ENSAE NLP PROJET 2024

Tâche 2 : Prédiction du chef de foyer

Authors
Ilias Razig

Contents

1 Présentation et description des données

1.1 Import et description des données

La tâche abordée ici est la prédiction de la mention chef de foyer pour un individu afin de pouvoir regrouper les individus par foyer. Pour ce faire, nous disposons d'un fichier json qui regroupe les informations des listes de rencensements de 1836 à 1936. Les informations de chaque individu sont regroupées par tokens. Ces tokens peuvent être utilisés comme features en entrée pour les modèles de prédiction et sont les suivants :
- *age* : âge de l'individu
- *birth_date* : date de naissance de l'individu
- *lob* : lieux de naissance de l'individu
- *civil_status* : status civil de l'individu (homme marié, femme mariée, garçon...)
- *education_level* : niveau d'éducation de l'individu
- *employer* : employeur de l'individu
- $firstname$: prénom de l'individu
- *surname* : nom de famille de l'individu
- *household_surname* : nom de foyer de l'individu
- *maiden_name* : nom de jeune fille de l'individu
- *link* : lien de l'individu avec son chef de foyer
- *occupation* : occupation de l'individu
- *nationality* : nationalité de l'individu
- *observation* : informations complémentaires concernant l'individu

Après importation initiale du fichier et mise en forme des tokens sous forme de colonnes d'un dataframe, celui-ci contient 25448 lignes, qui proviennent de 1218 pages de recensement différentes. Parmi ces lignes, 367 sont vides. Après nettoyage de ces lignes (qui représente 1,4% du dataframe initial) les données restantes sont au nombre de 25081 lignes issues de 851 pages de recensement différentes.

Les champs dont nous disposons ne sont pas uniformément peuplés puisque certains comportent plus de valeurs nulles que d'autres commen présenté dans le graphique suivant :

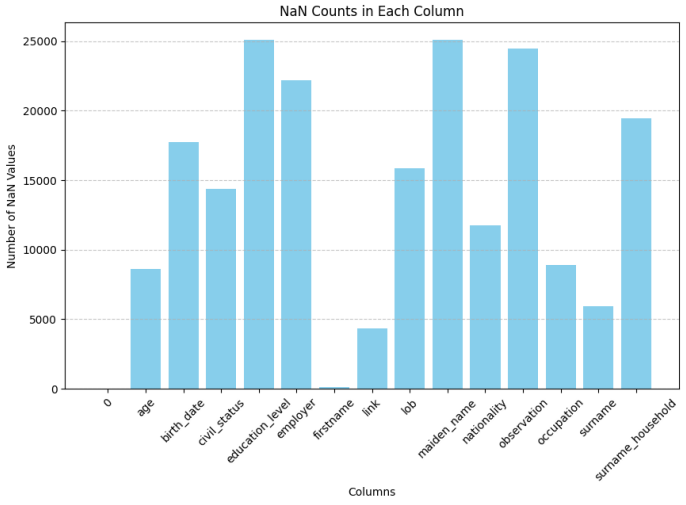

Figure 1: Nombre de valeurs nulles par variable

Nous pouvons observer que les colonnes les plus densément peuplées sont les colonnes '*firstname*', '*surname*' et '*occupation*'. Tandis que les colonnes les moins peuplées sont les colonnes '*education_level*', '*maiden_name*', et '*observation*'. Cependant bien que la majorité des colonnes présentent un grand nombre de valeurs manquantes, il est possible des remplir ces valeurs en effectuant certains nettoyage et traitements sur les données.

1.2 Statistiques descriptives et nettoyage des données disponibles

Le premier traitement qui a été effectué sur les données est le remplacement des noms des famille des invidus n'en ayant pas par leur nom de foyer lorsque celui-ci est disponible. Le choix a été réalisé de ne conserver au sein des données que les individus disposant d'un nom de famille compte tenu de l'importance de celui-ci et du faible nombre d'individus sans nom de famille résultant du traitement. En effet, la variable nom de famille est importante pour déterminer le chef d'un foyer; cette opération permet de réduire le nombre d'individus sans nom de famille de 5992 à 282 individus (soit près de 1% du nombre total d'individus).
Le second traitement qui a été effectué est le remplissage de toutes les valeurs 'idem' par la valeur les précédent puisque le dataframe comporte une notion d'ordre. La mise en minuscule des caractères de certaines colonnes a aussi été réalisée afin d'uniformiser les valeurs puisque pour un même mot, le fait qu'il commence par une majuscule ou non constitue deux mots différents pour certains modèles d'apprentissage. Les colonnes ayant reçu ce traitement sont : '*occupation*', '*civil_status*', '*observation*', '*link*', '*nationality*' et '*employer*'. Par exemple, pour la variable '*link*' de passer de 937 liens différents à 880 liens différents.

Nous observons que malgré le passage aux caractères minuscules pour les variables '*link*' et '*occupation*' que celles-ci présentent toujours une trop grande variété de valeurs uniques comme nous pouvons l'observer sur les graphiques [2],[3] qui présentent les 50 valeurs les plus nombreuses pour chacune des variables.

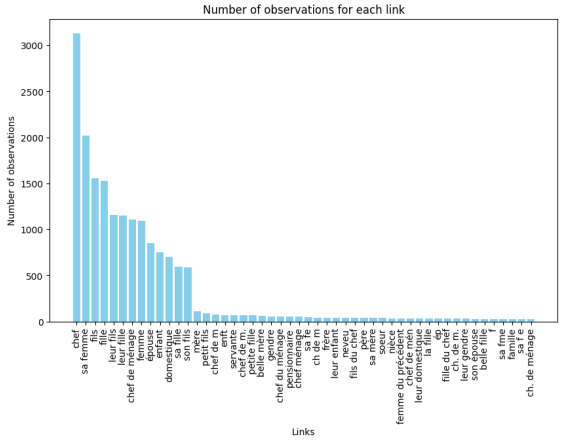

Figure 2: Liens les plus nombreux

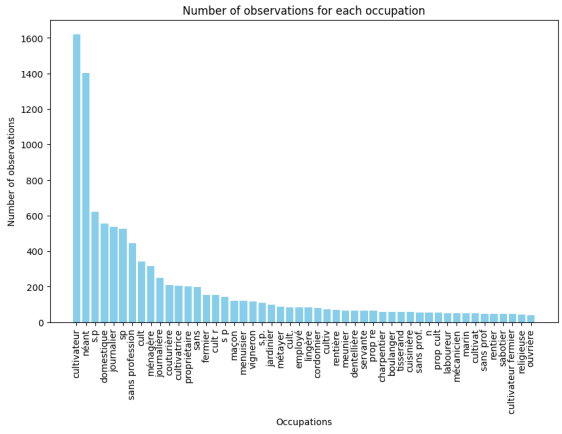

Figure 3: Occupations les plus nombreuses

Afin de réduire cette variété dans les données, des expressions régulières (regex) ont été utilisées sur les deux variables précédentes afin d'uniformiser les noms du lien 'chef' et pour uniformiser les noms de l'activité 'sans profession' pour l'occupation. En effet, ces deux mots présentaient un grand nombre de variantes ayant toutes le même sens. De plus, ces deux mots font partie des valeurs les plus nombreuses de leurs variables

respectives ce qui implique que leur uniformisation est importante afin d'aider les modèles à identifier les individus de manière plus précise puisque 'chef' est notre variable d'intérêt. Par ailleurs, la valeur 'chef' a aussi été affectée aux individus qui au sein d'un même document comportaient un nom de famille unique (ce qui signifie qu'il sont seuls dans leur foyer)
Ces traitements ont permis d'obtenir les répartitions par variable de mots suivantes :

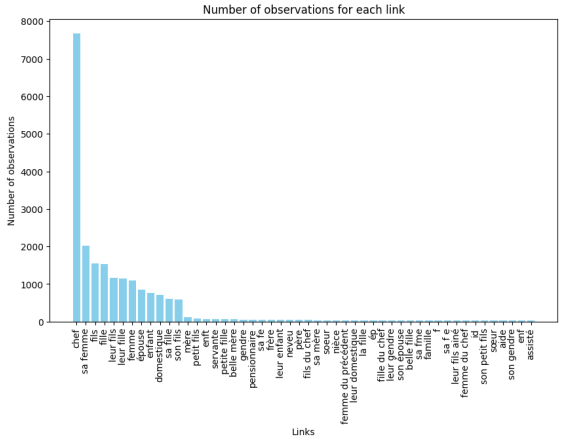

Figure 4: Liens les plus nombreux après traitement

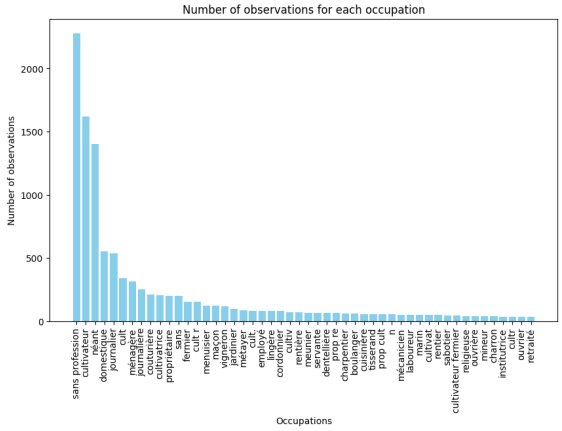

Figure 5: Occupations les plus nombreuses après traitement

La variable '$civil_status$' nécessitait aussi un traitement puisque celle-ci comportait un grand nombre de valeurs nulles (57%). Par ailleurs, cette variable ne peut prendre que 6 valeurs différentes : 'garçon', 'fille', 'homme marié', 'femme mariée', 'veuve', 'veuf'. Après une anaylse des données, certaines de ces valeurs peuvent être trouvées (au moins en partie) au sein d'autres colonnes telles que '*link*'. Cette dernière a donc été utilisée pour remplacer les valeurs nulle de la colonne '$civil_status$'. Des regex ont par la suite été utilisées sur la colonne pour trouver et normaliser les variantes des valeurs possibles de '$civil_status$', les valeurs ne correspondant pas ont par la suite été transformées en 'non disponible'. Ce traitement a donc permis de passer de 57% de valeurs nulles à 38%.

Enfin, en vue de réaliser une tâche de classification binaire puisque nous somme intéressé par la situation de chef ou non d'un individu, la variable '*link*' a été séparée en deux labels : 'chef' et 'not chef'. Ces labels présentent la répartition suivante[7] : 31% de 'chef' et 69% de 'not chef'

2 Analyse des modèles et Expérimentation

2.1 Identification du type de tâches à réaliser et choix des variables

En raison de la significativité des variables vis-à-vis de notre problème ainsi que de leur taux de remplissage comme présenté précédement, toutes les variables présentes dans la base n'ont pas été conservées pour réaliser les différents modèles de prédiction. Ainsi, seules les variables $civil\_status$, $firstname$, $surname$, $household\_surname$, $occupation$ ont été conservées en vue de prédire la variable $link$ car la combinaison de ces variables comporte des informations permettant de renseigner sur le status d'un individu. Par exemple un individu 'garçon', 'sans profession' et dont le nom de foyer est différent du sien a peu de chance d'être un chef de foyer.

2.2 Implémentation naïve

Afin de pouvoir comparer la perfromance des modèles qui seront utilisés dans les parties suivantes une méthode naïve a été implémentée afin qu'elle serve de 'baseline'. Compte tenu du fait que les données sont issues de registres de recensement datant du 19e siècle. Il serait logique que dans la plus part des cas de figure, un homme marié serait un chef de foyer. Nous avons donc prédit la variable 'chef' (qui a été encodée) selon si l'homme est marié ou non. De plus, une analyse des liens par statut conforte cette idée :

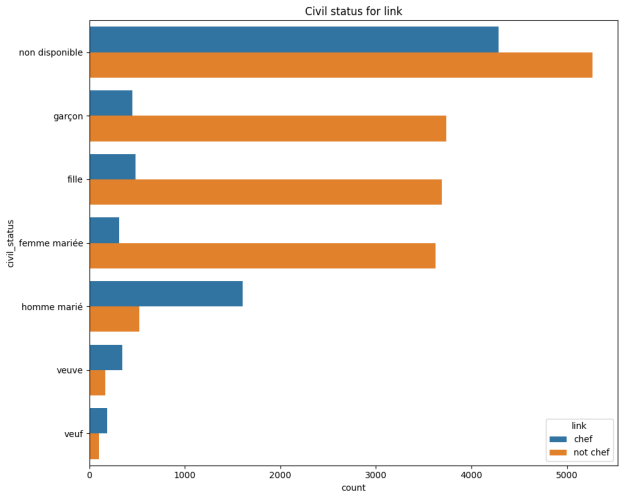

Figure 6: Statut civil par lien

Cette méthode donne donc sur toute la base de données une précision de 73,3% et un score F1 de 0,32. Cette méthode naïve donne déjà de bons résultats mais réalise souvent des faux négatifs[3], puisqu'elle néglige les autres statuts civil.

2.3 Naive Bayes

Le premier modèle d'apprentissage implementé est Naive Bayes, ce modèle d'apprentissage statistique présente l'avantage d'être simple d'utilisation et facile a entraîner. Le modèle prend en entrée les information de chaque colonne concaténées pour chaque individu et vectorisée avec un countvectorizer et donne (sur l'ensemble de test qui représente 20% de la base de donnée) une précision de 79,1% avec un score F1 de 0,63 ce qui représente une amélioration par rapport à la méthode naïve. Les erreurs de classifications[8] semblent bien distribuées cependant, le modèle réalise plus de faux négatifs que de faux positifs.

2.4 CAMEMBERT

Le second modèle d'apprentissage utilisé est un modèle pré-entrainé : CAMEMBERT. Ce modèle a été sélectionné en raison de ses bonnes performances de classifications. Le modèle a été fine-tune pour mieux répondre à notre tâche de classification. Le modèle prend en entrée une phrase basée sur les informations des colonnes et qui comprend aussi les informations de la personne précédente dans le registre ainsi que celles de la personne suivante afin de donner du contexte puisque pour bien classifier une personne comme chef ou non, il convient de disposer des informations des membres de son ménage. Cette méthode permet de prendre en compte ce contexte de manière partielle. Le modèle donne sur l'ensemble de test une précision de 92,2% avec un score F1 de 0,86 ce qui représente une amélioration par rapport au modèle précédent. Les erreurs de

classification[10] sont bien distribuées cependant, le modèle réalise toujours légèrement plus de faux négatifs que de faux positifs.

2.5 LLAMA 2

Le troisième modèle d'apprentissage utilisé est un LLM, LLAMA 2, développé par META. En raison de contraintes de mémoire, le modèle qui a été utilisé est la plus petite version. Sur la base des phrases de contexte générées, il est demandé au modèle de répondre à la question suivante : "À partir de ce texte, répond à la question suivante : Suis-je le chef de foyer ?". La réponse est ensuite interprété comme 'chef' si elle comprend le mot 'oui'. Le modèle donne sur l'ensemble de test une précision de 59,7% avec un score F1 de 0,27 ce qui est bien inférieur aux modèles précédents ainsi que la baseline. En observant la matrice de confusion[11] ainsi que les réponses, le modèle a du mal à prédire la classe 'chef' car il interprète souvent la personne précédente comme étant le chef et ne fait pas le lien entre les questions posées.

2.6 BART

Le dernier modèle d'apprentissage utilisé est BART, un modèle pré-entrainé de type zero-shot classification. Le modèle est utilisé de la même manière que CAMEMBERT, en lui spécifiant les classes que l'on souhaite classifier. Le modèle donne sur l'ensemble de test une précision de 34,7% avec un score F1 de 0,45 ce qui est aussi bien inférieur aux deux premiers modèles ainsi que la baseline. En observant la matrice de confusion[12], on réalise que le modèle prédit dans 91% des cas 'not chef' ce qui indique que le modèle n'est pas adapté à la tâche et nécessiterait d'être fine-tune.

2.7 Choix d'un modèle

Les résultats de chaque modèle sont présentés dans un tableau en annexe[13]. Compte tenu de ces résultats, le modèle retenu est les modèle CAMEMBERT fine-tune en raison de ses performances significativement plus élevées ainsi que les cas auxquels il s'applique. En effet, après analyse des résultats des modèles, les cas les plus difficiles à classifier sont les cas des foyers mono-personne puisque lorsqu'une personne est seule dans son foyer, elle est généralement non mariée, elle hérite ainsi du statut 'garçon' ou 'fille' cependant lorsqu'un garçon ou une fille est dans un foyer plus grand, il n'est jamais chef. C'est principalement cet élément qui explique le biais des faux positifs dans la matrices de confusion de tous les modèles utilisés.

3 Conclusion

Ainsi ces expérimentations sur la tâche de classification ont permis de tirer plusieurs conclusions. La première étant que compte tenu de la spécificité de notre tâche de classification, il est préférable d'entraîner ou bien de fine-tuner des modèles pré-entraînés (puisque ce sont ceux ayant les meilleures performances dans notre étude). Enfin, plusieurs axes d'améliorations de notre méthode sont possibles, dont les suivants :
- Inclure d'autres variables telles que l'âge (un nourisson ne peut pas être chef de foyer par exemple)
- Traiter et utiliser les informations complémentaires de la variable '*observation*'. - Créer une meilleure phrase de contexte[14] (car celle-ci est difficile à interpréter même pour un humain) ou bien une meilleure question en faisant du prompt engeenering. - Fine-tune un LLM pour qu'il se spécialise dans les tâches de classification (cette approche a été explorée mais en raison de contrainte de ressources (besoins importants en mémoire et en GPU) elle n'a pu être ménée à terme. - Réaliser un fine-tuning plus fin de CAMEMBERT pour obtenir de meilleures performances car celui utilisé est générique (provient de la documentation d'HuggingFace).

4 Annexes

Lien du github du code du projet : `https://github.com/iliasrazig/NLP-Project.git`

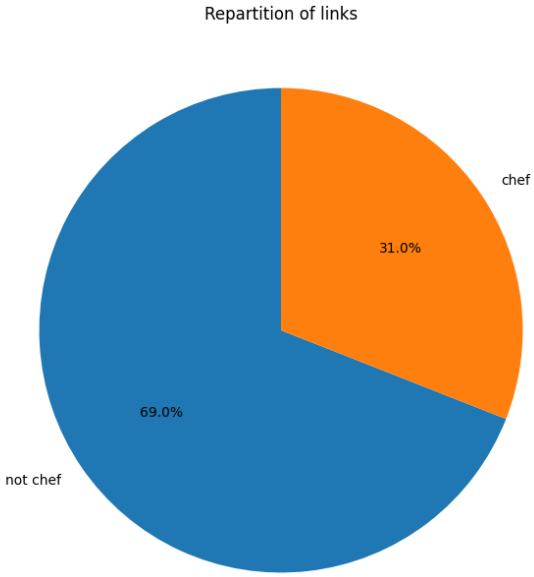

Figure 7: Répartition des labels

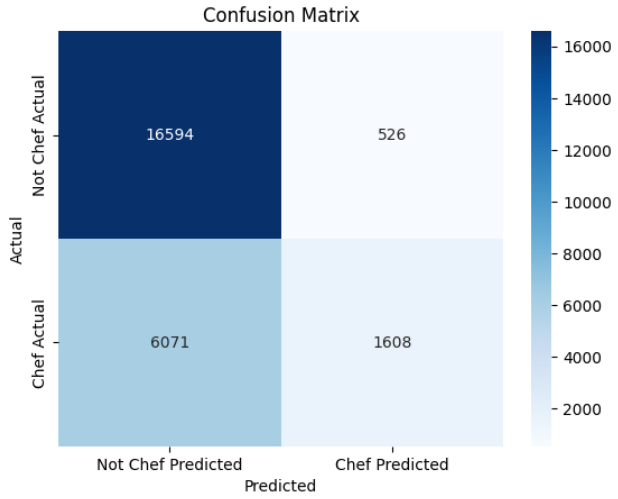

Figure 8: Matrice de confusion de l'approche naïve

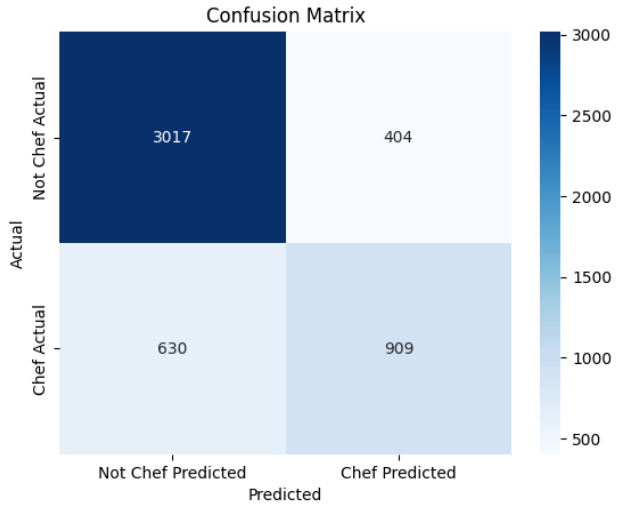

Figure 9: Matrice de confusion du modèle Naive Bayes

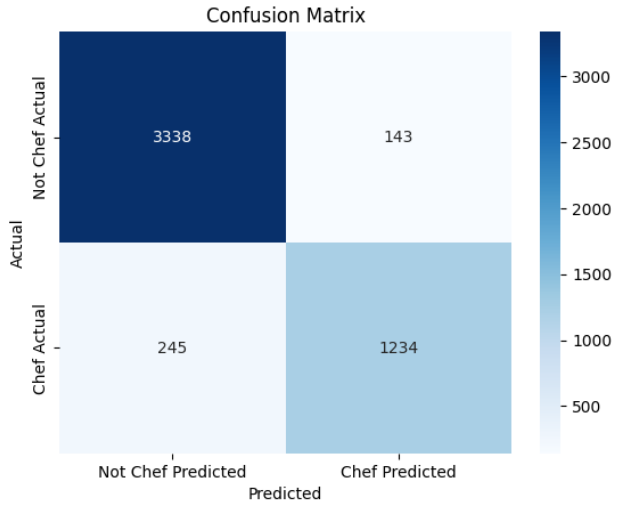

Figure 10: Matrice de confusion du modèle CAMEMBERT

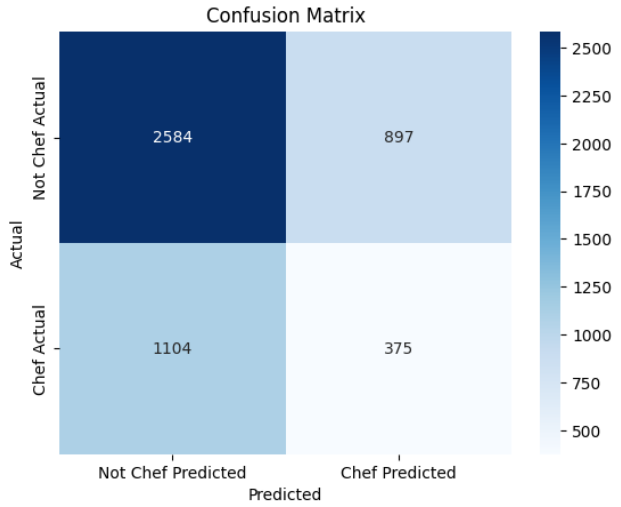

Figure 11: Matrice de confusion du modèle LLAMA 2

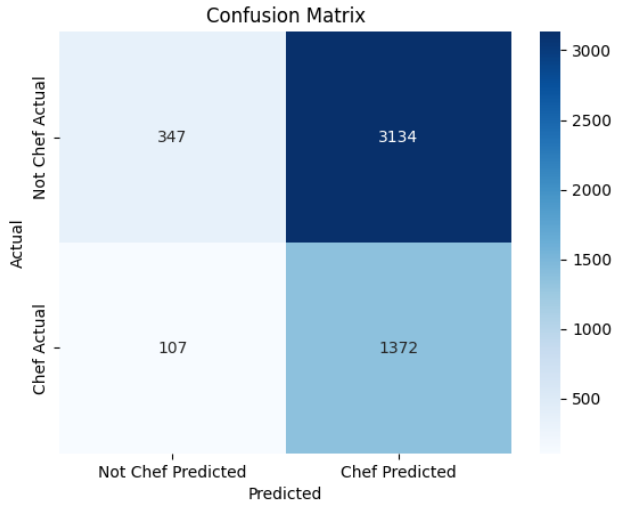

Figure 12: Matrice de confusion du modèle BART

Modèle	Accuracy	Score F1	Training / Fine-tuning
Baseline	73,3%	0,31	
Naive Bayes	79,1%	0,63	OUI
CAMEMBERT	92,2%	0,86	OUI
LLAMA 2	59,7%	0,27	NON
BART	34,7%	0,45	NON

Figure 13: Tableau comparatif des performances de chaque modèle

'Je suis Adèle Dutertre, mon nom de foyer est Allemant, mon statut civil est femme mariée et mon occupation est ouvrière. La personne qui me précède est Philippe Allemant, son nom de foyer est Allemant, son statut civil est homme marié et son occupation est tailleur. La personne qui me suit est Joachim Florentin Gaucheron, son nom de foyer est Gaucheron, son statut civil est homme marié et son occupation est pharmacien.'

Figure 14: Exemple de phrase de contexte d'un individu