Chapitre 1

Identification des demandes

Résumé. *Ce chapitre aborde le problème d'identification automatique, dans une décision, des éléments structurants les demandes formulées. L'identification manuelle réalisée à travers la lecture exige beaucoup d'effort à cause de la complexité du contenu dans lequel les demandes sont mélangées à d'autres informations (des demandes de nature différente, des arguments, des faits, etc.). L'automatisation de cette tâche métier vise à aider les experts à rapidement comprendre les réclamations des parties et les réponses correspondantes des juges. Une demande est abstraite par cinq attributs : la norme qui la fonde, son objet, l'interprétation du résultat (s_r), le quantum demandé (q_d), et celui obtenu (q_r). La norme et l'objet forment ensemble la catégorie de la demande. L'annotation manuelle des données d'évaluation suit un protocole que nous avons précisément défini avec l'expert du projet. Ce protocole recommande des cycles d'annotation consistant chacun à constituer un ensemble de décisions et à y identifier toutes les demandes d'une seule catégorie donnée car il serait difficile d'annoter simultanément des données pour toutes les catégories qui sont très nombreuses. L'approche proposée extrait à chaque application les demandes d'une seule catégorie et est formulée en trois tâches. La présence de la catégorie est déterminée par classification de la décision. Ensuite, les quanta et le sens du résultat sont identifiés à proximité de termes appris de la catégorie dans les sections adéquates identifiées à l'aide d'un modèle à base de CRF comme décrit au chapitre* **??**. *Enfin, les demandes sont formées en mettant en correspondance les éléments précédemment déterminés. Réalisées par validation croisées, nos expérimentations comparent une douzaine de méthodes statistiques d'extraction de termes-clés et quatre algorithmes de classification pour l'extraction de 6 catégories prédéfinies de demandes. Les résultats montrent que la détection de catégorie est facile quelque soit l'algorithme utilisé (F_1-mesure comprise entre 98.8 % et 100 %). Il résulte aussi que l'extraction des demandes nécessite de sélectionner la méthode d'extraction de terminologie la mieux adaptée à la catégorie. Cette sélection préalable permet d'observer, sur les données de test, des F_1-mesures comprises entre 33.09 % et 71.43 % pour les champs q_d, q_r, et s_r, et entre 28.65 % et 58.99 % pour les triplets (q_d, s_r, q_r).*

1.1 Introduction

Au cœur de l'analyse des décisions de justice se trouve le concept de demande. Il s'agit d'une réclamation ou requête effectuée par une ou plusieurs parties aux juges. Une partie peut demander des dommages-intérêts en réparation d'un préjudice subi ou à l'issu d'un divorce, des indemnités auxquelles elle pense avoir droit, ou encore une étude d'expert, etc. Les demandes sont fondamentales car l'argumentation au cours d'une affaire a deux buts : faire accepter ses demandes, et faire rejeter celles de la partie adverse. L'extraction des demandes et des résultats correspondants, dans un corpus, permet ainsi de récolter des données informant de la manière dont sont jugés des types de demandes d'intérêt. Les informations qui nous intéressent sont la catégorie de la demande, le quantum (montant) demandé, le sens du résultat (par ex. la demande a-t-elle été acceptée ou rejetée ?), et le quantum obtenu (décidé par les juges). Pour pouvoir extraire les demandes et les résultats, il est nécessaire de comprendre comment ceux-ci sont exprimés et co-référencés dans les décisions jurisprudentielles. Leur énoncé peut comporter des expressions plus ou moins complexes, dont souvent des références à des jugements antérieurs, des agrégations ou des restrictions (Figure 1.1 Page 3).

1.1.1 Données cibles à extraire

1.1.1.1 Catégorie de demande

Une catégorie c de demande regroupe les prétentions qui sont de même nature par le fait qu'elles partagent deux aspects : l'objet demandé (par ex. dommages-intérêts, amende civile, déclaration de créance) et le fondement c'est-à-dire les règles ou normes ou principes juridiques qui fondent la demande (par ex. article 700 du code de procédure civile). Des noms particuliers sont utilisés pour identifier les catégories (Tableau 1.1).

1.1.1.2 Sens du résultat

Le sens du résultat est l'interprétation de la décision des juges sur une demande. Nous le notons s_r. En général, le sens peut être positif si la demande a été acceptée, et négatif si elle a été rejetée. Il arrive aussi que le résultat soit reporté à un jugement futur ; il s'agit dans ce cas d'un sursis à statuer.

Jennifer M., Catherine M. et Sandra M. ... demandent à la Cour de :
- les recevoir régulièrement appelantes incidentes du jugement du 23/05/2014 ;
- infirmer le dit jugement en toutes ses dispositions ; ...
Statuant à nouveau ...
- les condamner au paiement d'une somme de 3 000,00 € pour procédure abusive et aux entiers dépens ;

(a) Exemples d'énoncés de demandes

La cour, ...
CONFIRME le jugement entreprise en toutes ses dispositions.
Y ajoutant
CONSTATE que Amélanie Gitane P. épouse M. est défaillante à rapporter la preuve d'une occupation trentenaire lui permettant d'invoquer la prescription acquisitive de la parcelle BH 377 située [...].
DEBOUTE Amélanie Gitane P. épouse M. de sa demande en dommages et intérêts.
CONDAMNE Amélanie Gitane P. épouse M. aux dépens d'appel.
DIT n'y avoir lieu à l'application de l'article 700 du Code de Procédure Civile.

(b) Exemple d'énoncés de résultats

Source : *extraits de la décision 14/01082 de la cour d'appel de Saint-Denis (Réunion).*
Légende : *énoncés simples en gris, références en bleu, et agrégations en marron.*

Figure 1.1 – Illustrations de la complexité des énoncés de demandes et de résultats.

1.1.1.3 Quantum demandé

Le quantum demandé quantifie l'objet de la demande. Nous le notons q_d. Par exemple, dans l'exemple de la Figure 1.1a, "3000 €" est le quantum demandé au titre des dommages-intérêts pour procédure abusive. Bien que cette étude ne porte que sur des sommes d'argent, le quantum peut être d'une autre nature comme par exemple une période dans le temps (garde d'enfant, ou emprisonnement, etc.). Toutes les catégories demandes n'ont pas de quantum (par ex. une demande de divorce) et seul le sens du résultat sera la donnée à extraire dans ce cas.

1.1.1.4 Quantum obtenu ou résultat

Le quantum obtenu quantifie le résultat ou la décision des juges. Nous le notons q_r. Il ne peut qu'être inférieur ou égal au quantum demandé. Si la demande est rejetée, q_r est nul même si cela n'est pas explicitement mentionné dans le document. A noter qu'il doit être de la même nature que le quantum demandé (somme d'argent ou durée).

Label	Expression nominative	Objet	Fondement
acpa	amende civile pour abus de procédure	amende civile	Articles 32-1 code de procédure civile + 559 code de procédure civile
concdel	dommages-intérêts pour concurrence déloyale	dommages-intérêts	Article 1382 du code civil
danais	dommages-intérêts pour abus de procédure	dommages-intérêts	Articles 32-1 code de procédure civile + 1382 code de procédure civile
dcppc	déclaration de créance au passif de la procédure collective	déclaration de créance	L622-24 code de commerce
doris	dommages-intérêts pour trouble de voisinage	dommages-intérêts	principe de responsabilité pour trouble anormal de voisinage
styx	frais irrépétibles	dommages-intérêts	Article 700 du code de procédure civile

Les labels ont été définis particulièrement dans le cadre du projet, et par conséquent, ils n'existent pas dans le langage juridique.

Tableau 1.1 – Exemples de catégories de demandes

1.1.2 Expression, défis et indicateurs d'extraction

Les demandes sont en général décrites à la fin de la section d'exposé des faits, procédures, moyens et prétentions des parties (section Litige cf. **??** et **??**). Elles rentrent donc dans les "moyens et prétentions des parties" qui regroupent les demandes et les arguments des parties. Quant aux résultats, ils sont décrits dans la section Dispositif et dans la section Motifs (raisonnement des juges). Les demandes sont exprimées dans différents paragraphes qui correspondent soit à une partie, soit à un groupe de parties partageant les mêmes demandes (par ex. des époux). Les paragraphe sont parfois organisés en liste dont chaque élément exprime une ou plusieurs demandes, ou fait référence à un jugement antérieur. Les résultats ont aussi la forme de liste dans la section Dispositif. Par contre, dans les motifs de la décision, les raisonnements sont organisés en paragraphes, et ordonnés catégorie après catégorie. Le résultat est donné à la fin du groupe de paragraphes associé à la catégorie.

Cette pseudo-structure n'est pas standard et impose de nombreux dé-

fis à relever. En effet, une décision jurisprudentielle porte sur plusieurs demandes de catégories différentes ou similaires. Il est important de faire correspondre un quantum demandé extrait au sens et quantum du résultat qui font référence à la même demande. La séparation des demandes et des résultats rend difficile cette mise en correspondance. Ce problème peut aussi être causé par la redondance des quanta ; par exemple, les résultats exprimés dans les Motifs sont résumés dans le Dispositif. D'autre part, les références aux jugements antérieurs exigent de résoudre des références aux résultats de jugements antérieurs qui sont, généralement, rappelés dans le même document. Notons aussi que les difficultés liées aux agrégations (par ex. "*infirmer ... en toutes ces dispositions*") et aux restrictions/sélections (par ex. "*infirme le jugement ... sauf en ce qu'il a condamné M. A. ...*") méritent d'être résolues. Par ailleurs, les catégories de demandes sont nombreuses [1] mais ne sont pas toutes présentes à la fois dans les décisions. Tous ces aspects rendent difficile l'annotation manuelle des données de référence et la modélisation d'une approche d'extraction adéquate. Nous avons cependant identifié des indicateurs qui pourraient être utiles.

On pourrait au préalable annoter les candidats potentiels de quanta. Nous nous sommes intéressés aux demandes dont les quanta sont des sommes d'argent. Les mentions de somme d'argent sont généralement de la forme « `[valeur] [monnaie]` » (par ex. `3000 €`, `15 503 676 francs`, `un euro`, `339.000 XPF`). Des centimes apparaissent parfois (par ex. `dix huit euros et soixante quatorze centimes`, `26'977 € 19`). Ainsi, il est possible d'annoter les sommes d'argent à l'aide d'une expression régulière. Même s'il est difficile de reconnaître des sommes d'argent écrites en lettre, il faut remarquer que l'équivalent en chiffre est généralement mentionné tout près (par ex. `neuf mille cinq cent soixante six euros et quatre vingt sept centimes (9566,87 € )`).

La terminologie utilisée est aussi un bon indicateur pour reconnaître des demandes et des résultats. En effet, le vocabulaire utilisé est très souvent propre aux catégories de demandes. Par exemple le dernier élément de la Figure 1.1a comprend le terme "*pour procédure abusive*" qui est près d'une somme d'argent (*3000 €*) ; il est donc probable que ce type de terme assez particulier soit un bon indicateur de la position des quanta. Par ailleurs, des verbes particuliers sont utilisés pour exprimer les demandes et résultats : infirmer, confirmer, constater, débouter, dire ...

1. plus de 500 selon la nomenclature des affaires civiles NAC+.

1.1.3 Formulation du problème

Nous avons tenu compte de deux principaux aspects du problème :

1. Une décision comprend plusieurs demandes de catégories similaires ou différentes ;
2. Il existe un grand nombre de catégories (500+) ; ce qui rend difficile l'annotation d'exemples de référence pour couvrir toutes ces catégories.

L'idée est de pouvoir ajouter progressivement de nouvelles catégories. Nous avons par conséquent opté pour une extraction par catégorie. Cette stratégie permet par ailleurs d'ajouter facilement de nouvelles classes sans avoir à redéfinir les classes déjà entraînées. Une exécution du système d'extraction permet ainsi d'extraire les demandes d'une seule catégorie. Le problème est décomposé en deux tâches :

Tâche 1 : Détecter les catégories présentes dans le document pour appliquer l'extraction uniquement à ces catégories ;

Tâche 2 : Pour chaque catégorie c identifiée, extraire les demandes :

1. identification des valeurs d'attributs : quanta demandés (q_d), quanta obtenus (q_r), et sens du résultat (s_r) ;
2. mise en correspondance des attributs pour former les triplets (q_d, s_r, q_r) correspondants aux paires demande-résultat.

1.2 Travaux connexes

Chacune des tâches précédentes se rapproche d'une tâche couramment traitée en fouille de texte. En effet, la détection de catégories dans les décisions peut être modélisée comme un problème de classification de documents. La tâche d'extraction se rapproche plus quant à elle des problématiques comme l'extraction d'évènements, le remplissage de champs, ou encore l'extraction de relations et la résolution de référencement.

1.2.1 Extraction d'éléments structurés

Les demandes ressemblent aux structures telles que les relations ou les évènements. En effet, les champs définis par la compétition d'Extraction Automatique de Contenus ACE (*Automatic Content Extraction*), dans LDC [2008], pour les relations et LDC [2005] pour les évènements, se rapprochent de ceux visés lors de l'extraction des demandes comme l'illustre

le Tableau Tableau 1.2 Page 7. Plus précisément, une catégorie de demandes correspond à un type d'évènement ou de relation entre deux entités. Les arguments qui participent à l'évènement « demande » ou à la relation « demande-résultat » sont le quantum demandé et le quantum résultat. Le sens du résultat représente la classe de la structure « demande ».

	Relation [LDC, 2008]	**Événement [LDC, 2005]**	**Analogie chez les demandes**
Type	Org-Aff.Student-Alum	Die	Catégorie="Dommages-intérêts pour procédure abusive"
Passage (*extend*)	*Card graduated from the University of South Carolina*	"Il est mort hier d'une insuffisance rénale."	(*Figure 1.1*)
Déclencheur (*trigger*)	-	"mort"	"procédure abusive"
Participants ou Arguments (*arguments*)	Arg1="Card" Arg2="the University of South Carolina"	Victim-Arg="il" Time-Arg="hier"	Quantum-demandé="3000€" Quantum-obtenu="0 €"
Classes (*attributes*, *classes*)	Asserted	Polarity=POSITIVE, Tense=PAST	Sens-résultat="Rejeté"

Tableau 1.2 – Exemples d'analogie entre relations, évènements et demandes

1.2.2 Approches d'extraction d'éléments structurés

L'extraction d'éléments structurés repose généralement sur une approche modulaire du problème qui le décompose en tâches plus simples. D'une part, on dispose de l'identification des déclencheurs [2] et des arguments. D'autre part, une mise en correspondance relie les arguments et déclencheurs qui participent à la même relation ou au même évènement. Les classes peuvent être déterminées par classification du passage associé. Cette décomposition a permis à de nombreuses méthodes de voir le jour.

L'approche traditionnelle consiste en une chaîne de traitements enchaînant des modules adaptés à une tâche simple. La sortie d'une étape est l'entrée de la suivante. C'est ainsi que Ahn [2006] définit un enchaînement de modèles de classification (k-plus-proches-voisins [Cover & Hart, 1967]

2. Terme-clés indiquant la présence d'un évènement [LDC, 2005].

vs. classificateur d'entropie maximum [Nigam *et al.* , 1999]), pour extraire des champs d'évènements dans le corpus d'ACELDC [2005]. Bien que les différents modules soient plus faciles à développer, ce type d'architecture souffre de la propagation d'erreurs d'une étape à la suivante, ainsi que de la non exploitation de l'interdépendance entre les tâches. Par conséquent, l'inférence jointe des champs est préconisée. Celle-ci peut-être réalisée par une modélisation graphique probabiliste ou neuronale. Par exemple, pour l'extraction d'évènements, Yang & Mitchell [2016] estiment la probabilité conditionnelle jointe du type d'entité t_i, les rôles des arguments $r_{i.}$ et les types d'entités qui remplissent ces rôles $a.$: $p_\theta(t_i, r_{i.}, a.|i, N_i, x)$, i étant un déclencheur candidat, N_i l'ensemble des entités candidates qui sont de potentiels arguments pour i, et x est le document. Cette approche obtient 50.6% de F_1-mesure moyenne pour la détection des valeurs d'arguments et 48.4% pour leur classification dans leur rôle respectif. Par ailleurs, Nguyen *et al.* [2016] illustrent l'utilisation des réseaux de neurones profonds avec une couche pour la prédiction du déclencheur, une autre pour le rôle des arguments, et la dernière encode la dépendance entre les labels de déclencheurs et les rôles d'arguments. Cette approche obtient 62.8% de F_1-mesure moyenne pour la détection des valeurs d'arguments et 55.4% pour leur classification dans leur rôle respectif.

L'annotation du corpus de l'ACE est un marquage des champs dans le texte, et par conséquent, la position ou l'occurrence des champs est indiquée (« annotation au niveau du segment de mots »). Comme dans notre cas, les données peuvent être annotées dans un tableau, hors des textes d'où elles sont issues. Il est donc nécessaire de retrouver leur position sans supervision. Palm *et al.* [2017] proposent dans cette logique une architecture de réseaux de neurones point-à-point qu'ils ont expérimentés sur des corpus de requêtes de recherche de restaurant et films [Liu *et al.* , 2013] ou de réservation de billets d'avion [Price, 1990]. Ils se sont intéressés au problème de remplissage de champs en apprenant la correspondance entre les textes et les valeurs de sorties. Leur modèle est basé sur les réseaux de pointeurs [Vinyals *et al.* , 2015] qui sont des modèles séquence-à-séquence avec attention, dans lesquels la sortie est une position de la séquence d'entrée. Le modèle proposé consiste en un encodeur de la phrase et des contextes, plusieurs décodeurs (un pour chaque champ). L'application de cette architecture à l'extraction des demandes serait confrontée à deux obstacles majeurs auxquels il faut répondre au préalable. Premièrement, les décisions judiciaires ont des contenus de plusieurs centaines à plusieurs milliers de lignes contrairement aux requêtes manipulées par Palm *et al.* [2017] dont la plus longue ne comprend que quelques dizaines de mots. La complexité des architectures neuronales de TALN augmente

rapidement en espace et en temps avec la longueur des documents manipulés. Deuxièmement, nous disposons de très peu de données annotées ; entre 23 et 198 documents annotés dans notre cas contre plusieurs milliers pour les expérimentations de Palm *et al.* [2017].

L'avantage de l'utilisation des réseaux de neurones vient de leur capacité à apprendre automatiquement des caractéristiques pertinentes contrairement aux modèles probabilistes qui exigent très souvent une ingénierie manuelle des caractéristiques. Par contre, il est beaucoup plus facile d'utiliser les modèles probabilistes sur des corpus de faible taille et de longs textes comme c'est le cas pour le problème d'identification des demandes judiciaires.

1.2.3 Extraction de la terminologie d'un domaine

L'identification des attributs peut être facilitée grâce à leur proximité avec des termes-clés caractéristiques des catégories de demandes au même titre que les « déclencheurs » aident à identifier les évènements. Ne disposant pas au préalable de la liste des termes pertinents pour l'extraction des demandes, il est possible de les apprendre. Il existe à cet effet plusieurs métriques statistiques de pondération de termes généralement employées en recherche d'information et en classification de texte comme méthodes de sélection de caractéristiques. Ces métriques sont qualifiées de poids globaux car calculées à partir des occurrences dans un corpus, à la différence des poids locaux (Tableau 2.1) calculés à partir des occurrences dans un document. Quelques métriques sont formulées ici en utilisant les notations du Tableau 1.3 définies pour une base d'apprentissage.

1.2.3.1 Métriques non-supervisées

Les métriques non-supervisées affectent un score à un terme en rapport avec l'importance de ce dernier dans le corpus global D. Parmi ces métriques, on retrouve par exemple la fréquence inverse de document (*inverse document frequency*) idf [Sparck Jones, 1972] et ses variantes $pidf$ [Wu & Salton, 1981] et $bidf$ [Jones *et al.* , 2000] accordent plus d'importance aux termes rares. Elles considèrent en fait qu'un terme rare est plus efficace pour la distinction entre des documents. Par conséquent, elles sont efficaces en recherche d'information mais moins indiquées en classification de textes où le but est plutôt de séparer des catégories [Wu *et al.* , 2017]. Elles se formulent comme suit :

$$idf(t) = \log_2\left(\frac{N}{N_t}\right), pidf(t) = \log_2\left(\frac{N}{N_t} - 1\right), bidf(t) = \log_2\left(\frac{N_{\bar{t}} + 0.5}{N_t + 0.5}\right)$$

Notation	Description
t	un terme
d	un document
$\|t\|$	longueur de t (nombre de mots)
c	la catégorie (domaine ciblé)
$\overline{c}$	la classe complémentaire ou négative
D	ensemble global des documents de taille $N = \|D\|$
D_c	ensemble des documents de c de taille $\|D_c\|$
$D_{\overline{c}}$	ensemble des documents de $\overline{c}$ de taille $\|D_{\overline{c}}\|$
N	nombre total de documents
N_t	nombre de documents contenant t
$N_{\overline{t}}$	nombre de documents ne contenant pas t
$N_{t,c}$	nombre de documents de c contenant t
$N_{\overline{t},c}$	nombre de documents de c ne contenant pas t
$N_{t,\overline{c}}$	nombre de documents de $\overline{c}$ contenant t
$N_{\overline{t},\overline{c}}$	nombre de documents de $\overline{c}$ ne contenant pas t
$DF_{t\|c}$	proportion de documents contenant t dans le corpus de c ($DF_{t\|c} = \frac{N_{t,c}}{\|D_c\|}$)
$DF_{c\|t}$	proportion de documents appartenant à c dans l'ensemble de ceux qui contiennent t

Tableau 1.3 – Notation utilisée pour formuler les métriques

Il est possible de prendre explicitement en compte le fait que les termes peuvent comprendre plusieurs mots (n-grammes) et avoir des tailles différentes (nombre de mots). La C-value [Frantzi *et al.* , 2000], par exemple, distingue la fréquence du terme et de ses sous-termes (termes imbriqués) par la formule :

$$\text{C-value}(t) = \begin{cases} \log_2(|t|) \cdot (N_t - \frac{1}{|T_t|} \cdot \sum_{b \in T_t} N_b), & \text{si } t \text{ est imbriqué} \\ \log_2(|t|) \cdot N_t, & \text{sinon,} \end{cases}$$

T_t étant l'ensemble des termes candidats qui contiennent t.

1.2.3.2 Métriques supervisées

Les métriques supervisées mesurent l'information contenue dans les labels des documents de la base d'apprentissage. Pour un terme t, elles expriment généralement la différence de proportion qui existe entre les occurrences de t dans D_c et ses occurrences dans $D_{\overline{c}}$. Elles sont ainsi mieux

adaptées à la distinction entre catégories. Parmi les nombreuses métriques existantes, nous avons expérimenté les suivantes :

La différence de fréquence Δ_{DF} consiste simplement à calculer la différence entre les proportions de documents contenant t respectivement dans c et $\overline{c}$:

$$\Delta_{DF}(t,c) = DF_{t|c} - DF_{t|\overline{c}}$$

Le gain d'information *ig* [Yang & Pedersen, 1997] estime la quantité d'information apportée par la présence ou l'absence d'un terme t sur l'appartenance d'un document à une classe c :

$$ig(t,c) = \begin{array}{l} \frac{N_{t,c}}{N}\log_2\left(\frac{N_{t,c}N}{N_t}\right) + \frac{N_{\overline{t},c}}{N}\log_2\left(\frac{N_{\overline{t},c}N}{N_{\overline{t}}|D_c|}\right) \\ \quad + \frac{N_{t,\overline{c}}}{N}\log_2\left(\frac{N_{t,\overline{c}}N}{N_t|D_{\overline{c}}|}\right) + \frac{N_{\overline{t},\overline{c}}}{N}\log_2\left(\frac{N_{\overline{t},\overline{c}}N}{N_{\overline{t}}|D_c|}\right) \end{array}$$

La fréquence de pertinence *rf* [Lan *et al.* , 2009] a comme intuition de considérer que plus la fréquence d'un terme t est élevée dans D_c relativement à sa fréquence dans $D_{\overline{c}}$, plus il contribue à distinguer les documents de c de ceux de $\overline{c}$. Elle est calculée par la formule :

$$rf(t,c) = \log\left(2 + \frac{N_{t,c}}{max(1, N_{t,\overline{c}})}\right)$$

Le coefficient du χ^2 [Schütze *et al.* , 1995] estime le manque d'indépendance entre t et c. Par conséquent, une grande valeur de $\chi^2(t,c)$ indique une relation étroite entre t et c. Elle est calculée par la formule :

$$\chi^2(t,c) = \frac{N((N_{t,c}N_{\overline{t},\overline{c}}) - (N_{t,\overline{c}}N_{\overline{t},c}))^2}{N_t N_{\overline{t}}|D_c||D_{\overline{c}}|}$$

Le coefficient de corrélation *ngl* de Ng, Goh et Low [Ng *et al.* , 1997] est la racine carré du χ^2 [Schütze *et al.* , 1995] :

$$ngl(t,c) = \frac{\sqrt{N}(N_{t,c}N_{\overline{t},\overline{c}}) - (N_{t,\overline{c}}N_{\overline{t},c})}{\sqrt{N_t N_{\overline{t}}|D_c||D_{\overline{c}}|}}.$$

L'intuition est de ne regarder que les termes qui proviennent de D_c et qui indiquent l'appartenance à c. Une valeur positive de *ngl* signifie que t est corrélé avec c, lorsqu'une valeur négative signifie que t est corrélé à $\overline{c}$.

Le coefficient *gss* de Galavotti, Sebastiani, et Simi [Galavotti *et al.* , 2000] est une fonction simplifiée du *ngl* [Ng *et al.* , 1997] :

$$gss(t,c) = (N_{t,c}N_{\bar{t},\bar{c}}) - (N_{t,\bar{c}}N_{\bar{t},c}).$$

Le facteur N a été éliminé car il est le même pour tous les termes. Le facteur $\sqrt{N_t N_{\bar{t}}}$ est supprimé car il accentue les termes extrêmement rares qui ne sont pas efficaces pour la classification de textes. Le facteur $\sqrt{|D_c||D_{\bar{c}}|}$ est éliminé car il accentue les catégories extrêmement rares, ce qui tend à réduire l'efficacité micro-moyennée (efficacité calculée globalement sur le corpus de test sans distinction du label des éléments).

Le coefficient de Maracuilo (*mar*) [Marascuilo, 1966] qui se calcule par la formule :

$$mar(t,c) = \frac{\left(\begin{array}{l}(N_{t,c} - N_t N_{t,c}/N)^2 \\ \quad + (N_{t,\bar{c}} - N_t|D_{\bar{c}}|/N)^2 \\ \quad\quad + (N_{\bar{t},c} - |D_c|N_{\bar{t}}/N)^2 \\ \quad\quad\quad + (N_{\bar{t},} - N_{\bar{t}}|D_{\bar{c}}|/N)^2 \\ \\ \end{array}\right)}{N}.$$

C'est un test de proportion multivariée. Nous proposons de l'utiliser pour comparer les proportions d'occurrences d'un terme t dans différents corpus.

Le « delta lissé d'*idf* **»** , *dsidf* [Paltoglou & Thelwall, 2010], est une version lissée du delta *idf* (*didf*) de Martineau & Finin [2009] ($didf(t,c) = \log_2\left(\frac{|D_{\bar{c}}|N_{t,c}}{|D_c|N_{t,\bar{c}}}\right)$). *dsidf* se formule comme suit :

$$dsidf(t,c) = \log_2\left(\frac{|D_{\bar{c}}|(N_{t,c}+0.5)}{|D_c|(N_{t,\bar{c}}+0.5)}\right)$$

.

Le delta BM25 d'*idf* , *dbidf* [Paltoglou & Thelwall, 2010], est une autre variante plus sophistiquée du *didf* qui se calcule comme suit :

$$dbidf(t,c) = \log_2\left(\frac{(|D_{\bar{c}}| - N_{t,\bar{c}} + 0.5)|(N_{t,c}+0.5)}{(|D_c| - N_{t,c} + 0.5)(N_{t,\bar{c}}+0.5)}\right)$$

1.2.3.3 Discussions

A l'exception de la C-value, ces métriques ne tiennent pas explicitement compte de la taille des termes dans les situations où on souhaiterait manipuler des termes de tailles différentes. Brown [2013] propose que soit affecté à un n-gramme t le poids $\left(\frac{N_t}{N}\right)^{0.27} |t|^{0.09}$, une formule obtenue empiriquement pour l'identification du langage d'un document. Par ailleurs, la méthode C-value [Frantzi *et al.* , 2000] propose un produit similaire avec le logarithme de la longueur à la place des puissances. Il est par conséquent évident que le produit lissé de la longueur du terme (puissance ou logarithme) avec les métriques décrites précédemment, permet de favoriser les longs termes qui, bien que rares, sont très souvent plus pertinents que certains termes plus courts. Aussi, le temps pour calculer ces différentes métriques devient rapidement long, surtout pour des n-grammes de mots de taille variée (nombre de mots). Pour compter rapidement les occurrences des n-grammes des corpus, nous avons utilisé la librairie SML [3] [Harispe *et al.* , 2013] lors des expérimentations.

1.3 Méthode

1.3.1 Détection des catégories par classification

Étant donné l'ensemble $D_{\bar{c}}$ des documents ne comprenant aucune demande de la catégorie d'intérêt c, nous proposons de modéliser la tâche de détection des catégories en une tâche de classification de documents. Pour chaque catégorie c, un modèle de classification binaire est entraîné pour déterminer si un document d contient une demande de la catégorie c. Nous avons particulièrement expérimenté quatre algorithmes traditionnellement utilisés comme approches de base. Il s'agit du classifieur bayésien naïf [Duda *et al.* , 1973], de l'arbre de décision C4.5 [Quinlan, 1993], des k-plus-proches-voisins (kNN) [Cover & Hart, 1967], de la machine à vecteurs de support (SVM) [Vapnik, 1995]. Ces algorithmes sont décrits en détail dans le chapitre 2 qui est axé sur la classification des documents. Les labels utilisés correspondent aux catégories d'intérêt. Par exemple, un document sera labellisé *danais* s'il contient des demandes de dommages-intérêts pour abus de procédure, et *nodanais* sinon. Etant donné le grand nombre de métriques de pondération existantes, la métrique choisie est celle qui fournit la meilleure performance sur les données d'apprentissage.

3. `http://www.semantic-measures-library.org`

1.3.2 Extraction basée sur la proximité entre sommes d'argent et termes-clés

Diverses approches d'extraction d'information existent (§ 1.2.2). Il est important de proposer dans un premier temps une approche basique explorant la solvabilité du problème du fait de ses multiples spécificités dont l'annotation d'une seule catégorie dans un document qui en contient plusieurs, l'annotation dans un tableau et donc à l'extérieur du document, la très faible quantité des données annotées, la multiplicité des demandes et des catégories dans un même document. Par conséquent, nous proposons ici une chaîne d'extraction à base de termes-clés, applicable pour chaque catégorie de demande. Il s'agit d'une approche qui tente de reproduire une lecture naïve du document en se basant sur des expressions couramment employées pour énoncer les demandes et résultats. La méthode consiste en deux phases dont une phase d'apprentissage des termes-clés de la catégorie, à proximité desquels seront identifiés les attributs durant la phase d'extraction des demandes comme l'illustre la Figure 1.2 Page 14. On remarque en effet que, naïvement, le seul fait que 1500 euros soit aussi proche des termes-clés *amende civile* et *pour procédure abusive* signifie bien qu'il s'agit du quantum demandé comme amende civile pour procédure abusive.

" ...
- débouter M. S. de ...
- le condamner à payer une amende civile de 1.500 euros pour procédure abusive ...
- le condamner à payer la somme ..."

(a) Extrait original d'un énoncé de demande avant marquage

" ...
- débouter M. S. de ...
- le **<demande categorie="acpa">**condamner à payer une <terme-clef categorie="acpa">**amende civile**</terme-clef> de <argent> **1.500 euros** </argent> <terme-clef categorie="acpa"> **pour procédure abusive**</terme-clef> ...
- le**</demande>** condamner à payer la somme ..."

(b) Énoncé, sommes d'argent, et termes-clés marqués

Figure 1.2 – Illustration de la proximité des quantas et termes-clés

1.3.2.1 Pré-traitement

Le pré-traitement est nécessaire pour :

1. sectionner le document en 4 sections Entête, Litige, Motifs, Dispositif ;
2. annoter les sommes d'argent (en chiffre) à l'aide de l'expression régulière « `[0-9]([0-9]|['.,]|\s)*\s*([Ee]uro[s]{0,1} |franc[s]{0,1} |€|F|XPF|CFP|EUR|EUROS|[i])( |$)` » ;
3. annoter les énoncés de demandes et de résultats respectivement dans les sections Litige et Dispositif : pour cela, les mots introductifs du Tableau Tableau 1.4 Page 15 sont employés car ils indiquent le début d'un énoncé indépendamment de la catégorie ; cette technique de recherche de passages à l'aide de listes de termes-clés prédéfinies a déjà été employé par Wyner [2010] pour annoter les énoncés de résultats en considérant toute phrase contenant un terme de jugement : *affirm, grant, deny, reverse, overturn, remand, ...*

Demande	**Résultat** (organisé par polarité ou sens)		
	accepte	**sursis à statuer**	**rejette**
accorder, admettre, admission, allouer, condamnation, condamner, fixer, laisser, prononcer, ramener, surseoir	*accorde, accordons, admet, admettons, alloue, allouons, condamne, condamnons, déclare, déclarons, fixe, fixons, laisse, laissons, prononce, prononçons*	*réserve, réservons, sursoit, sursoyons*	*déboute, déboutons, rejette, rejetons*

Tableau 1.4 – Mots introduisant les énoncés de demandes et de résultats

1.3.2.2 Apprentissage des termes-clés d'une catégorie

Les termes-clés sont identifiés à l'aide de méthodes statistiques d'extraction ou sélection de terminologie. La base d'apprentissage comprend les corpus D_c et $D_{\bar{c}}$ dont les documents ont été pré-traités. Le processus d'apprentissage des termes se déroule comme suit :

1. Restreindre le contenu de chaque document de D_c à la concaténation des énoncés de demande et résultats contenant des sommes d'argent de valeur égale à celle des quanta annotés.
2. Restreindre chaque document de $D_{\bar{c}}$ à la concaténation des énoncés de demande et résultats contenant des sommes d'argent.

3. A l'aide d'une métrique global g, calculer le score des termes du corpus $D_c \cup D_{\bar{c}}$. Ce score est multiplié par le logarithme de la longueur du terme pour favoriser les termes longs : $g'(t,c) = \log_2(|t|) \times g(t,c)$.
4. Normaliser les scores en appliquant à chaque score original ($g'(t,c)$) la formule $g'_{norm}(t,c) = \frac{g'(t,c) - \min_{t_k}(g'(t_k,c))}{\max_{t_k}(g'(t_k,c)) - \min_{t_k}(g'(t_k,c))}$.
5. Trier les termes par ordre décroissant de score.
6. Sélectionner les premiers termes qui obtiennent les meilleurs performances sur la base d'apprentissage.

1.3.3 Application de l'extraction à de nouveaux documents

A l'aide des termes-clés appris, l'extraction des données de couples demandes-résultats se déroule comme suit :

1. reconnaître et marquer les occurrences des termes dans le document ;
2. extraire les quanta demandés (q_d) et résultats (q_r) à proximité des termes-clés respectivement dans les énoncés de demande et résultat qui contiennent des sommes d'argent et un terme-clé ;
3. le mot introductif de l'énoncé résultat indique le sens du résultat (s_r) tel que catégorisé dans le Tableau 1.4 ;
4. relier les attributs (q_d, s_r, q_r) de chaque paire demande-résultat :
   (a) former les paires (énoncé de demande, énoncé de résultat) similaire (nous utilisons la métrique de « la plus longue sous-séquence commune » [Hirschberg, 1977; Bakkelund, 2009])
   (b) pour chaque paire d'énoncés formée, relier les quanta demandés et quanta résultats en considérant que les quanta correspondants apparaissent dans le même ordre dans les deux énoncés.

1.4 Résultats expérimentaux

Nous analysons ici la capacité de l'approche proposée à reconnaître efficacement les catégories de demandes présentes dans les documents, et à extraire les valeurs des attributs des différentes paires demandes-résultats qui y sont exprimées. Sont discutées les données et métriques d'évaluation employées, ainsi que des résultats expérimentaux observés avec des exemples annotés pour les six catégories du Tableau 1.1.

1.4.1 Données d'évaluation

L'annotation manuelle d'exemples s'effectue pour une catégorie à la fois afin que la tâche soit plus facile pour les experts. Le protocole d'annotation se déroule en 3 étapes :

1. définir une catégorie c par son objet et sa norme juridique ;
2. former un corpus D_c de documents contenant des demandes de c, et un autre $D_{\overline{c}}$ de documents n'en contenant pas ;
3. extraire toutes les demandes de catégories c mentionnées dans D_c, pour annoter les données des paires demande-résultat dans un tableau comme celui illustré par le Tableau 1.5 ;

IDENTIFICATION DE LA DECISION			DESCRIPTION DE LA PRETENTION				DESCRIPTION DU RESULTAT	
Type	Ressort	RG	OBJET	NORME	QUANTUM DEMANDE	POSITION	RESULTAT	QUANTUM RESULTAT (obtenu)
CA	Lyon	14/06911	dommages-intérêts	700 Code de Procédure Civile	3 500,00 €	Demandeur initial	rejette	0,00 €
CA	Lyon	14/06911	dommages-intérêts	700 Code de Procédure Civile	2 000,00 €	Défendeur initial	accepte	1 500,00 €

Les noms des champs sont sur les 2 premières lignes et les demandes sont données en exemple pour la catégorie *dommages-intérêts sur le fondement de l'article 700 du code de procédure civile* (décision 14/06911 de la cour d'appel de Lyon).

Tableau 1.5 – Extrait du tableau d'annotations manuelles des demandes.

La répartition des données d'évaluation est donnée par la Figure 1.3.

Il faut aussi noter que bien que l'annotation manuelle des demandes et des résultats soit réalisée dans un tableau (annotation externe au contenu), elle reste une tâche très difficile. Le très faible nombre de documents annotés manuellement en témoigne. Le nombre maximum de documents annotés pour une catégorie est seulement de 198 (barres vertes de *danais*).

1.4.2 Métriques d'évaluation

Reconnaissance de catégories par classification La classification des documents est évaluée en utilisant les métriques précision (P), rappel (P), f1-mesure (F1).

Extraction des attributs des paires demande-résultat Nous évaluons les approches proposées sur l'extraction de 3 données : le quantum demandé

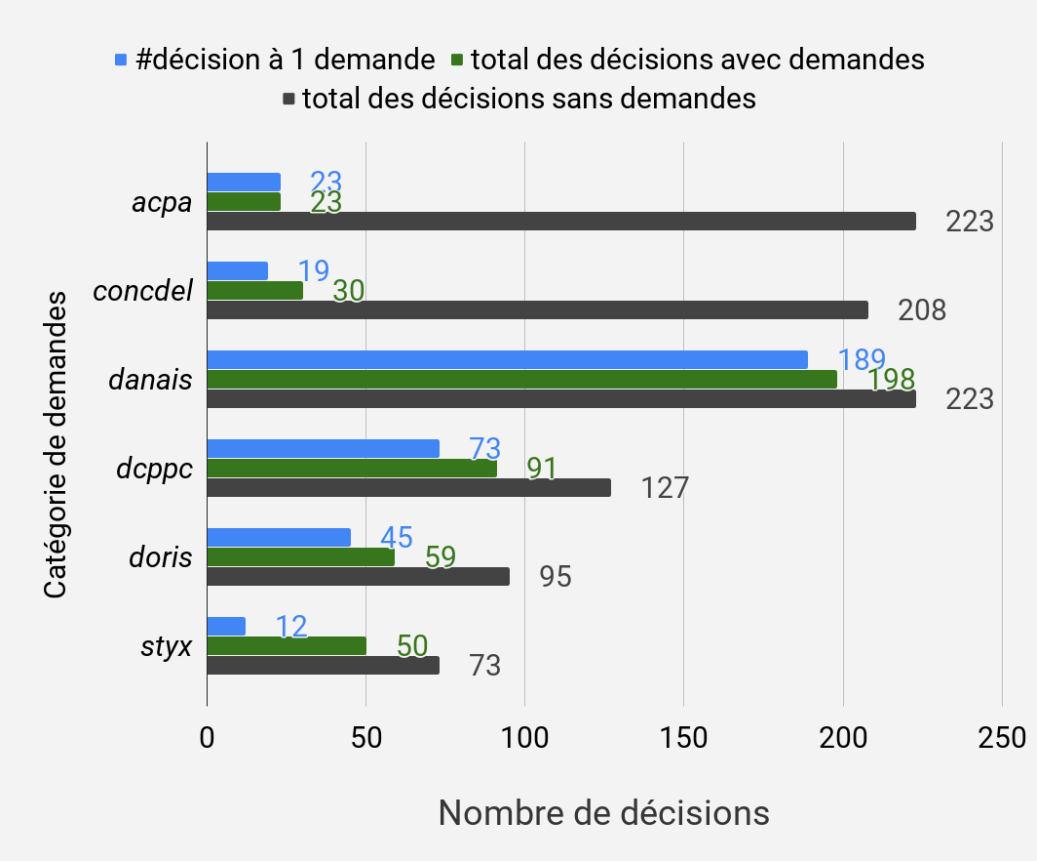


Figure 1.3 – Répartitions des demandes dans les documents annotées.

q_d, le sens du résultat s_r et le quantum obtenu q_r. Une demande est donc un triplet (q_d, s_r, q_r). Il est possible d'évaluer le système pour un sous-ensemble x de $\{q_d, s_r, q_r\}$ sur les demandes extraites d'un corpus annotées D de test. Nous utilisons les métriques traditionnellement employées en extraction d'information : la précision ($Precision_{c,x,D}$), le rappel ($Rappel_{c,x,D}$), et la F1-mesure ($F1_{c,x,D}$). Ces mesures sont définies à partir des nombres de vrais positifs (TP), faux positifs (FP) et faux négatifs (FN) calculés au niveau d'un document d :

- $TP_{c,x,d}$ est le nombre de demandes extraites de d par le système, qui sont effectivement de la catégorie c (demandes correctes) ;
- $FP_{c,x,d}$ est le nombre de demandes extraites de d par le système, mais qui ne sont pas des demandes de c (demandes en trop) ;
- $FN_{c,x,d}$ est le nombre de demandes annotées comme étant de c mais qui n'ont pas pu être extraites par le système (demandes manquées).

Au niveau d'un corpus d'évaluation D, ces métriques sont sommées : $TP_{c,x,D} = \sum_{d \in D} TP_{c,x,d}$ $FP_{c,x,D} = \sum_{d \in D} FP_{c,x,d}$ $FN_{c,x,D} = \sum_{d \in D} FN_{c,x,d}$.

Une donnée observée (par exemple « 3 000 € ») est bien extraite automatiquement si sa valeur (le nombre 3000) correspond à celle du quantum annoté dans le tableau. Nous considérons que les unités monétaires, entre les quanta extraits et ceux manuellement annotés, sont identiques.

1.4.3 Détection des catégories par classification

Les implémentations de la bibliothèque Weka [Frank *et al.* , 2016] ont permis d'utiliser plusieurs modèles de classification : le modèle Bayésien naïf (NB), l'arbre de décision C4.5 (implémenté sous l'appelation J48), les k-plus-proches-voisins (KNN), et le SVM. A chaque entraînement, s'exécute une sélection de modèles par validation croisée sur les données d'entraînement. Elle a pour but de sélectionner la métrique locale et la métrique globale appropriée. Les résultats obtenus par 5-*folds* validation croisée sont présentés sur le Tableau 1.6 Page 19.

	NB			C4.5			KNN			SVM		
	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
acpa	1.0	1.0	1.0	0.996	0.955	0.972	1.0	1.0	1.0	0.996	0.955	0.972
concdel	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.995	0.967	0.979
danais	0.988	0.989	0.988	0.996	0.995	0.995	0.995	0.995	0.995	0.993	0.993	0.993
dcppc	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
doris	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
styx	1.0	1.0	1.0	0.984	0.983	0.983	1.0	1.0	1.0	1.0	1.0	1.0

P= Précision, R=Rappel, F_1 = F_1-mesure

Tableau 1.6 – Evaluation de la détection de catégories.

D'après les résultats, la détection de catégorie par classification binaire est relativement aisée pour les algorithmes traditionnels qui détectent parfaitement la présence ou non d'une catégorie dans les documents. Par conséquent, pour toute catégorie c, les résultats de l'extraction, dans la suite, ne sont discutés que pour les documents de c, car, grâce à l'efficacité de la phase de classification, aucun document de $\overline{c}$ ne sera traité par la phase d'extraction.

1.4.4 Extraction de données des paires demandes-résultats

Les scores des termes-clés candidats étant normalisés, si on sélectionne les termes dont les scores sont supérieurs à un seuil fixé, on remarque que

chaque métrique d'extraction a un niveau d'efficacité différent entre les catégories de demande (Tableau 1.7 Page 20 avec 0.5 comme seuil fixé).

	acpa	*concdel*	*danais*	*dcppc*	*doris*	*styx*	**Moyenne**
$bidf$	37.33	32.73	23.96	20.46	8.08	28.43	25.17
χ^2	**54.55**	25.88	43.97	28.35	13.11	52.73	36.43
$dbidf$	37.58	24.63	56.25	**29.06**	11.58	52.73	35.31
Δ_{DF}	**54.55**	25.55	48.16	28.1	**19.64**	52.73	**38.12**
$dsidf$	37.58	25.25	**56.42**	26.05	8.72	**53.46**	34.58
gss	**54.55**	25.11	48.16	28.1	**19.64**	52.73	**38.05**
idf	38.78	32.73	22.31	20.53	8.27	25.22	24.64
ig	4	12.4	45.21	14.99	16.74	51.13	24.08
$marascuilo$	**54.55**	23.65	43.97	26.67	17.91	52.73	36.58
ngl	42.02	23.97	52.31	27.21	13.29	53.2	35.33
$pidf$	26.19	**33.71**	21.83	20.46	8.76	27.68	23.11
rf	41.11	33.09	55.72	28.56	14.93	51.23	37.44

Tableau 1.7 – Comparaison des pondérations globales suivant la F_1-mesure.

Par conséquent, la métrique et le seuil doivent être bien sélectionnés en fonction de la catégorie de demandes traitée. En choisissant, pour ces hyper-paramètres, les valeurs les plus efficaces pour l'extraction sur la base d'apprentissage, les résultats du Tableau 1.8 Page 21 sont observés. Les améliorations sont à noter notamment pour trois catégories. Le score F_1 sur l'extraction des triplets (q_d, s_r, q_r) passe de 54.55 au maximum à 58.99 (plus de 4%) pour *acpa*, de 29.06 à 29.41 pour *dcppc*, de 19.64 à 29.08 (près de 10%) pour *doris*. Les baisses de performances observées pour les autres catégories est comparativement très faibles (moins de 2%).

Ces résultats détaillés font remarquer que les attributs, pris individuellement, présentent d'assez bonnes performances. Cependant, la mise en correspondance des attributs peine toujours à montrer des performances du même rang. On remarque néanmoins que les scores F_1 des triplets (q_d, s_r, q_r) sont proches de celles des attributs qui présentent le plus de difficulté. En effet, la sélection préalable permet d'observer sur les données de test, des F_1-mesures comprises entre 33.09 % et 71.43 % pour les champs q_d, q_r, et s_r, et entre 28.65 % et 58.99 % pour les triplets (q_d, s_r, q_r). L'échec de l'extraction des attributs est une des principales causes des faibles performances observées pour la liaison des attributs de paires similaires demande-résultat. Par ailleurs, les données sur le résultat, s_r et q_r, sont en générale plus faciles à extraire(F_1-mesures entre 42.12 et 71.43 sauf pour *concdel*) que le quantum demandé q_d (F_1-mesures entre 41.75 et 63.61 sauf pour *concdel*). Remarquons aussi que la précision est en géné-

			Données d'entraînement				Données de test			
c	Données	$\lvert V_c \rvert$	P	R	F_1	%Docs	P	R	F_1	%Docs
acpa	q_d	1	86.4	56.37	68.13	56.37	68.33	54	58.99	46
	q_r	1	100	65.09	78.74	65.09	93.33	63	71.43	55
	s_r	1	100	65.09	78.74	65.09	93.33	63	71.43	55
	(s_r, q_r)	1	100	65.09	78.74	65.09	93.33	63	71.43	55
	(q_d, s_r, q_r)	1	86.4	56.37	68.13	56.37	68.33	54	58.99	46
concdel	q_d	26	49.33	44.02	45.31	24.17	73.2	29.72	33.29	26.67
	q_r	26	48.3	42.66	44.1	22.5	75.73	28.89	34.3	26.67
	s_r	26	46.52	40.89	42.36	22.5	74.93	26.39	33.09	26.67
	(s_r, q_r)	26	46.52	40.89	42.36	22.5	74.93	26.39	33.09	26.67
	(q_d, s_r, q_r)	26	42.43	37.41	38.68	20.83	68.27	23.06	28.65	23.33
danais	q_d	37	77.71	48.71	59.68	37.3	79.25	47.5	59	37.3
	q_r	37	77.68	48.71	59.67	37.03	77.78	46.46	57.79	36.22
	s_r	37	77.05	48.33	59.19	37.03	77.78	46.46	57.79	36.22
	(s_r, q_r)	37	77.05	48.33	59.19	37.03	77.78	46.46	57.79	36.22
	(q_d, s_r, q_r)	37	74.45	46.65	57.16	35.81	74.41	44.38	55.23	34.59
dcppc	q_d	35	45.71	36.64	40.66	34.05	44.64	40.73	41.75	31.4
	q_r	35	78.99	63.21	70.2	59.33	75.48	64.51	68.41	53.82
	s_r	35	84.73	67.85	75.33	63.24	81.21	69.14	73.51	57.43
	(s_r, q_r)	35	78.99	63.21	70.2	59.33	75.48	64.51	68.41	53.82
	(q_d, s_r, q_r)	35	34.2	27.39	30.41	28.03	31.66	28.55	29.41	25.37
doris	q_d	8	31.98	35.76	32.94	7.75	37.48	35.9	36.63	7.12
	q_r	8	35.73	39.72	36.69	8.63	39.43	38.47	38.89	7.12
	s_r	8	35.06	39.56	36.24	9.06	42.91	41.44	42.12	8.94
	(s_r, q_r)	8	32.61	36.16	33.45	8.2	38.14	37.04	37.54	7.12
	(q_d, s_r, q_r)	8	24.48	27.16	25.13	5.61	29.7	28.53	29.08	7.12
styx	q_d	4	69.34	59.55	64.04	33.5	69.3	59.49	63.61	32
	q_r	4	75.87	65.17	70.08	31.5	74.86	64.08	68.63	28
	s_r	4	75.87	65.17	70.08	31.5	74.86	64.08	68.63	28
	(s_r, q_r)	4	75.87	65.17	70.08	31.5	74.86	64.08	68.63	28
	(q_d, s_r, q_r)	4	57.61	49.44	53.19	25.5	57.24	48.36	52.08	24

P = Précision, R = Rappel, F_1 = F_1-mesure
%Docs : proportion de documents dont l'ensemble des données extraites est égale à l'attendu (documents parfaitement traités)
$\lvert V_c \rvert$: nombre moyen de termes-clés identifiés pour la catégorie c

Tableau 1.8 – Résultats détaillés pour l'extraction des données avec sélection automatique de la méthode d'extraction des termes-clés

ral supérieure au rappel ; ce qui signifie que la méthode a plus tendance à éviter les valeurs erronées au risque de manquer un nombre important de valeurs correctes. Enfin, la proportion de documents parfaitement traités (dans lesquels toutes les demandes de la catégorie ont été extraites) reste inférieur à la moyenne même pour *acpa* donc le corpus ne comprend que

des décisions à une seules demande de cette catégorie. L'unique terme-clef appris ne semble pas suffisant pour identifier toutes les demandes de *acpa* (l'« article 32-1 » est un bon indicateur par exemple). La catégorie *doris* enregistre la plus faible valeur pour cette proportion, probablement à cause de la présence dans une même décision de plusieurs demandes pour des raisons très variés du trouble du voisinage (préjudice moral, nuisance sonore, préjudice matériel, préjudice de jouissance, etc.) donc malheureusement les termes-clés ne sont pas tous captés par les méthodes statistiques employées (cf. Tableau 1.12 Page 24).

1.4.5 Analyse des erreurs

En extraction d'éléments structurés, on retrouve trois types d'erreurs [Yang & Mitchell, 2016] : les données manquées (faux négatifs), les données en plus des attendues (faux positifs), et les mauvaises classifications (confusions). La confusion n'est pas discutée ici car les annotations ne sont faites que pour une seule classe. Etant donné que la précision est en général supérieure au rappel d'après nos résultats, il est certain que les erreurs sont majoritairement dues aux données manquées comme le confirme le Tableau 1.9 Page 22.

	Données d'entraînement		Données de test	
	%erreurs FP	**%erreurs FN**	**%erreurs FP**	**%erreurs FN**
q_d	36.90	63.10	36.52	63.48
q_r	32.30	67.70	34.32	65.68
s_r	31.72	68.28	34.11	65.89
(s_r, q_r)	32.32	67.68	34.39	65.61
(q_d, s_r, q_r)	37.77	62.23	37.72	62.28

Tableau 1.9 – Types et taux d'erreurs (pourcentage en moyenne sur les 6 catégories de demandes)

Trois raisons peuvent expliquer le fait que peu de données attendues soient extraites.

Premièrement, certaines valeurs d'attributs ne sont pas mentionnées dans les sections Litige et Dispositif utilisées (pourcentages inférieurs à 100 dans les Tableaux 1.10 et 1.11). Par exemple, les quanta résultat de *doris* sont plus présents dans les Motifs que dans le Dispositif.

En second, la sélection des termes-clés n'est pas parfaite (Tableau 1.12). D'une part, l'ensemble sélectionné ne couvre pas toutes les situations d'expression de la catégorie (par exemple, pour la catégorie *styx*, le terme « frais irrépétibles » est souvent utilisé à la place de « article 700 du code

	$\#q_d$	$\#q_d \neq NUL$	# dans doc.	# dans Litige	# dans Motifs	# dans Dispositif
acpa	23	16	16 (100%)	16 (100%)	9 (56.25%)	5 (31.25%)
concdel	58	56	55 (98.21%)	55 (98.21%)	7 (12.5%)	2 (3.57%)
danais	208	182	182 (100%)	179(100%)	39 (21.43%)	23 (12.64%)
dcppc	126	126	122 (96.83%)	109 (86.51%)	71 (56.35%)	65 (51.59%)
doris	94	83	83 (100%)	82 (98.80%)	21 (25.30)%	6 (7.23%)
styx	89	86	86 (100%)	86 (100%)	12 (13.95%)	9 (10.47%)

Les pourcentages ne sont calculés que pour les valeurs non nulles

Tableau 1.10 – Taux de quanta demandés (q_d) mentionnés dans les documents annotés

	$\# q_r$	$\# q_r \neq NUL$	# dans doc.	# dans Litige	# dans Motifs	# dans Dispositif
acpa	23	6	6 (100%)	3 (50%)	6 (100%)	5 (83.33%)
concdel	58	8	8 (100%)	2 (25%)	8 (100%)	6 (75%)
danais	208	23	23 (100%)	15 (65.22%)	22 (95.65%)	20 (86.96%)
dcppc	126	76	75 (98.68%)	55 (72.37%)	56 (73.68%)	64 (84.21%)
doris	94	44	44 (100%)	28 (63.64%)	40 (90.91)%	24 (54.55%)
styx	89	30	29 (96.67%)	16 (53.33%)	22 (73.33%)	29 (96.67%)

Les pourcentages ne sont calculés que pour les valeurs non nulles

Tableau 1.11 – Taux de quanta accordés (q_r) mentionnés dans les documents annotés

de procédure civile », mais dans très peu d'exemples annotés). D'autre part, certains termes sont trop spécifiques à la base d'apprentissage (par exemple, pour la catégorie *concdel*, des sommes d'argent et autres termes comme « condamner in solidum les sociétés » apparaissent dans la liste).

Enfin, les expérimentations ont été réalisées sur des décisions d'appel mais les énoncés de demande et résultat renvoyant aux décisions de jugements antérieurs ne sont pas encore traités. Ces références aux décisions antérieures représentent une part importante des demandes des décisions d'appel. Il est donc nécessaire de les intégrer explicitement dans le processus d'extraction, pour compléter les données extraites.

1.5 Conclusion

Ce chapitre décrit le problème d'extraction de données pertinentes relatives aux paires demande-résultat mentionnées dans les décisions de justice. Les divers défis relatifs à la tâche y sont discutés en remarquant des analogies avec d'autres tâches classiques de la fouille de données textuelles. Il a été démontré la solvabilité du problème par la proposition et l'expérimentation d'une approche d'extraction basée sur la terminologie

Catégorie	Termes-clés appris
acpa	amende civile
concdel	titre de la concurrence déloyale, somme de 15000euros à titre, réparation de son préjudice financier, payer la somme de 15000euros, condamner in solidum les sociétés, agissements constitutifs de concurrence déloyale
danais	dommages et intérêts pour procédure, 32-1 du code de procédure, intérêts pour procédure abusive, titre de dommages-intérêts pour procédure, intérêts pour procédure, article 32-1 du code, dommages-intérêts pour procédure abusive
dcppc	admet la créance déclarée, admet la créance, passif de la procédure collective, passif de la procédure, hauteur de la somme, créance déclarée, titre chirographaire, admission de la créance, rejette la créance,
doris	préjudices, abusive, condamner solidairement, solidairement, réparation du préjudice, réparation, titre de dommages et intérêts, dommages, titre de dommages, dommages et intérêts, titre de dommages-intérêts, payer aux époux, jouissance
styx	700 du code de procédure, article 700 du code, 700 du code, article 700, 700

Les termes candidats sont des n-grammes de taille variant d'1 à 5 mots consécutifs

Tableau 1.12 – Premiers termes sélectionnés lors de la première itération de la validation croisée

de la catégorie des demandes à extraire et autres connaissances du domaine judiciaire telles que les motifs d'énoncés de demandes et de résultats, ainsi que leur position conventionnelle dans les documents. Les expérimentations démontrent que l'approche permet d'extraire plus ou moins bien des demandes selon la catégorie traitée. Même si nos résultats ont été obtenus à partir de terminologies apprises, une liste de termes fournis par les experts pourrait être plus précise et mettrait à l'abris des biais liés aux échantillons d'apprentissage. A cause de la forte dépendance aux subtilités de rédaction des décisions judiciaires, la méthode proposée rencontre des limites qui ne peuvent être surmontées qu'en la rendant beaucoup plus complexe qu'elle ne l'est déjà. Des approches d'apprentissage automatique sont recommandées comme perspectives. Elles devront être capables d'apprendre l'emplacement des données à extraire de manière semi-supervisée à l'aide de faibles quantités de longs documents annotés.

Chapitre 2

Identification du sens du résultat

Résumé. *L'extraction des demandes a été présentée dans le chapitre précédent comme l'identification du quantum demandé, du quantum résultat et du sens du résultat pour chaque demande d'une catégorie donnée. Le sens du résultat est l'information la plus importante pour le métier car elle donne une idée du taux d'acceptation des demandes dans les tribunaux pour une analyse descriptive du sens du résultat. Il a été proposé d'identifier le sens du résultat en interprétant simplement le verbe de décision de l'énoncé du résultat. Cette technique a l'inconvénient de dépendre de l'identification explicite du passage exprimant le résultat qui est donc une source supplémentaire d'erreurs. Le présent chapitre adresse cet inconvénient en reformulant l'identification du sens du résultat par la classification binaire (« accepte » / « rejette ») de la décision représentée en entrée sous forme vectorielle. Une décision pouvant comprendre plusieurs demandes de la catégorie traitée, nous ne traitons que le cas des décisions à une seule demande de la catégorie. Cependant, le défi de ce problème est le déséquilibre observé entre les 2 classes sur les données d'apprentissage. Nous observons en effet que la tendance est de rejeter une forte majorité des demandes, ce qui serait aussi le cas en général en justice pour la majorité des catégories. Sur la base de l'expertise d'une partie de l'encadrement de cette thèse, nous proposons une application de la méthode Gini-PLS de Mussard & Souissi-Benrejab [2018] qui a été conçue pour rendre robuste aux valeurs aberrantes, de vecteurs représentatifs, la régression par analyse des moindres carrés partiels (PLS). Nos expérimentations la compare à d'autres algorithmes de classification et sous différentes conditions de représentation dont divers modèles vectoriels et restriction du document à des sous-parties. Les meilleures résultats sont obtenus avec les arbres de classification avec une moyenne de F_1-mesure sur les catégories de ? ?%, maximale de ? ?%, minimale de ? ?%. Le Gini-PLS étant à ... ?*

2.1 Introduction

Comme le précédent, ce chapitre est relatif à l'extraction de données sur les demandes et résultats correspondants. Cependant, il est question ici d'extraire uniquement le sens du résultat d'une demande connaissant sa catégorie. Cette étude est intéressante parce que le problème devient

plus simple. En se passant de la localisation précise de l'énoncé du résultat, l'extraction du sens du résultat peut être formulée comme une tâche de classification de documents. Nous modélisons la tâche comme un problème de classification binaire consistant à entrainer un algorithme à reconnaitre si la demande a été rejetée (sens = rejette) ou acceptée (sens = accepte). Cette modélisation est proposée sur une restriction du problème définie par les postulats 2.1.1 et 2.1.2 basés sur nos observations des données annotées manuellement.

Postulat 2.1.1 *Pour toute catégorie de demande, les décisions ne contenant qu'une demande de cette catégorie sont majoritaires.*

Ce postulat est légitime car les statistiques sur les données labellisées de la Figure 1.3 montrent bien que dans chaque catégorie, les décisions contiennent en majorité une seule demande d'une même catégorie. On remarque néanmoins l'exception de la catégorie STYX (dommage-intérêt sur l'article 700 CPC), où dans la majorité des documents, on a plutôt 2 demandes. Cette exception peut se justifier par le fait que chaque partie fait généralement ce type de demande car elle porte sur le remboursement des frais de justice. Ce postulat présente cependant un inconvénient dû au fait que la majorité des demandes d'une catégorie peuvent se retrouver dans des décisions comprenant plus d'une demande de cette catégorie. Pour la catégorie *concdel* par exemple, 58 demandes ont été annotées manuellement (Figure 2.1 Page 27) mais 19 décisions sur 30 (63,33%) ont une seule demande de cette catégorie (Figure 1.3 Page 18). Ces 19 décisions comprennent donc seulement 32,75% ($100 \times 19/58$) des demandes annotées pour *concdel*. Par conséquent, 67.24% de ces demandes sont dans les décisions annotées ayant plus d'une demande de cette catégorie. Il est donc possible de manquer un grand nombre de demandes.

Postulat 2.1.2 *Le sens du résultat est généralement binaire : accepte ou rejette.*

Ce postulat est justifié car les sens de résultat ont majoritairement l'une de ces deux valeurs (Figure 2.1 Page 27). Les autres valeurs sont très rares.

Cette étude porte sur l'analyse de l'impact de différents aspects techniques en général impliqués dans la classification de textes qui consistent en général en une combinaison de représentations des documents et d'algorithmes de classification. Cette analyse permettra de savoir s'il existe une certaine configuration permettant de déterminer le sens du résultat à une demande sans identifier précisément cette dernière dans le document.

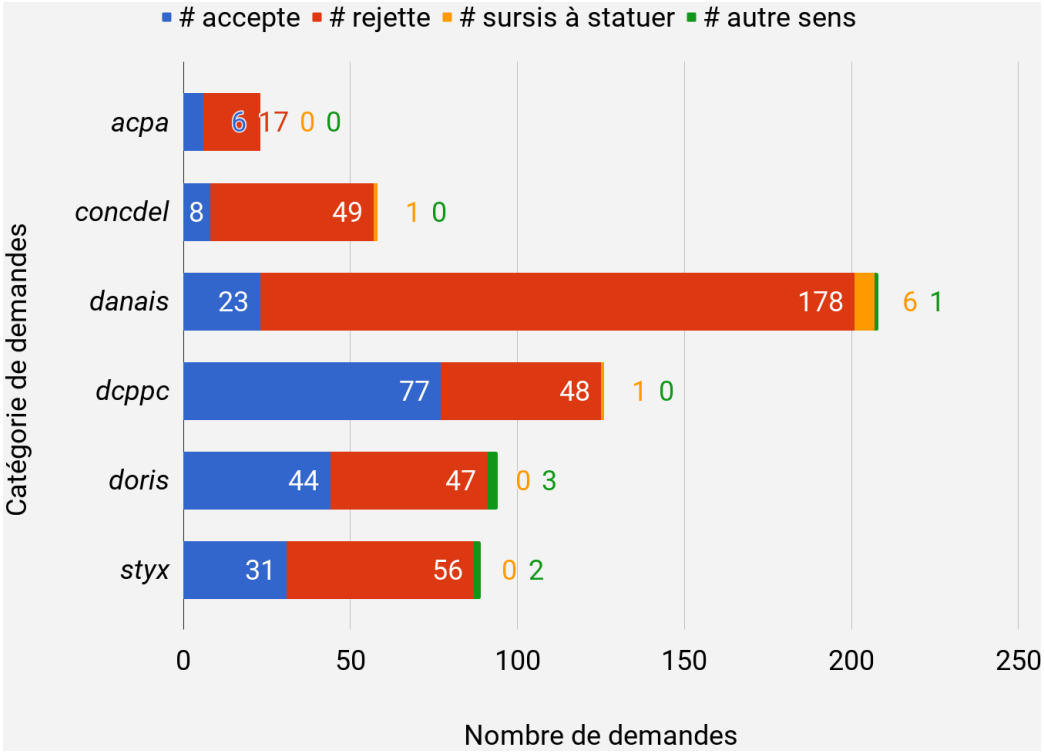


Figure 2.1 – Répartition des sens de résultat dans les données annotées.

2.2 Classification de documents

La classification de textes permet d'organiser des documents dans des groupes prédéfinis. Elle reçoit depuis longtemps beaucoup d'attention. Deux choix techniques influencent principalement les performances : la représentation des textes et l'algorithme de classification. Dans la suite, la variable à prédire est notée y, et la base d'apprentissage comprend les observations de l'échantillon $D = \{(x_i, y_i)_{i=1..n}\}$. Les notations du Tableau 1.3 sont utilisées dans cette section.

2.2.1 Représentation de textes

Chaque document d est représenté sous une forme vectorielle du type TF-IDF (*term frequency - inverse document frequency*) proposé par Salton & Buckley [1988] dont chaque dimension k est identifiée par un terme t_k. Tout document $d \in D$ est une séquence de mots $d = (d[1], \ldots, d[|d|])$, où d_i est le mot à la position i dans d. Sa représentation vectorielle est notée $\vec{d} = (\vec{d}[1], \vec{d}[2], \ldots, \vec{d}[m])$. Pour un modèle vectoriel de type TF-IDF de vo-

cabulaire $T = \{t_1, t_2, \ldots, t_m\}$, $\vec{d}[k] = w(t_k, d)$ le poids du terme $t_k \in T$ dans le texte d (cf. § 1.3.1). Le poids $w(t_k, d)$ affecté à ce dernier est le produit normalisé d'un poids global $g(t_k)$ de t_k dans le corpus d'entraînement et d'un poids local $l(t_k, d)$ de t_k dans le document d : $w(t_k, d) = l(t, d) \times g(t) \times nf(d)$, où nf est un facteur de normalisation tel que la norme euclidienne $\sqrt{\sum_k (w(t_k, d))^2}$. Le poids global est calculé à partir d'une des méthodes de la section § 1.2.3. Le poids local est calculé à partir de la fréquence d'occurrence du terme dans le document à l'aide d'une des méthodes du Tableau 2.1 Page 28.

Description	Formule
Décompte brute du terme [Salton & Buckley, 1988]	$tf(t,d) =$ nombre d'occurrences de t dans d
Présence du terme [Salton & Buckley, 1988]	$tp(t,d) = \begin{cases} 1 & \text{, si } tf(t,d) > 0 \\ 0 & \text{, sinon} \end{cases}$
Normalisation logarithmique	$logtf(t,d) = 1 + \log\left(tf(t,d)\right)$
Fréquence augmentée et normalisée du terme [Salton & Buckley, 1988]	$atf(t,d) = k + (1-k)\frac{tf(t,d)}{\max_{t \in T} tf(t,d)}$
Normalisation basée sur la fréquence moyenne du terme [Manning *et al.* , 2009] (avg représente la moyenne)	$logave(t,d) = \frac{1+\log tf(t,d)}{1+\log \operatorname{avg}_{t \in T} tf(t,d)}$

Tableau 2.1 – Métriques locales.

2.2.2 Algorithmes traditionnels de classification de données

Bien que la classification de documents voit se développer récemment des algorithmes propres aux textes, un grand nombre de méthodes ont été développées dans des contextes détachés des considérations applicatives. Ces méthodes sont généralement basée sur une représentation des textes dans un espace vectoriel $\mathcal{X}$ d'entrée et délimitent une frontière entre les classes dans un espace multidimensionnel. C représente l'ensemble des classes.

2.2.2.1 Le classifieur bayésien naïf (NB)

Le classifieur naïf bayésien [Duda *et al.* , 1973] est un modèle probabiliste qui estime la probabilité qu'un texte appartienne à une classe à l'aide du théorème de Bayes [Raschka, 2014] :

$$\text{probabilité a posteriori} = \frac{\text{probabilité conditionnelle} \cdot \text{probabilité a priori}}{\text{évidence}}$$

La probabilité a posteriori peut être interprétée pour la classification de documents par la question "Quelle est la probabilité que le document d soit de la classe $y = c \in C$?". La réponse à cette question se formalise comme suit :

$$\mathbb{P}(y = c|d) = \frac{\mathbb{P}(d|c)\mathbb{P}(c)}{\mathbb{P}(d)}, \forall c \in C$$

ou plus simplement $\mathbb{P}(y = c|d) \propto \mathbb{P}(c)\mathbb{P}(d|c)$ car $\mathbb{P}(d)$ ne change pas en fonction de la classe et peut donc être ignorée [Rish, 2001]. d est catégorisé dans la classe c pour laquelle $\mathbb{P}(c|d)$ est maximale :

$$y = \underset{c \in C}{\text{argmax}}\, \mathbb{P}(c|d).$$

La phase d'entrainement, appliquée à des exemples déjà labellisés, permet d'estimer les paramètres $\mathbb{P}(c)$ et $\mathbb{P}(d|c)$ qui servent à calculer $\mathbb{P}(c|d)$.

$\mathbb{P}(c)$ est estimée par la proportion de documents classés dans c parmi les exemples d'apprentissage : $\mathbb{P}(c) = \frac{N_c}{N}, \forall c \in C$.

$\mathbb{P}(c|d)$ est estimé grâce l'hypothèse 2.2.1 d'indépendance conditionnelle des descripteurs (termes). Une hypothèse naïve dont la violation, par les données réelles, n'empêche pas le NB de bien fonctionner [Rish, 2001].

Hypothèse 2.2.1 (indépendance conditionnelle des descripteurs) *[Un modèle naïf bayésien étant de type génératif], étant donnée la catégorie du texte, la position de chaque mot dans le texte est générée indépendamment de tout autre mot.*

Si l'ensemble des termes de d est $\{t_1, \ldots, t_K\} \subset T$ (vocabulaire), alors grâce à l'hypothèse 2.2.1, $\mathbb{P}(d|c) = \mathbb{P}(t_1, \ldots, t_K|c) = \prod_{k=1}^{K} \mathbb{P}(t_k|c)$, et pour un terme t_k, la probabilité conditionnelle $\mathbb{P}(t_k|c)$ est la proportion d'exemples de c qui contiennent t_k : $\mathbb{P}(t_k|c) = \frac{N_{t_k,c}}{|D_c|}, \forall k \in \{1, \ldots, K\}$.

2.2.2.2 Machine à vecteurs de support (SVM)

La classification binaire par une machine à vecteurs de support (SVM) [Vapnik, 1995] affecte à tout objet en entrée x la classe y qui correspond au coté d'un hyperplan, séparant les exemples d'entraînement des classes

candidates, où x se trouve. La phase d'apprentissage consiste à déterminer l'hyperplan optimal $w^T x + b = 0$ i.e. dont la marge [1] est maximale (Figure 2.2 [2]).

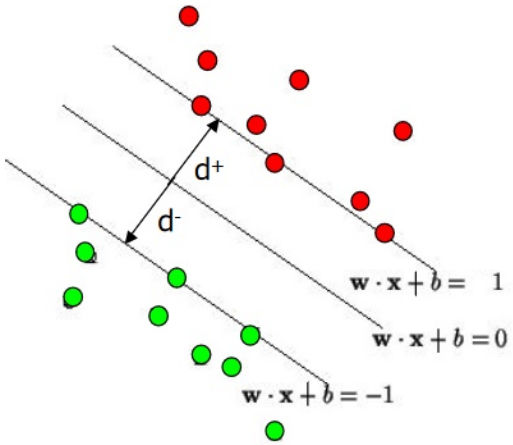


Figure 2.2 – Hyperplan optimal et marge maximale d'un SVM.

Le vecteur w des poids des caractéristiques et le biais b sont déterminés par le problème d'optimisation du « SVM à marges molles » de Cortes & Vapnik [1995] :

$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i \\ \text{s.c.} \quad & y_i(w^T + b) \geq 1 - \xi_i, \xi_i \geq 0. \end{aligned}$$

où C est la constante de régularisation pour éviter un sur-apprentissage ou un sous-apprentissage, les ξ_i sont des variables ressort (*slack variables*) qui permettent à des points de se retrouver dans la marge.

La classification par SVM ne s'applique que lorsque les exemples d'apprentissage des classes sont linéairement séparables. Cependant, ils ne le sont pas toujours dans l'espace $\mathcal{X}$. Ainsi, une fonction « noyau » (*kernel*) $K : \mathcal{X} \longrightarrow \mathcal{F}$ doit être choisie pour transformer chaque donnée entrée x de l'espace original $\mathcal{X}$ vers un nouvel espace $\mathcal{F}$ dit de caractéristiques dans lequel les classes sont linéairement séparables. Par conséquent, la fonction de classification s'écrit

$$f(x) = \sum_{i=1}^{n} \alpha_i K(x, x_i) + b$$

1. La plus petite distance entre les échantillons d'apprentissage et l'hyperplan séparateur

2. `http://web.mit.edu/6.034/wwwbob/svm-notes-long-08.pdf`

où les α_i sont les coefficients de la combinaison linéaire des exemples d'apprentissage égale à w ($w = \sum_{i=1}^{n} \alpha_i x_i$) [Ben-Hur & Weston, 2010]. Parmi les multiples formes qu'il peut prendre, le noyau peut être, par exemple, soit linéaire($K(x, x_i) = x^T x_i + c$), soit polynomial ($K(x, x_i) = (\gamma x^T x_i + c)^d$), soit Gaussien ou RBF[3] ($K(x, x_i) = \exp -\gamma \|x - x_i\|$), soit une sigmoïde ($K(x, x_i) = \tanh(\gamma x^T x_i + c)$) [Amami *et al.* , 2013].

2.2.2.3 k-plus-proches-voisins (kNN)

L'algorithme des k-plus-proches-voisins [Cover & Hart, 1967] est un algorithme simple qui consiste à affecter à un nouvel objet x la classe majoritaire y' parmi ceux des k points d'exemples d'entrainement $\{(x_i, y_i)\}_{1 \leq i \leq k}$, les plus proches du point x selon la distance Dis choisie. Ainsi, trois éléments clés influencent l'efficacité de la classification :

1. Si les les données d'entrainement sont trop nombreuses, le processus de classification peut devenir coûteux en temps de calcul. En effet, la distance du nouvel objet à chaque point annoté est calculée.
2. Le nombre de voisins (c'est-à-dire la valeur de k) ne doit être trop petit pour limiter la sensibilité aux bruits / *outliers*. Il ne doit non plus être trop grand au risque d'avoir dans le voisinage trop de points d'une autre classe que celle attendue. La sensibilité au nombre de voisins peut être atténuée en pondérant les points par leur distance à l'objet à classer. Il a été proposé plusieurs variantes de cette stratégie de « vote pondéré par la distance », comme par exemple :
   - $y' = \underset{c}{\text{argmax}} \sum_{(x_i, y_i)} \frac{1}{Dis(x, x_i)^2} \times I(c = y_i)$ [Dudani, 1976]

     $$\text{où } I(c = y_i) = \begin{cases} 1 & \text{si } c = y_i \\ 0 & \text{sinon} \end{cases}$$

   - $y' = \underset{c}{\text{argmax}} \sum_{(x_i, y_i)} w_i \times I(c = y_i)$ [Gou *et al.* , 2011]

     $$\text{avec } w_i = \begin{cases} \frac{Dis(x, d_k) - Dis(x, d_i)}{Dis(x, d_k) - Dis(x, d_1)} & \text{si } Dis(x, d_k) \neq Dis(x, d_1) \\ 1 & \text{sinon.} \end{cases}$$

3. La métrique de calcul de distance doit être adéquate pour le type de donnée et la tâche. Par exemple, la distance cosinus est souvent préférable à la distance euclidienne pour la classification de documents.

3. *radial base function*

En effet, la distance euclidienne se dégrade lorsque le nombre de dimensions augmente [Sohangir & Wang, 2017; Aggarwal *et al.* , 2001].

2.2.2.4 Arbre de décision

Un arbre de décision est une structure arborescente qui associe un label prédéfini à des objets (classification), ou prédire la valeur d'une variable continue (régression). Il comprend des nœuds internes qui correspondent chacun à un test sur la valeur d'un attribut (test uni-varié), des arêtes correspondant à une sortie du test, et enfin des feuilles ou nœuds terminaux qui correspondent chacun à une prédiction. La classification d'un objet x consiste à faire passer successivement les tests en fonction des valeurs des attributs de x, de la racine jusqu'à une feuille dont le label est retourné comme classe de x (Algorithme 1).

Algorithme 1 : Classification par arbre de décision

Données : Objet x, Arbre A
Résultat : label
$n := racine(A)$;
tant que *n n'est pas une feuille* **faire**
    Effectuer sur x le test associé à n;
    $n :=$ noeud fils de n correspondant au résultat du test ;
retourner le label associé à la feuille n;

La construction de l'arbre (phase d'apprentissage) consiste à générer une hiérarchie de tests, aussi courte que possible, qui divise successivement l'ensemble D d'exemples d'apprentissage en sous-ensembles disjoints de plus en plus purs [4]. L'arbre est construit de la racine aux feuilles en divisant les données d'entraînement S_t à chaque nœud (t) de sorte à minimiser le degré d'impureté des sous-ensembles d'exemples S_{t_i} dans les nœuds fils (t_i). Le critère de coupe est généralement défini à partir d'une métrique d'impureté comme par exemple :

- l'entropie de la distribution des classes dans S_t :
  $h_C(S_t) = -\sum_{c \in C} [p(c|S_t) \log_2 p(c|S_t)]$;
- l'indice de Gini mesurant la divergence entre les distributions de probabilité des valeurs de la variable prédite : $g_C(S_t) = 1 - \sum_{c \in C} [p(c|S_t)]^2$;
- l'erreur de classification définie par : $e_C(S_t) = 1 - \max_{c \in C} [p(c|S_t)]$.

4. homogénéité des labels

Pour ces métriques, $p(c|S_t)$ représente la proportion d'exemples du nœud t appartenant à c. Parmi les critères de séparation les plus populaires associés à ces métriques d'impureté, on retrouve :

- le gain d'information apporté par le test t portant sur l'attribut a (qui divise S_t en des sous-ensembles S_{t_i}) utilisant l'entropie comme métrique d'impureté, et est définie par la différence entre l'entropie de t et la moyenne des entropies des fils de t :
  $ig(S_t, a) = h_C(S_t) - i(S_t, t, a) = h_C(S_t) - \sum_i \frac{|S_{t_i}|}{|S_t|} \cdot h_C(S_{t_i})$;
- le rapport des gains, qui corrige le gain d'information, biaisé en faveur des tests ayant un grand nombre d'alternatives (sorties du nœud), en prenant en compte l'information intrinsèque $h_t(S_t)$ de la séparation de S_t suivant le test t en sous-ensembles S_{t_i} :
  $gr(S_t, t, a) = \frac{ig(S_t,t,a)}{h_t(S_t)}$ avec $h_t(S_t) = \sum_i \frac{|S_{t_i}|}{|S_t|} \log_2 \left(\frac{|S_{t_i}|}{|S_t|}\right)$
- le critère binaire de "doublage" (*twoing criteria*) qui ne s'emploie que pour les arbres binaires :
  $tc(t) = \frac{P(S_{t_R}|S_t)P(S_{t_L}|S_t)}{4} \left[\sum_{c \in C} |p(c|t_L) - p(c|t_R)|\right]^2$ où $P(S_{t_R}|S_t)$ et $P(S_{t_L}|S_t)$ sont les proportions de S_t qui vont respectivement dans les fils t_R et t_L après séparation suivant le test t.

Les variables nominales peuvent être divisées soit en utilisant autant de partitions que de valeurs distinctes, soit uniquement en des partitions binaires suivant des tests booléens nécessitant de rechercher la division optimale. Les variables numériques sont divisées quant à elles soit par discrétisation de leur domaine en les transformant en variables catégoriques ordinales, soit en recherchant la meilleure division binaire parmi toutes les séparations possibles.

La construction de l'arbre est une division récursive qui peut continuer tant qu'il est possible d'améliorer la pureté des nœuds, ce qui peut engendrer un arbre très grand résultant en un sur-apprentissage [5], et une forte complexité temporelle et spatiale lors de la prédiction. Pour s'arrêter plus tôt ("pré-élagage"), plusieurs conditions sont possibles comme par exemple, l'atteinte d'un seuil minimum par la taille des données ($|S_t|$), ou l'atteinte par l'arbre d'une profondeur maximale, ou l'amélioration du critère de division est très faible, etc. Le post-élagage [6] est appliqué après

5. Un modèle trop précis a un très faible taux d'erreur sur les données d'entraînement (erreur d'apprentissage) mais un fort taux d'erreur sur les données de test (erreur de test).

6. Suppression de sous-arbres superflus ou en trop après génération de l'arbre.

construction de l'arbre toujours dans le but de minimiser le sur-apprentissage et la complexité. Le post-élagage peut être basé, par exemple, soit sur la réduction du taux d'erreur (éliminer successivement les feuilles si cela ne fait pas croître le taux d'erreur sur les données d'entrainement), soit sur la stratégie coût-complexité de Breiman *et al.* [1984].

Les algorithmes de construction d'arbres diffèrent ainsi par leur critère de séparation, leur stratégie d'élagage, et leur capacité à gérer les types d'attributs, les valeurs manquantes et extrêmes. Singh & Gupta [2014] comparent les deux algorithmes CART [Breiman *et al.* , 1984] (critère de « doublage », élagage coût-complexité) et C4.5 [Quinlan, 1993] (rapport des gains, élagage à réduction d'erreur).

2.2.2.5 Analyses discriminantes linéaires et quadratiques

L'analyse discriminante comprend l'ensemble des méthodes déterminant les combinaisons linéaires de variables qui permettent de séparer le mieux possible K catégories ou variables qualitatives. Les analyses linéaires et quadratiques sont des méthodes probabilistes basées sur la probabilité conditionnelle d'appartenance d'un objet X à une classe y_k :

$$P(Y = y_k|X) = \frac{P(Y = y_k)P(X|Y = y_k)}{P(X)} = \frac{P(Y = y_k)P(X|Y = y_k)}{\sum_{j=1}^{K} P(Y = y_j)P(X|Y = y_j)}.$$

La classe de X est donc $y_{k*} = \underset{k}{\operatorname{argmax}}\, P(Y = y_k|X)$, avec $P(Y = y_k|X) \propto P(Y = y_k)P(X|Y = y_k)$ car le dénominateur est le même pour toutes les classes. Dans cette expression, $P(Y = y_k)$ est la proportion d'exemples de classes y_k dans l'ensemble des données d'apprentissage. Il ne reste donc qu'à déterminer $P(X|Y = y_k)$, pour trouver y. Deux hypothèses simplifient les calculs :

1. l'hypothèse de normalité statuant que la probabilité conditionnelle $P(X|Y)$ suit une loi normale multivariée :

$$P(X|Y = y_k) = \frac{1}{\sqrt{2\pi det(V_k)}} e^{-\frac{1}{2}(X-\mu_k)'V_k^{-1}(X-\mu_k)}$$

   μ_k étant le centre de gravité conditionnel, et V_k la matrice de variance-covariance de la classe y_k ;

2. l'hypothèse d'homoscédasticité statuant que les matrices de variance co-variance conditionnelles sont identiques i.e. :

$$\forall j, k \in \{1, ..., K\}, V_j = V_k = V.$$

L'analyse discriminante linéaire (LDA) [Fukuaga, 1990; Duda *et al.* , 2000] est définie par une simplification de $P(X|y_k)$ sous ces deux hypothèses. En effet, grâce à la proportionnalité de la probabilité conditionnelle à :$\ln[P(X|y_k)] \propto -\frac{1}{2}(X-\mu_k)'V^{-1}(X-\mu_k)$(Vérifier la position de la transposée), on déduit une fonction discriminante (ou de classement) linéaire proportionnelle à $P(y_k|X)$:

$$d(y_k, X) = \ln[P(Y = y_k)] + \mu_k V^{-1} X' - \frac{1}{2}\mu_k V^{-1}\mu_k'.$$

Ainsi $y_{k*} = \underset{k\in\{1,\dots,K\}}{\operatorname{argmax}}\, d(y_k, X)$.

L'analyse discriminante quadratique (QDA) [McLachlan, 1992] considère l'hétéroscédasticité (i.e. $\exists k \neq j, V_k \neq V_j$), et donc ne s'appuie que sur la première hypothèse (normalité). Dans ce cas, on obtient une règle quadratique de classification $k* = \underset{k\in\{1,\dots,K\}}{\operatorname{argmax}}\, Q_k(X)$ où :

$$Q_k(X) = (X-\mu_k)'V_k^{-1}(X-\mu_k) - 2\ln(\pi_k) + \ln(det(V_k))$$

est la fonction quadratique de classement de la classe k.

2.2.3 Algorithmes dédiés aux textes

Les algorithmes dédiés aux textes intègrent leur propre représentation de document, contrairement aux algorithmes opérant sur des espaces vectoriels aux axes et poids paramétrables à volonté comme le SVM. Actuellement, les algorithmes NBSVM [Wang & Manning, 2012] et FastText [Grave *et al.* , 2017] sont les plus populaires pour la classification de documents avec une très bonne précision pour l'analyse de sentiments [citation] et exemples [Mohammad, 2018; Joulin *et al.* , 2016; Shirsath, 2017; Elsaadawy *et al.* , 2018].

2.2.3.1 NBSVM

Le NBSVM [Wang & Manning, 2012] est un classifieur binaire (deux labels $\{-1;1\}$) dont le principe consiste à transformer les poids $f^{(k)}$ caractéristiques V des textes $x^{(k)}$, réduits à leur simple présence $\widehat{f}^{(k)}$ en réalisant leur produit élément à élément ($\tilde{f}^{(k)} = r \circ \widehat{f}^{(k)}$) avec le vecteur de poids r du classifieurs bayésien multinomial (calculé avec le vecteur présence de caractéristique) : $r = \log\left(\frac{p/||p||_1}{q/||q||_1}\right)$ avec $p = \alpha + \sum_{k:y^{(k)}=1} f^{(k)}$,

$q = \alpha + \sum_{k:y^{(k)}=-1} f^{(k)}$. L'ensemble des caractéristiques V est constitué de n-grammes de mots. Le nouveau vecteur issu de ce produit représente le texte ($x^{(k)} = \tilde{f}^{(k)}$) en entrée d'un SVM classique. La classe de $x^{(k)}$ est prédite par : $y^{(k)} = sign(\mathbf{w}^T x^{(k)} + b)$, $\mathbf{w}$ et b étant appris lors de l'entraînement du SVM. Une interpolation entre le bayésien multinomial et le SVM est nécessaire pour assurer la robustesse du NBSVM et des performances excellentes pour toute tâche de classification de documents ; les poids $\mathbf{w}$ sont réajustés par le modèle $\mathbf{w}' = (1-\beta)\overline{w} + \beta\mathbf{w}$, où $\overline{w} = ||\mathbf{w}||_1/|V|$ et $\beta \in [0;1]$.

2.2.3.2 FastText

FastText [Grave *et al.* , 2017], quant à lui, est un modèle de réseau de neurones dont l'architecture est semblable à celle de la variante CBOW de la méthode de plongement sémantique Word2Vec [Mikolov *et al.* , 2013] dans laquelle le mot du milieu a été remplacé par le label de la classe du texte et au-dessus de laquelle la fonction softmax $f(z) = \left[\frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}\right]_{\forall j \in \{1,\dots,K\}}$ est rajoutée pour réaliser la classification à partir de la représentation distribuée du texte. L'entraînement, sur un corpus manuellement annoté de n documents, consiste à minimiser $-\frac{1}{n} y_i \cdot \sum_{i=1}^{n} y_i \cdot \log f(B \cdot A \cdot x_i)$ qui estime la distribution de probabilité des classes ; A et B étant les matrices des poids à apprendre, et x_i la représentation vectorielle sous forme de « sac de N-grammes »[7] du i^{eme} document. Comment se déroule l'entrainement et l'apprentissage ?

2.2.4 Techniques d'amélioration de l'efficacité

La faible quantité [Ruparel *et al.* , 2013] et le déséquilibre des données sont susceptibles d'être des obstacles à l'entraînement des modèles de classification. De nombreuses techniques permettent néanmoins d'optimiser l'apprentissage en fonction des données. La sélection de modèle consiste à choisir les meilleures valeurs des hyper-paramètres (par exemple C et γ chez le SVM) en testant différentes combinaisons de valeurs candidates sur une fraction de la base d'entraînement (base de développement). La

7. Modèle sac-de-mot calculé sur des occurrences de N-grammes (segments N mots consécutifs dans un texte.)

combinaison de classifieurs est aussi une méthode très étudiée [Kittler *et al.* , 1996; Kuncheva, 2004; Tulyakov *et al.* , 2008] notamment par l'exemple des forêts aléatoires [Breiman, 2001], ou de SVM ensembliste (*Ensemble SVM*) [Dong & Han, 2005]. Par ailleurs, la représentation vectorielle des textes résulte généralement en des vecteurs de haute dimension dont les coordonnées sont en majorité nulles. Par conséquent, les techniques de réduction ou transformation des dimensions, comme les analyses discriminantes, permettent de d'obtenir des vecteurs plus pertinents pour la classification.

2.3 Application du PLS à la classification des textes

Justification : Pourquoi le PLS ? à quel problème cette méthode répond spécifiquement ? Qu'est ce que moi j'ai fait/ajouté ? : le modèle a été déjà publié en temps que méthode de régression. nous l'adaptons pour de la classification de textes pourquoi ? : ça n'a jamais été appliqué en classif. de texte, orienté par l'expertise de l'encadrement sur cette technique déjà publiée.

La méthode ou l'analyse des moindres carrés partiels PLS [Wold, 1966] explique la dépendance entre une ou plusieurs variables y (dite dépendantes) et des K variables $x_1, x_2, ..., x_K$ (dites explicatives ou indépendantes). Elle consiste principalement à transformer les variables explicatives en un nombre réduit de h composantes principales orthogonales $t_1, ..., t_h$. Il s'agit donc d'une méthode de réduction de dimension au même titre que l'analyse en composantes principales, l'analyse discriminante linéaire (LDA), et l'analyse discriminante quadratique (QDA). Les composantes t_h sont construites par étapes en appliquant l'algorithme du PLS de façon récurrente sur les données mal prédites (résidus). Plus précisément, à chaque itération h, la composante t_h est calculée par la formule $t_h = w_{h1}x_1 + \cdots + w_{hj}x_j + \cdots + w_{hp}x_K$ dont les coefficients w_{hj} sont à estimer. L'analyse PLS présente plusieurs avantages [Lacroux, 2011] dont la robustesse au problème de haute-dimension [8] comme on peut l'observer dans nos données (faible quantité de données annotées), et aussi prend en compte la multicolinéarité qui peut exister entre les variables explicatives, notamment quand celles-ci sont des mots/termes co-occurrents dans les documents. Cette méthode est étendue et appliquée avec succès pour divers problèmes de régression [Lacroux, 2011] ou de classification de données vectorielles en général [Liu & Rayens, 2007; Durif *et al.* , 2017; Bazzoli &

8. Lorsque le nombre de variables explicatives est très grand devant le nombre d'observations ($n << K$).

Lambert-Lacroix, 2018], et de textes en particulier [Zeng *et al.* , 2007]. Nous nous sommes intéressés particulièrement à deux extensions : la régression Gini-PLS [Mussard & Souissi-Benrejab, 2018] dont l'intérêt est de réduire la sensibilité aux valeurs aberrantes des variables, et la méthode Logit-PLS [Tenenhaus, 2005] combinant la régression logistique et la PLS. Une combinaison de ces deux approches (Logit-Gini-PLS) est décrite dans la suite de cette section (Section § 2.3.3.2).

2.3.1 L'opérateur Gini covariance

Soit $\bar{x}_k$ la moyenne arithmétique de la variable $x_k, \forall k \in [1 \,..\, m]$ sur les n observations d'apprentissage. L'opérateur de Gini covariance proposé par Schechtman & Yitzhaki [2003], encore appelé opérateur co-Gini est donné par :

$$\text{cog}(x_\ell, x_k) := \text{cov}(x_\ell, F(x_k)) = \frac{1}{n}\sum_{i=1}^{n}(x_{i\ell} - \bar{x_\ell})(F(x_{ik}) - \bar{F}_{x_k}), \qquad (2.1)$$

où $F(x_k)$ est la fonction de répartition de x_k, $\bar{F}_{x_k}$ sa moyenne, avec $\ell \neq k = 1, \ldots, K$. Lorsque $k = \ell$ le co-Gini mesure la variabilité entre une variable et elle-même (l'équivalent de la variance mesurée sur la norme ℓ_2). Le co-Gini est une mesure basée sur la distance de Manhattan (distance de métrique ℓ_1), en effet :

$$\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}|x_{ik} - x_{jk}| = 4\text{cog}(x_k, x_k).$$

D'autre part, lorsque $k \neq \ell$, le co-Gini produit une mesure de la variabilité jointe entre deux variables. Puisque le co-Gini n'est pas symétrique :

$$\text{cog}(x_k, x_\ell) := \text{cov}(x_k, F(x_\ell)) = \frac{1}{n}\sum_{i=1}^{n}(x_{ik} - \bar{x_k})(F(x_{i\ell}) - \bar{F}_{x_\ell}).$$

Définissons les rangs croissants d'une variable alétoire afin de fournir un estimateur de F,

$$R_\uparrow(x_{i\ell}) := nF(x_{i\ell}) = \begin{cases} \#\{x \leq x_{i\ell}\} & \text{si aucune observation similaire} \\ \frac{\sum_{i=1}^{p}\#\{x \leq x_{i\ell}\}}{p} & \text{s'il existe } p \text{ valeurs similaires } x_{i\ell}. \end{cases}$$

Alors, un estimateur du co-Gini est donné par,

$$\widehat{\text{cog}}(x_\ell, x_k) := \frac{1}{n}\sum_{i=1}^{n}(x_{i\ell} - \bar{x_\ell})(R_\uparrow(x_{ik}) - \bar{R}_{\uparrow_{x_k}}),\ \forall k, \ell = 1, \ldots, K, \qquad (2.2)$$

avec $\bar{R}_{\uparrow_{x_k}}$ la moyenne arithmétique du vecteur rang de la variable x_k.

2.3.2 Gini-PLS

Le premier algorithme Gini-PLS a été proposé par Mussard & Souissi-Benrejab [2018]. Nous le décrivons dans les lignes qui suivent. Il s'agit d'une méthode de compression avec débruitage qui consiste à réduire les dimensions de l'espace généré par X afin de trouver des composantes principales débruitées, dans le même esprit qu'une ACP débruitée, néanmoins l'approche est supervisée dans la mesure où une variable cible y est prise en compte dans le changement d'espace. Le sous-espace formé par les composantes principales $\{t_1, t_2, \cdots\}$ est construit de telle sorte que le lien entre les variables explicatives $X = [x_1, x_2, \cdots]$ et la cible y est maximisé.

• **Étape 1 :** La régression Gini permet de concevoir un nouveau type de lien entre la variable expliquée et les variables explicatives tout en évitant l'influence des valeurs aberrantes. Ceci est permis grâce notamment à l'opérateur co-Gini dans lequel le rôle de la variable explicative est remplacé par celui de son vecteur rang dans un espace muni d'une métrique ℓ_1. Ainsi, il est possible de créer un nouveau vecteur de poids w_1 qui renforce le lien (co-Gini) entre la variable expliquée y et les régresseurs X dans le cadre d'une régression (linéaire ou non linéaire).
La solution du programme,

$$\max \operatorname{cog}(y, Xw_1) \text{ , s.c. } \|w_1\| = 1 \text{ , est}$$

$$w_{1j} = \frac{\operatorname{cog}(y, x_j)}{\sqrt{\sum_{k=1}^{K} cog^2(y, x_k)}} \text{ , } \forall j = 1 \ldots, p \text{ .}$$

La pondération est équivalente à :

$$w_{1k} = \frac{\operatorname{cov}(y, R(x_k))}{\sqrt{\sum_{k=1}^{K} \operatorname{cov}^2(y, R(x_k))}} \text{ , } \forall k = 1 \ldots, K \text{ .}$$

Comme dans la régression PLS, on régresse y sur la composante t_1 qui est construite de la manière suivante :

$$t_1 = \sum_{k=1}^{K} w_{1k} x_k \implies y = \hat{c}_1 t_1 + \hat{\varepsilon}_1 \text{ .}$$

• **Étape 2 :** On régresse le vecteur rang de chaque régreseur $R(x_k)$ sur la composante t_1 par moindres carrés ordinaires afin de récupérer les résidus

$\hat{U}_{(1)k}$:

$$R(x_k) = \hat{\beta} t_1 + \hat{U}_{(1)k} \ , \ \forall k = 1, \cdots, K \ .$$

On construit le nouveau vecteur de pondération en utilisant les rangs des résidus des régressions partielles :

$$\max \operatorname{cog}(\hat{\varepsilon}_1, \hat{U}_{(1)} w_2) \ , \text{ s.c. } \|w_2\| = 1 \implies w_{2k} = \frac{\operatorname{cog}(\hat{\varepsilon}_1, \hat{U}_{(1)k})}{\sqrt{\sum_{k=1}^{K} \operatorname{cog}^2(\hat{\varepsilon}_1, \hat{U}_{(1)k})}}.$$

On utilise à présent les composantes t_1 et t_2 pour établir un lien entre y et les régresseurs x_k :

$$t_2 = \sum_{k=1}^{K} w_{2k} \hat{U}_{(1)k} \implies y = \hat{c}_1 t_1 + \hat{c}_2 t_2 + \hat{\varepsilon}_2 \ .$$

La validation croisée permet de savoir si t_2 est significative.

• **Étape h :** Les régressions partielles sont réitérées en ajoutant l'influence de t_{h-1} :

$$R(x_k) = \beta t_1 + \cdots + \gamma t_{h-1} + \hat{U}_{(h-1)k} \ , \ \forall k = 1, \cdots, K.$$

D'où, après maximisation :

$$w_{hk} = \frac{\operatorname{cog}(\hat{\varepsilon}_{h-1}, \hat{U}_{(h-1)k})}{\sqrt{\sum_{k=1}^{K} \operatorname{cog}^2(\hat{\varepsilon}_{h-1}, \hat{U}_{(h-1)k})}} \ ,$$

$$t_h = \sum_{k=1}^{K} w_{hk} \cdot \hat{U}_{(h-1)k} \implies y = \alpha_2 + c_1 t_1 + \cdots + c_h t_h + \varepsilon_h \ .$$

La procédure s'arrête lorsque la validation croisée indique que la composante t_h n'est pas significative. L'algorithme Gini-PLS1 est valable si toutes les composantes t_h et t_l sont orthogonales, $\forall h \neq l$.

La validation croisée permet de trouver le nombre optimal $h > 1$ de composantes à retenir. Pour tester une composante t_h, on calcule la prédiction du modèle avec h composantes comprenant l'observation i, $\hat{y}_{h_i}$, puis sans l'observation i, $\hat{y}_{h_{(-i)}}$. L'opération est répétée pour tout i variant de 1 à n : on enlève à chaque fois l'observation i et on ré-estime le modèle. [9]

9. Les observations peuvent être éliminées par blocs *Cf.* Tenenhaus (1998), p. 77.

Pour mesurer la robustesse du modèle, on mesure l'écart entre la variable prédite et la variable observée :

$$PRESS_h = \sum_{i=1}^{n} \left(y_i - \hat{y}_{h_{(-i)}} \right)^2 .$$

La somme des carrés résiduels obtenue avec le modèle à $(h-1)$ composantes est :

$$RSS_{h-1} = \sum_{i=1}^{n} \left(y_i - \hat{y}_{(h-1)_i} \right)^2 .$$

Le critère RSS_h (Residual Sum of Squares) du modèle à h composantes et $PRESS_h$ (PRedicted Error Sum of Squares) sont comparés. Leur rapport permet de savoir si le modèle avec la composante t_h améliore la prédictibilité du modèle :

$$Q_h^2 = 1 - \frac{PRESS_h}{RSS_{h-1}} .$$

La composante t_h est retenue si : $\sqrt{PRESS_h} \leq 0,95\sqrt{RSS_h}$. Autrement dit, lorsque $Q_h^2 \geq 0,0975 = (1 - 0,95^2)$, la nouvelle composante t_h est significative, elle améliore la prévision de la variable y. Pour la significativité de la composante t_1, on utilise :

$$RSS_0 = \sum_{i=1}^{n} (y_i - \bar{y})^2 .$$

2.3.3 Régressions Gini-PLS généralisée

Schechtman & Yitzhaki [2003] ont récemment généralisé l'opérateur co-Gini afin d'imposer plus ou moins de poids en queue de distribution. Notons $r_k = (R_\downarrow(x_{1k}), \ldots, R_\downarrow(x_{nk}))$ le vecteur rang décroissant de la variable x_k, autrement dit, le vecteur qui assigne le rang le plus petit (1) à l'observation dont la valeur est la plus importante x_{ik} :

$$R_\downarrow(x_{ik}) := \begin{cases} n+1-\#\{x \leq x_{ik}\} & \text{pas d'observation similaire} \\ n+1-\frac{\sum_{i=1}^{p} \#\{x \leq x_{ik}\}}{p} & \text{si } p \text{ observations similaires } x_{ik}. \end{cases}$$

L'opérateur co-Gini est généralisé grâce au paramètre ν :

$$\text{cog}_\nu(x_\ell, x_k) := -\nu \text{cov}(x_\ell, r_k^{\nu-1});\ \nu > 1. \tag{2.3}$$

Afin de bien comprendre le rôle de l'opérateur co-Gini, revenons sur la mesure du coefficient de corrélation linéaire généralisé au sens de Gini :

$$GC_\nu(x_\ell, x_k) := \frac{-\nu \text{cov}(x_\ell, r_k^{\nu-1})}{-\nu \text{cov}(x_\ell, r_\ell^{\nu-1})} ;\ GC_\nu(x_k, x_\ell) := \frac{-\nu \text{cov}(x_k, r_\ell^{\nu-1})}{-\nu \text{cov}(x_k, r_k^{\nu-1})}.$$

Property 1 – Schechtman & Yitzhaki [2003] :

(i) $GC_\nu(x_\ell, x_k) \leq 1$.

(ii) *Si les variables x_ℓ et x_k sont indépendantes, pour tout $k \neq \ell$, alors* $GC_\nu(x_\ell, x_k) = GC_\nu(x_k, x_\ell) = 0$.

(iii) *Une transformation monotone des données φ n'affecte pas le coefficient de corrélation,* $GC_\nu(x_\ell, \varphi(x_k)) = GC_\nu(x_\ell, x_k)$.

(iv) *Pour une transformation linéaire φ,* $GC_\nu(\varphi(x_\ell), x_k) = GC_\nu(x_\ell, x_k)$ [*comme le coefficient de corrélation de Pearson*].

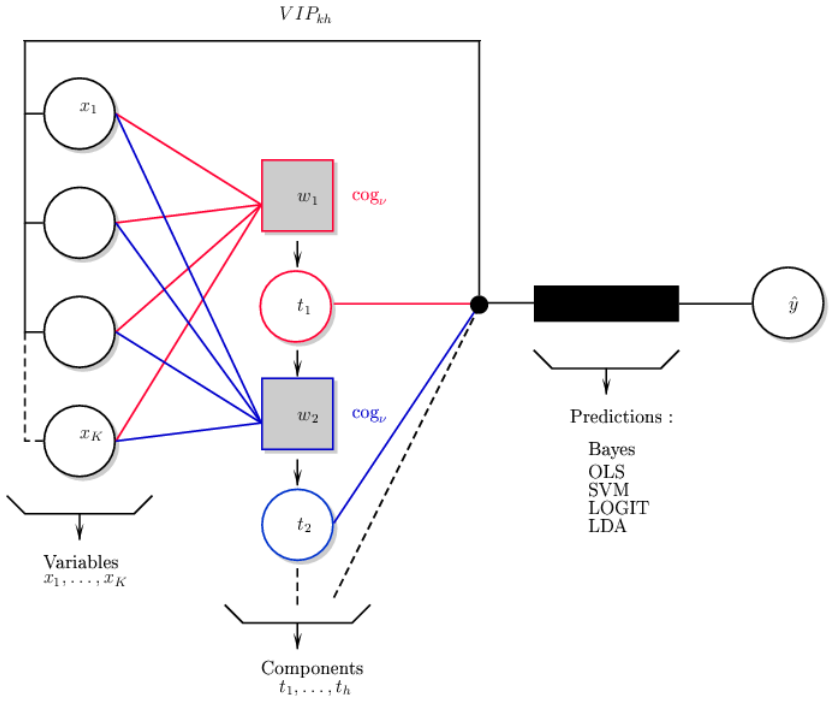
(v) *Si x_k et x_ℓ sont deux variables échangeables à une transformation linéaire près, alors* $GC_\nu(x_\ell, x_k) = GC_\nu(x_k, x_\ell)$.

Le rôle de l'opérateur co-Gini peut être expliqué de la manière suivante. Lorsque $\nu \to 1$, la variabilité des variables est atténuée de telle sorte que $\text{cog}_\nu(x_k, x_\ell)$ tend vers zéro (même si les variables x_k et x_ℓ sont fortement corrélées). Au contraire, si $\nu \to \infty$ alors $\text{cog}_\nu(x_k, x_\ell)$ permet de se focaliser sur les queues de distribution x_ℓ. Comme le montrent Olkin & Yitzhaki [1992], l'emploi de l'opérateur co-Gini atténue la présence d'outliers, du fait que le vecteur rang agit comme un instrument dans la régression de y sur X (régression par variables instrumentales).

Ainsi, en proposant une régression Gini-PLS basée sur le paramètre ν, nous pouvons calibrer la puissance du débruitage grâce à l'opérateur co-Gini qui va localiser le bruit dans la distribution. Cette régression Gini-PLS généralisée devient une régression Gini-PLS régularisée où le paramètre ν joue le rôle de paramètre de régularisation.

2.3.3.1 L'algorithme Gini-PLS généralisé

Dans ce qui suit nous généralisons la régression Gini-PLS de Mussard & Souissi-Benrejab [2018] avec renforcement du pouvoir de débruitage par l'intermédiaire du paramètre ν.



La première étape consiste à trouver des poids de débruitage associés à chaque variable x_k afin d'en déduire la première composante t_1 (ou première variable latente). Cette opération est bouclée jusqu'à la composante t_{h^*}, où h^* est le nombre optimal de variable latentes. Ainsi, le modèle est estimé :

$$y = \sum_{h=1}^{h^*} c_h t_h + \varepsilon_h. \tag{2.4}$$

La statistique VIP_{hj} est mesurée afin de sélectionner la variable x_j qui a l'impact significatif le plus important sur y estimé. Les variables les plus significatives sont celles dont $VIP_{hj} > 1$ avec :

$$VIP_{hj} := \sqrt{\frac{K \sum_{\ell=1}^{h} Rd(y; t_\ell) w_{\ell j}^2}{Rd(y; t_1, \ldots, t_h)}}$$

et

$$Rd(y; t_1, \ldots, t_h) := \frac{1}{K} \sum_{\ell=1}^{h} \text{cor}^2(y, t_\ell) =: \sum_{\ell=1}^{h} Rd(y; t_\ell).$$

où $\text{cor}^2(y, t_\ell)$ est le coefficient de corrélation de Pearson entre y et la composante t_ℓ. Cette information est rétro-propagée dans le modèle (une seule

fois) afin d'obtenir les variables latentes t_{h^*} et leurs coefficients estimés $\hat{c}_{h^*}$ sur les données d'entraînement. La variable cible y est ensuite prédite grâce à (2.4). Cette prévision est comparée aux modèles standards SVM, LOGIT, Bayes et LDA lorsque les données tests sont projetées dans le sous-espace $\{t_1, \ldots, t_{h^*}\}$.

Algorithme 2 : Gini-PLS Généralisé

Résultat : Prédiction du juge $y = 0; 1$

répéter
  répéter
    $\max \text{cog}_\nu(y, w_h X)$ s.t. $\|w_h\| = 1 \Longrightarrow$ poids w_h de X ;
    MCO équation : $y = \sum_h c_h t_h + \varepsilon_h$;
    MCO équation : $R(x_j) = \sum_h \beta_h t_h + \epsilon_k \; \forall k = 1, \ldots, K$;
    $X := (\hat{\epsilon}_1, \ldots, \hat{\epsilon}_K)$ $y := \hat{\varepsilon}_h$;
  jusqu'à $h = 10$ $[h = h + 1]$;
  Mesurer VIP_{kh}, Q^2_h ;
  Sélectionner le nombre optimal de composantes h^* ;
jusqu'à $\nu = 14$ $[\nu = \nu + 2]$;
Déduire le paramètre optimal ν^* qui minimise l'erreur ;
retourner Prédiction $\hat{y}$ avec Gini-PLS (h^*, ν^*) ;
retourner Prédiction $\hat{y}$ avec SVM, LOGIT, Bayes, LDA sur les composantes $(t_1, \ldots, t_{h^*})$;

2.3.3.2 L'algorithme LOGIT-Gini-PLS généralisé

Comme nous le constatons dans l'algorithme Gini-PLS généralisé que nous avons proposé dans le section précédente, les poids w_j proviennent de l'opérateur co-Gini appliqué à une variable booléenne $y \in \{0; 1\}$. Afin de trouver les poids w_j qui maximisent le lien entre les variables x_j et la variable cible y, nous proposons d'utiliser la régression LOGIT, autrement dit, une sigmoïde qui est mieux adaptée aux variables booléennes. Ainsi, dans chaque étape de la régression Gini-PLS nous remplaçons la maximisation du co-Gini par la mesure de la probabilité conditionnelle suivante :

$$\mathbb{P}(y_i = 1/X = X_i) = \frac{\exp\{X_i\beta\}}{1 + \exp\{X_i\beta\}} \qquad \text{(LOGIT)}$$

où X_i est la i-ème ligne de la matrice X (observation des caractéristiques/-dimensions de la décision juridique i). L'estimation du vecteur β se fait par

maximum de vraisemblance. On en déduit alors les pondérations w_j :

$$w_j = \frac{\beta_j}{\|\beta\|}$$

L'algorithme LOGIT-Gini-PLS généralisé est donc le suivant :

Algorithme 3 : LOGIT-Gini-PLS Généralisé

Résultat : Prédiction du juge $y = 0; 1$

répéter
  répéter
    LOGIT équation : $\Longrightarrow$ poids w_j de X ;
    MCO équation : $y = \sum_h c_h t_h + \varepsilon_h$;
    $X := (\hat{\epsilon}_1, \ldots, \hat{\epsilon}_K)$ $y := \hat{\varepsilon}_h$;
  jusqu'à $h = 10$ $[h = h + 1]$;
  Mesurer VIP_{kh}, Q_h^2 ;
  Sélectionner le nombre optimal de composantes h^* ;
jusqu'à $\nu = 14$ $[\nu = \nu + 2]$;
Déduire le paramètre optimal ν^* qui minimise l'erreur ;
retourner Prédiction $\hat{y}$ avec Gini-PLS (h^*, ν^*) ;
retourner Prédiction $\hat{y}$ avec SVM, LOGIT, Bayes, LDA sur les composantes $(t_1, \ldots, t_{h^*})$;

2.4 Expérimentations et résultats

Nous discutons ici les performances de divers algorithmes populaires et l'impact de la quantité et du déséquilibre des données, de l'heuristique, et de la restriction explicite des documents aux passages relatifs à la catégorie de demandes, ainsi que leur capacité à faire abstraction des autres demandes du document. Ces expériences visent aussi à comparer l'efficacité du Gini-Logit-PLS par rapport à d'autres analyses discriminantes. Comme Im *et al.* [2017], différentes combinaisons d'algorithmes de classification et méthodes de pondération sont comparées ; ce qui représente un total de 8 algorithmes * 11 métriques globales * 5 métriques locales = 440 configurations (+ 2 algorithmes i.e. FastText et NBSVM). La méthode Gini-Logit-PLS réalisant une projection dans un espace de petite dimension, elle a été comparée aussi à d'autres méthode de réduction de dimension.

2.4.1 Protocole d'évaluation

Deux métriques d'évaluation sont utilisées : la précision et la F_1-mesure. Pour tenir compte du déséquilibre entre les classes, la moyenne macro est préférée. Il s'agit de l'agrégation de la contribution individuelle de chaque classe : $F_{1macro} = \frac{2 \times P_{macro} \times R_{macro}}{P_{macro} + R_{macro}}$, où les macro-moyennes de la précision (P_{macro}) et du rappel (R_{macro}) sont calculées en fonction des nombres moyens de vrais positifs ($\overline{TP}$), faux positifs ($\overline{FP}$), et faux négatifs ($\overline{FN}$) comme suit [Van Asch, 2013] : $P_{macro} = \frac{\overline{TP}}{\overline{TP}+\overline{FP}}$, $R_{macro} = \frac{\overline{TP}}{\overline{TP}+\overline{FN}}$.

Les données utilisées sont une restriction, des données du chapitre précédent, aux documents n'ayant qu'une seule demande annotée pour chacune des catégories de demande. Le déséquilibre entre les classes est illustrée par la figure 2.3. En effet, les demandes sont en majorité rejetées pour les catégories ACPA, CONCDEL, DANAIS et STYX. Le contraire est observé pour DCPPC, et le rapport est légèrement équilibré pour DORIS.

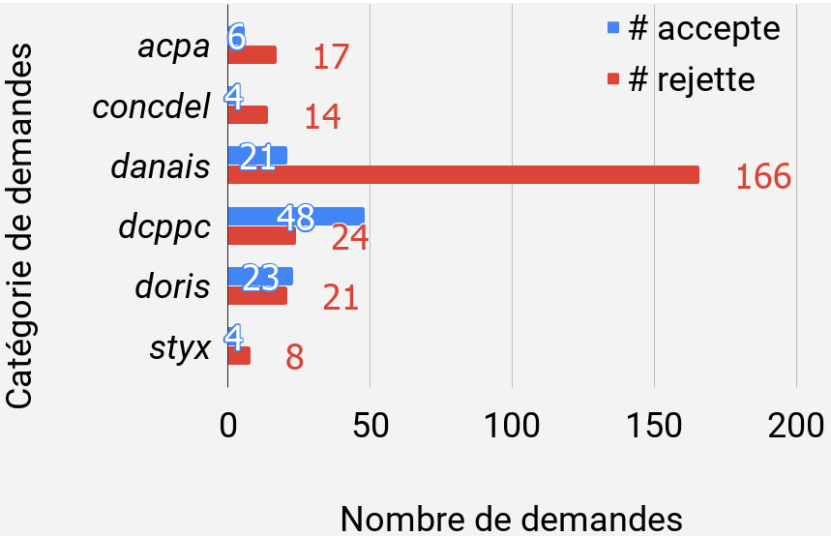


Figure 2.3 – Répartition des documents entre *accepte* et *rejette*.

L'efficacité des algorithmes dépend souvent des méta-paramètres dont il faut déterminer des valeurs optimales. Scikit-learn implémente deux stratégies de recherche des ces valeurs : RandomSearch et GridSearch. Malgré la rapidité de la méthode RandomSearch, elle est non déterministe et les valeurs qu'elle trouve donnent une prédiction moins précise que les valeurs par défaut. Idem pour la méthode GridSearch, qui est très lente, et donc peu pratique face au grand nombre de configurations à évaluer. Par conséquent, les valeurs utilisées pour les expérimentations sont les valeurs par défaut définies par Scikit-learn (Tableau 2.2).

algorithmes	hyper-paramètres
SVM	$C = 1.0; \gamma = \frac{1}{n\_features * var(X)}; noyau = RBF$ (fonction de base radiale)
KNN	$k = 5, weights =' uniform', algorithm =' auto'$
LDA	$solver =' svd', n\_components = 10$ [10]
QDA	
Arbre	critère de séparation='gini'
NBSVM	$--ngram = 123, linearSVM,$
FastText	-minCount=5, -wordNgrams=1, -lr=0.05,
Gini-PLS	$h = n\_components = 10$
Logit-PLS	$h = n\_components = 10$
Gini-Logit-PLS	$h = n\_components = 10; \nu = 14$

Tableau 2.2 – Valeurs utilisées pour les méta-paramètres des algorithmes de classification.

2.4.2 Classification de l'ensemble du document

En représentant l'ensemble du document à l'aide de diverses représentations vectorielles, les algorithmes sont comparés avec les représentations qui leurs sont optimales. On remarque d'après les résultats du Tableau 2.3 que les arbres sont en moyenne meilleurs sur l'ensemble des catégories même si en moyenne la F_1-mesure moyenne est limité à 0.668. Les résultats des extensions du PLS ne sont pas très éloignées de ceux des arbres avec des différences de F_1 à moins de 0.1 (si on choisit le bon schéma de "vectorisation").

ajouter les F_1 ou erreur de rejette et de accepte
mettre aussi en avant les analyses discriminantes comme réducteur de dimension et comme classifieurs

Vecteur	**algorithme**	F_1	**min**	**Cat. min**	**max**	**Cat. max**	F_1 **- 1er**F_1	**max - min**	**rang**
GSS*TF	Arbre	0.668	0.5	doris	0.92	dcppc	0	0.42	1
AVG-G*TF	LogitPLS	0.648	0.518	danais	0.781	dcppc	0.02	0.263	13
AVG-G*TF	StandardPLS	0.636	0.49	danais	0.836	dcppc	0.032	0.346	24
DELTADF*TF	GiniPLS	0.586	0.411	danais	0.837	dcppc	0.082	0.426	169
DELTADF*TF	GiniLogitPLS	0.578	0.225	styx	0.772	dcppc	0.09	0.547	220
-	NBSVM	0.494	0.4	styx	0.834	dcppc	0.174	0.434	
-	FastText	0.412	0.343	doris	0.47	danais	0.256	0.127	

Tableau 2.3 – Comparaison des algorithmes sur une représentation globale des documents pour la détection du sens du résultat.

Les scores F_1 moyens des algorithmes NBSVM et FastText n'excédent en général pas 0.5 malgré qu'ils soient spécialement conçus pour les textes. On peut estimer qu'ils sont très sensibles au déséquilibre des données entre les catégories (plus de rejets que d'acceptations), soit il est plus difficile de détecter l'acceptation des demandes. En effet, ces algorithmes classent toutes les données test avec le label (sens) majoritaire i.e. le rejet, et par conséquent, ils ne détectent quasiment pas d'acceptation de demande. Le cas des catégories DORIS et DCPPC pour le NBSVM ($F_{1macro} = 0.834$)

tend à démontrer la forte sensibilité aux cas négatifs de ces algorithmes puisque même avec presque autant de labels "accepte" que "rejette", la F_1-mesure de "rejette" est toujours supérieure à celle de "accepte" (Tableau 2.4).

Cat. Dmd.	**Algo.**	**Préc.**	**Préc. équi.**	**err-0**	**err-1**	$F_1(0)$	$F_1(1)$	F_{1macro}
dcppc	nbsvm	0.875	0.812	0.375	0	0.752	0.916	**0.834**
danais	fasttext	0.888	0.5	0	1	0.941	0	0.47
danais	nbsvm	0.888	0.5	0	1	0.941	0	0.47
concdel	fasttext	0.775	0.5	0	1	0.853	0	0.437
concdel	nbsvm	0.775	0.5	0	1	0.873	0	0.437
acpa	fasttext	0.745	0.5	0	1	0.853	0	0.426
acpa	nbsvm	0.745	0.5	0	1	0.853	0	0.426
doris	nbsvm	0.5	0.492	0.85	0.167	0.174	0.63	0.402
dcppc	fasttext	0.667	0.5	1	0	0.8	0	0.4
styx	fasttext	0.667	0.5	1	0	0.8	0	0.4
styx	nbsvm	0.667	0.5	0	1	0.8	0	0.4
doris	fasttext	0.523	0.5	1	0	0	0.686	0.343

$0 ==' accepte'; 1 ==' rejette'$

Tableau 2.4 – Détails des résultats de FastText et NBSVM.

2.4.3 Réduction du document aux régions comprenant le vocabulaire de la catégorie

Etant donné que les décisions portent sur plusieurs catégories de demande, nous avons expérimenté la restriction du document aux passages comprenant du vocabulaire de la catégorie d'intérêt : demande, résultat, résultat antérieur (resultat_a), paragraphes dans les motifs (motifs). Les combinaisons passages-représentation vectorielle-algorithme sont comparées dans le Tableau 2.5. Les résultats s'améliorent énormément avec les réductions, sauf pour la catégorie DORIS. La meilleure restriction combine les passages comprenant le vocabulaire de la catégorie dans la section Litige (demande et résultat antérieur), dans la section Motifs (contexte), et dans la section Dispositif (résultat).

2.5 Conclusion

L'étude de ce chapitre tente de simplifier l'extraction du sens du résultat rendu par les juges sur une demande de catégorie donnée. Elle a

Catégorie	zone	Vecteur (pondération)	classifieur	F_1
acpa	**demande_resultat_a_resultat_context**	**DBIDF*TF**	**Tree**	**0.846**
	litige_motifs_dispositif	DELTADF*TF	StandardPLS	0.697
	litige_motifs_dispositif	AVERAGEGlobals*TF	LogitPLS	0.683
concdel	**litige_motifs_dispositif**	**GSS*TF**	**Tree**	**0.798**
	motifs	IDF*TF	GiniLogitPLS	0.703
	context	DBIDF*LOGAVE	StandardPLS	0.657
danais	**demande_resultat_a_resultat_context**	**CHI2*AVERAGELocals**	**Tree**	**0.813**
	demande_resultat_a_resultat_context	AVERAGEGlobals*ATF	LogitPLS	0.721
	demande_resultat_a_resultat_context	AVERAGEGlobals*ATF	StandardPLS	0.695
dcppc	**demande_resultat_a_resultat_context**	**CHI2*TF**	**Tree**	**0.985**
	demande_resultat_a_resultat_context	CHI2*TF	LogitPLS	0.94
	litige_motifs_dispositif	MARASCUILO*TP	StandardPLS	0.934
doris	**litige_motifs_dispositif**	**DSIDF*TP**	**GiniPLS**	**0.806**
	litige_motifs_dispositif	DSIDF*TP	GiniLogitPLS	0.806
	litige_motifs_dispositif	IG*ATF	StandardPLS	0.772
styx	**motifs**	**DSIDF*TF**	**Tree**	**1**
	demande_resultat_a_resultat_context	DSIDF*LOGAVE	GiniLogitPLS	0.917
	litige_motifs_dispositif	RF*TF	GiniPLS	0.833

Tableau 2.5 – Détection du sens du résultat : Comparaison des réductions du document.

consisté à formuler le problème comme une tâche de classification de documents. On évite ainsi de passer par la détection ad-hoc [11] des passages et données à l'aide de termes-clés qui est un inconvénient de la méthode à règles du chapitre précédent car elle n'est peut-être pas généralisable à tous types de décisions (i.e. il pourrait être nécessaire d'établir de nouvelles listes de mots-clés pour d'autres domaines). Au total dix algorithmes de classification on été expérimentés sur 55 méthodes de représentations vectorielles de texte. Nous avons remarqué que les résultats de classification sont principalement influencés par 3 caractéristiques de nos données. Tout d'abord, le très faible nombre d'exemples d'entraînement défavorise certains algorithmes (sensibilité aux valeurs aberrantes ou *outliers*), comme par exemple FastText qui nécessite plusieurs milliers d'exemples pour mettre à jour le pas du gradient (*learning rate*). Ensuite, le fort déséquilibre entre les classes ("accepte" vs. "rejette") rend difficile la reconnaissance de la classe minoritaire qui est généralement la classe "accepte". Le fort gap entre les erreurs sur "rejette" et celles sur "accepte", ainsi que les bons résultats obtenus sur DCPPC en sont la preuve. Enfin, la présence d'autres catégories de demande dans le document dégrade l'efficacité de la classification parce que les algorithmes ne parviennent pas seuls à retrouver les éléments en rapport direct avec la catégorie choisie. Ceci est démontré par l'impact positif de la restriction du contenu à classer à certains passages particuliers, même si la restriction adéquate est fonction de

11. i.e. spécialement conçu pour nos données.

la catégorie.

Au final, les arbres de décision sont adaptés pour la tâche, mais l'usage du Gini-PLS et du Gini-Logit-PLS permet d'obtenir des performances assez proches de celles des arbres. Il serait intéressant de combiner ces variantes de l'analyse PLS, à d'autres comme le Sparse-PLS qui pourrait peut-être aider à résoudre le problème de vecteurs/matrices creuses dont sont victimes les représentations vectorielles de texte. Il existe aussi un grand nombre d'architectures neuronales pour la classification de document et de très grands nombres de métriques de pondération de termes pour la représentation des textes, mais aucune ne semble s'adapter à toutes les catégories. Par conséquent, une étude sur l'usage des représentations par plongement sémantique comme Word2Vec [Mikolov *et al.* , 2013], Sent2Vec [Pagliardini *et al.* , 2018] ou Doc2Vec [Le & Mikolov, 2014] serait intéressante.

Bibliographie

Aggarwal, Charu C., Hinneburg, Alexander, & Keim, Daniel A. 2001. On the surprising behavior of distance metrics in high dimensional space. *Pages 420–434 of : International Conference on Database Theory*. Springer.

Ahn, David. 2006. The stages of event extraction. *Pages 1–8 of : Proceedings of the Workshop on Annotating and Reasoning about Time and Events*. Association for Computational Linguistics.

Amami, Rimah, Ayed, Dorra Ben, & Ellouze, Noureddine. 2013. Practical Selection of SVM Supervised Parameters with Different Feature Representations for Vowel Recognition. *International Journal of Digital Content Technology and its Applications (JDCTA)*, **7**(9).

Bakkelund, Daniel. 2009. An LCS-based string metric. *Olso, Norway : University of Oslo*.

Bazzoli, Caroline, & Lambert-Lacroix, Sophie. 2018. Classification based on extensions of LS-PLS using logistic regression : application to clinical and multiple genomic data. *BMC bioinformatics*, **19**(1), 314.

Ben-Hur, Asa, & Weston, Jason. 2010. A User's Guide to Support Vector Machines. *Chap. 13, pages 223–239 of : Data Mining Techniques for the Life Sciences*. Totowa, NJ : Humana Press.

Breiman, Leo. 2001. Random Forests. *Machine Learning*, **45**(1), 5–32.

Breiman, Leo, Friedman, J. H., Olshen, R. A., & Stone, C. J. 1984. *Classification and Regression Trees*. Statistics/Probability Series. Belmont, California, U.S.A. : Wadsworth Publishing Company.

Brown, Ralf D. 2013. Selecting and weighting n-grams to identify 1100 languages. *Pages 475–483 of : International Conference on Text, Speech and Dialogue*. Springer.

Cortes, Corinna, & Vapnik, Vladimir. 1995. Support-vector networks. *Machine Learning*, **20**(3), 273–297.

Cover, Thomas, & Hart, Peter. 1967. Nearest Neighbor Pattern Classification. *IEEE Transactions on Information Theory*, **13**(1), 21–27.

Dong, Yan-Shi, & Han, Ke-Song. 2005. Boosting SVM classifiers by ensemble. *Pages 1072–1073 of : Special Interest Tracks And Posters Of The 14th International Conference On World Wide Web*. ACM.

Duda, Richard O., Hart, Peter E., *et al.* . 1973. *Pattern Classification And Scene Analysis*. Vol. 3. New York : John Wiley & Sons.

Duda, Richard O., Hart, Peter E., & Stork, David G. 2000. *Pattern classification*. 2nd edn. John Wiley & Sons.

Dudani, Sahibsingh A. 1976. The Distance-weighted k-Nearest-Neighbor Rule. *IEEE Transactions on Systems, Man, and Cybernetics*, **SMC-6**(4), 325–327.

Durif, Ghislain, Modolo, Laurent, Michaelsson, Jakob, Mold, Jeff E., Lambert-Lacroix, Sophie, & Picard, Franck. 2017. High dimensional classification with combined adaptive sparse PLS and logistic regression. *Bioinformatics*, **34**(3), 485–493.

Elsaadawy, AbdAllah, Torki, Marwan, & Ei-Makky, Nagwa. 2018. A text classifier using weighted average word embedding. *Pages 151–154 of : 2018 international japan-africa conference on electronics, communications and computations (jac-ecc)*. IEEE.

Frank, Eibe, Hall, Mark A., & Witten, Ian H. 2016. *The WEKA workbench*. Fourth edn. Morgan Kaufmann. `https://www.cs.waikato.ac.nz/ml/weka/Witten_et_al_2016_appendix.pdf`. Page 128 p.

Frantzi, Katerina, Ananiadou, Sophia, & Mima, Hideki. 2000. Automatic recognition of multi-word terms :. the c-value/nc-value method. *International journal on digital libraries*, **3**(2), 115–130.

Fukuaga, Keinosuke. 1990. *Introduction to Statistical Pattern Recognition*. Academic Press.

Galavotti, Luigi, Sebastiani, Fabrizio, & Simi, Maria. 2000. Experiments on the use of feature selection and negative evidence in automated text categorization. *Pages 59–68 of : International Conference on Theory and Practice of Digital Libraries*. Springer.

Gou, Jianping, Xiong, Taisong, & Kuang, Yin. 2011. A Novel Weighted Voting for K-Nearest Neighbor Rule. *JCP*, **6**(5), 833–840.

Grave, E., Mikolov, T., Joulin, A., & Bojanowski, P. 2017. Bag of tricks for efficient text classification. *Pages 427–431 of : Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*.

Harispe, Sébastien, Ranwez, Sylvie, Janaqi, Stefan, & Montmain, Jacky. 2013. The semantic measures library and toolkit : fast computation of semantic similarity and relatedness using biomedical ontologies. *Bioinformatics*, **30**(5), 740–742.

Hirschberg, Daniel S. 1977. Algorithms For The Longest Common Subsequence Problem. *Journal of the ACM (JACM)*, **24**(4), 664–675.

Im, Chan Jong, Mandl, Thomas, *et al.* . 2017. Text Classification for Patents : Experiments with Unigrams, Bigrams and Different Weighting Methods. *International Journal of Contents*, **13**(2).

Jones, K. Sparck, Walker, Steve, & Robertson, Stephen E. 2000. A Probabilistic Model Of Information Retrieval : Development And Comparative Experiments. *Information Processing & Management*, **36**(6), 809–840.

Joulin, Armand, Grave, Edouard, Bojanowski, Piotr, Douze, Matthijs, Jégou, Hérve, & Mikolov, Tomas. 2016. Fasttext. zip : Compressing text classification models. *arxiv preprint arxiv :1612.03651*.

Kittler, Josef, Hater, Mohamad, & Duin, Robert P.W. 1996. Combining classifiers. *Pages 897–901 of : Proceedings of 13th international conference on pattern recognition*, vol. 2. IEEE.

Kuncheva, Ludmila I. 2004. *Combining pattern classifiers : methods and algorithms*. John Wiley & Sons.

Lacroux, Alain. 2011. Les avantages et les limites de la méthode «Partial Least Square »(PLS) : une illustration empirique dans le domaine de la GRH. *Revue de gestion des ressources humaines*, **80**(2), 45–64.

Lan, Man, Tan, Chew Lim, Su, Jian, & Lu, Yue. 2009. Supervised and traditional term weighting methods for automatic text categorization. *IEEE transactions on pattern analysis and machine intelligence*, **31**(4), 721–735.

LDC, (Linguistic Data Consortium). 2005. *ACE (Automatic Content Extraction) English Annotation Guidelines for Events*. 5.4.3 edn. Linguistic Data Consortium. `https://www.ldc.upenn.edu/sites/www.ldc.upenn.edu/files/english-events-guidelines-v5.4.3.pdf`.

LDC, (Linguistic Data Consortium). 2008. *ACE (Automatic Content Extraction) English Annotation Guidelines for Relations*. 6.2 edn. Linguistic Data Consortium. https://www.ldc.upenn.edu/sites/www.ldc.upenn.edu/files/english-relations-guidelines-v6.2.pdf.

Le, Quoc, & Mikolov, Tomas. 2014. Distributed representations of sentences and documents. *Pages 1188–1196 of : International conference on machine learning*.

Liu, Jingjing, Pasupat, Panupong, Cyphers, Scott, & Glass, Jim. 2013. ASGARD : A Portable Architecture For Multilingualdialogue Systems. *Pages 8386–8390 of : 2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE.

Liu, Yushu, & Rayens, William. 2007. PLS and dimension reduction for classification. *Computational Statistics*, **22**(2), 189–208.

Manning, Christopher D, Raghavan, Prabhakar, & Schütze, Hinrich. 2009. Scoring, term weighting and the vector space model. *Chap. 6, pages 109–133 of : Introduction to information retrieval*. Cambridge : Cambridge university press.

Marascuilo, Leonard A. 1966. Large-sample multiple comparisons. *Psychological bulletin*, **65**(5), 280.

Martineau, Justin, & Finin, Tim. 2009. Delta TFIDF : An Improved Feature Space for Sentiment Analysis. *In : Third International AAAI Conference on Weblogs and Social Media (ICWSM)*.

McLachlan, Geoffrey J. 1992. *Discriminant analysis and statistical pattern recognition*. John Wiley & Sons.

Mikolov, Tomas, Chen, Kai, Corrado, Greg, & Dean, Jeffrey. 2013. Efficient estimation of word representations in vector space. *In : Proceedings of the International Conference on Learning Representations (ICLR)*.

Mohammad, Fahim. 2018. Is preprocessing of text really worth your time for toxic comment classification ? *Pages 447–453 of : Proceedings on the international conference on artificial intelligence (icai)*. The Steering Committee of The World Congress in Computer Science, Computer ....

Mussard, Stéphane, & Souissi-Benrejab, Fattouma. 2018. Gini-PLS Regressions. *Journal of Quantitative Economics*, April, 1–36.

Ng, Hwee Tou, Goh, Wei Boon, & Low, Kok Leong. 1997. Feature selection, perceptron learning, and a usability case study for text categorization. *Pages 67–73 of : ACM SIGIR Forum*, vol. 31. ACM.

Nguyen, Thien Huu, Cho, Kyunghyun, & Grishman, Ralph. 2016. Joint Event Extraction via Recurrent Neural Networks. *Pages 300–309 of : HLT-NAACL*.

Nigam, Kamal, Lafferty, John, & McCallum, Andrew. 1999. Using maximum entropy for text classification. *Pages 61–67 of : IJCAI-99 Workshop on Machine Learning for Information Filtering*, vol. 1.

Olkin, Ingram, & Yitzhaki, Shlomo. 1992. Gini regression analysis. *International Statistical Review/Revue Internationale de Statistique*, 185–196.

Pagliardini, Matteo, Gupta, Prakhar, & Jaggi, Martin. 2018. Unsupervised Learning of Sentence Embeddings using Compositional n-Gram Features. *In : NAACL 2018 - Conference of the North American Chapter of the Association for Computational Linguistics*.

Palm, Rasmus Berg, Hovy, Dirk, Laws, Florian, & Winther, Ole. 2017. End-to-End Information Extraction without Token-Level Supervision. *In : Proceedings of the Workshop on Speech-Centric Natural Language Processing*.

Paltoglou, Georgios, & Thelwall, Mike. 2010. A study of information retrieval weighting schemes for sentiment analysis. *Pages 1386–1395 of : Proceedings of the 48th annual meeting of the association for computational linguistics*. Association for Computational Linguistics.

Price, Patti J. 1990 (Jun). Evaluation Of Spoken Language Systems : The ATIS Domain. *Pages 91–95 of : Proceedings of the Speech and Natural Language Workshop of the Human Language Technology Conference*.

Quinlan, J. Ross. 1993. C4.5 : Programming for machine learning. *Morgan Kauffmann*, **38**, 48.

Raschka, Sebastian. 2014. *Naive Bayes and Text Classification I : Introduction and Theory*. preprint arXiv :1410.5329 [cs.LG].

Rish, Irina. 2001. An Empirical Study Of The Naive Bayes Classifier. *Pages 41–46 of : IJCAI 2001 Workshop On Empirical Methods In Artificial Intelligence*, vol. 3. IBM New York.

Ruparel, Nidhi H, Shahane, Nitin M, & Bhamare, Devyani P. 2013. Learning from small data set to build classification model : A survey. *Internationla Journal of Computer Applications*, **975**(8887), 23–26.

Salton, Gerard, & Buckley, Christopher. 1988. Term-weighting Approaches In Automatic Text Retrieval. *Information Processing & Management*, **24**(5), 513–523.

Schechtman, Edna, & Yitzhaki, Shlomo. 2003. A family of correlation coefficients based on the extended Gini index. *The Journal of Economic Inequality*, **1**(2), 129–146.

Schütze, Hinrich, Hull, David A, & Pedersen, Jan O. 1995. A comparison of classifiers and document representations for the routing problem. *Pages 229–237 of : Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM.

Shirsath, Ashlesha. 2017. Two stage smart crawler with nbsvm classifier. *International research journal of engineering and technology*, **4**(07), 2051–2054.

Singh, Sonia, & Gupta, Priyanka. 2014. Comparative study ID3, CART and C4.5 decision tree algorithm : a survey. *International Journal of Advanced Information Science and Technology (IJAIST)*, **27**, 97–103.

Sohangir, Sahar, & Wang, Dingding. 2017. Improved sqrt-cosine similarity measurement. *Journal of Big Data*, **4**(1), 25.

Sparck Jones, Karen. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, **28**(1), 11–21.

Tenenhaus, Michel. 2005. La regression logistique PLS. *Chap. 12, pages 263–276 of :* Droesbeke, Jean-Jacques and Lejeune, Michel and Saporta, Gilbert (ed), *Modèles statistiques pour données qualitatives*. Editions Technip.

Tulyakov, Sergey, Jaeger, Stefan, Govindaraju, Venu, & Doermann, David. 2008. Review of classifier combination methods. *Pages 361–386 of : Machine learning in document analysis and recognition*. Springer.

Van Asch, Vincent. 2013. *Macro- and micro-averaged evaluation measures*. Tech. rept. Computational Linguistics & Psycholinguistics (CLiPS), Belgium. `https://pdfs.semanticscholar.org/1d10/6a2730801b6210a67f7622e4d192bb309303.pdf`.

Vapnik, Vladimir N. 1995. *The Nature of Statistical Learning Theory*. Springer.

Vinyals, Oriol, Fortunato, Meire, & Jaitly, Navdeep. 2015. Pointer networks. *Pages 2692–2700 of : Advances in Neural Information Processing Systems*.

Wang, Sida, & Manning, Christopher D. 2012. Baselines and bigrams : Simple, good sentiment and topic classification. *Pages 90–94 of : Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics : Short Papers-Volume 2*. Association for Computational Linguistics.

Wold, Herman. 1966. Estimation of principal components and related models by iterative least squares. *Multivariate Analysis*, 391–420.

Wu, Haibing, Gu, Xiaodong, & Gu, Yiwei. 2017. Balancing between over-weighting and under-weighting in supervised term weighting. *Information Processing & Management*, **53**(2), 547–557.

Wu, Harry, & Salton, Gerard. 1981. A comparison of search term weighting : term relevance vs. inverse document frequency. *Pages 30–39 of : ACM SIGIR Forum*, vol. 16. ACM.

Wyner, Adam Z. 2010. Towards annotating and extracting textual legal case elements. *Informatica e Diritto : special issue on legal ontologies and artificial intelligent techniques*, **19**(1-2), 9–18.

Yang, Bishan, & Mitchell, Tom. 2016. Joint Extraction of Events and Entities within a Document Context. *Pages 289–299 of : Proceedings of NAACL-HLT*.

Yang, Yiming, & Pedersen, Jan O. 1997. A comparative study on feature selection in text categorization. *Pages 412–420 of : ICML*, vol. 97.

Zeng, Xue-Qiang, Wang, Ming-Wen, & Nie, Jian-Yun. 2007. Text classification based on partial least square analysis. *Pages 834–838 of : Proceedings of the 2007 ACM symposium on Applied computing*. ACM.