Détecter les BadBuzz grâce au Deep Learning

P7 - Sylvia Bankowska - Septembre 2022

Sommaire

Enjeux et objectifs

Enjeux

- Anticiper des tweets potentiellement **nuisibles**
- Prédire le **sentiment** associé à un tweet

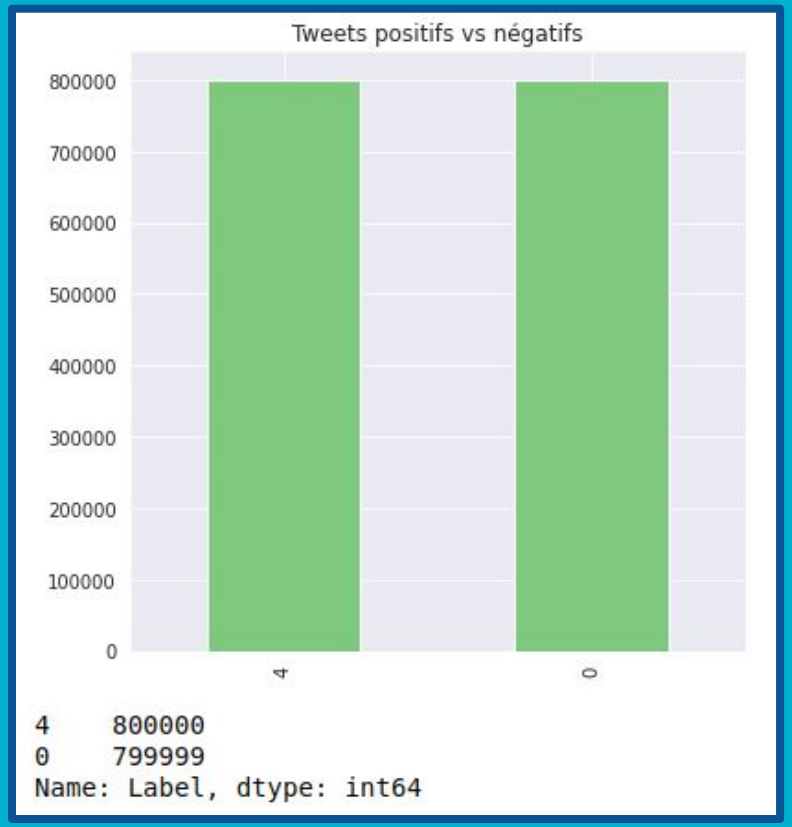
Objectifs

- **Comparer** plusieurs modèles de réseaux de neurones
- **Créer le prototype** d'un produit IA

Données

Analyse du dataset

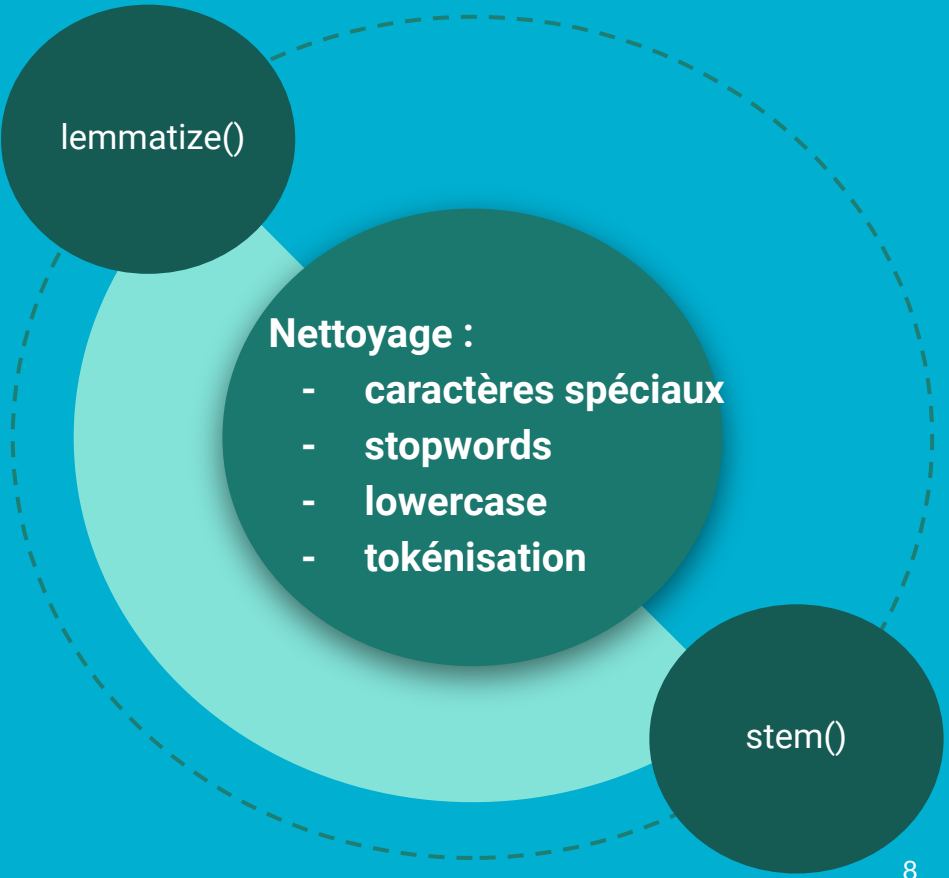
- **Jeu de données :**
  - 1 600 000 tweets
  - propre
- **Echantillon :**
  - contenu d'un tweet + label
  - 8000 pos + 8000 neg
- **Target :**
  - variable binaire
  - classes très équilibrées
- **Métrique :**
  - score F1 + AUROC



Prétraitement

Transformer le texte en chiffres

Stemming	Lemmatisation
Forme tronquée : ➢ suppression des suffixes ➢ suppression des flexions	Forme canonique : ➢ analyse morpho



Transformation du texte en chiffres

Vectorisation

fit_on_texts()

Sequencing

texts_to_sequences()

Padding

pad_sequences()

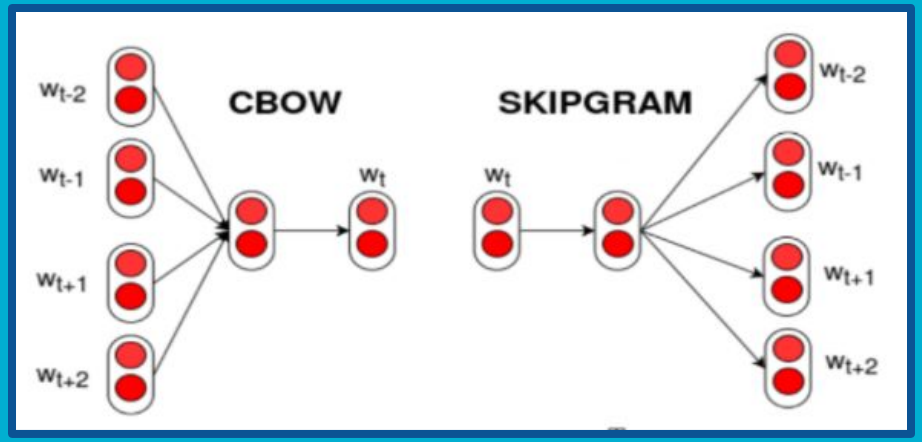
```
{'good': 1,
 'go': 2,
 'like': 3,
 'work': 4,
 'love': 5,
 'quot': 6,
 'today': 7,
 'time': 8,
 'well': 9,
 'make': 10,
 'back': 11,
 'miss': 12,
 'feel': 13,
 'want': 14,
 'know': 15,
 'think': 16,
 'amp': 17,
 'really': 18,
 'need': 19,
```

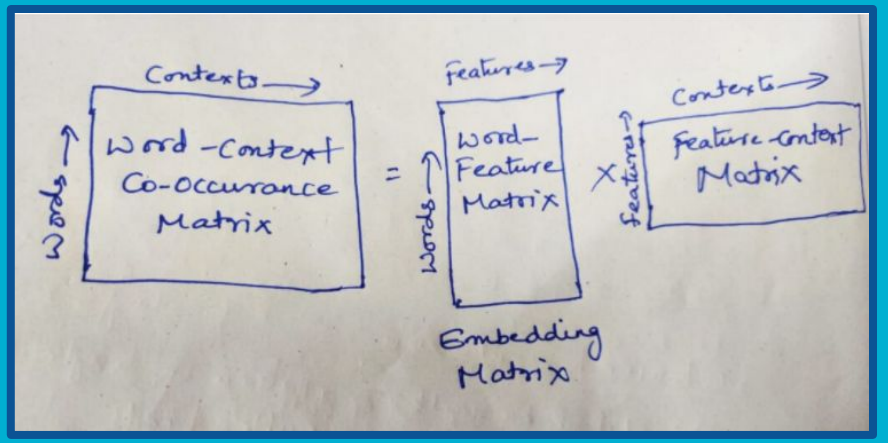
```
[186, 251, 42],
[25, 21, 24, 4531],
[641, 604, 642, 518],
[1892, 3024, 4532, 160, 4533],
[7, 3, 965, 3025, 168, 49, 1608, 466],
[95, 60, 192, 229],
[1, 3026, 2310, 1270, 109, 123, 5, 4534, 494],
[727, 145, 1893, 1429],
[1609],
[4535, 77, 1894, 337, 4536, 316, 4537],
[158, 1610, 2, 2311, 18, 210, 12, 128],
[72],
[78, 14, 78, 14, 4538],
[877, 100, 2312, 4539, 1153],
[341, 390, 19, 500],
[573, 5],
```

```
[[ 412,  324,    2, ...,    0,    0,    0],
 [  19, 1251, 4456, ...,    0,    0,    0],
 [1252,  251,   78, ...,    0,    0,    0],
 ...,
 [2992, 2992, 2992, ...,    0,    0,    0],
 [  16,   23,  108, ...,    0,    0,    0],
 [  36,  863,    0, ...,    0,    0,    0]],
```

Word Embeddings

Word2Vec vs GloVe





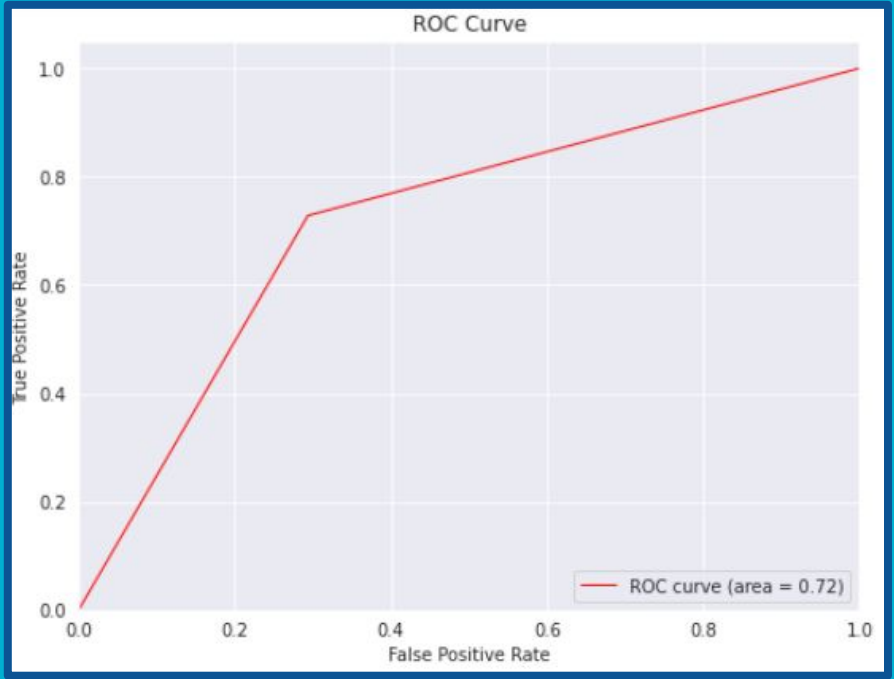
- réseau de neurones feed forward
- capture le contexte similaire

- calcul basé sur les cooccurrences de mots sur l'ensemble du corpus
- capture la probabilité que 2 mots apparaissent ensemble

Source

Modèle sur mesure simple

Régression Logistique



La **FONCTION SIGMOIDE** permet de mesurer si une entrée a dépassé le seuil de classification.

F1 score: 0.7197274698048932

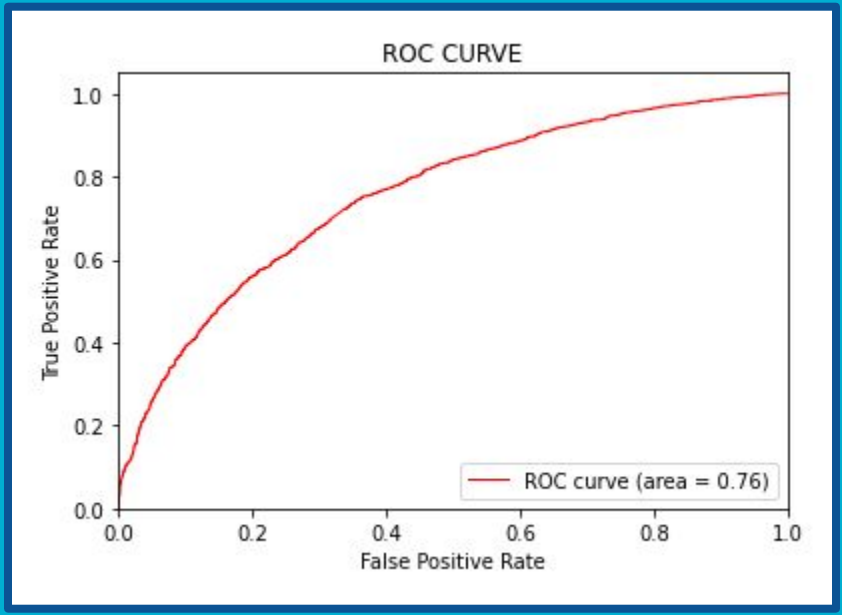
Classification Report

	precision	recall	f1-score	support
0	0.72	0.71	0.71	1604
1	0.71	0.73	0.72	1596
accuracy			0.72	3200
macro avg	0.72	0.72	0.72	3200
weighted avg	0.72	0.72	0.72	3200

Modèles sur mesure avancés

Keras + Embedding / RNN & LSTM / CNN / BERT

Couche Embedding de Keras

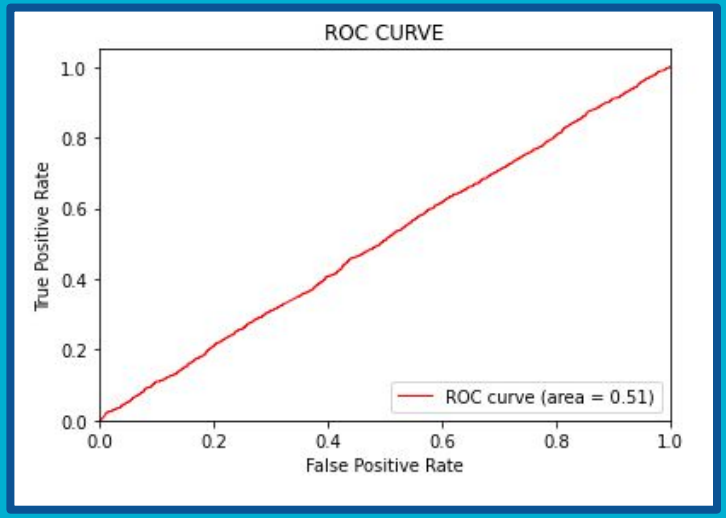


	precision	recall	f1-score	support
0	0.52	0.45	0.48	2452
1	0.50	0.57	0.53	2348
accuracy			0.51	4800
macro avg	0.51	0.51	0.50	4800
weighted avg	0.51	0.51	0.50	4800

- optimisation d'hyperparamètres
- tuner Hyperband de Keras Tuner
- tuning de l'hypermodèle :
  - best learning rate
  - best epochs
  - nb de neurones couche Dense

RNN & LSTM

- Entraînement :
  - avec Word2Vec et Glove
  - sur un corpus lemmatisé et stemmé
- Résultats : médiocres
- Le meilleur score F1 => corpus prétraité avec le stemmeur :
  - Word2Vec : 0.54
  - GloVe : 0.51

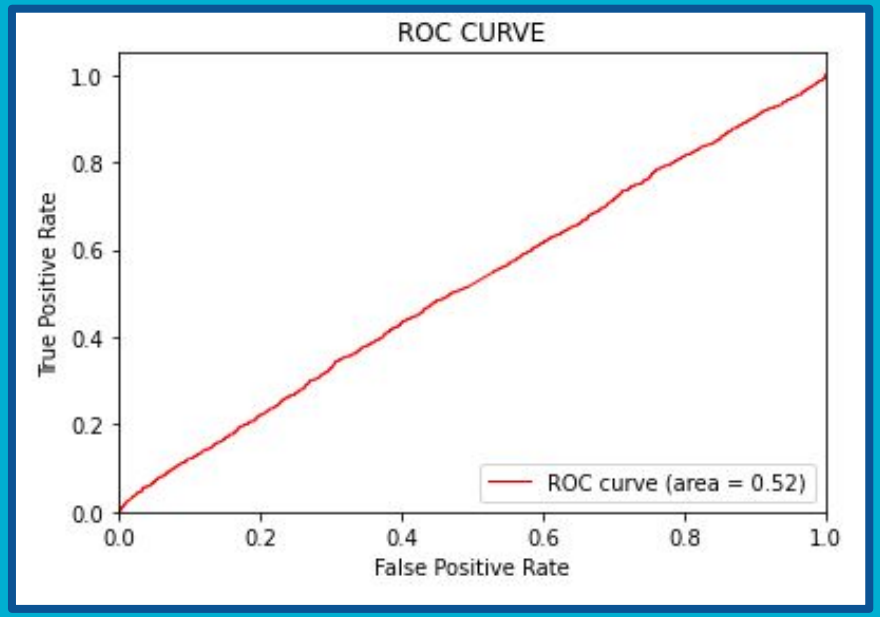



```
predict ("I don't like cats", model_rnn)
1/1 [==============================] - 1s 1s/step
('Label: ', 'POSITIVE', 'Score: 0.60', 'Elapsed_time: 1.46')
predict ("I love cats", model_rnn)
1/1 [==============================] - 0s 24ms/step
('Label: ', 'POSITIVE', 'Score: 0.60', 'Elapsed_time: 0.06')
```

CNN

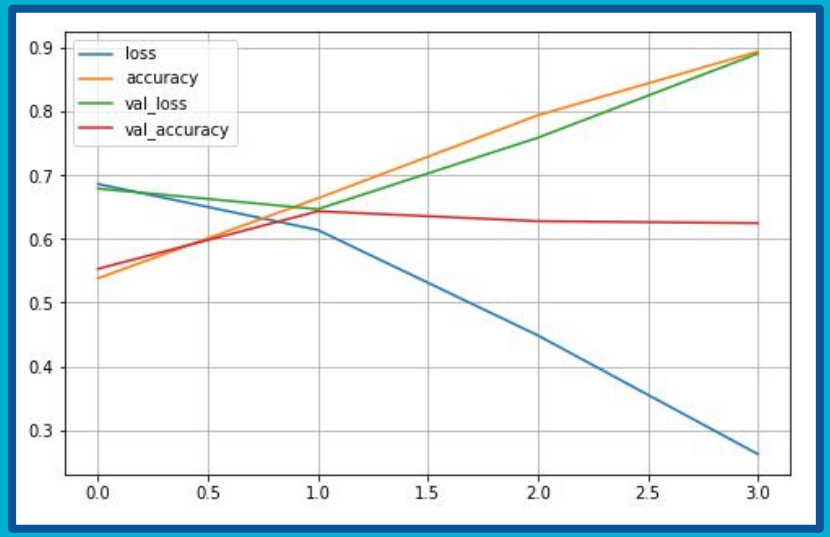
- **Résultats :**
  - **médiocres**
  - **meilleurs que RNN**

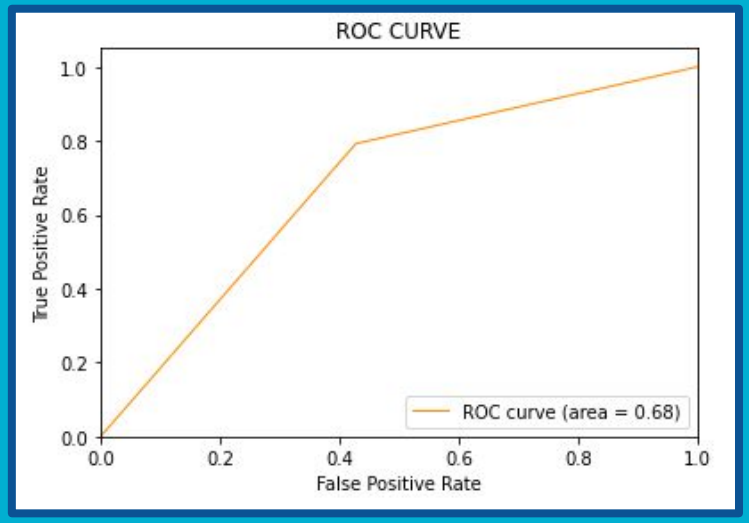
- **Le meilleur score F1 =>**
  corpus prétraité avec le lemmatiseur :
  - **Word2Vec : 0.54**
  - **GloVe : 0.54**



BERT

- Lenteurs à l'exécution
- Résultats :
  - meilleurs que RNN ou CNN
- Le score F1 :
  - corpus lemmatisé : 0.70
  - corpus stemmé : 0.62

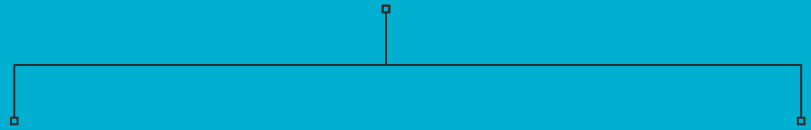




Résultats

Comparaison des scores

Corpus



Stemming

	Loss	Acc	AUROC	Time	F1
Logistic Regression	/	0.72	0.72	1.19	0.72
Keras Emb STEM	0.6	0.69	0.76	6.88	0.47
W2V RNN / LSTM STEM	0.91	0.5	0.5	123.71	0.54
W2V CNN STEM	0.87	0.51	0.51	21.72	0.52
Glove RNN / LSTM STEM	0.9	0.5	0.51	47.84	0.51
Glove CNN STEM	0.8	0.51	0.52	7.64	0.53
BERT STEM	0.73	0.49	0.62	2016.06	0.62

Lemmatisation

	Loss	Acc	AUROC	Time	F1
Logistic Regression	/	0.72	0.72	1.19	0.72
Keras Emb LEMMA	0.6	0.69	0.76	5.18	0.53
W2V RNN / LSTM LEMMA	0.81	0.51	0.51	49.0	0.51
W2V CNN LEMMA	0.81	0.5	0.5	11.88	0.54
Glove RNN / LSTM LEMMA	0.89	0.5	0.5	37.26	0.48
Glove CNN LEMMA	0.84	0.51	0.51	7.63	0.54
BERT LEMMA	0.67	0.6	0.68	1940.3	0.7

Mise en production

Fast API + GitHub + Heroku

FastAPI

HEROKU

Curl

```
curl -X 'POST' \
  'https://fastapi-projet.herokuapp.com/prediction' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
  "review": "I enjoyed the flight, thank you."
}'
```

Request URL

```
https://fastapi-projet.herokuapp.com/prediction
```

Server response

Code	Details
200	Response body

```
{
  "prediction": "The sentiment is positive :-)"
}
```

Curl

```
curl -X 'POST' \
  'https://fastapi-projet.herokuapp.com/prediction' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
  "review": "This flight was absolutely awful."
}'
```

Request URL

```
https://fastapi-projet.herokuapp.com/prediction
```

Server response

Code	Details
200	Response body

```
{
  "prediction": "The sentiment is negative :-("
}
```

Conclusions

Envisager des actions supplémentaires pour obtenir de meilleurs résultats :

- augmenter la volumétrie de l’ échantillon
- mieux prétraiter les données textuelles
- une recherche d'hyper paramètres plus poussée


Avez-vous des questions ?