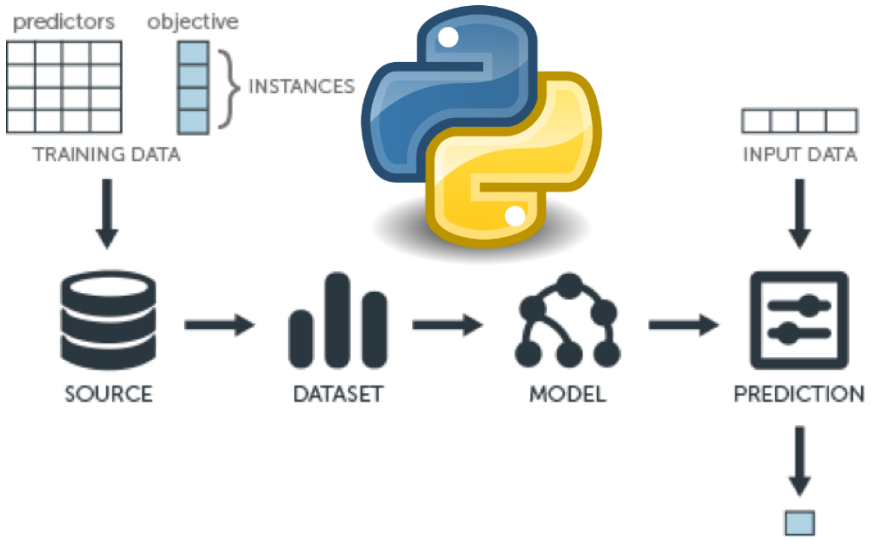
ECOLE PRATIQUE DES HAUTES ETUDES COMMERCIALES

Avenue du Ciseau, 15
1348 Louvain-la-Neuve

ANTICIPATIONS DE PANNES VIA LE M.L.

Travail de fin d'études présenté en vue de l'obtention du diplôme de bachelier en Informatique et Systèmes
orientation Technologie de l'Informatique

Constantin Mirica



Rapporteur : Arnaud Dewulf

Année Académique 2019 – 2020

Table des matières

Remerciements

Je tiens particulièrement à remercier toutes les personnes qui ont contribué au bon déroulement de mon TFE et qui signeront, je l'espère, cette étape de ma vie.

Je voudrais dans un premier temps remercier mon rapporteur de TFE M. Dewulf A. qui était extrêmement réactif à tous mes mails et qui, de par ses démarches, m'a aidé à conclure ce stade de mon cursus. Durant 8 mois, j'ai pu faire appel à lui sans aucun souci et bénéficier d' une réponse bienveillante ou de conseils afin de progresser dans mon travail.

Je remercie également M. Gerard J., responsable informatique de la faculté de droit, qui m'a recommandé cette entreprise et conseillé ce sujet. Il m'a de nombreuses fois proposé son aide et son soutien que, pour des raisons personnelles, j'ai tout de même préféré ne pas saisir.

J'adresse par la suite un grand merci à monsieur J. Gonze et l'entreprise Joassin Mazout pour un très bon début de collaboration et pour les données partielles, fournies.

Je tiens également à faire part de toute ma reconnaissance envers les personnes suivantes pour l'aide prodiguée lors de la réalisation de ce mémoire :

Madame Vroman M-N, qui a mis à disposition des étudiants les informations relatives au TFE ainsi que la manière de le rédiger et qui, avec l'ensemble de l'équipe pédagogique, me rappelait les différents deadlines à respecter.

Ma compagne, Mérrone Barbiau, qui m'a soutenu durant les moments difficiles et qui n'hésitait pas à m'encourager lorsque mes « pauses » se faisaient trop longues. Elle a eu la patience est la bonne volonté de relire et de corriger ce travail

1. Cahier de charges

i. Contexte

Durant les hivers froids et pluvieux de Belgique, certaines entreprises rencontrent régulièrement un problème peu commun : elles ont « trop de clients ». Ceci est le cas de l'entreprise Joassin, le plus important fournisseur en mazout de Namur et un des plus influents de Wallonie. Chaque hiver, des clients distraits appellent, contraints par l'urgence de leur situation, afin d'à nouveau remplir leur cuve. Malheureusement, à cause de ce flux de demandes conséquent, répondre d'une manière rapide et efficace s'avère laborieux, en grande partie à cause des facteurs matériels mais aussi organisationnels. Ceci entraine des retards importants et risque même d'engendrer des soucis de crédibilité et de réputation auprès des clients.

ii. Objectifs

Afin de palier à cela, le but du projet est de créer une application qui sera capable, grâce aux informations reçues de la base de données principale de l'entreprise, de créer un pattern permettant d'anticiper les pannes des clients. Ce système sera principalement utilisé pour éviter la surcharge de travail durant l'hiver, sachant que la plupart des retards sont observés lorsque l'agenda est le plus chargé. Pendant cette période, les commandes inattendues sont inacceptables et une organisation presque militaire est nécessaire pour pouvoir répondre aux demandes de tous les clients. Ayant pour objectif de limiter les conséquences financières ainsi que les soucis organisationnels, l'application démontre bel et bien sa nécessité et sa pertinence dans le cadre de l'entreprise.

iii. Description de l'existant

L'entreprise a à sa disposition un prototype qui est actuellement en test sur 59 personnes et qui, grâce aux degrés-jour[1], parvient à prévoir les pannes d'une manière relativement précise. Cependant, le problème majeur de cette solution est qu'elle ne prend pas en compte d'autres facteurs tels que : la taille de la maison, le nombre de membres dans la famille, le type de maison, etc. La solution est donc en quelque sorte oubliée et mise de côté, car elle implique une analyse journalière du fichier. Le nouveau projet fera lui aussi usage du principe des degrés-jours, car

[1] Le **degré jour unifié** (**DJU**) est la différence entre la température extérieure et une température de référence

l'idée et le développement de cette solution peuvent éventuellement être utilisés pour une meilleure approximation.

iv. Critères d'acceptabilité du produit

La première condition pour que le projet soit implémenté au sein de l'entreprise est que la Machine Learning soit capable de montrer des résultats concluants dans les phases de test ainsi qu'une amélioration dans la qualité et la précision des résultats.

Deuxièmement, la nouvelle application se doit d'être plus performante que le système-prototype de degrés-jour, et également capable de procéder à une évolution autonome.

v. Technologies utilisées

I. Stockage

Pour la base de données, la place de premier choix a dans un premier temps été attribuée à MySQL suite aux soucis de comptabilité entre WinDev et les autres bases de données. Le but final étant une implémentation totale sur le serveur de l'entreprise, la db se devait par la force des choses d'être compatible avec le reste de l'infrastructure.

En tenant compte du fait que le gestionnaire informatique ne répondait plus à mes mails, je ne pouvais pas risquer une incompatibilité entre MySQL et WinDev. Les données ont donc été stockées au format .CSV, ledit format étant assurément compatible avec les bases de données de WinDev.

II. Programmation – Python

Python sera utilisé autant pour la machine learning que pour la gestion de données sortantes de la base de données existante de l'entreprise. Particulièrement polyvalent grâce à ses nombreuses bibliothèques spécialisées dans ce genre de travail, Python m'a immédiatement semblé être le choix idéal pour la réalisation de mon projet. Afin de faire parvenir les résultats obtenus à l'utilisateur, le programme enverra un mail au responsable client dans le cadre de l'entreprise ou aux personnes présentant le risque de tomber en panne durant la prochaine semaine.

vi. Expression des besoins

I. Besoins fonctionnels

Les fonctions primaires / obligatoires sont :

- La connexion à la DBJ[2] pour pouvoir extraire les informations nécessaires
- Calculer les prévisions pour les clients
- Envoyer par mail, un fichier csv avec les résultats des prochaines panes, une semaine à l'avance (à discuter encore avec l'entreprise)

II. Besoins non fonctionnels

- La DBJ ne sera jamais impactée/modifiée par le logiciel
- Les données seront temporairement stockées dans une base de données afin que l'application puisse les traiter
- L'application utilisera Python 3 et un fichier CSV
- L'application prendra toujours en compte les nouvelles données
- L'application adaptera ses calculs en fonction des nouvelles données
- Chaque module sera testé avant d'être intégré à l'ensemble du programme

2. Méthodologie

En tenant compte du fait que le travail fourni sera véritablement implémenté et utilisé au sein d'un cadre professionnel, cela nécessite l'utilisation d'une méthodologie Scrum. Ladite méthodologie a été employée jusqu'à un certain stade où il n'a malheureusement pas été possible de conserver un dialogue continu avec le client. A partir du mois de Mars, la connexion avec l'entreprise qui gère le système informatique de Joassin s'est en effet rompue (voir mail envoyé en annexe). Dans la mesure où j'avais une partie des données à ma disposition, j'ai choisi d'utiliser la méthodologie Scrum Solo. En conservant mes timings préétablis, j'ai tenté de fonctionner par petit sprint d'une semaine.

Toutes mes heures de travail ont été comptabilisées via le site Clockify afin d'avoir une image globale des Sprints et du travail fourni.

[2] DBJ – DataBaseJoassin – la base de données principale de l'entreprise, celle qui contient l'intégralité des informations conservées par l'entreprise

3. Contraintes

i. Coûts

Suite à mon analyse, les coûts engendrés seront inexistants pour la simple raison qu'aucune pièce ni licence ne devra être achetée pour la bonne réalisation du projet.

ii. Stockage

Car l'application implique l'utilisation d'une machine virtuelle, un stockage d'au moins 60Go est conseillé.

iii. Délais

Le délai final pour la réalisation de l'application sera le mois d'aout 2020, lors de la fin de ma thèse de Travail de Fin d'Etudes.

iv. Autres contraintes

Comme l'entreprise Joassin respecte les règles RGPD[3], l'application ainsi que la base de données principale mais aussi la base de données propre doivent être parfaitement sécurisées dans le transfert des données. De plus, toute information qui se trouve dans les DB doit rester confidentielle.

4. Analyse de la problématique

Je me suis aperçu en analysant le système déjà mis en place avec les degrés jour que malgré sa logique qui devait être bonne, son implémentation a probablement dû rencontrer quelques soucis. Le problème est que malgré la bonne utilisation de l'application existante, le projet présente une maigre fiabilité de 18%. Cette dernière a été observée lorsque l'application indiquer qu'il fallait contacter les clients alors qu'ils n'avaient absolument pas besoin que leur cuve soient remplie.

[3] « GPD est l'acronyme de Règlement Général sur la Protection des Données »… « Il s'agit du Règlement (UE) 2016/679 du Parlement européen et du Conseil du 27 avril 2016 relatif à la protection des personnes physiques à l'égard du traitement des données à caractère personnel et à la libre circulation de ces données, et abrogeant la directive 95/46/CE. »- Baumann. A. Définition de RGPD. Dictionnaire du droit privé. https://www.dictionnaire-juridique.com/definition/rgpd.php

Le PDG de l'entreprise, monsieur Gonze, a suite à cela pris la décision de s'en remettre à un système plus complexe, prenant en compte davantage de valeurs que la simple variation de température. Malheureusement, il n'avait pas anticipé le fait que l'entreprise informatique avec laquelle il collaborait allait lui tenir rigueur de son manque de confiance quant au travail qu'elle avait effectué sur l'application.

Le système de prédiction est l'un des systèmes les plus complexes au monde, et fait partie intégrante de bien des épisodes de notre vie. Prédire la tendance en interprétant ce qui semble chaotique a toujours été un sujet attrayant pour les chercheurs.

Dans ce projet, nous faisons recours à différentes techniques d'apprentissage automatique, comme :

- le réseau neuronal
- la régression logistique
- l'arbre décisionnel
- la machine vectorielle de support

dans le but de prédire si les clients de l'entreprise Joassin Mazout vont tomber en panne dans la semaine qui suit l'alerte. Cette analyse se basera sur un an de données fournies par le client, ce qui équivaut à un total de 26 000 entrées dans notre fichier de stockage. La performance de l'implémentation a été analysée sous différents algorithmes en comparant la précision moyenne parmi l'ensemble de test.

Suite aux tests effectués sur lesquels nous nous pencherons plus tard, j'ai décidé de laisser l'algorithme le plus performant (le réseau neuronal) fonctionner avec l'application. Le but de tester et comprendre le fonctionnement de plusieurs algorithmes était d'offrir le meilleur taux de fiabilité au client.

i. Introduction analyse

Premièrement, je me suis tourné vers une solution existante que j'ai en vain essayé de faire évoluer : celle des degrés-jours de M. Vancutsem. Celle-ci ayant un taux de réussite de seulement 18% et en ne voyant aucun moyen de l'améliorer d'une quelconque manière, je me suis par la suite orienté vers un système de machine learning.

I. Technologies pour l'application

Python

J’ai choisi de programmer en Python afin de bénéficier de sa riche pile technologique et de son vaste ensemble de bibliothèques traitant de l'intelligence artificielle et l'apprentissage automatique. Ces avantages réduisent le temps de développement de manière conséquente.

Source – dans la bibliographie

Scikit-learn

Scikit-learn est une bibliothèque Python que j’ai dédiée au machine learning car elle regroupe toutes les fonctions dont j’ai eu besoin dans le cadre de ce projet :

Source – dans la bibliographie


```
from, sklearn, import, svm # SVM
from, sklearn, import, linear_model # régression
logistique
from, sklearn, import, tree # arbre décisionnel
from, sklearn.metrics, import, *
from, sklearn.neural_network, import, MLPClassifier
```

Numpy

Utile à la gestion des collections de données que l’application devait importer à partir du fichier CSV, Numpy est une bibliothèque Python qui s’est avérée fort utile dans ce champ de compétences :

Source – dans la bibliographie


```
From, sklearn, import, svm # SVM
from, sklearn, import, linear_model # 6egression
logistique
from, sklearn, import, tree # arbre décisionnel
from, sklearn.metrics, import, *
from, sklearn.neural_network, import, MLPClassifier
```

Pandas

Pandas est une bibliothèque Python à laquelle j’ai fait recours pour la gestion des collections de

Source – dans la bibliographie

données que l'application devait importer à partir du fichier CSV.

```
    """ Pendant les 4 lignes qui suivent, nous allon
s écrire - lire dans le fichier csv """
    # Create a dataframe from csv
    df = pd.read_csv('data.csv', delimiter=',')
    # User list comprehension to create a list of li
sts from Dataframe rows
    list_of_rows = [list(row) for row in df.values]
    # Insert Column names as first list in list of l
ists
    list_of_rows.insert(0, df.columns.to_list())
    # Print list of lists i.e. rows
    print(list_of_rows[-1])
```

Matplotlib

Textblob

Seaborn

A été utilisé afin d'exposer les graphiques illustrant les statistiques créées par **la régression logistique**. Cette bibliothèque s'est montrée particulièrement utile pour faciliter l'amélioration de cette régression logistique mais aussi la compréhension de son fonctionnement.

II. Hardware

Quant au matériel utilisé lors de la réalisation du projet, je suis équipé d'un portable doté d'un Intel i7 7820HQ et 16Go de mémoire vive. Ma carte graphique intégrée n'étant pas suffisamment puissante que pour pourvoir utiliser la technologie CUDA, j'ai été contraint d'utiliser les performances de mon processeur et de ma mémoire vive afin de prévoir les pannes. Cela explique notamment pourquoi j'ai choisi d'éviter le réseau neuronal au plus que possible ; j'avais en effet conscience que ce dernier nécessiterait une quantité de ressources considérable. En tenant compte les résultats des autres méthodes, j'ai tout de même dû implémenter un réseau de MPL. En résultent des heures d'attentes avec mon PC, ce dernier chauffant assez que pour alimenter un petit kot…

Nom	Statut	99% Processeur	61% Mémoire	0% Disque	0% Réseau	60% Processeur grap...
Visual Studio Code (4)		77,9%	2 868,5 Mo	0,1 Mo/s	0 Mbits/s	53,0%
Python (32 bits)		8,3%	2 485,7 Mo	0,1 Mo/s	0 Mbits/s	0%

III. Pourquoi le Machine Learning et l'intelligence artificielle ?

L'intelligence artificielle a toujours été un domaine passionnant pour les chercheurs et les entreprises. Anticiper sa trajectoire avec précision est chose difficile en raison de la multitude d'informations disponible sur son sujet. Une bonne compréhension des relations avance-retard parmi de nombreux facteurs environnants, saisir l'importance de ces relations et apprendre où les variables sont les plus importantes à surveiller, en les considérant comme des signaux qui prédisent les mouvements du marché, sont des éléments nécessaires pour anticiper les tendances. Les chercheurs en la matière ont mis au point de nombreuses bases théoriques et mathématiques, et ont développé beaucoup de méthodes dans le but d'anticiper le marché à l'aide de la technologie informatique actuelle. Parmi ces méthodes, les techniques d'apprentissage automatique sont très populaires grâce à leur capacité d'identifier les tendances, du comportement des clients et de l'énorme quantité de données qui régissent l'action sous-jacente et dynamique des achats.

Le modèle utilisé dans ce travail de fin d'études fait recours à des techniques d'apprentissage automatique robustes telles que Réseau neuronal, la régression logistique, l'arbre décisionnel. Il prend en charge les algorithmes de Support Vector Machine afin de prédire les éventuelles pannes à venir sur une période d'une semaine.

Afin de former ledit modèle, j'ai collecté des données de l'entreprise stockées dans une base de données requises pour la comptabilité. 75% des données obtenues ont été utilisées pour la création de mon modèle, et les 25% restants pour les tests.

Les précisions obtenues étaient comparées entre chaque algorithme sous différents facteurs. La méthodologie et les algorithmes d'apprentissage automatique utilisés pour résoudre ce problème sont mentionnés et explicités dans la partie suivante de ce rapport, où la manière avec laquelle les caractéristiques sont extraites des données ainsi que les résultats analysés pour obtenir des conclusions vous y seront détaillés.

Par la suite, nous nous pencherons sur les autres améliorations pouvant accroître le taux de précision des prédictions et par extension, la satisfaction du client.

ii. Les Données

Comme indiqué, l'ensemble de données utilisé est constitué de données historiques relatives aux stocks des commandes de mazout de

l’entreprise sur la dernière année. L’entreprise qui s’occupe actuellement du système informatique de Joassin ne m’a malheureusement pas fourni plus de données.

Plus précisément, chaque entrée dans l'historique des données comprend les éléments suivants :

• Date (string)
• Quantite(flottant)
• Prix (flottant)
• Capacite_citerne(flottant)
• Actif (entier)
• Temperature (entier)
• Produit (string)
• ID (entier)

Explications de données :

- Date - date de l’entrée dans le stockage
- Quantité – la quantité avec laquelle la citerne a été remplie lors de la dernière intervention
- Prix unitaire – Le montant de la dernière transaction
- Capacité, citerne(flottant) – la capacité que la citerne du client peut emmagasiner
- Actif – si le client a à nouveau établi un contact avec l’entreprise dans les derniers 3 mois
- Produit – le produit commandé : mazout ou mazout+ gaz
- ID – identification du client dans la base de données de l’entreprise

L'ensemble de données historiques était collecté avec le logiciel WinDev à l'aide de la plateforme créée par le gérant du système informatique.

La température a été ajoutée manuellement car l’entreprise ne m’a, comme énoncé précédemment, jamais fourni les données pourtant sollicitées au préalable. Si nous disposons de ces informations, c’est parce qu’elles ont été fournies gratuitement par l’IRM (Institut Météorologique Belge) pour ce travail uniquement.

La prévision des pannes a été réalisée en utilisant les caractéristiques susmentionnées, et les algorithmes ci-dessous ont été développés à l'aide de sci-kit learn library. Nous divisons l'ensemble de données en deux parties, à savoir 75% pour la formation et 25% pour tester les implémentations.

iii. Méthodes utilisées

Au vu de la complicité des différents algorithmes utilisés, j'inaugurerai cette partie du rapport par une introduction technique du perceptron multicouche, suivie d'explications simplifiées. Chacune sera accompagnée d'un exemple de code en python afin d'au mieux visualiser les divers éléments logiques.

I. Perceptron multicouche

Un réseau de neurones artificiels (ANN) est **un modèle de calcul** basé sur la structure et les fonctions des réseaux de neurones. Les informations qui circulent à travers le réseau impactent la structure de l'ANN, car un réseau neuronal change en fonction de cette entrée et sortie. Chaque unité neuronale individuelle a pour but d'additionner et donc de combiner les valeurs de toutes ses entrées ensemble.

Un perceptron multicouche (MLP) est **un modèle de réseau neuronal** artificiel fonctionnant par anticipation et qui combine des ensembles de données d'entrée et un ensemble de sorties adéquates. Le MLP est composé de plusieurs couches de nœuds, chaque couche étant entièrement connectée à la suivante. Chacun des nœuds est un neurone (ou un élément de traitement) ayant une fonction d'activation (à l'exception des nœuds d'entrée). Après une couche d'entrée basse, il y a généralement un nombre indéterminé de couches intermédiaires ou cachées, précédant une couche de sortie. Si les neurones d'une couche sont entièrement connectés aux neurones des couches adjacentes, il n'y a pas en revanche d'interconnexion au sein d'une même couche de nœuds.

Les poids synaptiques (PS) sont à l'origine du degré de corrélation entre les niveaux d'activité des neurones qu'ils connectent. Le poids synaptique, qui peut être défini dans les réseaux simulés et non biologiques, c'est le poids d'une relation entre deux neurones. Autrement dit, c'est la probabilité de voir s'établir une relation entre **un élément postsynaptique** (région présentant une densité élevée d'électrons) et **un élément présynaptique** (région convexe de l'axone où se trouve un neurotransmetteur contenu dans les vésicules synaptiques, et où l'on constate une activité particulièrement marquée).

Un vecteur d'entrée externe est fourni au réseau en fixant aux nœuds la valeur de la couche d'entrée. Pour les problèmes de classification conventionnels, pendant l'apprentissage, le nœud de sortie appropriée est fixé à l'état 1 tandis que les autres sont bloqués à l'état 0. Il s'agit de la sortie souhaitée.

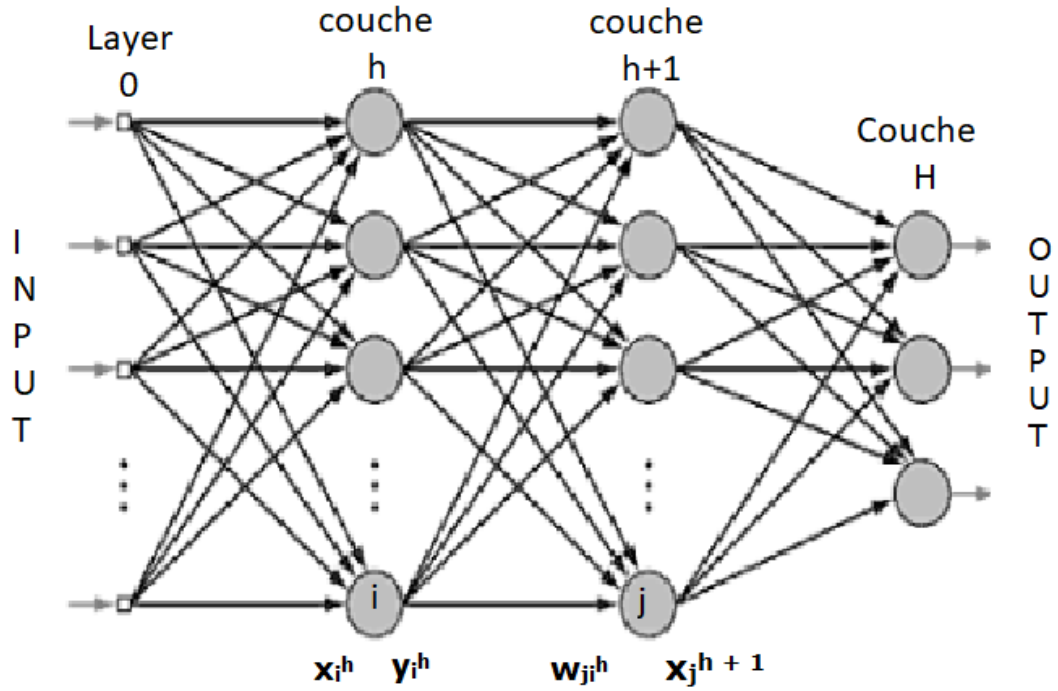


Fig.1 – Structure MLP avec 2 couches cachées – Source dans la bibliographie

Dans le réseau donné sur la figure 1, l'entrée totale, x_j^{h+1}, reçue par le neurone j dans la couche h+1 est donnée par la formule suivante :

$$\textbf{(1)} \qquad x_j^{h+1} = \sum_i y_i^h w_{ji}^h - \theta_y^{h+1}$$

- y_i^h est l'état du i ème neurone dans la hi ème couche précédente
- w_{ji}^h est le poids de la connexion du i ème neurone de la couche h au j ème neurone de la couche h + 1
- θj h + l est le seuil du j ème neurone de la couche h + l. Le seuil θj h + 1 peut être éliminé en donnant à l'unité j de la couche h + 1 une ligne d'entrée supplémentaire avec un niveau d'activité fixe de 1 et un poids de -θj h + 1.

La sortie d'un neurone dans n'importe quelle couche autre que la couche d'entrée (h> 0) est **une fonction non linéaire monotone** de son entrée totale, et est donnée par la formule suivante :

$$\textbf{(2)} \qquad y_j^h = \frac{1}{1+e-x_j^h}$$

Pour les nœuds de la couche d'entrée :

$$\textbf{(3)} \quad y_j^0 = x_j^0$$

Où xj 0 est la jième composante du vecteur d'entrée proche du niveau de la couche d'entrée. Tous les neurones d'une couche ont leurs états définis par les formules 1 et 2 en parallèle, et les différentes couches ont leurs états définis séquentiellement (en bas et en haut) les états des neurones dans le la couche de sortie H soient déterminées. La procédure d'apprentissage a pour objectif de déterminer les paramètres internes des unités, et ce sur la base de sa connaissance des entrées et des sorties. L'apprentissage consiste en somme à rechercher un très grand espace de paramètres et est donc rarement lent.

La rétropropagation est une méthode courante d'entraînement des réseaux de neurones artificiels qui répète une phase en deux mises à jour du cycle, de la propagation et du poids synaptique. Quand un vecteur d'entrée est présenté au réseau, il est propagé vers l'avant à travers le réseau, traversant couche après couche, jusqu'à ce qu'il atteigne finalement la couche de sortie. La sortie du réseau est ensuite comparée à la sortie souhaitée, en utilisant une fonction de perte, et une marge d'erreur la valeur est calculée pour chacun des neurones de la sortie couche. Les valeurs d'erreur sont alors à nouveau propagées vers l'arrière cette fois, à partir de la sortie, jusqu'à ce que chaque neurone ait une valeur d'erreur associée représentative de sa contribution à la sortie d'origine.

Les MLP sont largement utilisés pour la classification, la reconnaissance, la prédiction et l'approximation des formes. Le Perceptron multicouche a la capacité de résoudre des problèmes qui ne sont pas linéairement séparables.

Explication simplifiée :

Pour ce cas, j'ai entrepris de chercher et sélectionner un client ayant un comportement dit « parfait » : Actif et prévoyant, celui-ci commandait sans faute 90 % de sa cuve sans pour autant être en panne. S'il procède de la sorte, cela implique par déduction qu'il lui reste toujours 10% de cuve vide. Si l'on s'en réfère aux observations du personnel de Joassin, cela signifie qu'à partir de ce moment, en plein hiver, la cuve est vouée à être vide endéans une semaine.

Ensuite, j'ai dû compter le nombre de différents éléments dont j'allais faire utilisation et leur attribuer une valeur de base.

A l'intérieur du réseau neuronal existent de nombreux calculs destinés premièrement à dénicher les clients « parfaits », et deuxièmement à trouver une similitude invisible à l'œil humain à l'aide de l'algorithme existant. Grâce à cette similitude, ils pourront créer un pattern et appliquer ledit pattern aux clients suivants.

```
    """
    Solver
    Utilisé pour optimiser le poids des néurones.
    Pour des "petits" ensemble de données, 'lbfgs'
    peut calculer plus vite la bonne distribution des poids,
    donc l'application ira mieux.


    Activation
    La fonction d'activation d'un des noeuds, en foction de leur poids

    Alpha
    Paramétre de 'tuning'; il s'occupe des réglages précis de la la couche 2 c
achée
    """
    clf = MLPClassifier(solver='lbfgs', activation='relu', alpha=1e-5,
                        hidden_layer_sizes=(12, 8, 7), max_iter=300, random_st
ate=1)
    clf.fit(train_features, train_labels)
```

Afin d'éviter l'overfitting[4], j'ai utilisé plusieurs « features [5]», ceux-ci ayant fait en sorte que le MLP soit diversifié. A cela j'ai ajouté un paramètre « alpha » que j'ai dû changer plusieurs fois afin de tester sa fiabilité. Au plus la valeur d'alpha est grande, au plus la valeur des caractéristiques est proche de zéro. Les caractéristiques ne peuvent pas être laissées au maximum au risque d'entraîner l'overfitting.

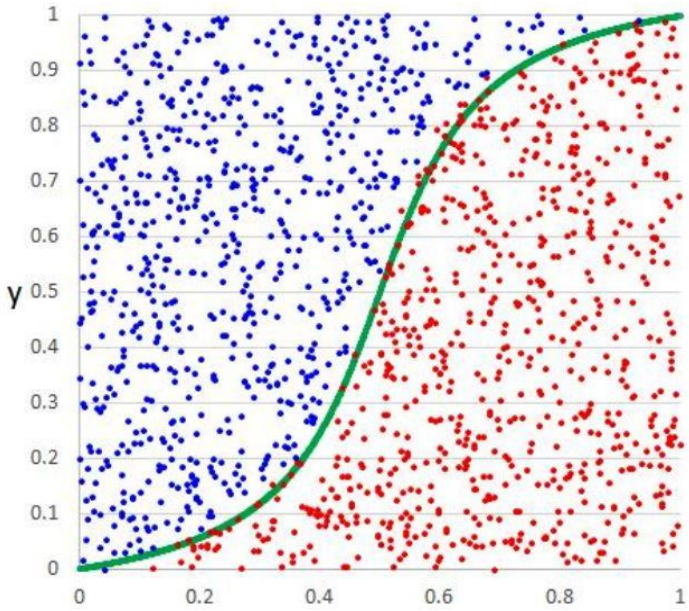
II. Régression Logistique

La régression logistique est utilisée dans le but de déterminer l'ampleur de relations entre les variables, pour modéliser les relations entre les variables, ou encore pour les prédictions basées sur Les modèles. La régression linéaire simple ou linéaire multiple est applicable lorsque la relation en question s'avère être linéaire. Cependant, de nombreuses techniques non linéaires peuvent être utilisées pour obtenir une régression plus précise si la relation entre les variables n'est pas linéaire.

La régression logistique est favorisée dans le cas où la variable de réponse ne peut prendre que des valeurs binaires (oui ou non). Sur le marché de mazout, le prix de prédiction et la tendance du prix du lendemain peuvent être catégorisés en deux classes, à savoir si le prix est croissant

[4] Le fait qu'une fonction ne rentre pas dans son espace dédié d'entrainement. C'est lorsque le système de ML apprend des réponses par cœur. Il ne calcule plus, il utilise le même exemple chaque fois.
[5] Des patterns qui peuvent être informatifs, discriminants et indépendant. En fonction de leur importance, les données de base deviennent des features de tel ou tel type.

ou décroissant, pour que la régression logistique puisse être utilisée. Nous utilisons la même logique pour les températures et les quantités relatives aux cuves.



● les faux échantillons
● les vrais échantillons
Fig.2 – graphique créé avec seaborn

L'importance de la régression logistique peut être évaluée par **le test de vraisemblance logarithmique**, en donnant comme test la statistique de Wald[6].

Le modèle de régression logistique est représenté par :

$$\textbf{(4)}\quad p = \frac{exp(c_0+c_1x_1+c_2x_2+\cdots+c_kx_k)}{1+exp(c_0+c_1x_1+c_2x_2+\cdots+c_kx_k)}$$

Pour parvenir à prédire la tendance de vidange de la cuve, le prix du mazout et la température, devraient avoir une tendance à la hausse si la valeur de la régression logistique pi (4) est proche de 0 ou est égale à 0. Dans le cas contraire, la date de vidange s'éloignera, car elle aura la tendance à ne plus s'approcher de son point 0 (moment où le client appelle).

En tenant compte, sur base de la figure 2, de la manière avec laquelle la régression logistique pourrait classer les données, nous avons sélectionné un ensemble de caractéristiques (température, prix) qui, à partir des informations contenues dans notre historique, permettent aux

[6] « Chaque fois que nous avons une relation au sein des ou entre les éléments de données qui peuvent être exprimées comme un modèle statistique avec des paramètres à estimer, et tout cela à partir d'un échantillon, le test de Wald peut être utilisé pour « tester la vraie valeur du paramètre » - WikiPedia

points de données d'être étroitement regroupés par caractéristiques. Cela permet la séparabilité.

Avec deux données utilisées seulement, l'algorithme est capable d'obtenir des résultats très poussés bien que sa fiabilité ne s'élève qu'à 80,75% sans jamais dépasser ce taux. Le souci est que les prévisions qu'il effectue ne sont valables que pour un laps de temps très court, dû au fait qu'il interprète toujours les données du jour suivant et pas au-delà. A cause de cela, il ne pourra pas être utilisé en pratique pour l'entreprise.

III. Machine à vecteurs de support avec noyau RBF

Une machine à vecteurs de support (SVM) est un classificateur défini par un hyperplan[7] de séparation. En d'autres termes, étant donné les données d'apprentissage étiquetées – les data du CSV – (apprentissage supervisé), l'algorithme produit un hyperplan optimal employé pour catégoriser l'ensemble de données de test. Tout hyperplan peut être écrit tel que l'ensemble des points x :

$$\textbf{(5)} \quad \vec{w} \cdot \vec{x} - b = 0$$

Si SVM est certes un classificateur linéaire binaire ne recourant pas aux calculs de probabilité, il permet la classification linéaire et peut efficacement effectuer une classification non linéaire en utilisant ce que l'on appelle « l'astuce du noyau », mappant implicitement leurs entrées dans des espaces de caractéristiques de haute dimension. Ce type d'implémentation est appelé SVM de fonction de base radiale (RBF). Le noyau RBF sur deux échantillons x et x ', représenté comme des vecteurs de caractéristiques dans un certain espace d'entrée, est défini comme

$$\textbf{(6)} \quad K(x, x') = exp\left(-\frac{\|x - x'\|^2}{2a^2}\right)$$

Basé sur la valeur de σ, l'espace des fonctionnalités du noyau a un nombre infini de dimensions.

[7] Estimation du modèle de régression multiple par la méthode des moindres carrés. – « permet de comparer des données expérimentales, généralement entachées d'erreurs de mesure, à un modèle mathématique censé décrire ces données » - Wikipédia

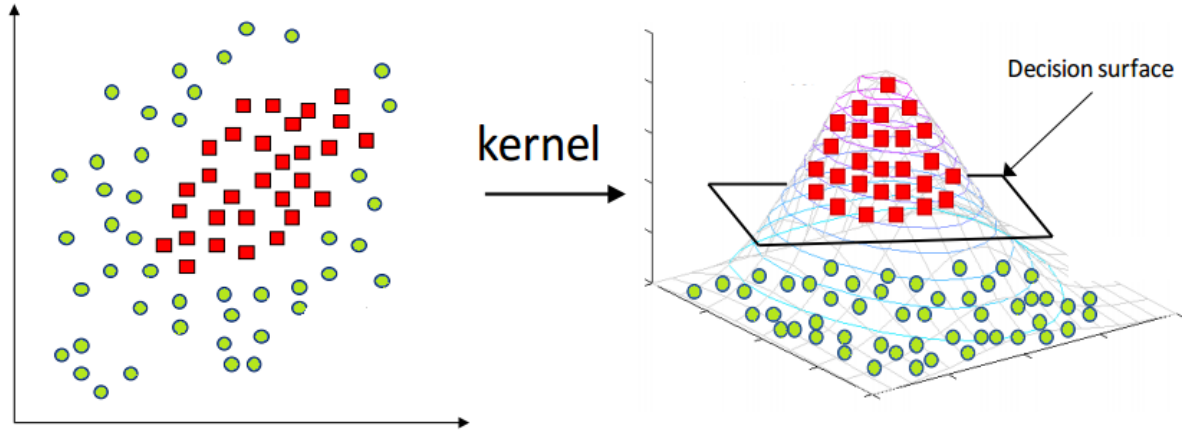


Fig.3 – Avec ou sans Kernel – Source dans la bibliographie

Comme démontré sur la figure 3, les performances de SVM avec la fonction de base radiale sont distinctivement supérieures au SVM linéaire. Étant donné que la prévision de pannes implique une grande quantité de données non linéaires, j'en conclus que SVM avec noyau RBF se voit être le meilleur choix pour la classification.

Explication simplifiée

La régression linéaire a pour fonction de prédire la valeur numérique d'une variable ou d'un ensemble de variables. Cependant, un inconvénient majeur des machines linéaires à marges rigides est leur incapacité à traiter le cas où l'ensemble de points d'entraînement n'est pas linéairement séparable. Pour contrer ce défaut, les machines à marges souples ont étés introduites en autorisant la mauvaise classification de quelques exemples d'entrainement.

Une fois la prédiction pour un client x effectuée, je la compare à la prévision d'un client « parfait » et observe les similitudes. Grâce à la Constante C, j'indique au système à quel point il doit accepter les différences entre le cas du client x en cours de traitement et celui du client « parfait ». « C » est un paramètre qui contrôle la pénalisation des erreurs de classification.

```
    ################################ Logistic Regression ####################
############
    max_acc = 0
    max_c = 0
    max_g = 0
    C_Val = [10e-7]
    for c in C_Val:
        clf = linear_model.LogisticRegression(
            C=c).fit(train_features, train_labels)
        curr_acc = accuracy_score(test_labels, clf.predict(test_features))
```

IV. Arbres de décision

Les arbres de décision sont des graphiques ayant la particularité d'être conçus sous-forme d'arbres dans lesquels chaque nœud représente une règle et chaque arête sortante, une valeur possible de cette règle. L'algorithme divise à plusieurs reprises l'ensemble de données en fonction d'un critère qui maximise la séparation des données, résultant en une structure arborescente.

Les nœuds (les feuilles) constituent dans leur ensemble une parmi plusieurs classes candidates possibles. Les arbres de décision sont principalement utilisés pour la classification. Le coût de l'utilisation de ceux-ci dans les prédictions est relatif au journal du nombre de points de données utilisés pour entraîner l'arbre (dans notre cas le fichier .CSV).

Dans la figure ci-dessous (Fig4[8]), nous avons l'occasion d'observer à quoi ressemblerait idéalement un arbre de décision permettant de prédire les pannes à venir des clients. Il établirait des règles sur chaque nœud en fonction de différents paramètres dans le but de classer les données et attribuer une étiquette au nœud en question. Un paramètre appelé « min sample split » est utilisé par le prédicteur, ce dernier indiquant le nombre d'échantillons minimum dont l'arbre de décision a besoin pour fractionner un nœud.

```
############################ Decision Trees ##############################
clf = tree.DecisionTreeClassifier(
    min_samples_split=80, min_impurity_split=1e-5)
clf.fit(train_features, train_labels)
```

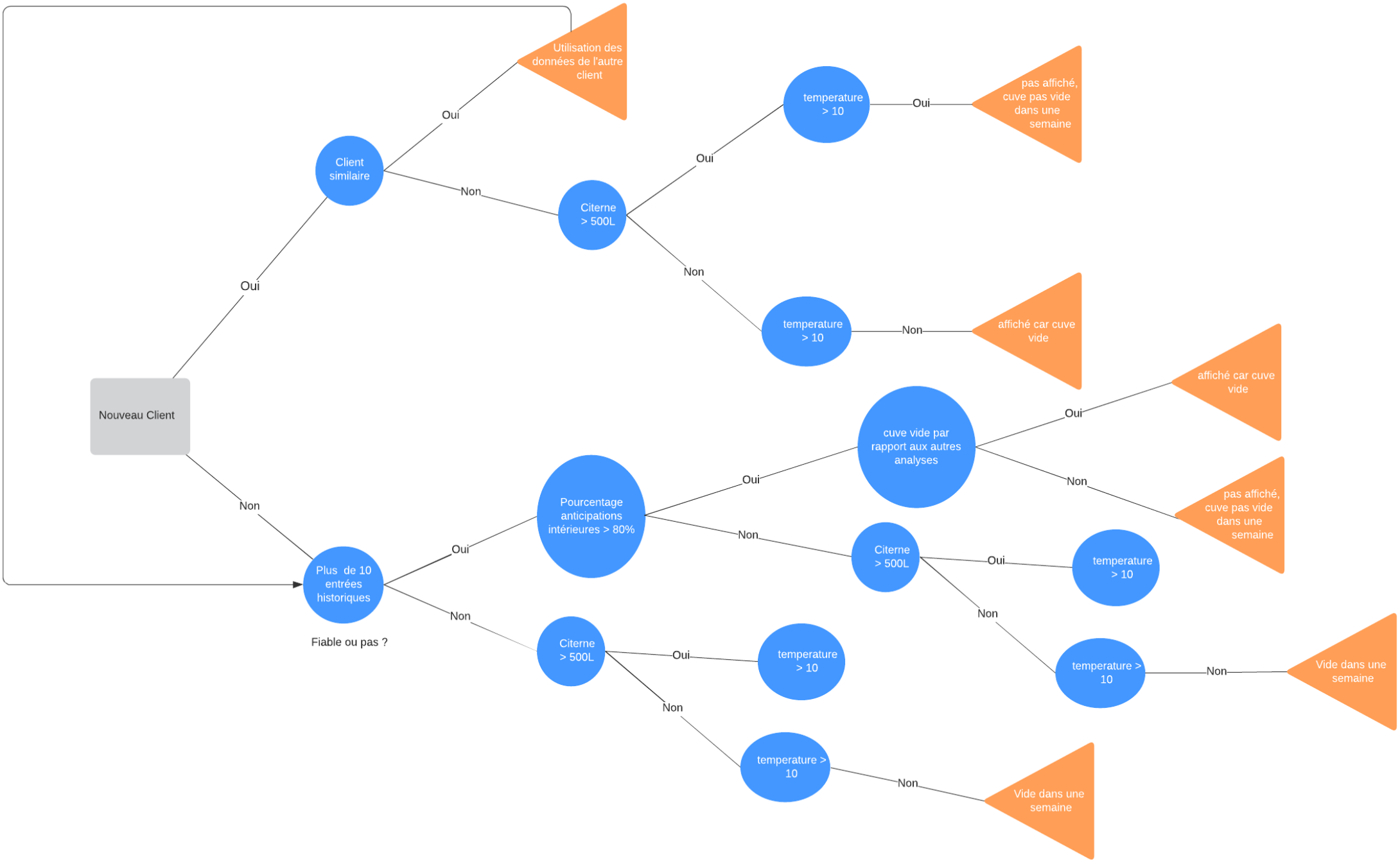
Afin de créer un arbre de plus en plus fiable, j'ai modifié petit à petit les min_samples_split et min_impurity_split. A chaque fois, j'ai remarqué les changements et suis par la suite parvenu à obtenir de bonnes performances. Malheureusement, la solution de l'arbre décisionnel reste la moins utilisable en terme de fiabilité, avec un pourcentage de prévision de seulement 77%.

Un inconvénient majeur des arbres décisionnels est qu'ils peuvent créer des arbres trop complexes qui généralisent de manière inadéquate les données, ce qui peut entraîner un surajustement ou un sous-ajustement de celles-ci. Un autre inconvénient réside dans le fait que les variables continues sont implicitement fractionnées à chaque fois qu'elles perdent les informations nécessaires en cours de processus.

[8] Fig 4 – a été créée via le site web LucidChart

Exemple d'Arbre de décision

Mirica Constantin | August 17, 2020



iv. Sélection et traitement des fonctionnalités

Le groupe initial d'ensembles de fonctionnalités comprenait la recherche différente entre chaque attribut de l'action, en consultant l'historique, par rapport à l'ID de chaque client. J'ai progressivement ajouté d'autres fonctionnalités jusqu'à obtenir une précision davantage optimale.

J'ai donc testé les fonctionnalités suivantes :

- Ouvrir le fichier data .CSV
- Recherche des nouveaux enregistrements
- Passer ses données dans le réseau neuronal de multiples perceptrons
- Trouver des clients similaires sur base de la température, citerne, prix du mazout, et la date du dernier remplissage (s'il s'agit d'un client existant)
- Faire passer les cas similaires dans le réseau neuronal de perceptrons afin de l'adapter
- Afficher les cas qui sont voués à tomber en panne car la citerne contient moins de 20% de combustible restant

v. Discussion sur les résultats

La précision obtenue à partir des différents algorithmes est représentée sur la figure 6. La plus haute précision :

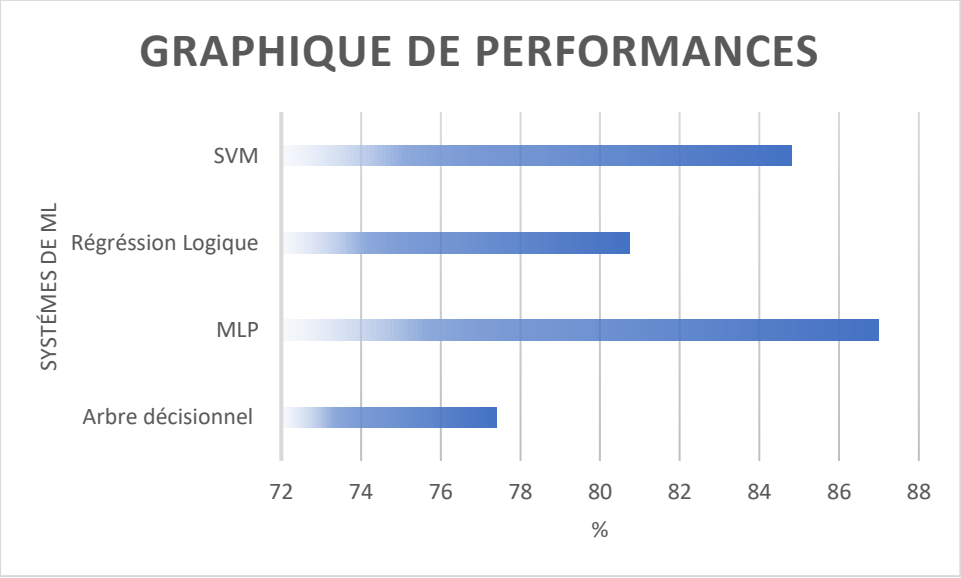


Fig. 7 – créée avec Excel à base des informations récupérées

Les performances de l'arbre de décision sont, comme démontré dans le graphique ci-dessus, relativement faibles. Le nombre minimum d'échantillons à fractionner le paramètre (min sample split) dans l'arbre de décision, est celui qui détermine le nombre d'échantillons dont l'algorithme doit tenir compte avant de diviser un nœud.

La précision obtenue à l'aide de la régression logistique s'élève à 80,75%. Puisque les données classifiables sont non linéaires, on s'attend à ce que SVM-RBF fonctionne de manière plus performante que la régression logistique, pour qui la précision s'élève, elle, à 84,8%. Les performances SVM-RBF dépendent de la valeur C et de la valeur gamma. La valeur C. est une pénalité attribuée pour une mauvaise classification (le SVM n'a foncièrement pas la même classification de données que la régression logistique). Un grand C prodigue une faible tendance, variance élevée et vice versa. Le paramètre gamma définit dans quelle mesure l'influence d'un seul exemple d'entraînement a un effet sur l'ensemble, avec des valeurs faibles signifiant que le système peut aller plus « loin », et des valeurs élevées signifiant « fermer » . Il n'y a donc plus aucune boucle nécessaire pour établir des corrélations.

vi. Finalité du processus d'analyse

Dans cette analyse, nous avons parcouru l'application de différents algorithmes d'apprentissage automatique ayant pour objectif la prédiction des pannes. Nous avons explicité la théorie derrière le perceptron multicouche, la régression logistique, SVM avec noyau RBF et algorithme d'arbre de décision. Comme le montre la figure 6 (avec MLP donnant le score de précision le plus élevé), les résultats obtenus dans tous ces cas étaient relativement précis.

Malgré le fait qu'un client puisse avoir beaucoup de facteurs, il est possible de tous les collecter, de fournir un modèle d'apprentissage automatique, et d'ajouter des informations supplémentaires. Cela aidera d'autant plus à obtenir un score de précision élevé. Ainsi, nous pouvons utiliser des algorithmes d'apprentissage automatique pour prédire les pannes des clients de l'entreprise Joassin Mazout avec précision. Ce qui était auparavant considéré comme presque imprévisible (prévisible à 18%) est maintenant prévisible à 85%.

5. Historique du projet

i. Planification

- 27.09.2019
  - Rencontre P.D.G.Joassin – Julien Gonze
  - Retrouver le problème du point de vue du PDG
  - Comprendre le core business de l'entreprise

- 10.10.2019
  - Rencontre représentant IT de l'entreprise
  - Comprendre le fonctionnement actuel du point de vue IT de l'entreprise
  - Trouver le problème du point de vue informatique et envisager des pistes

- 13.11.2019
  - Rencontre représentant IT de l'entreprise
  - Trouver le moyen de connexion à la DB
  - Accord sur les données à utiliser

- 21.01.2020
  - Récupération des données en format Excel

- 22.01.2020
  - Défense technique - sujet
- 03.02.2020
  - Demande les données relatives à la température à M. Bernard – responsable clients
- 21.02.2020
  - Demande du reste des informations par mail à S. Vancutsem

Plus de contact avec l'entreprise externe depuis.

ii. Evolution

Une fois la défense technique passée, j'ai essayé de reprendre contact avec le responsable informatique externe de l'entreprise. Je lui ai donc envoyé 2 mails sur ses 2 boites mails et n'ai cependant jamais eu de retour de sa part. Mon planning a de ce fait été perturbé, car je devais malgré cela trouver toutes les températures de la région de Namur sans pour autant dépenser une petite fortune.

Suite à cet évènement, je suis entré en contact avec l'Institut Météorologique Belge. Après avoir expliqué la situation quelque-peu critique dans laquelle je me trouvais, les informations dont j'avais besoin m'ont finalement été fournies.

iii. Git

Durant l'intégralité du TFE, j'ai travaillé avec GitHub afin d'avoir une sauvegarde permanente et un outil de versioning. En tenant compte du fait que j'étais seul à travailler sur l'application, je n'ai jamais éprouvé le besoin de faire des pull requests afin de vérifier mon propre code.

iv. Clockify

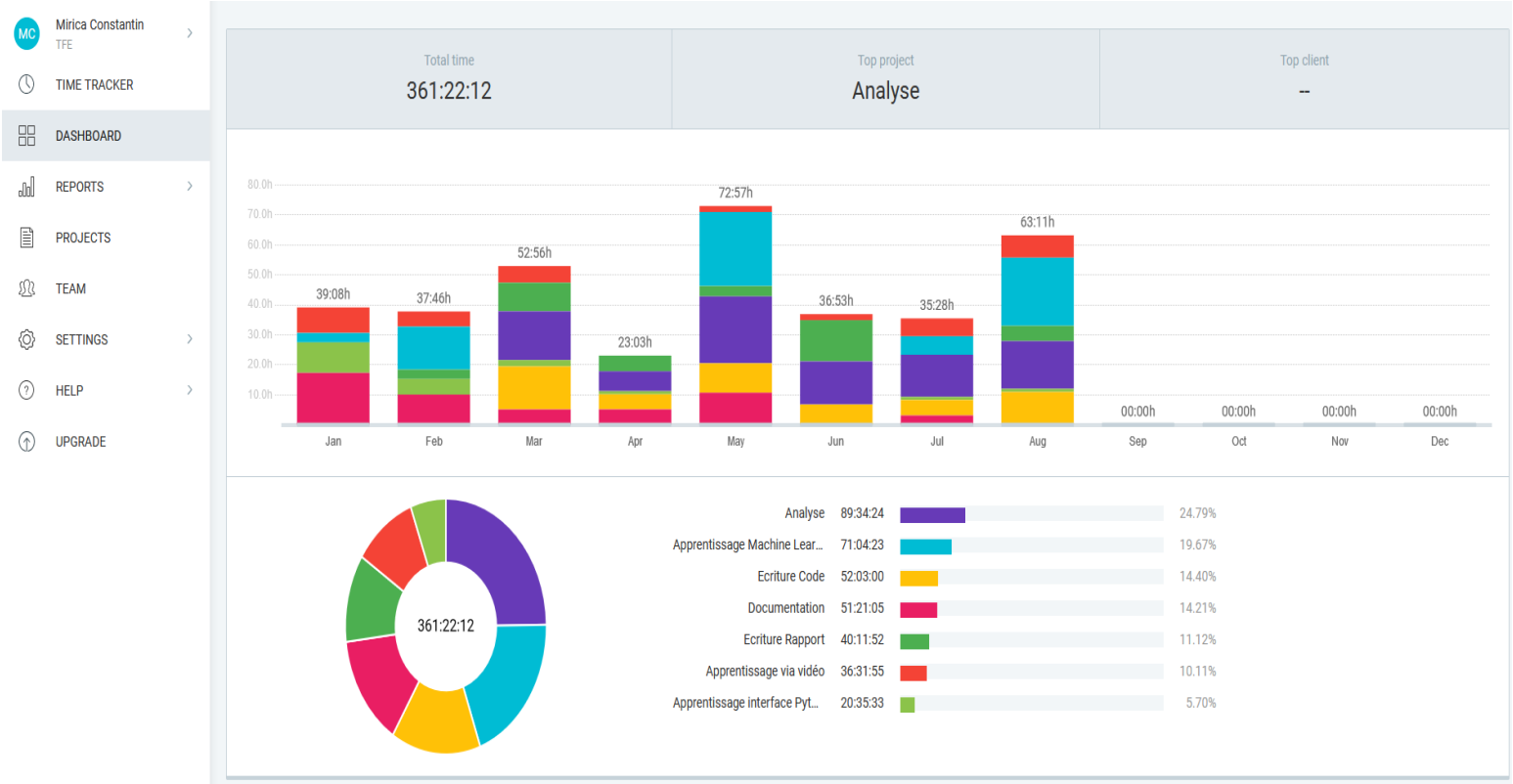
Afin de comptabiliser et visualiser aisément le temps passé sur ce projet, j'ai utilisé le site clockify. Le but était de voir où je me situais par rapport à la charge de travail demandée, mais aussi voir ce qu'il me restait à faire pour finaliser mon projet.

Avec un total de plus de 360h, mon travail se partage en plusieurs parties, parmi lesquelles on compte :

- La documentation (50h)
- L'apprentissage machine learning (70h)
- L'apprentissage interface graphique Python (20h)
- L'apprentissage via Vidéo (35h)
- L'écriture rapport (40h)
- L'écriture code (50h)
- L'analyse (90h)

L'apprentissage a constitué une partie relativement conséquente du temps total consacré à ce projet, pour la simple et bonne raison que mes connaissances sur le sujet étaient nulles.

Outre l'apprentissage, le plus grand challenge a été celui de comprendre exactement comment utiliser les vastes ressources que je venais d'accumuler. Pour ce faire, j'ai dû procéder à une analyse très approfondie et créer le code au fur et à mesure que je m'occupais de l'analyse. Malheureusement, je n'ai pas tenu compte du temps pour l'interface, mais celui-ci doit être aux environs de 10h, apprentissage compris



6. Produit fini

Le produit fini est une application exécutable contenant l'ensemble du calcul créé par la ML et par les algorithmes utilisés. Une fois lancée, l'application désigne les clients susceptibles de tomber en panne dans la semaine suivant l'alerte.

Dans le cadre du TFE, l'interface aura également pour option celle d'ajouter des entrées afin de tester le calcul.

7. Sécurité

En tenant compte des concurrents de l'entreprise Joassin, cette application représente un atout majeur de démarchage ainsi qu'un service sans précèdent dans ce secteur en Belgique. C'est pour cela que l'entreprise tient particulièrement à sécuriser l'application autant que possible.

La solution la plus sécurisée et par extension la plus pertinente aux besoins de l'entreprise, est d'insérer l'application dans une machine virtuelle Windows ne possédant pas de connexion internet et qui partage exclusivement un seul dossier contenant le serveur de l'entreprise.

Pratiquement ?

- Mettre la machine virtuelle sur le serveur Windows préexistant

- Créer un service dans l'application de comptabilité qui va, chaque semaine, exporter un fichier .CSV dans un dossier partagé avec la machine virtuelle
- Se connecter à la machine virtuelle avec les crédenciales fournies
- Exécuter l'application prenant en compte les changements dans le fichier .CSV et qui affichera les ID des clients susceptibles de tomber en panne

La machine virtuelle sera aussi copiée sur le NAS de l'entreprise une fois par mois. On pourra y trouver 3 versions de la MV.

8. Conclusion

i. Personnelle

En conclusion, j'ai eu durant ce projet la possibilité de gérer des situations problématiques, reflétant ainsi de manière assez fidèle les aléas de la vie professionnelle. J'ai appris que malgré mes efforts, il y aura toujours une inconnue, un moment où les choses ne se dérouleront pas en la faveur de mon travail ou de mes objectifs. Le plus important a été de parvenir à finaliser ce projet, d'en sortir grandi et d'apprendre de mes expériences. Mes compétences ont été mises à rude épreuve car, je ne me suis pas aperçu immédiatement de la complexité mathématique des sujets dont j'allais devoir traiter. Suite à de nombreuses lectures, beaucoup de recherches et de sessions essais-erreur je suis arrivé à comprendre et à expliquer du mieux que je peux les complexes concepts mathématiques qui se cachent derrière la machine learning.

ii. Par rapport au cahier de charges

Si l'on se réfère au cahier de charges, l'application exécute tout ce qui est commandé par l'utilisateur. Le projet en soi a dépassé les attentes fixées car plusieurs systèmes de machine learning ont été testés et utilisés afin de parvenir à atteindre le meilleur résultat, qui n'est autre que le Multi Layer Perceptron.

Du point de vue de l'application et du stockage de données, une base de données ne s'est pas montrée nécessaire. L'application lit et écrit dans un fichier CSV qui se trouve dans la machine virtuelle. Ce même fichier est exporté par l'application comptable en WinDev, présent sur le serveur de l'entreprise.

iii. Points forts projet

Malgré son apparence simpliste, l'application offre des avantages notables :

- Taux de fiabilité élevé
- Calcul relativement rapide des nouveaux cas
- Adaptabilité – le système apprends en fonction des nouveaux cas

iv. Points faibles projet

Parmi les points faibles, il y a :

- Son aspect
- Le manque de données faisant en sorte que l'application reste à améliorer

v. Améliorations envisageables

Afin d'améliorer le système, j'ai pensé à des nombreuses solutions telles que :

- Envoi automatique des mails aux clients – en créant un service de mailing
- Un service de robot, qui appelle les clients via VoIp, avec des messages pré-enregistrés
- En ayant plus de données, créer une application plus précise (Il suffit d'ajouter toute la liste de clients de Joassin et les calculs s'effectueront automatiquement)
- Plus de techniques d'exploration de données

vi. Plan pour le futur

Le futur reste une inconnue et ce même pour l'intelligence artificielle, je ne suis donc pas suffisamment qualifié pour tenter de le prédire …

En ce qui concerne l'application en revanche, j'ai pour projet de la présenter à mes parents car je suis satisfait d'avoir mené à bien un projet qui demande tant de « matière grise ».

Par rapport à ma vie professionnelle, je commence le 3 septembre à travailler chez Deloitte en tant que Développeur Java EE, et j'espère que cette dernière année riche en complexité se terminera sans dégâts notoires.

L'année prochaine, je compte entamer et -je l'espère- obtenir un Master en cours du soir en Sciences de Travail à ULB afin d'au mieux comprendre le fonctionnement technique du travail en entreprise et de pouvoir par la suite être reconnu en tant que bon leader.

9. Bibliographie

Barry P., Griffiths D. (2009). Head First: Programming. Edition O'Reilly. United States of America. Consulté en ligne.
http://index-of.es/Python/A.learner's.guide.to.programming.using.the.Python.language.David.Griffiths.2009.pdf

Burkov A. (2019). The Hundred-Page Machine Learning Book. Consulté en ligne.
https://github.com/tirthajyoti/Papers-Literature-ML-DL-RL-AI/blob/master/General-Machine-Learning/The%20Hundred-Page%20Machine%20Learning%20Book%20by%20Andriy%20Burkov/Links%20to%20read%20the%20chapters%20online.md

Downey A.B. (2009). Python for Software Design. Edition Cambridge. Royaume-Uni. Consulté en ligne.
http://index-of.es/Python/Python.for.Software.Design.How.to.Think.Like.a.Computer.Scientist.Allen.Downey.2009.pdf

Fig1 - https://lh3.googleusercontent.com/SpBLZQIh-Dpmmgidkb-IpP8W3nBqy32AhXrbYkXe_DmJxLAvMdUhIVHzaPue6XlEQbsnfSV5zlLP-QtLAWkODRw1YPVcWyo9lZqvhvd7eMBMXLikcKmiRi_lmUqBUAP_SBDR-Wff
Consulté le 05.06.20

Edureka!. (2018). AI vs Machine Learning vs Deep Learning | Machine Learning Training with Python | Edureka. Consulté en ligne.
https://www.youtube.com/watch?v=WSbgixdC9g8
Consulté le 03.06.20

Edureka!. (2019). Artificial Intelligence with Python. Consulté en ligne.
https://www.youtube.com/watch?v=7O60HOZRLng
Consulté le 03.06.20

freeCodeCamp.org. (2018). Learn Python - Full Course for Beginners [Tutorial]. Consulté en ligne.
https://www.youtube.com/watch?v=rfscVS0vtbw
Consulté le 11.06.2020

Géron A. (2019). Hands-On Machine Learning With Scikit-Learn, Keras, and TensorFlow, 2nd edition. Edition O'reilly. Consulté en ligne.
https://github.com/Akramz/Hands-on-Machine-Learning-with-Scikit-Learn-Keras-and-TensorFlow

Consulté le 19.06.2020

Han. J, Kamber.M. (2006). The Morgan Kaufmann Series in Data Management Systems. Edition Elsevier. San Francisco. Consulté en ligne. https://mitmecsept.files.wordpress.com/2017/04/data-mining-concepts-and-techniques-2nd-edition-impressao.pdf
Consulté le 25.05.2020

Hetland M. Lie. (2008) . Beginning Python: From novice to professional. New-York. Edition Apress. Consulté en ligne.
https://www.academia.edu/39515710/Beginning_Python_From_Novice_to_Professional_Third_Edition
Jaworski M, Ziadé T. (2019). Expert Python Programming. Edition Packt. Consulté en ligne.
https://github.com/PacktPublishing/Expert-Python-Programming-Third-Edition
Consulté le 02.07.2020

PyCharm. Learning Python. Consulté en ligne.
https://docs.python-guide.org/intro/learning/
Consulté le 17.05.2020
Shaw Z. (2013). Learn Python the hard way: A very simple Introduction to the terrifyingly Beautiful World of Computers and Code. Editions Pearson Addison-Wesley. Consulté en ligne.
https://github.com/cenuno/learning_python3_the_hard_way/tree/master/Python
Consulté le 29.02.2020

Tech With Him. (2019). Python Machine Learning Tutorial #1- Introduction.
https://www.youtube.com/watch?v=ujTCoH21GlA
Consulté le 06.03. 2020

VanderPlas. J. (2016). Python Data Science Handboook. Edition O'Reilly. Consulté en ligne
https://tanthiamhuat.files.wordpress.com/2018/04/pythondatasciencehandbook.pdf
Consulté le 18.01.2020

Image Préface
https://bbvaopen4u.com/sites/default/files/img/embed/new/cibbva_modelo.png - consulté le 15.08.2020
Consulté le 24.07.2020

Image SVM avec et sans Kernel

https://miro.medium.com/max/838/1*gXvhD4IomaC9Jb37tzDUVg.png – consulté le 16.08.2020

Image Scikit-learn
https://upload.wikimedia.org/wikipedia/commons/thumb/0/05/Scikit_learn_logo_small.svg/1200px-Scikit_learn_logo_small.svg.png
consulté le 23.08.2020

Image NumPy
https://en.wikipedia.org/wiki/NumPy#/media/File:NumPy_logo_2020.svg
consulté le 23.08.2020

Image Pandas
https://numfocus.org/wp-content/uploads/2016/07/pandas-logo-300.png
consulté le 23.08.2020