DATA PUBLICA

C-Radar
BUSINESS GROWTH ENGINE

Data Science au service de la Business Intelligence

Raphaël COURIVAUD

ESIEE PARIS
une école de la
CCI PARIS ILE-DE-FRANCE

Rapport De Stage

Raphaël Courivaud

ESIEE Paris
Data Publica

September 28, 2016

Contents

List of Figures

Introduction

Ce stage s'inscrit dans le cadre de ma quatrième année en école d'ingénieur au sein d'ESIEE Paris. J'ai eu la chance de le réaliser dans la startup Data Publica au sein d'une équipe Recherche & Développement dynamique et passionnée. Il s'intègre parfaitement dans ma scolarité et mon projet professionnel. En effet, je suis dans une filière spécialisée dans l'analyse de données, les réseaux, et les objets connectés. C'est parce que je souhaite continuer dans la voie de l'analyse de données que j'ai trouvé ce stage très enrichissant.

Durant ce stage j'ai travaillé sur un projet client avec deux autres personnes, j'ai pu participer à toutes les étapes du projet, du lancement jusqu'à la réunion finale. J'ai également eu l'opportunité de travailler sur des projets plus orientés produit. J'ai pu découvrir plusieurs technologies et consolider les bases que j'avais développées durant ma scolarité et en dehors.

Mon propos est de présenter de façon structurée les connaissances et les technologies utilisées lors des missions qui m'ont été confiées et de façon plus générale l'expérience acquise au sein de Data Publica.

Voici ci-dessous (Figure 1) un emploi du temps détaillé de mes différentes missions.

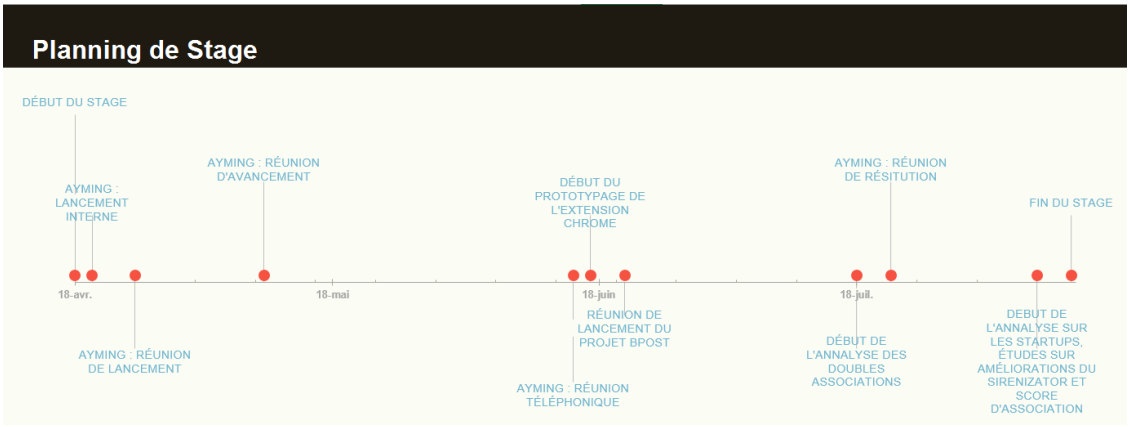


Figure 1: Planning

Dans un premier temps je vais brièvement présenter l'entreprise dans laquelle j'ai travaillé pendant quatre mois, je détaillerai ensuite le produit qu'ils proposent autant d'un point de vue commercial que technique et en expliquant les algorithmes de machine learning qui sont utilisés. Je décrirai ensuite les différentes missions qui m'ont été confiées dans un ordre chronologique et finirai par analyser les apports et les bienfaits de ce stage.

Remerciements

Je tiens tout d'abord à remercier Francois BANCILHON et Christian FRISCH pour m'avoir accueilli dans leur startup. Je remercie Anne VADAINE qui a été mon premier contact chez Data Publica et qui a permis de rendre ce stage possible.

Je remercie aussi Clément CHASTAGNOL pour m'avoir accompagné et aidé tout au long de mon stage en tant que tuteur. Ses conseils ont été précieux et sa disponibilité constante. Il m'a permis de consolider les bases de Data Science et Statistiques que j'avais commencé à acquérir à ESIEE Paris.

Je remercie également toute l'équipe technique de Data Publica, Thomas DUDOUET, Loic PETIT, Vincent YSMAL, Clément DEON et Kamal BENKIRAN pour leur grande disponibilité, leur écoute et leur assistance dans tous les domaines. Toujours disponibles et ouverts, ils m'ont donné accès à leur expertise ce qui m'a permis de progresser rapidement sur mes projets et d'acquérir des connaissance dans des domaines connexes (architecture des applications, Python, Java, JavaScript, systèmes d'information, etc.

Enfin je remercie toute l'équipe commerciale, Emmanuel JOUANNE, Philippe SPENATO, Justine POURRAT, Mathilde CIMIA, Karim SIFOUANE et Thibault DU CLEUZIOU pour leur accompagnement sur mon projet client, et plus généralement pour leur disponibilité et leur accueil.

Au final, tout le monde m'a permis de réaliser un stage très enrichissant au sein d'une équipe motivée, passionnée et compétente dans une ambiance de travail productive et toujours plaisante. Cet environnement startup, un mélange de liberté, de créativité et de motivation, est réellement motivant et épanouissant.

Note au lecteur

Pour un meilleur confort de lecture, il est recommandé de lire ce document dans sa version électronique (http://perso.esiee.fr/ courivar/rapportdatapublica.pdf) pour profiter des liens hypertexte qui ont été insérés.

Attestation sur l'honneur

Je soussigné, Raphaël Yoann Courivaud, certifie sur l'honneur que les travaux soumis en mon nom dans ce rapport sont le fruit de mes propres efforts et réflexions personnelles. Toute idée ou tout document utilisé pour étayer ce travail et ne constituant pas une réflexion personnelle est en conséquence cité en référence et signalé dans l'endroit précis de son utilisation.

1 Description de l'entreprise

L'évolution des technologies et leurs usages ont fait exploser la quantité de données produites. Selon IBM, 2.5 milliard de gigabytes (GB) de données ont été générées chaque jour de l'année 2012.
De plus, cette quantité de données double tous les deux ans. L'exploration ou la fouille de données (« data mining ») consiste à en extraire ou à en déduire des informations utiles. Ceci peut s'avérer non seulement très fructueux, mais devenir stratégique au vu de la quantité de données produites.

La question principale qui se pose est de savoir comment utiliser intelligemment cette immense masse de données pour en tirer une plus-value ? C'est le rôle des entreprises spécialisées dans l'exploitation de ces données dont Data Publica est un pionnier.

1.1 Histoire

Data Publica est un des précurseurs de l'Open Data en France. Cette société, qui a bénéficié d'investissements technologiques faits en 2010 dans le cadre d'un projet de R&D, a été financée initialement par un groupe de business angels et le fonds d'amorçage IT Translation.
Data Publica est une start-up spécialisée dans les données entreprises, l'Open Data, le Big Data et la Dataviz. C'est une société relativement jeune, axée R&D. Alimenté par une équipe très dynamique et compétente, Data Publica est dans la recherche constante du dépassement technique.

Historiquement, Data Publica ne faisait que de l'Open Data. C'est-à-dire que la société se servait de données accessibles à tous (provenant d'institutions gouvernementales notamment) pour créer des jeux de données sur mesure pour des entreprises. Un autre pan important de l'activité initiale de Data Publica a été de mettre en place le premier annuaire Open Data français, qui a ensuite été supplanté par data.gouv.fr. Ainsi, la société s'est spécialisée dans l'identification des sources de données, leur extraction et leur transformation en données structurées.

Depuis quelques années, Data Publica se spécialise dans les données sur les entreprises françaises en abandonnant progressivement son activité Open Data. En réalité, l'apparition de data.gouv.fr a très rapidement rendu son activité assez obsolète. Aujourd'hui, les services qu'elle propose ne sont plus tout-à-fait les mêmes. En effet, Data Publica réutilise les données Open Data concernant les entreprises françaises dans son produit phare : C-Radar. Ce produit est lui-même conçu pour les entreprises du B2B[1]. Le produit est décrit plus en détail dans la partie 2.

Data Publica participe également à de nombreux projets de recherche français et européens tels que XDATA, Diachron ou Poqemon, en partenariat avec l'INRIA, et scanR avec le ministère de l'enseignement supérieur et de la recherche.

L'application scanR est un moteur de recherche basé sur la technologie C-Radar. Il permet de rechercher les publications rattachées à des laboratoires de recherches qui ont potentiellement collaboré avec des entités privées ou publiques. Cela permet de visualiser un écosystème autour des projets de recherche afin de mieux les comprendre. Les sites des laboratoires sont crawlés tous les mois afin de récupérer les publications ainsi que les thèses.

[1] Business to Business : Définit toutes les activités des entreprises visant d'autres entreprises

1.2 Contexte Concurrentiel

La concurrence dans le domaine de l'Open Data en France comme à l'international n'est pas encore très développée, le mouvement étant encore jeune. Mais son côté prometteur pousse des entreprises à investir ce marché. Une dizaine de concurrents ayant un discours proche de celui de C-Radar existent à ce jour dont ZEBAZ, MixData, Corporama, Orb Intelligence.

D'autres existent mais ne proposent pas autant de services que C-Radar et sont simplement des annuaires d'entreprises. Nous pouvons citer Europages ou encore Creditsafe.

1.3 Clients

Data Publica réalise des projets d'analyse, de croisement et d'exploitation de données pour des entreprises travaillant dans de nombreux domaines. Plus de 70 entreprises et organismes ont fait appel à Data Publica depuis sa création.

Il faut aussi inclure parmi ces clients toutes les entreprises ayant souscrit à C-Radar. C-Radar se décompose en deux offres différentes, une destinée aux services commerciaux qui permet d'accéder aux recherches ainsi qu'aux fiches entreprises détaillées et une autre plus orientée marketing qui permet de réaliser des segmentations de marché par similarité sémantique et d'en déduire des prospects qualifiés. C-Radar compte aujourd'hui plus de 2950 inscrits.
Data-Publica compte entre autres parmi ses clients :

- des entreprises publicitaires et média telles que Medialex ;
- des banques et compagnies d'assurance telles que la Banque Publique d'Investissement et Euler-Hermès ;
- des industriels tels que Renault, Bouygues Telecom, Santé Clair, ERDF ou encore la Poste belge ;
- des entreprises d'e-commerce, telle que The French Talents.

1.4 L'équipe

Data Publica emploie quatorze personnes réparties également en deux équipes : une équipe Marketing/Commerciale (sept personnes) et une équipe Technique (sept développeurs). Les deux équipes travaillent chacune dans son open-space. Pendant mon stage, j'ai été immergé au sein de l'équipe Technique.

L'équipe Technique est composée de sept développeurs (ordonnés par ancienneté) :

- Christian FRISCH, directeur technique ;
- Thomas DUDOUET, développeur Back-end Java ;
- Loïc PETIT, développeur Back-end Java, Architecture ;
- Clément CHASTAGNOL, data scientist Python et mon maître de stage ;
- Clément DEON, développeur Front-end ;
- Vincent YSMAL, développeur Back-end Java, Architecture ;
- Kamal BENKIRAN, data scientist Python.

L'équipe Marketing et Commerciale est composée de sept commerciaux et responsables marketing (ordonnés par ancienneté) :

- François BANCILHON, directeur général ;
- Emmanuel JOUANNE, business Developpment Manager ;
- Philippe SPENATO ingénieur d'affaire ;
- Justine POURRAT, responsable Communication et Marketing ;
- Mathilde CIMIA, responsable Commercial ;
- Thibault DU CLEUZIOU, directeur Marketing ;
- Karim SIFOUANE, responsable Commercial.

2 C-Radar

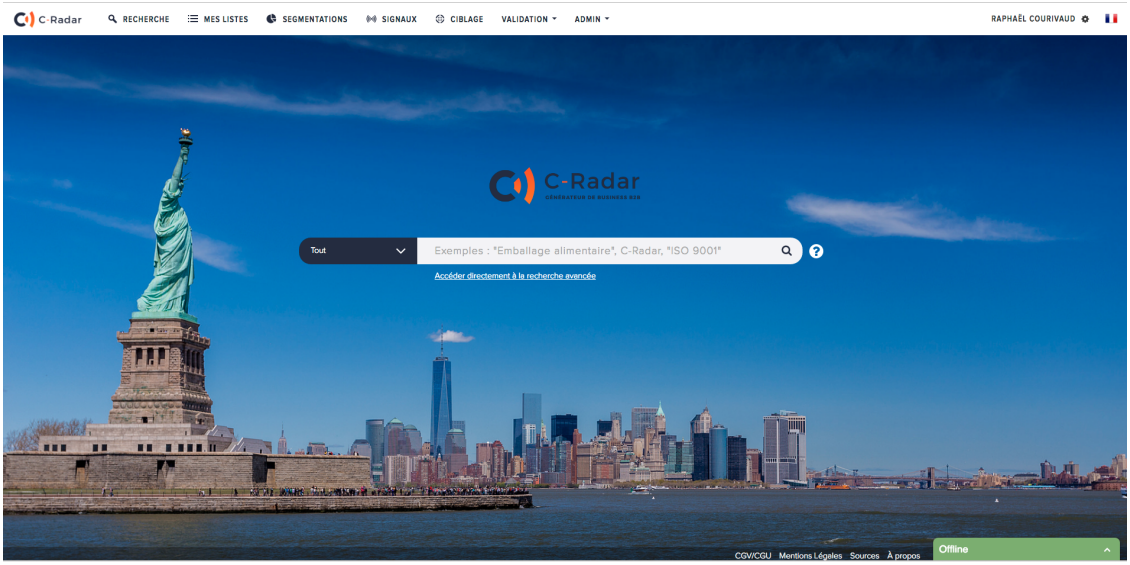


Figure 2: Page d'accueil de C-Radar

2.1 Présentation Commerciale

Ce produit est un moteur de recherche B2B. Celui-ci a pour objectif de permettre aux services de ventes et marketing des entreprises B2B de vendre plus et mieux en qualifiant leurs prospects et en ayant accès à des bases de données riches et à jour sur les entreprises.

Ce moteur de recherche, appelé C-Radar, est un produit de vente prédictive construit sur une base de référence des entreprises françaises. Il regroupe beaucoup d'informations de différents types, notamment administratives, financières, toutes celles qui découlent de leur communication sur les réseaux sociaux et de leur site web.

C-Radar est un concentré de technologies Big Data. En effet, il utilise diverses technologies comme le crawling, le scraping ou encore le machine learning. Ceci afin d'offrir à l'utilisateur diverses fonctionnalités : moteur de recherche d'entreprises, fiche d'activité d'entreprises avec contacts commerciaux, détection de nouveaux prospects, scoring de prospects existants, segmentation automatique d'entreprises, identification de marché.

2.2 L'application

L'application permet de présenter des informations concernant les entreprises aux utilisateurs, c'est un reporting des données produites par le Workflow (Cf. partie 2.4.3). Elle permet de faire des recherches, d'observer la répartition géographique, de créer des listes, etc.
Les utilisateurs ont aussi accès à plusieurs outils permettant de faciliter leur prospection. Ces outils mettent en place différents algorithmes de machine learning utilisant la sémantique des sites web des entreprises.

Figure 3: Fiche entreprise C-Radar

Elle permet d'avoir des données consises sur une même entreprise, rassemblées au même endroit (sur une fiche, présentée sur la figure 3). Ces données présentent une forte valeur ajoutée pour nos clients. La section "RÉSEAUX SOCIAUX" regroupe tous les comptes sociaux détectés sur le site web de ainsi que les posts provenant du compte Facebook, Twitter et flux RSS de l'entreprise. Un historique du nombre de like Facebook a aussi été mis en place qui permet d'avoir un aperçu de l'activité du compte. Cela permet de d'afficher ces données à l'utilisateur sous la forme d'un graphique (Figure 4).

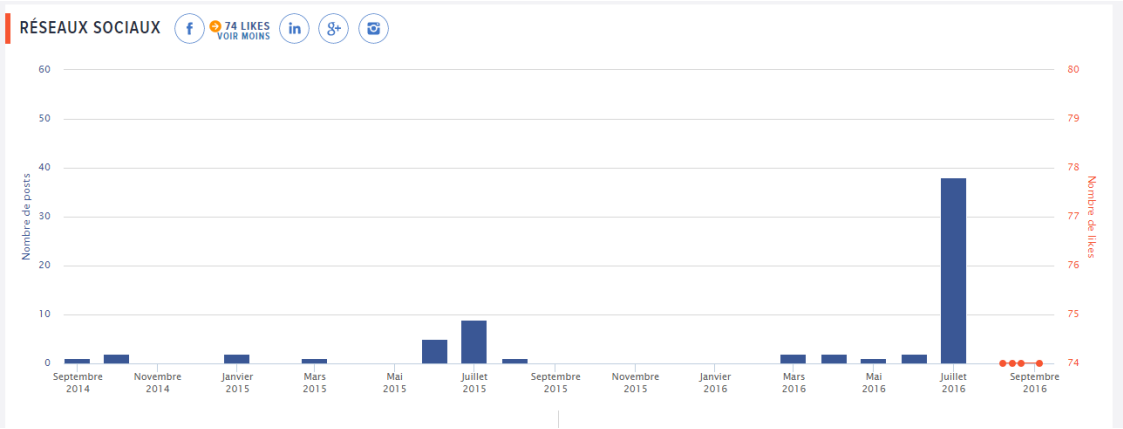


Figure 4: Affichage de l'activité du compte Facebook de l'entreprise

Le moteur de recherche avancé permet de réaliser des requêtes sur les mots clés présents sur les sites des entreprises. De nombreux filtres ont été mis en place sur le chiffre d'affaire, le code NAF (ou APE) ou encore sur la présence ou non sur les réseaux sociaux. On peut voir un exemple de recherche sur la figure 5.

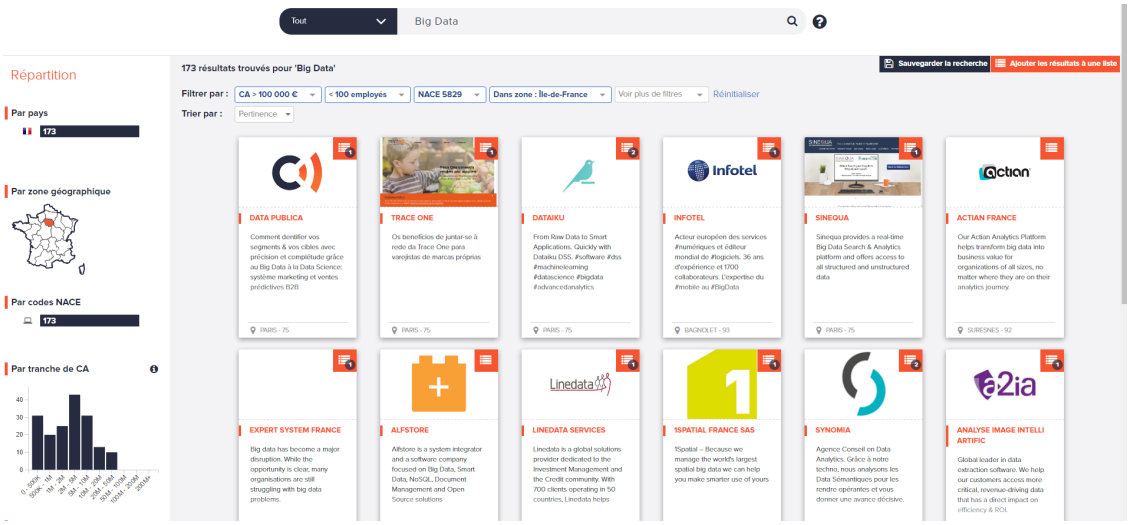


Figure 5: Interface de recherche de C-Radar

Elle permet aussi de créer des listes d'entreprises (Figure 6) pouvant regrouper le résultat d'une recherche. Il est aussi possible d'importer directement son portefeuille de clients pour y effectuer les analyses.

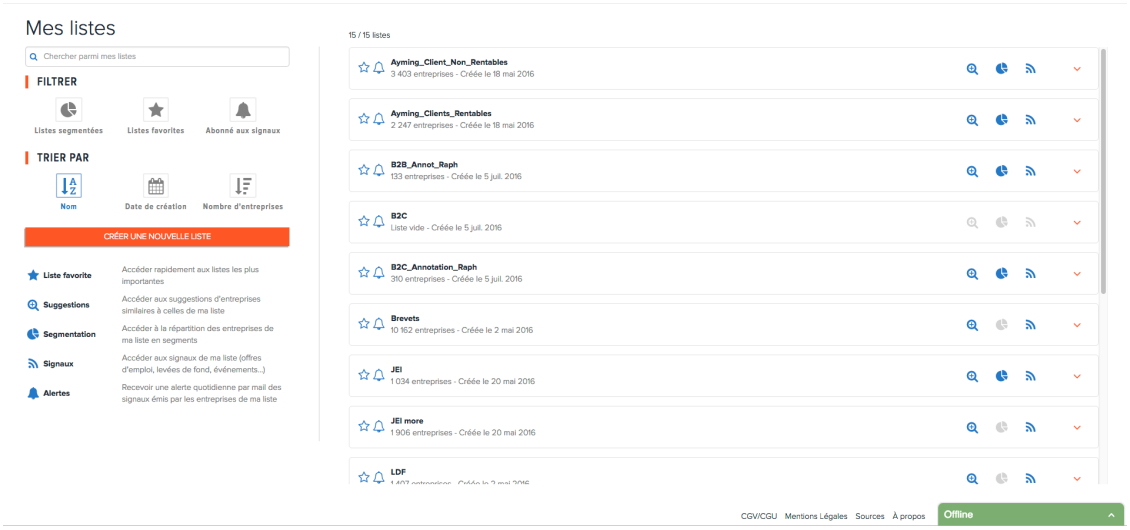


Figure 6: Affichage des listes dans C-Radar

2.3 Présentation Technique

Pour répondre aux problématiques auxquelles Data Publica se confronte, la société a acquis 4 expertises majeures :

- le web crawling / web scraping : la récupération des données ;
- le data mining / text mining : l'analyse, l'extraction et l'enrichissement des données ;
- le machine learning : l'apprentissage automatique à partir de données structurées ;
- la dataviz : la visualisation des données.

2.3.1 Le crawling

Le crawling est l'action réalisée par un script, appelé web crawler, qui va récupérer toutes les informations présentes sur un site web. Il permet de récupérer les différentes pages et d'en extraire le texte brut et ainsi, de récupérer toutes les données présentes de façon non structurée. Pour cela il part de la racine du site, parcourt la page et extrait les liens à partir des attributs href des balises <a> (HTML). Toutes les pages du site sont parcourues de façon récursive, en fixant une valeur limite (100 pages) pour limiter la taille du crawl. Une limite en profondeur d'exploration existe également, fixée à trois liens suivis par page. Les liens pointant vers des fichiers ne sont pas traités pour des question de stockage et d'optimisation.

2.3.2 Le scraping

Le scraping est l'action réalisée par un script permettant d'extraire de l'information structurée d'un site web. Contrairement au crawling, il est question d'extraire des données précises, et non pas la totalité des données disponibles sur le site. Le site « scrapé » et sa structure doivent donc être connus et analysés à l'avance afin d'adapter le scraper. Ce processus de scraping est donc manuel et assez long à mettre en oeuvre. Cette technique n'est pas généralisable contrairement au crawling. Au sein de Data Publica, de nombreux scrapers ont été développés spécifiquement pour des projets clients (récupérations des entreprises adhérentes d'un pôle de compétitivité par exemple).

Par exemple, si on veut scraper le site du FBI et récupérer le contenu du titre de la page, il faut effectuer le code suivant :

```
from bs4 import BeautifulSoup
import requests

url = "https://www.fbi.gov/"
r = requests.get(url)
soup = BeautifulSoup(r.text.replace("\n", ""), "lxml", from_encoding="utf-8")
soup.find(class_="main-header").find(class_="brand").text.strip()

Out[3]: 'Federal Bureau of Investigation'
```

2.3.3 Le data mining / text mining

Une fois les sites web crawlés et scrapés, des processus d'extraction d'informations peuvent être appliqués. Ils permettent de collecter des données variées et de grande valeur.
Cette méthode permet de récupérer les comptes des réseaux sociaux, les numéros de téléphone, les adresses, les numéros SIREN, les certifications, etc.
Ces processus utilisent essentiellement des expressions rationnelles[2].

2.3.4 Le machine learning

Le machine learning recouvre les techniques d'apprentissage automatique, c'est une sous-partie de l'intelligence artificielle. Cela revient à la conception, l'analyse et la mise en place d'algorithmes permettant de faire apprendre à une machine la prise d'une décision automatique.
L'algorithme nécessite un ensemble d'observations assez conséquent pour apprendre cette prise de décision, il permet ensuite de prédire la catégorie ou la valeur en fonction de nouvelles données d'entrée. Il existe de nombreuses applications dans des domaines très divers.
Plusieurs algorithmes sont utilisés dans C-Radar, ils sont décrits plus précisément dans la partie 2.5.

2.3.5 La Dataviz

La dataviz est un raccourci pour parler de data visualisation. C'est la dernière étape du cycle de vie de la donnée et elle est loin d'être la moins importante. Elle permet de présenter les données de manière visuelle, interprétable et compréhensible, sous forme de graphiques ou de représentations plus attractives que de simples tableaux de chiffres.
Les figures 7 et 8 sont des exemples de data visualisation plus ou moins avancés. La première provient de la fiche C-Radar. La deuxième est extraite d'un article publié par John Nelson (anciennement été directeur de la visualization chez IDV Solutions) sur LinkedIn. Il explique comment réaliser une HeatMap[3] directement depuis Excel.

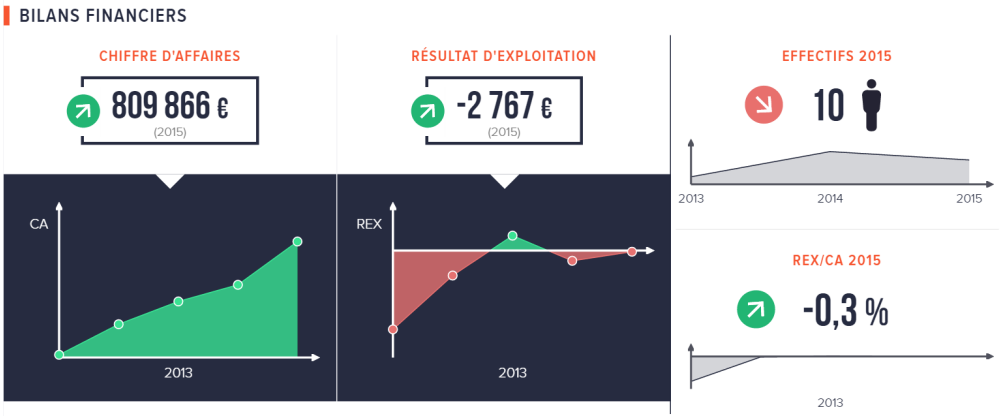


Figure 7: Présentation financière d'une entreprise, extraite de la fiche C-Radar

[2]Motif décrivant un ensemble de chaînes de caractères possibles selon une syntaxe précise.
[3]Graphique où les données sont représentées avec différentes teintes de couleurs

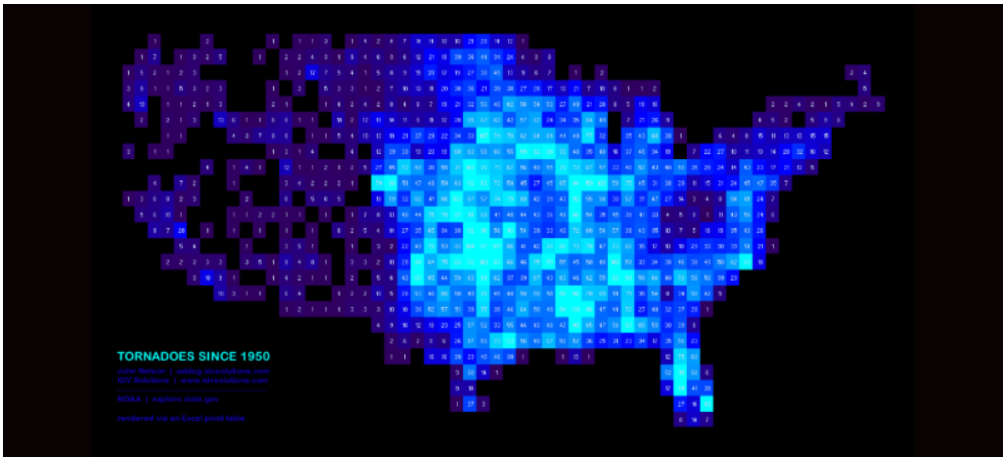


Figure 8: Répartition des appararition des Tornades aux Etats Unis depuis 1950 (source)

Un des tous premiers exemples de dataviz concerne les pertes napoléoniennes lors de la campagne de Russie. Cette carte représente 5 variables de façon extrémement concise.

Les technologies actuelles permettent de traiter de gros volumes de données, d'ajouter de l'interactivité, etc.

Ainsi, cela permet de comprendre plus facilement mais aussi d'analyser ou d'extraire plus rapidement les informations importantes provenant de données potentiellement très complexes. C'est une partie très importante, qui permet souvent d'expliquer et de justifier le travail du data scientist.

2.4 L'architecture

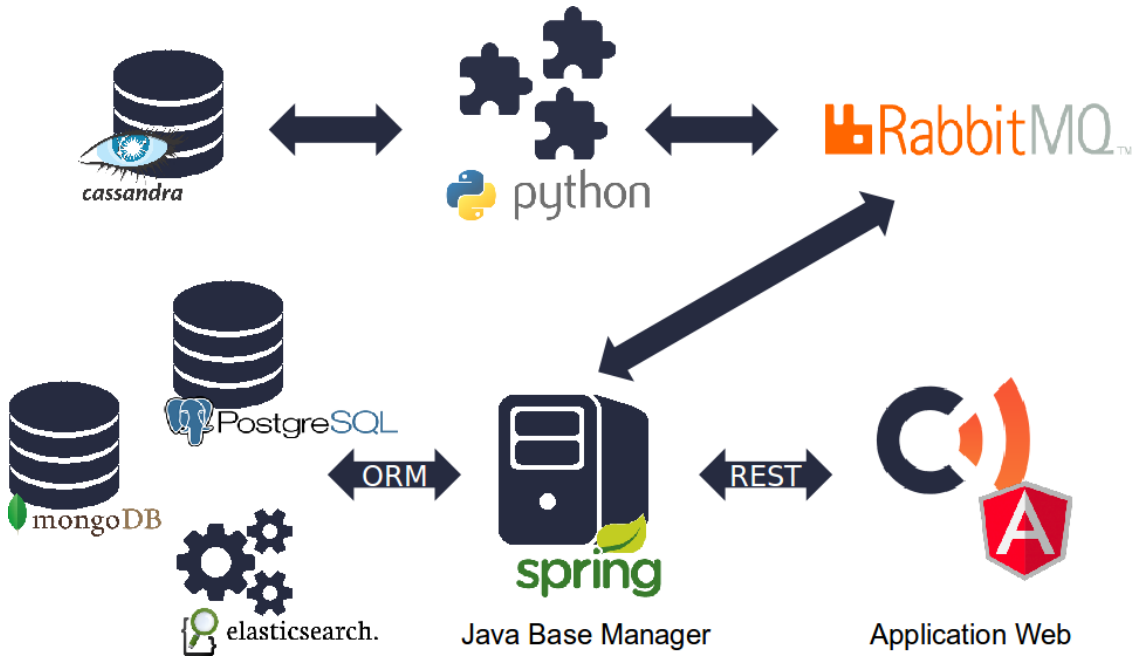


Figure 9: Schéma architecture C-Radar

L'architecture de C-Radar présentée sur la figure 9 peut être divisée en plusieurs parties :

- différentes bases de données sont utilisées pour stocker des données de natures diverses (par exemple, PostgreSQL pour les données fortement relationnelles, MongoDB pour des données très structurées et peu adaptées à un modèle relationnel, ou encore Cassandra pour les données volumineuses...) ;
- un moteur de recherche sémantique : Elastic Search ;
- un gestionnaire de files d'attentes et de messagerie : RabbitMQ ;
- différents plugins Python s'interfaçant avec le gestionnaire de fichier RabbitMQ ;
- un gestionnaire de flux entre les différents composants précédents, le Workflow ;
- une application Java s'interfaçant avec Elastic Search et les bases de données Mongo et PostgreSQL

2.4.1 Stockage des données

Une des parties les plus importantes de l'infrastructure est le stockage, la persistance et la mise à jours des données existantes. Pour cela, trois grands types de bases de données sont utilisées au sein de l'infrastructure de l'application. Ces trois différents types de stockages ont leurs caractéristiques propres qui leur donne leur intérêt.

- **PostGreSQL** : est une base de données relationnelles. Elle est utilisée pour stocker les données géographiques ainsi que les listes des entreprises créées par les utilisateurs ;
- **MongoDB** : est une base de données NoSql[4]. Elle permet de stocker un grand volume de données totalement hétérogènes. Ces dernières doivent être rassemblées sous forme de "modèle" de données. Ils permettent de représenter au mieux les différents types de données afin de faciliter leur utilisation ;
- **Cassandra** : est une base de données distribuée qui permet de passer à l'échelle très facilement. Comme MongoDB c'est une base de données non-relationnelle. Elle est entre autres utilisée pour stocker les crawls des sites.

2.4.2 Le Java Base Manager

Le JBM est le projet Java qui rassemble tout le Workflow ainsi que l'application web. Ce projet a été conçu et construit sur la base de Spring. Le framework Spring est une plate-forme Java JEE qui fournit une architecture complète permettant de développer des applications web. Spring gère l'architecture de l'application, cela permet de déclarer toutes les dépendances ou de mettre en place les points d'API[5] par exemple mais c'est aussi ce qui permet de prendre en charge le modèle MVC[6], la sécurité de l'application, l'interface avec les gestionnaire de fichiers, etc.

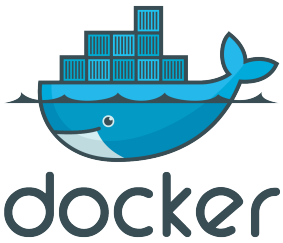
2.4.3 Le Workflow

Le Workflow est le gestionnaire de flux permettant de lancer les différents processus de récupération et d'analyse de données. Il gère tout ce qui est « computing ». C'est lui qui lance des tâches en les écrivant dans RabbitMQ. Ensuite les plugins Python écoutent les queues RabbitMQ pour obtenir de nouvelles tâches à effectuer. Une fois les sites webs crawlés, le plugin Python va les stocker dans Cassandra sans passer par le Workflow mais en lui indiquant que le crawl est terminé (c'est la seule fois où un plugin Python écrira directement dans une base).

Le workflow gère aussi tout ce qui est exécution des plugins Python du crawling au scraping en passant par la capture et catégorisation des signaux. Il gère aussi le stockage des données produites à l'issue de ces exécutions ainsi que l'indexation dans Elastic Search. Pour résumer, il s'occupe de l'orchestration de tous les processus permettant de passer d'un SIREN à une fiche d'entreprise complète et attractive à l'utilisateur. Ce numéro de SIREN permet d'identifier chaque entreprise distinctement. Il est utilisé par tous les organismes publics et les administrations en relation avec l'entreprise, il est attribué par l'INSEE lorsque l'entreprise est inscrite dans le répertoire Sirene, il est constitué de 9 chiffres.

[4]Not Only SQL : n'étant pas fondées sur l'architecture classique des bases de données relationnelles.
[5]Application programming interface, interface de programmation applicative en francais
[6]Modèle Vue Controleur

2.4.4 Docker



Depuis peu, toute l'infrastructure de C-Radar est passée sous Docker. Que ce soit les plugins Python, en passant par l'application, tous les serveurs utilisés se basent sur une architecture Docker. Pour des raisons financières, une infrastructure basique est de segmenter une machine physique en plusieurs serveurs. En effet, il est plus intéressant financièrement d'utiliser une seule grosse machine que plusieurs petites.

L'architecture classique est de monter sur l'OS de la machine physique un `hypervisor` qui gère plusieurs `virtual machine` et plusieurs OS hôtes. Cela permet de mettre en place plusieurs OS différents sur une même machine physique.

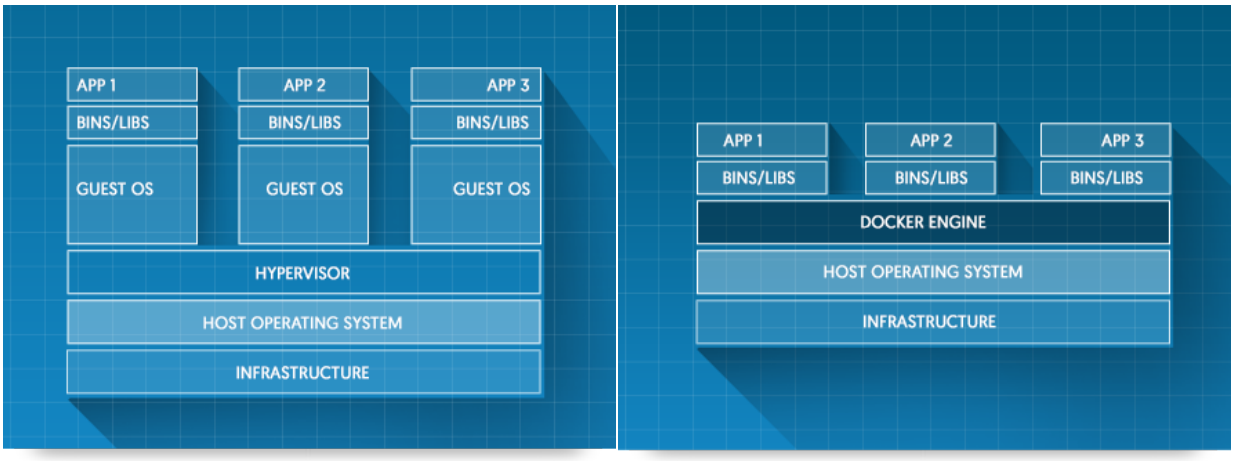


Figure 10: Comparaison de l'architecture classique/Docker

Un des avantages de Docker est de simplifier cette mise en place, puisque le Docker engine permet de segmenter une même machine physique en utilisant le même kernel.
Le principal avantage est d'isoler toutes les dépendances des conteneurs. On peut facilement isoler le système de fichier, le réseau, les devices, le CPU, la mémoire, les débits d'entrées et sorties (IO).

Tous les conteneurs ne peuvent pas avoir accès aux fichiers des autres et ils ne peuvent pas communiquer entre eux par défaut. Cela permet de laisser une machine propre de toutes les installations gênantes. Lorsqu'on supprime un conteneur, il n'en reste vraiment aucune trace. Toutes les librairies ainsi que tous les fichiers sont supprimés.

Pour des raisons d'architecture on peut quand même définir des liens entre deux conteneurs, ces liens sont exclusifs, ils ne marchent que dans un sens. On peut aussi faire un lien entre un dossier local et un dossier du conteneur pour pouvoir rendre persistantes les données créées.

L'importante communauté autour de Docker permet d'avoir des images dans beaucoup de domaines d'applications. Elles sont disponibles sur le DockerHub, un dépôt regroupant des images docker publiques ou privées. Il permet de télécharger des images qui instancient un conteneur avec son environnement et toutes ses dépendances.

Il existe par exemple des images pour instancier un environnement avec notebook Jupyter, une distribution Debian, ou encore avec une base de données Mongo.

2.5 Data Science

Nous allons maintenant faire un tour d'horizon des différents algorithmes utilisés dans tous les outils mis à disposition de l'utilisateur de C-Radar.

2.5.1 Text Mining

La plus grosse partie des analyses de données est faite à partir du crawl des sites web et donc de données textuelles. Ces données n'étant pas directement numériques, on doit effectuer un changement de représentation par exemple avec le modèle Bag of Words[7]. L'opération s'appelle alors la vectorisation, cette technique permet de projeter un corpus sur un vecteur représentant les différents mots du vocabulaire. On a alors pour chaque échantillon une correspondance entre le mot et son nombre d'occurrence.

Différentes étapes :
Pour obtenir des données "propres", cohérentes et pas trop importantes, sur des données textuelles, il faut réaliser plusieurs étapes importantes. La première étape très importante est de normaliser le texte cette étape permet de retirer tous les caractères qui produisent du bruit.

```
import re
text = """<p><b>Text mining</b>, also referred to as <i>text
<a href="/wiki/Data_mining" title="Data mining">data mining</a></i>,
roughly equivalent to <b><a href="#Text_mining_and_text_analytics">
text analytics</a></b>, refers to the process of deriving high-quality
<a href="/wiki/Information" title="Information">information</a> from
<a href="/wiki/Plain_text" title="Plain text">text</a>. High-quality
information is typically derived through the devising of patterns and
trends through means such as <a href="/wiki/Pattern_recognition"
title="Pattern recognition">statistical pattern learning</a>."""

text = re.sub('[\.,;!\?|&\(\)\'\\\\<>:/"’\+\*\}\{«»\[\]=#·“”]+', ' ', text)
text = re.sub('[\s]+', ' ', text).replace(" href ", " ")
text = re.sub(' [a-z] ', ' ', text).replace(" href ", " ")
text
```

```
Out[17]: ' b Text mining also referred to as text wiki Data_mining title
Data mining data mining i roughly equivalent to a
Text_mining_and_text_analytics text analytics b refers to the process
of deriving high-quality wiki Information title Information information
from wiki Plain_text title Plain text text High-quality information is
typically derived through the devising of patterns and trends through
means such as wiki Pattern_recognition title Pattern recognition statistical
pattern learning '
```

La deuxième étape est de supprimer les mots sans grand intérêt pour le sens du texte. Ces mots sont appelés des `StopWords` ils sont là pour lier les phrases mais n'apportent pas d'intérêt particulier. Cela permet de retirer une partie du vocabulaire, ce qui n'est pas de trop vu la quantité

[7] Représentation d'un texte sous la forme d'une liste de mots

de mots différents créant ce vocabulaire.
Ces `StopWords` sont disponibles sous forme de listes différentes selon les langues. La langue du site web est automatiquement détectée ce qui permet d'affiner les analyses pour les différents cas. L'exemple ci-dessous permet d'extraire aléatoirement 10 mots des stopwords francais. La listes est constituée de 155 mots, comprenant les différentes formes des pronoms et les conjugaisons des deux verbes auxiliaires être et avoir.

```
from nltk.corpus import stopwords
import random
[stopwords.words('french')[i] for i in
            sorted(random.sample(range(len(stopwords.words('french'))), 10)) ]
```

```
['aux', 'la', 'moi', 'mon', 'toi', 'une', 'c', 'fusses', 'avaient', 'eût']
```

Voici toutes les langues disponibles et implémentées dans l'application :

```
Languages  = {
"DA" : 'danish' , "DE" : 'german' , "EN" : 'english' , "ES" : 'spanish' ,
"FI" : 'finnish' , "FR" : 'french' , "HU" : 'hungarian' , "IT" : 'italian' , "NL"
"NO" : 'norwegian' , "PT" : 'portuguese' , "RU" : 'russian' , "SV" : 'swedish'
}
```

Ensuite afin de réduire la dimensionnalité du vocabulaire on "stemme" tous les mots du vocabulaire. Le stemming permet de ne garder que la racine d'un mot, permettant donc de regrouper tous les mots ayant la même signification ou au moins le même "sens" sans perdre trop d'information.
Il existe une autre méthode assez proche mais plus intelligente qui est la Lemmatization. De nombreux algorithmes sont disponibles pour "Lemmatizer" des corpus en langue anglaise, mais ils sont très pauvres en ce qui concerne la langue française.

Enfin, nous pouvons réaliser la transformation des échantillons et créer la matrice d'occurrence. Cette matrice est créée à partir de chaque mot du vocabulaire. Le crawl du site web est projeté sur le vecteur du vocabulaire. Et une normalisation est effectuée permettant de mettre en évidence la rareté d'un mot par rapport à sa présence dans le corpus total. On voit ici la décomposition du calcul de cette normalisation du terme i dans le document j.

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} \tag{1}$$

$$idf_i = \log \frac{|D|}{|d : t_i \in d|} \tag{2}$$

où :

- $|D|$: nombre total de documents dans le corpus ;
- $|\{d_j : t_i \in d_j\}|$: nombre de documents où le terme t_i apparaît (c'est-à-dire $n_{i,j} \neq 0$).

source

On a finalement $mathrmtfidf_{i,j} = \mathrm{tf_{i,j}} \cdot \mathrm{idf_i}$ avec TFIDF qui signifie Term Frequency-Inverse Document Frequency.

Cela donne finalement une très grande matrice comportant N nombre de lignes et M nombre de colonnes, N étant le nombre de site web présent sur la base et M le vocabulaire total de la langue française et anglaise formé par le corpus global. La base C-Radar France comporte les données administratives et juridiques de plus de 4,5 millions d'entreprises. Parmi toutes ces entreprises seules 450 000 sont rattachées à un nom de domaine. Le site web correspondant peut être alors crawlé. La matrice fait donc N=450 000 lignes sur M=1,2 million de colonnes.

Même si le type de matrice est optimisé avec une matrice creuse (CSR_matrix) qui est une matrice de lignes compressées, cela reste quand même très volumineux et lourd à monter en mémoire puisqu'elle fait plus de 4 Giga-octets sous sa forme compressée et environ 40 Téraoctets sous sa forme dense. Cette instruction correspond à la définition d'une matrice compréssée par lignes.

```
csr_matrix((data, (row_ind, col_ind)), [shape=(M, N)])
        where data, row_ind and col_ind satisfy the relationship :
            [row_ind[k], col_ind[k]] = data[k].
```

2.5.2 Big Mama

Description :

Big Mama (pour Big Matrix Manager) est le nom donné au plugin permettant de créer et de gérer la matrice sémantique des sites français. Cette matrice est utilisée par de nombreux plugins Python qui permettent de classifier et segmenter des entreprises grâce au texte présent sur leur site web. Elle peut être mise à jour au fil de l'eau reflétant donc à tout instant l'état actuel de la base.
Elle est stockée en mémoire sous une forme compressée sur la mémoire physique du serveur. Cette représentation persistante de la base de crawls vient remplacer une version éphémère, dans laquelle une matrice était construite spécifiquement pour chaque tâche de segmentation (Cf. partie 2.5.3), en incluant uniquement les données des entreprises concernées. Cette première version avait l'avantage d'être simple et d'avoir une faible empreinte mémoire, mais les temps de traitement pouvaient être relativement longs.

En effet, lorsque les plugins sont instanciés ils montent en mémoire la matrice ce qui prend quelques secondes, comparé à plusieurs minutes voire dizaines de minutes dans la première version. Les calculs sur la sémantique (s'appuyant sur la bibliothèque Scikit-learn constituée de wrappers Python autour de briques de code en C ou FORTRAN) sont considérés comme suffisamment rapides (quelques secondes) pour ne pas chercher à les optimiser. L'expérience utilisateur est donc nettement améliorée.

Cependant cela engendre de nouveaux problèmes. Cette matrice est énorme comme nous l'avons vu dans la partie 2.5.1 et elle n'est mise en place que pour quelques pays individuels (France, Royaume-Uni et Belgique). La génération de cette matrice pour la version mondiale de C-Radar n'est pas viable et la matrice ne tient pas en mémoire puisqu'il y a un peu plus de 5 millions de sites web analysés dans le monde entier et un vocabulaire tout aussi grand.

Amélioration :

Aujourd'hui, on arrive aux limites de l'infrastructure existante, la version mondiale étant lancée et réclamée par de nombreux client, il y a un gros travail à faire de mise en place d'une nouvelle infrastructure pour pouvoir mettre à l'échelle.

Il faudrait repenser toute l'infrastructure en l'axant sur les technologies Big Data existantes comme Apache Spark ou Hadoop qui permettent une approche Map/Reduce du problème. Cette

approche permet de décentraliser le volume de données et de paralléliser les calculs. Les accès peuvent être fait depuis la mémoire physique (Hadoop) ou depuis la mémoire vive (Spark) ce qui diminue d'un facteur 100 les temps de calculs.

2.5.3 Segmentation et génération de prospects

La segmentation permet de regrouper les entreprises d'une liste dans des clusters cohérents selon la sémantique extraite des sites web. Chaque cluster est nommé par le mot le plus pertinent extrait des sites des entreprises concernées.

La matrice possède un nombre très important de variables ce qui peut être problématique pour le modèle. Il est donc intéressant de réduire la dimensionnalité avec une LSA (Latent Semantic Analysis) ou SVD (Singular Value Decomposition) qui permet de regrouper des groupes de features entre elles. Cette méthode est intéressante puisqu'elle opère directement sur les vecteurs et peut donc être appliquée à une matrice compressée.

De plus, on réalise un Spectral Embedding qui permet de déterminer une représentation à faible dimension des variables de notre ensemble de données. Ces deux opérations poursuive le même objectif de réduction de dimension. Il faudrait donc faire des tests pour mesurer l'apport de ces deux opérations successives.

Ensuite un apprentissage non-supervisé est utilisé pour créer un nombre de clusters défini au préalable par l'utilisateur. Dans ce cas on utilise l'algorithme des K-means qui permet avec un calcul de distance assez simple de regrouper les échantillons en clusters cohérents. On peut voir ci-dessous un résultat d'un segmentation sur une liste provenant d'une recherche sur les jeunes entreprises innovantes (Figure 11).

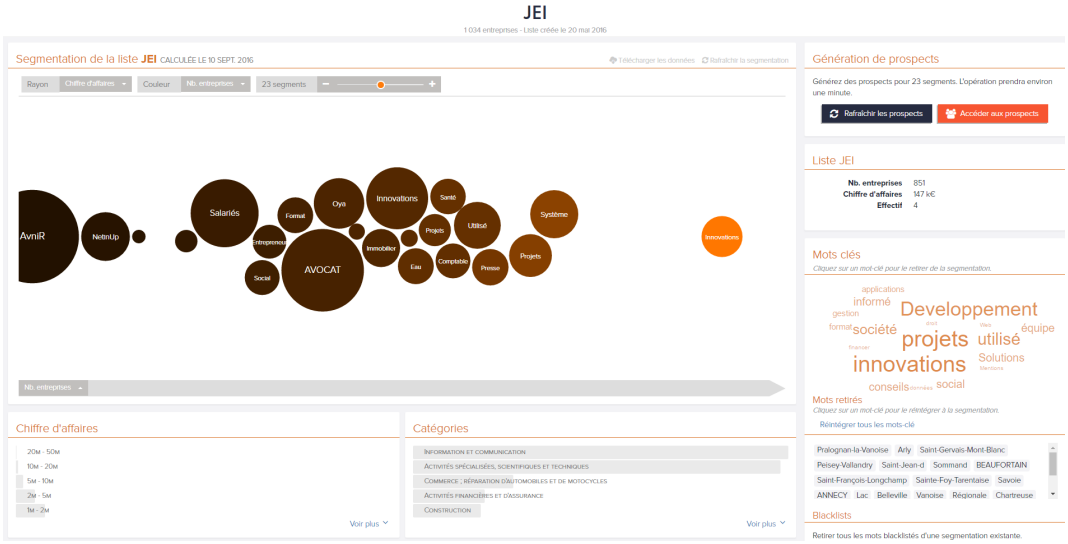


Figure 11: Interface de segmentation

2.5.4 Targeting

Le Targeting ou ciblage permet à l'utilisateur de créer son modèle. L'utilisateur se voit proposer des entreprises similaires en fonction des entreprises d'une liste donnée. Il doit alors déterminer si une entreprise lui parait pertinente. Cela permet alors de créer les deux classes du modèle, les

échantillons positifs et négatifs. ci-dessous (Figure 12), une entreprise à été sélectionnée par le ciblage et nécessite une intervention manuelle pour sélectionner une action parmi les trois suivantes :

- Positif : intègre l'ensemble d'apprentissage positif ;
- Negatif : intègre l'ensemble d'apprentissage négatif ;
- A revoir : sera reproposée ultérieurement ;
- Ignorer : écartée de la liste des propositions.

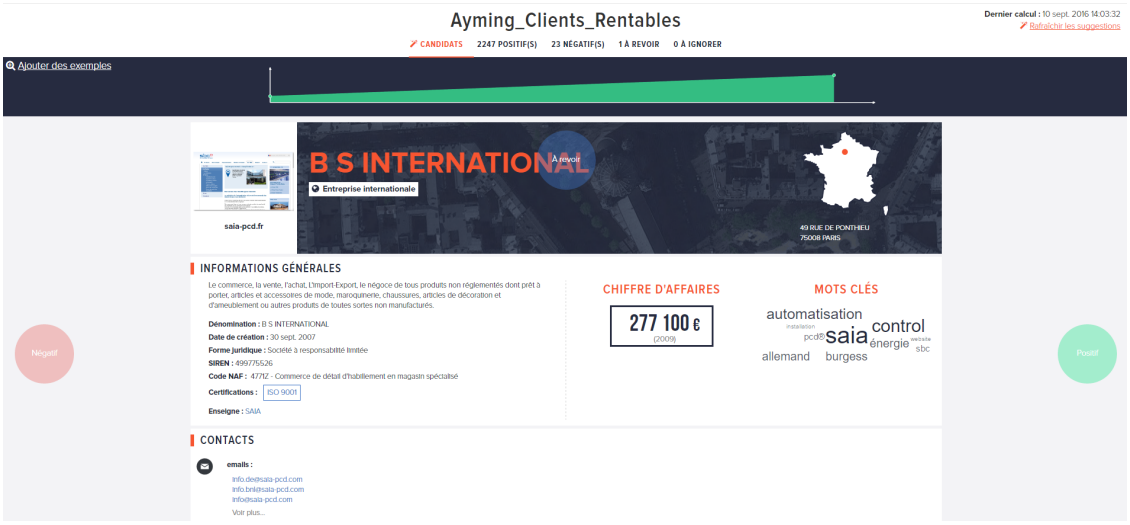


Figure 12: Capture d'écran de l'interface du ciblage

Plus l'utilisateur donne d'échantillons, plus le modèle possèdera un ensemble d'apprentissage important, et plus il sera cohérent. Pour faciliter l'apprentissage du modèle on rajoute aléatoirement un ensemble d'entreprises dans les négatifs pour équilibrer les deux classes (si elles ne le sont pas déjà). Cela est très important puisque le modèle utilisé est un modèle Naive Bayesien. Ce modèle prend en compte la probabilité a priori dans l'élaboration du modèle de sortie. Si les deux classes sont vraiment très disproportionnées le modèle aura une probabilité a priori trop importante et donc tous les échantillons seront prédits comme faisant partie de la classe prépondérante.

Cet algorithme implémente Naive Bayes pour les différentes variables. Cela est très utilisé dans la classification de texte où les données ont une représentation sous la forme de dénombrement de vocabulaire par rapport à un échantillon. La probabilité finale repose sur la probabilité de chaque terme sachant la classe.

Naive Bayes est un modèle conditionnel. Il prend comme hypothèse que toutes les variables sont indépendantes (ce qui n'est pas forcément vrai dans l'absolu), il est dit naïf. On peut écrire la probabilité conditionnelle de la variable X en fonction des autres variables $Fi,\ \forall i \in [1 \dots N]$.

$$p(X|F_1, \dots, F_N) = \frac{p(X)\ p(F_1, \dots, F_N|X)}{p(F_1, \dots, F_N)}. \tag{3}$$

En tenant compte de l'hypothèse d'indépendance on peut écrire,

$$\forall\, i \neq j,\ p(F_i|X, F_j) = p(F_i|X) \tag{4}$$

source

La sélection de variable de type Chi2 permet de réaliser un test du Chi2 et de sélectionner les k variables avec les scores les plus importants. Le test du Chi2 permet de déterminer la dépendance de deux vecteurs entre eux. Ces deux vecteurs sont créés par la variable testée en la découpant selon les deux classes positive et négative. Si ces deux vecteurs sont indépendants, donc que la variable testée n'est pas dépendante de la classe prédite, c'est que cette variable n'est pas pertinente pour la classification, elle aura donc un score proche de 0.

2.5.5 Catégorisation

Les entreprises possèdent un code NAF qui permet de plus ou moins définir leur secteur d'activité mais il existe plusieurs autres catégories auxquelles les entreprises peuvent appartenir. Quatre d'entres elles ont été mises en place dans C-Radar : eCommerce, International, Startup et B2B/B2C[8].

Le machine learning et des algorithmes d'apprentissage supervisé permettent de classifier des entreprises dans les catégories B2B/B2C et Startup. Un score composé de plusieurs métriques permet d'affecter les entreprises dans les deux autres catégories (eCommerce et International).
L'opération de catégorisation est effectuée à chaque mise à jour du crawl. Le plugin permettant de catégoriser utilise des données sémantiques mais aussi des données présentes sur C-Radar comme la présence de réseaux sociaux, les données financières ou encore les données juridiques comme la date de création.

Ces catégories sont ajoutées dans l'interface C-Radar par le biais d'une étiquette sur la fiche entreprise. Par exemple Data Publica est identifié comme une entreprise du digital et internationale (Figure 13).

Figure 13: Exemple d'affichage de catégories

2.5.6 QA (Quality Assessment)

La plupart du temps il n'existe pas de données de référence pour le problème de classification que l'on cherche à résoudre. Il faut alors créer des données "à la main". Nous avons besoin des données pour apprendre un modèle mais aussi pour le tester.
Le rôle d'un QA est de créer ces ensembles manuellement. Ces données sont souvent assez longues et difficiles à déterminer, c'est pour cela que plusieurs fois par mois, des sessions d'environ une heure sont organisées durant lesquelles toute l'équipe se rassemble pour un maximum d'efficacité.

[8]Business to Consumers : toutes les activités d'entreprises se rapportant aux consommateurs finaux

3 Mes Missions

Durant mon stage j'ai travaillé sur de nombreux sujets. Des sujets clients, mais aussi plusieurs projets axés sur l'amélioration du produit.

J'ai travaillé sur l'intégralité d'un projet client (Cf. Projet Ayming partie 3.1) avec deux autres collègues.
Après cela, j'ai pu ajouter deux fonctionnalités au produit :

- un plugin Python permettant détecter les plateformes technologiques utilisées par les entreprises sur leur site web et de créer un screenshot de la page de garde.
- une extension Chrome permettant d'afficher des informations basiques sur l'entreprise du site web visité par l'utilisateur.

Enfin, j'ai fait une étude sur les mauvaises associations des sites web à des entreprises.

Je vais détailler ces trois missions dans les paragraphes qui suivent.

3.1 Projet Ayming : scoring de prospects

3.1.1 Contexte

Ayming est une société de conseil aux entreprises. Ils souhaitaient faire appel à Data Publica pour optimiser leur recherche de prospects afin d'augmenter leur rentabilité. Nous avons travaillé avec le service d'aide aux entreprises permettant de rechercher des financements et des crédits d'impôt recherche.

3.1.2 Description de mission

Le but de la mission était d'aider Ayming à différencier les clients rentables des clients non rentables sur la base de critères qu'il fallait identifier. Cette notion de rentabilité était assez floue. Pour faire simple, c'était une moyenne de revenu pour Ayming supérieure à 5000€ sur les années de collaborations.
Ensuite il nous était demandé de scorer les prospects, suspects, les entreprises innovantes et les prospects générés par la segmentation. Ces quatre catégories ont été définies en séances lors des réunions de lancement et d'avancement :

- Les prospects sont les entreprises faisant partie du pipeline commercial[9].

[9] Opportunités d'affaires de l'entreprise

- Les suspects sont des entreprises n'ayant pas encore été contactées par un commercial mais qui ont été catégorisées comme intéressantes pour Ayming, ce sont les leads[10].
- Les entreprises innovantes sont des entreprises respectant au moins un critère d'innovation (Ayant levé des fonds, ayant déposé un brevet, faisant partie d'un pôle de compétitivité, etc.).
- Les prospects générés par la segmentation, sont la projection des segments déterminés après nettoyage de la segmentation (Cf. partie 2.5.3).

Nous avions à notre disposition le portefeuille complet d'Ayming comprennant 72 531 entreprises regroupées également en quatre catégories mais différentes de celles que l'on vient de détailler.

Statut	Compte
Suspect	66 490
Clients rentables	2 238
Client non-rentables	3 421
Prospects	382

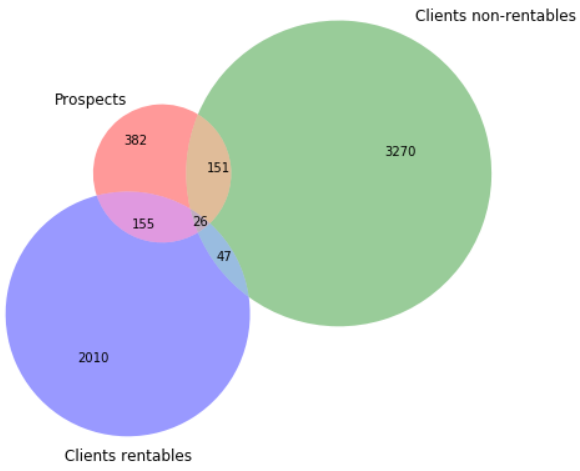


Figure 14: Recouvrement des différentes catégories

Les catégories du portefeuille d'Ayming étaient recouvrantes comme le montre la figure 14.

[10]Clients potentiels

3.1.3 Récupération des données

Pour créer l'ensemble d'apprentissage il a fallu identifier les dimensions pouvant être intéressantes pour la classification des clients d'Ayming. Certaines données ont pu être récupérées à partir d'anciens projets, d'autres ont dû être mises à jour. Nous avons utilisé la base de C-Radar (Données d'innovation, Analyse sémantique de la description, Titres de postes) et récupéré de nouvelles données depuis des sources Open Data (Crédits d'Impots Recherche et Innovation, Projets Européens). Détaillons à présent ces cinq catégories.

Crédits d'Impots Recherche, Innovation :
Dans un premier temps nous avons utilisé des données sur des critères d'innovation. Un bon facteur d'innovation est d'avoir reçu l'agrément CIR[11] ou CII[12] (Crédit Impôt Recherche ou Crédit Impôt Innovation). Ces données sont disponibles sur le site du ministère sous forme de fichier PDF. Nous avons utilisé Tabula qui permet de créer des règles et donc de récupérer des tables de données si elles sont formatées de la même façon sur toutes les pages. On peut alors créer un fichier CSV contenant toutes les données. Ces données sont accompagnées du SIREN de l'entreprise qui nous permet de les intégrer directement à notre modèle de données.

Projets Européens :
Nous avons aussi récupéré des données sur les fonds de financements européens H2020 et FP7 (les données des projets plus anciens sont obsolètes et incomplètes et n'ont pu être utilisées). Ces données comprennent les dates des projets, les descriptions, ainsi que les montants des financements reçus. Cependant il n'y a pas de SIREN associé à ces données ce qui empêche de les matcher à C-Radar dont la base du modèle de données repose sur le SIREN des entreprises (qui est à différencier du numéro SIRET qui identifie chaque établissement de l'entreprise).

Pour rendre ces données utilisables, il est nécessaire d'affecter un SIREN à ces données. Pour cela, un outil à été développé : le Sirenizator qui permet d'associer un numéro SIREN à des informations basiques de celles ci. Cet outil, qui n'en est encore qu'à sa phase de prototypage, a été mis en service lors d'un friton[13] que Data Publica organise deux fois par an. Le script utilise des données fournies en entrée comme l'adresse, le site web, la ville, le code postal pour "réconcilier" ces informations avec celles déjà présentes dans C-Radar .
En général, 30 à 40% des entreprises sont réconciliées automatiquement. Dans les autres cas, aucune correspondance exacte n'a été trouvée. Un score est alors calculé en prenant en compte la distance entre les données fournies et celles des candidats potentiels. Une validation manuelle est alors nécessaire pour trouver la bonne entreprise.

Données d'innovation :
Concernant les données d'innovation, plusieurs sources de données sont à notre disposition :

- La base C-Radar par le biais du modèle de données FullCompany qui regroupe les données financières, juridiques et extraites des sites web.
- Des bases MongoDB qui regroupent des données extraites dans le cadre de projets précédents.

[11] Crédit Impôt Recherche
[12] Crédit Impôt Innovation
[13] Hackaton Interne

Les données stockées sur Mongo ne sont pas accessibles directement depuis des API, il faut réaliser des exports de fichiers directement depuis le terminal Mongo ou depuis Python grâce au package `pymongo`. Ce package permet de réaliser toutes les opérations de base sur une collection Mongo : l'authentification, requêtes, etc.
Ces bases Mongo regroupent des données relatives :

- à l'appartenance aux pôles de compétitivité ;
- à des prix obtenus lors concours d'innovation ;
- au statut de jeune entreprise innovante (JEI).

Et depuis peu, l'INPI rend accessible ses données sur les dépôts de brevets. Ces données ont aussi été intégrées au modèle.

Analyse sémantique de la description :
Une analyse sémantique de la description ou de l'objet social des différentes entreprises a aussi été mise en place. La description est directement extraite des sites web ou du compte Facebook, alors que l'objet social est déclaré lorsque l'entreprise est créée. Il permet de déterminer plus précisément que le code NAF le secteur d'activité de l'entreprise.

Pour effectuer cette analyse de texte on cherche à créer une matrice TF-IDF comme celle utilisée pour l'analyse sémantique du contenu des sites web (Cf. partie 2.5.1). Nous ne gardons que les 500 mots les plus fréquents ce qui correspond à 0.5% du corpus tout entier. Cette matrice ne comprend qu'un peu plus de 100 000 échantillons, c'est pour cette raison que nous pouvons la manipuler localement.

Nous avons utilisé le package d'extraction de features de Scikit Learn et les corpus de NLTK pour réaliser ces opérations.

Titres de postes :
Une analyse des titres et fonctions des employés présents au sein l'entreprise a aussi été intégrée. Nous récupérons les différents titres des postes des employés de l'entreprise en crawlant son site web et en faisant le lien avec d'éventuels comptes LinkedIn détectés sur ce site. Ayming a fourni une liste de titres qui leurs semblaient intéressants et qu'ils utilisaient dans leur prospection. Certains étaient "impactant" ou "non impactant" (classification internes à Ayming) selon l'importance qu'ils avaient dans la prospection. Une approche TF-IDF a été mise en place pour pour normaliser ces données.

Impactant	Non impactant
Chercheur	Chargé de Mission
Directeur Recherche Développement	Chef de Projet
Directeur Scientifique	Coordinateur
Directeur de Laboratoire	Directeur de Projet
Directeur de Projet	Directeur technique
Ingénieur	Responsable Qualité
Responsable R&D	Responsable Technique

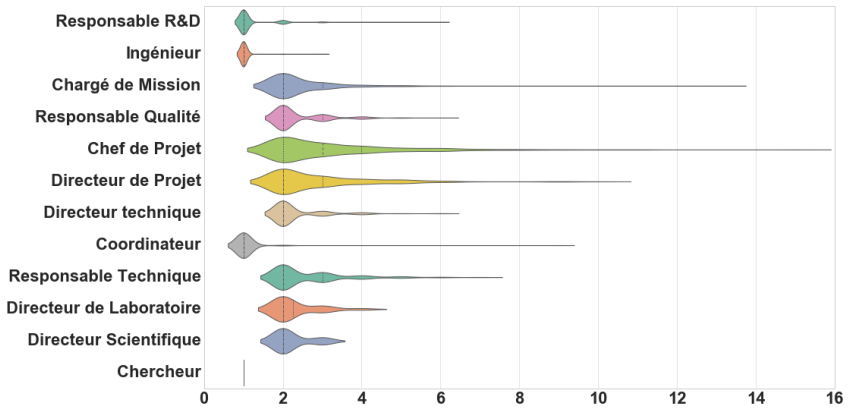


Figure 15: Distribution des différents postes

On peut voir la distribution des différents postes sur la figure 15.

Récapitulatif des données :

En regroupant les données des cinq catégories que nous venons de détailler, nous arrivons à environ 1300 variables.

Pour faciliter les analyses, nous avons regroupé les données en 9 catégories qui sont explicitées par le préfixe de la variable :

- **Titres** : Analyse des titres des fonctions présentes dans les entreprises analysées.
- **TextMining** : Chaque mot du vocabulaire forme une colonne.
- **Social** : Une colonne par réseau social utilisé. LinkedIn, Facebook et Twitter.

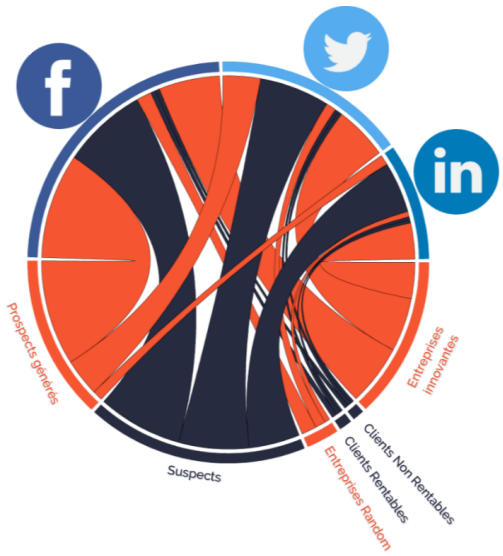


Figure 16: ChordDiagram représentant le dénombrement des différentes catégories dans les 3 réseaux sociaux

- **Pole** : Ces variables représentent l'appartenance aux différents pôles de compétitivité. 24 des plus gros pôles sont référencés sur les 72 existants. Ces derniers totalisent la moitié des entreprises faisant partie d'un pôle en France.

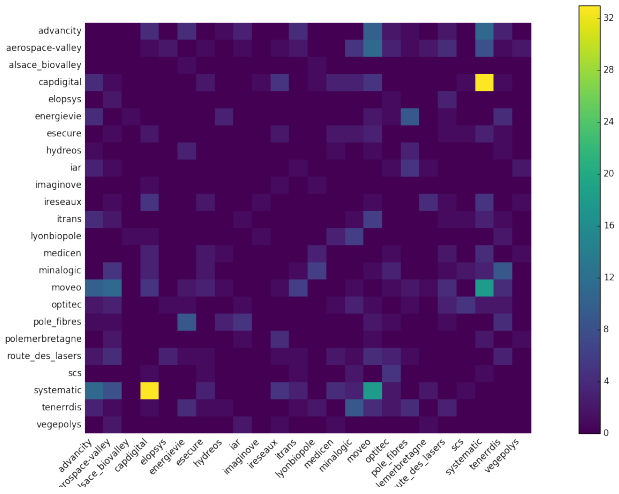


Figure 17: Intersection des entreprises entre les différents pôles

On peut observer (Figure 17) le recouvrement des différents pôles. Ce recouvrement représente le dénombrement des entreprises présentes dans deux pôles à la fois. Certaines entreprises font parties de plus de deux pôles, cela peut aller jusqu'à 6.

- **NAF5** : Tous les codes NAF de précision maximum (niveau 5) ont été pris en compte. Cela ajoute 735 colonnes au modèle mais environ 200 colonnes sont vides puisque de nombreux codes NAF ne sont pas représentés dans la base. La transformation des codes NAF en codes numériques de 0 à 734 sur une colonne permet de réduire le nombre de dimensions. Mais il est ensuite beaucoup plus difficile d'analyser et d'interpréter le comportement de la variable dans la décision finale.
  D'après O. Grisel (un des contributeurs principaux de Sklearn), lors de la conférence PyData Paris 2016, si la cardinalité des données catégorielles est élevée, l'encodage par des entiers fonctionne aussi bien, si ce n'est mieux que des variables binaires pour des algorithmes d'arbres (random forest, arbres boostés). Cependant notre but premier est de rendre accessibles les résultats à des équipes commerciales. Nous avons donc choisi de rester sur un encodage binaire.

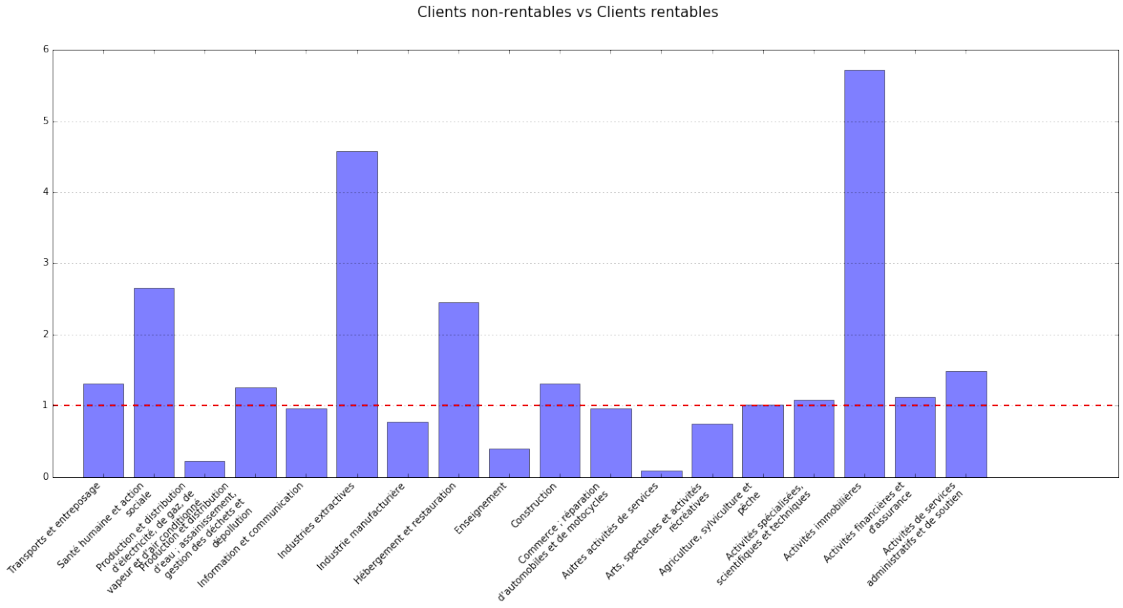


Figure 18: Quel code NAF représente le mieux les clients rentables ?

Ce graphique (Figure 18) représente la répartition des entreprises rentables et non rentables selon les différents code NAF de niveau 0. Toutes les codes NAF au dessus de la ligne rouge (qui représente l'équi-répartition entre les deux classes) sont des secteurs plus présents dans les entreprises rentables.

- **Innov** : Ce sont toutes les variables d'innovation dont nous avons parlé plus haut. Ce sont des données importantes pour Ayming qui travaille avec des entreprises souhaitant acquérir l'agrément CIR.

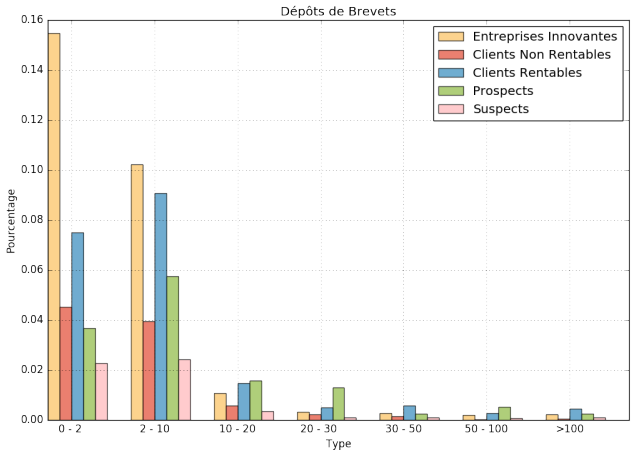


Figure 19: Quelle catégorie dépose le plus de brevets ?

- **Financ** : Ce sont les données financières de base. Le chiffre d'affaires, le budget Recherche & Développement et le budget de propriété intellectuelle (IP) ont été intégrés. Les budgets R&D et IP sont déclarés sur le compte de résultats de l'entreprise. Cependant il faut faire attention à l'exploitation du budget IP. En effet, il englobe les dépôts de brevets, qui est un bon marqueur d'innovation. Cependant les dépôts de marques ainsi que les droits d'auteur (des films ou musiques par exemple) et les achats de franchises en font aussi partie alors que ce ne sont pas des marqueurs de l'innovation.

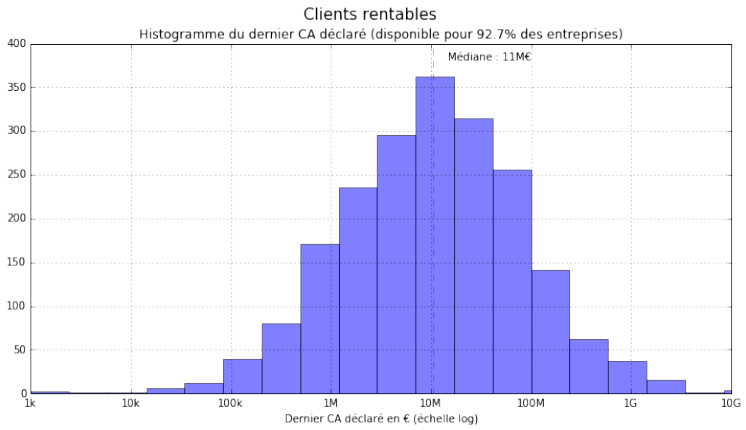


Figure 20: Distribution du chiffre d'affaire pour les clients rentables

- **Entity** : Type d'entités qui représente le statut juridique de l'entreprise. Cela permet par exemple d'identifier les associations à but non lucratif ;
- **Certif** : Toutes les certifications ISO, AQAP.

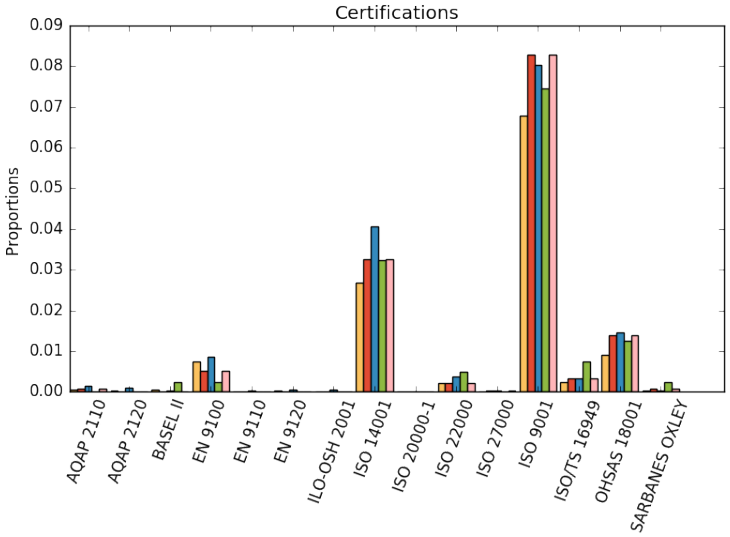


Figure 21: Décompte des certifications détectées par catégorie

3.1.4 Analyses

Modèles de données :
Le code suivant permet de créer une structure de données Python `pandas` à partir de fichiers CSV et EXCEL. Cette structure de données est nécessaire pour la suite du projet.

```
client_non_rentables = pd.read_csv("./data/client_non_rentables.csv")
client_rentables = pd.read_csv("./data/client_rentables.csv")
df_random = pd.read_csv("./data/df_random.csv")
df_client_data = pd.read_excel("./data/df_client_data.xlsx")

def normalize(X, id_col=""):
 scaler = sklearn.preprocessing.StandardScaler()
 if not id_col:
     return pd.DataFrame(scaler.fit_transform(X.as_matrix()), columns=X.columns)
 else:
     xids = X._id
     X = pd.DataFrame(scaler.fit_transform(X.drop(id_col, axis=1).as_matrix()),
                      columns=X.drop(id_col, axis=1).columns)
     X[id_col]= list(xids)
     return X

Y = np.array(["Clients"] * (len(client_non_rentables.index)+
        len(client_rentables.index)) +
        ["Random"] * (len(df_random.sample(n=5000).index)))
```

Nous avons testé différents modèles de données pour essayer de trouver un compromis qualité/cohérence convenant à Ayming. Trois modèles n'ont pas été retenus à cause de leurs performances moyennes :

1. un modèle cherchant à classifier les client rentables des clients non-rentables : le faible ensemble d'apprentissage et la forte proximité des deux classes ne permettait pas de les différencier. En effet, les performances moyennes pour les différents algorithmes testés étaient de 60% ;

2. un modèle avec un plus grand nombre classes a été entraîné. Les clients non-rentables constituaient la première classe et 5 classes ont été mises en place en fonction du taux de rentabilité. Ce modèle présentait des performances très moyennes et aucune corrélation n'a été trouvée entre les classes prédites et le taux de rentabilité sur l'ensemble de test. Les algorithmes spécifiques à la classification multilabel n'ont cependant pas été testés dans ce cas et pourrait potentiellement être une solution ;

3. un modèle visant à déterminer la rentabilité d'une entreprise. La classe négative est constituée de l'ensemble des clients non-rentables ainsi que d'entreprises tirées aléatoirement dans l'ensemble des entreprises françaises. Cela permet dans un certain sens de réduire le biais de la prospection d'Ayming. Cependant les deux types d'entreprises constituant la classe négative étaient vraiment très éloignées et les performances du modèle étaient moins bonnes que celles du premier (distinguer un client rentable d'un client non-rentable) ;

Parmi tous les modèles testés, un seul à été conservé. Cependant il ne permettait pas de prédire la rentabilité d'une entreprise mais seulement sa probabilité de devenir un client

pour Ayming. Pour modéliser la classe négative nous avons tiré au hasard 5000 entreprises françaises. Cela permet d'avoir un panel représentatif de l'ensemble des entreprises et un ensemble d'apprentissage conséquent. La classe positive était composée de 5 659 entreprises clientes d'Ayming, rentables et non rentables.

Pour résumer, nous essayons de déterminer la probabilité qu'une entreprise soit dans un contexte favorable pour l'obtention de CIR, c'est à dire, potentiellement un futur client d'Ayming.

Algorithmes :
Tous ces modèles de données ont été testés avec de nombreux algorithmes :

- Logistic Regression ;
- Random Forest (ou Forêt Aléatoire) ;
- Support Vector Machine (SVM) ;
- AdaBoost ;
- Bagging ;
- Gradient Tree Boosting.

Finalement seule la Random Forest a été conservée. Elle représente un bon compromis entre vitesse de calculs et performances. En effet, elle ne nécessite que très peu d'ajustements au niveau de ses paramètres contrairement à une SVM ou des arbres boostés par exemple. Elle permet aussi très généralement d'obtenir de meilleures performances qu'un modèle linéaire simple. Cependant ce modèle est assez compliqué à expliquer à une audience non scientifique. Le modèle retourne une importance moyenne des variables dans la décision contrairement à un modèle linéaire qui renvoie des pondérations pour chacune d'entre elles.
Même s'il existe un package Python `treeinterpreter` permettant d'interpréter plus facilement les Random Forest, cela ne marche que pour des observations locales. En effet, l'interpréteur utilise chaque observation localement et calcule l'importance de chaque feature pour sa prédiction. Cela permet d'interpréter alors des pondérations comme pour un modèle linéaire. Cette méthode est intéressante pour la description d'une prédiction individuelle mais pas d'un modèle dans son ensemble.

Le code suivant permet d'automatiser la création des courbes de précision/rappel ainsi que de permettre à l'utilisateur de préciser un seuil pour la classe positive et/ou négative et d'observer les valeurs de précision et rappel moyennes en fonction de ces seuils.

```
def create_precision_recall_curve(y_true, pred_proba, x1=None, x2=None):
     precision1, recall1, threshold1 = precision_recall_curve(y_true=y_true,
                probas_pred=np.array(pred_proba)[:,0], pos_label=0)
     precision2, recall2, threshold2 = precision_recall_curve(y_true=y_true,
            probas_pred=np.array(pred_proba)[:,1], pos_label=1)

     fig, (ax1, ax2) = plt.subplots(1,2)
```

```
    fig.set_size_inches(18, 6)
    ax1.plot(list(threshold1) + [1], precision1, label="Precision")
    ax1.plot(list(threshold1) + [1], recall1 , label="Rappel")
    ax1.set_title("Classe Negative")
    ax2.plot(list(threshold2) + [1], precision2, label="Precision")
    ax2.plot(list(threshold2) + [1], recall2 , label="Rappel")
    ax2.set_title("Classe Positive")
    if x1:
        print("Seuil 1 : ")
        ax1.axvline(x1, color="black")
        try:
            pres_thresh1 = np.max([y for x,y in zip(threshold1, precision1) if
                                   np.isclose(x,x2, rtol=1e-03)])
            recall_thresh1 = np.max([y for x,y in zip(threshold1, recall1) if
                                   np.isclose(x,x2, rtol=1e-03)])
            pprint(["%.2f"%pres_thresh1, "%.2f"%recall_thresh1])
        except:
            pass
    if x2:
        print("Seuil 2 : ")
        ax2.axvline(x2, color="black")
        try:
            pres_thresh2 = np.max([y for x,y in zip(threshold2, precision2) if
                                   np.isclose(x,x2, rtol=1e-03)])
            recall_thresh2 = np.max([y for x,y in zip(threshold2, recall2) if
                                   np.isclose(x,x2, rtol=1e-03)])
            pprint(["%.2f"%pres_thresh2, "%.2f"%recall_thresh2])
        except Exception as e:
            print(e)
    ax1.grid(color="white")
    ax2.grid(color="white")
    ax1.legend(loc=0)
    ax2.legend(loc=0)
```

Le code python ci-dessous permet de mettre en place une Régression Logistique ainsi que d'afficher les principales indications sur les performances (matrice de confusion, rapport de classification et courbes de précision/rappel). Ces performances sont calculées en validation croisée (Cf. 3.1.6).

```
from sklearn.preprocessing  import StandardScaler
from tqdm import tqdm
scaler = StandardScaler()
cols_to_drop= ['Innov_STATUS', "TextMining_Unnamed: 0", "Unnamed: 0", "_id"]
X = df_to_predict.drop(cols_to_drop, axis=1).fillna(0).copy()
X = scaler.fit_transform(X)
cols = df_to_predict.drop(cols_to_drop, axis=1).columns

clf = LogisticRegression()
```

```
y_true_array = []
pred_array = []
pred_proba_array =[]
#for train_index, test_index in tqdm(StratifiedKFold(Y, n_folds=4)):
for train_index, test_index in StratifiedKFold(Y, n_folds=4):
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = Y[train_index], Y[test_index]
    clf.fit(X_train, y_train)
    y_true_array.extend(y_test)
    pred_array.extend(clf.predict(X_test))
    pred_proba_array.extend(clf.predict_proba(X_test))

print(confusion_matrix(y_true_array, pred_array))
print(classification_report(y_true_array, pred_array))
```

```
   Matrice de confusion :

[[4964  695]
 [ 597 4403]]

 Rapport de classification :

             precision    recall  f1-score   support

    Clients       0.89      0.88      0.88      5659
     Random       0.86      0.88      0.87      5000

avg / total       0.88      0.88      0.88     10659
```

```
y_true_num = [0 if x == "Clients" else 1 for x in y_true_array]
create_precision_recall_curve(y_true_num, pred_proba_array);
```

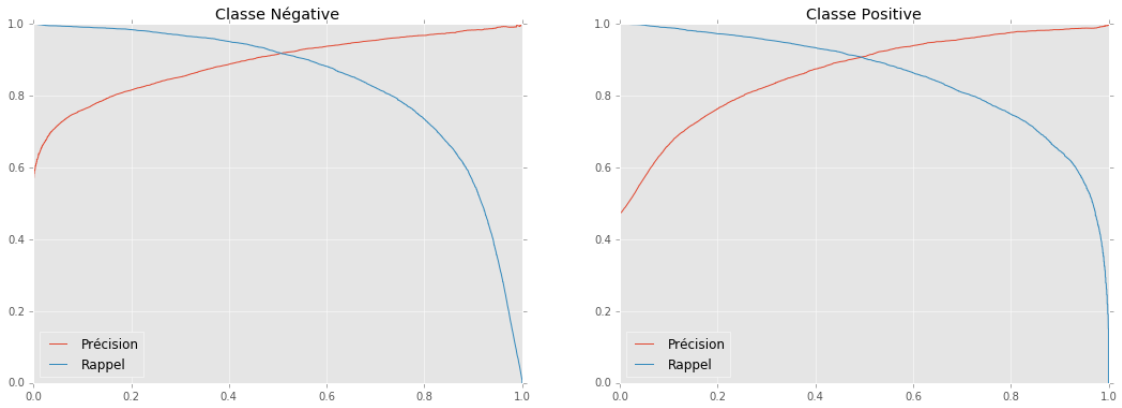


Figure 22: Courbes de précision rappel pour la régression logistique

Le code suivant permet de mettre en place une Random Forest.

```
scaler = StandardScaler()
cols_to_drop= ['Innov_STATUS', "TextMining_Unnamed: 0", "Unnamed: 0", "_id"]
X = df_to_predict.drop(cols_to_drop, axis=1).fillna(0).copy()
X = scaler.fit_transform(X)
cols = df_to_predict.drop(cols_to_drop, axis=1).columns

clf = RandomForestClassifier(n_estimators=1000, n_jobs=4, max_features=30)

y_true_array = []
pred_array = []
pred_proba_array =[]
#for train_index, test_index in tqdm(StratifiedKFold(Y, n_folds=2)):
for train_index, test_index in StratifiedKFold(Y, n_folds=2):

    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = Y[train_index], Y[test_index]
    clf.fit(X_train, y_train)
    y_true_array.extend(y_test)
    pred_array.extend(clf.predict(X_test))
    pred_proba_array.extend(clf.predict_proba(X_test))

print(confusion_matrix(y_true_array, pred_array))
print(classification_report(y_true_array, pred_array))
```

```
   Matrice de confusion :

[[5206  453]
 [ 484 4516]]

 Rapport de classification
             precision    recall  f1-score   support

    Clients       0.91      0.92      0.92      5659
     Random       0.91      0.90      0.91      5000

avg / total       0.91      0.91      0.91     10659
```

```
y_true_num = [0 if x == "Clients" else 1 for x in y_true_array]
create_precision_recall_curve(y_true_num, pred_proba_array)
```

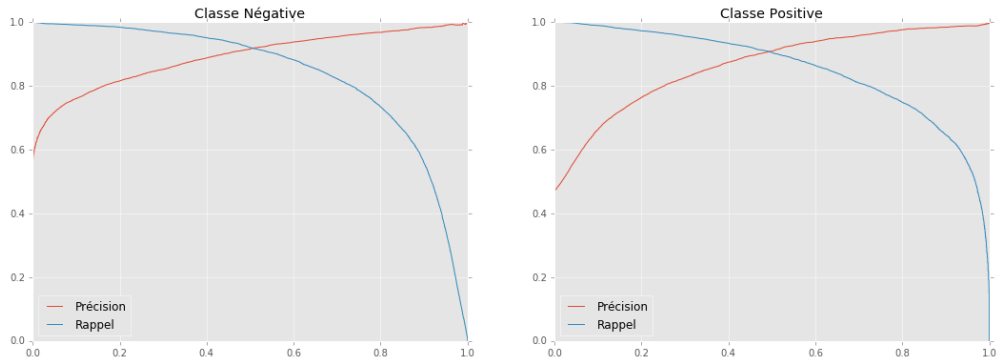


Figure 23: Courbes de précision rappel pour la Random Forest

Finalement, on voit ici que les résultats de la Random Forest (91%) sont nettement supérieurs à ceux de la Régression Logistique (88%). Ce modèle permet d'avoir des performances intéressantes sur les deux classes positive et négative. Pour que la précision et le rappel soient interprétables il faut observer une certaine équité entre les deux classes. C'est pour cela qu'autant d'entreprises aléatoires que d'entreprises clientes ont été tirées, pour équilibrer l'ensemble d'apprentissage, la précision et rappel moyens étant toujours pondérés au support.
Ici le package `tqdm` permet de créer une barre de progression. Cela permet d'avoir un aperçu de l'avancement des calculs. Les lignes ont été commentées pour faciliter l'export Jupyter Notebook -> Latexpour des causes de compilation LateX.

3.1.5 Random Forest

La Random Forest est un algorithme ensembliste supervisé qui réalise un grand nombre d'arbres de décision sur un sous-ensemble d'échantillons tiré aléatoirement et sur un sous-ensemble de variables.

Le principe de l'apprentissage ensembliste est de combiner un ensemble de prédiction d'un grand nombre de classifieurs. Ces méthodes ensemblistes permettent d'augmenter la généralisation du modèle et sa robustesse.
La prédiction finale repose sur un "vote" de tous les arbres. La classe prédite sera alors la classe déterminée par le plus grand nombre d'arbres. La probabilité de sortie sera la proportion d'arbres (ou plus généralement de classifieurs) ayant donné la classe positive, sur le nombre d'arbre total. Plus un échantillon sera "élu" à l'unanimité, plus sa probabilité de sortie sera proche de 1.

3.1.6 Validation croisée

La validation croisée permet de tester un modèle sur plusieurs ensembles d'apprentissage et de test pour évaluer les réelles performances du modèle sur des ensembles totalement différents. Cela permet de prouver la généralisabilité du modèle sur d'autres ensembles. On peut très facilement réaliser cette validation croisée depuis `sklearn` avec le module `Stratified KFolds`. Elle permet contrairement à une simple validation croisée (`KFolds`) de garder, pour les exercices de classification, la part de chaque classe dans le découpage des ensembles d'apprentissages et de tests.

3.1.7 Segmentation et Génération de Prospects

Après la génération du modèle, un segmentation sur les clients d'Ayming a été effectuée. Cela permet de découper les différents clients de façon uniforme en prenant en compte la sémantique de leurs sites web. Pour cela nous avons utilisé l'outil de segmentation dont nous avons parlé un peu plus haut (Cf. partie 2.5.3). Lorsque le clustering est réalisé, il faut faire un nettoyage approfondi et une validation manuelle des entreprises présentes dans les différents segments pour avoir une segmentation propre et cohérente avec des ensembles les plus homogènes possibles.

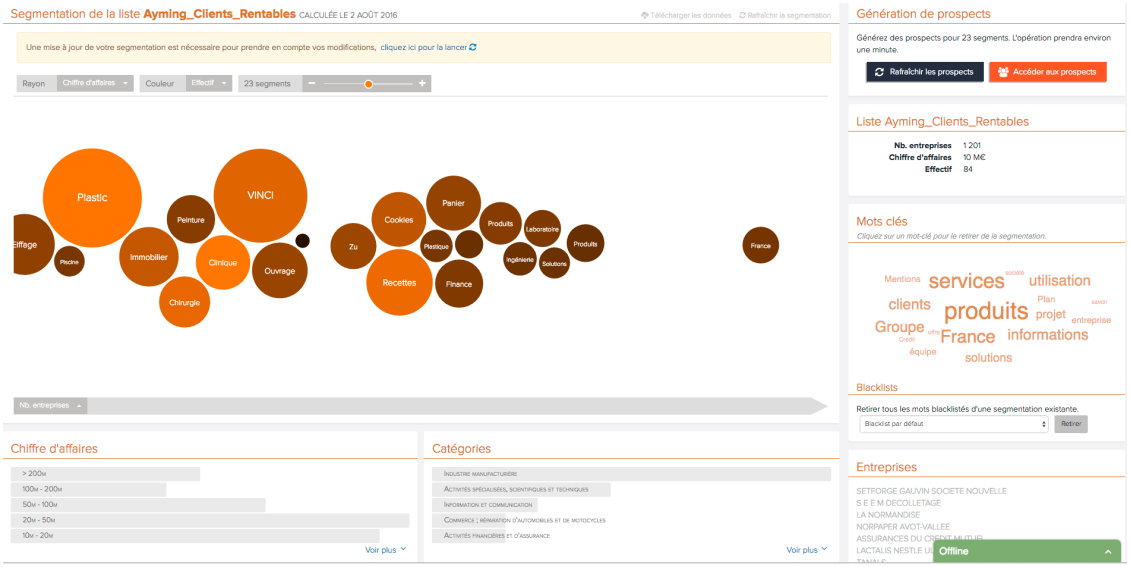


Figure 24: Interface de segmentation

Ensuite, les 600 000 entreprises de la base C-Radar possédant un site web sont projetées sur les segments obtenus. Cela permet de trouver des entreprises similaires aux segments retenus : c'est la génération de prospects (Cf. partie 2.5.3). Nous ne gardons que les 8 segments les plus intéressants sur les 25 segments déterminés par l'algorithme. Les prospects ne sont générés que sur ces 8 segments. Cette projection extrait 40 000 entreprises proches du centre des clusters.

Prédiction, Scoring et validation manuelle Après la génération de prospects, le modèle mis en place doit être appliqué ce qui permet de déterminer si une entreprise est un client potentiel de Ayming. Quatre types d'entreprises doivent être alors scorés: les prospects générés par la segmentation, les prospects en cours de contact par Ayming, l'ensemble du portefeuille de prospection, et les entreprises innovantes (Cf. description des types d'entreprise partie 3.1.2). La projection du modèle est ensuite appliquée aux entreprises.

Un fichier Excel est créé, il contient sur chaque page les prédictions pour les différentes catégories. Différentes informations peuvent être ajoutées pour faciliter l'utilisation du fichier ou pour le CRM[14] d'Ayming.

[14]Customer Relationship Management : La gestion de la relation client (GRC) en francais

3.1.8 Gestion de projet

Nous étions 3 personnes à travailler sur ce projet. J'ai travaillé avec Emmanuel Jouanne qui s'occupait de la gestion du projet et qui faisait l'intermédiaire avec le client. J'ai également travaillé avec Clément Chastagnol sur toute la partie R&D et Data Science.

Nous avons animé 4 réunions avec différents responsables d'Ayming :

- La première servait de réunion de lancement, elle a permis de fixer les bases et les limites du projet, et d'avoir une explication plus approfondie sur les données fournies.
- La deuxième servait de réunion de livraison pour la première partie de mission, soit la partie analyse quantitative et qualitative des critères permettant de différencier les différentes entreprises de leur portefeuille. Nous avons pour cela réalisé de nombreux graphiques (Cf. partie 3.1.3) permettant de comprendre le contexte ainsi que d'expliquer les distributions des critères les plus importants et potentiellement les plus discriminants pour la classification. Cette première approche nous a permis de bien comprendre les données et de mieux estimer comment aborder les prochaines étapes de la mission. Ces deux réunions ont été réalisées dans les locaux d'Ayming à Gennevilliers.
- La troisième était une réunion téléphonique permettant d'expliquer les premières analyses de données effectuées avec les algorithmes de machine learning. Nous avons alors exposé tous les modèles de données et différents algorithmes que nous avions testé pour que nous soyons bien clair sur les résultats de notre étude. Cela nous a aussi permis de préparer la réunion finale qui était prévue avec les vendeurs et tous les directeurs commerciau d'Ayming pour leur présenter les conclusions et les convaincre d'utiliser les travaux réalisés.
- La dernière servait de réunion semi-finale (rajoutée au dernier moment), pour expliquer les analyses à la responsable du service. Elle était très proche de la précédente, avec quelques améliorations et la livraison du fichier de résultat. Il restera une dernière réunion fin septembre avec tous les directeurs commerciaux pour présenter le projet pour qu'ils puissent l'utiliser dans les meilleures conditions possibles.

3.2 Détection de technologies Web : package Python

3.2.1 Description

Le but de la mission était de détecter les technologies utilisées sur les sites web des entreprises. Ces technologies peuvent apporter des informations concernant le niveau d'appropriation des nouvelles technologies de la part de l'entreprise. Cette mission s'inscrit dans deux projets importants :

- Un projet client pour BPost SME, la poste belge ;
- L'ajout de nouvelles données dans C-Radar pour enrichir les fiches entreprises.

De plus, de nombreuses demandes ont été faites par des clients de C-Radar, pour connaître :

- le type de serveur utilisé pour l'hébergement des sites, ce qui permet de détecter des clients pour la maintenance aux entreprises ;
- les plateformes de chat intégrées sur les sites web ;
- les plateformes de e-commerce. Aujourd'hui les sites de e-commerce sont déjà détectés en utilisant des règles prenant en compte la valeur moyenne des prix présents, s'il y a présence d'un panier etc... En détectant les technologies comme Magento, OpenCart ou PrestaShop, on pourrait améliorer significativement la détection de sites e-commerce.

3.2.2 Approche

Dans un premier temps un tour d'horizon des différentes solutions disponibles sur le marché a été réalisé. Il existe plusieurs solutions payantes permettant de détecter très facilement et très précisément de nombreuses technologies. Cependant ces solutions sont coûteuses et non viables puisque la plus sérieuse proposée, Builtwith, proposaient un service autour de 6000€ pour un million d'appels depuis leur API. Sachant que 5 millions d'entreprises sont mises à jour tous les mois cela revient vite extrêmement cher.

Une autre solution est d'utiliser le plugin Open Source Wappalyzer.

Il peut être lancé à partir de plusieurs plate-formes (Python, Java, PhantomJS, etc...), mais :

- Python et Java ne disposent pas d'un environnement d'exécution JavaScript et permettent seulement une évaluation statique à partir d'expressions rationnelles appliquées sur le code source HTML de la page.
- PhantomJS permet de charger le contenu dynamique de la page et en particulier les variables globales du langage JS. Ce fonctionnement permet d'avoir une meilleure détection. Cependant, cette méthode ne peut accéder aux headers HTTP et PhantomJS est très instable, ce qui n'est pas viable dans une chaîne de production.

Finalement, le meilleur compromis est d'utiliser Wappalyzer avec un browser headless à travers le package python `Selenium` (développé initialement pour tester les développements web). Ce package permet de lancer un browser Mozilla Firefox ou Google Chrome pour charger une page et effectuer des opérations simples sur la page en question.

Utiliser une extension directement intégrée dans le navigateur headless permet de récupérer plus d'informations que simplement en injectant un script depuis la console, puisqu'elle peut accéder aux requêtes effectuées par le navigateur (et donc utiliser l'information présente dans les headers HTTP). Cependant il est impossible de récupérer les informations de l'extension depuis le DOM de la page et Selenium ne permet pas d'y accéder directement. **La raison est que le contexte JavaScript d'une extension est totalement différent de celui de la page elle-même**.

Le projet Wappalyzer étant Open Source, nous avons accès directement au code de l'application. Le code source sur Github n'étant pas à jour il a fallu le récupérer depuis les fichiers d'installation de Chrome.

On peut voir l'ID de Wappalyzer dans la copie d'écran du gestionnaire d'extension ci-dessous.

Figure 25: Capture d'écran de l'interface de gestion d'extensions

3.2.3 Fonctionnement

Pour détecter les différentes technologies, un fichier `apps.json` est disponible. Les clés décrivent les règles de détection à partir d'expressions rationnelles. Chaque clé JSON correspond à une source d'information différente :

- env : les variables d'environnement ;
- headers : les headers des requetes HTTP ;
- icon : les icônes ;
- script : les sources des scripts JavaScript permettant d'incorporer une technologie au site ;
- website : le code source HTML, détecte le site web de la technologie.

Un extrait du fichier `apps.json` est donné ci-dessous. Il permet de définir les règles permettant de détecter la technologie Google Analytics.

```
{
    "Google Analytics": {
            "cats": [
                10
            ],
            "env": "^gaGlobal$",
            "headers": {
                "Set-Cookie": "__utma"
            },
            "icon": "Google Analytics.svg",
            "script": "^https?://[^\\/]+\\.google-analytics\\.com\\/(?:ga|urchin|
            "website": "google.com/analytics"
        }
}
```

Ces règles doivent être constamment mises à jour. Après quelques réglages et comparaisons elles permettent d'accéder aux technologies les plus utilisées. Ces règles fonctionnent parfaitement lorsque les technologies sont implémentées de façon conventionnelle et comme préconisé par les développeurs des plate-formes.

On exécute un script JS à l'intérieur du script de l'extension. Ce script permet d'ajouter une balise de texte (avec un ID spécifique) dans laquelle sont insérées les informations récupérées à partir de l'extension. Cette balise est alors utilisée par le plugin Python géré par Selenium. Cette balise est rafraîchie à chaque fois que l'extension détecte une nouvelle technologie.

Un dictionnaire python est ensuite construit à partir des technologies détectées. Les clés de ce dictionnaire sont les technologies, les valeurs sont les catégories de ces technologies. Techniquement, JavaScript forge une string qui possède la syntaxe d'un dictionnaire Python. Cette string est ensuite évaluée par le package Python `AST`.

3.2.4 Plugin

Ces fonctionnalités sont intégrées au workflow de l'application C-Radar avec un plugin Python. Le plugin s'exécutera après le crawl des sites. Le but de ce plugin n'est pas seulement de récupérer les technologies détectées mais aussi de gérer la création du screenshot de la page principale du site web. Ce screenshot est alors affiché sur la fiche entreprise lorsque l'algorithme d'extraction de données ne permet pas de détecter le logo de l'entreprise (sur le site web ou sur les photos de profil des réseaux sociaux).

Au sein de l'architecture C-Radar, tous les plugins sont basés sur le même principe, ils reçoivent un JSON de paramétrage en entrée et retournent un JSON de résultats. Chaque plugin est abonné à une queue RabbitMQ qui gère une file d'attente. Il existe une queue pour chaque plugin. Cette queue est identifiée par un nom et un numéro de port. Cette architecture permet non seulement de gérer des événements (et donc d'appeler les plugins périodiquement) mais aussi de gérer la parallélisation des différentes instances de chaque plugin. En effet, le plugin de détection met environ 6 secondes par site pour réaliser les différentes tâches :

- charger la page ;

- récupérer les technologies ;
- prendre la capture d'écran.

S'il n'existait qu'une seule instance du plugin, cela prendrait plus de 900 heures pour analyser les 650 000 sites français et 2400 heures pour analyser l'ensemble des sites web dans le monde, ce qui représente 14 semaines. Si on lance 100 instances du plugin en parallèle cela ne prend plus qu'une journée.

Le plugin se charge de gérer le lancement du browser headless et de lui passer toutes les options de timeout, de taille de page, etc... Ensuite l'extension est installée à partir du `fichier.crx` compilé par Chrome. On ajoute aussi une extension AdblockPlus pour éviter les nombreuses pubs et les pop-ups qui réduiraient les performances du plugin et parasiteraient l'aperçu des screenshots.

La détection et la récupération des technologies est lancée lorsque la page est chargée par le biais de l'extension modifiée. On récupère le dictionnaire directement dans la balise créée à cet effet. On effectue ensuite un screenshot de la page web quand elle est "totalement" chargée. Nous avons ajouté un délai d'attente au timeout du browser avant de lancer le screenshot, par exemple pour éviter les pages blanches sur les sites dynamiques qui apparaissent avec des transitions.

On renvoie ensuite un JSON avec l'URL, le dictionnaire des technos, et l'image du screenshot en Base64 (permet de stocker une image sous forme de string, plus efficace pour le transfert de données).

ci-dessous un exemple d'exécution du plugin sur le site du FBI. On peut observer le dictionnaire Python produit ainsi que le screenshot.

```
In [33]: bh = BrowserHandler({"timeout":10,
                               "delay":2,
                               "width":1280,
                               "height":620})
         url = "https://www.fbi.gov/"
         bh.get_page(url)
         bh.get_dict_of_tech(url)

Out[33]: {'Backbone.js': ['javascript-frameworks'],
          'CloudFlare': ['cdn'],
          'Facebook': ['social-network'],
          'Google Analytics': ['analytics'],
          'MediaElement.js': ['video-players'],
          'Nginx': ['web-servers'],
          'RequireJS': ['javascript-frameworks'],
          'Twitter Bootstrap': ['web-frameworks'],
          'Underscore.js': ['javascript-frameworks'],
          'Youtube': ['social-network'],
          'jQuery': ['javascript-frameworks']}

In [34]: screen = bh.get_screenshot(url)
```

```
In [35]: display(Image(screen))
```

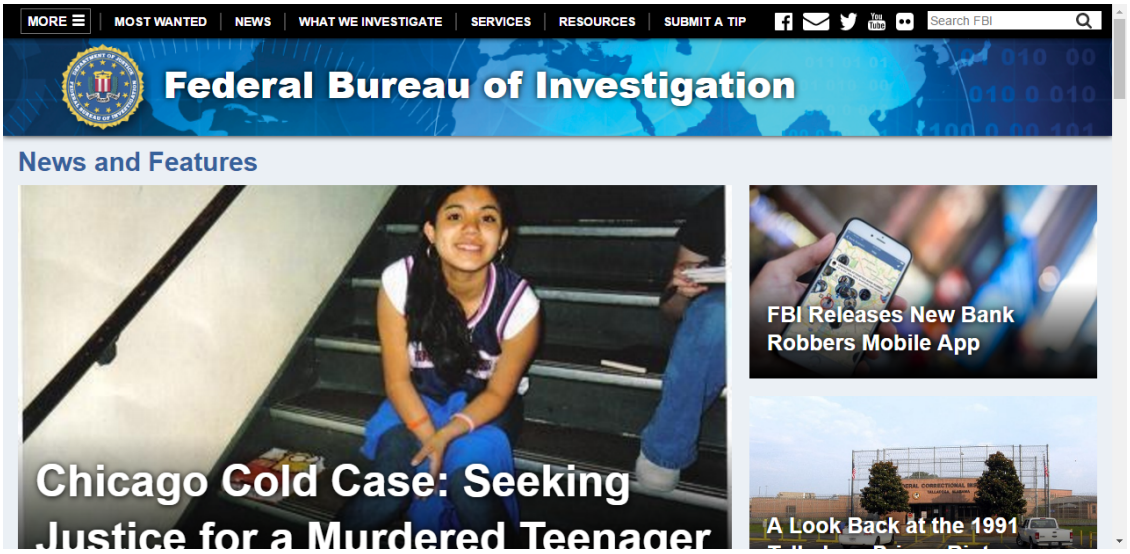


Figure 26: Capture d'écran provenant du plugin

3.2.5 La production

Pour la mise en production de ce plugin il a fallu intégrer les solutions à plusieurs problèmes qui sont survenus lors de la phase de test :

- Le navigateur qui tombe : il faut détecter le crash et relancer le navigateur ;
- les sites qui mettent en place des alertes JS, prenant la main sur le navigateur ;
- etc . . .

Après mon départ, la partie front a été mise en place. Depuis la fiche entreprise on peut maintenant voir le profil technique du site et donc toutes les technologies détectées par le plugin. C'est une valeur ajoutée à l'application C-Radar.

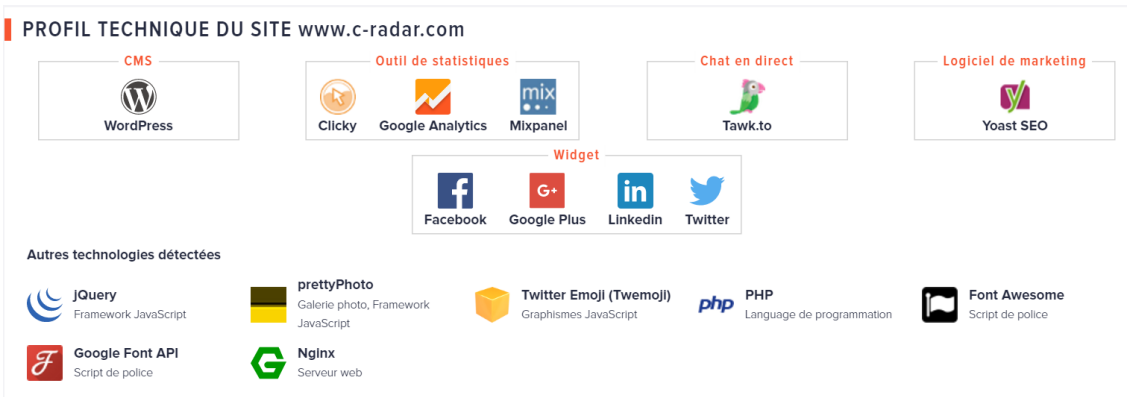


Figure 27: Affichage du résultat de détection de technologies sur la fiche entreprise

3.3 L'extension Chrome : affichage de données stratégiques

J'ai pris l'initiative de réaliser une extension pour le navigateur Chrome à destination des clients de l'application et des commerciaux de Data Publica. Cette extension a pour but d'afficher des informations concernant l'entreprise (dans une fenêtre supplémentaire) lorsque l'on visite son site web. Les données financières étant sous license, cette extension sera accessible via une identification (login/password) sur le serveur de C-Radar.

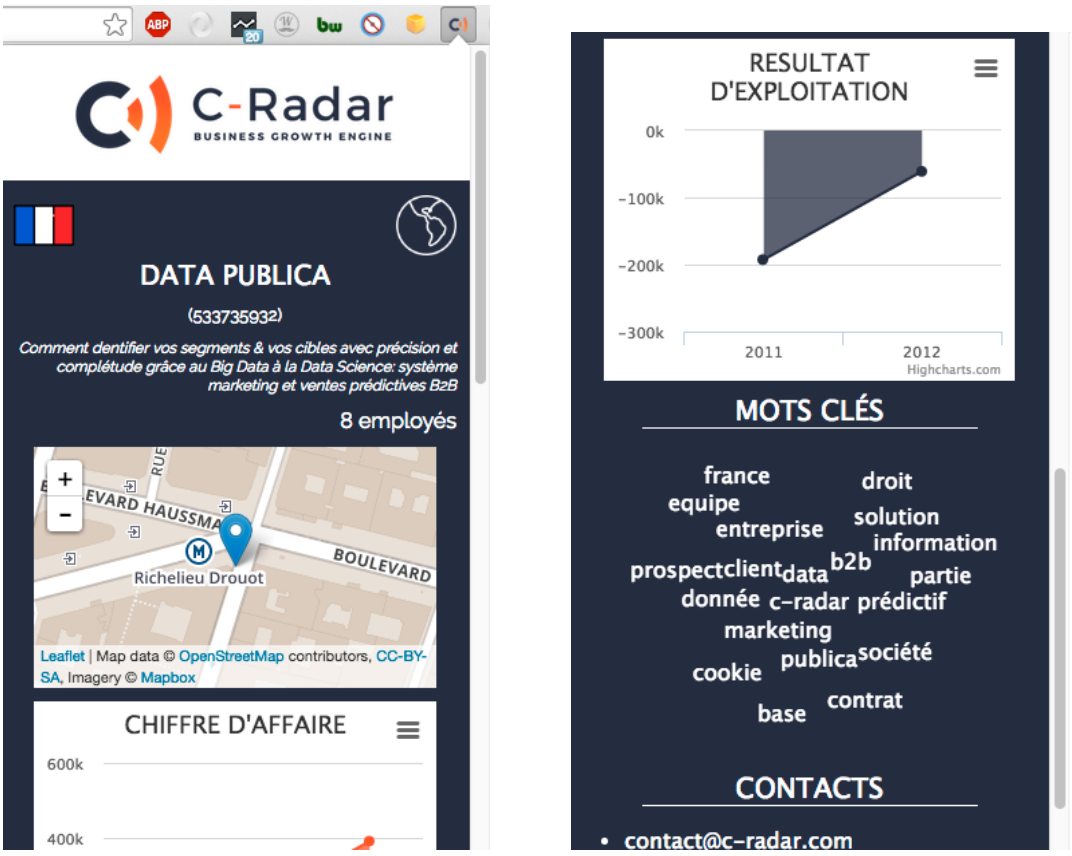


Figure 28: Screenshot du design de l'extension

3.3.1 Description technique

Les extensions Chrome sont basées sur les technologies web (HTML, CSS, JavaScript) et reposent sur un principe simple. Un fichier `manifest.json` qui permet de définir les informations de base de l'extension : nom, logos, version, etc. Ce fichier permet aussi de définir les permissions nécessaires à l'extension. Par exemple, la permission d'accéder aux url des différents onglets, de réaliser des requêtes HTTP, etc.

Le comportement et le design de l'extension sont respectivement définis par du code JavaScript et des fichiers HTML, CSS.
Pour identifier l'entreprise correspondante à l'url visitée et pour récupérer les informations principales de celle-ci, on effectue deux appels d'API depuis une requête HTTP asynchrone avec l'XMLHttpRequest de JavaScript. Un appel permet de d'identifier le SIREN correspondant à l'url, l'autre permet de récupérer les données sous la forme d'un fichier JSON.

On crée ensuite les graphiques correspondant au chiffre d'affaire et au résultat d'exploitation.

Pour cela on utilise la bibliothèque Highchart. Elle permet de faire simplement des graphiques interactif en JS.

Un exemple de fichier de configuration est donné ci-dessous.

```
option_chart = {
              "chart": {
                "type": 'area'
              },
              "title": {
                  "text": "Résultat d'exploitation de Data Publica",
              },
              "xAxis": {
                  "categories": [2011,2012, 2013, 2014, 2015]
              },
              "yAxis": {
                  "title": {
                      "text": ""
                  }
              },
              "legend": {
                  "layout": 'vertical',
                  "align": 'right',
                  "verticalAlign": 'middle',
                  "borderWidth": 0
              }
          }
```

A titre d'exemple, le code Python ci-dessous produit un graphique Highchart. Ce graphique est présenté à titre d'exemple. Dans l'extension il sera produit par du code JS.

```
from highcharts import Highchart
chart = Highchart()
chart.add_data_set([-192669,-60999,36508,-24997, -2767],series_type='area',
           threshold=0,showInLegend=False,name='Example Series',
           color='#f75531',negativeColor= '#262B40')
chart.set_dict_options(option_chart)
chart
```

```
Out[38]: <highcharts.highcharts.highcharts.Highchart at 0x2bb22a81d68>
```

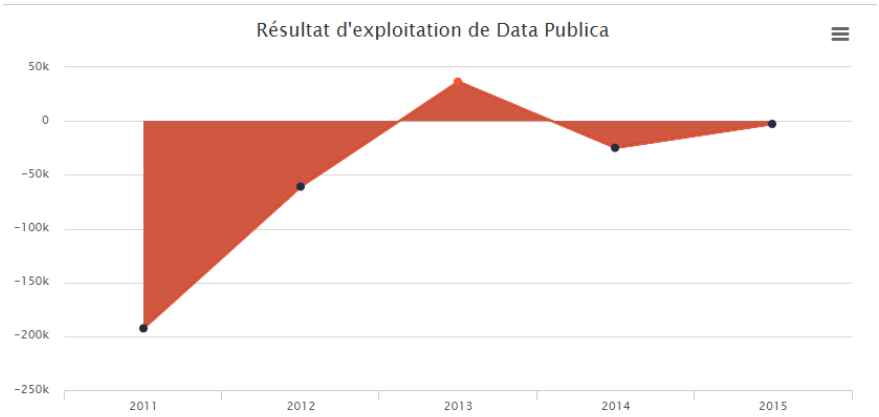


Figure 29: Highchart représentant le résultat d'exploitation de Data Publica

Les informations affichées sont : la description de l'entreprise, le nombre d'employés, le pays d'origine, sa présence à l'international, etc. Un nuage de mots (Cf. 28) a été également mis en place, permettant d'afficher les mots les plus présents sur le site.

Pour la sécurisation des données, un système d'authentification est utilisé. Un appel d'API permet de tester la validité des informations de login/password. En cas de succès, l'API renvoie un Cookie qui permet de garder la connexion active pendant une durée définie par le point d'API. Le Cookie expire 24h00 après avoir été délivré. Cela permet une navigation agréable pour l'utilisateur tout en gardant une sécurité optimale puisque le mot de passe et le nom d'utilisateur ne sont pas stockés par le navigateur.

3.4 Analyse des Doublons

3.4.1 Description du problème

Il existe un problème récurrent dans l'analyse sémantique des sites d'entreprises : l'association du site web au SIREN correspondant. Pour certaines d'entre elles il est difficile de le déterminer avec certitude, lorsque le nom ne permet pas une requête pertinente (nom de l'entreprise non discriminant, adresse ambiguë, etc.).

L'association entre un site et un SIREN prend en compte diverses informations : la présence du SIREN, l'adresse, numéro de téléphone, la distance sémantique13 entre le domaine et le nom de l'entreprise, etc. Un score est pour cela produit, et on considère que l'association est valide s'il dépasse un seuil fixé à partir de données d'apprentissage.

Le modèle utilisé pour calculer le score est un modèle linéaire dont les pondérations ont été mise en place de façon empirique. Des algorithmes ont permis de montrer qu'elles étaient cependant quasi-optimales. Il est probable que le score pourrait être amélioré avec un modèle non-linéaire, généralement plus performant.

Pour chaque entreprise on dispose de 3 scores associés aux 3 url retournées par la requête Bing. Les faux positifs sont minimisés avec un algorithme dédié. Le score obtenu est comparé au seuil pour déterminer l'association. Cependant, il est possible que plusieurs entreprises soient associées au même site web.

Dans certains cas cette association multiple est légitime. A l'inverse, certaines entreprises sont associées à un mauvais site web. Cela pose de gros problèmes pour les algorithmes qui se basent sur l'analyse sémantique puisque le texte analysé ne correspond pas à l'entreprise. Aujourd'hui, 61 379 sites web sont associés à plusieurs entreprises ce qui correspond à 194 869 entreprises associées à plusieurs sites web.

ci-dessous le code python permettant d'afficher le nombre de site web en fonction du nombre d'association :

```
df_site = pd.read_csv("./data/websites-raph.csv")
df_site_fr = df_site[df_site["_id"].str.contains("FR-")]
df_agg = df_site_fr.groupby("website.domain", as_index=False).agg({"Number":sum})

def plot(amplitude):
    value = amplitude
    fig, ax = plt.subplots()
    fig.set_size_inches(12, 6)
    df_agg[df_agg.Number>1].hist(range=[1, 40], bins=40, ax=ax)
    y, x = np.histogram(df_agg[(df_agg.Number>1) &(df_agg.Number <=40)]["Number"],
    #ax2 = ax.twinx()
    ax.plot(0.5 * (x[1:] + x[:-1]), np.cumsum(y), "r-", label="Somme cumulée")
    ax.set_xlim(1, 40)
    ax.axhline(sum(y[0:40]))
    ax.axhline(sum(y[0:value-1]))
    ax.axvline(value)
```

```
    pourcentage = int((1-(sum(y[0:40])-sum(y[0:value-1]))/sum(y[0:40]))*100)
    ax.text(10, sum(y[0:40])/2, str(pourcentage)+"%", size=30, va="center", ha="ce
            bbox=dict(alpha=0.2))
    ax.legend()
```

Les simulations ont été conduites dans l'environnement Jupyter Notebook. Il est possible de rendre le graphique intéractif avec un slider dans un widget. On voit le pourcentage de site en fonction du nombre d'association.

```
In [42]: a_slider = widgets.IntSlider(min=0, max=10, step=1, value=5)
         w=widgets.interactive(plot,amplitude=a_slider)
         display(w)
```

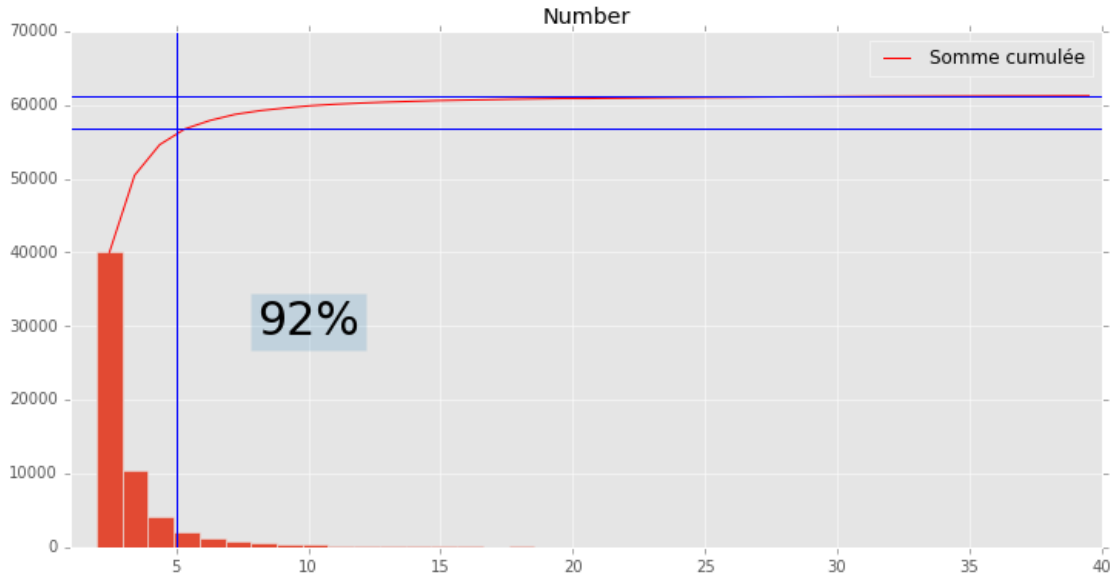


Figure 30: Nombre de site web vs nombre d'entreprises

On constate que 92% des associations concerne les sites web associés à moins de 4 entreprises.

On peut décomposer les associations multiples en trois catégories :

- Catégorie 1 : Les groupes et les franchises. Toutes les entreprises sont effectivement rattachées au même site web (Ex: Mcdonald's, Crédit Mutuel). Ces associations sont légitimes.
- Catégorie 2 : Les annuaires professionnels (Ex : PagesJaunes Pro). Ils référencent avec une même url un grand nombre d'entreprises. Le nombre d'annuaires étant assez limité, une solution simple et efficace est de les blacklister pour éviter qu'ils ne soient pris en compte dans la détection.
- Catégorie 3 : Tous les autres associations multiples. Dans le cadre de ma mission seules les doubles associations ont été étudiées. On peut à nouveau les décomposer en 3 catégories. Considérons 2 entreprises A et B et un site web W :

- A ou B est correctement associée à W ;
- A et B sont correctement associées à W ;
- ni A ni B ne sont correctement associées à W.

Dans un premier temps, on va chercher à identifier les différentes catégories. Et ensuite nous chercherons une solution au problème posé par la catégorie 3.

3.4.2 Détection du type de double association

Pour délimiter les 3 catégories un modèle a été mis en place. A partir de la sémantique du site web il s'agit de détecter si ce site va être associé à un ou plusieurs numéros de SIREN. Pour cela on crée un ensemble d'apprentissage de 2214 entreprises :

- 1922 sites "uniques" associés à une seule entreprise ;
- 147 sites "multiples" (soit site de groupe ou de franchise) ;
- 145 annuaires.

Pour construire le modèle on utilise la matrice sémantique (contenant plus d'un million de termes) des différents sites web. On fait une sélection de variables de type Chi2 pour réduire la dimensionnalité de la matrice et on garde uniquement les 2500 termes les plus discriminants.
On met ensuite en place un algorithme de Forêt aléatoire (Cf. partie 3.1.5) pour classifier les différentes catégories de sites web.

Le code ci-dessous permet de mettre en place une Random Forest :

```
df_prediction_type_site =  pd.read_csv("./data/df_for_predict_type_site.csv")
y = np.array(df_prediction_type_site["Catégorie"])

X_new = np.load("./data/X_new.npy")

skf = StratifiedKFold(y, n_folds=4)
y = np.array(y)
predict = []
predict_proba = []
y_true = []
for train_index, test_index in skf:
    X_train, X_test = X_new[train_index], X_new[test_index]
    y_train, y_test = y[train_index], y[test_index]
    clf = RandomForestClassifier(n_estimators=500, max_features=250, n_jobs=2)
    clf.fit(X_train,y_train)
    predict.extend(clf.predict(X_test))
    predict_proba.extend(clf.predict_proba(X_test))
    y_true.extend(y_test)

print(confusion_matrix(y_true, predict))
print(classification_report(y_true, predict))
```

```
    Matrice de confusion

[[  47   24   74]
 [  20   75   52]
 [   7    3 1912]]

    Rapport de classification
              precision    recall  f1-score   support

   Annuaire       0.64      0.32      0.43       145
   Multiple       0.74      0.51      0.60       147
     Unique       0.94      0.99      0.97      1922

avg / total       0.90      0.92      0.91      2214
```

Les résultats ci-dessus montrent que pour la catégorie "Annuaire" seuls 32% sont détectés comme annuaires ; ils représentent 64% des sites classifiés comme annuaires.

Ce premier modèle permet donc de distinguer les trois catégories mais les performances ne sont pas satisfaisantes pour la détection des annuaires et des multiples associations. Un autre modèle permettant de décomposer le problème en seulement deux catégories a été mis en place: les associations uniques et les associations multiples.

Le code ci-dessous permet de réaliser cette opération.

```
y_light = np.array(["multiple" if elt in ["Annuaire","Multiple"] else "unique" for
 skf = StratifiedKFold(y_light, n_folds=3)

 predict = []
 predict_proba = []
 y_true = []
 #for train_index, test_index in tqdm(skf):
 for train_index, test_index in skf:
     X_train, X_test = X_new[train_index], X_new[test_index]
     y_train, y_test = y_light[train_index], y_light[test_index]
     clf = RandomForestClassifier(n_estimators=500, max_features=250, n_jobs=4)
     clf.fit(X_train,y_train)
     predict.extend(clf.predict(X_test))
     predict_proba.extend(clf.predict_proba(X_test))
     y_true.extend(y_test)
```

```
print(confusion_matrix(y_true, predict))
print(classification_report(y_true, predict))
```

```
    Matrice de confusion :

[[ 173  119]
 [  19 1903]]

    Rapport de classification :

             precision    recall  f1-score   support

   multiple       0.90      0.59      0.71       292
     unique       0.94      0.99      0.97      1922

avg / total       0.94      0.94      0.93      2214
```

seulement 59% des associations multiples sont détectées.

Les performances sont meilleures. La précision et le rappel sur la classe "unique" sont satisfaisants contrairement au rappel sur la classe "multiple". Cela permet quand même de bien discriminer la classe "unique" et d'appliquer une solution spécifique à cette catégorie.

Ci-dessous les courbes de précision et rappel.

```
y_true_clean = [1 if elt=="unique" else 0 for elt in y_true]

create_precision_recall_curve(y_true_clean, predict_proba)
```

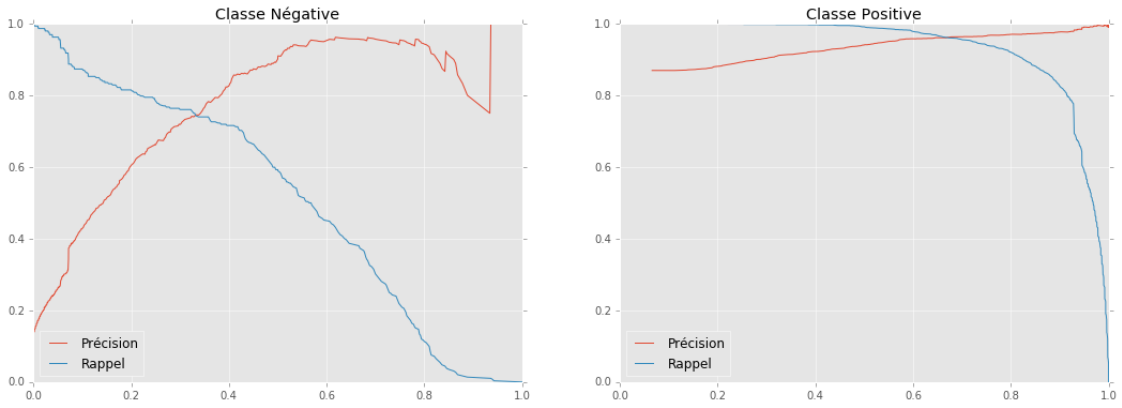


Figure 31: Courbes de précision rappel

Ici la classe négative est un site web pouvant avoir potentiellement plusieurs associations et la classe positive, un site associé à une seule entreprise.

Sur le graphique de la classe positive, pour un seuil à 0.8, la précision est de 0.98 et le rappel de 0.90. Ce qui signifie que les sites de l'échantillon devant être associés à une seule entreprise sont presque parfaitement détectés.

3.4.3 Détection de bonne association

En observant simplement la dépendance site web $\leftrightarrow$ entreprise déduite du scoring, on ne peut pas forcément conclure sur l'association correcte des deux. Considérons deux entreprises A et B et un site web W. Dans ce qui précède, seules les interactions A $\leftrightarrow$ W et B $\leftrightarrow$ W ont été prises en compte, et de façon indépendante. Ces interactions sont basées sur des informations juridiques "sûres" (provenant de bases de données d'entreprises) et sur les informations tirées de chacun des sites web.
Les résultats obtenus ne sont pas entièrement satisfaisants.

L'idée est alors d'étudier également l'interaction (A $\leftrightarrow$ B) $\leftrightarrow$ W.
Pour cela une validation manuelle a été effectuée sur 500 sites web associés à deux entreprises. Chaque double association a été catégorisée :

- Catégorie 3 : les deux entreprises sont bien associées ;
- Catégories 1 ou 2 : l'une ou l'autre est bien associée ;
- Catégorie 0 : aucune des deux n'est bien associée.

Le problème peut être alors modélisé sous la forme : "Sachant la deuxième association, la première est-elle légitime ?". Dans cette représentation du problème "Oui" sera notre classe positive et "Non" notre classe négative.

Après que l'ensemble d'apprentissage ait été formé, il faut déterminer les métriques qui pourraient permettre de différencier les catégories ci-dessus. Pour cela, on intègre les métriques utilisées dans le score d'association. Pour aller plus loin, des variables modélisant la différence entre les deux associations ont été créées :

- la différence de score d'association ;
- la distance sémantique entre les noms des deux entités.
- la différence de présence des variables : ville, adresse, numéro de téléphone, code postal, etc.

Une normalisation du nom des sociétés a été mise en place pour gérer le cas des éventuelles particules (SARL, EURL, GROUP, etc.).
Le code ci-dessous permet de mesurer les performances du modèle `RandomForestClassifier`.

```
df_for_prediction = pd.read_csv("./prediction_bonne_association.csv")
```

```
from sklearn.preprocessing import StandardScaler
def createModel(df):
    X = df.drop(["Comp1_siren", "Comp2_siren", "Categorie"], axis=1).fillna(0)._ge
    #X =  X[["diff_score", "Comp1_association_score"]]
```

```
    X=StandardScaler().fit_transform(X)
    y = np.array(df.Categorie)
    predictions_array = []
    ytrue_array = []
    predict_proba_array = []

    cross_val = StratifiedKFold(y, n_folds=5)
    #cross_val = StratifiedShuffleSplit(y, n_iter=5, test_size=0.5)
    for train_index, test_index in cross_val:

    #for train_index, test_index in tqdm(cross_val):
        #X_train, X_test = X.iloc[train_index], X.iloc[test_index]
        X_train, X_test = X[train_index], X[test_index]

        y_train, y_test = y[train_index], y[test_index]

        clf = RandomForestClassifier(n_estimators=1000, n_jobs=4)
        clf.fit(X_train, y_train)
        predictions_array.extend(clf.predict(X_test))
        ytrue_array.extend(y_test)
        predict_proba_array.extend(clf.predict_proba(X_test))
    return clf, predictions_array, ytrue_array, predict_proba_array

mdl, pred, y_true, pred_proba = createModel(df_for_prediction)

print(confusion_matrix(y_true, pred))
print(classification_report(y_true,pred))
```

```
    Matrice de confusion :

[[422  75]
 [ 93 386]]

  Rapport de classification :

             precision    recall  f1-score   support

          0       0.82      0.85      0.83       497
          1       0.84      0.81      0.82       479

avg / total       0.83      0.83      0.83       976
```

On voit ici que 82% des mauvaises associations (Catégorie 0) sont correctement détectées.

On crée ici les courbes de précision rappel avec un seuil placé à 0.7.

```
In [28]: create_precision_recall_curve(y_true, pred_proba, 0.7)

Seuil 1 :
['0.89', '0.70']
['0.92', '0.69']
```

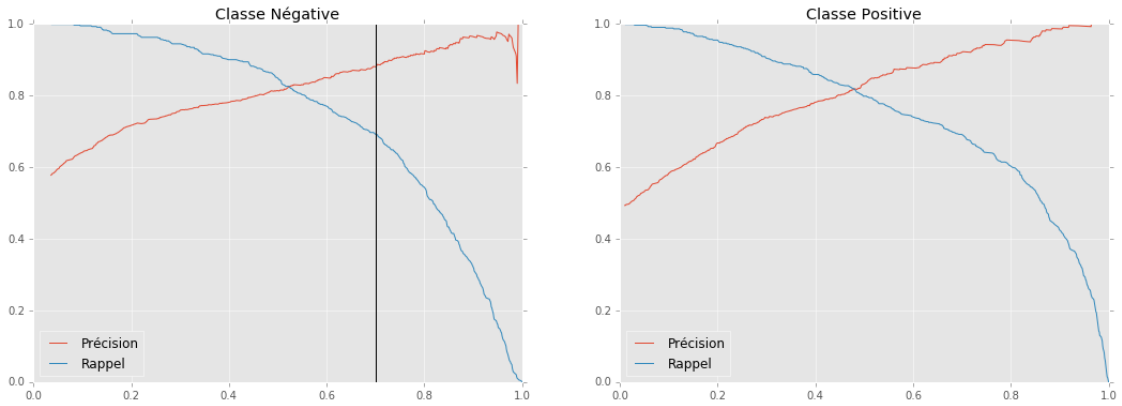


Figure 32: Courbes de précision rappel

On voit ci-dessus que si l'on place le seuil de détection à 0.7, que la précision est de 89% avec un rappel de 70% sur la classe négative. On peut ici privilégier la précision sur le rappel puisqu'on veut minimiser les faux positifs et donc éviter de désassocier une entreprise à son site web à tort.

3.5 Classification de startups

La dernière semaine de mon stage a été consacrée à des missions de moindre importance que je me suis fixé en complète autonomie :

- Amélioration du score d'association des entreprises avec leur site web en ajoutant le matching entre les marques déposées et l'url. Par exemple, Data Publica a déposé la marque "C-Radar" et son site est www.c-radar.com et non www.datapublica.com. Au lieu de les détecter, on pourrait potentiellement réduire les mauvaises associations en améliorant le modèle linéaire ou en utilisant un modèle non-linéaire (quelques tests non concluants ont été réalisés).
- Amélioration du code Java du sirenizator dont nous avons parlé plus haut (Cf. partie 3.1.3).
- Continuation du travail d'une autre stagiaire qui avait travaillé sur la classification de startups.

Seul ce dernier point est développé dans ce qui suit.

L'identification du statut de startups peut être une réelle valeur ajoutée aux données de C-Radar. En effet, de nombreux clients (proposant des prestations d'aides aux jeunes entreprises par exemple) sont demandeurs d'annuaires de startups.

Pour commencer, l'ensemble d'apprentissage (initialement 300 exemples positifs) a été fortement augmenté en scrapant l'annuaire de startups de l'Usine Digitale. Les informations récupérées sont utilisées par le sirenizator pour trouver les SIREN. Il permet de réconcilier 1172 startups sur les 5000 récupérées.

Le nombre de variables explicatives a également été augmenté en ajoutant :

- certaines données récupérées pour le projet Ayming (Cf. partie 3.1) ;
- des données financières comme le chiffre d'affaire, ou les levées de fonds, etc ;
- les données concernant l'appartenance aux pôles de compétitivité ;
- les dépots de brevets.

Le code ci-dessous permet de mettre en place la classification de Startups à l'aide d'une forêt aléatoire.

```
df_final = pd.read_csv("./data/df_final_for_startups_prediction.csv")
 X = df_final._get_numeric_data().drop("Categorie", axis=1).fillna(0)
 y = np.asarray([elt == 1 for elt in df_final.Categorie.values])

y_true = []
 pred = []
 pred_proba = []

 #for train_index, test_index in tqdm(StratifiedKFold(y, n_folds=5)):
 for train_index, test_index in StratifiedKFold(y, n_folds=5):
     X_train, X_test = X.iloc[train_index], X.iloc[test_index]
     y_train, y_test = y[train_index], y[test_index]
     clf = RandomForestClassifier(n_jobs=-1, n_estimators=100, max_features=40)

     clf.fit(X_train, y_train)
     y_true.extend(y_test)
     pred.extend(clf.predict(X_test))
     pred_proba.extend(clf.predict_proba(X_test))

In [73]: print(classification_report(y_true, pred))
 print(confusion_matrix(y_true, pred))

    Rapport de classification :
              precision    recall  f1-score   support

      False       0.95      0.99      0.97     11372
       True       0.85      0.45      0.59      1172
avg / total       0.94      0.94      0.93     12544
```

```
Matrice de confusion :

[[11280    92]
 [  643   529]]
```

Ici seule 45% des startups sont détectées sur les 1172 du support d'apprentissage. Cependant le modèle est assez précis puisque seules 92 entreprises sont prédites comme des startups alors qu'elles ne le sont pas (Faux Positifs). Le seuil par défaut est 0.5.

La représentation graphique avec un seuil de 0.33 :

```
create_precision_recall_curve(y_true, pred_proba, x2=0.33)
```

```
    Rapport de classification avec le nouveau seuil :

['0.93', '1.00']
['0.71', '0.68']
```

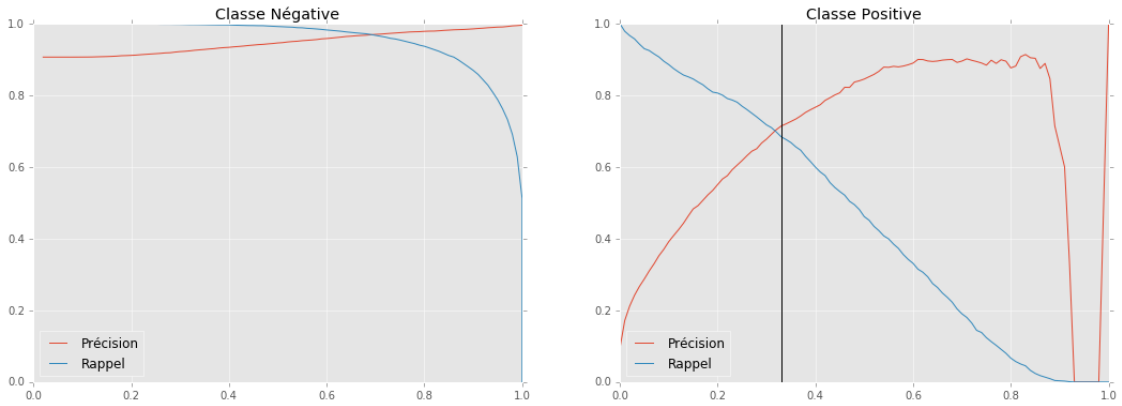


Figure 33: Courbes de précision rappel

On voit sur le graphique de droite que le rappel décroît très fortement, ce qui ne permet pas d'augmenter le seuil pour améliorer la précision. La ligne verticale permet de matérialiser le nouveau seuil (0.33).

Le code ci-dessous permet d'afficher les importances calculées des différentes variables les plus discriminantes.

```
df_features = pd.DataFrame({"Pond":clf.feature_importances_, "Name":X.columns})

fig, ax = plt.subplots()
df_to_plot = df_features.sort_values("Pond", ascending=False).head(15)
df_to_plot.plot(kind='barh', ax=ax)
ax.set_yticklabels(df_features.Name);
```

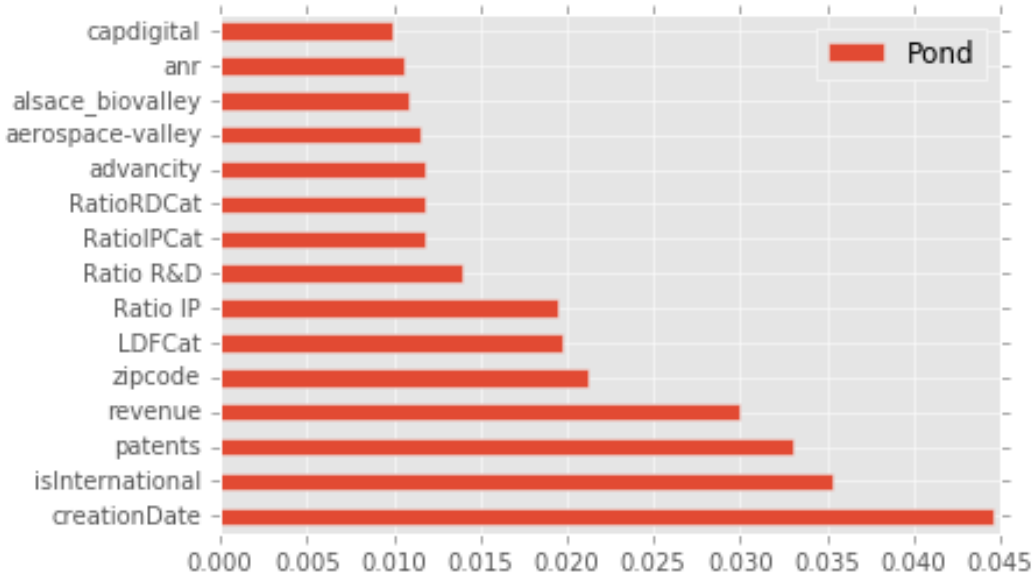


Figure 34: Importance des 20 variables les plus discriminantes triées par importance croissante

Parmi plusieurs milliers de variables, le graphique ci-dessus (34) présente les 20 variables les plus discriminantes. On voit ici que la date de création (`creationDate`), la présence à l'international (`isInternational`), les dépots de brevets (`patents`), etc. sont des variables discriminantes. Le chiffre d'affaire (`revenue`), les levées de fond (`LDFCat`), l'appartenance à des pôles de compétitivité (`capdigital, advancity, etc.`) sont aussi utilisés. Ici l'importance d'une variable ne correspond pas à un coefficient que l'on pourrait avoir avec un modèle linéaire, mais à une importance moyenne de chaque variable dans la création du modèle.

82% des variables ont une importance inférieure à 0.001 dans la décision. Certaines d'entre elles ont même un coefficient nul. Ce qui traduit une importance moindre voire nulle dans la prédiction. Comme l'algorithme de Random Forest utilise un ensemble aléatoire de variables explicatives, il est souvent intéressant de ne garder que les plus pertinentes. Pour conserver uniquement les variables ayant une importance supérieure à 0.001 :

```
X_light = X[df_features[df_features.Pond>0.001].Name]
 y_true = []
 pred = []
 pred_proba = []

 #for train_index, test_index in tqdm(StratifiedKFold(y, n_folds=5)):
 for train_index, test_index in StratifiedKFold(y, n_folds=5):
     X_train, X_test = X_light.iloc[train_index], X_light.iloc[test_index]
     y_train, y_test = y[train_index], y[test_index]
     clf = RandomForestClassifier(n_jobs=-1, n_estimators=100, max_features=4)
     clf.fit(X_train, y_train)
     y_true.extend(y_test)
     pred.extend(clf.predict(X_test))
     pred_proba.extend(clf.predict_proba(X_test))
```

```
print(classification_report(y_true, pred))
print(confusion_matrix(y_true, pred))
```

```
    Rapport de classification :

             precision    recall  f1-score   support

      False       0.94      0.99      0.97     11372
       True       0.86      0.42      0.56      1172
avg / total       0.93      0.94      0.93     12544

    Matrice de confusion :
[[11289    83]
 [  682   490]]
```

La précision de la classe positive augmente d'un point au détriment de la classe négative (0.86 au lieu de 0.85).

Le modèle est performant sur la précision : très peu d'entreprises sont prédites comme des startups alors qu'elles ne le sont pas (83/12544). Cependant le rappel n'est pas suffisant : de nombreuses de startups ne sont pas détectées. On peut améliorer légèrement la précision en augmentant légèrement le seuil de détection mais le rappel diminue extrêmement rapidement (Figure 35) :

```
create_precision_recall_curve(y_true, pred_proba, x2=0.6)
```

```
    Rapport de classification avec le nouveau seuil :
['0.96', '0.98']
['0.88', '0.28']
```

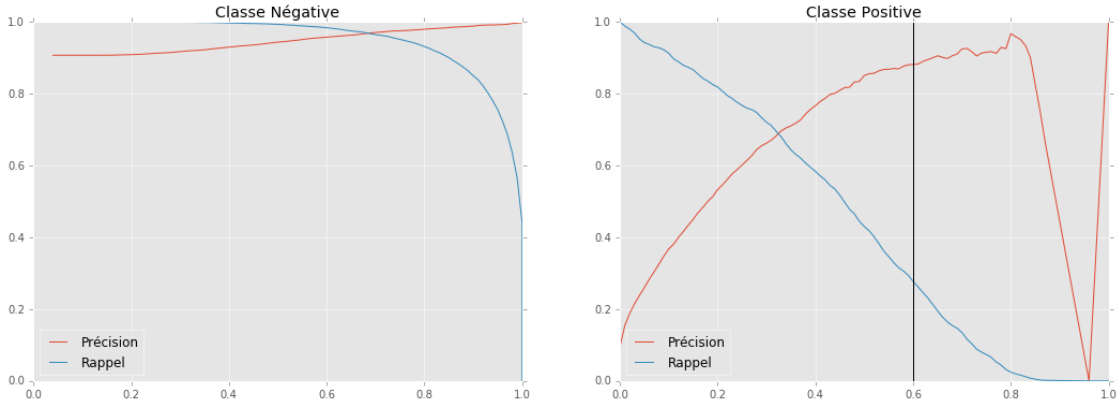


Figure 35: Courbes de précision rappel

En plaçant le seuil à 0.6, la précision passe à 0.88 mais seulement 28% des startups sont détectées. Ces performances ne sont pas suffisantes pour être intégrées dans le produit.

4 Bilan

Après ces quatre mois de stage, je peux faire un point sur toutes les compétences aussi bien techniques que relationnelles et organisationelles que j'ai eu la chance de développer au sein de Data Publica. Le cadre et l'ambiance de travail étaient idéaux pour grandir dans ce métier et dans le secteur de la Data Science. Je faisais partie intégrante d'une équipe R&D de 7 personnes ayant chacun leur domaine d'expertise: architecture, langages Java et Python, analyse de données, etc.

4.1 Compétences techniques :

Data Science : J'ai pu mettre en oeuvre tous les aspects du métier de Data Scientist, de l'extraction et la récupération de données, au nettoyage et l'extraction de métriques pertinentes, en passant par la mise en place d'algorithme de **machine learning**, jusqu'à la Data Visualisation. J'ai utilisé, pour toutes les analyses de données, le package Python `Scikit-Learn` qui est un projet Open Source très bien documenté, que nous avons déjà assez bien parcouru lors de nos enseignements. La plupart des algorithmes relatifs au machine learning y sont implémentés et assez faciles à mettre en oeuvre. J'ai pu aussi explorer de nombreux algorithmes afin de comprendre en détails leur fonctionnement pour les mettre en oeuvre efficacement et de pouvoir détecter plus facilement les comportement anormaux, lorsqu'il y en a.

J'ai pu aussi mettre en place des fonctions permettant de faciliter et d'améliorer la productivité des analyses. Ces fonctions permettaient de tester très facilement différents algorithmes de classification (**RandomForest, Arbres boostés, SVM, Regressions Lineaires**, etc.) et d'afficher les métriques de bases, afin de déterminer les performances de chacun d'entre eux.

J'ai eu la chance de participer dans le cadre de mon stage à la conférence PyData 2016 et au Scikit-Learn Day dans les locaux de l'ESILV (École Supérieure d'Ingénieurs Léonard de Vinci) animés par les créateur du package.

Python : J'ai pu consolider et énormément progresser sur la connaissance du langage Python. J'ai notamment travaillé sur des questions d'**optimisation de mémoire** lors de l'exécution de scripts par exemple. Même si nous travaillons la plupart du temps depuis des machines distantes par le biais d'une connexion **SSH** pour faire tourner des **notebook Jupyter**, les grands volumes de données ne pouvaient pas être manipulés et montés en mémoire facilement. De plus, ces machines étaient munis d'un grand nombre de processeurs, j'ai donc pu mettre en place des scripts basés sur le **multiprocessing**[15], permettant de paralléliser les calculs.

J'ai aussi pu apprendre les **bonnes pratiques** de la programmation, en mettant en oeuvre un **package Python**. Le package comprend la détection de technologies web et la mise en place des différents outils pour la mise en production du package, comme la **génération de logs**, ou un système permettant de prendre en considération toutes les **exceptions** ou erreurs pouvant subvenir. J'ai dû aussi suivre toutes les étapes de la création d'un package Python permettant de le rendre utilisable automatiquement et le plus facilement possible.

Systèmes d'information : J'ai aussi acquis des compétences de travail en équipe, j'ai dû me former à l'utilisation de **git** qui permet gérer les versions d'un projet. J'ai aussi pu appliquer tous les enseignements de sécurité que j'ai suivis durant ma scolarité, en utilisant un environnement de serveur et d'**authentification SSH**.
Enfin j'ai pu confirmer les connaissances en bases de données que j'avais acquise, en utilisant très

[15] Permet une optimisation du temps de traitement en utilisant plusieurs coeurs

souvent des bases de données **MongoDB**. J'ai également dû manipuler une base **Cassandra** sans avoir besoin de maitriser le sujet. J'ai aussi eu l'occasion de travailler sous une infrastructure **Docker** et d'apprendre à utiliser cette technologie.

Technologies Web : J'ai aussi pu améliorer mes connaissances dans les technologies web comme le **HTML** et **CSS** mais aussi le **JavaScript**. Lors de deux différents projets j'ai pu parcourir le langage JavaScript que j'avais déjà quelque peu découvert lors de la création de mon site web personnel. J'ai pu vraiment aborder ce sujet lors de la création d'une **extension Chrome** et lors de la modification d'un **projet Open Source** déjà existant (`Wappalyzer`). J'ai pu mieux comprendre le fonctionnement, les avantages et le contexte d'utilisation de ce langage qui, combiné aux technologies HTML et CSS, permet de créer des **interfaces web** attractives. La création de l'extension Chrome m'a permis de mettre en place une interface utilisateur en HTML et CSS, permettant de faire de la **Data Visualisation** dans le contexte web avec différents librairies JS (**HighCharts** pour les graphiques et **ICloud** pour le nuage de mots). J'ai utilisé la connaissance de ces technologies pour réaliser un scraper efficace.

4.2 Compétences relationnelles et organisationnelles :

J'ai été suivi tout au long de mon stage par un docteur es informatique et data scientist, Clément Chastagnol. En plus de m'avoir aidé à consolider mes bases en analyse de données, il m'a surtout permis d'acquérir une méthodologie et une démarche pour l'analyse des problèmes qui m'ont été posés. J'ai ainsi pu apprendre à formaliser un problème et à avoir une démarche structurée pour le résoudre.

J'ai eu la chance de travailler avec lui sur l'entièreté d'un projet client et donc de participer à toutes les étapes de la conception à la restitution. Concernant la restitution, j'ai participé à, et préparé, plusieurs réunions clients. Cela m'a permis d'apprendre à formaliser mes conclusions, et à les présenter.

Enfin, l'ambiance au sein de l'équipe ainsi que la motivation et la compétence de tous les membres permet de créer une vraie dynamique de travail. Il n'est jamais difficile d'obtenir de l'aide quel que soit le domaine. Cela permet de ne pas rester bloquer et d'avancer assez rapidement. Chacun a son domaine d'expertise qu'il sait faire partager pour en faire profiter toute l'équipe. Pour ma part, j'ai dû régulièrement faire un point d'avancement sur mon travail devant toute l'équipe pour valider les choix effectués.

Conclusion

Après une période de formation, j'ai passé mes deux premiers mois sur un projet client. Ce projet était une étude de cas visant à qualifier de futurs clients potentiels d'une société de conseil aux entreprises. J'ai pu faire le lien entre ce que j'ai appris en cours et les réalités du terrain. J'ai pu consolider les bases que j'avais acquises en Data Science et pu progresser sur la restitution d'un projet complexe. La dernière réunion se déroulera en septembre où les responsables commerciaux seront présents pour expliquer les tenants et les aboutissants du projet et l'intérêt que le client a, à utiliser nos travaux.

A la fin de cette période, j'ai pris part à un autre projet client : le projet BPost SME. Contrairement au projet précédant, ce projet a permis de développer C-Radar en enrichissant la base de données sur les entreprises. Ma contribution à ce projet à été de rechercher une solution pour détecter les technologies utilisées sur les sites web des entreprises. Ce projet est effectif et prend part dans le workflow de l'application C-Radar. Il est intégré au Front-End sur la fiche entreprise et permet de faire des recherches filtrées par technologies utilisées.

Dans une phase de transition j'ai pris l'initiative de créer une extension Chrome permettant d'afficher les données de C-Radar lorsque l'utilisateur visite le site web d'une entreprise. Cette extension est prête et fonctionne correctement mais il faudra un peu de temps pour pouvoir l'intégrer à l'offre proposée. Il subsiste cependant un problème de données sous licence (bilans financiers par exemple) ce qui interdit pour le moment de rendre disponible l'extension sur le Chrome Store.

Finalement, dans le temps qu'il me restait j'ai travaillé sur l'analyse des associations entre sites web et entreprises. Cette étude a permis à l'équipe d'avoir un aperçu qualitatif et quantitatif des problèmes existants. Un peu de travail est nécessaire pour finaliser mes travaux et rendre opérationnel cette fonctionnalité.

Ce stage m'a permis de confirmer mon intérêt pour la Data Science et l'analyse de données en étant immergé dans un environnement professionnel entièrement tourné vers cette problématique.

Annexes

Les annexes ci-dessous présentent quelques exemples de code ainsi que le poster présenté lors de la journée des projets.

.1 Main.py du package de détection de technologies

```
import json
from companies_plugin import extractor

from comp_webanalyzer.webanalyzer import BrowserHandler
from comp_webanalyzer import LIB_PATH


class Extractor(extractor.Extractor):
    def __init__(self, batch_name, wanted_fields, conf):
        """
        The configuration MUST have a misc called "webanalyzer" with the fields
        required by BrowserHandler

        :param batch_name:
        :param wanted_fields:
        :param conf:
        :return:
        """
        super().__init__(batch_name, wanted_fields, conf)
        self.browser_handler = BrowserHandler(conf.misc["webanalyzer"])

    def extract(self, headers, properties, message):
        """
        Webanalyzer service. Take as input a {url: } document
        Returns {url: , png: <base64>, techno:dict}

        :param headers:
        :param properties:
        :param message:
        :return:
        """
        reply_to = properties["reply_to"]
        msg = json.loads(message)

        url = msg["url"]

        is_reached = self.browser_handler.get_page(url)
        if is_reached:
            png = None
            techs = self.browser_handler.get_dict_of_tech(url)
            if techs is not None:
```

```
                png = self.browser_handler.get_screenshot(url)

            result = {
                "url": url,
                "png": png,
                "techno": techs
            }
        else:
            result = {
                "url": url,
                "png": None,
                "techno": None
            }
        self.browser_handler.reset()

        return json.dumps(result), reply_to


class Main(extractor.Main):
    pass


if __name__ == "__main__":
    m = Main(batch_name="WEBANALYZER",
             queue_name="WEBANALYZER",
             extractor_class=Extractor,
             mod_path=LIB_PATH)
    m.launch()
```

.2 Webanalyzer.py du package de détection de technologies

```
import logging, os, ast, json
from selenium import webdriver
from selenium.common.exceptions import WebDriverException, TimeoutException,
 UnexpectedAlertPresentException
from selenium.webdriver.support.wait import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.common.by import By

class BrowserHandler:
    def __init__(self, webanalyzer_conf):
        self.logger = logging.getLogger("webanalyzer")
        self.logger.setLevel(logging.INFO)

        self.LIB_PATH = os.path.dirname(os.path.abspath(__file__))

        # Id of the div added to the html page source code, hardcoded in
                #the plugin driver
        self.div_id = "datapublicadiv"
        self.timeout_pageload = webanalyzer_conf["timeout"]
        self.delay = webanalyzer_conf["delay"]
        self.width = webanalyzer_conf["width"]
        self.height = webanalyzer_conf["height"]

        # path to Custom Extension
        self.option = webdriver.ChromeOptions()
        # Add Extension to Chrome Option to install it in Headless Browser
        self.option.add_extension(self.LIB_PATH + '/custom_wappalyzer/2.46_0.crx')
        self.option.add_extension(self.LIB_PATH + '/adblock.crx')
        self.browser = self.create_browser()
        self.reset()
        self.base_html_content = self.get_page_source()
        self.logger.info(self.base_html_content)

        # Get Categories labels form Categories.json
        self.categories = self.load_categories()
        self.logger.info("Browser up !")

    def create_browser(self):
        browser = webdriver.Chrome(chrome_options=self.option)
        browser.set_page_load_timeout(self.timeout_pageload)
        #From http://stackoverflow.com/questions/37181403/how-to-set-browser-viewpo
        window_size = browser.execute_script("""
            return [window.outerWidth - window.innerWidth + arguments[0],
              window.outerHeight - window.innerHeight + arguments[1]];
            """, self.width, self.height)
        browser.set_window_size(*window_size)
```

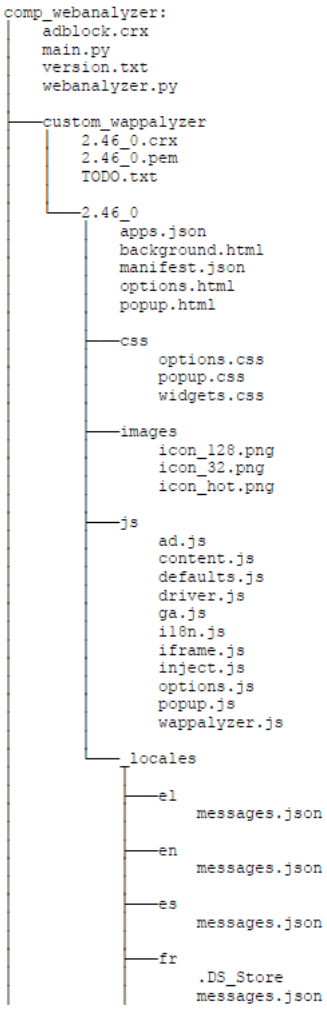
```
        self.tries_over_decount = 2
        return browser

    # Load page in the headless Browser
    def get_page(self, url, tries_crash=2, tries_timeout=1):

        if tries_timeout==1 and tries_crash==2 and
                                self.get_page_source() != self.base_html_content:
            self.logger.info("[%s] : Restarting Browser..." %  url)
            self.kill_browser()
            self.browser = self.create_browser()
            self.reset()
        if tries_crash == 0:
            self.logger.info("[%s] : Tries crash over" %  url)
            return False
        if tries_timeout == 0:
            self.logger.info("[%s] : Tries timeout over" %  url)
            return True
        self.logger.info("[%s] : Try to reach url ... " %  url)
        try:
            self.browser.get(url)
            self.logger.info("[%s] : Page Loaded." % url)
            return True
        except TimeoutException:
            self.logger.info("[%s] : Timedout" % url)
            # self.reset()
            return self.get_page(url, tries_crash=tries_crash,
                                        tries_timeout=tries_timeout-1)
        except WebDriverException as e:
            if "Message: chrome not reachable" in str(e):
                self.logger.info("[%s] : Browser crashed"% url)
                self.browser = self.create_browser()
                self.reset()
                self.logger.info("[%s] : Browser up"% url)
                self.logger.info("[%s] : Retry to reach the page..."
                                                                % url)
                return self.get_page(url, tries_crash=tries_crash-1,
                                        tries_timeout=tries_timeout)
            else:
                self.logger.warn(e)
                return False

    # Analyse Technologies on the page
    def get_dict_of_tech(self, url, tries_alert=2, tries_timeout=1):

        if tries_alert == 0 or tries_timeout==0:
            self.logger.info("[%s] : Tries over" % url)
            self.tries_over_decount -= 1
```

```
            return None
        self.logger.info("[%s] : Getting technologies...." % url)
        try:
            WebDriverWait(self.browser, self.delay).until(
                                EC.presence_of_element_located((By.ID, self.div_id)
            list_elems = self.browser.find_elements_by_id(self.div_id)
            string_of_tech = list_elems[-1].text
            dict_of_tech = ast.literal_eval(string_of_tech)
            final_dict = {tech: [self.categories[str(c)] for c in cat]
                                for tech, cat in dict_of_tech.items()}
            self.logger.info("[{!r}]: Techno detected : {!r}".format(url,
                                ", ".join(list(final_dict.keys()))))
            return(final_dict)
        except TimeoutException as e:
            self.logger.info("[%s] : Not able to find the div" % url)
            return(self.get_dict_of_tech(url, tries_alert=tries_alert,
                                tries_timeout=tries_timeout-1))
        except UnexpectedAlertPresentException as e:
            self.logger.info("[%s] : Alert detected" % url)
            alert = self.browser.switch_to_alert()
            alert.accept()
            self.logger.info("[{!r}] : Retry to get technos... ({!r} times)".format
                                tries_alert-1))
            return(self.get_dict_of_tech(url, tries_alert=tries_alert-1,
                                tries_timeout=tries_timeout))

    # Get dictoniary from categories.json
    def load_categories(self):
        with open(self.LIB_PATH + "/custom_wappalyzer/2.46_0/apps.json", 'r') as f:
            return json.load(f)["categories"]

    # get Page source code
    def get_page_source(self):
        return self.browser.page_source

    # Get Base64 string screenshot
    def get_screenshot(self, url):
        self.logger.info("[%s] : Shoot screenshot" % url )
        return self.browser.get_screenshot_as_base64()

    #Reset the tab with empty page
    def reset(self):
        self.logger.debug("Resetting the tab")
        self.browser.get('data:,')

    def kill_browser(self):
        self.logger.debug("Kill browser")
        self.browser.quit()
```

.3 Extrait du driver.js permettant d'insérer les technologies dans une div HTML

```
var resultstring='{';
for(app in w.detected[url]){
        resultstring += "'" + app + "':[" +  w.apps[app].cats + '] , ';}

var stringcode = 'var div = document.createElement("div");\
                        div.id = "datapublicadiv";\
                        div.style.fontSize = "0px";\
                        div.textContent = "' + resultstring +'}";\
                        document.body.appendChild(div);'
        chrome.tabs.executeScript(tab.id, { code:stringcode});
```

.4 Architecture du package de technologies web



.5 Poster présenté lors de la journée des projets

C-Radar
BUSINESS GROWTH ENGINE

29 Septembre 2016

Data Science au service de la Business Intelligence

RAPHAËL COURIVAUD

ESIEE PARIS

Plan

1. Présentation générale
    1. Contexte
    2. Présentation de l'entreprise
    3. Présentation de l'application
2. Data Science
    1. Data Mining : Text Mining
    2. Principaux outils et algorithmes de machine learning de l'application
3. Mes missions
    1. Etude de cas : qualification de prospects.
    2. Extraction de données : détection de technologies web.
    3. Affichage temps réel du profil web de l'entreprise.
    4. Analyse des doubles associations sites web <-> entreprises.
    5. Classification de Startups.

Présentation générale

STAGE DE QUATRIÈME ANNÉE

Présentation du contexte BI

Business Intelligence ou Informatique Décisionnelle

Définition : « a set of techniques and tools for the acquisition and transformation of raw data into meaningful and useful information for business analysis purposes. »

Source : Wikipédia

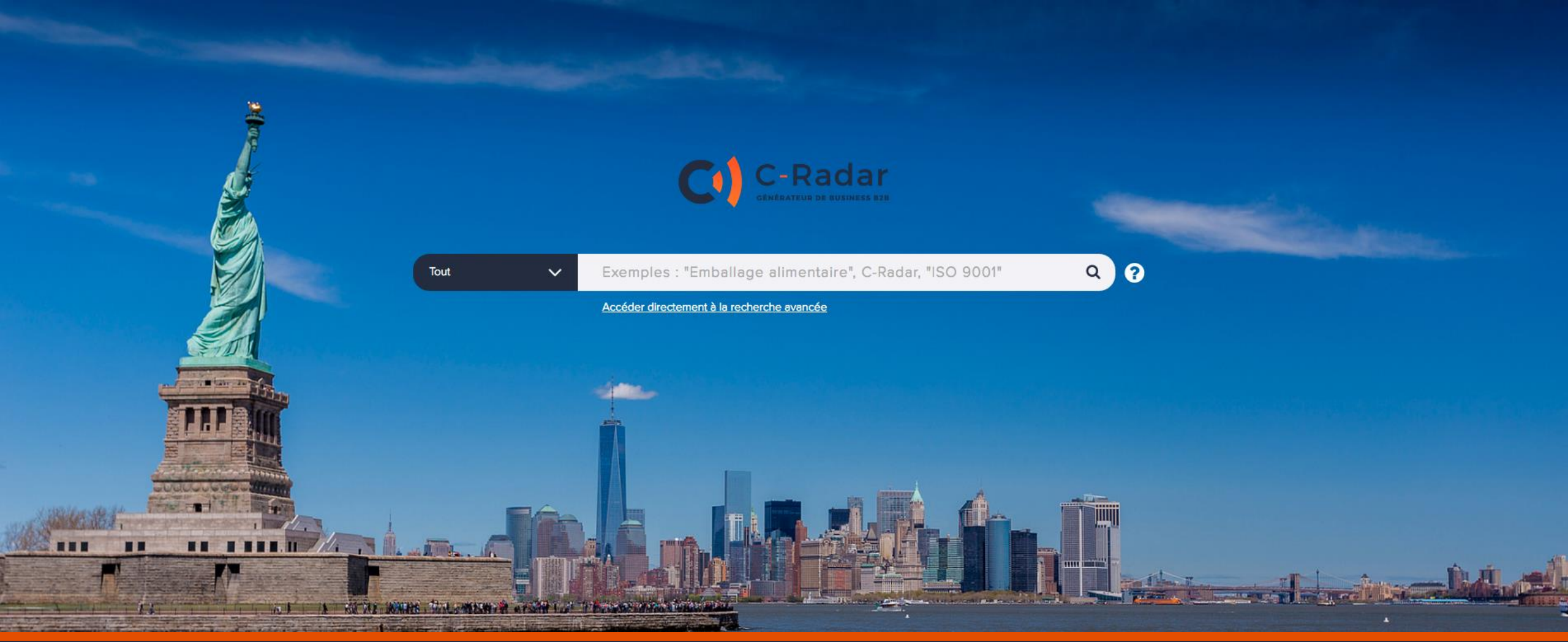
Présentation de l'entreprise

Historiquement Data Publica a mis en place le premier annuaire Open Data.

En janvier 2014, Data Publica lance l'offre C-Radar, la première offre de vente prédictive en France.

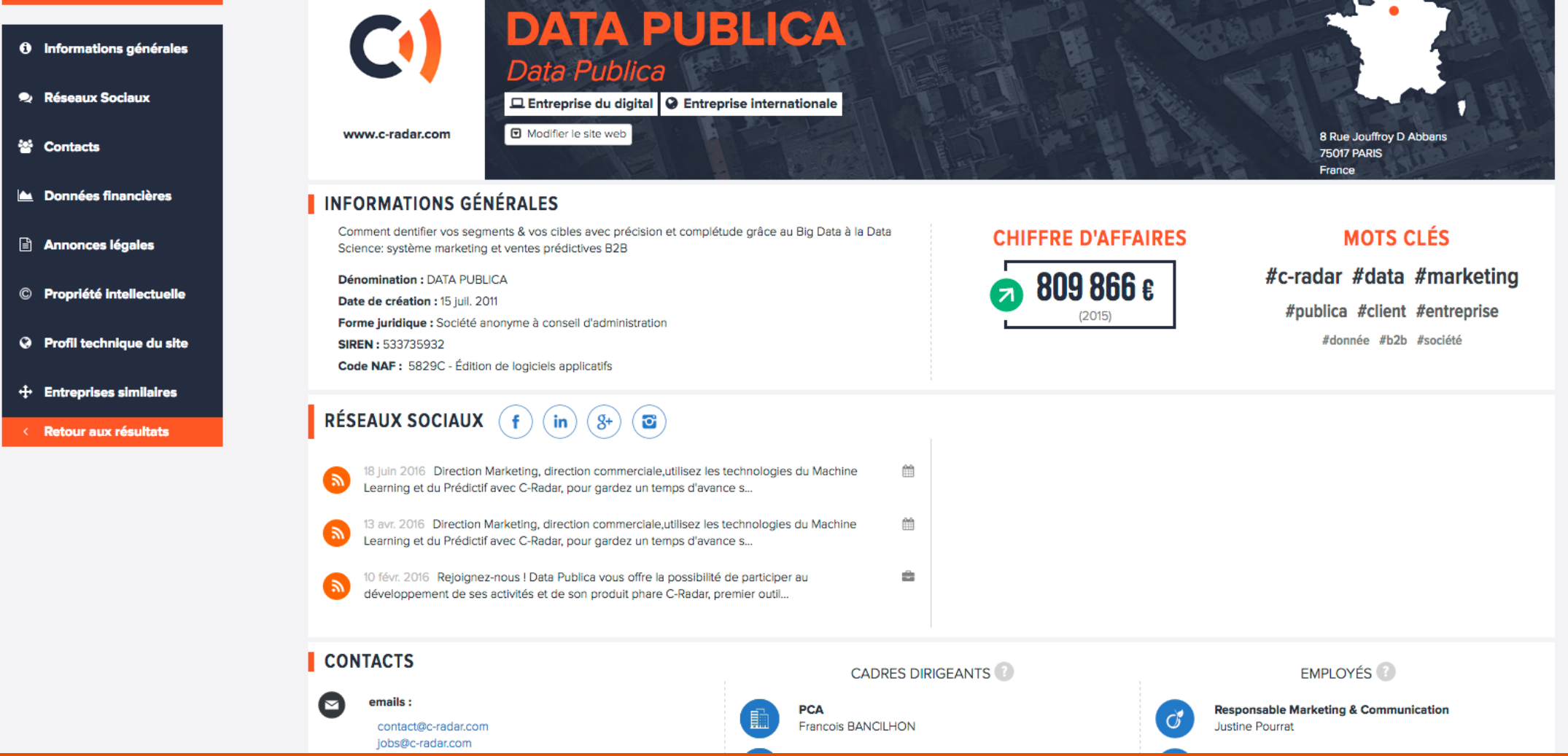
Data Publica participe à de nombreux projets de recherche comme Diachron ou ScanR en collaboration avec le MESR

DATA PUBLICA



L'application C-Radar

Un moteur de recherche B2B combiné à des outils de qualification de prospects utilisant le machine learning



Fiche entreprise

Permet de regrouper au même endroit des données : financières, sur les réseaux sociaux, de contacts, la propriété intellectuelle ainsi que les entreprises similaires sémantiquement

Les différentes étapes du processus d'analyse de données

Pour accéder à toutes ces données Data Publica est experte dans 4 domaines :

Web Crawling/Scraping :

Récupération de données (structurées ou non) depuis un site web.

Data Mining/Text Mining :

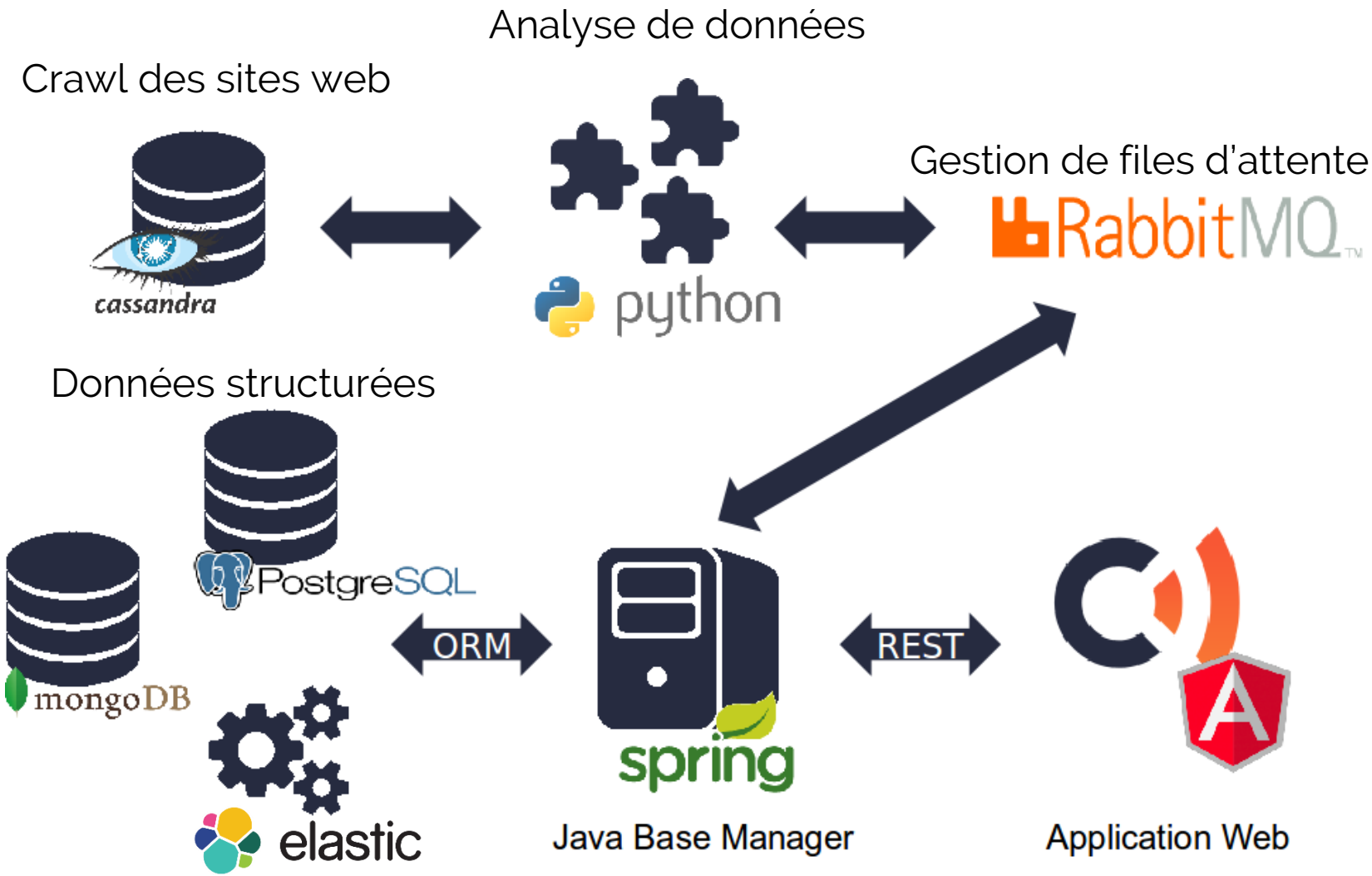
Extraction de valeur ajoutée de données brutes.

Machine Learning :

Utilisation des données récupérées et extraites pour mettre en place des algorithmes prédictifs.

Data Visualization :

Affichage de données complexes sous la forme de graphiques et visualisations attractifs et clairs.



L'architecture de l'application

Elle est constituée des technologies les plus récentes.

Data Science

PRÉSENTATION DES PRINCIPAUX ALGORITHMES DE L'APPLICATION

WordCount et Vectorisation :

Change la représentation du texte en vecteur correspondant au mot et son occurrence dans le texte.

Suppression des StopWords :

Les StopWords sont des mots ne donnant aucune information dans une phrase. (Ex : pronoms, auxiliaires, etc…)

Stemmer le vocabulaire :

On ne garde que la racine grammaticale du vocabulaire. Cela permet de réduire la dimension en supprimant les mots de même sens.

Calculer la normalisation TFIDF :

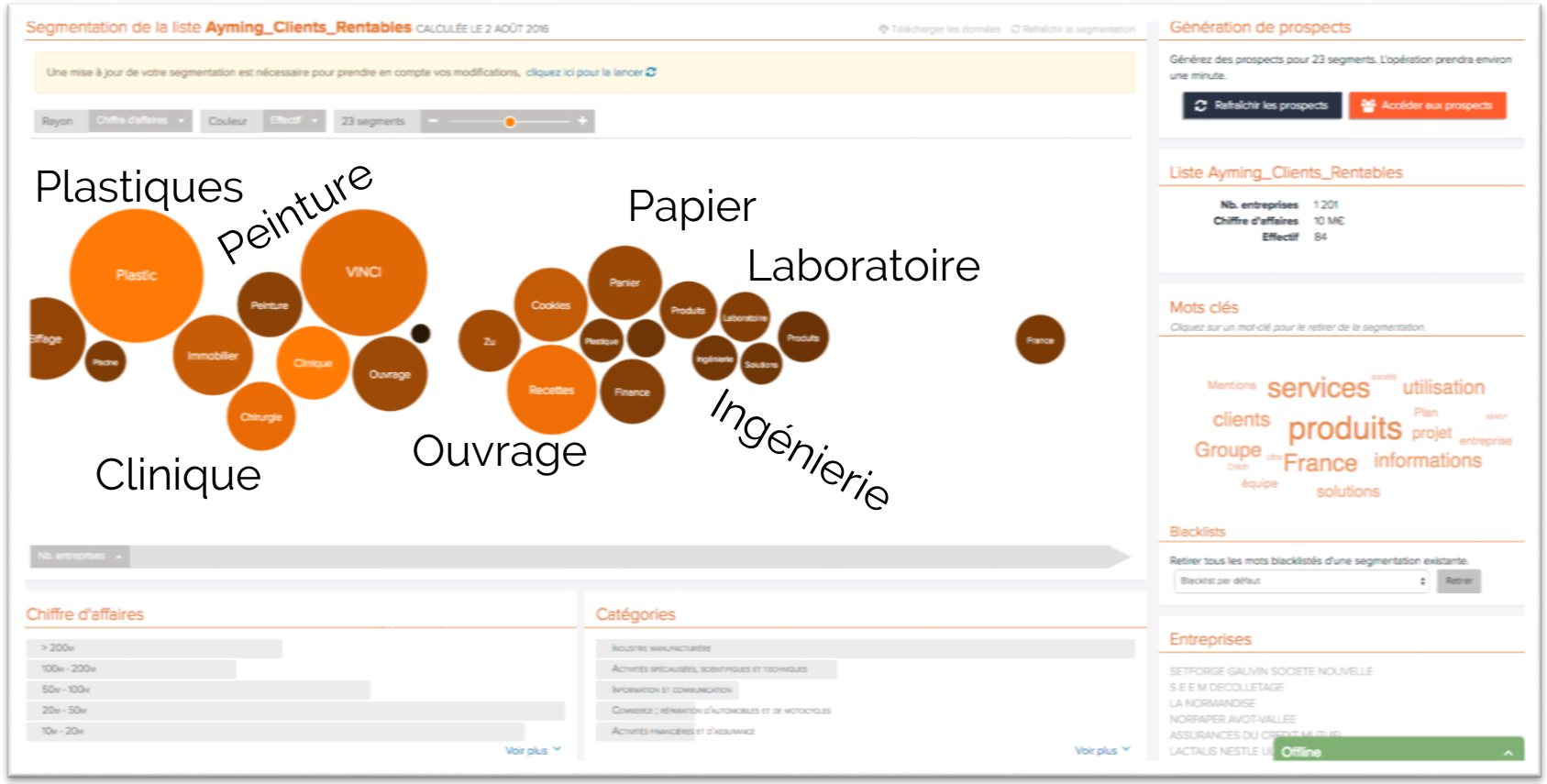
Cela signifie Term Frequency Inverse Document Frequency. Cela permet de créer une métrique de rareté d'un mot dans un document par rapport au corpus total.

Text Mining

Il se décompose en plusieurs étapes qui permettent de réaliser une matrice :

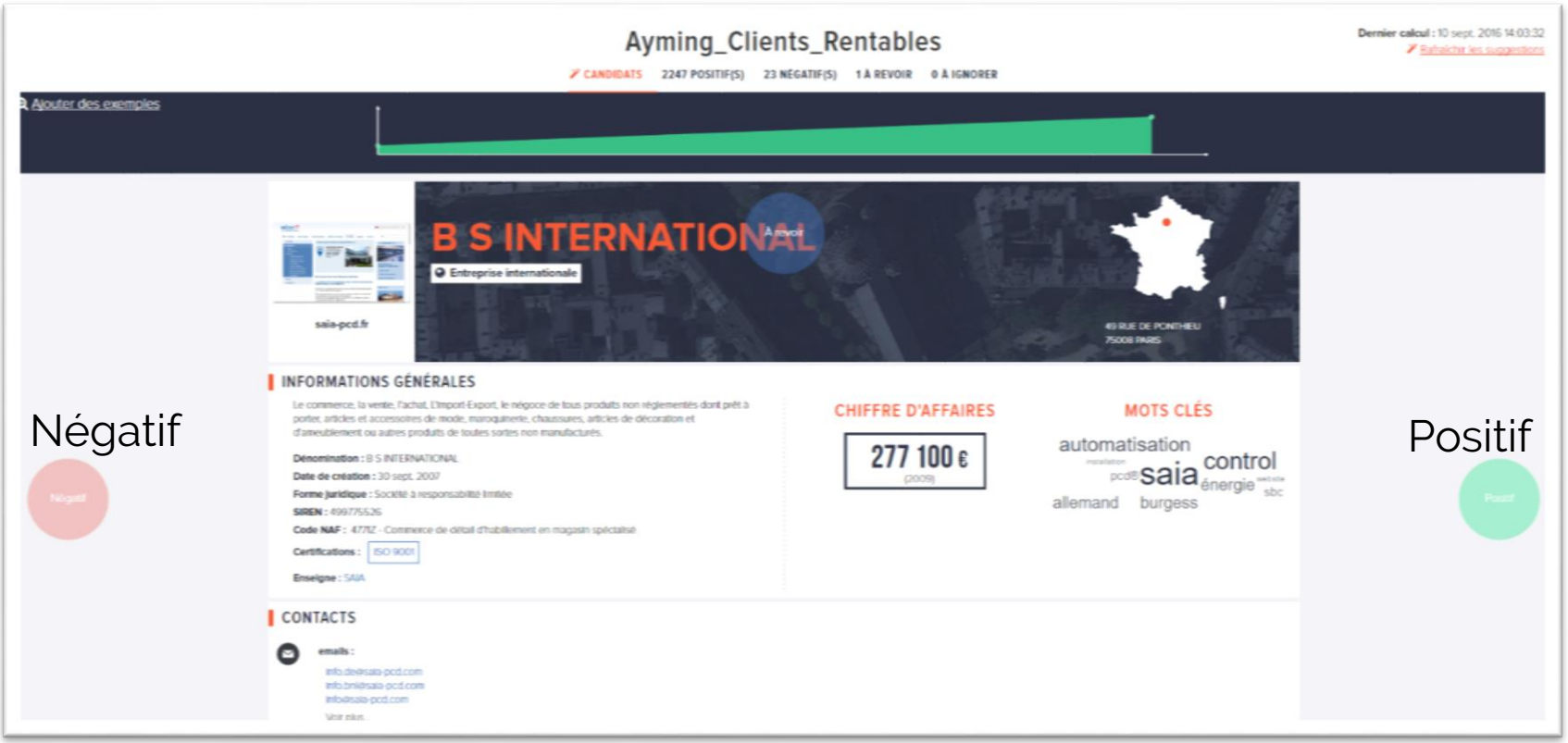
Segmentation et génération de prospects

Ces deux outils liés permettent de segmenter une liste d'entreprises afin de les regrouper en cluster cohérents en fonction du secteur d'activité. La génération de prospect permet de projeter les différents clusters sur l'entièreté de la base afin de trouver des entreprises similaires.



Utilisation de l'algorithme **K-Means** associé à deux étapes de réduction de dimensionnalité :

- Latent Semantic Analysis ou Singular Value Decomposition
  - Permet de regrouper les features entre elles
- Spectral Embedding
  - Permet de déterminer une représentation à faible dimension des variables



Négatif

Positif

Utilisation de l'algorithme **Multinomial Naïve Bayésien**.

Cet algorithme utilise une méthode Naïve Bayésienne sur toutes les variables en prenant comme hypothèse leur indépendance.

Il est très dépendant de la probabilité a priori des deux classes.

On observe une réduction du nombre de variables grâce à une sélection de type Chi2.

Targeting ou Ciblage

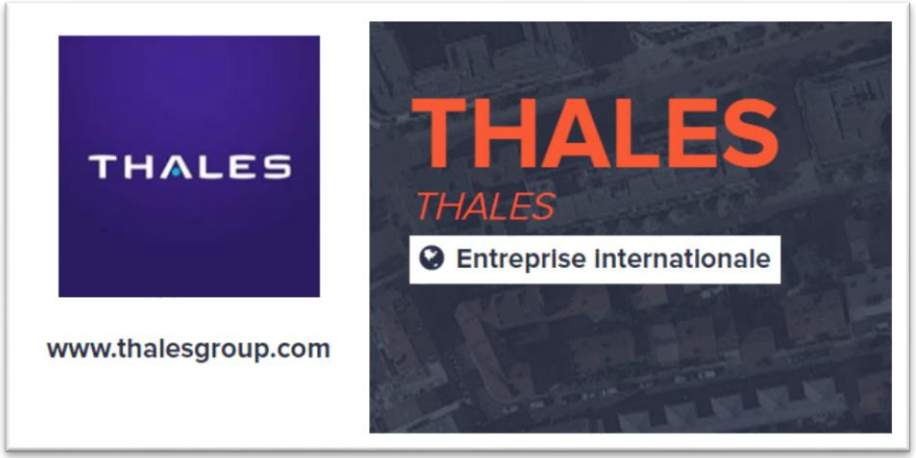
Le Targeting permet à l'utilisateur de mettre en place son propre modèle. Différentes entreprises lui sont proposées et c'est à lui de déterminer si elles font parties de la classe positive ou négative.

Catégorisation

Les algorithmes de catégorisation permettent d'appliquer des étiquettes aux entreprises françaises :

- E-Commerce
- B2B/B2C
- Internationale

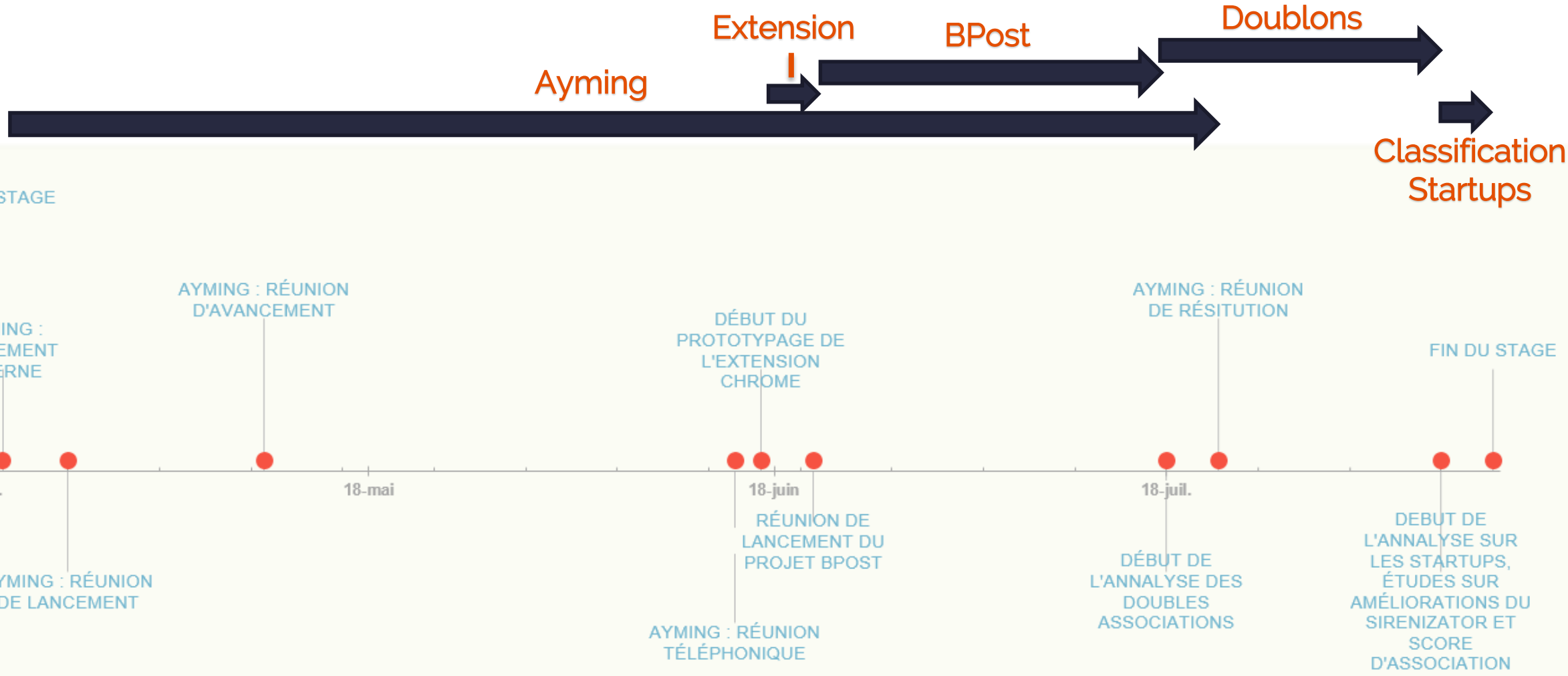
A la fin de chaque crawl les entreprises sont re-catégorisées.



Les algorithmes de catégorisation (score, régression linéaire, etc.) utilisent les données sémantiques des crawls mais aussi des données extraites :

- Présence sur les réseaux sociaux
- Différentes plateformes web de e-commerce
- Langues détectées.

Mes Missions



Planning de stage

Projet Ayming

Ayming est une société de conseil aux entreprises.

La mission : optimiser la recherche de prospects afin d'augmenter la rentabilité.

ayming | business performance consulting

Statut	Compte
Suspect	66 490
Clients rentables	2 238
Client non-rentables	3 421
Prospects	382

Description du portefeuille

Récupération des données

Sources des données :

- base C-Radar;
- extractions d'articles financiers pour les levées de fonds;
- données de l'INPI pour les dépôts de brevets;
- diverses provenances Open Source.

Crédit Impôt Recherche et Innovation :

Données structurées dans des fichiers PDF. Utilisation de Tabula pour extraire les données.

Projets européens :

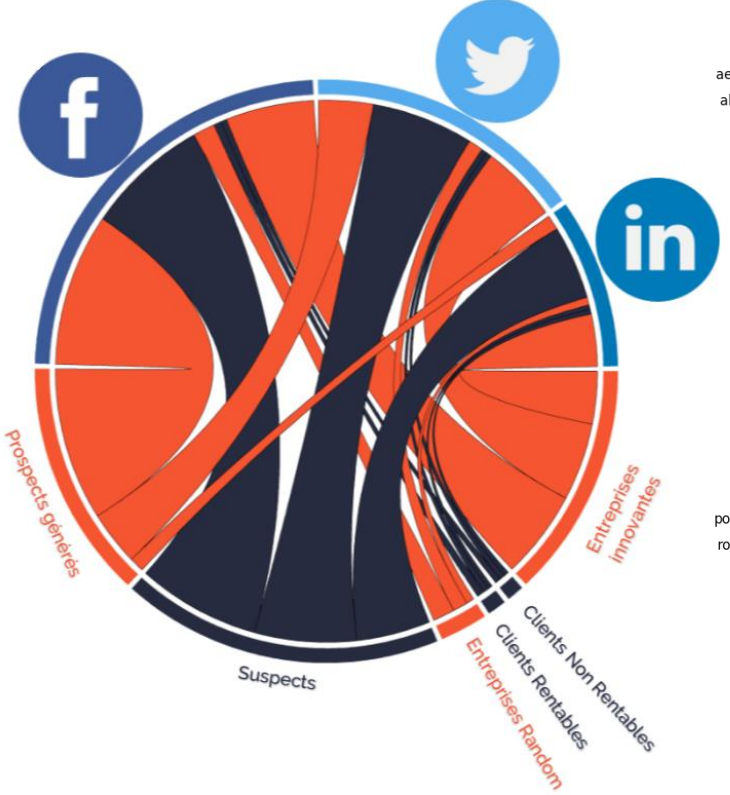
Fichiers Excel comprenant le nom des entreprises et les montants des financements. Utilisation du Sirenizator.

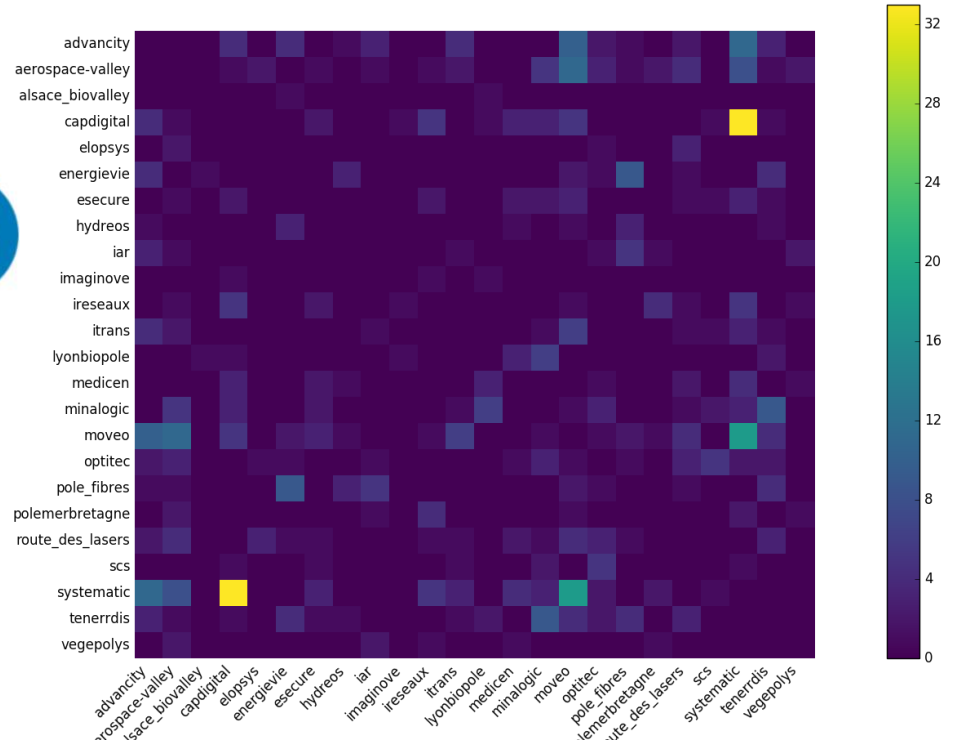
Analyse sémantique de la description :

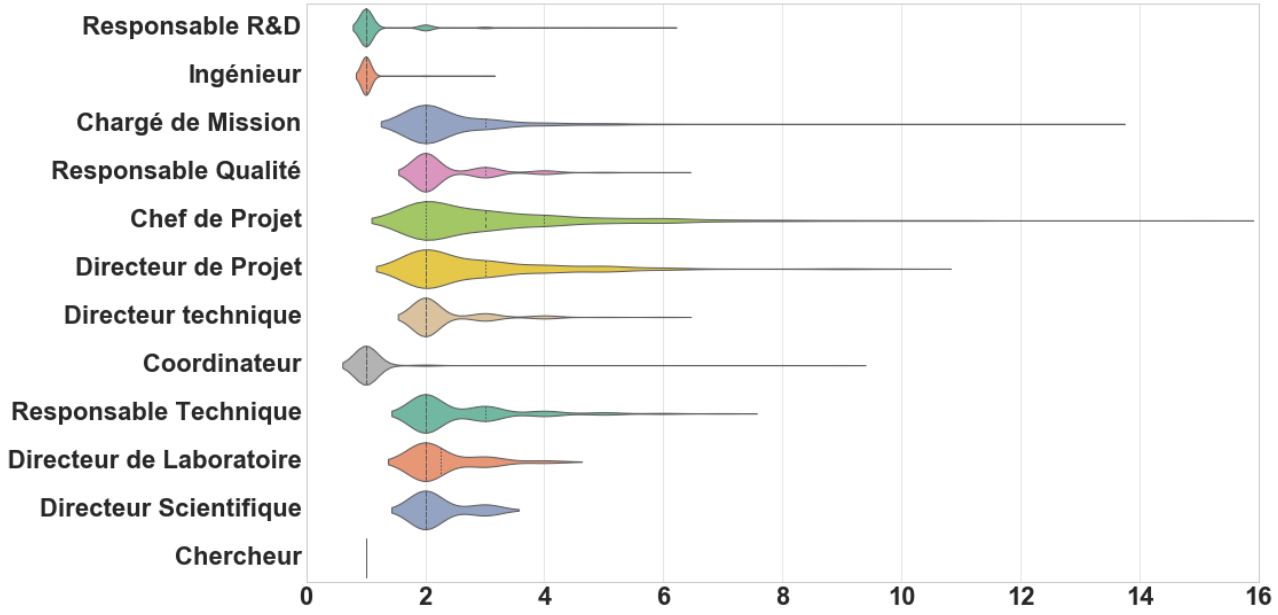
Récupération de la description de l'entreprise (ou objet social) et mise en place d'une méthode de Text Mining.

Données d'innovation :

Données d'innovations récupérées depuis des bases MongoDB d'anciens projets et mise à jours des données concernant les brevets et les levées de fonds.







Exemples de visualisation de données

Utilisation des librairies MatplotLib et Seaborn pour créer des graphiques expliquant les différentes données utilisées.

Analyses

Algorithmes testés :

- Logistic Regression ;
- Random Forest ;
- Support Vector Machine ;
- AdaBoost ;
- Bagging ;
- Gradient Tree Boosting.

Plusieurs modèles de données ont été testés.

Un seul a été conservé. Prédire si une entreprise pourrait être client d'Ayming :

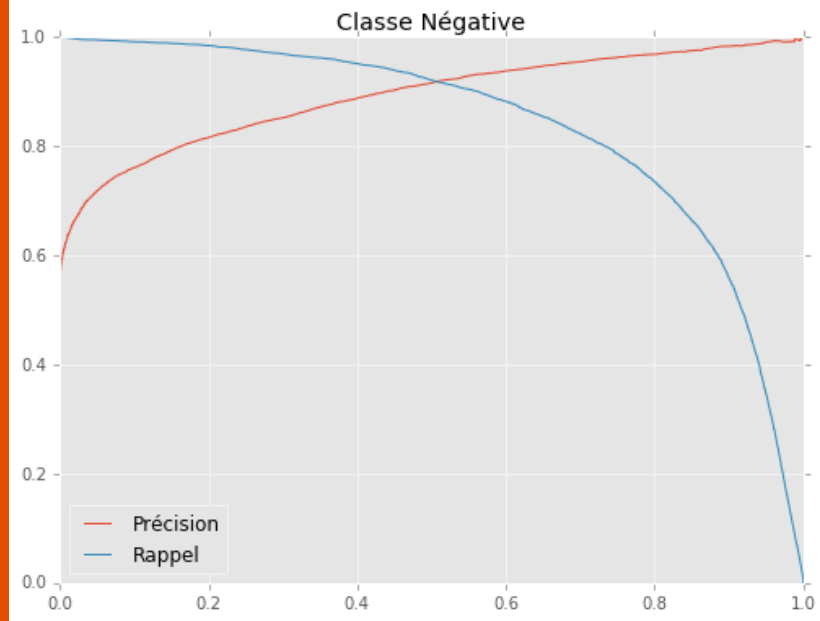
- Classe Positive : clients rentables (2 238) et non-rentables (3 421)
- Classe Négative : 5 000 entreprises tirées aléatoirement.

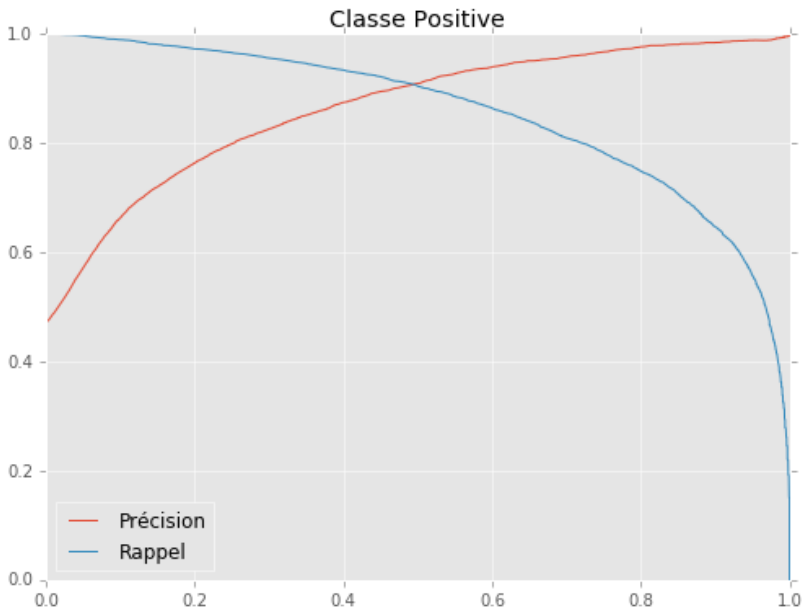
Matrice de confusion :

```
[[5206  453]
 [ 484 4516]]
```

Rapport de classification

	precision	recall	f1-score	support
Clients	0.91	0.92	0.92	5659
Random	0.91	0.90	0.91	5000
avg / total	0.91	0.91	0.91	10659





Les étapes suivantes.

1. Une segmentation sur 25 segments et une validation manuelle des différents segments pour ne garder que les plus cohérents.
2. Génération de prospects sur les 8 segments choisis.
3. Projection du modèle sur les 4 ensembles de données :
    1. Entreprises innovantes : toutes les entreprises observant un critère d'innovation (Levée de fonds, dépôts de brevets, etc …)
    2. Suspects du portefeuille d'Ayming
    3. Les prospects
    4. Les prospects générés par la segmentation
4. Validation manuelle et statistiques sur les résultats pour vérifier la cohérence du modèle.



Détection de technologies Web

Utilisation d'un package OpenSource

Modification du code JavaScript de Wappalyzer pour insérer dans une div HTML les technologies détectées.

Les différents étapes :

1. Instancier un Browser Headless depuis Python à l'aide de Selenium
2. Installer l'extension modifiée dans le navigateur.
3. Charger la page voulue.
4. Détecter les technologies et insertion de la div HTML
5. Récupérer les technologies détectées depuis le code de la page.
6. Prendre le screenshot et renvoyer le résultat

Process de détection

On utilise :

- Le contexte JavaScript de la page (variables d'environnement) ;
- Les en-têtes des requêtes HTTP ;
- Le contenu de la page HTML ;
- Les scripts importés.

Le code de Wappalyzer évalue des expressions régulières afin de détecter les technologies à partir de différentes sources :

```
{
    "Google Analytics": {
            "cats": [
                10
            ],
            "env": "^gaGlobal$",
            "headers": {
                "Set-Cookie": "__utma"
            },
            "icon": "Google Analytics.svg",
            "script": "^https?://[^\\/]+\\.google-analytics\\.com
            "website": "google.com/analytics"
        }
}
```

Architecture du plugin

Tous les plugins python sont abonnés à une queue RabbitMQ.

Résultats au format JSON :

1. L'url ;
2. Les technologies détectées avec leurs catégories ;
3. Un screenshot de la page du site web en base64.

Traitement des erreurs avec des exceptions Python.

Gestion de Logs.

Instanciation dans un conteneur Docker pour les paralléliser.

PROFIL TECHNIQUE DU SITE www.c-radar.com

CMS

WordPress

Outil de statistiques

Clicky · Google Analytics · Mixpanel

Chat en direct

Tawk.to

Logiciel de marketing

Yoast SEO

Widget

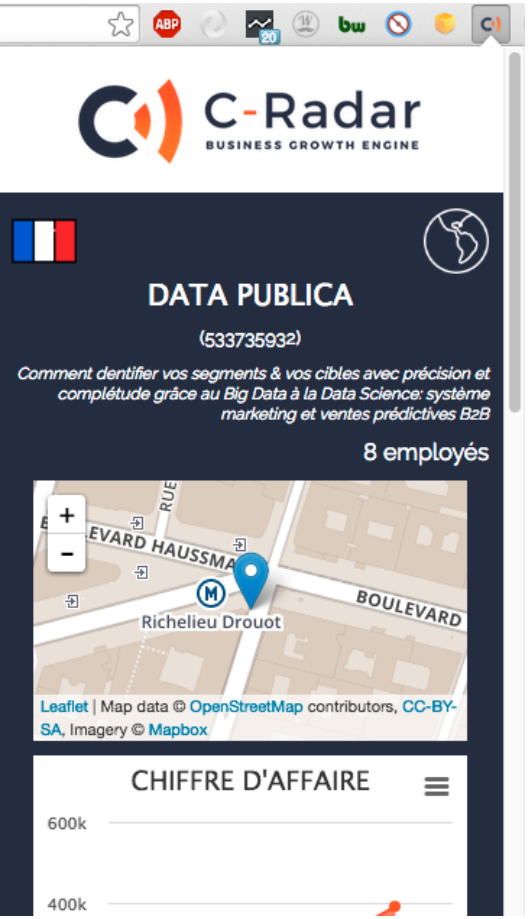
Facebook · Google Plus · Linkedin · Twitter

Autres technologies détectées

- **jQuery** — Framework JavaScript
- **prettyPhoto** — Galerie photo, Framework JavaScript
- **Twitter Emoji (Twemoji)** — Graphismes JavaScript
- **PHP** — Language de programmation
- **Font Awesome** — Script de police
- **Google Font API** — Script de police
- **Nginx** — Serveur web

Profil technologique du site web de l'entreprise

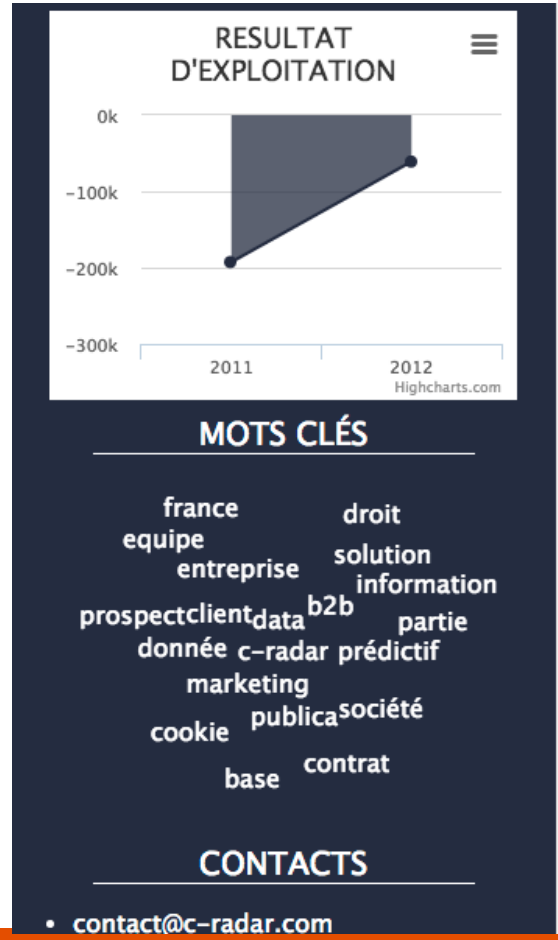
L'extension Chrome



Affichage de données pertinentes lors de la visite d'un site web.

- Pays d'origine et présence à l'international
- Nom et numéro de Siren
- Description détectée sur les réseaux sociaux
- Nombre d'employés
- Géolocalisation
- Données financières sous formes de graphiques HighCharts
- Nuage de mots les plus présents (ICloud)
- Contacts détectés

Mise en place d'un système d'authentification par Cookie.




Analyse des doublons

Le processus d'association d'une entreprise à un site web n'est pas robuste à 100% : certaines entreprises ne sont pas associées au bon site web.

Chaque association entre un site web et une entreprise est scoré.

Le score est composé de plusieurs métriques calculant la différence entre les données de l'entreprise et les données détectées sur le site web.

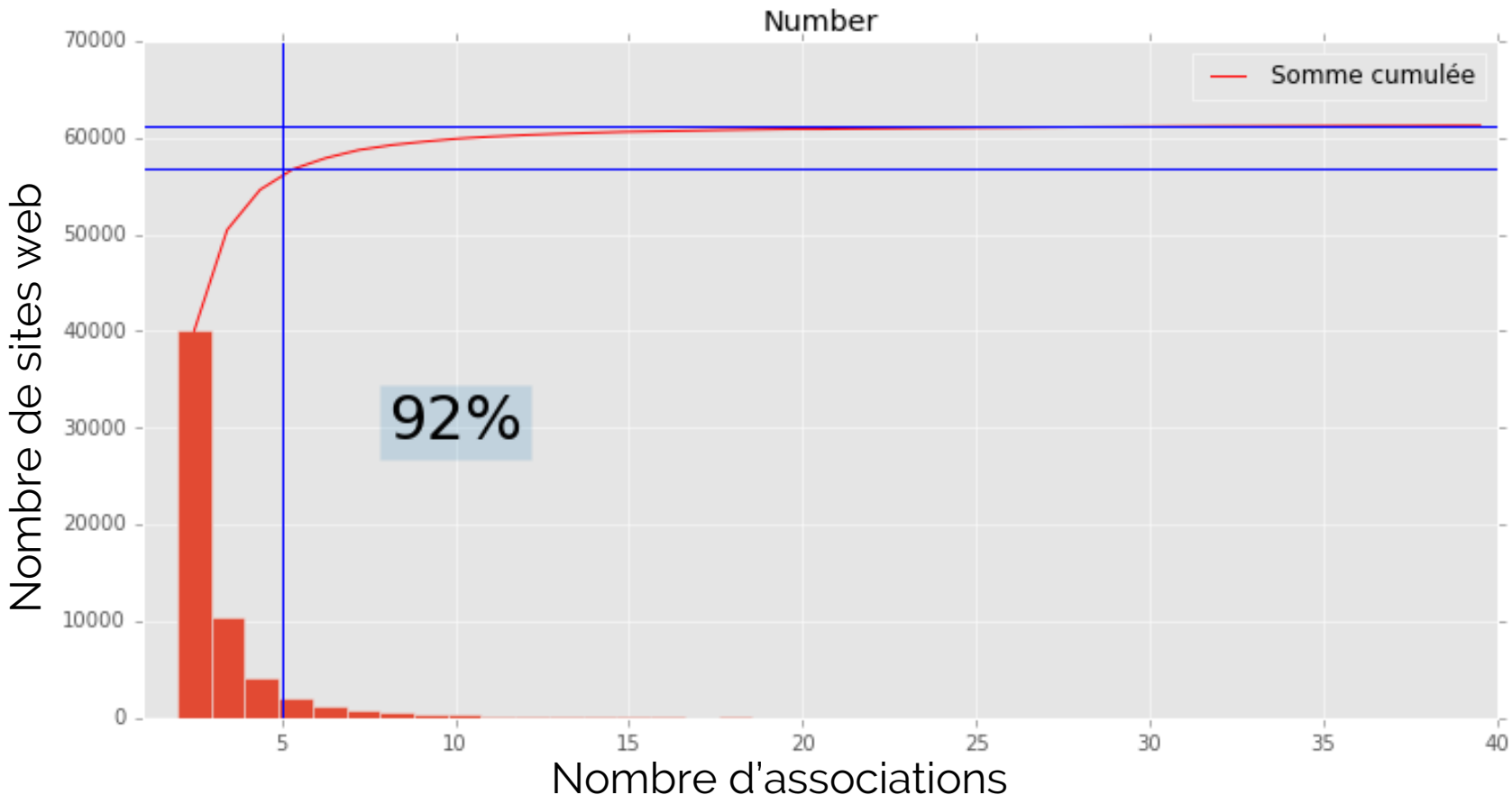
L'entreprise est associée si le score calculé est supérieur au seuil.

Quantification des associations multiples

Aujourd'hui 61 379 sites web sont associés à plusieurs entreprises. Ce qui correspond à 194 869 entreprises associées à plusieurs sites web.

92% des multiples associations concernent des associations à moins de 5 entreprises.

Le fait de solutionner les **doubles** associations permet de retirer 65% des multiples associations.

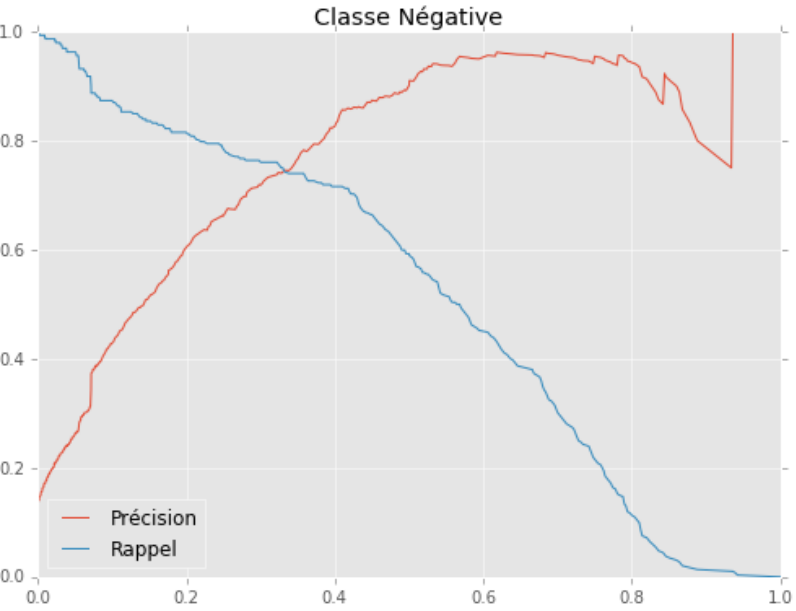


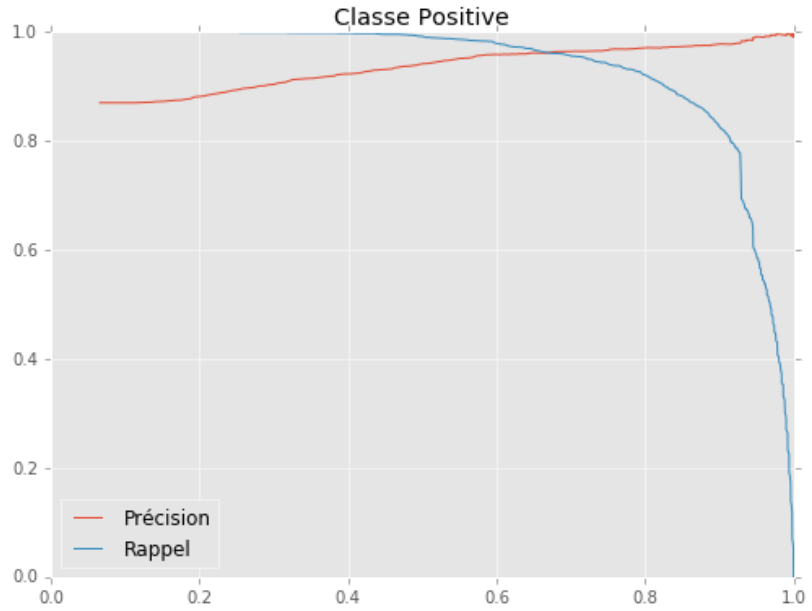
Multiples associations ?

- Utilisation de la sémantique du site
- Réduction de dimensions testés : LSA, PCA , Spectral Embedding
- Sélection de variables Chi2

```
    Matrice de confusion :

[[ 173  119]
 [  19 1903]]

    Rapport de classification :

             precision    recall  f1-score   support

   multiple       0.90      0.59      0.71       292
     unique       0.94      0.99      0.97      1922

avg / total       0.94      0.94      0.93      2214
```

Ici 119 sites à plusieurs associations sont prédit comme uniques : 40%.





Prédiction du type d'association

Classification d'un site web en **2 catégories** :

1. Un site de franchise, groupe ou annuaires;
2. Les autres, devant être associés une seule entreprise.

Détection de mauvaise association

Modélisation du problème :

Considérons deux entreprises A et B et un site W.

“Sachant la deuxième association B <-> W, la première A <-> W est-elle légitime ?”.

- « Oui » : Classe positive
- « Non » : Classe négative.

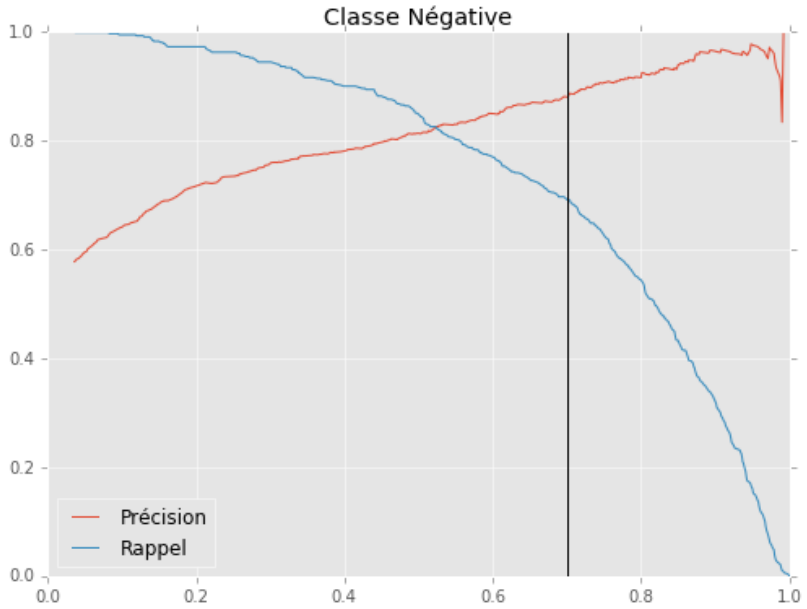
On crée des métriques représentant ces associations :

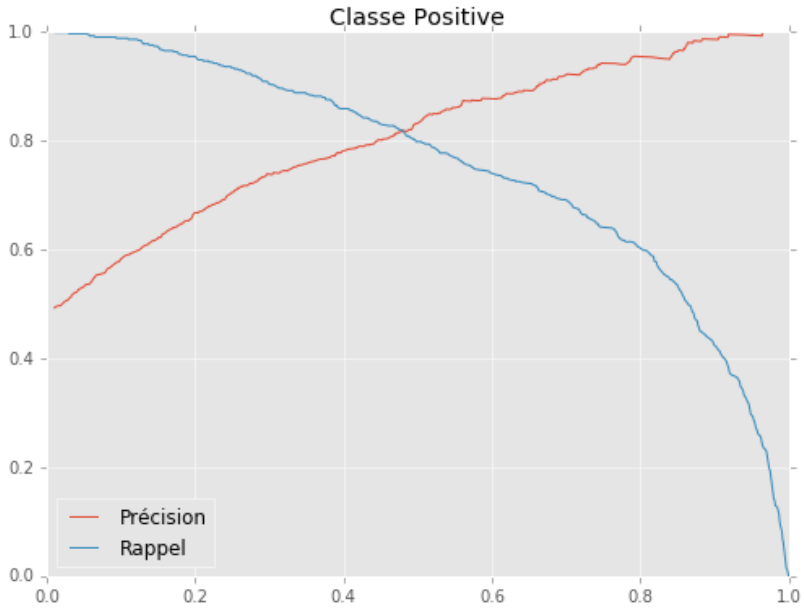
1. Distance entre informations du site W et des entreprises A et B;
2. Différence de score d’association entre A et B ;
3. Distance sémantique entre les noms des deux entités A et B ;
4. Différence de présence des variables pour A et B sur W : ville, adresse, numéro de téléphone, code postal, etc

```
Seuil 1 :
['0.89', '0.70']
['0.92', '0.69']
```

Rapport de classification :

	precision	recall	f1-score	support
0	0.82	0.85	0.83	497
1	0.84	0.81	0.82	479
avg / total	0.83	0.83	0.83	976





Classification de Startups

Catégorisation des entreprises de l'entièreté de la base des entreprises françaises.

Mise en place d'un scraper pour récupérer les startups de l'annuaire de l'usine Digitale.

Données utilisées et modèle

Variables explicatives :

1. données récupérées pour le projet Ayming ;
2. données financières (chiffre d'affaire, levées de fonds, etc) ;
3. données d'appartenance aux pôles de compétitivité ;
4. dépôts de brevets

Nombreux modèles testés. Seulement Random Forest utilisée.

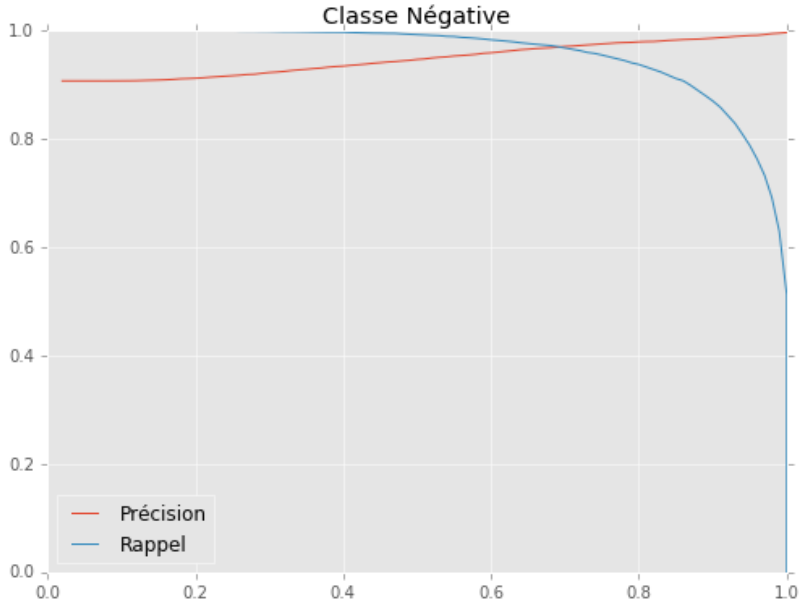
Classe positive : « Est une startup »
Classe négative : « N'est pas une startup »

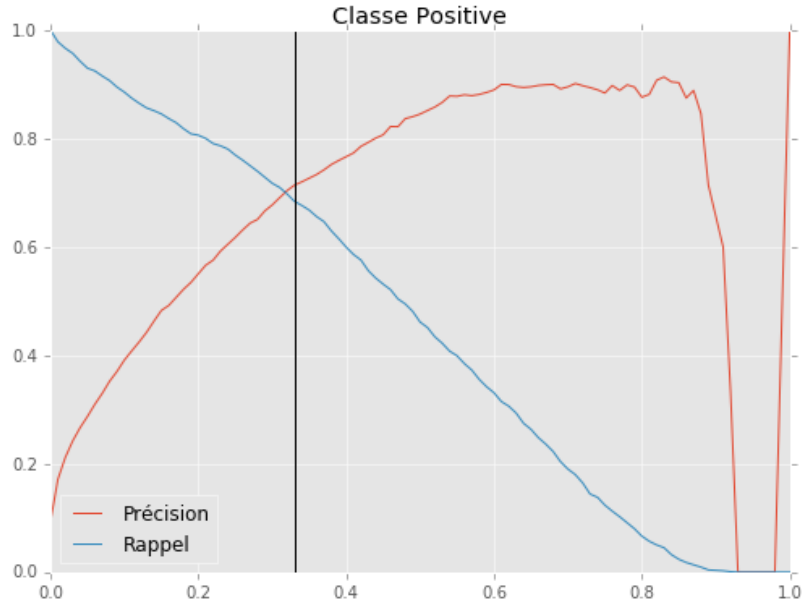
Matrice de confusion :

```
[[11280    92]
 [  643   529]]
```

Rapport de classification :

	precision	recall	f1-score	support
False	0.95	0.99	0.97	11372
True	0.85	0.45	0.59	1172
avg / total	0.94	0.94	0.93	12544

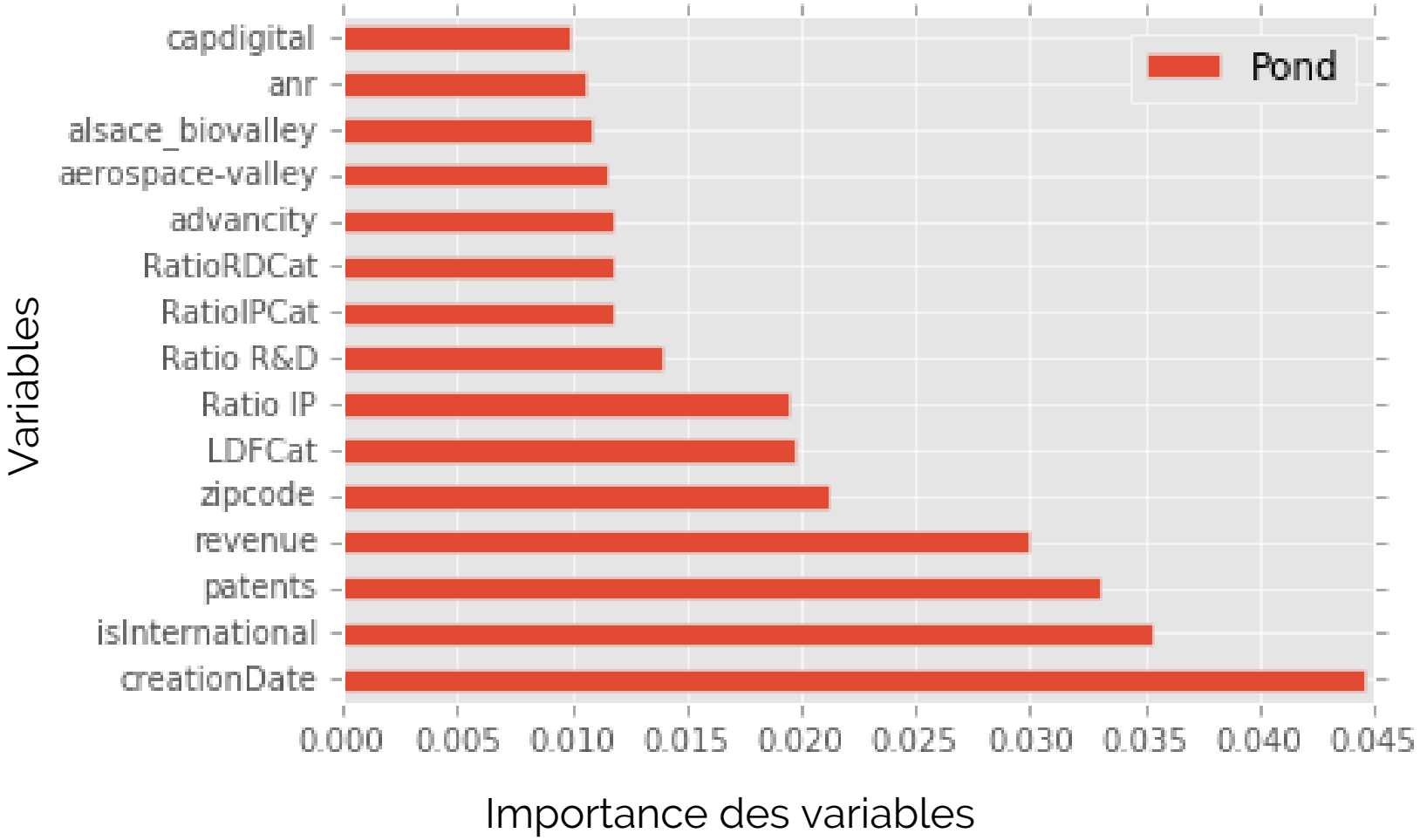




Rappel insuffisant pour être utilisé (45%).

La Random Forest renvoie une importance moyenne de chaque variable et non une pondération comme un modèle linéaire.

15 variables les plus discriminantes :



Description des variables les plus importantes

Données constituées de :

1. Pôles de compétitivité : capdigital, advancity
2. Données financières : Ratio IP et R&D, revenue
3. Données d'innovation : patents et LDF
4. Données calculées : présence à l'international.

Conclusion

Nombreuses compétences acquises ou renforcées

Technologies web

Algorithmes de Machine learning

Génie logiciel

Architecture d'applications

Contexte Start Up

Bilan personnel très positif

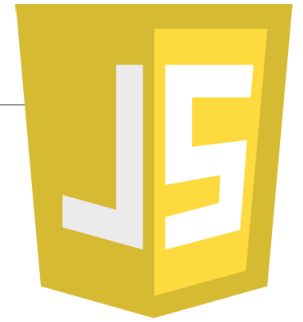
Premier stage en entreprise, travail en équipe, environnement Data Science

Conclusion

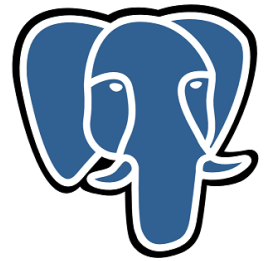

Java EE™


cassandra


spring




elastic




mongoDB
RabbitMQ™
PostgreSQL
docker
git
scikits
learn
machine learning in Python

Annexes

Random Forest

1. La Random Forest est un algorithme ensembliste
2. Elle met en place un grand nombre d'arbres de décisions (par défaut 10)
3. Chaque arbre observe un apprentissage sur
    1. un ensemble de variables tirées aléatoirement (par défaut $\sqrt{nombre\ de\ variables}$).
    2. Et un ensemble d'échantillons aléatoires.
4. La prédiction finale repose sur un vote.
    1. La proportion d'arbre ayant prédit la classe sur l'ensemble des arbres créer la probabilité finale.
    2. Plus le nombre d'arbre prédisant la classe est elevé plus la probabilité sera proche de 1.

Big Mama

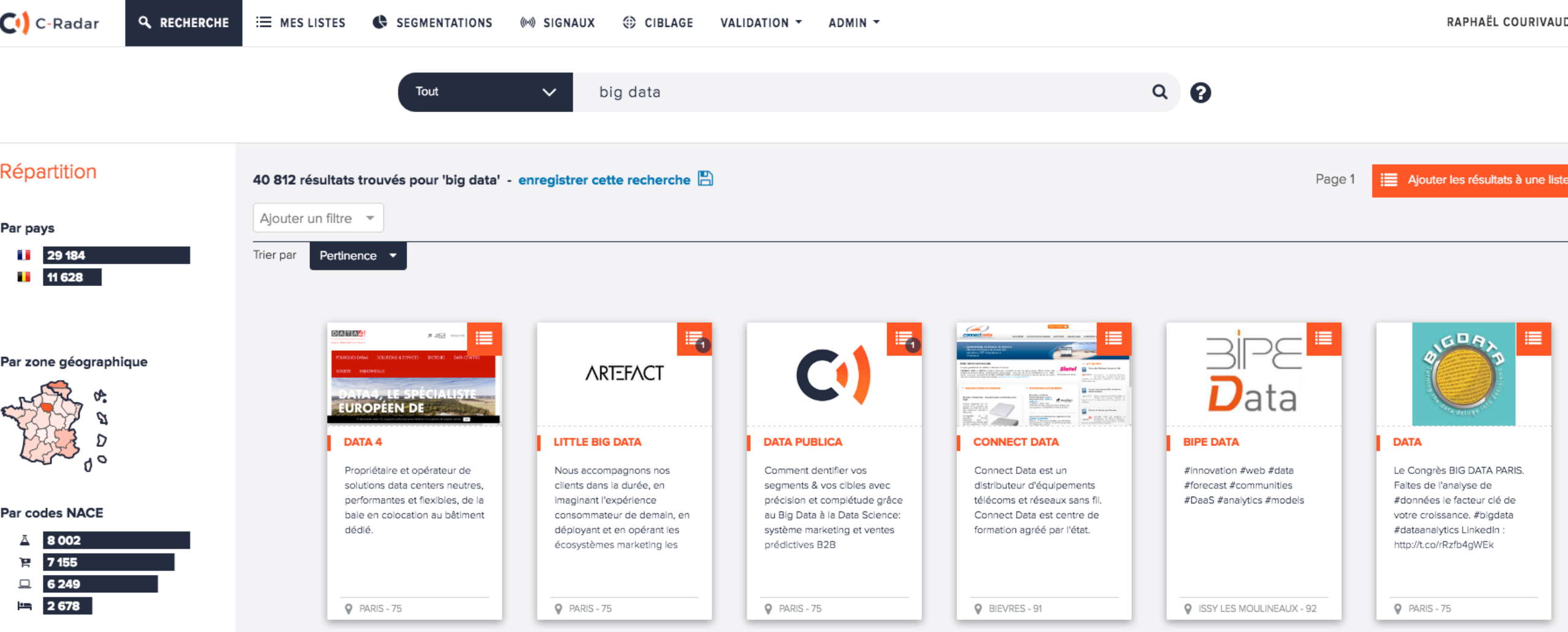
Big Mama est une matrice de 450 000 lignes et 1,2 million de colonnes créé à partir de l'analyse des crawls des sites.

Auparavant elle était régénérée à chaque fois qu'un algorithme devait être exécuté.

Elle est stockée sur disque pour accélérer les temps de calcul des différents algorithmes.

Elle est stockée sous la forme d'une matrice compressée de lignes pour optimiser l'espace mémoire.

Avec la version Mondiale on arrive aux limites de l'architecture actuelle.



Résultats de la recherche

Permet d'effectuer une recherche par mots clés (ici « big data »). Affiche quelques données de répartition des entreprises correspondantes à la recherche.

Bilan de chacun des projets (1)

1. Projet Ayming :
   - Les travaux ont été remis à toute l'équipe commerciale et peuvent être dès à présent utilisés.
2. Le plugin de détection de technologies :
   - Le code de l'extension est directement exécuté dans le navigateur pour plus de stabilité.
   - Il est effectif et fait partie du workflow de l'application.
   - Le projet étant Open Source, les règles sont automatiquement mises à jour. On peut également en rajouter pour des besoins spécifiques.
3. L'extension Chrome :
   - Elle est effective et fonctionne correctement. Il faudra encore du temps pour l'intégrer dans l'offre du produit.
   - Un ouverture des données financières (sous licence) permettrait de diffuser gratuitement cette extension directement sur le Chrome Store et la rendre accessible au grand public.

MongoDB

Base de données NoSql, permettant de stocker des données hétérogènes plus efficacement.

Cassandra

Base de données distribuée et non-relationnelle. Permet de stocker des données volumineuses avec un temps d'accès réduit.

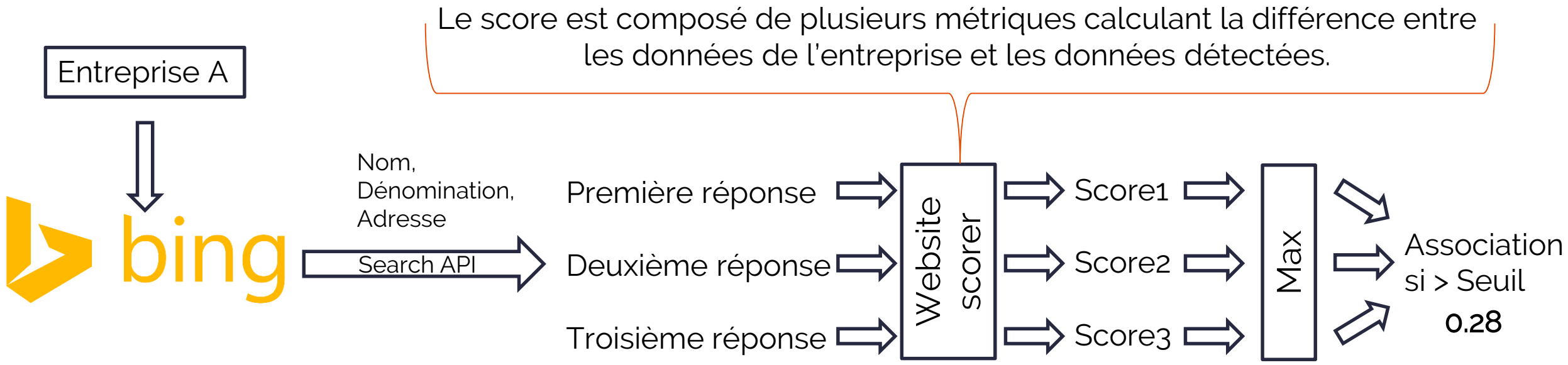
PostGreSQL

Base de données relationnelle : optimisée pour stocker des données géographiques. Utilisée pour les listes d'entreprises des utilisateurs.

Stockage de données

Analyse des doublons

Le processus d'association d'une entreprise à un site web n'est pas robuste à 100% : certaines entreprises ne sont pas associées au bon site web.



Bilan de chacun des projets (2)

4. Analyse des doublons :
   - L'étude permet d'avoir un idée quantitative du problème et de prouver qu'il existe une solution viable pour la résolution des mauvaises associations.
   - Une étude plus approfondie sur le score d'association doit être effectuée pour l'améliorer en ajoutant des métriques ou en utilisant un modèle non linéaire.
5. Classification de startups :
   - Le modèle ne possède pas encore un rappel suffisant pour être intégré à l'application.
   - Un refactoring est aussi nécessaire pour que les algorithmes de catégorisation puissent accéder à des données externes : des travaux ont été lancés sur le sujet.