Prédiction de votre type MBTI à l'aide des données texte

Rapport du projet

Hoang Phong Vu TRANG Nahel KHELLAF

1 Introduction

1.1 Myers-Briggs Type Indicator

Le **Myers-Briggs Type Indicator** (MBTI) est un des indicateurs les plus utilisés pour déterminer la personnalité d'une personne. Cet outil a été inventé par Katharine Cook Briggs et sa fille Isabel Briggs Myers, basé sur la théorie de Carl Jung sur des types psychologiques.

Aujourd'hui, l'outil est souvent utilisé par Les particuliers pour mieux comprendre leurs propres personnalités ou par les organisations pour pouvoir améliorer son environnement de travail. L'indeicateur est basé sur quatre catégories binaires:

Table 1: Les quatre catégories du MBTI

D'où puisons-nous notre énergie ?	
Extraversion	**I**ntroversion
Comment recueillons-nous et utilisons-nous l'information ?	
Sensation	I**N**tuition
Comment choisissons-nous nos critères de décision ?	
Thinking	**F**eeling
Comment gérons-nous notre temps et notre espace ?	
Jugement	**P**erception

La personnalité d'une personne est caractérisée par une suite de quatre lettres, une par catégorie parmi les catégories présentées ci-dessus. Par exemple, pour quelqu'un dont le type est **ENFP**, il est donc extravertie, intuitive, sentimentale et perceptive. C'est la personne qui arrive toujours à trouver une raison pour sourire ou donner le sourire. Ce type correspond à des personnes qui aiment entreprendre et créer.

1.2 Proposition

Notre projet a pour but d'exploiter des procédés et algorithmes de Machine Learning pour évaluer le type MBTI d'une personne. En d'autres termes, l'utilisateur n'a qu'à fournir au programme des messages textuels qu'il a écrit pour que ce programme prédise son type MBTI. Cela lui évite de passer par de longs questionnaires.

Par ailleurs, dans le monde où l'utilisation des réseaux sociaux est en croissance exponentielle, nous souhaitons aussi savoir s'il y a une relation entre la façon dont une personne s'exprime en ligne et sa personnalité. Si une telle relation existe, on pourrait se dire que son style d'écriture sur ces réseaux sociaux correspond à sa personnalité. Ainsi, on pourrait dire que le comportement d'une personne sur les réseaux sociaux est le même que son comportement dans sa vie quotidienne. S'il n'existe pas de telle relation, on pourrait réfuter cette hypothèse.

1.3 Données

Les données que nous avons utilisées sont récupérées depuis *(MBTI) Myers-Briggs Personality Type Dataset* du Kaggle. Il s'agit des données collectées sur le forum *PersonalityCafe forum* puisqu'il fournit clairement les types MBTI des personnes, ainsi que leurs styles d'expression. Elles contiennent 8675 lignes, chaque ligne correspond à une personne :

- son type MBTI
- ses 50 derniers commentaires, séparés par le symbole "|||"

Ci-dessous quelques exemples de données :

Table 2: Quelques exemples de données

	type	posts
0	INFJ	'http://www.youtube.com/watch?v=qsXHcwe3krw|||...
1	ENTP	'I'm finding the lack of me in these posts ver...
2	INTP	'Good one _____ https://www.youtube.com/wat...
3	INTJ	'Dear INTP, I enjoyed our conversation the o...
4	ENTJ	'You're fired.|||That's another silly misconce...

2 Pipeline

2.1 Exploration de données

À première vue, les 4 catégories binaires nous donnent en totalité 16 possibilités, et Elles ne sont pas uniformément distribuées.

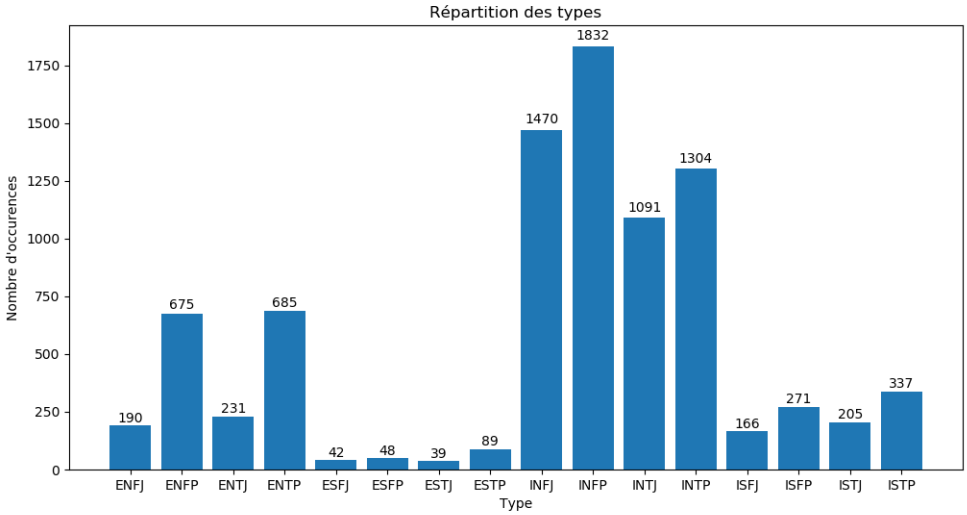

Figure 1: La répartition des types

Ici, nous pouvons observer que l'étiquette ayant le plus d'occurrences, INFP, apparaît 1832 fois, tandis que l'étiquette ESTJ n'apparaît que 39 fois. Il y a donc un déséquilibre des catégories au sein des données. Dans les parties suivantes nous allons discuter sur comment les traiter.

Par ailleurs, 80% des données sont utilisées pour le jeu d'entraînement et 20% pour le jeu de test.

2.2 Définition de la tâche, et choix d'une métrique de performance

Nous avons décidé de séparer les types en 4 catégories binaires différentes : E-I, S-N, T-F et J-P. Pour chaque catégorie, 0 correspond à la première lettre et 1 correspond à la deuxième. Il s'agit donc d'un problème de classification multi-label (ou la classification multi-étiquettes) binaire, c'est-à-dire, le domaine de définition de la sortie est $\{0, 1\}^4$.

Nous trouvons que la métrique 'classique' **accuracy** est simple pour interpréter. Néanmoins, face au déséquilibre des données, ce n'est pas une bonne idée de l'utiliser. Deux métriques, **F1-score** et **AUC** (pour Area Under Curve), seront donc prises en compte pour mésurer la performance du modèle. L'AUC présente cependant des inconvénients dans le cadre de notre projet, nous en discuterons plus concrètement dans la partie de l'entraînement du rapport.

2.3 Preprocessing

Avant de réaliser l'entrainement, nous avons besoin de traiter les textes bruts des données. Comme il s'agit de commentaires en ligne, ces textes contiennent des émojis, des liens http, etc.

Nous avons du prendre pas mal de temps pour les traiter.

Ce que nous avons réalisé:

- convertir en minuscules les majuscules
- convertir les émojis et les liens http en tokens spécifiques
- éliminer les symboles "|||"
- éliminer les ponctuations
- lemmatiser les textes

Du fait que l'apprentissage automatique ne peut se faire sur les textes directs mais sur plusieurs *variables d'entrée* (features), nous avons donc utilisé le modèle *Bag-of-words*. Dans ce modèle, on génère un dictionnaire de mots (bag) à partir des textes, des fréquences d'apparition sont attribuées à chaque instance du dictionnaire.

Une conversion de ces instances en tokens est effectuée dans le but de prendre en compte leur nombre d'occurrences dans le texte. Nous avons également choisi de tenir compte des émojis et des liens http car nous pensons que leurs fréquences d'utilisation sont liées à la personnalité d'une personne.

2.4 Optimisations

2.4.1 Support Vector Machines (SVM)

Support Vector Machine (SVM) est un modèle supervisé efficace et robuste. Cet algorithme permet de trouver un hyperplan dans un espace à N-dimensions qui sépare les classes. Pour les séparer, on peut choisir plusieurs hyperplans possibles. L'objectif est de maximiser la marge (*margin*, la distance entre l'hyperplan et les points qui en sont plus proche).

Formellement, avec un ensemble d'apprentissage $(\vec{x_n}, y_n)_{n=1}^{N}$ de taille N, avec $\vec{x_n} \in \{-1, 1\}$, on cherche à trouver $\vec{w}, b$ tels que:

$$minimiser \qquad \frac{1}{2}\|\vec{w}\|^2 + C\sum_{i=n}^{N}\zeta_i \tag{1a}$$

$$\text{sous les contraintes } \forall n, y_n(\vec{w}\cdot\vec{x_n} + b) >= 1 - \zeta_n \tag{1b}$$

$$\forall n, \qquad \zeta_n >= 0 \tag{1c}$$

Les *slack variables* ζ_n permet d'autoriser que les points soient mal classés. Pour les points correctement classifiés ou sur la marge $\zeta_n = 0$, les points situés dans la marge ont $0 < \zeta_n \leq 1$, et les points mal classés ont $\zeta_n > 1$.

Pour résoudre plus efficacement le problème, nous avons utilisé un modèle équivalent mais plus efficace que SVM, le modèle *Stochastic gradient descent* (SGD) avec la fonction de perte *hinge loss* (ce qui donne la marge maximale).

2.4.2 Choix d'hyper-paramètres

Nous avons choisi trois hyper-paramètres importants pour la performance de l'algorithme: la pénalité, l'alpha, le nombre d'itérations (*epochs*).

La pénalité désigne la norme utilisée dans la pénalisation, où 'L2' est le standard du modèle SVC (SVM pour des problèmes de classification). 'L1' est comme 'L2' ,mais la différence principale est que 'L1' permet d'éliminer les variables d'entrée (*feature*) les moins importantes, ainsi que de supprimer certains *features*. Elle fonctionne comme la technique *Feature Selection*. A priori, le modèle Bag-of-words nous donne en résultat un grand nombre de features, nous pensons que 'L1' donne de meilleurs résultats en termes de performances car nous n'effectuons pas *Feature Selection* précédémment.

Le paramètre 'alpha' dans le modèle *SGD* est inversement proportionnel au paramètre C dans le modèle *SVM* si-dessus. Enfin, le nombre d'itérations désigne le nombre de passes effectuées sur le jeu d'entraînement.

Nous avons ensuite essayé d'optimiser les hyper-paramètres grâce à la *Cross-Validation*, ci-dessus les hyper-paramètres optimisés que nous avons trouvés:

Table 3: Les hyper-paramètres optimisés

alpha	max_iter	penalty
0.0001	100	l1

2.4.3 Test

Avec les hyperparamètres optimisés obtenus à la phase précédente, nous effectuons la mesure de la performance sur le jeu de test.

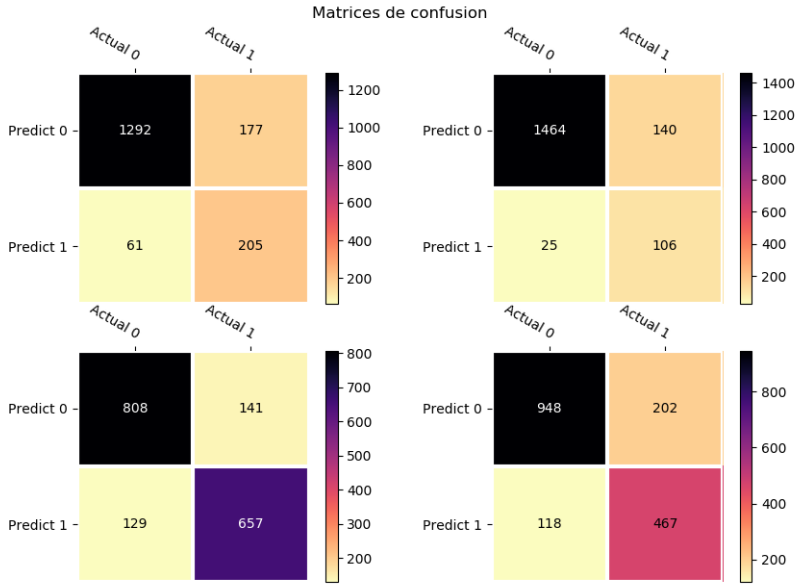

Figure 2: Les matrices de confusion des quatre catégories, en haut à gauche la catégorie E-I, à droite S-N, en bas à gauche la catégorie T-F, à droite J-P

En observant les matrices de confusion pour chaque catégorie, nous trouvons qu'il y a une grande différence entre le nombre de Positive et le nombre de Negative dans chaque matrice, notamment celles des catégories E-I et S-N. Cette différence implique une difficulté pour déterminer le seuil de décision, ce n'est donc pas évident d'utiliser AUC comme métrique. On utilise donc F1-Score.

Table 4: les F1-Score sur les 4 catégories

Catégorie	F1-score
E-I	0.63
S-N	0.56
T-F	0.82
J-P	0.74

Pour ce modèle, nous utilisons le mode de calcul micro-average pour F1-score. Contrairement au mode macro-average où la moyenne de F1-score de 4 catégories sont pris en compte, dans le mode micro-average, on calcule en utilisant le nombre total de TP, TN, FP, FN. Comme ça, la déséquilibre entre des classes est prise en charge. Nous trouvons donc un F1-score de **0.74**

2.5 Conclusion

Etant donné le F1-Score de 0.74 pour notre programme, la performance de ce dernier n'est pas très élevée. Cela pourrait s'expliquer par certains facteurs. Les données exploitées proviennent du forum PersonalityCafe spécialisé dans les types MBTI. Il aurait été plus intéressant de collecter et d'exploiter des données provenant d'une population plus grande et dont les sujets abordés seraient plus nombreux et plus vastes. D'autre part, le MBTI est un indicateur auto-évalué, il existe certainement des erreurs lorsqu'on effectue les tests. Par ailleurs, la performance faible peut également s'expliquer par l'inconhérence entre le style de langage utilisé en ligne par une personne et sa personnalité.

Nous pourrions améliorer la performance du modèle en extrayant des features qui nous semblent importantes comme les caractéristiques des émojis (content ou pas content, etc.), les caractéristiques des liens http (vidéo ou nouvelles, etc), etc.. Nous pourrions également utiliser les modèles Deep Learning de NLP (Natural Language Processing) pour l'améliorer.