REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEURE ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE IBN KHALDOUN - TIARET

MEMOIRE

Présenté à :

FACULTÉ MATHEMATIQUES ET INFORMATIQUE
DÉPARTEMENT D'INFORMATIQUE

Pour l'obtention du diplôme de :

MASTER

Spécialité :génie logiciel

Par :

CHEHBELAINE Walaa

Sur le thème

Analyse de l'état mental des utilisateurs de Twitter

Soutenu publiquement le octobre ...../2020 à Tiaret devant le jury composé de :

MrCHADLIAbdelhafid	Grade	Université Ibn-Khaldoun Tiaret	Président
Mr BENAOUDA Habib	Grade	Université Ibn-Khaldoun Tiaret	Examinateur
Mme LAKHDARI Aicha	Grade	Université Ibn-Khaldoun Tiaret	Encadreur

Dédicace

A mon père et ma mère pour leur dévouement à mon égard

A mon grand-père que j'aime beaucoup

A ma grand-mère que j'adore

A mes frères,

A toute la famille,

- *Chehebelaine*

Enfin: A tous ceux que j'ai oublié, qu'ils m'en excusent

Walaa

Remerciement

Tout d'abord je remercie le bon dieu le tout puissant et miséricordieux, qui m'a donné la force et la patience d'accomplir ce modeste travail.

Je tiens à exprimer ma profonde gratitude à Mme ***LAKHDARI Aicha****, pour avoir encadré et dirigémon présent travail. Je la remercie pour la confiance qu'elle a bien voulu m'accorder, ses conseils et remarques constructives qui m'ont permis d'améliorer la qualité de mon projet de fin d'études.*

Qu'elle soit ici assurée de mon très grand respect et de ma profonde reconnaissance.

Je tiens à exprimer ma gratitude et mes remerciements les plus sincères aux jurys pour leur travail et l'attention consacrée à l'égard de mon travail.

Je remercie ceux qui m'ont aidé et témoigner leur sympathie.

Résumé

De nos jours, notre style de vie conduit à la dépression, même chez la jeune génération. Comme des études indiquent que le taux de dépression augmente de jour en jour. Les rapports de l'OMS (organisation mondiale de la santé) et d'autres organisations montrent que les conséquences sont pires, voire conduisent au suicide. La majorité des travaux sur la détection des publications liées à la dépression sont basés sur les contributions recueillies auprès du grand public des medias sociaux à l'aide de questionnaires. En tant qu'une solution à cette problématique, nous abordons le problème de l'analyse des sentiments sur un ensemble de tweets préalablement tirés du big-social data "Twitter"en passant par une ressource lexicale linguistique (dictionnaire donné ou SentiWordnet) pour la phase de prétraitement et en utilisant un certain nombre de modèles d'apprentissage automatique traditionnel et en profondeur pour la classification. Il s'agit donc de la classification automatique de tweets en dépressifs et non-dépressifs. Le Deep Learning, approche incontournable en Machine Learning, s'avère particulièrement utile et performant pour cette classification.

Mots clés : Dépression, Tweet, SentiWordnet, Apprentissage Automatique traditionnel, DeepLearning, classification.

Table des matières

Liste des figures

Liste des tableaux

Liste des abréviations

AUC: Area Under The Curve

BOW: Bag of Words

CNN: Convolutional Neural Networks

DT: Decision Tree

FN: False Negative

FP: False Positive

K-NN: K nearest Neighbor

LSTM: Long Short-Term Memory

NB: Naïve Bayes

NLP: Natural Language Processing

NLTK: Natural Language Toolkit

NN: Neural Network

RF: Random Forest

RNN: Recurrent Neural Network

ROC: Receiver Operating Characteristic

SVM: Support Victor Machine

TN: True Negative

TP: True Positive

INTRODUCTION GÉNÉRALE

Introduction générale

La maladie mentale est très répandue dans le monde et la dépression est l'un des problèmes psychologiques les plus courants. La dépression non traitée augmente le risque de comportements dangereux. Plus de 300 millions de personnes dans le monde souffrent de dépression, mais la probabilité pour un individu de rencontrer un cas dépressif sur une période d'un an est de 3% à 5% pour les hommes et de 8 à 10% pour les femmes [1].Le trouble dépressif s'est également présenté comme la principale cause d'incapacité dans le monde chez les personnes de cinq ans et plus, et a été corrélé à un risque plus élevé de fractures chez les femmes. Plus des deux tiers des 30 000 suicides signalés l'année dernière étaient dus à la dépression. Dégradant ainsi considérablement la qualité de vie d'un individu et de son voisinage, le trouble dépressif devient donc un problème hautement prioritaire pour les sociétés à résoudre[1].

Les internautes utilisent de plus en plus les plateformes de médias sociaux pour partager leurs pensées et leurs désirs les plus intimes et exprimer leurs opinions. Les publications sur ces sites sont faites de manière naturaliste, et fournissent donc une solution à la manipulation que rencontrent souvent les questionnaires auto-déclarés sur un état de santé quelconque comme l'état dépressif. Les médias sociaux fournissent un moyen de capturer l'état mentald'un internaute et sont même efficaces pour représenter les sentiments d'inutilité, de culpabilité, d'impuissance et les niveaux de haine de soi qui caractériseraient souvent la dépression. L'hypothèse selon laquelle les médias sociaux, grâce à la vectorisation des mots, peuvent être utilisés pour construire des modèles statistiques afin de détecter et même de prédire les troubles dépressifs et non dépressifs, et peut-être même de compléter et d'étendre les approches traditionnelles orienté questionnaire sur les social-medias.

Dans ce projet de fin d'études (PFE), nous conduisons une étude pilote pour découvrir l'efficacité que ce soit des modèles traditionnels de machine-learning (ML) ou du Deep-learning en ce qui concerne l'analyse des sentiments appliquée à des tweets liés à la dépression et préalablement collectés de Twitter. Pour cela, nous avons utilisé une approche basée sur un lexiqueenrichiepar deux ressources linguistiques (dictionnaire donné &SentiWordnet) pour la phase de prétraitement des tweets. Cette phase consiste à identifierles mots présentsdansle dictionnaire demotspré-étiquetés(chacunayantreçuunevaleursurune échellecomposéede 03 intensitésdesentiments (polarités): -1,0,+1)et à comptabiliserlessentimentschiffrésafind'arriver à unscorepolairetotalpourchaquetexte de Tweet.

Les modèles traditionnels à entrainer et à tester font référence à des techniques d'apprentissage automatique classiques (classificateur d'apprentissage supervisé), telles que le classificateur Naïve Bayes (NB), le classificateur d'arbres de décision (DT), les machines à vecteurs de support (SVM), le classificateur k-plus proche-voisins (KNN) et le classificateur de forêt aléatoire (RF).Pour l'apprentissage en profondeur, nous utilisons le modèle de réseau neuronal récurrent(RNN).

Cependant, notre objectif est d'étudier, expérimenter et comparer ces différents classificateursen tant qu'une contribution à la problématique de l'analyse desproblèmes de santé mentale sur le big social data"Twitter".

Afin de parvenir à notre but, le présent document est divisé en deux volets. Un volet théorique ainsi qu'un volet pratique. Le volet théorique (chapitre 1) explore l'essence du textmining, l'apprentissage automatique (algorithmes traditionnels et deep learning), les méthodes de classifications disponibles ainsi que les algorithmes fréquemment utilisés. Le second volet, chapitre 2 & chapitre 3, débute par la conception de notre outil d'analyse de l'état mental des utilisateurs web et explore les méthodes et les critères d'évaluations qui ont été utilisés lors de cette étude. Ce volet se termine ainsi par une présentation détaillée des résultats.

ANALYSE DES SENTIMENTS ET LE DEEP LEARNING

CHAPITRE I : ANALYSE DES SENTIMENTS ET LE DEEP LEARNING

I.1 Introduction

Récemment, l'analyse des sentiments a reçu beaucoup d'attention non seulement de la part de la recherche scientifique mais aussi par d'autres domaines commela publicité, le marketing,.... Cela peut être attribué aux récentes avancées dans les réseaux sociaux et à la rapidité du relais de l'information. Les grandes masses de données réelles issues des réseaux sociaux sont largement utilisées pour, justement, l'analyse des sentiments.

Sentiment analysis est l'analyse des sentiments à partir de sources textuelles dématérialisées sur de grandes quantités de données (big-social data). Ce procédé apparait au début des années 2000 et connait un succès grandissant dû à l'abondance de données provenant de réseaux sociaux, notamment celles fournies par Twitter.Les sentiments extraits peuvent ensuite faire l'objet de statistiques sur le ressenti général d'un individu ou d'une communauté[2].

I.2 Analyse des sentiments

Un domaine utilisant les techniques de IR (*Recherche d'Informations*), TC (*Catégorisation de Texte*), ML (*Apprentissage Automatique*) ou Fouille de Texte est notamment le domaine de *l'Analyse des Sentiments*, connu sur le nom de (ang : *Opinion Mining*). Les recherches dans ce domaine couvrent plusieurs sujets, notamment l'apprentissage de l'orientation sémantique des mots ou des expressions, l'analyse sentimentale de documents et l'analyse des opinions et attitudes à l'égard de certains sujets ou produits. Cependant, L'analyse de sentiments est l'étude computationnelle et sémantique des parties de textes en fonction des opinions, des sentiments et des émotions exprimés dans le texte [3].

Généralement l'expression « *analyse des sentiments* » est utilisée pour désigner la tâche de classification automatique des unités de texte en fonction de leur polarité (positive, négative, neutre).

I.3 Catégorisation des sentiments

Les phrases sont objectives ou subjectives. Lorsqu'une phrase est objective, aucune autre tâche fondamentale n'est requise. Lorsqu'une phrase est subjective, ses polarités (positive, négative ou neutre) doivent être estimées (figure 1).

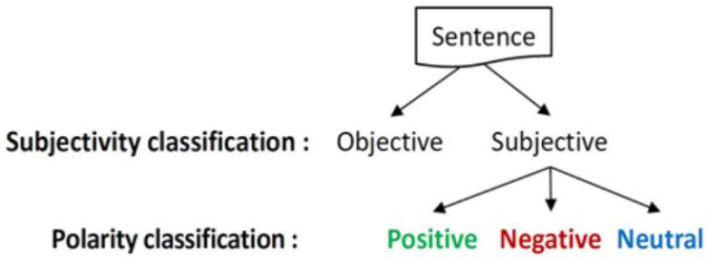


Figure 1 Workflow de l'analyse des sentiments[3]

La classification de subjectivité (Subjectivity classification) est la tâche qui distingue les phrases exprimant des informations objectives, des phrases exprimant des vues et opinions subjectives (figure 2).

L'iPhone est un smartphone – phrase objective

L'iPhone est génial – phrase subjective

Figure 2 Un exemple représentant la distinction entre la subjectivité et l'objectivité[3]

La classification de polarité et intensité de sentiment(Polarity classification) est la tâche qui distingue les phrases qui expriment des polarités positives, négatives ou neutres.[3]

L'intensité décrit à quel point la polarité d'une opinion est forte. Par exemple, dans une opinion à polarité positive, *aimer* est plus intense qu'*apprécier*, ou encore, dans une opinion de polarité négative, *haïr* est plus intense que de ne pas *aimer*[4].

I.4 Types d'analyse des sentiments

Il existe de nombreux types d'analyses de sentiments allant des systèmes qui se concentrent sur la classification de la polarité (*positif, négatif, neutre*) aux systèmes qui détectent des émotions (en *colère, heureux, triste*, etc.) ou identifient des intentions (par exemple, *intéressé, pas intéressé*). Nous abordons donc les types les plus importants [3]:

I.4.1 Analyse fine des sentiments

Au lieu de parler de phrases positives, négatives ou neutres, nous considérons les catégories suivantes :

- *Très positive*
- *Positive*
- *Neutre*
- *Négative*

- *Très négative*

Certains systèmes offrent également différentes classifications de polarité en identifiant si le sentiment positif ou négatif est associé à un sentiment particulier, tel que la *colère*, la *tristesse* ou des inquiétudes (*sentimentsnégatifs*) ou du *bonheur*, de l'*amour* ou de l'*enthousiasme* (*sentiments positifs*).

I.4.2 Détection d'émotion

La détection des émotions vise à détecter des émotions telles que le bonheur, la frustration, la colère, la tristesse, etc. De nombreux systèmes de détection d'émotions sont basés sur l'utilisation de lexiques de sentiments (c'est-à-dire des listes des émotions) ou sur des algorithmes d'apprentissage automatique complexes.

I.4.3 Analyse de sentiments à base d'aspects

Au lieu de classer le sentiment général d'un texte en positif ou en négatif, l'analyse de sentiments à base d'aspects permet d'analyser le texte afin d'identifier différents aspects et de déterminer le sentiment correspondant pour chacun. Les résultats sont plus détaillés, intéressants et précis car l'analyse à base d'aspects examine de manière précise les informations contenues dans un texte.

I.5 Les défis d'analyse des sentiments

Comme défis[5]:

- ✓ *L'état émotionnel du locuteur :* Les sentiments du locuteur peuvent être compatibles ou contraires aux déclarations faites par le locuteur.
- ✓ *Succès ou échec d'un côté avec respect à un autre :* Exemple 'Yay! France Beat Germany 3–1': si on a un supporteur de la France émotion positif, si on a un supporteur de l'Allemagne émotion négative.
- ✓ *Déclaration neutre des informations validées :* Si on a aucune indication sur l'émotion mais on a des citations valides, donc on ne sait pas et ce que on a des déclarations neutre ou une déclaration négatif d'émotion.
- ✓ *Sarcasme :* Est de déclarer des sentiments positifs, même s'ils sont négatifs. Exemple : « croyez-vous vraiment ce que vous dites ? »
- ✓ *Différent sentiment vers différent cible d'opinion :* L'orateur peut exprimer l'opinion à propos de cibles multiples, et le sentiment envers les différentes cibles pourrait être différent.

- ✓ *Déterminer précisément la cible de l'opinion :* Parfois, il est difficile de préciser la cible d'opinion.Par exemple, sur un sujet relatif aux conditions de travail ou aux prestations du restaurant, c'est le salarié concerné qui sera interrogé.

I.6 Difficultés d'Analyse des Sentiments

Parmi les difficultés d'analyse [5]:

- ✓ *Ambiguïté de certains mots positifs ou négatifs* selon les contextes et qui ne peut pas être toujours levée.
- ✓ *Difficulté due aux structures syntaxiques et sémantiques d'une phrase et l'expression de l'opinion:* Par exemple " l'histoire du film est intéressante mais les acteurs étaient mauvais ". Dans ce cas la polarité de la deuxième partie est opposée à la première.
- ✓ *Difficulté due au contexte :* la nécessite d'une bonne analyse syntaxique du texte, analyse qui peut se révéler particulièrement difficile dans des cas de coordination entre plusieurs parties d'une phrase. Par exemple "ma tonte a bien préparé le gâteau, son décor est bonne mais je n'ai pas aimée le goût", l'opinion de la dernière partie de la phrase est la plus importante.
- ✓ *Difficulté due à l'analyse de la phrase par " paquets de mots ":* Les deux phrases suivantes contiennent les mêmes paquets de mots sans pour autant exprimer les mêmes sentiments. La première phrase contient un sentiment positif alors que la deuxième est négative : " Je l'ai apprécié pas seulement à cause de ...", " Je l'ai pas apprécié seulement à cause de ... " ou se présente la gestion de négation.
- ✓ *Difficulté due au langage qu'utilisent les internautes pour s'exprimer*: Les ponctuations ne sont pas forcément utilisées pour marquer les fins de phrases, des mots spécifiques sont utilisés tel que : «ha haha», «Goood», «super».
- ✓ *Difficulté de déterminer un lexique adapté à l'analyse de l'ensemble des textes d'opinion*.

I.7 Domaines d'application d'analyse des sentiments

L'importance de l'analyse des sentiments est présentée dans plusieurs domaines, ainsi que plusieurs applications ont vu le jour dans ce contexte. Nous mentionnonsbrièvementquelques applications ci-dessous (figure 3)[6]:

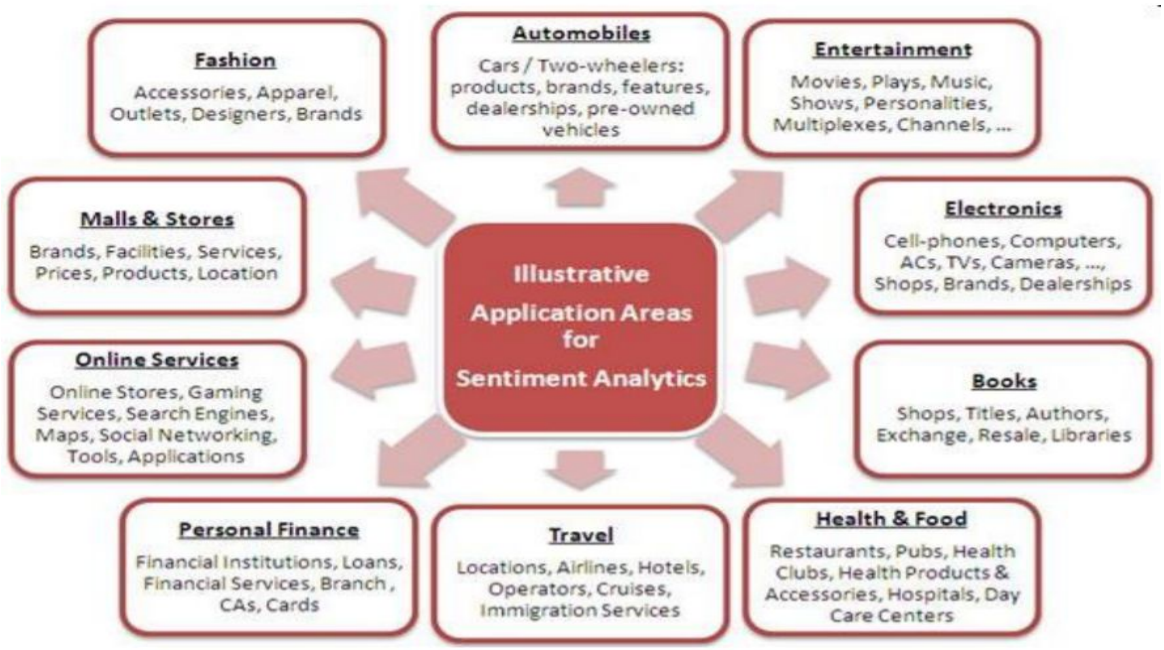


Figure 3 Domaines d'application d'analyse des sentiments[2]

- ✓ *Politique* **:** Grace à l'analyse de sentiments et via généralement des sondages, les décideurs de politique prennent en considération les avis et opinions de leurs citoyens sur certaines politiques. Cette solution politique va les aider à améliorer ou à créer de nouvelles politiques qui conviennent à leurs sociétés.
- ✓ *Prise de décision :* L'opinion et l'expérience des gens sont un élément très utile dans le processus de prise de décision.
- ✓ *Les systèmes de recommandations* **:** À travers l'analyse des sentiments, on peut classer les opinions des individus en positive ou négative, le système définit qui devrait prendre la recommandation et qui ne devrait pas prendre la recommandation.
- ✓ *Domaine de Transport:* Pour assembler et analyser les opinions de public sur le statut de transport. Exemple : Système de transport intelligent moderne.
- ✓ *Domaine médical :* Analyse d'opinions des médecins, patients, médicaments, services hospitalier....
- ✓ *Domaine éducation* :Développer le niveau d'enseignement à travers l'analyse et l'interprétation de l'opinion de l'étudiant à travers les méthodes d'enseignement (.i.e améliorer l'enseignement et l'apprentissage).

- ✓ *Marketing :* Du côté entreprises, permet au fournisseur plus de connaissances à propos des besoins des consommateurs, du côté client il peut donner son opinion, s'inspirer des opinions d'autres clients pour l'aidé à sa décision et aussi comparer les produits avant de les acquérir.

I.8 Exploration de textes

Plusieurs disciplines ont une relation directe ou moins directe avec l'analyse de sentiments et l'opinion mining. La fouille de textes (Text-mining en anglais)et même Data miningoffrent des outils et algorithmiques indispensables pour le traitement et la classification des sentiments [3].

Text-miningpeut être définie comme le processus au cours duquel le texte est transféré dans des données pouvant être analysées. Cela entraîne les procédures de création d'un index pour les termes individuels, basé sur l'emplacement du terme dans le texte d'origine, ou basé sur d'autres techniques ou protocoles. Les mots et les index peuvent ensuite être utilisés pour diverses méthodes d'analyse.

L'idée derrière l'indexation n'est pas de stocker l'emplacement du mot abstrait. Le processus d'indexation dans l'exploration de texte consiste plutôt à stocker la signification ou le concept du mot dans le contexte donné. Les concepts sont ensuite stockés dans la base de données. Cette dernière est utilisée pour une analyse et un traitement plus poussés. Les concepts sont extraits du texte donné viaune série de techniques linguistiques pour les processus d'extraction.

L'exploration de données (Data mining en anglais), contrairement à l'exploration de texte, est le processus de découverte de modèles et de tendances au sein de grands ensembles de données. L'exploration de données est un domaine interdisciplinaire, impliquant des constructions issues des domaines de l'apprentissage automatique, de l'intelligence artificielle, des systèmes de bases de données et des méthodes statistiques. Les deux opérations les plus essentielles du processus d'exploration de données sont le regroupement et la classification [7].

I.8.1 Raisons de l'exploration de texte

- ✓ Enrichir le contenu, en améliorant l'indexation du texte.
- ✓ Revue systématique de la littérature, en révisant systématiquement un grand contenu. Ce processus peut aider à réduire le temps et les efforts nécessaires à l'examen de la littérature générale.

- ✓ Découvrir de nouvelles perspectives: l'exploration de texte n'apporte pas à elle seule de nouvelles idées ou connaissances. L'exploration de données du contenu indexé résultant du processus d'exploration de texte peut en fait obtenir de nouvelles connaissances.
- ✓ Recherche linguistique computationnelle, où l'exploration de texte est considérée comme un outil pour des recherches linguistiques plus importantes.

I.8.2 Domaines de l'exploration de texte dans l'exploration de données

Les domaines de l'exploration de texte (figure 4) dans l'exploration de données sont [8]:

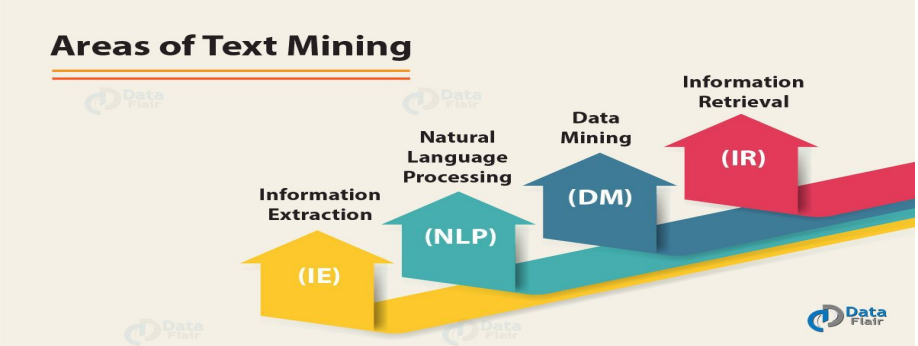


Figure 4 les domaines de l'exploration de texte dans l'exploration de données [8]

I.8.2.1 Recherche d'informations (RI)

La RI est considérée comme une extension de la recherche de documents. Que les documents renvoyés soient traités pour se condenser. Ainsi, la recherche documentaire suit une étape de synthèse de texte. Cela se concentre sur la requête posée par l'utilisateur. Les systèmes IR aident à restreindre l'ensemble des documents qui sont pertinents pour un problème particulier. Comme l'exploration de texte implique l'application d'algorithmes très complexes à de grandes collections de documents. En outre, IR peut accélérer considérablement l'analyse en réduisant le nombre de documents.

I.8.2.2 Exploration de données (DM)

L'exploration de données peut être décrite vaguement comme la recherche de modèles dans les données. Il peut davantage se caractériser par l'extraction de données cachées. Les outils d'exploration de données peuvent prédire les comportements et les tendances futures.

En outre, il permet aux entreprises de prendre des décisions positives fondées sur les connaissances. Les outils d'exploration de données peuvent répondre aux questions commerciales.

En particulier, cela a traditionnellement pris trop de temps à résoudre. Ils recherchent des bases de données pour des modèles cachés et inconnus.

I.8.2.3 Traitement du langage naturel (TALN)

TALN est l'un des problèmes les plus anciens et les plus difficiles. C'est l'étude du langage humain. La recherche TALN poursuit la vague question de savoir comment nous comprenons le sens d'une phrase ou d'un document? Quelles sont les indications que nous utilisons pour comprendre qui a fait quoi à qui? Le rôle de le TALN dans l'exploration de texte est de fournir le système dans la phase d'extraction d'informations en tant qu'entrée(figure 5) [8]:

A. Traitement morphologique

Le traitement morphologique traite des composants significatifs des mots. En morphologique, les chaînes sont décomposées en ensembles de jetons correspondant à la ponctuation, aux mots, aux sous-mots etc. Cependant en morphologique, la transformation ne se fera pas par l'ajout d'un préfixe ou d'un suffixe mais pourrait se faire par d'autres changements majeurs [9]

B. Syntaxe et analyse sémantique

L'analyse syntaxique traite de l'étude des relations structurelles entre les mots. Il peut être utilisé de deux manières. La première utilisation est d'identifier que la phrase est bien structurée ou non et la deuxième utilisation est de décomposer la phrase en une structure qui donne une relation syntaxique correcte. L'analyse syntaxique peut également être acquise par un ensemble de règles syntaxiques et à l'aide d'un parseur en prenant le mot du dictionnaire.[9]

C. Analyse pragmatique

L'analyse pragmatique porte sur l'étude de l'utilisation de la langue pour atteindre des objectifs. Cela fait partie du processus pour extraire les informations particulières du texte.[9]

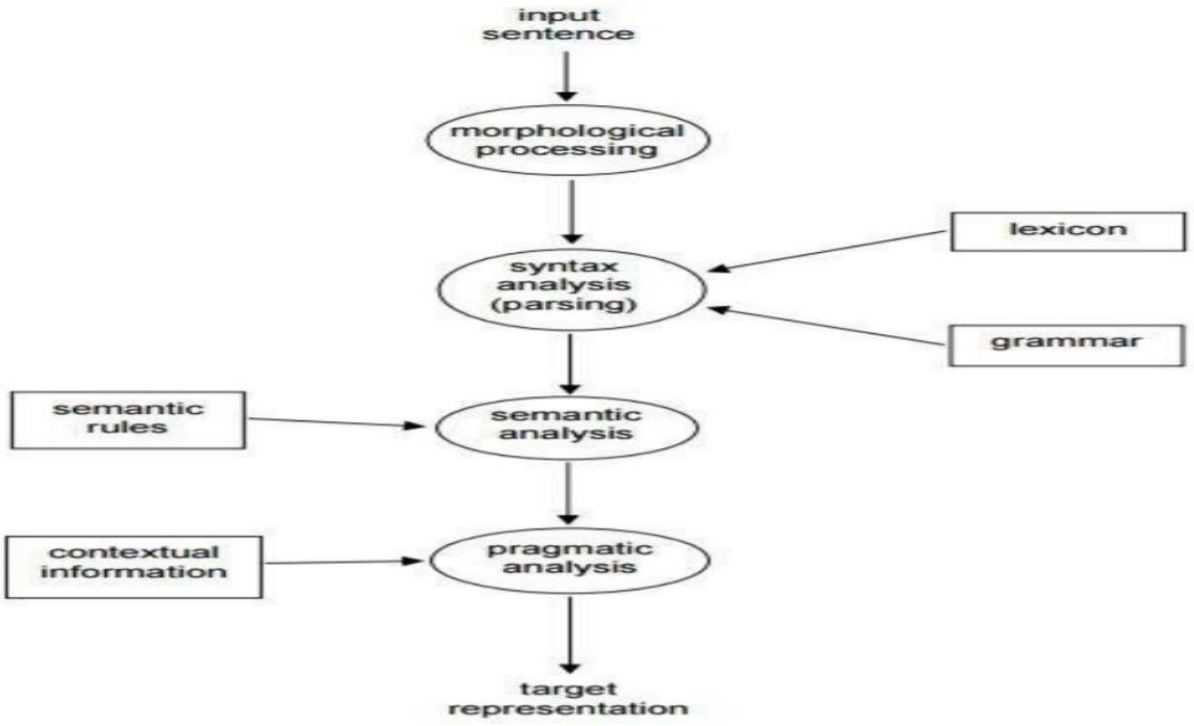


Figure 5 Étapes du traitement du langage naturel (TALN)[9]

I.8.2.4 Extraction d'informations (IE)

L'extraction d'informations est la tâche d'extraire automatiquement des informations structurées non structurées. Dans la plupart des cas, cette activité comprend le traitement de textes en langage humain au moyen de la TALN.

I.9 Méthodes de classification des sentiments

Dans la littérature, il existe de nombreuses méthodes et algorithmes pour mettre en œuvre des systèmes d'analyse des sentiments, qu'on peut les classer comme suit (Figure 6):

- *Approche d'apprentissage automatique* : systèmes qui s'appuient sur des techniques d'apprentissage automatique à partir de données.
- *Approche basée sur le lexique(TALN)*: systèmes qui effectuent une analyse des sentiments basée sur un ensemble de règles.
- *Approche hybride* : systèmes combinant à la fois des approches basées sur des règles et des approches automatiques.

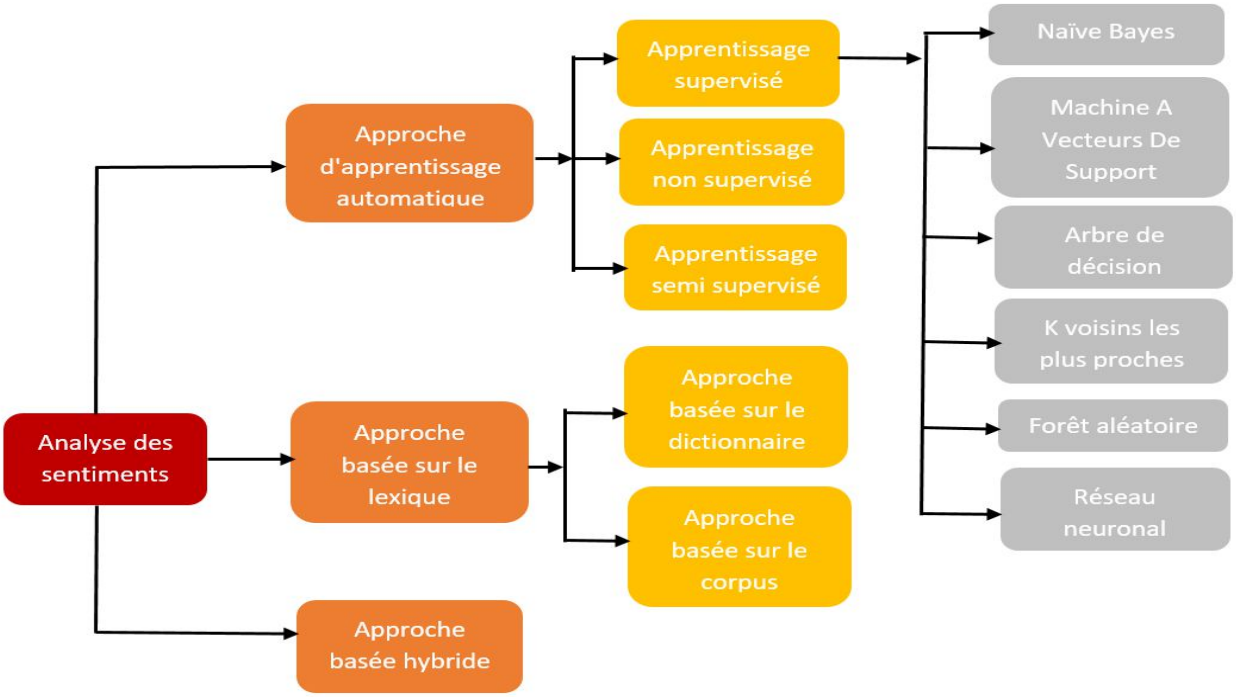


Figure 6 les approches d'analyse des sentiments

I.9.1 Approche D'apprentissage Automatique

L'apprentissage automatique est une tentative de comprendre et reproduire la faculté de l'apprentissage humain dans des systèmes artificiels.

Il s'agit de concevoir des algorithmes capables, à partir d'un nombre important d'exemples, d'en assimiler la nature afin de pouvoir appliquer ce qu'ils ont ainsi appris aux cas futurs. Ainsi, le but essentiel de l'apprentissage automatique est de déterminer la relation entre les objets et leurs catégories pour la prédiction et la découverte des connaissances.

On distingue ainsi trois types d'apprentissage: l'apprentissage supervisé, l'apprentissage non supervisé et l'apprentissage semi-supervisé.[10]

I.9.1.1 Apprentissage supervisé

L'apprentissage supervisé (ou classification) consiste à construire un modèle basé sur un jeu d'apprentissage et des labels (nom des catégories ou des classes) et à l'utiliser pour classer des données nouvelles.

Cette technique est utilisée dans plusieurs applications telles que les diagnostics médicaux, la prédiction des pannes et la détection des opinions trompeuses dans les réseaux sociaux. [11]

Il existe plusieurs algorithmes et techniques utilisés pour la classification supervisée telles que :

A. Naïve bayes(NB)

L'algorithme Naïve Bayes est un algorithme largement utilisé pour la classification de documents telles que les applications de détection de courriels (ou Spams) pour séparer les bons courriels des mauvais. Il y a beaucoup d'études qui ont été utilisées avec cette méthode. Basée sur le théorème populaire de Bayes à partir de statistiques, elle repose sur une hypothèse sous-jacente selon laquelle chaque entité est indépendante d'une autre, simplifiant ainsi considérablement l'espace de calcul. Par exemple, un fruit peut être classé comme une pomme s'il est rouge, rond et d'environ 10 centimètres de diamètre. Sous l'hypothèse d'indépendance de l'algorithme Naïve Bayes, ces caractéristiques seraient indépendantes les unes des autres, quelle que soit la corrélation possible entre la taille, la forme et la couleur d'un fruit. [1]

Certains utilisent un modèle multivarié de Bernoulli, c'est-à-dire un réseau bayésien sans dépendance entre les mots et les caractéristiques de mots binaires (par exemple Larkey et Croft 1996, Koller et Sahami 1997). D'autres utilisent un modèle multinomial, c'est-à-dire un modèle de langage uni-gramme avec des nombres de mots entiers (par exemple Lewis et Gale 1994, Mitchell 1997). [12]

Les équations (1) et (2) détaillent l'algorithme Naïve de Bayes, tandis que l'équation (3) détaille l'approche multinomiale du théorème de Bayes.

$$p(C_k|x) = \frac{p(C_k)\, p(x|C_k)}{p(x)} \qquad (1)$$

$$p(C_k|x_1, \dots, x_n) = \frac{1}{Z} p(C_k) \prod_{i=1}^{n} p(x_i|C_k) \qquad (2)$$

$$p(X|C_k) = \frac{(\sum_i x_i)!}{\prod_i x_i!} \prod_i {p_{ki}}^{x_i} \qquad (3)$$

B. Support Victor machine (SVM)

C'est une technique d'apprentissage automatique supervisé. SVM est bien connu dans l'analyse des sentiments. Il fonctionne mieux que les bayes naïfs en termes de problème de classification. En utilisant l'hyperplan, nous implémentons la classification SVM, la régression et d'autres travaux. Ce sont les groupes d'hyper-plans et travaillent dans un espace de grande dimension. Par conséquent, nous pouvons conclure que pour obtenir une bonne séparation entre toute classe, la distance doit être maximale pour le point de données le plus proche qui est également appelé marge fonctionnelle. Ainsi, si la marge était inférieure, l'erreur de généralisation entre le classificateur serait moindre. Une instance hyperplan est illustrée ci-dessous (figure 7). [9]

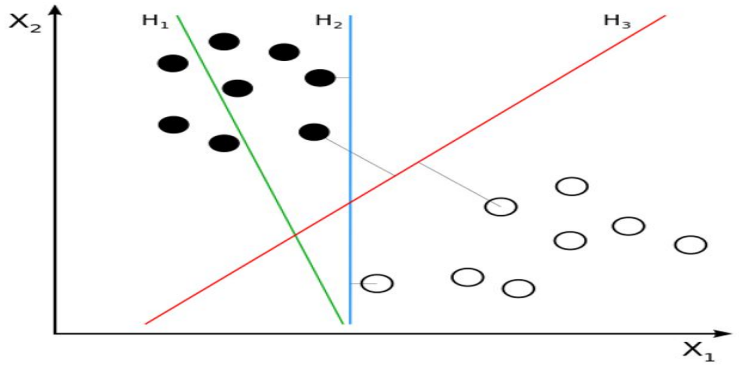


Figure 7 Visualisation d'une machine à vecteurs de support [1]

C. Arbre de décision (DT)

L'arbre de décision (Decisiontree en anglais) est un algorithme de classification qui relève des techniques d'apprentissage automatique supervisé. C'est une représentation graphique de toutes les solutions possibles à une décision basée sur certaines conditions (Figure 8). Par exemple, pour chaque nœud, il y a deux chemins «si nous devons le traverser, ce n'est pas le cas» et cela viendra jusqu'à ce que nous atteignions notre objectif. Dans le domaine de la prédiction et de la classification, l'arbre de décision joue un rôle majeur. Il existe différentes applications de l'arbre de décision. L'arbre de décision est utilisé dans la conception du produit lorsqu'il y a une série de décisions et que le résultat suivant dépend des résultats précédents. Une autre application de l'arbre de décision est lorsque l'utilisateur a un objectif qu'il souhaite atteindre au maximum. Profit ou il peut vouloir optimiser le coût. [9]

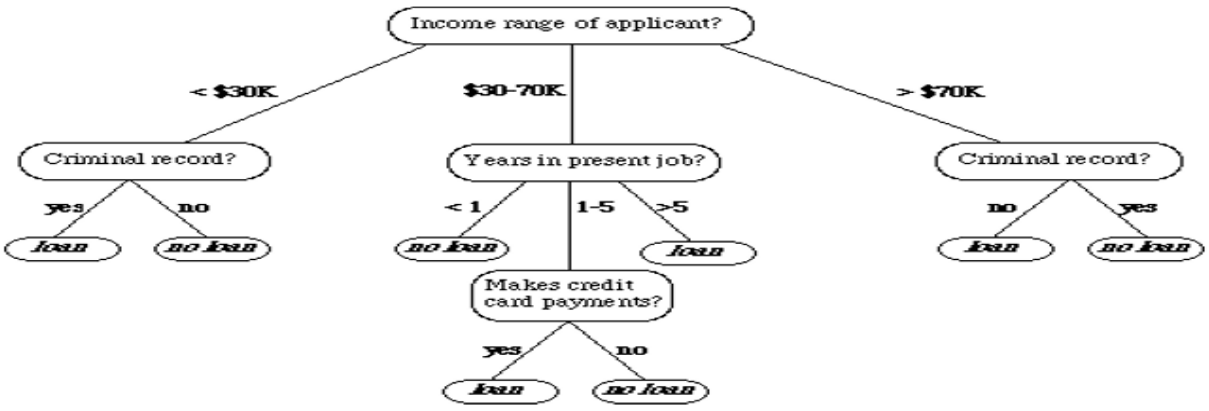


Figure 8 Une visualisation d'un algorithme d'arbre de décision évaluant s'il faut ou non approuver un prêt pour un demandeur [1]

D. K-voisins les plus proches (KNN)

k-NN est un algorithme très simple qui stocke tous les cas disponibles et classe les nouvelles données ou cas en fonction de la mesure de similarité. Il utilise l'ensemble de données dans sa phase d'apprentissage(Figure 9).

Par exemple, si la pomme ressemble le plus à la banane, l'orange, le melon plutôt qu'un singe, un chien ou un chat appartient probablement au groupe des fruits. En général, k-nn est utilisé dans l'application de recherche où vous recherchez l'élément similaire. Dans KNN, k désigne le nombre de voisins les plus proches qui détiennent la classe des nouvelles données ou des données de test. KNN est utilisé au niveau des industries.

Le plus grand cas d'utilisation de la recherche KNN est le système recommandé. Le système recommandé est une forme automatisée de type au comptoir de magasin. Lorsque vous demanderez le produit, il vous montrera non seulement le produit pertinent, mais vous suggérera ou vous recommandera également le produit lié à votre produit pertinent que vous souhaitez acheter. Algorithme KNN s'appliquant à la recommandation de produits comme dans Amazon et à la recommandation de médias dans Netflix. Dans le diagramme ci-dessus, indiquez la recherche de concept ou la recherche de documents sémantiquement similaires et la classification de documents contenant des sujets similaires. [13]

Figure 9 Système de recommandation utilisant l'algorithme KNN [10]

E. Forêts d'arbres décisionnels

C'est une application de graphe en arbres de décision permettant ainsi la modélisation de chaque résultat sur une branche en fonction des choix précédents. On prend ensuite la meilleure décision en fonction des résultats qui suivront. On peut considérer ceci comme une forme d'anticipation.En particulier dans la fouille d'opinions, ils ont l'avantage d'être lisibles et très faciles à comprendre et à interpréter ce qui est une des raisons de leurs succès. [10]

F. Le Boosting

Il s'agit d'une méthode de classification émettant des hypothèses qui sont au départ de moindre importance. Plus une hypothèse est vérifiée, plus son indice de confiance augmente. Ce qui prend de l'importance dans la classification.[10]

G. Réseau neuronal

Les réseaux neuronaux "NNs", parfois appelés réseaux neuronaux artificiels "ANNs", est un modèle de traitement de l'information qui simule le fonctionnement d'un système nerveux biologique. C'est similaire à la façon dont le cerveau manipule l'information. L'élément central de ce modèle est la structure du système de traitement. Il est composé d'un grand nombre d'éléments de traitement hautement interconnectés (neurones) travaillant à l'unisson pour résoudre des problèmes spécifiques.

G.1 Le modèle du perceptron

Le perceptron a été introduit en 1958 par Franck Rosenblatt. Il s'agit d'un neurone artificiel inspiré par la théorie cognitive de Friedrich Hayek et celle de Donald Hebb. Dans sa version la plus simple, le perceptron n'a qu'une seule sortie y à laquelle toutes les entrées xi sont connectées (figure 10), ses entrées et sorties étant booléennes. [14]

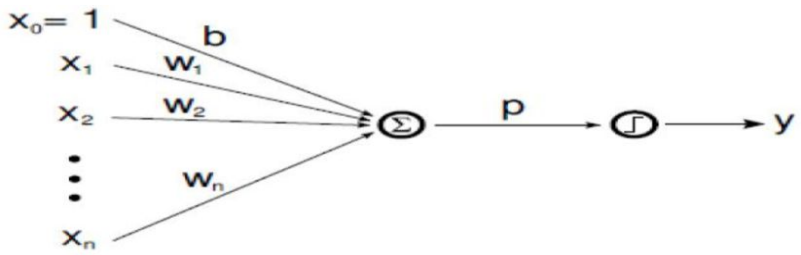


Figure 10 Modèle du perceptron [14]

La somme pondérée des entrées par les poids Wi associés aux entrées est appelée potentiel, noté p: $\boldsymbol{p} = \sum wi.xi$

Ce potentiel est alors soumis à une fonction seuil de type Heaviside :

$$Y = \{0 \ sis < 0$$

$$1 \ sis \geq 0\}$$

De nombreuses variantes ont été développées, notamment la version la plus couramment utilisée où les entrées et sorties sont des nombres flottants. Les valeurs de sortie -1 et 1 remplacent

fréquemment la valeur 0 et 1. Une valeur particulière, appelée biais a également été introduite. Ce biais peut être vu comme une entrée x0 supplémentaire dont la valeur est toujours de 1. Celui–ci a notamment été introduit pour un ajustement automatique du seuil, ou encore afin de pouvoir classer facilement un vecteur d'entrée dont toutes les composantes seraient nulles. Le principal obstacle de ce modèle est la détermination des poids.

Pour pallier ce problème, l'algorithme de rétro propagation du gradient a été mis au point. Soient S0 et S1 deux sous–ensembles de RN représentant les exemples négatifs et positifs de l'échantillon à apprendre.

L'algorithme consiste à présenter successivement les éléments de S0 et S1 et de mettre à jour les poids pour que la sortie se rapproche de la sortie désirée.

Cet algorithme repose sur le calcul du gradient de sortie puis sur la rétro propagation de celui-ci à travers la fonction de seuil puis des poids. C'est pourquoi la fonction de seuil doit être dérivable. La fonction d'activation de Heaviside est donc remplacée par des fonctions d'activation lui ressemblant et qui sont dérivables

G.2 Le perceptron multicouche (PMC)

Dans le modèle du Perceptron Multicouches, les perceptrons sont organisés en couches(Figure11). Les perceptrons multicouches sont capables de traiter des données qui ne sont pas linéairement séparables. Avec l'arrivée des algorithmes de rétro propagation, ils deviennent le type de réseaux de neurones le plus utilisé. Les PMC sont généralement organisés en trois couches, la couche d'entrée, la couche intermédiaire (dite couche cachée) et la couche de sortie. [14]

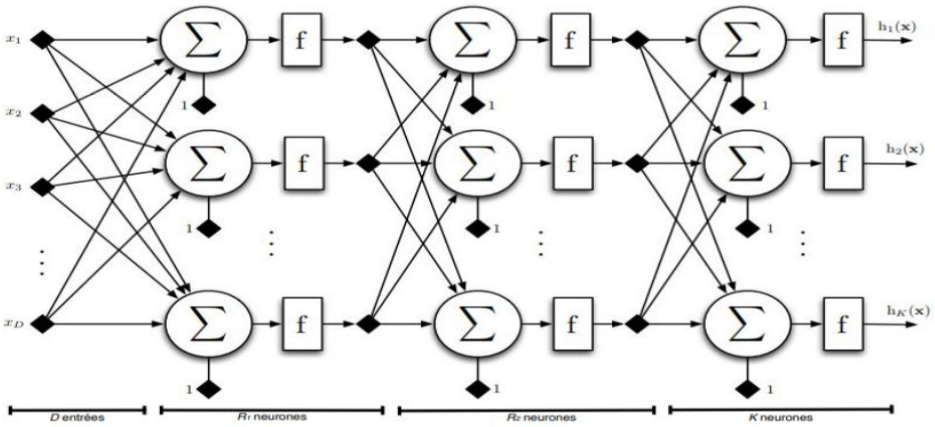


Figure 11 Schéma d'un perceptron multicouche [14]

G.3 Les fonctions d'activation

La fonction d'activation est une caractéristique importante du réseau neuronal. Neurone devrait être activé ou non est décidé par la fonction d'activation. Il calcule la somme pondérée des entrées et ajoute le biais. C'est une transformation non linéaire de la valeur d'entrée. Après la transformation, cette sortie est envoyée à la couche suivante. Sans une fonction d'activation, un réseau de neurones est juste un modèle de régression linéaire. [14]

- ✓ Types de fonction d'activation

*Fonction sigmoïde :*est comme une fonction pas à pas mais de nature non linéaire. Sa sortie varie entre 0 et 1 et a une courbe en forme de S. D'après la figure 12, il est facile de remarquer que les valeurs de X entre -2 et 2, les valeurs de Y sont très raides. Cela signifie qu'une petite modification de X entraînera un changement significatif des valeurs de Y.

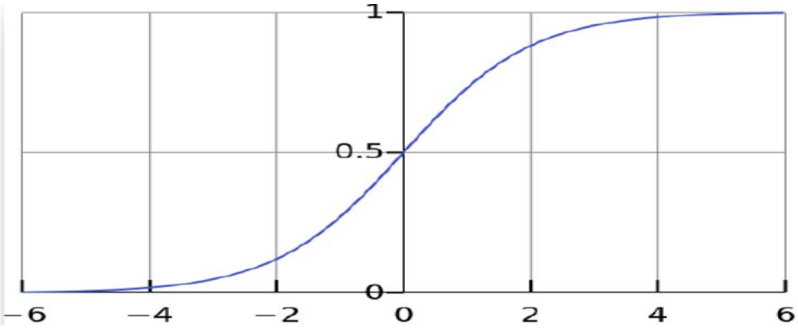


Figure 12 Représentation du graphe de fonction sigmoïde [14]

Fonction ReLu: C'est une fonction d'activation très simple. Supposons que l'entrée est la valeur X et si X est positif, la sortie sera X sinon 0. La fonction ReLu (unité linéaire rectifiée) est:

Fonction (X) = max (0, X)

Dans la figure 13, il y a une ligne droite et on dirait que c'est une fonction linéaire mais ReLu est non-linéaire dans la réalité. La plage de ReLu est [0, infini]. Calculer ReLu est moins cher que d'autres fonctions d'activation, car il a un fonctionnement mathématique simple. [14]

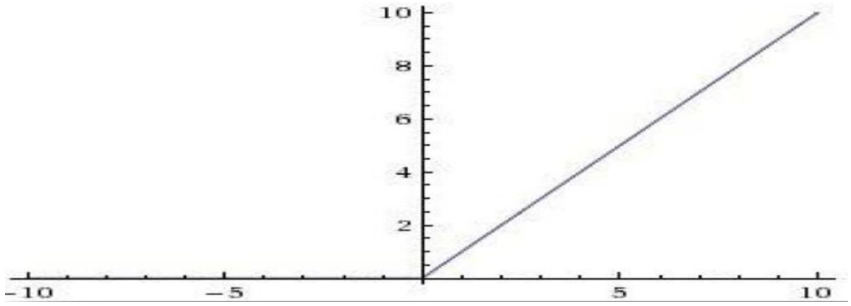


Figure 13 Représentation graphique de la fonction ReLu [14]

Fonction Softmax: Elle réalise une exponentielle normalisée (c'est-à-dire que la somme des sorties totalise 1)(Figure14). En combinaison avec la fonction d'erreur d'entropie croisée, elle permet de modifier les réseaux perceptrons multicouches pour l'estimation des probabilités de classes.

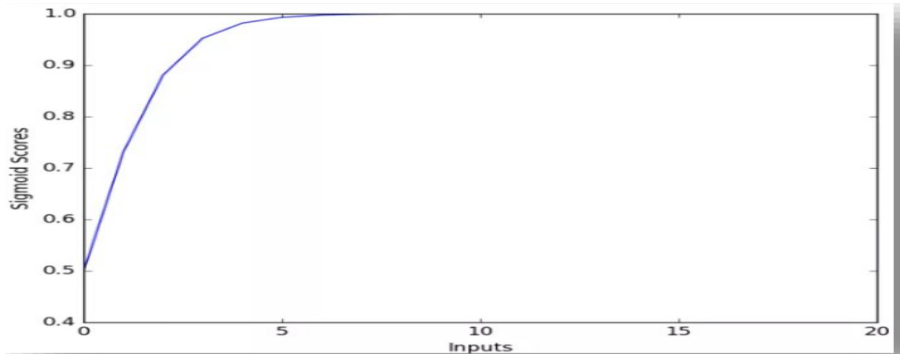


Figure 14 Représentation du graphe de fonction softmax [14]

G.4 Couche Dropout

Le technique dropout utilisé évite le sur-ajustement. En apprenant trop bien les données d'entraînement, on parle alors d'over-fitting. Pendant la formation, lorsqu'un modèle apprend le bruit et les détails dans une mesure qui a un impact négatif sur la performance des nouvelles données.

Pendant l'entraînement sur une couche spécifique, la moitié des neurones est éteinte. Ce processus améliore la généralisation car il oblige la couche à apprendre le même "concept" avec différents neurones. [14]

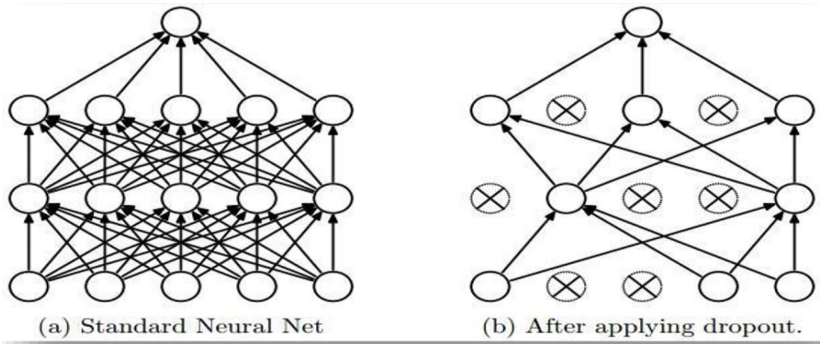


Figure 15 Réseau avant et après dropout [14]

La figure 15 .a est le réseau avant le processus dropout. Tous les neurones sont connectés à d'autres neurones tandis que la figure 15 .b montre les neurones désactivés. La suppression est désactivée dans la phase de prédiction.

H. Avantages et inconvénients des techniques d'apprentissage supervisé

Toutes les techniques d'apprentissage automatique supervisé ont leurs avantages et leurs inconvénients. Ils sont résumés dans le tableau 1:

Technique	***Avantages***	***Inconvénients***
SVM	*-Très haute robustesse* *-Moins de sur apprentissage* *-Robuste au bruit.*	*-Coûteux en calcul Performance lente.*
Naïve Bayes	*-Rapide dans la tâche d'entrainement et de classification.* *-Pas sensible aux caractéristiques non pertinentes.*	*-Suppose l'indépendance de la caractéristique.* *-Moins précis que SVM.*
K-NN	*-Entraînement très rapide.* *-Simple et facile à comprendre.* *-Gère bien les grands ensembles de données.*	*-Biaisée par la valeur de K.* *-Haute complexité de calcul. -Se laisse facilement tromper par des attributs non pertinents.*
DecisionTrees	*-faciles à comprendre.* *-Facilité à manipuler des données « symboliques ».* *-Multi-classe par nature* *-Interprétabilité de l'arbre!* *-Classification très efficace*	*-Sensibilité au bruit et points aberrants.* *-Stratégie d'élagage délicate*
RandomForest	*– Bien adaptées aux très grandes dimensions* *– Très simples à mettre en œuvre* *– Nombreux succès applicatifs ces dernières années Elles sont en général plus efficaces que les simples arbres de décision.*	*- possède l'inconvénient d'être plus difficilement interprétables* *- Apprentissage souvent long Valeurs extrêmes souvent mal estimées dans cas de régression*

Tableau 1 Résumé des avantages et des inconvénients des techniques d'apprentissage automatique supervisé[15]

I.9.1.2 Apprentissage non supervisé

L'apprentissage non supervisé (en anglais clustering) vise à construire des groupes (clusters) d'objets similaires à partir d'un ensemble hétérogène d'objets .Chaque cluster issu de ce processus doit vérifier les deux propriétés suivantes [10] :

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

Le processus de «clustering » repose sur une mesure précise de la similarité des objets qu'on veut regrouper. Cette mesure est appelée distance ou métrique.

Le « clustering » est utilisé dans plusieurs applications telles que le traitement d'images, les études démographiques, la recherche génétique, le forage des données et l'analyse des opinions. On distingue plusieurs algorithmes de clustering, par exemple KMeans.

I.9.1.3 Apprentissage semi-supervisé

L'apprentissage semi-supervisé utilise un ensemble de données étiquetées et non-étiquetés. Il se situe ainsi entre l'apprentissage supervisé qui n'utilise que des données étiquetées et l'apprentissage non-supervisé qui n'utilise que des données non-étiquetées.

L'utilisation de données non-étiquetées, en combinaison avec des données étiquetées, permet d'améliorer de façon significative la qualité de l'apprentissage. Un autre avantage vient du fait que l'étiquette de données nécessite l'intervention d'un utilisateur humain. Lorsque les jeux de données deviennent très grands, cette opération peut s'avérer fastidieuse.

Dans ce cas, l'apprentissage semi-supervisé, qui ne nécessite que quelques étiquettes, revêt un intérêt pratique évident et indiscutable.[10]

I.9.2 Approches basées sur le lexique

La méthode basée sur le lexique utilise un lexique composé de termes avec des scores de sentiment respectifs pour chaque terme:

I.9.2.1 Approche Basée Sur Le Dictionnaire

L'idée principale derrière l'approche basée sur un dictionnaire est d'utiliser des bases de données lexicales avec des mots d'opinion pour extraire le sentiment du document. La procédure de recherche est itérative. À chaque itération, l'algorithme prend un ensemble de mots mis à jour (ensemble étendu) et effectue une nouvelle recherche jusqu'à ce qu'il n'y ait plus de nouveaux mots à inclure. En fin de compte, un ensemble de mots de sentiment peut être examiné dans le but de supprimer les erreurs.[14]

I.9.2.2 Approche Basée Sur Le Corpus

Il est principalement utilisé pour trouver les nouveaux mots opiniâtres d'un corpus en utilisant une liste de mots sentimentaux connus. Et l'autre utilisation principale de l'approche basée sur le corpus est de créer un dictionnaire des sentiments à l'aide d'autres mots. En général, cette approche n'est pas dominante par rapport au dictionnaire, car elle nécessite un dictionnaire de tous les mots anglais. L'approche basée sur le corpus est divisée en deux techniques, à savoir sémantique et statique.[9]

I.9.3 Approches hybride

Le concept de méthodes hybrides est très intuitif : combinez simplement le meilleur des deux approches, celui basé sur des règles(lexique) et celui automatique. Généralement, en combinant les deux approches, les méthodes peuvent améliorer la précision. [3]

I.10 Le Deep Learning

Deep Learning a permis de faire des progrès importants dans les domaines de la classification des textes et du traitement du langage par exemple.

Deep Learning est un sous-champ d'apprentissage automatique des représentations de données d'apprentissage exceptionnel et efficace pour l'apprentissage des modèles (figure 16)[16].

Les modèles de Deep Learning sont bâtis sur le même modèle que les perceptrons multicouches précédemment décrits. Cependant, il convient de souligner que les différentes couches intermédiaires sont plus nombreuses. Chacune des couches intermédiaires va être subdivisée en sous partie, traitant un sous problème, plus simple et fournissant le résultat à la couche suivante, et ainsi de suite.

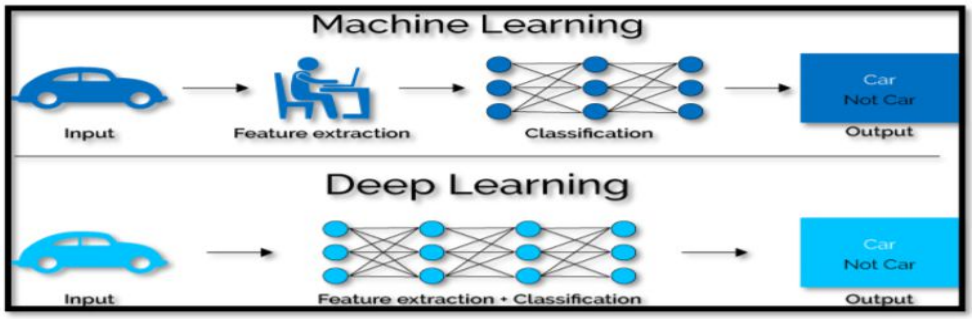


Figure 16ML vers Deep Learning. [16]

I.10.1Les réseaux neuronaux récurrents

L'idée derrière les RNN est d'utiliser des informations séquentielles. Dans un réseau neuronal traditionnel, nous supposons que toutes les entrées (et sorties) sont indépendantes les unes des autres. Mais pour de nombreuses tâches, c'est une très mauvaise idée. Si vous voulez prédire le mot suivant dans une phrase, il vaut mieux savoir quels mots l'ont précédé (figure 17). Les RNN sont appelés récurrents car ils effectuent la même tâche pour chaque élément d'une séquence, la sortie dépendant des calculs précédents. Une autre façon de penser aux RNN est qu'ils ont une «mémoire» qui capture des informations sur ce qui a été calculé jusqu'à présent. En théorie, les RNN peuvent

utiliser des informations dans des séquences arbitrairement longues, mais en pratique, ils se limitent à ne regarder que quelques étapes en arrière. Voici à quoi ressemble un RNN typique: [17]

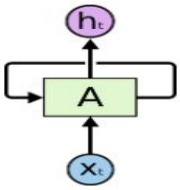


Figure 17 Les réseaux de neurones récurrents ont des boucles[17]

Dans le diagramme ci-dessus, un morceau de réseau neuronal, A, regarde une entrée X_t et génère une valeurh_t. Une boucle permet de transmettre des informations d'une étape du réseau à la suivante.

Ces boucles rendent les réseaux de neurones récurrents un peu mystérieux. Cependant, si vous réfléchissez un peu plus, il s'avère qu'ils ne sont pas si différents d'un réseau de neurones normal. Un réseau neuronal récurrent peut être considéré comme plusieurs copies du même réseau, chacune transmettant un message à un successeur. Considérez ce qui se passe si nous déroulons la boucle:

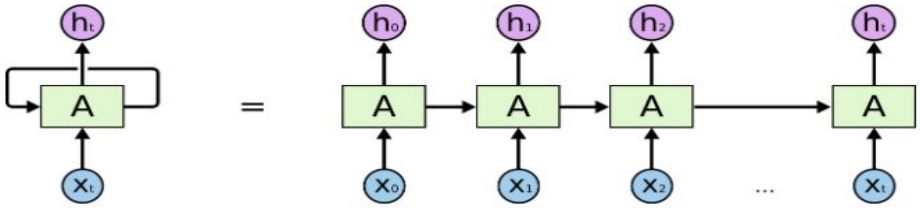


Figure 18 Un réseau neuronal récurrent déroulé. [17]

Cette nature en forme de chaîne révèle que les réseaux de neurones récurrents sont intimement liés aux séquences et aux listes. Ils constituent l'architecture naturelle du réseau neuronal à utiliser pour de telles données. Et ils sont certainement utilisés Au cours des dernières années, l'application des RNN a connu un succès incroyable. L'essentiel de ces succès est l'utilisation de «LSTM», un type très spécial de réseau neuronal récurrent qui fonctionne, pour de nombreuses tâches.

I.10.2Les réseaux Long Short-Term Memory (LSTM)

Les réseaux de mémoire à long terme à court terme généralement appelés simplement «LSTM» - sont un type spécial de RNN, capable d'apprendre les dépendances à long terme. Ils ont été introduits par Hochreiter&Schmidhuber (1997) et ont été affinés et popularisés par de

nombreuses personnes dans le cadre de leurs travaux. Ils travaillent très bien sur une grande variété de problèmes et sont maintenant largement utilisés.

Les LSTM sont explicitement conçus pour éviter le problème de dépendance à long terme. Se souvenir d'une information pendant de longues périodes est pratiquement leur comportement par défaut, pas quelque chose qu'ils ont du mal à apprendre.

Tous les réseaux neuronaux récurrents ont la forme d'une chaîne de modules répétitifs de réseau neuronal. Dans les RNN standard, ce module répétitif aura une structure très simple, telle qu'une couche de tanh unique. [17]

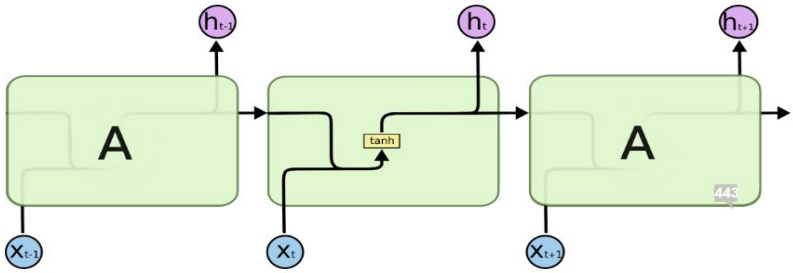


Figure 19 le module de répétition dans le réseau neuronal récurrent [17]

Les LSTM ont également cette structure en forme de chaîne, mais le module répétitif à une structure différente. Au lieu d'avoir une seule couche de réseau neuronal, il y en a quatre, interagissant d'une manière très spéciale.

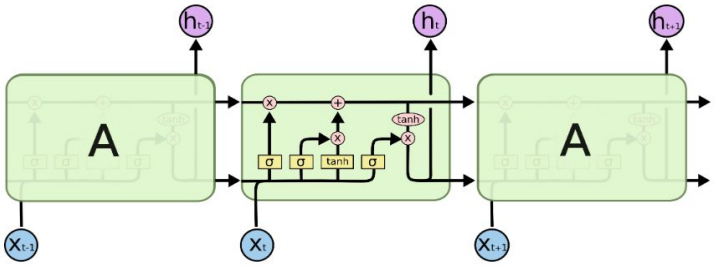
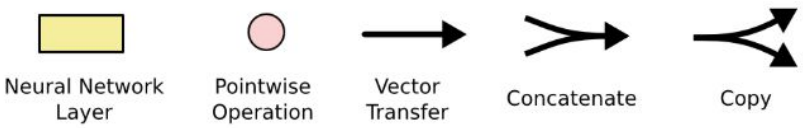


Figure 20 Le module de répétition dans un LSTM [17]



Sur la figure 20, chaque ligne contient le mot vecteur, généralement de la sortie du bloc à la suivante en entrée. Les opérations ponctuelles comme la multiplication ou l'addition de vecteurs sont effectuées par des cercles roses. Les rectangles jaunes sont une couche de réseaux de neurones appris. Les lignes fusionnant les unes avec les autres effectuent une concaténation. Le contenu est

copié et va à différents endroits avec la ligne de Branchement où $X_t$est l'entrée à l'étape t et $h_t$est la sortie à l'instant t.

I.10.2.1Description d'une cellule de Lstm

Une cellule Lstm agit comme un convoyeur d'information modulée à l'aide de 3 portes configurables (par entraînement)

- *Porte d'oubli* :décide quoi garder du contexte précédent pour mettre à jour de l'état de la cellule, basé sur l'entrée courante :

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

- *Porte entrée et circuit de mémoire:*$i_t$décide quoi contribuer de la mémoire ($\tilde{C}_t$ est équivalent à l'état caché dans un RNN standard) pour mettre à jour de l'état de la cellule, basé sur X_t

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

- *Mise à jour de la cellule* : On combine ce qui a été retenu de l'état précédent de la cellule avec celui retenu de la mémoire

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

- *Porte de sortie:* Décide quoi rendre publique du nouvel état de la cellule

$$o_t = \sigma\left(W_o \; [h_{t-1}, x_t] \; + \; b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

- X_t : vecteur d'entrée dans l'unité LSTM.
- f_t : vecteur d'activation des porte d'oubli.
- i_t : vecteur d'activation des portes d'entrée.
- o_t: vecteur d'activation des portes de sortie.
- h_t: vecteur de sortie de l'unité LSTM.
- c_t : vecteur d'état de cellule.
- W, U, b: les matrices des poids et des bais.

I.10.2.2 LSTM empilé (Stacked LSTM)

Le modèle LSTM d'origine est composé d'une seule couche LSTM cachée suivie d'une couche de sortie standard à anticipation.Le LSTM empilé (Stacked LSTM) est une extension de ce modèle qui a plusieurs couches LSTM cachées où chaque couche contient plusieurs cellules de mémoire.

A. L'augmentation de la profondeur

L'empilement de couches cachées LSTM rend le modèle plus profond, gagnant plus précisément la description en tant que technique d'apprentissage en profondeur.

C'est la profondeur des réseaux de neurones qui est généralement attribuée au succès de l'approche sur un large éventail de problèmes de prédiction difficiles

Des couches cachées supplémentaires peuvent être ajoutées à un réseau de neurones Perceptron multicouche pour le rendre plus profond. Les couches cachées supplémentaires sont censées recombiner la représentation apprise des couches précédentes et créer de nouvelles représentations à des niveaux élevés d'abstraction. Par exemple, des lignes aux formes en passant par les objets.

Un Perceptron multicouche à couche cachée unique suffisamment grand peut être utilisé pour approximer la plupart des fonctions. L'augmentation de la profondeur du réseau fournit une solution alternative qui nécessite moins de neurones et s'entraîne plus rapidement. En fin de compte, ajouter de la profondeur est un type d'optimisation de la représentation.

B. Architecture LSTM empilée

Étant donné que les LSTM fonctionnent sur des données de séquence, cela signifie que l'ajout de couches ajoute des niveaux d'abstraction des observations d'entrée au fil du temps. En effet, regrouper les observations dans le temps ou représenter le problème à différentes échelles de temps.[18]

Des LSTM empilés ou des LSTM profonds ont été introduits par Graves et al. Dans leur application des LSTM à la reconnaissance vocale, battant une référence sur un problème standard difficile.

Dans le même travail, ils ont constaté que la profondeur du réseau était plus importante que le nombre de cellules mémoire dans une couche donnée pour modéliser les compétences.

Les LSTM empilés sont désormais une technique stable pour résoudre les problèmes de prédiction de séquence. Une architecture LSTM empilée peut être définie comme un modèle LSTM

composé de plusieurs couches LSTM. Une couche LSTM ci-dessus fournit une sortie de séquence plutôt qu'une sortie de valeur unique vers la couche LSTM ci-dessous. Plus précisément, une sortie par pas de temps d'entrée, plutôt qu'un pas de temps de sortie pour tous les pas de temps d'entrée (figure 21).

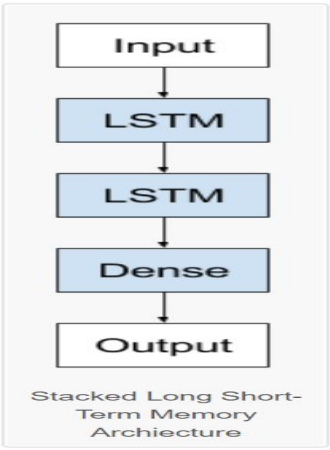


Figure 21 Architecture de mémoire empilée à long et court terme [18]

I.10.3 Utilité du deep learning

- Les fonctionnalités conçues manuellement sont souvent sur-spécifiées, incomplètes et prennent beaucoup de temps à concevoir et à valider.
- Les fonctionnalités apprises sont faciles à adapter, rapides à apprendre.
- L'apprentissage en profondeur fournit un cadre très souple, universel (presque ?) Et pouvant être appris pour représenter des informations mondiales, visuelles et linguistiques.
- Peut apprendre à la fois sans surveillance et sous surveillance.
- Apprentissage efficace du système conjoint de bout en bout.
- Utiliser de grandes quantités de données d'entraînement [16].

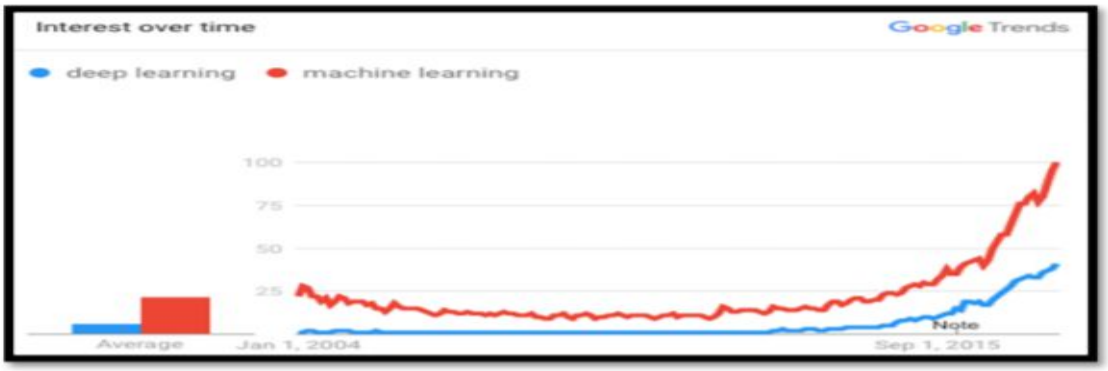


Figure 22 Google Trends l'Utilisation DL vers ML[16]

I.11 Outils d'analyse des sentiments

I.11.1 SentiWordNet

SentiWordNet[19]a été généré à l'aide du dictionnaire WordNet[1]. Chaque synset (c'est-à-dire un groupe de termes synonymes sur une signification particulière) dans WordNet est associé à trois scores numériques indiquant le degré d'association du synset avec du texte positif, négatif et objectif. En générant le lexique, les synsets initiaux (positifs et négatifs) ont été étendus en exploitant les relations de synonymie et d'antonymie dans WordNet, grâce à quoi la synonymie se conserve tandis que l'antonymie inverse la polarité avec un synset donné. Puisqu'il n'y a pas de relation synonyme directe entre les synsets dans WordNet, les relations: see_also, similar_to, pertains_to, derived_from et attribute ont été utilisées pour représenter la relation de synonymie tandis que la relation antonyme directe a été utilisée pour l'antonymie. Des glosses (c.-à-d. Les définitions textuelles des ensembles élargis de synsets ainsi que celle d'un autre ensemble supposé être composé de synsets objectifs) ont été utilisés pour former huit classificateurs ternaires. Les classificateurs sont utilisés pour classer chaque synset et la proportion de classification pour chaque classe (positive, négative et objective) ont été considérés comme les scores du synset. Dans une version améliorée du lexique (SentiWordNet 3.0), les scores ont été optimisés par une marche aléatoire en utilisant l'approche PageRank. Avec les synsets sélectionnés manuellement, puis propage la polarité des sentiments (positive ou négative) à un synset cible en évaluant les synsets qui se connectent au synset cible à travers l'apparence de leurs termes dans le brillant du synset cible.

SentiWordNet peut être vu comme ayant une structure arborescente comme indiqué sur la figure 23. Le nœud racine de l'arbre est un terme dont les nœuds enfants sont les quatre balises PoS de base dans WordNet (c.-à-d. nom, verbe, adjectif et adverbe). Chaque POS peut avoir plusieurs sens du mot en tant que nœuds enfants. Les scores de sentiment illustrés par un point dans l'espace triangulaire du diagramme sont attachés aux sens des mots. La subjectivité augmente (tandis que l'objectivité diminue) de bas en haut, et la positivité augmente (tandis que la négativité diminue) de droite à gauche du triangle.

1https://wordnet.princeton.edu/wordnet/

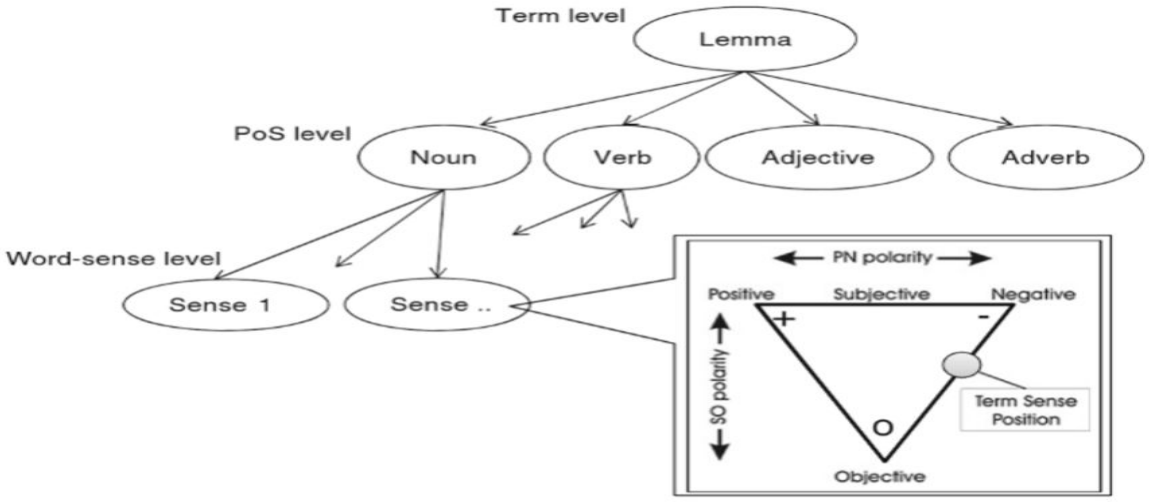


Figure 23 Diagramme montrant la structure de SentiWordNet[19]

I.11.2 Sentiment140

Sentiment140 (anciennement connu sous le nom "Twitter sentiment") est un outil en ligne gratuit qui a été créé par trois étudiants en computer science de Stanford, donc c'est un projet académique. Cet outil, contrairement à la plupart des autres sites d'analyse de sentiments, n'utilise pas de listes de mots positifs ou négatifs mais est fondé sur les algorithmes d'apprentissage automatique. Sentiment140 permet de découvrir des sentiments des tweets d'une marque, un produit ou un sujet sur Twitter. [20]

I.11.3 Tweetfeel

Tweetfeelest un service qui s'appuie sur les capacités temps réels de Twitter pour vous donner le sentiment des utilisateurs de Twitter sur un mot clé, une marque ou encore une star. L'évaluation de TweetFeel se fait sur la base de présence de mots clés précis dans les tweets tels que Good, Bad, etc... (Uniquement anglais pour l'instant), a quand un service similaire en français. Ensuite un pourcentage est calculé selon le nombre de tws ou négatifs un sentiment global de Twitter sur la marque. [21]

I.11.4 Twitrratr

Twitrratr est un outil en ligne gratuit, qui a émergé à partir d'un projet Startup Weekend. Twitrratr fonctionne à partir d'une liste de mots positifs et d'une liste de mots négatifs (40). Cet outil classe une opinion sur le mot clé de la requête s'il est capable de le croiser avec un mot d'une

des deux listes. Les mots positifs et négatifs qui servent à classer les tweets sont surlignés dans l'interface. [22]

I.11.5 Tweet Sentiments Analyses

Tweet Sentiments Analyses est un outil en ligne gratuit et open source d'analyse du sentiment sur Twitter. Il peut donner des sentiments positifs, négatifs et neutres des tweets sur le mot clé lancé dans la requête. Il peut travailler sur 12 langues. Il donne les résultats sous forme graphique. [23]

I.12 Mesures de performance de la classification des sentiments

Mesurer les performances d'un algorithme de Machine Learning est une étape indispensable et extrêmement importante pour produire un algorithme de qualité et qui répond aux attentes métiers.

En règle générale, pour mesurer les performances de la classification des sentiments, Les normes prédéfinies suivantes sont utilisés :

I.12.1 Confusion Matrix

C'est un tableau croisé entre les valeurs réelles et les prédictions (figure 24). Cette matrice permet d'identifier 4 catégories de résultats :

- *Vrais positifs (TP)* : les vrais positifs sont les cas où la classe réelle du point de données était 1 (Vrai) et la prédiction est également 1 (Vrai).
- *Vrais négatifs (TN):* les vrais négatifs sont les cas où la classe réelle du point de données était 0 (faux) et la prédiction est également 0 (faux).
- *Faux positifs (FP):* Les faux positifs sont les cas où la classe réelle du point de données était 0 (False) et la prédiction est 1 (Vrai). False est dû au fait que le modèle a prédit de manière incorrecte et positive parce que la classe prédite était positive. (1)
- *Faux négatifs (FN):* les faux négatifs sont les cas où la classe réelle du point de données était 1 (Vrai) et la prédite est 0 (Faux). False est dû au fait que le modèle a prédit de manière incorrecte et négative parce que la classe prédite était négative. (0)

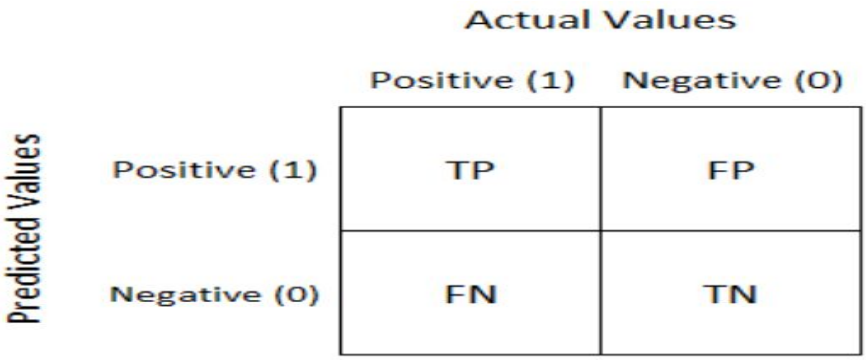


Figure 24 Confusion Matrix [24]

I.12.2 Accuracy, precision, recall and F-measure

À partir de cette table de confusion, on peut calculer différentes mesures pour évaluer le model :

- ✓ **Accuracy**

Accuracy est utilisée pour trouver et évaluer une matrice afin de voir l'efficacité de l'algorithme de classificateur pour calculer la précision que la formule est utilisée est:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

- ✓ **Precision**

C'est le nombre de résultats positifs corrects divisé par le nombre de résultats positifs prédits par le classificateur.

$$Precision = \frac{TP}{TP + FP}$$

- ✓ **Recall**

Il s'agit du nombre de résultats positifs corrects divisé par le nombre de tous les échantillons pertinents (tous les échantillons qui auraient dû être identifiés comme positifs).

$$Recall = \frac{TP}{TP + FN}$$

- ✓ **F-measure**

Les mesures F sont une combinaison de précision et de rappel, elles produiront une seule matrice de précision et de rappel. Pour calculer les mesures F, utilisez l'équation suivante.

$$F - measures = 2 * \frac{Precision * Recall}{Precision + Recall}$$

I.12.3 La courbe ROC et AUC

A. La courbe ROC

Une courbe ROC (receiver operating characteristic) est un graphique représentant les performances d'un modèle de classification pour tous les seuils de classification. Cette courbe trace le taux de vrais positifs en fonction du taux de faux positifs (figure25) [25]:

- ✓ TPR (True Positive Rate) / Rappel /Sensitivité

La sensibilité indique la probabilité d'un individu d'appartenir à la classe sachant qu'il devrait y appartenir.

$$TPR/Recall/Sensitivity = \frac{TP}{TP + FN}$$

- ✓ Spécificité

La spécificité indique, quant à elle, la probabilité qu'un individu n'appartienne pas à la classe à juste titre.

$$Specificity = \frac{TN}{TN + FP}$$

- ✓ FPR (false positive rate)

$$FPR = 1 - Specificity = \frac{FP}{TN + FP}$$

Voilà à quoi ressemblent des courbes de ROC :

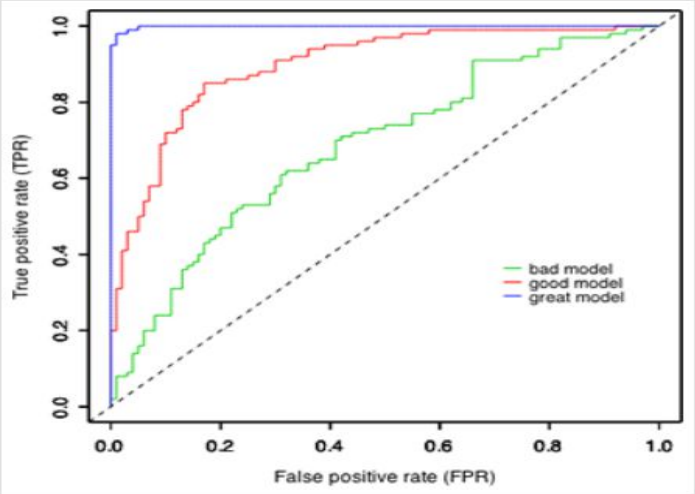


Figure 25 La courbe ROC [25]

B. L'AUC

AUC signifie "aire sous la courbe ROC". Cette valeur mesure l'intégralité de l'aire à deux dimensions située sous l'ensemble de la courbe ROC (par calculs d'intégrales) de (0,0) à (1,1) (figure 26) .L'AUC fournit une mesure agrégée des performances pour tous les seuils de classification possibles [25].

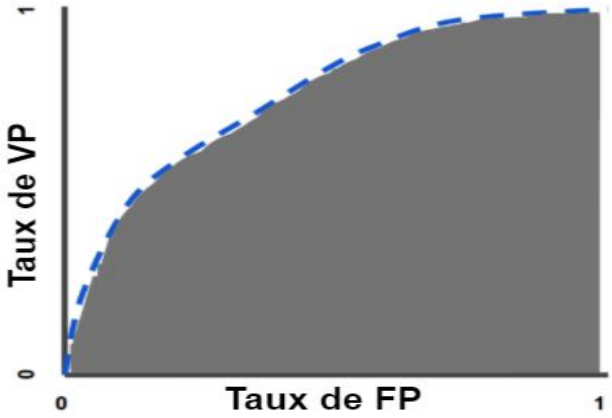


Figure 26 AUC (aire sous la courbe ROC [25]

I.13 Conclusion

Dans ce chapitre, nous avons présenté la revue de littérature pour l'analyse des sentiments. Ceci comprend un survole sur les principaux fondements, méthodes et techniques d'analyse et classification des sentiments. Le chapitre suivant sera dédié au volet pratique afin de concevoir et mettre en œuvre notre système d'analyse de l'état mental des internautes.

ANALYSE ET CONCEPTION

CHAPITRE II : ANALYSE ET CONCEPTION

II.1 Introduction

La collecte des données d'un big-social data n'est pas une tâche très simple. Nous devons en tenir compte des obstacles rencontrés. Cependant, un ensemble de donnéespréalablement collecté (data-set) est utilisé pour entraîner et tester deux modèles d'apprentissage: algorithmes traditionnelles et le Deep Learning.

Notre application d'analyse du sentiment de dépression se base sur une ressource lexicale (un dictionnaire donné ou SentiWordNet) au moment du prétraitement. Cette ressource linguistique aide à donner de l'information sur la polarité des mots afin de calculer le sentiment de chaque tweet. Les tweets reflètent les statuts postés sur le réseau social Twitter.

Ce chapitre étudie la manière dont les données seront traitées, analysées et principalement comment les classifier?. Avant de passer à cette chose, parlons de l'architecture proposée.

II.2 Architecture proposée

Cette section est consacrée à la description de la collection des données utilisée et au processus général de la méthodologie de classification des tweets du big-social data "Twitter":

II.2.1 Collection de données (Dataset)

Twitter est rapidement devenu l'un des médias sociaux les plus populaires depuis son lancement, il conseille plus de 313 millions d'utilisateurs actifs qui produisent plus de 6000 tweets sur Twitter chaque seconde chaque année. En faveur de la collecte des données relatives à la dépression, les développeurs continuaient à surveiller chaque tweet en streaming qui inclut le mot et les mots proches à «dépression» dans toute la communauté Twitter [26].

Pour gagner du temps et surpasser les problèmes de collectes et de scrapping via l'API-twitter, nous avons réutiliséce dataset[27] créé par Coppersmith et al. pour la tâche partagée de la linguistique computationnelle et de la psychologie clinique (CLPsych) 2015 [1]. Les données ont été consultées auprès des utilisateurs de Twitter qui ont déclaré un diagnostic de dépression, puis ont été normalisées dans une distribution démographiquestandard. Chaque utilisateur de cet ensemble de données est anonymisé à des fins de confidentialité.Avant de diffuser l'ensemble de données aux participants, Coppersmith et al. a permis de concevoir un ensemble de données qui comprenait 574 personnes (~ 63% de l'ensemble de données) sans problème de santé mentale et 326 utilisateurs (~ 36% de l'ensemble de données) avec un problème de santé mentale de la dépression. Pour les

besoins de recherche, cela a abouti à 1 253 594 documents (tweets) comme variables de contrôle et 742 560 documents avec un état de santé mentale de dépression [1].

II.2.2 Processus général de la méthodologie du système

Dans ce PFE, Une série d'expérimentations est mené sur le data-set stocké dans un fichier texte (.txt). Il est formé de tweets collectés selon les mots clefs ('depression', 'anxiety', 'mental health', 'suicide', 'stress', 'sad') pour une durée d'une heure. La figure 27 représente le processus général de la méthodologie du système.

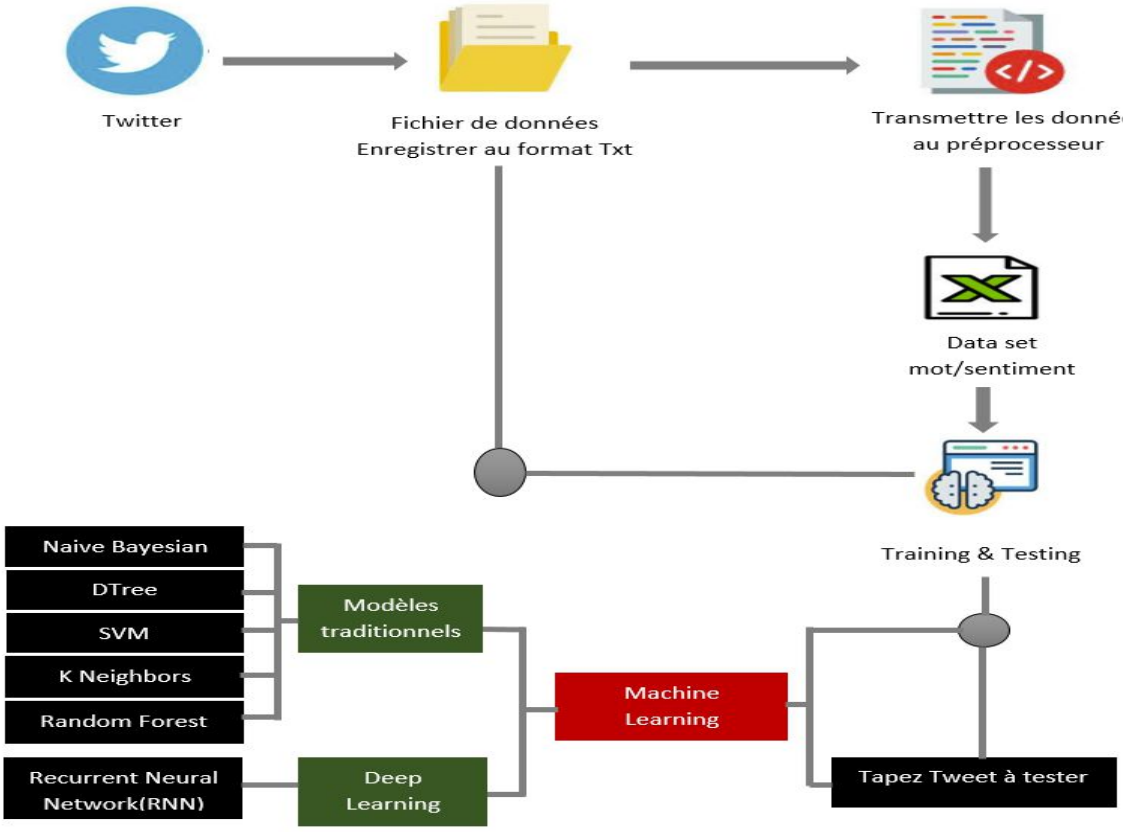


Figure 27 Processus général de la méthodologie du système

Les étapes à suivre :

1. Une étape de prétraitement (§ II.2.2.1) passe par le data set (.txt) via une ressource lexicale (Sentiwordnet (§ II.2.2.2 - Algorithme 1) ou un dictionnaire donné (.tsv)). Ce dictionnaire (.tsv) contient des mots avec leur polarité correspondante, qui est essentielle pour calculer le sentiment de chaque texte du tweet. Chaque mot extrait du texte est comparé aux mots de la ressource lexicale et un score est donné.

2. Une fois le prétraitement terminé. La sortie est un ensemble de données traitées (.xlsx) contenant l'ID, le texte et le sentiment de chaque tweet annoté avec des libellés positifs, négatifs ou neutres séparés en colonnes. Avec cette sortie, nous avons maintenant un ensemble de données Twitter et son sentiment correspondant filtré par les mots-clés (Positif, Neutre ou Négatif).
3. Pour le sous-processus de training et prédiction, notre code passe par le fichier (.xlsx) et en même temps récupère le tweet correspondant à l'id de chaque sentiment. En utilisant cela, nous utilisons ainsi les données originales et les transmettons tous à nos classificateurs. Lorsque tout est terminé, nous devrions avoir tous les résultats à évaluer des différents classificateurs.
4. Optionnellement, nous avons également la possibilité de saisir un exemple de tweet, ce dernier passera par l'un des classificateurs qui ont un niveau élevé d'AUC pour prédire les sentiments correspondants.

Que pourrait signifier le résultat? *Positif:* signifie qu'il est peu probable que la personne souffre de dépression ou d'anxiété. *Neutre:* est le niveau intermédiaire dans lequel l'utilisateur peut ou non avoir une dépression, mais peut également être plus enclin à être déprimé. À ce stade, l'utilisateur peut afficher des symptômes comme la dépression. Enfin, *Négatif* est le niveau le plus bas où les symptômes de dépression et d'anxiété sont détectés via les tweets des utilisateurs. Les mots les plus négatifs que l'utilisateur utilise signifient que le tweet a une émotion plus négative.

II.2.2.1 Prétraitement de données

Étant donné que les données recueillies à partir des tweets sont biaisées et bruyante, le nettoyage des tweets est notre première tâche. Ce prétraitement consiste à structurer et à faciliter l'utilisation des données textuelles originales [10].

Cependant, Le prétraitement des données est le processus de nettoyage et de préparation des tweets pour la catégorisation. Cette première étape (données textuelles) est la reconnaissance des termes, des ponctuations, des fins de paragraphes et des phrases. Nous devons aussi unifier l'écriture des lettres en minuscule afin de faciliter la mise en correspondance.

Les tweets en ligne contiennent généralement beaucoup de bruit et des parties non informatives telles que des balises HTML, des scripts et des publicités. De plus, au niveau des mots, de nombreux mots dans les tweets n'ont pas d'impact sur l'orientation générale de celui-ci. Garder

ces mots rend la dimensionnalité duproblème élevée et donc la classification plus difficile puisque chaque mot dans les tweets est traité comme une dimension.

Voici l'hypothèse d'un prétraitement correct des données: réduire le bruit dans les tweets devrait aider à améliorer les performances et accélérer le processus de classification, aidant ainsi à l'analyse des sentiments en temps réel.

Les phases de prétraitement choisis ci-dessous (Figure 28) ont le plus d'impact et d'influence sur l'obtention d'un meilleur résultat:

Figure 28 Processus de prétraitement

A. Filtrage

L'opération de filtrage a pour but la suppression des métadonnées contenues dans les tweets, nous pouvons citer comme exemples [28] :

- ✓ Liens URL : si le message contient un lien URL (exemple : http://bit.ly/KCairo). L'opération de filtrage le supprime, étant donné qu'il ne contient aucune information qui influe sur l'opinion exprimée dans le message (figure 29).

Figure 29 Exemple d'un tweet contenant lien url[6]

- ✓ Les symboles : dans le cas des tweets et des postes, souvent les utilisateurs utilisent des symboles (exemple : le hashtag « # ») pour mettre en évidence un mot précis, ces symboles doivent être supprimés de façon à pouvoir utiliser les mots par la suite.

Figure 30 Exemple de Tweet avec des hashtags[6]

Après analyse du tweet (figure 30), le processus de filtrage supprime le hashtag Tweet devient : (FIFAarrestsFIFAgate) a la place de #FIFAarrests #FIFAgate

B. Suppression des mots vides

Dans ce cas, nous effectuons une suppression des mots qui n'influent pas sur l'opinion exprimée dans le message et qui, de plus, augmente considérablement et inutilement le nombre de mots dans le vocabulaire. Ces mots sont : [10]

- Les conjunctions de coordination (for, and, nor, but, or, yet, so).
- Les determinants (a/an, the, this, that, these, those).
- Les prepositions (at, in, to).

Pour cibler et éliminer les mots vides, nous avons importé une liste des mots les plus fréquemment utilisés à partir de *NLToolkit* au début avec de *nltk.corpus import stopwords*. Après exécuter *stopwords.word (insérer la langue)* pour obtenir une liste complète pour chaque langue. [29]

Exemple:

Tweet contentant des mots vides :

"How to Deal with Stress, Anxiety and BipolarDisorder–Anxietyis one of manytroublingsymptoms of bipolar",

Après l'analyse du tweet:

"How Deal Stress, AnxietyBipolarDisorder - Anxiety one manytroublingsymptomsbipolar".

C. Suppression des ponctuations

Dans de nombreux travaux, il est courant de supprimer les signes de ponctuation en prétraitement. Une façon de le faire est de parcourir la série avec la compréhension de la liste et de conserver tout ce qui n'est pas dans *string.punctuation*, une liste de toute la ponctuation que nous avons importée au début avec *import string* [30].Cet ensemble de symboles sont : [!"#$%&'()*+,./:;<=>?@[\]^_`{|}~]:[29]

D. Lowercase

La conversion de toutes nos données en minuscules aide au processus deprétraitement et aux étapes ultérieures de l'application NLP, lorsque nous effectuons l'analyse (figure 31).[31]

```
text = "I am a person. Do you know what is time now? "
text = text.lower()
print(text)
```

```
i am a person. do you know what is time now?
```

Figure 31 Exemple de conversion de données en lettres minuscules (lowercase)

E. Tokenization

Dans cette partie, l'opération de Tokenization[29] est l'acte de décomposer une séquence de chaînes en morceaux tels que des mots, des mots-clés, des phrases, des symboles et d'autres éléments appelés jetons (tokens). Dans le processus de tokenization, certains caractères comme les signes de ponctuation sont ignorés. [10]

Les jetons deviennent l'entrée d'un autre processus comme l'analyse syntaxique et l'exploration de texte. La tokenization joue un rôle important dans le processus d'analyse lexicale.

Figure 32 Tweet composé du texte et des signes de ponctuation [12]

Après analyse du tweet, le processus de tokenisation transforme le texte d'un tweet en liste de tokens (mot, ponctuation,..) donc le tweet (figure 32) devient : ('Alma' , 'ne' , 'déprime' , 'pas' , 'please' , '.' , 'Tu' , 'méritais' , 'bien' , 'mieux' , '.').

F. POS tagger

Chaque mot du tweet à son rôle syntaxique qui définit comment le mot est utilisé. Ces rôles syntaxiques font partie du discours lié à ce mot. En anglais, il y a huit parties du discours qui sont connues comme le verbe, le nom, le pronom, l'adjectif, l'adverbe, la préposition, la conjonction et l'interjection. [10]

Le processus consistant à classer les mots dans leurs parties du discours et à les étiqueter, en conséquence est connu sous le nom de marquage de partie de discours. On a utilisée des bibliothèques externes pour la segmentation des mots. Voici un exemple (figure 33) :

```
Entrée [4]: from nltk import word_tokenize, pos_tag
            text = "my anxiety and depression the whole day"
            tagged_sentence = pos_tag(word_tokenize(text))

Entrée [5]: tagged_sentence

Out[5]: [('my', 'PRP$'),
         ('anxiety', 'NN'),
         ('and', 'CC'),
         ('depression', 'NN'),
         ('the', 'DT'),
         ('whole', 'JJ'),
         ('day', 'NN')]
```

Figure 33 POS tagging exemples

Ici, nous voyons cela 'anxiety ' et 'depression ' et 'day' sont NN (un nom), 'and' est CC (une conjonction de coordination), 'whole' est JJ (un adjectif).

G. Lemmatisation

La lemmatisation est l'étape qui désigne l'analyse lexicale chargée de faire regrouper les mots d'une même famille qui partagent le même suffixe lexical. Chacun des mots du texte se trouve ainsi réduit en une entité appelée « Lemme ». Ce lemme désigne la forme canonique des mots. La lemmatisation regroupe les différentes formes que peut avoir un mot.

Par exemple, un nom en pluriel va être réduit au singulier, un verbe à son infinitif, etc.La lemmatisation est étroitement liée à la racinisation. La différence est qu'un stemmer fonctionne sur un seul mot sans connaissance du contexte, et ne peut donc pas fairela distinction entre des mots qui ont des significations différentes selon la partie du discours. Cependant, les stemmers sont généralement plus faciles à implémenter et à exécuter plus rapidement, et la précision réduite peut ne pas avoir d'importance pour certaines applications. La normalisation des verbes est une forme de la lemmatisation.

II.2.2.2 Catégorisation sémantique du texte

Une fois l'étape de prétraitement terminé, l'étape suivante consiste à analyser et à classifier chaque tweet.

Cette étape est composée de deux composants principaux comme le montre la figure 34. Il s'agit de l'extraction des polarités et de la catégorisation du texte. Le texte d'entréeest prétraité, les polarités extraites des lexiques sont modifiées, normalisées et agrégées pour effectuer une classification de sentiment binaire (positive / négative).

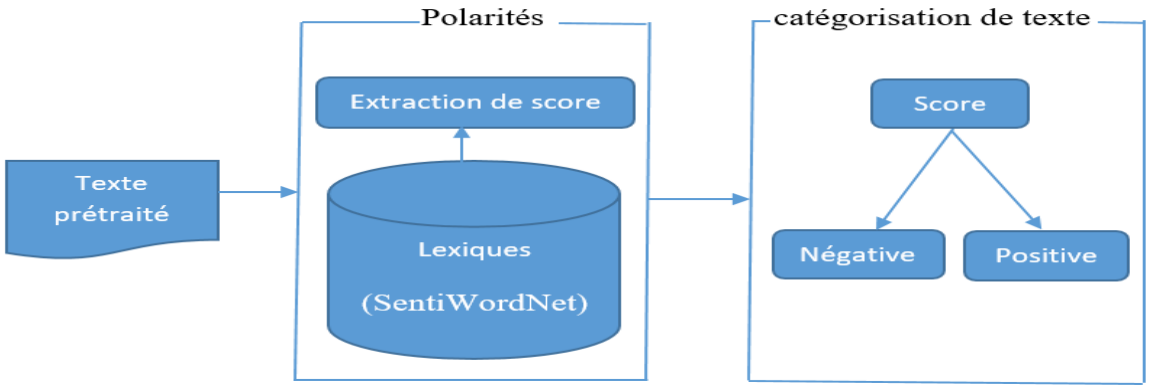


Figure 34 Catégorisation sémantique du texte de tweet

A. Extraction des scores de polarité

Dans SentiWordNet, chaque mot contient un score de positivité et un score de négativité. Tout d'abord, Le texte d'entrée est transmis au composant du prétraitement. À partir duquel les jetons (les mots) sont déterminés.Pour chaque mot qui n'est ni un mot vide ni une marque de ponctuation, nous cherchons le mot dans WordNet en utilisant son étiquetage morpho-syntaxique (PoStagging) et son lemme pour trouver les mots les plus proches du sens de mot recherché.Le mot sera rejeté dans les trois cas suivants :

- ✓ S'il ne s'agit pas d'un nom, d'un adjectif, d'un adverbe ou d'un verbe.
- ✓ Si le mot de texte n'est pas réduit à l'entité "Lemme".
- ✓ Si le mot n'obtient pas la liste de tous les synsets.

Si tous les mots du texte sont rejetés, le texte suivant sera traité. La fonction senti_synsets retourne automatiquement un positive_score, et un negative_score pour le premier synset le plus courant.

Deux scores différents pour la polarité du texte soit calculés. Un score contient l'addition du score des mots trouvés positifs (noté *score_positive*), l'autre contient l'addition du score des mots

trouvés négatifs (noté *score_negative*). Et puisque les scores de chaque mot sont des valeurs absolues, nous calculons le score du mot avec formule suivante:

Word_Sentiment = score_positive – score_negative.

Le score de sentiment pour un post p est déterminé en utilisant l'équation suivante:

$$score(p)_{dim} = \frac{\sum_{w_j \in p} score(w_j)_{dim}}{|n|}$$

Où $score(p)_{dim}$ est le score du post p dans la dimension sentiment de dim (dim est positif ou négatif) tandis que $score(w_j)_{dim}$ est le score de sentiment du terme w_j en p extrait de SentiWordNet, et $|n|$ est le nombre de termes porteurs de sentiments en p.

L'utilisation de senti_wordnet se résume par l'algorithme1 suivant :

Algorithme 1 : Calculer le score d'un texte de tweet

```
Get_sentiment_NLTK_sentiword
Defpenn_to_wn(tag):
   if tag.startswith('J'):
      return wn.ADJ
elseiftag.startswith ('N'):
      return wn.NOUN
elseiftag.startswith('R'):
      return wn.ADV
elseiftag.startswith('V'):
      return wn.VERB
return None
defsentiment_sentiwordnet ():
counter = 0 ➔ compter le nombre des texts
for text in  tweets_dataset['text']:
         raw_sentences = sent_tokenize(text)
         tokens_count = 0 ➔ compter le nombre de mots
         sentiment_score = 0
         for raw_sentence in raw_sentences:
tagged_sentence = pos_tag(word_tokenize(raw_sentence))
```

```
        for word, tag in tagged_sentence:
        wn_tag = penn_to_wn(tag)
        if wn_tag not in (wn.NOUN, wn.ADJ, wn.ADV, wn.VERB):
        continue ➔ mot suivant
        lemma = lemmatizer.lemmatize(word, pos= tag)
        if not lemma:
        continue ➔ mot suivant
        synsets = wordnet.synsets(lemma, pos= tag)
        if not synsets:
        continue ➔ mot suivant
        synset = synsets[0]
        swn_synset = sentiwordnet.senti_synset(synset.name())
        sentiment_score += swn_synset.pos_score() - swn_synset.neg_score()
        tokens_count += 1
if tokens_count == 0:
        counter += 1
        continue ➔ texte suivant
sentiment_score = sentiment_score /tokens_count
```

B. Catégorisation de texte

Quand l'algorithme finit de parcourir tous les mots du tweet en entrée, le sentiment du message sera :

- Si la polarité est < 0 : le message exprime une opinion négative (état dépressif).
- Si la polarité est >= 0 : le message exprime une opinion positive (état non dépressif).

Le pseudo code montré dans l'algorithme 2 est appliqué pour classifier les tweets:

Algorithme 2:Mesure de polarité

```
if sentiment_score>= 0:
        Return "Possitive"
else:
        Return "Negative"
```

II.2.2.3 Catégorisation lexico-syntaxique du texte

Dans cette étape,il s'agit d'extraction des polarités du dictionnaire (.tsv), le calcul du score, et de la catégorisation multi-classes du texte (positif, neutre, négatif) (figure 35) :

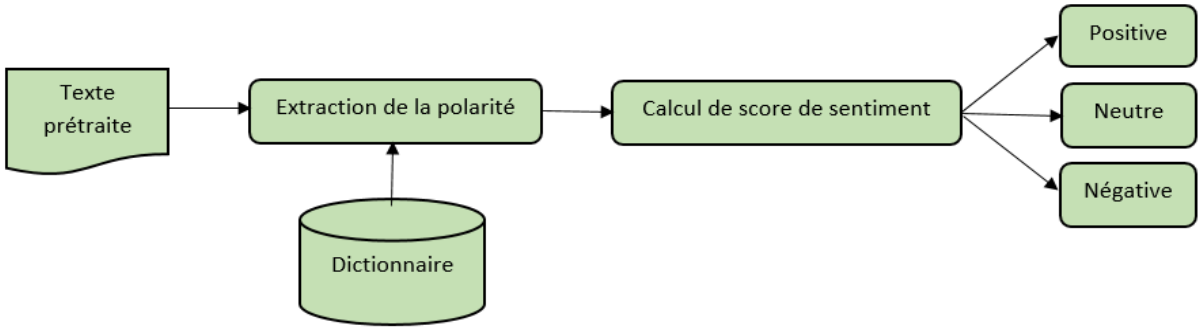


Figure 35 Catégorisation lexico-syntaxique du texte

A. Extraction des polarités et calcul du score

A partir de l'étape de prétraitement, les jetons (les mots) du texte d'entrée sont obtenus, et Chaque mot est recherché dans le dictionnaire, si le mot n'est pas trouvé, il est rejeté et le mot suivant est recherché. Si le mot se trouve dans le dictionnaire, la polarité de ce mot est tirée du dictionnaire (positive, négative ou neutre).Le score de sentiment du terme est calculé comme suit :

- ✓ Si la polarité du mot est positive, retourne un 1.
- ✓ Si la polarité du mot est négative, retourne un -1.
- ✓ Si la polarité du mot est neutre, retourne un 0.

Le score du sentiment pour un texte (t) est déterminé en utilisant l'équation suivante :

$$score(t)_{dim} = \frac{\sum_{w_i \in p} score(w_i)_{dim}}{|m|}$$

Où $score(t)_{dim}$ est le score du texte$t$ dans la dimension sentiment de dim (dim est positif, négatif ou neutre) tandis que $score(w_i)_{dim}$ est le score de sentiment du terme w_i en t(mentionné au-dessus), et $|m|$est le nombre de termes en t trouvés dans le dictionnaire.

B. Catégorisation de texte

Une fois que nous avons le score de sentiment textuel, la polarité du texte sera:

- ✓ Si le score est >= 0.2 : le tweet exprime une opinion positive (état non dépressif).
- ✓ Si le score est < 0.2 et le score est > -0.5: le tweet exprime une opinion neutre (état neutre).
- ✓ Si le score est<= -0.5 : le tweet exprime une opinion négative (état dépressif).

II.2.2.4 Sauvegarde des données

Une fois la classification des sentiments terminés. Les données sont stockées dans un fichier.xlsx(chaque approche a un fichier.xlsx). En l'ouvrant, l'ID (tweet), le texte original, le texte après la phase du prétraitement des données et le sentiment de chaque tweet sont séparés en 4 colonnes. Avec cette sortie, Nous aurons un ensemble de données Twitter avec le sentiment correspondant.

II.2.2.5 Training et Prédiction

Pour cette phase, nous testons deux architectures différentes: le modèle d'apprentissage traditionnel et le modèle d'apprentissage en profondeur.

Les modèles traditionnels font référence à des techniques d'apprentissage automatique classiques (classificateur d'apprentissage supervisé), telles que le classificateur Naïve Bayes (NB), le classificateur d'arbres de décision (DT), les machines à vecteurs de support (SVM), le classificateur k-plus proche-voisins (KNN) et le classificateur de forêt aléatoire (RF).Pour l'apprentissage en profondeur, nous avons utilisé le modèle de réseau neuronal récurrent(RNN).

Pourl'apprentissage automatique traditionnelle, les fonctionnalités sont définies et extraites manuellement ou à l'aide de méthodes de sélection de fonctionnalités. Par contre, l'apprentissage profond est utilisé en couches pour créer un réseau neuronal artificiel qui peut apprendre et extraire les fonctionnalités automatiquement, et prendre des décisions intelligentes par lui-même, ce qui améliore la précision et les performances. Le deep-learning offre actuellement les meilleures solutions à de nombreux problèmes dans les domaines du traitement du langage naturel:

A. Modèles traditionnels

La figure 36 suivante représente l'architecture du modèle :

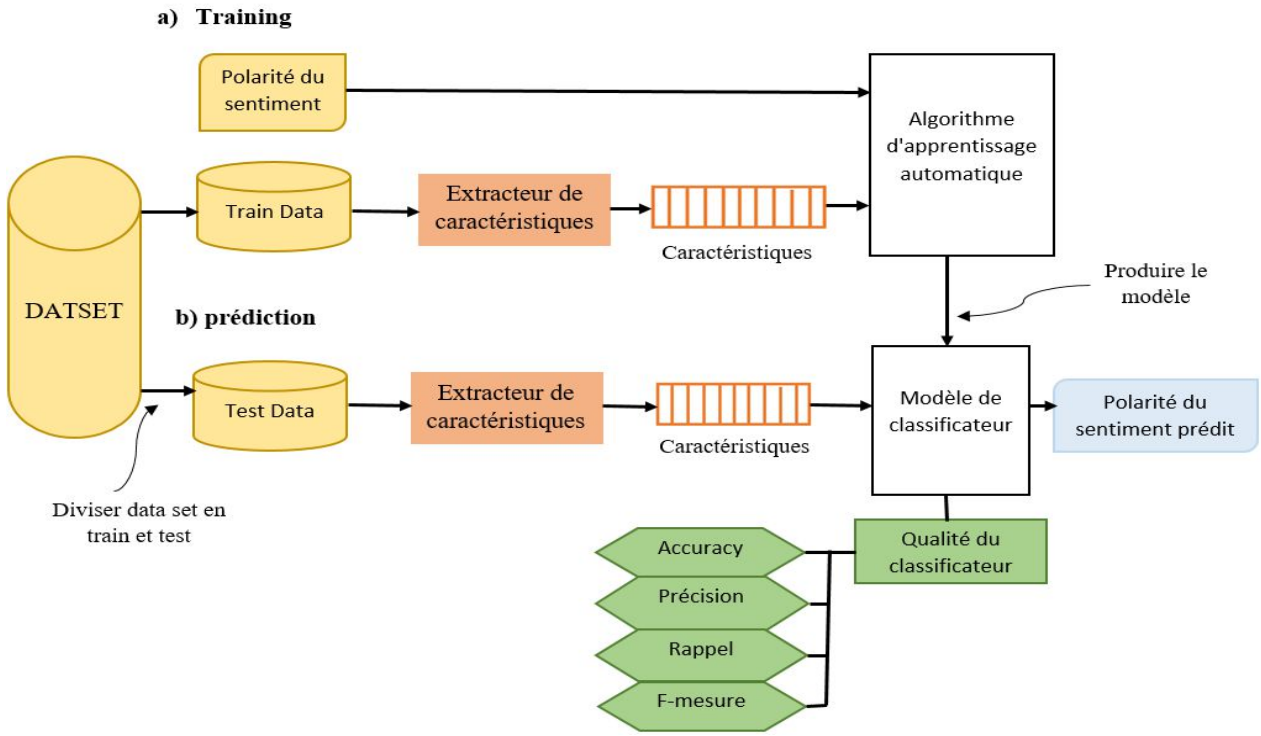


Figure 36 Schéma de principe du processus des modèles traditionnels

a) Pendant la formation, un extracteur de caractéristiques est utilisé pour convertir chaque valeur d'entrée (Train data) en un ensemble de caractéristiques. Ces ensembles de caractéristiques capturent les informations de base sur chaque entrée qui doivent être utilisées pour la classer. Des paires d'ensembles de caractéristiques et de polarité des sentiments sont introduites dans l'algorithme d'apprentissage automatique pour générer un modèle.

b) Pendant la prédiction, le même extracteur de caractéristiques est utilisé pour convertir les entrées (Test data) en ensembles de caractéristiques. Ces ensembles de caractéristiques sont ensuite introduits dans le modèle, qui génère une polarité des sentiments prédits, Enfin, la qualité du classificateur est évaluée.

A.1 Diviser les données

Nous avons choisi de diviser les données en deux morceaux: train et test

- Training: L'échantillon de données utilisé pour l'apprentissage.
- Ensemble de test: échantillon de données utilisé uniquement pour évaluer performance d'un modèle final.

On utilisant la fonction *train_test_split*, Quelques variables importantes dedans :

- *test_size*: la quantité de données à tester, si donné 0.5, cela prend 50% des données.
- *train_size*: identique à la définition de test_size mais pour les données d'apprentissage, et n'a pas besoin d'être défini et de donner une valeur si la taille des données à tester est définie.
- *random_state*: si random_state n'est pas spécifié dans le code, chaque fois que nous exécutons le code, une nouvelle valeur aléatoire est générée et les ensembles de données Train et Test auront des valeurs différentes à chaque fois.Cependant, si une valeur fixe est définie comme random_state = 1, quel que soit le nombre de fois que nous exécutons le code, le résultat sera le même, c'est-à-dire les mêmes valeurs dans les ensembles de données d'entraînement et de test.

diviser les données :
X_train, X_test, y_train, y_test = train_test_split(df['text'], df['sentiment'], test_size=0.2, random_state=1)

A.2 Extraction de caractéristiques

Après le prétraitement, l'ensemble de données sera toujours constitué d'échantillons de données texte, qui sont des données non structurées qui ne peuvent pas être transmises au classeur dans sa forme initiale. Ainsi, cette étape comprend la conversion des données textuelles en un espace de caractéristiques organisé, de sorte que nous nous appuyons sur la méthode d'extraction de caractéristiques(Feature extraction en anglais)sur le comptage à l'aide du vecteur de comptage.

Cette section est basée sur l'implémentation dans le package *Scikit-Learn*populaire que nous utilisons. Un vecteur de comptage (*count vectorizer* en anglais) est un sac de mots (*bag of words*), c'est-à-dire qu'il ignore la grammaire ou l'ordre des mots, mais maintient un nombre de mots unique. Pour nos besoins, l'ensemble de données est d'abord divisé (comme mentionné précédemment). Le vectoriseur de comptage est ensuite ajusté sur les échantillons de l'ensemble d'apprentissage.Cela signifie que chaque mot unique dans tout le vocabulaire de l'ensemble d'apprentissage reçoit un identifiant entier unique. Une fois que le vectoriseur de comptage a été ajusté, l'utilisation de la fonction de transformation avec n'importe quelle entrée de texte renvoie une sortie codée.L'exemple suivant devrait suffire à des fins de clarté:

1. L'ensemble d'entraînement se compose d'un document, avec le texte «the quick brown fox jumps over the lazy dog». Une instance du vectoriseur de comptage est adaptée au document.
2. Le vectoriseur a ainsi appris l'encodage suivant, chaque mot unique étant attribué un identifiant entier, dans l'ordre alphabétique: {'brown': 0, 'dog': 1, 'fox': 2, 'jump': 3, 'lazy ': 4,' over ': 5,' quick ': 6,' the ': 7}.
3. Un exemple de texte est transformé à l'aide du vectoriseur ajusté. Le texte est «the quick dog jumps the sheep». Il en résulte le vecteur codé [0 1 0 1 0 0 1 2], chaque indice vectoriel correspondant au codage donné ci-dessus. Ainsi, l'entier au premier index du vecteur, «0», correspond au nombre de fois où le premier mot du vocabulaire, «brown», apparaît dans l'échantillon en cours de transformation. Le mot «sheep», qui ne fait pas partie du vocabulaire sur lequel le vectoriseur a été installé, n'est pas pris en compte dans le vecteur renvoyé.

Lorsque le vectoriseur est entraîné sur un grand ensemble d'apprentissage, il aura un vocabulaire étendu. Chaque échantillon transformé sera donc constitué d'une majorité de zéros représentant chacun des mots qui apparaissent dans le vocabulaire de l'ensemble d'apprentissage mais pas dans l'échantillon. Dans la bibliothèque Scikit-learn, chaque échantillon transformé par le vectoriseur de comptage est ainsi renvoyé sous la forme d'une matrice creuse pour une efficacité de calcul, avec des comptages correspondant à chaque index de mot non nul.

Le vectoriseur de comptage est donc un vectoriseur efficace basé sur le modèle du sac de mots, et la représentation qu'il renvoie prend en compte la fréquence d'apparition des mots dans un jeu de données. Cependant, il présente diverses limites. Il ne prend pas en compte le contexte ou l'ordre dans lequel les mots apparaissent. Il prend également en compte le nombre de chaque mot unique apparaissant séparément. Enfin, il ne prend pas non plus en compte les similitudes entre les mots.

Convertir les données :
vectorizer = CountVectorizer(stop_words='english') train_features = vectorizer.fit_transform(X_train) test_features = vectorizer.transform(X_test).toarray()

A.3 Les classificateurs

Les algorithmes peuvent être perçus comme le cœur des systèmes à apprentissage automatique. Sans ces ensembles d'instructions, les systèmes ne pourraient pas apprendre seuls. En ce qui concerne l'apprentissage automatique supervisé, plusieurs algorithmes peuvent être mis en pratique et divisés en deux groupes. D'un côté se trouvent les algorithmes basés sur la logique comme les arbres à décisions, de l'autre se trouvent les algorithmes statistiques comme les réseaux neuronaux et les SVM.

Pour cela, nous avons construit un classificateur qui se compose de divers classificateurs d'apprentissage supervisé et qui classent les tweets en classes binaires positives et négatives. La bibliothèque Python a été utilisée pour créer un classificateur, scikit-learn.

- *Scikit-learn* est une bibliothèque qui est utilisée pour effectuer un apprentissage automatique en python. Par conséquent, il dispose de nombreux outils de classification, de visualisation, de régression, de clustering, etc. Grâce à une simple commande en python, nous pouvons installer scikit-learn dans notre système tel que 'pipinstallscikit-learn'. Il existe plusieurs classificateurs relevant du scikit-learn. Pour cette étape, nous allons implémenter les algorithmes d'analyse de sentiments et principalement implémenter les algorithmes classiques supervisés. Ces algorithmes sont : Support Vector Machines (SVM), DecisionTree (DT), Random Forest (RF), K-plus proche voisin (k-NN) et Naïve Bayes (NB):

Cependant, nous avons utilisé l'algorithme général de l'analyse suivant :

Algorithme
    Importer les bibliothèques
    Lire le Dataset
    Lire le dictionnaire
Début
    Extraire les fonctionnalités
    Préparation
    Appeler aux classificateurs
    Afficher les résultats
Fin

- **Naïve bayes(NB)**

```
#nbTrain
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, roc_auc_score
start_timenb = time.time()

train_features = vectorizer.fit_transform(X_train)
actual = y_test
nb = MultinomialNB()
nb.fit(train_features, y_train)
test_features = vectorizer.transform(X_test).toarray()
predictions = nb.predict(test_features)
```

- **Arbre de décision (DT)**

```
#datree
from sklearn import tree
start_timedt = time.time()

train_featurestree = vectorizer.fit_transform(X_train)
actual1 = y_test
test_features1 = vectorizer.transform(X_test).toarray()
dtree = tree.DecisionTreeClassifier()
dtree = dtree.fit(train_featurestree, y_train)
prediction1 = dtree.predict(test_features1)
```

- **Support Victor Machine(SVM)**

```
#Tsvm
from sklearn.svm import SVC
start_timesvm = time.time()

train_featuressvm = vectorizer.fit_transform(X_train)
actual2 = y_test
test_features2 = vectorizer.transform(X_test).toarray()
svc = SVC()
svc = svc.fit(train_featuressvm, y_train)
prediction2 = svc.predict(test_features2)
```

- **K-plus proche voisin (k-NN)**

```
#knN
from sklearn.neighbors import KNeighborsClassifier
start_timekn = time.time()

train_featureskn = vectorizer.fit_transform(X_train)
actual3 = y_test
test_features3 = vectorizer.transform(X_test).toarray()
kn = KNeighborsClassifier(n_neighbors=1)
kn = kn.fit(train_featureskn, y_train)
prediction3 = kn.predict(test_features3)
```

- **forêt aléatoire(RF)**

```
#RanFo
from sklearn.ensemble import RandomForestClassifier
start_timerf = time.time()

train_featuresrf = vectorizer.fit_transform(X_train)
actual4 = y_test
test_features4 = vectorizer.transform(X_test).toarray()
rf = RandomForestClassifier(max_depth=2, random_state=0)
rf = rf.fit(train_featuresrf, y_train)
prediction4 = rf.predict(test_features4)
```

B.Modèle d'apprentissage en profondeur

Nous avons utilisé le modèle de réseau neuronal récurrent. Les RNN sont une sorte de réseau neuronal bien connu pour bien fonctionner sur des données séquentielles, comme le cas des données textuelles. Dans ce cas, il a été implémenté un type spécial de RNN qui est LSTM (Long-Short Term Memory). Les LSTM sont l'une des versions améliorées des RNN, essentiellement les LSTM ont montré de meilleures performances en travaillant avec des phrases plus longues.L'architecture du modèle est généralement constituée des composants suivants (figure 37) :

L'idée de ce classificateur est basée sur la tokenisation, puis chaque jeton sera transformé en sa représentation basée sur un index.

Ensuite, Pour apprendre les relations contextuelles entre les mots dans les données d'apprentissage, nous utilisons le modèle d'incorporation de mots. Nous avons utilisé l'une des

couches du module *keras* qui est appelée couche embedding à cet effet. L'incorporation de mots se fait à l'aide de cette couche.

Chaque jeton est transmis séquentiellement à la couche embedding, cette couche produira une représentation intégrée de chaque jeton qui est passé à travers un réseau neuronal LSTM à deux empilements (stacked LSTM). Nous avons appliqué aussi une couche Dropout pour la régularisation.

Enfin, nous ajoutons une couche dense avec softmax comme fonction d'activation puisque chaque phrase ne peut appartenir qu'à une seule classe dans notre scénario. Le nombre de nœuds dans la couche dense est égal à 2 c'est la classe positive et négative dans notre problème.

Le modèle est entraîné avec l'ensemble de données d'entraînement. Et il est complètement construit à ce stade. Le modèle de classification des sentiments formé est testé avec l'ensemble de données de test et l'évaluation du modèle est notée.

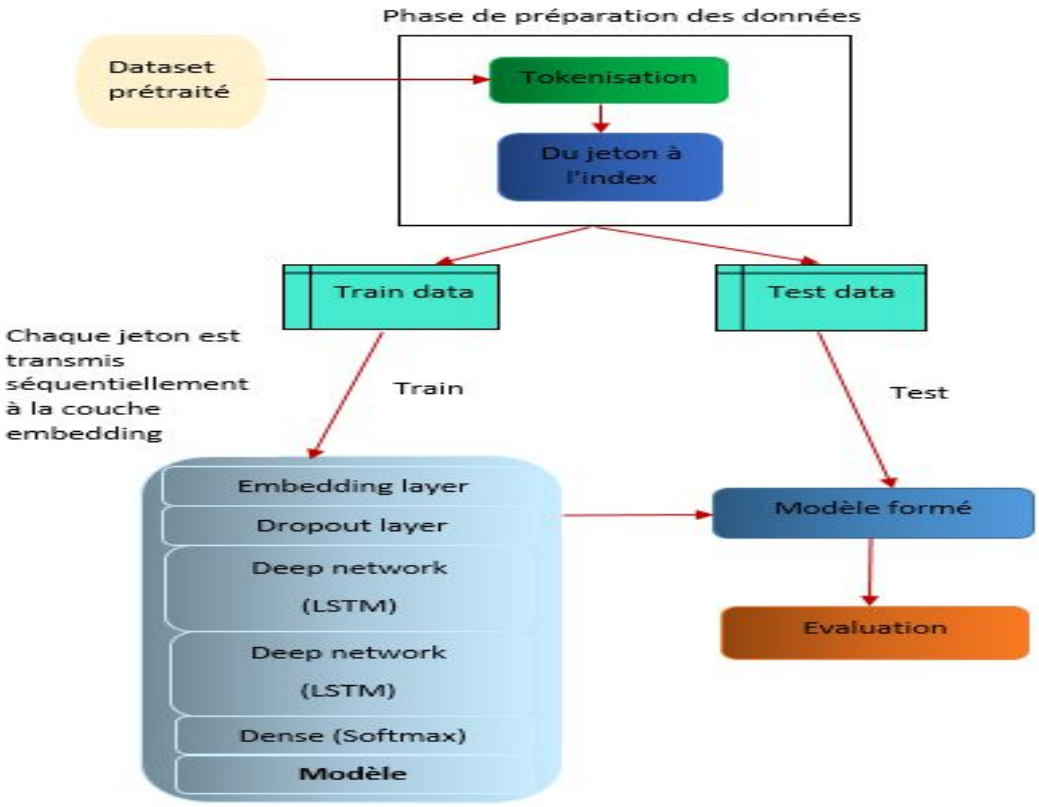


Figure 37 Architecture du modèle d'apprentissage en profondeur (RNN)

B.1 Préparation des données

Nous avons appliqué la technique de tokenisationce qui signifie que chaque texte sera tokenisé, et transformé chaque jeton en sa représentation basée sur un index. Ce processus est

expliqué dans l'extrait de code suivant:Il y a quelques hyper paramètres fixes qu'il vaut la peine de mentionner :

```
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences
tokenizer = Tokenizer()
tokenizer.fit_on_texts(df['text'].values)

max_len = max([len(s.split()) for s in df['text']])
vocab_size = len(tokenizer.word_index) + 1

X = tokenizer.texts_to_sequences(df['text'].values)
X = pad_sequences(X, maxlen=max_len) # padding our text vector so they all have the same length
```

- *vocab_size* : est la taille du vocabulaire dans les données de texte.
- *max_len* : Il s'agit de la longueur des séquences d'entrée.
- *La fonction fit_on_texts* : Met à jour le vocabulaire interne basé sur une liste de textes. Cette méthode crée l'index de vocabulaire basé sur la fréquence des mots. Donc, si vous lui donnez quelque chose comme, " I feeldepressed" Il va créer un dictionnaire à la place. word_index ["I"] = 1; word_index ["feel"] = 2. Donc chaque mot obtient une valeur entière unique.
- *La fonction text_to_sequences* : Transforme chaque texte des textes en une séquence d'entiers. Donc, il prend fondamentalement chaque mot dans le texte et le remplace par sa valeur entière correspondante du dictionnaire word_index.
- *La fonction pad_sequences* : qui complète les séquences qui ne correspondent pas à la longueur de séquence requise par des zéros.

B.2 La couche embedding

Un mot embedding est une représentation apprise pour le texte où les mots qui ont le même sens ont une représentation similaire. En d'autres termes, il représente des mots dans un système de coordonnées où les mots liés, basés sur un corpus de relations, sont rapprochés.

Dans les Framework d'apprentissage en profondeur tels que *TensorFlow*, *Keras,* cette partie peut être apprise dans le cadre de la formation d'un réseau neuronal. Une couche embedding est ajoutée au modèle, qui stocke une table de recherche pour mapper les mots représentés par des index numériques à leurs représentations vectorielles denses.

La couche embedding est définie comme la première couche masquée d'un réseau. Il doit spécifier 3 arguments :

- *input_dim*: la taille du vocabulaire.

- *output_dim*: C'est la taille del'espace vectoriel de 256 dimensions dans lequel les mots seront incorporés. Il définit la taille des vecteurs de sortie de cette couche pour chaque mot.
- *input_length*: la longueur des séquences d'entrée.

```
model.add(Embedding(vocab_size, 256, input_length=max_len))
```

B.3 La couche dropout

Le décrochage est ajouté pour surmonter la tendance au sur-ajustement, un problème très courant avec les réseaux basés sur RNN.

B.4 Réseau profond

Prend la séquence de embedding de vecteurs comme entrée et les convertit en une représentation compressée. La représentation compressée capture efficacement toutes les informations dans la séquence de mots dans le texte. La partie network profonde est un LSTM empilé (Stacked LSTM).

B.5 La couche entièrement connectée

La couche entièrement connectée prend la représentation profonde du LSTM et la convertit en classes de sortie finales égales à 2 et fournit par la fonction d'activation softmax.

B.6 Configuration du modèle

Nous sommes maintenant prêts à définir notre modèle de réseau neuronal (tableau 2):

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, 28, 256)           3704320
_________________________________________________________________
dropout_1 (Dropout)          (None, 28, 256)           0
_________________________________________________________________
lstm_1 (LSTM)                (None, 28, 256)           525312
_________________________________________________________________
lstm_2 (LSTM)                (None, 256)               525312
_________________________________________________________________
dense_1 (Dense)              (None, 2)                 514
=================================================================
Total params: 4,755,458
Trainable params: 4,755,458
Non-trainable params: 0
_________________________________________________________________
```

Tableau 2 Configuration du modèle

- Le nombre de paramètres d'embedding :
- Param = vocabulaire* dimension

  Param = 14470 * 256 =3704320

- Le tweet passe d'abord à la couche Lstm.

  Le nombre de paramètre =4*((size input + 1)* size output +$sizeoutput$^2).

  Param =4((256 +1)* 256 +256^2) = 525312.
- La dernière couche utilisée la fonction d'activation un softmax :

  Le nombre de paramètre = (input * output) + biais. Param = (256*2) + 2=514.
- Totale paramètre =$\sum Param.$

  Total param =3704320+0 +525312 +525 312+514 = 4 755 458.
- Trainableparameter =embedding+lstm+lstm + dense

  Trainableparam =3704320+525 312 +525 312+514 = 4 755 458.
- Non_ Trainable paramètre = totale – trainable.

  Non_ Trainable = 4 755 458- 4 755 458=0.

II.3 Conclusion

Dans ce chapitre, nous nous sommes principalement concentrés sur le processus général de la méthodologie de notre système d'analyse de la santé mental des internautes en expliquant le rôle de chaque sous-processus.

Nous avons utilisé les techniques de prétraitement les plus communes pour réduire le bruit dans le texte et avoir une meilleure et plus efficace analyse. Après avoir prétraiter les tweets, l'étape qui suit est la classification des textes de tweets en fonction du sentiment exprimé: positif, négatif ou neutre

Le chapitre suivant va porter sur les outils et les bibliothèques utilisés pour l'implémentation etla mise en œuvre de notre système.

IMPLÉMENTATION ET MISE EN ŒUVRE

CHAPITRE III : IMPLÉMENTATION ET MISE EN ŒUVRE

III.1 Introduction

Ce chapitre comprend l'analyse des sentiments des tweets recueillis à partir de Twitter, avec divers analyseurs de sentiments. De plus, il présente la visualisationet la validation des résultats d'évaluation obtenus à partir de chaque classificateur d'apprentissage automatique(traditionnel et deep learning). Les expériences sur les données de Twitter montreront quelle technique a la meilleure capacité de mesurer la précision de la prédiction de sentiments.

Cependant, notre outil s'intègre dans le domaine d'intelligence artificielle précisément "Machine Learning" car la précision de la classification augmente à chaque fois quand on exécute l'algorithme de classification.

III.2 Outils utilisés

Pour l'implémentation de l'application, nous avons utilisé :

III.2.1 Environnement Python

 Nous avons utilisé le langage de programmation Python la version 3.7. Python est un langage de programmation de haut niveau interprété (il n'y a pas d'étape de compilation) et orienté objet avec une sémantique dynamique. Il est très sollicité par une large communauté de développeurs et de programmeurs. Python est un langage simple, facile à apprendre et permet une bonne réduction du cout de la maintenance des codes. Les bibliothèques (packages) python encouragent la modularité et la réutilisabilité des codes. Python et ses bibliothèques sont disponibles (en source ou en binaires) sans charges pour la majorité des plateformes et peuvent être redistribués gratuitement. [14]

III.2.1 Anaconda

Anacondaest une distribution libre et open source des langages de programmation Python (et R) appliqué au développement d'applications dédiées à la science des données et à l'apprentissage automatique (traitement de données à grande échelle, analyse prédictive, calcul scientifique), qui vise à simplifier la gestion des paquets et de déploiement. Les

versions de paquetages sont gérées par le système de gestion de paquets conda.La distribution propose deux moyens d'accéder à ses fonctions. Soit de manière graphique avec Anaconda-Navigator, soit en lignes de commande [32]. La distribution Anaconda est utilisée par plus de 6 millions d'utilisateurs et comprend plus de 250 paquets populaires en science des données adaptés pour Windows, Linux et MacOs.

III.2.2 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 du texte narratif. Les utilisations incluent le nettoyage et la transformation des données, la simulation numérique, la modélisation statistique, la visualisation de données, l'apprentissage automatique et bien plus encore. [16]

III.2.3 Bibliothèque Nltk

Natural LanguageToolKit (NLTK) est une plate-forme leader pour la construction de programmes Python pour travailler avec des données de langage humain. Il fournit des interfaces faciles à utiliser pour plus de 50 ressources corporelles et lexicales telles que WordNet. Ainsi qu'une suite de bibliothèques de traitement de texte pour la classification, tokenization, stemming, étiquetage, analyse et raisonnement sémantique, wrappers pour les bibliothèques NLP " Natural LanguageProcessing" de puissance industrielle, et un forum de discussion actif.[10]

III.2.4 Package re (Regular expressions)

Ce module fournit des opérations correspondant aux expressions régulières. Les expressions régulières utilisent le caractère barre oblique inverse ('\') pour indiquer des formes spéciales ou pour permettre l'utilisation de caractères spéciaux sans invoquer leur signification particulière. [10]

III.2.5 Scikit-learn

Scikit-learn estune bibliothèque libre Python destinée à l'apprentissage automatique. Elle est développée par de nombreux contributeurs notamment dans le monde académique par des

instituts français d'enseignement supérieur et de recherche comme Inria. Elle comprend notamment des fonctions pour estimer des forêts aléatoires, des régressions logistiques, des algorithmes de classification, et les machines à vecteurs de support. Elle est conçue pour s'harmoniser avec d'autres bibliothèques libres Python, notamment NumPy et SciPy [33].Scikit-learn est livré avec de nombreuses fonctionnalités :

- *Algorithmes d'apprentissage supervisé*: Pensez à tout algorithme d'apprentissage automatique supervisé dont vous pourriez avoir entendu parler et il y a de très fortes chances qu'il fasse partie de scikit-learn. À partir de modèles linéaires généralisés (par exemple, régression linéaire), de machines vectorielles de soutien (SVM), d'arbres de décision et de méthodes bayésiennes tous font partie de la boîte à outils scikit-learn. La diffusion des algorithmes d'apprentissage automatique est l'une des principales raisons de la forte utilisation de scikit-learn. J'ai commencé à utiliser scikit pour résoudre des problèmes d'apprentissage supervisé et je le recommanderais également aux nouveaux utilisateurs de scikit / machine learning.
- *Validation croisée*: Il existe différentes méthodes pour vérifier l'exactitude des modèles supervisés sur des données invisibles à l'aide de sklearn.
- *Algorithmes d'apprentissage non supervisé*: encore une fois, il existe une large gamme d'algorithmes d'apprentissage automatique dans l'offre - du clustering, de l'analyse factorielle, de l'analyse des composants principaux aux réseaux de neurones non supervisés.
- *Divers jeux de données sur les jouets*: cela s'est avéré utile lors de l'apprentissage de *scikit-learn*.
- *Extraction de fonctionnalités*: *Scikit-learn* pour extraire des fonctionnalités d'images et de texte (par exemple, un sac de mots) [34]

III.2.6 TensorFlow

TensorFlow est un Framework de programmation pour le calcul numérique qui a été rendu Open Source par Google en Novembre 2015. Depuis son release, TensorFlow n'a cessé de gagner en popularité, pour devenir très rapidement l'un des Framework les plus utilisés pour le Deep Learning et donc les réseaux de neurones. Son nom est notamment inspiré du fait que les opérations courantes sur des réseaux de neurones sont principalement faites via des tables de données

multidimensionnelles, appelées Tenseurs (Tensor). Un Tensor à deux dimensions est l'équivalent d'une matrice. Aujourd'hui, les principaux produits de Google sont basés sur TensorFlow: Gmail, Google Photos, Reconnaissance de voix. [14]

III.2.7 Keras

Keras est une API de réseaux de neurones de haut niveau, écrite en Python et capable de fonctionner sur TensorFlow ou Theano. Il a été développé en mettant l'accent sur l'expérimentation rapide. Être capable d'aller de l'idée à un résultat avec le moins de délai possible est la clé pour bonnes recherches. Il a été développé dans le cadre de l'effort de recherche du projet ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System), et son principal auteur et mainteneur est François Chollet, un ingénieur Google.

En 2017, l'équipe TensorFlow de Google a décidé de soutenir Keras dans la bibliothèque principale de TensorFlow. Chollet a expliqué que Keras a été conçue comme une interface plutôt que comme un cadre d'apprentissage end to end. Il présente un ensemble d'abstractions de niveau supérieur et plus intuitif qui facilitent la configuration des réseaux neuronaux indépendamment de la bibliothèque informatique de backend. Microsoft travaille également à ajouter un backend CNTK à Keras aussi. [14]

III.2.8 Pandas

Pandas est une bibliothèque écrite pour le langage de programmation Python permettant la manipulation et l'analyse des données. Elle propose en particulier des structures de données et des opérations de manipulation de tableaux numériques et de séries temporelles. Pandas est un logiciel libre sous licence BSD.[35]

III.2.9 Numpy

Cette bibliothèque logicielle libre et open source fournit de multiples fonctions permettant notamment de créer directement un tableau depuis un fichier ou au contraire de sauvegarder un tableau dans un fichier, et manipuler des vecteurs, matrices et polynômes.

III.2.10 Matplotlib

Matplotlib est une bibliothèque du langage de programmation Python destinée à tracer et visualiser des données sous formes de graphiques. Elle peut être combinée avec les bibliothèques python decalcul scientifique NumPy et SciPy. Matplotlib est distribuée librement et gratuitement sous une licence de style BSD. [36]

III.2.11 Seaborn

C'est l'un des meilleurs packages de visualisation, quel que soit l'outil ou la langue. Seaborn me donne la même sensation générale.Cela nous donne la possibilité de créer des données visuelles amplifiées. Cela nous aide à comprendre les données en les affichant dans un contexte visuel pour découvrir les corrélations cachées entre les variables ou les tendances qui pourraient ne pas être évidentes au départ. Seaborn a une interface de haut niveau par rapport au faible niveau de Matplotlib. [37]

III.2.12 Flask

Flask est un framework open-source de développement web en Python. Son but principal est d'être léger, afin de garder la souplesse de la programmation Python. Flask est basé sur la boîte à outils Werkzeg WSGI et le moteur de modèles Jinja2 (template) [38]:

- WSGI : L'interface de passerelle de serveur Web a été utilisée comme norme pour le développement d'applications Web Python. WSGI est la spécification d'une interface commune entre les serveurs Web et les applications Web.
- Werkzeug : est une boîte à outils WSGI qui implémente des requêtes, des objets de réponse et des fonctions utilitaires. Cela permet de créer un cadre Web dessus. Le frameworkFlask utilise Werkzeg comme l'une de ses bases.
- jinja2 : est un moteur de modèle populaire pour Python. Un système de modèle Web combine un modèle avec une source de données spécifique pour rendre une page Web dynamique.

III.3 Dataset

La collection des données des tweets créées par la tâche partagée Linguistique computationnelle et psychologie clinique (CLPsych) 2015(mentionnés précédemment) se présente sous forme d'un fichier d'extension (.txt). La figure 38 suivante représente un fragment de cet ensemble de données :

{"created_at":"Fri Jun 02 00:04:00 +0000 2017","id":870430762255953920,"id_str":"870430762255953920","text":"Hey, look
"Central Time (US & Canada)","geo_enabled":true,"lang":"en","contributors_enabled":false,"is_translator":false,"profile
"place":null,"contributors":null,"is_quote_status":false,"retweet_count":0,"favorite_count":0,"entities":{"hashtags":[]

{"created_at":"Fri Jun 02 00:04:02 +0000 2017","id":870430770141253632,"id_str":"870430770141253632","text":"RT @shanno
one":"Pacific Time (US & Canada)","geo_enabled":true,"lang":"en","contributors_enabled":false,"is_translator":false,"pr
null,"coordinates":null,"place":null,"contributors":null,"retweeted_status":{"created_at":"Wed May 31 23:43:03 +0000 20
s_count":5980,"created_at":"Tue Jul 30 20:55:51 +0000 2013","utc_offset":null,"time_zone":null,"geo_enabled":true,"lang
o":null,"coordinates":null,"place":null,"contributors":null,"is_quote_status":false,"retweet_count":570,"favorite_count

Figure 38 Extrait du data set CLPsych 2015

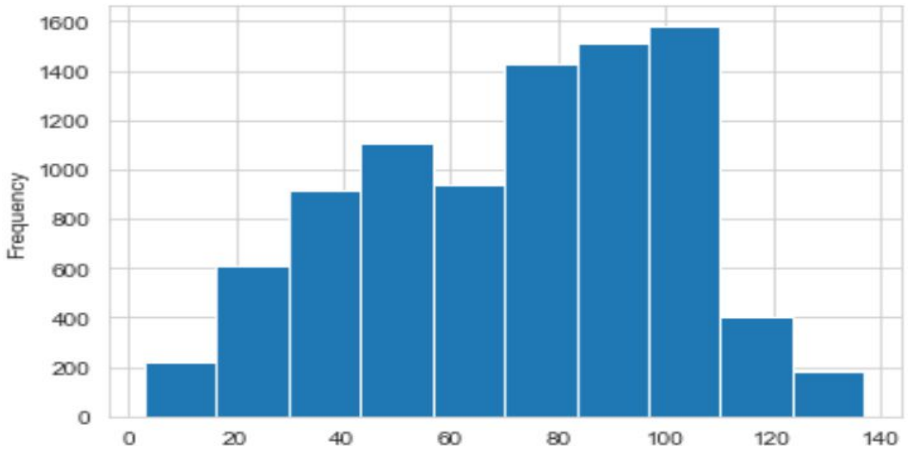


Figure 39 Représentation graphique du data set

Les données fournies sont reformatées à partir de la version originale en utilisant des méthodes de prétraitement. Pour l'approche sémantique, les tweets sont classés selon deux étiquettes positives (non déprimés) et négatives (déprimés). Par contre, pour l'approche lexico-syntaxique, il y a la présence de la polarité neutre. Chaque entrée de notre ensemble de données est structurée comme suit :

- Id : un identifiant du tweet.
- Texte original : contient le texte original du tweet publié par l'utilisateur.
- Texte : contient le texte traité des données.
- Sentiment : étiquette du tweet, qui peut être ("1 → positif", "0→ neutre","-1 → négatif").

Les tableaux suivants montrent le fichier (.xlsx) résultant des extraits des données stockées suite au prétraitement:

id	original tex	text	sentiment
870430762	Hey, look -	hey look i	1
870430770	RT @shani	rt shannon	1
870430771	RT @HRoy	rt hroyalth	-1
870430772	How to De	how deal s	-1
870430776	RT @COCC	rt coconut	-1
870430779	@sabbunn	sabbunny i	-1
870430780	RT @loopz	rt loopzoo	-1
870430781	Gossip Girl	gossip girl	-1
870430784	RT @lucas	rt lucasbav	-1
870430796	I'm glad I u	im glad i ur	-1
870430797	RT @huma	rt humans	-1
870430799	RT @RTFF	rt rtffacts	1
870430802	Anyone loo	anyone loo	-1
870430804	RT @oralt	rt oraltwjn	-1
870430804	UR PRECIC	ur precious	1
870430807	RT @petty	rt pettybla	-1
870430807	I am deligh	i delighted	1
870430814	It's time fo	its time ph	1
870430815	RT @bpHo	rt bphoper	-1

Tableau 3 Fichier .xlsx résultant des données stockées et de leurs sentiments(Approche-sémantique)

id	original tex	text	sentiment
870430762	Hey, look -	hey look i	-1
870430770	RT @shani	rt shannon	-1
870430771	RT @HRoy	rt hroyalth	-1
870430772	How to De	how deal s	-1
870430776	RT @COCC	rt coconut	0
870430779	@sabbunn	sabbunny i	-1
870430780	RT @loopz	rt loopzoo	-1
870430781	Gossip Girl	gossip girl	0
870430784	RT @lucas	rt lucasbav	0
870430796	I'm glad I u	im glad i ur	0
870430797	RT @huma	rt humans	0
870430799	RT @RTFF	rt rtffacts	-1
870430802	Anyone loo	anyone loo	0
870430804	RT @oralt	rt oraltwjn	-1
870430804	UR PRECIC	ur precious	1
870430807	RT @petty	rt pettybla	-1
870430807	I am deligh	i delighted	1
870430814	It's time fo	its time ph	1
870430815	RT @bpHo	rt bphoper	0

Tableau 4 Fichier .xlsx résultant des données stockées et de leurs sentiments (Approche lexico-syntaxique)

Cet ensemble de données contient 8956 tweets, répartis en 80% tweets de trainet 20% tweets de test. La répartition des données est illustrée dans les tableaux et les figures suivantes :

	Dataset	Train set	Test set	Positive	Negative
Shape	8898	(7118,)	(1780,)	4258	4640

Tableau 5 Description dudataset(approche semantique)

	Dataset	Train set	Test set	Positive	Negative	Neutral
Shape	(8956,)	(7164,)	(1792,)	1813	3946	3197

Tableau 6 Distribution du dataset(approche lexico-syntaxique)

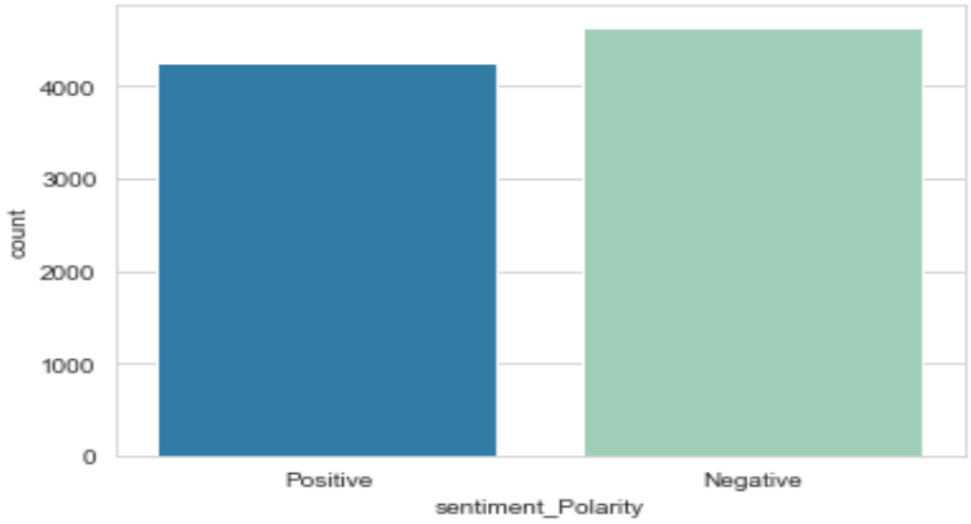


Figure 40 Image visualisant les données Positive/ Négative(approche sémantique)

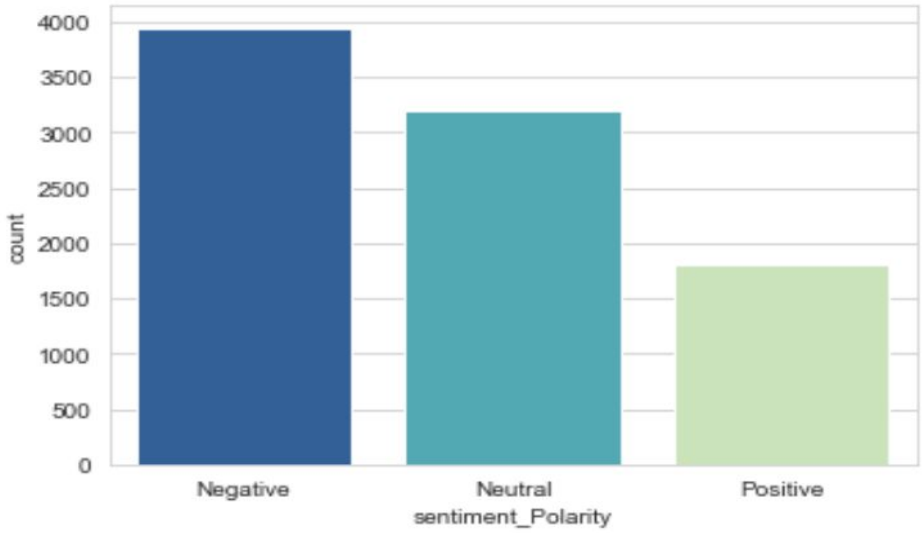


Figure 41 Image visualisant les données Positive/ Négative/Neutre (approche lexico-syntaxique)

III.4 Discussions des résultats obtenus

Chaque modèle a été appliqué sur le même jeu de données, l'évaluation de nos modèles s'est faite en terme de 6 métriques qui sont confusion *Matrix*, *l'accuracy*, la *précision*, le *rappel*, la *F-mesure*, *la courbe ROC et AUC (aire sous la courbe ROC):*

III.4.1 Résultats via les modèles traditionnels

III.4.1.1 Approche syntaxique (dictionnaire donné)

A. Résultats du modèle Naïve Bayes

Voici la matrice de confusion pour le classificateur Naïve Bayes :

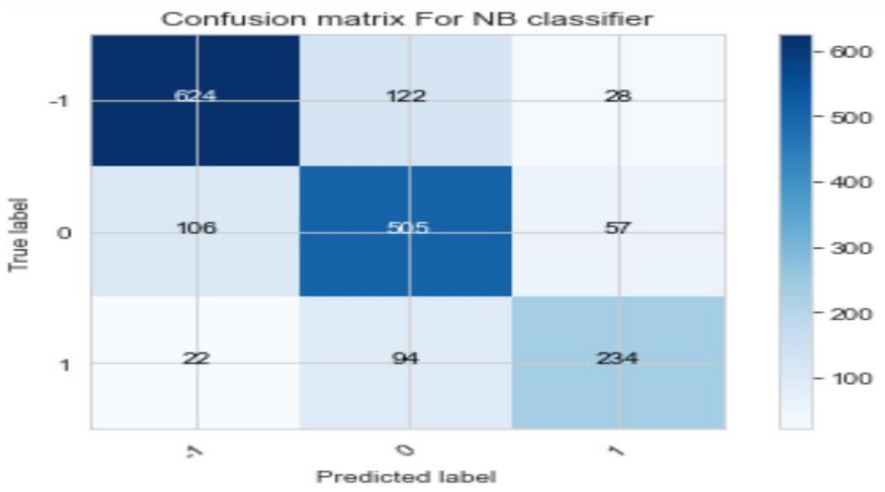


Figure 42 Matrice de confusion pour le classificateur Naïve Bayes

À partir de la matrice de confusion, nous pouvons voir que:

- Vrais positifs :
  - TP (-1) = 624.
  - TP (0) = 505.
  - TP (1) = 234.
- Vrais négatifs :
  - TN (-1) = 505+235+57+94 = 891.
  - TN (0) = 624+234+28+22 = 908.
  - TN (1) = 624+505+106+122 = 1357.
- Faux positifs :
  - FP (-1) = 106+22 = 128.
  - FP (0) = 122+94 = 216.
  - FP (1) = 28+57 = 85.
- Faux négatifs :
  - FN (-1) = 122+28 = 150.

- FN (0) = 106+57 = 163.
- FN (1) = 94+22 = 116.

Voici les résultats de performance du classificateur naïve bayes :

	Accuracy	Precision	Recall	F-measures
Naive Bayes	76.060268	75.458188	74.3587	74.817259

Figure 43 Résultats de performance du classificateur naïve bayes

La performance du classificateur naïve bayes sur notre jeu de données est de 75% pourla précision, et il donne 74% en termes de rappel et de mesure f,la meilleure performanceest l'accuracy à 76%.

B. Résultats du modèle arbres de décision

Voici la matrice de confusion pour le classificateur d'arbres de décision :

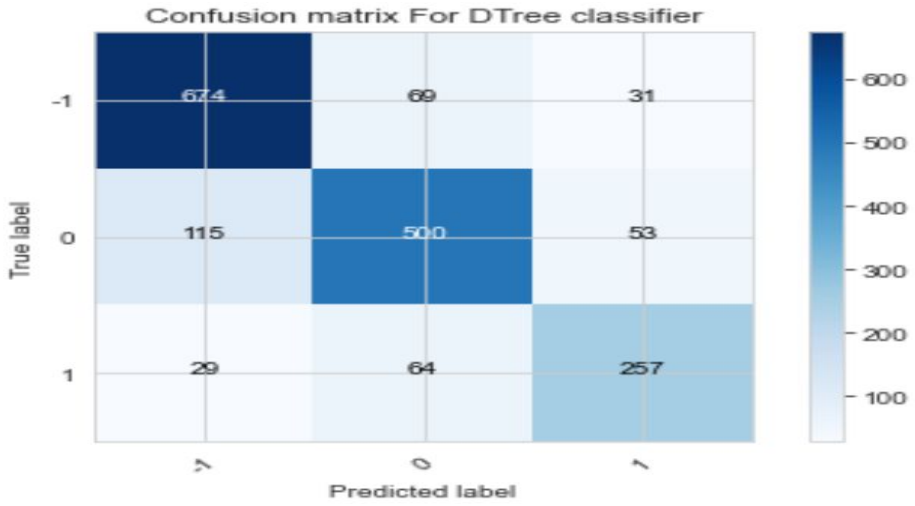


Figure 44 Matrice de confusion pour le classificateur d'arbres de décision

À partir de la matrice de confusion, nous pouvons voir que:

- Vrais positifs :
  - TP (-1) = 674.
  - TP (0) = 500.
  - TP (1) = 257.
- Vrais négatifs :
  - TN (-1) = 500+257+53+64= 874.
  - TN (0) = 674+257+29+31 = 991.

- TN (1) = 500+115+29+64 = 708.

- Faux positifs :
  - FP (-1) = 115+29 = 144.
  - FP (0) = 69+64 = 133.
  - FP (1) = 53+31 = 84.
- Faux négatifs :
  - FN (-1) = 69+31 =100.
  - FN (0) = 115+53 =168.
  - FN (1) = 29+64 = 93.

Voici les résultats de performance du classificateur d'arbres de décision:

	Accuracy	Precision	Recall	F-measures
Decision tree	79.854911	78.917199	78.452991	78.640756

Figure 45 Résultats de performance du classificateur d'arbres de décision

La performance du classificateur d'arbre de décision dans notre ensemble de données est de 78% pour le rappel,la mesure f et la précision, la meilleure performance estl'accuracyà 79%.

C. Résultats dumodèle Support Vector Machine

Voici la matrice de confusion pour le classificateur Support Vector Machine :

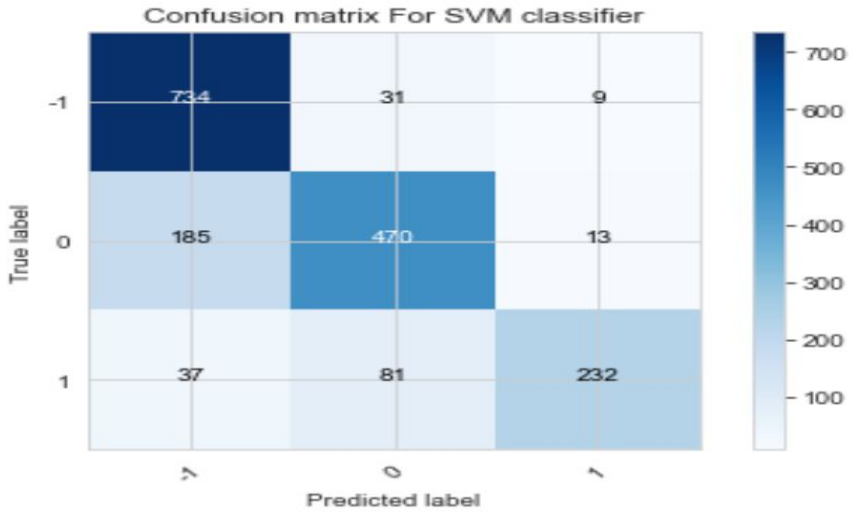


Figure 46 Matrice de confusion pour le classificateur Support Vector Machine

À partir de la matrice de confusion, nous pouvons voir que:

- Vrais positifs :
  - TP (-1) = 734.
  - TP (0) = 470.
  - TP (1) = 232.
- Vrais négatifs :
  - TN (-1) = 470+232+13+81 = 796.
  - TN (0) = 734+9+37+232 = 1012.
  - TN (1) = 734+470+185+31 = 1420.
- Faux positifs :
  - FP (-1) = 185+37 = 222.
  - FP (0) = 31+81 = 112.
  - FP (1) = 9+13 = 22.
- Faux négatifs :
  - FN (-1) = 31+9 = 40.
  - FN (0) = 185+13 = 198.
  - FN (1) = 37+81 = 118.

Voici les résultats de performance du classificateur Support Vector Machine :

	Accuracy	Precision	Recall	F-measures
Support vector machine	80.133929	82.957613	77.159012	78.958894

Figure 47 Résultats de performance du classificateur Support Vector Machine

La performance du classificateur Support Vector Machine sur notre jeu de données est de 80% pour l'accuracy. Il donne un rappelet une mesure f de 77% et 78% respectivement, etil donne de meilleures performances en termes de précision soit 82%.

D. Résultats du modèle k-plus proches-voisins

Voici la matrice de confusion pour le classificateur k-plus proches-voisins :

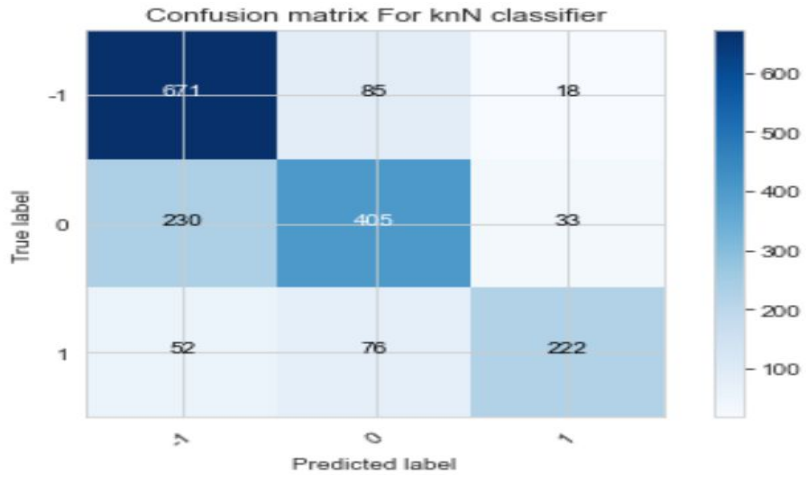


Figure 48 Matrice de confusion pour le classificateur k-plus proches-voisins

À partir de la matrice de confusion, nous pouvons voir que:

- Vrais positifs :
  - TP (-1) = 671.
  - TP (0) = 405.
  - TP (1) = 222.
- Vrais négatifs :
  - TN (-1) = 405+222+76+33 = 736.
  - TN (0) = 671+222+18+52 = 963.
  - TN (1) = 671+405+230+85 = 1391.
- Faux positifs :
  - FP (-1) = 230+52 = 282.
  - FP (0) = 76+85 = 161.
  - FP (1) = 33+18 = 51.
- Faux négatifs :
  - FN (-1) = 85+18 = 103.
  - FN (0) = 230+33 = 263.
  - FN (1) = 76+52 = 128.

Voici les résultats de performance du classificateur k-plus proches-voisins :

	Accuracy	Precision	Recall	F-measures
K neighbors classifier	72.433036	74.427562	70.24994	71.53842

Figure 49 Résultats de performance du classificateur k-plus proches-voisins

La performance du classificateur k-plus proches-voisins dans notre ensemble de données est de 72% pourl'accuracy, Il donne un rappel et une mesure f de 70% et 71% respectivement, lameilleure performance estla précision à 72%.

D. Résultats du modèle de forêt aléatoire

Voici la matrice de confusion pour le classificateur de forêt aléatoire :

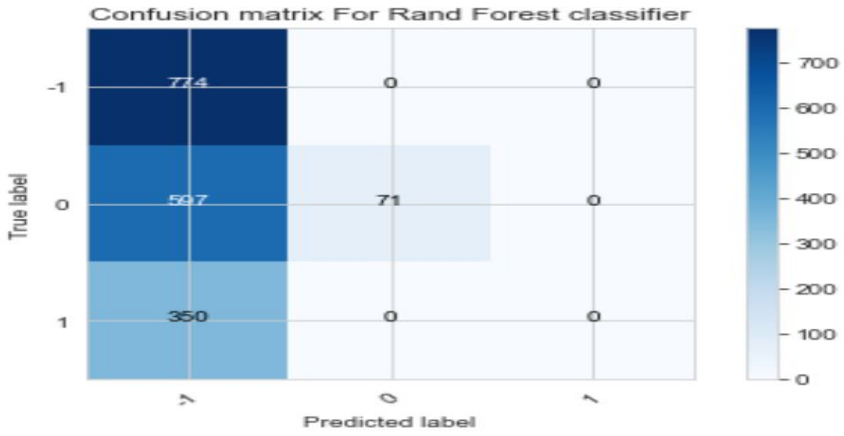


Figure 50 Matrice de confusion pour le classificateur de forêt aléatoire

À partir de la matrice de confusion, nous pouvons voir que:

- Vrais positifs :
  - TP (-1) = 774.
  - TP (0) = 71.
  - TP (1) = 0.
- Vrais négatifs :
  - TN (-1) = 71+0+0+0 = 71.
  - TN (0) = 774+350+0+0 = 1124.
  - TN (1) = 774+597+71+0 = 1442.

- Faux positifs :
  - FP (-1) = 597+350 = 947.
  - FP (0) = 0.
  - FP (1) = 0.
- Faux négatifs :
  - FN (-1) = 0.
  - FN (0) = 597.
  - FN (1) = 350.

Voici les résultats de performance du classificateur de forêt aléatoire :

	Accuracy	Precision	Recall	F-measures
Random Forest	47.154018	48.324617	36.876248	27.086415

Figure 51 Résultats de performance du classificateur de forêt aléatoire

La performance du classificateur de forêt aléatoiredans notre ensemble de données est de 47% pour l'accuracy. Et en termes du rappel et mesure f de 36% et 27% respectivement, la meilleure performance est laprécision de 48%.

III.4.1.2 Approche sémantique (SentiWordnet)

A. Résultats du modèle Naïve Bayes

Voici la matrice de confusion pour le classificateur Naïve Bayes :

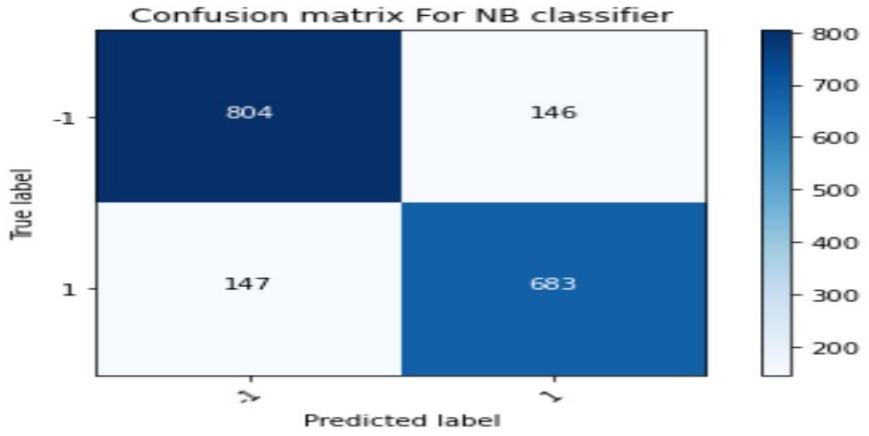


Figure 52 Matrice de confusion pour le classificateur Naïve Bayes

À partir de la matrice de confusion, nous pouvons voir que:

- 683 individus sont correctement identifiés comme étant non déprimés par le modèle (TP).
- 804 individus sont correctement identifiés comme déprimés par le modèle (TN).
- 146individus déprimés identifiés comme non déprimés par le modèle (FP).
- 147 individus non déprimés qui ont été identifiés comme déprimés par le modèle (FN).

Voici les résultats de performance du classificateur naïve bayes :

	Accuracy	Precision	Recall	F-measures
Naive Bayes	83.539326	82.38842	82.289157	82.338758

Figure 53 Résultats de performance du classificateur naïve bayes

En général, ce modèle fonctionne sur un grand ensemble de données et c'est rapide.La performance du classificateur naïve bayes sur notre jeu de données est de 83% pour l'accuracy,et en termes du rappel, la mesure f et la précision, il donne 82%.

B. Résultats du modèle arbres de décision

Voici la matrice de confusion pour le classificateur d'arbres de décision :

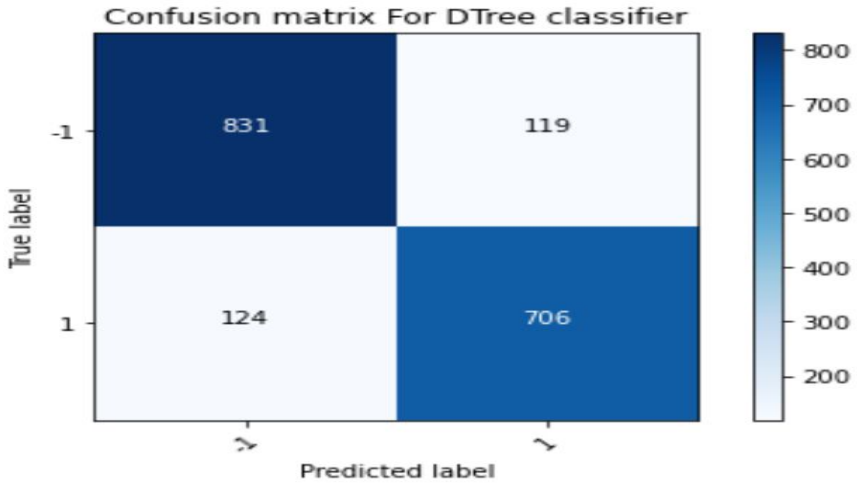


Figure 54 Matrice de confusion pour le classificateur d'arbres de décision

À partir de la matrice de confusion, nous pouvons voir que:

- 706 individus sont correctement identifiés comme étant non déprimés par le modèle (TP).
- 831 individus sont correctement identifiés comme déprimés par le modèle (TN).
- 119individus déprimés identifiés comme non déprimés par le modèle (FP).
- 124individus non déprimés qui ont été identifiés comme déprimés par le modèle (FN).

Voici les résultats de performance du classificateur d'arbres de décision :

	Accuracy	Precision	Recall	F-measures
Decision tree	86.348315	85.575758	85.060241	85.317221

Figure 55 Résultats de performance du classificateur d'arbres de décision

La performance du classificateur d'arbre de décision dans notre ensemble de données est de 85% en termes de la précision, le rappel et la mesure f, la meilleure performance estl'accuracyà 86%.

C. Résultats dumodèle Support Vector Machine

Voici la matrice de confusion pour le classificateur Support Vector Machine :

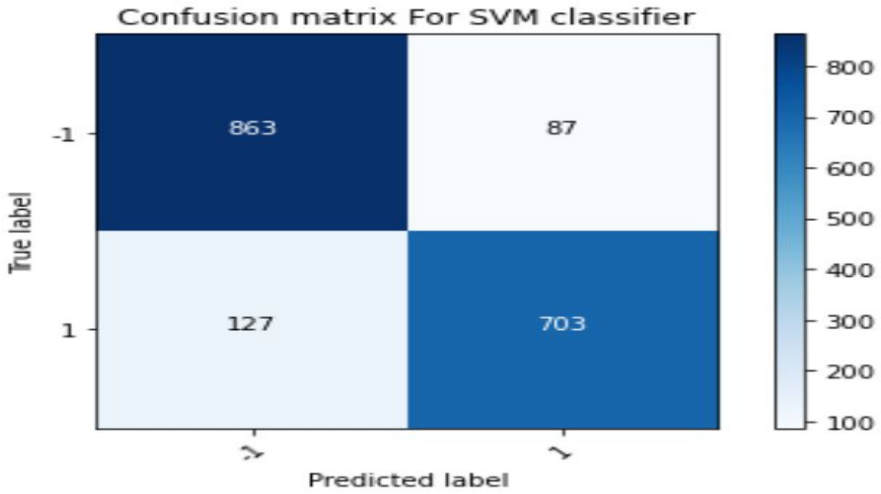


Figure 56 Matrice de confusion pour le classificateur Support Vector Machine

À partir de la matrice de confusion, nous pouvons voir que:

- 703 individus sont correctement identifiés comme étant non déprimés par le modèle (TP).
- 863 individus sont correctement identifiés comme déprimés par le modèle (TN).
- 87individus déprimés identifiés comme non déprimés par le modèle (FP).
- 127 individus non déprimés qui ont été identifiés comme déprimés par le modèle (FN).

Voici les résultats de performance du classificateur Support Vector Machine :

	Accuracy	Precision	Recall	F-measures
Support vector machine	87.977528	88.987342	84.698795	86.790123

Figure 57 Résultats de performance du classificateur Support Vector Machine

La performance du classificateur Support Vector Machine sur notre jeu de données est de 87% pour l'accuracy. Il donne un rappel de 84%, et une mesure f de 86% respectivement, et il donne de meilleures performances en termes de la précision à88%.

D. Résultats du modèle k-plus proches-voisins

Voici la matrice de confusion pour le classificateur k-plus proches-voisins :

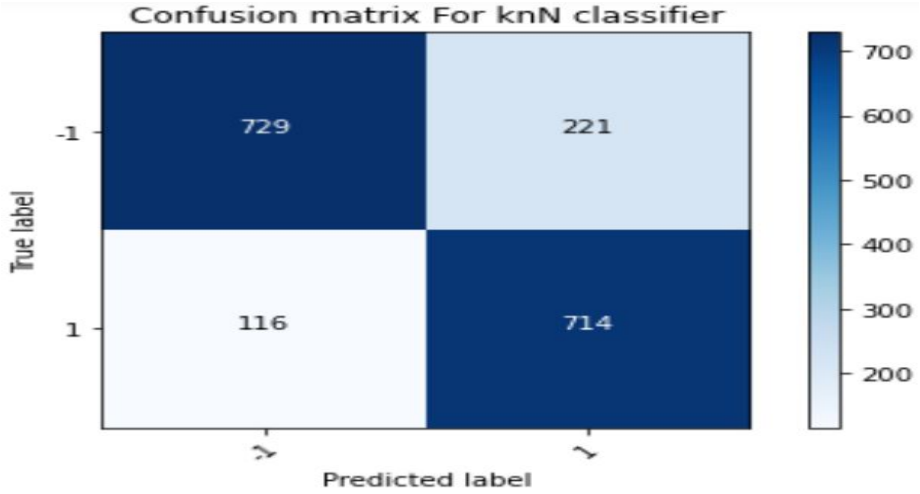


Figure 58 Matrice de confusion pour le classificateur k-plus proches-voisins

À partir de la matrice de confusion, nous pouvons voir que:

- 714 individus sont correctement identifiés comme étant non déprimés par le modèle (TP).
- 729 individus sont correctement identifiés comme déprimés par le modèle (TN).
- 221individus déprimés identifiés comme non déprimés par le modèle (FP).
- 116 individus non déprimés qui ont été identifiés comme déprimés par le modèle (FN).

Voici les résultats de performance du classificateur k-plus proches-voisins

	Accuracy	Precision	Recall	F-measures
K neighbors classifier	81.067416	76.363636	86.024096	80.906516

Figure 59 Résultats de performance du classificateur k-plus proches-voisins

La performance du classificateur k-plus proches-voisins dans notre ensemble de données est de 81% pour l'accuracy. Il donne une précision et une mesure f de 76% et 80% respectivement, lameilleure performance est lerappel à86%.

E. Résultats du modèle forêt aléatoire

Voici la matrice de confusion pour le classificateur de forêt aléatoire :

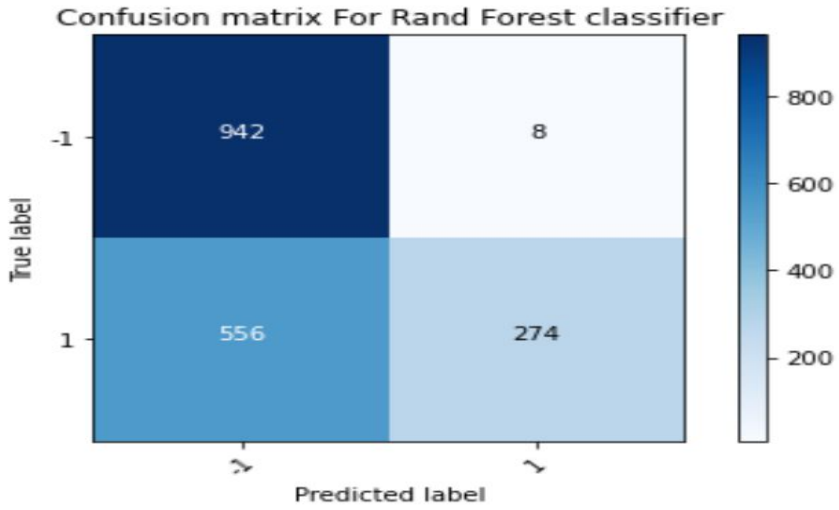


Figure 60 Matrice de confusion pour le classificateur de forêt aléatoire

À partir de la matrice de confusion, nous pouvons voir que:

- 274 individus sont correctement identifiés comme étant non déprimés par le modèle (TP).
- 942individus sont correctement identifiés comme déprimés par le modèle (TN).
- 8individus déprimés identifiés comme non déprimés par le modèle (FP).
- 556 individus non déprimés qui ont été identifiés comme déprimés par le modèle (FN).

Voici les résultats de performance du classificateur de forêt aléatoire :

	Accuracy	Precision	Recall	F-measures
Random Forest	68.314607	97.163121	33.012048	49.280576

Figure 61 Résultats de performance du classificateur de forêt aléatoire

La performance du classificateur de forêt aléatoire dans notre ensemble de données est de 68% pour l'accuracy. Il donne 33% et 49% pour la précision et la mesure f respectivement. La meilleure performance est le rappel à97%.

III.4.2 Résultats du modèle apprentissage en profondeur (RNN)

Les bibliothèques Keras et Tensorflow ont été utilisées pour tester notre modèle:

A. Phase de formation

Nous entraînons le modèle sur l'ensemble d'entraînement pour lancer l'apprentissage. On a comme paramètre nos tweets de train avec leurs polarités, taille du lot = 64, le nombre d'épochè = 4 ainsi des tweets pour la validation.

Nous pouvons voir ci-dessous qu'à chaque fois qu'on augmente le nombre des épochès de notre modèle, l'accuracy de l'apprentissage et de test est amélioré (figure 62). Après quelques périodes, nous atteignons accuracy de validation d'environ 87% (tableau 7).

```
Train on 7118 samples, validate on 1780 samples
Epoch 1/4
7118/7118 [==============================] - 55s 8ms/step - loss: 0.4303 - accuracy: 0.7782 - val_los
s: 0.3066 - val_accuracy: 0.8562
Epoch 2/4
7118/7118 [==============================] - 52s 7ms/step - loss: 0.1560 - accuracy: 0.9404 - val_los
s: 0.2874 - val_accuracy: 0.8798
Epoch 3/4
7118/7118 [==============================] - 51s 7ms/step - loss: 0.0610 - accuracy: 0.9803 - val_los
s: 0.3374 - val_accuracy: 0.8680
Epoch 4/4
7118/7118 [==============================] - 51s 7ms/step - loss: 0.0301 - accuracy: 0.9900 - val_los
s: 0.4230 - val_accuracy: 0.8753
```

Figure 62 Phase de formation

Epoch	val_loss	val_accuracy	loss	accuracy
1	0.306571	0.856180	0.430280	0.778168
2	0.287387	0.879775	0.155975	0.940433
3	0.337410	0.867977	0.061000	0.980332
4	0.423029	0.875281	0.030064	0.990025

Tableau 7 phase de formation

B. Phase de test

Nous pouvons tester notre modèle avec l'ensemble de test pour vérifier comment il prédit le sentiment de chaque texte comme ci-dessous (figure 63):

```
thank cox north psychological department raising money mental health awareness month we [0.2972494
0.70275056] [0 1]
wisest words ive heard dont allow depression control life control depression its hard possible days
[0.88641256 0.11358742] [1 0]
```

Figure 63 Phase de test

La sortie donne le texte, la prédiction ainsi que les valeurs de sentiments réels (il y a deux classes: négative et positive respectivement).En comparant la prédiction aux valeurs de sentiments réels, nous pouvons voir dans chaque texte que le modèle est très proche du vrai résultat. Par exemple, dans le premier exemple, nous avons 70% pour la deuxiemecatégorie de prédiction, et d'autre part, nous remarquons que dans les valeurs de sentiment réel ce texte a une polarité de sentiment positive(1 pour la deuxiemecatégorie "positive"), donc c'est juste. Et aussi pour le deuxième exemple nous avons 88% pour la première classe de prédiction, et dans les valeurs de sentiment réel, ce texte a une polarité de sentiment négative (1 pourla premièrecatégorie "négative"), et c'est juste aussi.

C. Phase d'évaluation

Voici la matrice de confusion pour le modèle de réseau neuronal récurrent :

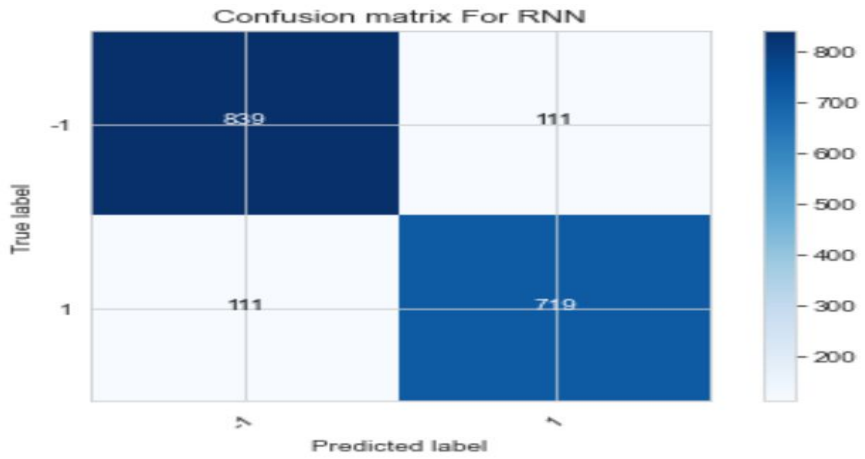


Figure 64 Matrice de confusion pour un modèle de réseau neuronal récurrent

À partir de la matrice de confusion, nous pouvons voir que:

- 719 individus sont correctement identifiés comme étant non déprimés par le modèle (TP).
- 839 individus sont correctement identifiés comme déprimés par le modèle (TN).
- 111 individus déprimés identifiés comme non déprimés par le modèle (FP).
- 111 individus non déprimés qui ont été identifiés comme déprimés par le modèle (FN).

Voici les résultats de performance du modèle de réseau neuronal récurrent :

	Accuracy	Precision	Recall	F-measures
Recurrent Neural Network	87.528092	86.626506	86.626506	86.626506

Figure 65 Résultats de performance du modèle de réseau neuronal récurrent

La performance du classificateur RNN dans notre ensemble de données montre de meilleurs résultats, il donneles mêmes résultats en termes de la précision, du rappel et la mesure f à 86%.Et il donne de meilleures performances en termes d'accuracy 87%.

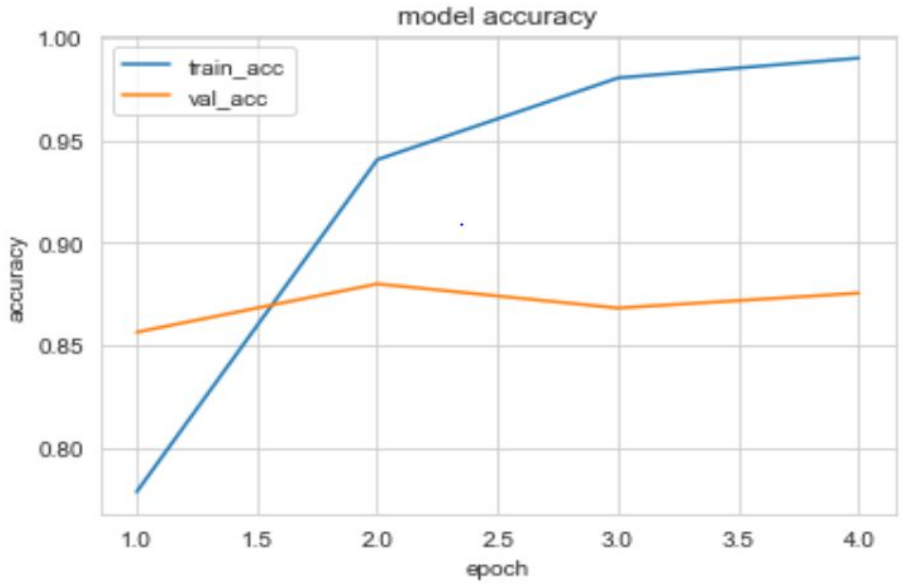


Figure 66 Graphique de précision du modèle (Formation et validation)

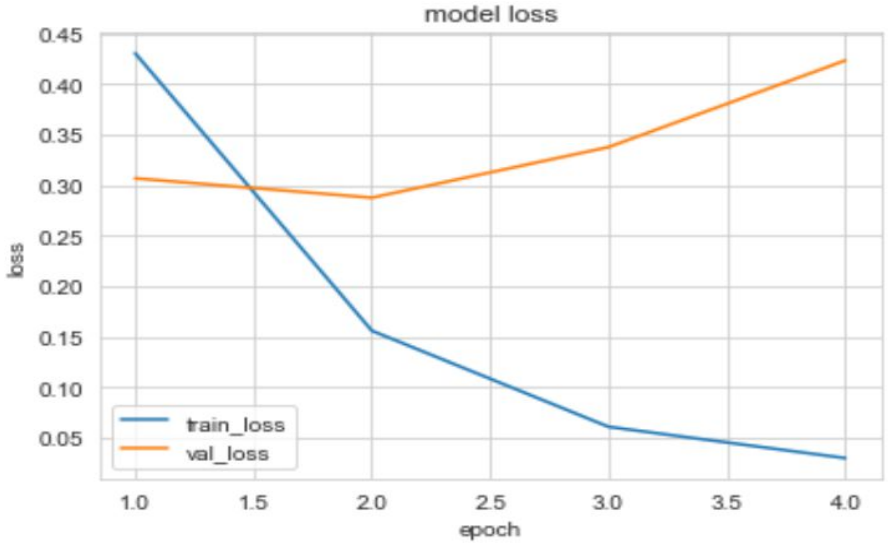


Figure 67 Graphique d'erreur du modèle (Formation et validation)

D'après la Figure 66 la précision de l'apprentissage augmente avec le nombre d'époque, ceci reflète qu'à chaque époque le modèle apprenne plus d'informations.

De même, d'après la Figure 67, l'erreur d'apprentissage a diminué et la validation augmente avec le nombre d'époque.

III.5 Analyse comparative des classificateurs choisis

Dans le tableau 8 suivant, nous résumons tous les résultats de performance obtenus et nous effectuons une analyse comparative des différents classificateurs choisis afin de trouver celui qui est le meilleur parmi tous :

Approches	Classificateurs	Accuracy	Précision	Rappel	F-mesure
Approche lexicale pour les modèles traditionnels	Naive Bayes	76.06	75.45	74.35	74.81
	Decision tree	79.85	78.91	78.45	78.64
	Support vector machine	80.13	82.95	77.15	78.95
	K Nearest Neighbor	72.43	74.42	70.24	71.53
	Random Forest	47.15	48.32	36.87	27.08
Approche sémantique pour les modèles traditionnels	Naive Bayes	83.53	82.38	82.28	82.33
	Decision tree	86.34	85.57	85.06	85.31
	Support vector machine	87.97	88.98	84.69	86.79
	K Nearest Neighbor	81.06	76.36	86.02	80.906
	Random Forest	68.31	97.16	33.01	49.28
Approche d'apprentissage profond	Recurrent Neural Network	87.528	86.626	86.626	86.626

Tableau 8 Performances des approches de classification

Pour les deux approches lexico-syntaxique et sémantique (algorithmes traditionnels de ML), On remarque une dégradation des performances de l'approche lexico-syntaxique par rapport à l'approche sémantique.

Si on prend par exemple le classifier SVM, il est de meilleur performance dans les deux approches, il donne accuracy de 87.9%, une précision de 88,9%, un rappel de 84.6% et une mesure f de 86.7% pour l'approche sémantique. Par contre pour l'approche lexicale, le classifier SVM donne accuracy de 80.1%, une précision de 82,9%, un rappel de 77.1% et une mesure f de 78%

Donc comme nous l'avons noté, les modèles traditionnels font le mieuxpour l'approche sémantique plutôt que pour l'approche lexico-syntaxique en critère deperformance de classification (accuracy, précision, rappel, mesure-f).

En comparant l'approche sémantique pour les modèles traditionnels par rapport à l'approche d'apprentissage en profondeur, on constate que les deux modèles SVM et RNN donnentle même bon résultat en termes d'accuracy 87%, et en termes de mesure f 86% par rapport aux autres classificateurs.Nous pouvons donc conclure que SVM et RNN sont préférable pour trouver le résultat exact. Passons à la précision, RF a une précision de 97%, la plus élevée parmi toutes. Cela signifie que RF donne des résultats substantiellement pertinents que les résultats non pertinents. Et d'autre part,RNN et KNN ont eu un rappel de 86% qui est le plus élevé de tous les classificateurs, ce qui signifie que nos classificateurs ont renvoyéles résultats les plus pertinents. Les résultats sont présentés dans le diagramme à barres ci-dessous (figure 68) :

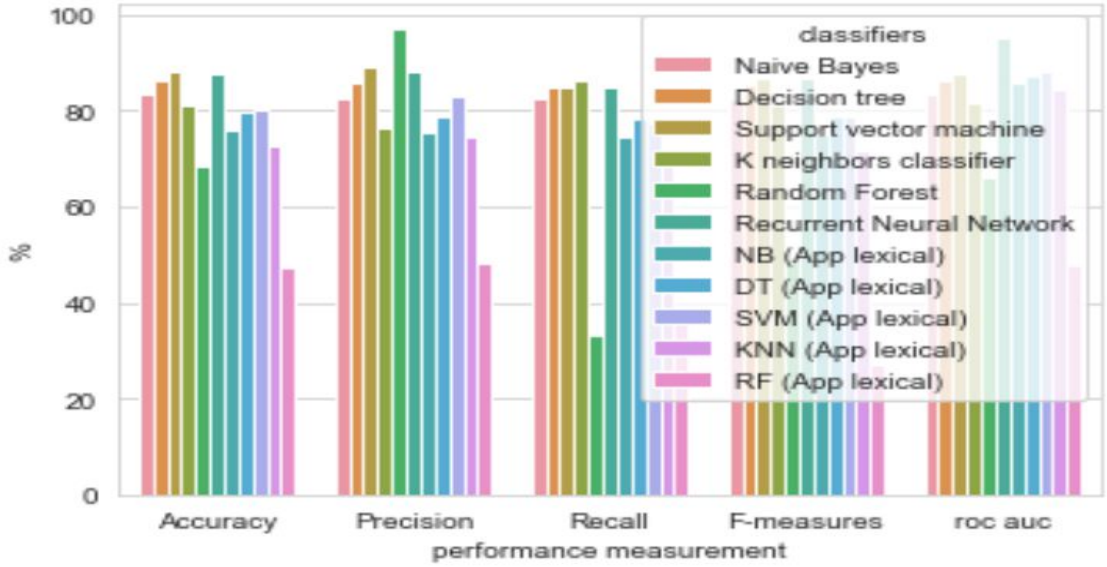


Figure 68 Représentation graphique des résultats de performance

Pour confirmer les résultats précédents et obtenir une vue plus large des performances des classificateurs, nous avons utilisé la courbe ROC qui montre les tests exacts (plus le graphique est proche de la diagonale, moins le test est précis).

Nous avons également utilisé l'AUC, qui fournit une mesure globale de la performance pour tous les seuils de classification possibles afin de démontrer la précision du test. Voici notre espace ROC pour les trois approches (figure 69):

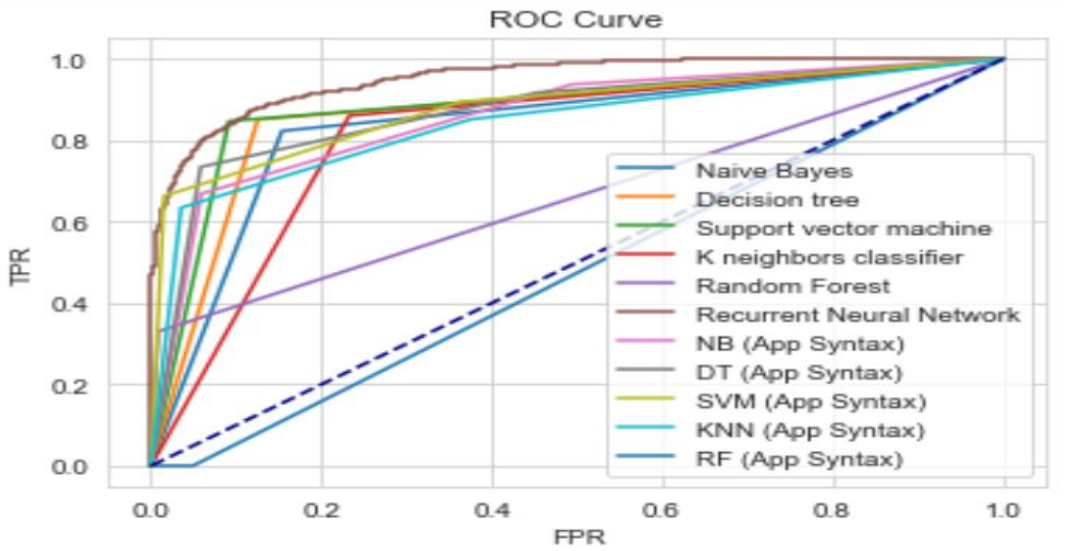


Figure 69 Représentation de la courbe roc des algorithmes de classification

Dans le tableau suivant nous résumons tous les AUC obtenus des modèles (tableau 9) :

Classificateurs	NB	DT	SVM	KNN	RF	NB (App Lexical)	DT (App Lexical)	SVM (App Lexical)	KNN (App Lexical)	RF (App Lexical)	RNN
AUC	82.83	84.59	87.18	78.89	63.14	81.14	81.65	83.20	79.85	47.02	**95.01**

Tableau 9 AUC des classificateurs

En tant que résultat final, grâce à l'espace ROC et AUC, nous concluons que les algorithmes MLpour l'approche sémantique obtiennent les meilleures performances par rapport à ce qu'ils ont réalisé dans l'approche lexico-syntaxique, ce qui nous prouve la précision, l'exactitude et l'efficacité de l'utilisation de l'approche sémantique pour analyser une grande quantité de données afin de déduire les sentiments dépressifs qui ont été exprimés.

Nous constatonségalement que le modèle de DL le RNN surpasse les autres algorithmes, Comme nous pouvons l'observer dans la Figure 69, le modèle RNN a un graphique proche des bordures supérieure et gauche, Ainsi, Il est très surperformé avec un score AUC élevé de 95% (tableau 9), donc ce classificateur donne de bonnes performances et il est très précis et prédit très bien parmi tous.

Cela nous montre l'avantage de l'application d'une approche DL dans ce type de tâches d'analyse de sentiments et la bonne performance que nous obtenons par rapport aux algorithmes ML traditionnels.

III.6 Conclusion

Dans ce chapitre, nous avons utilisé des techniques de domaines de catégorisation du texte et d'apprentissage automatique pour les besoins d'analyse des sentiments des tweets ou nous nous sommes concentrés particulièrement sur la tâche de leurs classifications. Les expérimentations menées en passant par deux ressources lexicales et entrainant et testant deux modèles de ML nous ont permis de valider notre application et d'afficher les différents résultats. Aussi que,les représentations graphiques offertes par notre application permettent de mieux comparer et analyser les différents classificateurs.

La comparaison des résultats trouvés a montré que le modèle RNN/LSTM a donné des résultats satisfaisants et que plus le niveau d'analyse est profond, plus les résultats sont spécifiques.

CONCLUSION GÉNÉRALE

Conclusion générale

Tout le monde le sait, analyser le sentiment des contenus postés par les internautes sur les réseaux sociaux est bien plus compliqué qu'il n'y apparaît.

Dans ce PFE, nous avonsprésenté en détail les aspects théoriques et pratiques liés à l'implémentation de notre système d'analyse des sentiments. Il s'agit donc de la classification automatique des textes de tweets en trois classes : positive, négative ou neutre via les algorithmes traditionnels et en profondeur de ML en passant par une ressource lexicale (dictionnaire donné ou SentiWordnet) pour la phase de prétraitement de données originales.

Le potentiel de l'utilisation dubig-social data "Twitter" comme outil de mesure et de prédiction du trouble dépressif a été soigneusement démontré et les expérimentations menées ont permis de valider notre application.Les expériences faites ont montré que plus le niveau d'analyse est profond plus les résultats sont spécifiques.

Ce travail nous a permis de mettre plus en pratique nos connaissances théoriques sur les algorithmes traditionnelles de ML et les réseaux de neurones ainsi que de les enrichir, et le plus important est que nous fassions le premier pas vers l'apprentissage profond, un des champs les plus importants de l'Intelligence Artificielle.

Parmi les orientations futures:

- Assemblage (hybridation/combinaison)des modèles de ML doit être une forme de technique d'algorithmes de méta-apprentissage dans laquelle nous proposons de combiner différents classificateurs afin d'améliorer la précision de la prédiction.
- Tester l'outil implémenté sur d'autres jeux de données (emojis, images, videos...) et sur d'autres plateformes comme facebook, youtube ou autres
- Application d'autres classificateurs et utilisation d'autres fonctionnalités
- Création d'un lexique orienté santé mental

RÉFÉRENCES BIBLIOGRAPHIQUES

Bibliographie

[1] M. Nadeem, M. Horn, et G. Coppersmith, « IdentifyingDepression on Twitter », p. 9.

[2] Lohard, D. OPINION MINING ET SENTIMENT ANALYSIS. Marseille. : s.n., 2012. p. 35.

[4] AIT BACHIR S, OUSSALAH A., Diffusion des opinions dans les réseaux sociaux en temps-réel. Mémoire, Université des Sciences et de la Technologie Houari Boumediene, 2016. p. 63.

[5] Atique1, Harshali P. Patil1 and Mohammad. Applications, Issues and Challenges in Sentiment analysis and Opinion Mining– A User's perspective, international journal of control theory and applications. 2017. p. 789.

[6] HaseenaRahmath P, Opinion Mining and Sentiment Analysis - Challenges and Applications, Dept.of Computer Science and Engineering, Al-FalahSchool of Engineering, Dhauj, Haryana, India, May 2014. p. 20.

[10] H. Bachir, G. Ayyoub, et M. M. Riadh, « Détection d'opinions à partir de Twitter », p. 74.

[11] Ribeiro, C. S. (2010). Inductive infeTencefoT large scaletext classification. berlinheidelberg: springer-verlag. p. 94.

[15] M. Abdelkader, « L'ANALYSE DU SENTIMENT UTILISANT LE DEEP LEARNING », p. 84.

[19] « The International Conferenceon Advanced Machine Learning Technologies and ... - Google Livres ». https://books.google.dz/books?id=N0-NDwAAQBAJ&pg=PA891&lpg=PA891&dq=wordnet+preprocessing&source=bl&ots=4DR_HowZj3&sig=ACfU3U0KzkIeiJtWORZF7PDF3qxl_XCoOw&hl=fr&sa=X&ved=2ahUKEwjn9YTW4dfpAhVFC2MBHV3fBg0Q6AEwDnoECAcQAQ#v=onepage&q=wordnet%20preprocessing&f=false.p. 891.

[27] A. Ramji, *AshwanthRamji/Depression-Sentiment-Analysis-with-Twitter-Data*. 2020.

[28] Mehdi, M. Fouille d'opinion dans les reseaux sociaux. alger : s.n., 2016-2017. p. 58-65.

[31] Liu, B. Sentiment analysis and opinion mining. Synthesis lectures on humanlanguage technologies. 2012. p. 106-112.

Webographie

[3] M. Hadji, « Analyse des sentiments : Généralités », *Medium*, août 22, 2019. https://medium.com/@mehdihadji/analyse-des-sentiments-g%C3%A9n%C3%A9ralit%C3%A9s-99ab87503a5e (consulté le juin. 20, 2020).

[7] « TextBlob: SimplifiedTextProcessing — TextBlob 0.16.0 documentation ». https://textblob.readthedocs.io/en/dev/ (consulté le juin. 20, 2020).

[8] « WhatisTextMining in Data Mining - Process& Applications », *DataFlair*, févr. 27, 2018. https://data-flair.training/blogs/text-mining/ (consulté le juin. 20, 2020).

[9] « abhilash final thesis.pdf ». Consulté le mai. 24, 2020. [En ligne]. Disponible sur: http://dspace.dtu.ac.in:8080/jspui/bitstream/repository/17036/1/abhilash%20final%20thesis.pdf.

[12] « Yassine Bensaoucha.pdf ». Consulté le: juillet. 03, 2020. [En ligne]. Disponible sur: http://dspace.univ-msila.dz:8080/xmlui/bitstream/handle/123456789/2208/Yassine%20Bensaoucha.pdf?sequence=1&isAllowed=y.

[13] « K-NearestNeighbor in Machine Learning ». https://www.knowledgehut.com/blog/data-science/knn-for-machine-learning (consulté le mai. 15, 2020).

[14] « Une approche Deep Learning pour l'analyse des Sentiments Sur Twitter.pdf ». Consulté le: juin. 10, 2020. [En ligne]. Disponible sur: http://dspace.univ-km.dz/xmlui/bitstream/handle/123456789/2625/Une%20approche%20Deep%20Learning%20pour%20l%E2%80%99analyse%20des%20Sentiments%20Sur%20Twitter.pdf?sequence=1&isAllowed=y.

[16] « un Architecture Basé sur l'Apprentissage Profond (Deep Learning) pour la Détection d'Intrusion.pdf». Consulté le: mai. 27, 2020. [En ligne]. Disponible sur: https://dspace.univ-adrar.edu.dz/jspui/bitstream/123456789/3175/1/D%C3%A9veloppement%20d%E2%80%99une%20Architecture%20Bas%C3%A9e%20sur%20l%E2%80%99Apprentissage%20Profond%20(Deep%20Learning)%20pour%20la%20D%C3%A9tection%20d%E2%80%99Intrusion%20dans%20les~.

[17] « Understanding LSTM Networks -- colah's blog ». https://colah.github.io/posts/2015-08-Understanding-LSTMs/ (consulté le mai. 30, 2020).

[18] J. Brownlee, « Stacked Long Short-Term Memory Networks », *Machine Learning Mastery*, août 17, 2017. https://machinelearningmastery.com/stacked-long-short-term-memory-networks/ (consulté le juin. 08, 2020).

[20] « Sentiment140 - A Twitter Sentiment AnalysisTool ». http://www.sentiment140.com/ (consulté le juin. 08, 2020).

[21] « tweetfeel». http://www.tweetfeel.com (consulté le juin. 26, 2020).

[22] « TwitrRatr | Latest Twitter & Social Media News, Trends & Updates ». http://twitrratr.com/ (consulté le juin. 26, 2020).

[23] « streamcrab». http:// smm.streamcrab.com (consulté le juin. 26, 2020).

[24] S. Narkhede, « Understanding Confusion Matrix », *Medium*, août 29, 2019. https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62 (consulté le juillet. 29, 2020).

[25] « Classification : ROC et AUC | Cours d'initiation au machine learning », *Google Developers*. https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc?hl=fr (consulté le juillet. 29, 2020).

[26] « Twitter », *Wikipédia*, Consulté le: août. 13, 2020. [En ligne]. Disponible sur: https://fr.wikipedia.org/w/index.php?title=Twitter&oldid=176157348.

[29] D. Monsters, « TextPreprocessing in Python: Steps, Tools, and Examples », *Medium*, oct. 15, 2018. https://medium.com/@datamonsters/text-preprocessing-in-python-steps-tools-and-examples-bf025f872908 (consulté le août. 12, 2020).

[30] R. Koenig, « NLP for Beginners: Cleaning&PreprocessingText Data », *Medium*, juill. 29, 2019. https://towardsdatascience.com/nlp-for-beginners-cleaning-preprocessing-text-data-ae8e306bef0f (consulté le août. 12, 2020).

[32] « Distribution Anaconda pour Python - Numérique et sciences informatiques ». https://numerique-sciences-informatiques.discip.ac-caen.fr/distribution-anaconda-pour-python (consulté le août. 15, 2020).

[33] « Scikit-learn », *Wikipédia*. oct. 20, 2020, Consulté le: août. 15, 2020. [En ligne]. Disponible sur: https://fr.wikipedia.org/w/index.php?title=Scikit-learn&oldid=175750983.

[34] « SKLearn | Scikit-Learn In Python | SciKitLearn Tutorial », *AnalyticsVidhya*, janv. 05, 2015. https://www.analyticsvidhya.com/blog/2015/01/scikit-learn-python-machine-learning-tool/ (consulté le août. 16, 2020).

[35] « Pandas », *Wikipédia*. juin 20, 2020, Consulté le: août. 16, 2020. [En ligne]. Disponible sur: https://fr.wikipedia.org/w/index.php?title=Pandas&oldid=172173577.

[36] « Matplotlib », *Wikipédia*. juin 20, 2020, Consulté le: août. 16, 2020. [En ligne]. Disponible sur: https://fr.wikipedia.org/w/index.php?title=Matplotlib&oldid=172172874.

[37] « Seaborn », *Wikipédia*. Consulté le: août. 16, 2020. [En ligne]. Disponible sur: https://fr.wikipedia.org/wiki/Seaborn.

[38] « Flask (framework) », *Wikipédia*. oct. 09, 2020, Consulté le: août. 20, 2020. [En ligne]. Disponible sur: https://fr.wikipedia.org/w/index.php?title=Flask_(framework)&oldid=175415025.