Prédiction du sexe d'un individu à partir de ses données personnelles

Matthieu Bricaire - matthieu.bricaire@ensae.fr - Code associé au rapport

1 Présentation du problème

Dans le cadre du projet Socface, une grande quantité de documents de recensements français doivent être exploités et analysés. Pour ce faire, les données textuelles présentes sur ces archives manuscrites sont extraites et structurées grâce à un modèle de reconnaissance automatique (YOLO). Une difficulté majeure associée à cette tâche réside dans le fait que les formats tabulaires des recensements varient en fonction de la date à laquelle ils ont été menés. En particulier, dans certains recensements, le sexe des individus n'apparaît pas, alors qu'il s'agit d'une information cruciale pour l'analyse des ménages et le suivi des personnes correspondantes. Ainsi, à partir de données labellisées, nous nous intéressons à la prédiction du sexe d'un individu à partir des informations personnelles disponibles à son sujet. Un tel outil de prédiction pourrait ensuite être inclus dans le pipeline global de traitement des recensements.

En plus des problématiques inhérentes au traitement du langage naturel, cette tâche de prédiction présente plusieurs difficultés supplémentaires. D'une part, lors de la retranscription des archives manuscrites, le modèle de reconnaissance automatique commet des erreurs (caractères textuels incorrectement reconnus, informations mal classifiées), ce qui introduit du bruit dans les données. D'autre part, la prédiction du sexe concerne un nombre conséquent d'individus, et ne peut donc pas être réalisée manuellement par un humain. Enfin, dans le cadre de notre étude, nous disposons d'un petit jeu de données, extrait de la base principale. Il s'agit donc de développer des méthodes qui soient à la fois performantes, robustes aux données bruitées et *scalables* à de gros jeux de données ; tout en étant peu gourmandes en données d'entraînement ainsi qu'en ressources numériques.

2 Description des données et *preprocessing* général

Dans le cadre de notre étude, nous disposons de deux base de données séparées. Il s'agit donc de les combiner et de les mettre en forme de sorte à les rendre exploitables par les méthodes qui nous intéressent.

2.1 Première base : transcriptions automatiques et sexe associé

La première base de données mise à notre disposition se nomme *transcriptions_with_sex*, et constitue notre table principale. Elle contient, pour 241 individus, leurs données personnelles, ainsi que leur sexe. Plus précisément, pour chaque individu, nous disposons des informations suivantes :

— une transcription *groundtruth* réalisée manuellement par un humain (variable *groundtruth*)

— la transcription réalisée par le modèle de reconnaissance automatique (variable *prediction*)

— le sexe de l'individu (variable *sex*)

Dans l'optique d'intégrer l'outil de prédiction du sexe dans le pipeline de traitement automatique des archives de recensement, nous ne nous intéressons qu'aux variables *prediction* et *sex*.

D'une part, la variable *prediction* contient, pour chaque individu, une chaîne de caractères qui regroupe les informations personnelles disponibles à son sujet. Chaque chaîne de caractères est de la forme : « nom : Dupont prénom : Jean Paul date_naissance : 12 relation : fils profession : plombier ». A l'aide d'un expression régulière, nous extrayons les 9 types d'informations disponibles, que nous stockons dans autant de nouvelles colonnes de la base de données. Parmi ces différentes variables, plusieurs sont non informatives, ou ont trop de valeurs manquantes pour être utilisées avec pertinence par un modèle de *Machine Learning*. Ainsi, nous abandonnons les colonnes suivantes :

— **'nom'** : non informatif par définition

— **'date_naissance'** : mélange de dates de naissance et d'âges sans possibilité de convertir les dates en âge (et réciproquement), nombreuses valeurs aberrantes

— **'éducation'** : 238 NaN sur 241 individus

— **'état_civil'** : 205 NaN sur 241 individus, et valeurs aberrantes ou non informatives pour le reste

— **'employeur'** : 209 NaN sur 241 individus, et valeurs aberrantes ou non informatives pour le reste

— **'lieux_naissance'** : non informatif par définition

A l'inverse, plusieurs colonnes semblent particulièrement pertinentes dans le cadre de la prédiction du sexe, et sont donc retenues pour l'étude :

— **'prénom'** : 0 NaN, différences claires entre les prénoms masculins et féminins (dans la plupart des cas)

— **'relation'** : 88 NaNs, mais potentiel informatif (mots masculins et féminins à détecter)

— **'profession'** : 183 NaNs, mais potentiel informatif (mots masculins et féminins à détecter)

D'autre part, la variable *sex* contient trois modalités : « homme », « femme », et « ambigu » lorsque le sexe de l'individu ne peut être déterminé de manière certaine à partir de ses informations personnelles. Nous abandonnons les échantillons pour lesquels le sexe est ambigu. En effet, ils sont peu nombreux (9), et si un humain n'est pas capable de déterminer leur sexe avec précision, un modèle de *Machine Learning* n'y arrivera pas non plus. Ainsi, nous ne conservons que les 232 échantillons pour lesquels le sexe est clairement déterminé, et nous nous restreignons donc à un problème de classification binaire.

2.2 Seconde base : occurrences des prénoms en fonction du sexe

La seconde base de données dont nous disposons se nomme *firstname_with_sex*. Elle contient, pour 6946 prénoms (variable *firstname*), le nombre de fois où ils ont été observés chez des hommes (variable *male*) et des femmes (variable *female*). A partir des colonnes *male* et *female*, nous construisons la colonne *male_freq*, qui indique le pourcentage d'occurrence masculine de chaque prénom présent dans la base. Nous intégrons ensuite la variable *male_freq* à la table principale, en réalisant une fusion sur la colonne des prénoms (variable *firstname*).

A l'issue de cette fusion, nous observons que parmi les 110 prénoms présents dans la table principale, 35 ne figurent pas dans la base *firstname_with_sex*, et n'ont donc pas de fréquence d'occurrence masculine associée. Parmi ces 35 prénoms, il y a quelques prénoms reconnaissables pour un humain mais incorrectement restitués par le modèle de reconnaissance automatique, des valeurs aberrantes (mots qui n'ont rien à voir avec des prénoms), mais également plusieurs vrais prénoms, qui ne figurent tout simplement pas dans la table *firstname_with_sex* (non exhaustive). Lorsque cela sera nécessaire, nous procéderons à une imputation par la moyenne des valeurs manquantes de la variable *male_freq*.

3 Modèles

A l'issue de la phase de *preprocessing* général présentée ci-dessus, nous disposons donc d'une base de données de 232 échantillons, qui contient 4 *features* (*firstname*, *relation*, *profession* et *male_freq*) ainsi que la variable *sex* à prédire. Dans la suite de ce rapport, nous nous intéressons aux différentes méthodes développées pour notre tâche de prédiction du sexe.

3.1 Règles de décision construites à la main

Tout d'abord, avant d'explorer le potentiel de modèles de *Machine Learning*, il semble pertinent d'étudier les performances d'une méthode relativement naïve, basée sur des règles de décision construites à la main. Une telle méthode a de bonnes chances de bien fonctionner, puisque dans la majorité des cas, les informations personnelles des individus étudiés contiennent des marqueurs indiscutables de masculinité ou de féminité. Elle peut donc constituer une référence solide en termes de performance.

Dans le cadre de notre étude, nous établissons plusieurs règles de décision, classées par ordre de priorité :

— D'abord, pour un individu donné, si la fréquence d'occurrence masculine de son prénom (*male_freq*) est disponible et supérieure (resp. inférieure) à 0.95 (resp. 0.05), l'individu est classifié comme un homme (resp. une femme).

— Sinon, si la *relation* associée à l'individu contient un marqueur de masculinité ('père', 'pere', 'fils', 'frere', 'frère' ou 'chef') ou de féminité ('mère', 'mere', 'fille', 'soeur', 'femme' ou 'brue'), la classification est faite en conséquence.

— Sinon, si la *profession* associée à l'individu contient un marqueur de masculinité ('fils', 'ier', 'eur' ou 'patron') ou de féminité ('ière' ou 'euse'), la classification est faite en conséquence.

— Sinon, si *male_freq* est disponible et supérieure (resp. inférieure) à 0.5, l'individu est classifié comme un homme (resp. une femme).

— Enfin, si aucun des critères précédents n'a permis de conclure sur le sexe de l'individu, il est classifié de manière arbitraire comme un homme. Ce dernier choix est motivé par le fait que les données contiennent légèrement plus d'hommes que de femmes (52%).

Cette méthode a l'avantage d'être peu gourmande en ressources de calcul, et de proposer un bon niveau d'explicabilité. Toutefois, elle repose sur plusieurs choix arbitraires, tels que la définition des seuils de masculinité et de féminité (variable *male_freq*), ou encore la classification automatique d'un individu comme un homme si aucune des règles établies n'a permis de déterminer son sexe. Enfin, cette méthode manque de robustesse, puisqu'elle est sensible aux fautes d'orthographe, elle ne prend en compte qu'un faible nombre de marqueurs de masculinité ou de féminité parmi l'ensemble de ceux qui existent, et n'est pas capable de déceler d'éventuelles subtilités.

3.2 Méthodes ensemblistes basées sur des arbres de décision

Dans un second temps, nous nous intéressons à des méthodes ensemblistes, qui reposent sur des arbres de décision. Plus précisément, nous nous intéressons aux modèles *XGBoost* (*boosting*) et *Random Forest* (*bagging*). En effet, ces méthodes sont connues pour être performantes sur des données tabulaires, et semblent donc être des solutions adaptées à notre problème de classification.

Tout d'abord, nous procédons à un *split* des données, que nous séparons en *sets* d'entraînement (70%) et de test (30%). Ensuite, puisque les méthodes qui nous intéressent ne peuvent traiter que des variables numériques, nous appliquons un *preprocessing* spécifique à chaque variable :

— La variable cible (*sex*) est binaire, ses modalités « homme » et « femme » sont respectivement converties en 1 et 0.

— Les variables *firstname* et *relation* sont catégorielles, et sont encodées à l'aide du *TargetEncoder* de *scikit-learn*.

— La variable *profession* contient beaucoup de valeurs manquantes. Nous choisissons de considérer que les individus associés sont inactifs. Nous procédons alors à une binarisation de la variable *profession*, à partir des catégories « Inactif » (classe 0, personnes sans profession ou dont la profession n'est pas renseignée) et « Actif » (classe 1, personne dont la profession est explicitement renseignée).

— Les valeurs manquantes de la variable *male_freq* sont imputées par la moyenne de cette variable dans le *set* d'entraînement. Elle est ensuite standardisée à l'aide du *StandardScaler* de *scikit-learn*.

Les modèles de *Machine Learning* tels qu'*XGBoost* ou *Random Forest* ont l'avantage d'être relativement peu gourmands en termes de données d'entraînement, par opposition aux

modèles de *Deep Learning*. Ainsi, ils consituent une solution frugale à notre problème, plutôt adaptée au peu de données dont nous disposons. Ces modèles ont également le potentiel de détecter des relations plus subtiles entre les variables que ne peut le faire la méthode naïve présentée ci-dessus. Toutefois, ils ne prennent pas en compte la sémantique des mots qui constituent nos données, ce que pourrait par exemple faire un modèle de langage avancé basé sur des *Transformers*.

3.3 Utilisation d'un LLM pré-entraîné

Enfin, dans l'idée d'obtenir des représentations « riches » des informations contenues dans nos données textuelles, nous construisons un modèle de classification basé sur un LLM pré-entraîné sur un corpus de textes français. Plus précisément, nous utilisons le modèle CamemBERT [1], couplé à une couche de classification construite pour notre tâche de prédiction. Le modèle CamemBERT nous permet d'encoder les données personnelles de chaque individu et de les agréger sous la forme d'un *embedding* de taille 768. La couche de classification permet ensuite de prédire le sexe de chaque individu à partir de l'*embedding* de ses données personnelles.

Pendant l'entraînement, le modèle CamemBERT est figé, seule la couche de classification est entraînée. En effet, la quantité de données dont nous disposons est tout juste suffisante pour entraîner la couche de classification, et il n'est donc pas envisageable de procéder à un *fine tuning* d'un modèle aussi gros et complexe que CamemBERT (*Transformer-based*).

De la même manière que précédemment, nous procédons à un *split* des données, que nous séparons en *sets* d'entraînement (60%), de validation (20%) et de test (20%). Avant d'être envoyées dans CamemBERT, les données doivent être *tokenisées*. Ainsi, à partir des données tabulaires que nous avions obtenues à l'issue de la phase de *preprocessing* initiale, nous reconstituons une chaîne de caractères globale par individu, qui sera ensuite donnée au *tokenizer*. Cette chaîne de caractères présente la même structure que la variable *prediction* brute dont nous disposions au départ. Lors de la reconstitution de la chaîne de caractères, le *preprocessing* suivant est appliqué :

— Dans les variables *relation* et *profession*, les valeurs manquantes sont remplacées par le mot « inconnue ».

— Les valeurs manquantes de la variable *male_freq* sont imputées par la moyenne de cette variable dans le *set* d'entraînement. Elle est ensuite encodée et incluse dans la chaîne de caractères, dans un champ « tendance : ». Deux façons d'encoder cette variable sont considérées. La première méthode consiste à arrondir la valeur de *male_freq* et à l'inclure telle quelle dans la chaîne de caractères (le soin est laissé au *tokenizer* de traiter les nombres comme il le souhaite). La seconde méthode consiste à remplacer cette valeur par « masculin » (resp. « feminin ») si elle est supérieure (resp. inférieure) à 0.5.

L'utilisation du *tokenizer* permet potentiellement d'améliorer la robustesse du modèle. En effet, en découpant des mots en *tokens* élémentaires, il devient possible d'isoler des fragments de mots utiles pour la prédiction du sexe, et ce même si le mot était initialement mal orthographié. Par exemple, on peut imaginer que la *tokenisation* permette l'isolation du

fragment « thilde » dans le mot « bathilde », censé retranscrire le prénom « mathilde ». On peut ensuite imaginer que le *tokenizer* fasse de même avec d'autres occurrences du prénom « mathilde » bien orthographié, permettant ainsi au modèle de se concentrer sur le suffixe du prénom, indépendamment de la faute de retranscription.

Par rapport aux méthodes précédentes, ce modèle présente plusieurs avantages. Tout d'abord, il permet d'obtenir une représentation des données textuelles qui capte la sémantique qu'elles véhiculent (notamment grâce au mécanisme d'attention utilisé dans CamemBERT). De plus, CamemBERT a été entraîné sur un corpus de textes français, ce qui le rend particulièrement adapté à nos données. Toutefois, il s'agit d'une approche basée sur du *Deep Learning*, qui est donc particulièrement gourmande en données d'entraînement ainsi qu'en ressources de calcul.

4 Expériences et résultats obtenus

4.1 Règles de décision construites à la main

Pour cette méthode, il n'y a pas de phase d'entraînement du modèle, puisque la prédiction repose simplement sur une séquence de règles de décision. Ainsi, il suffit d'appliquer ces règles aux données dont nous disposons, et de mesurer les performances de la méthode. Les différents scores obtenus sont regroupés dans le tableau 1 :

Classe	Precision	Recall	F1-score
femme	0.97	0.91	0.94
homme	0.92	0.98	0.95

TABLE 1 – Scores obtenus

Nous pouvons également visualiser les résultats précédents sous la forme d'une matrice de confusion :

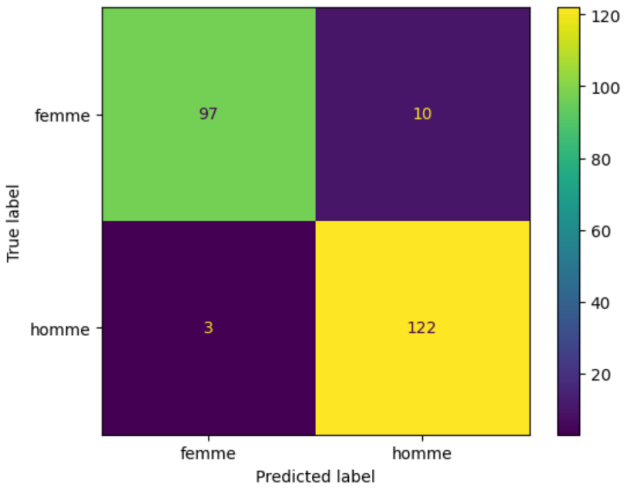

FIGURE 1 – Matrice de confusion

Ces résultats correspondent à une *accuracy* globale de 0.94, ce qui est plutôt impressionnant pour une méthode aussi basique. Il convient maintenant d'étudier les performances des autres méthodes qui nous intéressent, pour voir si elles offrent un meilleur compromis entre leurs performances, leur complexité et leurs avantages.

4.2 Méthodes ensemblistes : *XGBoost* et *Random Forest*

Pour ces deux modèles, nous *splittons* les données en *sets* d'entraînement et de test, comme expliqué précédemment. L'optimisation est réalisée par validation croisée (5-*fold*) sur une grille d'hyperparamètres, à l'aide de l'outil *GridSearchCV* de *scikit-learn*. L'estimateur qui donne la meilleure *accuracy* est retenu et utilisé pour l'évaluation finale de chaque modèle.

Pour *XGBoost* comme pour la *Random Forest*, nous observons un *overfitting* sur les données d'entraînement (tous les scores sont à 1), et ce malgré l'utilisation de la validation croisée. Au final, la *Random Forest* offre de bien meilleures performances qu'*XGBoost*, et atteint notamment une *accuracy* de 0.94 sur les données de test (contre 0.77 pour *XGBoost*). Les différents scores obtenus par la *Random Forest* sur les données de test sont regroupés dans le tableau 2 :

Classe	Precision	Recall	F1-score
femme	1.0	0.87	0.93
homme	0.91	1.0	0.95

TABLE 2 – Scores obtenus

La matrice de confusion associée est montrée sur la figure 2 :

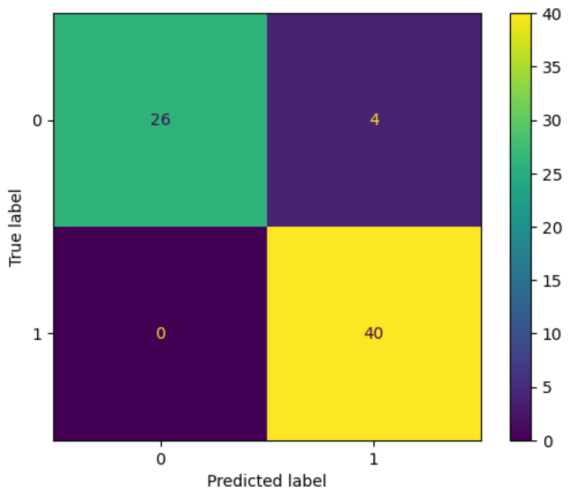

FIGURE 2 – Matrice de confusion

Il semble donc que la *Random Forest* soit une alternative valable aux règles de décision construites à la main. Toutefois, il apparaît que les performances du modèle dépendent du *split* des données. En effet, en réentraînant plusieurs fois le modèle avec des *splits* différents

à chaque fois, l'*accuracy* sur les données de test varie entre 0.88 et 0.94. Cette variabilité des performances est un effet indésirable, qui est probablement dû au fait que nous travaillons avec peu de données.

4.3 LLM pré-entraîné

Enfin, nous nous intéressons à l'entraînement de notre architecture de *Deep Learning*, qui combine un modèle CamemBERT pré-entraîné et une couche de classification. Comme expliqué précédemment, les données sont *splittées* en *sets* d'entraînement, de validation et de test. De plus, seule la couche de classification est entraînée, le modèle CamemBERT reste figé. Nous entraînons le modèle pendant 20 *epochs*, avec une *batch_size* de 8 et un *learning_rate* de 0.1.

A l'issue de l'entraînement, nous évaluons le modèle sur les données de test. Nous obtenons une *accuracy* de 0.96 environ, ce qui dépasse les performances des modèles précédemment étudiés. De plus, la performance finale sur les données de test ne semble pas être influencée par la façon dont la variable *male_freq* est encodée. Toutefois, comme pour la *Random Forest*, et dans des proportions encore plus importantes, nous observons une dépendance entre le *split* des données et la performance du modèle entraîné. En effet, en fonction du *split*, l'*accuracy* du modèle final varie entre 0.80 et 0.96. Comme précédemment, cet effet serait sans doute fortement atténué si nous disposions de plus de données d'entraînement.

5 Recommandations

Dans l'ensemble, les trois types de méthodes étudiées atteignent des performances satisfaisantes, avec des *accuracies* proches de 95%. En théorie, elles constituent donc toutes des solutions acceptables à notre problème de prédiction. En pratique, si nous ne disposons pas de beaucoup plus de données d'entraînement, il vaut probablement mieux choisir la première méthode, qui repose sur les règles de décision construites manuellement. Malgré sa simplicité, cette dernière fonctionne très bien, requiert peu de ressources de calcul, et ne souffre pas du manque de données qui affecte les modèles de *Machine Learning*. Si nous avions accès à beaucoup plus de données, il pourrait être intéressant de travailler avec un LLM comme CamemBERT, soit en entraînant seulement la dernière couche de classification comme nous l'avons fait, soit en allant jusqu'à *fine tuner* certaines de ses couches cachées. Dans le cadre de ce modèle, l'étape de *tokenisation* est particulièrement intéressante, puisqu'elle ouvre la voie à une décomposition robuste de données bruitées en plusieurs *tokens*, certains étant potentiellement discriminants pour la classification finale.

Enfin, le jeu de données proposé pour cette étude n'est probablement pas très représentatif de l'ensemble de la base de données. Ainsi, il est possible que certaines colonnes abandonnées parce qu'elles présentaient trop de valeurs manquantes (*état_civil* notamment) soient en réalité plus fournies, et présentent un réel potentiel informatif pour notre tâche de prédiction du sexe. Le cas échéant, il pourrait être intéressant de reprendre l'étude en les y incluant.

Références

[1] Louis Martin et al. "CamemBERT : a Tasty French Language Model". In : *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2020. DOI : 10.18653/v1/2020.acl-main.645. URL : http://dx.doi.org/10.18653/v1/2020.acl-main.645.