AIR PARADIS

Détectez les Bad Buzz grâce au Deep Learning

1. INTRODUCTION

Les équipes IA de MIC (Marketing Intelligence Consulting) ont été mandaté par la société AIR PARADIS sur une problématique Marketing : “La détéction d’analyse de sentiment au travers des réseaux sociaux”.

L’objectif d’AIR PARADIS est d’automatiser la détection de sentiment négatif, afin de limiter sa propagation et répondre rapidement à la demande client.

2. DÉMARCHE

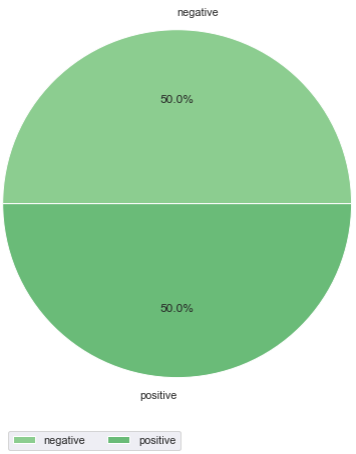
La société AIR Paradis ne disposant pas de jeu de données, nous avons utilisé un jeu de donnée provenant de Kaggle: Sentiment140 dataset with 1.6 million tweets.
Chacun des tweets référencés est labéllisé avec un sentiment négatif (=0) ou positif(=4), nous allons concevoir et proposer à AIR PARADIS différentes solutions de classifications de texte pour lesquelles nous comparerons les performances.

A. Présentation du jeu de donnée

Nous téléchargeons un fichier d’une taille de 227 Mo, contenant 6 colonnes :

- target (label positif / négatif : Conservé)
- id (Non conservé)
- date (Non conservé)
- flag (Non conservé)
- user (Non conservé)
- text (texte à classifier : Conservé)

Nous notons que la distribution des classes est parfaitement équilibrée (50/50%).



B. Présentation des outils de classifications

Voici les différents outils de classifications utilisés lors de notre étude:

1. Machine Learning - Apprentissage supervisé : Nous utiliserons ici une partie du jeu de donnée pour entrainer nos modèles de classification binaires.
2. Deep Learning - les réseaux de neurones : Les modèles de Deep learning ayant tendance à mieux fonctionner avec une grande quantité de données, nous accroitront la taille du jeu d'entrainement.

C. Présentation des métriques de performance

Pour chacun de nos modèles détaillés ci-dessous, nous fournirons les valeurs associée à la matrice de confusion.
Pour rappel: La matrice de confusion permet de mesurer le nombre de prédiction d'un algorithme, séparé en quatre grandes catégories:

- *Les vrais positifs*
- *Les vrais négatifs*
- *Les faux positifs*
- *Les faux négatifs*

A partir de cette matrice, nous obtiendrons différentes métriques permettant de comparer les modèles, en fonction du besoin métier:

- *Accuracy* : Somme de tous les vrais positifs et vrais négatifs qu'il divise par le nombre total d'instances ⇒ (VP + VN) / (VP + VN + FP + FN)
- *Précision* : Rapport entre les prévisions positives correctes et le nombre total de prévisions positives ⇒ VP / (VP + FP).
- *Rappel* : Paramètre qui permet de mesurer le nombre de prévisions positives correctes sur le nombre total de données positives ⇒ VP / (VP + FN)
- *Score F1* : Moyenne harmonique de la précision et du rappel ⇒ 2 x (Précision x Rappel) / (Précision + Rappel).
- *FBetaScore* : Équivalent au F1-score si Beta=1. Dans notre cas: Beta = 0.5 donc accorde plus d'importance à la précision (et donc à la détection des faux positifs) ⇒

En apprentissage supervisé, la notion de **perte d'information** (*loss* en anglais) due à l'approximation doit être minimisée.
Aussi, nous veillerons à mesurer l'**accuracy** et la **loss** lors de nos phases d'apprentissage, afin d'éviter le risque de sur-aprentissage du modèle.

D. Présentation des challenges

Dans notre cas, nous travaillerons les technologies de "traitement du texte" appelé NLP (Natural Language Processing). Cependant, l'une des difficultés rencontrée consiste à gérer la diversité des textes et de leur qualité:

- Syntaxe
- Orthographe
- Caractère spéciaux et URLs
- Longueur des champs

Ainsi, nous devons mettre en place une étape de Pré-traitement permettant de nettoyer le texte avant l'entrainement du modèle, et avant la classification de sentiment.

Voici une description succinte des outils de pré-traitement analysés:

- Construction d'un preprocess NLP classique (voir explication ici)
    - Tokenisation
    - Suppression Regex
    - Changement LowerCase
    - Suppression Stop-Words
    - Lemmatisation ou/et Stemmatisation
- Utilisation d'une librairie spécifique pour le Tweet préprocessing

Nous détaillerons ci-dessous les résultats pour lesquelles les textes ont été pré-traité via la librairie Tweet-Preprocessing, car elle est plus performante dans le cadre d'une analyse de sentiment sur des réseaux sociaux.

3. COMPARATIFS DES SOLUTIONS

Le coté pragmatique de la mesure de performance ne suffit pas à évaluer un modèle de classification. Ainsi, je développerai quelques aspects subjectifs, comme le coût de Developpement, le Coût de Déploiement ou la consommation de ressources machines.

A. Machine Learning - Apprentissage supervisé

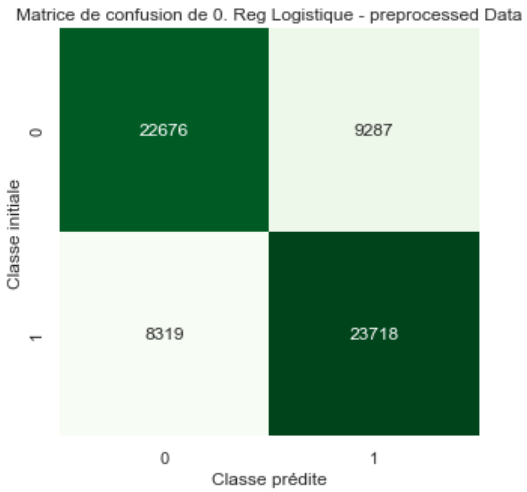
Les algorithmes expliqués dans ce chapitre sont construit de bout en bout. Suite au pré-traitement du texte, nous construirons un Bag of Words consistant à représenter le document par un ensemble des mots qu'il contient.

En pratique, cela résulte en un encodage des mots en vecteurs de fréquence, distanciés par les fréquence d'apparition au travers de la matrice TF-IDF.

a. Régression Logistique

On se sert de la classe LogisticRegression du module sklearn.
La régression logistique est un modèle mathématique qui estime la probabilité, se situant entre 0 et 1. Quand la valeur est supérieure à 0,5, le sentiment est classifié positif et inversement.

L'Accuracy obtenu est de 0.723 (Pourra être accru à 0.76 via une optimisation des Hyper-paramètres et du Bag of Words).



Model Name	seuil	Accuracy_score	FBeta-Score	F1-Score
0. Reg Logistique - preprocessed Data	0.5	0.724906	0.722859	0.729313

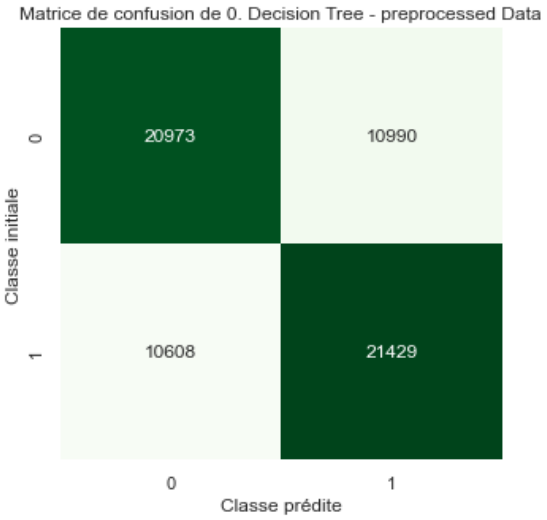
Notes:

Temps de Dev	Temps de Déploiment	Ressources machines	Performance
★★★☆☆	★★★★☆	★★★★☆	★★★☆☆

b. Decision Tree

Nous utilisons ici la classe DecisionTreeClassifier du module sklearn. L'apprentissage par arbre de décision désigne une méthode basée sur l'utilisation d'un arbre de décision comme modèle prédictif.

L'Accuracy obtenu est de 0.66.



Model Name	seuil	Accuracy_score	FBeta-Score	F1-Score
0. Decision Tree - preprocessed Data	0.5	0.662531	0.662563	0.664919

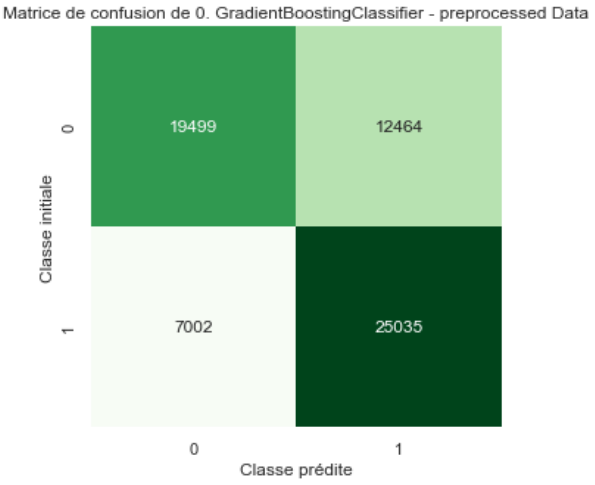
Notes:

Temps de Dev	Temps de Déploiment	Ressources machines	Performance
★★★☆☆	★★★★☆	★★★☆☆	★☆☆☆☆

c. Gradient Boosting

Nous utilisons ici la classe GradientBoostingClassifier du module sklearn. Le renforcement du gradient est une technique populaire en raison de sa précision. Il est peu adapté aux données contenant de nombreuses features/variables, comme le texte.

L'Accuracy obtenu est de 0.69.



Model Name	seuil	Accuracy_score	FBeta-Score	F1-Score
0. GradientBoostingClassifier - preprocessed Data	0.5	0.695844	0.68765	0.720059

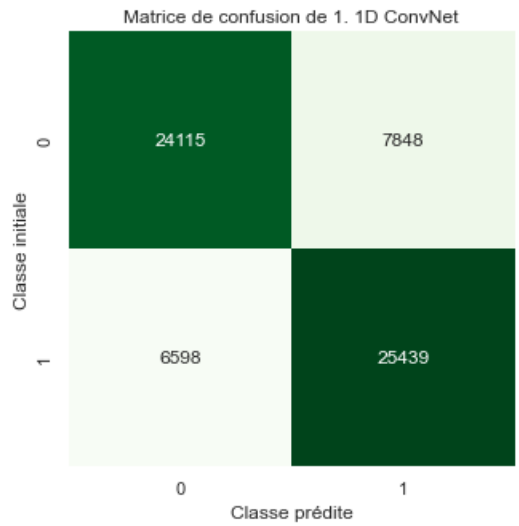
Notes:

Temps de Dev	Temps de Déploiment	Ressources machines	Performance
★★★☆☆	★★★★☆	★★☆☆☆	★★☆☆☆

B. Deep Learning - Réseaux de Neurones

Nous prototypons ici différents réseaux de neurones CNN, en utilisant la librairie Keras de Tensorflow.

a. 1DConvNet



Un réseau CNN est construit par la succession d'une couche de convolution et d'une couche d'aggrégation de l'information (Pooling) et se termine par une couche de neurones totalement connecté.

Nous allons dans notre cas utiliser un réseau de neurones qui "empile" :

- Deux couches de convolution
- Une couche de pooling
- des couches Dropout et Flatten

```
Accuracy: 77.428126
Loss: 48.209625
```

Model Name	seuil	Accuracy_score	FBeta-Score	F1-Score
1. 1D ConvNet	0.5	0.743188	0.752273	0.73135

L'Accuracy obtenu est de 0.774.

Notes:

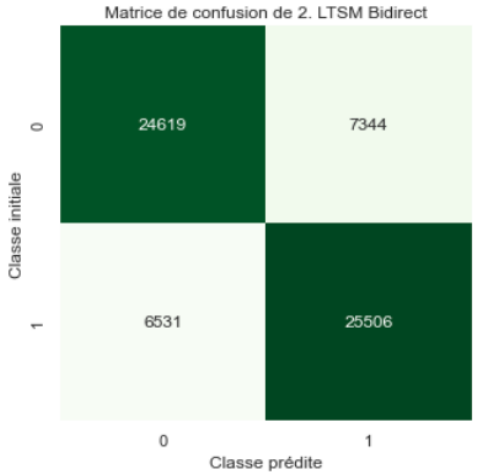
Temps de Dev	Temps de Déploiment	Ressources machines	Performance
★★☆☆☆	★★☆☆☆	★★☆☆☆	★★★☆☆

b. Bi-LSTM

Un réseau LSTM comporte une cellule de mémoire, typiquement une couche de neurones, ainsi que trois portes : une porte d'entrée, une porte de sortie et une porte de l'oubli.

Bidirectional Long Short-Term Memory permet d'utiliser l'information après et avant les données étudiées par le réseau au temps t.

Pour la couche bi-LSTMs de l'extracteur de représentations, nous avons fixé sa dimension à 64.



L'Accuracy obtenu est de 0.784.

Model Name	seuil	Accuracy_score	FBeta-Score	F1-Score
2. LTSM Bidirect	0.5	0.784	0.783553	0.78484

Notes:

Temps de Dev	Temps de Déploiment	Ressources machines	Performance
★★☆☆☆	★★☆☆☆	★★☆☆☆	★★★☆☆

c. Transfer Learning Glove

Le Transfer Learning, ou apprentissage par transfert en français, désigne l'ensemble des méthodes qui permettent de transférer les connaissances acquises à partir de la résolution de problèmes donnés pour traiter un autre problème.

En utilisant des modèles pré-entraînés comme point de départ, le Transfer Learning permet de développer rapidement des modèles performants et résoudre efficacement des problèmes complexes.

Dans ce cas, nous utilisons les vecteurs Glove (Pennington et al., 2014) pré-appris sur 42 milliards de mots du Web, et nous ré-entrainons sur le réseau LSTM précédemment détaillé.

L'Accuracy obtenu est de 0.788 (Pourra être accru à 0.794 via une augmentation de la taille du jeu d'entrainement)

Model Name	seuil	Accuracy_score	FBeta-Score	F1-Score
4. Glove Re-trained embedding Bi-LSTM	0.5	0.787016	0.789759	0.785227

Notes:

Temps de Dev	Temps de Déploiment	Ressources machines	Performance
★☆☆☆☆	★★☆☆☆	★★☆☆☆	★★★★☆

d. Transfer Learning Fasttext

Dans ce cas, nous utilisons les vecteurs pré-appris avec FastText, et nous ré-entrainons sur le réseau LSTM précédemment détaillé.

L'Accuracy obtenu est de 0.782.

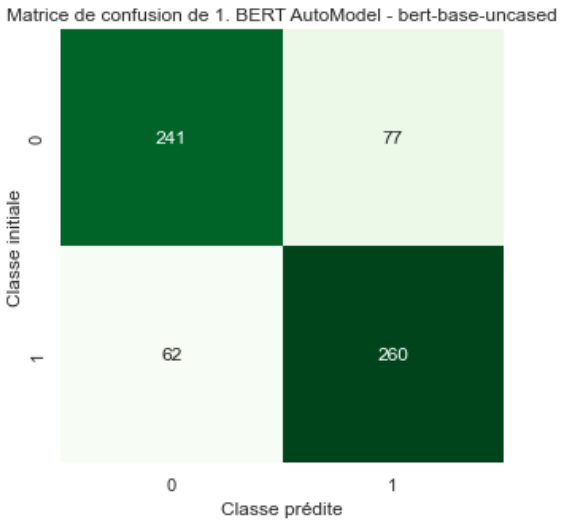
	Model Name	seuil	Accuracy_score	FBeta-Score	F1-Score
8.	Fasttext Re-trained embedding Bi-LSTM	0.5	0.781922	0.787889	0.77729

Notes:

Temps de Dev	Temps de Déploiment	Ressources machines	Performance
★☆☆☆☆	★★☆☆☆	★★☆☆☆	★★★★☆

C. Deep Learning - Réseaux de Neurones (BERT de HuggingFaces)

a. Transfer Learning BERT



BERT c'est pour Bidirectional Encoder Representations from Transformers. Ce modèle a été initialement entrainé à partir du corpus anglophone de wikipedia.
Nous l'entrainons en mode incrémental pour spécialiser le modèle rapidement avec notre jeu de donnée.

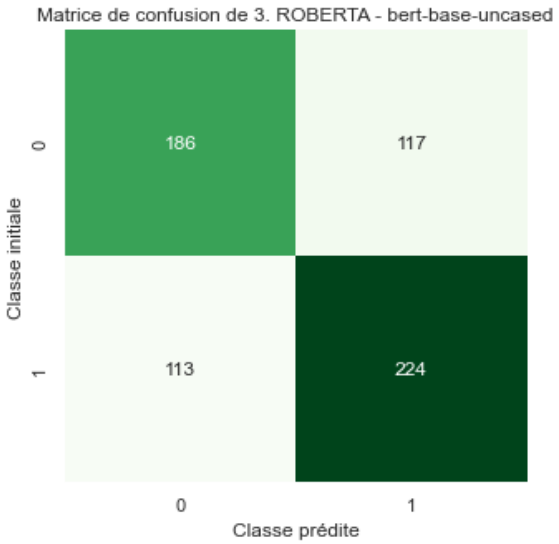
L'Accuracy obtenu est de 0.793 (Pourra être accru à 0.840 avec une taille du jeu d'entrainement correcte)

	Model Name	seuil	Accuracy_score	FBeta-Score	F1-Score
1.	BERT AutoModel - bert-base-uncased	0.5	0.79375	0.775047	0.788462

Notes:

Temps de Dev	Temps de Déploiment	Ressources machines	Performance
★★★★☆	★★★★☆	★☆☆☆☆	★★★★★

b. Transfer Learning RoBERTa



RoBERTa est une version de BERT pour laquelle certains hyperparamètres du pré-entraînement ont été modifiés pour la prédiction de texte.

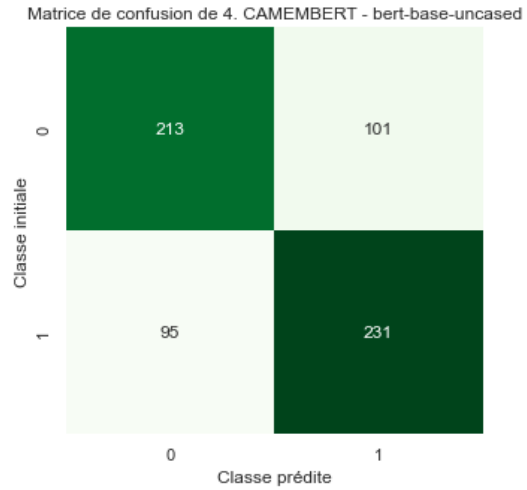
L'Accuracy obtenu est de 0.534.

	Model Name	seuil	Accuracy_score	FBeta-Score	F1-Score
3.	ROBERTA - bert-base-uncased	0.5	0.534375	0.443609	0.283654

Notes:

Temps de Dev	Temps de Déploiment	Ressources machines	Performance
★★★★☆	★★★★☆	★☆☆☆☆	★☆☆☆☆

c. Transfer Learning CamemBERT



CamemBERT (Martin et al., 2019) est un modèle linguistique de pointe pour le français, basé sur l'architecture RoBERTa pré-entraînée sur le corpus multilingue OSCAR.

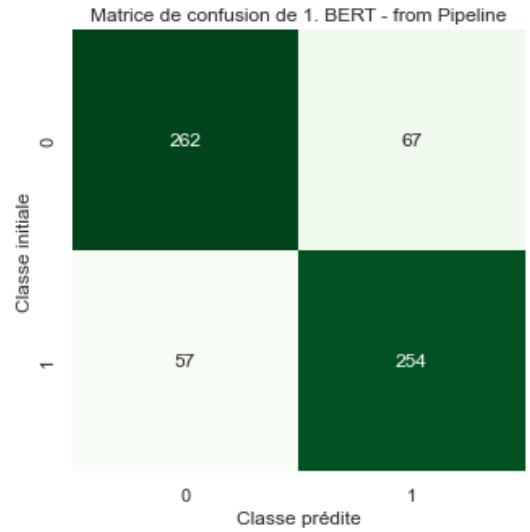
L'Accuracy obtenu est de 0.643.

	Model Name	seuil	Accuracy_score	FBeta-Score	F1-Score
4.	CAMEMBERT - bert-base-uncased	0.5	0.64375	0.643225	0.663717

Notes:

Temps de Dev	Temps de Déploiment	Ressources machines	Performance
★★★★☆	★★★★☆	★☆☆☆☆	★★☆☆☆

d. Native Pipeline BERT



Nous utilisons ici de façon native le pipeline sentiment-analysis pré-entrainé de la librairie Transformers.

L'idée est de comparer les résultats avec le modèle BERT ré-entrainé sur notre jeu de donnée.

	Model Name	seuil	Accuracy_score	FBeta-Score	F1-Score
1.	BERT - from Pipeline	0.5	0.726562	0.73509	0.707846

Notes:

Temps de Dev	Temps de Déploiment	Ressources machines	Performance
★★★★★	★★★★☆	★☆☆☆☆	★★☆☆☆

4. CONCLUSION

Nous préconisons ainsi pour la société AIR PARADIS le modèle BERT Fine-tuned, qui apporte 3 plus-values distinctes:

- Performance des résultats dans la détéction de sentiment
- Facilité de mise en oeuvre et facilité de déploiement
- Maintenance aisée, car utilisation d'une librairie prête à l'emploi

Le seul inconvénient de ce modèle réside dans sa consommation de ressource, pour lequel il nécessitera de ressource Cloud type GPU.

Model Name	Accuracy	FBeta (Beta = 0,5)	F1Score	Temps de Dev	Temps de Déploiment	Ressources machines	Performance	Resultat Etude
BERT	0,840	0,838	0,841	4	4	1	5	3,5
Transfer Learning Glove	0,787	0,790	0,785	1	2	2	4	2,25
Transfer Learning Fasttext	0,782	0,788	0,777	1	2	2	4	2,25
Reg Logistique	0,760	0,764	0,770	3	4	4	3	3,5
Conv1D + global max pooling	0,774	0,770	0,778	2	2	2	3	2,25
LTSM Bidirect	0,784	0,784	0,784	2	2	2	3	2,25
Gradient Boosting	0,696	0,688	0,720	3	4	2	2	2,75
CAMEMBERT	0,694	0,698	0,702	4	4	1	2	2,75
PIPELINE	0,727	0,735	0,708	5	4	1	2	3
Decision Tree	0,663	0,663	0,665	3	4	4	1	3
ROBERTA	0,534	0,443	0,284	4	4	1	1	2,5