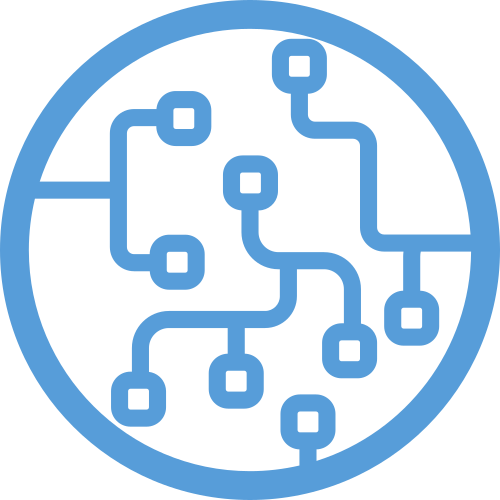
**Financial Sentiments**

****

***Projet P3***

***Capocasale Romain***

***INF3dlm-A***

***He-Arc, Suisse***

***2019***

Table des matières

[1. Résumé 3](#_Toc22299517)

[2. Introduction 3](#_Toc22299518)

[3. Cahier des charges 4](#_Toc22299519)

[4. Planning 6](#_Toc22299520)

[5. Analyse 7](#_Toc22299521)

[5.1. Etat de l’art 7](#_Toc22299522)

[5.2. Schéma du pipeline des processus 7](#_Toc22299523)

[7](#_Toc22299524)

[6. Solutions développées 8](#_Toc22299525)

[6.1. Métrique d’évaluation du modèle 8](#_Toc22299526)

[6.2. Jeu de données 8](#_Toc22299527)

[6.2.1. Récupération des données 8](#_Toc22299528)

[6.2.2. Analyse des données 9](#_Toc22299529)

[6.2.3. Comparaison avec des modèles existants 9](#_Toc22299530)

[6.3. Prétraitement des données 10](#_Toc22299531)

[6.3.1. Traitement du texte 10](#_Toc22299532)

[6.3.2. Vectorisation 10](#_Toc22299533)

[6.3.3. TF-idf 10](#_Toc22299534)

[6.4. Modèle de machine learning 10](#_Toc22299535)

[6.5. Évaluation des modèles de machines learning 10](#_Toc22299536)

[6.6. Calcul de sentiment sur de nouveau texte 10](#_Toc22299537)

[7. Conclusion 10](#_Toc22299538)

[8. Bibliographie 10](#_Toc22299539)

# Résumé

# Introduction

Dans le cadre du 5ème semestre de Bachelor en développement logiciel et multimédia, nous avons pour objectif de réaliser un projet individuel qui permet de nous permet de nous préparer le mieux possible au travail de Bachelor. J’ai alors sélectionné un projet dans le domaine du machine learning.

Dans le cadre du machine learning, le domaine de l'analyse de sentiments dans les textes a pris beaucoup d'intérêt ces dernières années grâce au succès et aux bonnes performances des algorithmes de classification de texte. Dans un premier temps cette technologie à été utilisé pour analyser les sentiments dans des domaines tels que les commentaires de films, les hôtels, les restaurants, etc

Le but du projet est d’appliquer cette technologie d’analyse de sentiment au domaine du marché financier dans le but de dire si un texte à plutôt un sentiment positif ou négatif. Ce sentiment pourrait alors être utilisé dans le domaine du FOREX (Foreign Exchange Market). Une information perçue comme positive pourrait indiquer une montée d’une devise par rapport à une autre tandis qu’une information négative pourrait en indiquer la chute.

L’objectif du projet est de créer un analyseur de sentiment pour le domaine de la finance. Le modèle sera entrainé avec des textes et vocabulaires propres au thème de la finance. Dans un premier temps, les solutions d’analyse de sentiment présente sur le web seront analysé pour résoudre ce problème. Dans un second temps, le but est de créé un modèle de machine learning dans le but de prédire un sentiment par rapport à un texte financier.

# Cahier des charges

Le but du projet est de créer un modèle d'analyse de sentiment pour le domaine de la finance. Un analyseur de sentiment est un programme informatique qui en entrée demande une phrase ou un texte et en sortie indique un sentiment pour cette phrase sur une échelle de -1 à 1. -1 pour un sentiment négatif et 1 pour un sentiment positif.

Étant donné que le domaine de la finance dispose de son propre langage, il est intéressant de développer un modèle spécialisé dans ce secteur dans le but d'obtenir de meilleur résultat qu'un analyseur de sentiment global qui aurait été entraîné à faire des prédictions sur tout le vocabulaire de la langue anglaise.

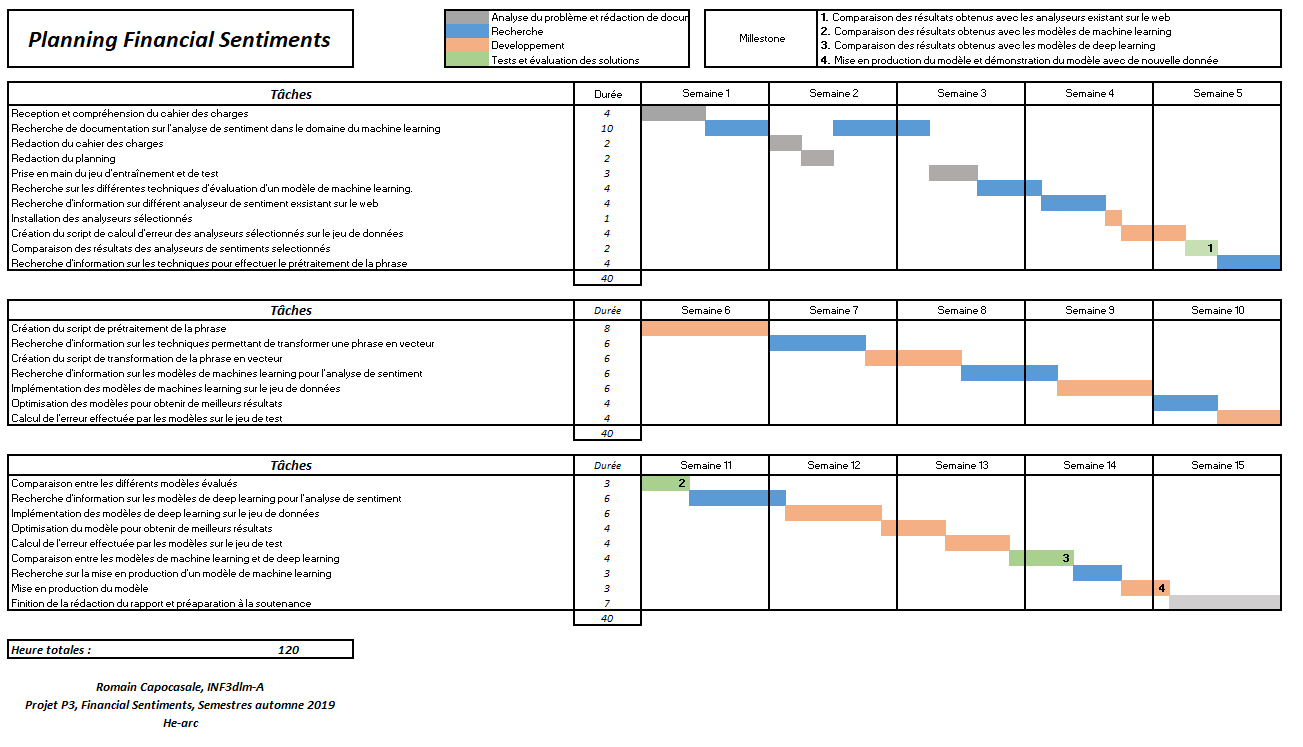
Pour l'entraînement et le test du modèle, un ensemble de titres d'article de finance avec la valeur du sentiment correspondant a été fourni. Le modèle sera réalisé en Python.

Pour ce projet, il faudra dans un premier temps, comparer le score des analyseurs de sentiment existant sur le web avec le jeu de donnée fourni. Par la suite, le but est de développer un modèle spécialisé dans l'analyse de sentiment pour la finance. La première étape consistera à prétraiter la phrase pour enlever la ponctuation, les déterminants, etc. Il faudra par la suite, transformer la phrase en vecteur dans le but d'être utilisé dans un modèle de machine learning. Différents modèles de machine learning devront être implémentés et testés à l'aide du jeu de donnée fourni. Par la suite, un modèle plus complexe utilisant des techniques de deep learning pourra être implémenté. Il faudra également comparer ces différents modèles entre eux pour sélectionner celui effectuant le moins d'erreurs de prédiction.

Le but final(hors projet), serait d'utiliser ce modèle d'analyse de sentiment dans le but de suivre la tendance haussière et baissière dans les pairs de devises (par exemple USD/CHF et EUR/USD). Des sources textuelles seraient récoltées depuis le site de Reuters et d'autres canaux financiers spécialisés dans le but d'en prédire un sentiment (entre -1 et 1) d'une période de trading (une heure, une demi-journée, journée, semaine ...) et relatif à un sujet (par exemple Brexit, échanges commerciaux avec la Chine, ....).

Ce classificateur avec d'autres en cours de développement, sera intégré dans un outil de prédiction du taux change.

# Planning



# Analyse

# Etat de l’art

Il existe de nombreux analyseur de sentiment sur le web comme par exemple : Textblob, VADER (nltk), etc. Ces analyseurs sont très simples à utiliser, il suffit d’indiquer la phrase et la librairie va indiquer la polarité de la phrase entre -1 et 1. Ces analyseurs ont cependant été entrainé sur tout le vocabulaire de langue anglaise et non sur le vocabulaire de la finance spécifiquement. Ils auront alors tendance à commettre plus d’erreur qu’un classificateur entrainé sur un vocabulaire spécifique. Un modèle entrainé avec un vocabulaire spécifique donnera une pondération plus importante aux mots qu’il considère plus important dans le contexte donné.

Si on regarde dans le domaine de la finance spécifiquement, d’après mes recherches ils n’existent pas de libraire ou d’analyseur de sentiment spécifique à la finance. Ils existent cependant de nombreux tutoriels qui expliquent comme créé un modèle à partir de texte contenant du vocabulaire financier. On trouve également pas mal d’article qui explique l’utilité du NLP (Natural Language Processing) appliqué au domaine de la finance.

Avec ces différentes recherches, on constate donc qu’il serait intéressant de créer un modèle de Machine Learning entrainé avec un vocabulaire spécifique à la finance.

# Schéma du pipeline des processus

# 

Ce schéma montre les différentes étapes qui seront effectués dans ce projet dans le but d’obtenir un modèle de machine learning.

Dans un premier temps, les données provenant de différentes sources devront être recueillis puis séparé entre les textes (phrase à analyser) et les étiquettes (sentiment pour une phrase). C’est également dans cette étape que les données devront être analysé dans le but d’en ressortir les tendances, subtilité, etc.

Dans un second temps, les données doivent être prétraités, dans le but de pouvoir être introduit dans un algorithme de machine learning. Le texte doit d’abords être traité dans le but de retiré le bruit. C’est-à-dire enlever la ponctuation, enlever certain mots, etc (pour plus de détails lire la partie dans la chapitre du traitement des données).

Après avoir traité le texte, il faut maintenant le découper en vecteur. En effet, les algorithmes de machine learning ne peuvent travailler qu’avec des nombres et non du texte. Par la suite, une transformation Tf-idf est appliqué sur le vecteur dans le but de données une pondération supérieur au mot qui se trouve souvent dans l’ensemble des textes (corpus).

A présent, différents algorithmes de machines learning doivent être étudié et implémenté pour résoudre le problème. Les données seront alors séparées en données d’entrainement dans le but d’entrainé le modèle et en données de test pour effectuer l’évaluation du modèle.

Une fois le modèle conçu, il faut l’évaluer avec les données de test et calculer différents scores qui permettront de se faire une idée de la fiabilité ou non du modèle. Selon les scores obtenus, il peut être intéressant de modifier certains paramètres des étapes vu précédemment dans le but d’améliorer les modèles. Certains paramètres du modèle (hyper-paramètres) peuvent aussi être adapté et modifié.

Une fois qu’un modèle satisfaisant a été obtenu, il est maintenant possible d’effectuer de nouvelle prédiction sur de nouveau texte. Le modèle prédira un score entre -1 et 1 pour ce texte (regression). Il peut également être intéressant de convertir ces résultats en classes(classification) dans le but de savoir si un texte est positif ou négatif sans se préoccuper du score précisément. Pour effectuer cette étape, les prédictions du modèle devront être placé dans la classe positive ou négative selon un certain seuil.

# Solutions développées

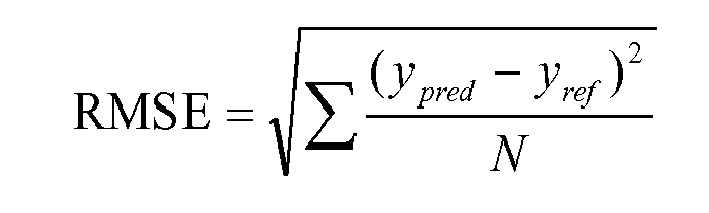
Voici les différentes solutions développées ainsi que l’argumentation sur le choix de ces solutions sur la base des recherches effectué en amont sur le domaine.

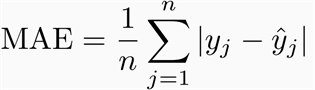
## Métrique d’évaluation du modèle

Avant de commencer à parler des solutions développées, il est important de définir les métriques qui nous permettront d’évaluer notre modèle. C’est grâce à cela, qu’on pourra dire si un modèle est viable ou non.

Les métriques d’évaluations choisi pour ce projet sont d’une part la ***RMSE*** (Root Mean Square Error) et la ***MAE*** (Mean Absolute Error) qui sont des métriques sur des modèles de régression et ***l’accuracy*** qui est une mesure sur les modèles de classification.

* La RMSE est la racine carrée de l’erreur quadratique moyenne, elle permet de donner une idée de l’importance des erreurs de prédictions commise par le système.



* La MAE est l’erreur absolue moyenne, elle permet de donner moins d’importance au valeurs extrêmes.
* L’accuracy permet de déterminer l’exactitude des prédictions, mais avant de pouvoir utiliser cette métrique, il faut d’abords convertir le vecteur des prédictions qui contient des valeurs entre -1 et 1. Pour effectuer cela, il faut fixer un seuil (par défaut 0) qui permet de dire qu’une valeur en dessous de seuil sera considéré comme positive et une valeur supérieure au seuil sera positive. L’accuracy se calcule alors en divisant le nombre de prédiction correcte par le nombre de prédiction totale.

## Jeu de données

## Récupération des données

Pour ce projet, deux jeux de données de deux sources différentes m’ont été fourni. Les fichiers était au format JSON. Les 2 jeux de données contenaient chacun une colonne avec la phrase et le sentiment correspondant. Cependant, chaque jeu de données avait des colonnes supplémentaires que l’autre jeu de donnée n’avait pas avec d’autres informations. Un contenait par exemple l’entreprise qui était le sujet du texte et dans l’autre jeu de données une colonne contenait la provenance des textes. Même si ces informations auraient pu être intéressant à inclure dans le modèle, j’ai préféré ne garder que les colonnes avec la phrase et le sentiment correspondant dans le but d’avoir exactement les mêmes colonnes pour les deux jeux de données. Cela me permet d’accroitre la taille du jeu de données final. De plus, si d’autre variables avaient été inclus dans le modèle, il aurait fallu les renseigner lors de future prédiction ce qui peut être contraignant.

J’ai alors créé un script python (présent sous data/rebuild\_data.py) permettant de récupérer les deux jeux de donnés et d’en ressortir un unique avec seulement les informations sélectionnées précédemment. Les données retournées sont sous forme d’un fichier au format JSON.

# Analyse des données

Une fois les données récupérées, il est important d’effectuer une analyse des données dans le but de voir différentes statistiques sur les données, des tendances ou des subtilités. Ceci dans le but de bien comprends comment sont structuré, afin de construire un modèle efficace.

On remarque alors différentes statistiques sur les données, voici les plus importantes (pour plus de statistiques, consulter le notebook) :

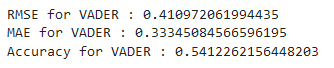
* Il y a en tout 2866 lignes (2866 phrases et 2866 sentiment correspondant). On remarque qu’il y a assez peu de données.
* Il y a 1756 phrases positives, 1045 phrases négatives et 65 phrases neutres. On remarque alors qu’il y a plus de phrase positive que négative. Cette réparation est normale étant donné qu’en général il y a plus de texte ou de phrase positive que négative.
* Il y a environ 7,5 mots par phrase. Ce qui est de nouveau un nombre assez petit.

# Comparaison avec des modèles existants

Pour le projet, il m’a été demandé de comparer les scores d’analyseur de sentiment présent sur le web. Les scores obtenus ici seront utilisés comme score de référence pour l'évaluation des futurs modèles. Le but des futurs modèles est de battre ces scores. Sur la base des recherches effectués, deux modèles d’analyse de sentiment ont été retenu : la librairie textblob et VADER un sous package de la libraire NLTK.

Voici les scores qu’obtiennent ces deux analyseurs :

* Pour textblob :



* Pour VADER :

On constate que les résultats obtenus sont plutôt mauvais. Même si la RMSE et la MAE ne sont pas si mauvaise, la précision est quant à elle très mauvaises. Les 2 modèles obtiennent une précision d’a peine plus de 50%, ce qui corresponds à un choix presque aléatoire de la classe. Au vu des scores, on constate qu’il est très intéressant de construire un modèle de machine learning avec pour but de battre les scores ci-dessus.

# Prétraitement des données

Ici, le but est de prétraités les données dans le but de pouvoir les introduire dans un algorithme de machine Learning. Ces différents processus seront alors ajoutés dans l’ordre (TextProcessor, Vectoroisation et Tf-idf) dans un pipeline sklearn.

## Traitement du texte

Voici les différents processus qui ont été effectué pour prétraiter la phrase :

* **Tokenisation** : Cette étape consiste à découper la phrase en mot.
* **Suppression des stop word** : Cette étape consiste à supprimer des mots qui ont peu d’importance dans la phrase comme « the », « are », etc.
* **Tagging et suppression de certains mots tagués** : Le tagging consiste à détecter à quelles classes appartient un mot comme par exemple nom, adjectif, verbe, etc. Par la suite, il faut supprimer le type de mot que l'on ne veut pas garder dans la phrase. Ceci peut être utile pour enlever par exemple les déterminants.
* **Suppression des mots en dessous d’une certaine taille** : Ce processus consiste à ne garder que les mots d'une certaine taille dans la phrase.
* **Suppression de la ponctuation** : Cette étape consiste à supprimer les caractère comme !, ?, etc.
* **Lemmatisation** : Ce processus consiste à ne garder que la racine des mots. Par exemple, les mots comme « runs », « running » seront transformer en « run ». Ceci permet de réduire le nombre de mot dans le dictionnaire final.

Il est a noté que les différents processus ci-dessus ne sont pas tous effectué systématiquement lors du traitement de la phrase. Le but est de regarder quels processus fait améliorer ou non le score du modèle.

Le traitement du texte est effectué avec la librairie Spacy. Pour effectuer ceci, j’ai créé une classe nommée « TextProcessor » héritant de « BaseEstimator » et de « TransformerMixin ». Le fait d'hériter de ces 2 classes permettra d'inclure cette classe dans le pipeline sklearn. Les différents processus vu ci-dessus peuvent être ajouté ou non au traitement via le constructeur de la classe. Le principale avantage de cette méthode est qu'elle se combine avec l'utilisation de la classe GridSearchCV() ou RandomizedSearchCV() qui permet de chercher les meilleurs hyper-parametrès pour un modèle.

## Vectorisation

## TF-idf

# Modèle de machine learning

Les 3 principaux modèles de machine Learning développé sont :

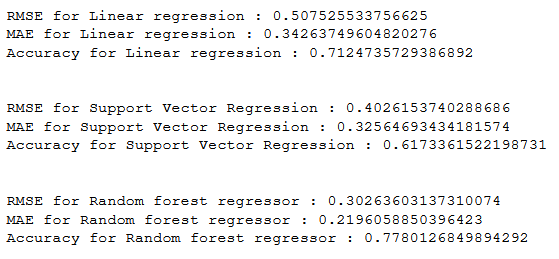
* La régression linéaire (LinearRegression)
* La machine à vecteur de support (SVR)
* La foret aléatoire (RandomForestRegressor)

La régression linéaire à été choisi car c’est un modèle de machine learning basique pour les tâches de régression. La régression linéaire est surtout utilisée pour s’assurer que les autres modèles ont de meilleurs résultats qu’un modèle basique. Les 2 autres modèles ont été sélectionné car ce sont 2 modèles puissant pour les tâches de régression.

Le modèle de machine learning est ajouté à la suite du pipeline. Les différents modèles sont alors entrainés sur le jeu d’entrainement.

# Évaluation des modèles de machines learning

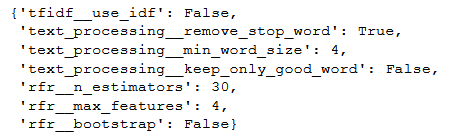
En évaluant ces modèles sur le jeu de test on obtient les résultats suivants :



Si on compare ces résultats avec les résultats obtenus précédemment par VADER et TextBlob on remarque que, au niveau de la RMSE et de la MAE la régression linéaire et la machine à vecteur de support on des résultats quasiment équivalemment aux résultats précédent (sauf la RMSE de la régression linéaire qui est supérieur). Seule la forêt aléatoire obtient de meilleur résultat sur la RMSE et la MAE. Au niveau de la précision on remarque que tous les nouveaux modèles entrainés ont de bien meilleurs résultats. La forêt aléatoire obtient même une précision d’environ 78%. Sur les différentes métriques ont remarqué que le meilleur modèle est le foret aléatoire. C’est donc ce modèle qui sera conservé.

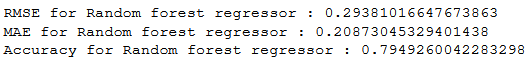
## Recherche par quadrillage

Dans le but d'améliorer encore la performance du modèle, il est possible d'effectuer une recherche par quadrillage pour trouver les meilleures hyperparamètres du modèle. La classe RandomizedSearchCV() de sklearn est utilisé pour rechercher de manière aléatoire les différents hyperparamètres. Un autre avantage d’utilisé cette classe est que, le modèle est entrainé par validation croisé ce qui permet de s’assurer que le modèle ne surévalue pas trop sur le jeu d’entrainement. La recherche est effectuée sur la forêt aléatoire étant donné que c’est le modèle qui a donné les meilleurs résultats. Après l’entrainement on obtient le dictionnaire suivant :



La clé correspond à l’hyperparamètre évalué et la valeur corresponds à la meilleure valeur de cet hyperparamètre. On déduit donc que, il ne vaut mieux pas utiliser l’idf, qu’il est mieux d’enlever les top word, qu’il vaut mieux supprimer les mots de moins de 4 caractères et qu’il vaut mieux garder tous les types de mots.

Les hyperparamètre du modèle de machine learning à proprement parlé (ceux préfixé par rfr\_) ont été choisi par rapport à l’exemple du livre.

Voici les métriques obtenues pour ce nouveau modèle : 

On remarque que le modèle c’est très légèrement amélioré sur toutes les métriques mais rien de très impressionnant. On obtient presque une précision de 80%.

# Calcul de sentiment sur de nouveau texte

Une fois le meilleur modèle trouvé, il est possible de l’exporter pour pouvoir conserver sont état actuel et ne pas devoir le ré entrainer à chaque fois. Pour exporter le modèle, j’utilise la libraire Joblib. Le modèle s’enregistre comme un fichier en « .joblib ». Il est mtn possible d’importer ce modèle dans un nouveau script et d’effectuer de nouvelle prédiction sur de nouveaux textex.

# Conclusion

# Bibliographie