Projet NLP - ENSAE 2023/2024

Samuel Bazaz samuel.bazazjazayeri@ensae.fr

April 18, 2024

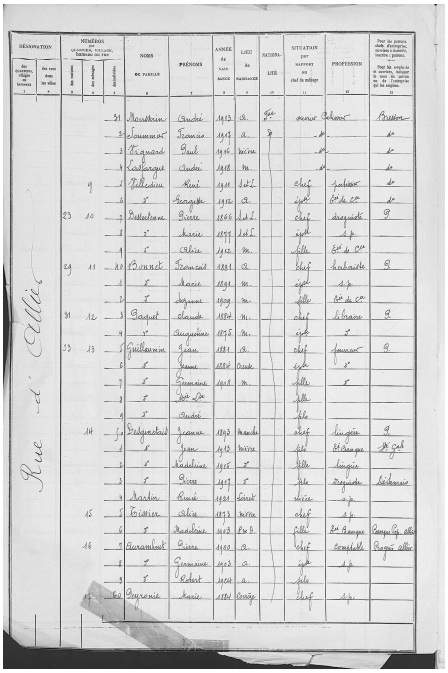

Figure 1: Moulins, Allier 1886

Abstract

Cette étude s'insère dans le projet de recensement Socface en construisant une étape de classification homme/femme sur des données entre 1836 et 1936, à travers une implémentation python disponible sur github[1]. Ce projet étudie différents modèles de classification binaire à partir de données textuelles et catégorielles, et compare les performances des 'classifier' sur des données de natures différentes: annotation humaine, OCR, avec ou sans information statistique. On verra que l'information statistique a disposition est de très bonne qualité pour la classification et que les modèles de Machine Learning peinent à apporter une plus value quand ce dernier est présent. L'enjeu est alors de retrouver des performances comparables sans l'information statistique et ou sur des données bruité issus de de l'OCR. Nous verrons par la suite si ces objectifs sont atteints.

[1] https://github.com/S-bazaz/intro_NLP_projet.git

1 Données et statistiques descriptives

1.1 Données brutes et volumétrie

Pour notre classification on dispose de deux bases:

- **transcriptions_with_sex.csv** concentre des informations tabulaires1 de recensement et le sex de 241 individus. Il y a deux versions de ces données tabulaires :
  - **groundtruth** que l'on notera Gr, pour des données annotées par des humains.
  - **prediction** que l'on notera Pr, pour des données issues d'un OCR.
- **firstname_with_sex.csv** est notre information statistique donnant pour 6946 noms le nombre d'hommes et le nombre de femmes les portant dans la population française.

On peut déjà constater que le volume des données est faible, ce qui motive le choix de modèles peu expressifs ou pré-entraînés, ainsi qu'une attention particulière au préprocessing afin de manuellement diminuer le bruit.

1.2 Préselection des données de transcriptions

On pourrait directement travailler sur les chaînes de caractères des transcriptions (ex:4.3). Hélas, l'information n'est pas toujours pertinente, est bruitée, et le manque de données nous limite dans la sélection automatique que pourrait réaliser un mécanisme d'attention. Une première étape est de mettre sous format tabulaire les transcriptions en ajoutant les préfixes gr et pr pour identifier les colonnes d'origines, puis d'harmoniser les types et les valeurs manquantes. On regarde suite à cela la proportion de valeurs vides:

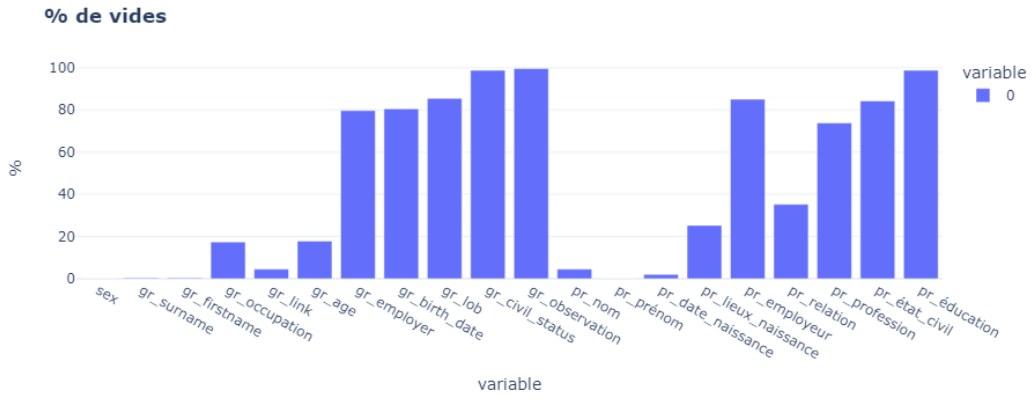

Figure 2: Proportions de valeurs vides des données de transcriptions

Ensuite, une analyse des histogrammes des différentes variables (cf: notebook 1_Preprocessing) permet de mesurer qualitativement la qualité et le caractère genré a priori de ces dernières. Comme attendu les données prédites par OCR sont beaucoup plus bruitées. Suite à cela, on conserve pour les transcriptions: le sex, le nom, le métier, les liens familiaux, la civilité. Aussi, pour nos 'labels'(voir 7) on remarque la présence d'une classe 'ambigu' de très faible proportion, cependant nous prenons le partie de ne pas tenir trop compte de cette catégorie en utilisant l'**accuracy** car les deux catégories 'homme femme' sont suffisamment équilibrés, et que l'on a pas de préférence entre faux positifs et faux négatifs.

1.3 Ajout de l'information statistique

On agrège ensuite l'information de la deuxième base en une colonne appelée **feminite_nom** qui fait la différence des logarithmes du nombre de femmes et du nombre d'hommes portant le même nom que l'individu. Lorsque le nom n'est pas dans la liste on choisit le nom le plus proche selon la distance d'édition de Levenshtein.

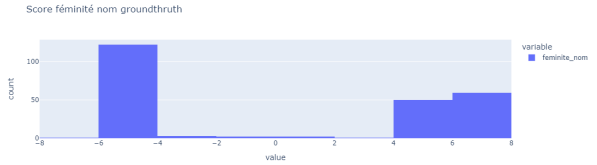

Figure 3: Le score de féminité du nom Gr

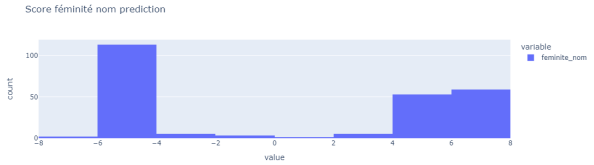

Figure 4: Le score de féminité du nom Pr

Par construction le signe du score nous donne le sex le plus co-occurent au nom, et on observe ci dessus (Figures 3-4) une distribution bimodale bien séparable ce qui est bon signe.

1.4 Amélioration des données textuelles

On remplace ensuite certains caractères et mots de manière à harmoniser les catégories comme les "sans profession" par exemple. Dans un deuxième temps on élimine les 'stopwords' en ajoutant 'idem' et certains mots a priori non genrés vu que l'on ne dispose pas de suffisamment d'information pour reconstruire le signal et que l'on veut réduire la variabilité des données.

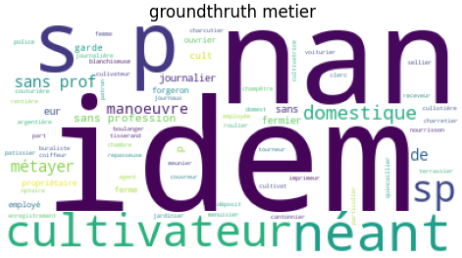

Figure 5: Word cloud métiers

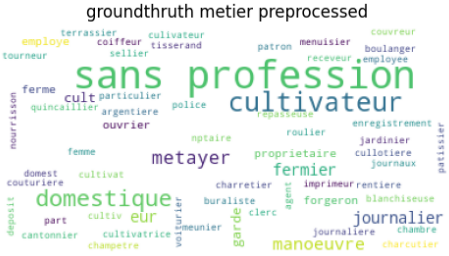

Figure 6: Word cloud métiers après préprocessing

Plus de visualisations 'Wordcloud' sont disponibles en annexe 4.3. On remarque que les données de prédictions ne respectent pas toujours la tabulation avec des informations associées au métier présents dans la colonne civilité par exemple.
Notons ensuite l'absence de racinisation par un 'stemmer' vu que l'on veut conserver au maximum l'information du genre.
On sépare enfin 60% des données en données d'entraînement et 40% en données de tests. Cette séparation est randomisée afin d'obtenir des les mêmes proportions entre les époques si une chronologie existe.

Par la suite nous utiliserons ces données et comparerons la performance des modèles sur différentes bases : non traité ou traité, Gr ou Pr.

2 Modèles

2.1 Un Modèle statistique déterministe

Le modèle le plus simple est un classifier statistique qui seuil à 0 le score de féminité du nom. Ce dernier reposant sur l'information statistique, nous allons étudier par la suite des modèles de Machine Learning pour essayer de n'utiliser que les transcriptions. On combinera ces derniers au modèle statistique en ajoutant la prédiction du classifier dans la colonne **genre_nom**.

2.2 Fasttext

Fasttext[2] est un modèle de machine learning relativement simple, dont le fonctionnement est présenté dans les grandes lignes en annexe B (4.3). Cependant son utilisation est assez difficile car aucun framework l'utilise, nous conduisant à réaliser un travail conséquent pour préparer les données d'entrées et pour reformatter les résultats. Les données d'entraînement sont alors mis dans un fichier .txt comme dans l'exemple suivant:

__label__femme metier: nom:rose lien_famille:mere genre_nom:femme civilite: feminite_nom:5.927

2.3 Transformers

Une autre limitation de Fasttext est qu'il ne peut apprendre des liens qu'entre des tokens consécutifs par l'utilisation de n_words et de n_gram. Pour contextualiser un nom mixte par le métier ou la civilité, on peut tirer partie du mécanisme d'attention. On obtient alors un nouvel embedding pour chaque contexte comme combinaison convexe de l'embedding et de celles des autres tokens présents. N'ayant pas un volume de données suffisant pour apprendre un modèle de langage pertinent, on contourne cette difficulté par l'utilisation de transfert learning sur des transformers préentrainés. Il existe un nombre important d'architectures transformers et pour des raisons de comparabilité nous allons étudier des modèles relativements simples qui ont déjà fait leur preuve en production: DistilBERT[14] et CamemBERT[12]. DistilBERT est une version plus légère de BERT[5] et nécessite donc moins de données pour le réentrainement. CamemBERT est une version de RoBERTa[10] spécialisée aux corpus français par l'équipe ALMAnaCH de l'Inria. En comparant ces deux modèles, nous pourront déterminer si il est préférable de s'orienter vers des modèles compactes ou spécialisés au français.

3 Expérimentation

3.1 Protocole expérimentale

A ce niveau nous disposons de 8 bases de données:
(brute ou traité) et (Gr ou Pr) et (train ou test).

A partir de ces bases, nous testons différents combinaisons de variables, afin de mesurer l'effet de l'information statistique et la pertinence des features présélectionnés. Aussi nous disposons de 5 modèles: Statistique, Fastext, Fastext préentrainé, DistilBERT préentrainé, et CamemBERT préentrainé. Pour chaque modèle de machine learning on crée un modèle hybride ML+Statistique pour l'ajout de genre_nom dans les variables. On collecte ensuite après entraînement l'accuracy obtenue sur les données de tests, ces derniers sont présents en Annexes C (4.3.3).

3.2 Analyse des résultats

Premièrement, regardons le cas où l'on dispose de l'information statistique (1) avec le score de féminité du nom. Le classifier Statistique est déjà très bon avec 0.97 d'accuracy pour Gr et 0.94 pour Pr ! La différence s'explique par un recours plus important à la distance d'édition dans le cas de la prédiction ce qui est source d'erreur malheureusement. Lorsque l'on combine le classifier à un modèles ML le score reste proche de celui du classifier, et seuls les transformers performent mieux allant jusqu'à 0.99 pour Gr et 0.96 pour PR, bien que la performance soit davantage variable d'une run à une autre: on tombe sur un arbitrage biais variance.
Dans un deuxième temps, on peut aussi noter que sans le classifier statistique, les modèles Fasttext performent moins biens que ce dernier et particulièrement mal lorsque les données sont non traités. Les transformers eux sont plus robusts et l'apport du preprocessing apparait ici marginale voir ambigu. Enfin, entre DistilBERT et CamemBERT, DistilBERT semble meilleure même si des statistiques seraient nécessaires pour trancher.

Regardons ensuite le cas sans informations statistique (2). Concernant les variables, on remarque que la civilité et les liens familiaux améliorent les prédictions alors que les métiers les détériorent. Concernant Fasttext, le transfert learning n'améliore pas vraiment les résultats et une raison à cela est que l'espace latent associé aux poids disponibles sur internet est de trop faible dimension. Aussi, les données d'entrées ont une structure assez éloignée d'un corpus classique et les ngram les plus présents sont de natures très différentes. Le preprocessing améliore grandement les performance permettant d'atteindre 0.91 pour Gr et 0.82 pour Pr.
Concernant les Transformers, on remarque encore une fois leur plus grande robustesse, avec une plus value du preprocessing relativement faible. DistilBERT domine avec un maximum de 0.97 pour Gr et 0.93 pour Pr ce qui est proche des résultats du classifier statistique ! Enfin, CamemBERT semble trop lourd pour que le volume de réentrainement suffisse et un modèle spécialisé au français plus compacte serait à tester.

4 Conclusion

4.1 Recommandations

Dans le cadre de l'OCR, l'utilisation d'un transformers est à prioriser en raisons d'une meilleure robustesse aux bruit et afin de diminuer le travail de preprocessing nécessaire si les ressources humaines sont limitantes. Comme on la vu, cela permet aussi d'estimer à la volé avec une précision relativement bonne sans nécessiter de croiser une étude stastistique. Cependant, si les contraintes le permettent, l'utilisation des statistiques des noms par les résultats du classifier statistique, combiné à l'information du nom, des liens familiaux, et de la civilité, semble être la meilleure stratégie au vu de cette étude.
Ensuite, la faible volumétrie des données nous conduit à prioriser des modèles préentrainés distillés (dont l'apprentissage tire partie de modèles plus gros), comme pour DistilBERT dont les résultats sont déjà très satisfaisants. Cependant si le volume de traitement est conséquent, utiliser le classifier statistique sera bien moins coûteux et offrira déjà un faible taux d'erreur.

4.2 Pistes d'amélioration

Les architectures LLM évoluent rapidement et des modèles distillés plus récents comme Zéphyr-7B [15] seraient à tester. Ensuite, une source d'erreur importante est l'utilisation de la distance d'édition pour le calcul du score de féminité du nom. On pourrait très bien imaginer utiliser la métrique euclidienne dans l'espace latent du modèle transformers pour améliorer le choix du nom le plus proche dans la liste statistique.

D'autres part, l'optimisation des hyperparamètres serait à améliorer avec par exemple un grid search ou un bayesian search[16]. Aussi, les comparaisons entre les différentes stratégie manquent de robustesse. Pour palier cela on peut s'inspirer des méthodes de Monte Carlo[3] et de Bootstrap[6], et échantillonner plusieurs scores à partir d'un échantillonnage de plusieurs bases de train et test. Ce faisant, on obtient des distributions de scores que l'on peut comparer par un test de rang de Wilcoxon-Mann-Whitney[11].

4.3 Conclusion

Il apparaît ainsi qu'un modèle de Machine Learning n'est pas toujours nécessaire pour déterminer le genre avec une précision correcte, bien que les mécanismes d'attention, en contextualisant des noms bruité par l'OCR permettent une meilleure robustesse. Il en résulte un choix entre une règle de décision peu coûteuse et un modèle plus précis mais dont la conception n'a pas été sans conséquences.
On peut enfin nuancer l'approche statistique si l'on veut faire une classification plus récente, car les Marie et les Jean sont aujourd'hui moins répandus, bien que les noms portent encore souvent l'information du genre.

References

[1] R. Bellman et al. *Dynamic Programming*. Rand Corporation research study. Princeton University Press, 1957. ISBN: 9780691079516. URL: https://books.google.fr/books?id=wdtoPwAACAAJ.

[2] Piotr Bojanowski et al. *Enriching Word Vectors with Subword Information*. 2017. arXiv: 1607.04606 [cs.CL].

[3] George Casella Christian P.Robert. *Monte Carlo Statistical Methods*. https://link.springer.com/book/10.1007/978-1-4757-4145-2. Springer, 1999.

[4] Essam Debie and Kamran Shafi. "Implications of the curse of dimensionality for supervised learning classifier systems: theoretical and empirical analyses". In: *Pattern Analysis and Applications* 22 (2019), pp. 519–536.

[5] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv: 1810.04805 [cs.CL].

[6] Bradley Efron. "Bootstrap Methods: Another Look at the Jackknife". In: *Annals of Statistics* 7 (1979), pp. 1–26. URL: https://api.semanticscholar.org/CorpusID:227312712.

[7] Jerome H Friedman. "On bias, variance, 0/1—loss, and the curse-of-dimensionality". In: *Data mining and knowledge discovery* 1 (1997), pp. 55–77.

[8] Ian Goodfellow et al. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[9] Andrei Koudriavtsev et al. "Maxwell — Boltzmann Statistics". In: *The Law of Mass Action*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 1–42. ISBN: 978-3-642-56770-4. DOI: 10.1007/978-3-642-56770-4_1. URL: https://doi.org/10.1007/978-3-642-56770-4_1.

[10] Yinhan Liu et al. *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. 2019. arXiv: 1907.11692 [cs.CL].

[11] H. B. Mann and D. R. Whitney. "On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other". In: *The Annals of Mathematical Statistics* 18.1 (1947), pp. 50–60. DOI: 10.1214/aoms/1177730491. URL: https://doi.org/10.1214/aoms/1177730491.

[12] 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: https://doi.org/10.18653%2Fv1%2F2020.acl-main.645.

[13] Donald B Percival and Andrew T Walden. *Spectral analysis for physical applications*. cambridge university press, 1993.

[14] Victor Sanh et al. *DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter*. 2020. arXiv: 1910.01108 [cs.CL].

[15] Lewis Tunstall et al. *Zephyr: Direct Distillation of LM Alignment*. 2023. arXiv: 2310.16944 [cs.LG].

[16] Jia Wu et al. "Hyperparameter Optimization for Machine Learning Models Based on Bayesian Optimizationb". In: *Journal of Electronic Science and Technology* 17.1 (2019), pp. 26–40. ISSN: 1674-862X. DOI: https://doi.org/10.11989/JEST.1674-862X.80904120. URL: https://www.sciencedirect.com/science/article/pii/S1674862X19300047.

Annexes A: Preprocessing et Statistiques descriptives

Exemple transcription Gr : 'surname: Chardon firstname: Marie occupation: idem link: fille age: 30 '

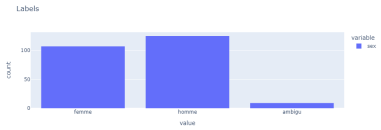

Figure 7: Proportions des labels

Stop words : ('idem', 'id', 'nan', 'neant', 'son', 'sa', 'le', 'la', 'leur', 'de', 'enf', 'enfant', 'menage', 'famille', 's', 'e', 'd', 'f', 'c', 'p', 'b', 'v', 'm', 'p')

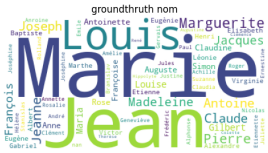

Figure 8: Nom Gr

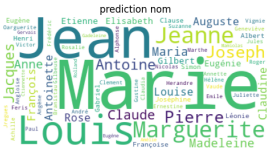

Figure 9: Nom Pr

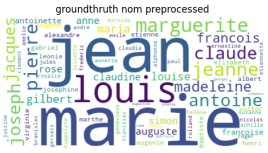

Figure 10: Nom Gr après

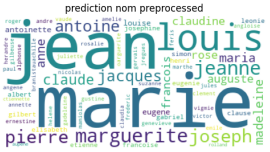

Figure 11: Nom Pr après

Figure 12: metier Gr

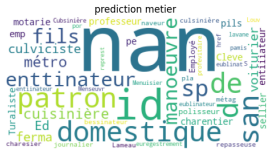

Figure 13: metier Pr

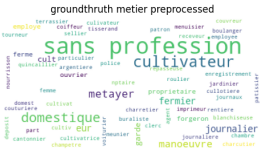

Figure 14: metier Gr après

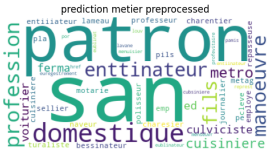

Figure 15: metier Pr après

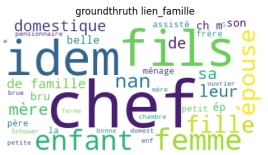

Figure 16: famille Gr

Figure 17: famille Pr

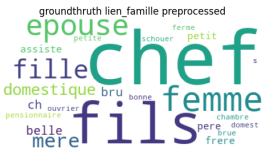

Figure 18: famille Gr après

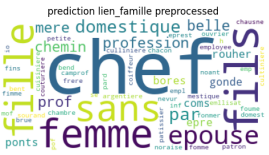

Figure 19: famille Pr après

Figure 20: civilite Gr

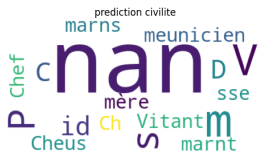

Figure 21: civilite Pr

Figure 22: civilite Gr après

Figure 23: civilite Pr après

Annexes B: Fasttext

Voyons ici quelques intuitions autour du fonctionnement de Fasttext. Pour l'exemple, on prendra une phrase déjà formatée comme entrée du modèle: EST ETUD AGE22

4.3.1 Une analyse fréquentielle des sous-mots: de la phrase au vecteur

La première étape de l'algorithme est un travail de découpage. On découpe les mots, mais aussi des *nwords*, des groupe de mots typiquement de 3 mots, et des *ngram*, des chaines de caractère de longueurs fixes (ici égale à 3). De cette manière on garde en mémoire l'ordre des mots, certaines racines, et séquences de lettres utiles à une interprétation sémantique de la phrase.

EST, ETUD, AGE22, EST ETUD AGE22, EST, STE, TET, ETU, TUD, UDA, DAG, AGE, GE2, G22

L'objectif est ensuite d'obtenir un jeu de coordonnées pour nos futurs vecteurs associés aux phrases. Pour se faire, l'algorithme répertorie toutes les chaînes de caractères obtenues à travers le corpus et leur associe une coordonnée. Si le nombre de coordonnées est insuffisant, deux mots distincts peuvent être associés à la même dimension.

Enfin, une simple analyse fréquentielle permet de calculer les coordonnées de notre "vecteur phrase". Dans notre exemple, chaque fragment apparaît une unique fois parmi les 14 fragments associés à la phrase, donc chaque coordonnée a pour valeur la fréquence 1/14. Remarquons que l'analyse spectrale est une méthode largement employée en physique[13], ayant fait ses preuves en traitement du signal, il n'est ainsi pas étonnant que son utilisation soit courante en NLP.

4.3.2 La matrice d'embedding : passer des fréquences au sens

L'idée ici est de se ramener à un espace vectoriel plus petit afin d'éviter la malédiction des hautes dimensions[1] qui rend les classifier très instables[7, 4]. L'algorithme choisit par optimisation la meilleure transformation linéaire dans un espace de dimension fixée.

Les "vecteurs phrases" sont ainsi plongés dans un "espace sémantique", appelée "espace latent" ou "espace d'embedding". Dans cet espace, si l'optimisation est bien réalisée, deux "vecteurs sémantiques" aussi appelés "embedding", sont proches si le sens des phrases initiales le sont. La matrice associée au "plongement" est appelée "matrice d'embedding".

On comprend ainsi comment le choix de la représentation d'une variable catégorielle (code ou texte en claire) peut influencer notre classification. Ajouter de l'information textuelle peut conduire à modifier la base et les coordonnées de nos vecteurs de façon potentiellement délétère.

4.3.3 Un classifier linéaire one vs all : comment séparer les catégories

Un perceptron est enfin utilisé pour la classification. Chaque catégorie qui dans notre cas est le sex, est associé à un hyperplan de notre espace sémantique déterminée par optimisation. Savoir si notre phrase est associée à une catégorie revient à calculer la distance de son vecteur sémantique avec l'hyperplan associé, en prenant en compte l'orientation pour le signe.

Enfin, afin d'obtenir des probabilités d'appartenance à une catégorie donnée, la fonction softmax est utilisée sur ces différentes distances relatives. Cette fonction déjà connue en physique statistique[9], donne plus de poids aux valeurs maximales et ce de manière différentiable d'où son intérêt pour la descente de gradient et son utilisation en Deep learning[8].

Annexes C: Résultats

Model	**epochs**	**lr**	**variables**	**Gr**	**Pr**
Statistique	0		feminité nom	0.97	0.94
Fastext	200	4e-2	raw txt + feminité nom	0.85	0.52
			feminité nom, nom, lien famille, métier,civilité	0.90	0.83
Fastext + Statistique	200	4e-2	raw txt + feminité nom	0.97	0.94
			feminité nom, nom, lien famille, métier,civilité	0.96	0.94
Fastext préentrainé	200	3e-2	raw txt + feminité nom	0.81	0.52
			feminité nom, nom, lien famille, métier,civilité	0.91	0.82
Fastext préentrainé+ Statistique	200	3e-2	raw txt + feminité nom	0.97	0.94
			feminité nom, nom, lien famille, métier,civilité	0.96	0.94
DistilBERT préentrainé	5	5e-5	raw txt + feminité nom	**0.99**	**0.95**
			feminité nom, nom, lien famille, métier,civilité	**0.99**	0.94
DistilBERT préentrainé+Statistique	5	5e-5	raw txt + feminité nom	0.96	0.93
			feminité nom, nom, lien famille, métier,civilité	**0.99**	**0.96**
CamemBERT préentrainé	15	5e-5	raw txt + feminité nom	0.97	0.94
			feminité nom, nom, lien famille, métier,civilité	**0.99**	**0.95**
CamemBERT préentrainé+Statistique	15	5e-5	raw txt + feminité nom	**0.99**	0.94
			feminité nom, nom, lien famille, métier,civilité	**0.98**	0.93

Table 1: Accuracy avec l'information statistique

Model	epochs	lr	variables	Gr	Pr
Fastext	200	4e-2	raw txt	0.86	0.59
			nom	0.82	0.79
			nom, lien famille	0.91	0.79
			nom, civilite	0.82	0.80
			nom, lien famille, civilite	0.90	0.82
			nom, lien famille, métiers	0.89	0.77
			nom,lien famille, métier,civilité	0.90	0.79
Fastext préentrainé	200	3e-2	raw txt	0.68	0.64
			nom	0.82	0.78
			nom, lien famille	0.91	0.79
			nom, civilite	0.84	0.79
			nom, lien famille, civilite	0.90	0.81
			nom, lien famille, métiers	0.88	0.76
			nom,lien famille, métier,civilité	0.90	0.79
DistilBERT préentrainé	5	5e-5	raw txt	**0.96**	0.88
			nom	0.94	0.89
			nom, lien famille	**0.97**	0.87
			nom, civilite	0.95	0.86
			nom, lien famille, civilite	**0.97**	**0.90**
			nom, lien famille, métiers	0.94	**0.93**
			nom,lien famille, métier,civilité	0.95	0.88
CamemBERT préentrainé	15	5e-5	raw txt	0.91	0.89
			nom	0.91	**0.90**
			nom, lien famille	0.93	**0.91**
			nom, civilite	0.84	0.83
			nom, lien famille, civilite	0.93	0.88
			nom, lien famille, métiers	0.92	0.89
			nom,lien famille, métier,civilité	0.88	0.89

Table 2: Accuracy sans information statistique