UNIVERSITÉ DU QUÉBEC À MONTRÉAL

APPROCHES D'APPRENTISSAGE AUTOMATIQUE POUR LA DÉTECTION DU SPAM WEB : EXPLORATION DE DIVERSES CARACTÉRISTIQUES

MÉMOIRE

PRÉSENTÉ

COMME EXIGENCE PARTIELLE

DE LA MAÎTRISE EN INFORMATIQUE

PAR

FATIMA AIT MAHAMMED

MARS 2018

REMERCIEMENTS

En premier lieu, je remercie Dieu de m'avoir permis de mener ce travail de recherche à terme.

Ce travail n'aurait pu aboutir sans la contribution d'un nombre de personnes, ainsi se présente l'occasion de les remercier.

Je tiens à remercier mon directeur de recherche, Monsieur Hakim Lounis, qui a supervisé mon travail tout en me laissant une grande marge de liberté. Je le remercie pour son encadrement, sa disponibilité et la pertinence de ses remarques tout au long de la réalisation de ce projet de maîtrise et aussi pour son soutien financier.

Merci également à tous les professeurs de l'UQÀM avec qui j'ai suivi des cours tout au long de la maîtrise. Je remercie également la faculté des sciences de l'UQÀM pour les bourses d'excellence que j'ai reçues durant mes études de maîtrise.

Merci à ma mère qui m'est la plus chère, pour sa patience, ses conseils qui ont éclairé mon chemin, et soutenu tout au long de ma vie. À toi maman, tu m'as toujours poussé vers le sérieux et le travail, et maintenant c'est grâce à toi et pour toi que j'arrive là.

Ce mémoire n'aurait pas vu le jour sans la contribution de mon cher mari, Méziane, qui été toujours à côté de moi dans les moments délicats. Je ne saurai assez le remercier, pour son soutien moral et sa présence. C'est grâce à ton aide et à ta patience avec moi que ce travail a pu voir le jour.

Je tiens à remercier ma famille, qui ont toujours trouvé les mots pour m'encourager. Ma plus profonde reconnaissance pour votre soutien.

DÉDICACE

À tous ceux qui se sentent fiers de ce travail

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

RÉSUMÉ

Face à l'augmentation de l'information disponible sur le Web, la quantité de données textuelles disponibles pour les utilisateurs est devenue très importante. Selon un dernier sondage[1], la taille du Web est d'au moins 4,8 milliards de pages, dont plusieurs sont, soit dupliquées soit du spam. Les pages qui trompent les algorithmes de classement dans les moteurs de recherche afin d'avancer leur classement dans les résultats des moteurs de recherche forment le Spam Web. Étant donné que les utilisations malveillantes dans le Web sont devenues massives, le besoin en techniques automatisées, capables d'analyser des données afin de détecter les sources malveillantes, est devenu primordial. L'application des techniques d'apprentissage automatique dans le contexte de la cybercriminalité est très prometteuse et commence à donner des résultats en termes d'applications conçues et d'articles publiés. Ces techniques sont de plus en plus accessibles et utilisées de manière intensive.

De nombreux chercheurs travaillent à détecter les pages de spam. Cependant, il n'existe pas de technique efficace universelle jusqu'ici qui puisse détecter toutes les pages de spam. Ce travail est un effort dans cette direction. Nous proposons une approche basée sur le contenu pour identifier les pages spam. Dans ce travail, nous explorons des caractéristiques pour classer une page Web comme spam ou non-spam. Nous expérimentons quelques méthodes d'apprentissage automatique pour classer deux ensembles de données, l'un avec les attributs que nous avons extraits et l'autre avec une combinaison des meilleurs attributs explorés et des attributs existants pour détecter les hôtes spam. Nous avons utilisé pour cela, l'ensemble de données Web Spam UK-2007. Les résultats ont été comparés à certaines approches existantes. Un bon taux de F-mesure (0,968) et de surface sous la courbe ROC (AUC) démontre l'efficacité des méthodes d'apprentissage pour la détection de spam dans le Web.

MOTS-CLÉS : Apprentissage automatique, détection de spam Web, préparation des données, extraction de caractéristiques à base de contenu, spam de contenu

[1] http://www.worldwidewebsize.com

INTRODUCTION

0.1 Introduction générale

Supposons un scénario dans lequel, vous formulez une requête dans un moteur de recherche populaire comme Google, mais vous n'obtenez pas de réponse satisfaisante, même au sein des meilleurs résultats. C'est ce que l'on appelle le spam Web : c'est une tentative de manipuler le classement des algorithmes de moteur de recherche afin de stimuler le classement de pages spéciales dans les résultats des moteurs de recherche (Shekoofeh et Alireza, 2013).

Le Web est en train de devenir une source importante de divertissement, de communication, de recherche, de nouvelles et de commerce. La raison en est que les gens utilisent les moteurs de recherche plus fréquemment qu'auparavant. Les moteurs de recherche (Google, Yahoo !, Bing, etc.) sont essentiels pour les utilisateurs qui souhaitent accéder à des informations pertinentes.

Selon une étude réalisée par Jansen et Spink (Bernard et Amanda, 2003), environ 80% des utilisateurs de moteurs de recherche ne prennent pas en considération les résultats placés au-delà de la troisième page. Cependant, les sites Web sont en concurrence pour attirer l'attention des utilisateurs et avoir de la visibilité, grâce à des stratégies malveillantes qui tentent de contourner les moteurs de recherche. Ces sites sont connus comme des spams Web où les pages non pertinentes ont un rang plus élevé que les pages pertinentes dans les résultats du moteur de recherche. Ils sont généralement considérés comme des résultats insuffisants et inappropriés pour l'utilisateur (Rajendra et *al.*, 2016).

Définissons tout d'abord le spam Web. Un hôte est dit spam si sa popularité est non justifiable (Zhou et *al.*, 2008). Cependant, il est souvent difficile d'étiqueter un hôte Web comme étant spam ou non-spam. Le but est de perturber les utilisateurs

d'Internet et les moteurs de recherche, car il endommage la fiabilité du moteur de recherche et le bénéfice des utilisateurs du Web, et il dégrade la qualité de la recherche d'information sur le Web (Shou-Hong *et al.*, 2014). La lutte contre le spam Web est devenue de plus en plus importante dans les travaux de recherche associés au Web (Zhou et *al.*, 2008).

Selon des études récentes (Renato *et al.*, 2012) et (Kwang et Ashutosh, 2015), indiquent que la quantité de spam Web augmente considérablement : 36% des résultats générés par les moteurs de recherche les plus populaires contiennent des URL malveillantes.

La cybercriminalité est l'une des nouvelles formes de criminalité et de délinquance, dont les conséquences peuvent être particulièrement graves pour la sécurité informatique. C'est un phénomène prenant de l'importance de nos jours (Beebe, 2009). Toutefois, même si les systèmes d'information sont de plus en plus sécurisés, la criminalité informatique augmente, et la demande en analyse forensique augmente d'autant (Duval *et al.*, 2005).

En raison de la croissance rapide de la technologie employée par les spammeurs, et (Becchetti *et al.*, 2006a) des quantités massives de données à analyser (Kwang et Ashutosh, 2015), des systèmes automatiques de détection de spam Web sont nécessaires, car ils sont plus efficaces, génériques et faciles à adapter.

Au sein de ces problématiques, de nombreuses études ont porté sur les différentes méthodes de détection de spam Web. La plupart d'entre elles reposent sur des techniques d'apprentissage machine ou « machine learning ». L'application de ces dernières dans la détection de spam Web peut être considérée comme un problème de classification binaire, où un classificateur est utilisé pour prédire si un hôte Web donné est un spam ou non (Najork, 2009).

En effet, l'apprentissage machine s'intéresse à concevoir des algorithmes, capables à partir d'un nombre suffisant d'exemples, d'en assimiler la nature afin de pouvoir appliquer ce qu'ils ont appris aux cas futurs. Cependant, pour une détection efficace du spam Web, un ensemble de données d'apprentissage fiable est nécessaire. De plus,

l'échelle du Web est énorme, et ne cesse d'augmenter. Il est donc difficile de recueillir et de maintenir un ensemble d'entraînement suffisant pour la détection du spam.
Quelles sont les différentes approches qui ont été proposées dans la littérature pour la détection des spams Web ? Qu'est-ce qui caractérise les hôtes spam ?
Pour parvenir à démontrer l'importance des méthodes d'apprentissage automatique dans le contexte de détection de spam Web, nous avons étudié différentes approches qui ont été proposées dans la littérature. Nous avons aussi exploré diverses caractéristiques à inclure dans l'ensemble d'apprentissage.

0.2 Objectifs et contribution

Dans ce travail de recherche, nous évoquons les applications et les études les plus récentes dans le domaine de l'apprentissage automatique appliqué pour la classification des pages Web.
L'objectif de notre travail consiste premièrement, à étudier et comparer un ensemble de méthodes d'apprentissage automatique que nous avons découvert pendant notre étude de l'état de l'art dans la communauté de la détection de spams Web. Deuxièmement, notre objectif est de préparer les données en faisant l'extraction de nouvelles caractéristiques pour traiter cette problématique. Finalement, nous réaliserons des expérimentations en utilisant différentes caractéristiques du spam Web pour comparer les méthodes d'apprentissage automatique, suivis d'une discussion des résultats obtenus.
Nous tentons donc de faire une évaluation de la performance de plusieurs techniques d'apprentissage machine, utilisées pour détecter automatiquement les hôtes qui diffusent le spam Web. Cependant, il est utile de tester et de comparer ces méthodes en utilisant diverses caractéristiques dans l'ensemble d'apprentissage.
Cette comparaison va nous permettre de connaître les limites de chacune des méthodes, et ainsi, améliorer l'efficacité du système de détection du spam Web.
Essentiellement, voici comment se structure le document :

Le premier chapitre vise à l'introduction des concepts du domaine de l'apprentissage automatique pour la détection du spam Web. Il définit le spam Web, les techniques et les méthodes permettant de les détecter. De plus, il établit la nécessité de la détection des spams sur le Web et il donne une vue générale sur l'apprentissage machine via ses deux sous-domaines principaux, l'apprentissage supervisé et l'apprentissage non supervisé.

Le second chapitre expose un état de l'art sur la détection du spam Web au moyen de diverses méthodes d'apprentissage automatique.

Le troisième chapitre présente et explique notre méthode de détection de spam Web. Elle repose sur l'exploration de différentes caractéristiques. Nous exposons l'architecture et le processus de la préparation de données, en étalant tout le processus qui va de la collecte des données brutes, puis l'extraction de caractéristiques jugées pertinentes à partir du contenu, jusqu'à l'ensemble de données que nous utilisons pour expérimenter les méthodes d'apprentissage machine. Il traite également des propriétés que nous utilisons dans notre méthode pour caractériser le spam Web.

Le quatrième chapitre montre les algorithmes que nous avons choisis, ainsi que les résultats obtenus par chacun d'eux. Nous comparons également nos résultats à ceux consignés dans la littérature. Nous discutons aussi l'importance des attributs les plus pertinents sur la qualité de la prédiction de ces méthodes.

Les expérimentations montrent que les résultats avec nos attributs surpassent les résultats obtenus par d'autres travaux, à la fois en termes de F-mesure que de AUC (Area Under a ROC Curve).

Enfin, nous terminons par une conclusion tout en relevant certaines limites de cette recherche.

CHAPITRE I

DÉFINITION DU DOMAINE : DÉTECTION DU SPAM WEB

1.1 Introduction

Le concept du spam Web a été introduit en 1996 et a été reconnu comme l'un des principaux défis pour l'industrie des moteurs de recherche (Spirin et Han, 2012). Le spam attaque aujourd'hui toutes les sources d'information ouvertes comme les blogs, les wikis, les forums, les sites collaboratifs et les réseaux sociaux. Les enjeux économiques et sociaux sont devenus extrêmement importants pour les différents acteurs du Web et pour les utilisateurs (Ashish *et al.*, 2015).

Il est important que le site offre des contenus pertinents, des installations de navigation et d'autres fonctionnalités qui profitent aux visiteurs. Le problème est que plusieurs sites préfèrent investir dans des techniques de SEO (*Search Engine Optimization*) non éthiques et contourner les moteurs de recherche pour obtenir une pertinence sans mérite. Les plus touchés sont les utilisateurs qui, lorsqu'ils font leurs requêtes dans les moteurs de recherche, reçoivent des réponses inattendues et non pertinentes, souvent infectées par des contenus malveillants. De telles techniques sont connues sous le nom de spam ou spamdexing (Renato *et al.*, 2012), et selon (Kwang et Ashutosh, 2015), elles peuvent être divisées en deux catégories principales: spam de contenu et spam de lien (Renato *et al.*, 2012).

Le spamming Web se réfère à l'utilisation de pratiques contraires à l'éthique des moteurs de recherche pour obtenir un meilleur rang sur la page de résultats. De nombreuses techniques ont tenté d'affecter les résultats de recherche et leur classement avec l'intention de tromper le moteur de recherche pour qu'il renvoie des résultats qui ne sont pas utiles à l'utilisateur (Rajendra et *al.*, 2016). Le spam Web

nuit gravement à la qualité de la recherche d'informations sur le Web ce qui entraîne une diminution de l'efficacité du moteur de recherche et gaspille également beaucoup de temps. Ce phénomène amène un besoin urgent d'identifier les pages spam afin d'utiliser efficacement le moteur de recherche.

La lutte contre le spam Web est devenue de plus en plus importante dans la recherche Web. Un système de récupération d'information robuste et efficace peut être construit si l'on peut identifier et éliminer toutes les pages de spam. C'est la raison pour laquelle des moteurs de recherche efficaces, et qui peuvent produire des résultats de qualité et selon la requête de l'utilisateur sont nécessaires (Rajendra et *al.*, 2016).

Le classement des pages Web est essentiel dans la recherche Web et pour les moteurs de recherche. Toutefois, juger une page Web afin de déterminer si elle est un spam ou non, est une tâche compliquée, car différents moteurs de recherche ont des normes différentes. De plus, les pages spam et non-spam démontrent différentes caractéristiques statistiques (Ntoulas *et al.*, 2006), et enfin, les spammeurs utilisent des techniques différentes pour atteindre leurs objectifs. Sur cette base, plusieurs algorithmes ont été proposés pour classer les pages de spam distinctes des pages normales (Gadhvi et Madhu, 2013).
Les moteurs de recherche utilisent donc des algorithmes sophistiqués pour classer les pages Web afin d'éviter de donner un haut rang aux pages de spam. Le spamming basé sur le contenu, le spamming basé sur les liens et le cloaking sont l'objet principal de différents techniques anti-spam (Muhammad et Malik, 2015). Même si ces techniques anti-spam ont eu beaucoup de succès, elles sont encore confrontées à des problèmes lors de la lutte contre un nouveau type de techniques de spam.

Dans ce qui suit, nous définissons le spam Web et ses enjeux. Nous présentons ensuite la taxonomie des techniques utilisées par les spammeurs ainsi que la

taxonomie de détection de spam. Enfin, nous présentons les méthodes d'apprentissage automatique utilisées pour détecter ces pages spam.

1.2 Qu'est-ce que le Spam Web?

Voyons la définition que donne (Becchetti *et al.*, 2008a) au spam Web ou spamdexing : «Toutes les actions trompeuses qui tentent d'augmenter le classement d'une page dans les moteurs de recherche. Une page ou un hôte de spam est soit utilisé pour le spamming, soit reçoit une quantité importante de son score à partir d'autres pages de spam ». Plus précisément, le spam Web, fait référence à toute action délibérée apportant à des pages Web sélectionnées une pertinence ou une importance injustifiée (Zhou et *al.*, 2008), ce qui est l'un des principaux obstacles à la récupération d'information de qualité sur le Web.

La définition du spamming Web peut aussi être décrite comme l'ajout de contenu immatériel ou des liens vers la page HTML, dans le seul but d'atteindre un rang plus élevé que ce que la page Web mérite en réalité (Ntoulas *et al.*, 2006).

D'une manière générale, les techniques légales sont connues sous le nom de Search Engine Optimization (SEO), alors que l'algorithme de classement trompeur est appelé spam Web (Gadhvi et Madhu, 2013).

1.3 But des spammeurs

Les spammeurs profitent des utilisateurs d'Internet en les attirant sur leurs sites Web à l'aide de diverses techniques de spam intelligentes. Leur but ultime est d'améliorer le classement de leurs pages Web dans les résultats de recherche.

Il y a trois objectifs différents pour diffuser une page de spam (Shekoofeh et Alireza, 2013).

- Le premier est d'attirer les internautes à visiter leurs sites afin d'améliorer le score de la page dans le but d'augmenter les avantages financiers pour les propriétaires du site.
- Le deuxième objectif est d'encourager les gens à visiter leurs sites afin d'introduire leurs entreprises et leurs produits, et de convaincre les visiteurs d'acheter ces derniers.
- Le dernier objectif, et non des moindres, est d'installer des logiciels malveillants sur l'ordinateur de la victime.

1.4 Taxonomie des techniques de spam Web

Il existe différentes manières de construire du spam Web. Parmi les techniques les plus courantes de spam Web, nous avons les méthodes basées sur le contenu, les liens, le cloaking et la redirection.
La combinaison des techniques de spam Web ci-dessous peut également être utilisée pour distraire les utilisateurs et rendre leur détection plus difficile (Victor *et al.*, 2012).

❖ **Spam de Contenu**

Le spam de contenu est le type de spam Web le plus populaire et le plus répandu. Il est très populaire en raison du fait que les moteurs de recherche utilisent des modèles de récupération d'information basés sur le contenu de la page (tels que le modèle d'espace vectoriel) pour les classer. Ainsi, les spammeurs analysent les faiblesses de ces modèles et les exploitent (Spirin et Han, 2012). Il s'agit d'une technique basée sur la modification du contenu ou des mots-clés (Victor *et al.*, 2012). Elle est également connue sous le nom de bourrage de mots clés ou *spamming term* (Gadhvi et Madhu, 2013; Gongwena *et al.*, 2016). Par exemple, la répétition des termes importants à plusieurs reprises sur une page cible ou mettre tous les termes du dictionnaire sur une page cible, sont quelques-unes des techniques par lesquelles les spammeurs tentent

d'augmenter le score d'une page dans les résultats de recherche (Rajendra et *al.*, 2016).

Généralement, il existe cinq sous-types de spam de contenu en fonction de la structure d'une page Web (Rajendra et *al.*, 2016; Spirin et Han, 2012). Ils sont :

- Spam de titre (*Title Spamming*)
- Spam du corps (*Body Spamming*)
- Spam de Méta-tags (*Meta-tags Spamming*)
- Spam de texte d'ancrage (*Anchor Text Spamming*)
- Spam d'URL (*URL Spamming*)

❖ **Spam de Lien**

Le spam de lien est la création de spam Web au moyen de l'ajout de liens artificiels entre les pages dans le but d'augmenter leur popularité (Victor *et al.*, 2012). Dans ce type de spam, les attaquants profitent de la structure de lien des pages Web pour créer des pages de spam. Il existe deux grandes catégories de spam de lien (Spirin et Han, 2012) : les liens sortants et les liens entrants.

- Lien sortant du spam : dans le spam de lien sortant, le spammeur a un accès direct à la page cible et peut donc ajouter de nombreux liens qui pointent vers la page cible pour modifier les résultats de recherche (Rajendra et *al.*, 2016).

- Lien entrant du spam : fait référence à la création d'une page qui pointe vers beaucoup d'autres pages (Gadhvi et Madhu, 2013). Il est également possible de créer des "link farms", qui sont des pages et des sites interconnectés entre eux (Victor *et al.*, 2012). En faisant cela, l'importance de la page cible sera augmentée (Rajendra et *al.*, 2016).

❖ **Cloaking**

Le cloaking est une technique de dissimulation qui est largement utilisée et qui trompe le moteur de recherche et l'utilisateur. Dans cette méthode de dissimulation, les spammeurs essaient de livrer des contenus différents aux robots d'exploration et aux visiteurs normaux afin de tromper les algorithmes de classement des moteurs de recherche (Rajendra et *al.*, 2016). Il s'agit d'un moyen de fournir des versions différentes d'une page pour les utilisateurs en fonction des informations contenues dans la requête. Elle génère dynamiquement un contenu différent pour certains clients (les navigateurs), mais pas pour les autres (les systèmes d'exploration) (Victor *et al.*, 2012). En conséquence, les moteurs de recherche fournissent des informations erronées aux utilisateurs selon la structure qui leur a été représentée par le cloaking (Gadhvi et Madhu, 2013).

❖ **Redirection**

La redirection consiste à générer des clics frauduleux avec l'intention de favoriser le fonctionnement du clic vers les sites cibles. Pour atteindre cet objectif, les spammeurs soumettent des requêtes à un moteur de recherche pour cliquer sur les liens pointant vers leurs pages cibles (Spirin et Han, 2012). Il est généralement utilisé conjointement avec le spam de contenu, en servant une page qui redirige immédiatement le navigateur de l'utilisateur vers une page différente (soit via un script côté client ou le code HTML "méta tag"), ce qui augmente la probabilité que la page spam soit retournée à la suite d'une recherche (Spirin et Han, 2012).

1.5 La nécessité de la détection du spam Web

De nos jours, le concept de détection de spam est considéré comme un sujet de recherche prolifique dans de nombreuses communautés de recherche à travers le monde (Najork, 2009).

Le spam Web a de nombreux effets négatifs sur l'utilisateur final et le moteur de recherche. Les pages de spam ne font pas seulement perdre que de l'espace, mais aussi des ressources importantes, ainsi que du temps. Comme le moteur de recherche doit indexer et stocker un grand nombre de pages Web, il lui faut donc plus d'espace de stockage. De même, lorsque le moteur de recherche doit rechercher des pages Web en fonction d'une requête d'utilisateur, la recherche aura lieu dans un grand corpus et, par conséquent, plus de temps est requis. Cela réduit l'efficacité du moteur de recherche et diminue la confiance de l'utilisateur envers le moteur de recherche (Rajendra et *al.*, 2016).

Grâce à certaines techniques connues sous le nom de spam Web, le classement de certaines pages est stimulé dans les résultats des moteurs de recherche (Shekoofeh et Alireza, 2013). Elles sont apparues dans le but d'obtenir une pertinence injuste des pages Web ou des sites. De nombreuses personnes et organisations utilisent le spam Web pour nuire à des tiers (généralement concurrents) ou pour augmenter le PageRank de leurs pages/sites afin d'obtenir une meilleure position et augmenter leur revenu de la publicité ou de la vente. Finalement, le but est de gagner de l'argent illégalement (Victor *et al.*, 2012).

1.6 Taxonomie des techniques de détection du spam Web

En général, les méthodes de détections de spam peuvent être classées en trois groupes: les méthodes basées sur le contenu, sur les liens et celles combinant les deux approches.

Les méthodes basées sur le contenu : pour cette catégorie d'algorithmes, des propriétés (informations) de contenu des pages Web telles que le contenu, le titre, l'URL, la longueur d'URL, etc. sont utilisées.

Dans le tableau ci-dessous, sont consignées les caractéristiques utilisées par quelques travaux se basant sur le contenu.

Auteur/Année	Informations utilisées
(Fetterly *et al.*, 2004)	La longueur des noms d'hôte, le nombre de tirets, de points, de chiffres dans les noms d'hôtes, le nombre de mots dans chaque page.
(Piskorski *et al.*, 2008)	Fonctionnalités linguistiques en utilisant divers outils de NLP (*Natural Language Processing*).
(Luckner *et al.*, 2014)	Une nouvelle fonctionnalité basée sur le lexical.
(Jacint *et al.*, 2008; Martinez-Romo et Araujo, 2009)	Un modèle de langage statistique.
(Ntoulas *et al.*, 2006)	Le nombre de mots dans la page, le nombre de mots dans le titre, la longueur moyenne des mots, la quantité de texte d'ancrage, la fraction du contenu visible et la compressibilité.
(Gongwena *et al.*, 2016)	Le nombre de mots dans le titre et le ratio de compression des pages Web

Les méthodes basées sur les liens : de nombreuses études se basent sur les liens via le graphique de la structure des liens Web, pour détecter le spam Web. L'algorithme PageRank, basé sur le graphe du lien Web (Gongwena *et al.*, 2016), classe les pages par la valeur de contribution du lien entre les pages Web. Il est l'un des algorithmes de classement le plus utilisé.

Dans le tableau ci-dessous sont répertoriées les caractéristiques utilisées par quelques travaux basés sur les liens.

Auteur/Année	Information utilisée
(Gyongyi *et al.*, 2006)	PageRank : algorithme d'analyse des liens participant au système de classement des pages Web utilisé par le moteur de recherche Google. Il mesure la popularité d'une page Web.
(Becchetti *et al.*, 2006a)	Truncated PageRank algorithme : calcul qui permet de réduire la contribution directe des premiers niveaux de liens dans le classement fournis par l'algorithme PageeRank
(Benczur *et al.*, 2006b)	Mesures de similarité basées sur un lien hypertexte, telles que la co-citation, les plus proches voisins et SimRank.
(Becchetti *et al.*, 2008a)	Le score de TrustRank[2] d'une page. L'intuition derrière TrustRank est une page avec un PageRank élevé.
(Goh *et al.*, 2014)	Des propriétés du poids comme influence d'un nœud Web vers un autre nœud Web.

Les méthodes combinées : Elles utilisent des propriétés de la base de contenu avec la technique basée sur des liens des pages Web afin d'entraîner un classificateur. Cette combinaison peut améliorer les performances du moteur de recherche pour détecter les pages de spam (Rajendra et *al.*, 2016). Cependant, très peu de travaux de recherche ont été réalisés avec des approches combinées, à l'exception des travaux suivants:

Auteur/Année	Informations utilisées
(Benczur *et al.*, 2006a)	Nouvelles caractéristiques basées sur les liens avec les modèles basés sur le modèle de langage (LM).
(Castillo *et al.*, 2007)	Informations de contenu et de topologie.
(Wei *et al.*, 2012)	Données de clic et les URL de quelques pages.
(Gongwena *et al.*, 2016)	Diversité des liens et les caractéristiques de contenu.

[2] Un noeud avec un PageRank élevé

1.7 Apprentissage automatique

La plupart des approches de détection de spam Web reposent sur des techniques d'apprentissage machine ou «machine learning». Dans cette partie, nous essayons de donner une vue générale sur l'apprentissage machine et ses deux déclinaisons : l'apprentissage supervisé et l'apprentissage non supervisé.

L'apprentissage machine est le domaine s'intéressant à comprendre et reproduire la faculté de l'apprentissage humain par des systèmes artificiels. Il s'agit, très schématiquement, de concevoir des algorithmes et des méthodes permettant d'extraire l'information pertinente de données, ou d'apprendre des comportements à partir d'exemples. Ainsi, le but essentiel de l'apprentissage machine est de déterminer la relation entre les objets et leurs catégories pour la prédiction et la découverte des connaissances (Bouguessa, 2015b). Nous distinguons ainsi l'apprentissage supervisé de l'apprentissage non supervisé.

1.7.1 Apprentissage supervisé

L'apprentissage supervisé a pour but d'établir des règles de comportement à partir d'une base de données contenant des exemples de cas déjà étiquetés. La base de données est en principe un ensemble de couples entrées / sorties {(X, Y)}. Le but est d'apprendre à prédire pour toute nouvelle entrée X, la sortie Y (El Fouz, 2013).

La technique de la détection de page Web spam correspond à un problème de classification supervisé (Gadhvi et Madhu, 2013). Dans la classification supervisée, les pages précédemment classées forment un ensemble d'entraînement pour décider si la page est spam ou non.

Il existe plusieurs algorithmes et techniques utilisés pour la classification supervisée des spam Web que nous détaillerons dans le chapitre 4, tel que :

- **Classification Bayésienne.** Il s'agit d'une méthode de classification statistique qui se base principalement sur le théorème de Bayes. Elle est utilisée dans plusieurs applications telles que la détection de courriels spam, pour séparer les bons courriels des mauvais (c'est-à-dire les pourriels).

- **Machine à vecteurs de support (SVM).** Il s'agit d'un ensemble de techniques destinées à résoudre des problèmes de discrimination (prédiction d'appartenance à des groupes prédéfinis) et de régression (analyse de la relation d'une variable par rapport à d'autres).

- **Réseau neuronal.** À l'inverse des algorithmes de déduction, ces derniers sont des algorithmes de type induction, c'est-à-dire que par le biais d'observations limitées, ils essayent de tirer des généralisations plausibles. C'est un système basé sur l'expérience qui se constitue une mémoire lors de sa phase d'apprentissage (qui peut être aussi non supervisée), appelée entrainement.

- **Arbre de décision.** Un arbre de décision est un enchaînement hiérarchique de règles logiques qui permettent de diviser la base d'exemples en sous-groupes, en fonction de la valeur des variables d'entrées. L'arbre est construit en recherchant à chaque niveau le paramètre le plus discriminant pour classifier un exemple (Naffakhi, 2004). Différents algorithmes peuvent être utilisés pour développer l'arborescence. Les arbres de décision suscitent un large engouement en fouille de données, car ils sont simples et rapides tout en restituant de manière compréhensible les relations existantes entre les variables d'entrée et le phénomène à modéliser.

- **Forêt d'arbres décisionnels (Random Forest).** Il s'agit d'une application de graphe en arbres de décision permettant ainsi la modélisation de chaque résultat sur une branche en fonction des choix précédents. On prend ensuite la

meilleure décision en fonction des résultats qui suivront.

1.7.2 Apprentissage non supervisé

Contrairement à l'apprentissage supervisé, le non supervisé traite le cas où on dispose seulement des entrées {X} sans avoir au préalable les sorties.
L'apprentissage non supervisé ou le « clustering » vise à construire des groupes (clusters) d'objets similaires à partir d'un ensemble hétérogène d'objets (Bouguessa, 2015b). Chaque cluster issu de ce processus doit vérifier les deux propriétés suivantes:

- La cohésion interne (les objets appartenant à ce cluster sont les plus similaires possible).
- L'isolation externe (les objets appartenant aux autres clusters sont les plus distincts possible).

En effet, le processus de « clustering » repose sur une mesure précise de la similarité des objets que l'on veut regrouper. Cette mesure est appelée distance ou métrique.
On distingue plusieurs algorithmes de « clustering », comme :

K-moyennes (KMeans). KMeans est un algorithme de partitionnement des données en K nombre de groupes ou clusters. Chaque objet sera associé à un seul cluster. Le nombre K est fixé par l'utilisateur.

Fuzzy KMeans. Il s'agit d'une variante du précédent algorithme proposant qu'un objet ne soit pas associé qu'à un seul groupe.

Espérance-Maximisation (EM). Cet algorithme utilise des probabilités pour décrire qu'un objet appartient à un groupe. Le centre du groupe est ensuite recalculé par rapport à la moyenne des probabilités de chaque objet du groupe.

Regroupement hiérarchique. Deux sous-algorithmes en découlent, à savoir d'une part le «bottom up» qui a pour fonction d'agglomérer des groupes similaires, donc en réduire le nombre (les rendre plus lisibles) et d'en proposer un ordre hiérarchique, et d'autre part, le «top down» qui fait le raisonnement inverse en divisant le premier groupe, récursivement, en sous-ensembles.

1.8 Conclusion

La détection de spam Web est une tâche d'apprentissage automatique difficile. La première génération adopte souvent un modèle d'apprentissage supervisé, où un ensemble d'apprentissage composé de pages étiquetées spam ou non, est utilisé pour former un classificateur. Celui-ci est ensuite utilisé pour classer d'autres pages Web.
Cette tâche doit obligatoirement passer par une étape primordiale qui consiste en le prétraitement des données textuelles. Cette étape est longue et suit des procédures soigneusement organisées pour former des données structurées sous forme de matrice. Cette étape commence par une collecte de données qui peuvent être publiques, ou obtenues suite à un processus de récupération de données, afin de passer d'un ensemble de données textuelles brutes vers un vecteur numérique.
Dans le prochain chapitre, nous présentons l'application des méthodes d'apprentissage machine sur des données Web. Un état de l'art sur la détection de spam Web en utilisant le contenu au moyen de diverses méthodes d'apprentissage automatique sera aussi détaillé.

CHAPITRE II

DÉTECTION DU SPAM WEB AU MOYEN DE DIVERSES MÉTHODES D'APPRENTISSAGE AUTOMATIQUE : UN ÉTAT DE L'ART

2.1 Introduction

La sécurité informatique en tant que discipline a été étudiée dès le début des années 1970 (Beebe, 2009). Selon (Ariu *et al.*, 2011), une des premières pierres angulaires des travaux d'apprentissage machine appliquée à la sécurité informatique, est certainement représenté par le travail proposé par (Denning., 1987), qui a introduit le premier modèle de détection d'intrusions. Depuis lors, une abondance de différentes applications de l'apprentissage machine à la sécurité informatique a été proposée (Altheide et Carvey, 2011). Détecter le spam Web est une tâche d'exploration Web difficile (Zhou et *al.*, 2008). La première génération de méthodes de détection de spam Web adopte souvent un modèle d'apprentissage supervisé. Un ensemble d'apprentissage composé de pages étiquetées spam ou non est utilisé pour former un classificateur. Celui-ci est ensuite utilisé pour classer d'autres pages Web.

Le spamming Web est une technique qui permet d'augmenter le nombre de pages Web spam renvoyées par le moteur de recherche. Récemment, les recherches dans ce domaine ont été très actives (Shekoofeh et Alireza, 2013; Zhou et *al.*, 2008; Renato *et al.*, 2012; Kwang et Ashutosh, 2015; Ashish *et al.*, 2015). Il y a plusieurs façons d'appliquer la détection des spams au moyen des méthodes d'apprentissage automatique. Plusieurs recherches anti-spam se sont concentrées sur l'analyse du contenu pour la classification. D'autres se sont focalisées sur la détection basée sur des liens. Toutefois, la plupart des recherches appliquent les méthodes

d'apprentissage automatique supervisées afin de classer les pages. Les méthodes non supervisées sont généralement utilisées dans quelques travaux tels que (Shyam *et al.*, 2013), exclusivement pour soutenir les méthodes d'apprentissage supervisées, comme dans les travaux de (Castillo *et al.*, 2007).

Au sein de l'approche supervisée, certaines pages Web sont collectées comme données d'apprentissage et étiquetées comme spam ou non-spam par un expert. Ensuite, un modèle de classificateur est construit à partir de ces données. On peut utiliser n'importe quel algorithme d'apprentissage supervisé pour construire ce modèle. Ainsi, le modèle est utilisé pour classer n'importe quelle page Web comme spam ou non-spam. Le problème principal est de déterminer les caractéristiques à utiliser lors de l'apprentissage. Nous nous concentrons sur le rôle qui peut être joué par des techniques d'apprentissage automatique dans la détection de spam dans le Web. La performance de ces méthodes diffère d'un travail à un autre, et plusieurs outils de mesure des performances ont été utilisés.

Dans ce chapitre, nous commençons par présenter les approches utilisées dans la classification des pages Web aux moyens de méthodes d'apprentissage machine. Nous établissons ensuite les techniques de mesure de performances adoptées dans les recherches les plus récentes dans ce domaine. Enfin, nous présentons une synthèse de quelques travaux antérieurs sur la détection de spam Web.

2.2 Métriques d'évaluation

En général, pour évaluer la performance d'une solution, on divise l'échantillon de données déjà classées et disponibles en deux ensembles : l'ensemble d'entraînement sur lequel le classificateur fait son apprentissage et l'ensemble de tests sur lequel on peut évaluer sa performance. L'ensemble de tests contient des pages Web dont on connaît à l'avance les classes auxquelles elles devraient appartenir. On pourra ainsi comparer les décisions prises par le classificateur automatique, aux classes réelles, et calculer ainsi un score de performance (matrice de confusion).

2.2.1 Matrice de confusion

La qualité d'un système de classification est mesurée à l'aide de la matrice de confusion. Les colonnes de cette matrice représentent la répartition des objets dans les classes réelles. Les lignes quant à elles, représentent la répartition des points dans les classes estimées par un algorithme de classification (Bouguessa, 2015a).

Lors de la classification multiclasses de pages Web, c'est-à-dire lorsque $|C| > 2$, une approche commune consistant à « couper » le processus de classification en sous-problèmes est requise. Chaque sous-problème concerne uniquement une classe et l'objectif est alors de juger si la nouvelle page appartient ou non à cette classe par opposition aux autres.

Tableau 2. 1 Matrice de confusion

		Classes réelles	
Classes estimées	$C = \{c_1, c_2, \ldots, c_{\|C\|}\}$	c_i	$\neg c_i$
	c_i	$VP = \sum_{i=1}^{\|C\|} VPi$	$FP = \sum_{i=1}^{\|C\|} FPi$
	$\neg c_i$	$FN = \sum_{i=1}^{\|C\|} FNi$	$TN = \sum_{i=1}^{\|C\|} TPi$

Lors de l'évaluation de tels classificateurs à partir d'un ensemble de tests, quatre nombres sont importants pour chaque classe (voir la table 2.1) :

- le nombre de pages correctement classées comme appartenant à la classe i, noté VPi (pour Vrai Positif).
- le nombre de pages incorrectement classées comme appartenant à la classe i, noté FPi (pour Faux Positif).
- le nombre de pages incorrectement rejetées, noté FNi (pour Faux Négatif).
- le nombre de pages correctement rejetées, noté VNi (pour Vrai Négatif)

Parmi les ratios cités précédemment, les plus importants sont le ratio de faux positifs et le ratio de faux négatifs sur lesquelles s'appuient les mesures de performance. Le

ratio de faux positifs correspond aux pages normales qui ont été classifiées comme du spam par le classificateur, contrairement au ratio de faux négatifs, qui représentent les spams qui ont été classifiés comme des pages normales.

De cette table 2.1, différentes mesures peuvent être calculées pour mesurer les performances des classificateurs, chacune s'intéressant à un aspect de la classification. Dans la suite de la présente section, nous allons présenter les mesures de performance souvent utilisées dans la littérature.

Tableau 2. 2 Mesures de performances utilisées dans divers travaux récents

Auteurs	Approche	Métriques
(Victor *et al.*, 2012)	Une technique (SAAD) basée sur un ensemble d'heuristiques pour traiter tous les types de spam à la fois.	Précision, Rappel et AUC
(Karimpour *et al.*, 2012).	Combiner un algorithme génétique et un algorithme ICA (Imperialist Competitive Algorithm) (Atashpaz et Lucas, 2007) pour sélectionner un sous-ensemble optimal d'attributs.	F-mesure
(Gadhvi et Madhu, 2013)	Évaluer quatre algorithmes de classification différents.	Précision, TP et FP
(Kwang et Ashutosh, 2015)	Comparer plusieurs algorithmes d'apprentissage dans le cadre de la détection de spam Web.	AUC : sous la courbe ROC
(Muhammad et Malik, 2015)	Évaluer quatre algorithmes de classification différents en sélectionnant plusieurs caractéristiques de contenu WebspamUK-2007.	Précision, Rappel et AUC
(Gongwena *et al.*, 2016)	Un nouvel algorithme combinant les caractéristiques de contenu et la diversité des liens des pages Web.	Précision, Rappel et F-mesure
(Rajendra et *al.*, 2016)	Combiner les caractéristiques en se basant sur le contenu et sur les liens.	Précision, Rappel et F-mesure

Les mesures de performance les plus connues dans le contexte de la recherche d'information (précision, rappel, la F-mesure, TP, FP et AUC) seront utilisées pour évaluer la performance de notre approche.

2.2.2 Précision et Rappel

Pour chaque classe c_i, deux mesures sont calculées, la précision notée πi et le rappel noté pi. On définit la précision en apprentissage, comme la probabilité conditionnelle qu'un exemple choisi aléatoirement soit bien classé par le système. Il s'agit du rapport entre le nombre de bonnes prédictions positives et le nombre de prédictions positives. Dans notre contexte, elle mesure le niveau de sécurité, c'est-à-dire le degré avec lequel les pages bloquées sont vraiment des spams. Le rappel mesure la « largeur de l'apprentissage » et correspond au rapport entre le nombre de bonnes prédictions positives et le nombre total d'exemples. Il mesure le pourcentage de pages spam que le filtre a bloqué, à savoir son efficacité. Le seuil de rentabilité est atteint lorsque la précision et le rappel sont égaux (Namburu *et al.*, 2005).

La précision et le rappel sur toutes les classes peuvent être calculés à travers une moyenne des résultats obtenus pour chaque classe.

$$Rappel = \frac{VP}{VP + FN} = \frac{\sum_{i=1}^{|c|} VPi}{\sum_{i=1}^{|c|}(VPi + FNi)} \qquad Precision = \frac{VP}{VP + FP} = \frac{\sum_{i=1}^{|c|} VPi}{\sum_{i=1}^{|c|}(VPi + FPi)}$$

2.2.3 La F-Mesure

La F-mesure est définie comme la moyenne harmonique de la précision et du rappel. Il s'agit d'une mesure qui est un compromis entre la précision et le rappel.

$$F - mesure = \frac{2 \times precision \times rappel}{precision + rappel}$$

Une valeur proche de 1 indique que la classification est de très bonne qualité.

2.2.4 La surface sous la courbe (AUC)

L'AUC est mesurée par la surface sous la courbe ROC (Receiver Operating Characteristic) (Fogarty *et al.*, 2005). La courbe ROC permet de régler le compromis entre les types d'erreurs FP et TP, en mettant en relation dans un graphique, les taux de FP (en abscisse) et les taux de VP (en ordonnée) (Bouguessa, 2015a).

Chaque classificateur produit un point (taux FP, taux TP) dans la courbe.

- Le taux vrai positif (TPR) est le nombre de pages classées correctement de la classe Ci, divisé par le nombre total de pages de la classe Ci (indiqué dans l'équation 1).
- Le taux de faux positifs (FPR) est le nombre des pages mal classées de la classe Ci, divisé par le nombre total de pages de la classe Ci (indiqué dans l'équation 2).

$$TPRi = \frac{VPi}{VPi + FNi} \qquad (1) \qquad\qquad FPRi = \frac{FPi}{FPi + VNi} \qquad (2)$$

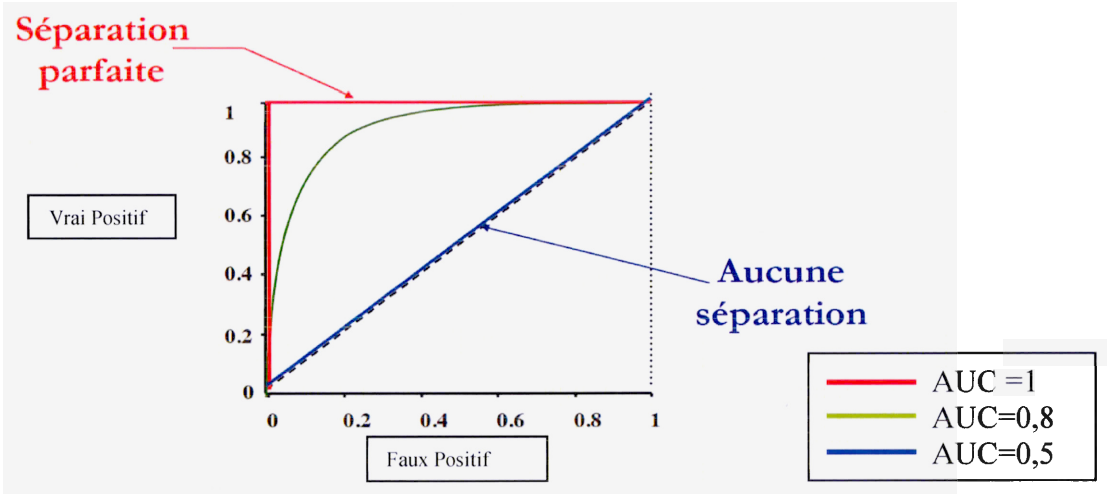


Figure 2.1 Interprétation de la courbe ROC

2.2.5 Taux de succès et taux d'erreur

Le taux de succès (A) et le taux d'erreur (E) sont deux mesures souvent utilisées par la communauté de l'apprentissage automatique. Le taux de succès (traduction de «*accuracy rate*») désigne le pourcentage d'exemples bien classés par le classificateur, tandis que le taux d'erreur («*error rate*») désigne le pourcentage d'exemples mal classés. Les deux taux sont estimés comme suit:

$$A = \frac{\mathrm{VP} + \mathrm{VN}}{\mathrm{VP} + \mathrm{VN} + \mathrm{FP} + \mathrm{FN}} \quad et \quad E = \frac{\mathrm{FP} + \mathrm{FN}}{\mathrm{VP} + \mathrm{VN} + \mathrm{FP} + \mathrm{FN}} = 1 - A$$

Remarque :

Toutes les méthodes de prédiction doivent pouvoir être en mesure de maîtriser ces risques et viser à rendre le ratio de faux positifs petit et de baisser au maximum le ratio de faux négatifs.

Le ratio de faux positifs est le plus critique, car il a un impact plus dangereux sur l'utilisateur que le ratio de faux négatifs, en rejetant une page normale que le classificateur considère comme un spam.

Toutes ces mesures sont utilisées essentiellement pour comparer les différentes approches et les différentes méthodes utilisées dans la classification des pages Web.

2.3 Aperçu sur la détection de spam Web au moyen des méthodes d'apprentissage automatique

Le spam Web a commencé à être pris en compte au milieu des années 1990 et il a pris de l'importance avec l'extension d'Internet. Cependant, l'étude du spam Web au sein de la communauté académique est tout à fait récente. La taxonomie des pages Web spam a été suggérée par (Gyongyi *et al.*, 2004), la plupart des recherches se concentrant sur certains des principaux types de spam Web, comme par exemple, le contenu, le cloaking et le lien.

En général, les auteurs considèrent la détection de spam comme étant un problème de classification binaire, dont les deux classes sont spam et non-spam, et qui met en jeu plusieurs techniques d'apprentissage (Najork, 2009). Un travail d'investigation met en relief que les travaux se concentrent sur le rôle des techniques d'apprentissage machine dans la détection de spam, sur la façon de préparer et de traiter les données, mais aussi, d'extraire les caractéristiques pertinentes. Généralement, les travaux se concentrent sur la proposition de nouvelles techniques d'apprentissage, ou sur l'exploration de nouveaux attributs pertinents. Ainsi, plusieurs travaux de recherche se penchent sur cette dernière tendance.

Nous les avons répartis en trois catégories, en nous basant sur la façon avec laquelle les données sont préparées. La première catégorie met en jeu les méthodes d'apprentissage automatique en se basant sur le contenu des pages. La seconde est basée sur la prise en compte des liens des pages Web. Enfin, la troisième catégorie combine plusieurs caractéristiques comme attributs servant à l'apprentissage.

La première catégorie, qui consiste à utiliser le contenu des pages Web, se caractérise principalement par l'analyse de contenu et des propriétés des pages Web normales et spam. Différentes études ont été menées pour analyser l'importance du contenu de la page Web et des propriétés associées pour détecter le spam Web (Victor *et al.*, 2012; Fetterly *et al.*, 2004; Muhammad et Malik, 2015; Rajendra et *al.*, 2016).

En se référant à ce type d'approches, la plupart des études ont détecté le spam Web en extrayant des caractéristiques des pages Web normales. Par exemple, le travail de (Urvoy *et al.*, 2008) a identifié les pages Web spam en fonction de la comparaison de similarités du style de ces dernières. En 2006 (Ntoulas *et al.*, 2006) présentent un certain nombre de méthodes heuristiques pour détecter le spam en se basant sur le contenu. Ils proposent de nouvelles caractéristiques en fonction des techniques de pondération des mots. Les auteurs présentent des expériences réalisées sur une collection de 105, 484 et 446 pages, collectées par le moteur MSN Search. Une autre contribution de leur travail fut l'utilisation des techniques d'apprentissage

automatique combinées avec leur méthode pour créer un algorithme de détection de spam efficace et précis. Les résultats montrent que certaines méthodes de détection de spam sont plus efficaces que d'autres. Ils les ont donc combinées pour modifier le classificateur C4.5 qui est très précis. Dans (Luckner *et al.*, 2014), les auteurs ont proposé une nouvelle caractéristique basée sur le lexique pour une détection de spam Web stable. Ils ont comparé les classificateurs SVM formés et testés sur les ensembles de données WebspamUK-2006 et 2007. Les résultats expérimentaux montrent que la précision et la spécificité de leur approche sont statistiquement stables. Dans l'étude de (Fetterly *et al.*, 2004), la prévalence du spam a été analysée en fonction de certaines propriétés basées sur le contenu des pages Web. Cette étude révèle que des caractéristiques telles que de longs noms d'hôtes, des noms d'hôtes contenant plusieurs tirets, des points et des chiffres, et le nombre de mots invariables dans chaque page, sont de bons indicateurs des pages Web spam.

En 2016, en analysant les caractéristiques de contenu des pages Web dans l'ensemble de données WebspamUK-2007 (Gongwena *et al.*, 2016) proposent de nouvelles caractéristiques comme le nombre de mots dans le titre et le ratio de compression des pages Web. Un autre travail qui a recours à l'analyse de l'efficacité des algorithmes d'apprentissage les plus utilisés pour la classification supervisée, est celui de (Muhammad et Malik, 2015). Il sélectionne plusieurs caractéristiques basées sur le contenu des pages Web spam, avec l'hypothèse que de telles caractéristiques pourraient être communes aux ensembles de données WebSpamUK-2006 et WebSpamUK-2007. Pour évaluer cette hypothèse, ils ont utilisé des données publiques WebspamUK-2007, et ont comparé leurs résultats à d'autres, avec des classificateurs bien connus. De bons résultats du RIPPER (Jrip) et des arbres de décision (J48) ont été démontrés par rapport aux deux autres méthodes, Naïve Bayes et One-attribute-Rule (OneR).

La deuxième approche qui classe les pages Web en s'appuyant sur les liens concerne les pages spam qui déjouent les algorithmes de classement des moteurs de recherche

en ajoutant des liens supplémentaires ou en induisant d'autres liens pour y accéder. La technique la plus courante est celle des «link farm», qui est une méthode utilisée pour augmenter artificiellement l'importance d'un site ou d'un groupe de sites dans les moteurs de recherche. Le principe est de créer un certain nombre de pages qui pointent vers les pages cibles spéciales. Étant donné que les pages cibles ont une grande quantité de liens entrants, elles peuvent obtenir un classement plus élevé grâce à l'algorithme de tri (Gongwena *et al.*, 2016). L'étude de (Adali *et al.*, 2005) a montré que la technique de spam à base de liens la plus efficace est celle qui fait en sorte que toutes les pages Web spam pointent vers une page cible.

De nombreuses études dans cette approche s'appuient sur le graphe de la structure des liens Web, pour la détection du spam de liens (Becchetti *et al.*, 2006a ; Castillo *et al.*, 2007). Par exemple (Liu et Zhang, 2008) proposent certaines caractéristiques de comportement des utilisateurs, extraites des journaux d'accès du serveur Web d'une page. Ces caractéristiques représentent les comportements des utilisateurs lorsqu'ils atteignent une page (spam ou non-spam). Ces modèles sont utilisés pour séparer les pages de spam des non-spam, indépendamment des techniques de spam utilisées (Karimpour *et al.*, 2012). L'algorithme PageRank est également l'un des algorithmes de classement les plus célèbre, basé sur le graphe des liens Web (Gongwena *et al.*, 2016). En 2007 (Castillo *et al.*, 2007) ont présenté aussi un système de détection de spam qui utilise la topologie du graphe du Web en exploitant les liens entre les pages Web et le contenu des pages elles-mêmes. Ils ont utilisé le résultat d'un algorithme de clustering pour améliorer la prédiction obtenue à partir de l'algorithme de classification. Intuitivement, si la majorité d'un cluster correspond à du spam, alors ils changent la prédiction pour tous les hôtes du cluster comme spam, et vice et versa. Le résultat est un système précis pour la détection de spam Web qui peut être appliqué dans la pratique à des bases de données de grande dimension. Dans (Wei *et al.*, 2012), la structure de liaison du graphe des clics a été utilisée afin de détecter la spamicité qui se propage itérativement entre les requêtes et les URL à travers les clics, pour découvrir d'autres sites qui sont susceptibles d'être du spam.

La troisième approche se caractérise principalement par le fait de combiner plusieurs caractéristiques comme attributs d'apprentissage. Plusieurs travaux de recherche se penchent sur ce type d'approche. En 2006 (Becchetti *et al.*, 2006b) intègrent les attributs de lien et de contenu pour construire un système pour détecter le spam Web. Dans (Erdélyi *et al.*, 2011), les auteurs évoquent la nécessité de procéder à la classification avec un grand espace d'attributs en fonction des progrès récents dans le filtrage du spam Web. Ils montrent que les techniques d'apprentissage, y compris la sélection d'ensembles, LogitBoost et Random Forest, améliorent considérablement la précision. Une nouvelle approche qui s'appelle SAAD (Spam Analyzer and Detector) est aussi proposée par (Victor *et al.*, 2012). Elle consiste à utiliser un ensemble d'heuristiques telles que, la longueur moyenne des mots, des phrases spécifiques communes aux pages spam, le ratio du nombre d'octets de code vs le nombre total d'octets, les mots populaires, etc. Cette approche permet de traiter tous les types de spam à la fois (Cloaking, Redirection Spam, Content Spam et Link Spam). Deux ensembles de données publiques, Email Spam (Web Spam Corpus) et Yahoo!, ont été utilisés pour tester et comparer SAAD avec d'autres études antérieures. Les résultats montrent que l'algorithme C4.5 en appliquant Boosting peut donner de bons résultats avec une amélioration variant de 6% à 10%, par rapport aux résultats de (Ntoulas *et al.*, 2006; Gonzalez et Cristina, 2009), avec une surface sous la courbe ROC (AUC) jusqu'à 0,99% avec l'ensemble de Yahoo. Un autre travail qui a recours à une combinaison des caractéristiques pour détecter les liens favorisés (nepotists links), en utilisant des modèles de langues, est celui de (Benczur *et al.*, 2006a). Il combine plusieurs caractéristiques basées sur les liens avec les modèles basés sur le modèle de langage. Il s'agit d'un système de détection de spam client efficace, où un lien a un faible poids si les modèles linguistiques de la page source et de la page cible ont un grand désaccord. En 2012 (Renato *et al.*, 2012) compare plusieurs techniques d'apprentissage automatique sur le même ensemble de données, WebspamUK-2006. Il utilise trois ensembles de caractéristiques, le premier composé de 96

caractéristiques basées sur le contenu, le second composé de 41 caractéristiques basées sur les liens et le troisième composé de 138 caractéristiques basées sur les liens transformés. Les résultats ont montré une précision des arbres de décision, Perceptron multicouche et Random Forest, supérieure à celle des SVM et KNN. (Karimpour *et al.*, 2012) ont combiné un algorithme génétique (GA) et un algorithme ICA (Imperialist Competitive Algorithm) (Atashpaz et Lucas, 2007), pour trouver un sous-ensemble optimal de caractéristiques à partir de l'ensemble de données WebspamUK-2007. Les expériences montrent que cette combinaison améliore la précision de la classification du spam Web. En outre, la sélection basée sur ICA surpasse la sélection basée sur GA dans la détection de spam Web. Les expériences de (Rajendra et *al.*, 2016) sur l'ensemble de données WebspamUK-2006, et qui se basent sur le contenu, la densité des termes, le rapport POS[3] (Part-Of-Speech Tagging) et les caractéristiques basées sur les liens, ont aussi donné de bons résultats de classification avec un taux de F-mesure de 75,2%. En 2016 (Gongwena *et al.*, 2016) ont proposé un nouvel algorithme de classement de pages Web qui calcule le score de classement des pages Web par la méthode TrustRank, en combinant la diversité des liens et des caractéristiques de contenu. De leur côté, les auteurs (Mahmoudi *et al.*, 2010) appliquent la sélection des caractéristiques pour réduire la dimension des données en utilisant le gain d'information (GI). L'importance d'utiliser différentes catégories de caractéristiques anti-spam Web, pour augmenter la précision de la classification des spam, a aussi été étudiée par (Erdélyi *et al.*, 2011).

En 2017 (Kumar et *al.*, 2017) proposent trois groupes de caractéristiques de spam qualitatif (liens, contenus et cloaking) pour améliorer la précision d'identification du spam. Ils ont utilisé un classificateur d'apprentissage qui combine les trois types de caractéristiques, en obtenant une précision qui fait progresser les résultats de chaque type de spam séparément, et celles réalisées par d'autres recherches. En comparant les résultats obtenus dans (Spirin et Han,

[3] L'étiquetage grammatical

2012) pour l'ensemble de données WebspamUK-2007, dont la F-mesure était de 82% pour la détection de spam de contenu et de 80% pour le spam de lien. En ajoutant leurs nouvelles caractéristiques de contenu et de liens, les auteurs ont obtenu 97% et 98% respectivement.

Nous abordons dans ce qui suit, un aperçu des travaux de détection de spam Web selon l'approche d'apprentissage machine utilisée.

Dans (Niu et Ma, 2010), les auteurs abordent la façon de détecter le spam Web par la programmation génétique. Les résultats sur l'ensemble de données WebspamUK-2006 montrent que cette méthode peut améliorer les performances du rappel de spam de 26%, de 11% en termes de F-mesure et de 4% en termes de précision, par rapport à SVM. Dans (Shyam *et al.*, 2013), les auteurs ont présenté une méthode de clustering pour détecter les pages Web spam. Celle-ci étudie la distance entre l'ensemble des attributs de la page. D'après leur étude, la meilleure combinaison pour les grands ensembles de données est FCM (Fuzzy C-Means) avec KNN (K Nearest Neighbour). De plus, FCM est l'un des meilleurs algorithmes de clustering nécessitant un seul balayage de l'ensemble de données. En 2014, dans (Shou-Hong *et al.*, 2014), une méthode de détection de spam Web inspirée par l'algorithme «Ant Colony Optimization» (ACO) a été présentée. Elle consiste en deux étapes : le prétraitement et la détection. Pour la première étape, le problème de classe déséquilibrée est résolu en utilisant la technique de clustering K-means. Lors de la deuxième étape, le modèle de détection de spam est construit sur la base de l'algorithme d'optimisation de colonie de fourmis. Les résultats expérimentaux obtenus sur l'ensemble WebspamUK-2006 démontrent qu'un tel prétraitement permet la détection de spam plus efficacement, et révèlent aussi que cette approche peut atteindre les mêmes ou de meilleurs résultats avec moins de caractéristiques.

En 2015, dans (Ashish *et al.*, 2015), les auteurs ont proposé une détection de spam basée sur les réseaux de neurones, qui selon eux, ont très peu été sollicités dans des travaux de ce domaine. Dans cette étude, ils évaluent trois algorithmes d'apprentissage supervisé de réseaux de neurones artificiels. Ils ont donc créé un corpus de 368 cas de pages web sélectionnées manuellement, dans lequel environ 30 % des cas ont été marqués comme spam et les autres ont été étiquetés comme non spam. Ils ont ensuite choisi, au hasard, environ 80 % des enregistrements du corpus pour l'apprentissage et 20% pour les tests. Ils ont extrait un total de 31 caractéristiques et les ont classées en 3 catégories : 10 URL, 16 contenus et 5 liens. Ils ont finalement montré que l'algorithme de rétropropagation « Résilient » est plus rapide et fonctionne mieux en termes de précision. De plus, « Gradient conjugué » donne la meilleure sensibilité, alors que l'algorithme de « Levenberg-Marquardt » donne la meilleure spécificité, mais il est plus lent lorsque le temps d'apprentissage est considérable. Un autre travail de recherche qui compare plusieurs algorithmes d'apprentissage au sein de la communauté de détection de spam Web est proposé par (Kwang et Ashutosh, 2015). Leur expérimentation est faite sur deux ensembles de données publics, bien connus, WebspamUK-2006 et WebspamUK-2007. Celle-ci a montré que l'algorithme Random Forest avec des variations de AdaBoost atteint 0,937 AUC avec WebspamUK-2006 et 0.852 AUC avec WebspamUK-2007.

2.4 Synthèse des travaux

Dans le tableau ci-dessous, nous résumons quelques études réalisées ces dix dernières années, en précisant les tâches effectuées, les méthodes utilisées, les caractéristiques retenues et les données exploitées, ainsi que les résultats obtenus

Tableau 2. 3 Synthèse des travaux

Année	Auteurs	Tâches effectuées	Méthodes	Caractéristiques	Données	Résultats/conclusion
2006	(Ntoulas *et al.*, 2006)	Proposition de nouvelles caractéristiques en fonction des techniques de pondération des mots	Arbre de décision	Contenu	Collection de 105, 484 et 446 Pages web, en utilisant MSN search crawler	97% de précision
2010	(Niu et Ma, 2010)	Propose d'apprendre une fonction discriminante par programmation génétique	Algorithme génétique	Liens	WebspamUK-2006	Performant par rapport au SVM
2010	(Jayanthi et Subramani, 2010)	Un algorithme DBSpamClust est proposé pour la détection de spam par lien	Clustering (fuzzy)	Contenu et liens	Données obtenues en utilisant Web Crawler	D'autres pages potentielles de spam pourraient être identifiées à l'aide de DBSpamClust en intégrant le poids de la pertinence du contenu de la page Web
2011	(Erdélyi *et al.*, 2011)	Techniques de classification (la sélection d'ensembles) avec un grand espace d'attributs	Méthodes de sélection d'ensembles, comme LogitBoost et Random Forest	Contenu, liens et liens transformés	WebspamUK-2007 et Discovery Challenge DC2010	Un sous-ensemble d'attributs peu coûteux est supérieur à tous les résultats précédemment publiés.
2012	(Karimpour *et al.*, 2012).	Sélection d'un sous-ensemble optimal des caractéristiques et classification dans un ensemble de données existant	Un algorithme génétique et un compétitif impérialiste / (SVM, réseau bayésien, C4.5)	Liens	WebspamUK-2007	La sélection des caractéristiques par ICA et GA améliore la précision de classification. La sélection par ICA améliore plus par rapport à la sélection par GA
2012	(Renato *et al.*, 2012)	Nouvelles méthodes avec l'ensemble de données existant	Bagging, Random Forest, KNN, Adaboost, C4.5, SVM, LogitBoost, et Perceptron	Contenu et liens	WebspamUK-2006	Les techniques d'agrégation, Bagging et Adaboost, donnent la meilleure performance. Ils sont recommandés pour les prochaines comparaisons

2012	(Victor *et al.*, 2012)	Une technique (SAAD) basée sur un ensemble d'heuristiques pour traiter tous les types de spam à la fois	C4.5	Contenu et liens	Web Spam Corpus WebspamUK-2006, et WebspamUK-2007	C4.5 amélioré en appliquant Boosting. Avec Web Spam Corpus : amélioration de 6% à 10% des résultats présentés dans (Ntoulas et al., 2006). Dans le cas de WebspamUK-2006/2007 AUC jusqu'à 0,99.
2013	(Gadhvi et Madhu, 2013)	Évaluation de quatre algorithmes de classification différents	JRIP, J48, Random Forest, LAD Tree	Contenu, Liens et liens transformés	WebspamUK-2007	Randon Forest plus efficace que d'autres techniques pour les caractéristiques basées sur le contenu et sur les liens. LAD Tree efficace avec liens transformés.
2013	(Keyhanipour et Moshiri, 2013)	Propose un modèle de programmation génétique multicouche. Applique l'analyse du coefficient de corrélation pour la réduire l'espace de caractéristique.	Algorithme génétique	Contenu, liens et liens transformés	WebspamUK-2007	En utilisant cette méthode, on obtient des résultats acceptables, qui sont comparables à ceux d'autres méthodes présentées et qui utilisent toutes les caractéristiques
2015	(Muhammad et Malik, 2015)	Évaluer quatre algorithmes de classification différents, en sélectionnant plusieurs caractéristiques de contenu	Naïve Bayes, OneR, JRip, J48	Contenu	WebspamUK-2007	Les résultats montrent que Jrip et J48 ont de bons résultats par rapport aux autres deux méthodes.
2015	(Kwang et Ashutosh, 2015)	Comparer plusieurs algorithmes d'apprentissage	C4.5, SVM, Random Forest, KNN, Bagging, Adaboost, Naïve Bayes	Contenu et liens	WebspamUK-2006 et WebspamUK-2007	La F-mesure pour le Random Forest avec variation de Adaboost atteint : 0,937 avec WebspamUK-2006 et 0,862 avec WebspamUK-2007

2015	(Ashish *et al.*, 2015).	Évaluation de trois algorithmes d'apprentissage supervisé de réseau de neurones artificiels.	Conjugate Gradient, Resilient Backpropagation learning, Levenberg-Marquardt	Contenu et liens	Sélection de 368 pages manuellement	Resilient Backpropagation learning est plus rapide et plus précis. Conjugate Gradient donne la meilleure sensibilité. Levenberg-Marquardt avec bayésien précis, mais plus lent.
2016	(Rajendra et *al.*, 2016)	Proposer une approche combinant les caractéristiques en se basant sur le contenu et sur les liens	Une nouvelle approche pour identifier les pages de spam.	Contenu et liens	WebspamUK-2006	Une F-mesure très bonne et prometteuse de 75,2% par rapport à d'autres techniques existantes.
2017	(Kumar et *al.*, 2017)	Proposer de nouvelles caractéristiques pour identifier les trois types de spam	un classificateur d'apprentissage qui combine les trois types de caractéristiques	Contenu, liens et cloaking	WebspamUK-2007, ClueWeb-2009 et ECML-PKDD-2011	Avec un total de 20 nouvelles caracteristiques qui ont été introduites et elles ont amélioré le taux de F-mesure de 97% par rapport à d'autres techniques existantes.

2.5 Conclusion

Nous avons proposé dans ce chapitre, un aperçu sur les travaux qui ont été faits pour classifier les pages Web au moyen de méthodes d'apprentissage machine. Notre état de l'art a été établi selon deux points de vue (i) sur les méthodes de classification utilisées, et (ii) sur la façon dont les travaux ont procédé avec l'exploration des attributs pour construire l'ensemble de données, avant la classification. La plupart des travaux ont donné de bons résultats. L'approche qui utilise le contenu des pages est utilisée dans plusieurs travaux, car le spam contenu est la méthode de spam privilégiée par les spammeurs, en raison du fait que la plupart des moteurs de recherche appliquent les modèles de récupération d'information basés sur le contenu de la page pour classer les pages Web, comme le modèle d'espace vectoriel (Salton *et al.*, 1975), ou des modèles statistiques de langage (ChengXiang, 2008). Par conséquent, les spammeurs analysent les faiblesses de ces modèles et les exploitent (Muhammad et Malik, 2015). Nous nous concentrons dans le prochain chapitre sur le prétraitement des données. Nous présentons ainsi notre méthode de détection de spam qui repose sur l'exploration de différentes caractéristiques à partir du contenu, afin de préparer l'ensemble de données que nous utiliserons pour expérimenter des méthodes d'apprentissage automatique dans le chapitre 4.

CHAPITRE III

LE PROCESSUS DE PRÉPARATION DES DONNÉES

3.1 Introduction

En raison des points que nous avons abordés dans les chapitres précédents, nous devrions créer des mécanismes de protection pour les utilisateurs qui perdent leur temps et leur argent, les propriétaires des pages qui souhaitent obtenir des clients par des moyens et des méthodes éthiques et pour les entreprises qui fournissent des moteurs de recherche. Ces derniers sont très touchés puisqu'ils ne perdent pas seulement en réputation lorsqu'ils montrent des pages Web spam parmi leurs résultats, mais ils gaspillent de l'argent et des ressources pour analyser, indexer et afficher les résultats des pages qui ne devraient pas être affichés. Ne pas protéger certaines de ces entités signifie une perte économique (Victor *et al.*, 2012).

Pour affecter le classement des moteurs de recherche, le spamming de contenu est censé être la première technique de spam Web utilisée (Muhammad et Malik, 2015). C'est la méthode recommandée par les spammeurs en raison du fait que la plupart des moteurs de recherche appliquent les modèles de récupération d'information basés sur le contenu de la page pour classer les pages Web, comme le modèle d'espace vectoriel (Salton *et al.*, 1975), ou des modèles statistiques de langage (ChengXiang, 2008). Par conséquent, les spammeurs analysent les faiblesses de ces modèles et les exploitent. Par exemple, les spammeurs trompent les moteurs de recherche en falsifiant le score fréquence (TFIDF) dans leurs sites Web (Muhammad et Malik, 2015).

Comme nous l'avons souligné dans le chapitre 2, la plupart des travaux ont donné de bons résultats. Mais nous avons vu que leur performance dépend de plusieurs paramètres. Cependant, il est utile de comparer les méthodes d'apprentissage sur diverses caractéristiques d'ensembles d'apprentissage et sur des problèmes variés.
Le but de notre travail est de proposer une méthode de détection de spam qui repose sur l'exploration de différentes caractéristiques à partir du contenu pour, étudier et analyser l'impact de certains attributs, et celui des méthodes d'apprentissage sur la performance de classification dans le contexte de détection de spam Web. Nous voulons aussi comparer notre approche de détection de spam avec celles proposées dans des travaux précédents.
Cette étude va nous permettre de connaitre ce qui caractérise les hôtes spam et elle va nous permettre également de proposer une combinaison entre les caractéristiques les plus pertinentes que nous avons extraites et celles proposées dans (Castillo *et al.*, 2006) pour améliorer l'efficacité du système de détection de spam Web.

Dans ce chapitre, nous décrivons les données de WebspamUK-2007 et présentons la démarche poursuivie. Nous commençons par une présentation générale du processus, en détaillant les différentes étapes par des figures illustratives. Ensuite, nous présentons chaque module en allant de la collecte des pages Web jusqu'à l'extraction des caractéristiques. Puis, nous mettons en avant la base de données que nous avons conçue, en décrivant chacun de ses attributs. Enfin, nous décrivons le processus d'évaluation de la méthode proposée, qui débute par la sélection de caractéristiques, se poursuit par la classification et enfin se termine par la comparaison des résultats.

3.2 Description des données WebspamUK-2007

L'une des tâches initiales est de trouver une collection de référence pour tester la détection de spam Web. Dans ce contexte (Castillo *et al.*, 2006) ont mis à disposition le premier corpus spécifiquement conçu pour la recherche sur le spam Web (WebspamUK-2006). Plus tard, une version mise à jour et améliorée a été étiquetée par un groupe de bénévoles, en créant la version WebspamUK-2007 du corpus, qui est elle aussi publiquement disponible.

Cet ensemble de données est composé de quatre sous-ensembles d'attributs différents : "caractéristiques basées sur le contenu", "caractéristiques basées sur les liens", "caractéristiques basées sur les liens transformés" et "caractéristiques générales". Parmi ces quatre ensembles, nous avons pris en considération seulement les caractéristiques basées sur le contenu. Ce type de caractéristiques comprend la taille des mots, la longueur des titres, le nombre de mots dans la page, etc.

Nous présentons dans ce qui suit, l'ensemble des caractéristiques basées sur le contenu[4] de WebspamUK-2007:

Il comprend 98 attributs dont:

- l'attribut Hostid, l'identifiant de l'hôte.
- l'attribut Hostname, le nom de l'hôte.

Les 96 attributs restants sont partagés selon 4 catégories de 24 attributs chacune. Pour chaque catégorie, les mêmes attributs sont calculés et sont définis comme suit :

- de HST1- HST24 concernent la page d'index,
- de HMG25 - HMG48 concernent la page avec le max PageRank,
- de AVG49 - AVG72 concernent la valeur moyenne pour les pages dans l'hôte
- de STD73 - STD96 concernent l'écart type pour les pages dans l'hôte.

[4] http://chato.cl/webspam/datasets/uk2007/features/uk-2007-05.content_based_features.txt

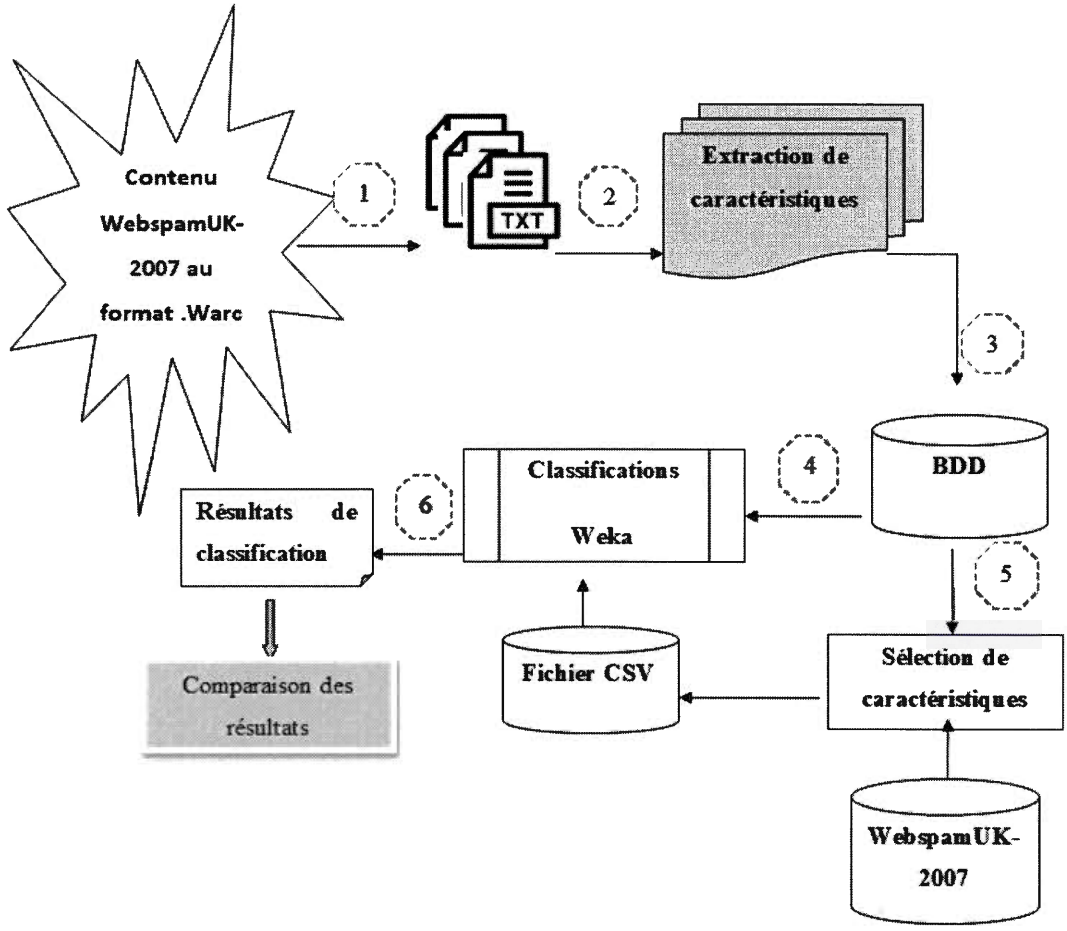
Tableau 3. 1 Description des caractéristiques du contenu WebspamUK-2007

Attributs				Définition
Page d'index	Max PageRank	La valeur moyenne des pages dans l'hôte	L'écart type des pages dans l'hôte	
HST-1	HMG25	AVG49	STD73	nombre de mots
HST-2	HMG26	AVG50	STD74	nombre de mots dans le titre
HST-3	HMG27	AVG51	STD75	longueur moyenne des mots
HST-4	HMG28	AVG52	STD76	fraction du texte d'encre
HST-5	HMG29	AVG53	STD77	fraction du texte visible
HST-6	HMG30	AVG54	STD78	Taux de compression
HST-7	HMG31	AVG55	STD79	top 100 de précision corpus
HST-8	HMG32	AVG56	STD80	top 200 de précision corpus
HST-9	HMG33	AVG57	STD81	top 500 de précision corpus
HST-10	HMG34	AVG58	STD82	top 1000 de précision corpus
HST-11	HMG35	AVG59	STD83	top 100 de rappel corpus
HST-12	HMG36	AVG60	STD84	top 200 de rappel corpus
HST-13	HMG37	AVG61	STD85	top 500 de rappel corpus
HST-14	HMG38	AVG62	STD86	top 1000 de rappel corpus
HST-15	HMG39	AVG63	STD87	top 100 de précision requête
HST-16	HMG40	AVG64	STD88	top 200 de précision requête
HST-17	HMG41	AVG65	STD89	top 500 de précision requête
HST-18	HMG42	AVG66	STD90	top 1000 de précision requête
HST-19	HMG43	AVG67	STD91	top 100 de rappel requête
HST-20	HMG44	AVG68	STD92	top 200 de rappel requête
HST-21	HMG45	AVG69	STD93	top 500 de rappel requête
HST-22	HMG46	AVG70	STD94	top 1000 de rappel requête
HST-23	HMG47	AVG71	STD95	l'entropie
HST-24	HMG48	AVG72	STD96	l'indépendance LH

Nous avons présenté dans cette section l'ensemble de caractéristiques basées sur le contenu, proposées dans WebspamUK-2007. Dans la prochaine section, nous allons voir l'ensemble de caractéristiques que nous avons proposé pour construire notre version WebspamUQAM-2017 à partir du contenu HTML des pages de WebspamUK-2007.

3.3 Organisation de la démarche de construction de WebspamUQAM-2017

Le schéma suivant illustre les domaines qui interviennent durant le processus :



1 : Collecter et décrypter le contenu de WebspamUK-2007 au format .txt
2 : Extraire des caractéristiques à partir du contenu
3 : Stocker dans la base de données
4 : Classifier selon plusieurs méthodes d'apprentissage
5 : Sélectionner les attributs pertinents
6 : Comparer les résultats obtenus

Figure 3.1 Processus de la démarche

Dans notre cas, nous exploitons comme corpus de données un ensemble de pages Web afin de les catégoriser en pages spam et pages normales. L'apprentissage automatique prend part au processus lors de l'utilisation des techniques et des méthodes de «Data Mining» sur l'ensemble des pages. Notre approche prépare, en premier lieu, les données entrées en input pour former un contexte formel, c'est-à-dire, la matrice documents-mots (hôtes-caractéristiques), sur laquelle nous pouvons appliquer les techniques d'apprentissage automatique.

3.4 La collecte des données (pages Web)

La nécessité d'une collection de référence capable de garantir la reproductibilité des résultats et d'assurer une comparaison correcte des approches novatrices a été mise en évidence par l'expérience acquise au fil des années par la communauté de spam Web.
Pour mener à bien nos expériences, l'accès au contenu HTML de l'ensemble de données WebspamUK-2007[5] crawler par le Laboratoire de Web Algorithmics[6] de l'université de Milan, nous a été fourni par l'auteur. Le choix de se tourner vers cet ensemble a été motivé par le fait qu'il soit complet, c'est-à-dire qu'il contient des pages HTML brut conservant le format d'origine.
Ces données ont plusieurs caractéristiques :
- Large: la collection devrait inclure de nombreux exemples de spam et des contenus non-spam.
- Propre: la collection devrait contenir peu ou pas d'erreurs de classement.
- Uniforme: la collection devrait représenter un échantillon aléatoire uniforme sur un ensemble de données.

[5] http://chato.cl/webspam/
[6] http://law.di.unimi.it/. URL retrieved 02/2017

- Vaste: la collection devrait inclure autant de techniques de spam Web différentes que possible.
- Ouvert: la collection devrait être disponible gratuitement pour les chercheurs.

La compilation du jeu de données WebspamUK-2007 a été étiquetée au niveau de l'hôte par un groupe de personnes qui travaillent sur le domaine de la détection de spam. Ces hôtes sont marqués comme «spam», «non spam» et «inconnu» par l'évaluateur.

Les pages HTML collectées dans ce corpus sont stockées au format WARC, de sorte qu'un domaine donné est composé de plusieurs fichiers WARC, partagés en 8 fichiers (LAW0-LAW7) de 6 GO chacun. En outre, ce corpus a largement été utilisé dans des études antérieures (Abernethy *et al.*, 2008; Araujo et Martínez-Romo, 2010; Castillo *et al.*, 2006; Erdélyi *et al.*, 2011; Han et Levenberg, 2012). L'ensemble de données WebspamUK-2007 contient jusqu'à 114 529 hôtes, dont 6 475 étiquetés à l'aide de trois catégories différentes (voir tableau 3.3). De plus, l'ensemble est limité à un large échantillon de 400 pages par hôte. Les fichiers WARC GZIP d'origine ont été compressés par blocs.

Une description de ce corpus a été présentée dans le Web Spam Challenge 2008, par Castillo, Chellapilla et Denoyer (2008) (voir tableau 3.2).

Figure 3.2 Description des 8 fichiers WARC

Tableau 3. 2 Description de WebspamUK-2007

	Version complète	Version sommaire
Contenu	Toutes les pages: 105 Mg pages	Jusqu'à 400 pages par hôte: 12 Mg pages
Taille	560 Go compressés	46 Go compressés 200 Go non compressés
Format physique	8 fichiers de ~ 70 Go chacun non disponibles	8 fichiers de ~ 6 Go chacun disponibles en ligne (*mot de passe requis*)

Tableau 3. 3 Répartition de l'ensemble de données WebspamUK-2007

Classe	Nombre d'instances
Non spam	5706 (88%)
Spam	344 (5.3%)
Inconnu	425 (6.5%)
Total	6475

3.5 Décrypter le contenu de WebspamUK-2007 du format .WARC au format .TXT

Les contenus de WebspamUK-2007 sont fournis au format WARC, une norme proposée par Internet Archive. Afin de décrypter le fichier (.WARC), la bibliothèque LAW library 1.3+ comprend des méthodes pour lire ces enregistrements à partir de Java. Cette bibliothèque est un logiciel gratuit, pouvant être modifiée aux termes de la Licence Publique Générale de GNU[7] publiée par le Free Software Fondation.

[7] Https://www.gnu.org/copyleft/gpl.html

Pour cette étape, nous avons exploité certains algorithmes de LAW pour lire les 8 fichiers (.WARC) contenant les pages Web des hôtes, mais nous avons également dû ajouter du code source à LAW pour répondre à nos besoins.

Comme nous l'avons souligné au début, l'ensemble de données WebspamUK-2007 contient jusqu'à 114 529 hôtes, dont 6 479 sont étiquetés, c'est-à-dire dont on connait la classe. Afin d'optimiser le programme, nous avons décidé de récupérer le contenu des pages Web des hôtes dont la classe est connue. Dans ce qui suit, nous décrivons les étapes que nous avons suivies.

Étape 1 : Récupérer dans une table la classe de chaque nom d'hôte

Nous avons tout d'abord, récupéré la classe de chaque hôte à travers le fichier *WebSpam-UK2007-hostnames*, qui contient les hostid et les noms d'hôte.

Hostid : un numéro unique de 0 à 114 528 identifiant chaque hôte dans la collection des noms d'hôte. Dans les deux fichiers, *WebSpam-UK2007-set1-labels* et *WebSpam-UK2007-set2-labels* disponibles dans (Castillo C. , Webspam-UK2007), le premier contient 4275 hôtes, et représente l'ensemble d'apprentissage, alors que le second dispose de 2204 hôtes, et constitue l'ensemble de test. Nous avons créé une base de données (Bdd_labels) et avons stocké le fichier *WebSpam-UK2007-hostnames* dans la table « *Hosts* », et les fichiers *WebSpam-UK2007-set1-labels* et *WebSpam-UK2007-set2-labels* dans la table « *labels_data* ». Nous avons ensuite effectué une jointure afin d'obtenir l'identifiant, le nom et la classe de chaque hôte, tel qu'illustré ci-dessous :

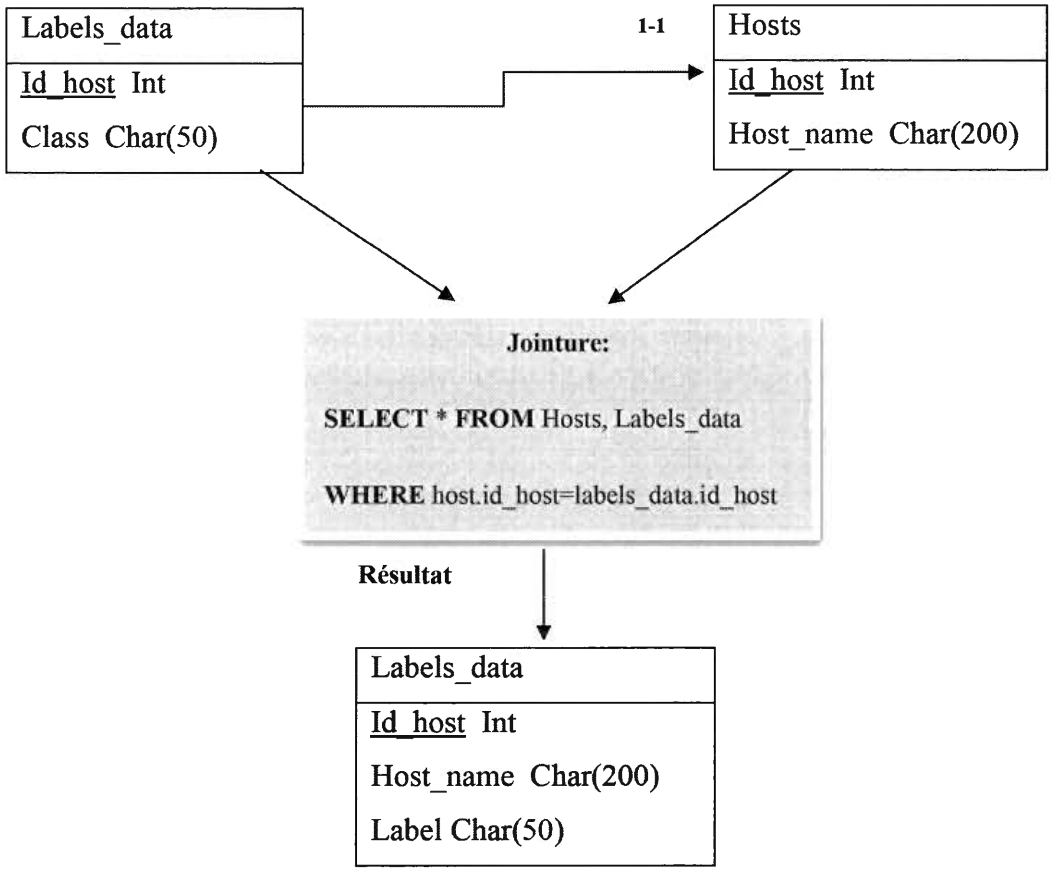


Figure 3.3 Schéma de la jointure

Étape 2 : Récupérer le contenu des hôtes dont la classe est connue

Pour récupérer le contenu des hôtes, nous avons conçu un algorithme qui parcourt chaque LAW_i pour lire les pages Web, et stocke toutes celles qui appartiennent au même hôte en créant un fichier portant le nom de celui-ci. L'algorithme est résumé dans le pseudo-code suivant.

- *Entrées : fichier law_i, Labels_data*
- *procédure pour décompresser le format Warc : GZWarcRecord*
- *Sortie : un dossier **data** qui contient les hôtes. Chaque hôte correspond à un fichier nommé avec son identifiant, contenant toutes les pages qui lui appartiennent.*

(1) Appliquer la procédure GZWarcRecord pour décrypter law_i

(2) Pour chaque page Web lue dans law_i

Si la page est vide alors afficher -1

Sinon

(3) Récupérer le nom de l'hôte

(4) Vérifier si l'hôte existe dans le labels_data

Si l'hôte existe dans le Labels_data

Vérifier dans le dossier data

Si le fichier au nom de cette hôte existe alors

Ajouter le contenu de cette page dans ce fichier

Sinon

Créer un nouveau fichier portant le nom de cette hôte

Fin si

Fin si

Fin si

Fin pour

Figure 3.4 Algorithme : Récupération du contenu des hôtes dont la classe est connue

Résultat de l'étape 2

Le dossier data contient 6475 fichiers des hôtes, en format texte.

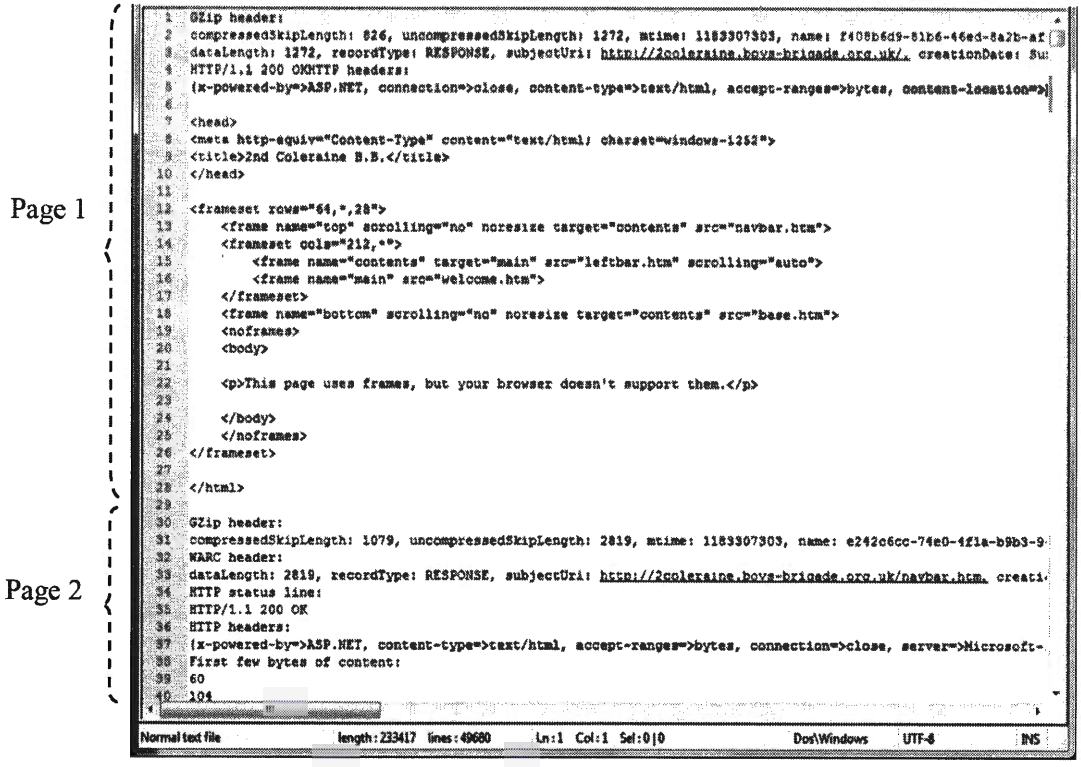


Figure 3.5 Exemple du contenu d'un hôte

Une autre considération importante est l'existence de domaines contenant des pages avec un contenu vide ou sans signification (par exemple, comme des redirections vers des pages d'erreur). Par conséquent, il est obligatoire de prétraiter tout le corpus afin d'éliminer ces domaines. Le tableau 3.4 montre la répartition finale de chaque instance utilisée dans l'étude.

Tableau 3. 4 Répartition finale des instances utilisées dans l'étude

	Corpus Existant	Corpus résultant
Non spam	5706 (88%)	4726 (86%)
Spam	344 (5.3%)	344 (6.3%)
Inconnue	425 (6.5%)	425 (7.7%)
Nombre d'instances	6475	5495

Comme on peut le constater à partir du tableau 3.4, le corpus résultant est déséquilibré, contenant 5495 instances réparties asymétriquement (c'est-à-dire, 344 spam, 425 inconnues et 4726 non spam).

3.6 Extraction de caractéristiques à partir du contenu

Comme les pages de spam Web visent à améliorer le score de classement des moteurs de recherche, la majeure partie du contenu Web est l'accumulation d'une grande quantité de mots-clés et d'un contenu non pertinent ajouté de manière malveillante.

Pour pouvoir exploiter un ensemble de données avec de l'apprentissage machine, nous avons besoin de le prétraiter et de l'organiser sous une forme bien structurée. Nous avons donc analysé les caractéristiques de contenus des pages Web dans le jeu de données WebspamUK-2007 fourni par le moteur de recherche Yahoo. Après avoir analysé le contenu des pages d'index de spam et des pages d'index normales, plusieurs caractéristiques de contenu nous ont semblé intéressantes à extraire pour améliorer les résultats de classification pour WebspamUK-2007. Ces caractéristiques sont classées en quatre catégories : titre, en-tête, mots clés et corps.

3.6.1 Caractéristiques de titre

- **Le nombre de caractères dans le titre et cohérence du titre**

Les moteurs de recherche d'aujourd'hui donnent généralement un poids plus élevé aux termes qui apparaissent dans le titre d'un document. Par conséquent, il est logique que les spammeurs essayent d'inclure les termes du spam dans le titre du document.

Le nombre de caractères dans le titre

Lors de l'utilisation des moteurs de recherche, nous entrons généralement des mots clés pour trouver ce dont nous avons besoin, de sorte que dans de nombreuses pages

spam, une grande quantité de mots-clés qui ne sont pas liés aux contenus Web se forment comme un titre de page.

Cohérence du titre

Le titre devrait être en adéquation avec le contenu global de la page qu'il désigne. Les spammeurs essayent généralement de nommer le titre d'une page avec un titre générique pouvant être très recherché, mais sans lien avec le contenu de la page. Pour cette raison à travers la cohérence du titre, on calcule le taux du nombre de mots clés utilisés dans le titre de la page et n'apparaissant pas dans les mots clés de la page, en utilisant le ratio suivant :

$$\textbf{\textit{Cohérence du titre}} = \frac{Nbr\ de\ mots\ du\ titre\ qui\ apparaissent\ dans\ le\ contenu}{Nbr\ de\ mots\ dans\ le\ contenu}$$

1. *Récupérer le titre entre les balises <title> </title>*
2. *Si le titre existe {*

   ***Alors** retirer les ponctuations*

   - Tokenization

   - Supprimer ponctuation

   - Stocker les mots dans une liste result1

   - Récupérer tout le contenu de la page sans les balises html

   ***Pour** chaque mot du titre {*

   Chercher l'existence du mot dans le contenu {

   Si le mot se trouve dans le contenu

   Alors compteurMot++;

   Fin si

   Fin pour

   *Cohérence titre= compteurMot*100/result1.size()*

   Enregistrer dans la Bdd

   Sinon

Cohérence titre= null

Fin si

Figure 3.6 Algorithme de calcul de la cohérence du titre

3.6.2 Caractéristiques de mots clés

Pour les prochaines caractéristiques, nous nous sommes basés sur l'information contenue dans les zones suivantes :

- **Titre :** Le titre a une grande importance descriptive sur une page HTML. D'ailleurs, Google tient compte de cet élément pour générer son classement.
- **Méta Description :** Il s'agit d'une description du contenu de la page. Elle sert souvent de descriptif lorsque la page en question apparait sur les résultats d'un moteur de recherche comme Google. Les mots-clés recherchés apparaissent en gras dans ce descriptif, ce qui peut constituer un avantage. Le contenu de la description est une cible pour les spammeurs.
- **Méta Keyword :** Les méta keywords sont une liste de mots-clés (séparés par une virgule) ayant un rapport direct avec la page.
- **L'attribut Alt de la balise image :** En HTML, la balise img implique l'usage de l'attribut alt. *().* Le texte de l'attribut alt des balises images permet notamment, l'indexation des graphiques dans les pages de résultats dédiées aux images. De plus, Google associera l'image en question aux mots-clés présents dans l'attribut alt. Il est donc intéressant pour les spammeurs de choisir intelligemment la valeur de l'attribut alt.
- **Body :** il s'agit du corps de la page

La raison pour laquelle nous avons choisi les informations situées dans la zone HEAD (les balises META) en plus du BODY, est que cette zone est la cible

privilégiée des spammeur. En effet l'information contenue dans cette zone n'est pas vue par les utilisateurs des navigateurs. Les informations dans les balises META sont principalement utilisées pour communiquer avec les navigateurs Web et les moteurs de recherche : la déclaration de mots clés ou la fourniture d'une description de contenu.

- **Nombre de mots clés**

Comme les pages spam visent à améliorer le score de classement des moteurs de recherche, la majeure partie du contenu Web est composé de l'accumulation d'une grande quantité de mots-clés et d'un contenu non pertinent ajouté de manière malveillante. Nous avons calculé le nombre de mots clés sans compter les mots vides et les doublons. Le processus est le suivant :

1. Extraire le contenu des zones citées précédemment en utilisant *Jsoup* qui est une bibliothèque de méthodes Java open source, conçues pour extraire et manipuler des données stockées dans des documents HTML.
2. Tokenisation : StringTokenizer
3. Supprimer les mots vides
   stopWords[]={"a", "add", " ", "all", "also", "an", "and", "are", "as", "at", "be", "been", "but", "for", "had", "has", "have", "he", "her", "him", "his", "in", "is", "it", "its", "of", "on", "other", "our", "she", "than", "that", "the", "their", "this", "to", "u", "was", "we", "where", "what", "who", "with"};
4. Supprimer la ponctuation
5. Stemming (EnglishStemmer) : qui consiste à regrouper les mots ayant la même racine. La racine d'un mot correspond à la partie du mot restante une fois que l'on a supprimé son préfixe et suffixe.
6. Supprimer les doublons.

De façon très basique, il s'agit de compter le nombre d'occurrences d'un mot dans une page, puis de faire un rapport avec le nombre total de mots trouvés dans cette même page.

- Mot unique

Le nombre de mots dont le nombre d'occurrences est égal à un, c'est-à-dire le nombre de mots clés qui ne se répètent pas.

- Taux de mots uniques

Les spammeurs ont tendance à répéter plusieurs fois un mot clé pour favoriser sa prise en compte par les moteurs de recherche.
Le taux de mots uniques consiste à mesurer le taux des mots sans répétitions.

$$Taux\ de\ mots\ uniques = \frac{nombre\ de\ mots\ uniques}{nombre\ de\ mots\ clés}$$

Par exemple, si la page contient 299 mots clés uniques sur 501 mots clés, dans ce cas-là, le taux de mots uniques sera de 59%.
Ainsi, nous pouvons facilement déduire le taux de répétition des mots clés à partir du taux mots, en faisant *(1- taux de mots unique)*.

- Densité de mot clé

L'indice de densité de mot clé (ID) correspond au nombre d'occurrences du mot clé (nombre de répétitions du mot) ramené au nombre total de mots de la page. Le calcul de densité de mot-clé aide à déterminer si le mot est présent de nombreuses fois dans la page.
Pendant longtemps en référencement, l'élément de la densité avait un poids comme critère SEO (*Search Engine Optimization*). Plus un mot était répété dans une même page, mieux c'était pour son référencement. C'est ainsi que les pages obtiennent de

bons résultats lors d'une recherche Web. En sachant qu'une page Web est considérée comme du spam si elle a été créée dans le but d'augmenter son classement, grâce à l'utilisation de contenu qui n'ajoute aucune valeur à la page Web, il faut donc examiner l'intention de l'auteur.

$$\textit{Indice de densité de mot-clé (ID)} = \frac{\textit{nombre d'occurrences du mot}}{\textit{nombre total des mots dans la page}}$$

Par exemple, si une page contient 300 mots avec un mot-clé qui apparait 3 fois, alors l'ID de ce mot clé est de 1%.

Dans notre cas, nous cherchons à récupérer cette information de densité globale pour toute la page. Nous procédons en deux étapes. La première consiste à calculer la densité pour chaque mot de la page, et la seconde, est de calculer la densité globale sous quatre mesures différentes :

- la densité maximale, qui renvoie la fréquence maximale ;
- le mode est la valeur observée qui a la fréquence la plus élevée. Dans notre cas, cela correspond à la densité des mots qui reviennent le plus souvent dans la page ;
- la densité médiane est le point milieu de l'ensemble qu'elle divise en deux moitiés. Il s'agit d'une valeur *m* qui permet de découper les valeurs en deux parties égales, mettant d'un côté une moitié des valeurs qui sont toutes inférieures ou égales à *m*, et de l'autre côté, l'autre moitié des valeurs qui sont toutes supérieures ou égales a *m*;
- l'écart-type de la densité est défini comme étant une mesure de dispersion d'un ensemble de valeurs autour de leur moyenne. Plus l'écart-type est faible, plus les valeurs sont homogènes.

♦ **Similarité cosinus**

Nous avons décidé de calculer le degré de similarité entre les mots de la page et les mots qui se trouvent dans les balises méta, pour voir les points de ressemblance et de dissemblance qu'il y a entre ces derniers. Plus précisément nous voulons voir si les

mots les plus fréquents dans la page, visible pour l'utilisateur, et les mots dans les zones méta, visibles seulement par le navigateur, sont les mêmes.

La similarité cosinus est fréquemment utilisée en tant que mesure de ressemblance entre deux documents. Il pourrait s'agir de comparer les textes issus d'un corpus dans une optique de classification, ou de recherche d'information. Dans ce cas, un document vectorisé est constitué par les mots de la requête et est comparé par mesure de cosinus de l'angle avec des vecteurs correspondant à tous les documents présents dans le corpus. On évalue ainsi lesquels sont les plus proches.
La mesure d'angle entre deux vecteurs ne pouvant être réalisés qu'avec des valeurs numériques, il faut imaginer un moyen de convertir les mots d'un document en nombres. On partira d'un index correspondant aux mots présents dans les documents puis on attribuera à ces mots des valeurs. En règle générale, pour mesurer finement la similarité entre des séquences de texte, les vecteurs sont construits d'après un calcul de type *TF-IDF* (term frequency–inverse document frequency), qui permet d'estimer l'importance d'un mot par rapport au document qui le contient, en tenant compte aussi du poids de ce mot dans le corpus complet[8].

En pratique, l'utilisation du *TF-IDF* donnera souvent le meilleur résultat pour le calcul de similarité. Le calcul effectif de la distance dans ce cas est appelé la «similarité cosinus» et a été largement utilisée pour la recherche d'information (Weiss *et al.*, 2005).
Le calcul de la similarité-cosinus se fait de la manière suivante, sachant que le poids du mot dans le document *Wj* est calculé par la formule *TF-IDF*, où *j* est le *jème* mot dans le dictionnaire, *TFj* est sa fréquence dans le document, *N* est le nombre de documents dans la collection d'apprentissage et enfin, *DFj* est le nombre des documents dans lesquels le mot apparait :

[8] http://fr.wikipedia.org/wiki/TF-IDF

Soit A et B deux documents.

$$W_j = TF_j * log_2 (N/DF_j)$$

$$Similarity\ cos = cos(\theta) = \frac{A \times B}{\|A\| \times \|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n}(A_i)^2} \times \sqrt{\sum_{i=1}^{n}(B_i)^2}}$$

Le cosinus de l'angle entre deux vecteurs détermine ainsi si deux vecteurs pointent à peu près dans la même direction.

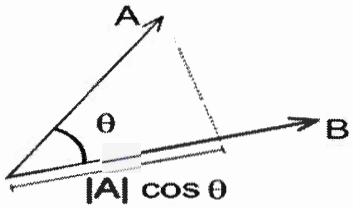


Figure 3.7 Distance entre vecteurs exprimés en cosinus

3.6.3 Caractéristiques de l'entête

La technique la plus récente pour le spamming est la méthode **tag-based**. Cette méthode permet de trouver des pages camouflées mieux que les algorithmes précédents qui sont basés sur les termes et les liens (Shekoofeh et Alireza, 2013).
Les méta-tags qui apparaissent dans l'en-tête du document HTML ont toujours été la cible du spam. En raison du gros volume de spam, les moteurs de recherche donnent actuellement peu de priorité à ces balises, voire les ignorent complètement.
Pour cette partie, nous considérons les caractéristiques suivantes :

- Densité-head (max, min, mode, médiane, écart-type)

Le calcul de cette caractéristique est similaire à celui du calcul de la densité pour les mots clés à la seule différence qu'il concerne seulement la densité des mots contenus dans les balises méta.

- Répétition méta description et keyword

Les méthodes de spam basées sur le contenu adaptent essentiellement le contenu des champs de texte dans les pages HTML pour rendre les pages de spam plus pertinentes pour certaines requêtes. Ce type de spam est également appelé spamming term. Il existe deux techniques principales de spam de contenu, qui créent simplement des contenus synthétiques contenant des termes de spam. La première consiste à répéter certains termes importants et la seconde à rajouter de nombreux termes non liés (Karimpour *et al.*, 2012).

Voici deux exemples :

*<meta name="**keywords**" content="bathroom accessories, bathroom accessories uk, bathroom taps, bathroom accessories, bathroom, accessories, bathroom accessories uk, bathroom accessory, bathroom accessoires, bathrooms, cheap bathroom accessories, bathroom taps, taps, shower, bathroom taps">*

*<meta name="**description**" content="Information about Yamaha Motorcycle, Super Cars - suzuki, kawasaki and yamaha bikes made. motorcycle chains sprockets partsuzuki kawasaki and yamaha bikes [motorcycle chains sprockets partsuzuki, kawasaki and yamaha motorcycle brake pads and tyre supplier">*

Grâce à la répétition d'un ou de plusieurs termes spécifiques que les spammeurs atteignent, une pertinence accrut pour une page en ce qui concerne un petit nombre de termes de requête.

Dans le but de récupérer cette information, nous avons calculé la répétition pour les zones meta-keyword et meta-description. Nous avons procédé comme suit :

1. *Extraire le contenu des Meta keywords ou description en utilisant Jsoup qui est une bibliothèque Java open source de méthodes conçues pour extraire et manipuler des données stockées dans des documents HTML.*
2. *Tokenisation : StringTokenizer*

3. *Supprimer les mots vides*
4. *Supprimer la ponctuation*
5. *Stemming (EnglishStemmer)*
6. *Calculer le nombre de termes*
7. *Supprimer les doublons*
8. *Calculer le nombre de termes sans répétition*
9. *Calculer la Répétition Meta* $= 100 - \frac{nombre\ de\ mots\ cles\ sans\ repetition * 100}{nombre\ de\ mots\ clés\ avec\ repetion}$
10. *Enregistrer dans la base de données*

- Nombre de ponctuations dans l'URL

Parmi les caractéristiques qui nous ont semblé intéressantes à extraire lors de notre analyse des pages spam et des pages normales, nous avons aussi choisi la présence de signes de ponctuation (*e.g, les* «-») dans l'URL. En effet, les spammeurs ont souvent tendance à séparer des mots avec des ponctuations pour les rendre plus difficiles à détecter *(e,g,: http://www.animated.gifs.btinternet.co.uk/casino-match-bonus)*.

- HTTP statut

HTTP est un protocole de communication utilisé pour communiquer avec un serveur connecté au réseau. Le protocole http permet à un serveur web de transmettre des informations et des pages à un navigateur Web. Il indique un état ou un statut. Les codes de statuts http permettent d'indiquer aux navigateurs les résultats d'une requête menée ou tout simplement d'une erreur.

Toutefois, la page remise au navigateur Web peut être la page Web normale ou un lien brisé comme "HTTP 404: fichier non trouvé» ou «HTTP 503: service non disponible".

Nous avons donc extrait cette information à partir de l'entête et l'avons stockée au niveau du champ http-statut de la base de données.

- Redirection

Il est généralement utilisé en conjonction avec le contenu spam, en servant une page qui redirige immédiatement le navigateur de l'utilisateur vers une page différente (soit via un script côté client ou via le code HTML "méta tag"), ce qui augmente la probabilité que la page soit retournée à la suite d'une recherche. Le procédé est le suivant :

1. *Chercher la balise « location » dans l'entête*
2. *Si elle existe, attribuer la valeur 1, ce qui veut dire que la page est redirigée*
3. *Sinon attribuer la valeur 0.*

3.6.4 Caractéristique du corps (body)

- **POS (NN, VB, ADV, ADJ, PR, DT)**

Le ratio POS (Part of Speech) est basé sur les fonctionnalités linguistiques par lesquelles nous détectons une page comme spam ou non-spam.

Nous avons procédé à une annotation syntaxique du contenu, qui vise à découper le texte en unités lexicales et à attribuer à chaque unité son étiquette lexicale (POS). Pour ce faire, nous avons choisi de travailler avec l'API Stanford Tagger[9]. Nous avons ainsi écrit un programme qui traite le contenu brut et génère son arbre syntaxique. À partir de ce dernier, nous dégageons ensuite la distribution statistique de chaque étiquette syntaxique attribuée auparavant dans le contenu.

Cette analyse peut être utile, car les pages spam contiennent souvent plus de noms que dans les pages non spam (Westbrook et Greene, 2002), du fait que la plupart des termes de requête contiennent des noms.

Nous avons ainsi calculé le pourcentage des fréquences d'utilisation des noms (NN), verbes (VB), adverbes (ADV), adjectifs (ADJ), pronoms (PR) et déterminants (DT), pour chaque page.

---

[9] https://nlp.stanford.edu/software/

Nous avons procédé comme suit :

1. *Extraire le contenu de la page sans les balises HTML en utilisant Jsoup*
2. *Crée un objet StanfordCoreNLP, avec le marquage POS, la lemmatisation,*
3. *Obtenir les phrases contenues dans le texte en utilisant la clé CoreMap*
4. *Itérer sur tous les tokens dans une phrase*
5. *Récupérer et ajouter le POS pour chaque mot dans la liste des POS*
6. *Calculer le pourcentage de chaque POS dans la page*
7. *Enregistrer dans la base de données*

- **Sentiment (VP, P, Neutral, VN, N)**

L'analyse des sentiments consiste à analyser une grande quantité de données pour déterminer les opinions ou les sentiments exprimés dans les textes (Faath, 2015). Elle peut être utilisée pour détecter les opinions des utilisateurs sur des sujets divers (Pang et Lee, 2008). Le but est alors d'attribuer une polarité (positive, négative ou neutre) à des opinions, des sentiments exprimés et présents dans les pages.

Cette analyse peut être utile, car nous estimons que le fait que la plupart des pages spam visent à faire de la publicité ou tout simplement d'attirer l'attention des utilisateurs, favorise le fait que les pages spam contiennent plus de sentiments positifs.

Nous avons extrait les textes des balises HTML qui ont ensuite été segmentés en phrases afin d'annoter chacune d'entre elles selon sa polarité.

Nous avons mis en place une méthode d'annotation en utilisant la libraire Stanford. Cette méthode permet à la fois de segmenter les phrases et d'annoter les opinions contenues dans la page. Par ailleurs, nous avons dû travailler au niveau le plus fin d'annotation, c'est-à-dire l'annotation de chaque phrase contenue dans la page. Nous avons calculé le pourcentage des sentiments très positifs (VP), positifs (P), neutres (Neutral), très négatifs (VN) et négatifs (N) pour chaque page.

Nous avons procédé comme suit :

1. *Extraire le contenu de la page sans les balises HTML en utilisant Jsoup*
2. *Segmenter le texte en utilisant SentenceUtils.listToString qui renvoie la phrase en tant que chaîne avec un espace entre les mots.*
3. *Annoter la liste de phrases tokenisées en utilisant l'annotation de Stanford*
4. *Un arbre est attaché à chaque phrase dans le SentencesAnnotation via SentimentCoreAnnotations.SentimentAnnotatedTree*
5. *Stocker tous les sentiments VP, P, Neutral, VN, N dans une liste*
6. *Calculer le pourcentage de chaque sentiment dans la page*
7. *Enregistrer dans la base de données*

- **NER (Named Entity Recognition)**

Le NER reconnaît les entités nommées, les entités numériques et les entités temporelles. Le terme d'« entité nommée » est utilisé dans le domaine du traitement automatique des langues (TALN).

L'extraction d'entités comprend deux parties, l'identification de l'entité et la catégorisation. Cette dernière consiste à attribuer un type sémantique à l'entité identifiée. Repérer et catégoriser les EN permet un accès particulièrement pertinent au contenu des documents et, de ce fait, représente un enjeu crucial pour l'analyse et la compréhension automatique des textes.

Les entités identifiées sont organisées en trois grandes catégories :

- Entités nommées : « Person », « Organization », « Location », « Misc »
- Entités temporelles : « Date », « Time »
- Entités numériques : « Money », « Percent »

Nous avons procédé comme suit :

1. *Extraire le contenu de la page sans les balises HTML en utilisant Jsoup*

2. *Crée un objet StanfordCoreNLP, avec le marquage (lemmatisation, NER)*
3. *Appeler la procédure identifyNER qui permet d'identifier les entités et de retourner Map<List>. Ses paramètres sont :*
   - *texte,*
   - *modèle : le nom du modèle parmi les 3 modèles de Stanford*
   - *nom de l'hôte.*

   La procédure retourne la liste des noms dans les phrases et leurs entités pour chaque hôte.
4. *Parcourir la liste de Map et compter le nombre de chaque entité*
5. *Stocker dans la base de données le nombre correspondant à chaque entité (Person, Organization, Location, Misc, Date, Time, Money, Percent)*

- **Nombre d'images**

Nous avons calculé le nombre d'images dans une page, et lorsqu'on trouve qu'une page a beaucoup d'images et que le nombre de mots dans la page est faible, alors on peut suspecter qu'il s'agit d'une page spam. Souvent, il s'agit de pages qui ne font que de la publicité avec très peu de contenu, comme des sites qui offrent des catalogues de produit qui réorientent réellement vers d'autres marchands sans fournir de valeur supplémentaire.

Pour chaque page :

1. *Extraire les balises des images en utilisant Jsoup*
2. *Compter le nombre d'images*
3. *Stocker dans la base de données*

3.7 Base de données

Pour sauvegarder les différentes caractéristiques extraites des pages Web, nous avons créé une base de données sous *MySQL* dans l'environnement *EasyPHP,* appelée « *BDD_Labels*». Nous avons choisi *MySQL* comme système de gestion de base de données afin de pouvoir l'interroger. À cause de sa rapidité, sa robustesse et sa facilité d'utilisation et d'administration. Un autre avantage majeur de *MySQL* est sa documentation très complète et bien structurée.

Cette base de données est caractérisée par la table (data) qui comprend les attributs suivants :

Tableau 3. 5 Description de la table data

Nom attribut	Description
Id	Identifiant de l'hôte
Nom_hote	Nom de l'hôte
Nbr_page	Nombre de pages dans l'hôte
Nb-P-Url	Nombre de ponctuations dans l'URL
http_statut	Statut HTTP
Coh_Titre	Cohérence du titre
Redirection	Valeur binaire. Si la page est redirigée ou pas.
NbCarTitre	Nombre de caractères dans le titre
NbCarContent	Nombre de caractères dans le contenu
NbCarMeta	Nombre de caractères dans les balises méta Keyword
NbImg	Nombre d'images dans la page d'index
Max-Dens-Head	Densité maximale dans le head
Min-Dens-Head	Densité minimale dans le head

Nom attribut	Description
NbrMotPage	Nombre de mots dans la page d'index
Negative	Pourcentage des phrases négatives
VNegative	Pourcentage des phrases très négatives
Neutral	Pourcentage des phrases neutres
Positive	Pourcentage des phrases positives
VPositive	Pourcentage des phrases très positives
NN%	Pourcentage des noms dans la page d'index
ADJ%	Pourcentage des adjectifs dans la page d'index
ADV%	Pourcentage des adverbes dans la page d'index
VB%	Pourcentage des verbes dans la page d'index
DT%	Pourcentage des déterminants dans la page d'index

Mod-Dens-Head	Mode de densité dans le head
Median-Dens-Head	Densité médiane dans le head
EcatT-Dens-Head	Écart-type de la densité dans le head
SimCosinus	Mesure de ressemblance entre deux vecteurs
RepetitionMetaDes	Répétition de mots dans les Méta-descriptions
RepetitionMetaKey	Répétition de mots dans les Méta-descriptions
Max-Dens	Densité maximale
Mode-Dens	Mode de la densité
Median-Dens	Densité médiane
EcartT-Dens	Écart type de la densité
MotCle	Nombre de mots clés
MotUnique	Nombre de mots uniques
Taux Mots	Taux des mots clés sans répétition

PR%	Pourcentage des pronoms dans la page d'index
Person	Nombre d'entités de type nom de personne
Organi	Nombre d'entités de type nom d'organisation
Locat	Nombre d'entités de type nom de lieu
Percent%	Nombre d'entités de type pourcentage
Money	Nombre d'entités de type argent
Misc	Nombre d'entités de type divers
Time	Nombre d'entités de type heure
Date	Nombre d'entités de type date
Label	Classe de la page

3.8 Sélection

La sélection des attributs est effectuée dans le but de choisir les éléments les plus pertinents en formant la valeur d'un attribut sur son utilité au classement d'une page dans le corpus. Cette étape est cruciale, car l'utilisation de tous les attributs nécessite beaucoup de temps de calcul et d'espace, ce qui peut aussi influer considérablement sur la performance de certains classificateurs.

À cet effet, deux méthodes de sélection de caractéristiques bien connues sont appliquées pour trouver le degré d'importance des attributs sélectionnés. Une première méthode classe les caractéristiques en fonction de leur pertinence en gain d'information et un sous-ensemble de caractéristiques est alors sélectionné. Le Gain

d'Information (GI) estime l'information mutuelle entre un attribut et la classe cible pour choisir l'attribut qui va le mieux diviser l'ensemble des instances en deux groupes homogènes. La seconde méthode dite analyse en composantes principales (PCA), projette les points originaux dans un sous-espace vectoriel de dimension plus réduite afin d'accomplir une réduction de dimension. Il s'agit de résumer l'information contenue dans un ensemble en un certain nombre de variables synthétiques, qui sont une combinaison linéaire des variables originelles. Ces variables synthétiques sont appelées « composantes principales ».

Nous avons identifié les meilleurs attributs en utilisant ces deux méthodes, pour ensuite réappliquer les mêmes algorithmes sur le même ensemble de données, mais avec les attributs sélectionnés seulement.

3.9 Classification

Avant de lancer la classification d'une ou plusieurs instances, une phase de préparation de l'expérience est nécessaire. Elle consiste à :

- Choisir les données d'apprentissage et de test. Nous avons importé les différents corpus à partir de la base de données qui nous permet de produire des fichiers CSV qui sont reconnus par l'outil WEKA (Hall *et al.*, 2009). Ce dernier nous a permis d'avoir le format ARFF, de WEKA
- Choisir des algorithmes pour la classification, la sélection de caractéristiques, etc.
- Fixer les paramètres pour certains algorithmes puis lancer la classification.

Nous obtenons plusieurs résultats pour cette expérience, soit une matrice de confusion ainsi que différentes mesures de performances qui nous permettent d'évaluer la classification.

3.10 Comparaison

La comparaison consiste à récupérer dans un premier temps, les résultats de classification des algorithmes d'apprentissage les plus performants en utilisant nos attributs seulement. Dans un deuxième temps, les résultats de classification des méthodes d'apprentissage en utilisant les attributs les plus pertinents issus de la combinaison entre notre ensemble et celui proposé par (Castillo *et al.*, 2006). Nous avons ainsi établi des graphiques qui résument notre comparaison. Ceci va nous permettre de faire une analyse générale sur les performances et aussi d'identifier ce qui caractérise le spam Web.

3.11 Conclusion

Différentes études ont été faites pour analyser l'importance du contenu de la page Web et des propriétés associées pour détecter le spam Web.
Compte tenu de cela, nous avons proposé une méthode d'exploration qui utilise du contenu Web pour identifier les pages Web spam.
Dans ce chapitre nous avons détaillé la méthode proposée ainsi que son fonctionnement. Ensuite, nous avons présenté chacune des étapes en donnant les algorithmes requis, les différents modules et les différentes méthodes et fonctions utilisées pour chaque caractéristique extraite. Le chapitre suivant détaillera les techniques d'apprentissage machine prises en compte, ainsi que les différentes expériences menées pour valider ce travail.

CHAPITRE IV

ÉVALUATION DE L'ENSEMBLE DE DONNÉES PRÉPARÉES

4.1 Introduction

Après avoir exposé la méthodologie de préparation de données, nous présentons dans ce chapitre, les expérimentations faites. Dans une première partie, nous présentons les ensembles d'apprentissages sur lesquels nous expérimentons plusieurs méthodes d'apprentissage automatique. Dans la deuxième partie, nous exécutons diverses méthodes d'apprentissage automatique sur deux ensembles de données. Le premier est obtenu avec nos caractéristiques seulement, et cela pour une classification à deux et trois classes. Le deuxième est obtenu avec une combinaison d'attributs, c'est-à-dire nos caractéristiques avec ceux proposés dans WebspamUK-2007. Par la suite, nous discutons et commentons les résultats de chaque expérimentation. Enfin, dans la dernière partie, nous comparons les résultats de notre approche aux résultats d'autres travaux.

4.2 Présentation des ensembles d'apprentissages

Nous allons utiliser deux méthodes d'évaluation. Nous prenons d'abord l'ensemble d'apprentissage que nous avons construit avec nos attributs, puis nous utilisons en second lieu l'ensemble d'apprentissage original WebspamUK-2007, duquel nous sélectionnons les meilleurs attributs, que nous combinons avec l'ensemble d'attributs que nous avons proposé, pour créer un nouvel ensemble combiné.

4.2.1 Ensemble d'entraînement avec les caractéristiques existantes

Nous avons utilisé un ensemble de données WebSpamUK-2007, dans lequel nous avons pris en considération seulement les caractéristiques basées sur le contenu. Ce type de caractéristiques comprend la taille des mots, la longueur des titres, le nombre de mots dans la page, etc.
Le tableau suivant résume le profil de ces données.

Nombre d'instances	5070	Tâches associées	Classification
Nombre d'attributs	98	Nombre de classes	2

4.2.1 Ensemble d'entraînement avec les nouvelles caractéristiques

Nombre d'instances	5070	Tâches associées	Classification
Nombre d'attributs	44	Nombre de classes	2

Ces données que nous nommerons WebspamUQAM-2017 ont été obtenues à partir de notre étude sur la proposition de nouveaux attributs (approche décrite dans le chapitre 3), et en utilisant le contenu de l'ensemble de données WebspamUK-2007.
À partir de l'ensemble de données présentées dans le tableau ci-dessus et en tenant compte du fait que nous travaillons avec un scénario binaire (c'est-à-dire de spam ou de non spam) dans cette partie, les instances appartenant à la 3ème catégorie « inconnue » ne seront pas utilisées.

4.3 Algorithmes d'apprentissage automatique utilisés

Dans cette étude, nous nous intéressons à quelques techniques de classification en utilisant le logiciel Weka (Hall *et al.*, 2009) qui est un outil pratique de forage de données.

Weka[10] est un logiciel de Data Mining libre, très populaire dans la communauté «Apprentissage Machine». Il intègre un grand nombre de méthodes, articulées essentiellement autour des approches supervisées et non supervisées.

Les algorithmes considérés dans ce travail sont : les forêts d'arbres de décision (Random Forest), Logitboost, AdaBoost, les réseaux de neurones multicouches, Logistic Model Tree (LMT), les arbres de décisions (J48), les tables de décision (decision table), Repeated Incremental Pruning (JRip), les K-plus proches voisins (KNN) et les machines à vecteur de support (SVM).

Nous avons utilisé cette liste d'algorithmes sur deux ensembles de données. Dans un premier temps avec WebspamUQAM-2017, puis avec une combinaison WebspamUK-2007 et WebspamUQAM-2017, dans le but d'évaluer leur performance dans la détection de spam Web.

Nous présentons dans ce qui suit, les méthodes d'apprentissage automatique que nous avons utilisées.

4.3.1 J48

L'algorithme J48 de Weka a été utilisé pour générer des arbres de décision. Il s'agit d'une mise en œuvre de l'algorithme C4.5 de Quinlan (1993) dans le cadre de l'apprentissage supervisé. Cet algorithme permet la construction d'un arbre de décision élagué ou non. Cette méthode a pour but global de générer un arbre de décision simple capable de classifier les nouvelles instances avec un certain degré d'erreur tolérée. L'algorithme C4.5 commence par construire l'arbre de décision en divisant récursivement l'ensemble d'apprentissage.

Pour la construction des sous-arbres, C4.5 utilise le taux de gain d'information (IGR : information gain rate) pour chacun des attributs possibles qui pourraient potentiellement être utilisés pour diviser les données. L'attribut ayant la plus grande

[10] https://www.cs.waikato.ac.nz/ml/weka/

valeur de gain d'information est choisi comme racine d'un sous-arbre. Cette méthode de construction de sous-arbres est appliquée récursivement jusqu'à ce que l'arbre résultant classifie toutes les instances de l'ensemble d'apprentissage (Yazid et Lounis, 2006).

Pour avoir un bon classificateur, l'arbre de décision doit être élagué. L'élagage de l'arbre de décision s'effectue en remplaçant un sous-arbre entier par une feuille. Cette substitution a lieu si une règle de décision établit que le taux d'erreur attendu dans le sous-arbre est supérieur à celui d'une simple feuille (Bouslama, 2012).

Une méthode alternative consiste à arrêter la construction de l'arbre une fois que l'ensemble d'apprentissage a été suffisamment subdivisé en utilisant un critère.

4.3.2 JRip

L'algorithme JRIP de Weka correspond à la méthode Ripper de William W. Cohen (Cohen., 1995). Il permet de construire itérativement des règles pour couvrir les instances qui n'ont pas été couvertes auparavant. Les règles sont générées de la manière habituelle, mais de la classe la plus rare à la classe la plus fréquente. Ripper possède les mêmes avantages que C4.5, tout en étant bien plus efficace et surtout, il peut manipuler les données bruitées.

JRIP produit des règles indépendantes. Il intègre une première procédure de post-élagage afin de retirer les propositions inutiles, et une seconde procédure pour réduire le nombre de règles dans la base. Il en résulte souvent un classifieur plus compact par rapport aux autres algorithmes d'apprentissage (Bouslama, 2012).

4.3.3 Adaboost

La méthode Adaboost (Adaptive Boosting) permet de combiner plusieurs règles simples pour créer une autre plus performante. L'algorithme Adaboost a été proposé par Freund et Schapire (1996). L'idée de base du Boosting est de combiner des

«règles» simples pour créer un ensemble dont la performance de chaque élément est meilleure que celle de l'ensemble. Plusieurs variantes de la méthode Boosting existent, parmi elles la méthode AdaBoost.M1 conçue spécifiquement pour la classification. Cette méthode peut utiliser n'importe quel algorithme d'apprentissage de classification. Cependant, l'algorithme d'apprentissage doit être capable de manipuler des instances pondérées (le poids doit être un nombre positif). La présence des poids d'instances change la manière de calculer l'erreur par le classificateur qui correspond à la somme des poids des instances mal classifiées divisée par le poids total de toutes les instances, au lieu de la fraction des instances mal classifiées (Yazid et Lounis, 2006).

4.3.4 Logitboost

Les fondateurs de l'approche AdaBoost, Freund et Schapire en 1995, ont toujours essayé d'améliorer cet algorithme, ce qui explique l'apparition de plusieurs variantes de cette méthode qui optimisent différemment la pondération. On s'intéresse ici à l'algorithme LogitBoost qui répond au mieux au problème de classification qui est basé, selon la règle de Bayes, sur la détermination de la probabilité P(y) = [j /x] ou j représente la classe à laquelle une observation x appartient. Il réduit au minimum son critère pour adapter un modèle de régression logistique en optimisant le logarithme de la vraisemblance (Bahri et Maddouri, 2008).

La différence entre Adaboost et Logitboost provient de la fonction de coût à minimiser lors de l'apprentissage qui est celle d'une régression logistique pour ce dernier.

4.3.5 Random Forest

La méthode des forêts aléatoires a été développée en 2001 par Breiman et Cutler. Il s'agit d'une approche d'ensemble pour la classification et la régression qui fonctionne

en construisant une multitude d'arbres de décision au moment de l'entrainement (Gadhvi et Madhu, 2013). La prédiction ou classification se fait alors en fonction d'un système de vote majoritaire au sein de ces différents arbres. Le principe de la forêt aléatoire est alors de chercher à tirer profit de cette instabilité en les agrégeant entre eux.

La forêt aléatoire se construit en concevant un arbre sur un sous-échantillon tiré aléatoirement (ou échantillon « out-of-bag »). Ensuite, pour chacun des arbres à construire, un sous-ensemble de $q \leq$ P (variables explicatives) est sélectionné aléatoirement et sert à leur élaboration respective.

L'objectif de cette approche est de rendre les arbres construits plus indépendants entre eux ce qui offre de meilleures performances lors de l'agrégation en forêt. L'approche possède l'avantage de pouvoir être utilisée même sur des données de haute dimension et d'être simple à mettre en œuvre. L'utilisation de la forêt aléatoire permet également de s'affranchir de toute phase d'élagage et de tout problème lié à la multicolinéarité des variables (Mayotte, 2015).

4.3.6 Réseau de neurones

Le Perceptron multicouche est un classificateur linéaire de type réseau neuronal formel organisé en plusieurs couches au sein desquelles une information circule de la couche d'entrée vers la couche de sortie. Chaque couche est constituée d'un nombre variable de neurones, les neurones de la couche de sortie correspondant toujours aux sorties du système (Haccoun, 2012).

Les réseaux de neurones forment une structure composée d'une succession de couches de nœuds, qui définissent une fonction de transformation non linéaire des vecteurs d'entrées aux sorties. Ces vecteurs représentent les caractéristiques des pages pouvant servir après apprentissage, à reproduire une forme de raisonnement humain. Ces caractéristiques permettent d'ajuster les coefficients synaptiques du réseau de neurones durant la phase d'apprentissage. Cette dernière se fait à partir d'une

collection de pages (spam, non spam). Une fois l'apprentissage réalisé, la structure fonctionne comme un filtre antispam classique. Le nombre de couches déployées et la disposition des neurones dans le réseau influencent le résultat de la classification.
Comparés aux autres méthodes de classification supervisée, les réseaux de neurones sont rapides et permettent de régler le taux de mauvaise classification (Faux Positifs) en ajustant le seuil de sensibilité, mais nécessitent également un entraînement long et laborieux, car demandant une certaine expertise pour optimiser les différents paramètres.

4.3.7 LMT

L'algorithme Logistic Model Tree (LMT) (Landwehr *et al.*, 2005) qui combine arbres de décision et modèles de régression logistique est un arbre composé d'une structure d'arbre de décision standard de taille réduite, avec des fonctions de régression logistique au niveau des feuilles. Les paramètres des fonctions de régression logistique sont calculés pour maximiser les probabilités sur les données observées. Le LMT est un classifieur probabiliste dont les résultats sont généralement pertinents lorsque l'on dispose de peu de données d'apprentissage. La structure d'arbre permet de minimiser les erreurs d'entraînement, tandis que la régression logistique évite le surapprentissage en limitant la taille de l'arbre (Charton *et al.*, 2013).

4.3.8 Les tables de décision

Cette approche a pour but de construire et d'utiliser un classificateur de table de décision.
Elle a une représentation simple avec une règle qui prend par défaut la classe majoritaire. Cette représentation appelée DTM (Tableau de décision à majorité) a deux composants : un schéma qui est un ensemble d'attributs inclus dans la table, et un corps se composant d'instances étiquetées de l'espace défini par les attributs dans

le schéma (Kohavi, 1995). Si aucune instance n'est trouvée, la classe de majorité du DTM est retournée, sinon, la classe majorité de toutes les instances correspondantes est retournée. Pour construire un DTM, l'algorithme d'induction doit décider quels attributs sont à inclure dans le schéma et quelles instances sont à stocker dans le corps (Yazid et Lounis, 2006).

4.3.9 SVM

Les SVM sont des algorithmes qui utilisent une transformation non linéaire des données d'apprentissage. Ils projettent les données d'apprentissage dans un espace de plus grande dimension que leur espace d'origine. Dans ce nouvel espace, ils cherchent l'hyperplan qui permet une séparation linéaire optimale des données d'apprentissage en utilisant les vecteurs de support et les marges définies par ces vecteurs.

La technique SVM fait partie des techniques classiques de fouille de données. Elle fait partie des méthodes d'apprentissage qui ont réalisé des performances meilleures que les méthodes statistiques traditionnelles en matière de classification.

L'algorithme implanté au niveau de l'outil Weka est la version SMO développée par John Platt (Platt, 1998). Il remplace toutes les valeurs manquantes et transforme les attributs nominaux en binaires. Par défaut, il normalise tous les attributs; ainsi la sortie est basée sur ces attributs normalisés et non pas sur les données originales. Il est utilisé seulement pour des problèmes de classification.

4.3.10 KNN

L'idée de base de l'algorithme des *k-plus proches voisins*, traduction de *k-nearest neighbor (kNN)* en anglais, consiste à classer une nouvelle page P non étiquetée sur la base de la classe dominante des K plus proches voisins dans l'espace d'apprentissage,

en mesurant la distance entre les pages d'apprentissage et la nouvelle page P non étiquetée.

Nous avons appliqué l'algorithme IBK (Instance-based learning algorithm) conçu par D. W. Aha et al, et qui est intégré dans l'outil Weka.

Il fonctionne selon le principe intuitif que les objets les plus proches ont plus de chances d'appartenir à la même classe. La classe majoritaire des K voisins les plus proches (ou la moyenne des distances pondérées si la classe est numérique) est assignée à la nouvelle instance.

4.4 Application de plusieurs approches d'apprentissage automatique sur 2 classes de données

Nous avons utilisé la validation croisée (cross validation) avec 10 sous-échantillons de test pour augmenter la signification statistique des résultats. Le principe de la validation croisée consiste à diviser les données d'apprentissage en 10 sous-échantillons de tailles égales, puis de retenir l'un de ces échantillons et de rouler l'algorithme d'apprentissage sur les 9 restants. Recommencer ce processus pour chaque échantillon. Nous avons adopté la F-mesure moyenne, la surface sous la courbe ROC (AUC), le taux de vrais positifs (VP), le taux de faux positifs (FP), la précision et le rappel, comme paramètres d'évaluation pour mesurer les performances de ces algorithmes d'apprentissage. Toutes les méthodes d'apprentissage doivent être en mesure de maîtriser ces risques en visant à maximiser les taux de la F-mesure et d'AUC et en baissant le ratio de faux positifs.

Nous présentons dans ce qui suit les résultats obtenus pour chaque expérimentation.

Tableau 4. 1 Résultats de classification de l'algorithme J48

	F-mesure	AUC	Précision	Rappel	VP	FP	Matrice
Tous	0,931	0,632	0,942	0,946	0,946	0,713	a b ← classified as 4714 12 | a = nonspam 263 81 | b = spam
GI	0,930	0,604	0,944	0,946	0,946	0,724	a b ← classified as 4718 7 | a = nonspam 263 77 | b = spam
PCA	0,931	0,712	0,928	0,937	0,937	0,589	a b ← classified as 4626 100 | a = nonspam 217 127 | b = spam

Tableau 4. 2 Résultats d'exécution de l'algorithme JRIP

	F-mesure	AUC	Précision	Rappel	VP	FP	Matrice
Tous	0,931	0,625	0,938	0,945	0,945	0,699	a b ← classified as 4704 22 | a = nonspam 258 86 | b = spam
GI	0,932	0,633	0,936	0,945	0,945	0,686	a b ← classified as 4698 28 | a = nonspam 253 91 | b = spam
PCA	0,929	0,643	0,927	0,940	0,940	0,673	a b ← classified as 4668 58 | a = nonspam 248 96 | b = spam

Tableau 4. 3 Résultats de classification de l'algorithme Adaboost

	F-mesure	AUC	Précision	Rappel	VP	FP	Matrice
Tous	0,964	0,839	0,966	0,967	0,967	0,407	a b <-- classified as 4710 16 | a = nonspam 150 194 | b = spam
GI	0,964	0,840	0,966	0,967	0,967	0,407	a b <-- classified as 4710 16 | a = nonspam 150 194 | b = spam
PCA	0,899	0,744	0,869	0,932	0,932	0,932	a b <-- classified as 4726 0 | a = nonspam 344 0 | b = spam

Tableau 4. 4 Résultats de classification de l'algorithme Logitboost

	F-mesure	AUC	Précision	Rappel	VP	FP	Matrice
Tous	0,964	0,838	0,967	0,968	0,968	0,407	a b <-- classified as 4712 14 | a = nonspam 150 194 | b = spam
GI	0,964	0,842	0,967	0,968	0,968	0,407	a b <-- classified as 4712 14 | a = nonspam 150 194 | b = spam
PCA	0,902	0,742	0,907	0,932	0,932	0,916	a b <-- classified as 4721 5 | a = nonspam 338 6 | b = spam

Tableau 4. 5 Résultats de classification de l'algorithme Random Forest

	F-mesure	AUC	Précision	Rappel	VP	FP	Matrice
Tous	0,939	0,790	0,948	0,950	0,95	0,648	a b <-- classified as 4713 13 | a = nonspam 239 105 | b = spam
GI	0,940	0,791	0,948	0,951	0,951	0,640	a b <-- classified as 4713 13 | a = nonspam 236 108 | b = spam
PCA	0,938	0,840	0,949	0,950	0,950	0,661	a b <-- classified as 4717 9 | a = nonspam 244 100 | b = spam

Tableau 4. 6 Résultats de classification de l'algorithme Réseau de neurones

	F-mesure	AUC	Précision	Rappel	VP	FP	Matrice
Tous	0,964	0,835	0,965	0,967	0,967	0,38	a b <-- classified as 4698 28 | a = nonspam 140 204 | b = spam
GI	0,962	0,821	0,963	0,965	0,965	0,401	a b <-- classified as 4699 27 | a = nonspam 148 196 | b = spam
PCA	0,928	0,733	0,926	0,938	0,938	0,657	a b <-- classified as 4654 72 | a = nonspam 242 102 | b = spam

Tableau 4. 7 Résultats de classification de l'algorithme LMT

	F-mesure	AUC	Précision	Rappel	VP	FP	Matrice
Tous	0,964	0,834	0,965	0,967	0,967	0,391	a b <-- classified as 4703 23 | a = nonspam 144 200 | b = spam
GI	0,963	0,810	0,965	0,967	0,967	0,404	a b <-- classified as 4706 20 | a = nonspam 149 195 | b = spam
PCA	0,936	0,654	0,936	0,945	0,945	0,624	a b <-- classified as 4676 50 | a = nonspam 230 114 | b = spam

Tableau 4. 8 Résultats de classification de l'algorithme Decision Table

	F-mesure	AUC	Précision	Rappel	VP	FP	Matrice
Tous	0,968	0,821	0,971	0,971	0,971	0,371	a b <-- classified as 4718 8 | a = nonspam 137 207 | b = spam
GI	0,968	0,821	0,971	0,971	0,971	0,371	a b <-- classified as 4718 8 | a = nonspam 137 207 | b = spam
PCA	0,918	0,709	0,931	0,939	0,930	0,810	a b <-- classified as 4715 11 | a = nonspam 299 45 | b = spam

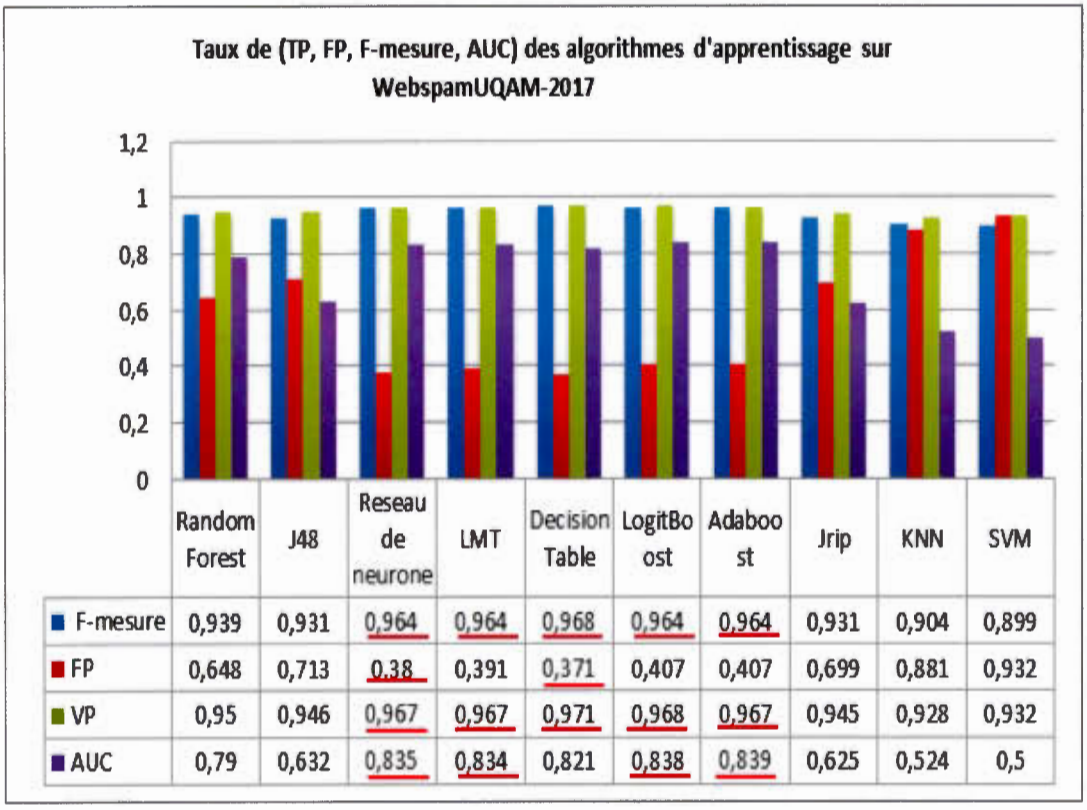
Tableau 4. 9 Résultats de classification de l'algorithme SVM

	F-mesure	AUC	Précision	Rappel	VP	FP	Matrice
Tous	0,899	0,500	0,869	0,932	0,932	0,932	a b <-- classified as 4724 2 | a = nonspam 344 0 | b = spam
GI	0,899	0,500	0,869	0,932	0,932	0,932	a b <-- classified as 4724 2 | a = nonspam 344 0 | b = spam
PCA	0,899	0,500	0,869	0,932	0,932	0,932	a b <-- classified as 4724 2 | a = nonspam 344 0 | b = spam

Tableau 4. 10 Résultats de classification de l'algorithme KNN

	F-mesure	AUC	Précision	Rappel	VP	FP	Matrice
Tous	0,904	0,525	0,894	0,928	0,928	0,881	a b <-- classified as 4687 39 | a = nonspam 325 19 | b = spam
GI	0,896	0,496	0,868	0,925	0,925	0,933	a b <-- classified as 4689 37 | a = nonspam 344 0 | b = spam
PCA	0,920	0,701	0,920	0,920	0,920	0,553	a b <-- classified as 4524 202 | a = nonspam 203 141 | b = spam

La figure ci-dessous nous permet de comparer la performance des différents algorithmes utilisés.



	Random Forest	J48	Reseau de neurone	LMT	Decision Table	LogitBoost	Adaboost	Jrip	KNN	SVM
F-mesure	0,939	0,931	0,964	0,964	0,968	0,964	0,964	0,931	0,904	0,899
FP	0,648	0,713	0,38	0,391	0,371	0,407	0,407	0,699	0,881	0,932
VP	0,95	0,946	0,967	0,967	0,971	0,968	0,967	0,945	0,928	0,932
AUC	0,79	0,632	0,835	0,834	0,821	0,838	0,839	0,625	0,524	0,5

Figure 4.1 Performance des algorithmes d'apprentissage sur l'ensemble d'apprentissage WebspamUQAM-2017

Après avoir confronté les données de WebspamUQAM-2017 aux algorithmes d'apprentissage, nous avons constaté que plus de 93% des instances ont été classées correctement. Nos caractéristiques ont donnée de bons résultats (sauf pour SVM et KNN) avec une précision entre 0.93 et 0.97, une F-mesure allant de 0.93 à 0.96, et des taux de surface sous la courbe ROC (AUC) entre 0.63 et 0.83.

Les réseaux de neurones, LMT, LogitBoost, Adaboost et Decision table ont dépassé les autres méthodes de classification, d'un point de vue quantitatif, mais, il est difficile d'interpréter le résultat obtenu puisqu'elles ne fournissent pas de connaissances explicites compréhensibles et exploitables.
Par exemple, le réseau de neurones correspond à une boite noire qui n'identifie pas des règles et des modèles de classification. Cependant, JRip et J48 produisent des résultats interprétables et explicites. Ainsi, en appliquant J48, nous avons obtenu un arbre de décision que nous présentons dans la figure suivante (Figure 4.2) :

J48 pruned tree : Attributes: 44

NbCaraContent <= 0
| taux <= 0.409091: spam (57.0)
| taux > 0.409091: nonspam (6.0/2.0)
NbCaraContent > 0
| Nb-P-Url <= 4
| | NbCaraTitre <= 90: nonspam (4714.18/236.5)
| | NbCaraTitre > 90
| | | NbCaraMeta <= 1
| | | | ADV% <= 1.07: spam (23.03/1.87)
| | | | ADV% > 1.07: nonspam (7.78/0.27)
| | | NbCaraMeta > 1
| | | | Median-Dens <= 0.1272
| | | | | RepetitionMetaKey <= 57
| | | | | | Median-Dens-Head <= 10: nonspam (18.55/2.0)

```
| | | | | | Median-Dens-Head > 10: spam (2.47/0.47)
| | | | | RepetitionMetaKey > 57: spam (4.7/0.7)
| | | | Median-Dens > 0.1272: nonspam (190.29/10.06)
| Nb-P-Url > 4
| | Locat <= 1: nonspam (29.0/1.0)
| | Locat > 1
| | | Max-Dens <= 4.3716: nonspam (7.0)
| | | Max-Dens > 4.3716: spam (10.0/2.0)
Number of Leaves :  12
```

Figure 4.2 Arbre de décision formé lors de l'application de J48 sur l'ensemble d'apprentissage WebspamUQAM-2017

Il contient 12 feuilles, donc 12 règles de décision interprétables. Par exemple, si on a une page qui a :

NbCaraContent > 0 AND

Nb-P-Url <= 4 AND NbCaraTitre <= 90: nonspam (4714.18/236.5) alors c'est une page non spam.

Si on a une page qui a: *NbCaraContent > 0 AND Nb-P-Url <= 4 AND NbCaraTitre > 90 AND NbCaraMeta <= 1 AND ADV% <= 1.07: spam (23.03/1.87)* Alors c'est une page spam.

L'algorithme Jrip produit aussi des règles de décision sur lesquelles il se base pour classifier les pages. Ces règles sont explicites et interprétables. La figure suivante (Figure 4.3) présente les règles que nous avons obtenues lors de l'application de Jrip sur l'ensemble de données avec nos caractéristiques.

```
JRIP rules:
==========
(NbCaraContent <= 0) => label=spam (63.0/4.0)
(NbCaraMeta <= 13) and (NbCaraTitre >= 91) and (ADV% <= 1.03) => label=spam (22.0/1.0)
(Organi <= 0) and (NbCaraContent >= 158) and (taux <= 0.692308) and (ADV% >= 4.17) =>
label=spam (17.0/5.0)
(NbCaraMeta <= 16) and (PR% >= 11.03) and (taux >= 0.767123) and (NN% <= 62.2) => label=spam
(8.0/0.0)
(Max-Dens >= 8.1731) and (Locat >= 7) and (DT% <= 1.18) => label=spam (11.0/3.0)
 => label=nonspam (4949.0/236.0)
Number of Rules : 6
```

Figure 4.3 Règles de décision obtenues lors de l'application de Jrip sur l'ensemble d'apprentissage WebspamUQAM-2017

Le résultat est donc compréhensible et il peut être compris et utilisé par un humain. Par exemple, si on a une page qui a : (NbCaraMeta <= 16) and (PR% >= 11.03) and (taux >= 0.767123) and (NN% <= 62.2) => label=spam (8.0/0.0), alors c'est une page spam.

En plus de produire des règles interprétables, Jrip et J48 associent une information sur la qualité de chaque règle. Ils donnent un poids à la fin de chaque règle, Weka affiche deux valeurs entre parenthèses (voir figure 4.2 et figure 4.3). Le premier nombre, à gauche, représente le nombre d'instances correctement classifiées, c'est-à-dire les instances parmi tous les exemples d'apprentissage où la règle vérifie toutes les conditions de chaque instance et le résultat de la classification est correct. Le deuxième nombre, à droite, représente les instances incorrectement classifiées, c'est-à-dire les instances parmi tous les exemples d'apprentissage où la règle vérifie toutes les conditions de chaque instance, mais le résultat de classification est incorrect. Par exemple, la règle NbCaraTitre >= 68 and ADV% <= 1.03 => label=spam est pondérée par les valeurs (24.0/2.0).

Une remarque importante par rapport aux algorithmes Random Forest, J48 et JRip est qu'ils atteignent un taux élevé de vrais positifs (spams réels) entre 0,94 et 0,95. Mais ils atteignent également un taux élevé de faux positifs (non spams classés comme spam) ce qui est plus grave que le fait de rater un spam. C'est pour cette raison qu'ils donnent un taux de F-mesure (0,931-0,939) moins bon que les algorithmes précédents (Réseau de neurones, LMT, Adaboost, Logitboost et Decision Table).

Nous constatons que le SVM donne de mauvais résultats. Son taux de surface sous la courbe ROC (AUC) est de 0,5, ce qui signifie qu'il n'y a aucune séparation entre les deux classes. Cela pourrait être expliqué par le fait que la validation de l'algorithme s'est effectuée sur 1/10 de l'ensemble d'apprentissage et que la surface séparatrice ne donne qu'une vue partielle sur l'ensemble des données, tout en tenant compte de l'effet qu'exercent les attributs non pertinents. Ces derniers vont diminuer la performance de l'algorithme, car ils affectent le calcul des distances qu'estime l'algorithme pour trouver la marge de sécurité maximale autour de la surface de séparation. Donc, même la représentation dans des espaces de grandes dimensions ne va pas aider à trouver les hyperplans de séparation.
En examinant les résultats de l'algorithme des k plus proches voisins, nous constatons qu'il se classe à la toute fin, juste avant SVM. On peut justifier cette faible performance par deux raisons. Premièrement, le choix du paramètre k qui est très difficile et deuxièmement, par le grand nombre de dimensions et l'influence des attributs non discriminants ou interdépendants qui biaisent le calcul des distances.

Après quelques tests et à partir des observations faites sur les règles obtenues, nous avons supposé qu'il y avait, peut-être, une corrélation entre les attributs lors du fonctionnement des algorithmes. Nous avons donc eu recours à deux méthodes de sélection d'attributs offertes par Weka qui sont « Info Gain Attribute Eval » (GI) et l'analyse en composantes principales (PCA). Nous avons identifié les 30 meilleurs

attributs afin de réappliquer la même liste d'algorithmes, mais avec les attributs sélectionnés seulement. Nous avons alors éliminé les attributs non discriminants.

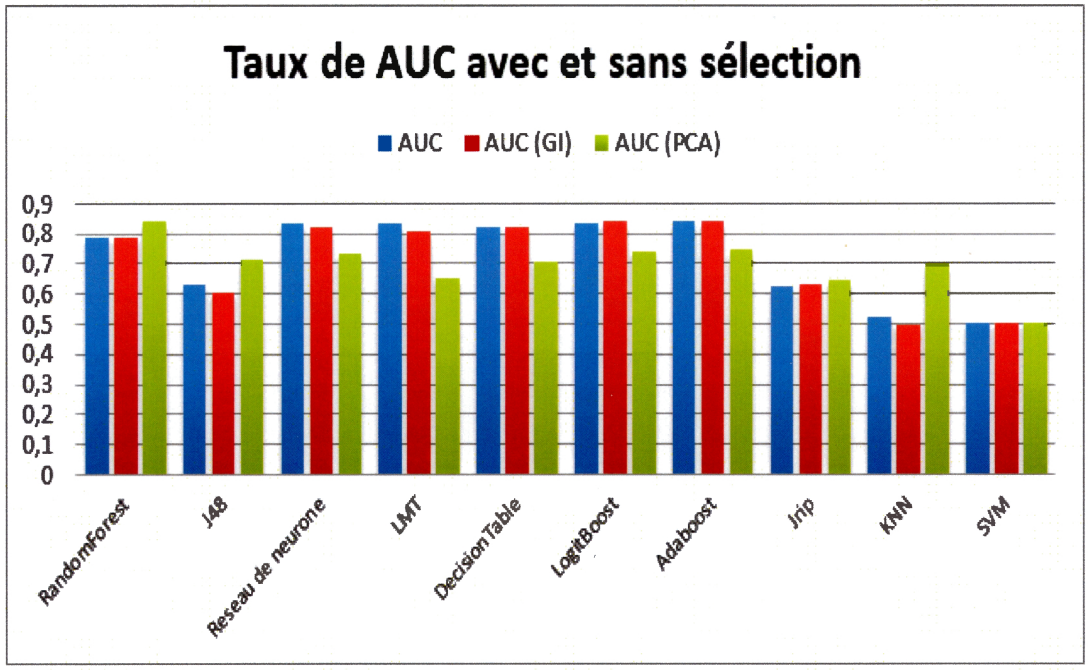


Figure 4.4 Taux de AUC avec et sans sélection d'attributs

En comparant les résultats obtenus après la sélection des attributs et ceux sans la sélection d'attributs, on remarque qu'une sélection par PCA pour les algorithmes J48, Jrip et Random forest a amélioré la surface sous la courbe Roc, soit une amélioration de 0,02 à 0,08. Tandis qu'une sélection par le gain d'information a causé une diminution de la valeur moyenne de la AUC du J48 de 0.63 à 0.60 soit une baisse de 0.03 qui peut être négligée.

L'amélioration la plus prononcée dans cette évaluation est celle de l'algorithme KNN sur le nouvel ensemble généré par PCA. Cette amélioration est le résultat de l'élimination des attributs non discriminants qui ont une influence directe sur la valeur de la distance entre les instances.

La performance de PCA est évidente, car cette technique utilise une transformation d'axes qui tente de représenter au mieux le nuage des points. La non-corrélation des attributs et la variance maximale entre eux permettent d'un côté de filtrer les attributs interdépendants et d'autre part, d'éliminer les attributs non discriminants et redondants. Ceci peut aussi être confirmé par la performance de l'algorithme KNN qui s'est amélioré et qui a donné de bons résultats avec une surface sous la courbe Roc de 0.70 en utilisant une évaluation par la sélection PCA, ce qui n'est pas le cas avec la sélection d'attributs par le gain d'information.
Par contre, pour les classifications avec Jrip, Random Forest, Decision table et SVM, on ne remarque aucune amélioration pour le gain d'information (GI). Il a engendré une diminution des valeurs de la surface sous la courbe ROC (AUC) pour les algorithmes J48, Réseau de Neurones, LMT et KNN.

Concernant Adaboost et Logitboost, ils offrent de meilleurs taux de classification après la sélection par le gain d'information (GI) avec une augmentation de la valeur sous la courbe ROC (AUC) de 0,01.

4.5 Application de plusieurs approches d'apprentissage sur 3 classes de données

Nombres d'instances:	5495	Tâches associées	Classification
Nombres d'attributs	44	Nombre de classes	3

Dans cette partie, nous avons réappliqué la même liste d'algorithmes sur l'ensemble de données avec nos attributs, mais avec trois classes cette fois-ci, c'est-à-dire que les instances appartenant à la catégorie « inconnue » seront utilisées pour la classification.

Tableau 4. 11 Résultats de classifications de plusieurs algorithmes sur l'ensemble WebspamUQAM-2017 avec 3 classes de données

Méthodes	F-Measure	AUC	VP	FP	Confusion Matrix
Random Forest	0,851	0,775	0,881	0,634	a b c <-- classified as 4682 9 35 | a = nonspam 233 93 18 | b = spam 333 24 68 | c = undecided
Réseau de neurones	**0,867**	0,763	0,886	0,532	a b c <-- classified as 4614 19 93 | a = nonspam 134 190 20 | b = spam 340 18 67 | c = undecided
LMT	**0,869**	0,784	0,896	0,581	a b c <-- classified as 4698 14 14 | a = nonspam 160 173 11 | b = spam 359 14 52 | c = undecided
DecisionTable	**0,869**	0,758	0,898	0,574	a b c <-- classified as 4706 13 7 | a = nonspam 150 183 11 | b = spam 363 19 43 | c = undecided
LogitBoost	**0,870**	**0,795**	0,899	0,573	a b c <-- classified as 4710 13 3 | a = nonspam 150 189 5 | b = spam 362 20 43 | c = undecided
Adaboost	0,853	0,750	0,892	0,618	a b c <-- classified as 4710 16 0 | a = nonspam 150 194 0 | b = spam 402 23 0 | c = undecided
J48	0,837	0,678	0,874	0,695	a b c <-- classified as 4686 28 12 | a = nonspam 262 73 9 | b = spam 359 22 44 | c = undecided

On remarque que la performance des algorithmes a diminué par rapport à la classification binaire, soit une diminution de la F-mesure de 0,09 et une diminution de la surface sous la courbe ROC (AUC) d'environ 0,04. Ceci confirme la supériorité des résultats de la classification binaire.

Les méthodes des réseaux de neurones, de LMT, LogitBoost et Decision table ont dépassé les deux autres méthodes de classification, d'un point de vue quantitatif. La même observation donnée avec la classification binaire demeure valide et les explications sont les mêmes. De plus, ils atteignent également un taux légèrement

élevé de faux positifs. C'est pour cette raison que leur taux de F-mesure est à la baisse par rapport à la classification binaire.
Les résultats sur les données avec trois classes confirment les interprétations données précédemment pour les données avec deux classes.

4.6 Combinaison d'attributs

Nous avons ensuite essayé de trouver d'autres améliorations à ces résultats de classification tout en cherchant les attributs les plus discriminants dans le processus de classification du spam Web. Nous avons, ainsi, combiné les attributs de WebspamUQAM-2017 avec ceux proposés dans WebspamUK-2007.
Suite aux expériences faites, à la sélection d'attributs, et à partir des observations faites sur les règles obtenues, nous avons alors éliminé les attributs non discriminants dans chaque ensemble. Nous avons identifié 30 des 44 attributs de WebspamUQAM-2017, comme par exemple : Taux_mots, Similarité-cosinus, Cohérence du titre, Pourcentage des déterminants (DT%), Organi etc.

C'est aussi le cas pour 38 des 98 attributs de WebspamUK-2007 tels que : le nombre de mots dans la page d'index (HST_1), le nombre de mots dans le titre de la page avec max Pagerank (HMG_26), top 100 de précision corpus dans la page d'index (HST_7), l'écart type de la fraction du texte d'ancre (anchor text) dans toutes les pages de l'hôte (STD_77), etc.
Ainsi, nous avons donc identifié les meilleurs attributs pour les deux ensembles de données, ce qui donne un totale de 68 attributs auxquels nous réappliquons les mêmes algorithmes d'apprentissage machine (voir ANNEXE A).

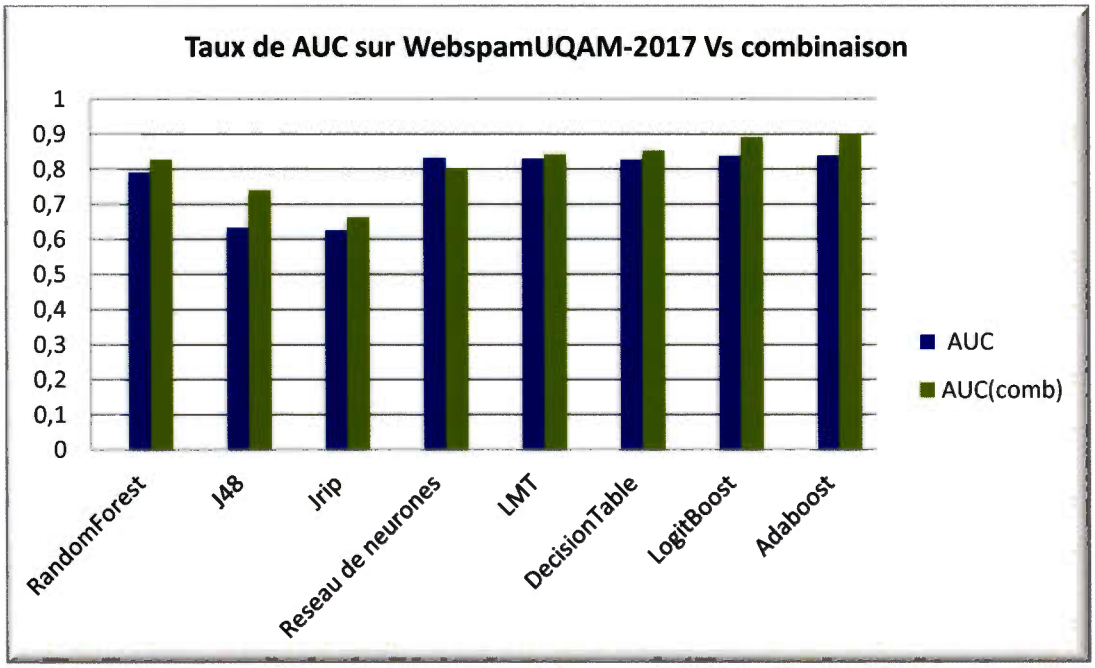


Figure 4.5 Taux de AUC sur WebspamUQAM-2017 vs Combinaison

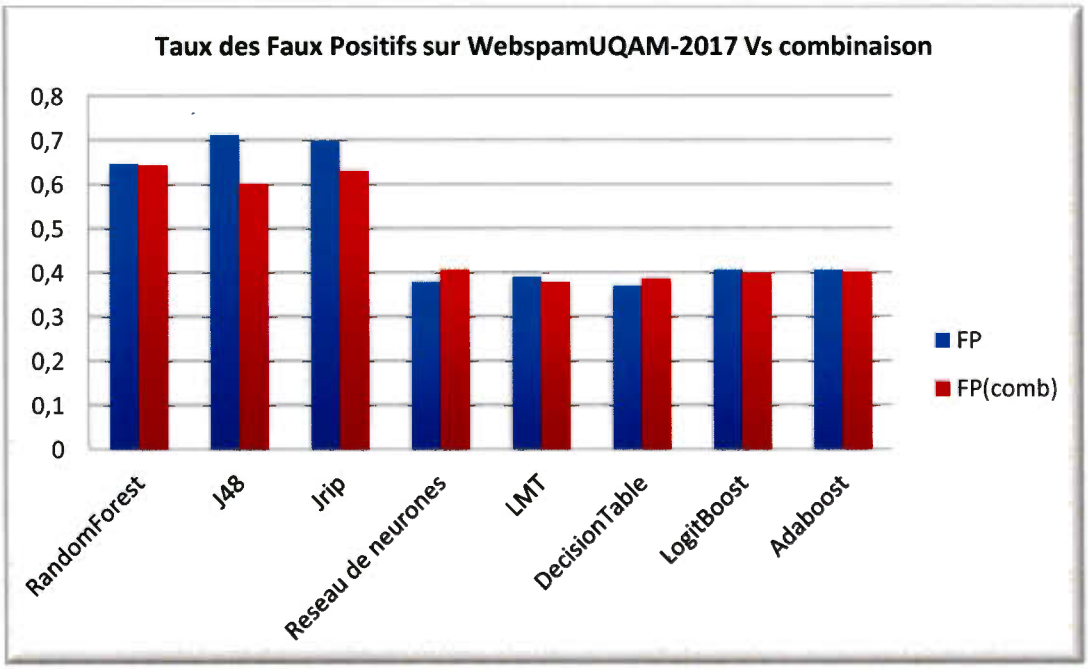


Figure 4.6 Taux des Faux Positifs sur WebspamUQAM-2017 vs Combinaison

En comparant les résultats obtenus après la combinaison de WebspamUQAM-2017 avec ceux de WebspamUK-2007, on remarque que ceux obtenus avec la combinaison ont amélioré la performance de la surface sous la courbe ROC (AUC) de (0.63-0.83) à (0.74-0.89) soit une amélioration de (0.01-0.11). Cette amélioration est le résultat du jumelage des attributs les plus discriminants. Tandis que pour le réseau de neurones, la combinaison cause une diminution du taux de performance de 0.83 à 0.80 soit une baisse de 0.03 qui peut être négligée.

Nous remarquons aussi que le nombre d'objets mal classés diminue pour l'ensemble des algorithmes. Cependant, il a augmenté légèrement pour les algorithmes LMT et réseau de neurones. Ceci nous confirme les interprétations données pour le AUC.

4.7 Comparaison avec les approches existantes

Nos résultats expérimentaux ont aussi été comparés aux approches existantes suivantes:

- (Araujo et Martinez-Romo, 2010)

Nous comparons nos résultats obtenus avec le travail de (Araujo et Martinez-Romo, 2010) pour détecter les pages spam. Ils ont utilisé un système de détection de spam basé sur un classificateur qui combine de nouvelles caractéristiques basées sur les liens avec des modèles de langage (LM).

Des caractéristiques basées sur les liens de la page, comme par exemple la capacité d'un moteur de recherche à trouver la page sur laquelle le lien pointe réellement, c'est-à-dire la cohérence entre une page et une autre par un de ses liens. Ils ont utilisé l'arbre de décision (C4.5), Bayésien Naïf (NB), SVM et Logistic Regression (LR) comme classificateurs pour leur travail expérimental. Les données WebspamUK-2007 ont été aussi utilisées. Les résultats des classificateurs de spam Web utilisant différentes caractéristiques sur l'ensemble d'apprentissage, ont donné une F-mesure

d'environ 0,40 et une AUC d'environ 0,76, ce qui est significativement inférieur à nos résultats.

- (Gongwena *et al.*, 2016)

Notre travail a aussi été comparé à (Gongwena *et al.*, 2016). Dans leur travail, un nouvel algorithme de classement de pages Web a été proposé. Dans cette méthode, le score de classement des pages Web est calculé par la méthode TrustRank combinant la diversité des liens et la distribution des caractéristiques de contenu des pages Web. Certaines caractéristiques, comme le nombre de mots de titre et le ratio de compression des pages Web, ont été extraites pour les caractéristiques basées sur le contenu. De même, pour la diversité des liens, ils ont analysé les informations de lien de pages Web. WebspamUK-2007 a été utilisé à des fins expérimentales. On trouve à partir des résultats de (Gongwena *et al.*, 2016) qu'ils ont atteint une F-mesure d'environ 0,822 avec un taux de précision de 0,926 et de rappel de 0,739, ce qui est aussi inférieur à nos résultats.

- (Rajendra et *al.*, 2016)

Enfin, nous avons comparé nos résultats à (Rajendra et *al.*, 2016). Ces derniers ont proposé une approche combinant du contenu et des techniques basées sur les liens pour identifier les pages spam. Les caractéristiques basées sur le contenu incluent, la densité des termes et le test du rapport des composantes du langage (Part Of Speech). Concernant l'approche basée sur les liens, ils ont exploré la détection collaborative à l'aide du classement de la page personnalisée pour classer la page Web comme spam ou non-spam. Ils ont utilisé l'ensemble de données WebspamUK-2006 pour leurs expériences et ont comparé leurs résultats à certaines approches existantes. Leur approche surpasse clairement quatre autres travaux (Egele *et al.*, 2011; Dai *et al.*, 2009; Becchetti *et al.*, 2008b ; Benczúr *et al.*, 2007) de détection de spam. Ils ont aussi atteint une F-mesure d'environ 0.752 avec une précision d'environ 0.729 et un rappel de 0.776, ce qui est encore inférieur à nos résultats.

À partir des comparaisons ci-dessus, il apparait que notre méthode surpasse clairement les trois approches de détection de spam citées (Tableau 4.12).

Tableau 4. 12 Comparaison de l'approche proposée avec les approches existantes

Approches	F-Mesure
Araujo et Martinez-Romo, 2010	0,4
Gongwena *et al.*, 2016	0,822
Rajendra *et al.*, 2016	0,752
Notre approche	0,968

- (Erdélyi *et al.*, 2011)

Ensuite, nos résultats expérimentaux sont comparés à (Erdélyi *et al.*, 2011). Dans leur travail, diverses classes de caractéristiques de spam Web ont été étudiées. Pour améliorer le classement du spam, ils ont construit trois ensembles différents sur les caractéristiques de contenu. En utilisant le mécanisme d'apprentissage supervisé, y compris la sélection d'ensemble, LogitBoost et Random Forest, leur approche conclue qu'avec des techniques d'apprentissage appropriées, un sous-ensemble de caractéristiques peu coûteux est probablement plus important que l'élaboration de nouvelles caractéristiques complexes. L'ensemble de données Webspam-UK2007 a été utilisé à des fins expérimentales. À partir du tableau 5 figurant dans (Erdélyi *et al.*, 2011), on a constaté qu'en utilisant leur approche, ils ont atteint une AUC de 0,893 qui est supérieur à nos résultats (AUC de 0,839).

Enfin après des tests et d'observations faites sur les règles et les arbres obtenus, nous avons pu identifier les 10 meilleurs attributs des caractéristiques de WebspamUQAM-2017 : Taux mots, Ecart-type Densité-head, Pourcentage des noms (NN%), Similarité-cosinus entre le corps et méta-déscription, Ecart-type Densité,

Cohérence du titre, Pourcentage des pronoms (PR%), Nombre de caractères contenu, Pourcentage des verbes (VB%), Nombre de ponctuation dans l'URL.

Pour confirmer que ces 10 attributs sont discriminants, nous avons effectué d'autres expériences en utilisant les attributs de WebspamUK-2007 seules ensuite nous avons rajouté ces 10 attributs seulement pour voir si les résultats se sont améliorés.

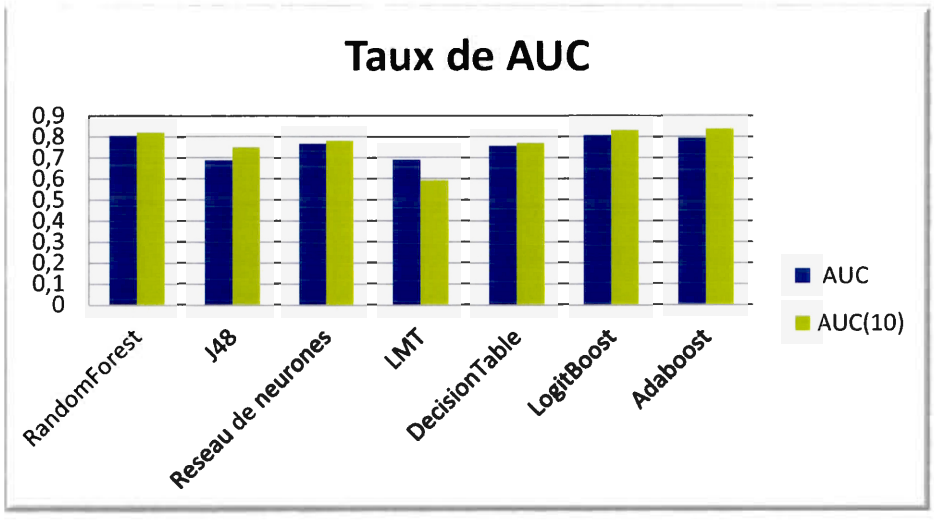


Figure 4.7 Taux des AUC sur WebspamUK-2007 vs WebspamUK-2007+10 attributs de WebspamUQAM-2017

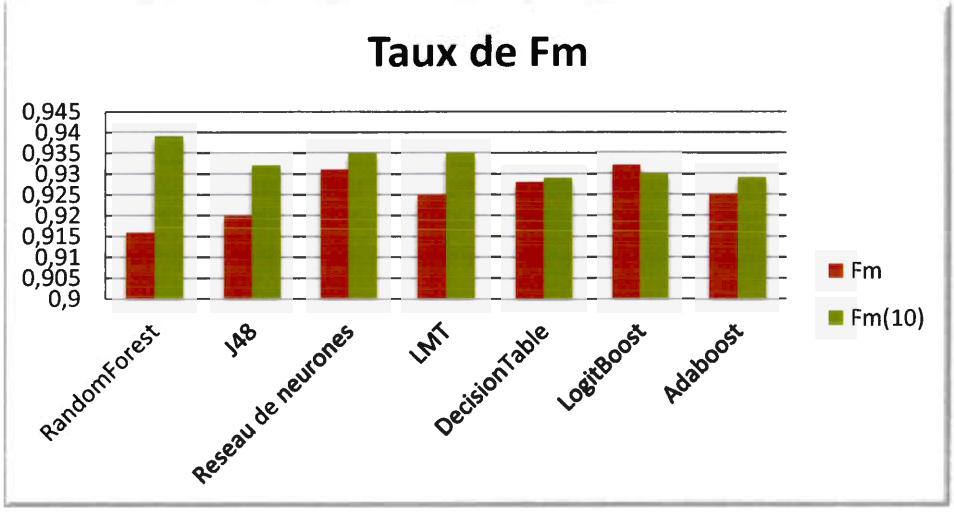


Figure 4.8 Taux des F-mesure sur WebspamUK-2007 vs WebspamUK-2007+10 attributs de WebspamUQAM-2017

Nous remarquons clairement que le fait de rajouter ces 10 attributs améliore le résultat pour la plupart des algorithmes. Cela confirme l'apport de nos caractéristiques dans la détection de spam Web.

4.8 Conclusion

Dans ce chapitre, nous avons vu à travers les expériences qui ont été faites, l'apport et la contribution de nos caractéristiques au monde de la détection de spam Web. Pour des résultats plus précis et pour prouver leur efficacité, nous avons utilisé deux ensembles d'apprentissage et deux types de classification dans les expériences. Les résultats ont été par la suite discutés et interprétés.

Les résultats montrent que quelques classificateurs sont capables d'atteindre une bonne performance telle que les réseaux de neurones, LMT, Decision Table, LogitBoost et Adaboost. De plus, les autres classificateurs obtiennent également une bonne performance, mais en deçà de ceux cités précédemment. Ceci confirme la fiabilité des méthodes d'apprentissage pour la détection de spam dans le Web.

On a vu aussi qu'avec WebspamUQAM-2017, les performances et la précision sont améliorées. En comparant notre approche avec d'autres travaux, on a pu constater que notre approche est très compétitive et reste au-dessus de leurs performances.

CHAPITRE V

CONCLUSION GÉNÉRALE

Dans ce chapitre final, nous récapitulons les éléments importants de cette recherche sur la sélection de caractéristiques pertinentes, la préparation des données et l'application des méthodes d'apprentissage machine pour classifier des pages Web. Nous faisons un survol de notre recherche et nous présentons notre contribution dans ce travail. Enfin, nous étalons les limites rencontrées lors de cette recherche et projetons des perspectives futures.

5.1 Survol de la recherche

Dans la première partie, nous avons commencé par une étude détaillée du domaine de la détection de spam dans le Web. Nous avons présenté la taxonomie des techniques utilisées par les spammeurs, ainsi que les méthodes d'apprentissage automatique utilisées dans ce domaine.

Dans une deuxième partie, nous avons établi un état de l'art sur la détection de spam Web au moyen de diverses méthodes d'apprentissages automatique. Généralement, les travaux se concentrent sur l'exploration de nouveaux attributs pertinents, ou sur la proposition de nouvelles techniques d'apprentissage.
Nous nous sommes tout d'abord concentrés sur la façon de préparer les données avant la classification. Cela nous a amenés à répartir les travaux en trois catégories. La première catégorie se base sur le contenu des pages. La seconde est basée sur la prise en compte des liens des pages Web. Et enfin, la troisième combine plusieurs

caractéristiques comme attributs servant à l'apprentissage. Nous avons ensuite fait un inventaire des méthodes de classification utilisées, en donnant un aperçu des travaux sur la détection de spam Web selon l'approche d'apprentissage machine exploitée. Enfin, nous avons présenté comment les travaux évaluent la performance des méthodes d'apprentissage automatique, selon plusieurs mesures. Une synthèse de quelques travaux antérieurs a ainsi été présentée.

Dans la troisième partie, nous avons introduit notre approche de préparation de données, de la collecte du contenu des pages Web, jusqu'à l'extraction des caractéristiques. Les pages obtenues suite à ce processus de collecte de données, nous obligeant à accomplir un effort supplémentaire de nettoyage et de standardisation. Le processus de préparation de données sert à passer d'un ensemble de données textuelles brut, vers un ensemble de vecteurs numériques.
Nous avons présenté la base de données que nous avons conçue, intitulée WebspamUQAM-2017, et nous avons décrit chacune de ses caractéristiques.

Enfin, dans la quatrième partie de ce travail, nous avons expérimenté plusieurs méthodes d'apprentissage machine sur deux ensembles de données. Le premier est WebspamUQAM-2017, et le second est obtenu par une combinaison d'attributs, c'est-à-dire les attributs de WebspamUQAM-2017 et ceux proposés dans WebspamUK-2007. Nous avons enfin entrainé et évalué différents classificateurs pour la détection de spam Web à base de contenu. Nous avons obtenu des résultats satisfaisants, qui démontrent la fiabilité des méthodes d'apprentissage pour détecter le spam Web. Nous avons aussi discuté les résultats obtenus en les comparant aux résultats d'autres travaux consignés dans la littérature associée au domaine.

5.2 Contribution de la recherche

L'objectif ultime de notre travail est de proposer une méthode de détection de spam Web, qui repose sur l'exploration de différentes caractéristiques à partir du contenu, pour analyser l'impact de certains attributs et méthodes d'apprentissage sur la performance de classification dans le contexte de détection de spam Web. Cette étude nous a permis de connaitre ce qui caractérise les hôtes spam et de proposer une combinaison entre les caractéristiques les plus pertinentes que nous avons sélectionnées et celles proposées dans (Castillo *et al.*, 2006), pour améliorer l'efficacité du système de détection de spam Web.

Nous avons comparé nos résultats de détection de spam avec ceux de travaux précédents. Pour cela nous avons effectué une recherche bibliographique des travaux réalisés par la communauté de la détection de spam Web, qui nous a permis de parvenir à démontrer l'importance des méthodes d'apprentissages automatique dans le contexte de détection du spam Web.

Nous avons abordé la problématique selon quatre étapes principales :

- Étudier et comparer un ensemble de méthodes d'apprentissage automatique.
- Effectuer l'étape de la préparation des données en faisant l'extraction des caractéristiques pour traiter cette problématique.
- Réaliser les expérimentations en utilisant deux ensembles de données, afin d'évaluer la performance de plusieurs méthodes d'apprentissage automatique, pour détecter les hôtes qui diffusent du spam Web.
- Discuter les résultats obtenus, en tirer des conclusions et enfin, les comparer avec d'autres travaux.

5.3 Limites de la recherche

Bien que cette recherche apporte une contribution théorique et pratique, les résultats et les contributions formulés au terme de cette étude sont toutefois limités par certaines contraintes. Nous avons montré dans le deuxième chapitre qu'il existe plusieurs façons de préparer les données avant l'apprentissage. Cependant, nous nous sommes concentrés seulement sur l'approche basée sur le contenu des pages Web. Nous n'avons pas testé l'approche basée sur les liens. Cette limite est due au fait que nous avons eu accès qu'au contenu HTML des pages par l'auteur. La deuxième contrainte rencontrée résidait dans l'incapacité de collecter toutes les pages offertes par la base de données publique WebspamUK-2007, qui contient 114 529 hôtes, mais de laquelle nous n'avons utilisé que 5 495 hôtes. Cette limite est due, d'une part, au serveur de notre laboratoire qui a une capacité de mémoire limitée, et d'autre part, au fait que seuls 6 475 hôtes sont associés à une classe. Enfin, la dernière limite concerne le fait que les données ne sont pas balancées, c'est-à-dire qu'il y a beaucoup plus de pages normales que de pages spam (4726 normales et 344 spam).

5.4 Recherches futures

Bien que les résultats témoignent de la précision des méthodes d'apprentissage machine pour détecter le spam Web, il serait intéressant d'essayer d'améliorer ce travail, notamment :

- d'étendre notre liste d'attributs pour que nous sélectionnions les attributs les plus prédictifs et améliorer ainsi la prédiction ;
- utiliser toutes les pages de l'hôte et non pas juste la page d'index, c'est-à-dire, calculer les attributs pour la page ayant le maximum « PageRank » et ensuite pour toutes les pages de l'hôte et prendre la valeur moyenne;
- faire la prédiction en utilisant plusieurs méthodes d'apprentissage (combiner deux classifieurs, par exemple, réseau de neurones et SVM).

ANNEXE A

Les attributs utilisés pour la combinaison :

Attributs sélectionnés de WebspamUK-2007	
Attributs	**Définition**
HST-1	nombre de mots dans la page d'index
HST-3	longueur moyenne des mots dans la page d'index
HST-5	fraction du texte visible dans la page d'index
HST-6	Taux de compression dans la page d'index
HST-7	top 100 de précision corpus dans la page d'index
HST-11	top 100 de rappel corpus dans la page d'index
HST-15	top 100 de précision requête dans la page d'index
HST-19	top 100 de rappel requête dans la page d'index
HST-23	l'entropie de HP dans la page d'index
HMG25	nombre de mots dans page avec le max page
HMG26	nombre de mots dans le titre dans la page ayant le max pagerank
HMG27	longueur moyenne des mots dans la page ayant le max pagerank
HMG28	fraction du texte d'encre dans la page ayant le max pagerank
HMG29	Taux de compression dans la page ayant le max pagerank
HMG30	fraction du texte d'encre dans la page ayant le max pagerank
HMG31	top 100 de précision corpus dans la page ayant le max pagerank
HMG35	top 100 de rappel corpus dans la page ayant le max pagerank
HMG39	top 100 de précision requête dans la page ayant le max pagerank
HMG43	top 100 de rappel requête dans la page ayant le max pagerank
HMG48	l'entropie de HP dans la page ayant le max pagerank
AVG51	La valeur moyenne des pages dans l'hôte de longueur moyenne des mots
AVG52	La valeur moyenne des pages dans l'hôte de fraction du texte d'encre
AVG53	La valeur moyenne des pages dans l'hôte de fraction du texte visible
AVG54	La valeur moyenne du taux de compression des pages dans l'hôte
AVG55	La valeur moyenne des pages dans l'hôte de top 100 de précision corpus
AVG59	La valeur moyenne des pages dans l'hôte de top 100 de rappel corpus
AVG63	La valeur moyenne des pages dans l'hôte de top 100 de précision requête

AVG67	La valeur moyenne des pages dans l'hôte de top 100 de rappel requête
AVG71	La valeur moyenne des pages dans l'hôte de l'entropie de HP
AVG72	La valeur moyenne des pages dans l'hôte de l'indépendance de LH
STD76	L'écart type des pages dans l'hôte de fraction du texte d'encre
STD77	L'écart type des pages dans l'hôte de fraction du texte visible
STD79	L'écart type des pages dans l'hôte de top 100 de précision corpus
STD87	L'écart type des pages dans l'hôte de top 100 de précision requête
STD95	L'écart type des pages dans l'hôte de l'entropie de HP
STD96	L'écart type des pages dans l'hôte de l'indépendance de LH
Label	Classe de la page

Attributs sélectionnés de WebspamUQAM-2017	
Attributs	**Définition**
Taux mots	Taux des mots clés sans répétition
DT%	Pourcentage des déterminants dans la page d'index
NbCarMeta	Nombre de caractères dans les balises méta Keyword
NbCarTitre	Nombre de caractères dans le titre
Redirection	Valeur binaire. Si la page est redirigée ou pas.
RepetitionMetaDes	Répétition de mots dans les Méta-descriptions
EcartT-Dens-Head	Écart-type de la densité dans le head
NN%	Pourcentage des noms dans la page d'index
SimCosinus	Mesure de ressemblance entre deux vecteurs
EcartT-Dens	Écart type de la densité
Coh_Titre	Cohérence du titre
NbrImg	Nombre d'images dans la page d'index
PR%	Pourcentage des pronoms dans la page d'index
Max-Dens	Densité maximale
Organi	Nombre d'entités de type nom d'organisation
Neutral	Pourcentage des phrases neutres
Max-Dens-Head	Densité maximale dans le head
Http_Statut	Statut http
VB%	Pourcentage des verbes dans la page d'index
Mode-Dens-Head	Mode de densité dans le head
VNegative	Pourcentage des phrases très négatives
Misc	Nombre d'entités de type divers
Median-Dens-Head	Densité médiane dans le head
Negative	Pourcentage des phrases négatives
Locat	Nombre d'entités de type nom de lieu
motUnique	Nombre de mots uniques
NbrMotPage	Nombre de mots dans la page d'index
NbCarContent	Nombre de caractères dans le contenu
Nb-P-Url	Nombre de ponctuations dans l'URL
motCle	Nombre de mots clés

APPENDICE A

L'agrément accepté lors de la demande d'accès aux contenus de WebspamUK-2007

==========

TO: Carlos Castillo

SUBJECT: Request for data collection WebspamUK-2007

I, a person engaging in scientific research, hereby apply to use the HTML contents of the UK-2006 and UK-2007 data set (WebspamUK-2007). In consideration of the provision of this data, I agree to:

1. **SCOPE OF AGREEMENT**

1.1 Use the Data only in accordance with this letter agreement (the Agreement), and to hold the Data in strict confidence.

1.2 You may use the Data only for research purposes in academic and/or commercial institutions. Summaries, analyses and interpretations of the Data may be derived and published provided it is not possible to reconstruct the Data from the publication. Small excerpts of the Data may be displayed to others or published in a scientific or technical context, solely for the purpose of describing your research and related issues.

1.3 You may grant access to the Data only to persons that are working under your supervision and control and have a valid purpose within the scope of this agreement to access the Data. You agree to ensure that such persons comply with the terms and conditions of this Agreement, and to accept responsibility for that compliance.

1.4 The Data has been obtained by crawling the Internet. All the Web pages contained in the Data are documents which have been at some time made publicly available on the Internet, and which have been collected using a process which respects the commonly accepted methods (such as robots.txt) indicating which documents should not be collected.

1.5 Owners of copyright of individual documents contained in the collection may choose to request deletion of these documents from the Data and you agree to promptly comply with such request.

2. **COMMENCEMENT AND DURATION** This Agreement will take effect from the date of signature and will expire two years from this date, unless terminated earlier.

3. **THE DATA**

3.1 The Data will be supplied by providing an unique username/password combination. determines.

3.2 You agree to delete the Data, or any portion thereof, from any media on which it has been stored, if required to do this for legal or regulatory reasons.

3.4 Unless expressly requested no attribution, all publications resulting from research carried out using the Data must provide an attribution. This attribution should preferably appear among the bibliographic citations in the publication, in the following form (edited to fit the citation style used in your publication):

"Web Collection UK-2006/UK-2007". http://chato.cl/webspam/ Crawled by the Laboratory of Web Algorithmics, University of Milan, http://law.di.unimi.it/. URL retrieved MM YYYY

<Fatima AIT MAHAMMED>
<Student>
<UQAM>
<04/04/2016>

BIBLIOGRAPHIE

Adali, S., Liu, T., et Magdon-Ismail, M. (2005). Optimal link bombs are uncoordinated. *In proccedings of the first international workshop on adversarial information retrieval on the web (AIRWeb'05), Japan* .

Altheide, C., et Carvey, H. (2011). Digital Forensics with open source Tools. *On investigating and analyzing computer systems and media using open source tools, Davidson.R (Eds), Elsevier* , pp. 1-8.

Araujo, L., et Martinez-Romo, J. (2010). Web Spam Detection: New Classification Features Based on Qualified Link Analysis and Language Models. *IEEE Transactions on Information Forensics and Security , 5* (3), pp. 581-590.

Ariu, D., Giacinto, G., et Roli, F. (2011). Machine learning in computer forensics (and the lessons learned from machine learning in computer security). *In Proceedings of the 4th ACM workshop on Security and artificial intelligence, AISec '11, ACM*, pp. 99-104.

Ashish, C., Mohammad, S., et Rizwan, B. (2015). Web spam classification using supervised artificial neural network algorithms. *Advanced Computational Intelligence: An International Journal (ACII) , 2* (1), pp. 21-30.

Atashpaz, G. E., et Lucas, C. (2007). Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition. *IEEE Congress on Evolutionary Computation (CEC 2007)* , pp. 4661-4667.

Bahri, E., et Maddouri, M. (2008). Une nouvelle approche du boosting face aux données bruitées. *In EGC , vol. RNTI-E-11* , pp. 349-360.

Becchetti, L., Castillo, C., Donato, D., Ricardo, B.-Y., et Stefano, L. (2008a). Link analysis for web spam detection. *ACM Transactions on the Web (TWEB) 2(1): 2* , pp. 1-42.

Becchetti, L., Castillo, C., Donato, D., Ricardo, B.-Y., et Stefano, L. (2006b). Link-based characterization and detection of Web Spam. *In AIRWeb*, USA, pp. 1–8.

Becchetti, L., Castillo, C., Donato, D., Ricardo, B.-Y., et Stefano, L. (2006a). Using rank propagation and probabilistic counting for link-based spam detection. *In Proceedings of the Workshop on Web Mining and Web Usage Analysis . ACM Press*

Becchetti, L., Castillo, C., Donato, D., Ricardo, B.-Y., et Stefano, L. (2008b). Web spam detection : Link-based and content-based techniques. *The European Integrated Project Dynamically Evolving, Large Scale Information Systems (DELIS), 222*, pp. 99–113.

Beebe, N. (2009). Digital Forensic research : The good, The bad and the unaddressed. In Springer (Ed.), *In Advances in Digital Forensics V : Fifth IFIP WG 11.9 International Conference on Digital Forensics, (Advances in Information and Communication Technology),Gilbert Peterson. 306*, USA: Advencee in digital forensics V, pp. 17-36.

Benczúr, A., Bíró, I., Csalogány, K., et Sarlós, T. (2007). Web spam detection via commercial intent analysis. *Proceedings of the 3rd international workshop on Adversarial information retrieval on the web, ACM*, pp. 89–92.

Benczur, A., Bıro, I., Csalogany, K., et Uher, M. (2006a). Detecting nepotistic links by language model disagreement. *In Proceedings of the 15th international conference on World Wide Web*, Scotland, pp. 939-940.

Benczur, A., Karoly, C., et Tamas, S. (2006b). Link-Based Similarity Search to Fight Web Spam. *In AIRWEB* .

Bernard, J., et Amanda, S. (2003 йил 23-26-june). An Analysis of Web Documents Retrieved and Viewed. *The 4th International conference on internet computing* , pp. 65-69.

Bouguessa, M. (2015a). Évaluation de l'apprentissage. *Cours-DIC9370* .

Bouguessa, M. (2015b). Forage de données. *notes de cours. Université du Québec à Montréal* , 88 pages.

Bouslama, R. (2012). *Vers un systeme d'aide a la décision pour la conception en génie logiciel: Une approche basee sur les connaissances.* UQAM.

Castillo, C. (n.d.). *Webspam-UK2007*. Retrieved 06 25, 2017 from http://chato.cl/: http://chato.cl/webspam/datasets/uk2007/

Castillo, C., Donato, D., Becchetti, L., et Boldi, P. (2006). A Reference Collection for Web Spam. *ACM SIGIR Forum* , pp. 11-24.

Castillo, C., Donato, D., Gionis, A., Murdock, V., et Silvestri, F. (2007). Know your neighbors: Web spam detection using the Web topology. *In Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, Netherlands: *ACM*, pp. 423-430.

Charton, E., Jean-Louis, L., Meurs, M. J., et Gagnon, M. (2013). Trois recettes d'apprentissage automatique pour un système d'extraction d'information et de classification de recettes de cuisine ∗. *TALN-RÉCITAL , Les Sables d'Olonne.*

ChengXiang, Z. (2008). Statistical Language Models for Information Retrieval. *Foundations and Trends in Information Retrieval* , pp. 137-213.

Cohen., W. W. (1995). Fast effective rule induction. *In Proceedings of the Twelfth International Conference on Machine Learning* , pp. 115-123.

Dai, N., Davison, B. D., et Qi, X. (2009). Looking into the past to better classify web spam. *Proceedings of the 5th international workshop on adversarial information retrieval on the web, ACM*, pp. 1-8.

Denning., D. (1987). An intrusion-detection model. *Software Engineering, IEEE Transactions on, SE-13(2)* , pp. 222-232.

Duval, T., Jouga, B., et Roger, L. (2005). XMeta : une approche bayésienne pour le computer forensics. *In French conference on security of information,Symposium on Security of Information and Communication Technologies, SSTIC '05, Feder.E (Eds).* Actes du symposium SSTIC05.

Egele, M., Kolbitsch, C., et Platzer, C. (2011). Removing web spam links from search engine results. *Journal in Computer Virology , 7* (1), pp. 51-62.

El Fouz, I. (2013). *Clustering des News.* Université de nice sophia antipolis.

Erdélyi, M., Garzó, A., et Benczúr, A. (2011). Web Spam Classification: a Few Features Worth More. *Proceeding WebQuality '11 Proceedings of the 2011 Joint WICOW/AIRWeb Workshop on Web Quality*, pp. 27-34.

Faath, É. (2015). *Annotation des opinions dans un corpus de comptes-rendus de lecture*. Retrieved 07 02, 2017 from http://lab.hypotheses.org/1384

Fetterly, D., Manasse, M., et Najork, M. (2004). Spam, damn spam, and statistics: using statistical analysis to locate spam web pages. *In Proceedings of the 7th International Workshop on the Web and Databases: colocated with ACM SIGMOD/PODS*, pp. 1-6.

Fogarty, J., Baker, R. S., et Hudson, S. E. (2005). Case studies in the use of roc curve analysis for sensor-based estimates in human computer interaction. *In Proceedings of Graphics Interface 2005, GI '05* , Canada, pp. 129-136.

Gadhvi, H., et Madhu, S. (2013). Comparative Study of Classification Algorithms for Web Spam Detection. *International Journal of Engineering Research et Technology (IJERT)* , pp. 2497-2501.

Goh, K. L., Patchmuthu, R. K., et Singh, A. K. (2014). Link-based web spam detection using weight properties. *J. Intell. Inf. Syst 43(1)* , pp. 129-145.

Gongwena, X., Xiaomeib, L., Zhijuna, Z., et Li'Naa, X. (2016). Web Spam Detection Based On Link Diversity and Content Features. *International Journal of Security and Its Applications , 10* (7), pp. 363-372.

Gonzalez J., B., et Cristina, A. (2009). Implementacion y evaluacion de un detector masivo de web spam.

Gyongyi, Z., et Garcia-Molina, H. (2004). *Web Spam Taxonomy.* Stanford InfoLab.

Gyongyi, Z., Berkhin, P., Garcia-Molina, H., et Pedersen, J. (2006). Link spam detection based on mass estimation. *VLDB '06 Proceedings of the 32nd international conference on Very large data bases* , pp. 439-450.

Haccoun, A. (2012). *Comparaison de méthodes de classifications.* From https://www.lri.fr/~antoine/Courses/Master-ISI/ISI-10/Projets_2012/Projet_DM.pdf

Hall, A., Eibe, F., Holmes, G., Pfahringer, B., Reutemann, P., et Witten, I. H. (2009). The WEKA data mining software: An update. *SIGKDD Explorations , 11* (1), pp.10-18.

Jacint, I. B., Andras, S., et Benczur, A. (2008). Latent Dirichlet Allocation in Web Spam Filtering. *In Proceedings of the 4 th international workshop on Adversarial information retrieval on the Web* , pp. 29-32.

Jayanthi, S., et Subramani, S. (2010). Link Spam Detection Based on Dbspamclust with Fuzzy c-Means Clustering . *International Journal of Next-Generation Networks, 2(4),* pp.1-10.

Karimpour, J., Noroozi, A., et Adeleh, A. (2012). The Impact of Feature Selection on Web Spam Detection. *I.J. Intelligent Systems and Applications* , pp. 61-67.

Keyhanipour, A. H., et Moshiri, B. (2013). Designing a Web Spam Classifier Based on Feature Fusion in the Layered Multi-Population Genetic Programming

Framework. *In 2013 16th International Conference Information Fusion (FUSION) ,IEEE,* pp.53-60.

Kumar, S., Xiaoying, G., et Ian, W. (2017). Novel Features for Web Spam Detection. *IEEE*, pp. 593-597.

Kwang, L. G., et Ashutosh, K. S. (2015). Comprehensive Literature Review on Machine Learning Structures for Web Spam Classification. *Proceedings of the 4th International Conference on Eco-friendly Computing and Communication Systems. 70*, India: *Elseiver*, pp. 434-441.

Liu, R. et Zhang, M. (2008). Identifying web spam with user behavior analysis. *In Proc. Of 4th Int. Workshop on Adversarial Information Retrieval on the Web (AIRWeb'08).*China , pp. 9-16.

Luckner, M., Gad, M., et Sobkowiak, M. (2014). Stable web spam detection using features based on lexical items. *Comput Security 46* , pp. 79–93.

Mahmoudi, M., Yari, A., et Khadivi, S. (2010). Web spam detection based on discriminative content and link features. *In 5th International Symposium on Telecommunications (IST)*, pp. 542-546.

Martinez-Romo, J., et Araujo, L. (2009). Web Spam Identification through Language Model Analysis. *In Proceedings of the 5 th International Workshop on Adversarial Information Retrieval on the Web* , pp. 21-28.

Mayotte, Y. (2015). *La forêt aléatoire.* From lemakistatheux: https://lemakistatheux.wordpress.com/2015/09/03/les-forets-aleatoires/

Muhammad, I., et Malik, M. A. (2015). Combating against Web Spam through Content Features. *International Journal of Computer Science Issues* , pp. 36-44.

Naffakhi, N. (2004). *Apprentissage supervisé pour la classification des images à l'aide de l'algèbre P-tree.* memoire.

Najork, M. (2009). Web Spam Detection. (M. By L. Liu, Ed.) *Encyclopedia of Database Systems* , pp. 3520-3523.

Namburu, S., Tu, H., Luo, J., et Pattipati, K. (2005). Experiments on Supervised Learning Algorithms for Text Categorization. *Aerospace Conference, IEEE*, pp. 1-8.

Niu, X., et Ma, J. (2010). Learning to Detect Web Spam by Genetic Programming. *Springer-Verlag Berlin Heidelberg* , pp. 18-27.

Ntoulas, A., Najork, M., Manasse, M., et Fetterly, D. (2006). Detecting spam Web pages through content analysis. *In Proceedings of the 15th International Conference on World Wide Web*, pp. 83-92.

Pang, B., et Lee, L. (2008). Opinion Mining and Sentiment Analysis. *Foundations and Trends® in Information Retrieval* , pp. 1-135.

Piskorski, J., Sydow, M., et Weiss, D. (2008). Exploring linguistic features for Webspam detection: a preliminary study. *In Proceedings of the 4th international workshop on Adversarial information retrieval on the Web,* China, pp. 25-28.

Rajendra, K. R., Asthana, S. R., Parikh, M. S., et Dhrevesh. (2016). Detection of spam web page using content and link-based techniques:A combined approach. *Indian Academy of Sciences* , pp. 193-202.

Renato, S., Akebo, Y., et Tiago, A. (2012). An Analysis of Machine Learning Methods for Spam Host Detection. *11th International Conference on Machine Learning and Applications*, USA: *IEEE*, pp. 227-232.

Salton, G., Wong, A., et Yang, C. S. (1975). A vector space model for automatic indexing. *Commun. ACM , 18.*

Shekoofeh, G., et Alireza, P. (2013). Detecting Cloaking Web Spam Using Hash Function. *Computer Science and Information Technology 1(1)* , pp. 33-40.

Shou-Hong, T., Yan, Z., Fan, Y., et Qing, X. (2014). Ascertaining Spam Web Pages Based on Ant Colony Optimization Algorithm. *Springer International Publishing Switzerland* , pp. 231–239.

Shyam, J., Libin, J., John, S., et DevaKumar, S. (2013). Web spam detection using fuzzy clustering. *International Journal on Recent and Innovation Trends in Computing and Communication* , pp. 928-938.

Spirin, N., et Han, J. (2012). survey on Web Spam Detection: Principles and Algorithms. *SIGKDD Explor* , pp. 50-64.

Urvoy, T., Chauveau, E., Filoche, P., et Lavergne, T. (2008). Tracking Web spam with HTML style similarities. *ACM Transactions on the Web , 2* (1).

Victor, M. P., Rafael, L. G., et Fidel, C. (2012). Analysis and Detection of Web Spam by means of Web Content. *Multidisciplinary Information Retrieval. IRFC*, pp. 43-57.

Wei, C., Liu, Y., Zhang, M., Ma, S., Ru, L., et Zhang, K. (2012). Fighting against web spam: a novel propagation method based on click-through data. *In Proceeding of*

the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval , pp. 395-404.

Weiss, S., Indukhya, N., Zhang, T., et Damerau, F. (2005). TEXT MINING: Predictive Methods for Analyzing Unstructured Information. *Springer* .

Westbrook, A., et Greene, R. (2002). Using Semantic Analysis to Classify Search Engine Spam. *Westbrook02usingsemantic* .

Yazid, H., et Lounis, H. (2006). *Les algorithmes d'apprentissage automatique offerts par l'environnement Weka.* From http://www.info2.uqam.ca/~lounis_h/dic938G-hiv2011/documents_weka/algos_weka.pdf

Zhou, B., Jian, P., et Zhaohui, T. (2008). A Spamicity Approach to Web Spam Detection*. *OSD: An Online Web Spam Detection System*, pp. 277-288.