Analyse de la distribution des noms de rues en France

Gomez Malaurie

Janvier - Juin 2022

Résumé. Historiquement, en France, les noms de rues ont été donnés et ont changés en suivant des logiques d'appellations propres aux époques. Aujourd'hui, les noms de rues reflètent le passage des époques, et sont ainsi très variés dans leur types et leurs significations. Ce projet a pour but d'analyser leur distribution afin de répondre à diverses interrogations telles que les tendances de dénomination dans une zone géographique donnée, ou la récurrence de certains noms en fonction de leur nature.
Mots-clés: odonymie, intégration de données, mesures de similarité, classification automatique.

1 Introduction

Dans le cadre du second semestre de master 1 d'informatique, j'ai travaillé sur le sujet de POM (Projet d'Orientation en Master) intitulé "Analyse de la distribution des noms de rues en France", sous l'encadrement des professeurs Fabien Duchateau et Franck Favetta, membres de l'équipe Base de Données du LIRIS (Laboratoire d'InfoRmatique en Image et Système d'information).

Les odonymes sont les noms propres par lesquels sont désignées les voies de communication, telles que les rues, mais aussi les routes, les places, les chemins, etc. Un odonyme se compose généralement de deux parties : un indicateur du type de voie (« rue », « avenue », ...) et d'un nom individuel (« De Gaulle », « de l'église », ...). C'est cette deuxième partie qui est particulièrement intéressante. Effectivement, des typologies spécifiques aux époques ont été utilisées pour nommer les voies [1] . Au Moyen Âge par exemple, on privilégie la logique fonctionnelle, et les noms de voies sont en rapport avec leur emplacement (« place du marché », « rue des jardins », ...). À partir de 1600, cette logique est peu à peu abandonnée, et on préfère donner des noms de personnalités importantes aux voies. Lors de la révolution française, des noms faisant référence à des valeurs étaient privilégiés (« rue de la libération », « place de la nation »). Au fil du temps, certains noms de rues ont été changés ou modifiés, d'autres ont gardé leurs noms d'origine. Ainsi, il y a une grande variété dans l'odonymie en France.

La distribution des odonymes a déjà été étudiée pour des villes particulières [2], mais l'étude de la distribution sur le territoire français reste actuellement limitée. L'objectif du projet est ainsi de permettre l'étude de l'odonymie en France à différentes échelles, dans des zones sélectionnées plus ou moins précises. L'un des défis consiste à détecter les noms de rue équivalents entre plusieurs communes, du fait de la réutilisation de noms de rue populaires sur tout le territoire. Pour simplifier l'étude des odonymes, une approche consiste à associer une catégorie à chaque nom de rue afin de produire des statistiques générales.

Pour répondre à cette problématique, le projet a été découpé en trois parties, comme le montre la figure 1. Tout d'abord, la déduplication des données permet de déterminer les noms de rues équivalents, par exemple, « De Gaulle » et « Ch. de Gaulle » (détails en **section 2**). Une étape de blocking a été nécessaire pour traiter efficacement les 9 millions de rues en France. La seconde étape concerne la classification, c'est-à-dire l'affectation d'une rue à une catégorie (détails en **section 3**). Pour cela, il a d'abord fallu procéder à la définition d'une taxonomie (catégories principales, et éventuellement des sous-catégories) puis à la création d'une expertise manuelle (« Ch. de Gaulle » réfère à une personne, «de l'église » réfère à un bâtiment). Enfin, l'utilisation d'algorithmes d'apprentissage a permis la prédiction de la catégorie de nouveaux odonymes. Lors de ces deux premières étapes, j'ai mis en place des expérimentations pour évaluer et peaufiner différents algorithmes dans le but d'améliorer les résultats. Enfin, la dernière étape concerne le développement d'une application web afin de présenter les résultats (détails en **section 4**).

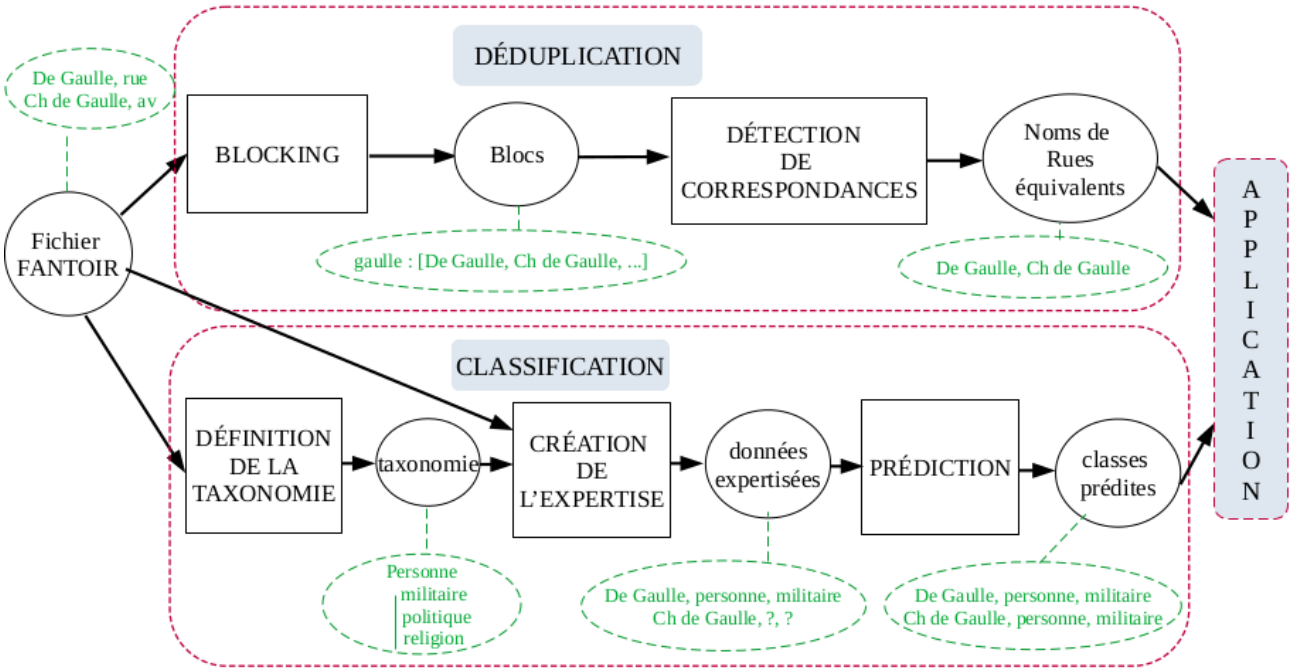


Figure 1. Aperçu de l'approche développée dans ce projet POM.

2 Déduplication des données

Dans cette section, je décris comment j'ai nettoyé le jeu de données des noms de rue en détectant les noms équivalents (déduplication et appariement d'entités [3]).

2.1 Base de données

Les données sont tirées du fichier FANTOIR[1] (Fichier ANnuaire TOpographique Initialisé Réduit). Il s'agit d'un fichier régulièrement mis à jour recensant les voies triées par communes, et contenant des informations sur chaque voie, comme son identifiant, le code de la commune et du département, la nature de la voie, etc. Ce fichier comporte environ 9 millions de lignes pour une taille de 995 Mo décompressé. Il était donc primordial d'intégrer les données dans une base de données pratique et performante afin d'y accéder et de les manipuler facilement.

À ces fins, j'ai utilisé le SGBD Postgresql, et plus spécifiquement l'instance hébergée sur un serveur de l'université. Le fichier comportant beaucoup d'informations sur chaque entrée, je me suis référée au descriptif du fichier afin de modéliser de manière appropriée le domaine (voir schéma entité-association en Annexe 1). J'ai favorisé la conservation d'une majorité d'informations en vue de l'étape d'apprentissage, pour laquelle différentes propriétés pourront être utiles.

Comme l'objectif est d'étudier les noms de rue, il est primordial de regrouper les odonymes équivalents (ceux qui représente la même entité, par exemple, "avenue Marie Curie" et "rue Marie Skłodowska-Curie" se réfèrent à la même personne), et ainsi, de nettoyé le jeu de données.

1 fichier FANTOIR

2.2 Blocking

Pour détecter les noms de rues équivalents, il va falloir comparer les noms deux à deux et calculer une similarité à l'aide d'algorithmes. Or, la base de données comportant 9 millions d'entrées, la comparaison du produit cartésien des rues, en particulier avec une mesure de similarité, serait extrêmement longue. C'est pour cela que je suis passé par une étape de blocking.

Afin de diminuer le nombre de comparaisons, il a fallu définir une clé de blocking permettant de savoir si oui ou non deux noms seront comparés plus finement. La clé utilisée est la présence d'un mot commun dans les noms des rues. En effet, deux noms de rues équivalents ont de fortes chances de partager au moins un mot en commun. Par exemple, "de Gaulle" et "Ch de Gaulle" ont "Gaulle" en commun et devront être comparés. Au contraire, "de Gaulle" et ''des fontaines" n'ont aucun terme en commun, et on peut éviter de les comparer.

En étudiant les données, je me suis rendue compte de trois failles dans cette logique. Tout d'abord, les mots tels que «le», «la», « du », etc sont présents dans une majorité de noms. Par exemple, « de la fontaine » et « Ch de Gaulle » ont bien un mot en commun ("de"), mais ils ne sont en rien similaires. Il est donc important de ne pas prendre en compte ce type de mots. Ensuite, la présence de pluriel et/ou de féminin dans les noms peut également impacter le blocking. Par exemple, « rue de la fontaine » et « rue des fontaines » sont extrêmement similaires, mais ils n'ont en soit aucun mot exact en commun. Afin de régler ce problème, j'ai appliqué une lemmatisation[2] à chaque mot des noms de rues, c'est-à-dire une extraction de la racine du mot. Ainsi, à partir de "fontaines", on obtient "fontaine", et on détecte ainsi bien un mot en commun. Enfin, la présence ou l'absence de ponctuation (les tirets par exemple) peut impacter les résultats. Par exemple, "des etats unis" et "des etats-unis", sont deux noms de rues très similaires. Pour détecter des mots en communs, il faut ici supprimer le tiret et le remplacer par un espace. Les noms de rues étant fournis en majuscules dans le fichier FANTOIR, les majuscules / minuscules et les accents ne posent pas de problèmes.

Après avoir effectué ces pré-traitements sur les données, j'ai développé un algorithme de blocking qui regroupe dans un même bloc tous les noms partageant un mot donné. Le résultat est un dictionnaire de 363 693 blocs dont les clés sont des mots (lemmatisés) et les valeurs associées sont des listes de noms de rues (non raccourcis et non lemmatisés).

2.3 Détection de correspondances entre les noms de rues

La détection des noms de rues équivalents se base sur le dictionnaire généré pendant la phase de blocking. Des algorithmes plus sophistiqués vont comparer uniquement les noms appartenant à un même bloc et détecter, par exemple, que "avenue Marie Curie" et "rue Marie Skłodowska-Curie" correspondent, tandis que "avenue Marie Curie" et "voie Pierre Curie" ne représentent pas la même entité.

Afin de calculer la similarité entre deux noms de rues, j'ai testé plusieurs algorithmes de similarité dans le but de définir le meilleur pour les besoins de ce projet. Chaque mesure retourne un score normalisé entre 0 (aucune similarité) et 1 (ressemblance totale). La première mesure de similarité est la distance de Levenshtein [4]. Elle se base sur le nombre minimal de caractères qu'il faut enlever / insérer / remplacer pour passer d'une chaîne de caractères à une autre. Pour cela, j'ai utilisé une fonction du module python Distance[3]. Un paramètre permet de sélectionner le facteur de normalisation : soit la longueur de la plus grande chaîne de caractère (levenshtein 1), soit celle de la plus petite (levenshtein 2). La deuxième mesure utilisée est la distance de Jaro-Winkler [5]. Elle se base sur la somme des caractères correspondants d'une chaîne de caractère à une autre. La troisième mesure est la distance de Jaccard. Elle exploite le rapport entre la taille de l'intersection des chaînes de

2 Spacy-leff, module python permettant la lemmatisation française
3 Distance, module python permettant le calcul de similarité entre des chaînes de caractères

caractères et la taille de leur union. J'ai utilisé deux variantes de cette mesure : jaccard 1 compare uniquement les caractères tandis que jaccard 2 compare des bi-grams (séquences de 2 caractères). Des exemples avec chacune de ces mesures sont donnés en Annexe 2.

De manière générale, lorsque l'on compare des entités, on combine plusieurs mesures afin de pallier les faiblesses de certaines (ex, peu adaptée à de petites chaînes) ou pour exploiter différentes propriétés. Une combinaison courante est une moyenne pondérée des scores de similarité calculés par différentes mesures. Le poids associé à une mesure est plus ou moins élevé en fonction de sa fiabilité, de son efficacité ou de sa pertinence dans le contexte. Je n'ai combiné que deux mesures en guise de tests, mais dans l'idéal, d'autres combinaisons devraient être testées. Cependant, l'utilisation d'un grand nombre de mesures a un impact négatif sur les performances. Décrivons maintenant les résultats obtenus pour la déduplication.

2.4 Expérimentations pour la déduplication

Dans cette section, je décris et analyse les résultats de mes expérimentations sur la déduplication.

2.4.1 Protocole d'expérimentation

Afin de mener les expérimentations à bien, plusieurs outils ont été utilisés. Deux expertises ont été créées. La première est précise, vérifiée manuellement, et basée sur un jeu de données restreint. La seconde est estimée, créée à partir des résultats précédents, et basée sur un jeu de données un peu plus grand. Le jeu de données utilisé varie donc. Pour l'évaluation de l'expertise précise et celle du blocking, l'échantillon de données utilisé inclut 26 communes autour de Lyon pour un total de 4603 voies. Pour l'évaluation de l'expertise estimée, l'échantillon utilisé inclut toutes les communes du département du Rhône, ce qui représente 27 273 noms de rues. Afin d'évaluer les résultats, je me base sur plusieurs mesures : la f-mesure, la précision et le rappel. Ces mesures impliquent l'utilisation de vrais positifs (TP), faux positifs (FP), vrais négatifs (TN) et faux négatifs (FN). Pour plus de mise en contexte, des exemples sont données en Annexe 3 pour ces termes. Ci-dessous les formules correspondantes aux mesures:

$$\text{précision} = TP / (TP + FP)$$
$$\text{rappel} = TP / (TP + FN)$$
$$\text{f-mesure} = 2 * \text{précision} * \text{rappel} / (\text{précision} + \text{rappel})$$

2.4.2 Évaluation avec une expertise précise

Afin d'évaluer l'algorithme et les résultats de chacune des mesures, j'ai construit un fichier d'expertise (dite "précise") sur un échantillon limité de données (26 communes pour 4603 voies). L'expertise contient 847 entrées (c'est-à-dire, 847 paires de noms de rues que j'estime équivalents, par exemple, "Ch de Gaulle" et "de Gaulle"). Afin qu'il soit le plus précis possible, ce fichier est réalisé en vérifiant manuellement les correspondances. Cela est faisable de part la taille limitée de l'échantillon.

L'objectif de cette expérimentation est d'évaluer chacune des mesures de similarité sur cette expertise exacte. La difficulté est de choisir une valeur seuil (au-dessus de laquelle deux rues sont considérées comme équivalentes) qui limite le taux d'erreur. Ici j'ai parcouru tous les seuils possibles (entre 0 et 1) avec un pas de 0,01 afin d'obtenir le meilleur résultat (en termes de f-mesure) pour chaque algorithme. Les résultats sont montrés dans le tableau de l'Annexe 4.

Au vu des résultats, je conclus que la meilleure mesure pour cette application est celle de jaccard2 (avec les bigrams), qui obtient la meilleure f-mesure (0.89), le meilleur rappel (0.84) et une très bonne précision (0.95). Globalement, la plupart des mesures obtiennent une précision élevée (donc peu de correspondances erronées sont détectées). L'algorithme jaccard2 se distingue par un rappel plus élevé que les autres (donc détecte davantage de correspondances de l'expertise). On

remarque également que les performances des mesures de levenshtein et de Jaro Winkler sont très similaires, et que la mesure jaccard1 semble la moins performante ici.

Il est fréquent en intégration de données de combiner des mesures de similarités afin d'améliorer les résultats et de contrebalancer les éventuels points négatifs d'une mesure. J'ai testé une combinaison de jaccard 2 et de levenshtein 1 (deux mesures assez différentes dans leur manière de fonctionner), avec chacune un poids de 0.5. En termes de résultats, la f-mesure et le rappel trouvés restent proches de ceux de levenshtein, et le nombre de résultats trouvé est le plus petit entre toutes les mesures. La précision, quant à elle, est très bonne (0.98), meilleure que chacune des mesures individuelles. Cela indique bien qu'il y a une éventualité d'amélioration des résultats avec des combinaisons de mesures. J'en conclus que cette combinaison n'est pas plus performante que la mesure de jaccard 2, mais la combinaison de mesures, idéalement avec des configurations différentes (poids et seuils) nécessiterait davantage de travaux.

Pour évaluer la cohérence des résultats, j'ai également implémenté un algorithme permettant de regrouper les noms de rues associés ensemble. Par exemple, si le couple "Ch de Gaulle", "de Gaulle" est considéré comme équivalent, ainsi que le couple "Ch de Gaulle", "Charles de Gaulle", un groupe ["Ch de Gaulle", "de Gaulle", "Charles de Gaulle"] sera formé. Ainsi, j'ai pu estimer visuellement la cohérence des résultats (519 groupes, avec 7 noms maximum par groupe sur cette expertise précise). Les groupes trouvés sont effectivement cohérents (extrait des premiers groupes en Annexe 5).

2.4.3 Évaluation avec une expertise estimée

L'évaluation avec un échantillon plus grand implique une expertise plus grande. Or, il est fastidieux de tout vérifier à la main, donc une autre solution doit être utilisée, quitte à avoir un expertise moins précise. Afin de confirmer ces résultats à une échelle plus grande, j'ai donc construit une expertise "estimée", en utilisant un échantillon de données plus large (toutes les communes du Rhône, c'est-à-dire 27 273 noms de rues) et en me basant sur les résultats de l'expertise précise. L'hypothèse est que les rues détectées comme équivalentes avec un seuil très élevé sont en majorité des correspondances correctes (parmi les plus faciles à détecter). Pour construire cette expertise, j'ai cherché, pour chaque mesure, le seuil à partir duquel 99% de précision est atteint, (seuils disponibles en Annexe 6). J'ai ensuite lancé les algorithmes de déduplication sur les communes du Rhône. Pour chacun, j'ai ajouté dans l'expertise estimée les couples de rue dont la mesure de similarité était supérieure au seuil respectif trouvé (seuil pour les 99%). Finalement, j'ai cherché les résultats en termes f-mesures, précision et rappel en appliquant les seuils trouvés précédemment (ceux de l'Annexe 4, seuils permettant d'avoir la meilleure f-mesure sur l'expertise précise). Ces résultats sont visibles en Annexe 7.

L'expertise estimée contient 10264 correspondances. Une vérification manuelle rapide (notamment des "moins similaires") a mis en évidence une qualité élevée de cette estimation. Dans l'ensemble, on peut remarquer que les précisions diminuent et que les rappels augmentent, ce qui fait au final augmenter la f-mesure.

La précision correspondant à la proportion de couples estimés à juste titre comme correspondances, et le rappel étant la proportion de couples non estimés comme correspondant alors qu'ils le sont, il est logique que l'un baisse quand l'autre augmente dans ce cas. Avec 99%, le seuil trouvé est haut, donc on ajoute peu de couples dans l'expertise. Le rappel est donc élevé, car étant donné que peu de couples ont été ajoutés dans l'expertise (seulement ceux avec une forte mesure de similarité), les couples sont facilement trouvés. A l'inverse, la précision diminue car trop de couples trouvés ne sont pas représentés dans l'expertise estimée. Les scores de précision sont donc des valeurs minimales : certaines mesures ont probablement détecté des paires de rues véritablement équivalentes, mais qui n'ont pas forcément été mises dans l'expertise estimée, ce qui contribue à diminuer la précision.

L'objectif de cette expérimentation était d'évaluer si les mesures obtenaient des résultats cohérents sur de plus grands échantillons, et c'est bien le cas.

2.4.4 Évaluation du blocking

L'étape de blocking étant faite pour gagner en temps, il a fallu vérifier son efficacité. J'ai donc mesuré les performances de la déduplication avec et sans blocking. L'impact sur la qualité est également étudié, puisque le blocking tend à la diminuer (deux entités qui ne sont pas dans le même bloc ne pourront pas être détectées comme correspondantes). Je réutilise l'expertise précise (4603 rues, pour 847 correspondances). Les seuils utilisés sont les seuils permettant la meilleure f-mesure dans chacun des cas, avec blocking. Le tableau de l'Annexe 8 présente les résultats obtenus.

J'en conclus tout d'abord qu'au niveau du temps d'exécution, l'algorithme de blocking est très efficace. Avec le blocking, il faut environ 30 secondes pour faire le blocking et la déduplication alors que sans, il faut plusieurs minutes, soit une exécution 10 à 30 fois plus rapide.

Notons que les seuils "avec blocking" ont été réutilisés pour les expérimentations sans blocking, ils ne sont donc pas optimaux (c'est-à-dire, la version sans blocking peut obtenir de meilleurs résultats avec des seuils plus élevés) mais permettent d'évaluer l'utilité du blocking. L'absence de blocking ajoute principalement des correspondances entre des noms de rues courts dont un mot contiendrait une ou plusieurs lettres différentes. Par exemple: "les condamines" et "les contamines", ou "charriere" et "la carriere". Ces noms sont effectivement similaires au niveau lexical, mais pas au niveau de leurs significations. Cela a un impact négatif sur la précision. Le rappel sans blocking est forcément supérieur ou égal : on constate ici que la version avec blocking affecte peu les résultats sur ce point. Cela signifie que le regroupement en blocs n'a pas d'impact significatif sur la qualité, mais permet d'améliorer grandement les performances.

3 Classification

Dans la section précédente, le jeu de données FANTOIR a été nettoyé en supprimant les doublons. L'objectif consiste désormais à associer une catégorie ou classe à chaque nom de rue. Pour l'atteindre, il faut d'abord définir une taxonomie (c'est-à-dire, des catégories organisées) puis prédire la catégorie pour chaque nom de rue. Comme mon approche est basée sur l'apprentissage supervisé, j'ai créé un jeu de données expertisé contenant des instances déjà classées. Je détaille ensuite mon approche pour la prédiction et présente les résultats obtenus dans la dernière partie.

3.1 Définition de la taxonomie

Suite à l'étude des données, et en m'inspirant de la taxonomie de Badariotti [1] , j'ai établi ma propre taxonomie, exposée en Annexe 9. Elle contient 13 classes, dont 4 qui sont divisées en sous-classes (de 8 à 18). Cette taxonomie présente des différences par rapport à celle de Badariotti. Les bâtiments et les lieux sont ici beaucoup plus répartis (beaucoup plus de sous-classes). Les dates quant à elles ne sont pas sous divisées, alors que Badariotti les avait divisées en différents types d'événements plus ou moins lointains. Des catégories ont été ajoutées par rapport à Badariotti, telles que les animaux, les professions ou les objets.

3.2 Création de l'expertise pour l'apprentissage

Comme mentionné précédemment, afin de faire de l'apprentissage supervisé, un fichier d'expertise est nécessaire. Pour le créer, j'ai dans un premier temps exploité Wikidata[4], une base de connaissances externe. Pour cela, j'ai implémenté un algorithme permettant, à partir d'un nom de rue sélectionné aléatoirement, d'essayer d'obtenir son type. À partir d'une recherche, on peut obtenir

4 Wikidata, base de connaissances externes

plusieurs types: un par résultat (s'il y a plusieurs résultats). Par exemple, avec “de gaulle”, Wikidata renvoie des types tels que "être humain” ou “militaire” ou “nom de famille”.

La difficulté est d'associer les types de wikidata aux classes de la taxonomie. Wikidata renvoie des types parfois très précis. Par exemple, “cheval”, “chèvre”, “chien”, etc, mais pas “animal”. Au contraire, il renvoie parfois des types plus abstraits ou généraux, le plus commun étant “taxon”, qui peut référer soit à un animal (mammifère comme insecte), soit à une plante. Un autre problème est que parfois, le type renvoyé est “page d'homonymie de Wikimedia”, ce qui signifie que la recherche correspond à plusieurs choses, mais aucun type précis n'est donné. Dernièrement, l'API prend parfois la liberté de modifier la requête d'origine afin d'obtenir un résultat, et le type renvoyé est donc erroné. Par exemple, la recherche de “beauvet” retourne le type “colline”. En regardant de plus près, on peut s'apercevoir que ce type est en fait associé au résultat “Beauvette Hill”, et non pas au nom de famille “beauvet”.

Pour contrer ces problèmes, j'ai constitué un dictionnaire de types dont un extrait est disponible en Annexe 10. Ses clefs sont les classes de la taxonomie, et les valeurs associées sont des listes de types que renvoie wikidata (et de même pour les sous-classes). J'ai donc constitué ce dictionnaire en lançant des recherches wikidata et en le remplissant peu à peu avec les types trouvés. Mais on se doute bien que ce dictionnaire ne sera jamais exhaustif. Au final, je me suis rendue compte que wikidata faisait quand même des erreurs, même avec ce dictionnaire. La majorité de l'expertise a donc dû être vérifiée manuellement. Même en classifiant à la main, des problèmes sont vite rencontrés, car beaucoup de noms de rues ont des significations que je ne connais pas. Classifier uniquement les noms dont je comprends la signification apporterait un biais à l'expertise, mais chercher la signification de tous les noms de rues peut vite devenir fastidieux. L'automatisation de la classification avec wikidata, comme la classification à la main, sont donc assez difficiles. C'est pour cela que le nombre d'entrées dans l'expertise est limité.

Finalement, afin de rajouter des entrées de manière efficace, j'ai ré-utilisé le blocking de la section précédente en le lançant sur toute la France. J'ai ainsi pu récupérer les mots les plus représentés, et ajouter des instances de noms contenant ces mots dans l'expertise. Par exemple, le mot le plus représenté est “fontaine”. Les noms de rue contenant ce mot réfèrent généralement à un bâtiment, donc il est facile de trouver des instances et de savoir comment les classer. Cependant, cette technique ne peut pas permettre d'ajouter massivement des exemples classifiés pour quelques termes sélectionnés, car cela peut favoriser l'overfitting des algorithmes d'apprentissage.

Le fichier d'expertise pour l'apprentissage résultant contient 2323 entrées, dont environ 1100 proviennent de wikidata, et environ 1200 qui ont été ajoutées manuellement. La répartition dans les différentes classes est montrée sur le tableau en Annexe 11. Quatre classes incluent plusieurs centaines d'exemples, tandis que les autres sont limitées à quelques dizaines. Aucune rue mentionnant une valeur (e.g., égalité, liberté) n'a été sélectionnée (faible présence) donc cette classe ne sera plus utilisée par la suite.

3.3 Prédiction

Pour prédire, j'ai d'abord identifié des critères intéressants, puis j'ai développé une approche utilisant différentes façons de classifier.

3.3.1 Les critères

Le fichier d'expertise pour l'apprentissage contient 18 critères. Ces critères sont de plusieurs types. Certains sont purement lexicaux, tels que le nombre de caractères ou le nombre de mots dans le nom de la rue. Certains sont tirés du fichier de la base de données (et donc du fichier FANTOIR) tels que la nature de la voie ou le caractère rural. Similairement, certains ont été créés à partir de la base de données, comme le nombre d'occurrences du nom de la rue dans la base. Certains ont été créés grâce

au fichier de blocking (voir section II), comme le nombre de fois où apparaît un mot du nom de la rue dans la base de données. Finalement, certains viennent de sources extérieures, comme la longueur de la voie (physique). Plus de détails sur les critères sont fournis en Annexe 12.

Un pré-traitement est également appliqué aux données. Les noms de rues passent dans un pipeline qui leur applique 2 estimateurs tirés de la bibliothèque Python scikit-learn[5]. Le premier est CountVectorizer, qui permet de compter les n-grams ou les caractères successifs, et renvoie un tableau. Le deuxième est TfidfTransformer, qui est appliqué sur le dictionnaire renvoyé par CountVectorizer afin de calculer les fréquences. Ce prétraitement peut être vu comme un critère supplémentaire. Les critères numériques (en bleu sur l'Annexe 12) sont passés dans un pipeline qui leur applique 3 estimateurs : SImpleImputer (qui permet de compléter les valeurs manquantes), StandardScaler (pour standardiser les valeurs) et MinMaxScaler (pour mettre les valeurs à l'échelle entre 0 et 1). Enfin, les critères textuels sont passés dans un pipeline qui leur applique un unique estimateur : OneHotEncoder, qui permet de les encoder proprement. Pour plus de clarté, ce prétraitement est schématisé en Annexe 13.

3.3.2 Méthodes pour la classification

Pour classifier les noms de rues, il y a plusieurs possibilités. On peut utiliser une méthode de classification générale, qui affecte une classe (ou une sous-classe) à chaque nom de rue. Par exemple, la rue "De Gaulle" sera classée en "personne". Il est également possible d'utiliser une méthode qui retourne un résultat binaire pour chaque classe. Par exemple, la rue "De Gaulle" peut être catégorisée comme appartenant à la classe "personne" et n'appartenant pas à la classe "animal". Si plusieurs classifiers binaires classent positivement la même rue, alors la rue est considérée comme multi-classe. Un algorithme doit donc décider de la classe la plus pertinente. Enfin, on peut combiner une classification binaire avec une classification générale. Le tableau en Annexe 14 présente mes 4 approches pour la classification avec leurs caractéristiques. On note qu'il n'y a pas de méthode de classification binaire pour les sous-classes car cela nécessiterait un temps de calcul conséquent (génération d'une quarantaine de classifiers) et générerait probablement de nombreuses ambiguïtés (odonyme associé à plusieurs catégories). La méthode combi_classes utilise des classifiers binaires pour les classes avec le plus d'instances (bâtiment, personne, lieu, nature, date) de façon séquentielle (i.e., une rue classée positivement dans une classe ne sera plus testée par les classifiers suivants). Si aucun classifieur binaire n'a retourné de résultat positif, alors un classifier général est utilisé pour choisir la classe parmi celles restantes. Cette méthode permet de prendre en compte le déséquilibre d'instances entre les classes (notamment pour les classes avec peu d'exemples, pour lesquelles un classifieur binaire peut ne pas être très fiable). Un résumé de cette approche est montré en Annexe 15.

Quelque soit l'approche utilisée, différents classifiers tirés de la bibliothèque Python scikit-learn peuvent être utilisés : LinearSVC (Linear Support Vector Classification, basée sur la maximisation de la marge), SGDClassifier, AdaBoostClassifier, MultinomialNB, KNeighborsClassifier, SVC, DecisionTreeClassifier et RandomForestClassifier.

3.4 Expérimentations pour la classification

La métrique souvent utilisée en classification est l'exactitude (accuracy) :

exactitude = nombre de prédictions correctes / nombre de prédictions total

Le cas de la classification binaire ajoute également les notions de vrais positifs (TP), faux positifs (FP), vrais négatifs (TN) et faux négatifs (FN). Dans ce cas :

exactitude = TP + TN / (TP + TN + FP + FN)

5 Scikit-learn, module python pour le machine learning

Les meilleurs résultats obtenus sont observés lors de la classification avec la méthode bin-classes (binaire sur les classes), avec LinearSVC, dont les résultats sont présentés en Annexe 16. On distingue des classes qui se catégorisent très bien : lieu, nature, bâtiment, personne et date. Ces classes obtiennent une bonne exactitude, un nombre de TP élevé, et des nombres de FN et FP assez bas. Les FP, notamment, étant assez bas, impliquent qu'un nom de rue sera rarement classé à tort dans la classe étudiée. Ces 5 classes sont par ailleurs les classes les plus représentées, et seront utilisées comme telles avec la méthode combi_classes. Les autres classes, bien qu'elles aient une bonne exactitude, ont 0 (ou proche de 0) TP. Leur bon score est en fait lié au nombre élevé de TN. Ces classes coïncident également avec les classes les moins représentées. Il faut noter que bien que les résultats soient les meilleurs, cette méthode amène un problème de décision au moment de la classification (un nom de rue peut être classé dans plusieurs classes).

La prédiction avec la méthode gen-classe, (c'est-à-dire, celle qui permet d'obtenir la bonne classe), fournit une prédiction d'exactitude 0.72, avec LinearSVC. Les résultats avec les autres classifieurs sont disponibles en Annexe 17. Si on exclut les classes les moins bien prédites, on peut atteindre 0.77 d'exactitude.

La prédiction avec la méthode combi-classes permet quant à elle d'obtenir, avec LinearSVC, 0.66 d'exactitude, ce qui est donc moins satisfaisant que les résultats précédents.

La prédiction d'une sous-classe dans une classe (méthode gen-sc) donne des résultats variés selon la classe en question (voir Annexe 18). Les résultats pour les sous-classes de "lieu", "nature" et "bâtiment" sont assez satisfaisants (0.80, 0.81, et 0.84 d'exactitude respectivement). Les résultats pour les sous-classes de la classe "personne" ne sont cependant pas convaincants (0.55 d'exactitude). Mais cela se justifie par le fait qu'il y a beaucoup de sous-classes (18) par rapport aux autres classes, et que les sous-classes font référence au métier des personnes. Il est donc assez difficile de les prédire.

Globalement, les méthodes binaires amènent les meilleurs résultats, même si elles apportent également un problème de décision. Il faut également noter l'impact de la présence ou non de critères. Pour chaque méthode, les résultats changent peu si on ne garde qu'un critère (le nom de la rue). Cela implique donc que les critères utilisés ne sont pas assez performants. Un des points faibles est notamment la présence de critères avec des valeurs nulles (même si elles sont comblées lors du prétraitement). Cela peut influencer leur efficacité s'il y a trop de valeurs nulles.

Une perspective d'amélioration de la classification aurait été la classification en multi-classe. Ainsi, un nom de rue aurait pu être associé à plusieurs classes. Par exemple: "pont Jean-Moulin" pourrait être classé comme bâtiment et comme personne. Le fameux problème de décision rencontré avec les méthodes binaires pourrait ainsi potentiellement être contré par cette approche, étant donné qu'appartenir à plusieurs classes ne poserait pas problème.

4 Développement application

Le développement d'une application web permet de facilement visualiser la distribution des odonymes. La maquette originale est disponible en Annexe 19. Des captures d'écran du produit final sont disponibles en Annexe 20.

L'application propose plusieurs options : on peut choisir la région, le département et la commune sur lesquels on veut lancer la recherche. La classification serait trop longue si il y a trop de données, donc le bouton « rechercher » reste inactif tant que la zone géographique sélectionnée est trop large. On peut également choisir le classifier qui sera utilisé. En termes de résultats, on peut choisir sur le menu à gauche si on veut les résultats sur toutes les classes ou sur une classe spécifique.

Les résultats proposés sont peu nombreux. Sur la deuxième capture d'écran, on voit qu'on peut voir les classes les plus représentées dans une zone géographique. Il y a également des boutons « dropdown » permettant d'afficher les noms de rues classées dans chacune des catégories.

Pour trouver les résultats, les travaux des deux sections précédentes (déduplication et classification) sont réutilisés. Les données sont d'abord nettoyées grâce à une étape de blocking et une étape de déduplication. Lors de la déduplication, la meilleure mesure et son seuil permettant la meilleur f-mesure trouvés précédemment sont utilisés (c'est-à-dire, jaccard 2 avec un seuil de 0.63).

Ensuite, l'apprentissage est effectuer avec les données du fichier d'expertise créé pour la classification, puis, le classifier est utilisé pour prédire les classes des noms de rues du lieu donné. Les méthodes retenues pour la classification sont gen-classes si on demande des informations générales, et bin-classes si on demande des informations sur une classe spécifique. L'application n'est pas finalisée, il y a donc beaucoup d'axes d'amélioration sur ce point.

5 Conclusion

L'objectif initial de mon projet POM était l'analyse des noms de rues en France. J'ai pour cela, dans un premier temps, travaillé sur la déduplication des données à partir des données du fichier FANTOIR. J'ai ainsi pu définir la meilleure mesure de similarité à utiliser pour des noms de rues ainsi que la valeur seuil à partir de laquelle deux noms de rues peuvent être considérés comme équivalents. Cela m'a notamment permis de nettoyer la base de données. J'ai également développé un algorithme avec une étape de blocking très efficace pour réduire le temps d'exécution sans avoir un impact important sur la qualité des résultats. Dans un deuxième temps, j'ai travaillé sur la classification des données. Pour cela, j'ai défini ma propre taxonomie et construit un fichier d'expertise assez fastidieux à réaliser. J'ai ensuite pu prédire la catégorie de chaque nom de rues en utilisant l'apprentissage supervisé. La prédiction se base sur divers critères et permet l'utilisation de plusieurs approches et classifiers. Enfin, j'ai commencé à développer l'application web permettant la visualisation des résultats, mais je n'ai malheureusement pas eu le temps de la finaliser.

Dans l'ensemble, il y a plusieurs perspectives d'amélioration. Pour la déduplication, afin d'obtenir des résultats, il faudrait faire des tests en combinant différentes mesures avec des poids différents afin de potentiellement améliorer les résultats. Pour la classification, un système multi-classes pourrait permettre d'améliorer les problèmes de décision lors de l'utilisation d'une approche binaire. L'acquisition de critères plus performants pourrait également améliorer les performances de la classification.

6 Références

[1] Dominique Badariotti. Les noms de rue en géographie. plaidoyer pour une recherche sur les odonymes. In Annales de Géographie, t. 111, n°625, 2002., 2002

[2] Gérard Quantin. Les noms de rues de Reims. Revue internationale d'onomastique, 2(3):177–192, 1950. Lien

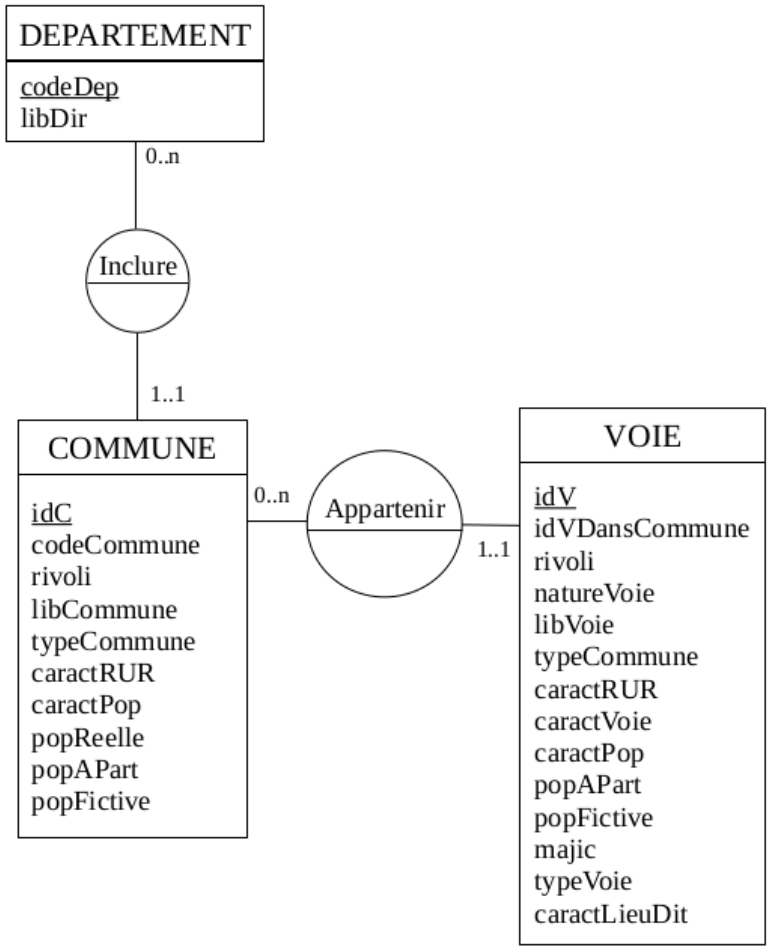
[3] Christen, P. (2012). The data matching process. In Data matching (pp. 23-35). Springer, Berlin, Heidelberg.

[4] Levenshtein, V.I. (1966) Binary Codes Capable of Correcting Deletions, Insertions and Reversals. Soviet Physics Doklady, 10, 707-710.

[5] Winkler, W. E., & Thibaudeau, Y. (1991). An application of the Fellegi-Sunter model of record linkage to the 1990 US decennial census. Washington, DC: US Bureau of the Census.

7 Annexes

7.1 Annexe 1 : Diagramme entité association de la base de données



7.2 Annexe 2 : Exemples de valeurs pour chaque mesure testée pour la déduplication, avec différents couples de noms de rues

Dans la base de données, les noms de rues sont donnés en majuscules, donc les accents ne sont pas pris en compte, ce qui explique pourquoi les noms des rues dans le tableau ci-dessous ne possèdent pas d'accents.

	"crx regis", "de la croix regis"	"pre moulin", "pre mutin"	"res les chataigniers", "gros chataigniers"
Levenshtein 1	0.8	0.8	0.68
Levenshtein 2	0.8	0.8	0.7
Jaro-Winkler	0.85	0.94	0.87
Jaccard 1	0.875	0.73	0.84
Jaccard 2	0.6	0.41	0.63

7.3 Annexe 3 : Détails sur les vrais positifs (TP), faux positifs (FP), vrais négatifs (TN) et faux négatifs (FN) dans le cadre de la déduplication des noms de rues

	Noms estimés comme équivalents	Noms estimés comme non équivalents
Noms équivalents ex : Ch de Gaulle, de Gaulle	TP (True positive, vrai positif)	FN (False negative, faux négatif)
Noms non équivalents ex : de Gaulle, du gaulois	FP (False positive, faux positif)	TN (True negatif, vrai négatif)

7.4 Annexe 4 : Résultats pour chaque mesure de similarité en utilisant les seuils qui permettent d'obtenir la meilleure f-mesure

	f-mesure	précision	rappel	seuil	nb résultats
levenshtein1	0.83	0.96	0.73	0.81	646
levenshtein2	0.83	0.96	0.73	0.81	647
jaroWinkler	0.82	0.96	0.72	0.95	640
jaccard1	0.78	0.83	0.74	0.87	761
jaccard2	0.89	0.95	0.84	0.63	751
levenshtein1 + jaccard2	0.84	0.98	0.72	0.72	630

7.5 Annexe 5 : Extraits du fichier regroupant les groupes de noms de rues créé avec l'expertise

Les noms d'un groupe sont tous équivalent. Chaque nom d'un groupe est considéré comme équivalent avec au moins un des autres noms du même groupe. Ce fichier a été testé pour vérifier si l'expertise avait bien été construite et que des équivalences ne se mélangeait pas.

```
DES GRANGES,LES GRANGES,DE LA GRANGE,GRANGE HTE,LA GRANGE ,DE LA GRANGE BASSE,DE GRANGE HAUTE
GRANDES TERRES,LES GRANDES TERRES,LES GDES TERRES,DES GRANDES TERRES,LA GRANDE TERRE,DE LA GRANDE TERRE
DES FONTAINES,LES FONTAINES,FONTAINE,DE LA FONTAINE,LA FONTAINE,DE FONTAINES
DE GRANDE VIGNE,GRANDE VIGNE,AUX GRANDES VIGNES,DES GDES VIGNES,LES GDES VIGNES
DU ONZE NOVEMBRE 1918,ONZE NOVEMBRE,DU ONZE NOVEMBRE,DU 11 NOVEMBRE 1918,DU 11 NOVEMBRE
CHAPULAY,GD CHAPULAY,CHAPULAY NORD,DE CHAPULAY,CHAPULAY SUD
DU GRAND CHAMP,GRAND CHAMP,GD CHAMP,GD CHAMP EST
DES COMBES,LES COMBES,DE LA COMBE,LA COMBE
DE LA CROIX,LES CROIX,DES CROIX,LA CROIX
GARE,LA GARE,DE LA GARE,LA GARE EST
DE LA COTE,LA COTE,COTE SUD,LES COTES
DU VILLAGE,LE VILLAGE,VILLAGE,VILLAGE NORD
DE LA GARENNE,LES GARENNES,DES GARENNES,LA GARENNE
DES VERCHERES,LES VERCHERES,DE LA VERCHERE,LA VERCHERE
CHATANAY,DE CHATANAY,DU CHATANAY,CHATANAY SUD
DES MURIERS,DU MURIER,AUX MURIERS,LES MURIERS
DE LA BALME,DES BALMES,LES BALMES,AUX BALMES
METRO LIGNE C,METRO LIGNE D,METRO LIGNE B,METRO LIGNE A
PETIT BOIS EST,PETIT BOIS OUEST,DU PETIT BOIS,LES PETITS BOIS
FORET SUD,LA FORET NORD,DE LA FORET,LA FORET
GDES TERRES EST,GDES TERRES OUEST,GDES TERRES,LA GDE TERRE
DU CHENE,LE CHENE,DES CHENES,AU CHENE
DE SAINT MARTIN,SAINT MARTIN,SAINT - MARTIN
DE LA PLAINE,LA PLAINE,PLAINE SUD
DE LA PLACE,LA PLACE,DES PLACES
DE L ILE,LES ILES,DES ILES
DU PUITS VIEUX,PUITS VIEUX,DU PUITS NEUF
DE LA CITE,DES CITES,LA CITE
BEL AIR,DU BEL AIR,DE BEL AIR
DES ROCHES,LA ROCHE,DE LA ROCHE
DE PLAMBOIS,PLAMBOIS,PLAMBOIS OUEST
DE VILLENEUVE,VILLENEUVE,VILLENEUVE NORD
DE LA FOUILLOUSE,LA FOUILLOUSE,FOUILLOUSE
DES SALINES,SALINE,DE SALINES
CRAPON,LE CRAPON,DE CRAPON
LE PILON,PILON,DU PILON
DES PACHOTTES,LES PACHOTTES,PACHOTTES
DES PAVILLONS,LE PAVILLON,DU PAVILLON
DE LA GRAVIERE,DES GRAVIERES,LA GRAVIERE
DE ROBELLY,ROBELLY,ROBELLY NORD
DE LALEAU,LALEAU,LALEAU NORD
DU BOUTAREY,LE BOUTAREY,LE BOUTAREY EST
DE BLANCHERIE,DE LA BLANCHERIE,LA BLANCHERIE
DE LA CHAPELLE,LA CHAPELLE,DES CHAPELLES
DES GAGERES,LES GAGERES,GAGERE
```

7.6 Annexe 6 : Seuils nécessaires pour atteindre au moins 99 % de précision pour chacune des mesures, et utilisés pour créer l'expertise estimée

	Seuil pour 99 %
levenshtein1	0.86
levenshtein2	0.86
jaroWinkler	0.98
jaccard1	1.00
jaccard2	0.78
levenshtein1 + jaccard2	0.79

7.7 Annexe 7 : Résultats pour chaque mesure de similarité, sur le département du Rhône, avec l'expertise estimée

L'expertise estimée est construire à partir des seuils permettant une précision de 99 % dans l'expertise précise (Annexe 6)

	f-mesure	précision	rappel	seuil	nb résultats
levenshtein1	0.91	0.88	0.94	0.81	11633
levenshtein2	0.88	0.82	0.94	0.79	12466
jaroWinkler	0.92	0.92	0.93	0.95	11006
jaccard1	0.80	0.68	0.97	0.87	15460
jaccard2	0.87	0.79	0.95	0.63	13039
levenshtein1 + jaccard2	0.94	0.94	0.94	0.72	10907

7.8 Annexe 8 : Comparaison des résultats de l'algorithme de déduplication avec et sans blocking, pour les 5 mesures

Les temps présentés dans le tableau ci-dessous, pour la partie « avec blocking », incluent à chaque fois le temps de blocking (environ 30 secondes) en plus du temps de déduplication.

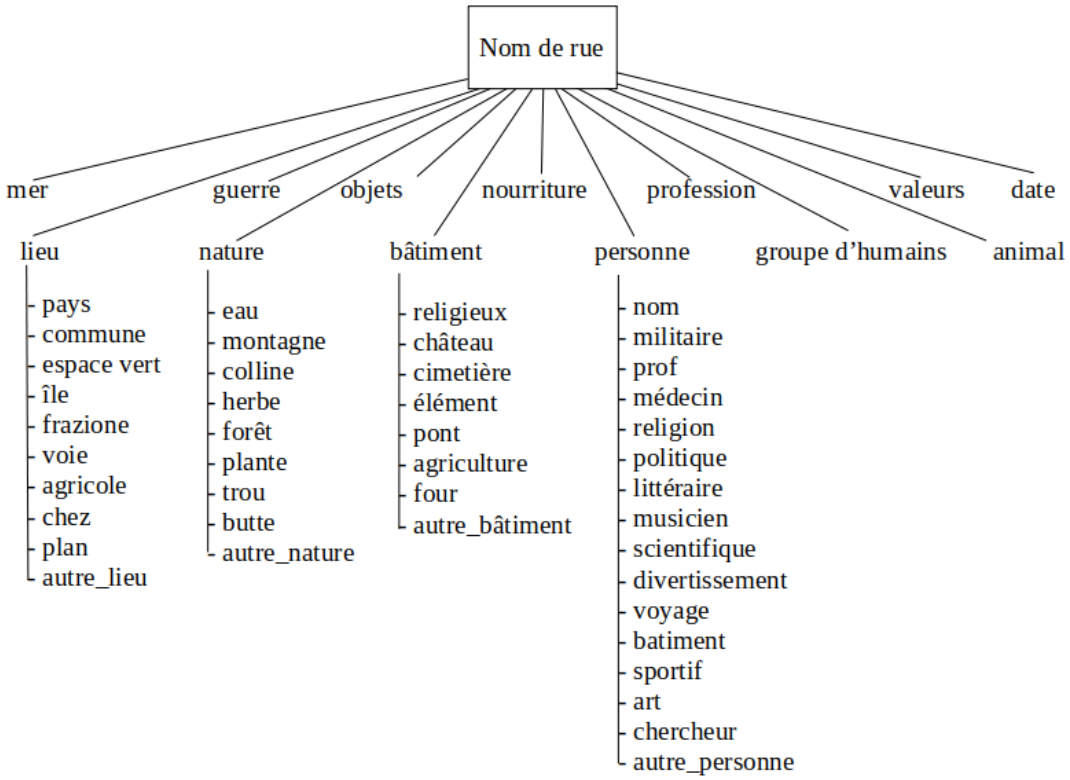
Avec blocking (22 503 comparaisons)

	f-mesure	précision	rappel	seuil	nb résultats	temps
levenshtein1	0.83	0.96	0.73	0.81	646	34.8 s
levenshtein2	0.83	0.96	0.73	0.81	647	36.3 s
jaroWinkler	0.82	0.96	0.72	0.95	640	33.6 s
jaccard1	0.78	0.82	0.74	0.87	761	33.1 s
jaccard2	0.89	0.94	0.84	0.63	751	33.2 s

Sans blocking (10 591 503 comparaisons)

	f-mesure	précision	rappel	seuil	nb résultats	temps
levenshtein1	0.65	0.58	0.73	0.81	1060	677 s
levenshtein2	0.65	0.58	0.73	0.81	1062	1037 s
jaroWinkler	0.67	0.63	0.72	0.95	981	337 s
jaccard1	0.42	0.29	0.75	0.87	2161	138 s
jaccard2	0.69	0.58	0.84	0.63	1212	170 s

7.9 Annexe 9 : Taxonomie utilisée



7.10 Annexe 10 : Extraits des dictionnaires des types de Wikidata associés aux classes et sous classes de la taxonomie

```
dico_types = {
    'personne': {
        "être humain": [["être humain"], []],
        "nom": [["nom de famille", "prénom féminin", "prénom masculin", "prénom"], []],
        "militaire": [["grade militaire"], []],
        "religion": [["dieu", "évêque catholique", "archevêque", "chef religieux"], []]
    },

    'batiment': {
        "religieux": [["croix monumentale", "abbaye","couvent", "monastère", "Église catholique",
                    "chapelle", "croix de chemin", "lieu mythique"], []],
        "château": [["château", "château fort", "fort", "fortification", "tour fortifiée"], []],
        "element": [["élément d'architecture"], []],
        "batiment": [
                        ["monument", "auberge", "cimetière", "gare", "raffinerie", "fontaine", "stade",
                        "pavillon", "rive", "puits à eau", "lycée", "gare ferroviaire", "port", "batiment", "usine"],
                        ["structure", "bâtiment", "architecture", "maison", "moulin"]
                    ]
    },

    'nature': {
        "plante": [["organe d'une plante", "bocage", "biome", "cépage"], ["plante"]],
        "montagne" : [["chaîne de montagnes", "montagne", "volcan", "sommet"], []],
        "eau" : [
                    ["rivière", "marais", "marais salant", "lac", "eau", "fleuve", "récif", "baie", "canal"],
                    ["'eau"]
                ],
        "colline" : [["colline"], []],
        "forêt" : [["forêt"], []],
        "nature" : [["campagne", "réserve naturelle", "réserve naturelle nationale", "aire protégée",  "vallée", "paysage",
                    "satellite naturel", "système planétaire"], []],
    },

    'lieu': {
        "ile" : [
                    ["terre émergée", "île", "archipel"],
                    ["commune", "municipalité", "lieu"]
                ],
        "pays" : [["pays"], []],
        "salle" : [["salle"], []],
        "espace vert" : [["parc"], ["jardin"]],
        "frazione" : [["frazione"], []],
        "village" : [["village"], []],
        "lieu" : [["ville", "quartier", "localité", "grande ville", "province de France", "région culturelle", "territoire dépendant",
                "domaine agricole"], []],
    },

    'voie': {
        "pont" : [[], ["pont"]],
        "voie" : [["route", "chemin", "estacade", "rue"], []],
    },
```

7.11 Annexe 11 : Utilisation de la taxonomie

Le tableau ci-dessous répertorie pour chaque catégorie de la taxonomie ses sous-classes, son nombre d'occurrences dans l 'expertise et le pourcentage d'entrées dans l'expertise qu'elle représente.

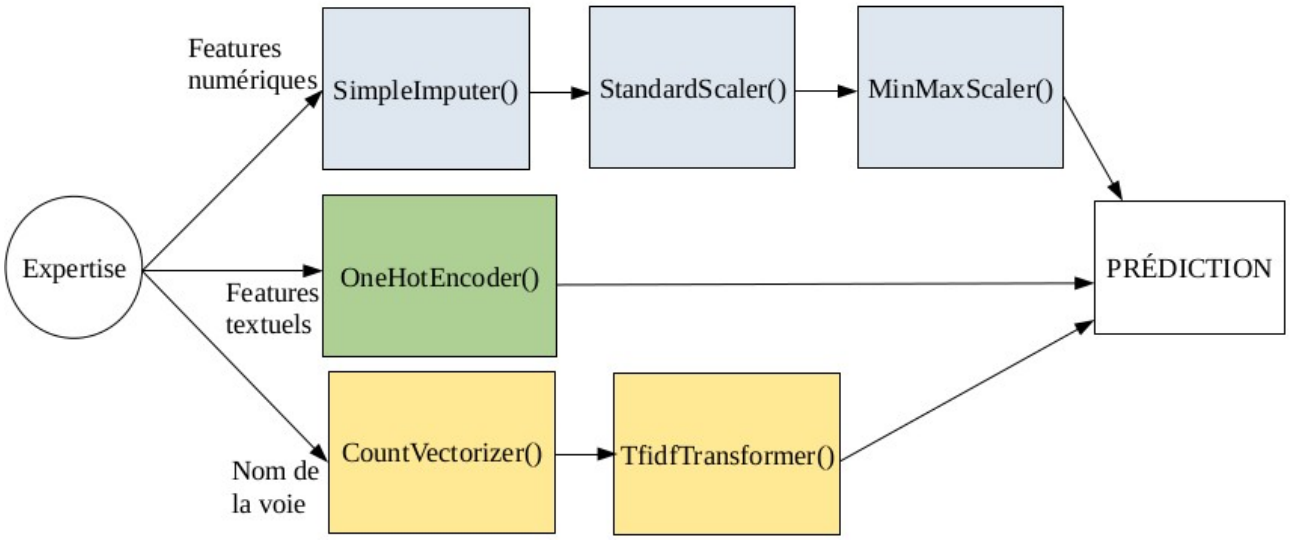
Classe	Sous classes	occurrences	%
lieu	10 (pays, commune, espace vert, île, frazione, voie, agricole, chez, plan, lieu)	507	21.8
nature	9 (eau, montagne, colline, herbe, forêt, plante, trou, butte, nature)	725	31.2
bâtiment	8 (religieux, château, cimetière, élément, pont, agriculture, four, bâtiment)	326	14
personne	18 (nom, miliaire, prof, médecin, religion, politique, littéraire, musicien, scientifique, divertissement, voyage, batiment, sportif, art, chercheur, autre, personne)	478	20.6
date		51	2.2
animal		56	2.4
profession		45	2
guerre		28	1.2
nourriture		14	0.6
objet		28	1.2
mer		21	0.9
groupe d'humains		21	0.9

7.12 Annexe 12 : Critères de l'expertise pour l'apprentissage

Les couleurs correspondent à celles des pré-traitements associés(bleu pour les critères numériques, vert pour les textuels et jaune pour le nom de la rue exploité avec TF/IDF).

Nom du critère	Explications	Valeurs possibles	Origine	Présence de valeurs nulles
nomRue	Nom de la rue	Chaîne de caractères	FANTOIR	non
nbChar	Nombre de caractères dans le nom de la rue	[0, n]	Calculs sur le nom de la rue	non
nbMots	Nombre de mots dans la nom de la rue	[0, n]	Calculs sur le nom de la rue	non
nbPetitsMots	Nombre de petits mots (taille < 3) dans le nom de la rue	[0, n]	Calculs sur le nom de la rue	non
nbGrandsMots	Nombre de grands mots (taille > 8) dans le nom de la rue	[0, n]	Calculs sur le nom de la rue	non
natureVoie	Nature de la voie	[rue, che, rte, av, ...]	FANTOIR	oui
caractRUR	Booléen pour si la voie est recensée ou non	[true, false]	FANTOIR	non
caractVoie	Booléen pour si la voie est privée ou non (publique)	[true, false]	FANTOIR	non
typeVoie	Type de la voie	[voie, ensemble immobilier, lieu-dit, pseudo-voie, voie provisoire]	FANTOIR	non
typeCommune	Booléen pour si la commune est recensée ou non	[true, false]	FANTOIR	non
caractLieuDit	Booléen pour si la voie est un lieu-dit bâti ou non	[true, false]	FANTOIR	non
occBD	Nombre de fois ou le nom exact de la rue est présent dans la base de données	[0, n]	Calculs sur la base de données	non
longMoy	Longueur de la voie. Les longueurs de toutes les voies ne sont pas accessibles, et certains noms de rues sont présent dans différent(e)s départements / communes. On a donc la longueur moyenne entre toutes les voies comportant le nom en question, la longueur minimale et la maximale.	[0, n]	Source externe	oui ~ 50 %
longMin				
longMax				
occMotsBdTot	Nombre de fois ou les mots présent dans le noms de la voie apparaissent dans la base (fait à partir du fichier de blocking sur la France). OccMotsBdTot et occMotsBdMoy : nombre de fois sommé / moyenne ou chaque apparaît. OccMotsBdMin et occMotsBdMax : nombre de fois ou le mot le moins / plus fréquent apparaît.	[0, n]	Fichier de blocking	non
occMotsBdMoy				
occMotsBdMin				
occMotsBdMax				

7.13 Annexe 13 : Prétraitements des données pour la classification



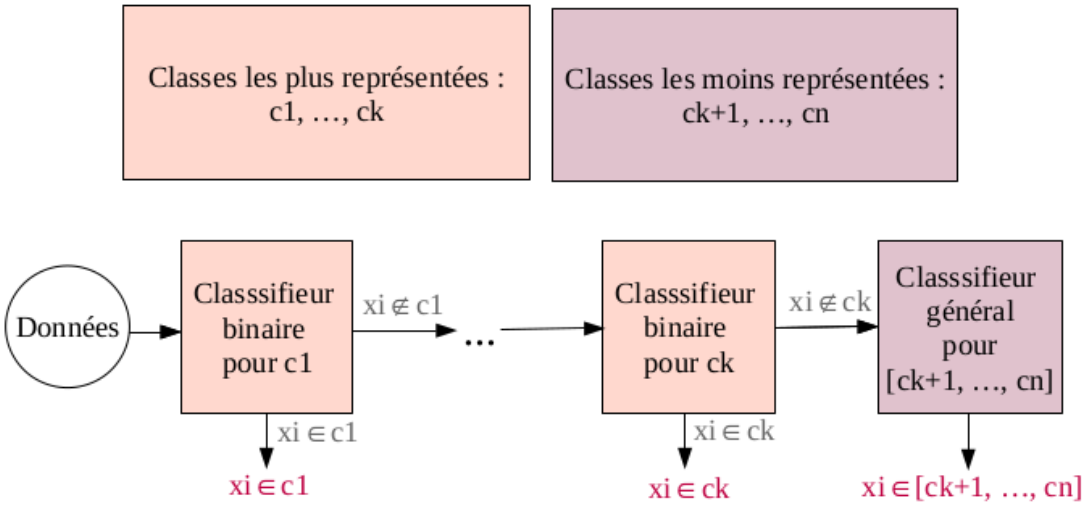
7.14 Annexe 14 : Les différentes approches utilisées pour la classification

Une approche binaire signifie qu'on cherche à savoir si un nom de rue appartient à une classe précise ou non (2 choix : oui ou non).

Une approche non binaire signifie qu'on cherche à savoir à quelle classe/sous-classe appartient un nom de rue (autant de choix que de classes / sous-classes).

Méthode	Non binaire (générale)	Binaire	Classe	Sous-classe
gen-classes	X		X	
bin-classes		X	X	
gen-sc	X			X
combi-classes	X	X	X	

7.15 Annexe 15 : Schéma représentant la méthode combi-classes



7.16 Annexe 16 : Résultats de la classification en binaire avec le classifier LinearSVC (méthode bin-classes)

	Accuracy	True positives	False positives	True negatives	False negatives
Lieu	0.91	65	4	359	37
Nature	0.91	109	6	313	35
Batiment	0.94	43	6	395	20
Personne	0.91	59	7	363	35
Date	1	9	0.1	454	1
Animal	0.98	0.26	10	454	2
Profession	0.98	0	0	456	8
Guerre	0.99	0	0	458	6
Nourriture	0.99	0	0	462	3
Objet	0.99	3	0	455	6
Mer	0.99	0	0	460	5
Groupe d'humains	0.99	0	0	461	4
Emotion	1	0	0	461	4
Valeur	1	0	0	463	1
Couleur	1	0	0	463	2

7.17 Annexe 17 : Résultats de la prédiction parmi toutes les classes avec différents classifiers (méthode gen-classes)

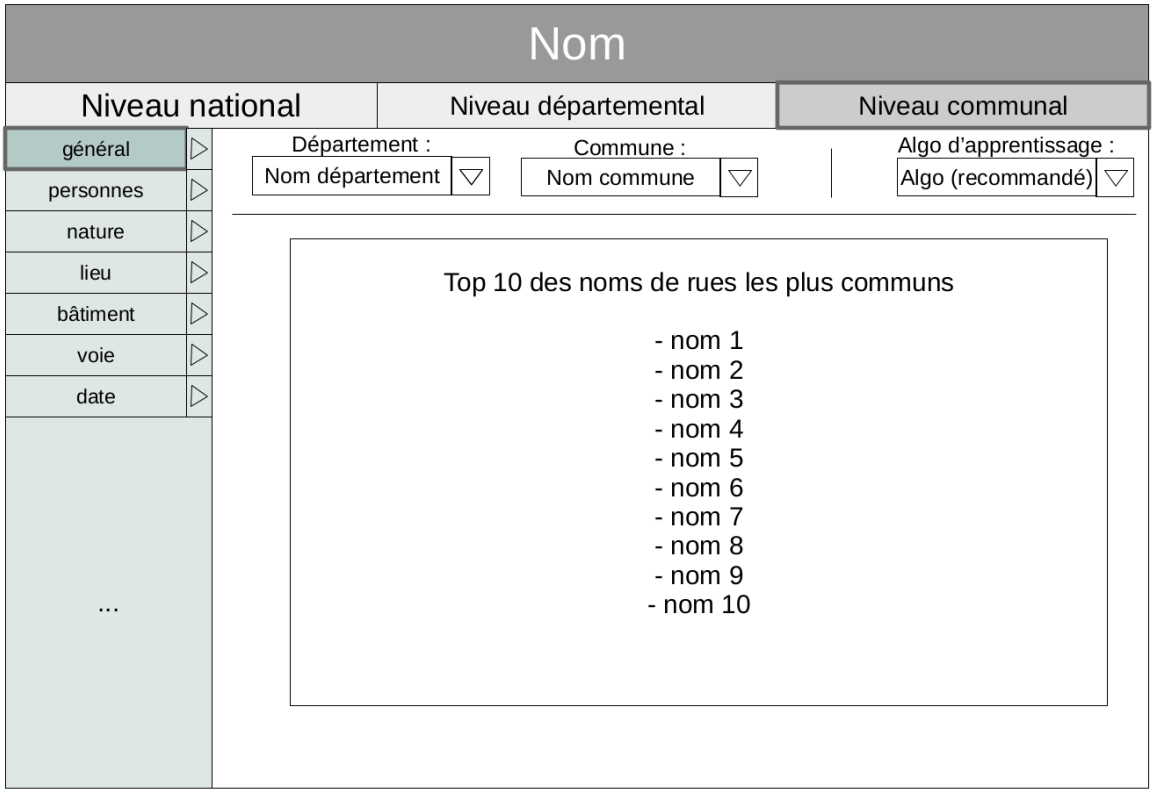
	Avec tous les critères	critères = [nomRue]
LinearSVC	0.72	0.72
SGDClassifier	0.66	0.71
AdaBoostClassifier	0.39	0.29
MultinomialNB	0.53	0.62
KNeighborsClassifier	0.59	0.63
SVC	0.62	0.7
DecisionTreeClassifier	0.66	0.54

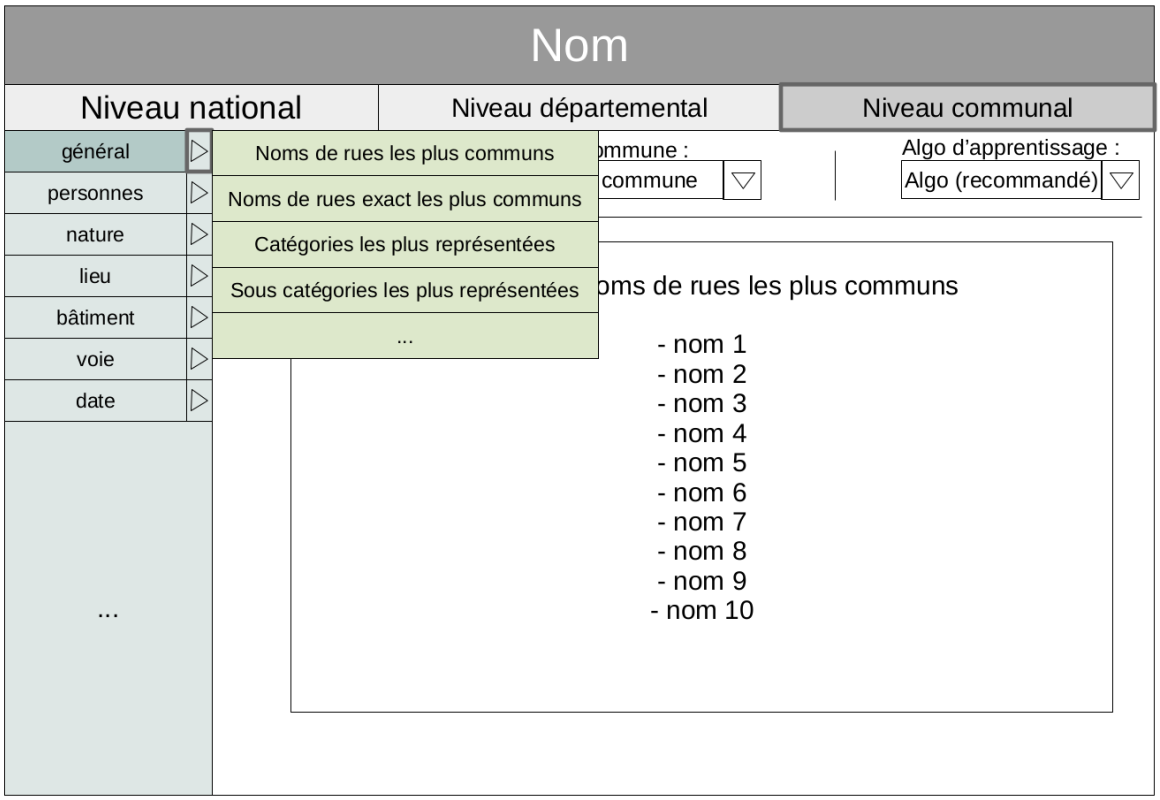
7.18 Annexe 18 : Résultats de la prédiction d'une sous-classe parmi une classe

Cette méthode est évidemment possible que pour les classes qui contiennent des sous-classes

	Accuracy	Positives	Negatives
Lieu	0.80	82	20
Nature	0.81	118	26
Batiment	0.84	54	10
Personne	0.55	65	52

7.19 Annexe 19 : Maquette de l'application web





7.20 Annexe 20 : Captures d'écran de l'application finale