

# RAPPORT DE STAGE D'APPLICATION

**Organisme d'accueil : Moroccan Center for Innovation  
and Social Entrepreneurship (MCISE)**

**Automatisation des processus de  
traitement des CV en utilisant le  
Machine Learning**

**Préparé par :**

*M.FEKHARI Abderrahmane*

*M.JABBAR Abdelkadir*

**Sous la direction de :**

*M.BENDIDI Ihab*

*M.ADDIOUI Adnane*

Année Universitaire : 2019-2020



# *Dédicace*

*A nos chers parents Pour tous  
leurs sacrifices,*

*A toute nos familles Pour les  
encouragements,*

*A nos amis*

*Pour tous les moments de bonheur que  
nous avons passé ensemble,*

*Nous vous dédions cet humble  
travail*

# *Remerciements :*

Nous tenons tout d'abord à remercier chaleureusement notre encadrant M. Ihab Bendidi pour son soutien, son accompagnement, son esprit de partage et de leadership tout au long de la période du stage.

Nous tenons à remercier également M. Adnane Addioui, pour la confiance, l'accompagnement depuis le début de notre projet de stage d'application et pour l'aide bienveillante sans qui ce projet n'aurait pas pu se concrétiser.

Nous tenons à remercier tout le corps professoral de l'INSTITUT NATIONAL DE STATISTIQUE ET D'ECONOMIE APPLIQUEE, pour leurs conseils précis et précieux.

Que tous ceux et celles qui ont contribué de près ou de loin à l'accomplissement de ce travail trouvent l'expression de nos salutations les plus chaleureuses.

# *Mots-clés:*

Curriculum Vitae

K-Nearest Neighbors

Machine Learning

Classification

Traitement du langage naturel

Term Frequency–Inverse Document Frequency

Bag Of Words

# *Les abbreviations:*

MCISE: Moroccan Center for Innovation and Social Entrepreneurship

CV : Curriculum Vitae

K-NN: K-Nearest Neighbors

TF-IDF: Term Frequency–Inverse Document Frequency

NLP: Natural language processing

# Table des matières

Dédicace .....	3
Remerciements : .....	4
Mots-clés: .....	5
Les abbreviations: .....	6
Table des matières .....	7
Liste des tableaux et des figures : .....	10
Introduction : .....	13
Chapitre 1 .....	14
I. Présentation du Centre Marocain pour l'Innovation et l'Entrepreneuriat Social	14
II. Bureau exécutifs est le suivant : .....	15
III. Equipe MCISE : .....	16
IV. La théorie du MCISE du changement : .....	17
V. Vision et motivation du projet .....	17
VI. Objectifs du projet .....	18
Chapitre 2 : Concepts et outils .....	19
I. Introduction à la Data Science : .....	19
I.1 Qu'est-ce que la data science ? .....	19
I.2 Le cycle de travail du data scientifique : .....	20
II. Introduction au Machine Learning : .....	27
II.1 Pourquoi le Machine Learning ? .....	28
II.2 Quelques applications des Machine Learning : .....	29

II.3	Les différents types d'apprentissage automatiques.....	30
II.4	Régression ou Classification .....	34
II.5	L'algorithme d'apprentissage .....	34
III.	Réseaux de neurones artificiels et deep Learning.....	35
III.1	Définition des réseaux de neurone : .....	35
III.2	Deep Learning .....	36
IV.	Natural language processing (NLP).....	37
IV.1	Aperçu .....	37
IV.2	Difficulté .....	38
IV.3	Tâches du traitement du langage naturel .....	38
V.	Outils technologiques utilisés : .....	42
V.1	GitHub : .....	42
V.2	Python : .....	42
V.3	Jupyter Notebook : .....	43
V.4	Scikit-learn : .....	44
Chapitre 3 : Application .....		46
I.	Représentation générale .....	46
II.	Collecte des données.....	47
III.	Nettoyage des CV .....	47
III.1	Préparation des données .....	48
III.2	Nettoyage des cv : .....	48
IV.	Vectorisation .....	52
IV.1	La transformation Sac de mots (Bag-Of-Words).....	52
IV.2	Term-frequency inverser document frequency : TF-IDF .....	55
V.	Choix de l'algorithme : KNN .....	56



V.1	Présentation et principe de l'algorithme.....	56
V.2	Exemple d'application pour un CV .....	57
VI.	L'approche Train-Test et mesure de l'accuracy .....	59
VII.	Choix du k : nombre de voisin optimal .....	60
	Bibliography/Webography .....	64

# *Liste des tableaux et des figures :*

Figure 1: Bureau exécutif de MCISE .....	15
Figure 2: Equipe MCISE .....	16
Figure 3: Le cycle de travail du data scientist .....	20
Figure 4: Données sur le loyer et la surface .....	22
Figure 5 : La droite de régression correspondant à la modélisation du nuage de points .....	24
Figure 6: L'intervalle de confiance (à 90 %) .....	24
Figure 7: Exemple du quartet d'Anscombe .....	25
Figure 8: Prédiction du prix d'un appartement d'une surface de 30 m <sup>2</sup> .....	27
Figure 9: Donnée d'entrée en format image .....	31
Figure 10: Visualisation 2D d'une méthode non supervisée qui permet de grouper les images par similarité .....	32
Figure 11: La représentation interne des concepts de "visage" et "chat" apprises par un algorithme non supervisé à partir d'images extraites de millions de vidéos YouTube .....	33
Figure 12: Illustration de la différence entre classification linéaire et régression linéaire .....	34
Figure 13: réseaux de neurone réel du corps humain .....	35
Figure 14: réseaux de neurone .....	36
Figure 15: Réseaux de neurone profonds .....	36

Figure 16: Modèle de neurone convolutif .....	37
Figure 17: Hiérarchique des différentes tâches NLP.....	41
Figure 18: Le processus complet de l'approche traditionnelle.....	41
Figure 19: Logo de GitHub .....	42
Figure 20: Logo du Python.....	43
Figure 21: Logo du Jupyter Notebook.....	43
Figure 22: Exemple d'illustration de Python dans la plateforme Jupyter.....	44
Figure 23: Logo de la bibliothèque Scikit-Learn .....	44
Figure 24: Représentation générale de l'application.....	46
Figure 25: Base de données dans Excel.....	47
Figure 26: Dataset « Useful_entities.txt » .....	49
Figure 27: Data Set «names.txt ».....	49
Figure 28: Dataset « unnecessary_entities.txt » .....	49
Figure 29: Code de la conversion du texte .....	50
Figure 30: CV_cleaning.....	51
Figure 31: Code du suppression du ponctations.....	52
Figure 32: Illustration de l'exemple du Sac de mots.....	53
Figure 33: Sac de mots en 2-grammes.....	54
Figure 34: Application de la transformation Tf-Idf.....	56
Figure 35: Représentation graphique de l'algorithme K-NN.....	57
Figure 36: Exemple d'un CV .....	58
Figure 37: Exemple de prédiction du domaine d'un CV .....	59
Figure 38: Mesure de la performance du modèle .....	60
Figure 39: Figure illustrant l'importance du choix du paramètre K .....	61

Figure 40: Représentation graphique de la performance du modèle en fonction du paramètre $K$ .....	62
---	----

# *Introduction :*

Dans le contexte de la Digitalisation dans le monde moderne, l'organisme MCISE a décidé de créer un site Web visant à compenser le rôle partiel du personnel des ressources humaines. En sélectionnant les employés les plus adaptés aux opportunités vacantes, le travail de création du site a été repris par l'équipe technique, et nous avons pris le travail de rédiger un algorithme permettant de classifier les nouveaux CV des candidats et associer chaque CV à chaque domaine.

Ce rapport montre comment le traitement du langage naturel et l'apprentissage automatique peuvent être combinés pour permettre aux demandeurs d'emploi de trouver une correspondance réciproque avec les recruteurs. L'une des méthodes permettant de relever ces défis est d'appliquer le Machine Learning à travers des techniques qui utilisent beaucoup de données pour modéliser les CVs des demandeurs d'emploi et domaines référencée, et d'établir un meilleur appariement entre eux.

Notre rapport se compose de trois chapitres résumant l'avancement de nos travaux :

Le premier chapitre présente l'organisation MCISE, les enjeux, la motivation, les objectifs du projet et les outils qui seront déployés pour atteindre notre objectif.

Le deuxième chapitre présente une introduction générale de data science et ses applications.

Le troisième chapitre présente notre application de traitement des CVs par NLP et le Machine Learning.

# *Chapitre 1 :*

## **I. Présentation du Centre Marocain pour l'Innovation et l'Entrepreneuriat Social**

Fondé en 2012 par un groupe de personnes enthousiastes à l'idée de changement social au Maroc, le Centre Marocain pour l'Innovation et l'Entrepreneuriat Social (MCISE) est une organisation à but non lucratif qui se consacre à la recherche de solutions entrepreneuriales et innovantes à tous les défis sociaux au Maroc.

Ils pensent que soutenir les entrepreneurs sociaux avec des idées de changement de système peut apporter des avantages au Maroc et à la communauté mondiale au sens large.

En 2017, la MCISE a rejoint le réseau Ashoka, un réseau international d'acteurs du changement.

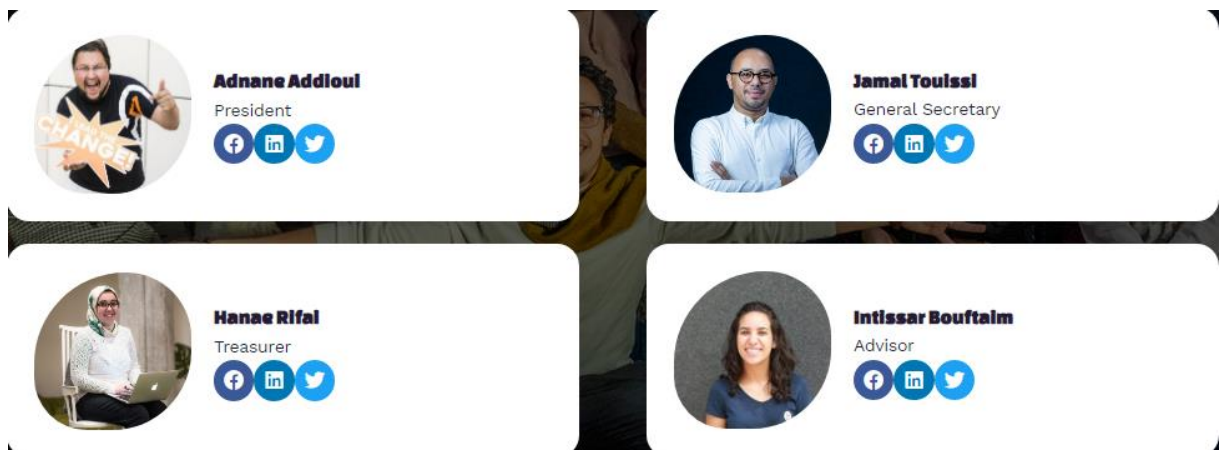
### **Vision Générale :**

Un monde où les idées et les opportunités innovantes sont au service du bien commun.

### **Missions :**

Trouver des solutions innovantes et entrepreneuriales pour chaque défi social au Maroc. Tout en respectant ses valeurs Empathie, Transparence, confiance, Ouverture, Excellence.

## II. Bureau exécutifs est le suivant :



*Figure 1: Bureau exécutif de MCISE*

### III. Equipe MCISE :

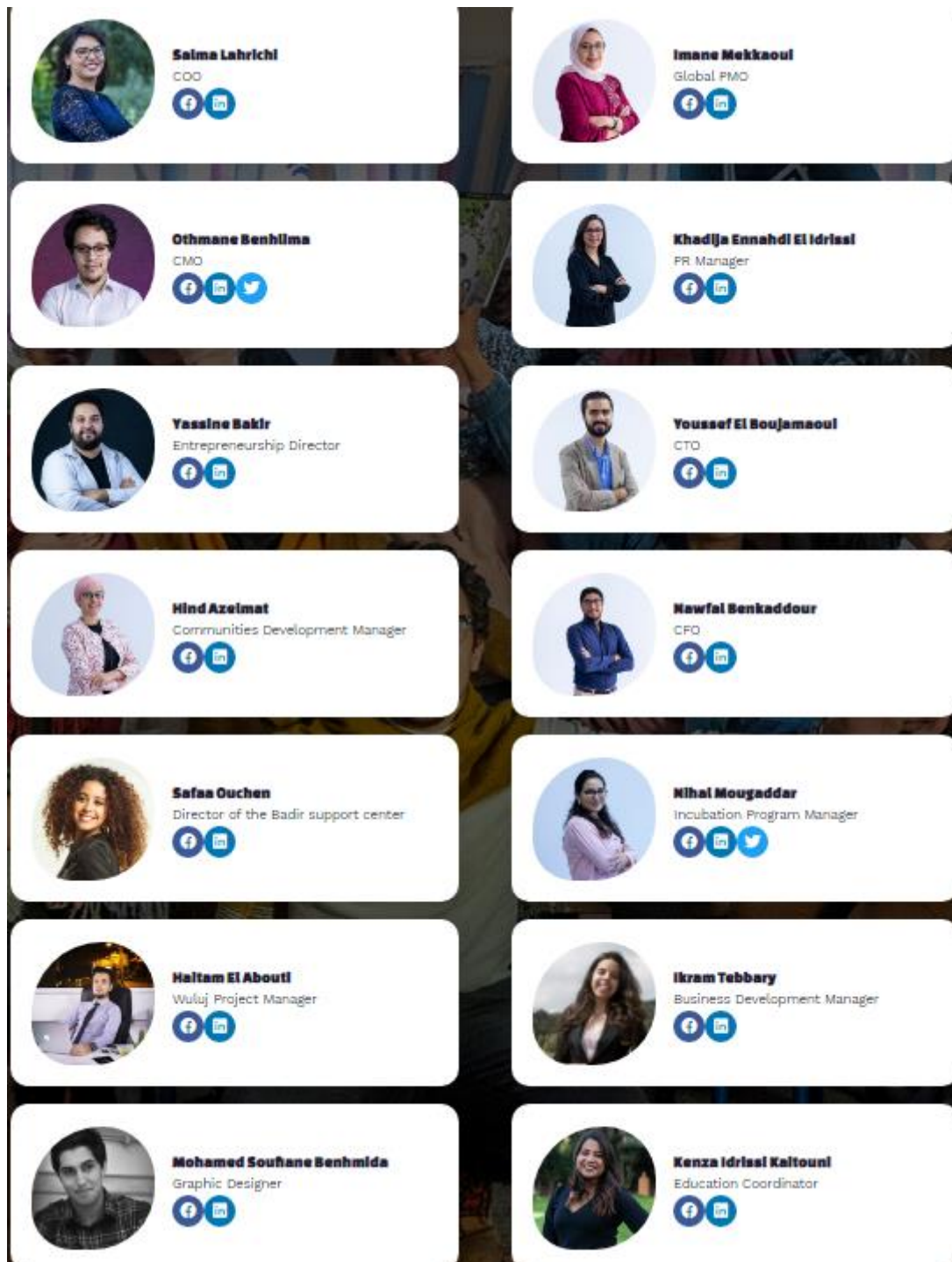


Figure 2: Equipe MCISE



## **IV. La théorie du MCISE du changement :**

### **INSPIRE**

Rendre les idées, les concepts et les croyances sur l'entrepreneuriat social accessibles aux individus, aux organisations, aux entreprises et au gouvernement.

### **ÉDQUER :**

Développer du matériel éducatif et des ressources matérielles pour les étudiants, les chercheurs, les praticiens et les professionnels qui veulent améliorer leurs connaissances de l'entrepreneuriat social.

### **DEVELOPPER :**

Lancer des initiatives, des projets et des programmes qui peuvent accompagner et guider les individus et les organisations vers un avenir d'entrepreneuriat social.

## **V. Vision et motivation du projet**

Dans le cadre de la stratégie de croissance de MCISE dont l'un des objectifs principaux est de se démarquer des autres organismes en réalisant des projets en interne toujours de plus en plus innovants et modernes.

MCISE trouve dans la digitalisation une opportunité à saisir pour développer ses services, un des services importants que MCISE cherche à automatiser est celui du capital humain.

MCISE décide de réaliser un projet ayant pour but de faire un appariement automatisé entre les postes vacants dans des domaines précises et les demandeurs d'emploi.

## **VI. Objectifs du projet**

Parmi les tâches à réaliser pour le bon déroulement du projet, nous citons :

- Participer à la modélisation de processus de recrutement d'une manière orientée données.
- Participer au choix et au développement des outils nécessaires.
- Compréhension de la qualité des données, définition d'hypothèses concernant les données manquantes.

Etudier et proposer des algorithmes de traitement de données.

# *Chapitre 2 : Concepts et outils*

## **I. Introduction à la Data Science :**

La data science (ou science des données en français) et le Machine Learning (ou apprentissage automatique en français) sont deux mots très en vogue lorsque l'on parle de la révolution Big Data, de prédiction des comportements ou tout simplement de la transformation numérique des entreprises. Et comme pour tous les domaines innovants, il est parfois difficile de s'y repérer.

### **I.1 Qu'est-ce que la data science ?**

Le premier objectif du data science est de produire des méthodes (automatisées, autant que possible) de tri et d'analyse de données afin d'en extraire des informations utiles.

Le besoin d'un data science est apparu pour trois raisons principales :

L'explosion de la quantité de données produites et collectées par les humains.

L'amélioration et l'accessibilité plus grande des algorithmes de traitement des données.

L'augmentation exponentielle des capacités de calcul des ordinateurs.

Une entreprise qui a bien intégré la data science sera capable de pondérer les intuitions humaines à l'aide des nouvelles informations suggérées par les données qu'elle possède.

## I.2 Le cycle de travail du data scientifique :

Le cycle de travail du data science peut se résumer par le schéma ci-dessous. Pour faire simple, nous partons de la réalité, nous récupérons les données, nous les nettoyons, nous les explorons puis nous utilisons nos algorithmes pour créer de l'intelligence (artificielle) qui aide à la décision. Dans la suite, nous allons détailler ces différentes étapes et voir quels sont les différents métiers sur la chaîne de traitement de la donnée.

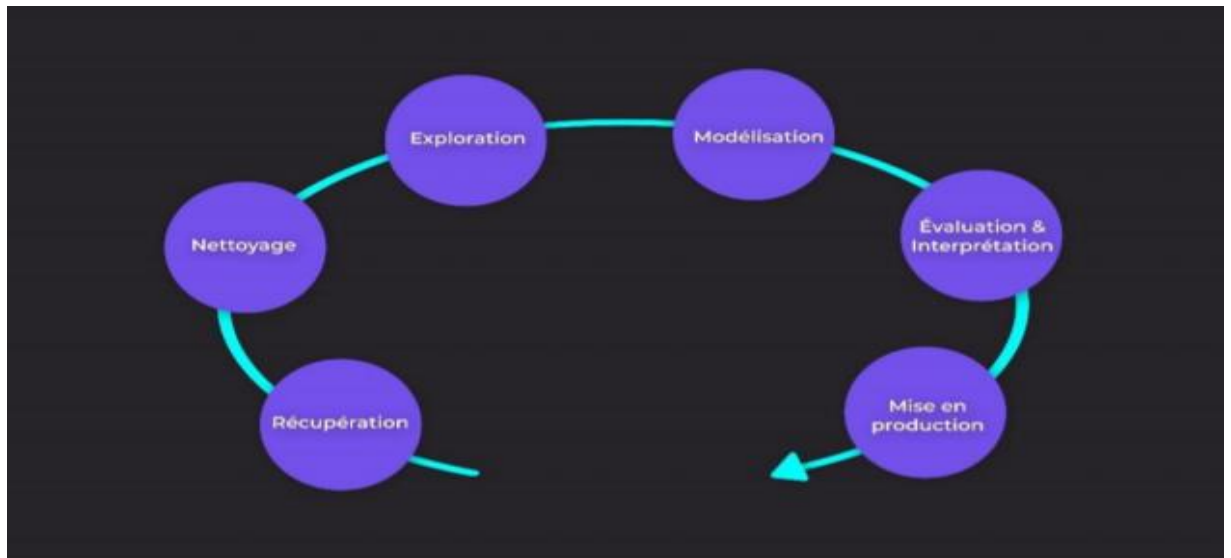


Figure 3: Le cycle de travail du data scientifique

- **Récupération des données :**

Une fois que nous avons décidé d'attaquer un problème, la première chose à faire est d'explorer toutes les pistes possibles pour récupérer les données. En effet, les données constituent l'expérience, les exemples que nous fournissons à notre algorithme de Machine Learning afin qu'il puisse apprendre et devenir plus performant.

- **Nettoyage des données :**

Une fois les données trouvées, il faut passer à l'étape de nettoyage. Nettoyer les données, c'est s'assurer qu'elles sont consistantes, sans valeurs aberrantes ni manquantes.

Une autre étape nécessaire en général est l'agrégation de ces données dans un data. Nettoyer les données signifie donc qu'elles sont toutes sous le même format, accessible au même endroit et au bon moment.

L'important, c'est de bien préparer le terrain pour les étapes suivantes.

Ces étapes seront grandement simplifiées si ce travail fastidieux est bien effectué en amont.

- **Exploration des données :**

Les données bien propres peuvent maintenant commencer à être explorées. Cette étape nous permet de mieux comprendre les différents comportements et de bien saisir le phénomène sous-jacent. Les meilleurs data scientists ne sont pas ceux qui connaissent les algorithmes les plus complexes mais ceux qui ont une très bonne connaissance des données, et ont préparé le terrain avec soin en amont.

À la fin de l'exploration, Nous devons être en mesure de :

Proposer plusieurs hypothèses sur les causes sous-jacentes à la génération du data set : "suite à l'exploration, il y a clairement une relation entre X et Y".

Proposer plusieurs pistes de modélisation statistique des données, qui vont permettre de résoudre la problématique de départ considérée.

Proposer si nécessaire de nouvelles sources de données qui aideraient à mieux comprendre le phénomène.

- **Modélisation des données à l'aide du Machine Learning :**

Nous pouvons enfin rentrer dans la partie la plus intéressante du métier, c'est à dire la création du modèle statistique associé aux données. C'est ce qu'on appelle Machine Learning (ou apprentissage automatique).

En Machine Learning et en data science plus généralement, l'objectif est de trouver un modèle (stochastique ou déterministe) du phénomène à l'origine des données. (Nous allons voir cette notion avec plus de détails par la suite)

C'est-à-dire que nous considérons que chaque donnée observée est l'expression d'une variable aléatoire générée par une distribution de probabilité.

Exemple :

Imaginons que nous voulons savoir si nous payons trop cher notre loyer. Nous avons récupéré sur un site de location une trentaine de prix des locations disponibles, ainsi que la surface associée :

loyer mensuel (en €)	surface (en $m^2$ )
1500	32
2120	65
2500	60
...	...

Figure 4: Données sur le loyer et la surface

Nous pouvons enfin rentrer dans la partie la plus intéressante du métier, c'est à dire la création du modèle statistique associé aux données. C'est ce qu'on appelle Machine Learning (ou apprentissage automatique).

En Machine Learning et en data science plus généralement, l'objectif est de trouver un modèle (stochastique ou déterministe) du phénomène à l'origine des données. (Nous allons voir cette notion avec plus de détails par la suite).

C'est-à-dire que nous considérons que chaque donnée observée est l'expression d'une variable aléatoire générée par une distribution de probabilité.

Bien sûr en réalité d'autres paramètres seraient probablement à prendre en compte (parties communes, voisinage, évolution des loyers au cours du temps, etc). Le but est ici d'appréhender un modèle simplifié afin de comprendre rapidement ce que veut dire "modéliser un phénomène à l'aide du Machine Learning.

Si nous affichons maintenant ces différents points sur un graphe qui représente le montant du loyer en fonction de la surface, on obtient le graphique suivant :

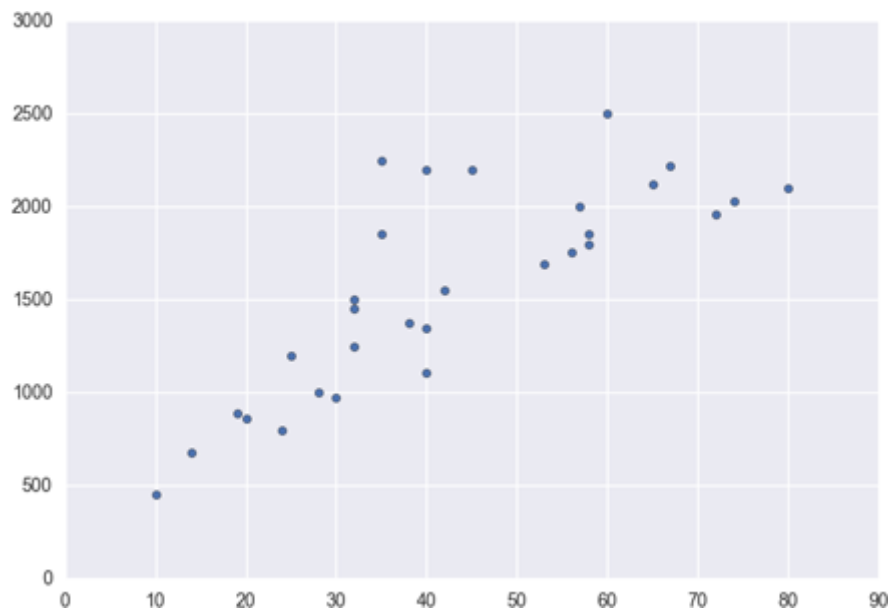


Tableau 1: Loyer mensuel en fonction de la surface du logement

Comme nous pouvions s’y attendre, nous remarquons une augmentation relativement linéaire du loyer par rapport à la surface de l’appartement. Une première modélisation simple du phénomène (le prix du loyer) serait donc simplement de considérer la droite la plus “proche” de l’ensemble des points (régression linéaire).

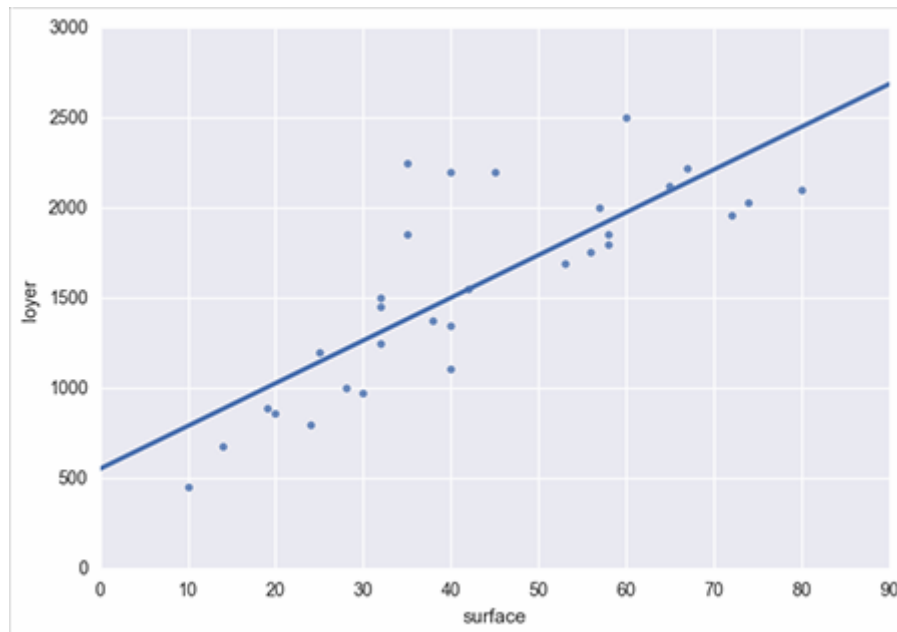


Figure 5 : La droite de régression correspondant à la modélisation du nuage de points

La droite représente donc notre modèle du phénomène, auquel nous pouvons ajouter l'intervalle de confiance dans laquelle nous pensons que se trouve la droite.

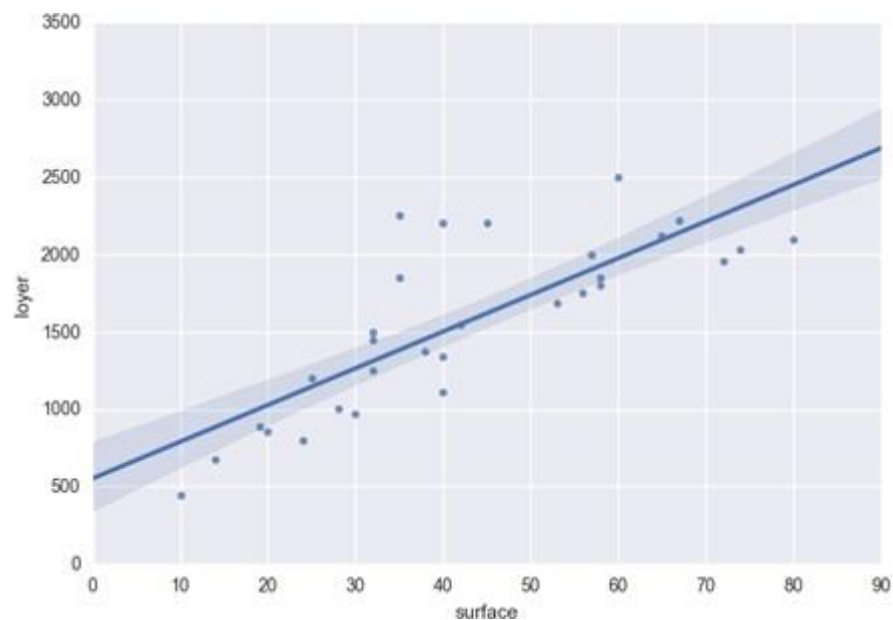


Figure 6: L'intervalle de confiance (à 90 %)



Pour résumer, le travail de modélisation consiste à trouver le bon modèle statistique (ici la droite et son intervalle de confiance) qui s'ajuste le mieux aux données d'exemple.

Le Machine Learning en particulier intervient pour trouver ce modèle de manière automatisée.

- **Évaluez et interprétez les résultats**

Une fois un premier travail de modélisation effectué, la suite de l'étude s'effectue par l'évaluation de la qualité de notre modèle, c'est à dire sa capacité à représenter avec exactitude notre phénomène.

Une représentation connue qui souligne la nécessité de l'évaluation est le quartet d'Anscombe. Il permet de montrer visuellement que pour 4 jeux de données très différents, on obtient la même droite de régression.

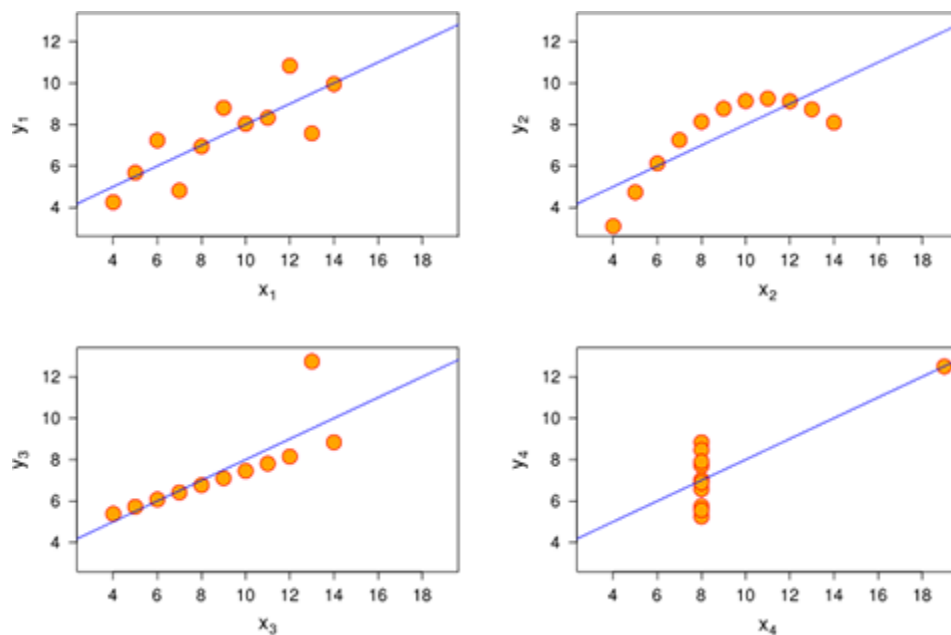


Figure 7: Exemple du quartet d'Anscombe

Le quartet d'Anscombe illustre bien le fait que si nous n'examinons pas assez les données, et nous ne mesurons pas de la bonne manière l'erreur de son modèle, nous pouvons facilement arriver à des aberrations de modélisation.

Il y a parfois clairement un problème dans notre modèle qui ne capture pas l'essence du phénomène. Pour nous aider à évaluer les résultats, mesurer l'erreur de notre modélisation vis-à-vis de nos données d'exemple constitue un premier indicateur de qualité.

Dans les cas ci-dessus, il faudrait clairement changer le modèle d'une droite que nous avons décidé au départ !

C'est donc un jeu d'allers-retours entre modélisation et évaluation qui s'effectue pour obtenir les performances les plus satisfaisantes possibles. Il est même possible dans certains cas de remettre en question certaines hypothèses de départ et de repartir dans une phase d'exploration pour mieux comprendre les données

- **Déploiement du modèle en production :**

Une fois que nous sommes satisfaits de la qualité des performances de notre modèle, nous allons pouvoir passer à l'étape suivante, qui est le rendu de nos résultats et le potentiel déploiement du modèle en production.

Imaginons si nous trouvons que notre modèle d'évaluation des loyers est très performant, et mériterait d'être partagé à plus de monde. Nous décidons donc de le déployer sur un serveur où tout le monde pourra obtenir une estimation de son loyer selon notre modèle, et ainsi déterminer s'il paie plus ou moins que les prix du marché ! Cela l'aidera sûrement dans sa décision de déménager.

Pour ce faire, il suffit de récupérer les paramètres du modèle et de faire passer la surface de l'appartement en entrée du modèle, afin d'obtenir le loyer associé en sortie, en suivant la droite.

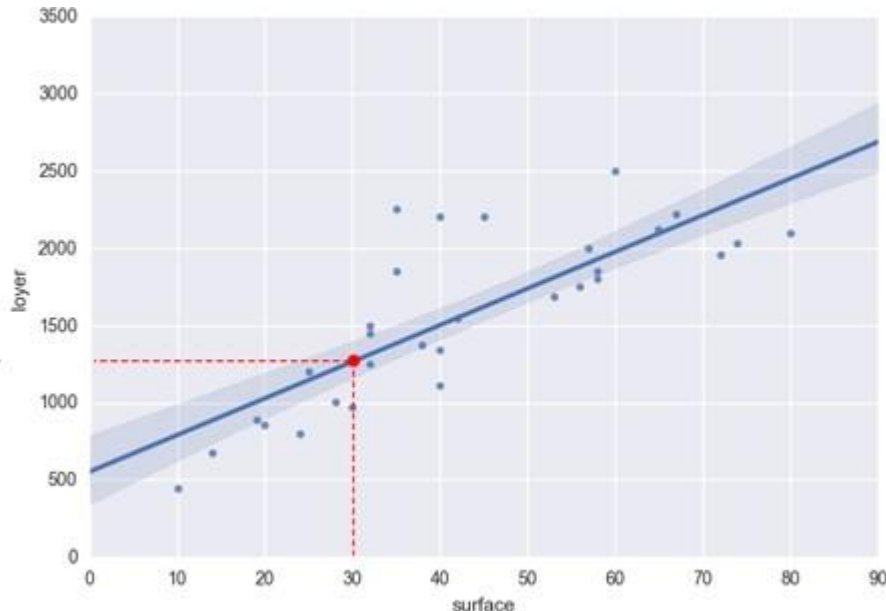


Figure 8: Prédiction du prix d'un appartement d'une surface de 30 m<sup>2</sup>

Par exemple pour un appartement de surface 30 mètres carrés (point en rouge), une estimation légitime du loyer se situerait aux alentours de 1300 euros selon notre modèle.

Pour des modèles plus complexes, le fonctionnement reste le même.

## II. Introduction au Machine Learning :

Le Machine Learning (ou l'apprentissage automatique) est un champ d'étude de l'intelligence artificielle qui se fonde sur des approches statistiques pour donner aux ordinateurs la capacité d'« apprendre » à partir de données, c'est-à-dire d'améliorer leurs performances à résoudre des tâches sans être explicitement programmés pour chacune. L'apprentissage automatique comporte généralement deux phases. La première consiste à estimer un modèle à partir de données, appelées observations, qui sont disponibles et

en nombre fini, lors de la phase de conception du système. L'estimation du modèle consiste à résoudre une tâche pratique, telle que traduire un discours, estimer une densité de probabilité, reconnaître la présence d'un chat dans une photographie ou participer à la conduite d'un véhicule autonome. Cette phase dite « d'apprentissage » ou « d'entraînement » est généralement réalisée préalablement à l'utilisation pratique du modèle. La seconde phase correspond à la mise en production : le modèle étant déterminé, de nouvelles données peuvent alors être soumises afin d'obtenir le résultat correspondant à la tâche souhaitée

## **II.1 Pourquoi le Machine Learning ?**

Les outils analytiques traditionnels ne sont pas suffisamment performants pour exploiter pleinement la valeur du Big Data. Le volume de données est trop large pour des analyses compréhensives, et les corrélations et relations entre ces données sont trop importantes pour que les analystes puissent tester toutes les hypothèses afin de dégager une valeur de ces données.

Le Machine Learning est très efficace dans les situations où les insights doivent être découvertes à partir de larges ensembles de données diverses et changeantes, c'est à dire : le Big Data.

Pour l'analyse de telles données, le Machine Learning se révèle nettement plus efficace que les méthodes traditionnelles en termes de précision et de vitesse.

Le Machine Learning est aujourd'hui utilisé dans de nombreux domaines, comme le développement de véhicules autonomes, les systèmes de recommandations en ligne comme ceux de Netflix et Amazon, l'analyse de sentiments de clients, ou encore la détection de fraude. Avec le Machine Learning, Il est désormais possible de produire rapidement et automatiquement des modèles capables d'analyser plus rapidement des volumes de données plus importants et plus complexes. Ce faisant, les entreprises ont davantage de chances d'identifier des opportunités lucratives ou d'éviter les risques inconnus.

## **II.2 Quelques applications des Machine Learning :**

### Finance – Assurance

- Modélisation d'indicateurs économiques.
- Personnalisation de l'expérience client.
- Evaluation de la solvabilité de l'emprunteur.
- Détection de fraudes.
- Analyse de marché et benchmarking.

### Marketing digital

- Segmentation clients au sein de DMP (Data Management Platform).
- Analyse multicanale.
- Scoring des prospects.
- Optimisation de stratégies SEA (Google AdWords).

### E-commerce

- Analyse de paniers
- Personnalisation de recommandations produits
- Analyse de sentiment sur les réseaux sociaux
- Evaluation de la satisfaction client
- Ventes additionnelles et ventes croisées

### Industrie

- Maintenance prédictive sur les données d'objets connectés (IoT)
- Détermination et ajustement de prix
- Prévion des stocks

- Relations clients (CRM)
- Surveillance de la réputation sur internet

#### Ressources humaines

- Automatisation de recherche de profils (avec le Natural Language Processing)
- Evaluation des risques de départs
- Optimisation du marketing de recrutement

## **II.3 Les différents types d'apprentissage automatiques**

Dans cette partie, on présentera les grandes familles d'algorithmes d'apprentissage existantes :

### Apprentissage "supervisé" ou "non supervisé"

Une première grande distinction à faire en Machine Learning est la différence entre apprentissage supervisé et non supervisé. En anglais ces deux notions se nomment respectivement supervised learning et unsupervised learning.

Pour bien comprendre la différence, reprenons un exemple :

Supposons que nous avons une nouvelle base de photos à catégoriser. On dispose de données d'exemple (training set) préalables pour entraîner notre modèle.

En apprentissage supervisé : nous allons récupérer des données dites annotées de leurs sorties pour entraîner le modèle, c'est-à-dire que nous leur avons déjà associé un label ou une classe cible et nous voulons que l'algorithme devienne capable, une fois entraîné, de prédire cette cible sur de nouvelles données non annotées. Dans notre exemple, les données d'entrée seraient des images, et la cible (ou Target en anglais) la catégorie de photos que nous voulons.

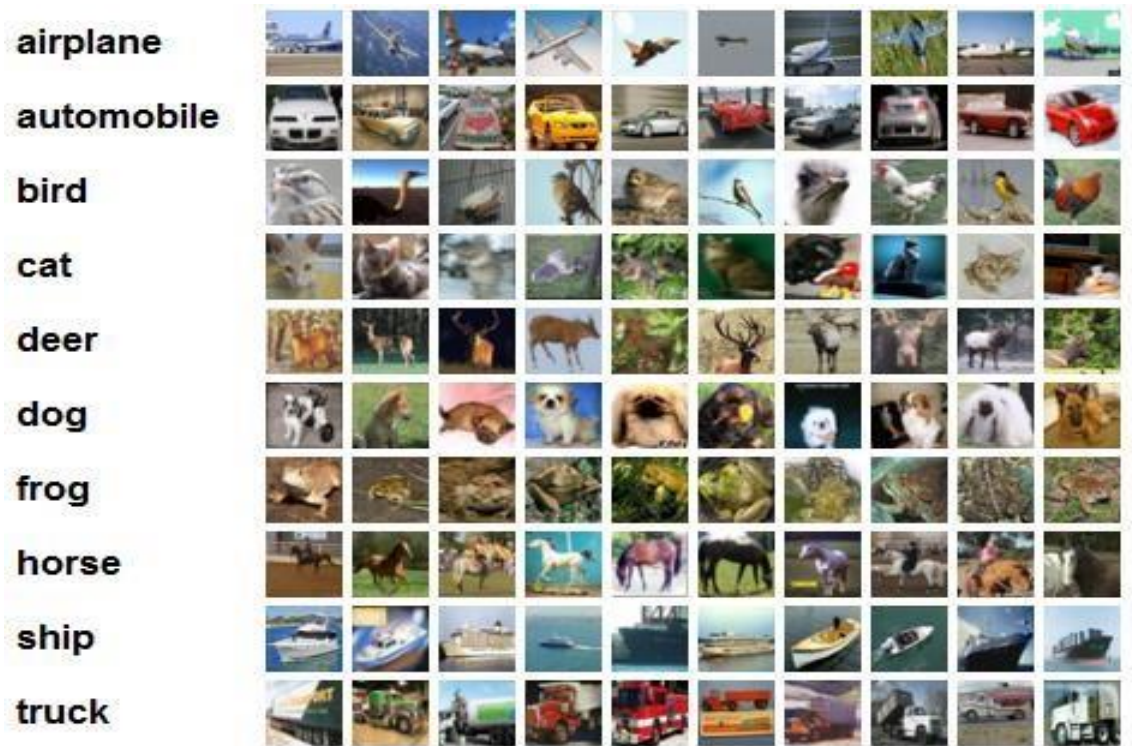


Figure 9: Donnée d'entrée en format image

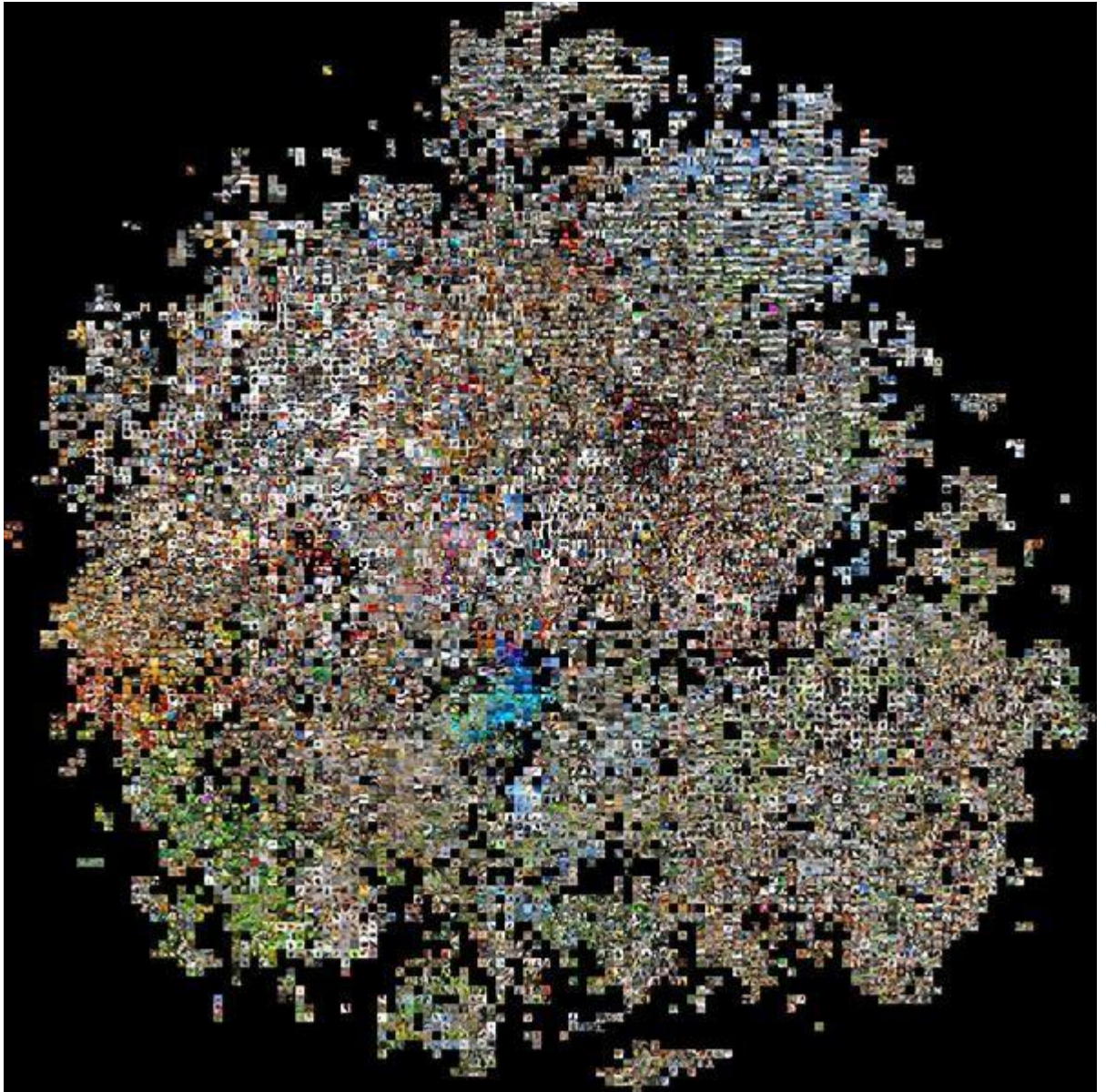
Chaque image utilisée pour entraîner l'algorithme est étiquetée avec sa catégorie.

En apprentissage non supervisé : les données d'entrées ne sont pas annotées. En effet, l'algorithme d'entraînement s'applique dans ce cas à trouver seul les similarités et distinctions au sein de ces données, et à regrouper ensemble celles qui partagent des caractéristiques communes.

Dans notre exemple, les photos similaires seraient ainsi regroupées automatiquement au sein d'une même catégorie.

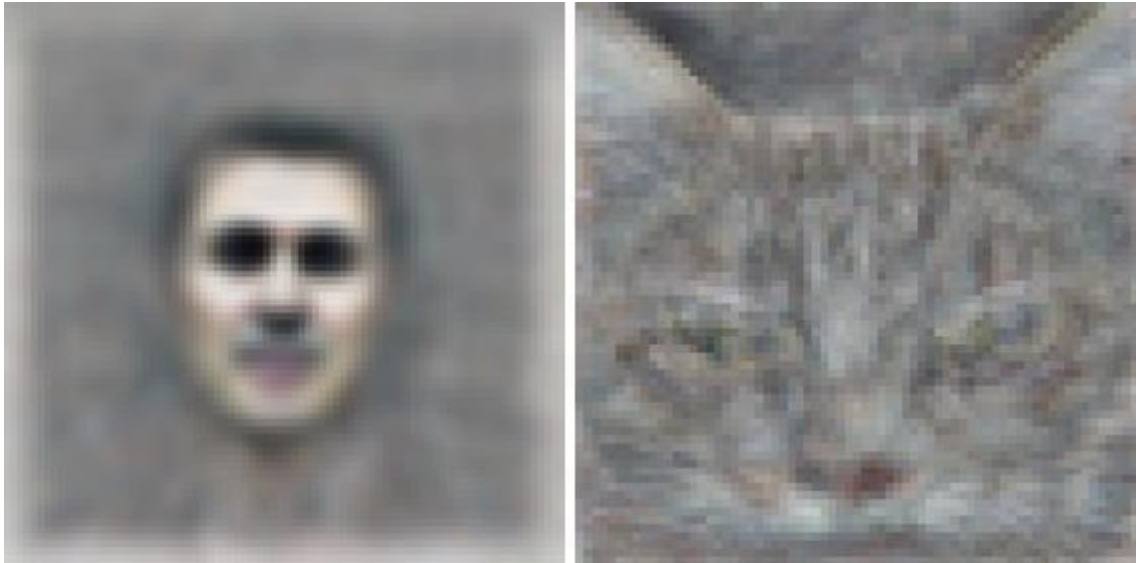
Voici une visualisation de l'utilisation d'une méthode non supervisée qui regroupe les images par similarité.





*Figure 10: Visualisation 2D d'une méthode non supervisée qui permet de grouper les images par similarité*





*Figure 11: La représentation interne des concepts de "visage" et "chat" apprises par un algorithme non supervisé à partir d'images extraites de millions de vidéos YouTube*

Une représentation un peu plus mathématique peut nous permettre d'éclaircir le concept :

En apprentissage non supervisé : nous recevons uniquement des observations brutes de variables aléatoires :  $x_1, x_2, x_3, x_4, \dots$  et nous espérons découvrir la relation avec des variables latentes structurelles :  $x_i \rightarrow y_i$

En apprentissage supervisé : nous recevons des données d'exemple annotées :  $(x_1, y_1), (x_2, y_2), (x_3, y_3), \dots$  et on espère prédire la sortie sur de nouvelles observations :  $x^* \rightarrow y^*$

- **Dans quel cas utilise-t-on l'un ou l'autre ?**

Dans le cas où nous avons un problème où nous pouvons annoter précisément pour chaque observation la cible que nous voulons en sortie, nous pouvons utiliser l'apprentissage supervisé.

Dans le cas où nous essayons de mieux comprendre notre base de données où d'identifier des comportements intéressants, nous pouvons utiliser l'apprentissage non supervisé.

## II.4 Régression ou Classification

Une autre distinction qui nous aidera dans le choix d'un algorithme de Machine Learning est le type de sortie que l'on attend de notre programme : est-ce une valeur continue (un nombre) ou bien une valeur discrète (une catégorie) ? Le premier cas est appelé une régression, le second une classification.

Par exemple, si nous voulons déterminer le coût par clic d'une publicité web, nous effectuons une régression. Si on veut déterminer si une photo est un chat ou un chien, nous effectuons une classification.

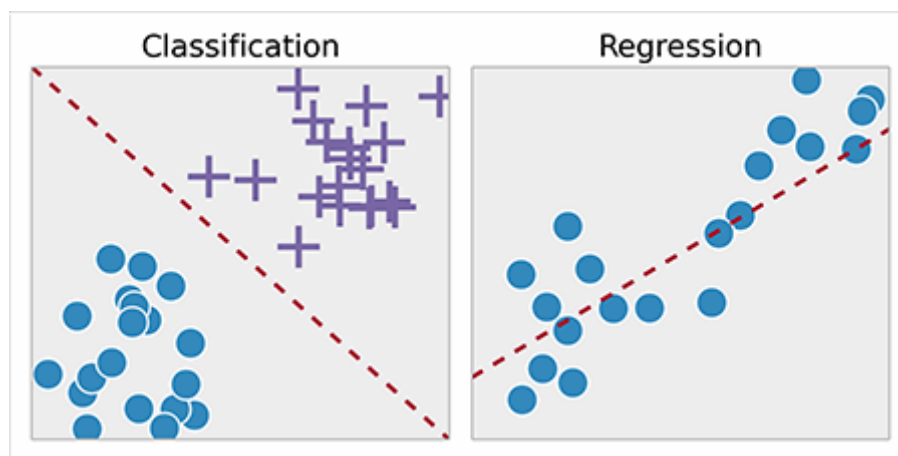


Figure 12: Illustration de la différence entre classification linéaire et régression linéaire

## II.5 L'algorithme d'apprentissage

L'algorithme d'apprentissage constitue la méthode avec laquelle le modèle statistique va se paramétrer à partir des données d'exemple. Il existe de nombreux algorithmes différents. Nous allons parler d'un type d'algorithme particulier en fonction du type de tâche que l'on souhaite accomplir et du type de données dont on dispose. En gros, quelle est l'entrée de l'algorithme et quelle est la sortie.

Quelques exemples d'algorithmes d'apprentissage de Machine Learning :

- La régression linéaire
- K-NN
- Les Support Vector Machine (SVM)
- Les réseaux de neurones
- Les random forests

## III. Réseaux de neurones artificiels et deep Learning

### III.1 Définition des réseaux de neurone :

Les réseaux de neurones artificiels tentent de reproduire la manière dont le cerveau traite l'information. Dans le cerveau, l'information est traitée par un réseau complexe de neurones.

Interconnectés. Les neurones des différentes régions du cerveau sont spécialisés dans des Traitements spécifiques.

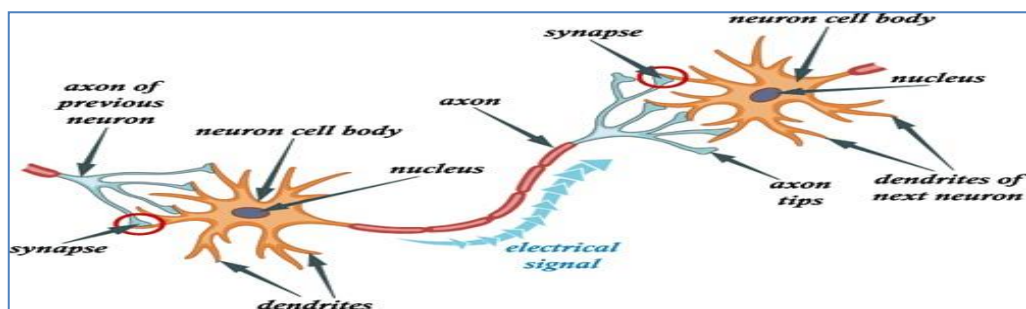


Figure 13: réseaux de neurone réel du corps humain

En effet, chaque neurone est représenté par une fonction prenant en entrée le signal des autres neurones, pondération spécifique à chacun d'entre eux, et effectue une transformation de la somme des signaux résultants, puis retourne le signal vers l'étage de Neurones suivant :

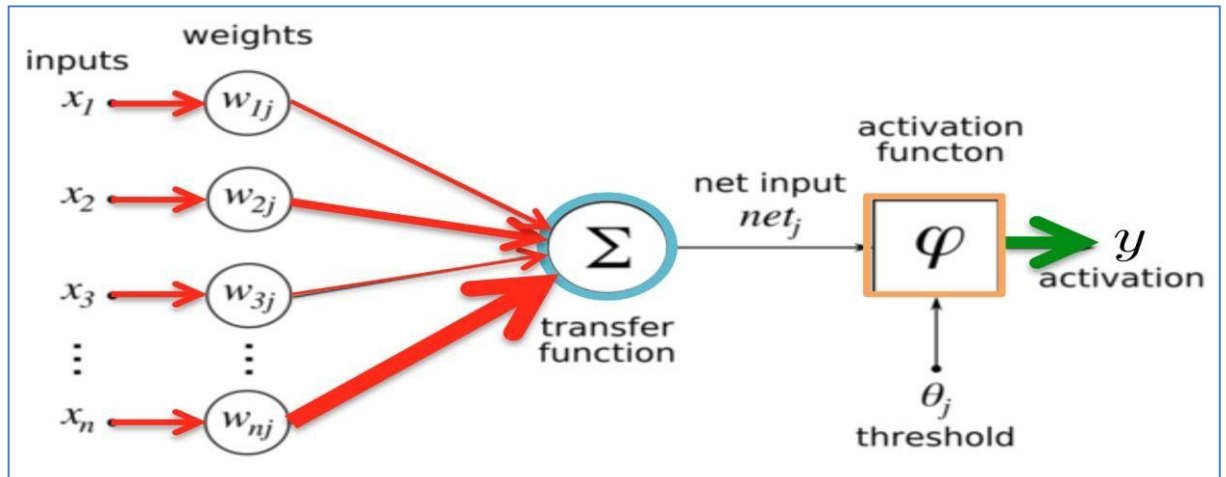


Figure 14: réseaux de neurone

Une approche totalement différente pour résoudre le même type de problème fait appel à ce que nous appelons des "réseaux de neurone profonds", c'est ce que nous appelons le "Deep Learning/ L'utilisation d'un neurone artificiel n'a rien de récent.

Il ne fonctionne jamais seul, mais dans le cadre d'un "réseau de neurone" dont le premier exemple historique fut le perceptron.

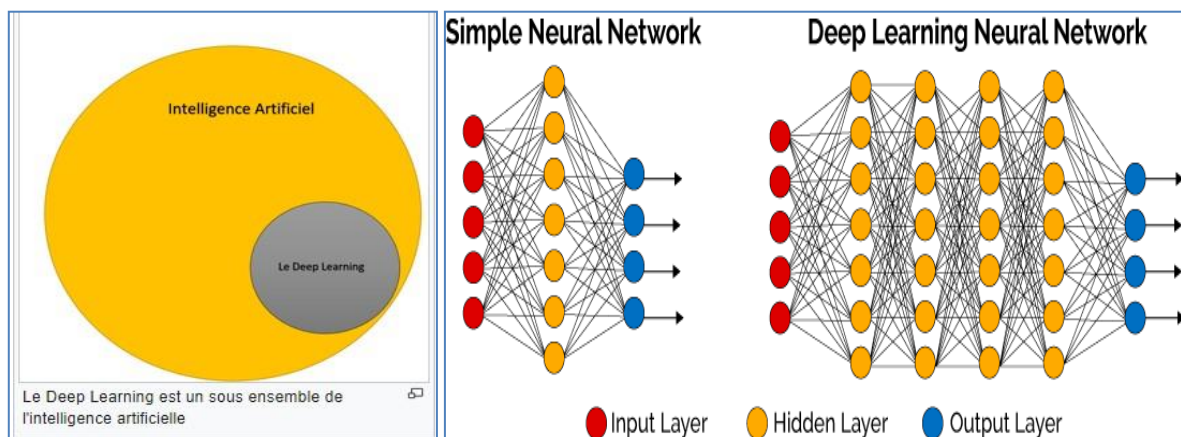


Figure 15: Réseaux de neurone profonds

## III.2 Deep Learning

Le Deep Learning, ou apprentissage profond, fait référence à un type d'Intelligence Artificielle particulier utilisant notamment le réseau du neurone et de certains modèles d'algorithmes particuliers comme le modèle de neurone convolutif

(convolutional neural network) afin de générer des modèles intelligents grâce à l'apprentissage.

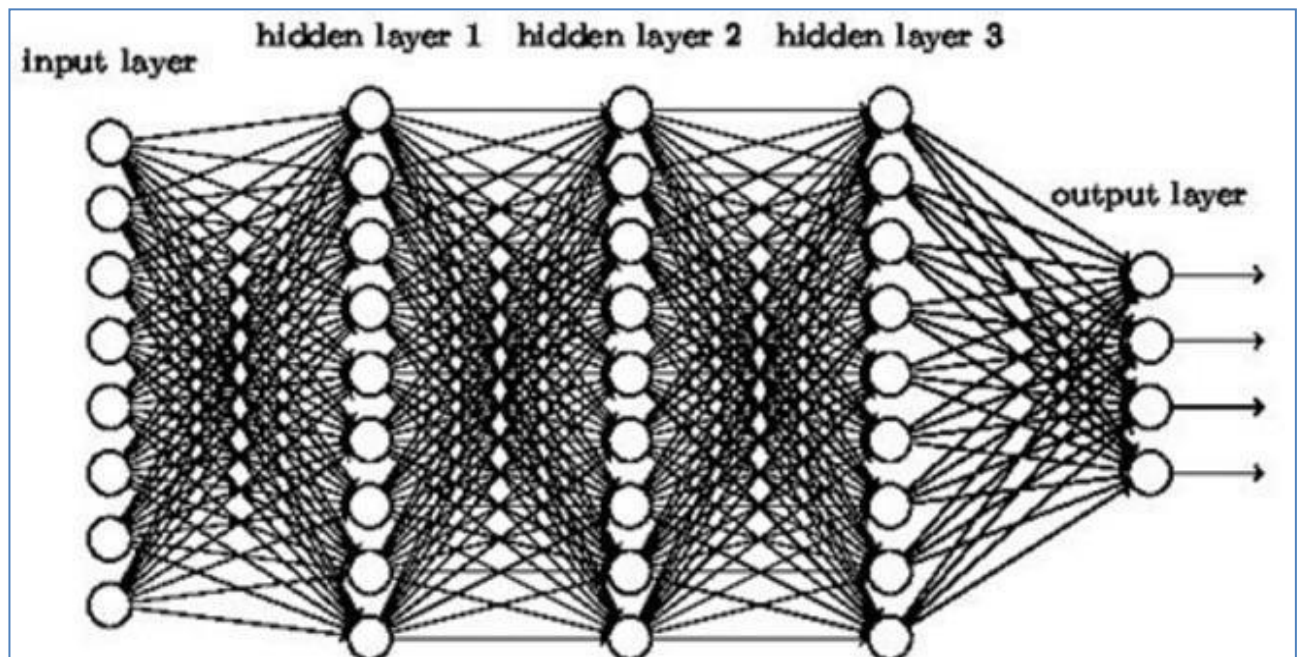


Figure 16: Modèle de neurone convolutif

## IV. Natural language processing (NLP)

### IV.1 Aperçu

En termes simples, L'objectif de la NLP est de faire en sorte que les machines comprennent nos langues parlées et écrites. De plus, la NLP est omniprésente et constitue déjà une part importante de la vie humaine. Les assistants virtuels, tels que Google Assistant, Cortana et Apple Siri, sont en grande partie des systèmes NLP.

De nombreuses tâches de la NLP ont lieu quand nous demandons à un VA (virtual assistant) : "Pouvez-vous me montrer un bon restaurant italien à proximité?". Tout d'abord, le VA doit convertir l'énoncé en texte (c'est-à-dire, parole à texte). Ensuite, il doit comprendre la sémantique de la demande (par exemple, l'utilisateur recherche un bon restaurant proposant une cuisine italienne) et formuler une demande

structurée (par exemple, cuisine = italienne, note = 3-5, distance <10 km ). Ensuite, le VA doit rechercher des restaurants filtrés par leur emplacement et cuisine, puis trie les restaurants en fonction des notes reçues. Pour calculer une note globale pour un restaurant, un bon système NLP peut examiner à la fois la note et la description fournie par chaque utilisateur. Enfin, une fois que l'utilisateur est au restaurant, le VA peut l'aider en traduisant divers éléments de menu de l'italien vers l'arabe. Cet exemple montre que la NLP est devenue une partie intégrante de la vie humaine.

## **IV.2 Difficulté**

La NLP est un domaine de recherche extrêmement difficile, car les mots et la sémantique ont une relation non linéaire extrêmement complexe, et il est encore plus difficile de saisir cette information sous forme de représentation numérique robuste. Pour aggraver les choses, chaque langue a sa grammaire, sa syntaxe et son vocabulaire. Par conséquent, le traitement de données textuelles implique diverses tâches complexes telles que l'analyse du texte (par exemple, la tokenisation et son enchaînement), l'analyse morphologique, la désambiguïsation du sens des mots et la compréhension de la structure grammaticale sous-jacente d'une langue. Par exemple, dans ces deux phrases, « je suis allé à la banque et j'ai longé la rive du fleuve », le mot banque a deux significations totalement différentes. Pour distinguer ou (sans ambiguïté) la banque de mots, nous devons comprendre le contexte dans lequel le mot est utilisé. L'apprentissage automatique est devenu un outil clé pour la NLP, aidant à accomplir les tâches mentionnées au moyen de machines.

## **IV.3 Tâches du traitement du langage naturel**

La NLP a une multitude d'applications dans le monde réel. Un bon système de NLP est celui qui effectue de nombreuses tâches de NLP. Lorsque vous recherchez la météo du jour sur Google ou utilisez Google Translate, découvrez comment dire "Comment allez-vous ?" en français, vous utilisez un sous-ensemble de ces tâches dans la NLP. Nous énumérerons certaines des tâches les plus omniprésentes :

Tokenisation : la tokenisation consiste à séparer un corpus de texte en unités atomiques (par exemple, des mots). Bien que cela puisse sembler trivial, la tokenisation

est une tâche importante. Par exemple, en japonais, les mots ne sont pas délimités par des espaces ni des signes de ponctuation.

Désambiguïsation du sens des mots (WSD, Word-sense Désambiguation): la tâche qui consiste à identifier le sens correct d'un mot. Par exemple, dans les phrases suivantes : Le chien a aboyé contre le facteur et l'écorce d'arbre est parfois utilisée comme médicament ; le mot écorce a deux significations différentes. WSD est essentiel pour des tâches telles que la réponse à une question.

Reconnaissance d'entités nommées (NER) : le NER tente d'extraire des entités (par exemple, une personne, un lieu et une organisation) d'un corps de texte donné ou d'un corpus de texte. Par exemple, la phrase, John a donné deux pommes à l'école lundi à l'école sera transformé en [John] nom a donné [nom à Mary] [deux] pommes numéro à l'organisation [de l'école] le [lundi.] Heure. Le TNS est un sujet impératif dans des domaines tels que la recherche d'informations et la représentation des connaissances.

Marquage Pos : le marquage Pos consiste à attribuer des mots à leurs parties respectives du discours. Il peut s'agir de balises de base telles que nom, verbe, adjectif, adverbe et préposition, ou granulaires telles que nom propre, nom commun, phrasal verb, verbe, etc.

Classification des phrases / synopsis : la classification des phrases ou des synopsis (critiques de films, par exemple) présente de nombreux cas d'utilisation tels que la détection du spam, la classification des articles d'actualité (politique, technologique et sportive, par exemple) et les évaluations de produits (c'est-à-dire positives ou négatif). Ceci est réalisé en formant un modèle de classification avec des données étiquetées (c'est-à-dire des revues annotées par des humains, avec une étiquette positive ou négative).

Génération linguistique : dans la génération linguistique, un modèle d'apprentissage exemple, réseau de neurones) est formé aux corpus de texte (une vaste collection de documents textuels), qui permettent de prédire le nouveau texte qui suit. Par exemple, la génération linguistique peut générer une histoire de science-fiction entièrement nouvelle en utilisant des histoires de science-fiction existantes pour la formation.

Réponses aux questions (QA): les techniques d'assurance qualité ont une grande valeur commerciale et se trouvent à la base des chat bots et des VA (par exemple, Google Assistant et Apple Siri). Les chats bots ont été adoptés par de nombreuses entreprises pour le support client.

Les chats bots peuvent être utilisés pour répondre et résoudre les problèmes simples des clients (par exemple, changer le forfait mobile mensuel d'un client), qui peuvent être résolus sans intervention humaine. L'QA touche à de nombreux autres aspects de la NLP, tels que la recherche d'informations et la représentation des connaissances. En conséquence, tout cela rend très difficile le développement d'un système d'assurance qualité.

Traduction automatique (MT): MT a pour tâche de transformer une phrase d'une langue source (par exemple, l'allemand) vers une langue cible (par exemple, l'anglais). C'est une tâche très ardue, car différentes langues ont des structures morphologiques très différentes, ce qui signifie qu'il ne s'agit pas d'une transformation individuelle. En outre, les relations mot à mot entre les langues peuvent être un à plusieurs, un à un, plusieurs à un ou plusieurs à plusieurs. C'est ce que l'on appelle le problème de l'alignement des mots dans la littérature MT.

Enfin, pour développer un système pouvant assister un humain dans les tâches quotidiennes (par exemple, un VA ou un chat bot), plusieurs de ces tâches doivent être effectuées ensemble. Comme nous l'avons vu dans l'exemple précédent où l'utilisateur demande : "Pouvez-vous me montrer un bon restaurant italien? proximité ", plusieurs tâches NLP différentes, telles que la conversion parole-texte, les analyses sémantiques et sentimentales, la réponse aux questions et la traduction automatique, doivent être complétées.

Dans la figure 18, nous présentons une taxonomie hiérarchique des différentes tâches NLP classées en plusieurs catégories. Nous avons deux grandes catégories : les tâches d'analyse (analyse du texte existant) et de génération (nouveau texte), puis divisées en trois catégories différentes : syntaxique (tâches basées sur la structure du langage), sémantique (tâches basées sur le sens). , et pragmatique (problèmes ouverts difficiles à résoudre):



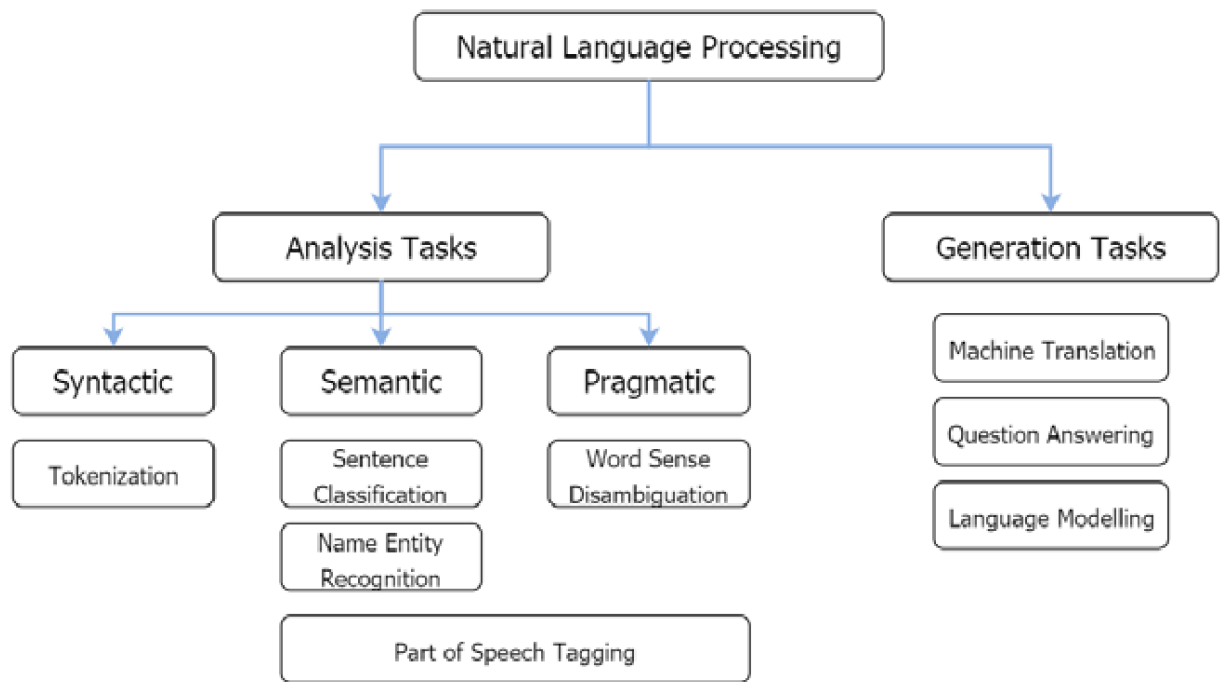


Figure 17: Hiérarchie des différentes tâches NLP

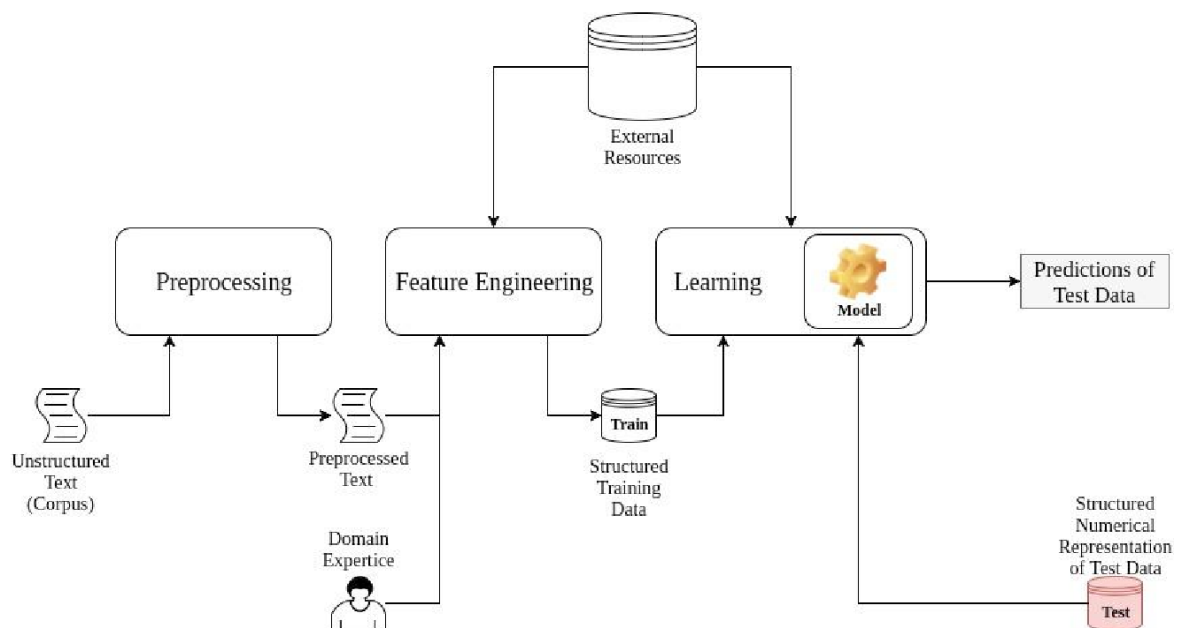


Figure 18: Le processus complet de l'approche traditionnelle

## **V. Outils technologiques utilisés :**

### **V.1 GitHub :**



*Figure 19: Logo de GitHub*

GitHub est un service web d'hébergement et de gestion de développement de logiciels, utilisant le logiciel de gestion de versions Git. GitHub propose des comptes gratuits pour les projets de logiciels libres. Le site assure également un contrôle d'accès et des fonctionnalités destinées à la collaboration comme le suivi des bugs, les demandes de fonctionnalités, la gestion de tâches et un wiki pour chaque projet.

En avril 2016, GitHub a annoncé avoir dépassé les 14 millions d'utilisateurs et plus de 35 millions de dépôts de projets le plaçant comme le plus grand hébergeur de code source au monde.

### **V.2 Python :**



Figure 20: Logo du Python

Python est un langage de programmation orienté objet, interprété et de haut niveau. Il est doté d'un typage dynamique fort, d'une gestion automatique de la mémoire par ramasse-miettes et d'un système de gestion d'exceptions.

Python a une philosophie de conception qui met l'accent sur la lisibilité du code, et une syntaxe qui permet aux programmeurs de coder en moins de lignes comparé à la plupart des autres langages. Il est créé par Guido van Rossum en 1998 et placé sous une licence libre et fonctionne sur la plupart des plates-formes informatiques, la version actuelle est 3.9 (le 5 octobre 2020), et contient une multitude de bibliothèques variées.

### V.3 Jupyter Notebook :



Figure 21: Logo du Jupyter Notebook

Jupyter Notebook est une application Web open source qui vous permet de créer et de partager des documents contenant du code en direct, des équations, des visualisations et même du texte narratif. Les utilisations comprennent : le nettoyage et la transformation des données, la simulation numérique, la modélisation statistique, la visualisation des données, l'apprentissage automatique et bien plus encore.

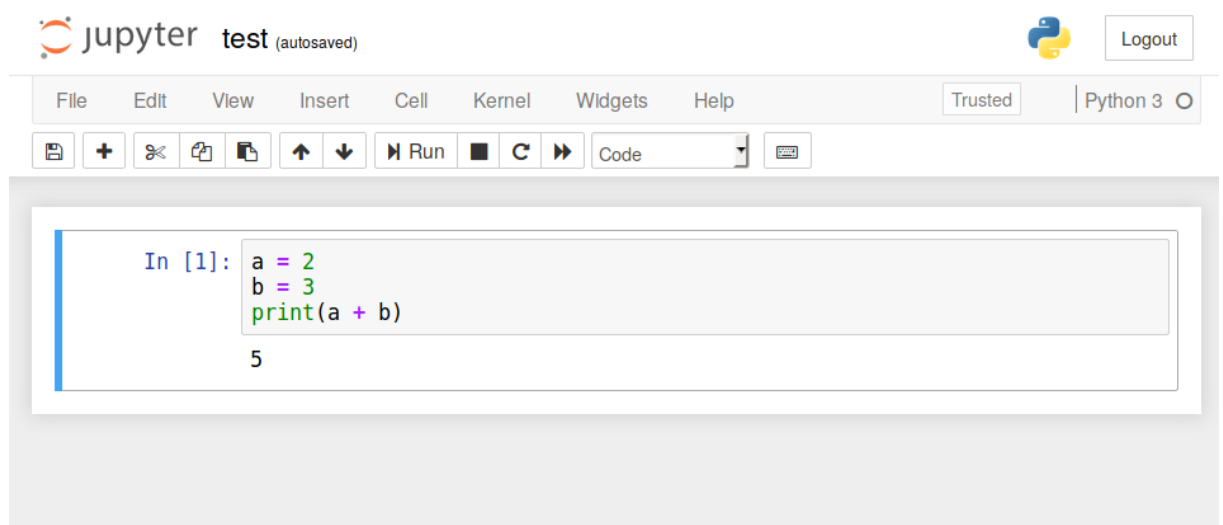


Figure 22: Exemple d'illustration de Python dans la plateforme Jupyter

## V.4 Scikit-learn :



Figure 23: Logo de la bibliothèque Scikit-Learn

Scikit-learn est une bibliothèque libre Python destinée à l'apprentissage automatique. Elle propose dans son Framework de nombreuses bibliothèques d'algorithmes à implémenter clé en main, à disposition des data scientists.

Elle comprend notamment des fonctions pour estimer tous les modèles du Maching Learning déjà mentionnés.

Son usage est facile, il suffit de savoir quelle algorithme choisir et préparer les données de telle façon que cette bibliothèque peut les utilisent pour entrainer le modèle.

# Chapitre 3 : Application

## I. Représentation générale

L'objectif de l'application est d'entraîner un modèle qui va prédire par la suite le domaine d'un CV à partir des informations extraites des plusieurs CV de différents profils.

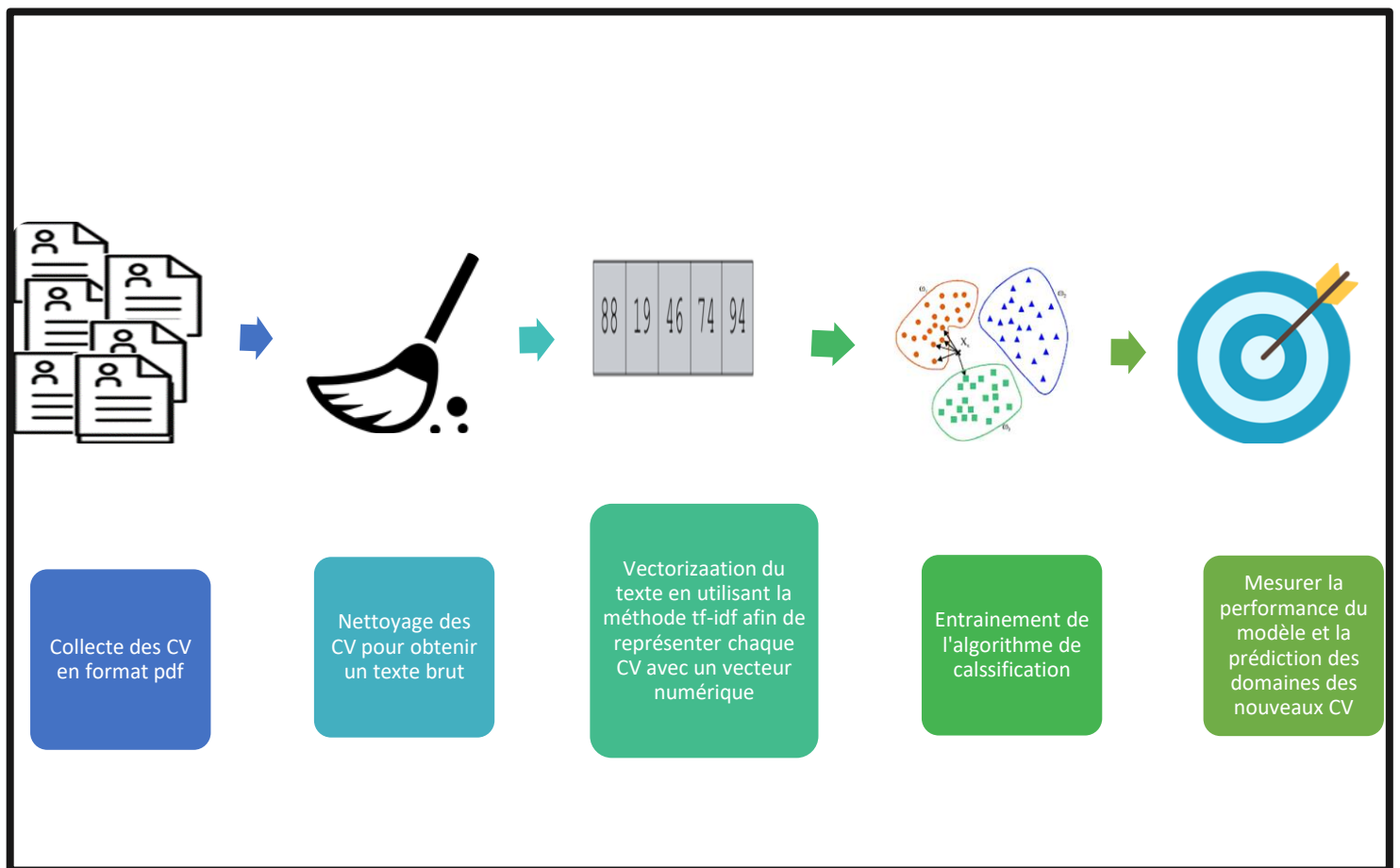


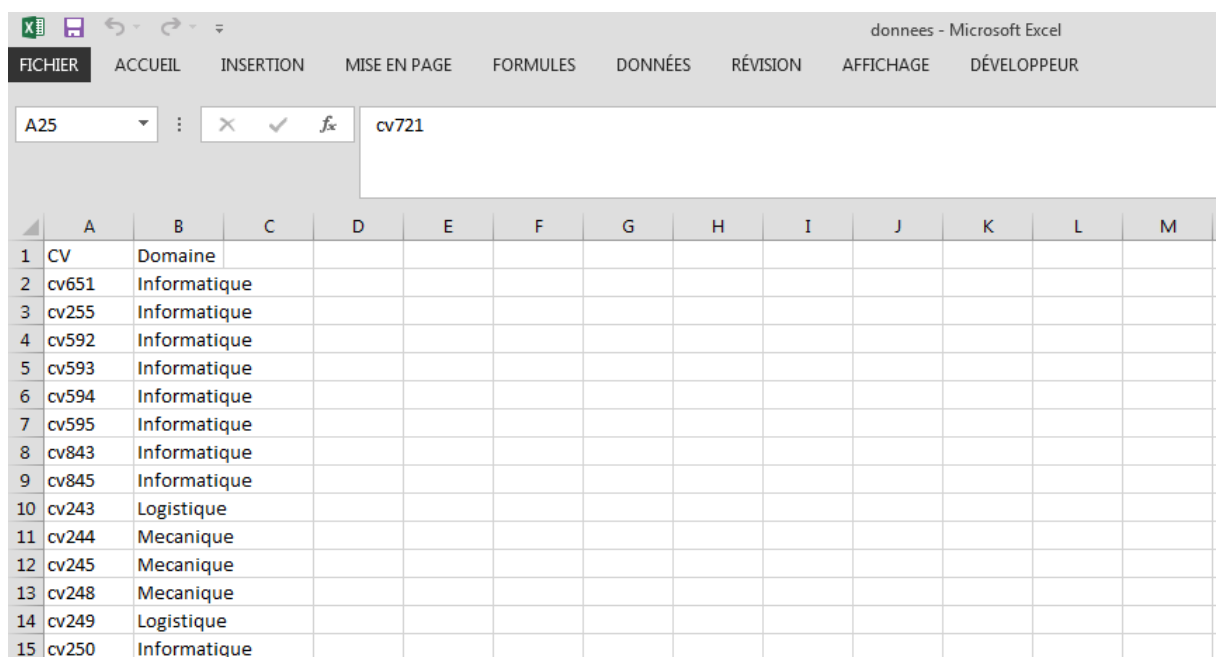
Figure 24: Représentation générale de l'application

## II. Collecte des données

La première phase consiste à collecter les données, à partir de plusieurs sites de web, nous avons pu collecter une base de données composée de 200 CV.

Ainsi un travail manuel nous a permis de classer les CV et à affecter à chacun d'eux un domaine parmi les 8 domaines suivants :

Informatique, Logistique, Mécanique, Economie, Commerce, Juridique, Ressources Humaines.



The screenshot shows a Microsoft Excel spreadsheet titled 'donnees - Microsoft Excel'. The ribbon includes 'FICHIER', 'ACCUEIL', 'INSERTION', 'MISE EN PAGE', 'FORMULES', 'DONNÉES', 'RÉVISION', 'AFFICHAGE', and 'DÉVELOPPEUR'. The formula bar shows 'cv721'. The spreadsheet contains a table with 15 rows and 13 columns (A-M). The first row (1) has headers 'CV' in column A and 'Domaine' in column B. The following rows contain data for various CV numbers and their corresponding domains.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	CV	Domaine											
2	cv651	Informatique											
3	cv255	Informatique											
4	cv592	Informatique											
5	cv593	Informatique											
6	cv594	Informatique											
7	cv595	Informatique											
8	cv843	Informatique											
9	cv845	Informatique											
10	cv243	Logistique											
11	cv244	Mécanique											
12	cv245	Mécanique											
13	cv248	Mécanique											
14	cv249	Logistique											
15	cv250	Informatique											

Figure 25: Base de données dans Excel

## III. Nettoyage des CV

### III.1 Préparation des données

Le prétraitement et le nettoyage de données sont des tâches primordiales qui doivent être effectuées avant d'utiliser un jeu de données à des fins d'apprentissage automatique. Les données brutes sont souvent bruyantes, peu fiables et incomplètes. Leur utilisation pour une fin de modélisation peut générer des résultats incorrects. Pour cela, nous avons construit un processus complet de nettoyage pour rendre les données sans bruit et prêtes à être analysées ce qui nous permet de consommer moins de capacité de calcul et avoir de bons résultats.

Ce processus est principalement composé des étapes suivantes :

- Suppression de tout morceau de texte inutile dans notre étude comme : e-mail, numéro de téléphone, les noms des candidats...
- Suppression des mots vides comme ; un, avec, dans ... etc.
- Suppression de la ponctuation
- La normalisation : C'est la conversion de toutes les disparités d'un mot dans leur forme normalisée (également appelée lemme).

### III.2 Nettoyage des cv :

Les CV sont un excellent exemple de données non structurées. Chaque CV a son propre style de formatage, possède ses propres blocs de données et présente de nombreuses formes de formatage des données. Cela rend la lecture des CV difficile, du point de vue de la programmation. Les recruteurs passent beaucoup de temps à parcourir les CV et à sélectionner ceux qui conviennent le mieux à leur emploi. Les géants de la technologie comme Google et Facebook reçoivent chaque jour des milliers de CV pour différents postes et les recruteurs ne peuvent pas les parcourir tous. C'est la raison pour laquelle les analyseurs de CV sont très utiles pour des personnes comme elles. Les analyseurs de CV permettent de sélectionner facilement le CV idéal parmi les nombreux CV reçus.

*Pour le nettoyage des cv, nous sommes basés sur une Data Set qui contient les éléments que on va supprimer des CV à savoir les noms et les prénoms, les parties standards formant le cv. Les parties qui ne sont pas nécessaires dans l'entraînement*

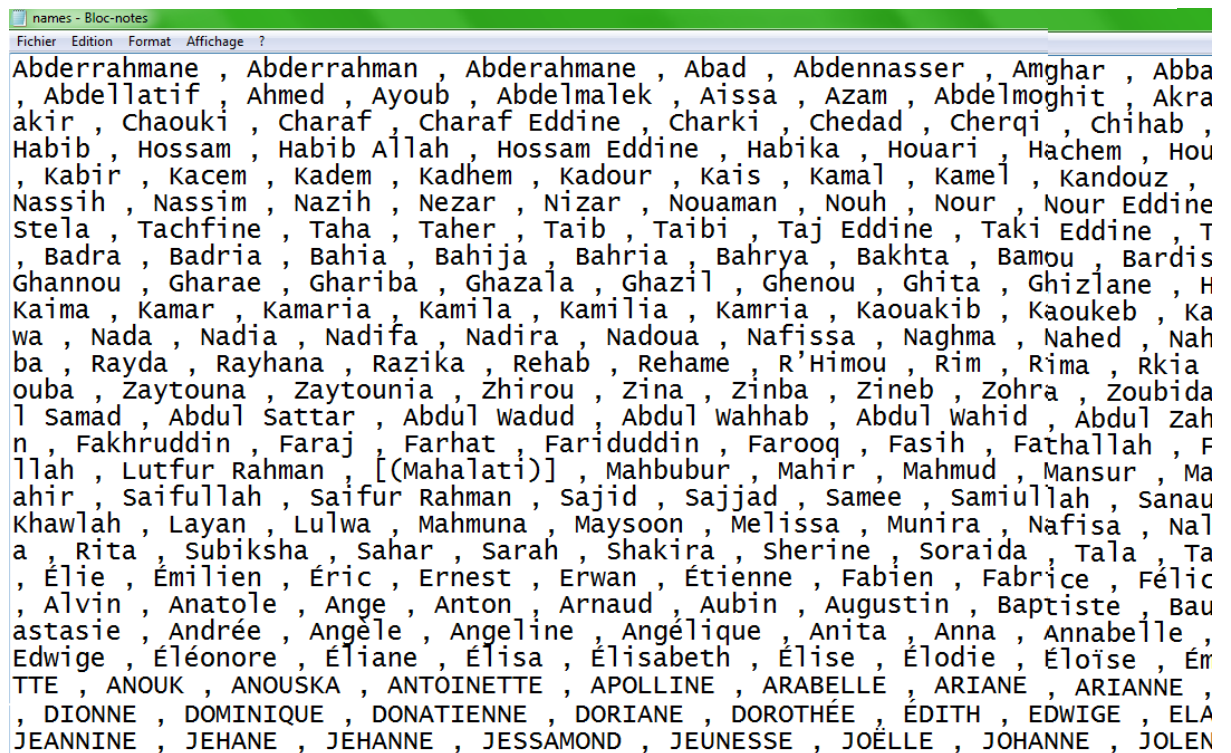


du modèle (Contact, Divers...), les entités des parties essentielles dans mon modèle, qui sont respectivement dans les Data Sets « names.txt », « unnecessary\_entities.txt » et « useful\_entities.txt ».

```
print(useful_entities)
```

['education', 'formations', 'formation', 'diplômes', 'diplomes', 'formations académiques', 'formations academiques', 'formation et diplômes', 'formation et diplomes', 'diplômes et formations', 'diplomes et formations', 'diplômes & formations', 'diplomes & formations', 'parcours de formation', 'études et diplômes', 'etudes et diplomes', 'logiciels', 'qualités', 'qualites', 'qualite s', 'informatique', 'outil informatique', 'expériences professionnelles', 'experiences professionnelles', 'expériences', 'experiences', 'expériences pro', 'experiences pro', 'projets & expériences professionnelles', 'projets & experiences professionnelles', 'expérience', 'experience', 'experience professionnelle', 'experience associative', 'experiences techniques', 'expérience', 'experience', 'compétences', 'competences', 'compétences technique', 'competences technique', 'compétences informatiques', 'competences informatiques', 'compétences linguistiques', 'competences linguistiques', 'compétences linguistiques & informatiques', 'competences linguistiques & informatiques', 'connaissances', 'compétences et connaissances', 'competences et connaissances', 'projets scolaires', 'projets', 'langues/ informatiques', 'langues', 'langues et informatique', 'langues et competences', 'informatiques', 'connaissances qualifications', 'experiences professionnelle', 'nom', 'e-mail', 'email', 'mail', 'mobile', 'telephon e']

Figure 26: Dataset « Useful\_entities.txt »



The screenshot shows a text editor window titled 'names - Bloc-notes'. The menu bar includes 'Fichier', 'Edition', 'Format', and 'Affichage ?'. The text content is a long list of names, including: Abderrahmane, Abderrahman, Abderahmane, Abad, Abdennasser, Amghar, Abba, Abdellatif, Ahmed, Ayoub, Abdelmalek, Aissa, Azam, Abdelmoghith, Akra akir, Chaouki, Charaf, Charaf Eddine, Charki, Chedad, Cherqi, Chihab, Habib, Hossam, Habib Allah, Hossam Eddine, Habika, Houari, Hachem, Hou, Kabir, Kacem, Kadem, Kadhém, Kadour, Kais, Kamal, Kamel, Kandouz, Nassih, Nassim, Nazih, Nezar, Nizar, Nouaman, Nouh, Nour, Nour Eddine Stela, Tachfine, Taha, Taher, Taib, Taibi, Taj Eddine, Taki Eddine, T, Badra, Badria, Bahia, Bahija, Bahria, Bahrya, Bakhta, Bamou, Bardis Ghannou, Gharae, Ghariba, Ghazala, Ghazil, Ghenou, Ghita, Ghizlane, H Kaima, Kamar, Kamaria, Kamila, Kamilia, Kamria, Kaouakib, Kaoukeb, Ka wa, Nada, Nadia, Nadifa, Nadira, Nadoua, Nafissa, Naghma, Nahed, Nah ba, Rayda, Rayhana, Razika, Rehab, Reham, R'Himou, Rim, Rima, Rkia ouba, Zaytouna, Zaytounia, Zhirou, Zina, Zinba, Zineb, Zohra, Zoubida l samad, Abdul Sattar, Abdul wadud, Abdul Wahhab, Abdul wahid, Abdul Zah n, Fakhruddin, Faraj, Farhat, Fariduddin, Farooq, Fasih, Fathallah, F llah, Lutfur Rahman, [(Mahalati)], Mahbubur, Mahir, Mahmud, Mansur, Ma ahir, Saifullah, Saifur Rahman, Sajid, Sajjad, Samee, Samiullah, Sanau Khawlah, Layan, Lulwa, Mahmuna, Maysoon, Melissa, Munira, Nafisa, Nal a, Rita, Subiksha, Sahar, Sarah, Shakira, Sherine, Soraida, Tala, Ta , Élie, Émilien, Éric, Ernest, Erwan, Étienne, Fabien, Fabrice, Félic , Alvin, Anatole, Ange, Anton, Arnaud, Aubin, Augustin, Baptiste, Bau astasie, Andrée, Angèle, Angeline, Angélique, Anita, Anna, Annabelle, Edwige, Éléonore, Éliane, Élisabeth, Élise, Élodie, Éloïse, Ém TTE, ANOUK, ANOUSKA, ANTOINETTE, APOLLINE, ARABELLE, ARIANE, ARIANNE , DIONNE, DOMINIQUE, DONATIENNE, DORIANE, DOROTHÉE, ÉDITH, EDWIGE, ELA JEANNINE, JEHANE, JEHANNE, JESSAMOND, JEUNESSE, JOËLLE, JOHANNE, JOLEN

Figure 27: Data Set « names.txt »

```
print(unnecessary_entities)
```

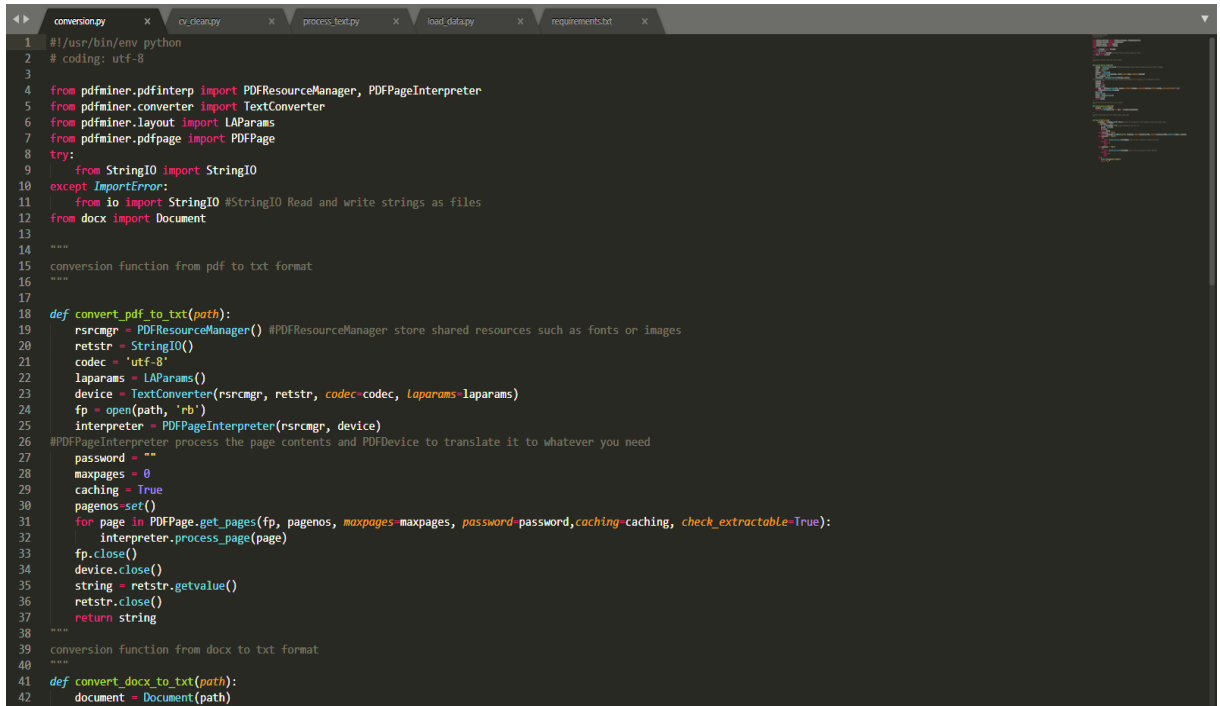
['\ufeeflangues', 'langues', 'date de naissance', 'langue', 'coordonnées', 'coordonnees', 'divers', "centres d'interet", 'info s', 'contact', 'loisirs', 'présentation', 'presentation', 'profil', 'hobbies', 'me contacter', 'objectifs', 'site web', 'person nalité', 'personnalite', 'intérêts', 'interets', 'réseaux sociaux', 'reseaux sociaux', 'une vie dynamique', 'objectif', 'inform ations complémentaires', 'personnalité', 'personnalite', 'engagements personnels', "centres d'intérêts", "centres d'interets", "centre d'intérêts", "centre d'interets"]

Figure 28: Dataset « unnecessary\_entities.txt »

Les étapes de nettoyages sont les suivantes :

**Text conversion** est la première et essentiel étape dans le processus de nettoyage de CV, consiste à transformer CV de son format original en texte

Code est le suivant :



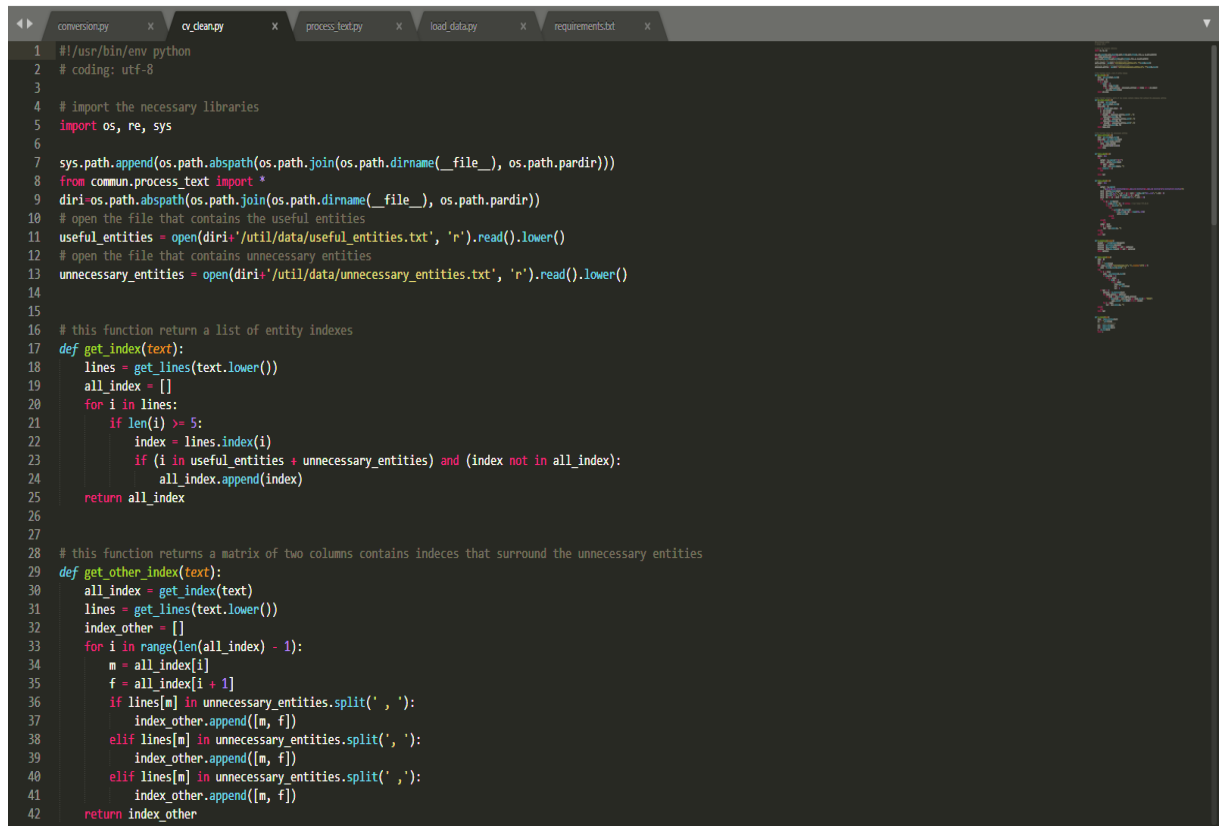
```

1  #!/usr/bin/env python
2  # coding: utf-8
3
4  from pdfminer.pdfinterp import PDFResourceManager, PDFPageInterpreter
5  from pdfminer.converter import TextConverter
6  from pdfminer.layout import LAParams
7  from pdfminer.pdfpage import PDFPage
8  try:
9      from StringIO import StringIO
10 except ImportError:
11     from io import StringIO #StringIO Read and write strings as files
12 from docx import Document
13
14 """
15 conversion function from pdf to txt format
16 """
17
18 def convert_pdf_to_txt(path):
19     rsrcmgr = PDFResourceManager() #PDFResourceManager store shared resources such as fonts or images
20     retstr = StringIO()
21     codec = 'utf-8'
22     laparams = LAParams()
23     device = TextConverter(rsrcmgr, retstr, codec=codec, laparams=laparams)
24     fp = open(path, 'rb')
25     interpreter = PDFPageInterpreter(rsrcmgr, device)
26     #PDFPageInterpreter process the page contents and PDFDevice to translate it to whatever you need
27     password = ""
28     maxpages = 0
29     caching = True
30     pagenos = set()
31     for page in PDFPage.get_pages(fp, pagenos, maxpages=maxpages, password=password, caching=caching, check_extractable=True):
32         interpreter.process_page(page)
33     fp.close()
34     device.close()
35     string = retstr.getvalue()
36     retstr.close()
37     return string
38 """
39
40 conversion function from docx to txt format
41 """
42 def convert_docx_to_txt(path):
43     document = Document(path)

```

Figure 29: Code de la conversion du texte

Et la deuxième cruciale étape est `cv_cleaning`, consiste à supprimer les données non nécessaires



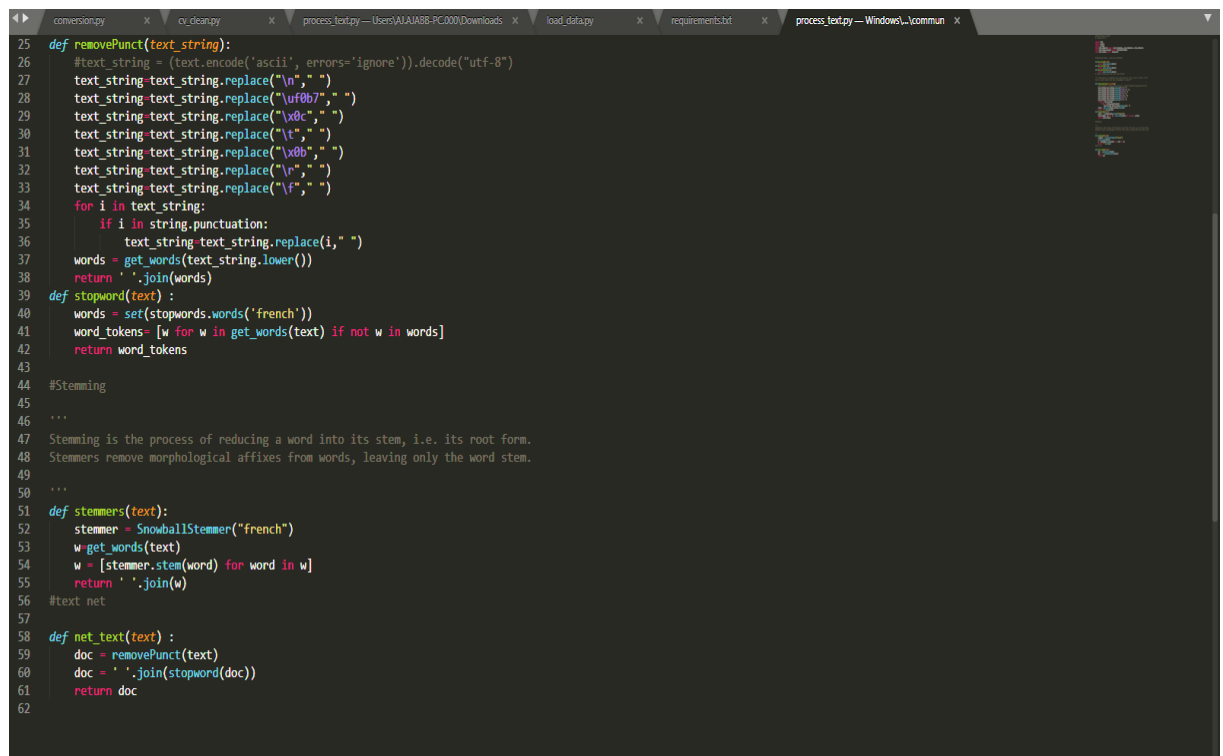
```

1  #!/usr/bin/env python
2  # coding: utf-8
3
4  # import the necessary libraries
5  import os, re, sys
6
7  sys.path.append(os.path.abspath(os.path.join(os.path.dirname(__file__), os.path.pardir)))
8  from comun.process_text import *
9  dir1=os.path.abspath(os.path.join(os.path.dirname(__file__), os.path.pardir))
10 # open the file that contains the useful entities
11 useful_entities = open(dir1+'util/data/useful_entities.txt', 'r').read().lower()
12 # open the file that contains unnecessary entities
13 unnecessary_entities = open(dir1+'util/data/unnecessary_entities.txt', 'r').read().lower()
14
15
16 # this function return a list of entity indexes
17 def get_index(text):
18     lines = get_lines(text.lower())
19     all_index = []
20     for i in lines:
21         if len(i) >= 5:
22             index = lines.index(i)
23             if (i in useful_entities + unnecessary_entities) and (index not in all_index):
24                 all_index.append(index)
25     return all_index
26
27
28 # this function returns a matrix of two columns contains indeces that surround the unnecessary entities
29 def get_other_index(text):
30     all_index = get_index(text)
31     lines = get_lines(text.lower())
32     index_other = []
33     for i in range(len(all_index) - 1):
34         m = all_index[i]
35         f = all_index[i + 1]
36         if lines[m] in unnecessary_entities.split(' '):
37             index_other.append([m, f])
38         elif lines[m] in unnecessary_entities.split(','):
39             index_other.append([m, f])
40         elif lines[m] in unnecessary_entities.split(';'):
41             index_other.append([m, f])
42     return index_other

```

Figure 30: *CV\_cleaning*

La troisième étape est `text_processing` que consiste à supprimer les ponctuations et `stop_words` :



```

25 def removePunct(text_string):
26     #text_string = (text_string.encode('ascii', errors='ignore')).decode("utf-8")
27     text_string = text_string.replace("n", " ")
28     text_string = text_string.replace("uf0b7", " ")
29     text_string = text_string.replace("x0c", " ")
30     text_string = text_string.replace("t", " ")
31     text_string = text_string.replace("x0b", " ")
32     text_string = text_string.replace("n", " ")
33     text_string = text_string.replace("f", " ")
34     for i in text_string:
35         if i in string.punctuation:
36             text_string = text_string.replace(i, " ")
37     words = get_words(text_string.lower())
38     return ' '.join(words)
39 def stopword(text):
40     words = set(stopwords.words('french'))
41     word_tokens = [w for w in get_words(text) if not w in words]
42     return word_tokens
43
44 #Stemming
45
46 '''
47 Stemming is the process of reducing a word into its stem, i.e. its root form.
48 Stemmers remove morphological affixes from words, leaving only the word stem.
49 '''
50
51 def stemmers(text):
52     stemmer = SnowballStemmer("french")
53     w = get_words(text)
54     w = [stemmer.stem(word) for word in w]
55     return ' '.join(w)
56
57 #text net
58
59 def net_text(text):
60     doc = removePunct(text)
61     doc = ' '.join(stopword(doc))
62     return doc

```

Figure 31: Code de suppression du ponctuations

## IV. Vectorisation

Après le nettoyage du CV et la suppression de toute information inutile, les modèles du Machine Learning ne peuvent pas traiter un texte directement, donc il fallait faire des transformations afin de rendre le texte du CV nettoyé compréhensible.

### IV.1 La transformation Sac de mots (Bag-Of-Words)

Une première étape pour transformer un texte en format numérique consiste à construire un dictionnaire de tous les mots et calculer l'occurrence de chaque mot cela nous permettra d'obtenir une première représentation numérique

Par exemple, considérons les phrases suivantes :

'Le coronavirus est un virus dangereux.',

'Ce virus a apparu à la fin de 2019 en Chine.',

'Le premier cas du Covid-19 au Maroc apparu en Mars.',

'Le vaccin du Covid-19 sera probablement prêt en 2021']

Le dictionnaire du vocabulaire pour ces phrases serait :

```
['19', '2019', '2021', 'apparu', 'au', 'cas', 'ce', 'chine', 'coronavirus', 'Covid', 'dangereux', 'de', 'du', 'en', 'est', 'fin', 'la', 'le', 'maroc', 'mars', 'premier', 'prêt', 'probablement', 'sera', 'un', 'vaccin', 'virus']
```

Ensuite, nous obtenons un vecteur de taille V (taille du vocabulaire) pour chaque phrase, en indiquant l'occurrence de chaque mot, c'est-à-dire le nombre de fois où chaque mot du vocabulaire apparaît dans la phrase. Dans cet exemple, les vecteurs caractéristiques des phrases seraient respectivement les suivants :

```
[[0 0 0 0 0 0 0 0 1 0 1 0 0 0 1 0 0 1 0 0 0 0 0 0 1 0 1]
 [0 1 0 1 0 0 1 1 0 0 0 1 0 1 0 1 1 0 0 0 0 0 0 0 0 0 1]
 [1 0 0 1 1 1 0 0 0 1 0 0 1 1 0 0 0 1 1 1 1 0 0 0 0 0 0]
 [1 0 1 0 0 0 0 0 0 1 0 0 1 1 0 0 0 1 0 0 0 1 1 1 0 1 0]]
```

```
In [1]: import sklearn
        from sklearn.feature_extraction.text import CountVectorizer

In [2]: text = ['Le virus Covid-19 est un virus dangereux.', \
... 'Ce virus a apparu à la fin de 2019 en Chine.', \
... 'Le premier cas du Covid-19 au Maroc apparu en Mars.', \
... 'Le vaccin du Covid-19 sera probablement prêt en 2021']

In [3]: vectorizer = CountVectorizer()
        X = vectorizer.fit_transform(text)
        print(vectorizer.get_feature_names())

['19', '2019', '2021', 'apparu', 'au', 'cas', 'ce', 'chine', 'coronavirus', 'covid', 'dangereux', 'de', 'du', 'en', 'est', 'fin', 'la', 'le', 'maroc', 'mars', 'premier', 'pret', 'probablement', 'sera', 'un', 'vaccin', 'virus']

In [4]: print(X.toarray())

[[0 0 0 0 0 0 0 0 1 0 1 0 0 0 1 0 0 1 0 0 0 0 0 0 1 0 1]
 [0 1 0 1 0 0 1 1 0 0 0 1 0 1 0 1 1 0 0 0 0 0 0 0 0 0 1]
 [1 0 0 1 1 1 0 0 0 1 0 0 1 1 0 0 0 1 1 1 1 0 0 0 0 0 0]
 [1 0 1 0 0 0 0 0 0 1 0 0 1 1 0 0 0 1 0 0 0 1 1 1 0 1 0]]
```

Figure 32: Illustration de l'exemple du Sac de mots

Parfois cette transformation est limitée car il ne prend pas en considération le contexte de chaque mot dans la phrase, donc comme alternative on peut opter pour une transformation avancée qui consiste à la décomposition en n-grammes au niveau des mots, par exemple pour la dernière phrase, la décomposition en bi-grammes donne les différentes combinaisons de 2 mots, le dictionnaire du vocabulaire sera le suivant :

```
['19 au', '19 est', '19 sera', '2019 en', 'apparu en', 'apparu la',
 'au maroc', 'cas du', 'ce virus', 'covid 19', 'de 2019', 'du covid',
 'en 2021', 'en chine', 'en mars', 'est un', 'fin de', 'la fin', 'le premier',
 'le vaccin', 'le virus', 'maroc appar', 'premier cas', 'pret en',
 'probablement pret', 'sera probablement', 'un virus', 'vaccin du', 'virus
 irus apparu', 'virus covid', 'virus dangereux']
```

## Bi-gram vectorizer

```
In [8]: vectorizer2 = CountVectorizer(analyzer='word', ngram_range=(2, 2))
X = vectorizer2.fit_transform(text)
print(vectorizer2.get_feature_names())

['19 au', '19 est', '19 sera', '2019 en', 'apparu en', 'apparu la', 'au maroc', 'cas du', 'ce virus', 'covid 19', 'de 2019', 'du covid', 'en 2021', 'en chine', 'en mars', 'est un', 'fin de', 'la fin', 'le premier', 'le vaccin', 'le virus', 'maroc appar', 'premier cas', 'pret en', 'probablement pret', 'sera probablement', 'un virus', 'vaccin du', 'virus apparu', 'virus covid', 'virus dangereux']

In [7]: print(X.toarray())

[[1 0 0 0 0 0 0 0 1 1 0 0 0 1 0 0 1 0 0 0 0 0 0 1 0 2]
 [0 1 0 1 0 0 1 1 0 0 1 0 1 0 1 1 0 0 0 0 0 0 0 0 0 1]
 [1 0 0 1 1 1 0 0 1 0 0 1 1 0 0 0 1 1 1 1 0 0 0 0 0 0]
 [1 0 1 0 0 0 0 0 1 0 0 1 1 0 0 0 1 0 0 0 1 1 1 0 1 0]]
```

Figure 33: Sac de mots en 2-grammes

Comme pour les deux exemples illustratifs, nous optons pour cette vectorisation pour chaque CV :

```
# Builds a dictionary of features and transforms documents to feature vectors and convert our text documents to a
# matrix of token counts (CountVectorizer)
count_vect = CountVectorizer()
X_train_counts = count_vect.fit_transform(data['Texte_CV'])
```

Et nous obtenons un dictionnaire de mots contenant 7740 mots, et une représentation vectorielle de chaque CV contenant les fréquences de chaque mot du dictionnaire.

## IV.2 Term-frequency inverser document frequency : TF-IDF

La formule TF-IDF (ce type de pondération avait déjà été systématisé par le chercheur en sciences informatiques Hans Peter Luhn en 1957) est utilisée souvent pour l'optimisation On-Page des sites Web, afin d'augmenter la pertinence d'un site pour les moteurs de recherche déterminé dans quelles proportions certains mots peuvent être évalués par rapport au reste du texte. Cette formule utilise le critère de fréquence ou occurrence.

L'objectif de la transformation TF-IDF, et non pas seulement l'occurrence est de donner des poids pour chaque mot afin de diminuer l'impact des mots qui sont fréquents mais sans utilité, et qui impact l'effet des mots qui sont moins fréquents et pertinents.

La formule de calcul est la suivante :

$$tf\ idf(t, d) = tf(t, d) * idf(t)$$

Avec :  $t$  : un terme donné

Et  $d$  : un document donné

Et  $tf(t, d)$  : la fréquence d'un terme  $t$  dans un document  $d$

Et 
$$idf(t, d) = \log\left(\frac{n}{df(t)}\right) + 1$$

Avec  $n$  : le nombre total des documents

Et

$df(t)$  : la fréquence du document, le nombre de document ou apparait le terme  $t$

Il faut noter qu’il existe plusieurs formules pour calculer le TF-IDF.

Ainsi, nous appliquons cette normalisation a notre la représentation précédente des CV et nous obtenons

```
# transform a count matrix to a normalized tf-idf representation (tf-idf transformer)
tfidf_transformer = TfidfTransformer()
X_train_tfidf = tfidf_transformer.fit_transform(X_train_counts)
```

Figure 34: Application de la transformation Tf-Idf

## V. Choix de l’algorithme : KNN

### V.1 Présentation et principe de l’algorithme

Comme c’est déjà énoncé, l’algorithme K-NN (K-Nearest Neighbors) est une méthode d’apprentissage supervisé, il est l’algorithme de classification le plus simple et il peut être utilisé pour la classification. Son fonctionnement peut être assimilé à l’analogie suivante “dis-moi qui sont tes voisins, je te dirais qui tu es...”.

Donc c’est une technique pour classer les nouveaux points de données en fonction de la relation avec les points de données proches, en utilisant une distance par exemple la distance euclidienne, distance de Minkowski ou même la distance de Manhattan.

Son principe est le suivant :



Comme la montre la figure ci-dessous, le diagramme de dispersion nous permet de calculer la distance entre deux points de données quelconques. Les points de données sur le diagramme de dispersion ont déjà été classés en trois groupes. Ensuite, un nouveau point de données dont la classe est inconnue est ajouté au tracé. Nous pouvons prédire la catégorie du nouveau point de données en fonction de sa relation avec les points de données existants. Cependant, nous devons d'abord définir «k» pour déterminer le nombre de points de données que nous souhaitons nommer pour classer le nouveau point de données. Si nous fixons k à 5, k-NN analysera uniquement la relation entre le nouveau point de données et les trois points de données les plus proches.

Et pour cet exemple l'algorithme prédira la classe numéro 1 pour le nouveau point x vu que quatre de ces cinq voisins appartiennent à la classe 1.

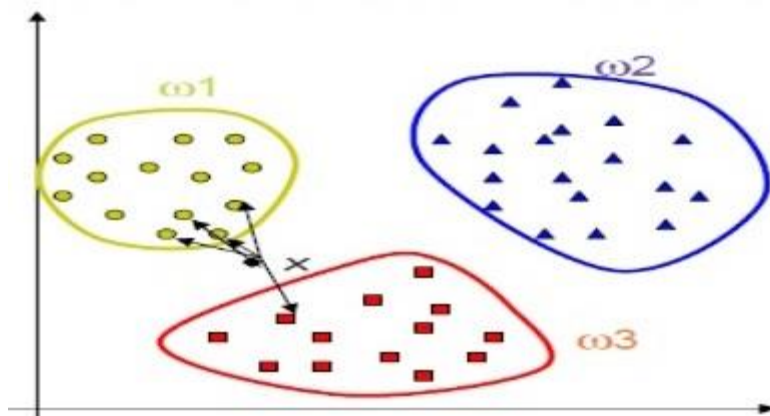


Figure 35: Représentation graphique de l'algorithme K-NN

## V.2 Exemple d'application pour un CV

En prenant un CV qui ne figure pas dans la base de données, par exemple un CV de profil spécialité Mécanique :



**Jad SAAB**

INGÉNIEUR GÉNÉRALISTE - SPÉCIALITÉ CONCEPTION MÉCANIQUE

✉ jadsaab28@gmail.com  
 ☎ 06 30 17 05 72  
 📍 16 avenue de Grammont  
 37000 Tours  
 📅 22 ans  
 📄 Permis B  
 🇱🇧 Libanais  
 🚗 Mobilité

#### Langues

Arabe

Français

Anglais

#### Voyages

Royaume Uni, Belgique, Pays-Bas,  
 Chypre, Dubaï, Qatar, Bahreïn.

#### Centres d'intérêt

Cinema

Voyages culturels

Sports: Football, Basketball

#### Expériences professionnelles

07-2018 - 09-2018

##### Stagiaire

Khatib & Alami (K&A) Beirut, Liban

- Conception destinations de pompage d'eau : modélisation, choix des pompes respectant un cahier des charges stricte
- Dimensionnement de centres d'épuration : respect des différentes étapes du processus
- Analyse des systèmes de filtration des déchets : sélection des dégrilleurs, dimensionnement des conteneurs pour la coagulation et la sédimentation, familiarisation au procédures chimiques de purification.

07-2017 - 08-2017

##### Stagiaire

Middle East Airlines (MEA) Beirut, Liban

- Contrôles techniques : annuels, non destructifs (CND)
- Maintenance : Procédures de réparation des bosses, changement des pneus, correction des dysfonctionnements éventuels sur le moniteur électronique centralisé des avions (ECAM) ou de l'avionique.

#### Diplômes et Formations

2016 - 2019

ÉCOLE D'INGÉNIEUR - « Mécanique et Conception des Systèmes » - Option mécanique avancée  
 Ecole polytechnique de l'université de Tours (Polytech Tours) Tours, France

2014 - 2016

LICENCE 2 - « Sciences Mécaniques et Ingénierie (SMI) »  
 Université de sciences et technologies de Lille (Lille 1) Lille, France

2013 - 2014

BACCALAURÉAT scientifique français - Spécialité Mathématique  
 Lycée franco-libanais Verdun Beirut, Liban  
 Mention « Assez Bien »

#### Informatique

Catia   
 Abaqus   
 AutoCad   
 Pack Office   
 Python   
 Java 

Figure 36: Exemple d'un CV

Le résultat prédit parmi les huit domaines (Informatique, Logistique, Mécanique, Economie, Commerce, Juridique, Ressources Humaines) est Mécanique, comme c'est illustré ci-dessous :

### Exemple de prédiction pour un CV donné

```
docs_new=[cv_cleaner(convert_pdf_to_txt('cv298.pdf'))]
X_new_counts = count_vect.transform(docs_new)
X_new_tfidf = tfidf_transformer.transform(X_new_counts)
predicted = clf.predict(X_new_tfidf)
print("Le domaine de ce CV est:",Categories_numbers[predicted[0]])

'Mecanique'
```

Figure 37: Exemple de prédiction du domaine d'un CV

## VI. L'approche Train-Test et mesure de l'accuracy

Notre modèle essaie donc de s'entraîner sur des données et va par la suite prédire le domaine des nouvelles données, une question qui se pose alors : Comment se généralisera-t-il sur des données qu'elle n'a pas encore "vu" lors de la phase d'apprentissage ?

C'est pour cela que les données d'apprentissage doivent être assez représentatives du problème étudié. Plus la base de données d'entraînement est représentatif du problème, mieux seront les informations capturées par le modèle prédictif.

C'est pour cette raison qu'on a opté pour une division de la base de données en une base pour l'entraînement et une autre pour le test (composée de 20% de la taille de la base globale c'est à dire 40 CV),

Afin de mesurer la performance du modèle, cette dernière sera mesurée d'une façon intuitive par la relation suivante :

$$\text{Accuracy} = \frac{\text{Nombre de CV prédit correctement dans la base de données de Test}}{\text{Nombre totale de CV dans la base de données de Test}}$$

En prédictant les domaines pour les CV de la base de données Test, nous retrouvons le résultat suivant :

86.6% des CV ont été correctement classés par notre modèle, cette qualité de prédiction est acceptable.

```
# Predicting our test data
predicted = clf.predict(X_test_tfidf)
print('Le modèle a pu prédire correctement', np.mean(predicted == Y_test)*100, '% parmi ceux de la base de données de Test.')
Le modèle a pu prédire correctement 86.66666666666667 % parmi ceux de la base de données de Test.
```

Figure 38: Mesure de la performance du modèle

## VII. Choix du k : nombre de voisin optimal

Un point spécifique pour l'algorithme KNN, c'est que son accuracy est influencé par le paramètre K : le nombre de voisin.

Dans la figure ci-dessous illustre cette importance, le point x sera classé dans la classe A si k=3, par contre il sera classé dans la classe B si k=7.

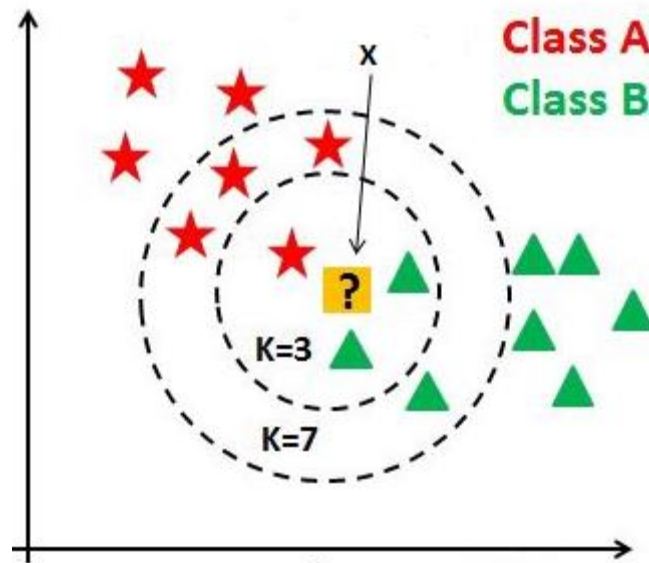


Figure 39: Figure illustrant l'importance du choix du paramètre  $K$

Lorsque nous diminuons la valeur de  $K$  à 1, nos prévisions deviennent moins stables.

Une valeur très grande de  $K$  ne donnera pas des résultats logiques car il donnera toujours la classe la plus dominante comme résultat.

Pour sélectionner le  $K$  qui convient à nos données, nous pouvons exécuter plusieurs fois l'algorithme KNN avec différentes valeurs de  $K$  et choisir le  $K$  qui réduit le nombre d'erreurs rencontrées (qui maximise le nombre de prédiction correcte).

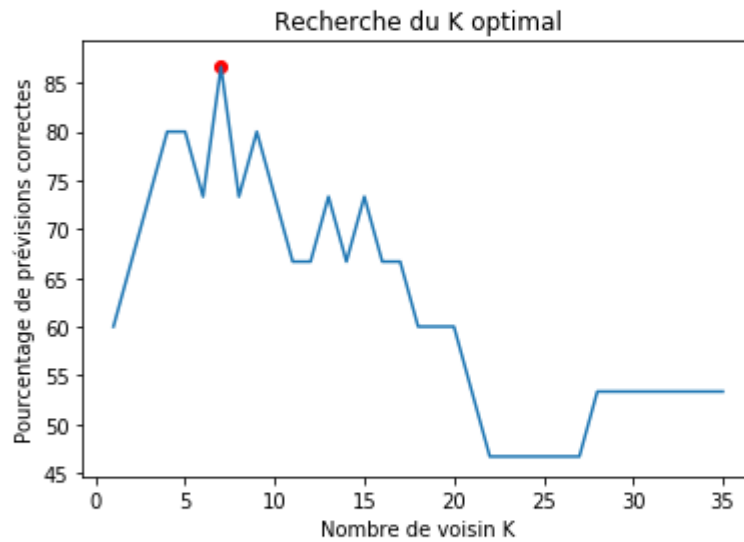
La figure ci-dessous est une représentation graphique pour les différentes valeurs de  $K$  ainsi que la valeur qui donne le meilleur résultat :

$K=7$ , qui correspondant à une performance de 86,66%.

```

K=np.arange(1,K5)
plt.plot(K,accuracy)
plt.xlabel('Nombre de voisin K')
plt.ylabel('Pourcentage de prévisions correctes')
plt.title("Recherche du K optimal")
plt.scatter(accuracy.argmax()+1,accuracy.max(),color='red')
plt.show()

```



```

print(accuracy.max())
print('La valeur optimale de K est :',accuracy.argmax()+1)

```

```

86.66666666666667
La valeur optimale de K est : 7

```

Figure 40: Représentation graphique de la performance du modèle en fonction du paramètre  $K$

# Conclusion

La découverte d'un nouveau domaine qui est la Data Science à travers le traitement automatique de la langue naturelle mais aussi du milieu de la recherche scientifique était un vrai défi pour la réussite de ce projet de ce stage d'application.

Nous avons appris plusieurs technologies notamment JSON, python et certaines de ses librairies etc... Ce qui est, sur le plan personnel, une vraie valeur ajoutée pour la suite de notre carrière professionnelle.

Nous avons utilisé des techniques de traitement automatique de la langue naturelle pour extraire automatiquement les métiers, les compétences et les expériences, afin de filtrer les CV sur lesquels nous avons travaillé.

Notre plus grande difficulté était dans la manipulation et la nature de données ; nous estimons aussi qu'avec plus de données propres et variées, nous aurions de meilleurs résultats.

# *Bibliography/Webography*

Text Classification of Patents using the k-Nearest Neighbor Algorithm (Catelyn School and John W. Sheppard, Gianforte School of Computing, Montana State University)

Perkins J., « Python text processing with NLTK 2.0 cookbook », Packt Publishing, 2010.

Weiss S., Indurkha N., Zhang T., Damerau F., « Text Mining – Predictive methods for analyzing unstructured information », Springer, 2005.

Feldman R., Sanger J., « The text mining handbook – Advanced approaches in analyzing unstructured data », Cambridge University Press, 2008.

Aggarwal C., Zhai C., « Mining Text Data », Springer, 2012.

Perkins J., « Python text processing with NLTK 2.0 cookbook », Packt Publishing, 2010.

Coelho L.P., Richert W., « Building Machine Learning Systems With Python », 2nd Edition, Packt Publishing, 2015.

<https://jupyter.org/>

<https://scikit-learn.org/>

<https://www.lebigdata.fr/python-langage-definitif/>



