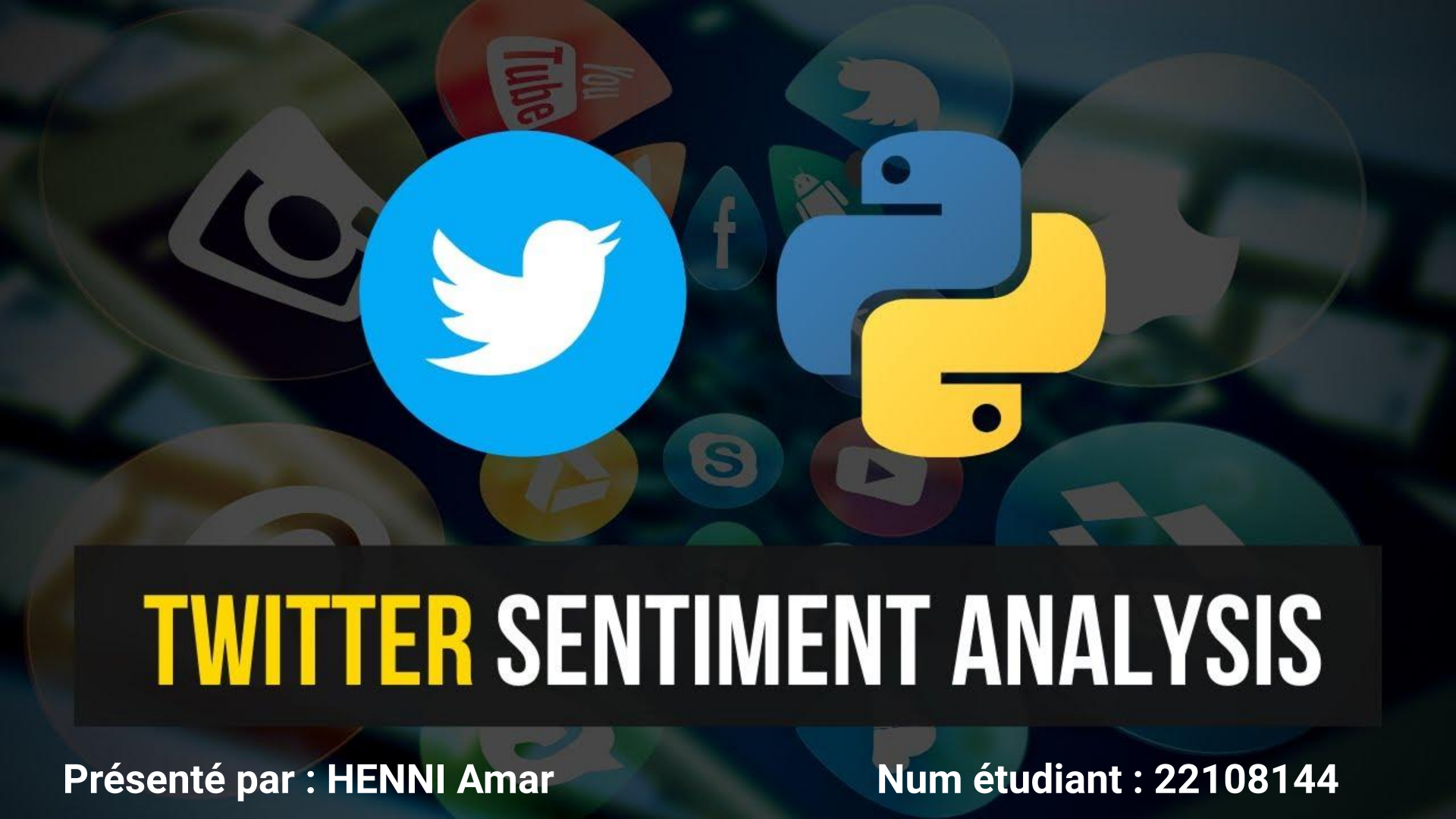


TWITTER SENTIMENT ANALYSIS

Présenté par : HENNI Amar

Num étudiant : 22108144

Pourquoi l’analyse de sentiments ?

Détails

Extrait d’une compétition

Nom : Natural Language Processing with Disaster Tweets

But : Prédire quels Tweets concernent de vraies catastrophes et lesquels ne le sont pas

Source : https://www.kaggle.com/c/nlp-getting-started/data?select=train.csv

73	ablaze	Sheffield Township, Ohio	Deputies: Man shot before Brighton home set ablaze http://t.co/gWNRhMSO8k	1
74	ablaze	India	Man wife get six years jail for setting ablaze niece http://t.co/eV1ahOUCZA	1
76	ablaze	Barbados	SANTA CRUZ ÛÓ Head of the St Elizabeth Police Superintendent Lanford Salmon has r ... - http://t.co...	0
77	ablaze	Anaheim	Police: Arsonist Deliberately Set Black Church In North CarolinaáÊAblaze http://t.co/pcXarbH9An	1

```
print(disaster_data.shape) #nombre de ligne et colonne de mon dataset
```

```
(7613, 5)
```

id : un identifiant unique pour chaque tweets

text : le text du tweet

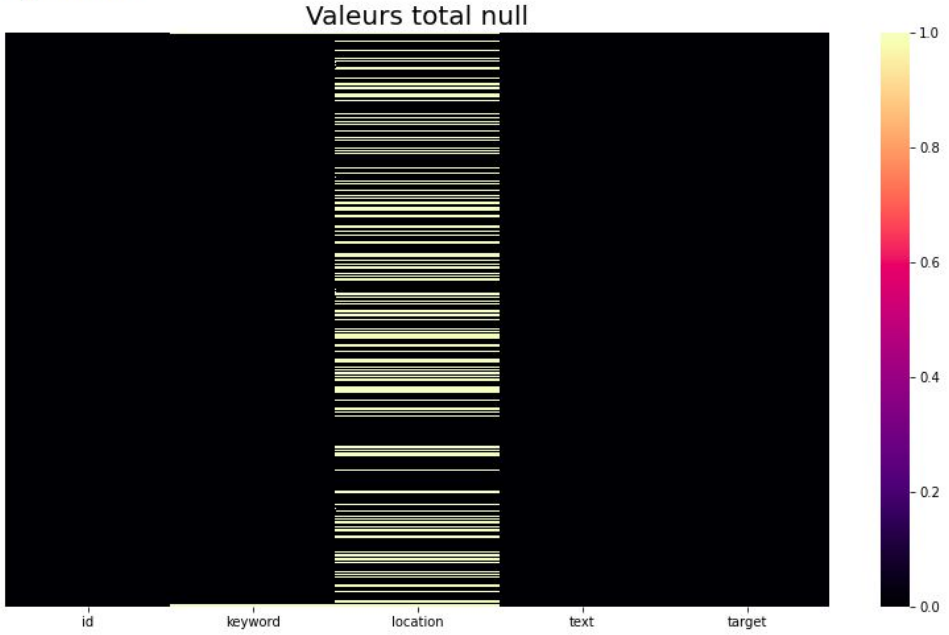
location : la localisation d'où le tweet a été émis (peut être vide)

keyword - Un mot particulier des tweet (peut être vide)

target -indique si un tweet parle d'un véritable désastre (1) ou non (0)

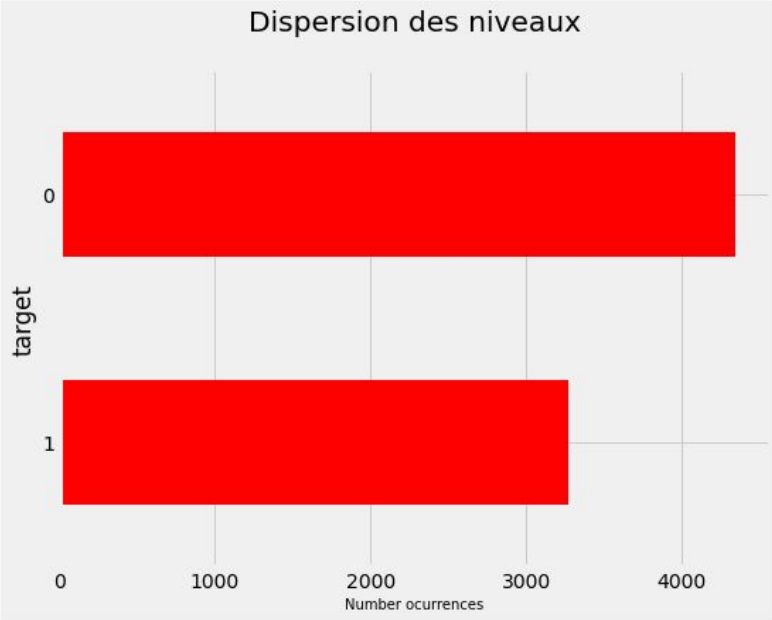
```
import seaborn as sns
cprint('Valeurs total null de notre dataset :','green')
print(disaster_data.isnull().sum()) # showing null values of train data
```

```
Valeurs total null de notre dataset :
id             0
keyword       61
location    2533
text           0
target         0
dtype: int64
```




```
#create a dataframe with random
dataframe_tweets = disaster_data.sample(7613, random_state=1).copy()
#Nombre de tweets Réels vs tweets Non_réel
print(dataframe_tweets['target'].value_counts())
```

```
0    4342
1    3271
Name: target, dtype: int64
```



Méthode choisie

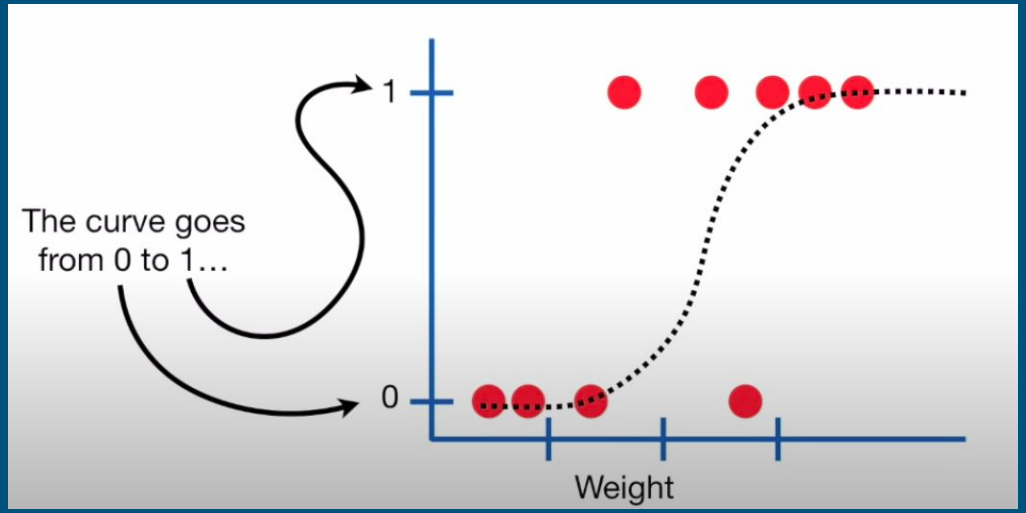
régression logistique

Régression logistique

Mesure de la relation entre une variable dépendante (dichotomique) et une ou plusieurs variables indépendantes

Utile dans la prédiction de la présence ou absence d'un comportement

succès ou échec, amélioration ou non …



Etapes :

1- Analyse de nos données

2- Prétraiter le texte pour le rendre propre et lisible.

3- Représenter les tweets sous forme d’un vecteur numériques (Mappage de dictionnaire)

4- S'appuyer sur une régression logistique pour prédire les sentiments des tweets

Objectif :

- Prédire de manière la plus précise possible en fonction de nos données
- Donc … il faut minimiser notre taux d’erreur

Tweet : Je suis heureux parce que j’apprends le NLP

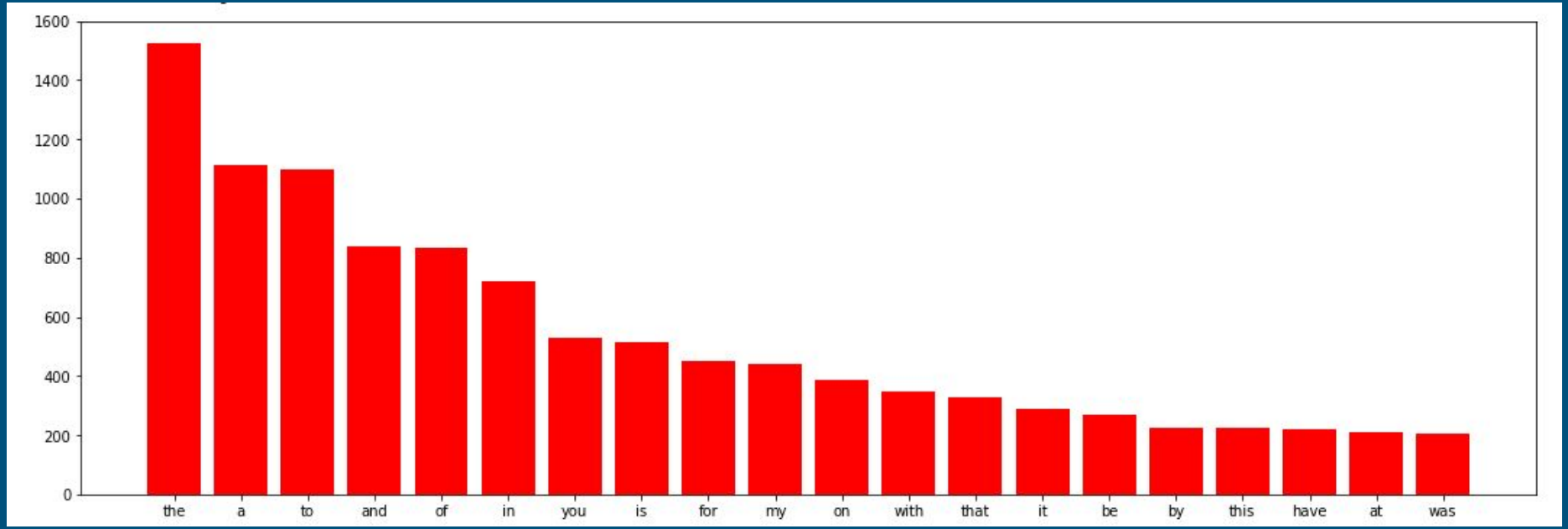
Real disaster

Not real disaster

Analyse des données

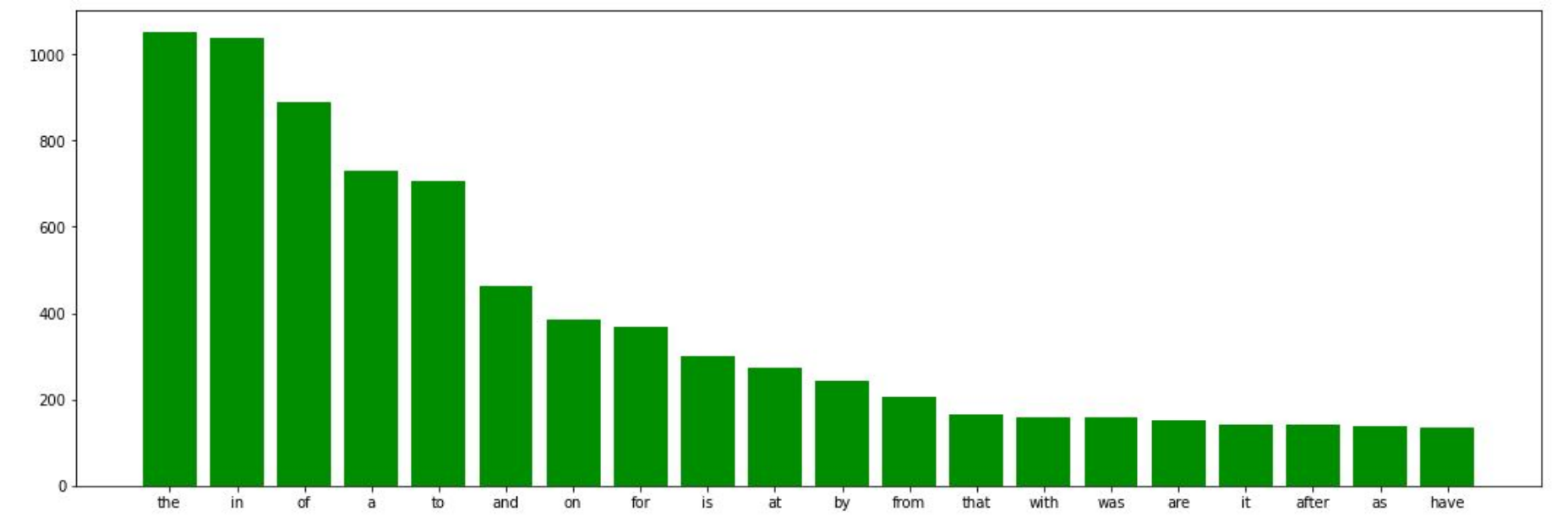
Analyse des données

Top 20 des mots vides (stop words) présents dans les tweets non réel (class 0)



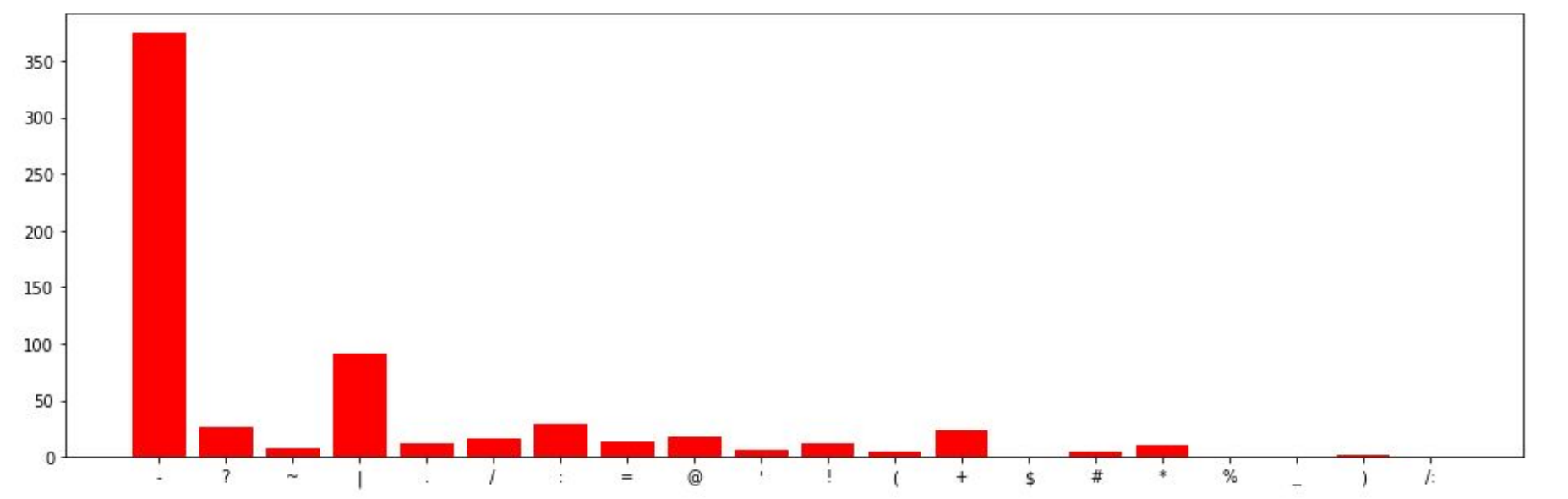
Analyse des données

Top 20 des mots vides (stop words) présents dans les tweets réels (class 1)



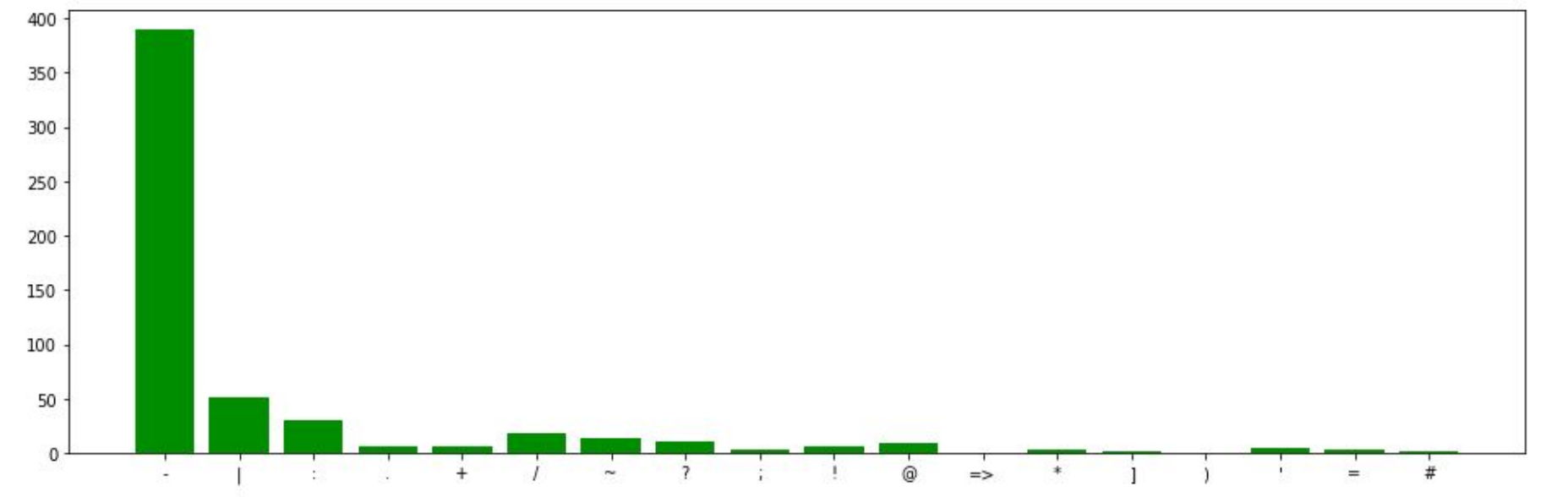
Analyse des données

Analyse de la ponctuation présents dans les tweets non réels (class 0)



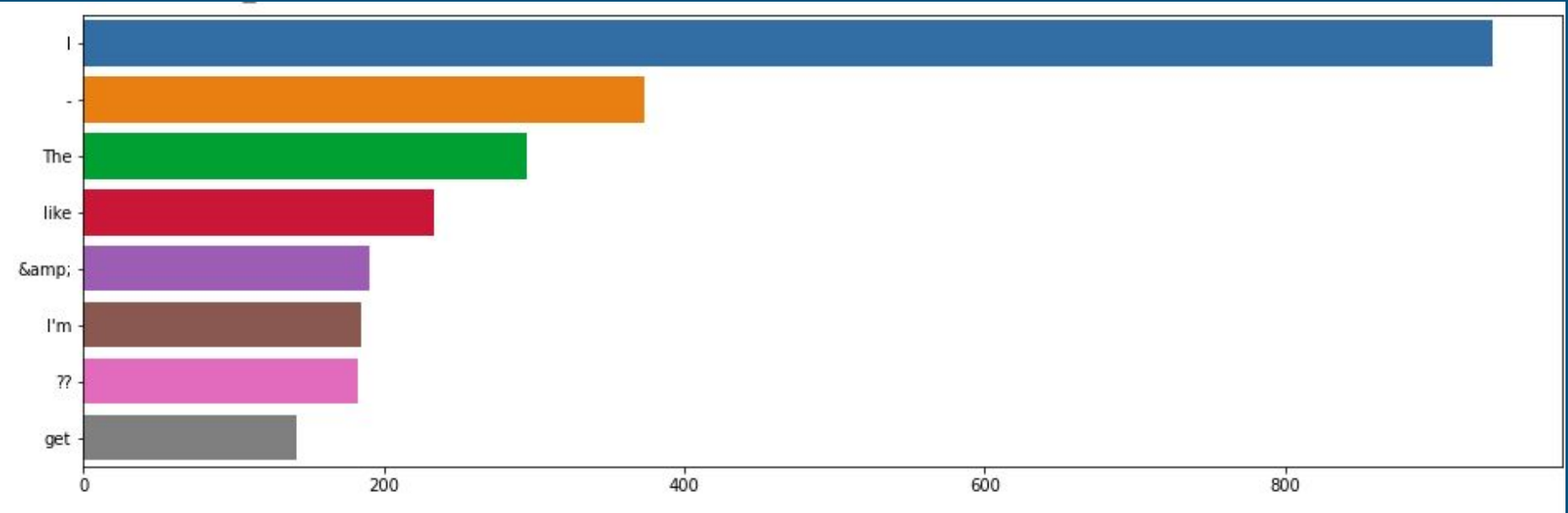
Analyse des données

Analyse de la ponctuation présents dans les tweets réels (class 1)



Analyse des données

Les mots les plus communs dans tous les tweets



https://dev.w3.org/html5/html-author/charref

Pré traitement des données

Preprocessing

Vérification régulière de la sortie du préprocessing

```
#affichage des 10 premier tweet positifs format text (pour afficher l'integralité des messages contenu dans chaque tweets réels)
for c in dataframe_tweets[dataframe_tweets["target"] == 1]["text"].head(10):
    print(c)
```

```
Goulburn man Henry Van Bilsen missing: Emergency services are searching for a Goulburn man who disappeared from hisÛ_ http://t.co/z99pKJzTRp
Crawling in my skin
These wounds they will not hea
Hollywood Movie About Trapped Miners Released in Chile: 'The 33' Hollywood movie about trapped miners starring... http://t.co/tyyfG4qQvM
The Catastrophic Effects of Hiroshima and Nagasaki Atomic Bombings Still Being Felt Today http://t.co/WC8AqXeDF7
Witness video shows car explode behind burning buildings on 2nd St this afternoon #Manchester  http://t.co/cgmJlSEYLo via @MikeCroninWMUR
Sweetpea's are running riot at the allotment - and brightening up a rainy day http://t.co/6NdBFOPK5m
After a violent afternoon storm more severe weather heads for Chicago tonight. Details ----&gt; http://t.co/6Peeip4y7W
I just added 'Sandy First Responders Lost Their Homes' to VIP Home Page Group on @Vimeo: https://t.co/lKXi6UXjaQ
#NJTurnpike å_ #NJTurnpike Reopens Hours After Truck Fire In? http://t.co/oABJZtbVyZ http://t.co/GPBXRrDc07
Wild fires in California... Must be Global Warming. Can't just be extreme heat combined with dry foliage ignited by some douchebag hiker.
```

Preprocessing

```python
import re

def preprocessing_dataset(tweet):


    #Suppression des URLs
    tweet = re.sub('https?://\S+|www\.\S+', '', tweet)

    #Supression des crochets
    tweet = re.sub('\[.*?\]', '', tweet)

    #Suppression des tags HTML
    tweet = re.sub('<.*?>+', '', tweet)

    #Suppression des \n
    tweet = re.sub('\n', '', tweet)

    #Supression des nombre
    tweet = re.sub('\w*\d\w*', '', tweet)
```

Les caractères spéciaux

Les contractions

```python
tweet = re.sub(r"he's", "he is", tweet)

    tweet = re.sub(r"there's", "there is", tweet)
    tweet = re.sub(r"We're", "We are", tweet)
    tweet = re.sub(r"That's", "That is", tweet)
```

Suppression d'emoji
Supression de la ponctuations
Suppression de mots vides

```python
#Mettre en minuscule tous les mots
```

Preprocessing

Correction d'orthographe

```
from spellchecker import SpellChecker
```

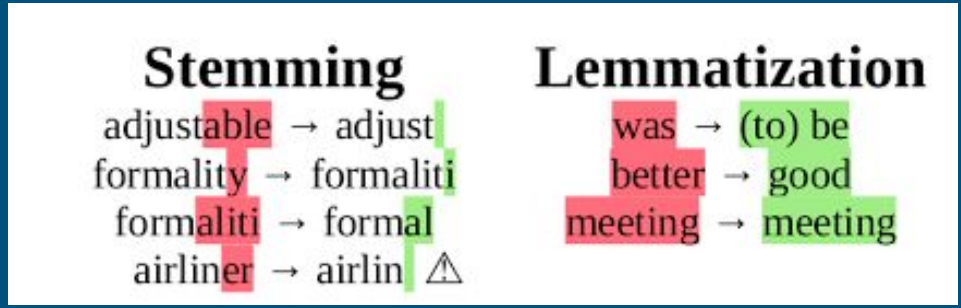
La lemmatisation

```
from nltk.stem import WordNetLemmatizer
import nltk
nltk.download('wordnet')
```

considère le contexte d’un mot et convertit le mot en saforme de base significative.

Stemming

transforme le mot à sa forme radical



Encodage des données

Encoding

Représentation des tweets sous forme d'un vecteur numériques

TF-IDF : une méthode qui nous permet de calculer l'importance de chaque mot dans un document, les mots les plus discriminants

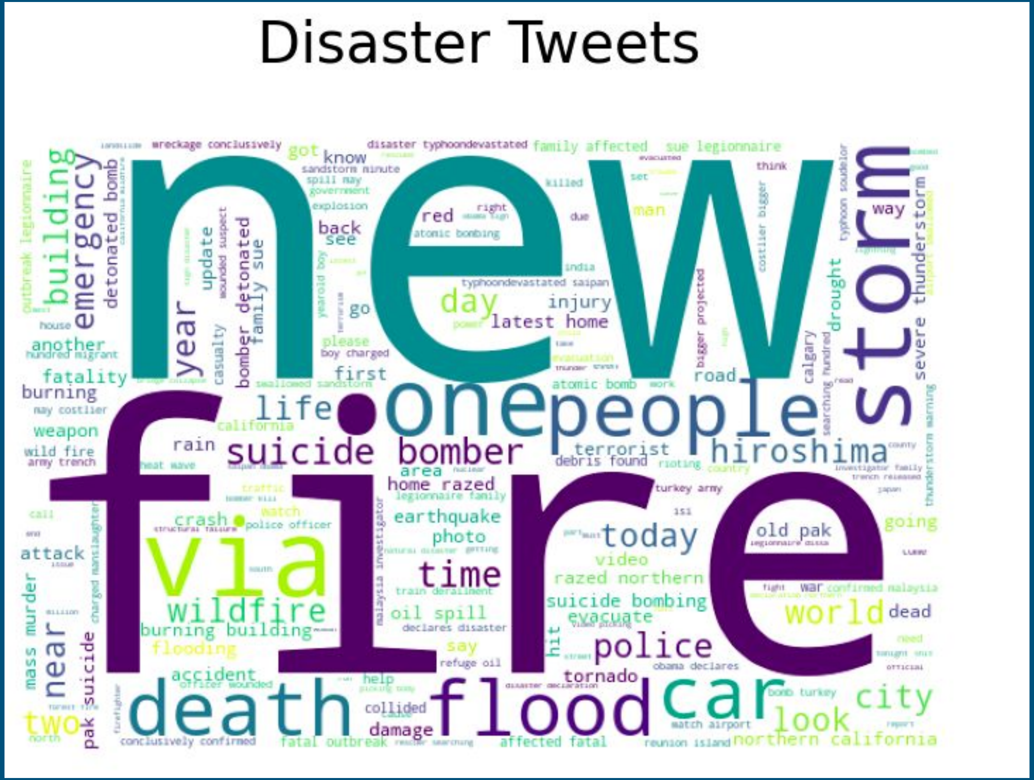
Tf = nombre de mot qui apparaît / nombre totale des mots

Idf = nombre des documents / nombre de document ou les mots apparaît

$$TFIDF_{t,d,D} = TF_{t,d} \times IDF_{t,D}$$

- $TFIDF_{t,d,D}$: Importance d'un terme t dans un document d
- $TF_{t,d}$: Fréquence d'un terme t dans un document d
- $IDF_{t,D}$: Importance du terme t dans l'ensemble des documents D

Juste pour le fun (WordCloud)



Test et Validation

Train / Validation

La métrique que j'ai tenu en compte pour évaluer le modèle c'est l'accuracy

Qui correspond à la proportion de prédictions correctes sur les prédictions totales.

et aussi l'erreur absolu moyenne Mean Absolute Error

On a pas bcp de point de données et cela n'a pas de sens de diviser nos données en train, validation et test. Sklearn a un objet cross_val_score qui nous permet de voir à quel point notre modèle se généralise.

```
from sklearn.feature_extraction.text import TfidfVectorizer,CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer

print(f'Début de l\'encodafe des données ... sous format TF-IDF: ')
tfidf = TfidfVectorizer(sublinear_tf=True, min_df=5,
                        ngram_range=(1, 2),
                        stop_words='english')

# Transformation en vecteur
features = tfidf.fit_transform(dataframe_tweets.text).toarray()
print('Encodage terminé')

X = features # Collection of documents
y = dataframe_tweets['target']
```

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import accuracy_score
from sklearn.model_selection import cross_val_score


# Split the data
print(f'Début du split du dataset ...')
X_train,X_test,Y_train,Y_test = train_test_split(X, y, test_size=0.30, random_state=1)
print(f'Split terminé : ')

print(f'\t Ensemble d\'entrainement  = 70% : ')
print(f'\t Ensemble de test  = 30% : ')

#On définie le modèle LR
model = LogisticRegression(random_state=1)
#model = RandomForestClassifier(n_estimators=100, max_depth=5, random_state=0)

#entrainnement de notre modèle
model.fit(X_train, Y_train)

#valeur à prédir
val_predictions = model.predict(X_test)
```

```
print('\nTrain/Test split results:')
print(model.__class__.__name__+" accuracy is %2.3f" % accuracy_score(Y_test, val_predictions))

#Erreur de prédiction pour les tweets (moyenne de toutes les erreurs)
print('\nMean Absolute Error (MAE) is ',mean_absolute_error(Y_test, val_predictions))

scores = cross_val_score(model, X_train, Y_train, cv=10)
print('\nCross-Validation Accuracy Scores', scores)

#Afficher la valeur min - moyenne - max de la cross validation
scores = pd.Series(scores)
scores.min(), scores.mean(), scores.max()

#nos prédictions sont erronées d'environ 0.20359019264448336
```

Résultats

```
Début du split du dataset ...
Split terminé :
         Ensemble d'entrainement  = 70% :
         Ensemble de test  = 30% :

Train/Test split results:
LogisticRegression accuracy is 0.796

Mean Absolute Error (MAE) is  0.20359019264448336

Cross-Validation Accuracy Scores [0.77673546 0.79362101 0.7804878  0.79362101 0.81050657 0.82363977
 0.81613508 0.7804878  0.81425891 0.78571429]
(0.776735459662289, 0.7975207719110158, 0.8236397748592871)
```

Discussion

J’aurai voulu utilisé :

Mon modèle sur un dataset beaucoup plus grands

D’autre représentation de mes données (autre encodage) tel que :

- Word2Vec

Cbow : continuous bag of words, prédit le mot étant donné son contexte

Skip gram : Le modèle est nourri par le mot cible, et prédit les mots du contexte

- BERT (Bidirectional Encoder Representations from Transformers)

J’aurai voulu voir les réseaux de neurones plus en profondeurs dans le NLP