Majeure Systèmes Informatiques

Compte-rendu de projet de convention d'études industrielles :

« Outil d'analyse de la réputation sur les réseaux sociaux »

Par :

Mme Salma LOUDARI
Mr Raphaël KERN

Encadré par :

Nacéra SEGHOUANI, professeure à CentraleSupélec
Daouda THIOYE, Data Scientist à TNP Consultants
Alexandre RABASSE, Consultant Confirmé à TNP Consultants

Avec la participation de :

Giulia ABELLO, Manager à TNP Consultants
Léa BUFFENOIR, Stagiaire à TNP Consultants
Aurélien LAURENS, Stagiaire à TNP Consultants

Remerciements

Nous tenons à remercier toutes les personnes qui ont participé à la réussite de ce projet et à son bon déroulement, aussi bien pendant les phases de réunions que tout au long des phases de travail.

Tout d'abord, merci à Mme Nacera Seghouani, Professeure à CentraleSupélec, de nous avoir accordé sa confiance en nous proposant ce projet CEI, et pour l'opportunité unique qu'elle nous a offerte en nous accompagnant tout au long de la réalisation de ce projet. Elle nous a permis pendant ces 5 mois de nous développer techniquement, professionnellement et personnellement.

Un grand merci à Daouda Thioye, DataScientist à TNP Consultants, pour son accueil chaleureux, et pour nous avoir suivi et accompagné pendant toute la durée du projet. Sa joie de vivre, sa disponibilité et sa générosité ont fait partie des éléments clefs pour le succès de ce projet. Merci aussi, d'avoir rythmé nos réunions de quelques blagues : sa bonne humeur est contagieuse.

Merci à Alexandre Rabasse, Consultant Confirmé à TNP Consultants, pour sa présence efficace lors des réunions clefs de ce projet. Ses conseils toujours pertinents et son sens de l'observation aiguisé nous ont été d'une grande aide.

Nous tenons également à remercier Léa Buffenoir et Aurélien Laurens, Stagiaires à TNP Consultants, pour leur réactivité et leur force de proposition. Merci d'avoir partagé avec nous vos parcours, vos connaissances et votre bienveillance.

Merci à Mme Yolaine Bourda, et à CentraleSupélec, d'offrir ce genre d'opportunité au sein de notre cursus d'étudiant ingénieur.

Enfin, nous remercions toutes les personnes qui nous ont accordé leur confiance et qui ont contribué, de près ou de loin, au bon déroulement de ce projet et sans qui la réalisation de celui-ci aurait été impossible.

Sommaire

I. Introduction

Dans le cadre de la troisième année du cursus ingénieur Supélec, pour faciliter la transition vers le monde de l'entreprise, les étudiants sont amenés à réaliser en binôme (ou trinôme) un projet de recherche ou de développement sur un sujet proposé à l'École par une entreprise dans le cadre d'une convention de partenariat. Il s'agit du CEI : Convention d'Études Industrielles.

Le présent rapport a pour sujet le CEI qui a eu lieu en 2018-2019 entre l'entreprise TNP Consultants et les deux étudiants de troisième année, Salma Loudari et Raphaël Kern. Il a pour objet : « outil d'analyse de web-réputation ».

1) Contexte et objectifs

De nos jours, pour vendre un produit et lui faire une place sur le marché il ne suffit plus de proposer un produit innovant, répondant à un besoin, bien conçu ou efficace. Une composante clef du commerce et de la vente est la communication. Ainsi, avec l'essor des réseaux sociaux, et l'omniprésence des moyens de communication sur internet, le marketing digital devient indispensable.

Dans ce contexte, les entreprises tendent à poster de plus en plus de contenus en ligne, soit pour faire la promotion de leurs produits, soit pour redorer l'image de marque, ou parfois tout simplement pour faire parler d'eux. Mais, devant des données de plus en plus volumineuses et nombreuses, il est compliqué de voir l'impact réel de toutes ces publications, de trouver des pistes d'amélioration, ou même de se positionner par rapport à ses concurrents.

C'est là que l'améliorateur de performance qu'est TNP Consultants entre en jeu. Des clients venant de secteurs d'activité diverses les contactent à ce sujet, et ils répondent à cette requête avec un outil d'analyse de la réputation sur les réseaux sociaux. C'est la mission qui nous a été confiée, et que nous avons acceptée.

2) Réseaux sociaux et limitations

De nombreux réseaux sociaux existent, beaucoup d'entre eux sont populaires et très utilisés, chacun avec leurs particularités.

- Facebook : Plus de 26 millions d'utilisateurs en France. C'est un réseau social qui permet de partager tout type de contenu et animer une conversation avec vos publics.
- Twitter : Près de 6,8 millions de comptes actifs en France. Le public qui l'utilise est généralement jeune de 15 à 34 ans. Près de 70-80% de journalistes sont sur Twitter et la plupart des hommes politiques et autorités publiques, les acteurs, sportifs, etc. ont un compte Twitter. Il est devenu une des principales sources d'information en temps réel C'est une plateforme de micro-blogging, ce qui signifie que vos posts sont limités en caractère – vous avez 280 symboles pour faire un message (auparavant 140).
- LinkedIn : Réseau professionnel par excellence, LinkedIn recense 6 millions de comptes actifs en France.
- Google + : Dès son lancement en 2011, ce réseau se voulait comme une alternative à Facebook. Malgré les efforts de Google pour l'imposer, Google + n'a pas vraiment rencontré son public. Bien qu'il revendique 300 millions de comptes dans le monde et environ 10 millions en France, l'activité des utilisateurs demeure relativement faible.

- Pinterest et Instagram : Applications concurrentes permettant de diffuser de l'information sous forme de visuels qui connaissent une forte progression en termes d'usage par les internautes en France. Les célébrités de la mode et du sport ainsi que la téléréalité sont très présentes sur Instagram, entre autres, dans le but de faire suivre leurs activités à leurs publics.
- YouTube : Depuis sa création en 2005 et son rachat dans la foulée par Google, la plateforme n'en finit pas d'imposer sa domination sur les contenus vidéos. Aujourd'hui, YouTube compte 1 milliard d'utilisateurs dans le monde et 22 millions en France. Le nombre d'heures de visionnage mensuelles sur YouTube augmente de 50 % chaque année tandis que 300 heures de vidéo sont mises en ligne chaque minute sur le réseau.

3) Déroulement et méthodologie

Notre travail sur ce projet a duré 6 mois. Le planning détaillé de notre travail est présenté en annexe 1 à la fin du présent rapport.

Notre travail a été rythmé par des réunions hebdomadaire ou bihebdomadaires avec :

- Une réunion interne avec Mme Nacéra Seghouani (réunion interne)
- Ainsi qu'une réunion avec l'équipe TNP.

Ce rapport est organisé en différents chapitres chacun correspondant chronologiquement à une étape du projet :

- Dashboard cible : ce chapitre décrit la conception du prototype de l'outil de web réputation « idéal » ainsi que les différentes fonctionnalités choisies

- Extraction et stockage des données : ce chapitre décrit comment et quelles données ont été extraites des différents réseaux sociaux, ainsi que la structure des tables de stockage construites

- Désambiguïsation des données : ce chapitre décrit l'algorithme conçue afin de désambiguïser les données extraites

- Analyse des données : ce chapitre décrit les différentes analyses réalisées sur les données

- Visualisation et mise en forme des données : ce chapitre décrit les choix d'implémentation visuelle des résultats de l'analyse des données

Les différentes étapes de notre méthodologie peuvent être résumées dans la figure 1. Cette figure représente également les interactions entre les différents acteurs des différentes étapes du projet.

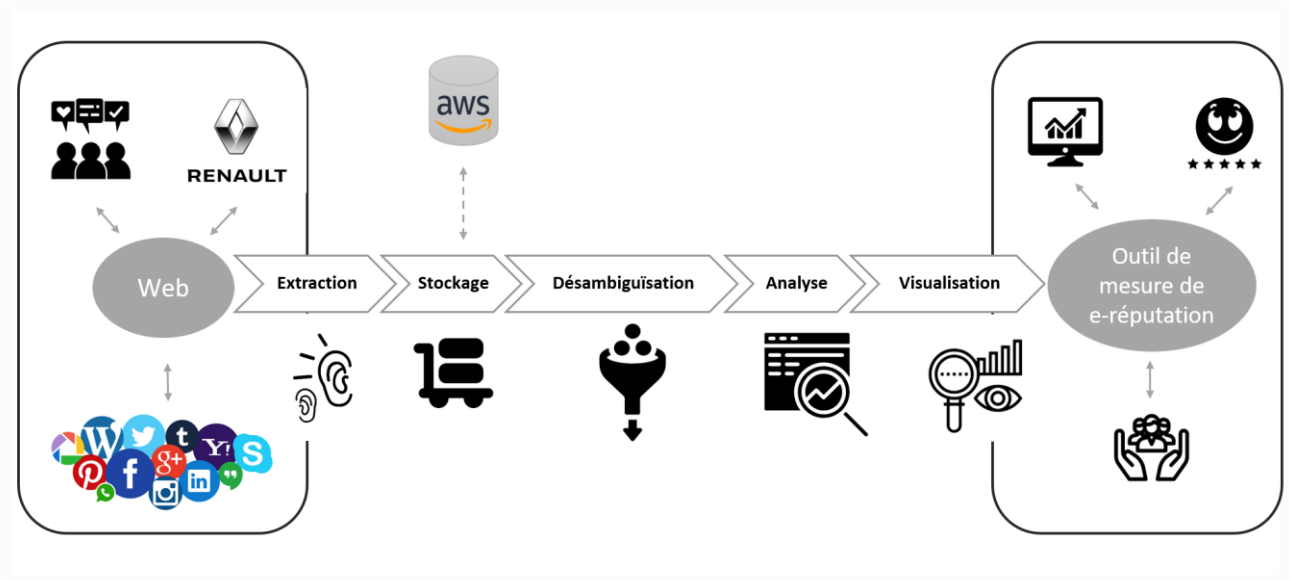

Figure 1 : Démarche, acteurs et interactions du projet

II. Dashboard cible

1) *Objectif*

La première étape de notre projet était de concevoir un prototype idéal de l'outil de e-réputation voulu. L'objectif de ce prototype étant de nous donner une idée précise de l'interface utilisateur voulu, du design ainsi que des métriques qui seront utiles pour l'utilisateur final de notre outil, à savoir un directeur marketing ou un community manager.
Nous avons pour cela utilisé le logiciel Adobe XD (disponible en version gratuite).

2) *Logiciel Adobe XD*

Adobe XD est une solution d'UX/UI design complète pour la conception de sites web, d'applications mobiles, etc.
Alliant rapidité, précision et qualité, Adobe XD permet aux designers de modifier et partager facilement des prototypes interactifs avec des collaborateurs sur l'ensemble des appareils et plates-formes, dont Windows, Mac, iOS et Android.

3) *Présentation du prototype*

Nous avons choisi de réaliser un prototype avec un design épuré en utilisant les couleurs de TNP (gris et vert clair).
Dès l'ouverture de l'outil, une fenêtre de paramétrage (Figure 2 et Figure 3) apparait qui permet d'indiquer :

- Les mots-clés qu'on souhaite étudier (ici : Renault)
- Les concurrents auxquels on veut se comparer

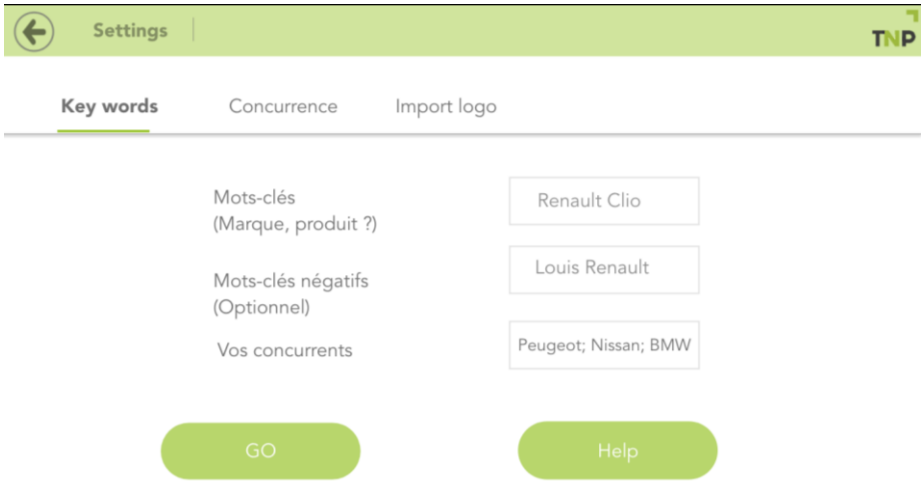

Fig. 2 : Page de paramétrage des mots-clés à utiliser pour l'extraction des données

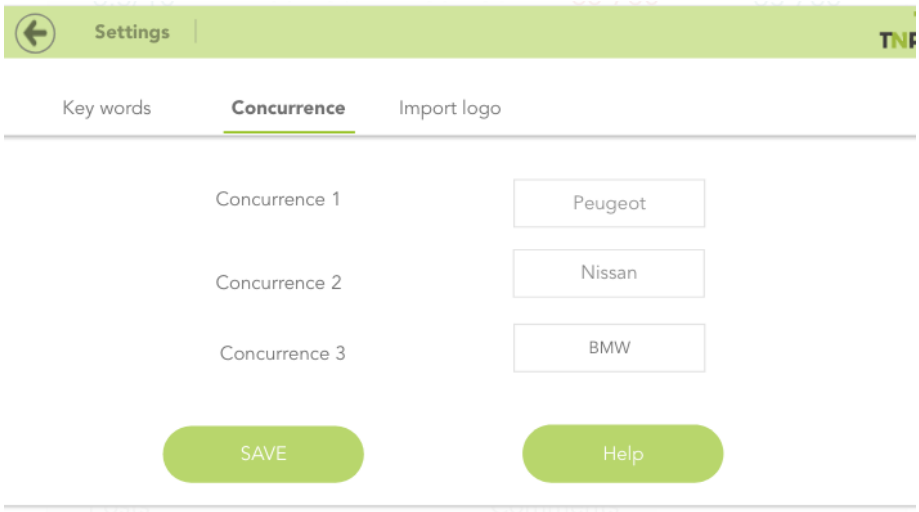

Fig. 3 : Page de paramétrage des noms des concurrents à utiliser pour l'extraction des données

Nous avons également inséré un onglet « alertes » en haut à droite de la fenêtre principale. Il permet de signaler, avec un système de notifications, les « bad buzz » : par exemple, un commentaire jugé négatif sur Twitter (Figure 4).

Fig. 4 : Zoom sur les notifications générées pour alerter des « buzz »

L'intérêt est de remonter cette information à l'utilisateur de l'outil de web analyse pour lui permettre de réagir rapidement et éviter que le message se répande sur le net : réponse directe au commentaire négatif, geste commercial, résolution du problème du client mécontent etc.

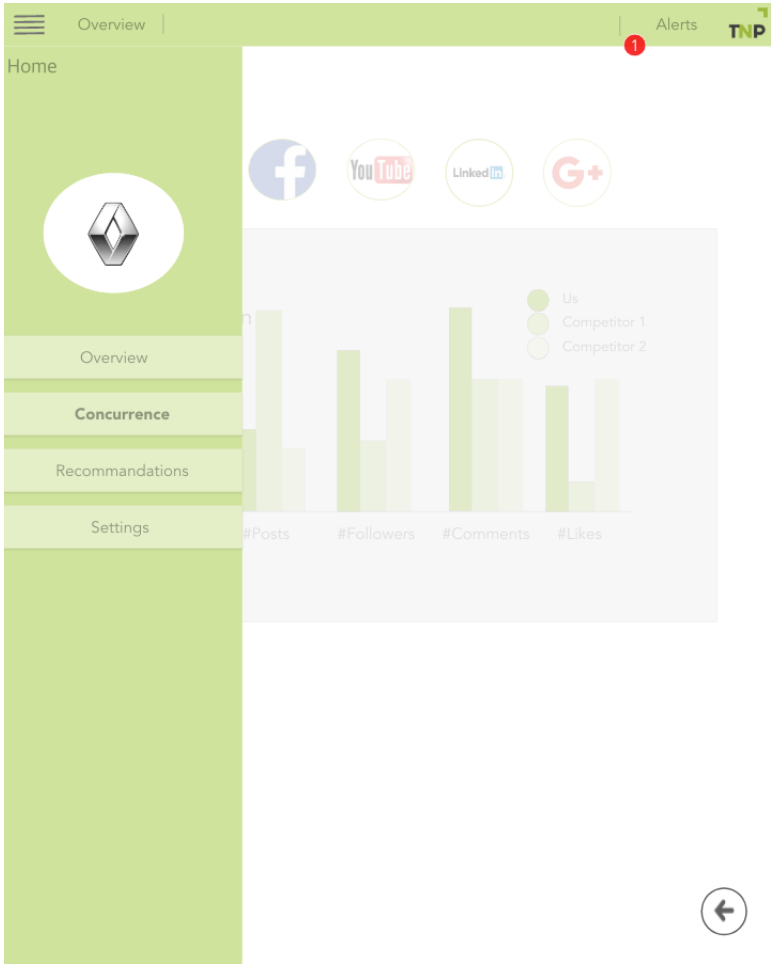

Fig. 5 : Slide-bar présentant le logo ainsi que les quatre onglets principaux du prototype

L'outil comporte une barre glissante (Figure 5) à gauche contenant les quatre sections principales suivantes :

- *Overview* : page donnant une vision globale des performances sur les réseaux sociaux
- *Concurrence* : page confrontant notre performance à celles des concurrents choisis
- *Recommandations* : page proposant des recommandations pour améliorer la présence sur les réseaux sociaux du client
- *Settings :* page de paramétrages du nom de la marque et des concurrents étudiés

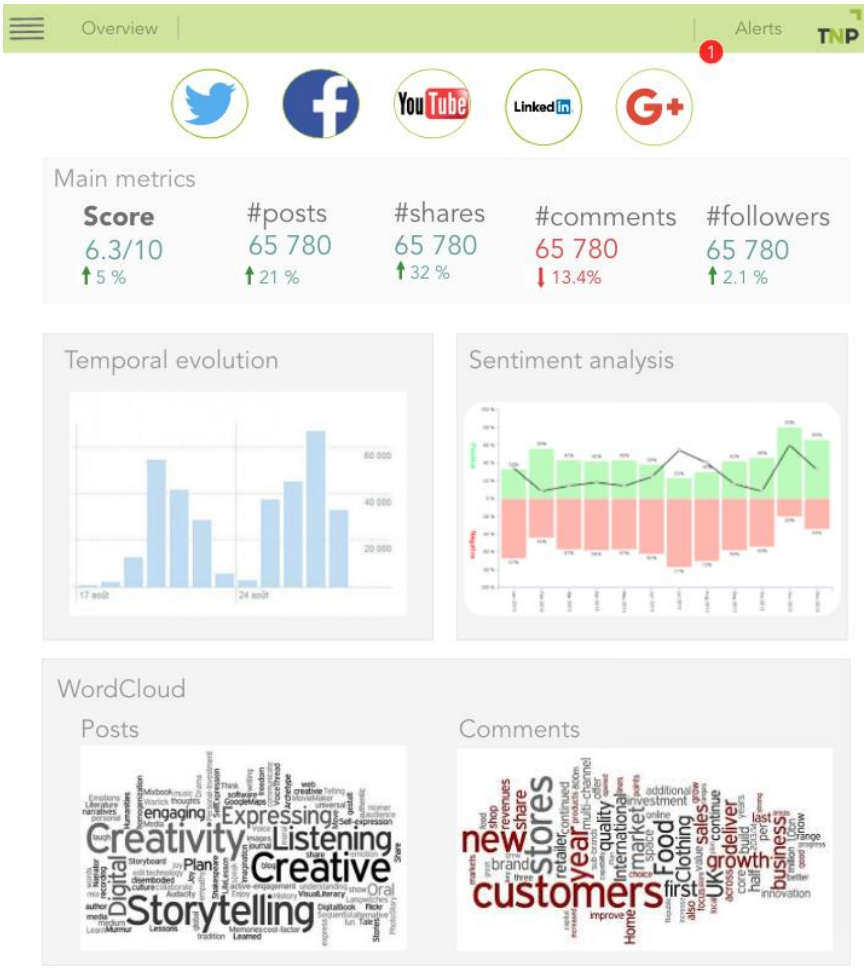

Fig. 6 : Page « Overview » du prototype qui présentent les performances de notre marque au global avec une possibilité de filtrer sur un seul réseau social

La page « Overview » (Figure 6) contient de haut en bas :

- Des boutons permettant de filtrer sur un réseau social en particulier (s'ils sont tous sélectionnés, les résultats seront agrégés sur l'ensemble des réseaux sociaux)
- Une évolution temporelle du nombre de posts et de commentaires (au global ou sur un réseau social particulier)
- Une évolution temporelle de l'analyse des sentiments
- Un nuage de mots sur les posts (en prenant en compte les tweets)
- Un nuage de mots sur les commentaires (en prenant en compte les réponses aux tweets et les commentaires des vidéos YouTube)

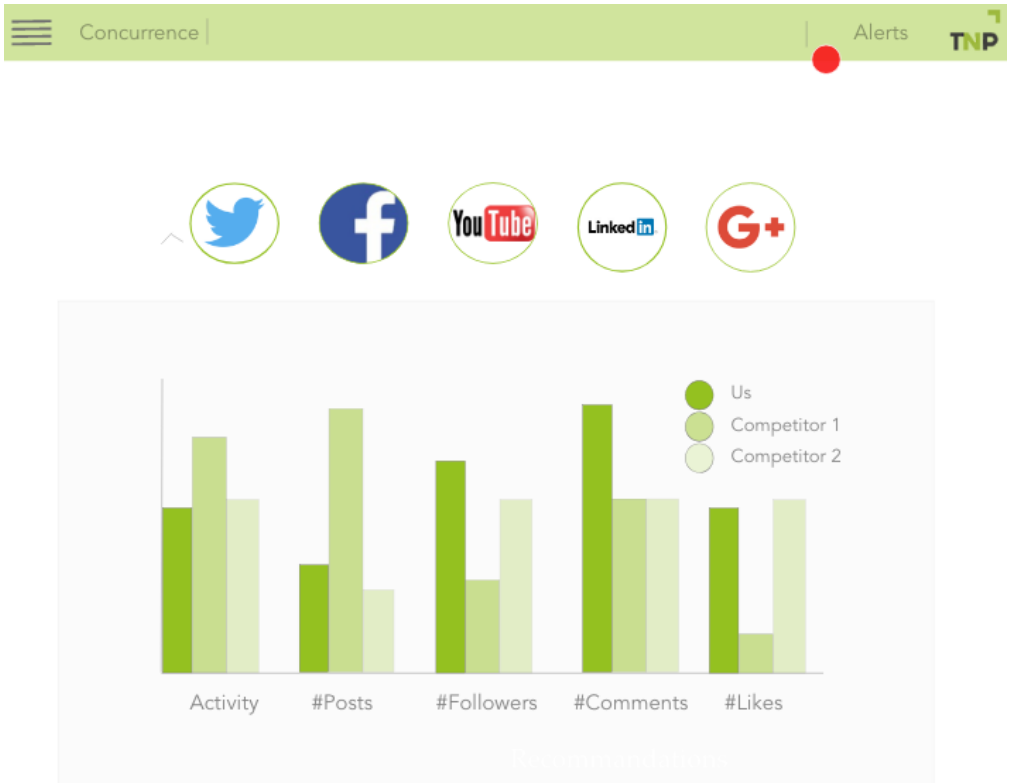

Fig. 7 : Page « Concurrence » du prototype qui confrontent les performances de notre marque et celle des concurrents choisis

La page « Concurrence » (Figure 7) contient de haut en bas :

- Des boutons permettant de filtrer sur un réseau social en particulier (s'ils sont tous sélectionnés, les résultats seront agrégés sur l'ensemble des réseaux sociaux)

- Un diagramme présentant en abscisse différentes métriques (activité, nombre de posts, nombre de *followers*, nombre de commentaires, nombre de *likes)* et en ordonnés les valeurs de ces métriques pour la marque (ou produit) de notre client versus celles des concurrents.

La page « Recommandations » (Figure 8) contient de haut en bas :

- Une section « When » qui indique le moment le plus opportun pour poster sur les réseaux sociaux (date et heure)
- Une section « Where » qui indique le réseau social où il est le plus opportun de publier
- Une section « What content » qui indique :
    - Les mots à fort impact présents dans les posts passés de notre client
    - Les mots à fort impact présents dans les postes passés des concurrents de notre client
- Une section « Influencers » qui indiques le classement des 5 plus grands influenceurs, en distinguant les experts du domaine étudié des célébrités.

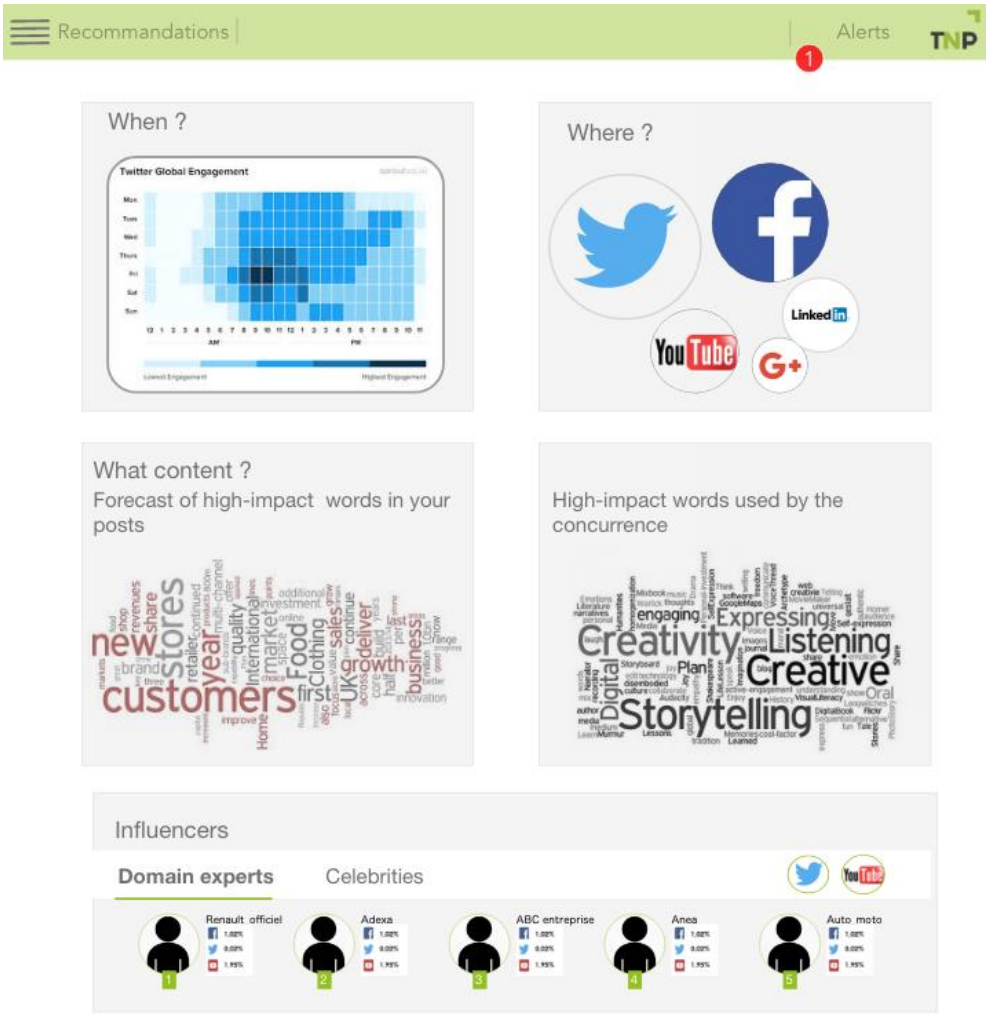

Fig. 8 : Page « Recommandations » du prototype

III. Extraction et stockage des données

1) Extraction et stockage des données Twitter

Nous avons extrait les données du réseau social Twitter. Nous les avons ensuite stockés dans une base de données de type relationnel (PostgreSQL) AWS dont l'accès nous a été octroyé par TNP.

a. Méthodologie et API utilisés

Après avoir créé une application Twitter, on utilise l'API « TweeterSearch » afin de récupérer les tweets correspondants aux mots-clés voulus. On a commencé par tester avec le mot-clé « Renault ».

Voici un extrait de l'ensemble des données d'un tweet auxquels on a accès grâce à l'API.

b. Choix d'implémentation

Les méthodes crées sont les suivantes :

- *collect_tweets()* : cette méthode permet de collecter une centaine de tweets contenant le mot-clé recherché et renvoie un dataframe où chacune des ligne correspond à un tweet et les colonnes correspondent aux différentes données qui nous intéressent.
- *create_table()* : cette méthode permet de créer une table dans la base de données AWS. On spécifie ici le type de chaque attribut.
- *enrich_table(dataframe)* : cette méthode prend en argument un objet de type dataframe et enrichie la table - préalablement crée – avec ce dataframe.
- *show_table()* : cette méthode permet l'affichage d'une portion de la table ainsi que de ses dimensions.

On obtient alors la table relationnelle suivante (Figure 9) pour le réseau social Twitter :

ID_USER	SCREENAME_USER	…	RETWEET_STATUS	TWEET_LANG

Attribut	*Type*	*Explications*
'Id_user'	Varchar(500)	Id de l'utilisateur
'Screename_user'	Varchar(500)	Pseudonyme de l'utilisateur
'Name_user'	Varchar(500)	Nom de l'utilisateur
'User_description'	Varchar(500)	Description du compte de l'utilisateur
'Verified_account'	Bool	Indique si le compte est vérifié par Tweeter (si célébrité par exemple)
'User_followers_count'	Varchar(500)	Nombre de followers du compte en question
'User_friends_count',	Varchar(500)	Nombre d'amis/de personnes qui suivent le compte en question
'User_favourites_count',	Varchar(500)	Nombre de comptes suivis
'User_statuses_count',	Varchar(500)	Nombre de tweets émis
'Account_date_creation',	Timestamp	Date de création du compte
'User_geo_enabled',	Bool	Indique si la localisation est activée
'User_location'	Varchar(500)	Localisation de l'utilisateur
'Tweet_geo',	Varchar(500)	Localisation du tweet
'Tweet_coordinates',	Varchar(500)	Coordonées du tweet
'Tweet_place',	Varchar(500)	Place
'Tweet_contributors',	Varchar(500)	Contributeurs
'Tweet_id',	Varchar(500)	Id du tweet
'Tweet_URL',	Varchar(500)	URL du tweet
'Tweet_text'	Varchar(500)	Texte du tweet
'Tweet_date_creation'	Timestamp	Date de création du tweet
'Tweet_retweet_count'	Varchar(500)	Nombre de retweet du tweet
'Tweet_hashtags'	Varchar(500)	Juxtaposition de hashtags présents dans le tweet
'Tweet_mentions',	Varchar(500)	Juxtaposition des mentions présentes dans le tweet
'Tweet_reply_to_user_name'	Varchar(500)	Nom de l'utilisateur auquel on répond
*'Tweet_reply_to_user_*	Varchar(500)	Id de l'utilisateur auquel on répond
'Tweet_reply_to_tweet_id'	Varchar(500)	Id du tweet auquel on répond

'Retweet_status'	Varchar(500)	Indique si c'est un retweet
'Tweet_lang'	Varchar(500)	Indique la langue du tweet (« fr » pour français, « en » pour anglais etc)

Figure 9 : Table des données extraites de Twitter

c. Problèmes et solutions

- Les données récoltées par l'API sont sous forme d'un dictionnaire, chaque attribut du tweet est de longueur variable. Par exemple, pour les hashtags présents dans un post, il peut y ont avoir 0, 1 ou plusieurs. Cela a posé problème lors du stockage dans notre base de données, étant donné le caractère relationnel, et donc figé des attributs.
  Solution : les attributs « mentions » ou « hashtags » sont des juxtapositions de mots séparés par des virgules.

- Le choix des types de chacun des attributs a été également difficile, puisqu'il s'agit de le rendre le plus flexible possible, pour l'ensemble des données possibles.
  On a opté pour le type *« varchar(500) »* qui est normalement en deçà de la taille des données possiblement récoltés.

- Les limites intrinsèques à l'API de TwitterSearch ont été contournées en réalisant un requêtage automatique et périodique, avec un délai de pause entre de requêtes suffisamment grandes.

- Le stockage de notre dataframe dans la base de données passe par la conversion du dataframe en un objet *IoString* où les différents attributs sont séparés par une tabulation \t.
  Afin de lever toute ambiguïté, on a traité le texte des tweets (description du compte et texte du post) en retirant tous les caractères de type tabulation.

- Chaque ligne de note table correspond soit un tweet soit à un commentaire (réponse à un tweet). On a choisi de laisser ces deux objets dans la même table en gardant en tête que l'attribut 'Tweet_reply_to_tweet_id' permet de les distinguer :
  - si 'Tweet_reply_to_tweet_id' = Null : la ligne correspond à un tweet
  - 'Tweet_reply_to_tweet_id' != Null : la ligne correspond à un commentaire à un tweet

2) Extraction des données YouTube

a. Méthodologie et API utilisés

Après avoir créé une application YouTube, on utilise l'API « GoogleApiClient » afin de récupérer les vidéos et les commentaires correspondants aux mots-clés voulus. On a commencé par tester avec le mot-clé « Renault ».

Voici un extrait (Figure 10) de l'ensemble des données auxquels on a accès grâce à l'API :

{'kind': 'youtube#searchResult', 'etag': '"XpPGQXPnxQJhLgs6enD_n8JR4Qk/0TEC5-yM_evuYEIct3y7AOqbv4Q"', 'id': {'kind': 'youtube#video', 'videoId': 'sbeCCVgZ0Ss'}, 'snippet': {'publishedAt': '2013-06-18T19:43:18.000Z', 'channelId': 'UCZHxCShu3eQwMOdWi9M_lsQ', 'title': 'Nuevo Renault Master. Todo el power para tu negocio.', 'description': 'El nuevo Renault Master ofrece un montón de ventajas pensadas específicamente para tu negocio. Tanto que, de hecho, podés convertirte en el Master of ...', 'thumbnails': {'default': {'url': 'https://i.ytimg.com/vi/sbeCCVgZ0Ss/default.jpg', 'width': 120, 'height': 90}, 'medium': {'url': 'https://i.ytimg.com/vi/sbeCCVgZ0Ss/mqdefault.jpg', 'width': 320, 'height': 180}, 'high': {'url': 'https://i.ytimg.com/vi/sbeCCVgZ0Ss/hqdefault.jpg', 'width': 480, 'height': 360}}, 'channelTitle': 'RenaultArg', 'liveBroadcastContent': 'none'}}

Fig. 10 : Format d'un exemple d'élément extrait par l'API de YouTube

Cet API nous permet à la fois d'accéder aux données :

- Des vidéos avec la méthode youtube.search()
- Des commentaires des vidéos avec la méthode youtube.commentThreads()

On a donc choisi de construire, pour chaque marque, deux tables :

- Une table contenant les données des vidéos : « Youtube_Renault », « Youtube_Suzuki », « Youtube_Hyundai », « Youtube_Tatamotors », « Youtube_Mahindra »
- Une table contenant les données des commentaires des vidéos : « Youtube_comments_Renault », « Youtube_ comments_Suzuki », « Youtube_ comments_Hyundai », « Youtube_ comments_Tatamotors », « Youtube_ comments_Mahindra ».

Il est essentiel de noter que la clé de jointure entre ces deux tables est l'id de la vidéo (attribut « video_id »).

b. *Choix d'implémentation pour l'extraction des vidéos*

Les méthodes crées sont les suivantes :

- *collect_youtube()* : cette méthode permet de collecter 50 vidéos contenant le mot-clé recherché et renvoie un dataframe où chacune des ligne correspond à une vidéo et les colonnes correspondent aux différentes données qui nous intéressent.
- *create_table()* : cette méthode permet de créer une table dans la base de données AWS. On spécifie ici le type de chaque attribut.
- *enrich_table(dataframe)* : cette méthode prend en argument un objet de type dataframe et enrichie la table - préalablement crée – avec ce dataframe.
- *show_table()* : cette méthode permet l'affichage d'une portion de la table ainsi que de ses dimensions.

On obtient alors la table relationnelle suivante (Figure 11) pour les vidéos YouTube:

Attribut	*Type*	*Explications*
Video_id	Varchar(500)	Id de la vidéo
Channel_id	Varchar(500)	Id de la chaine
Title	Varchar(500)	Titre de la vidéo
Date	Timestamp	Date de publication
Description	Varchar(500)	Description de la vidéo
viewCount	Varchar(500)	Nombre de vues de la vidée
likeCount	Varchar(500)	Nombre de likes (j'aime) de la vidéo
dislikeCount	Varchar(500)	Nombre de dislikes (j'aime pas) de la vidéo
favoriteCount	Varchar(500)	Nombre de fois que la vidéos a été mise favorite
commentCount	Varchar(500)	Nombre de commentaires de la vidéo

Figure 11 : Tables des vidéos extraites de YouTube.

c. *Choix d'implémentation pour l'extraction des commentaires des vidéos*

Les méthodes crées sont les suivantes :

- *collect _youtube_comments ()* : cette méthode permet de collecter les commentaires des vidéos contenant le mot-clé recherché et renvoie un dataframe où chacune des ligne correspond à un commentaires et les colonnes correspondent aux différentes données qui nous intéressent.
- *create_table_ comments()* : cette méthode permet de créer une table dans la base de données AWS. On spécifie ici le type de chaque attribut.
- *enrich_table_comments (dataframe)* : cette méthode prend en argument un objet de type dataframe et enrichie la table - préalablement crée – avec ce dataframe.
- *show_table_comments ()* : cette méthode permet l'affichage d'une portion de la table ainsi que de ses dimensions.

On obtient alors la table relationnelle suivante (Figure 12) pour les commentaires des vidéos YouTube:

Attribut	*Type*	*Explications*
comment_id	Varchar(500)	Id du commentaire
video_id	Varchar(500)	Id de la vidéo
author	Varchar(500)	Auteur du commentaire
Comment_text	Varchar(500)	Texte du commentaire
Date_publication	Timestamp	Date de publication du commentaires
likeCount	Varchar(500)	Nombre de likes (j'aime) du commentaire

Figure 12 : Table des commentaires des vidéos extraites de YouTube.

d. Tests

On obtient la table suivante (Figure 13) pour le mot-clé « Renault » :

	title	date
0	Renaud - Toujours debout (Clip officiel)	2016-02-26 10:54:44
1	RENAULT CAPTUR	2017-11-16 09:13:01
2	Renaud - Mistral gagnant (Clip officiel)	2016-03-16 15:59:15
3	Richard drives a F1 car round Silverstone - To...	2008-11-08 01:55:20
4	Renault Espace Convertible Challenge - Top Gea...	2009-01-10 02:14:37
5	Crash Test Renault	2007-11-29 13:06:37
6	Renault DUSTER | The True SUV	2019-02-06 08:16:37
7	Peugeot 208 GTi vs Renault Clio 200 Vs Ford Fi...	2014-02-09 21:00:01
8	Renault SPORT | 40 anos na Fórmula 1 | Narraçõ...	2017-11-08 23:34:43
9	AdBuster - Renault Megane Destroy!	2013-06-24 16:44:48
10	Renault y Thule unieron tus pasiones en un sol...	2018-03-21 15:14:36
11	Renault KWID - India's New Favourite Car	2016-07-01 11:33:58
12	Renault KWID 1.0L - Bookings Open	2016-08-22 13:21:02
13	Renault LIMITED | protagonizado por Jonathan K...	2018-11-07 09:36:34
14	Renault Captur – Le crossover urbain aux multi...	2013-03-05 14:11:21
15	Renault Duster 4x4 off-road Heavily Stuck in M...	2015-10-01 21:03:30
16	La technologie du moteur champion du monde dan...	2013-11-28 09:00:36
17	Renault F1 engine playing God Save the Queen	2006-07-15 10:14:05
18	Charlie Bit the family Renault ZOE EV	2013-12-06 07:07:12
19	Hyundai Creta vs Renault Duster | The Perfect ...	2015-12-22 14:27:30

Fig. 13 : Portion d'une table de données des vidéos Youtube avec le mot-clé Renault

Cet exemple nous montre la nécessité de désambiguïser nos données, puisque plusieurs lignes correspondent à des clips du chanteur Renault et n'ont aucun lieu avec la marque automobile française.

e. Problèmes et solutions

- Le choix des types de chacun des attributs a été également difficile, puisqu'il s'agit de le rendre le plus flexible possible, pour l'ensemble des données possibles.
  On a opté pour le type *« varchar(500) »* qui est normalement en deçà de la taille des données possiblement récoltés.

- Les limites intrinsèques à l'API « GoogleAPIclient » ont été contournées en réalisant un requêtage automatique et périodique, avec un délai de pause entre de requêtes suffisamment grandes.

- Le stockage de notre dataframe dans la base de données passe par la conversion du dataframe en un objet *IoString* où les différents attributs sont séparés par une tabulation \t.
  Afin de lever toute ambiguïté, on a traité le texte (description des vidéos et texte des commentaires) en retirant les caractères de type tabulation.

3) *Extraction des données Facebook*

Pour obtenir les données de Facebook, il faut se heurter à de nombreuses limitations, et trouver un moyen de les contourner en restant dans les frontières fixées par la loi. Facebook est très protecteur des données de ses utilisateurs, c'est pourquoi les données récupérées sont toutes anonymisées. Par ailleurs, il n'existe pas d'API sur python qui fonctionne pour récupérer les données de Facebook qui nous intéressent, il a donc fallu utiliser le logiciel de Microsoft : PowerBI.

a. Méthodologie

PowerBI présente une interface peu ergonomique pour récupérer les données de Facebook, mais il permet de récupérer les éléments suivants :

- Il faut dans un premier temps choisir le nom d'une page publique, celle dont le contenu nous intéresse. Il n'est pas possible de choisir un mot clef, et de récupérer les publications et commentaires qui le contiennent.
- Pour la page en question, 3 éléments sont accessibles :
    1. Le « feed » : les publications faites sur cette page par les utilisateurs de Facebook.
    2. Les « likes » : les pages publiques que la page en question like.
    3. Les « posts » : les publications réalisées par la page.
- Ce troisième élément est celui qui nous est le plus utile. On peut également récupérer tous les commentaires réalisés sur ces posts. Le reste des données (likes, réactions, etc…) n'est pas accessible.

b. Format de la table

Ces données sont ensuite exportées au format CSV, puis stockées via python et sa librairie pandas, dans un DataFrame dont un aperçu est le suivant (Figure 14) :

	Date_post	Id_post	Post	Date_comment	Id_comment	Comment
0	2019-01-22T13:51:13+0000	223968877680152_2002854476458241	Un design sublimé : Nouvelle Renault TWINGO se...	2019-01-22T14:08:12+0000	2002854476458241_2002872883123067	Et dedans ya du changement? \n\nPour certain.....
1	2019-01-22T13:51:13+0000	223968877680152_2002854476458241	Un design sublimé : Nouvelle Renault TWINGO se...	2019-01-22T16:33:26+0000	2002854476458241_2003037573106598	Perso proprio de la twingo edition one avec to...
2	2019-01-22T13:51:13+0000	223968877680152_2002854476458241	Un design sublimé : Nouvelle Renault TWINGO se...	2019-01-22T17:39:38+0000	2002854476458241_2003105469766475	Justine Peytout 🚗
3	2019-01-22T13:51:13+0000	223968877680152_2002854476458241	Un design sublimé : Nouvelle Renault TWINGO se...	2019-01-22T18:07:21+0000	2002854476458241_2003138893096466	La Twingo 1 reste la meilleure
4	2019-01-	223968877680152_2002854476458241	Un design sublimé : Nouvelle Renault TWINGO	2019-01-	2002854476458241_2003156953094660	Cedric Messiaen prépare un bon

Fig. 14 : Aperçu du DataFrame contenant les données extraites de Facebook.

Elles sont ensuite exportées, comme pour les 2 autres réseaux sociaux, vers la base de données AWS, et la table peut être mise à jour via des fonctions similaires.

La difficulté à automatiser le processus d'extraction des données de Facebook, s'ajoutant à la grande limitation des données extractibles et à l'anonymisation de ces données, nous a finalement poussé à ne pas poursuivre ce réseau social dans la suite de notre étude.

4) Automatisation de la collecte et données collectées

Bien que l'implémentation de l'extraction automatique des données n'est pas été complètement effectuée, celle-ci a été rendue facile pour un futur utilisateur, pour les réseaux Twitter et YouTube.

L'automatisation repose sur l'utilisation de la librairie « crontab » permettant d'exécuter automatiquement des scripts, des commandes ou des logiciels à une date et une heure spécifiées à l'avance, ou selon un cycle défini à l'avance.

Les scripts permettant de collecter les données et de mettre à jour les tables étant déjà existant, il suffira d'utiliser crontab pour planifier une mise à jour régulière. Il est recoomandé d'utiliser une machine fonctionnant sous Linux.

Pour les données que nous avons extraites tout au long de l'étude, l'extraction a été manuelle et ainsi les données ne sont pas continues dans le temps et sont parfois légèrement disparates.

On retrouve Figure 15 la taille (nombres d'instances) des tables pour les différents réseaux et les différentes marques.

	Renault	Suzuki	Mahindra	Hyundai	Tatamotors
Twitter (posts)	22 162	11 957	6 558	2 558	3 240
Twitter (commentaires)	3 139	6 537	1 161	204	2 742
YouTube (vidéos)	3 900	800	600	600	50
YouTube (commentaires)	15 494	3 122	13 665	15 358	17 805
Facebook (renault_fr)	10 000	10 000	10 000	10 000	10 000

Figure 15 : Tailles des datasets

IV. Désambiguïsation des données

1) Objectif

La désambiguïsation lexicale est la détermination du sens d'un mot dans une phrase lorsque ce mot peut avoir plusieurs sens possibles (exemple : Renault désigne à la fois la marque de voiture mais aussi le nom de famille d'un chanteur français). L'objectif est donc de remettre les informations dans leur contexte, et de ne traiter que les informations pertinentes pour notre étude.

2) Choix d'implémentation

Pour cela, on utilise une fonction qui :

- Prend en entrée : la table à traiter, les colonnes de la table à désambiguïser, une liste de mots « négatifs » et une liste de mots « positifs ».
- Envoie en sortie : la table avec uniquement les lignes qui ne sont hors de contexte.

La liste de mots négatifs doit être fournie par l'utilisateur et contient tous les mots jugés hors-contexte.
La liste de mots positifs doit également être fournie par l'utilisateur et contient les ancres linguistiques, c'est-à-dire les mots qui nous permettent de garder une ligne de la table même si elle contient un mot négatif.
Par exemple, pour notre étude sur la marque Renault on peut proposer les deux listes suivantes :

L mots négatifs = [chanson, chanteur, Renaud]

L mots positifs = [voiture, automobile]

L'algorithme utilisé repose sur l'arbre de décision suivant (Figure 16) :

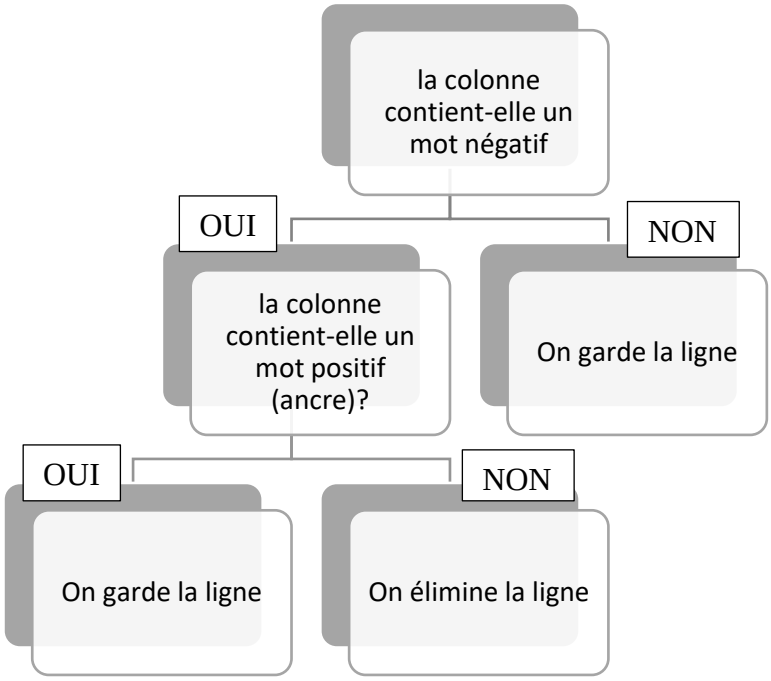

Figure 16 : Arbre de décision de l'algorithme de désambiguïsation

Avant d'utiliser cet algorithme, on commence par filtrer selon la langue : on ne conserve que les tweets en langue anglaise (Figure 17).

	Renault	Suzuki	Mahindra	Hyundai	Tatamotors
Twitter (posts)	40 %	41 %	89 %	51 %	90 %
Twitter (commentaires)	40 %	20 %	85 %	67 %	85 %

Figure 17 : Pourcentage des données en langue anglaise

On applique ensuite l'algorithme, comme le montre la figure 18.

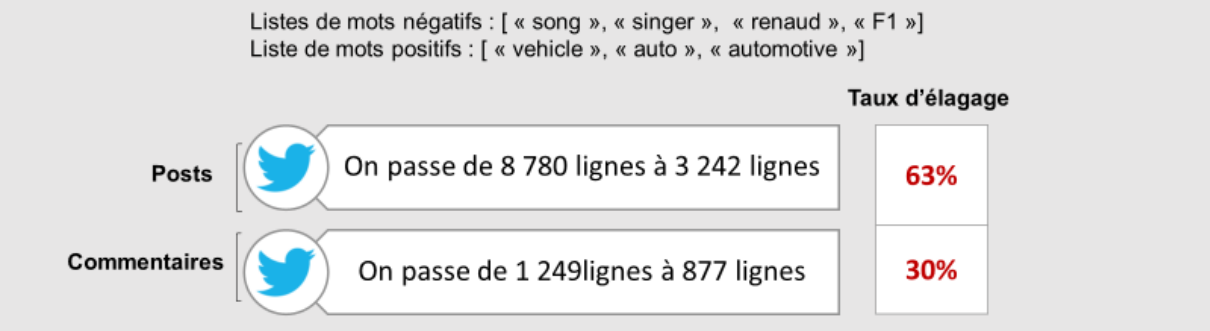

Figure 18 : Résultats de l'algorithme de désambiguïsation

V. Analyse des données

1) Analyse des sentiments

a. Objectif

L'analyse du sentiment, aussi appelé opinion mining, est un processus qui permet de déterminer la tonalité émotionnelle qui se cache derrière une série de mots. Cette analyse est utilisée pour mieux comprendre la perception, les opinions et les émotions exprimées dans une chaine de caractères.
L'analyse du sentiment est extrêmement utile en veille des médias sociaux car elle permet d'obtenir une vue d'ensemble sur l'opinion du public au sujet de certains thèmes.
Les utilisations de l'analyse du sentiment sont à la fois vastes et puissantes.
Être en mesure de rapidement comprendre les attitudes des consommateurs et de réagir en conséquence a été extrêmement utile pour l'équipe canadienne d'Expedia qui a observé une hausse soudaine des avis négatifs quant à la musique utilisée dans l'une de leur publicité télévisée.

b. Choix d'implémentation

On utilise l'analyse des sentiments des différents textes extraits (tweets, commentaires, retweets, posts etc.) Pour cela, on utilise la bibliothèque Python TextBlob qui permet le traitement de données textuelles.
En particulier, on importe :

- Blobber from textblob
- PatternTagger, PatternAnalyzer from textblob_fr (version en français)

Ainsi, on associe à chaque donnée textuelle un couple de valeurs :

- La première coordonnées correspondant à la polarité , il s'agit d'une valeur (type float) comprise entre -1(négatif) et 1(positif)
- La seconde coordonnée correspond à la subjectivité du texte analysé, il s'agit d'une valeur (type float) comprise entre 0(très objectif) et 1 (très subjectif).

On a choisi d'exploiter uniquement la polarité (première coordonnée).

c. *Limites*

Étant donné que l'API utilisée s'appuie sur un corpus de mots limité, l'analyse du sentiment a ses limites et ne peut pas être fiable à 100 %.
Tout comme n'importe quel procédé automatisé il y a des risques d'erreur, et l'œil humain est souvent nécessaire pour s'assurer de la justesse de l'analyse.

2) *Nuage de mots / WordCloud*

a. *Objectif*

Le nuage de mots va recenser tous les mots utilisés par le public et mettre en valeur les plus utilisés. Il permet donc de faire ressortir le vocabulaire le plus fréquent.

On a décidé de réaliser des nuages de mots sur les réseaux sociaux :

- Twitter distinguant les tweets eux-mêmes (texte du tweet et hashtags) et les commentaires (texte du commentaire et hashtags)
- YouTube (texte commentaires des vidéos)

L'objectif est de faire ressortir les mots les plus utilisés à la fois dans les posts et les commentaires pour :

- Avoir une idée des mots-clés associés à notre web-réputation
- Savoir quels sont les mots associés aux opinions négatives et les mots associés aux opinions positives
- S'inspirer des certains mots-clés à opinion positive pour la rédaction de nouveaux posts

b. *Choix d'implémentation*

Pour afficher les nuages de mots, nous utilisons des bibliothèques reposant sur la librairie wordcloud de python. Cette librairie permet de créer des nuages de mots au format de plot (librairie matplotlib de python) il faut ensuite utiliser d'autres librairies comme io pour convertir ces nuages de mots au format png. Elle permet également de donner une forme et des couleurs aux mots, suivant un « mask » qui est une image que l'on donne en argument.

c. *Problèmes et solution*

La librairie est difficile à prendre en main du fait du grand nombre de librairies sous-jacentes pas forcément interopérables.
Par ailleurs, il nous a été impossible de générer des nuages de mots interactifs et filtrables (comme initialement prévu). Cependant, nous avons pu associer aux nuages de mots un histogramme des fréquences des mots et bigrams utilisés, qui lui est filtrable.

Un exemple sur le texte de Moby Dick est donné Figure 19 pour illustrer les propos.

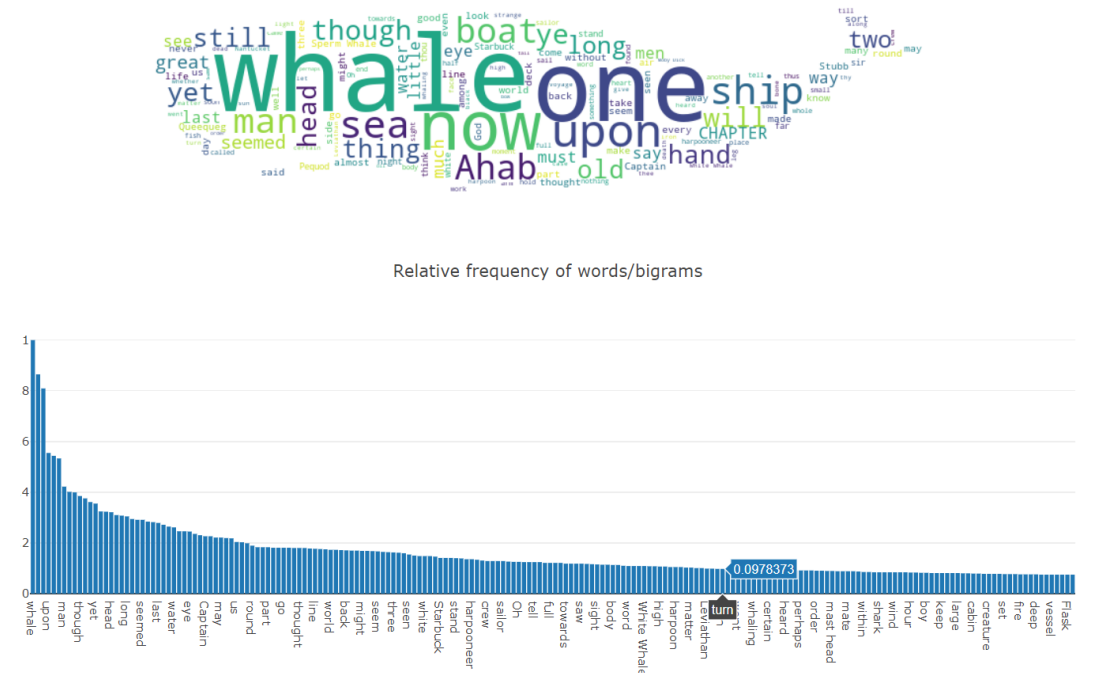

Fig. 19 exemple d'histogramme des fréquences pour un nuage de mot lié au texte de Moby Dick.

3) *Influenceurs*

a. *Objectif et réflexion*

Dans cette partie, nous souhaitons mesure l'influence des tweets en mesurant leur audience. Par cette analyse, le but est alors de représenter un classement des utilisateurs de Twitter les plus influents selon l'opinion (ou le sentiment) global exprimé par les Tweets.
Autrement dit, pour l'opinion globale de l'utilisateur (négative, positive ou neutre) fixé, nous allons trouver les utilisateurs qui ont les tweets les plus influents sur le sujet recherché par TwitterSearch.
Pour mesurer l'influence d'un Tweet, nous ne regardons que son audience, c'est à dire les personnes qui sont susceptibles de lire le tweet en question. Ceci est surtout représenté par :

- les followers de l'utilisateur (utilisateur signifiera toujours celui qui a émis le tweet initial)
- les retweeters qui ont retweeté le tweet initial
- les followers des retweeters (qui sont informés du tweet de la même manière que les followers de l'utilisateur).

Nous allons donc utiliser un score d'audience du tweet noté s qui mesure l'influence de ce tweet. On utilise les notations suivantes :

- F désigne l'ensemble des followers de l'utilisateur
- R l'ensemble des retweeters du tweet
- pour r un retweeter, Fr l'ensemble des followers du retweeter r, alors on définit arbitrairement la relation :

$$s = 5\,card\,(R) + card(\,(F \cup \bigcup_{r \in R} Fr)\,/R\,)$$

On la justifie de la manière suivante : le poids est plus important (5 points) pour les retweeters car en retweetant, on est sûr que le retweeter a lu le tweet (car il l'a repris), ce qui n'est pas forcément le cas des followers. Tous les autres followers (privés des retweeters pour éviter de compter ces derniers 2 fois) valent tous 1 point pour le score d'audience car ils sont touchés de la même manière par le tweet.

Enfin, nous pouvons établir les classements d'influence de l'utilisateur de la manière suivante :

1) Pour chaque utilisateur, on somme les opinions des tweets qu'il a émis (donné avec l'API textblob) pour connaître le sentiment global de l'utilisateur. On notera que pour un tweet, 1 équivaut à un tweet d'opinion positive, -1 équivaut à un tweet d'opinion négative et 0 équivaut à un tweet d'opinion neutre. Ainsi, le signe de la somme des opinions donne l'avis global de l'utilisateur (neutre si =0, positif si >0, négatif si <0).

2) Pour chaque utilisateur, on calcule la moyenne des scores des tweets et on garde uniquement les 50 meilleurs scores moyens pour chaque opinion

3) Pour chaque utilisateur, on calcule la somme des scores des tweets. On prend les 5 meilleurs sommes pour chaque opinion parmi les 50 filtrées précédemment.

4) Notation d'un post futur

a. Objectif

Une des taches principales des directeurs marketing et community managers est de faire parler de leur marque sur les réseaux sociaux, de prévenir les utilisateurs de la sortie de nouveaux produits, en faisant mieux que leurs concurrents.
Pour les aider à faire des posts compétitifs et leur permettre d'avoir une première idée de comment leur contenu sera perçu sur les réseaux avant même qu'ils ne le mettent en ligne, nous proposons de développer un outil de notation d'un post futur. Cela permettra, avant de publier un contenu, de le tester en le comparant à tous les posts passés, et d'obtenir un score pour ce post potentiel, indiquant s'il s'agit ou non d'une bonne idée de leur mettre en ligne.

b. Idée d'implémentation

La première étape pour développer cet outil est donc de définir un score pour chaque post. Ce score doit refléter le niveau global d'un post et de son impact. Il doit donc prendre en compte de nombreux facteurs :

- Le « buzz » créé : est-ce que ce post fait réagir ? Est-il vu par de nombreuses personnes ? A-t-il un grand nombre de like/retweet/commentaires ?
- Le sentiment créé : comment ce post fait-il réagir ? Il faut absolument éviter que ce post provoquent des sentiments négatifs.
- Les personnes qui interagissent avec ce post : y a-t-il des influenceurs ? des clients potentiels ?

Nous pouvons poser sans pour autant perdre de généralité, que ce score sera un nombre compris entre -10 (post à éviter) et +10 (post à faire absolument). Nous n'avons pas implémenté ce score.

La deuxième étape est d'attribuer à chaque post passé, fait par la marque ou par ses concurrents directs, son score selon le modèle que nous venons de définir. Ce score fera office pour la suite d'étiquettes (label) pour chaque post que nous considérions dès lors comme des instances.

La dernière étape consiste à utiliser un algorithme de classification basé sur le Machine Learning afin d'attribuer un score à des posts n'ayant pas été mis en ligne. Pour ces posts, impossible d'attribuer un score car nous ne disposons pas encore des réactions qu'il a entrainées. Ainsi en se basant sur les posts passé, et l'algorithme développé nous faisons une prédiction de ce que ce score serait.

D'un point de vue utilisateur, il suffira de rentrer dans l'outil le texte ainsi que les différents paramètres du futur post potentiel, avant d'obtenir un indice sur la pertinence d'un tel post. Faute de temps nous n'avons pas mis en place cet outil.

VI. Visualisation et mise en forme des données

1) Objectif

L'objectif de cette partie est de permettre la visualisation de l'ensemble des graphes et analyses faites auparavant dans le but de reproduire le prototype de l'outil présenté dans la partie « Dashboard cible » de ce rapport.

2) Présentation de Dash

On utilise la bibliothèque Python nommée Dash qui permet de construire des applications.
Celle-ci permet de:

- Explorer des données
- Construire et modifier des modèles /graphes
- Construire sa propre plateforme de business intelligence.

Le choix de cette librairie pour implémenter le Dashboard résulte de deux facteurs principaux.

1. C'est la libraire que les étudiants de la coding week ont utilisée et c'est donc logiquement la librairie que TNP nous a recommandé d'utiliser.
2. C'est une librairie qui parait très documentée et qui permet d'atteindre des résultats satisfaisants sur des projets simples très rapidement.

Malheureusement, la prise en main de cette librairie nous a posé de nombreux problèmes, car il s'est avéré que pour les projets plus complexes, celle-ci n'était pas aussi ergonomique que prévu.

Le principal avantage de Dash est qu'elle est « No Javascript required », et ainsi il n'y a pas besoin d'apprendre le language assez connu et reconnu pour le développement d'application qu'est JavaScript. Néanmoins, pour obtenir des résultats esthétiques, il faut se plonger dans des langages sur laquelle la librairie repose. Ainsi, il nous a fallu apprendre à comprendre html, css, et bootsrtap.

La dernière complexité, repose sur le fait que nous développons une application multipage. Ceci se reflète directement dans l'architecture finale du projet, qu'il nous a mis longtemps à trouver.

3) Choix d'implémentation

Pour éviter d'aller trop profondément dans l'utilisation de html, css et bootstrap, afin d'éviter d'utiliser un trop grand nombre de technologies parfois incompatibles, nous avons décider d'utiliser la sous-librairie de Dash : dash_bootstrap_components.

Cette librairie permet d'améliorer l'esthetique de l'app. Malheureusement, cette libre est encore en cours de développement et ainsi, il nous a fallu la mettre à jour régulièrement et certaines composantes n'était pas encore stable.

4) Aperçu du Dashboard final

Voici (Figure 20) un aperçu de l'outil finalement développé. On peut retrouver le code et ainsi essayer l'outil localement, dans les annexes.

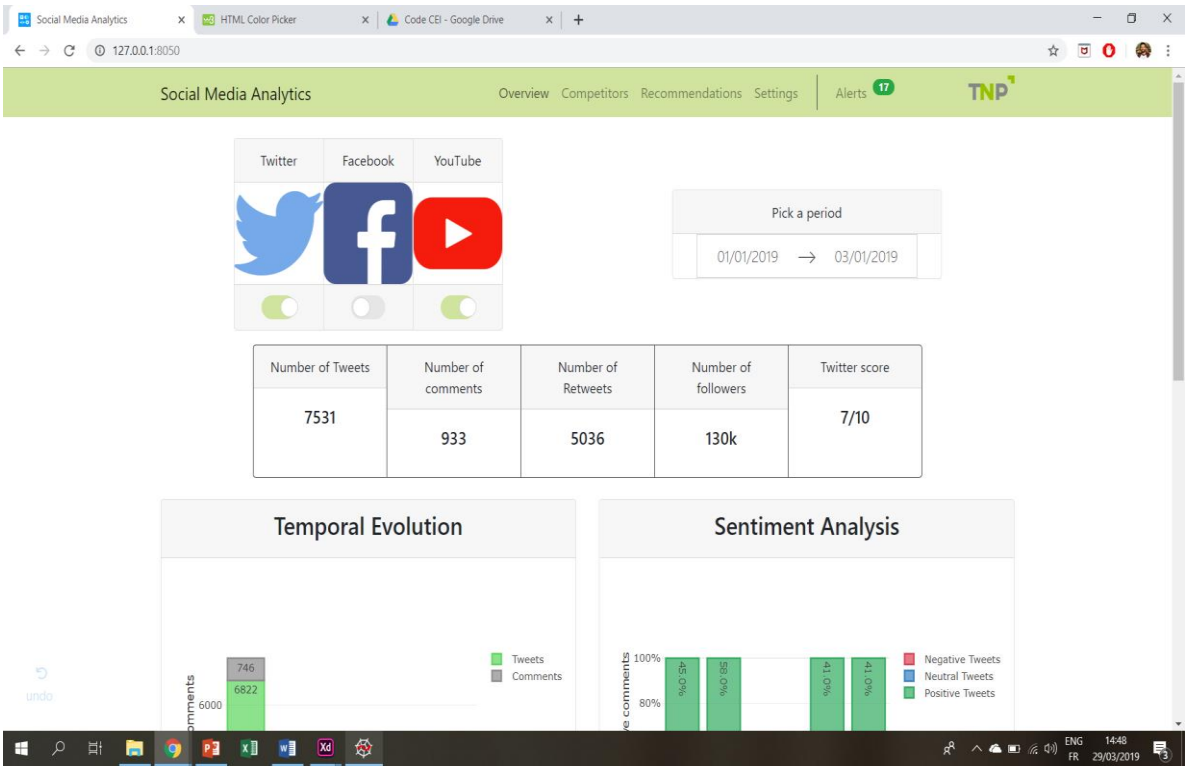

Figure 20 : Aperçu du Dashboard (page overview)

VII. Conclusion

Ce projet nous a permis de développer un outil qui permet de mesure la réputation d'une marque ou d'un produit sur les réseaux sociaux (essentiellement Twitter et YouTube).
Il était composé de 3 axes essentiels :

- La récolte et le stockage des données des réseaux sociaux
- L'analyse des données collectées pour créer des recommandations marketing et stratégiques
- La visualisation des résultats obtenus dans un tableau de bord efficace

Il nous a également permis d'entreprendre un projet du début à la fin en utilisant une multitude d'outils : Python, Dash , AdobeXD, Trello etc.
Nous avons réussi à construire un outil qui n'est certes pas une copie conforme du prototype cible mais qui est fonctionnel et qui permet d'obtenir des résultats cohérents.

Les principales limites qu'on a rencontrées sont :

- Les limites intrinsèques des API utilisés (TwitterSearch et GoogleApiClient)
- Les limitations des bibliothèques Python (analyse de sentiment)
- La prise en main de Dash

Les axes d'amélioration de notre travail :

- Automatiser la récolte et le stockage des données dans les tables
- Implémenter toutes les fonctionnalités (ranking des influenceurs etc.)
- Améliorer l'ergonomie du tableau de bord Dash

VIII. Annexes

1) Annexe 1 : Planning et documents de suivis.

Cf. fichiers joints

2) Annexe 2 : Code

Le code se trouve dans le projet Github à l'adresse suivante :
https://github.com/RaphKern/CEI-Social-Media-Analytics