine learning classifiers performed better on the selected American Airline Twitter dataset.

Comparing to the previous studies presented in the report, the results were partially in line with what was achieved previously. Similarly to other research, the best performant classifiers were MNB and SVM, with 82.3% and 82.6% accuracy respectively. However, the cross-validation and hyper-tuning helped improve the accuracy by at least 0.8% compared to previous results on other datasets. While it was found that emoticons do improve sentiment analysis, emojis, on the other hand, carried worse results, increasing the accuracy by approximately 0.1%, which was an unexpected outcome and would require further investigation to understand the causes of such behavior.

As future lines of work, it would be interesting to test these results with a larger dataset and also with a higher density of pictograms. During the dataset research, it was found that currently there are no large datasets that have been manually annotated and that contain a large density of emoticons or emojis. Hence, it is suggested to build a dataset that meets these criteria by extracting text from Twitter and manually assigning the sentiment of each tweet. This could help with future research in the area regarding pictograms and sentiment analysis. Additionally, since the preprocessing was performed before the dataset split, it would be recommended to check better some of the preprocessing steps for the textual data, guaranteeing that the models are trained using only the statistical properties of training data and not also of the test or validation sets. Besides that, it could also be useful to study how Deep Learning models would perform in this task, as well as to test more combinations of hyper-parameters for the presented models.

# Bibliography

[1] F. Eight, "Twitter US Airline Sentiment," *https://www.kaggle.com/crowdflower/twitter-airline-sentiment*.

[2] P. Yadav and D. Pandya, "Sentireview: Sentiment analysis based on text and emoticons," in *2017 International Conference on Innovative Mechanisms for Industry Applications (ICIMIA)*. IEEE, 2017, pp. 467–472.

[3] S. Fahlman, "Smiley lore," 2002. [Online]. Available: https://www.cs.cmu.edu/~sef/sefSmiley.htm

[4] G. S. Solakidis, K. N. Vavliakis, and P. A. Mitkas, "Multilingual sentiment analysis using emoticons and keywords," in *2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, vol. 2. IEEE, 2014, pp. 102–109.

[5] T. LeCompte and J. Chen, "Sentiment analysis of tweets including emoji data," in *2017 International Conference on Computational Science and Computational Intelligence (CSCI)*. IEEE, 2017, pp. 793–798.

[6] W. Wang, L. Chen, K. Thirunarayan, and A. P. Sheth, "Harnessing twitter" big data" for automatic emotion identification," in *2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing*. IEEE, 2012, pp. 587–592.

[7] D. Li, R. Rzepka, M. Ptaszynski, and K. Araki, "A novel machine learning-based sentiment analysis method for chinese social media considering chinese slang lexicon and emoticons," in *AffCon@ AAAI*, 2019.

[8] A. G. Prasad, S. Sanjana, S. M. Bhat, and B. Harish, "Sentiment analysis for sarcasm detection on streaming short text data," in *2017 2nd International Conference on Knowledge Engineering and Applications (ICKEA)*. IEEE, 2017, pp. 1–5.

[9] A. Pålsson and D. Szerszen, "Sentiment classification in social media: An analysis of methods and the impact of emoticon removal," 2016.

[10] A. Håkansson, "Portal of research methods and methodologies for research projects and degree projects," in *The 2013 World Congress in Computer Science, Computer Engineering, and Applied Computing WORLDCOMP 2013; Las Vegas, Nevada, USA, 22-25 July*. CSREA Press USA, 2013, pp. 67–73.

[11] Z. Yuan, "IMDB dataset (Sentiment analysis) in CSV format," *https://www.kaggle.com/columbine/imdb-dataset-sentiment-analysis-in-csv-format*.

[12] B. R. Go, A. and L. Huang, "Sentiment140 dataset with 1.6 million tweets," *https://www.kaggle.com/kazanova/sentiment140*.

[13] S. N. A. Project, "Amazon Fine Food Reviews," *https://www.kaggle.com/snap/amazon-fine-food-reviews*.

[14] J. Liu, "515K Hotel Reviews Data in Europe," *https://www.kaggle.com/jiashenliu/515k-hotel-reviews-data-in-europe*.

[15] S. Socolovschi and A. Igareta, "Models Implementation and Experiments - Google Colab," *https://colab.research.google.com/drive/1vKbpJWPZkqPcIBeAYPf2jaxL3L8wVayh?usp=sharing*.

# Appendix

## 4.1 Individual Models

### 4.1.1 Support Vector Machine

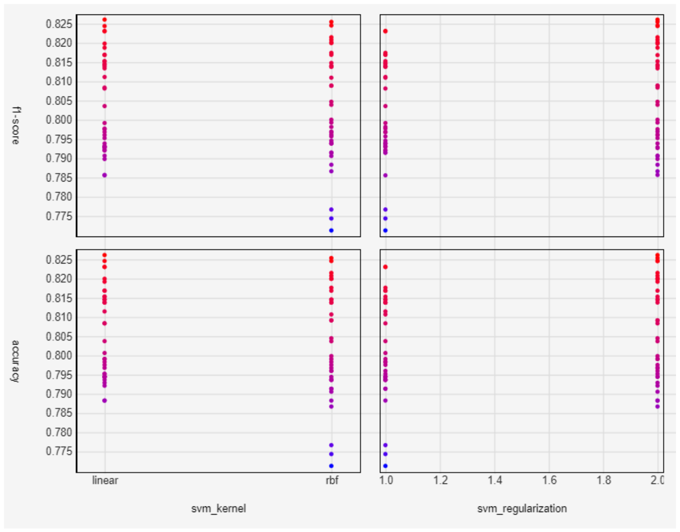


Figure 4.1: SVM - Hyperparameters Influence

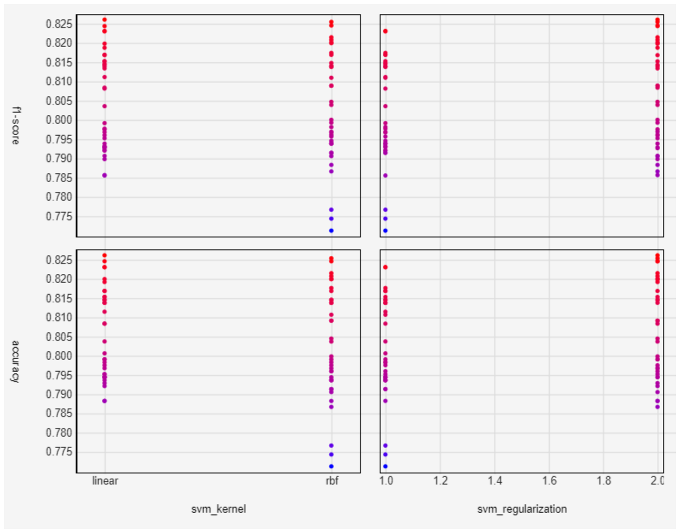


Figure 4.2: SVM - Emoticons and Emojis Influence

### 4.1.2 Multinomial Naive Bayes

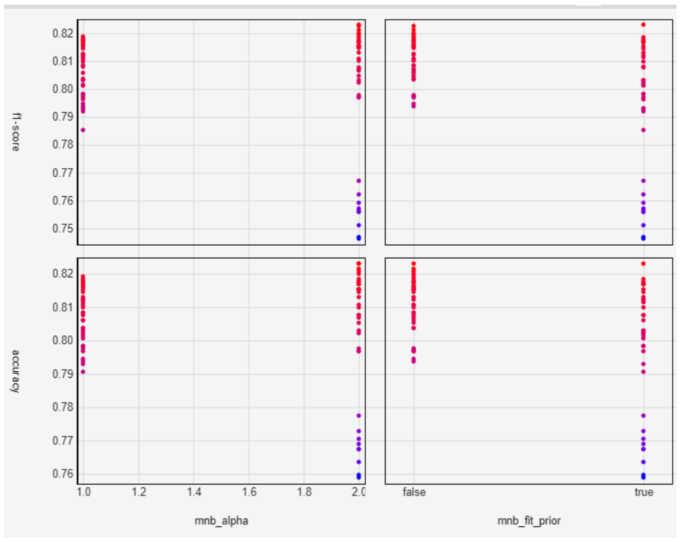


Figure 4.3: MNB - Hyperparameters Influence

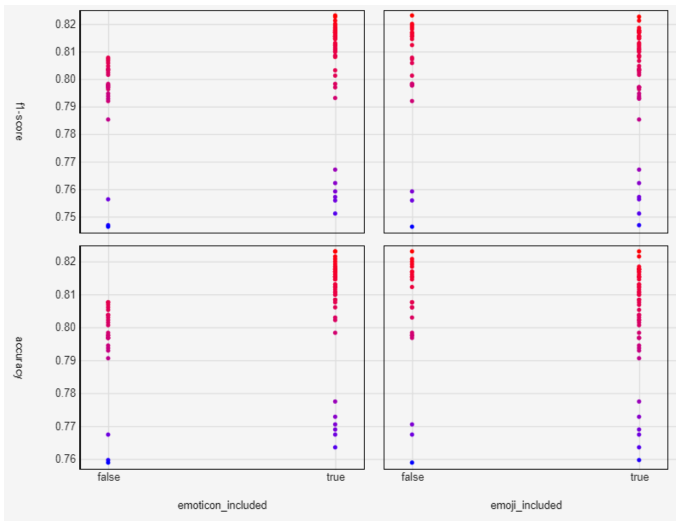


Figure 4.4: MNB - Emoticons and Emojis Influence

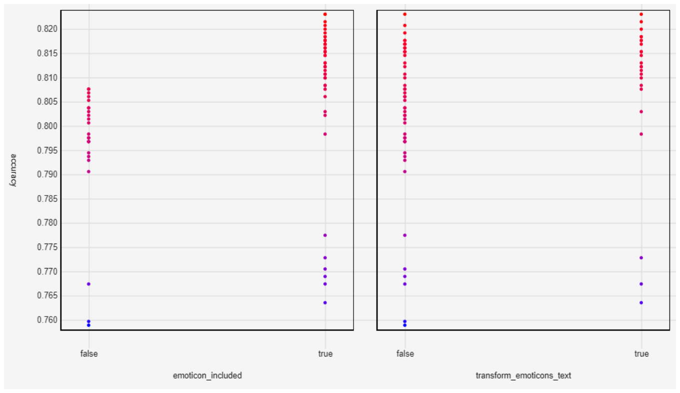


Figure 4.5: MNB - Emoticon Text Representation

### 4.1.3 K-Nearest Neighbors

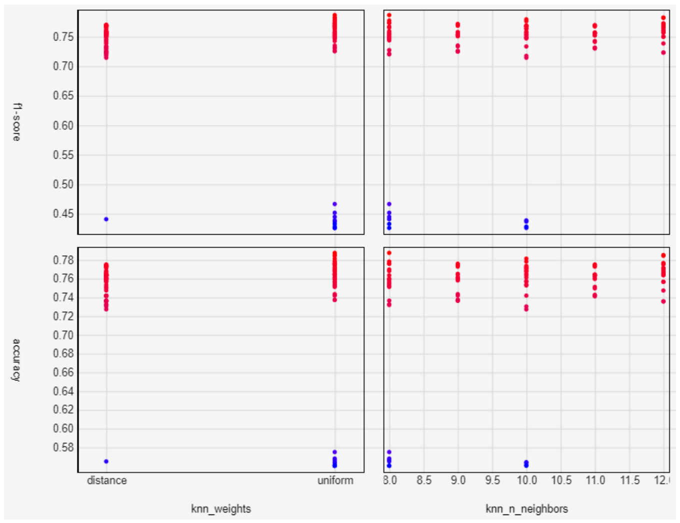


Figure 4.6: KNN - Hyperparameters Influence

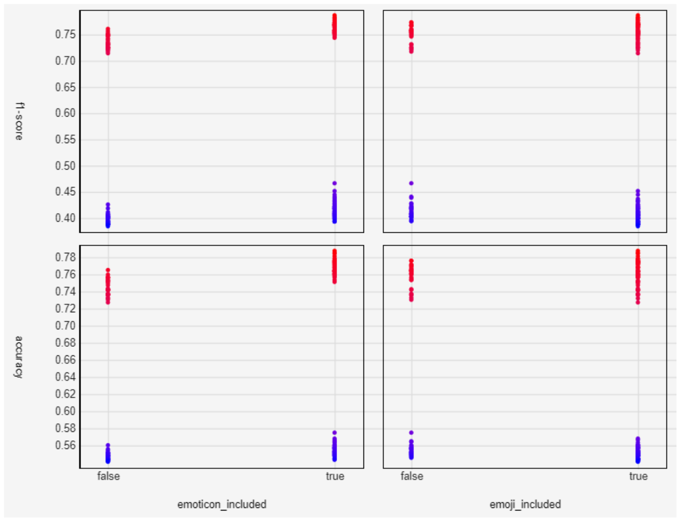
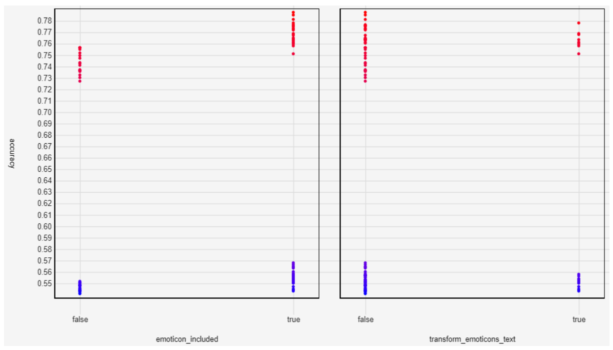
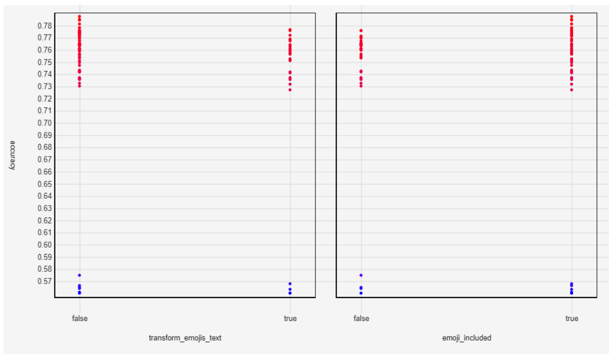


Figure 4.7: KNN - Emoticons and Emojis Influence



(a) KNN - Emoticon Text Transformation



(b) KNN - Emoji Text Transformation

### 4.1.4 Decision Tree

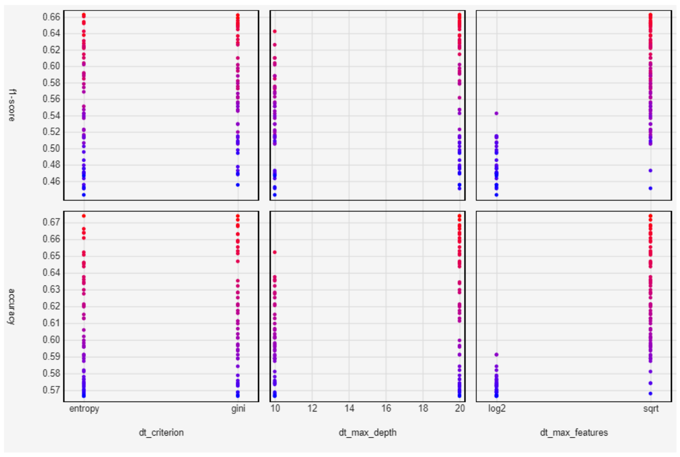


Figure 4.9: DT - Hyperparameters Influence

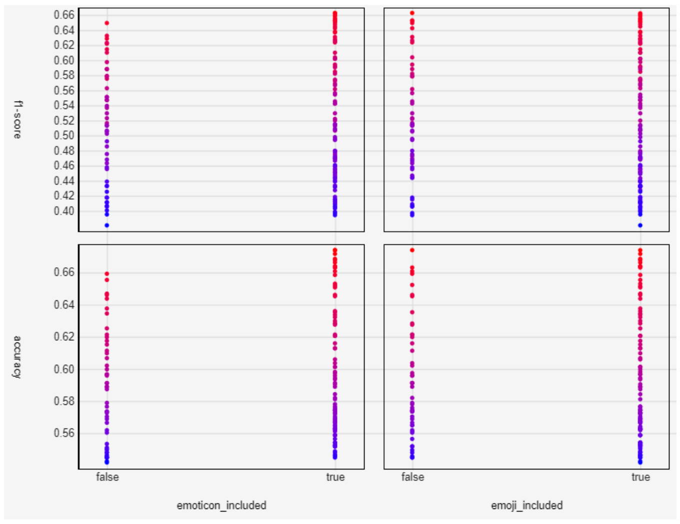


Figure 4.10: DT - Emoticons and Emojis Influence

### 4.1.5 Random Forest

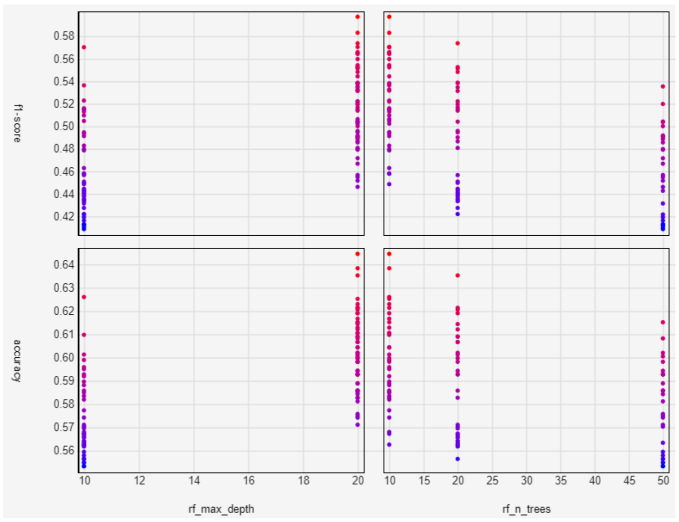


Figure 4.11: RF - Hyperparameters Influence

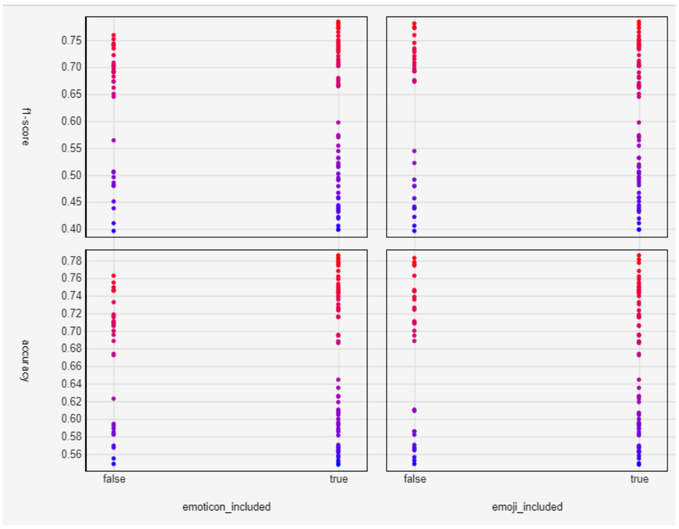
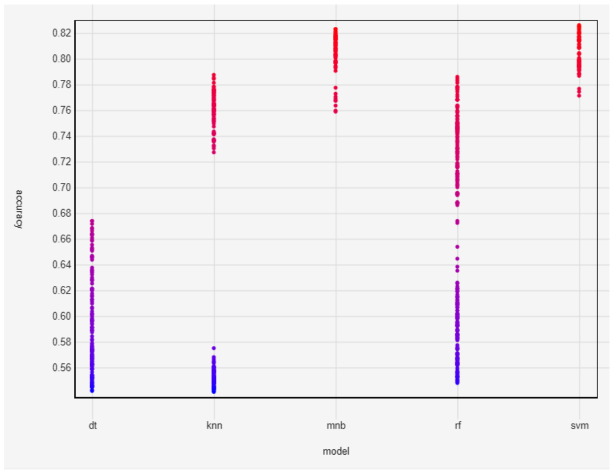
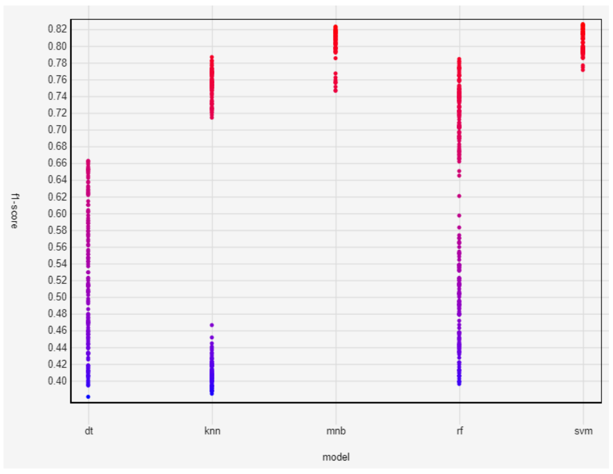


Figure 4.12: RF - Emoticons and Emojis Influence

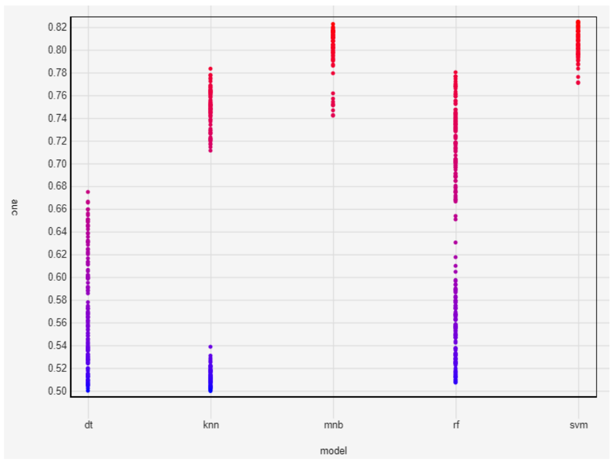
## 4.2 Metrics per model



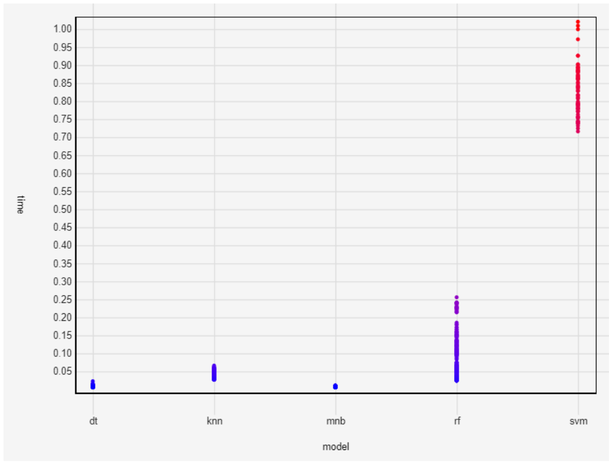
(a) Accuracy Overview per Model



(b) F1-Score Overview per Model



(a) AUC Overview per Model



(b) Time Overview per Model

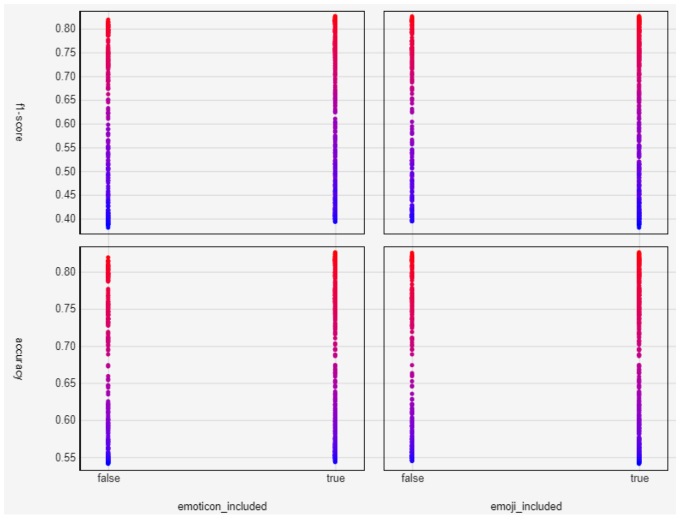


Figure 4.15: Overview - Emoticons and Emojis Influence