Master 1 Informatique Parcours ICO

Rapport de projet

EXTRACTION DE CONNAISSANCES POUR LE DOMAINE DU FACT-CHECKING

Réalisé par :
KIMOUCHE Aicha
BENARAB Hanane
RAIHANE Hamza
TERROUFI Safae

Encadré par :
M.Todorov Konstantin

Année Universitaire :2021/2022

Remerciements

Tout d'abord, nous adressons nos sincères remerciements à notre professeur, Mr Todorov qui nous a accompagnés, encadré durant tout le long du projet et pour le partage de son expertise au quotidien. Grâce à sa confiance nous avons pu accomplir totalement nos missions.

Nous tenons également à remercier l'ensemble de nos enseignants et professeurs du Master ICO, qui nous ont aidé à acquérir les compétences indispensables à la réussite de ce projet.

Sommaire

1 Introduction

Plus que jamais, la société de l'information est atteinte de ce virus meurtrier : fake news « Le nouveau coronavirus est une arme biologique développée par l'homme », « La 5G est en réalité à l'origine de l'épidémie »... cette augmentation prolifique de la diffusion de fausses informations et de fausses nouvelles mènent les gens à croire à des affirmations et des histoires fausses et potentiellement dangereuses d'ou la nécessité du Fact-Checking.

La vérification traditionnelle des faits par des experts et des analystes ne peut pas suivre le volume des informations nouvellement créées. Il est donc important et nécessaire d'améliorer notre capacité à déterminer par calcul si un énoncé de fait est vrai ou faux. Les informations exposées sur les réseaux sociaux par exemple, ne sont plus filtrées en étant au préalable passées au crible par des experts et journalistes. Elles sont à la place hiérarchisées a posteriori par des algorithmes de classement et de référencement qui dépendent en partie des clics et «likes» des internautes. Les informations sont donc vérifiées à la source en extrayant les données issues des sites web et en étudiant leurs véracités.

De ce fait, pour lutter contre la déformation, de nombreuses plateformes dédiées aux Fact-Checking ont été spécialement créées, pour notre projet on a utilisé la plateforme Fullfact qui est une organisation caritative britannique, basée à Londres,qui vérifie et corrige les faits rapportés dans les nouvelles ainsi que les allégations qui circulent sur les réseaux sociaux.

2 Objectif du projet

Le but du projet est de proposer des modèles de classification supervisée d'assertions faites par des figures politiques selon leurs véracités. Autrement dit, nous devons proposer une approche automatique de fact-checking. Pour le faire, nous avons sélectionné un ensemble de données pour nos expériences qui contient des informations provenant de plusieurs domaines (tels que la politique, la santé et la technologie) et un mélange d'articles vrai, faux et incertain. L'ensemble de dataset est sur le site FullFact.

Les méthodes manuelles d'identification d'un article de presse comme réel ou faux prennent beaucoup de temps et nécessitent des efforts considérables. Dans ce projet, un modèle d'apprentissage automatique (ML) a été développé afin d'automatiser ce processus. Des techniques de traitement du langage naturel (NLP) ont été utilisées pour analyser le contenu textuel des nouvelles et trouver des modèles qui classeront les nouvelles comme factuelles ou fausses.

3 Fact-checking

3.1 Définition et objectif :

La vérification des faits, est une technique consistant d'une part à vérifier la véracité et l'exactitude des faits et des chiffres présentés dans les médias et les réseaux sociaux par des personnes publiques, notamment des personnalités politiques et des experts, et, d'autre part, à évaluer le niveau d'objectivité des médias eux-mêmes dans leur traitement de l'information.

Mise en pratique par des journalistes dans le cadre de leur profession, la méthode s'est démocratisée grâce à des logiciels aidant les particuliers à vérifier les faits.

3.2 Machine Learning dans le Fact checking :

Le Machine Learning, ou apprentissage automatique en français, est un sous-domaine de l'intelligence artificielle (IA). Il a pour objectif de comprendre la structure des données et de les intégrer dans des modèles qui peuvent être compris et utilisés pour tester d'autres données. Le Machine Learning est né grâce aux technologies de reconnaissance de pattern et à la théorie selon laquelle les ordinateurs peuvent apprendre sans être programmés pour effectuer des tâches spécifiques (ex ; classification, régression).

Son utilisation dans le fact-checking revient au fait, que les méthodes manuelles d'identification d'une claim comme réel ou faux prennent beaucoup de temps et nécessitent des efforts considérables, ce processus se révèle incroyablement long et fastidieux. De plus, les modérateurs humains peuvent eux-mêmes faire preuve d'une subjectivité néfaste dans leurs jugements.

C'est la raison pour laquelle une solution reposant sur le Machine learning pourrait s'avérer extrêmement utile.

3.3 Extraction de données depuis FullFact

Full Fact est une organisation caritative britannique, qui vérifie et corrige les faits rapportés dans l'actualité dans le monde entier ainsi que les affirmations qui circulent sur les médias notamment les réseaux sociaux. Ce site web se présente de cette façon :

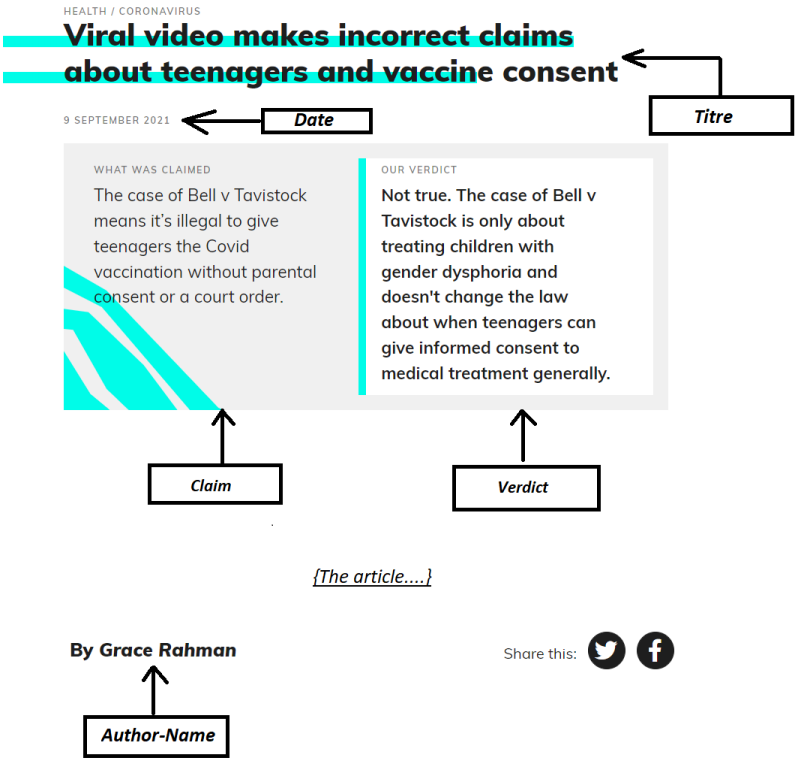


FIGURE 1 – La présentation de Full Fact

L'extraction des claims du site web FullFact, nous l'avons faite

de manière manuelle pour pouvoir observer l'importance des quantités de données utilisées et de la taille du dataset dans la précision des modèles.

Nous avons commencé par extraire 100 claims puis 200 claims et on a franchi la barre des 500 claims pour avoir une précision plus importante.

4 Conception du système

Notre système se base sur l'utilisation du Machine Learning pour détecter la véracité des claims et ainsi les Fake News. Le système prend en entrée un fichier CSV de données extraites du site web fullFast et les transforme en une base utilisable par la phase d'apprentissage. Cette transformation est appelée pré-traitement, elle effectue une série d'opérations telles que le nettoyage, le filtrage et la tokenisation.

La base pré-traitée est subdivisée en deux parties ; une pour l'entraînement et l'autre pour le test. Le module d'entraînement utilise la base d'entraînement et un algorithme d'apprentissage pour fournir un modèle de décision qui est appliqué sur la base de test. Si le modèle est accepté, c-à-d a pu atteindre un taux de reconnaissance acceptable, il sera conservé et utilisé par le module d'utilisation et l'entraînement se termine. Dans le cas contraire, les paramètres de l'algorithme d'apprentissage sont révisés dans le but d'améliorer le taux de précision.

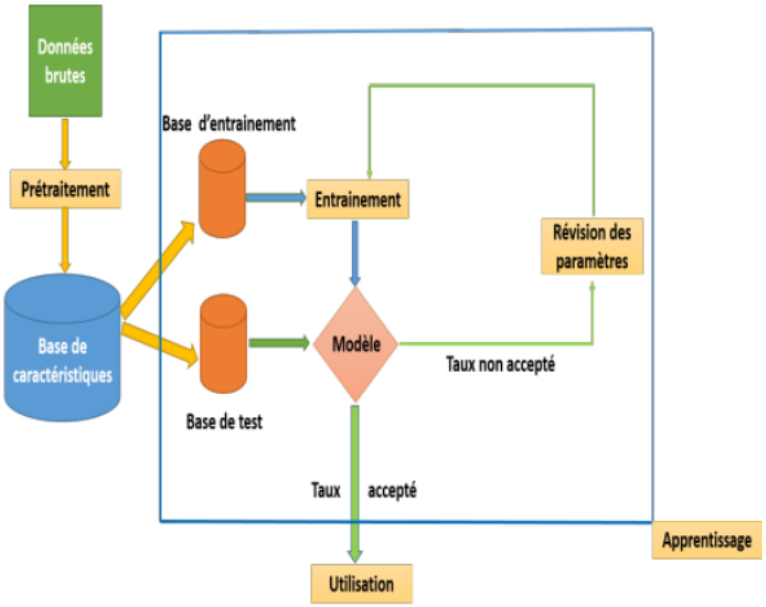


FIGURE 2 – Architecture générale

4.1 Pré-traitement des données

Le pré-traitement des données brute, est une étape primordiale dans le traitement de langage naturel, la qualité des données doit être vérifiée avant l'usage d'algorithmes d'apprentissage automatique, nous avons utilisé les techniques suivantes :

Nettoyage : Qui consiste à éliminer les mots vides tel que "a", "about", "am", "you", "are", "it"...ext et les caractères spéciaux tel que ! ? : ; , ...ext et toute information non utile.

Lemmatisation du texte : Lemmatisation du texte : il s'agit de la réduction de mots et considère le vocabulaire complet d'une langue pour appliquer une analyse morphologique aux mots, visant à supprimer uniquement les fins flexionnelles et à renvoyer la

forme de base ou de dictionnaire d'un mot.

Tokenisation : C'est un aspect clé (et obligatoire) du travail avec des données textuelles, c'est un moyen de séparer un morceau de texte en unités plus petites appelées jetons. Il consiste à identifier les unités de textes élémentaires qui peuvent être des mots, mais aussi des lettres, des syllabes, des phrases, ou des séquences de ces éléments.

Encodage : Permet convertir les fichiers texte en vecteurs de caractéristiques numériques. Pour ce faire, nous avons utilisé le modèle **bag-of-word** qui compte le nombre de fois où chaque mot apparaît dans chaque texte et enfin à l'aide de TF-ID pour obtenir les fréquences pondérées. ensuite il faut appliquer le **N-grammes** qui vont nous servir dans de l'analyse de texte dans lesquelles les séquences de mots sont appropriées. Les expressions de mots suivantes représentent 2 grammes : «New York» et 3 grammes : «Les trois mousquetaires».

Vectorisation du texte :L'application d'un algorithme quel qu'il soit implique le traitement des données sous forme numérique. Dans le cadre de l'analyse de texte, cette numérisation consiste à transformer un texte en un vecteur de nombres, Les observations ne changent pas dans cette opération et se rapportent au même texte, mais les colonnes deviennent alors des features où chaque feature se rapporte à un mot et contient la fréquence ou le compte d'apparition de chaque mot.

4.2 Apprentissage supevisé

L'apprentissage supervisé est une méthode de l'apprentissage automatique s'appuyant sur des données labellisées (étiquetée)

pour entraîner des modèles de l'IA prédictifs.

En se basant sur cette base d'apprentissage, les paramètres du modèle s'ajustent en vue ensuite de réagir efficacement face à des situations similaires c'est -à -dire des données non labellisées.

Au fur et à mesure de l'enrichissement du modèle, le résultat gagne en pertinence, réduisant la marge d'erreur.

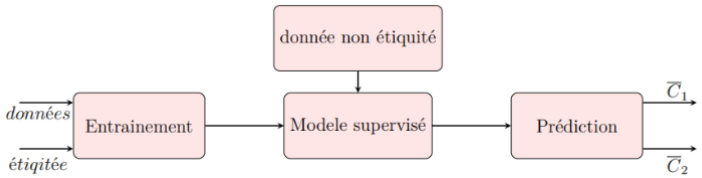


FIGURE 3 – Apprentissage supervisé

Il regroupe deux modules, l'entraînement et la validation utilisant chacun une partie de la base des caractéristiques subdivisée en deux parties, base d'entraînement et base de test. Le module d'entraînement utilise la base d'entraînement pour fournir un modèle de décision tandis que le module de validation utilise la base de test pour mesurer la performance du modèle fourni

→ Entrainement :

Pour entraîner notre modèle nous avons choisi plusieurs classifiers afin d'évoluer leurs performances sur notre base pré-traité tel que RandomForest, Naive Bayes, Logistic Regression.

Le résultat de l'entraînement est un modèle, qui représente l'analyse des données et leur transformation en informations utiles, en établissant des relations entre elles, Plusieurs métriques sont utilisées pour estimer la qualité du modèle qui se basent sur les valeurs

suivantes qui représente la matrice de confusion :

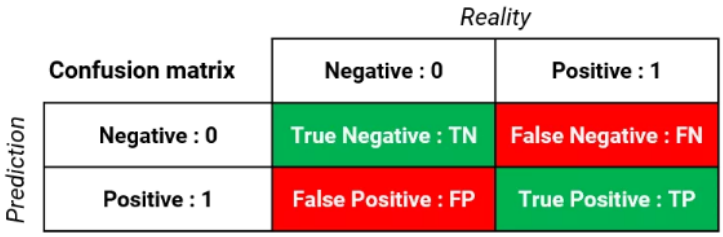


FIGURE 4 – Matrice de confusion pour la classification binaire

1. VP : les exemples positifs classés correctement. ;
2. FP : les exemples positifs mal classés. ;
3. VN : les exemples négatifs classés correctement. ;
4. FN : les exemples négatifs mal classés.

les principales mesures de performance utilisées afin d'évaluer l'efficacité des modèles de classification :

Performance Metric	***Formula***
Accuracy	(TP + TN) / (TP + TN + FP + FN)
Precision	TP / (TP + FP)
Recall (Sensitivity)	TP / (TP + FN)
F1 Score	(2 * recall * precision) / (recall + precision)

FIGURE 5 – Formules de mesure des performances

→ Validation :
Consiste à mesurer la capacité du modèle à reconnaître des nouvelles claims et à les classifier. Pour cela, la base est subdivisée

en deux parties dès le début en une partie d'entraînement et une partie de test. Son utilité consiste à éviter le sur-apprentissage, c-à-d tester le modèle sur la même base d'entraînement.

Pour cela nous avons utilisé la technique de Cross-Validation :

→ Cross-Validation :

Il évalue le modèle en utilisant différents morceaux de l'ensemble de données comme ensemble de validation. Nous divisons notre ensemble de données en K-plis. K représente le nombre de plis dans lesquels on souhaite diviser nos données. Si nous l'utilisons 5 fois, l'ensemble de données se divise en cinq sections. Dans différentes itérations, une partie devient l'ensemble de validation.

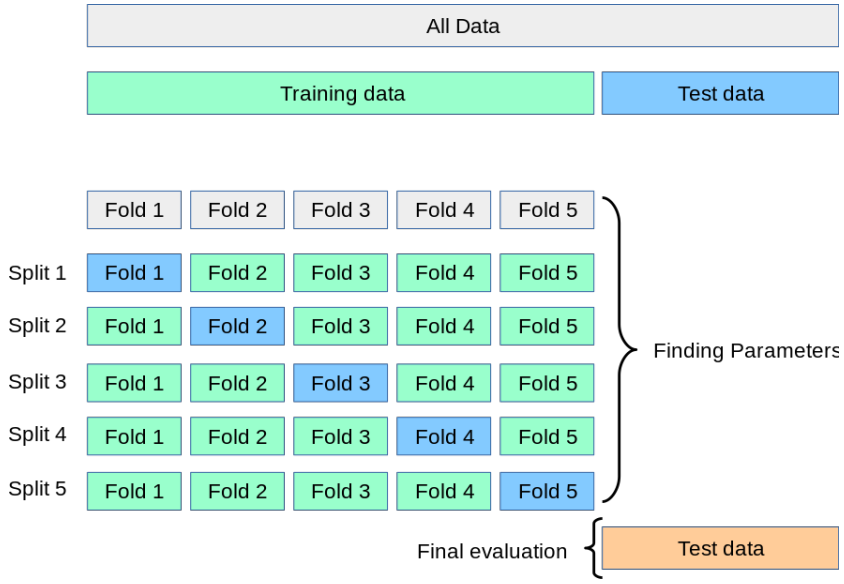


FIGURE 6 – Cross Validation

4.3 Implementation Résultats

4.3.1 Visualisation des données du dataSet

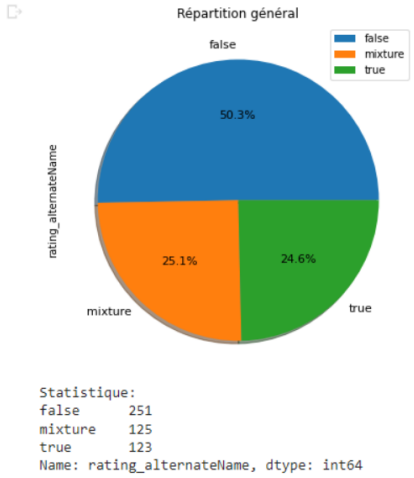


FIGURE 7 – Répartition des données du Dataset

La répartition des classes de notre dataset n'est pas équilibrée, Cela entraînera effectivement une mauvaise précision au niveau des modèles qu'on va construire, mais il existe des solutions afin d'y remédier.

4.3.2 Classification

Nous appliqué un premier classifier qui est le logistic Regression et nous avons obtenu les résultat suivant :

En observant la matrice de confusion, on constate que notre classifier ne classe pas correctement les claims de nature Mixture, nous avons essayé un autre classifier pour comparer les résultats avant de proposer une solution afin de pallier ce problème. On y observe également un phénoméne d'Overfitting (sur-

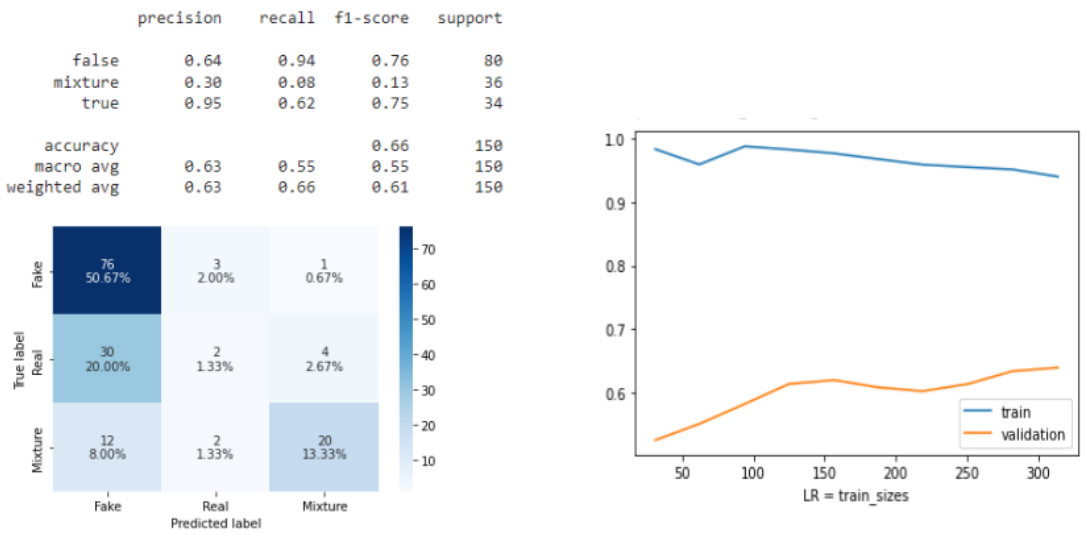


FIGURE 8 – Résultats de Logistic Regresion

apprentissage) qui désigne le fait que le modèle prédictif produit par l'algorithme de Machine Learning s'adapte bien au Training Set. Par conséquent, le modèle prédictif capturera tous les "aspects" et détails qui caractérisent les données du Training Set. Dans ce sens, il capturera toutes les fluctuations et variations aléatoires des données du Training Set. En d'autres termes, le modèle prédictif capturera les corrélations généralisables ET le bruit produit par les données.

→Naives Bayes :

En appliquant le Naïves Bayes sur notre base prétraité, nous avons eu les résultats suivant :

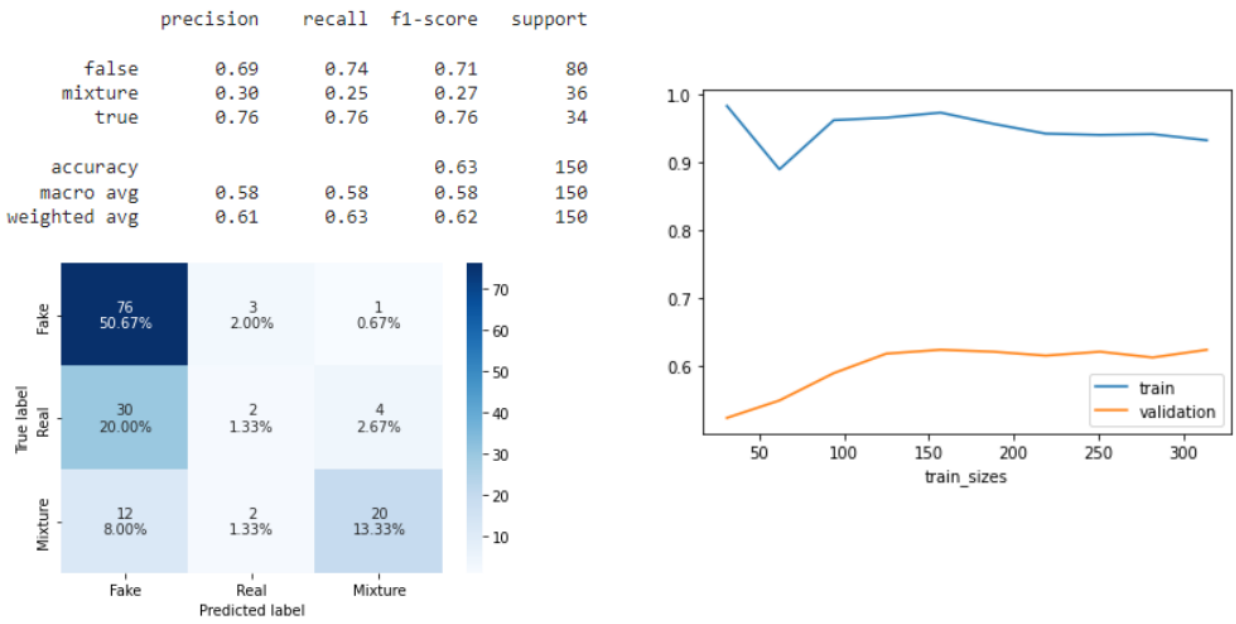


FIGURE 9 – Résultats de Naives Bayes

Au niveau des résultats, on voit qu'il agit mieux au niveau des classes mixture mais un peu moins avec les classes False, mais ça reste insuffisant.

Pour remédier à ce problème, nous avons effectué la technique de suréchantillonnage (upSampling) qui est une procédure dans laquelle des points de données générés synthétiquement (correspondant à une classe minoritaire) sont injectés dans l'ensemble de données. Après ce processus, les comptes des deux étiquettes sont presque les mêmes. Cette procédure de péréquation évite que le modèle ne penche vers la classe majoritaire.

→Aprés le upSampling :

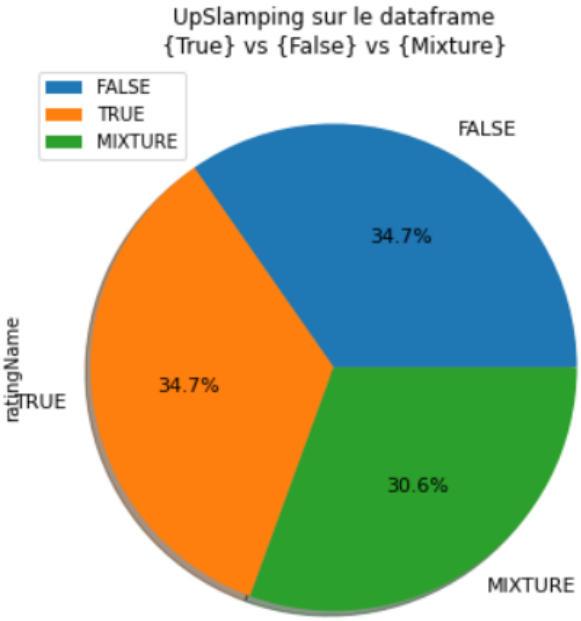


FIGURE 10 – UpSlamping sur le dataFrame

Après le upSampling, la répartition des classes est équilibrée, nous pouvons à présent observer l'impact de cet equilibration sur la classification et sur la matrice de confusion.

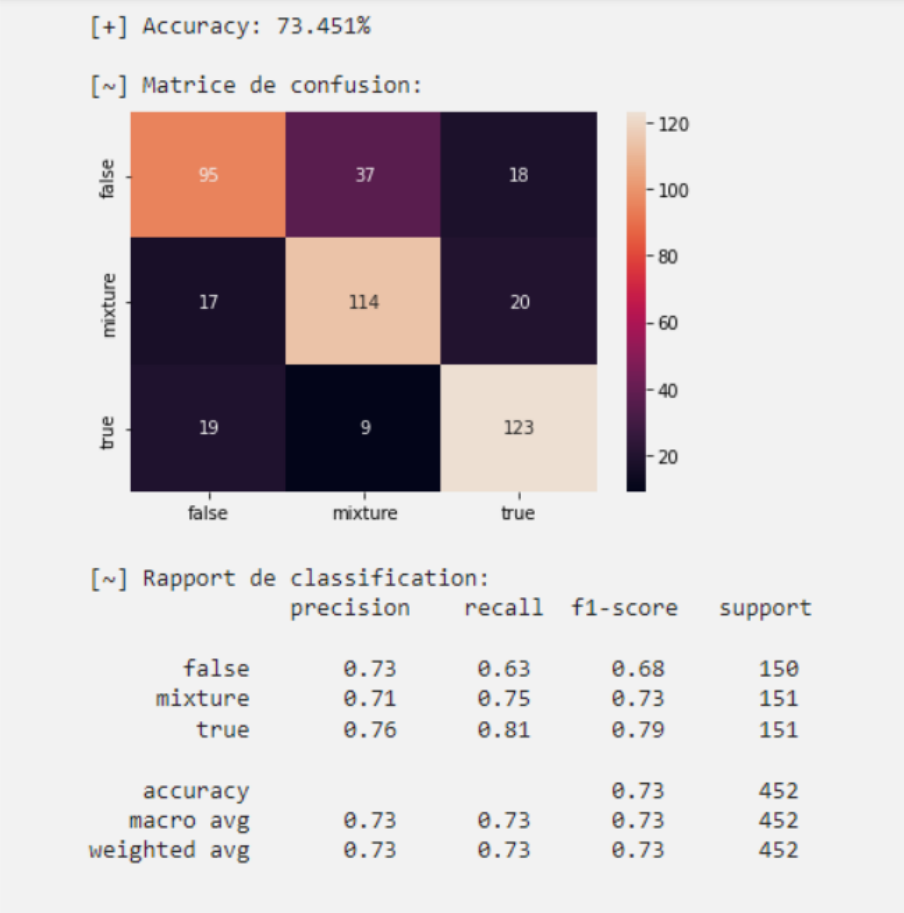


FIGURE 11 – Matrice de confusion après le UpSlamping

Nous observons une grande amélioration au niveau de la matrice de confusion et également la précision qui est passée de 65% à 73% d'où l'importance d'avoir des données équilibrées.

Nous allons désormais effectuer des techniques d'optimisation afin d'optimiser notre modèle et améliorer encore la précision.

5 Optimisation

5.1 Ajout de données Additionnelles

Jusqu'à présent seuls les textes principaux de nos articles étaient traités, pour améliorer notre précision nous pouvons ajouter de nouvelles données mais il faut les choisir avec soin.

En effet, de nouvelles données inutiles ajouteraient du bruit à l'estimation, de même si l'on ajoute des informations redondantes. Nous pouvons ainsi économiser du temps et gagner de la précision. Pour déterminer leurs importances dans l'amélioration de notre précision en ajoutant une unique colonne en plus de notre texte et nous avons obtenu les résultats suivants :

```
Features: ['claimReview_claim', 'creativeWork_author_name\t'] | Accuracy: 78.81%
Features: ['claimReview_claim', 'claimReview_claim'] | Accuracy: 82.45%
Features: ['claimReview_claim', 'claimReview_Title'] | Accuracy: 83.77%
Features: ['claimReview_claim', 'claimReview_Verdict'] | Accuracy: 92.72%
```

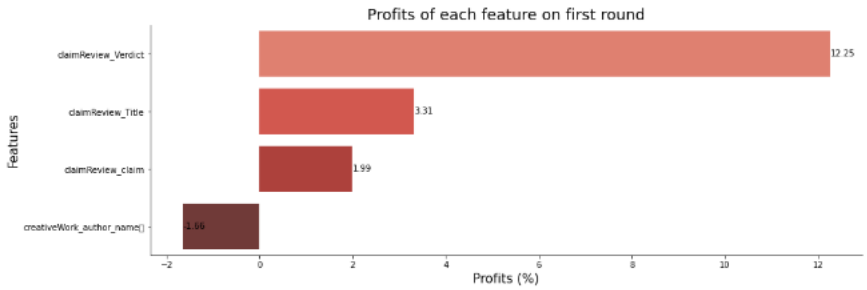


FIGURE 12 – Le profit aprés chaque ajout d'une donnée

Toutes les données sembles utiles à ajouter. Sauf author qui a un profit négatif.

Pour la suite du projet, nous allons les inclure dans la classifica-

tion.

5.2 GridSearch

Afin d'optimiser nos résultats en choisissant les bon hyper-paramètres qui s'adaptent le plus à nos classifiers pour avoir une meilleure performance du modèle, nous avons appliqué le Grid-Search qui va nous permettre de tester une série de paramètres et de comparer les performances pour en déduire le meilleur paramétrage pour les différents classifiers.

— →**Logistic Regression :**

```
[+] Accuracy: 92.699%
[+] Meilleurs paramètres:  {'clf__C': 2, 'clf__max_iter': 1000, 'clf__solver': 'lbfgs'}
```

— →**K-nearest neighbors :**

```
[+] Accuracy: 73.451%
[+] Meilleurs paramètres:  {'clf__leaf_size': 2, 'clf__n_neighbors': 5, 'clf__weights': 'distance'}
```

— →**Arbre de décision :**

```
[+] Meilleurs paramètres:  {'clf__criterion': 'gini', 'clf__min_samples_leaf': 1, 'clf__min_samples_split': 2, 'clf__n_estimators': 200}
```

6 Analyse discussion

Nous avons pu constater grâce à la technique upSampling, qui nous a permis d'équilibrer les classes, sachant que le déséquilibre au début, a fait que le modèle ne prédit pas les classes Mixture, à cause du manque de données dans cette classe et qui a causé également le phénomène de overfitting.

Nous avons pu constater que grâce à l'optimisation (Grid Search qui nous a permit de choisir les hyper-paramètres les mieux adapté

à notre classifier, la sélection des Features, et le rajout de données additionnelle qui nous a permit de voir son impact sur le modèle), on arrive à améliorer la précision de notre modèle comme le démontre la figure suivante :

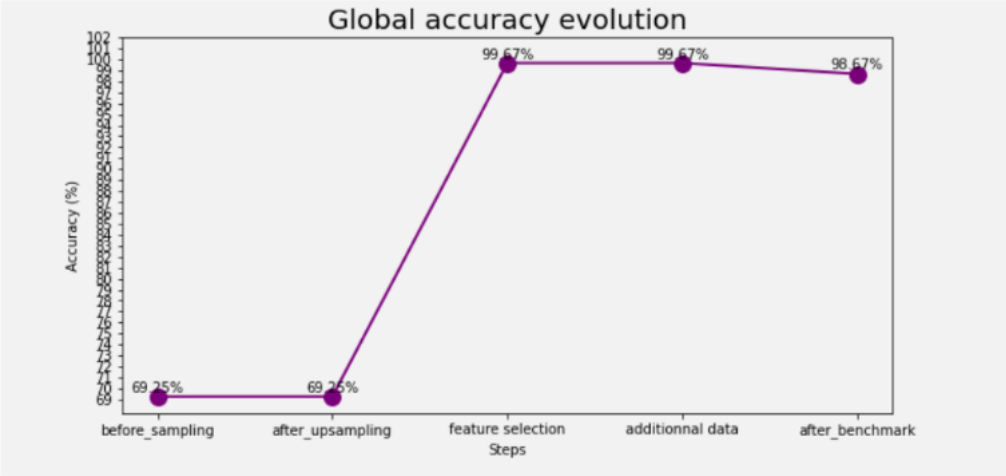


FIGURE 13 – évolution globale de l'acuracy

Finalement, Après le prétraitement, la classification et l'optimisation nous avons pu obtenir un modèle avec une précision de 100 % avec le classifier RandomForest.

Modèles	RandomForest	LogisticRegression	Naives Bayes	K- Nearest Neighbors
Accuracy	100%	92%	98%	73,451%

FIGURE 14 – Valeur de l'accuracy par rapport au differents classifers.

7 Gestion de Projet

Pour la réalisation de notre projet, nous avons adopté la méthode en Cascade,qui est un modèle de gestion linéaire qui divise les processus de développement en phases de projet successives. Contrairement aux modèles itératifs, chaque phase est effectuée une seule fois. Les sorties de chaque phase antérieure sont intégrées comme entrées de la phase suivante. Le modèle en cascade est principalement utilisé dans la Data Sciences.

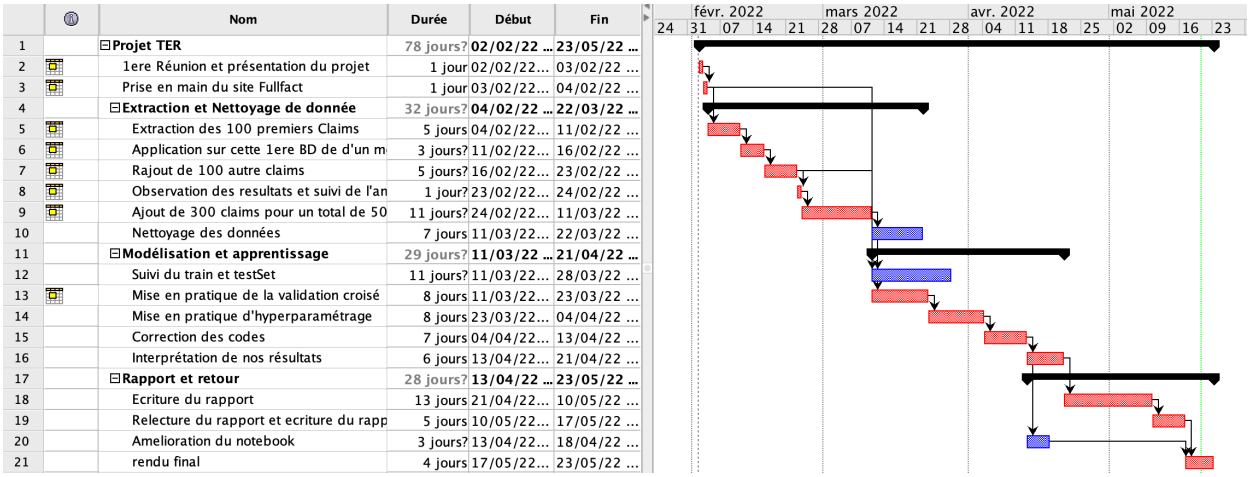
FIGURE 15 – Processus de dévoleppement

L'étude sur la prédiction des véracité des claims, nécessite l'anticipation des demandes des utilisateurs, la définition complète des données et la documentation exhaustive. Notre équipe se constituait d'un chef de projet, deux data scientists et un testeur.

Pour que cette méthode porte ses fruits, tout d'abord nous avons établi lors d'une réunion un cahier de charge détaillé qui explique chaque phase du projet et précise notamment les ressources nécessaires et les tâches à réaliser par chaque membre concerné par le projet. Ce qui nous a permis d'avoir une idée claire sur :

— -Les personnes chargées de chaque étape
— -Les principales dépendances
— -Les ressources nécessaires
— -La chronologie pour définir le temps nécessaire à chaque étape

Pour ce faire ,nous avons conçu un diagramme de Gantt ,avec des tâches définies.



	Nom	Durée	Début	Fin
1	**Projet TER**	78 jours?	**02/02/22 ...**	**23/05/22 ...**
2	1ere Réunion et présentation du projet	1 jour	02/02/22...	03/02/22 ...
3	Prise en main du site Fullfact	1 jour	03/02/22...	04/02/22 ...
4	**Extraction et Nettoyage de donnée**	32 jours?	**04/02/22 ...**	**22/03/22 ...**
5	Extraction des 100 premiers Claims	5 jours	04/02/22...	11/02/22 ...
6	Application sur cette 1ere BD de d'un m	3 jours?	11/02/22...	16/02/22 ...
7	Rajout de 100 autre claims	5 jours?	16/02/22...	23/02/22 ...
8	Observation des resultats et suivi de l'an	1 jour?	23/02/22...	24/02/22 ...
9	Ajout de 300 claims pour un total de 50	11 jours?	24/02/22...	11/03/22 ...
10	Nettoyage des données	7 jours	11/03/22...	22/03/22 ...
11	**Modélisation et apprentissage**	29 jours?	**11/03/22 ...**	**21/04/22 ...**
12	Suivi du train et testSet	11 jours?	11/03/22...	28/03/22 ...
13	Mise en pratique de la validation croisé	8 jours	11/03/22...	23/03/22 ...
14	Mise en pratique d'hyperparamétrage	8 jours	23/03/22...	04/04/22 ...
15	Correction des codes	7 jours	04/04/22...	13/04/22 ...
16	Interprétation de nos résultats	6 jours	13/04/22...	21/04/22 ...
17	**Rapport et retour**	28 jours?	**13/04/22 ...**	**23/05/22 ...**
18	Ecriture du rapport	13 jours	21/04/22...	10/05/22 ...
19	Relecture du rapport et ecriture du rapp	5 jours	10/05/22...	17/05/22 ...
20	Amelioration du notebook	3 jours?	13/04/22...	18/04/22 ...
21	rendu final	4 jours	17/05/22...	23/05/22 ...

FIGURE 16 – Diagramme de Gantt

7.1 Outils de collaborations

Ensuite à la phase de conception nous avons précisé le matérielle que nous utiliserons pour la communication entre les membres de l'équipe à savoir :

→**Discord :** Qui est un logiciel propriétaire gratuit de VoIP et de messagerie instantanée, pour l'organisation de réunions et le suivi d'avancement du projet

→**Google Colab :** C'est un environnement adapté au machine

learning, à l'analyse de données qui permet d'écrire et d'exécuter le code Python de son choix par le biais du navigateur.

→**OpenProject :** Pour la réalisation du diagramme de gantt et pour le suivi du projet.

→**LaTex :** Pour faciliter la rédaction du rapport à plusieurs.

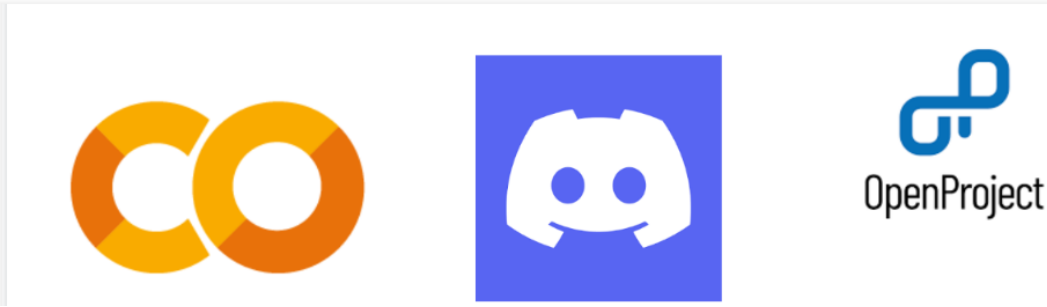


FIGURE 17 – Outils de collaborations

7.2 Difficultés rencontrées

L'extraction manuelle des claims et leur véracité au niveau du site web FullFact était un processus long et difficile. Pour certaines claims, il était difficile de prononcer leurs véracité. Il faut savoir que FullFact est un site web non structuré.

Notre recherche n'a porté que sur un dataset de petite taille (500 claims), cela peut effectivement affecter les résultats obtenus. Il serait plus pertinent d'étendre cette étude avec un dataset plus volumineux et d'évoluer son apprentissage supervisé.

Quant à l'optimisation, nous étions pas très informé des techniques qui existait pour optimiser notre modèle, nous avons effectué beaucoup de recherche pour a la fin utiliser le UpSampling qui a donné

effectivement des bons résultats.

8 Conclusion

A l'issue de ce projet, nous avons pu construire un modéle d'apprentissage automatique supervisé, pour la detection de fake news dans le cadre du Fact Cheking, en essayant de déterminer les meilleures caractéristiques et techniques permettant au mieux d'identifier une fausse nouvelle.

Nous avons conçu et implémenté un modéle qui se base sur l'utilisation des techniques de nettoyage, steaming, encodage par N-gram, sac à mots et TF-IDF pour le pré-traitement des textes brutes des claims, ensuite nous avons vu la technique d'up Sampling qui nous a permis d'équilibrer les classes.

Nous avons appliqué des algorithmes d'apprentissage sur notre base de prétraité pour construire un modèle permettant la classification des nouvelles informations.

Finalement, nous avons pu constater l'importance de l'optimisation (en appliquant GridSearch, la selection des features, l'ajout des données additionelle) sur la précision de notre modéle en allant jusqu'a atteindre une précision de 100 % avec RandomForest.