



جامعة تونس المنار

Université de Tunis El Manar



المدرسة الوطنية للمهندسين بتونس

école nationale d'ingénieurs de Tunis

**Information and Communication Technologies Department**

## **Sentiment Analysis**

### **Textual Data Sentiment Analysis**

**Realized by :**

**Amina Laakom and Firas Kachroudi**

**Class : 3ATEL2-Dasec**

**Supervised by :**

**Sabrine Krichen and Fares EL Kahla**

**Academic year 2022/2023**

# Table des matières

<b>Table des figures</b>	<b>iii</b>
0.1 Introduction . . . . .	iv
0.2 Data Preprocessing . . . . .	1
0.2.1 Loading Data . . . . .	1
0.2.2 Data Distribution . . . . .	2
0.2.3 Check missing data . . . . .	2
0.3 Text Preprocessing . . . . .	3
0.4 Train and test split . . . . .	4
0.5 Tokenization . . . . .	4
0.6 Sequences, Trunking and padding . . . . .	5
0.7 Model architecture . . . . .	5
0.8 Model Evaluation . . . . .	5
0.9 Conclusion . . . . .	6

# Table des figures

1	Raw data . . . . .	1
2	Final Dataset . . . . .	1
3	Data distribution . . . . .	2
4	Checking missing data . . . . .	2
5	Data after text preprocessing . . . . .	3
6	Word cloud visualization for positive words . . . . .	3
7	Word cloud visualization for negative words . . . . .	4
8	Train and test split . . . . .	4
9	Accuracy for train and test . . . . .	6
10	Confusion Matrix . . . . .	6
11	Classification report . . . . .	7

## 0.1 Introduction

Natural Language Processing or NLP is a branch of Artificial Intelligence which deal with bridging the machines understanding humans in their Natural Language. Natural Language can be in form of text or sound, which are used for humans to communicate each other. NLP can enable humans to communicate to machines in a natural way.

Text Classification is a process involved in Sentiment Analysis. It is classification of peoples opinion or expressions into different sentiments. Sentiments include Positive, Neutral, and Negative, Review Ratings and Happy, Sad. Sentiment Analysis can be done on different consumer centered industries to analyse people's opinion on a particular product or subject.

Sentiment Classification is a perfect problem in NLP for getting started in it. You can really learn a lot of concepts and techniques to master through doing project.

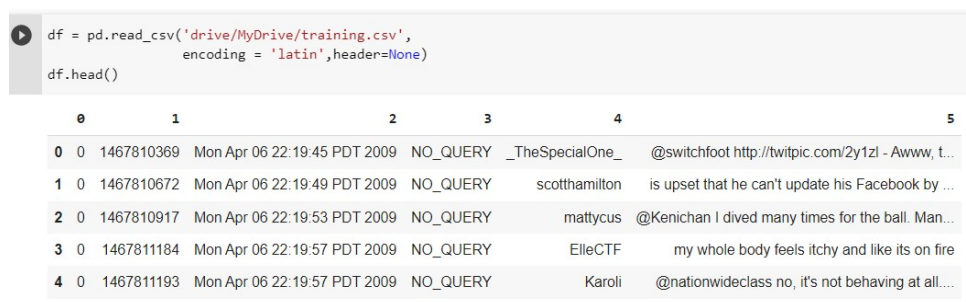
I will go through all the key and fundament concepts of NLP and Sequence Models in this work.

## 0.2 Data Preprocessing

In this project, I am using Sentiment-140 from Kaggle. It contains a labels data of 1.6 Million Tweets and I find it a good amount of data to train our model.

### 0.2.1 Loading Data

First of all, we load our data and convert it into DataFrame using pandas.



```
df = pd.read_csv('drive/MyDrive/training.csv',
                 encoding = 'latin', header=None)
df.head()
```

	0	1	2	3	4	5
0	0	1467810369	Mon Apr 06 22:19:45 PDT 2009	NO_QUERY	_TheSpecialOne_	@switchfoot http://twitpic.com/2y1zl - Awww, t...
1	0	1467810672	Mon Apr 06 22:19:49 PDT 2009	NO_QUERY	scotthamilton	is upset that he can't update his Facebook by ...
2	0	1467810917	Mon Apr 06 22:19:53 PDT 2009	NO_QUERY	mattycus	@Kenichan I dived many times for the ball. Man...
3	0	1467811184	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	ElleCTF	my whole body feels itchy and like its on fire
4	0	1467811193	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	Karoli	@nationwideclass no, it's not behaving at all....

FIGURE 1 – Raw data

As we can see, the columns are without any proper names, we rename them for our reference. Also we are going to train only on text to classify its sentiment, so we can remove all useless columns.

```
lab_to_sentiment = {0:"Negative", 4:"Positive"}

def label_decoder(label):
    return lab_to_sentiment[label]

df.sentiment = df.sentiment.apply(lambda x: label_decoder(x))
df.head()
```

	sentiment	text
0	Negative	@switchfoot http://twitpic.com/2y1zl - Awww, t...
1	Negative	is upset that he can't update his Facebook by ...
2	Negative	@Kenichan I dived many times for the ball. Man...
3	Negative	my whole body feels itchy and like its on fire
4	Negative	@nationwideclass no, it's not behaving at all....

FIGURE 2 – Final Dataset

Here are decoding the labels. We map 0 to Negative and 4 to Positive as directed by the dataset description.

### 0.2.2 Data Distribution

Now that we decoded we shall now analyse the dataset by its distribution. As we can see it's well balanced.

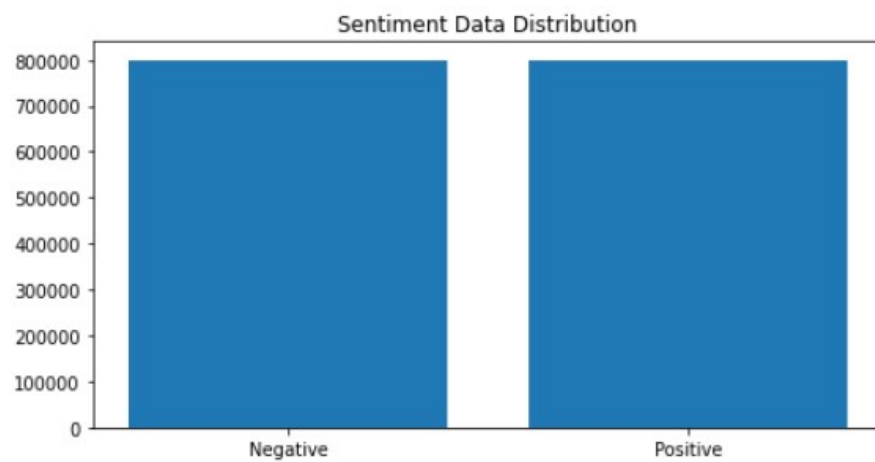


FIGURE 3 – Data distribution

### 0.2.3 Check missing data

```
[ ] #check missing data
df.isna().sum()

sentiment    0
text         0
dtype: int64
```

FIGURE 4 – Checking missing data

We don't have any missing data.

## 0.3 Text Preprocessing

Tweet texts often consists of other user mentions, hyperlink texts, emoticons and punctuations. In order to use them for learning using a Language Model. We cannot permit those texts for training a model. So we have to clean the text data using various preprocessing and cleansing methods.

we remove URLs and mentions. After, we remove stopwords, stopwords are commonly used words in English which have no contextual meaning in an sentence. So therefore we remove them before classification. Finally, we remove punctuations and numbers.

	sentiment	text
0	Negative	thats bummer shoulda got david carr third day...
1	Negative	upset cant update facebook texting it might cr...
2	Negative	dived many times ball managed save rest go bo...
3	Negative	whole body feels itchy like fire
4	Negative	no behaving all im mad here cant see there

FIGURE 5 – Data after text preprocessing

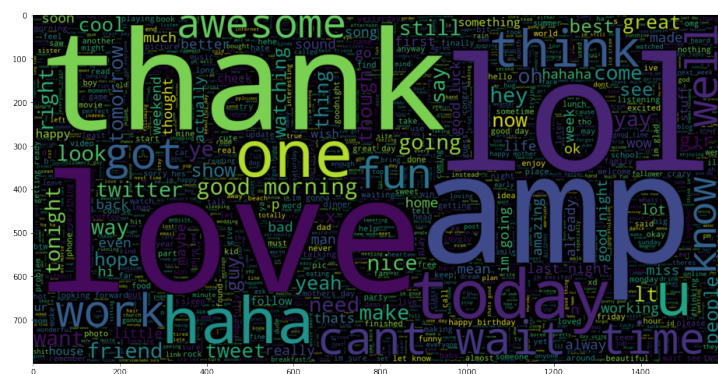


FIGURE 6 – Word cloud visualization for positive words



train/test split will shuffle the dataset and split it to gives training and testing dataset. It's important to shuffle our dataset before training.

We have 0.7 of data for train and 0.3 for test.

We have 0.7 of data for train and 0.3 for test.

FIGURE 8 – Train and test split

Given a character sequence and a defined document unit, tokenization is the task of chopping it up into pieces, called tokens , perhaps at the same time throwing away certain characters, such as punctuation. The process is called Tokenization.

- tokenizer : create tokens for every word in the data corpus and map them to a index using dictionary.
- word index : contains the index for each word.
- vocab size : represents the total number of word in the data corpus.



## 0.6 Sequences, Trunking and padding

Now we got a tokenizer object, which can be used to covert any word into a Key in dictionary (number). Since we are going to build a sequence model. We should feed in a sequence of numbers to it. And also we should ensure there is no variance in input shapes of sequences. It all should be of same lenght. But texts in tweets have different count of words in it. To avoid this, we seek a little help from pad sequence to do our job. It will make all the sequence in one constant length MAX SEQUENCE LENGTH.

## 0.7 Model architecture

The code defines a function called `create_model()` that creates a sequential neural network model using the TensorFlow library. The model takes three inputs : `vocabsize`, `embeddingdim`, and `maxlen`.

The model is composed of several layers :

1. Embedding layer which takes vocab size and embedding dim as input and is used to convert the input to dense embedding of fixed size.
2. A 1D convolutional layer with a filter size and kernel size specified by the `filters` and `kernel_size` variables (which are not defined in this code snippet) and uses the ReLU activation function.
3. Two bidirectional LSTM layers with dimensions specified by the `lstm1_dim` and `lstm2_dim` variables (which are also not defined in this code snippet)
4. Several dropout layers with a dropout rate of 0.3 to prevent overfitting. Several dense layers with different number of neurons and `relu` activation function.
5. The final dense layer with sigmoid activation function that outputs a probability between 0 and 1.

## 0.8 Model Evaluation

We train the model jut for 4 Epochs and we obtain around 0.8 accuracy.

```

Epoch 1/4
35000/35000 [=====] - 2752s 78ms/step - loss: 0.4705 - accuracy: 0.7763 - val_loss: 0.4534 - val_accuracy: 0.7897
Epoch 2/4
35000/35000 [=====] - 2751s 79ms/step - loss: 0.3733 - accuracy: 0.8340 - val_loss: 0.4680 - val_accuracy: 0.7796
Epoch 3/4
35000/35000 [=====] - 2744s 78ms/step - loss: 0.3417 - accuracy: 0.8485 - val_loss: 0.4975 - val_accuracy: 0.7721
Epoch 4/4
35000/35000 [=====] - 2742s 78ms/step - loss: 0.3202 - accuracy: 0.8583 - val_loss: 0.5002 - val_accuracy: 0.7769

```

FIGURE 9 – Accuracy for train and test

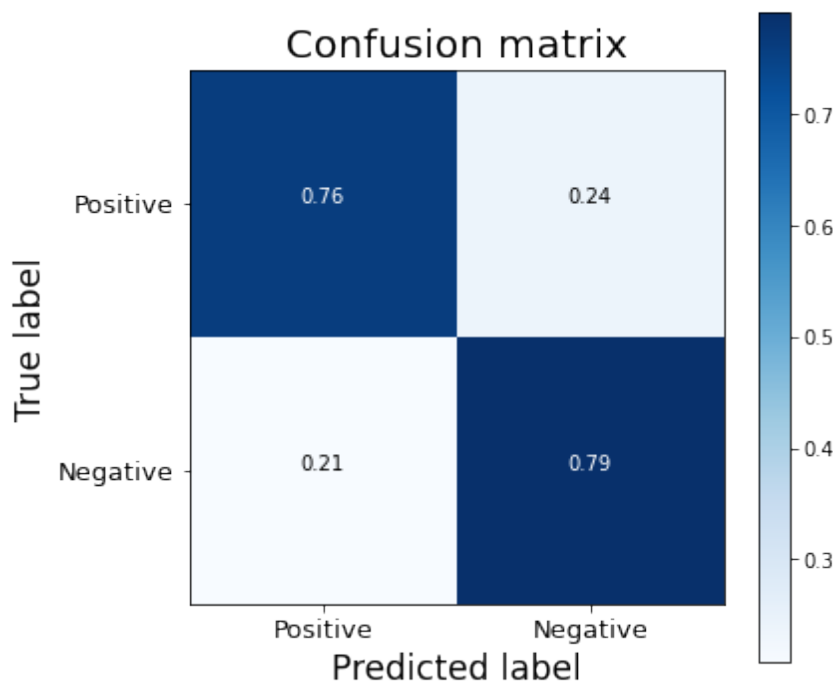


FIGURE 10 – Confusion Matrix

## 0.9 Conclusion

It's a pretty good model we trained here in terms of NLP. Around 80 per cent accuracy is good enough. Also, you may go on and explore the dataset, some tweets might have other languages than English. So our Tokenizing wont have effect on them. But on practical scenario, this model is good for handling most tasks for Sentiment Analysis.

```
print(classification_report(list(df_test.sentiment), y_pred_1d))
```

	precision	recall	f1-score	support
Negative	0.78	0.76	0.77	239361
Positive	0.77	0.79	0.78	240639
accuracy			0.78	480000
macro avg	0.78	0.78	0.78	480000
weighted avg	0.78	0.78	0.78	480000

FIGURE 11 – Classification report