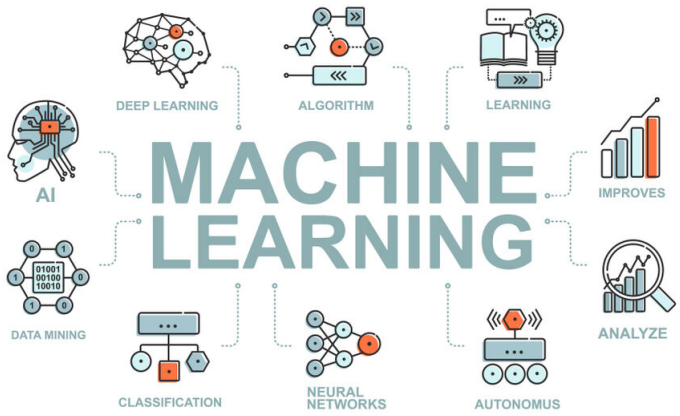
Université de Montpellier

FACULTÉ DES SCIENCES DE MONTPELLIER

M1 IASD

Projet de Machine learning

Détection automatique des fake news à partir de données textuelles (Fake News Detection)



Rédigé par :

Rania BENDAHMANE 21811387
Bachir HADJOUDJA 21811363
Youcef LABIAD 21710780
Rym ZEGGAR 21909615

Encadrant :

Pascal PONCELET

1er mai 2023

Table des matières

1 Introduction

Ce rapport vise à décrire les différentes étapes indispensables pour entraîner un modèle capable de détecter des Fake News. Dans un premier temps, nous procéderons à une sélection rigoureuse de nos données, qui représentent des articles étiquetés selon leur véracité (VRAI, FAUX, MIXTURE ou OTHER). Nous entreprendrons ensuite un pré-traitement de ces données afin de les rendre plus adaptées à notre algorithme en supprimant les informations superflues. Nous examinerons ensuite plusieurs classifieurs potentiels pour effectuer nos prédictions, avant de nous concentrer sur le plus performant. Enfin, nous effectuerons une recherche minutieuse des hyper-paramètres pour affiner notre classifieur et améliorer ainsi nos prédictions.

2 Prétraitements et ingénierie de données

En ce qui concerne cette étape, Nous avons fusionné les deux fichiers mis à notre disposition, New_train et New_test, afin d'obtenir un ensemble de données plus conséquent. Ensuite, nous avons effectué divers prétraitements sur ces données en utilisant la fonction MyCleanText et en ajustant ses paramètres. Plusieurs tests ont été réalisés pour identifier les paramètres les plus pertinents, c'est-à-dire ceux qui produisent les meilleurs résultats. Ces résultats sont présentés ultérieurement dans le rapport.
En ce qui concerne la vectorisation, nous avons expérimenté avec BagOfWords, mais nous avons principalement utilisé la méthode TF-IDF. Cette dernière permet de minimiser l'impact des termes fréquemment présents dans un corpus et qui sont donc moins informatifs que les autres termes dans le corpus d'apprentissage.
Afin d'obtenir un ensemble de données équilibré, nous avons opté pour le Downsampling, car nous ne souhaitions ni dupliquer les données au risque de provoquer un surapprentissage, ni générer de nouvelles données potentiellement incorrectes qui pourraient fausser le processus d'apprentissage. Ce choix nous a semblé le plus judicieux.
Lors de la séparation des données en ensembles d'apprentissage et de test, nous avons veillé à ce que chaque exécution utilise la même répartition pour toutes les classifications. C'est pourquoi nous avons choisi une graine aléatoire afin de garantir une répartition identique à chaque fois.

2.1 Reconnaissance d'entités nommées (Entity Recognition)

Nous avons appliqué des fonctions permettant d'extraire les entités nommées des articles mis à notre disposition, afin de tester et d'analyser l'efficacité de cette méthode dans l'apprentissage automatique. Nous avons intégré ces entités nommées aux tâches de classification, et les résultats obtenus seront présentés ultérieurement.

2.2 Modélisation par sujets (Topic Modeling)

En utilisant le modèle LDA entraîné, nous avons pu identifier les sujets dominants de chaque document ainsi que leurs mots-clés et leurs poids de contribution. Nous avons veillé à ce que la cohérence entre les sujets soit élevée et que la perplexité, c'est-à-dire la capacité du modèle à généraliser, soit faible. Pour ce faire, nous avons sélectionné le nombre approprié de sujets et les meilleurs paramètres pour le modèle. Nous avons également intégré ces éléments aux tâches de classification en ajoutant les mots-clés au texte, de manière à augmenter la fréquence des termes.

2.3 Classifications

Nous avons fait 3 classifications.

- **True vs False :** La première classification vise à prédire la classe True ou False. Nous avons réalisé un notebook de base pour cette classification, ainsi qu'un notebook intégrant les entités nommées et un autre intégrant le topic modeling. Cette approche a été adoptée pour toutes les classifications.
- **(True OR False) vs Other :** La deuxième classification a pour objectif de prédire si un article appartient aux catégories True ou False, ou s'il appartient à la catégorie Other.
- **True vs False vs Other vs Mixture :** Cette classification consiste à prédire les quatre classes. Étant donné la difficulté à distinguer les labels Mixture et Other, les résultats obtenus étaient relativement faibles. Nous avons donc envisagé une double classification :
  - **(True OR False) vs (Other OR Mixture)** : Cette étape prédit si un article appartient aux catégories True ou False, ou s'il appartient aux catégories Mixture ou Other.
  - **Other vs Mixture** : Ici, nous séparons les catégories Other et Mixture en testant divers classificateurs, paramètres et hyperparamètres afin de distinguer les deux classes.
  - **True vs False** : Nous réalisons à nouveau une classification entre True et False en utilisant le meilleur classificateur et les meilleurs paramètres.
  - **Concaténation** : Finalement, nous combinons les résultats des deux classifications précédentes.

3 Classifications et leurs résultats

Dans notre étude, nous avons cherché à identifier les meilleurs classifieurs et les paramètres optimaux (C, gamma et kernel pour SVC, n_estimators et max_features pour RF, etc.) afin d'obtenir les meilleurs résultats en termes d'accuracy, de précision et de rappel. Nous avons exploré plusieurs classifieurs, notamment le Support Vector Classifier (SVC), le Random Forest Classifier (RF), K-Nearest Neighbors (KNN), Decision Tree Classifier, Multinomial Naive Bayes et Logistic Regression. Pour chaque test, nous avons également pris soin d'afficher un "classification report".

Initialement, nous avons sélectionné les meilleurs classifieurs en utilisant une validation croisée (Cross_val) avec un k-fold de 10. Nous nous sommes basés sur la moyenne des accuracy et les écarts types pour choisir le meilleur classifieur. Par la suite, nous avons itéré sur les prétraitements, les classifieurs sélectionnés et les hyperparamètres afin de tester toutes les combinaisons possibles en utilisant GridSearch.

Dans les tableaux présentés, nous avons reporté les résultats des 4 meilleurs prétraitements pour le titre, le texte et le titre avec le texte pour chaque version de notre classification (basique, entités nommées et topic modeling), et ce, pour chaque classification. Pour chaque type de prétraitement, nous avons affiché l'accuracy correspondante. Les accuracy les plus élevées pour chaque type de classification ont été mises en bleu, et la colonne la plus prometteuse pour réaliser la classification avec les paramètres et prétraitements optimaux a été mise en vert.

Enfin, nous avons créé un notebook intitulé "Classifications finales" qui réalise les trois classifications avec les paramètres et les modèles qui ont donné les meilleurs résultats. De plus, nous avons affiché la classification report et les matrices de confusion des classifications finales présentant les meilleures performances.

Si l'accuracy est une mesure importante, d'autres mesures méritent également notre attention. Dans le contexte de la détection des fake news, le rappel revêt une importance particulière. En effet, si un article de fake news est classifié à tort comme véridique, cela pourrait entraîner des conséquences

graves, telles que la propagation de fausses informations et la désinformation du public. Par conséquent, il est essentiel que les modèles de détection de fake news présentent un rappel élevé pour garantir qu'ils identifient la majorité des fake news dans le jeu de données.

La précision est également importante dans ce contexte, car il est essentiel de ne pas classer les vraies nouvelles comme fake news (faux positifs). Une faible précision signifierait que certains articles légitimes sont classés comme fake news, ce qui pourrait nuire à la crédibilité des médias et semer la confusion parmi les lecteurs.

Les deux mesures sont importantes, cependant, le rappel est privilégié afin de minimiser le risque de laisser passer des fake news.

3.1 True vs False

TABLE 1 – Résultats des modèles de classification pour chaque méthode de prétraitement des documents

		TFIDF_lowcase	**TFIDF_lowstop**	**TFIDF_lowStopstem**	**TFIDF_brut**
Basique	Titre	88.8% (SVC C :1, gamma :'scale', Kernel :'rbf')	88.8% (SVC C :1, gamma :'scale', Kernel :'rbf')	89.3% (SVC C :1, gamma :1, Kernel :'rbf')	89.3% (SVC C :1, gamma :'scale', Kernel :'rbf')
	Texte	91.1% (SVC C :2, gamma :'scale', Kernel :'rbf')	91.7% (SVC C :1, gamma :'scale', Kernel :'rbf')	92.9% (SVC C :1, gamma :'scale', Kernel :'rbf')	89.9% (SVC C :2, gamma :'scale', Kernel :'rbf')
	Titre/Texte	88.2% (SVC C :5, gamma :'scale', Kernel :'rbf')	89.9% (SVC C :2, gamma :'scale', Kernel :'rbf')	91.1% (RF n_estimators :200, max_features :'sqrt')	90.5% (RF n_estimators :50, max_features :'sqrt')
Entités nommées	Titre	89.3% (SVC C :2, gamma :'scale', Kernel :'rbf')	86.4% (RF n_estimators :300, max_features :'sqrt')	87.6% (SVC C :2, gamma :'scale', Kernel :'rbf')	89.9% (RF n_estimators :300, max_features :'sqrt')
	Texte	92.9% (RF n_estimators :300, max_features :'sqrt')	91.7% (SVC C :2, gamma :'scale', Kernel :'rbf')	94.1% (SVC C :2, gamma :'scale', Kernel :'rbf')	92.3% (SVC C :2, gamma :'scale', Kernel :'rbf')
	Titre/Texte	91.1% (SVC C :5, gamma :'scale', Kernel :'rbf')	91.1% (RF n_estimators :100, max_features :'sqrt')	90.5% (SVC C :2, gamma :'scale', Kernel :'rbf')	89.9% (SVC C :2, gamma :'scale', Kernel :'rbf')
Topic Modeling	Titre et Keywords	87% (RF n_estimators :50, max_features :'sqrt')	88.2% (RF n_estimators :100, max_features :'sqrt')	89.3% (SVC C :7, gamma :'scale', Kernel :'rbf')	88.2% (SVC C :2, gamma :'scale', Kernel :'rbf')
	Texte et Keywords	92.3% (RF n_estimators :200, max_features :'sqrt')	90.5% (SVC C :2, gamma :'scale', Kernel :'rbf')	92.3% (SVC C :2, gamma :'scale', Kernel :'rbf')	91.1% (RF n_estimators :300, max_features :'sqrt')
	Titre/Texte et Keywords	88.8% (RF n_estimators :100, max_features :'log2')	89.3% (SVC C :2, gamma :'scale', Kernel :'rbf')	91.7% (RF n_estimators :50, max_features :'sqrt')	89.3% (RF n_estimators :50, max_features :'sqrt')

Pipeline	
Prétraitement	removestopwords=True,lowercase=True, getstemmer=True ,removedigit=False
TFIDF	lowercase=False
SVC	C : 2 , gamma : 'scale' , Kernel : 'rbf'

Accuracy : 0.941

	Précision	Recall	F1-score	Support
false	0.92941	0.95181	0.94048	83
true	0.95238	0.93023	0.94118	86

Accuracy			0.94083	169
Macro avg	0.94090	0.94102	0.94083	169
Weighted avg	0.94110	0.94083	0.94083	169

FIGURE 1 – Classification report sur True et False

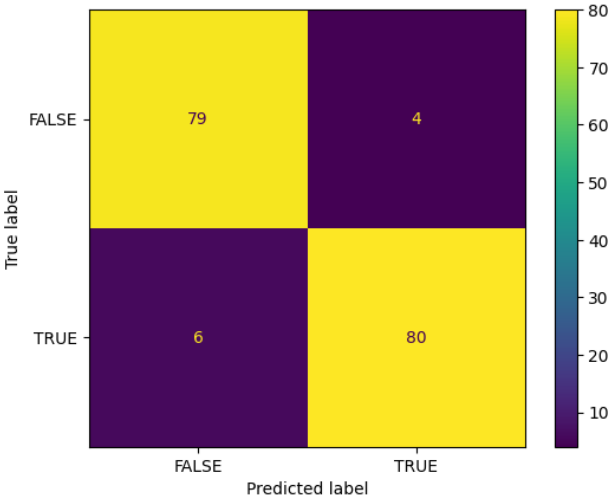


FIGURE 2 – Matrice de confusion

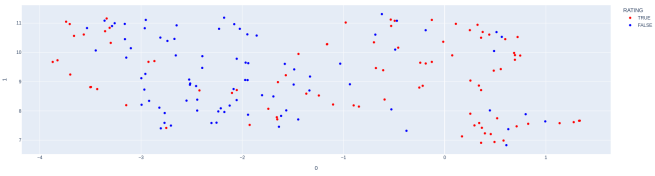


FIGURE 3 – Visualisation de ce qu'on était censés avoir

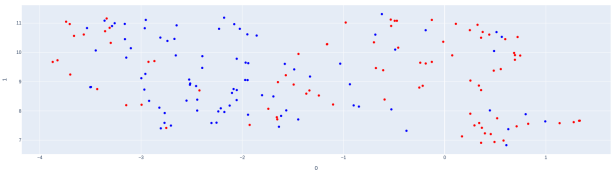


FIGURE 4 – Visualisation de ce qui a été prédit

Discussion des résultats : Dans cette classification, les prétraitements offrant les meilleurs résultats sont résumés dans le tableau situé au-dessus du classification report. Nous observons dans le premier tableau que, pour chaque type de classification (basique, avec entités nommées, etc.), la mise en minuscule (lowercase) des documents, la suppression des mots vides et la stemmatisation sont des prétraitements qui permettent d'obtenir de meilleurs résultats.

En effet, la mise en minuscule du texte permet d'uniformiser la casse des mots, évitant ainsi que des mots identiques soient considérés comme différents selon leur casse. Cela améliore la "term frequency", en accordant plus d'importance aux termes fréquemment utilisés. De plus, la suppression des stopwords, tels que "le", "la", "et", "de", "à", etc., réduit la dimensionnalité du vocabulaire et permet au modèle de se concentrer sur les mots les plus pertinents pour la classification.

Cependant, nous constatons aussi que le prétraitement consistant à supprimer les chiffres n'est pas très utile dans notre cas, car les pipelines incluant ce prétraitement n'offrent pas des résultats optimaux, étant donné que les chiffres n'ont pas une grande influence sur la classification des textes. Nous nous rendons également compte que la lemmatisation ne fournit pas des résultats optimaux. Nous nous sommes demandé pourquoi la stemmatisation était plus intéressante dans ce cas. Nous nous sommes rappelé que la stemmatisation permettait de regrouper les variantes d'un même mot, facilitant ainsi l'analyse et le traitement du texte. Par exemple, les mots "manger", "mangeant", "mangé" et "manges" ont la même racine "mang". Cela va augmenter notre term frequency. En appliquant la stemmatisation à ces mots, on peut les traiter comme étant le même mot. La lemmatisation, quant à elle, donne les racines exactes des mots, ce qui peut introduire plus de bruit dans les données que la stemmatisation, qui est une méthode plus simple et plus rapide pour réduire la dimensionnalité du vocabulaire. La stemmatisation peut être préférable, par exemple, si l'on veut simplement extraire des informations clés du texte sans se soucier des nuances grammaticales. Les meilleurs résultats sont obtenus grâce aux entités nommées pour cette classification, ce qui nous permet d'af-

firmer que les entités nommées en tant que caractéristiques sont utiles au modèle de classification. En effet, nous constatons que la meilleure accuracy pour la classification basique est de 92,9% et la meilleure précision obtenue grâce aux entités nommées est de 94,1%. Ainsi que le rappel et la précision sont de 94.1% pour les entités nommées et de 91.12% pour la classification basique. En comparant la matrice de confusion de la classification basique avec celle de la classification avec entités nommées, nous constatons moins de faux négatifs (2 de moins) et, par conséquent, plus de vrais positifs (2 de plus). Le topic modeling nous permet d'identifier les sujets les plus récurrents dans les articles FakeNews, mais l'ajout de ces mots-clés en tant que caractéristiques n'améliore pas l'accuracy dans notre cas. Nous remarquons que, dans les classifications offrant les meilleurs résultats, SVC est utilisé avec les hyper-paramètres suivants : C=2, gamma='scale' et kernel='rbf'. La valeur de C entraîne une amélioration de l'accuracy, mais peut également causer du surapprentissage si la valeur est trop élevée. Cette valeur de gamma semble être la mieux adaptée à notre modèle et à la distribution des données.
En ce qui concerne les visualisations, la figure 3 représente la distribution des données en fonction de leur labélisation réelle (y_test), tandis que la figure 4 montre comment notre classifieur les a classées (y_pred). Une légère différence (légèrement mauvaise classification) est observée entre les deux. En conséquence, notre modèle a montré une excellente performance sur l'ensemble de données de test.

3.2 True/False vs Other

Table 2 – Résultats des modèles de classification pour chaque méthode de prétraitement des documents

		TFIDF_lowcase	**CV_lowstop**	**TFIDF_lowStopstem**	**TFIDF_brut**
Basique	Titre	96.8% (SVC C :1, gamma :'scale', Kernel :'rbf')	96.8% (SVC C :1, gamma :'0.1', Kernel :'rbf')	96.8% (SVC C :1, gamma :'scale', Kernel :'rbf')	96.8% (SVC C :1, gamma :'scale', Kernel :'rbf')
	Texte	95.7% (RF n_estimators :100, max_features :'log2')	94.7% (SVC C :1, gamma :'scale', Kernel :'rbf')	94.7% (SVC C :2, gamma :'scale', Kernel :'rbf')	94.7% (SVC C :1, gamma :'scale', Kernel :'rbf')
	Titre/Texte	94.7% (SVC C :2, gamma :'scale', Kernel :'rbf')	96.8% (SVC C :1, gamma :0.1, Kernel :'rbf')	91.5% (RF n_estimators :300, max_features :'sqrt')	94.7% (SVC C :1, gamma :'scale', Kernel :'rbf')
Entités nommées	Titre	96.8% (SVC C :1, gamma :'scale', Kernel :'rbf')	96.8% (SVC C :1, gamma :'scale', Kernel :'rbf')	96.8% (SVC C :1, gamma :'scale', Kernel :'rbf')	96.8% (SVC C :5, gamma :'scale', Kernel :'rbf')
	Texte	93.6% (RF n_estimators :200, max_features :'sqrt')	93.8% (SVC C :2, gamma :0.7, Kernel :'rbf')	92.6% (RF n_estimators :200, max_features :'sqrt')	92.6% (RF n_estimators :300, max_features :'sqrt')
	Titre/Texte	93.6% (RF n_estimators :100, max_features :'log2')	97.9% (SVC C :1, gamma :'0.1', Kernel :'rbf')	89.4% (SVC C :2, gamma :'scale', Kernel :'rbf')	90.4% (SVC C :1, gamma :'scale', Kernel :'rbf')
Topic Modeling	Titre et Keywords	92.6% (SVC C :2, gamma :'scale', Kernel :'rbf')	93.6% (SVC C :1, gamma :0.5, Kernel :'rbf')	92.6% (SVC C :2, gamma :'scale', Kernel :'rbf')	89.4% (RF n_estimators :100, max_features :'log2')
	Texte et Keywords	92.6% (RF n_estimators :100, max_features :'sqrt')	93.6% (SVC C :1, gamma :0.1, Kernel :'rbf')	94.7% (RF n_estimators :300, max_features :'sqrt')	95.7% (RF n_estimators :100, max_features :'log2')
	Titre/Texte et Keywords	93.6% (RF n_estimators :300, max_features :'sqrt')	93.6% (SVC C :1, gamma :0.2, Kernel :'rbf')	92.6% (RF n_estimators :300, max_features :'sqrt')	93.6% (RF n_estimators :300, max_features :'sqrt')

Pipeline	
Prétraitement	removestopwords=True,lowercase=True ,getstemmer=False,removedigit=False
CV	lowercase=False
SVC	C : 1 , gamma : 0.1 , Kernel : 'rbf'

Accuracy : 0.979

	Précision	Recall	F1-score	Support
false	1.00000	0.95122	0.97500	41
true	0.96364	1.00000	0.98148	53

Accuracy			0.97872	94
Macro avg	0.98182	0.97561	0.97824	94
Weighted avg	0.97950	0.97872	0.97865	94

FIGURE 5 – Classification report sur True/False et Other

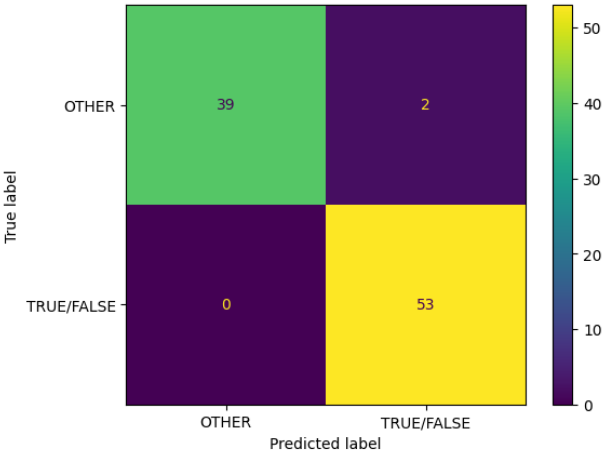


FIGURE 6 – Matrice de confusion

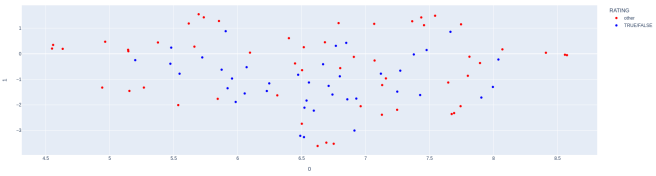


FIGURE 7 – Visualisation de ce qu'on était censés avoir

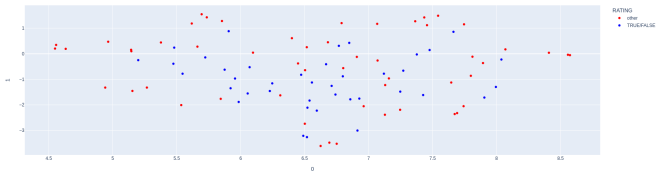


FIGURE 8 – Visualisation de ce qui a été prédit

Discussion des résultats : Nous avons remarqué que pour cette classification, la méthode de vectorisation Bag of Words donne les meilleurs résultats. Les paramètres de prétraitement qui ont donné les meilleurs résultats sont les mêmes que pour la classification précédente, à l'exception de la stemmatisation qui a été mise à faux cette fois-ci. Cela montre que la stemmatisation peut ne pas être efficace dans tous les cas, car elle peut supprimer des lettres importantes des mots et les rendre moins distincts, ce qui peut nuire à la performance du modèle de classification.

Nous avons constaté que les meilleurs résultats ont été obtenus grâce au modèle SVM avec les entités nommées sur le titre et le texte. Cela renforce l'idée que l'ajout des entités nommées en tant que features est utile pour la classification. La meilleure accuracy obtenue est de 97,9%, ce qui est supérieur à celle des classifications basiques (96,8%) et de la classification de topic modeling (95,7%). Le rappel est de 97.5% et la précision est de 98.18% pour les entités nommées et de 96.61% et de 96.89% respectivement pour la classification basique.
Nous avons également remarqué que l'accuracy est plus élevée que celle de la classification True vs False. Cela est cohérent avec le fait de regrouper les labels "vrai" et "faux" dans une seule classe "TRUE/FALSE" et de les différencier des autres labels dans la classe "OTHER".

Avec l'ajout des entités nommées, nous avons observé trois vrais positifs de plus pour la classe TRUE/FALSE, ce qui a réduit le nombre de faux négatifs de trois (des TRUE/FALSE qui ont été prédits comme OTHER dans la classification basique). Le classifieur qui donne la meilleure accuracy est SVC avec les hyper-paramètres suivants : C=1, gamma=0,1 et Kernel='rbf'.

En résumé, les résultats montrent que l'ajout d'entités nommées en tant que features peut améliorer la performance de la classification et que la méthode de vectorisation Bag of Words peut donner de bons résultats dans certains cas.

Concernant les graphiques, la figure 7 illustre la distribution des données selon leur étiquetage réel (y_test), tandis que la figure 8 montre comment notre classificateur les a classées (y_pred). Il y a une

petite différence (légèrement mauvaise classification) entre les deux. En conclusion, notre modèle a montré une très bonne performance sur le jeu de test.

3.3 True vs False vs Mixture vs Other

Comme nous l'avons expliqué précédemment, nous avons tenté de classifier les 4 classes en même temps, mais les accuracy étaient trop basses, car les classes Mixture et Other sont difficiles à séparer. Nous avons donc décidé de procéder à une double classification, la première sur True et False comme détaillé précédemment, et la deuxième sur Other et Mixture. Les résultats de cette deuxième classification sont fournis dans le tableau ci-dessous. Ensuite, nous avons concaténé les résultats des deux classifications.
Deuxième classification : Other vs Mixture

TABLE 3 – Résultats des modèles de classification pour chaque méthode de prétraitement des documents

		TFIDF_lowcase	**TFIDF_lowstop**	**TFIDF_lowStoptem**	**TFIDF_brut**
Basique	Titre	86.2% (SVC C :1, gamma :'scale', Kernel :'rbf')	85.1% (SVC C :1, gamma :'scale', Kernel :'rbf')	83% (SVC C :1, gamma :'scale', Kernel :'rbf')	86.2% (SVC C :1, gamma :'scale', Kernel :'rbf')
	Texte	84% (SVC C :1, gamma :'scale', Kernel :'rbf')	87.2% (SVC C :1, gamma :'scale', Kernel :'rbf')	86.2% (SVC C :2, gamma :'scale', Kernel :'rbf')	84% (SVC C :1, gamma :'scale', Kernel :'rbf')
	Titre/Texte	85.1% (SVC C :1, gamma :'scale', Kernel :'rbf')	87.2% (SVC C :1, gamma :'scale', KerneSl :'rbf')	87.2% (SVC C :1, gamma :'scale', Kernel :'rbf')	85.1% (SVC C :1, gamma :'scale', Kernel :'rbf')
Entités nommées	Titre	92.6% (SVC C :1, gamma :'scale', Kernel :'rbf')	91.5% (SVC C :1, gamma :'scale', Kernel :'rbf')	92.6% (SVC C :1, gamma :'scale', Kernel :'rbf')	92.6% (SVC C :1, gamma :'scale', Kernel :'rbf')
	Texte	91.5% (SVC C :1, gamma :'scale', Kernel :'rbf')	92.6% (SVC C :1, gamma :'scale', Kernel :'rbf')	92.6% (SVC C :1, gamma :'scale', Kernel :'rbf')	92.6% (SVC C :1, gamma :'scale', Kernel :'rbf')
	Titre/Texte	91.5% (SVC C :1, gamma :'scale', Kernel :'rbf')	92.6% (SVC C :2, gamma :'scale', Kernel :'rbf')	92.6% (SVC C :1, gamma :'scale', Kernel :'rbf')	91.5% (SVC C :1, gamma :'scale', Kernel :'rbf')
Topic Modelling	Titre avec Keywords	62.8% (SVC C :2, gamma :'scale', Kernel :'rbf')	60.6% (SVC C :2, gamma :'scale', Kernel :'rbf')	62.2% (SVC C :2, gamma :'scale', Kernel :'rbf')	62.2% (SVC C :1, gamma :'scale', Kernel :'rbf')
	Texte avec Keywords	73.4% (RF n_estimators :300, max_features :'sqrt')	72.3% (SVC C :1, gamma :'scale', Kernel :'rbf')	73.4% (SVC C :2, gamma :'scale', Kernel :'rbf')	71.8% (SVC C :2, gamma :'scale', Kernel :'rbf')
	Titre/Texte avec Keywords	72.9% (SVC C :2, gamma :'scale', Kernel :'rbf')	72.9% (SVC C :2, gamma :'scale', Kernel :'rbf')	71.8% (SVC C :1, gamma :'scale', Kernel :'rbf')	73.9% (SVC C :2, gamma :'scale', Kernel :'rbf')

Concaténation des résultats :

Pipeline	
Prétraitement	removestopwords=True,lowercase=True ,getstemmer=True ,removedigit=False
TFIDF	lowercase=False
SVC	C : 2 , gamma : 'scale' , Kernel : 'rbf'

Accuracy : 0.920

	Précision	Recall	F1-score	Support
false	0.92941	0.95181	0.94048	83
true	0.95238	0.93023	0.94118	86
mixture	0.82222	0.92500	0.87059	40
other	0.93878	0.85185	0.89320	54
Accuracy			0.922015	263
Macro avg	0.91070	0.91472	0.91136	263
Weighted avg	0.92254	0.92015	0.92037	263

FIGURE 9 – Classification report sur les 4 classes

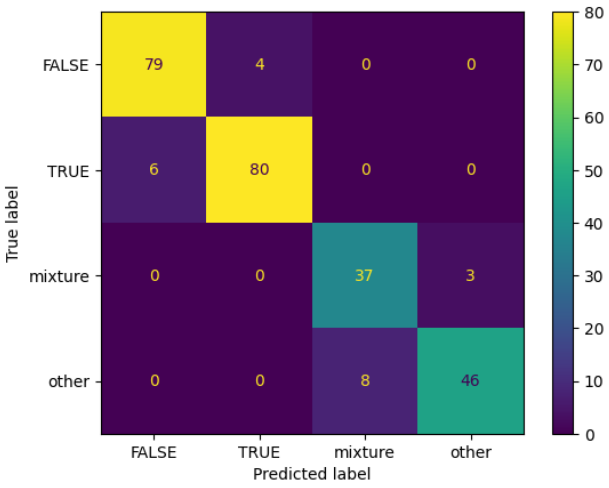


FIGURE 10 – Matrice de confusion

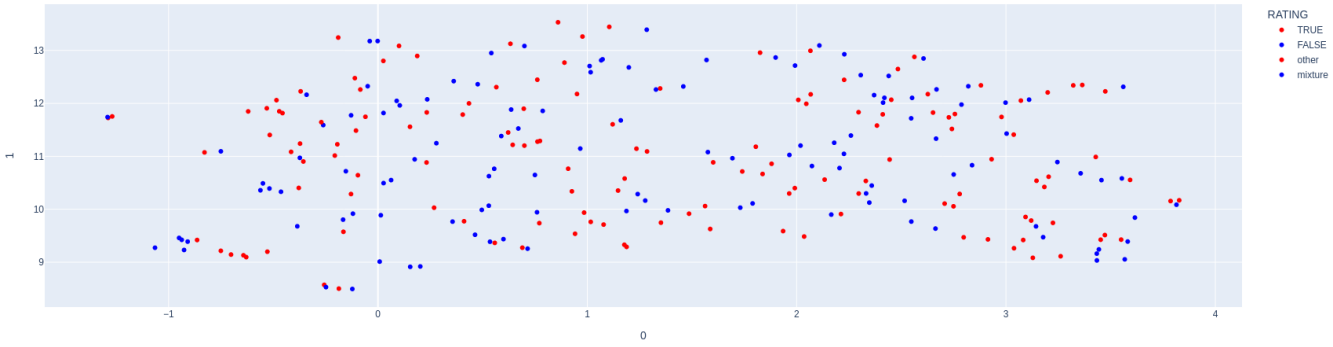


FIGURE 11 – Visualisation de ce qu'on était censés avoir

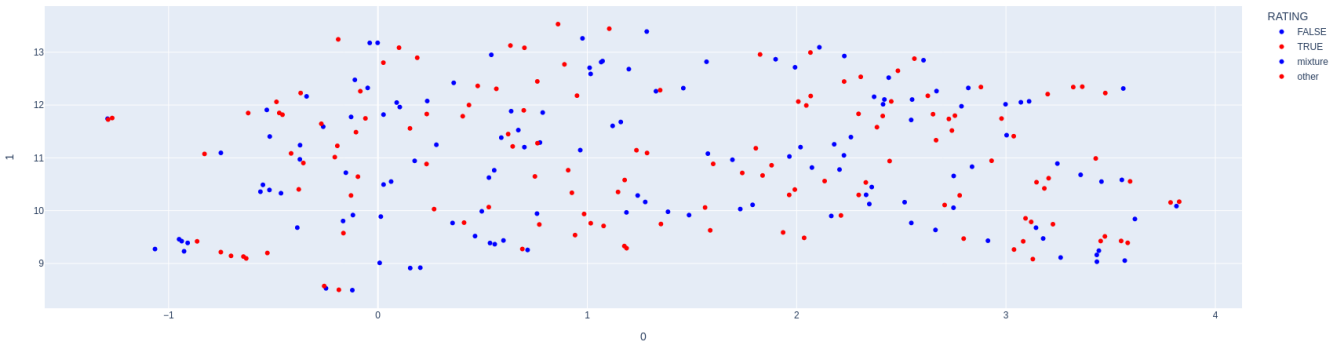


FIGURE 12 – Visualisation de ce qui a été prédit

Discussion des résultats :
Pour cette classification, la meilleure accuracy est donnée par la méthode TF_IDF avec les prétraitements suivants qui donnent un résultat optimal : la suppression des mots vides, la suppression des majuscules et la stemmatisation. Dans un souci de ne pas engendrer de redondances dans notre rapport, nous ne mentionnons pas les explications à l'optimisation de ces paramètres dans la mesure où nous les avons déjà expliquées dans la section True vs False.

Les meilleurs résultats pour la classification True vs False et la classification Other vs Mixture sont obtenus avec l'utilisation des entités nommées. Nous pouvons constater d'après le tableau de résultats des modèles de classification que la meilleure accuracy avec les entités nommées pour Other/-Mixture est de 92,6%, tandis que celle des classifications basiques est de 87,2%.

Pour la classification finale après concaténation sans entités nommées, nous avons obtenu une accuracy de 90,9%,un rappel de 93,6% et une précision de 93,2%.

En ce qui concerne la classification finale avec les entités nommées, les résultats sont fournis dans la figure 9. En moyenne, l'accuracy est de 92%, la précision est de 91% et le rappel est de 91,4%.

En comparant la matrice de confusion de la classification basique et celle où nous ajoutons les entités nommées, nous constatons que nous avons plus de vrais positifs dans la dernière (2 de plus), et par conséquent, moins de Faux négatifs pour "True vs False" et pour "Mixture vs Other".

En ce qui concerne les visualisations, la figure 11 montre la distribution des données selon leur vraie labélisations (y_test). Et sur la figure 12, nous voyons comment notre classifieur les a classées (y_pred). Nous remarquons une légère différence. Nous pouvons en conclure que notre modèle était très bon sur le jeu de test.

4 Conclusion

Au cours de ces trois classifications, nous avons obtenu des résultats optimaux en nous familiarisant avec les paramètres de prétraitement et les hyper-paramètres testés. Nous avons constaté que les paramètres de prétraitement et les hyper-paramètres ont un impact important sur la classification, car en les modulant, nous avons pu obtenir de meilleurs résultats. Nous avons remarqué que le modèle SVC permettait d'obtenir les meilleurs résultats pour chacune des classifications. Il est particulièrement utile lorsque les données sont linéairement séparables, ce qui signifie que les deux classes peuvent être séparées par une ligne droite ou un hyperplan. Cela explique son efficacité pour la prédiction de classification binaire selon true et false.
En outre , cela nous a également permis d'évaluer certaines méthodes en fonction de leur contribution à la classification. Nous avons constaté que l'ajout d'entités nommées dans les features de la classification a augmenté l'accuracy, le rappel, la précision et le nombre de prédictions correctes (vrais négatifs et vrais positifs). Cela s'explique par le fait que les entités nommées permettent d'identifier les noms de personnes, de lieux et d'organisations mentionnées dans un article de presse, ce qui peut aider à augmenter le term frequency.
Enfin, nous avons utilisé le topic modeling pour identifier les sujets les plus fréquents des articles de Fake News. Nous avons constaté que les sujets les plus récurrents étaient, par exemple : "climate change", "virus, test, pcr" (covid), "presidential election, vote", "high education, teaching, graduate earning". Bien que le topic modeling n'ait pas amélioré la performance du modèle, il a été utile pour identifier les sujets les plus fréquents dans les articles de Fake News.