Master 1 Informatique - Machine Learning 1
Détection de fausses nouvelles

AMAH Gnimdou Richard Numéro d'étudiant : 22216331
METE Bursa
BEN AMOR Oumayma
LOUDAGH Ali
LAOUINI Yasmine
Encadrants :

Pr. Pascal PONCELET
Pr. Konstantin TODOROV

1 Mai 2023

Table des matières

1 Introduction

L'objectif de ce projet qui conclut le module HAI817I est d'utiliser les méthodes d'apprentissage automatiques classiques, surtout celles de l'apprentissages supervisé pour classer des assertions (une assertion est une proposition que l'on avance et soutient comme vraie), et ensuite de trouver les modelès les plus performants pour déterminers si certaines assertions comme des données test sont vraies ou fausses. Notre jeu de données est ClaimsKG, collecté par sur sites de fact-checking par le LIRMM en collaboration avec plusieurs équipes de recheche européennes. Nous avons procédé par une visualisation de ces données pour basées notre approche face au problème de classification posé.

2 Préparation, prétraitements et ingénierie des données

2.1 Proportions des jeux de données

Les jeux de données qui nous ont été donnés pour le projet ont déjà été séparés en deux, un des jeux pour l'entraînement des modèles et l'autre pour l'evaluation et l'analyse de nos classifieurs, contrairement au cas classique où souvent, on a qu'un seul jeu de données que nous devrions nous-même répartir en un ensemble test et d'entrainement.
La taille de nos données d'entraînement étant très petite, elle presente quand une très grande disparité entre les classes présentes. Pour évitez les erreur liées aux proportions inégales des données de chaque classe, nous avons réquilibré ces classes et utilisé l'entièreté de ces nouvelles proportions.

2.2 Equilibrage des classes

En visualisant une première fois les données, initialement avant tout traitement et ingénierie textuelle, nous avons remarqué comme dit plus haut, une disproportion très marquèe entre les classes (ici, le type de "our rating"). On remarque facilement que les classes "true" et "other" sont les moins représentées.

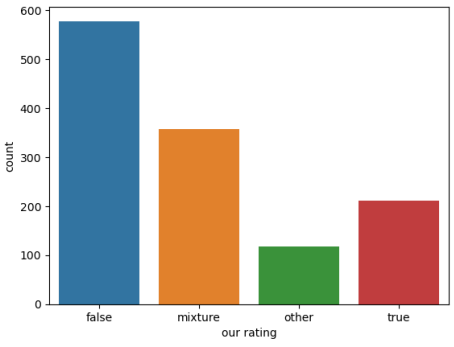

FIGURE 1 – Proportions brutes du jeu d'entrainement

Ce qui se traduit par certains écarts que nous obtenons par exemple au niveau du rappel des modèles de la classification Vrai vs Faux (on y reviendra un peu plus bas). On a choisi le down-sampling pour réequilibrer nos données pour tous les trois types de classification. Mais cette approche nous mène à une importante réduction de la quantité finale de données réellement utilisée pour l'entrainement de nos modèles.

2.3 Ingénierie textuelle

Generalement, les algorithmes de classification en Machine Learning se basent sur des fonctions mathématiques ayant comme des valeurs d'entrée des valeurs numeriques. Le probleme de la classification du texte réside dans la nature meme de ce texte utilisé comme donnée d'entrée des algorithmes de classification. Pour passer des valeurs textuelles en valeur numériques tout en ne perdant pas les informations importantes présentes, nous avons utilisé des techniques de traitement de données textuelles vues en cours. Nous avons choisi des approches classiques en NLP, telles que la tokénisation des phrases, la conversion en lettres minuscules de notre corpus, la suppression des stopwords (mots très communs et qui n'apportent rien lors de la phase d'apprentissage), nous avons supprimé les nombres et utilisé la méthode de stemming, donc de racinisation des mots. Concrètement, nous avons procédé à de diverses combinaison de ces pré-traitement que nous avons par la suite tester sur chacun de nos classifieurs.
Mais avant de tester ces combinaisons de pré-traitement, une étape importante était la transformation de nos texte pré-traités en valeur numériques comme sus-mentionné. Une méthode classique est de calculer le TF-IDF qui transforme un texte d'un corpus en un vecteur où chaque entrée est un nombre entre 0 et 1 correspondant à un mot du corpus. Nous avions ensuite plusieurs approches possibles, l'approche N-Gram qui consiste à considérer non plus un seul mot dans la vectorisation mais des paquets de N mots. Il faut quand meme noter que cette méthode reste encore peu éfficace.

3 Développement et Evaluation des modèles

En machine learning il existe de nombreux algorithmes de classification pour différents problèmes de classification (comme des classifications multiclasses, binaire, etc...) et différentes type des données (données labelisées ou non labelisées). Dans notre cas, nous travaillons avec des données labelisées. De cet fait, nos choix se sont portes sur un certain nombre de classifieurs.

3.1 Validation croisée des modèles

Nous avons choisi 7 classifieurs traditionnels. L'étape suivante consistait à tester les différentes combinaisons de pré-traitement en vue de trouver la meilleure propre à chaque classifieur pour obtenir des niveaux satisfaisants de précision. Celle-ci est ensuite utilisée pour effectuer la validation croisée K-fold du modèle.

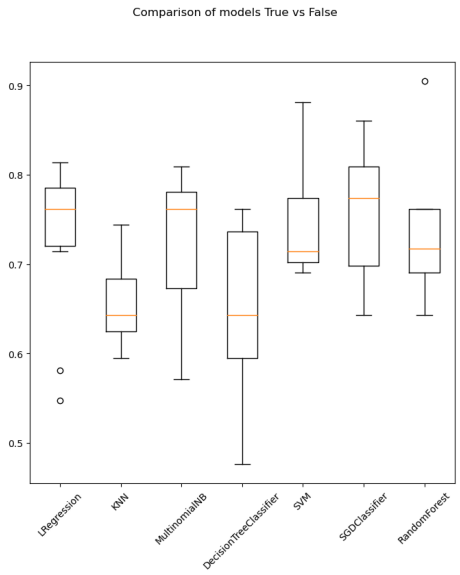

FIGURE 2 – Comparaison des modèles pour la classification Vrai vs Faux

La validation croisée nous permet de rendre moins ”biaisé” les évaluation de nos modèles. Nous avons preferé une validation croisée avec K=10 et de choisir les meilleurs modèles pour rechecher les hyper-paramètres optimaux pour ces modèles.

3.2 Recherche des meilleurs hyper-paramètres : Grid Search

Après avoir ”cross-validé” l'ensemble des classifieur que nous avons choisi au départ, nous avons sélectionné ceux qui ont une tendance à avoir une meilleur accuracy globalement pour les trois type de classifications, parmi lesquels la Regression Logistique, le Multinominal Naive Bayes, le RandomForest. Nous fixons un pré-traitement au fur et à mésure car la recherche des meilleur pré-traitement directement dans le pipeline dans cette phase, la rend quasiment interminable. Mais quand nous même avons fait cette phase juste avant de rechercher les paramètres des modèles en question. Notre pipeline normalise ensuite les textes pré-traités par le TD-IDF puis éffectue la recheche proprement dite. Notre critère de choix des meilleurs paramètres est l'accuracy. Le revers de la médaille avec ce critère est que nous avons obtenu des modèles qui sont très précis à l'apprentissage, mais peu précis avec de nouvelles données de test.

3.3 Evaluation des modèles sur le jeu de donnée test

Habituellement, pour mesurer les performances sur un jeu de données Test, un moyen est d'analyser les proportions obtenues après prédiction sur ces données par la matices de confusion. Ci-dessous nous avons la matrice de confusion du meilleur modèle de la regression logistique obtenu après la recherche de nos hyper-paramètres. Elle indique que ce modèle est un peu trop ”erroné” sur les données test.

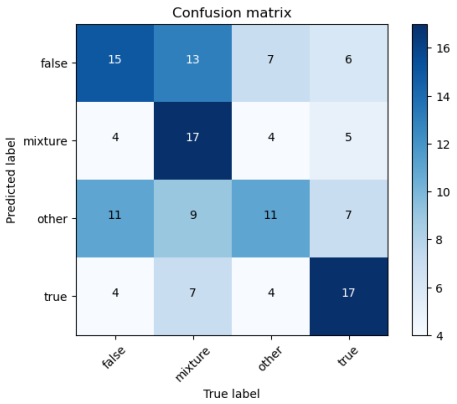

FIGURE 3 – Matrice de confusion de la classification Vrai vs Faux vs Mixture vs Other

En calculant l'erreur par rapport à la taille des données, nous pouvons constater que le modèle surapprend (Erreur presque nulle) et que sur les données tests, il ressort une différence bien marquée. Ceci est en partie dû à la quantité de données utilisée pour l'entrainement du modéle comme nous l'évoquions au début. Très marquant aussi dans cette classification, les classes "other" et "mixture" présentent les pires scores (accuracy, f1-measure, recall), différence qui peut s'expliquer par la très différente disparité entre les classes dans le jeu train par rapport au jeu Test.

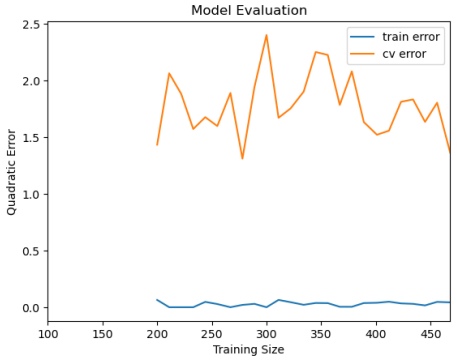

FIGURE 4 – Courbe d'apprentissage du modèle multinominal Naive Bayes

4 Conclusion

Ce projet nous a permis de d'abord comprendre les concepts de base du Machine learning, comment aborder les problèmes qui y sont posés. L'apprentissage automatique étant une intersection de plusieurs disciplines, ce projet, nous a aussi permis de mettre en application nos différents acquis venant d'autres modules. L'ensemble des

membres de notre groupe reste persuadé que ce projet nous a apporté un peu plus de connaissance sur le domaine de l'apprentissage automatique en particulier et de l'intelligence artificiel en général.