UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À
L'UNIVERSITÉ DU QUÉBEC À TROIS-RIVIÈRES

COMME EXIGENCE PARTIELLE
DE LA MAÎTRISE EN MATHÉMATIQUES ET INFORMATIQUE
APPLIQUÉES

PAR
Abdou Khadir DIA

MODÈLES PRÉDICTIFS PAR APPRENTISSAGE AUTOMATIQUE POUR LA CLASSIFICATION ET LA RÉGRESSION : APPLICATION EN SCIENCE DES DONNÉES

JUILLET 2023

Résumé

L'application de l'apprentissage automatique dans le domaine industriel est de plus en plus fréquente avec l'avènement de nouvelles technologies de collecte de données. Les industries utilisent l'apprentissage automatique pour créer de la valeur à partir des grandes quantités de données qu'elles collectent tous les jours. Les méthodes d'apprentissage automatique sont classées en trois catégories : l'apprentissage automatique supervisé, l'apprentissage automatique non supervisé et l'apprentissage automatique semi-supervisé. Ce mémoire s'intéresse aux modèles d'apprentissage supervisé pour les problèmes de régression et l'apprentissage non supervisé appliqués sur les données de la corrosion d'un pipeline. Le Self Organising Maps (SOM) combiné avec la Classification Ascendante Hiérarchique (CAH) sont appliqués sur les données de corrosion pour étudier les différents niveaux de perte d'épaisseur d'un pipeline. Grâce à ces modèles, les différents points du pipeline ont été classifiés dans trois catégories différentes : Pas de perte d'épaisseur, Perte d'épaisseur moyenne, Perte d'épaisseur élevée.

Les modèles d'apprentissage automatique pour la régression ont permis de prédire la perte d'épaisseur du pipeline avec une erreur de prédiction la plus faible possible. Les modèles forêt aléatoire, machine à vecteur support, Gradient Boosting Machine (GBM), k-voisins les plus proches et perceptron multicouche sont utilisés pour la prédiction de la perte d'épaisseur du pipeline. La sélection des variables les plus pertinentes a été effectuée à l'aide de la méthode d'élimination récursive des paramètres et de l'analyse en composantes principales. Pour améliorer les résultats des modèles, l'optimisation des hyperparamètres avec la méthode Grid Search a été utilisée et a permis d'améliorer les résultats du modèle machine à vecteur support qui a le plus faible RMSE.

Abstract

The application of machine learning in industry is becoming more common with the advent of new data collection technologies. Industries are using machine learning to create value from the large amounts of data they collect every day. Machine learning methods can be classified into three categories: supervised machine learning, unsupervised machine learning and semi-supervised machine learning.

This thesis focuses on supervised learning models for regression problems and unsupervised learning applied to pipeline corrosion data. Self Organising Maps (SOM) combined with Hierarchical Ascending Classification (HAC) are applied on corrosion data to study the different levels of thickness loss of a pipeline. Using these models, the different points of the pipeline were classified into three different categories: No thickness loss, Medium thickness loss, High thickness loss.

Machine learning models for regression were used to predict the pipeline thickness loss with the lowest possible prediction error. Random forest, support vector machine, Gradient Boosting Machine (GBM), k-nearest neighbour and multilayer perceptron models are used for the prediction of pipeline thickness loss. The selection of the most relevant variables was performed using recursive parameter elimination and principal component analysis. To improve the results of the models, hyperparameter optimisation with the Grid Search method was used and improved the results of the support vector machine model which has the lowest RMSE.

Remerciements

Je remercie Dieu le Tout-Puissant de m'avoir donné la santé et la volonté d'entamer et de terminer ce travail.

Je tiens à adresser mes sincères remerciements à ma directrice de recherche Mme Nadia Ghazzali pour avoir accepté de diriger mon projet de recherche. Je la remercie également pour sa disponibilité, la correction du document, son appui financier et pédagogique et surtout ses judicieux conseils qui ont contribué à alimenter ma réflexion.

Je tiens à saisir cette occasion et adresser mes profonds remerciements et ma profonde reconnaissance à Dr Axel Gambou Bosca pour son aide incommensurable durant mes études de maitrise, ses conseils, ses commentaires et suggestions sur le document.

Mes remerciements vont également au Fonds de recherche du Québec – Nature et Technologies (FRQNT) et au Conseil de Recherches en Sciences Naturelles et en Génie du Canada (CRSNG) pour leur soutien financier, au Centre de Métallurgie du Québec (CMQ) et au Agnico Eagle Mine Goldex pour la disponibilité des données.

Je tiens à remercier ma femme, qui aussi a droit à toute mon appréciation.

Enfin et surtout, je remercie mes parents, ma famille pour leur soutien, encouragement et aide et toutes les personnes qui ont participé de près ou de loin à la rédaction de ce document.

Table des matières

Liste des tableaux

Table des figures

Liste des abréviations

API	American Petroleum Institute
BMU	Best Matching Unit
CAH	Classification Ascendante Hiérarchique
CART	Classification And Regression Tree
CMQ	Centre de Métallurgie du Québec
CPSO	Chaos Particle Swarm Optimization
DNN	Deep Neural Network
FFANN	Feed-Forward Artificial Neural Network
GBDT	Gradient Boosting Decision Tree
GBM	Gradient Boosting Machine
ID3	Iterative Dichotomiser 3
KNN	K-Nearest Neighbors
MARS	Multivariate Adaptive Regression Splines
NACE	National Association of Corrosion Engineers
PSO	Particle Swarm Optimisation
RF	Random Forest
RMSE	Root Mean Square Error
SOM	Self Organising Maps
SVM	Support Vector Machine
SVR	Support Vector Regression

Introduction

Les méthodes d'apprentissage automatique ont connu un grand succès ces dernières années. Elles sont utilisées actuellement dans presque tous les secteurs (administration publique, entreprises, institution etc..) pour des besoins différents relatifs à la création de la valeur à partir des données. La donnée est devenue ainsi une ressource puissante. C'est ainsi, que Agnico Eagle mine Goldex, une entreprise minière compte utiliser l'apprentissage automatique en collaboration avec le Centre de Métallurgie du Québec (CMQ) pour l'évaluation et la prévision de la corrosion du pipeline qu'il utilise dans le cadre de ses activités. En effet, Agnico Eagle mine Goldex utilise depuis août 2008 un pipeline de 24 km de long, dont 16,4 km en acier, et le reste en HDPE. L'entreprise se soucie de la fiabilité de ses installations, notamment le risque de défaillance des pipelines. Il faut rappeler que cette défaillance aura de grands impacts négatifs sur la santé et la sécurité des opérateurs, sur l'environnement et les communautés, et bien sûr économiques. L'une des causes majeures de la défaillance des pipelines est l'attaque par la corrosion.

Les pipelines sont utilisés pour transporter des produits chimiques tels que le pétrole et d'autres hydrocarbures liquides, le gaz naturel, de concentrés minéraux sur de longues distances. Ils jouent un rôle central dans le transport des produits depuis les champs de production jusqu'aux satellites où ils sont traités et transportés vers les raffineries [1]. Le transport par pipeline a pour caractéristiques d'être très efficace, peu coûteux et de passer par diverses conditions de travail [2]. Les pipelines sont la méthode la plus lucrative de transport des hydrocarbures gazeux et liquides. Il existe environ 3,5 millions de km d'oléoducs et de gazoducs dans le monde [3,4]. Lorsqu'un accident de pipeline se produit, il entraîne non seulement d'énormes pertes économiques, mais aussi des victimes et une pollution de l'environnement. Il est donc

important de créer des méthodologies et des technologies qui réduisent les fuites et les ruptures dans ces structures, en les maintenant dans des conditions de fonctionnement sûres.

Cependant, en raison de la présence des boues abrasives ou corrosives, les pipelines sont continuellement soumis à une corrosion interne qui entraîne la perte de l'épaisseur de la paroi des tuyaux. La corrosion est un phénomène très complexe basé sur la dégradation d'un matériau ou de ses propriétés en raison de sa réaction avec l'environnement [5]. Cette dégradation fait intervenir de multiples facteurs particules et variables. Il est généralement admis que les canalisations des installations doivent être inspectées pour détecter les dommages en service tels que la corrosion. La corrosion des pipelines a causé de nombreux problèmes aux opérateurs des compagnies pétrolières et gazières en raison des coûts associés à sa gestion. Les auteurs de [4] ont montré qu'au Canada, au Royaume-Uni, en Europe et aux États-Unis, l'une des principales raisons des défaillances des pipelines est la corrosion. L'European Gas Pipeline Incident Data Group a reconnu que les dommages causés par les mécanismes de corrosion ont augmenté ces dernières années, passant de 16,7 à 26,6% [6]. La perte d'épaisseur à long terme peut entraîner des pertes énormes pour les entreprises. Par exemple, en Colombie-Britannique (Canada), environ 1 million de mètres cubes de gaz naturel ont fui en raison de l'endommagement des pipelines en 2012 [7]. Outre la menace que représente la corrosion pour l'intégrité structurelle, les coûts qui en découlent n'ont pas encore été particulièrement étudiés pour l'industrie des pipelines dans le monde ; cependant, selon un rapport publié par National Association of Corrosion Engineers (NACE) en 2016, le coût mondial estimé de la corrosion est d'environ 2 500 milliards de dollars [8].

L'estimation de la corrosion des pipelines est fondamentale pour l'analyse de leur fiabilité [9]. Pour ce faire, une méthodologie qui compromet le code d'inspection des pipelines de l'American Petroleum Institute (API) et le processus d'évaluation directe de la NACE est appliquée depuis 2005 [10]. La corrosion des pipelines peut être décrite comme une dégradation systématique de la paroi du pipeline due aux actions des paramètres d'exploitation sur le matériau du pipeline [9]. La plupart des méthodes existantes utilisent des techniques d'évaluation non destructives telles que les ondes ultrasonores (UT) pour détecter la perte d'épaisseur de la paroi et ainsi prédire la durée de vie restante. Des pratiques d'inspection et d'entretien des pipelines basées sur l'expérience sont aussi utilisées. Cependant, Ces procédures ne sont plus suffisantes et des méthodologies quantitatives basées sur le risque sont nécessaires. Des outils analytiques sont donc développés depuis plusieurs années dans le but, d'une part, de réduire l'impact économique des défaillances et, d'autre part, de limiter autant que possible l'impact que les défaillances peuvent avoir sur l'environnement, la santé et la sécurité.

L'objectif de ce mémoire par article est d'étudier la dégradation par corrosion du pipeline en utilisant les techniques d'apprentissage automatique. Deux articles scientifiques ont été rédigés pour répondre à cet objectif. Le premier a traité les méthodes d'apprentissage non supervisé à savoir le Self Organising Maps (SOM) combiné avec la Classification Ascendante Hiérarchique (CAH) pour mieux visualiser l'impact de la corrosion évaluée par les inspections périodiques par ultrasons. En effet, lors de la rédaction de ce premier article, on avait une seule variable disponible, l'épaisseur du tuyau mesurée sur 125 points. L'article 2 vient compléter le premier, en présentant 5 modèles d'apprentissage automatique (Forêt aléatoire ou Random Forest (RF), Machine à vecteur support ou Support Vector Machine (SVM), K plus

proches voisins ou K-Nearest Neighbors (KNN), Perceptron Multicouche ou Multilayer Perceptron (MLP), Machine à gradient boosting ou Gradient Boosting Machine (GBM)) pour prédire de manière précise la perte d'épaisseur des pipelines. Pour améliorer la précision des modèles, des techniques d'ingénierie des caractéristiques ont été utilisées à savoir l'élimination récursive des paramètres et l'Analyse en Composantes Principales (ACP). Aussi, l'optimisation des paramètres à travers le grid search a amélioré les résultats du modèle. Le mémoire par article sera divisé en cinq chapitres. Le premier sera consacré à la revue de la littérature des modèles d'apprentissage automatique utilisés en corrosion. Le deuxième chapitre traitera de la méthodologie utilisée dans les deux articles mis en annexe. Enfin, avant de passer à la conclusion générale, nous aborderons respectivement les résultats du premier et deuxième articles en chapitres 3 et 4 et la discussion générale en chapitre 5.

Chapitre 1 : Revue de la littérature

Depuis le développement des techniques des données massives, l'apprentissage automatique a montré des avantages significatifs dans la modélisation et l'exploration des données [11]. L'apprentissage automatique a été utilisé dans un certain nombre de problèmes liés à la corrosion, notamment pour la modélisation de la corrosion du CO2 [12], la détection de la corrosion à partir d'une analyse d'image automatisée [13], la modélisation de la croissance des défauts de corrosion dans les pipelines [1], l'inspection des matériaux [14], la modélisation du taux de corrosion dans un environnement marin [15], la détermination du temps d'initiation de la corrosion de l'acier encastré dans le béton armé [16], la prédiction des spectres d'impédance électrochimiques [17], la caractérisation de la distribution spatiale de la corrosion par piqûres [18], la modélisation du vieillissement des pipelines [19], la prédiction du taux de corrosion [20] et la détection de la corrosion [21]. Les auteurs dans [20] développent différents modèles pour prédire les dommages de la corrosion de l'acier galvanisé revêtu des structures métalliques exposées aux intempéries. Ils utilisent le modèle Multivariate Adaptive Regression Splines (MARS) pour compléter le traitement des données et la méthode d'apprentissage non supervisée SOM comprenant diverses couches (supersom) d'apprentissage supervisé et non supervisé pour définir la perte par corrosion de l'acier galvanisé au cours de la première année. Dans [22], les auteurs utilise le régresseur Gradient Boosting pour prédire la résistance à la corrosion dans des alliages à éléments principaux multiples. Les auteurs dans [23] ont formé six modèles d'apprentissage automatique sur les données dérivées des essais ultrasoniques pour prédire le degré de corrosion en fonction des caractéristiques ultrasoniques. Les résultats montrent que les modèles d'apprentissage automatique, à l'exception du modèle linéaire, peuvent prédire

de manière précise et robuste le degré de corrosion sous l'influence de l'amplitude des valeurs aberrantes et de la taille de l'ensemble d'apprentissage. Velázquez et al. [24] passent en revue les différentes méthodes probabilistes et statistiques utilisées dans la littérature pour étudier les problèmes de corrosion et leur application dans la vie réelle. Dans [25], on a établi un modèle de relation entre le potentiel de corrosion de l'acier faiblement allié de l'eau de mer de Sanya et ses facteurs d'influence par le biais d'un réseau neuronal artificiel et ont visualisé l'influence de plusieurs éléments d'alliage sur le potentiel de corrosion. Pour estimer la croissance de la profondeur des défauts de corrosion des pipelines âgés, les auteurs dans [1] mettent en œuvre une approche d'apprentissage automatique axée sur les données qui s'appuie sur l'ACP, Particle Swarm Optimisation (PSO), le réseau neuronal artificiel Feed-Forward Artificial Neural Network (FFANN), GBM, RF et Deep Neural Network (DNN). Zhao et al. [26] ont proposé les méthodes du rough set et de l'arbre de décision pour analyser la corrosion du sol des pipelines. Dans [27] on a effectué une régression à l'aide de différents algorithmes d'apprentissage automatique (MLP, RF, SVM, KNN, GBM) pour modéliser les données expérimentales des taux de corrosion variables dans le temps des spécimens d'acier doux, lorsque des inhibiteurs de corrosion ont été ajoutés au système à différentes concentrations et schémas de dosage. Ils ont constaté que RF était le meilleur algorithme qui prédisait le profil temporel complet des taux de corrosion avec une erreur quadratique moyenne allant de 0,005 à 0,093. Les auteurs de [28] ont proposé un nouvel algorithme intelligent hybride pour prédire le taux de corrosion de la canalisation à écoulement multiphase. Le modèle proposé combine Support Vector Regressor (SVR), l'ACP et Chaos Particle Swarm Optimization (CPSO). Ainsi, l'ACP est

utilisée pour réduire la dimension des données et le CPSO pour optimiser les paramètres hyperfins dans la SVR.

Chapitre 2 : Méthodologie

1. Classification Ascendante hiérarchique

Contrairement aux méthodes non hiérarchiques qui produisent une partition en un certain nombre (souvent fixe) de classes, la classification ascendante hiérarchique (CAH) produit des suites de partitions emboîtées d'hétérogénéités croissantes, entre la partition en n classes où chaque objet est isolé et la partition en une classe qui regroupe tous les objets [29]. La CAH peut être utilisée en se basant sur une mesure de ressemblance (distance, dissimilarité ou similarité). Il faut la définir entre deux objets et entre deux classes.

Voici le schéma de l'algorithme :

- Étape 1. Les classes initiales sont les objets.
- Étape 2. On calcule la ressemblance entre les classes.
- Étape 3. Les deux classes les plus proches sont fusionnées et remplacées par une seule classe.
- Étape 4. On reprend l'étape 2 jusqu'à ce qu'il n'y ait plus qu'une seule classe, qui contient tous les objets.

Les résultats d'une CAH sont fréquemment représentés à l'aide d'un dendrogramme. Un dendrogramme est un arbre dont les nœuds sont les classes de la hiérarchie. Cet arbre peut être coupé à une hauteur plus ou moins grande pour obtenir un nombre plus ou moins restreint de classes, ce nombre de classes pouvant être choisi pour optimiser des critères de qualité statistique. Il existe des méthodes sophistiquées basées sur des critères précis. Les auteurs de [30] utilisent plusieurs critères pour déterminer le nombre de classes optimal.

1.1 Les principales distances utilisées

Ainsi, l'algorithme CAH fonctionne en recherchant les classes les plus proches à chaque étape et en les fusionnant, et l'algorithme réside dans la définition de la ressemblance entre deux classes A et B. Nous rappelons les notions relatives à une distance comme mesure de ressemblance Pour plus d'informations, le lecteur peut consultée les références [29,31,32]. Quand les deux classes sont réduites chacune à un élément, la définition de leur distance est naturelle (il s'agit notamment de la distance euclidienne entre les deux éléments), mais dès qu'une classe a plus d'un élément, le concept de la distance entre deux classes est moins évident. Elle peut être définie de nombreuses façons, mais les définitions les plus courantes sont celles décrites ci-dessous.

La distance maximale entre deux classes A et B tend à produire des classes de même diamètre. Par définition, elle est très sensible aux valeurs aberrantes, et est donc peu utilisée. La CAH correspondante est dite « du saut maximum » ou « critère du diamètre » ou « compte linkage ».

$$d(A,B) = \max(x,y) \text{ avec } x \in A \text{ et } y \in B$$

La distance minimale entre deux classes A et définit une CAH dite « du saut minimum ». Son point faible est qu'elle est sensible à « l'effet de chaîne » (ou chaînage) : si deux classes très éloignées sont reliées par une chaîne d'individus proches les uns des autres, elles peuvent être regroupées.

$$d(A,B) = \min(x,y) \text{ avec } x \in A \text{ et } y \in B$$

La distance moyenne entre deux classes A et , qui définit une CAH dite « du saut moyen » ou « average linkage », est intermédiaire entre la distance maximale et la distance minimale, et moins sensible au bruit. Elle tend à produire des classes ayant la même variance.

$$d(A,B) = \frac{1}{n_A n_B} \sum_{i \in A} \sum_{j \in B} d(x_i, x_j)$$

La distance entre les barycentres de A et B, qui définit une CAH dite « distance des barycentres » ou « centroid method » (d'après le nom de centroïdes parfois donné aux barycentres), est plus robuste aux valeurs aberrantes mais moins précise. C'est la plus simple à calculer.

$$d(A,B) = d(\bar{x}_A, \bar{x}_B)$$

Avec $\bar{x}_A = \frac{1}{n_A} \sum_{i \in A} x_i$; $\bar{x}_B = \frac{1}{n_B} \sum_{j \in B} x_j$ et $\bar{x}_{AB} = \frac{\bar{x}_A n_A + \bar{x}_B n_B}{n_A + n_B}$ les barycentres respectivement de A, de B et de la réunion de A et B.

La méthode de Ward est l'une de celles qui correspondent le mieux à l'objectif de la classification. Elle est fréquemment utilisée car elle est une interprétation simple. Comme une bonne classification est une classification pour laquelle l'inertie interclasse est élevée (ou l'inertie intra classe faible), et comme le passage d'une classification en $k+1$ classes à une classification en k classes (regroupement de deux classes) ne peut que baisser l'inertie interclasse, on cherchera à fusionner les deux classes qui feront le moins baisser l'inertie interclasse. La notion de distance correspondant à cet objectif est la distance de Ward entre deux classes, définie comme la baisse d'inertie résultant de leur fusion et on cherche à la minimiser.

$$d(A,B) = \frac{n_A n_B}{n_A + n_B} d^2(\bar{x}_A, \bar{x}_B)$$

2. Self Organising Maps (SOM)

La carte autoadaptative (ou Self Organising Maps (SOM) en anglais) [30,31] est une technique d'apprentissage automatique non supervisée. C'est un type de réseau de neurones diffèrent de certains réseaux qui sont formés à

l'aide de l'apprentissage par correction des erreurs (rétropropagation). C'est aussi une technique de réduction de la dimensionnalité par la possibilité de présenter les données de grandes dimensions sous forme d'une carte à deux dimensions. SOM est formé à l'aide de l'apprentissage compétitif. L'apprentissage compétitif fait référence à trois étapes d'apprentissage :

- Compétition

Dans le cas du SOM, chaque nœud se voit attribué un vecteur de poids avec la même dimension que l'espace d'entrée (figure 1). Ensuite, la distance est calculée entre chaque nœud ou neurone (vecteur de poids) et le vecteur d'échantillon des données d'entrées. Le neurone qui a la distance la plus faible est le neurone gagnant de la compétition. Il est aussi appelé le Best Matching Unit (BMU, en anglais). La distance qui est souvent utilisée est la distance euclidienne :

$$D(X,Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

Où x_i et y_i représentent les neurones i.

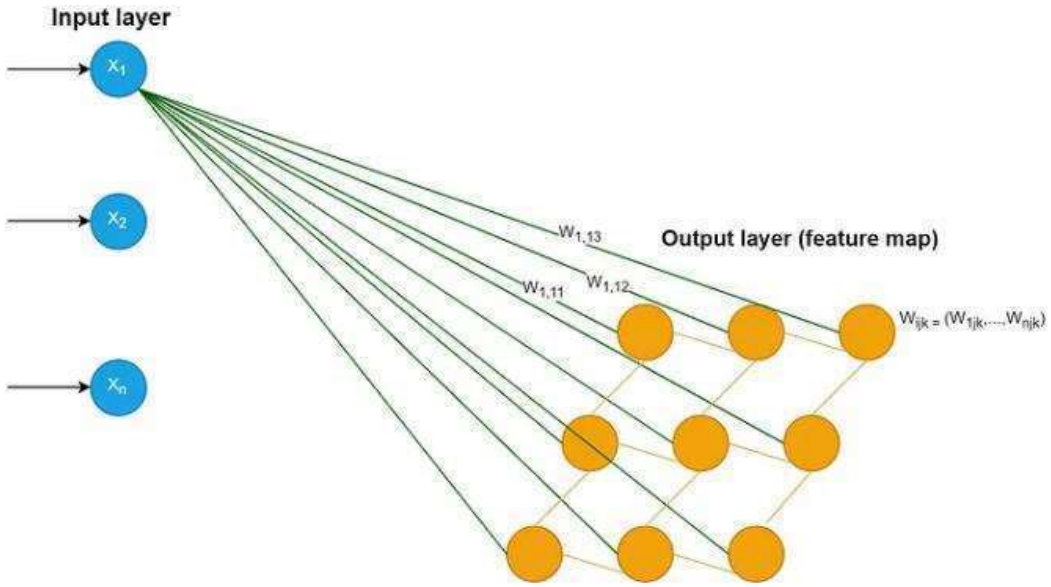

Figure 1: Attribution de chaque échantillon de l'espace d'entrée à un neurone

- Corporation et adaptation

Une mise à jour des poids est effectuée après le choix du BMU. Les vecteurs poids du BMU et des neurones voisins sont mis à jour. Pour déterminer les neurones voisins, la fonction de noyau de voisinage est calculée. C'est une fonction qui dépend de deux facteurs : le temps (temps incrémenté à chaque nouvelle donnée d'entrée) et la distance entre le neurone gagnant et l'autre neurone. La fonction de noyau la plus utilisée est donnée dans l'équation suivante :

$$h(r,s,t) = \exp\left(-\frac{\| \vec{r} - \vec{s} \|}{2\sigma^2(t)}\right)$$

$\vec{r}$ est le vecteur de position du neurone, $\vec{s}$ est le vecteur de position du neurone gagnant et $\sigma^2(t)$ le paramètre de lissage.

La formule de mise à jour du neurone est :

$$W_v(k+1) = W_v(k) + \theta(s,r,k) \cdot \alpha(k) \cdot \ D(t) - W_v(k)$$

$\theta(s, r, k)$ est la fonction de noyau, k est l'indice de l'étape, s est l'indice du noyau BMU pour le vecteur d'entrée $D(t)$, r est l'indice du neurone, $\alpha(k)$ est un coefficient d'apprentissage à décroissance monotone et W_v le poids du neurone v [35].

L'algorithme de SOM est :

1. Initialiser les poids w_{ij} aléatoirement pour chaque nœud avec des valeurs normalisées (on normalise les variables). Initialiser le taux d'apprentissage α.

2. Calculer la distance euclidienne au carré entre le vecteur d'entrée x_i et le vecteur de poids w_{ij} pour le nœud j sur la grille SOM :

$$D(j) = \sum_{i=1}^{n} \left(x_i(t) - w_{ij}(t)\right)^2$$

où $\boldsymbol{n}$ est le nombre de vecteurs d'entrée et $\boldsymbol{t}$ correspond au nombre d'itérations.

3. Trouver un nœud gagnant (BMU) avec la condition suivante :

$$BMU = argmin\, D(j)$$

4. Ajuster les poids des BMU et des nœuds voisins dans le rayon donné pour tous les vecteurs d'entrée en mettant à jour les nouveaux poids comme suit :

$$W_v(k+1) = W_r(k) + \theta(s, r, k) \cdot \alpha(k) \cdot \ D(t) - W_r(k)$$

3. K plus proches voisins/K-Nearest Neighbors (KNN)

La méthode des K plus proches voisins [36] ou (K-Nearest Neighbors en anglais) (KNN) est une méthode d'apprentissage supervisée non paramétrique

qui se base sur la distance pour prédire une nouvelle valeur. C'est une méthode facile à implémenter, qui s'adapte facilement et qui a peu d'hyperparamètres. Elle est utilisée pour des problèmes de classification et de régression. La classification ou la prédiction d'une nouvelle valeur est basée sur les valeurs des voisins les plus proches déterminés à l'aide d'une distance. La valeur K représente le nombre de voisins, s'il est égal à 1, pour un problème de classification, la classe prédite est la classe du voisin le plus proche et pour un problème de régression, la valeur prédite est la valeur du voisin le plus proche. Si K est supérieur à 1, la valeur prédite est la moyenne des valeurs de K voisins pour un problème de régression.

L'une des étapes les plus importantes de l'algorithme KNN est la détermination des voisins qui se fait à travers un calcul de distance. L'une des distances les plus utilisées et qui est utilisée dans ce document est la distance euclidienne. Soit deux échantillons, $X = \{x_1, x_2, \dots, x_n\}$ and $Y = \{y_1, y_2, \dots, y_n\}$, la distance euclidienne se calcule comme suit :

$$D(X, Y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

La détermination de la valeur de K est très importante étant donné qu'un mauvais choix de K peut conduire à un surajustement ou à un sous-ajustement. Les écarts élevés avec un biais faible sont souvent caractérisés par des valeurs de K inférieures. L'algorithme de KNN pour la régression est :

1. Calculer de la distance euclidienne entre les x_i associés à la variable à prédire et les x_i des données de l'entrainement.
2. Ordonner les distances obtenues de manière croissante.
3. Trouver un nombre heuristiquement optimal K de voisins les plus proches, basé sur métrique de performance d'un modèle (le Root Mean

Square Error (RMSE), par exemple). Ceci est fait en utilisant la validation croisée.

4. Calculer la moyenne pondérée par la distance inverse avec les K -plus proches voisins. La pondération par la distance inverse signifie que les voisins les plus proches ont un poids plus important dans le calcul de la moyenne que les voisins plus éloignés.

4. Forêt aléatoire/Random Forest (RF)

Forêt aléatoire (ou Random Forest en anglais) (RF) est une méthode d'apprentissage ensembliste qui est constitué de plusieurs arbres de décision. Elle est proposée par Tin Kam Ho en 1995 [37] et dont une extension a été proposée par Leo Breiman et Adele Cutler en 2001 [38]. Pour une meilleure compréhension de RF, nous allons présenter en premier lieu les arbres de décision.

4.1 Arbre de décision

L'arbre de décision [39] (ou decision tree en anglais) est une méthode d'apprentissage automatique supervisée non paramétrique. C'est un algorithme qui est utilisée à la fois pour les tâches de classification et de régression. Il est facile à interpréter, il ne nécessite pas beaucoup de préparations des données et il est flexible. Les arbres de décision ont une structure hiérarchique (figure 2) avec trois différents niveaux :

- Le nœud racine ou le nœud initial représente l'ensemble de l'échantillon et peut être divisé en plusieurs nœuds ;
- Les nœuds intérieurs représentent les caractéristiques d'un ensemble de donnée ;
- Les nœuds feuilles représentent le résultat.

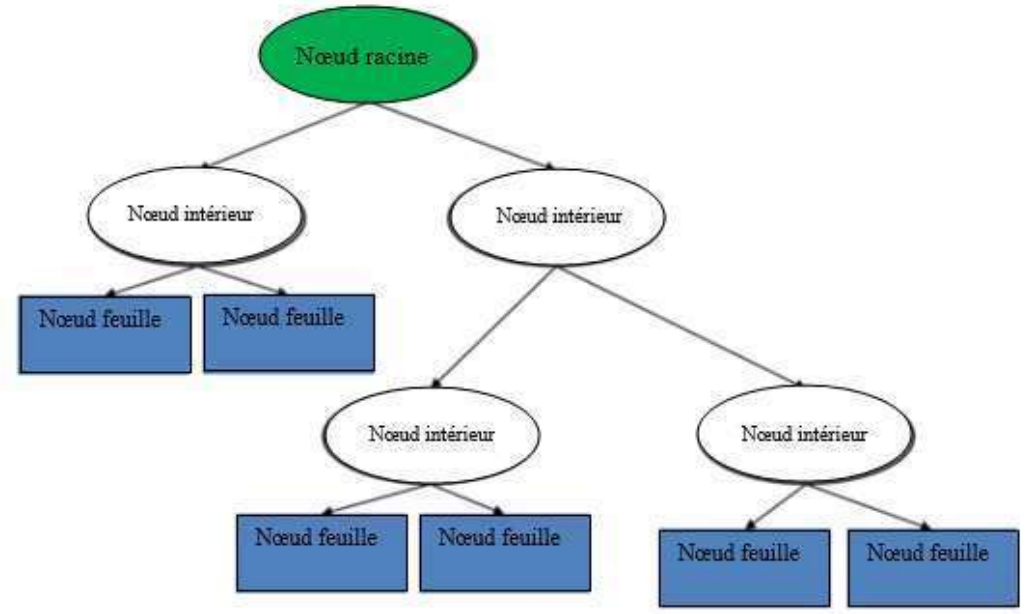

Figure 2: Structure d'un arbre de décision

Il existe plusieurs types d'algorithmes d'arbres de décision qui se diffèrent par leur manière de sélectionner la meilleure caractéristique (attribut) pour chaque nœud. Iterative Dichotomiser 3 (ID3) est un algorithme développé par Quinlan [40] qui utilise l'entropie et le gain d'informations comme mesures pour évaluer les fractionnements des candidats. L'entropie est définie par la formule suivante :

$$\mathrm{H(t)} = \mathrm{Entropie}\,(S) = -\sum_{c \in C} p(c)\log_2\, p(c)$$

S représente le jeu de données avec lequel l'entropie est calculée

c représente les classes dans le jeu de données, S

$p(c)$ représente la proportion de points de données appartenant à la classe c par rapport au nombre total de points de données du jeu de données, S.

Le gain d'informations est représenté par la formule suivante :

$$IG(S,A) = \mathrm{H}(S) - \sum_{t \in T} p(t)\mathrm{H}(t) = \mathrm{H}(S) - \mathrm{H}(S \mid A)$$

H S est l'entropie de l'ensemble S ;

T est le sous ensemble crée à partir de fractionnement S par attribut A tel que

$$\bigcup_{t \in T} t$$

$p(t)$ est la proportion du nombre d'éléments dans t au nombre d'éléments dans l'ensemble S ;

$\mathrm{H}(t)$ est l'entropie du sous ensemble t.

L'algorithme C4.5 est une itération ultérieure de l'algorithme d'ID3. Il est aussi développé par Quinlan [41]. Le nom de l'algorithme « C4.5 » signifie « Classification and Regression Tree algorithm » (algorithme d'arbre de classification et de régression). L'algorithme fonctionne en construisant un arbre de décision à partir d'un ensemble de données d'entraînement, où chaque nœud de l'arbre représente une décision basée sur une caractéristique (attribut) particulière de l'ensemble de données. La mesure utilisée pour le fractionnement au sein des arbres de décision peut être le gain d'informations ou le ratio de gain.

Classification And Regression Tree (CART) est l'algorithme d'arbre de décision le plus populaire. Il est développé par Breiman [39], il utilise l'impureté de Gini pour identifier l'attribut idéal pour effectuer le fractionnement pour les problèmes de classification. Pour les problèmes de régression, la mesure utilisée est le Mean Square Error (MSE) représenté par la formule suivante :

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

y_i représente les valeurs de la variable cible qui est une variable continue, $\hat{y}_i$ est la valeur prédite. Soit un échantillon

$(x_1, y_1),\ x_2, y_2\ , \ldots\ldots, (x_n, y_n)$ représenté comme suit :

X	Y
22	133
34	128
35	135
35	128
37	89
47	133
52	53
56	150
60	107

La première chose à faire est d'ordonner la variable X. Ensuite, on calcule la moyenne des premières valeurs de X ((22+34)/2=28). Les données de X sont séparées en deux parties, <28 et >28. Le but est de partitionner l'espace X par une séquence de division binaire. On divise les données de Y en deux parties, <28 et >28 et on calcule la moyenne dans chaque partie. Soit A, la partie <28 et B, celle >28. La moyenne de Y dans A est de 133 et celle de B est de 115,38. Ainsi, la valeur prédite pour les valeurs de Y inférieure à 28 est de 133 et pour les valeurs supérieures à 28 est de 115,38. On calcule le MSE en utilisant la vraie valeur de Y pour une valeur de X donnée et la valeur prédite de Y. Ensuite, on refait le même processus en calculant cette fois la moyenne des deux seconds nombres de X. On continue le processus jusqu'à calculer la moyenne des deux (n-1)$^{\text{-ème}}$ nombres de X. Ainsi, les n-1 erreurs quadratiques sont calculées, pour déterminer le point auquel diviser les données, on choisit le point où l'erreur quadratique moyen est le plus faible. Si on a plusieurs variables, à chaque nœud, les variables passeraient par le même

processus que X a traversé dans l'exemple ci-dessus. Les méthodes de validation croisée sont souvent utilisées pour arrêter l'élagage de l'arbre.

4.2 Forêt aléatoire

L'algorithme de régression de forêt aléatoire est un modèle composé d'une série de sous-arbres de décision. C'est un algorithme qu'on peut utiliser pour des problèmes de classification et de régression et il s'est vite adopté par sa souplesse d'utilisation et sa flexibilité. Pour un problème de classification, la classe prédite représente la classe prédite par la plupart des arbres. Pour un problème de régression, la valeur prédite, représentée dans l'équation suivante est la moyenne des valeurs prédites par les différents arbres.

$$\bar{h}(x) = \frac{1}{T}\sum_{t=1}^{T}\{h(x, \theta_t)\}$$

$\bar{h}(x)$ la fonction de prédiction moyenne pour une observation x ;
x est l'observation d'entrée dont la valeur de sortie doit être prédite.
T est le nombre total de modèles de prédiction inclus dans l'ensemble d'apprentissage.
θ_t sont les paramètres de chaque modèle individuel t, qui est utilisé pour effectuer une prédiction $h(x, \theta_t)$ pour l'observation x.
Le processus d'apprentissage du modèle foret aléatoire est le suivant :

1. En utilisant l'agrégation bootstrap, on échantillonne pour chaque arbre un ensemble de données d'entrainement avec remplacement qui est aussi appelé échantillon boostrap ;

2. Ensuite k caractéristiques sont sélectionnées de manière aléatoire pour la division des nœuds afin de construire un seul sous-arbre de décision de régression ;

3. On répète les étapes 1) et 2) pour construire T sous arbres de décision et former les forêts ;

4. Enfin, la prédiction du modèle forêt aléatoire est la moyenne des valeurs prédites par chaque arbre [23].

5. Machine à vecteur support/ Support Vector Machine (SVM)

Support Vector Machine (SVM) est un modèle d'apprentissage automatique supervisé utilisé pour les problèmes de régression et de classification. Il est développé pour la première fois par Vapnik et ses collègues [39,40]. Pour les problèmes de régression, le nom change et devient Support Vector Regression (SVR). Le principe est presque le même que pour les problèmes de classification à la différence de la variable continue qu'on doit prédire pour les problèmes de régression. L'objectif de l'algorithme de SVR est de trouver un hyperplan dans un espace à n dimensions qui ajuste le mieux les données. L'hyperplan est la ligne qui nous aidera à prédire la valeur continue ou la valeur cible. La fonction continue à approximer est représentée dans la formule suivante :

$$y = w.x + b$$

5.1 SVR linéaire

SVR formule ce problème d'approximation de fonction comme un problème d'optimisation comme présenté dans les équations suivantes :

$$min\,\frac{1}{2}\|w^2\| + c\sum_{i=1}^{n}\xi_i + \xi_i^*$$

Sous condition

$$\begin{cases} z_i - (w \cdot x + b) & \leq \varepsilon + \xi_i \\ (w \cdot x + b) - z_i & \leq \varepsilon + \xi_i^* \\ \xi_i \xi_i^* & \geq 0 \end{cases}$$

$$y = \sum_{i=1}^{N} (\alpha_i - \alpha_i^*) \cdot \langle x_i, x \rangle + b$$

Où $\|w^2\|$ est la magnitude du vecteur normal à la surface qui est approximée, $\xi_i \, et \, \xi_i^*$ déterminent combien de points peuvent être tolérés en dehors du tube, z_i représente la marge associée à chaque exemple d'entraînement x_i, $\alpha_i \, et \, \alpha_i^*$ les multiplicateurs de Lagrange associés aux contraintes de marge supérieure et inférieure, respectivement, pour le i-ème exemple d'entraînement, et C un paramètre de régularisation qui contrôle le compromis entre la complexité du modèle et son aptitude à s'adapter aux données d'entraînement [44].

5.2 SVR non linéaire

Dans le cas non linéaire (figure 3), on utilise des fonctions de noyau pour transformer les données pour permettre la séparation linéaire comme dans les équations suivantes :

$$min \frac{1}{2} \|w^2\| + c \sum_{i=1}^{n} \xi_i + \xi_i^*$$

Sous condition

$$y_i - w^T \varphi(x_i) \leq \varepsilon + \xi_i^* \quad i = 1, \ldots, N$$
$$w^T \varphi(x_i) - y_i \leq \varepsilon + \xi_i \quad i = 1, \ldots, N$$
$$\xi_i, \xi_i^* \geq 0 \qquad i = 1, \ldots, N$$
$$w = \sum_{i=1}^{N_{SV}} (\alpha_i^* - \alpha_i) \varphi(x_i)$$

$$y = \sum_{i=1}^{N} (\alpha_i - \alpha_i^*) \cdot \langle \varphi(x_i), \varphi(x) \rangle + b$$

$$y = \sum_{i=1}^{N} (\alpha_i - \alpha_i^*) \cdot K(x_i, x) + b$$

Où $\varphi(x_i)$ la fonction de transformation de caractéristiques (ou fonction noyau), N_{SV} le nombre total de vecteurs de support et $K(x_i, x)$ est la fonction de noyau.

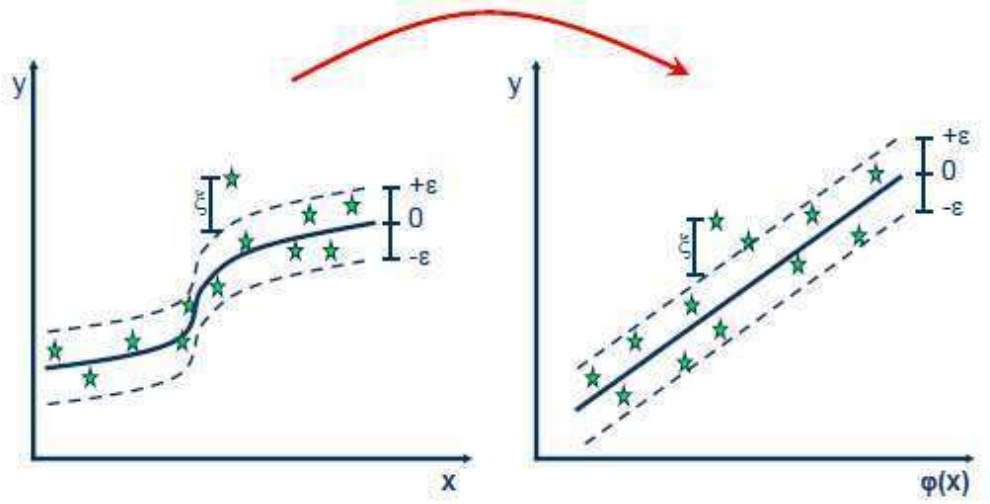

Figure 3: SVR non linéaire

6. Gradient Boosting Machine (GBM)

Gradient boosting est une méthode d'apprentissage automatique utilisée pour les problèmes de classification. C'est un modèle performant qui donne souvent de meilleurs résultats comparés aux arbres de décision et aux forêts aléatoires. Dans son processus d'estimation, le gradient boosting combine itérativement plusieurs modèles simples appelés « apprenant faibles » pour obtenir un « apprenant fort » avec une meilleure performance. Inspiré du gradient

boosting, le GBM est introduit par Friedman [45] pour les problèmes de régression.

Soit un échantillon d'entrainement $(x_i;\ y_i);\ i = 1; \dots n$. Nous supposons que l'on nous donne une classe de base d'apprenants $\mathcal{B}$ (modèles) et que notre classe de fonction cible est la combinaison linéaire de ces apprenants de base (notée $Lin(\mathcal{B})$). Soit $B = \{b_\tau(x) \in \mathbb{R}\}$ une famille d'apprenants paramétrée par $\tau \in$ T. La prédiction correspondant à un vecteur caractéristique x est donnée par un modèle additif de la forme [46]:

$$f(x) = \left(\sum_{m=1}^{M} \beta_m b_{\tau_m}(x) \right) \in \text{lin}\ (\mathcal{B})$$

Où $b_{\tau m}(x) \in \mathcal{B}$ est un apprenant faible et β_m est son coefficient additif correspondant.

Le but de GBM est d'obtenir une bonne estimation de la fonction f qui minimise approximativement la perte empirique [46]:

$$L^* = \min_{f \in \text{lin}(\mathcal{B})} \left\{ L(f) = \sum_{i=1}^{n} \ell(y_i, f(x_i)) \right\}$$

Où $\ell(y_i, f(x_i))$ est une mesure de la fidélité des données pour le i-ième échantillon pour la fonction de perte ℓ.

La méthode GBM peut être considérée comme un algorithme d'optimisation numérique qui vise à trouver un modèle additif qui minimise la fonction de perte [47]. L'algorithme débute par une initialisation du modèle par une première estimation. Le modèle qui est généralement utilisé est l'arbre de décision, le but est de réduire au maximum la fonction de perte. Ensuite à

chaque étape, un modèle qui s'adapte le mieux aux résidus est calculé et ajouté au modèle précèdent pour mettre à jour les résidus. L'itération continue jusqu'à ce qu'un nombre maximal d'itérations, fourni par l'utilisateur soit atteint. Les différentes étapes de l'algorithme [46] sont représentées ci-dessous :

Initialisation : Initialiser avec $f^0\ x\ = 0$

Pour $M = 0, \ldots\ldots, M - 1\ Faire$

Effectuer des mises à jour :

1. Calculer le pseudo-résidu : $r^m - [\frac{\partial \ell\ y_i, f^m\ x_i}{\partial f^m\ x_i}] i=1,\ldots,n$
2. Trouver les paramètres du meilleur apprenant faible :

$$\tau_m = \underset{\tau \in \mathcal{T}}{argmin} \sum_{i=1}^{n} \left(r_i^m - b_\tau(x_i)\right)^2$$

3. Choisir la taille du pas η_m par recherche de ligne

$$\eta_m = argmin_\eta \sum_{i=1}^{n} \ell(y_i, f^m(x_i) + \eta b_{\tau_m}(x_i))$$

4. Mise à jour du modèle : $f^{m+1}(x) = f^m(x) + \eta_m b_{\tau_m}(x)$
5. Sortie. $f^M(x)$

7. Perception multicouche/ Multilayer Perceptron (MLP)

Le perceptron multicouche (ou Multilayer Perceptron en anglais) (MLP) en anglais est un type de réseau neuronal artificiel organisé en plusieurs couches. C'est un réseau à propagation directe car l'information est de sens unique : de la couche d'entrée vers la couche de sortie. Chaque couche est constituée d'un nombre variable de neurones, les neurones de la dernière couche (dite « de sortie ») étant les sorties du système global.

Dans le perceptron multicouche à rétropropagation, les neurones d'une couche sont reliés à la totalité des neurones des couches adjacentes (figure 4). Ces liaisons sont soumises à un coefficient altérant l'effet de l'information sur le neurone de destination. L'algorithme de rétropropagation permet de déterminer ces coefficients qui est l'élément clef du fonctionnement du réseau.

La figure 4 montre un réseau multicouche. Chaque neurone i reçoit une série de signaux des neurones j situés aux couches précédentes. Le fonctionnement du réseau illustré est gouverné par l'équation suivante [48].

$$Z_i = \sum_{j=1}^{n_j} W_{ij} X_j + b_i$$

Où n_j est le nombre de neurones d'entrée et X_j est la valeur du signal transmis par le neurone j de la couche précédente. Les W_{ij} représentent les poids respectifs des connections entre les neurones j des couches précédentes et le neurone i de la couche actuelle. Les paramètres b_i, sont des valeurs de biais permettant une fonction de transfert non nulle à l'origine. Les entrées X_j sont pondérées par les poids W_{ij}. Une fois l'entrée alimentée, le neurone i la transforme et produit une sortie. Dans ce cas, Z_i et la sortie O_i, d'un neurone donné, sont reliées par une fonction de transfert de forme tangente hyperbolique illustrée dans l'équation suivante :

$$O_i = f(Z_i) = \frac{1 - e^{-2Z_i}}{1 + e^{-2Z_i}}$$

D'autres fonctions de transferts peuvent être utilisées telles que Relu, fonction linéaire et fonction exponentielle [49,50].

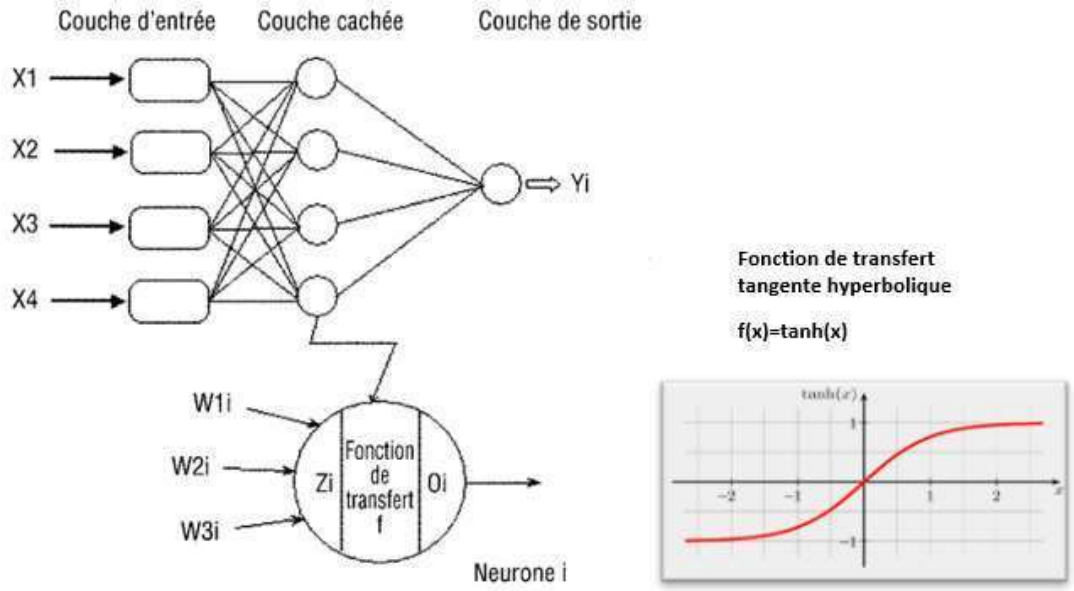

Figure 4: *Perceptron multicouche* [48]

L'erreur commise par le réseau à la sortie est calculée et ensuite minimisée. On parle de la méthode de rétropropagation des erreurs. Les poids du réseau W_{ij} sont corrigés afin de réduire l'erreur globale $\underline{E}$. La méthode de gradient descendant est utilisée pour minimiser l'erreur globale. Elle est représentée par l'équation suivante :

$$\underline{E} = \frac{1}{2}\sum_{k=1}^{n_k} (S_k - O_k)^2$$

Où S_k représente la valeur estimée et O_k la valeur observée et $\underline{E}$ l'erreur globale.

Les étapes de l'algorithme de rétropropagation des erreurs sont :

1 Présentation d'un motif d'entraînement au réseau ;

2 Comparaison de la sortie du réseau avec la sortie ciblée ;

3 Calcul de l'erreur en sortie de chacun des neurones du réseau ;

4 Calcul, pour chacun des neurones, de la valeur de sortie qui aurait été correcte ;

5 Définition de l'augmentation ou de la diminution nécessaire pour obtenir cette valeur (erreur locale) ;

6 Ajustement du poids de chaque connexion vers l'erreur locale la plus faible ;

7 Attribution d'un « crédit d'erreur » à tous les neurones précédents ;

8 Recommencer à partir de l'étape 4, sur les neurones précédents en utilisant le « crédit d'erreur » comme erreur.

Chapitre 3 : Article scientifique 1

Dans cette partie, nous présentons notre article (en annexe A) intitulé **Unsupervised Neural Network for Data-Driven Corrosion Detection of a Mining Pipeline** soumis et publié à la 35éme conférence de **The Florida Artificial Intelligence Research Society (FLAIRS), mai 2022.**

Résumé de l'article : La défaillance des pipelines souvent causée par la corrosion peut entraîner des problèmes de sécurité, environnementaux et économiques. Dans cette étude, un réseau neuronal non supervisé, les cartes auto-organisatrices ou Self Organising Maps (SOM) en anglais, sont utilisées pour créer des groupes représentant l'impact de la corrosion évaluée par des inspections périodiques par ultrasons. Ce nouvel aperçu de la représentation des données d'épaisseur devrait faciliter la planification des activités d'atténuation de la corrosion par le biais d'inspections basées sur les risques. Le modèle SOM a conduit à la réduction des variables dans des nœuds d'espace bidimensionnel. La CAH a ensuite été utilisée pour classer ces nœuds en regroupant les mesures de perte d'épaisseur. La méthode proposée, qui combine à la fois SOM et CAH, a permis de détecter l'étendue de la corrosion dans un pipeline minier.

3.1 Analyse des résultats

Ce chapitre présente les différents résultats obtenus de l'article scientifique 1.

3.1.1 Données

Dans cet article, nous avons utilisé les données obtenues auprès de Agnico Eagle Mine Goldex. Il s'agit de mesures d'épaisseur d'un pipeline qui est utilisé pour transporter les résidus (pulpe) du concentrateur au site du parc à résidus Manitou appartenant au Ministère de l'Énergie et des Ressources Naturelles. Les mesures ont été effectuées à l'aide d'une jauge d'épaisseur à ultrasons MMX-6 DL (Dakota Ultrasonics, USA). Pour effectuer les mesures d'épaisseur, la circonférence de la section du pipeline (figure 5) a été subdivisée en 24 points équidistants à partir desquels des lignes ont été tracées sur toute la longueur du pipeline. Des points de repère séparés de 1 pouce ont été marqués le long des 24 lignes. Au total, 125 points ont été marqués sur chaque ligne, allant de 1 (début) à 125 (fin) pour un total de 3000 points (125 x 24) sur la surface du pipeline pour les mesures d'épaisseur. Les points 1 à 125 représentent les variables et les lignes 1 à 24 sont les observations. En effet, les 125 variables sont toutes des mesures d'épaisseur. Ce travail porte sur les données collectées de 2016 à 2019.

Figure 5: Étape de marquage (gauche) et vue d'ensemble (droite) du tuyau témoin

3.1.2 Traitement et analyse des données

Au début du projet, nous disposions de deux bases de données différentes : une base de données contenant les variables opérationnelles et environnementales (température, tonnage, pression, débit, etc.) et une autre base de données contenant les mesures d'épaisseurs. La base de données opérationnelle s'étalait sur six années (2017-2021) et pour chaque année, les variables ont été collectées toutes les 5 minutes. Les mesures d'épaisseurs ont été collectées pendant quatre ans (2016-2019) et chaque année, une mesure d'épaisseur a été effectuée sur chaque point du pipeline. L'objectif du projet était de prédire la perte d'épaisseur en fonction des caractéristiques du pipeline. Comme pour tout projet de modélisation, le traitement et le nettoyage de la base de données ont été l'une des premières étapes. Les données obtenues étaient toutes complètes et les analyses effectuées n'ont pas révélé d'incohérences ou d'autres problèmes dans les données. Cependant, les données disponibles étaient insuffisantes pour utiliser certains modèles d'apprentissage automatique. Nous avons donc mis l'accent sur l'analyse des mesures d'épaisseurs dans ce premier article pour identifier les zones à risque nécessitant une surveillance. La détection rapide des points du pipeline présentant une perte d'épaisseur permet à la mine de prévoir une maintenance

préventive et de réduire les risques de fuite. Nous avons également effectué une analyse des variables opérationnelles et environnementales pour identifier les éventuelles corrélations entre ces variables.

La CAH, qui est une méthode d'apprentissage automatique non supervisée, est utilisée pour regrouper des données qui se ressemblent. Ainsi, le dendrogramme obtenu à partir de la CAH nous permet d'identifier rapidement les différences entre les lignes. Les lignes qui se détachent des autres ont peut-être des valeurs d'épaisseur plus élevées ou plus faibles que les autres. La figure 6 représente le dendrogramme des mesures d'épaisseur pour l'année 2019. Bien que les groupes ne contenant qu'un seul élément ne soient pas souhaitables dans une classification, nous avons effectué une classification en 3 classes avec une classe contenant la ligne 20 pour mettre en évidence sa particularité. L'analyse de la ligne 20 montre des valeurs très basses pour ce point du pipeline (en moyenne 5,45 mm). Dans la même lignée, l'algorithme de K-means est également utilisé pour identifier une séparation au niveau des lignes du pipeline. Comme le montre la figure 7, la ligne 20 s'éloigne des autres lignes, de même que les lignes 18, 19 et 21 s'éloignent du point représentant la position moyenne du cluster. Ces positions des lignes sur le dendrogramme montrent une variabilité des mesures d'épaisseur de ces lignes, ce qui nécessite une étude approfondie.

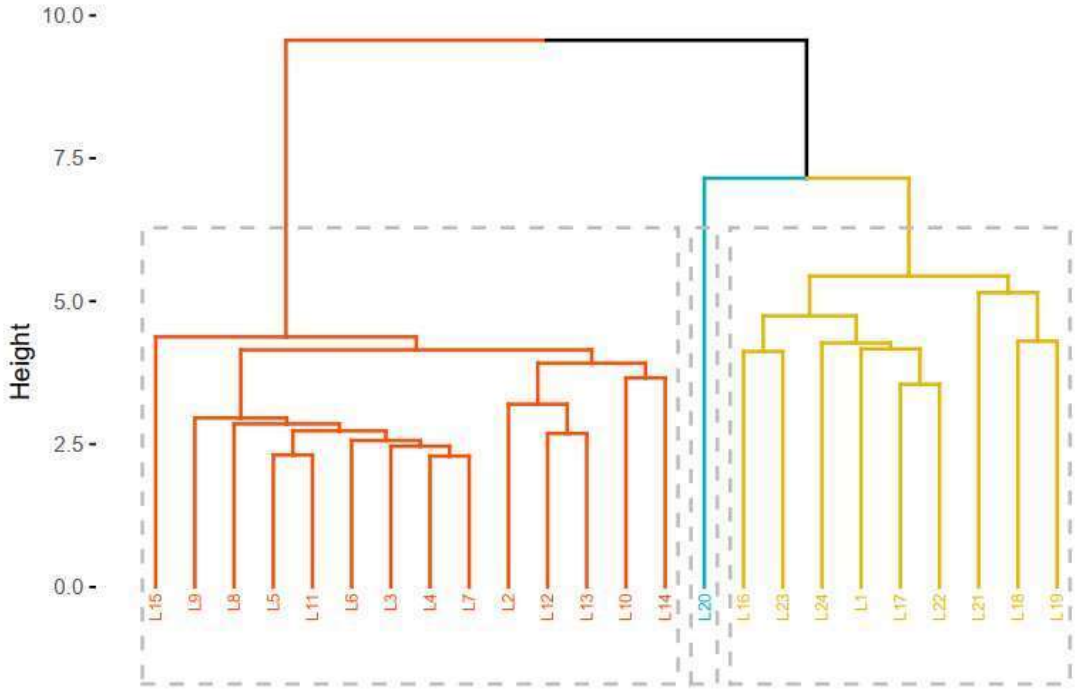

Figure 6: Dendrogramme

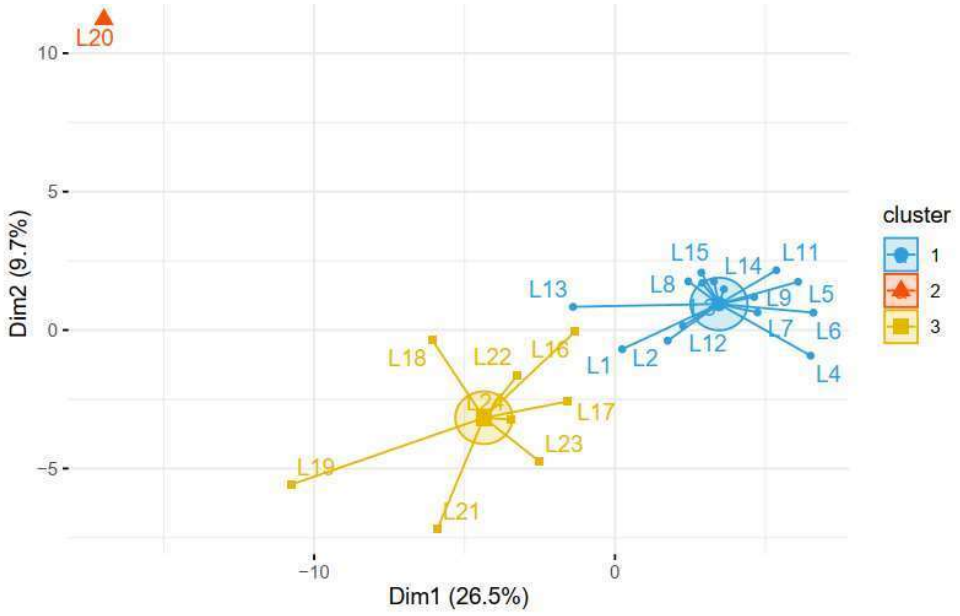

Figure 7: Classification par K-means

Pour l'analyse des variables opérationnelles et environnementales, une ACP a été effectuée sur l'ensemble des données afin d'étudier la colinéarité entre les variables et d'extraire l'information contenue dans l'ensemble des données. La figure 8 représente les corrélations entre les variables. On remarque une séparation des variables en deux blocs corrélés entre elles : les

variables (température au km 14, température pulpe flottation, température au km 0, pression au km 14) sont dans un bloc, tandis que les variables (pression au km 0, tonnage, débit de résidu, pH flottation, % solide résidu, TPH résidu calculé) sont dans un autre bloc. Les variables tonnage et débit de résidu sont fortement corrélées. L'analyse de la qualité de représentation (figure 9) révèle que les variables pression au km 0, température au km 0 et température pulpe flottation sont bien représentées sur les composantes principales. En revanche, les variables tonnage et pH flottation sont faiblement représentées. En d'autres termes, ces variables ne jouent pas un rôle important dans la variabilité des données, bien que leur calcul des contributions montre qu'elles contribuent le plus à la variabilité des données.

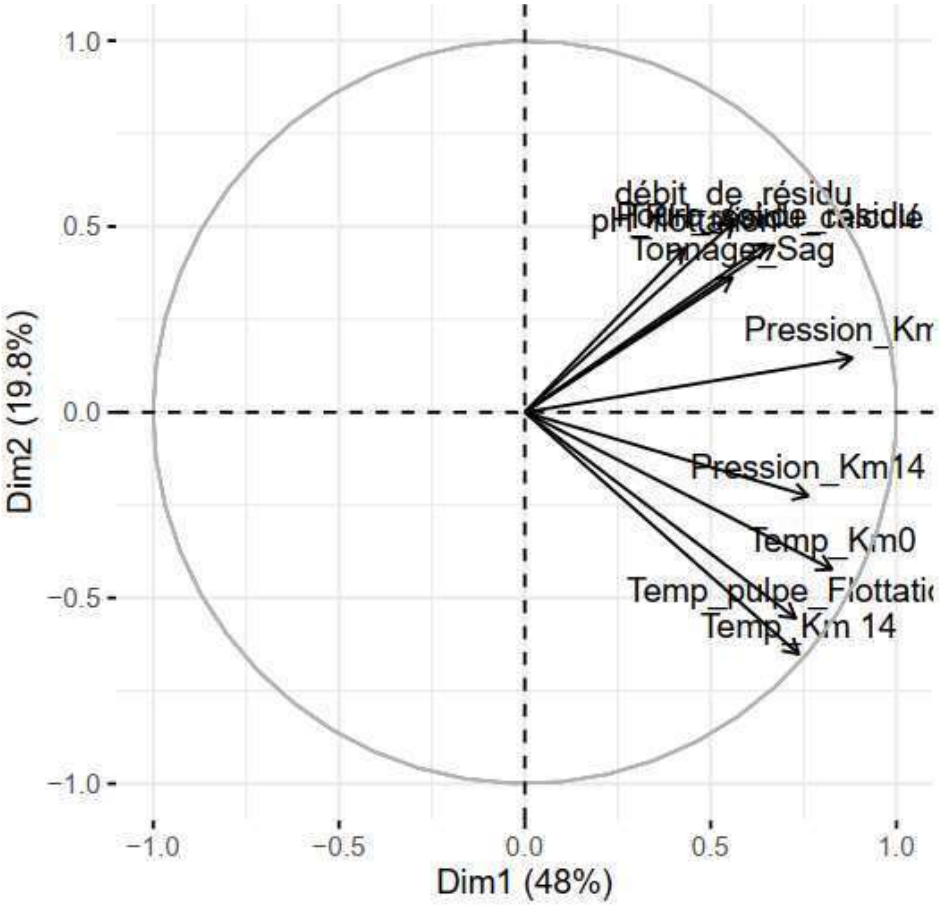

Figure 8: Cercle de corrélations des variables

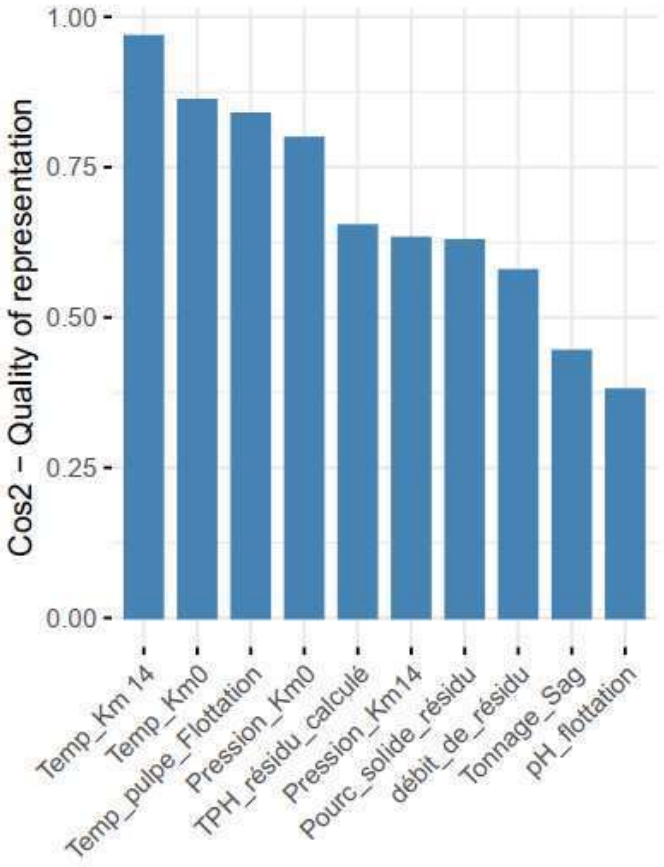

Figure 9: Qualité de représentation

L'article 1 est inspiré de ces analyses et met l'accent sur la classification des niveaux de corrosion en utilisant le Self Organising Maps combiné avec la classification ascendante hiérarchique.

3.1.1 Analyse des résultats de la modélisation

3.1.1.1 Analyse des mesures d'épaisseurs

L'analyse des mesures d'épaisseur est importante, elle permet d'identifier les zones à risque (faible épaisseur) au niveau du pipeline. Le contrôle dimensionnel de l'épaisseur de la paroi est de +15% à -12,5% de l'épaisseur nominale, qui est comprise entre 7,3 et 5,6 mm. L'épaisseur nominale moyenne est de 6,25 mm. La valeur de l'épaisseur minimale permet de déterminer la gravité de la corrosion. Elle est aussi utilisée pour estimer le temps avant fuite d'un pipeline donné. On remarque à la figure 10 qu'à

l'exception des lignes L4, L5, L9, L10 et L11, toutes les lignes présentent des valeurs minimales inferieures à l'épaisseur minimale autorisée par la firme (5,6 mm). Cependant, les lignes 18, 19 et 20 sont plus critiques avec des valeurs d'épaisseur minimale inférieures ou égales à 4,4 mm. On s'attend à ce que ces lignes présentent un temps de fuite court.

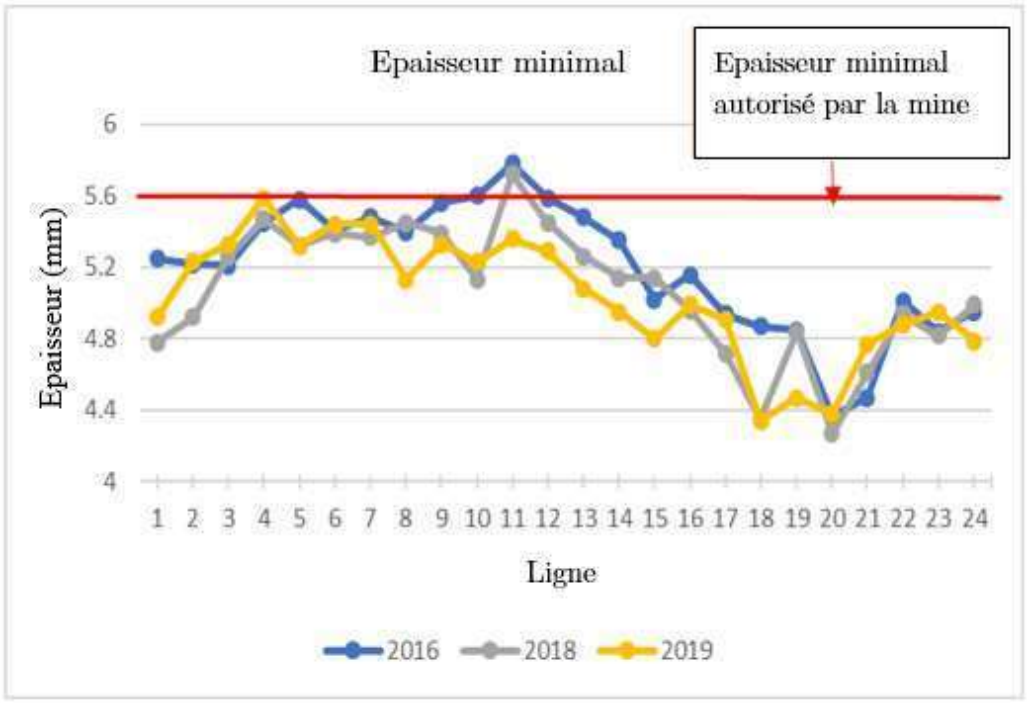

Figure 10: Distribution des valeurs d'épaisseur minimale

La Figure 11 illustre les corrélations obtenues entre les moyennes annuelles des conditions d'exploitation du pipeline et les valeurs moyennes de l'épaisseur des pipelines L2, L10, L20 et L16. Un certain nombre de relations significatives ont été identifiées.

Il est notable qu'il existe une corrélation négative prononcée entre des facteurs tels que le TPH résiduel calculé, la pression au Km 0 et la température au Km 0, et les valeurs d'épaisseur au niveau de la ligne 2. Cette relation suggère que

des augmentations de ces paramètres sont liées à une diminution de l'épaisseur du pipeline à ce niveau.

De façon similaire, l'épaisseur de la ligne 10 a montré une corrélation avec le tonnage Sag et la température au Km 0, indiquant que ces variables peuvent également jouer un rôle dans l'érosion de l'épaisseur du pipeline à cet endroit.

De plus, une corrélation négative a été observée entre le flux de résidu, le pourcentage de résidu solide, le TPH résiduel calculé et l'épaisseur au niveau de la ligne 16. Cette tendance implique que lorsque ces paramètres augmentent, l'épaisseur de la ligne 16 diminue.

Cependant, une relation positive a été constatée entre la température de la pulpe de flottation et l'épaisseur à la ligne 20, ce qui suggère que l'augmentation de la température pourrait potentiellement contribuer à maintenir ou même à augmenter l'épaisseur du pipeline à cet endroit.

Ces corrélations offrent un aperçu précieux de l'influence des conditions d'exploitation sur le taux de corrosion. Néanmoins, l'absence de données sur plusieurs années limite notre capacité à développer un modèle de prédiction de la corrosion qui pourrait être utilisé pour des interventions plus proactives et stratégiques dans la gestion de la corrosion des pipelines.

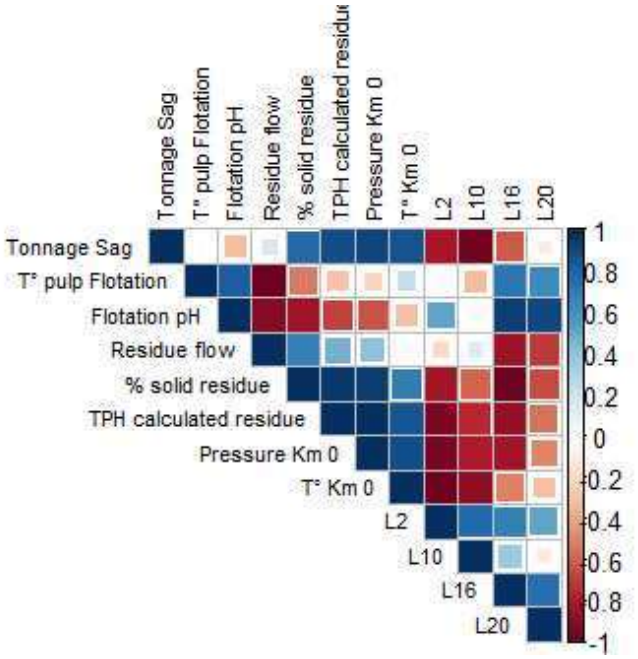

Figure 11: Corrélation entre les variables de procédés et l'épaisseur du pipeline

3.1.1.2 Résultats de la modélisation

Nous avons utilisé la méthode SOM pour réduire la dimensionnalité et représenter les nœuds sous forme d'une carte. Par la suite, la CAH est utilisée pour créer des classes aux nœuds définis par SOM. Les nœuds regroupent les données originales (ligne 1 à ligne 24). Le résultat du modèle SOM est un maillage de neurones hexagonaux 5×5 formés avec l'algorithme de Kohonen. Le maillage fournit une bonne représentation de l'espace d'échantillonnage. Il n'y a pas de zones très denses ou de cellules vides, au moins chaque cellule contient un élément. La carte entraînée résultante contient toutes les données dans une structure vectorielle de sorte que les données d'entraînement sont affectées sur chacun des neurones (figure 12).

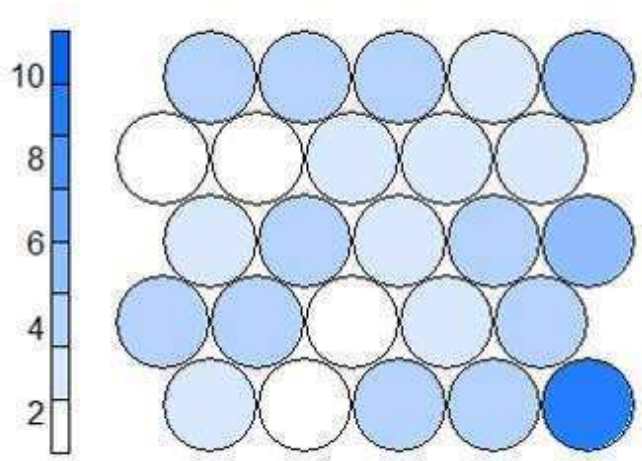

Figure 12: Représentation des nœuds par SOM

La première analyse des épaisseurs du pipeline peut être effectuée en calculant la distance entre les nœuds voisions (U-matrix). La distance de matrice unifiée (U-matrix) est une représentation graphique sous forme de carte de neurones dont la distance euclidienne entre les neurones est représentée par un gradient de couleur. A partir de la figure 13, on remarque que les neurones voisins sont représentés dans l'extrémité droite de la carte. Plus la couleur est foncée, plus les neurones sont proches. La partie de l'extrémité représente la partie de faibles valeurs d'épaisseur, autrement dit les lignes à risques sont représentées dans cette partie de la carte.

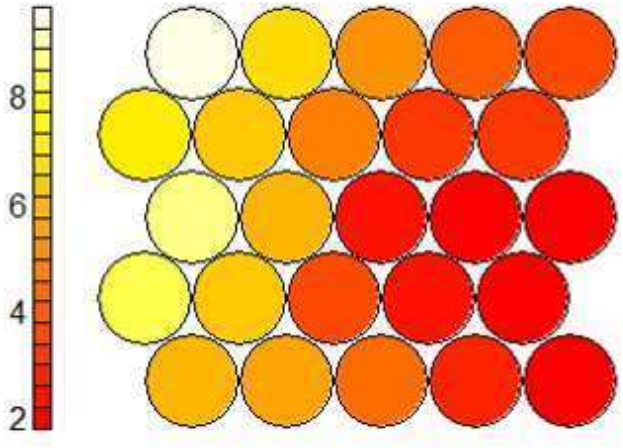

Figure 13: Distance entre voisins par SOM

La classification des lignes en fonction du degré de la corrosion est réalisée grâce à l'utilisation de la méthode de la classification ascendante hiérarchique. Après avoir calculé les vecteurs de poids des neurones par la méthode SOM, la CAH est utilisée pour regrouper les neurones en fonction de leur similarité. Le dendrogramme suggère la partition en trois classes. De plus, en utilisant le package Nbclust [30] d'autres indices (kl, ch, indice de hartigan etc) ont été calculés et la meilleure répartition reste trois classes. Ainsi, les 25 nœuds sont regroupés dans trois classes différentes comme représenté dans le dendrogramme (figure 14). La classe 3 contient les neurones V11, V16, V17, V21 et V22. L'analyse de la figure 15 montre que les nœuds de la classe 3 sont représentés dans la zone de faible valeur d'épaisseur, autrement dit cette classe de nœud contient les lignes à risque à surveiller.

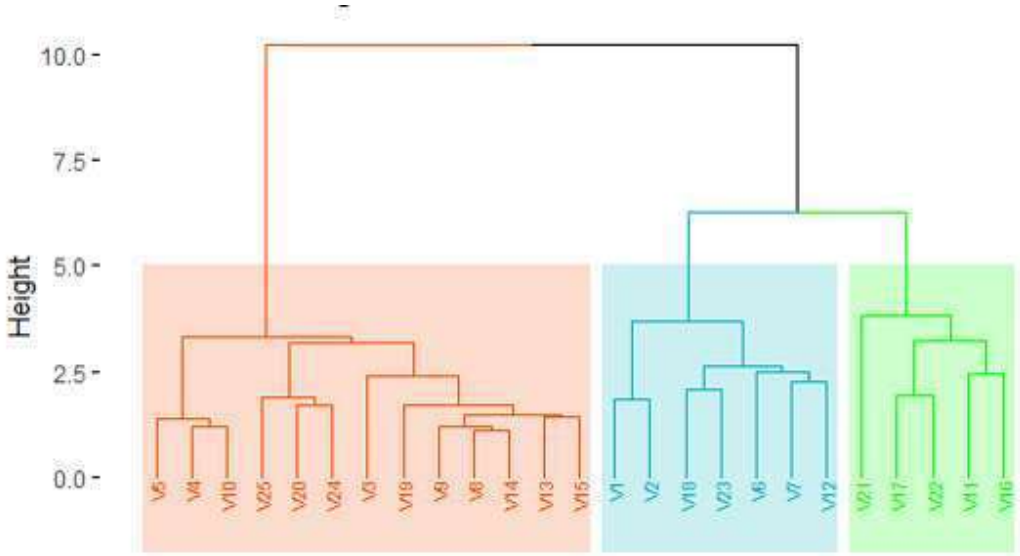

Figure 14: Dendrogramme des classes

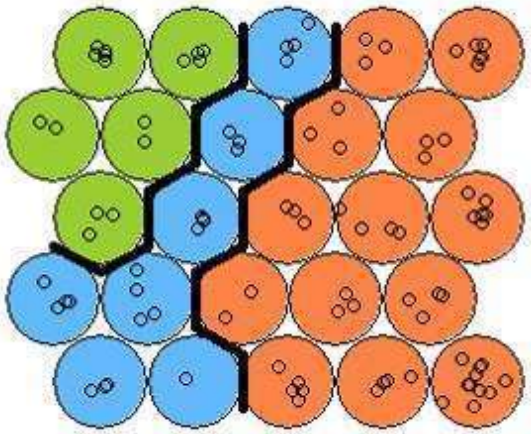

Figure 15: Représentation des classes dans SOM

La détermination des lignes au niveau des nœuds permet de regrouper les lignes en trois classes différentes à savoir : Pas de perte d'épaisseur, Perte d'épaisseur moyenne, Perte d'épaisseur élevée comme représenté dans le tableau 1.

Tableau 1: Répartition des lignes selon les 3 classes

Pas de perte d'épaisseur	Perte d'épaisseur moyenne	Perte d'épaisseur élevée
L1 ; L2 ; L13 ; L16 ; L17 ; L22 ; L23 ; L24	L3 ; L4 ; L5 ; L6 ; L7 ; L8 ; L9 ; L10 ; L11 ; L12 ; L14 ; L15	L18 ; L19 ; L20 ; L21

Chapitre 4 : Article scientifique 2

Dans cette partie, nous présentons notre deuxième article (en annexe 2) intitulé **Walk-through corrosion assessment of slurry pipeline using machine learning** soumis dans ce spécial numéro de MDPI **« Corrosion Prediction and Corrosion Protection ».**

Résumé de l'article. L'étude de la corrosion des pipelines est cruciale pour prévenir les pertes économiques, la dégradation de l'environnement et la sécurité des travailleurs. Dans cette étude, plusieurs méthodes d'apprentissage automatique telles que l'élimination récursive des caractéristiques, l'ACP, GBM, SVM, RF, KNN et MLP ont été utilisées pour estimer la perte d'épaisseur d'un pipeline. Ces différents modèles d'apprentissage automatique ont été appliqués aux données brutes (l'ensemble des variables), aux données avec des variables sélectionnées par la méthode de l'élimination récursive des paramètres et aux données avec les variables sélectionnées par l'ACP. Une analyse comparative a été effectuée pour déterminer l'influence de la sélection et de la transformation des données sur la performance des modèles. Les résultats montrent que les modèles sont plus performants sur les données avec sélection de variables par la méthode d'élimination récursive et que les meilleurs modèles sont : RF, SVM et GBM avec une racine d'erreur quadratique moyenne (ou Root Mean Square Error en anglais) (RMSE) de 0,017. En modifiant les hyperparamètres, le modèle SVM devient le meilleur modèle avec un RMSE de 0,011 et un coefficient de détermination (R^2) de 0,83.

4.1 Analyse des résultats

Cette partie présente les résultats et la discussion de l'article scientifique 2.

4.1.1 Traitement et analyse des données

Les données utilisées pour cette étude ont été obtenues auprès de Agnico Eagle Mine Goldex. Il s'agit de mesures d'épaisseur sur 8 positions du pipeline (figure 16) qui est utilisé pour transporter les résidus (pulpe) du concentrateur au site du parc à résidus Manitou appartenant au Ministère de l'Énergie et des Ressources Naturelles. Aussi des variables opérationnelles du pipeline et environnementale sont collectées par des sondes. Le tableau 2 présente les différentes variables et leurs statistiques descriptives.

Tableau 2: Liste des variables

Variables	Min	Max	Moyenne	Ecart type
Tonnage sag (t)	8,78	404,39	342,65	65,15
Température de la pulpe Flottation (degré)	13,92	34,92	23,07	4,68
pH de flottation	7,47	9,33	9,03	0,27
Débit du résidu (m^3/h)	18,54	495,97	437,59	90,22
% de résidu solide	0,02	46,13	23,54	10,17
TPH résidu calculé	0,02	301,95	141,31	68,20
Pression Km 0 (Pa)	624,61	3439,54	2209,96	618,89
T° Km 0 (degré)	3,76	30,86	16,25	5,70
Débit m^3/h (Rivière Thompson)	30,00	381,19	207,72	110,66
T° (rivière Thompson) (degré)	0,99	20,45	13,42	6,04
Débit m^3/h (bassin de sédimentation)	26,88	122,30	68,82	16,17
T° (bassin de sédimentation) (degré)	2,71	21,55	9,58	5,71
Débit m^3/h (South Park)	0,04	369,80	168,89	95,23
T° (Parc Sud) (degré)	0,73	16,23	5,21	4,53
Épaisseur (mm)	5,75	6,01	5,84	0,03

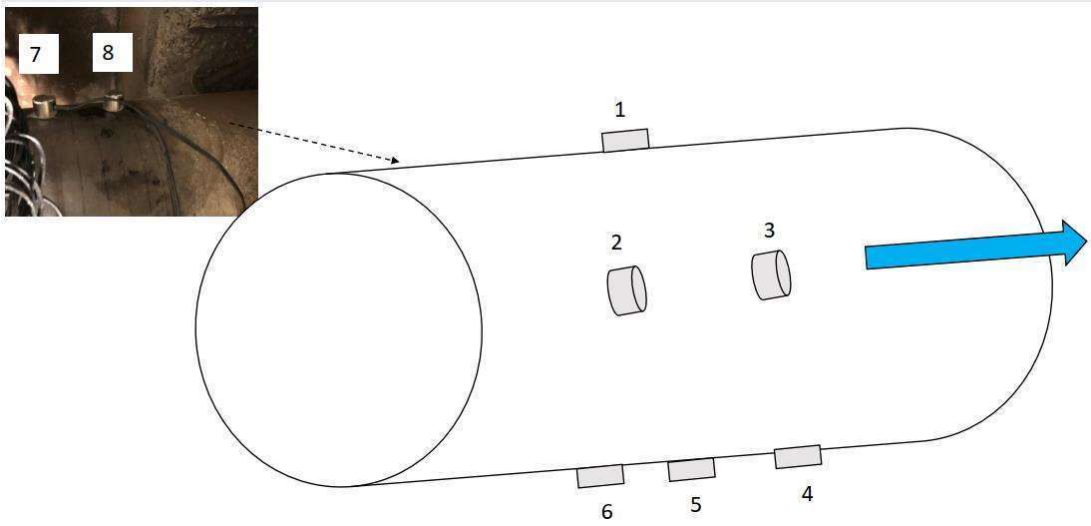

Figure 16: Position sur le tuyau

A ce stade du projet, les données s'étalaient sur 190 jours (entre 14/12/2021 et 21/06/2022). Les variables opérationnelles et environnementales sont collectées par 5 minutes tandis les mesures d'épaisseurs sont collectées par 24 h. Une base de données comprenant les mesures d'épaisseurs et les valeurs moyennes des variables opérationnelles et environnementales a été construite pour les besoins de la modélisation. Le but était de développer un modèle performant permettant de prédire l'épaisseur du pipeline. Les premiers travaux consistaient à traiter les données. Ainsi, l'analyse de la base de données a montré que les données étaient complètes et aussi toutes étaient continues (figure 17).

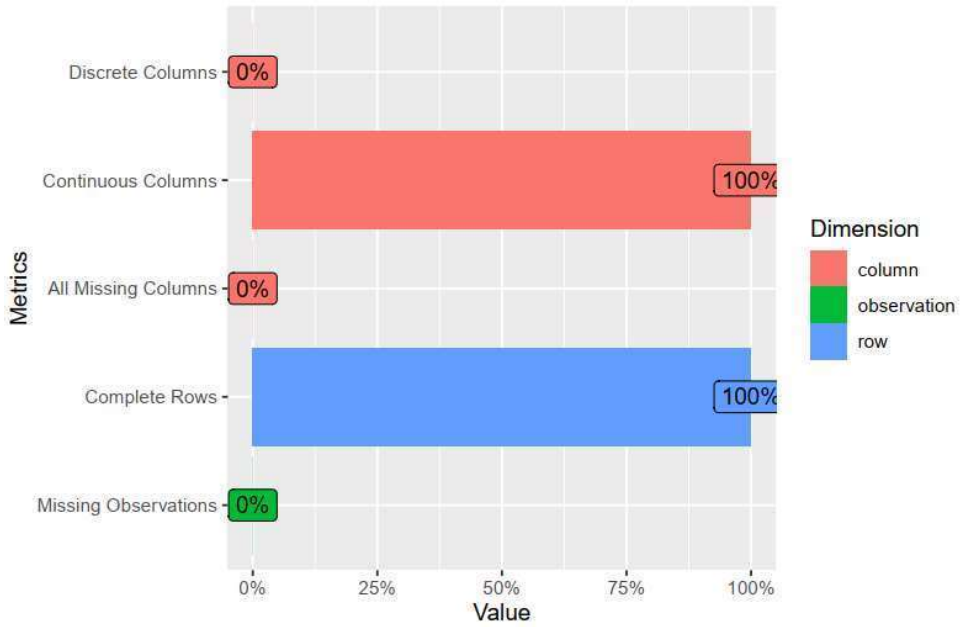

Figure 17: Visualisation des données

L'analyse des données d'entrée est très importante dans le processus de modélisation. Certains modèles exigent une certaine distribution des données d'entrée pour obtenir des résultats performants. Ainsi, un histogramme (figure 18) est construit pour chaque variable afin d'étudier graphiquement leur normalité. Il en ressort que certaines variables, telles que le % de solide résiduel, le TPH résiduel calculé et le débit m^3/h BS, ont une distribution proche de la normale. Pour les modèles qui requièrent une distribution normale des données, nous appliquerons des transformations à certaines variables afin de leur donner une distribution gaussienne.

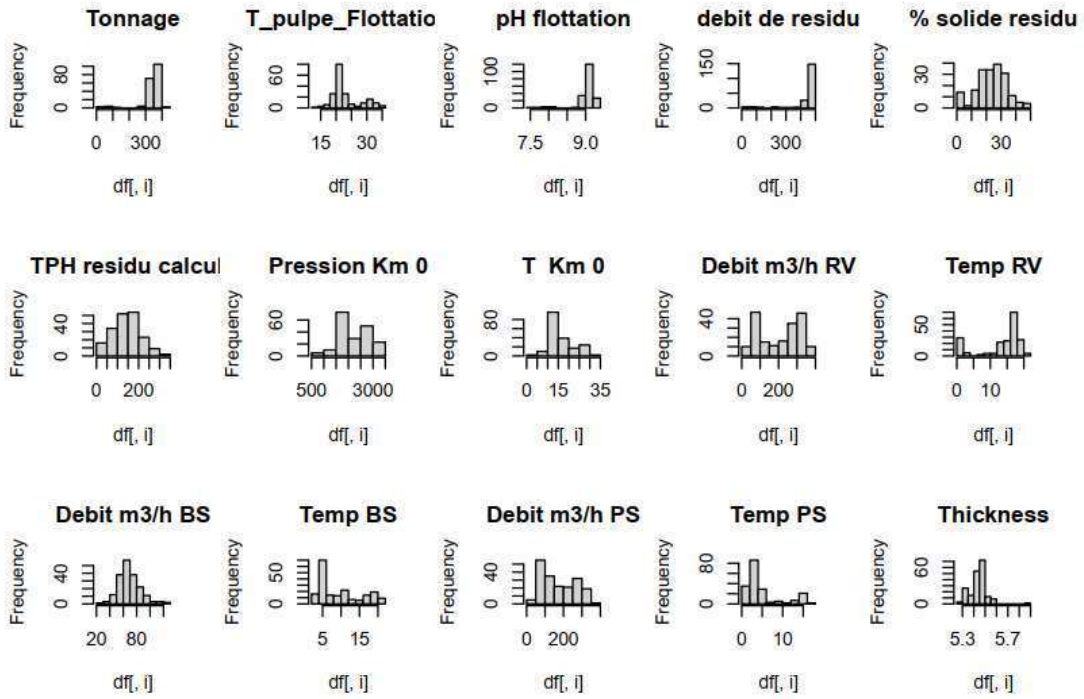

Figure 18: Histogramme des variables

L'analyse des boîtes à moustaches (figure 19) a permis de détecter certaines variables présentant des données aberrantes, telles que le pH de flottation, le débit de résidu, le tonnage, le débit m^3/h BS et le temps PS. Cette distribution peut être due à des problèmes de mesure causés par un dysfonctionnement des appareils ou des variations saisonnières. Ainsi, nous avons discuté avec des experts du domaine pour comprendre les causes de cette distribution des données et apporté les corrections nécessaires.

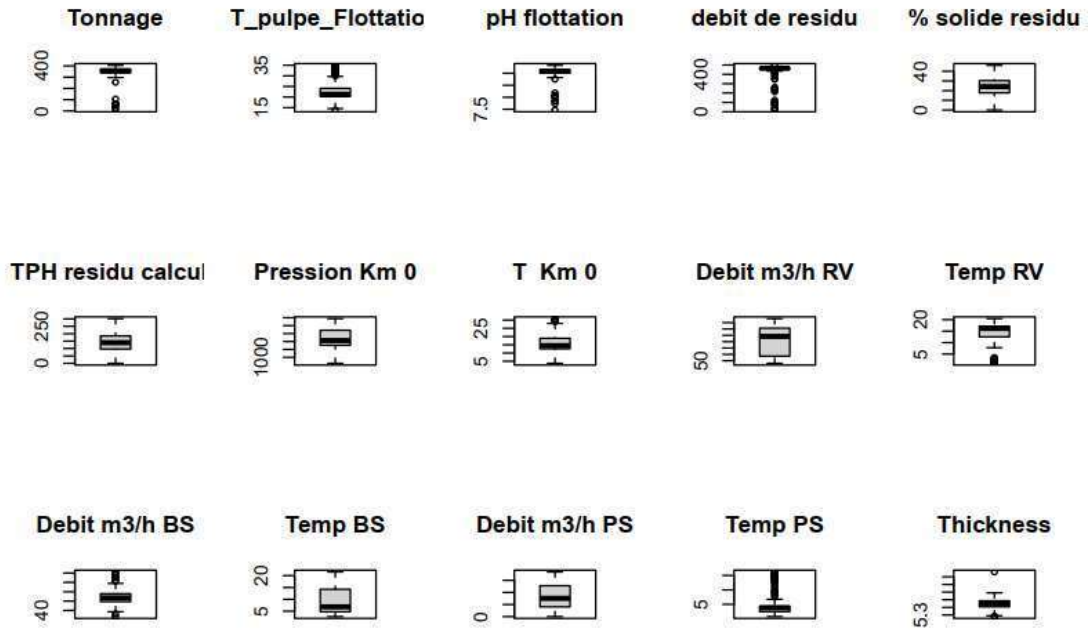

Figure 19: Boite à moustache des variables

4.1.2 Analyse des résultats de la modélisation

Plusieurs études ont été réalisées dans le domaine de la corrosion en utilisant plusieurs types de données. Les caractéristiques les plus rencontrées sont les caractéristiques chimiques du pipeline telles que la pression partielle de CO2, le type d'inhibiteur de corrosion [27], la concentration en ions sulfate et la concentration en ions chlorure [1]. Dans cette étude, nous avons collecté d'autres types de variables directement liées au pipeline (ph, débit résiduel, pression, TPH résiduel calculé) et des variables externes au pipeline comme le débit et la température des rivières et du bassin de sédimentation qui alimentent le pipeline. Nous avons réalisé cinq modèles sur trois ensembles de données différentes. Le premier modèle est effectué sur la base de données contenant l'ensemble des variables. Le deuxième considère les variables sélectionnées par la méthode d'élimination récursive des caractéristiques et le dernier utilise les composantes principales comme variables explicatives dans le modèle.

Les métriques utilisées pour évaluer la performance des modèles sont le RMSE et le coefficient de détermination (R^2). Ils sont représentés par les formules suivantes :

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Où y_i, $\hat{y}_i$, $\bar{y}_i$ représentent la valeur mesurée, la valeur prédite et la valeur moyenne de l'épaisseur du pipeline, respectivement.

La technique de validation croisée (validation croisée 5-fold répétée 3 fois) est utilisée pour éviter autant que possible les erreurs aléatoires. Les données d'entrainement représentent 80 % et le reste (20 %) sera utilisé pour la validation.

Les résultats (voir tableau 3) montrent que les RMSE pour les modèles RF (0,017), GBM (0,018) et SVM (0,018) sont sensiblement égaux. Le tableau 4 montre les scores de signification statistique par paire. Les scores de signification statistique par paire (ou paires de signification statistique) sont souvent utilisés pour comparer la performance de différents modèles statistiques ou algorithmes d'apprentissage automatique. Ces scores permettent de calculer la probabilité que la différence de performance entre deux modèles soit due au hasard. La diagonale inférieure du tableau montre les p-valeurs pour l'hypothèse nulle (les distributions sont les mêmes). La diagonale supérieure du tableau montre la différence estimée entre les distributions. La p-valeur entre SVM et KNN et SVM et KNN est estimée à 0,02. Ainsi, au seuil de 5%, on peut affirmer que la différence de performance entre les modèles SVM et KNN et SVM et RF n'est pas due au hasard mais plutôt à une différence significative entre les modèles. Par contre, on ne peut

pas conclure à une telle différence entre les modèles SVM et GBM, et RF et GBM car la p-valeur est égale à 1 supérieure au seuil de 5%. Le R^2 du modèle neuronal est faible par rapport aux autres modèles, c'est-à-dire que les variables explicatives expliquent moins la variation de la perte d'épaisseur. Ces résultats sont obtenus avec les données d'entrainement. Nous appliquerons les données de validation à la fin lorsque nous aurons trouvé le meilleur modèle.

Tableau 3: Résultat de la modélisation avec l'ensemble des variables

	Training set	
Modèle	**RMSE**	**R^2**
SVM	0,018	0,667
GBM	0,018	0,672
RF	0,017	0,715
KNN	0,021	0,504
MLP	0,026	0,182

Tableau 4: Scores de signification statistique par paire

	SVM	KNN	RF	GBM
SVM		-0,003	0,001	0,001
KNN	0,02		0,004	0,004
RF	0,02	0,00		0,00
GBM	1,00	0,01	1,00	

Nous effectuons une deuxième modélisation en sélectionnant quelques variables. La sélection des variables pertinentes est très importante dans le processus de modélisation. Dans cette étude, nous avons utilisé l'élimination récursive des caractéristiques comme méthode de sélection des variables. Cette méthode ajuste un modèle et élimine la ou les caractéristiques les plus faibles jusqu'à ce que le nombre spécifié de caractéristiques soit atteint. Le modèle utilisé ici est le RF car il possède un bon mécanisme intégré pour calculer l'importance des caractéristiques. Cette méthode tente d'éliminer les dépendances et la colinéarité qui peuvent exister dans le modèle.

La figure 20 représente la variation de la RMSE en fonction du nombre de variables sélectionnées dans notre modèle. Le nombre optimal de variables est de 10 : Tonnage sag, Température de la pulpe Flottation, % résidu solide, TPH résidu calculé, Pression Km 0, Température au Km 0, Température (Rivière Thompson), Débit m^3/h (bassin de sédimentation), Température (bassin de sédimentation), Débit m^3/h (South Park), Température (Parc Sud) avec une RMSE de 0,017. Ces 10 variables ont été utilisées pour étudier les autres modèles. Les résultats (voir tableau 5) montrent une faible variation des paramètres de performance. Les résultats du test de signification pour les modèles RF, SVM et GBM montrent qu'il n'y a pas de différence entre ces modèles au seuil de 5% (p-valeur = 1). Cependant, ces modèles sont meilleurs que KNN et MLP. Le MLP fonctionne mieux dans les situations où la taille de l'échantillon est très grande, ce qui n'est pas le cas pour notre étude.

Tableau 5: Résultat avec sélection de variables

	Training set	
Modèle	RMSE	R^2
SVM	0,016	0,735
GBM	0,017	0,707
RF	0,017	0,725
KNN	0,027	0,274
MLP	0,027	0,074

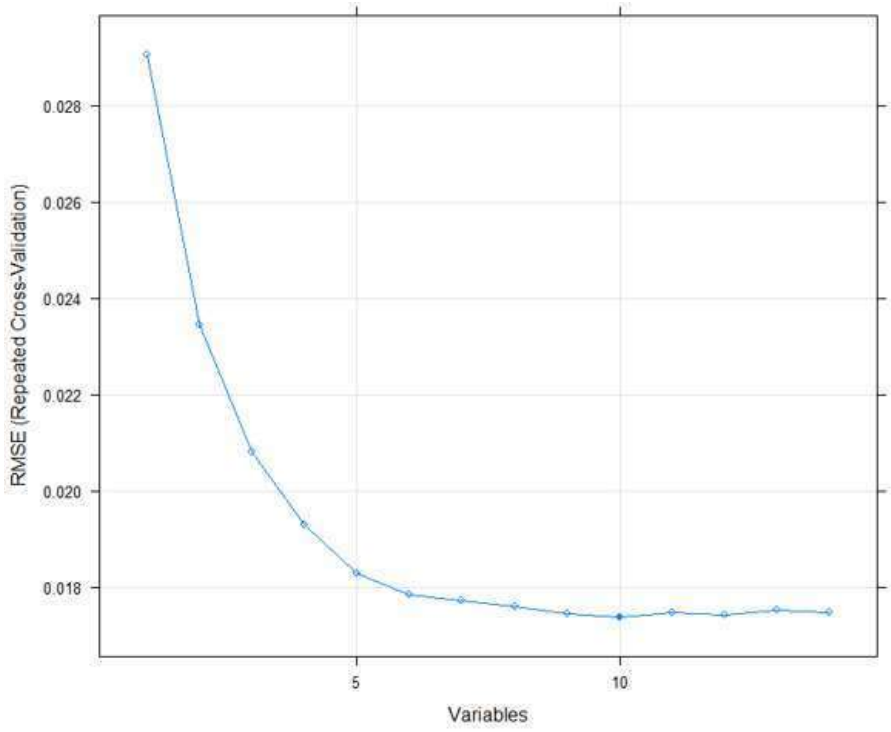

Figure 20: Elimination récursive des caractéristiques

Dans notre dernière modélisation, nous avons utilisé l'ACP comme méthode de sélection des variables en prenant comme variables explicatives, les composantes principales qui expliquent la plus grande variation. Nous avons effectué une ACP sur nos données d'entraînement et avons sélectionné les 8 composantes principales (figures 21 et 22) qui expliquent plus de 95% de la variation des données. Ces 8 composantes principales sont ensuite utilisées comme variables explicatives dans nos différents modèles. Les résultats (voir tableau 6) varient légèrement de ceux trouvés précédemment. Cependant, le modèle KNN performe mieux avec la transformation des données par l'ACP.

Tableau 6: Résultats avec la transformation des données par ACP

	Données d'entrainement	
Modèle	RMSE	R^2
SVM	0,018	0,66
GBM	0,02	0,51
RF	0,019	0,61
KNN	0,019	0,55

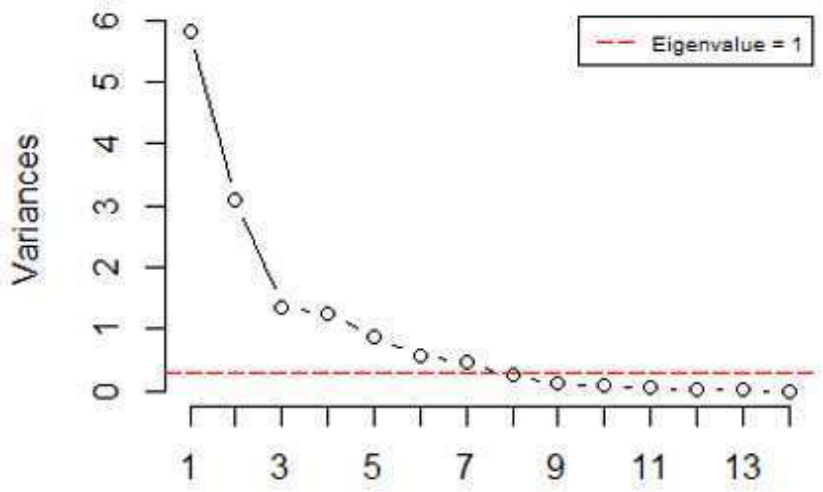

Figure 21: Variance des 14 composantes principales

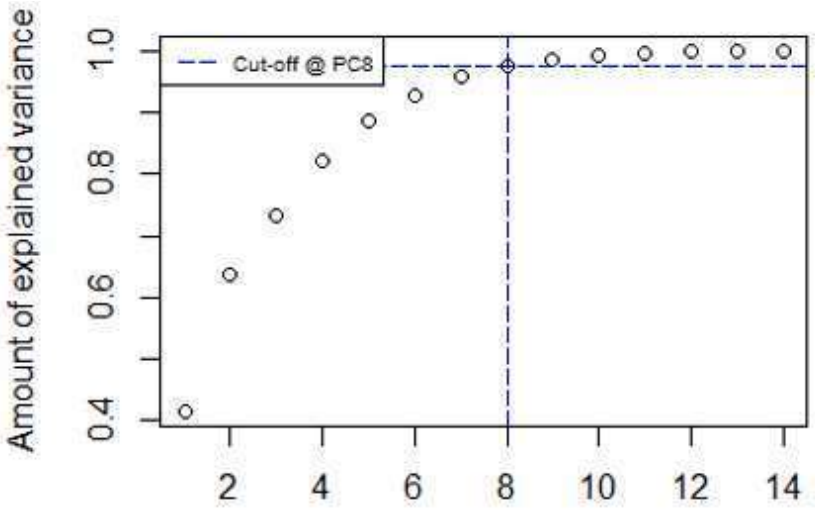

Figure 22: Variance cumulée des composantes principales

L'optimisation ou le réglage des hyperparamètres [51] est le problème qui consiste à choisir un ensemble d'hyperparamètres optimaux pour un algorithme d'apprentissage. Cette combinaison d'hyperparamètres maximise les performances du modèle, en minimisant une fonction de perte prédéfinie pour produire de meilleurs résultats avec moins d'erreurs. La méthode traditionnelle d'optimisation des hyperparamètres est la recherche sur grille (grid search en anglais) [52] ou le balayage des paramètres, qui consiste simplement en une recherche exhaustive dans un sous ensemble de l'espace des hyperparamètres d'un algorithme d'apprentissage spécifié manuellement. Un

algorithme de recherche sur grille doit être guidé par une certaine mesure de performance, généralement mesurée par validation croisée sur l'ensemble d'apprentissage. Pour le modèle SVM, nous recherchons la valeur optimale pour **C** et **Gamma**. **C** est le coût de la violation des contraintes. C'est la constante du terme de régularisation dans la formulation de Lagrange. Une valeur faible de **C** peut entraîner une classification incorrecte de certaines données d'apprentissage par le modèle, tandis qu'une valeur élevée peut entraîner un surajustement du modèle. Le surajustement crée une analyse trop spécifique pour l'ensemble de données actuel et éventuellement inadaptée aux données futures. **Gamma** est l'inverse du rayon d'influence des échantillons de données que nous avons sélectionnés comme vecteurs de support. Des valeurs élevées indiquent un petit rayon d'influence et de petites limites de décision qui ne tiennent pas compte des échantillons de données relativement proches. Ces valeurs élevées entraînent un surajustement. Des valeurs faibles indiquent l'effet significatif d'échantillons de données éloignés, de sorte que le modèle ne peut pas capturer les limites de décision correctes à partir de l'ensemble de données.

Pour le modèle RF, nous recherchons la valeur optimale de **Max_features** et de **N_estimators**. **Max_features** ou **mtry** est le nombre maximum de caractéristiques que le modèle est autorisé à essayer dans un arbre individuel. **N_estimators** ou **ntree** est le nombre d'arbres que nous souhaitons construire avant de prendre le vote maximum ou les moyennes des prédictions. Les meilleurs hyperparamètres pour le modèle RF sont : **max_features**=2 et **n_estimators**=2500. Avec ces hyperparamètres, la RMSE du modèle devient 0,016 et le R^2, 0,73. Nous pouvons voir dans la figure 23, la variation du RMSE en fonction de **max_features** avec **n_estimateurs**=2500.

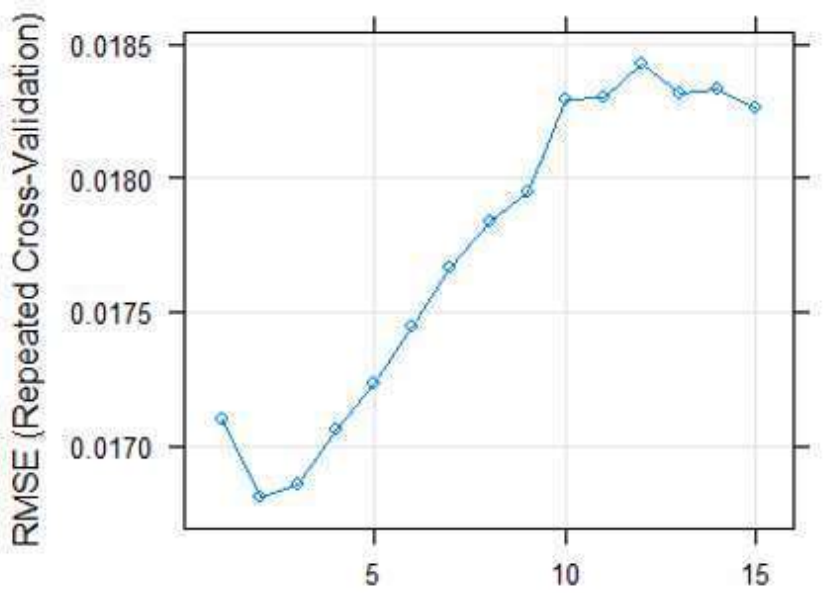

Figure 23: Réglage des hyperparamètres (max_features) - Modèle Forêt aléatoire

L'optimisation des hyperparamètres du modèle SVM produit des résultats plus intéressants. La figure 24 montre la variation du RMSE en fonction des hyperparamètres. Nous pouvons clairement voir que le plus petit RMSE (modèle plus performant) se trouve au point C (**cost**)=5 et Gamma (**sigma**)=0.05. Avec ces hyperparamètres, le RMSE est estimé à 0.0154. Le calcul du R^2 a donné 0.76. Ces estimations sont faites sur les données d'entraînement.

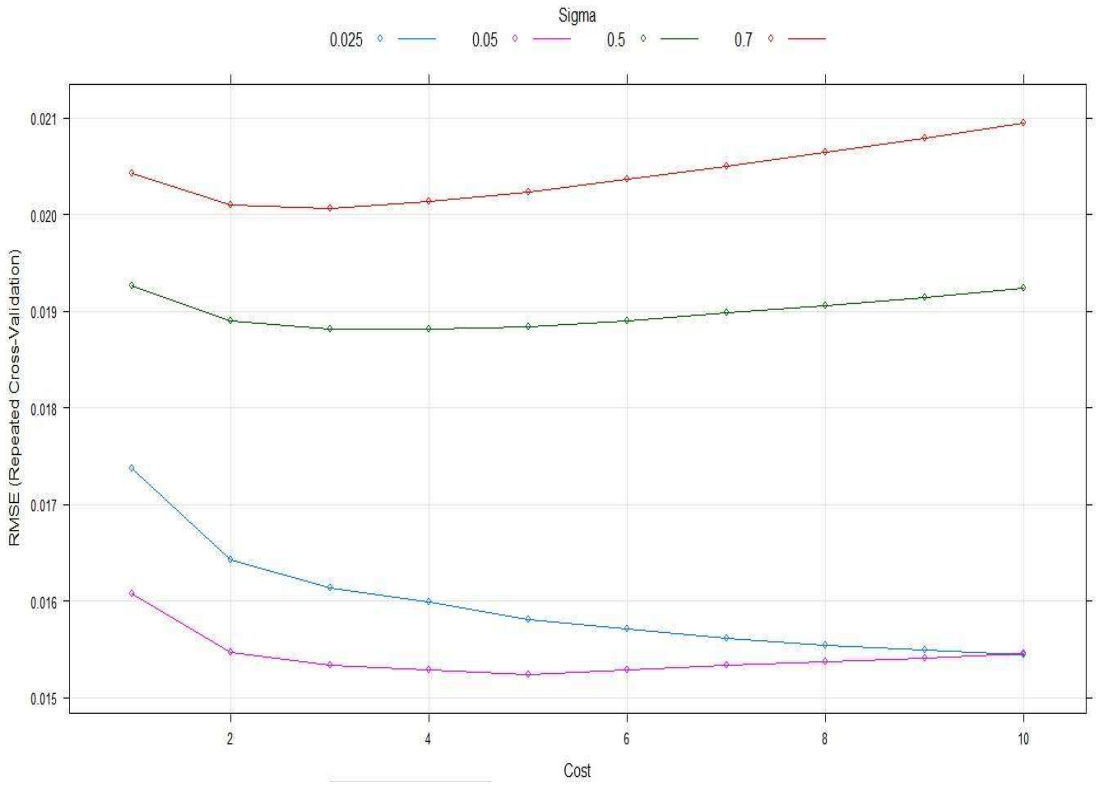

Figure 24: Réglage des hyperparamètres - Modèle Machine à vecteur support

Le modèle SVM semble être le meilleur modèle avec un RMSE plus faible que celui trouvé dans la plupart des modèles de la revue. Pour étudier la performance du modèle dans de nouvelles données, nous utilisons les données de validation (20% de l'ensemble des données) pour prédire les pertes d'épaisseur et estimer le RMSE. Le tableau 7 montre les résultats de l'estimation du modèle SVM sur les données de validation. Le modèle prédit bien la perte d'épaisseur avec un RMSE faible évalué à 0,011 et un R^2 de 0,83. Nous pouvons voir sur la figure 25 la corrélation entre les valeurs observées et prédites des mesures d'épaisseur sur le modèle de validation avec le meilleur modèle SVM et les hyperparamètres optimaux C=5 et sigma=0,05.

Tableau 7: Résultat sur les données de validation avec le modèle SVM

Model	**Validation set**	
	RMSE	**R^2**
SVM	0.011	0.83

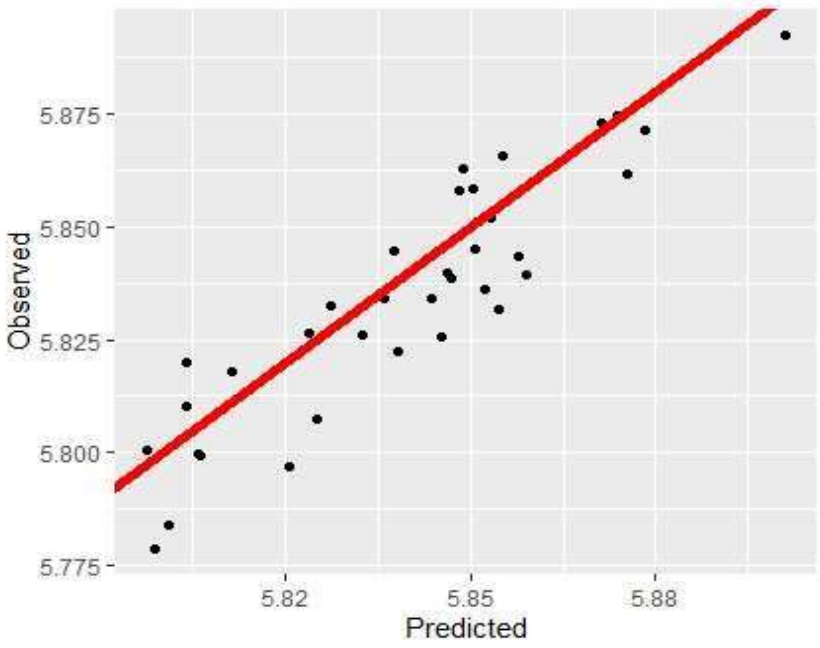

Figure 25: Epaisseur du pipeline : valeurs observées vs valeurs prédites

Chapitre 5 : Discussion Générale

Les articles 1 et 2 abordent le problème de la corrosion d'un pipeline de deux manières différentes. Le premier essaie d'identifier les positions du tuyau qui sont attaquées par le phénomène de la corrosion tandis que le deuxième essaie de prédire la perte d'épaisseur du pipeline. La corrosion des pipelines est un problème majeur dans l'industrie minière. En plus d'entraîner des pertes économiques considérables, elle peut également avoir des conséquences graves sur l'environnement et la sécurité des travailleurs. Ainsi, l'utilisation de méthodes d'apprentissage automatique pour estimer la perte d'épaisseur d'un pipeline peut être un moyen efficace de prévenir les accidents et les pertes économiques.

Dans les deux articles plusieurs modèles d'apprentissage automatique ont été utilisés pour estimer la perte d'épaisseur d'un pipeline ou pour identifier les zones à risque au niveau du pipeline. Dans le premier article, l'utilisation de Self Organising Maps nous a permis d'analyser les mesures d'épaisseurs et d'identifier les points qui se ressemblent au niveau du pipeline en termes de vitesse de fuite. L'identification des zones à risque pourrait être faite en calculant les minimums, maximums, quartiles, moyennes et médianes pour chaque ligne du pipeline, ce qui prendrait beaucoup de temps. L'avantage de l'utilisation du SOM combiné avec la CAH est que toutes les 24 lignes du pipeline seront analysées ensemble et regroupées en différentes classes indiquant le niveau de corrosion.

Dans le deuxième article, nous avons développé une méthodologie de modélisation de la perte d'épaisseur d'un pipeline en utilisant la majeure partie des modèles pour les problèmes de régression. À cause de la quantité, de la distribution et de la forme des données, nous avons ignoré certains modèles qui pourraient donner de meilleurs résultats. Les modèles utilisés ont été

appliqués aux données brutes ainsi qu'aux variables sélectionnées par l'élimination récursive des caractéristiques et l'ACP. Il faut rappeler qu'il existe plusieurs méthodes de sélection des variables. L'élimination récursive des caractéristiques a été mise en évidence dans cet article. Les résultats ont montré que les modèles sont plus performants sur les variables sélectionnées par l'élimination récursive des caractéristiques. L'optimisation des hyperparamètres a amélioré considérablement les résultats du modèle SVM.

Ces résultats suggèrent que les modèles d'apprentissage automatique peuvent être utilisés avec succès pour estimer la perte d'épaisseur des pipelines et qu'il est important de sélectionner les variables pertinentes et optimiser les hyperparamètres pour améliorer la performance du modèle. Cette étude souligne l'importance de la prévention de la corrosion des pipelines et l'utilisation de techniques avancées telles que l'apprentissage automatique pour améliorer la précision des estimations de perte d'épaisseur. Ces résultats pourraient également avoir des implications importantes pour l'industrie minière en termes de prévention des accidents et de réduction des pertes économiques. Enfin, cette étude peut encourager les chercheurs à explorer davantage l'utilisation de l'apprentissage automatique pour résoudre des problèmes similaires dans l'industrie.

Conclusion et Perspectives

Cette étude examine différentes techniques d'apprentissage automatique qui peuvent être utilisées pour modéliser les données dans le domaine minier. La corrosion est un phénomène courant dans les industries minières, ayant des impacts négatifs sur l'environnement, la santé des travailleurs et pouvant entraîner des pertes considérables pour l'industrie. Les modèles d'apprentissage automatique supervisé et non supervisé ont été utilisés pour étudier la corrosion des pipelines, avec cinq modèles supervisés et un modèle non supervisé. Les résultats ont montré que chaque modèle a des caractéristiques distinctes et performe différemment en fonction de la quantité de données, du nombre de caractéristiques et du réglage des hyperparamètres. Le perceptron multicouche est plus adapté pour les grandes bases de données, tandis que la sélection des caractéristiques avec la méthode d'élimination récursive des caractéristiques a permis de déterminer le nombre optimal de variables à considérer. L'optimisation des hyperparamètres est une étape cruciale de la modélisation, car chaque modèle a des hyperparamètres différents qui ont une grande influence sur les résultats finaux. Un bon réglage de ces hyperparamètres est nécessaire pour éviter le surajustement du modèle.

Bien que cette étude ait permis d'obtenir des résultats significatifs, elle présente encore certaines limites qui pourraient être surmontées dans des recherches futures. L'une de ces limites est la quantité limitée des données qui ne permet pas d'utiliser des modèles tels que les réseaux de neurones récurrents. En effet, les réseaux de neurones récurrents sont utilisés pour traiter des données séquentielles telles que des séquences de texte ou de temps. Ils ont une grande capacité de modélisation. Ils peuvent tenir en compte les informations précédentes pour prendre des décisions sur les données actuelles, ce qui en fait un choix naturel pour les tâches de prédiction. Nous pensons que

l'exploration et l'évaluation comparatives des différentes méthodes de modélisation des séries temporelles peuvent fournir des perspectives précieuses pour améliorer la précision de la prévision et la compréhension des dynamiques temporelles dans nos données. Cela pourrait potentiellement conduire à des résultats plus robustes et plus fiables qui pourraient enrichir le domaine d'étude.

Références

[1] C. I. Ossai, "A data-driven machine learning approach for corrosion risk assessment—a comparative study," *Big Data and Cognitive Computing*, vol. 3, no. 2, 2019, doi: 10.3390/bdcc3020028.

[2] T. Pootakham and A. Kumar, "Bio-oil transport by pipeline: A techno-economic assessment," *Bioresour Technol*, vol. 101, no. 18, 2010, doi: 10.1016/j.biortech.2010.03.136.

[3] Central Intelligence Agency, "CIA - The World Factbook - New Zealand," *CIA - The World Factbook*, 2021.

[4] H. Lu, T. Iseley, S. Behbahani, and L. Fu, "Leakage detection techniques for oil and gas pipelines: State-of-the-art," *Tunnelling and Underground Space Technology*, vol. 98. 2020. doi: 10.1016/j.tust.2019.103249.

[5] Z. Ahmad, *Principles of Corrosion Engineering and Corrosion Control*. 2006. doi: 10.1016/B978-0-7506-5924-6.X5000-4.

[6] "EGIG 11th Report of the European Gas Pipeline Incident Data Group," Dec. 2021.

[7] F. Caleyo, L. Alfonso, J. Alcántara, and J. M. Hallen, "On the estimation of failure rates of multiple pipeline systems," in *Journal of Pressure Vessel Technology, Transactions of the ASME*, 2008. doi: 10.1115/1.2894292.

[8] E. Bowman *et al.*, "International Measures of Prevention, Application, and Economics of Corrosion Technologies Study," 2016.

[9] C. I. Ossai, "Pipeline corrosion prediction and reliability analysis : A systematic approach with Monte Carlo Simulation and Degradation models," *International Journal of Scientific & Technology Research*, vol. 2, no. 3, 2013.

[10] A. R. Kowalski, J. C. Ruiz-Rico, and S. R. Finneran, "A successful methodology following API 570 & ANSI/NACE SP0502 for piping integrity assessment," in *NACE - International Corrosion Conference Series*, 2012.

[11] Y. Diao, L. Yan, and K. Gao, "Improvement of the machine learning-based corrosion rate prediction model through the optimization of input features," *Mater Des*, vol. 198, 2021, doi: 10.1016/j.matdes.2020.109326.

[12] S. Nesic, M. Nordsveen, N. Maxwell, and M. Vrhovac, "Probabilistic modelling of CO2 corrosion laboratory data using neural networks," *Corros Sci*, vol. 43, no. 7, 2001, doi: 10.1016/S0010-938X(00)00157-8.

[13] W. T. Nash, C. J. Powell, T. Drummond, and N. Birbilis, "Automated corrosion detection using crowdsourced training for deep learning," *Corrosion*, vol. 76, no. 2, 2020, doi: 10.5006/3397.

[14] G. Sanchez, W. Aperador, and A. Cerón, "Corrosion grade classification: a machine learning approach," *Indian Chemical Engineer*, vol. 62, no. 3, 2020, doi: 10.1080/00194506.2019.1675539.

[15] L. Yan, Y. Diao, Z. Lang, and K. Gao, "Corrosion rate prediction and influencing factors evaluation of low-alloy steels in marine atmosphere using machine learning approach," *Sci Technol Adv Mater*, vol. 21, no. 1, 2020, doi: 10.1080/14686996.2020.1746196.

[16] B. A. Salami, S. M. Rahman, T. A. Oyehan, M. Maslehuddin, and S. U. Al Dulaijan, "Ensemble machine learning model for corrosion initiation time estimation of embedded steel reinforced self-compacting concrete," *Measurement (Lond)*, vol. 165, 2020, doi: 10.1016/j.measurement.2020.108141.

[17] X. Gong, C. Dong, J. Xu, L. Wang, and X. Li, “Machine learning assistance for electrochemical curve simulation of corrosion and its application,” *Materials and Corrosion*, vol. 71, no. 3, 2020, doi: 10.1002/maco.201911224.
[18] A. Abass, K. Wada, H. Matsunaga, H. Remes, and T. Vuorio, “Quantitative Characterization of the Spatial Distribution of Corrosion Pits Based on Nearest Neighbor Analysis,” *Corrosion*, vol. 76, no. 9, 2020, doi: 10.5006/3551.
[19] R. Aulia, H. Tan, and S. Sriramula, “Prediction of corroded pipeline performance based on dynamic reliability models,” in *Procedia CIRP*, 2019. doi: 10.1016/j.procir.2019.01.093.
[20] M. Terrados-Cristos, F. Ortega-Fernández, G. Alonso-Iglesias, M. Díaz-Piloneta, and A. Fernández-Iglesias, “Corrosion prediction of weathered galvanised structures using machine learning techniques,” *Materials*, vol. 14, no. 14, 2021, doi: 10.3390/ma14143906.
[21] F. Hassan, A. K. Mahmood, L. Ab Rahim, M. Syed, A. Saboor, and M. Rimsan, “Clustering-Based Quantitative Evaluation Using Acoustic Emission Waveforms for Corrosion Detection,” *Hunan Daxue Xuebao/Journal of Hunan University Natural Sciences*, vol. 48, pp. 320–329, Aug. 2021.
[22] A. Roy, M. F. N. Taufique, H. Khakurel, R. Devanathan, D. Johnson, and G. Balasubramanian, “Machine-learning-guided descriptor selection for predicting corrosion resistance in multi-principal element alloys,” *Npj Mater Degrad*, vol. 6, p. 9, Jan. 2022, doi: 10.1038/s41529-021-00208-y.
[23] J. Zhang, M. Zhang, B. Dong, and H. Ma, “Quantitative evaluation of steel corrosion induced deterioration in rubber concrete by integrating ultrasonic testing, machine learning and mesoscale simulation,” *Cem Concr Compos*, vol. 128, 2022, doi: 10.1016/j.cemconcomp.2022.104426.
[24] J. C. Velázquez, E. Hernández-Sánchez, G. Terán, S. Capula-Colindres, M. Diaz-Cruz, and A. Cervantes-Tobón, “Probabilistic and Statistical Techniques to Study the Impact of Localized Corrosion Defects in Oil and Gas Pipelines: A Review,” *Metals*, vol. 12, no. 4. 2022. doi: 10.3390/met12040576.
[25] X. Wei, D. Fu, M. Chen, W. Wu, D. Wu, and C. Liu, “Data mining to effect of key alloying elements on corrosion resistance of low alloy steels in Sanya seawater environmentAlloying Elements,” *J Mater Sci Technol*, vol. 64, 2021, doi: 10.1016/j.jmst.2020.01.040.
[26] Z. Zhao, M. Chen, H. Fan, and N. Zhang, “Data Analysis and Knowledge Mining of Machine Learning in Soil Corrosion Factors of the Pipeline Safety,” *Comput Intell Neurosci*, vol. 2022, May 2022, doi: 10.1155/2022/9523878.
[27] M. Aghaaminiha *et al.*, “Machine learning modeling of time-dependent corrosion rates of carbon steel in presence of corrosion inhibitors,” *Corros Sci*, vol. 193, 2021, doi: 10.1016/j.corsci.2021.109904.
[28] S. Peng, Z. Zhang, E. Liu, W. Liu, and W. Qiao, “A new hybrid algorithm model for prediction of internal corrosion rate of multiphase pipeline,” *J Nat Gas Sci Eng*, vol. 85, 2021, doi: 10.1016/j.jngse.2020.103716.
[29] A. Bouchier, “La classification ascendante hiérarchique (C.A.H),” *L’analyse des données multivariées à l’aide du logiciel R*. 2006.
[30] M. Charrad, N. Ghazzali, V. Boiteau, and A. Niknafs, “NbClust: An R Package for Determining the Relevant Number of Clusters in a Data Set,” *J Stat Softw*, vol. 61, pp. 1–36, Oct. 2014, doi: 10.18637/jss.v061.i06.
[31] D. Si, W. Hu, Z. Deng, and Y. Xu, “Fair hierarchical clustering of substations based on Gini coefficient,” *Global Energy Interconnection*, vol. 4, no. 6, 2021, doi: 10.1016/j.gloei.2022.01.009.

[32] W. Hu and Q. he Pan, “Data clustering and analyzing techniques using hierarchical clustering method,” *Multimed Tools Appl*, vol. 74, no. 19, 2015, doi: 10.1007/s11042-013-1611-9.
[33] T. Kohonen, “Self-organized formation of topologically correct feature maps,” *Biol Cybern*, vol. 43, no. 1, pp. 59–69, 1982, doi: 10.1007/BF00337288.
[34] T. Kohonen, “Essentials of the self-organizing map,” *Neural Networks*, vol. 37, 2013, doi: 10.1016/j.neunet.2012.09.018.
[35] T. Kohonen and T. Honkela, “Kohonen network,” *Scholarpedia*, vol. 2, no. 1, 2007, doi: 10.4249/scholarpedia.1568.
[36] T. M. Cover and P. E. Hart, “Nearest Neighbor Pattern Classification,” *IEEE Trans Inf Theory*, vol. 13, no. 1, 1967, doi: 10.1109/TIT.1967.1053964.
[37] T. Ho, *Random decision forests*, vol. 1. 1995. doi: 10.1109/ICDAR.1995.598994.
[38] L. Breiman, “Random Forests,” *Mach Learn*, vol. 45, pp. 5–32, Oct. 2001, doi: 10.1023/A:1010950718922.
[39] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and regression trees*. 2017. doi: 10.1201/9781315139470.
[40] J. Ross Quinlan, “Induction of decision trees,” *Mach Learn*, vol. 1, 1986.
[41] J. R. Quinlan, “C4.5: Programs for Machine Learning (Morgan Kaufmann Series in Machine Learning),” *Morgan Kaufmann San Mateo California*, 1992.
[42] C. Cortes and V. Vapnik, “Support-vector networks,” *Chem. Biol. Drug Des.*, vol. 297, pp. 273–297, Jan. 2009, doi: 10.1007/%2FBF00994018.
[43] C. Cortes and V. Vapnik, “Support-vector networks,” *Mach Learn*, vol. 20, no. 3, pp. 273–297, 1995, doi: 10.1007/BF00994018.
[44] M. Awad and R. Khanna, *Efficient learning machines: Theories, concepts, and applications for engineers and system designers*. 2015. doi: 10.1007/978-1-4302-5990-9.
[45] J. H. Friedman, “Greedy function approximation: A gradient boosting machine,” *Ann Stat*, vol. 29, no. 5, 2001, doi: 10.1214/aos/1013203451.
[46] H. Lu, S. Karimireddy, N. Ponomareva, and V. Mirrokni, *Accelerating Gradient Boosting Machine*. 2019.
[47] S. Touzani, J. Granderson, and S. Fernandes, “Gradient boosting machine for modeling the energy consumption of commercial buildings,” *Energy Build*, vol. 158, 2018, doi: 10.1016/j.enbuild.2017.11.039.
[48] M. Bélanger, N. El-Jabi, D. Caissie, F. Ashkar, and J. M. Ribi, “Estimation de la température de l’eau de rivière en utilisant les réseaux de neurones et la régression linéaire multiple,” *Revue des sciences de l’eau*, vol. 18, no. 3, 2005, doi: 10.7202/705565ar.
[49] J. Schmidhuber, “Deep Learning in neural networks: An overview,” *Neural Networks*, vol. 61. 2015. doi: 10.1016/j.neunet.2014.09.003.
[50] N. Lange, C. M. Bishop, and B. D. Ripley, “Neural Networks for Pattern Recognition.,” *J Am Stat Assoc*, vol. 92, no. 440, 1997, doi: 10.2307/2965437.
[51] M. Feurer and F. Hutter, “Hyperparameter Optimization,” 2019, pp. 3–33. doi: 10.1007/978-3-030-05318-5_1.
[52] F. Tang, Y. Wu, and Y. Zhou, “Hybridizing Grid Search and Support Vector Regression to Predict the Compressive Strength of Fly Ash Concrete,” *Advances in Civil Engineering*, vol. 2022, p. 3601914, 2022, doi: 10.1155/2022/3601914.

Annexe A : Article scientifique 1

Unsupervised Neural Network for Data-Driven Corrosion Detection of a Mining Pipeline

Abdou Khadir Dia[1], Nadia Ghazzali[1], Axel Gambou Bosca[2]
[1]University of Québec at Trois Rivières, Department of Mathematics and Computer Science
(abdou.khadir.dia@uqtr.ca, Nadia.Ghazzali@uqtr.ca)
[2] Québec Mettalurgy Centre
(axel.gambou.bosca@cegeptr.qc.ca)

Abstract

Pipelines failure often caused by corrosion may result in safety, environmental and economic issues. In this study, an unsupervised neural network, Self-Organizing Maps (SOM), is applied to create clusters representing the corrosion impact assessed with ultrasound periodic inspections. Based on this work, it is expected that the new insight into thickness data representation using unsupervised neural network will facilitate planning of corrosion mitigation activities through risk-based inspections of mining slurry pipelines. As a result, SOM led to the reduction of the variables in two-dimensional space nodes. Hierarchical ascending classification (HAC) was then used to classify these nodes regrouping thickness loss measurements. The proposed method by combining both SOM and HAC succeeded in detecting the extent of corrosion in a mining pipeline.

Introduction

Globally and domestically the long-term sustainability and viability of both the mining industry and its related communities are of the utmost importance. Improving environmental performance and mitigating environmental impacts of mining are critical to ensure the social health and welfare of associated communities. In regard to their safety, efficiency and low cost, pipelines are widely used in transporting large quantities of oil and gas, minerals or oil sands slurry over long distances (Okonkwo and Adel 2014). As such, pipelines are critical assets of our civil infrastructure. Pipelines may suffer from different types of defects such as corrosion, fatigue cracks, stress corrosion cracking (SCC), bacterial corrosion, slurry erosion-corrosion, etc. (Raheem 2020). These defects, if not properly managed, may result in the asset failures including leak or rupture, which could lead to environmental hazards and very expensive downtime.

The overall annual corrosion cost (direct and indirect) in Canada was estimated to be approximately $46.4 billion in 2003 (Lou and al. 2003) which accounts for about 2.5% of the GDP. Furthermore, the impact study published by NACE International in 2016 estimated the global cost of corrosion to be $2.5 trillion, or 3.4% of the GDP by country (Koch and al. 2006). Most important was that it was demonstrated that 15 to 35% of the cost of corrosion could be saved using currently available corrosion control technologies and practices.

Corrosion is a very complex phenomenon based on the degradation of a material or its properties due to its reaction with the environment (Ahmad 2006). This degradation involves multiple factors (Chico and al. 2017), particles (Yin and al. 2020) and variables. It is a general understanding that facility piping should be inspected for in-service damage such as corrosion. Estimation of pipeline corrosion is fundamental to the analysis of pipeline reliability (Ossai 2013). To do so, a methodology that compromises the American Petroleum Institute (API) Piping Inspection Code and the National Association of Corrosion Engineers (NACE) Direct Assessment Process is applied since 2005 (Kowalski 2012). The corrosion of pipelines can be described as a systematic degradation of the pipeline wall due to the actions of operating parameters on the pipeline material (Ossai 2013). Most of the existing methods employ non-destructive evaluation techniques such as ultrasound testing (UT) waves to detect wall thickness loss and thus to predict the remaining asset life. For effective monitoring of pipeline reliability and remaining life prediction therefore, corrosion risk assessment is necessary. The advancement of technology such as the use of new data collection tools has allowed researchers to develop many methods to better understand the behavior of the collected data. In the field of corrosion, many methods have been used in recent years either to predict the corrosion rate (Nikoo and al. 2017, Cristos and al. 2021), or to cluster data (Hassan and al. 2021) in order to detect corrosion (loss of thickness in a pipeline for example). Roy and al. (2022) use the Gradient Boosting Regressor to predict corrosion resistance in multi-principal element alloys.

Among the machine learning and deep learning methods, depending on the available data, a supervised or unsupervised learning (Cristos and al. 2021) can be done. In the literature, these two methods have been used to model corrosion (Taffese and Sistonen 2016). Cristos and al. (2021) develop various models for predicting galvanized coated steel corrosion damage of metal structures exposed to weathering. They use Multivariate Adaptive Regression

Splines (MARS) to complete data-processing and Self-Organising Maps (SOM) (Kohonen 2013) including various layers (supersom) of both supervised and unsupervised learning to define the first-year corrosion loss of galvanized steel. A variant of SOM called Self-Organising Feature Map (SOFM) has been successfully used by Mohamed and al. (2015) as feature visualization tool for the purpose of selecting the most appropriate features produced by Magnetic Flux Leakage (MFL) in defect depth estimation of oil and gas pipelines. Later, Nikoo (2017) used SOFM to predict the corrosion current density in reinforced concrete. To prioritize inspection according to the permissible risk level involves the understanding of the consequences of failure of a component on a system and then predict the mean time for failure with numerical tools. Hatami and al (2016) consider temperature CO2 partial pressure, flow rate, and pH as inputs to study corrosion for oil pipelines using Support Vector Regression (SVR). Lunchun (2020) use machine learning method to simulate the marine atmospheric corrosion behavior of low-alloy steels. Abbas and al. (2018) applied the neural network method to the pipeline corrosion prediction. The prediction results were within the 95% confidence range, with the accuracy of ±3. Recently, Peng and al. (2020) proposed a new hybrid intelligent algorithm to predict the corrosion rate of the multiphase flow pipeline. The proposed model combines support vector regression (SVR), principal component analysis (PCA), and chaos particle swarm optimization (CPSO). Thus, PCA is utilized to reduce the data dimension and CPSO to optimize the hyperfine parameters in SVR.

While recent corrosion studies focus on the prediction of the corrosion rate (thickness loss/year) in the presence of various operating conditions, the primary objective of this work is to combine SOM with Hierarchical Ascending Classification (HAC) to better visualize the corrosion impact assessed with ultrasound periodic inspections. This to render UT a more efficient cost-effective approach to corrosion risk assessment. In fact, the present study focuses on a single variable (pipeline thickness) which is measured on 125 points of the pipeline representing sub-variables. SOM is used to aggregate the data obtained from the periodic nondestructive evaluation (NDE) of the pipeline, reduce the dimensionality to be able to represent these data on a space of dimension 2. Then, the unsupervised learning method HAC is used to create clusters at the nodes defined in the SOM. These nodes group the original data (rows 1-24 of the pipeline for each year) which are then grouped into clusters representing the corrosion level.

Materials and Methods

Data

Data for this study were obtained from Agnico Eagle Mine Goldex. These are thickness measurements of a pipeline that is used to transport residue (pulp) from the concentrator to the Manitou Residue Park site owned by the MERN (Ministère de l'Énergie et des Ressources Naturelles). The pipeline is 23 km in total (14 km steel and 9 km HDPE). A yearly excavated 3m section of the pipeline has been used to assess the residual wall thickness by UT analysis since 2016. The measurements were made using an ultrasonic thickness gauge MMX-6 DL (Dakota Ultrasonics, USA). The gauge was primarily calibrated using a standard block at different thicknesses. To make the thickness measurements, the circumference of the pipeline section was subdivided into 24 equidistant markers from which lines were drawn along the length of the pipeline. Marker points separated by 1 inch were marked along the 24 lines. Overall, 125 markers were marked on each line, ranging from 1 (start) to 125 (end) for a total of 3000 markers (125 x 24) on the pipeline surface for thickness measurements. Points 1 to 125 represent the variables and lines 1 to 24 are the observations. Indeed, all 125 variables are thickness measurements. This work deals with the data collected from 2016 to 2019.

Data Analysis

Figure 1 shows the minimum thickness values measured for the different lines on the inspected pipeline. The dimensional control of the wall thickness is +15% to -12.5% of the nominal thickness, which is comprised between 7.3 and 5.6 mm. The average nominal thickness is 6.25 mm. The minimum thickness value, which is the smallest of the 125 values collected for a given line, is important for analyzing the severity of corrosion. Indeed, short-term and long-term corrosion rates are calculated between previous and actual inspections in accordance with API 570. Thus, the minimal value is used to assess the time to leak for a given pipeline. Although, all lines except L4, L5, L9, L10, L11 have minimum thickness values below the allowed limit (5.6 mm), lines 18, 19 and 20 are more critical with minimum thickness values less than or equal to 4.4 mm. Hence, it will be expected for these lines to exhibit a short time to leak since there is a widely held belief that process is a simple one, where a pipeline corrodes to the point at which it can no longer withstand the applied internal and external forces, resulting in a main break. However, research has shown that the failure process is more complex than expected.

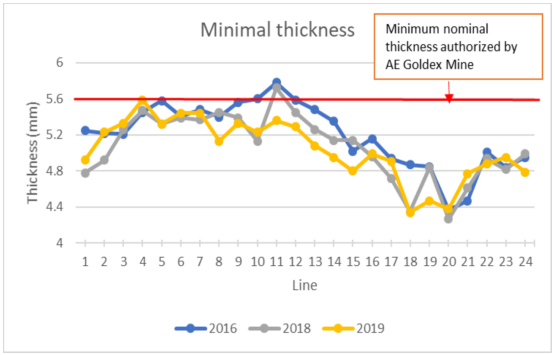

Figure 1: Minimal thickness values distribution

Figure 2 represents the whisker box of the pipeline thickness measurements in 2019. To study the distribution of thickness values on each line, the database is transposed to have rows (1 to 24) as columns and points 1 to 125 as rows (observations). Some lines will be chosen according to their minimum and average value to study their distribution. These are lines L4, L6, L15, L17, L19 and L20. Lines L4 and L6 have average thicknesses equal to 5.9 mm and minimum thicknesses of 5.4 mm. The average thickness of lines 15 and 17 are around 5.7 mm and the minimum thicknesses are 4.8 mm. While the average thickness of lines 19 and 20 are less than 5.6 mm and respectively equal to 5.5 mm and 5.4 mm, the minimum thicknesses are less than or equal to 4.4 mm. Lines 4 and 6 have 50% of their value between 5.8 and 5.9 mm. The loss of the thickness is almost non-existent. Line 4 has 75% of its values above 5.9 mm (value above the minimum allowed). Similarly, line 6 has the same proportion of values above 5.8 mm. While, line 20 has 50% of its values of thickness between 5.2 and 5.6 mm. Also, 75% of its values are less than or equal to the minimum allowed value indicating a high corrosion at this line. Line 19 is somewhat identical to line 20, with 50% of the values between 5.4 and 5.6 mm. The thickness at lines 15 and 16 remains normal with respectively 75% of the values between 5.6 and 5.9 mm.

After performing descriptive analyses of the pipeline data, machine learning models will be used to better understand the data and extract useful information. One of the unsupervised learning methods will be used along with other data mining methods. These are SOM and HAC. SOM is a neural method used to represent high-dimensional data into low-dimensional data. It is a powerful tool for data visualization and summarization. Like Principal Component Analysis (PCA), SOM allows for dimensionality reduction. It produces a mapping from the input space X to the reduced space Y (most common is a 2D network, creating Y a 2-dimensional space).

Pipelines fail due to factors that are operationally, structurally and environmentally induced. The operational factors are associated with the components of the fluid flowing through while the environmental factors deal with the electrochemical and mechanical interactions of the pipeline material and the immediate surroundings.

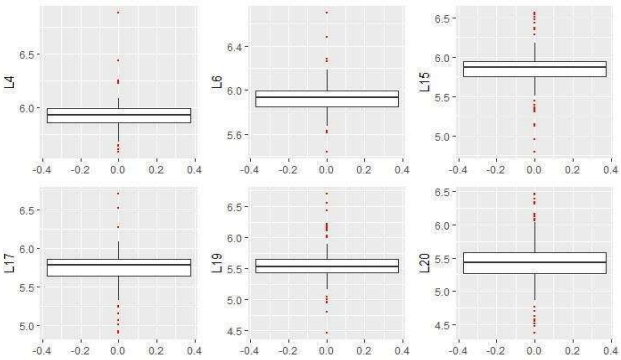

Figure 2: Box plot thickness in 2019

Figure 3 highlights the correlations that could be made between the annual average values of the pipeline operating conditions and the thickness average values of lines L2, L10, L20 and L16. A strong negative correlation was noted between parameters such as calculated residual TPH, pressure at Km0 and temperature at km 0 and thickness values at line 2. Similarly, the thickness at line 10 is correlated with Sag tonnage and temperature at km 0. Residue flow, solid residue percentage and calculated residue TPH are also negatively correlated with the thickness at line 16. However, the thickness at line 20 is positively correlated with the flotation pulp temperature. Although the observed correlations are indicative of the influence of operating conditions on the corrosion rate, the nature of the computed data (yearly averages) hinders the development of a corrosion predictive model.

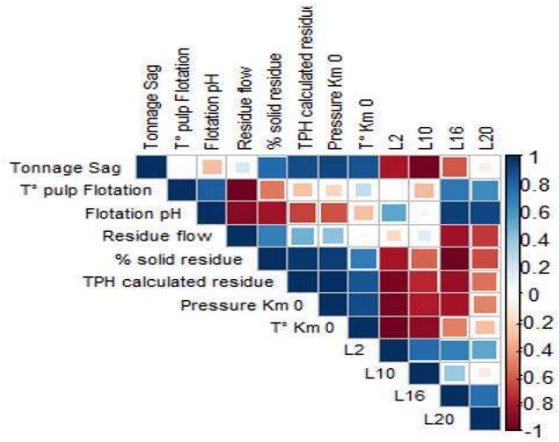

Figure 3: Correlation between process variables and pipe thicknesses

In this study, SOM will be used to reduce the dimensionality of the data and make an easier representation by taking into account the different dimensions. Thus, with the graphical representation, it will be possible to highlight the similarities in the data based on the similar thickness measurements. Then, the extracted code vectors will allow a classification with HAC.

SOM Algorithm

The SOM (Kohonen, 2013) is an unsupervised learning method based on the idea of competitive learning. It is mostly used as a tool for visualization by mapping a high-dimensional data onto a regular low-dimensional representation. The SOM algorithm is as follows (figure 5)

a) Initialize the weights $\boldsymbol{w_{ij}}$ randomly for each node with standardized values. Initialize the learning rate α SOM.

b) Calculate the squared Euclidean distance between the input vector $\boldsymbol{x_i}$ and the weight vector $\boldsymbol{w_{ij}}$ for $\boldsymbol{j^{th}}$ node on the SOM grid:

a. $$\boldsymbol{D(j) = \sum_{i=1}^{n}(x_i(t) - w_{ij}(t))^2}$$

where $\boldsymbol{n}$ is the amount of input vectors and $\boldsymbol{t}$ corresponds to iteration number.

c) Find a winning node (BMU) with following condition:

$$\boldsymbol{BMU = argmin_i\, D(j)}$$

d) Adjust the weights of BMU and neighbourhood nodes in the given radius for all input vectors by updating new weights as follows:

$$\boldsymbol{w_{ij}(t+1) = w_{ij}(t) + \alpha(t)(x_i(t) - w_{ij}(t))}$$

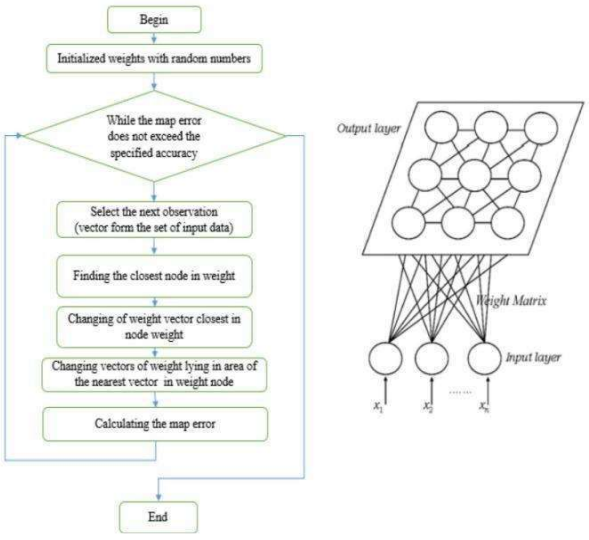

Figure 4. SOM Algorithm (Kumar & Saini, 2020)

Results

The result of the SOM model is a mesh of 5 × 5 hexagonal neurons trained with the Kohonen algorithm. The mesh provides a good representation of the sample space. There are no very dense areas or empty cells; at least each cell contains an element. The resulting trained map contains all the data in a vector structure so that the training data falls on each of the neurons (Figure 5).

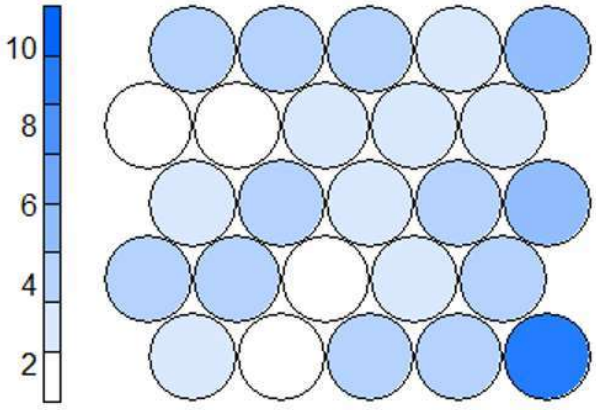

Figure 5: Node count

Figure 6 is the neighbour distance plot called Umatrix. The unified distance matrix (U-matrix) is a representation of SOM where the Euclidean distance between the codebook vectors of neighboring neurons is depicted in a range of colors. It shows the degree of similarity or difference between the samples through the distance between adjacent map units. At the same time, the distance between adjacent units can be indicated by the color gradient. Therefore, nodes that are close to each other are dark in color. It can be observed that they are concentrated at the right end of the map, the darker the color, the greater the loss of thickness. This suggests a good separation of groups in the topology.

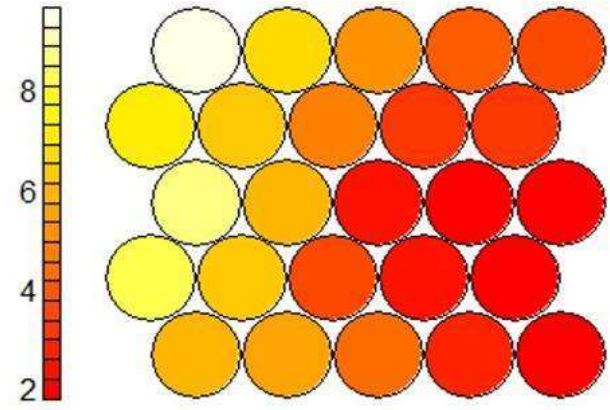

Figure 6: SOM Neighbour distance

Figure 7 represents the heatmap of the variable P3 chosen at random to analyze its distribution. A heatmap shows the distribution of a variable in the SOM. The high value areas are colored in red and the low value areas in blue. The southwestern zone is a high value zone. The low value areas (corrosion phenomenon) are located in the northeast. By doing the analysis combined with figure 5, it appears that the high value areas contain more observations than the low value areas.

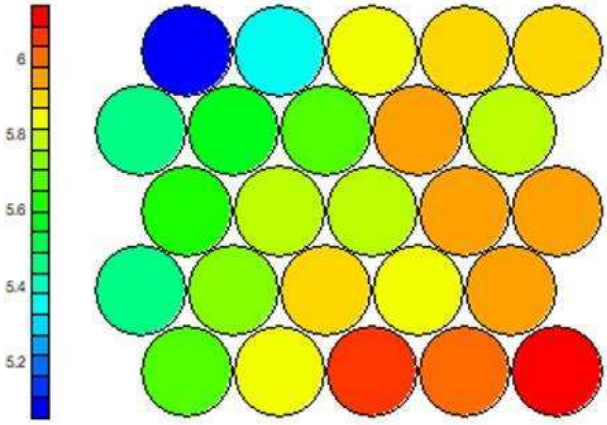

Figure 7: Heatmaps: Areas for high values (red) and low values (blue) for each variable

The third, fourth and fifth nodes each contain a sample with thickness values of about 6 mm and contain about 4 to 6 observations (third and fourth nodes) and more than 10 observations (fifth node). Therefore, the loss of the thickness is noticed on few lines.

Cluster analysis from the map

In order to classify the lines according to their loss of thickness, an HAC was performed after calculating the codebook vectors with the SOM. The classification will be done first on the nodes (25 in total). Each node contains observations (the lines delimited on the pipeline). The dendrogram (figure 8) suggests the repartition in 3 classes. In addition, other indices were calculated (kl, ch, Hartigan index, etc.) (Charrad and al 2014) and 3 clusters remains the best partition. Nodes V11, V16, V17, V21 and V22 are classified together in cluster 3. The first cluster contains many more nodes (13) compared to 7 for the second cluster. In figure 8, the nodes of the third cluster are located in the low value areas (blue color) which shows that this cluster contains the lines that were attacked by the corrosion phenomenon. Thus, the clusters can be categorized into high thickness loss (cluster 3), medium thickness loss (cluster 2) and very low thickness loss (cluster 1).

Figure 9 represents the different clusters with the number of observations in each node. Note the low value nodes are shown in the northeast and have a total of 16 observations (the thickness loss lines between 2016 and 2019).

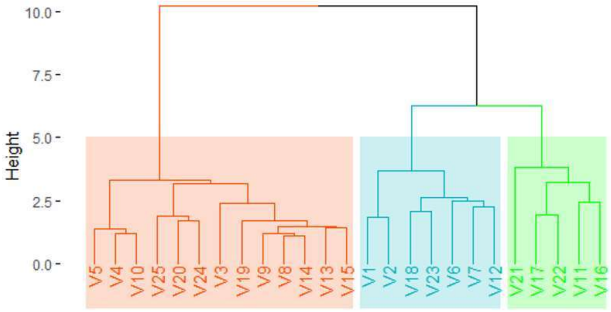

Figure 8: Cluster dendrogram

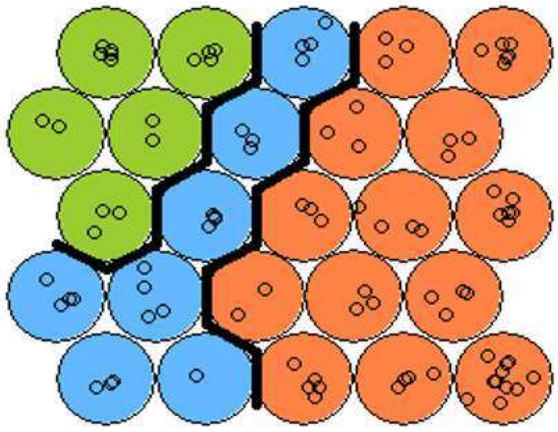

Figure 9: Representation of the clusters into the map

Table 1 represents the distribution of lines according to clusters for 2019. Lines 18 to L21 are the lines most affected by the thickness loss phenomenon.

Table 1 : Distribution of lines according to clusters for 2019

Cluster 1 (No thickness loss)	Cluster 2 (Medium thickness loss)	Cluster 3 (Important loss of thickness)
L1 ; L2 ; L13 ; L16 ; L17 ; L22 ; L23 ; L24	L3 ; L4 ; L5 ; L6 ; L7; L8 ; L9 ; L10 ; L11 ; L12 ; L14 ; L15	L18 ; L19 ; L20 ; L21

Conclusion and future work

In this paper, a new data representation is proposed to identify clusters representing corrosion levels in a pipeline based on ultrasound inspections. The neural method (SOM) is used to reduce the dimensionality and then represent the data in a smaller two-dimensional space. To identify the clusters, hierarchical ascending classification is applied on the nodes, resulting in three clusters representing the corrosion levels of the pipeline. This information is useful to pipeline corrosion experts who consistently plan corrosion mitigation activities through risk-based inspections. Future work will focus on the prediction of pipeline corrosion and failure rates by using in-line corrosion monitoring (ER and real-time erosion-corrosion probes) combine with models such as Random Forest, SVM, Multilayer Perceptrons or Convolutional Neural Networks will be used.

Acknowledgement

We thank Agnico Eagle Goldex Mine for support. This work was funded by the Natural Sciences and Engineering Research Council of Canada (NSERC) and the Québec Fonds de recherche nature et technologies (FRQNT).

References

Abbas, MH, Norman, R, Charles, A, 2018. Neural network modelling of highpressure CO2 corrosion in pipeline steels. *Process Saf. Environ. Protect*. 119: 36–45. https://doi.org/10.1016/j.psep.2018.07.006.

Ahmad, Z. 2006. Chapter 2 Basic Concepts in Corrosion. In Principles of Corrosion Engineering and Corrosion Control; Ahmad, Z., Ed.; Butterworth-Heinemann: Oxford, UK, 9–56, ISBN 978-0 -7506-5924-6. https://doi.org/10.1016/B978-0-7506-5924-6.X5000-4.

Charrad, M.; Ghazzali, N.; Boiteau, V.; and Niknafs, A. 2014. "NbClust : An R Package for determining the relevant number of clysters in a data set", *Journal of Statistical Software* 61(6): 1-36. https://doi.org/10.18637/jss.v061.i06.

Chico, B.; De la Fuente, D.; Díaz, I.; Simancas, J.; and Morcillo, M. 2017. Annual Atmospheric Corrosion of Carbon Steel Worldwide. An Integration of ISOCORRAG, ICP/UNECE and MICAT Databases. *Materials* 10(6). https://doi.org/10.3390/ma10060601.

Cristos, M. T.; Fernández, F. O.; Iglesias, G. A.; Piloneta, M. D.; Iglesias, A. F. 2021. Corrosion Prediction of Weathered Galvanised Structures Using Machine Learning Techniques. *Materials* 14(14). https://doi.org/10.3390/ma14143906.

Hassan, F.; Mahmood, A. K.; Rahim, L. A.; Jameel, S. M.; Saboor, A.; and Rimsan, M. 2021. Clustering-Based Quantitative Evaluation Using Acoustic Emission Waveforms for Corrosion Detection. *Journal of Hunan University (Natural Sciences)* 48(7): 320-329.

Hatami S, S.; Ghaderi-Ardakani A, A., and Niknejad-Khomami M. M. 2016. On the prediction of CO2 corrosion in petroleum industry. *Journal of Supercritical Fluids* 117: 108–112. http://dx.doi.org/10.1016/j.supflu.2016.05.047.

Koch, G.; Varney, J.; Thompson, N.; Moghissi, O.; Gould, M.; Payer, J. 2006. International measures of prevention, application, and economics of corrosion technologies study. *Nace Impact Rep*.

Kohonen, T. 2013. Essentials of the selforganizing map. *Neural Networks* 37(1): 52–65. https://doi.org/10.1016/j.neunet.2012.09.018.

Kowalski, A. R. 2012. A Successful Methodology Following API 570 & ANSI/NACE SP0502 for Piping Integrity Assessment. *Nace Corrosion*.

Kumar, K.; and Saini, R.P. 2020. Application of machine learning for hydropower plant silt data analysis. *Materials Today: Proceedings* 46(7). http://dx.doi.org/10.1016/j.matpr.2020.09.375.

Lou, J.; Elboujdaini, M.; Shoesmith, D.; and Patnaik, P C. 2003. Environmental degradation of materials and corrosion control in metals: *proceedings of the International Symposium*: Vancouver, British Columbia, Canada, Canadian Institute of Mining, Metallurgy and Petroleum.

Luchun, Y.; Yupeng, D.; Zhaoyang, L., and Kewei, Gao. 2020. Corrosion rate prediction and influencing factors evaluation of low-alloy steels in marine atmosphere using machine learning approach. *Science and Technology of Advanced Materials* 2(1): 359-370. http://dx.doi.org/10.1080/14686996.2020.1746196.

Mohamed, A. A.; Salah, H. M.; and Tahar, S. 2015. Self-Organizing Map-Based Feature Visualization and Selection for Defect Depth. *19th International Conference on Information Visualisation* 235-240. http://dx.doi.org/10.1109/iV.2015.50.

Nikoo, M.; Sadowski, Ł.; Nikoo, M. 2017. Prediction of the Corrosion Current Density in Reinforced Concrete Using a Self-Organizing Feature Map. *Coatings* 7(10). http://dx.doi.org/10.3390/coatings7100160.

Okonkwo, P. C.; and Adel, M. A. M. 2014. Erosion corrosion in oil and gas industry: a review. *International Journal of Metallurgical & Materials Science and Engineering (IJMMSE),* 4(3): 7-28.

Ossai, C. I. 2013. Pipeline Corrosion Prediction and Reliability Analysis: A Systematic Approach with Monte Carlo Simulation and Degradation Models. *International Journal of Scientific & Technology Research (IJSTR)* 2 (3): 58-69.

Peng, S.; Zhang Z.; Liu, E.; Liu, W.; and Qiao, W. 2021. A new hybrid algorithm model for prediction of internal corrosion rate of multiphase pipeline. *Journal of Natural Gas Science and Engineering* 85. https://doi.org/10.1016/j.jngse.2020.103716

Raheem, L. 2020. Erosion-corrosion study of carbon steel and duplex stainless steel elbows in potash brine-sand slurry, Department of Mechanical Engineering, University of Saskatchewan, Saskatoon. Canada.

Roy, A.; Taufique, M.F.N.; Khakurel, H.; Devanathan, R.; Johnson, D, D.; and Balasubramanian, G. 2022. Machine-learning-guided descriptor selection for predicting corrosion resistance in multi-principal element alloys. *npj Mater Degrad* 6(1). http://dx.doi.org/10.1038/s41529-021-00208-y.

Taffese, W. Z.; and Sistonen, E. 2016. Neural network based hygrothermal prediction for deterioration risk analysis of surface-protected concrete façade element. *Constr. Build. Mater,* 113: 34–48. http://dx.doi.org/10.1016/j.conbuildmat.2016.03.029.

Yin, C.; Cheng, X.; Liu, X.; and Zhao, M. 2020. Identification and Classification of Atmospheric Particles Based on SEM Images Using Convolutional Neural Network with Attention Mechanism. *Complexity* 2020(20). http://dx.doi.org/10.1155/2020/9673724 .

Annexe B : Article scientifique 2

Walk-Through Corrosion Assessment of Slurry Pipeline Using Machine Learning

Abdou Khadir DIA [1,*], Axel Gambou-Bosca [2] and Nadia Ghazzali [1]

[1] Université du Québec à Trois-Rivières, Department of Mathematics and Computer Science; Nadia.Ghazzali@uqtr.ca
[2] Québec Metallurgy Center; axel.gambou.bosca@cegeptr.qc.ca
* Correspondence: abdou.khadir.dia@uqtr.ca

Abstract: The study of pipeline corrosion is crucial to prevent economic losses, environmental degradation and worker safety. In this study, several machine learning methods such as Recursive Feature Elimination (RFE), Principal Component Analysis (PCA), Gradient Boosting Method (GBM), Support Vector Machine (SVM), Random Forest (RF), K Nearest Neighbors (KNN) and Multilayer Perceptron (MLP) were used to estimate the thickness loss of a slurry pipeline subjected to erosion corrosion. These different machine learning models were applied to the raw data (the set of variables), to the variables selected by RFE and to the variables selected by PCA (principal components) and a comparative analysis was carried out to find out the influence of the selection and transformation of the data on the performance of the models. The results show that the models perform better on the variables selected by RFE and that the best models are: RF, SVM, GBM with an average RMSE of 0.017. By modifying the hyperparameters, the SVM model becomes the best model with an RMSE of 0.011 and an R-squared of 0.83.

Keywords: Corrosion, Pipeline, Mining, machine learning, recursive feature elimination, principal component analysis.

1. Introduction

Slurry pumping technology is a well-established and favored method for transporting mineral concentrates through pipelines. These pipelines can be made from a variety of materials, including carbon steel, alloy steel, hardened steel, stainless steel, abrasion-resistant lined pipes, non-ferrous pipes, and HDPE, with the choice depending on the application, material being transported, and cost [1]. Despite the excellent safety record and favorable economics of long-distance slurry pipeline systems compared to traditional bulk transport systems, pipe abrasion and erosion loss remain a significant concern. While non-ferrous pipelines can extend the life of the transport system for mineral concentrates, carbon steel pipes are prone to internal corrosion, especially when dealing with abrasive or corrosive slurries. The pipeline structure and materials are continually being improved for various industries. For example, HDPE is extensively used for applications such as mine tailings due to its ultra-high molecular weight and resistance to abrasiveness, making it more durable than carbon steel pipes. In addition, non-ferrous materials are used to line the inside of steel pipes to protect against erosion and corrosion, and low wear resistance non-ferrous pipes such as polyurethane, polybutylene, PVC, PP, ABS, and fiberglass pipe with internal ceramic chips are also available for slurry transport.

Despite the use of carbon steel pipes, the high wear conditions caused by the large quantities and abrasive nature of slurry can result in leaks or ruptures, leading to significant maintenance costs in the mining industry due to erosion corrosion, especially in long pipelines spanning hundreds of kilometers. Other industries, such as the oil and gas sector, have also reported erosion corrosion as one of the top five forms of damage mechanisms, posing challenges to machinery and equipment with short lifecycles [2, 3]. Therefore, to mitigate these risks, it is crucial to implement pipeline integrity detection and monitoring, including an understanding of defect progression, condition-based maintenance, and lifecycle management [4].

Over the years, several non-destructive testing methods have emerged for inspecting pipelines while in use, including ultrasonic inspection (UT), which uses high-frequency sound waves to identify defects on materials or their surfaces. UT is effective at detecting cracks, crevices, metal losses, and other discontinuities at varying depths within samples due to the reflection, diffraction, and transmission characteristics of ultrasonic sound [4, 5]. A bulk wave ultrasonic thickness measurement technique for corrosion monitoring can be used by temporarily or permanently coupling a transducer to the outer surface of a pipe, and the wall thickness of the pipe can be determined based on the time difference between transducer excitation and reception of the reflected wave from the back-wall surface. Traditional inspection and maintenance practices based solely on experience are no longer sufficient, and pipeline operators now require quantitatively risk-based methodologies. To reduce the economic impact of failures and minimize their impact on the environment, health, and safety, analytical tools have been developed over the years [6].

After the emergence of big data techniques, machine learning (ML) has demonstrated significant benefits in modeling and data mining [7]. ML has been utilized in various corrosion-related issues, such as the modeling of CO2 corrosion [8], automated image analysis to detect corrosion [9], the modeling of corrosion defect growth in pipelines [10], material inspection [11], predicting corrosion rates in marine environments [12], determining the initiation time of embedded steel corrosion in reinforced concrete [13], predicting electrochemical impedance spectra [14], characterizing the spatial distribution of pitting corrosion [15], and modeling pipeline aging [16]. A variety of machine learning techniques have been employed to predict the rate of corrosion or identify the areas most affected by corrosion [17]. According to Jinrui et al., they trained six machine learning models on ultrasonic testing data to forecast the degree of corrosion based on ultrasonic characteristics [18]. The findings suggest that, except for the linear model, machine learning models can accurately and robustly forecast the corrosion degree despite the interference of outlier amplitude and training set size. In their study, Julio et al. examine various statistical and probabilistic methods that have been utilized in the literature to investigate corrosion issues and their practical applications [19]. Meanwhile, Wei et al. utilize an artificial neural network to establish a relationship model between the corrosion potential of low alloy steel in Sanya seawater and its influencing factors, allowing them to visualize the impact of different alloy elements on corrosion potential [20]. To estimate the corrosion defect depth growth of aged pipelines, Chinedu adopts a data-driven machine learning approach, relying on techniques such as Principal Component Analysis (PCA), Particle Swarm Optimization (PSO), Feed-Forward Artificial Neural Network (FFANN), Gradient Boosting Machine (GBM), Random Forest (RF), and Deep Neural Network (DNN), to estimate the growth of corrosion defect depth in aged pipelines [10]. Roy et al. use the Gradient Boosting Regressor to predict corrosion resistance in alloys with multiple principal elements [21], while Zhifeng et al. suggest using rough set and decision tree methods to analyze pipeline soil corrosion [22]. To model experimental data of time-varying corrosion rates in mild steel specimens when corrosion inhibitors are added to the system at varying concentrations and dose schedules, Mohammadreza et al. perform regression with several ML algorithms, ultimately finding Random Forest to be the best option [23]. Lastly, Peng et al. propose a new hybrid intelligent algorithm that combines SVR, PCA, and CPSO to predict the corrosion rate of multiphase flow pipelines, utilizing PCA to reduce data dimensionality and CPSO to optimize hyperfine parameters in SVR [24].

The literature review above highlights the increasing use of machine learning methods in the field of corrosion, which is attributed to the emergence of software solutions that reduce the need for extensive mathematical and statistical knowledge. However, the risk of obtaining false positive results in a "black box" automated process cannot be ignored. The purpose of this paper is not to present a complete machine learning model for predicting corrosion erosion degradation in a slurry pipeline but rather to describe the methodology in detail to provide a corrosion assessment using machine learning as a starting point for the corrosion community. Full research papers often struggle to explain the essential aspects of machine learning tools suitable for a specific dataset, as much emphasis is placed on results and discussion. Therefore, this paper aims to explain every step and parameter needed to draw robust and trustworthy predictive models, including the effect of feature engineering methods like recursive feature elimination and principal component analysis, as well as data transformation. The data used in this publication was obtained from the periodic non-destructive evaluation of a pipeline, and five machine learning models were applied and compared, including RF, KNN, SVM, MLP, and GBM, on both unprocessed and feature-engineered data. Hyperparameter optimization using the grid search method improved the model's results and made it more robust.

2. Materials and Methods

2.1 Ultrasonic monitoring

The issue of erosion corrosion in pipelines is significant in slurry pumping systems. To prevent risks and monitor structural health, corrosion models are constructed by quantitatively estimating the degradation. Conventionally, this involves placing coupons made of the same pipe material inside the pipe and measuring the resulting weight loss after exposure to the environment for a specified period [24]. However, this method is intrusive, costly, and time-consuming due to manual intervention. Alternatively, ultrasonic technology offers a non-intrusive way to monitor corrosion. An array of eight ultrasonic transducers with a diameter of 10 mm and a frequency of 5 MHz, smartPIMS distributed by Procon systems Canada, was used to monitor a section of a 12'' diameter, 24'' length pipeline in pulse-echo mode, as shown in Figure 1, for long-term monitoring since 2021.

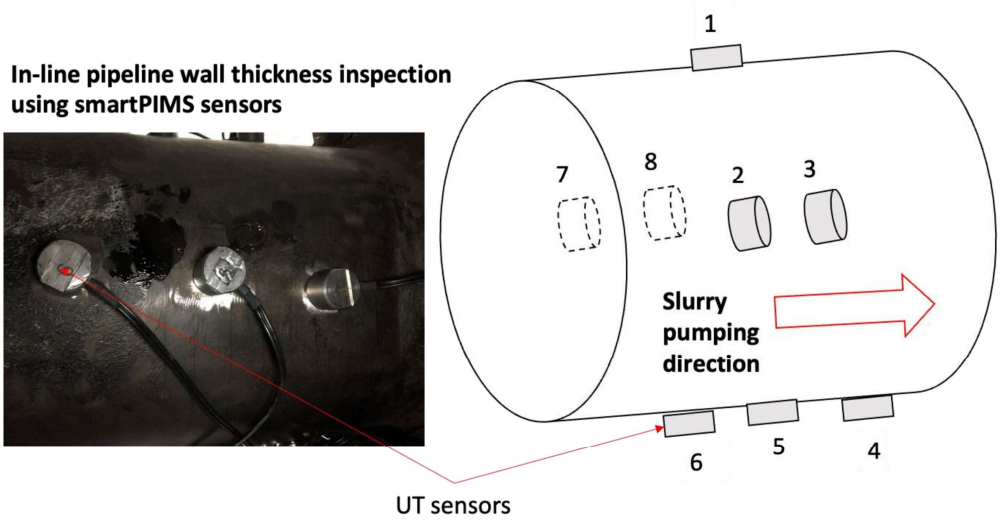

Figure 1: Ultrasonic transducers fixed at the pipe for continuous wall thickness monitoring.

2.2 Data Collection

Data for this study were obtained from Agnico Eagle Mine Goldex. These data are physical measurements from a pipeline that is used to transport residue (pulp/slurry) from the concentrator to the Manitou Residue Park site owned by the MERN (Ministère de l'Énergie et des Ressources Naturelles) in Val-d'Or, Quebec. Data include pipeline wall thickness values and process parameters such as pulp temperature, pH, pressure, etc. from the various in-line probes. While thickness measurements were taken on a daily basis, process variables were collected every five minutes. Table 1 lists the different variables and their descriptive statistics.

Table 1: List of features.

	Minimum	Maximum	Mean	Sd
Tonnage sag (t)	8.78	404.39	342.65	65.15
Flotation pulp temperature (°C)	13.92	34.92	23.07	4.68
Flotation pH	7.47	9.33	9.03	0.27
Residue flow (m^3/h)	18.54	495.97	437.59	90.22
% solid residue	0.02	46.13	23.54	10.17
TPH calculated residue	0.02	301.95	141.31	68.20
Pressure at Km 0 (Psi)	624.61	3439.54	2209.96	618.89
T° at Km 0 (°C)	3.76	30.86	16.25	5.70
Flow rate (m^3/h) (Thompson River)	30.00	381.19	207.72	110.66
T° (Thompson River) (°C)	0.99	20.45	13.42	6.04
Flow rate (m^3/h) (sedimentation basin)	26.88	122.30	68.82	16.17
T° (sedimentation basin) (°C)	2.71	21.55	9.58	5.71
Flow rate (m^3/h) (South Park)	0.04	369.80	168.89	95.23
T° (South Park) (°C)	0.73	16.23	5.21	4.53
Pipe wall thickness (mm)	5.75	6.01	5.84	0.03

2.3 Feature Selection

In this study, the process of feature selection involved choosing important features that contribute significantly to thickness loss. To achieve this, the authors employed the Recursive Feature Elimination (RFE) technique [25] which involves building a model on the entire set of predictors and assigning an importance score to each predictor. Less important predictors are then eliminated, and the model is rebuilt with the remaining predictors, and importance scores are computed again [26]. RFE is particularly useful for certain models like the random forest, which the researchers

used in their study of pipeline degradation data [27]. Along with RFE, the authors also used Principal Component Analysis (PCA) as a dimensionality reduction method to extract important information from the data and represent it as a set of new orthogonal variables called principal components [28]. Although PCA is not a variable selection method, it can be used to enhance model performance. The authors identified principal components that explained a significant proportion of the variance in the data set and used them as explanatory variables to compare models with the initial data.

2.4 Machine Learning Models

2.4.1 K-Nearest Neighbors (KNN)

The k-nearest neighbor method is a distance-based supervised learning method [29]. It is a method easily scalable and has few hyperparameters. It is used for classification and regression problems. The classification or prediction of a new value is based on the values of the nearest neighbors that are determined using a distance between those values. The k value represents the number of neighbors, if it is equal to 1, for a classification problem, the predicted class is the class of the nearest neighbor and for a regression problem, the predicted value is the value of the nearest neighbor. If k is greater than 1, the predicted value is the average of the values of k neighbors for a regression problem. One of the most important steps of the KNN algorithm is the determination of the neighbors which is done through a distance calculation. One of the most used distances and which is used in this paper is the Euclidean distance. Given two samples $X = \{x_1, x_2, \dots\dots, x_n\}$ *and* $Y = \{y_1, y_2, \dots\dots\}$, the Euclidean distance is calculated as follows:

$$D(X,Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \tag{1}$$

Determining the value of k is very important since a poor choice of k can lead to overfitting or underfitting. High deviations with low bias are often characterized by lower k values.

2.4.2 Random Forest (RF)

Random forest is an ensemble learning method that consists of multiple decision trees. It was proposed by Tin Kam Ho in 1995 [30] and an extension was proposed by Leo Breiman and Adele Cutler in 2001 [31]. It is an algorithm that can be used for both classification and regression problems and has been rapidly adopted because of its flexibility. For a classification problem, the predicted class is the class predicted by most trees. For a regression problem, the predicted value, represented in the following equation, is the average of the values predicted by the different trees.

$$\bar{h}(x) = \frac{1}{T}\sum_{t=1}^{T}\{h(x,\theta_t)\} \tag{2}$$

The learning process of the random forest model starts with bagging by sampling a training data set with replacement, also called boostrap sampling, and k predictors for each tree. Then a decision tree is trained on each sample. Finally, the prediction of the random forest model is the average of the values predicted by each tree.

2.4.3 Gradient Boosting Machine (GBM)

Gradient Boosting is a popular machine learning technique applied to classification tasks, known for its robustness and high performance compared to decision trees and random forest algorithms in certain cases. The method improves the accuracy of predictions by iteratively combining multiple "weak learners," which are simple models, to produce a "strong learner" with superior performance. On the other hand, the Gradient Boosting Machine, developed by Friedman, was inspired by Gradient Boosting and is utilized for regression problems [32].

Let be a training sample $(x_i, y_i)\ i = 1, \dots\dots n$. We make the assumption that we have a set of base learners $\mathcal{B}$ and our objective function can be expressed as a linear combination of these base learners, which is denoted $Lin(\mathcal{B})$. The set of learners, $\mathcal{B}$, is defined as $B = \{b_\tau(x) \in \mathbb{R}\}$ where $\tau \in \mathrm{T}$ represents the parameters of the learners. To predict the output for a given feature vector, x, we use an additive model represented by [33]:

$$f(x) := \left(\sum_{m=1}^{M} \beta_m b_{\tau_m}(x)\right) \in \mathrm{lin}(\mathcal{B}) \tag{3}$$

Where $b_{\tau_m}(x) \in \mathcal{B}$ is a weak-learner and β_m is its corresponding additive coefficient.

The objective of GBM is to derive an accurate approximation of the function f that can effectively reduce the empirical loss [33]:

$$L^{\star} = \min_{f \in \mathrm{lin}(\mathcal{B})} \left\{ L(f) \coloneqq \sum_{i=1}^{n} \ell(y_i, f(x_i)) \right\} \tag{4}$$

Where $\ell(y_i, f(x_i))$ is a measure of the data fidelity for the *i*-th sample for the loss function ℓ.

The objective of the GBM method, as a numerical optimization algorithm, is to minimize the loss function by finding an additive model. The algorithm (as shown in Fig. 2) initializes the model with a first estimation, typically a decision tree, and aims to minimize the loss function. With each iteration, the algorithm calculates a model that best fits the residuals and adds it to the previous model to update the residuals. The algorithm stops after reaching the maximum number of iterations specified by the user.

Initialization. Initialize with $f^0(x) = 0$.
For $m = 0, \ldots, M-1$ do:
Perform Updates:
(1) Compute pseudo residual: $r^m = -\left[\frac{\partial \ell(y_i, f^m(x_i))}{\partial f^m(x_i)}\right]_{i=1,\ldots,n}$.
(2) Find the parameters of the best weak-learner: $\tau_m = \arg\min_{\tau \in \mathcal{T}} \sum_{i=1}^{n} (r_i^m - b_\tau(x_i))^2$.
(3) Choose the step-size η_m by line-search: $\eta_m = \arg\min_{\eta} \sum_{i=1}^{n} \ell(y_i, f^m(x_i) + \eta b_{\tau_m}(x_i))$.
(4) Update the model $f^{m+1}(x) = f^m(x) + \eta_m b_{\tau_m}(x)$.
Output. $f^M(x)$.

Figure 2: Algorithm Gradient Boosting Machine (GBM) [33].

2.4.4 Support Vector Machine

Support Vector Machine (SVM) is a supervised machine learning model used for regression and classification problems. It was first developed by Vapnik and his colleagues [35, 36]. For regression problems, the name changes to Support Vector Regression (SVR). The principle is almost the same as for classification problems except that for regression problems the continuous variable must be predicted. The goal of the SVR algorithm is to find a hyperplane in an n-dimensional space that best fits the data. The hyperplane is the line that will help us predict the continuous value or the target value. The continuous function to be approximated can be written as in equation 5:

$$y = w.x + b \tag{5}$$

- SVR Linear

SVR formulates this function approximation problem as an optimization problem as presented in equations 6 and 7:

$$min \frac{1}{2} \|w^2\| + c \sum_{i=1}^{n} \xi_i + \xi_i^* \tag{6}$$

Subject to

$$\begin{cases} z_i - (w \cdot x + b) & \leq \varepsilon + \xi_i \\ (w \cdot x + b) - z_i & \leq \varepsilon + \xi_i^* \\ \xi_i \xi_i^* & \geq 0 \end{cases} \tag{7}$$

$$y = \sum_{i=1}^{N} (\alpha_i - \alpha_i^*) \cdot \langle x_i, x \rangle + b \tag{8}$$

Where $\|w^2\|$ indicates the size of the normal vector corresponding to the surface being approximated, the variables $\xi_i \, et \, \xi_i^*$ are responsible for determining the allowable number of points outside the tube, while C acts as a regularization parameter that can be adjusted to give greater importance to minimizing either the error or the flatness of the solution in this problem involving multiple objectives. This information is cited from reference [37].

- Non-linear SVR

In the nonlinear case (Fig. 3), kernel functions are used to transform the data to allow for linear separation as in equations 9 and 10.

$$min\frac{1}{2}\|w^2\| + c\sum_{i=1}^{n}\xi_i + \xi_i^* \tag{9}$$

Subject to

$$\begin{gathered} y_i - w^T\varphi(x_i) \leq \varepsilon + \xi_i^* i = 1, \ldots, N \\ w^T\varphi(x_i) - y_i \leq \varepsilon + \xi_i i = 1, \ldots, N \\ \xi_i, \xi_i^* \geq 0 i = 1, \ldots, N \\ w = \sum_{i=1}^{N_{SV}} (\alpha_i^* - \alpha_i)\varphi(x_i) \end{gathered} \tag{10}$$

$$\begin{gathered} y = \sum_{i=1}^{N} (\alpha_i - \alpha_i^*) \cdot \langle \varphi(x_i), \varphi(x) \rangle + b \\ y = \sum_{i=1}^{N} (\alpha_i - \alpha_i^*) \cdot K(x_i, x) + b \end{gathered} \tag{11}$$

Where $K(x_i, x)$ is the kernel function.

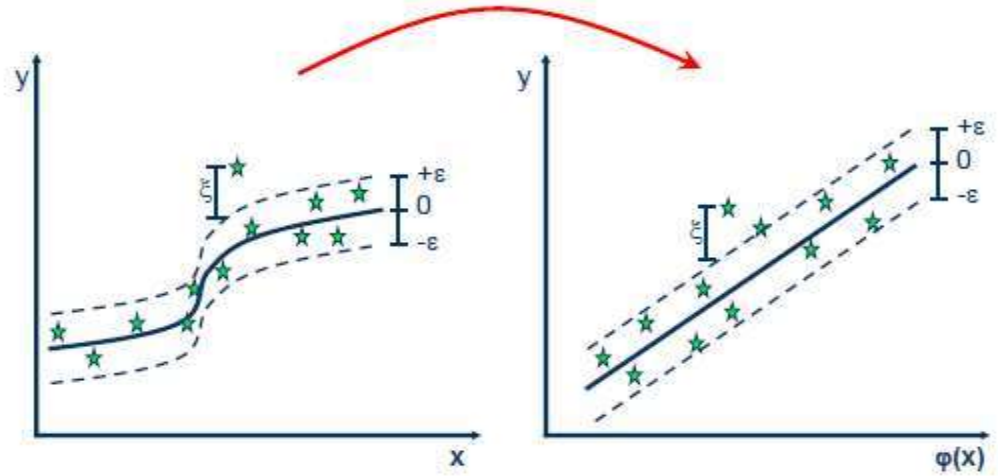

Figure 3: Non-Linear SVR.

2.4.5 Multilayer Perceptron

The MLP, a form of artificial neural network, is structured with multiple layers and operates through direct propagation from the input to output layer. The number of neurons in each layer varies, with the last layer being designated as the output layer. In the multilayer back-propagation perceptron, adjacent layers are interconnected, with the strength of these connections being determined by a coefficient that influences the destination neuron's response. The backpropagation algorithm is used to calculate these coefficients, which are essential to the network's functionality. Figure 4 shows a multilayer network. Each neuron i receives a series of signals from neurons j located at the previous layers. The operation of the illustrated network is governed by the following equation [38].

$$Z_i = \sum_{j=1}^{n_j} W_{ij} X_j + b_i \tag{12}$$

Where n_j is the number of input neurons and X_j is the value of the signal transmitted by neuron j of the previous layer. The W_{ij} represent the respective weights of the connections between the neurons j of the previous layers and the neuron i of the current layer. The parameters b_i, are bias values allowing a non-zero transfer function at the origin. The inputs X_j are weighted by the weights W_{ij}. Once the input is provided, neuron i transforms it and produces an output. In this case, Z_i and the output O_i, of a given neuron, are related by a transfer function of hyperbolic tangent form.

$$O_i = f(Z_i) = \frac{1 - e^{-2Z_i}}{1 + e^{-2Z_i}} \tag{13}$$

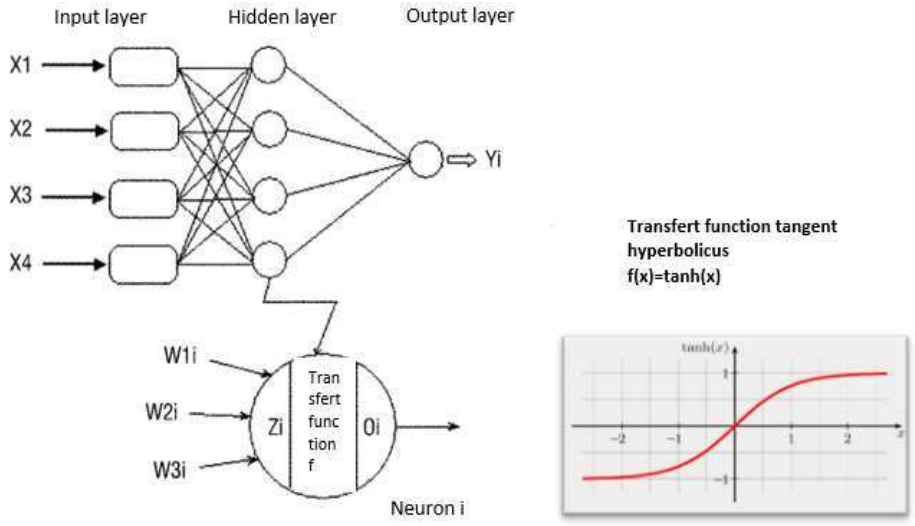

Figure 4: Multilayer Perceptron [38].

The error made by the network at the output is calculated and then minimized. This is referred to as the error backpropagation method. The weights of the network W_{ij} are corrected to reduce the overall error $\underline{E}$. The gradient descent method is used to minimize the global error. It is represented by the equation 14:

$$\underline{E} = \frac{1}{2} \sum_{k=1}^{n_k} (S_k - O_k)^2 \tag{14}$$

Where S_k represents the estimated value and O_k the observed value and $\underline{E}$ the overall error.
The steps of the error backpropagation algorithm are:

1 Presentation of a training pattern to the network.
2 Comparison of the network output with the target output.
3 Computes the output error of each neuron in the network.
4 Compute, for each neuron, the output value that would have been correct.
5 Definition of the increase or decrease necessary to obtain this value (local error).
6 Adjustment of the weight of each connection towards the lowest local error.
7 Assigning a blame to all previous neurons.
8 Repeat from step 4, on the previous neurons using the blame as error.

3. Results and Discussion

3.1. Models with all features

The prediction of thickness loss has been studied several times in the literature using different types of features and models. The features most often found in the literature are the chemical characteristics of the pipeline such as CO2 partial pressure, corrosion inhibitor type [23], sulfate ion concentration and chloride ion concentration [10]. In this study,

other types of variables directly related to the pipeline (pH, residue flow, pressure, calculated residual TPH) and variables external to the pipeline such as flow rate and temperature of the rivers and sedimentation basin that feed the pipeline were collected. Different set of models with all the variables were performed on the training data. Both the coefficient of determination (R^2) and Root-Mean-Square Error (RMSE) were employed for the model's predictive performance evaluation. They were defined by the following equations [7]:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \widehat{y_i})^2}{\sum_{i=1}^{n}(y_i - \overline{y_i})^2} \quad (15)$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y_i})^2} \quad (16)$$

where y_i , $\widehat{y_i}$ *and* $\overline{y_i}$ represent the measured value, the predicted value and the average value of the corrosion rate, respectively. During the training process, a cross-validation technique (i.e., 5-fold repeated cross-validation method) was utilized to avoid random errors as much as possible [38]. The models were trained on 80% of the data and the rest (20%) was used for validation.

The results presented in Table 2 show slight differences in RMSE for the RF (0.017), GBM (0.018) and SVM (0.018) models. The KNN and MLP models perform less well with an RMSE of 0.02. The R^2 of the MLP model is low compared to the other models, i.e. the explanatory variables explain less the variation of the thickness loss. Table 3 shows pairwise statistical significance scores. The table's lower diagonal displays p-values for the null hypothesis, indicating that the distributions are the same. Conversely, the upper diagonal shows the estimated difference between the distributions. It is evident from the table that there is no discernible difference between RF and GBM, and the differences between the distributions for RF, SVM, and KNN are minimal. The above results are those obtained with the training data. We will apply the validation data at the end when we have found the best model.

Table 2: Result of the models with all variables.

	Training set	
Models	**RMSE**	**R^2**
SVM	0.018	0.667
GBM	0.018	0.672
RF	0.017	0.715
KNN	0.021	0.504
MLP	0.026	0.182

Table 3: Pairwise statistical significance scores.

	SVM	**KNN**	**RF**	**GBM**
SVM		-0.003	0.001	0.001
KNN	0.02		0.004	0.004
RF	0.02	0.00		0.00
GBM	1.00	0.01	1.00	

3.2. Models with feature selection

To enhance the model's performance, feature engineering is a crucial stage in modeling that involves selecting the most significant variables using various methods. According to [7] research, the Gradient Boosting Decision Tree (GBDT) method and Kendall correlation analysis were employed as feature engineering approaches.

During the second stage of thickness loss data modeling, we employed the recursive feature elimination technique to enhance the model's effectiveness. This approach involves fitting a model and removing the weakest feature or features until the designated number of features is achieved. The model implemented in this process is RF, which has a reliable built-in feature importance calculation mechanism. The purpose of this method is to eliminate any dependencies and collinearity that could potentially exist in the model. Figure 5 illustrates the change in RMSE concerning the number of selected variables in our model. The optimal number of variables is 10: Tonnage sag, Pulp temperature

Flotation, % solid residue, TPH calculated residue, Pressure at Km 0, T° at Km 0, T° (Thompson River), Flow rate (sedimentation basin), T° (sedimentation basin), Flow rate (South Park), T° (South Park) with an RMSE of 0.017. These 10 variables were used to study the other models. The results (Table 4) show little variation in the performance metrics. The results of the significance test for the RF, SVM and GBM models show that there is no difference between these models. However, these models are better than KNN and MLP. MLP work better in situations where the sample size is very large which is not the case for our study.

Table 4: Result models with feature selection.

	Training set	
Models	**RMSE**	**R^2**
SVM	0.016	0.735
GBM	0.017	0.707
RF	0.017	0.725
KNN	0.027	0.274
MLP	0.027	0.074

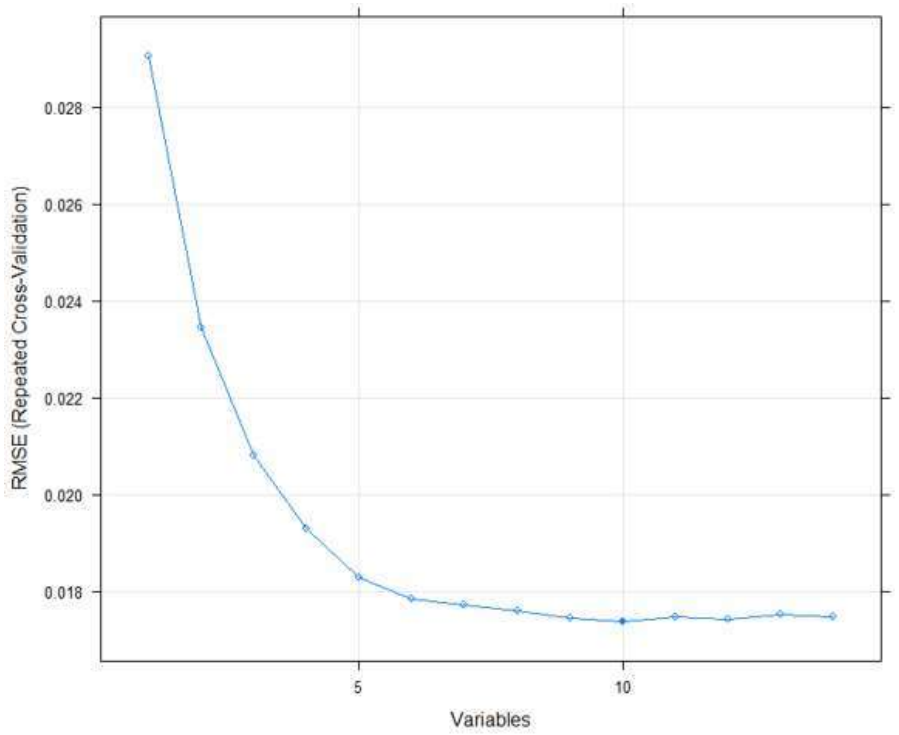

Figure 5: Recursive Feature Elimination.

3.2. Models wirh feature selection by PCA

Principal Component Analysis (PCA) is sometimes used as a method of feature engineering using as explanatory variables the principal components that explain the greatest variation. It is a method that sometimes gives excellent results [10]. We performed a PCA on our training data and selected the 8 Principal Components (PC) (figs. 6 and 7) that account for more than 95% of the variation in the data. These 8 principal components are then used as explanatory variables in our different models. The results (Table 5) vary slightly from those found previously. However, the KNN model performs better with the PCA transformation of the data.

Tableau 5: Result models with feature selection by PCA.

	Training set	
Models	**RMSE**	**R^2**
SVM	0.018	0.66
GBM	0.02	0.51
RF	0.019	0.61
KNN	0.019	0.55

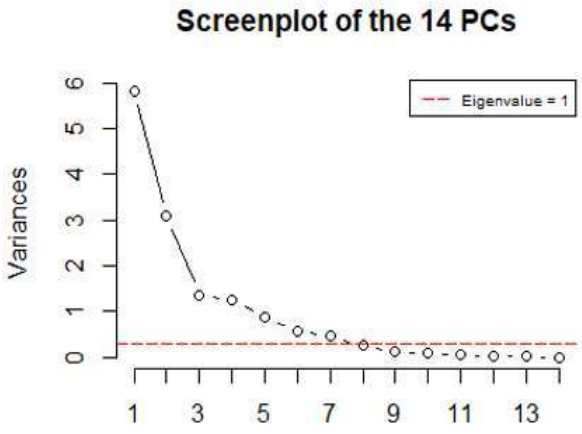

Figure 6: Screenplot of 14 PCs.

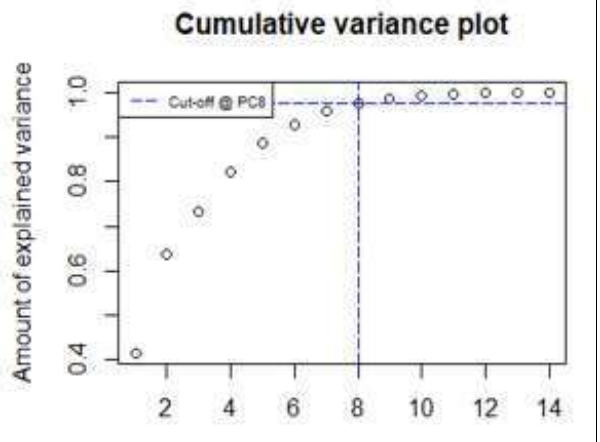

Figure 7: Cumulative variance plot.

3.2. Tuning hyperparameters

hyperparameter optimization [39,40] or tuning involves selecting an optimal set of hyperparameters for a learning algorithm that maximizes the model's performance and minimizes a pre-defined loss function to produce accurate results with fewer errors. Grid search, also known as parameter sweep, has traditionally been the preferred method for hyperparameter optimization, which involves an exhaustive search through a manually specified subset of the algorithm's hyperparameter space. Grid search is guided by a performance metric, which is typically measured by cross-validation on the training set [40,41]. For the SVM model, we aim to identify the optimal values for C and Gamma. C represents the cost of constraint violation and is the regularization term constant in the Lagrange formulation. A low value may cause the model to incorrectly classify some training data, while a high value may lead to overfitting, which results in an analysis that is too specific for the current data set and may not be suitable for future data. Gamma is the inverse of the influence radius of data samples chosen as support vectors. High values indicate a small radius of influence and small decision boundaries that do not consider relatively close data samples, leading to overfitting. Low values indicate a significant impact of distant data samples, causing the model to fail to capture the correct decision boundaries from the data set. For the RF model, we aim to find the optimal values of max_features and n_estimators. Max_features represents the maximum number of features that Random Forest can attempt in an individual tree, while n_estimators refers to the number of trees built before taking the maximum voting or averages of predictions. The best hyperparameters for the RF model are max_features=2 and n_estimators=2500, resulting in an RMSE of 0.016 and an R^2 of 0.73. Figure 8 illustrates the variation of RMSE as a function of max_features with n_estimators=2500.

Hyperparameter optimization of the SVM model produces more interesting results. Fig. 9 shows the variation of the RMSE according to the hyperparameters. We can clearly see that the smallest RMSE (better performing model) is at the point cost=5 and sigma=0.05. With these hyperparameters, the RMSE is estimated at 0.015 and the R^2 at 0.76. These estimates are made on the training data.

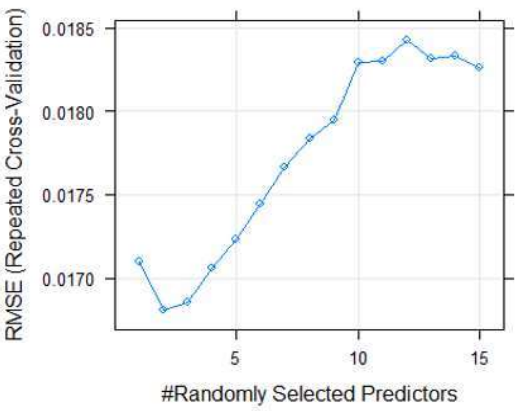

Figure 8: Hyperparameter tuning - Random Forest.

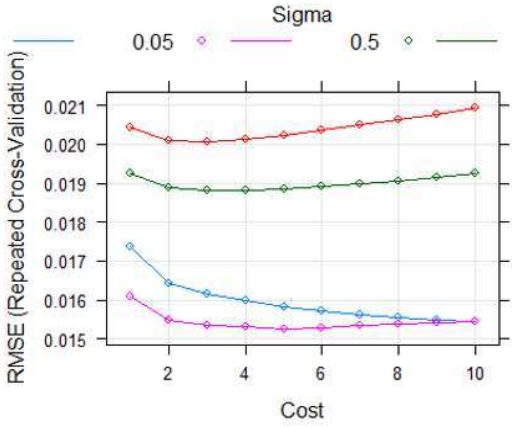

Figure 9: Hyperparameter tuning – SVM.

The SVM model appears to be the best model with a lower RMSE than found in most of the review. To investigate the performance of the model in new data, we use the validation data (20% of the data set) to predict thickness losses and estimate the RMSE. Table 6 shows the estimation results of the SVM model on the validation data. The model predicts the thickness loss well with a low RMSE evaluated at 0.011 and R^2 of 0.83. We can see in Fig 10, the small difference between the observed and predicted values of the thickness measurements by running the model on new data (validation data) with the best model (SVM) and the optimal hyperparameters C=5 and sigma=0.05.

Table 6: Result on validation set.

Model	Validation set	
	RMSE	R^2
SVM	0.011	0.83

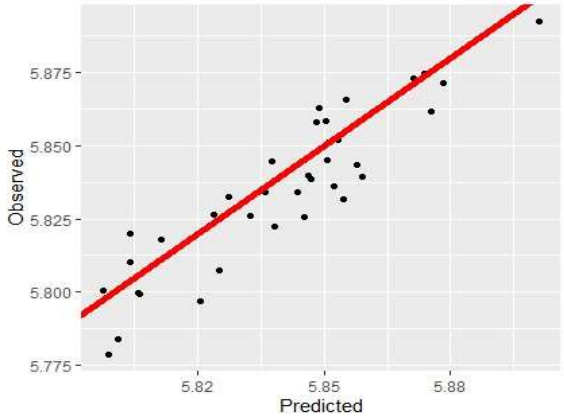

Figure 10: Pipeline wall thickness values: Observed vs. predicted.

5. Conclusions

In the current situation, the majority of the internal pipeline wall is deteriorating due to both slurry erosion and corrosion, which result in the gradual removal of material from the surface due to the impact of solid particles suspended in the liquid phase. As stated earlier, the intent of this paper was not to present the full machine learning model for predicting corrosion erosion degradation in a slurry pipeline, which will be done in a subsequent publication, but rather to take a unique opportunity to describe the methodology in detail as a walk-through corrosion assessment of slurry pipeline using machine learning. Explaining the essential aspects of machine learning tools suitable for a specific dataset is often difficult in a full research paper as much focus needs to be on results and discussion. Whilst the study combined several machine learning techniques to achieve better results, it was found that the SVM, RF and GBM models perform better on the initial data. On the other hand, the KNN model performs better on the principal component data. The change in hyperparameters was important in this analysis, as the SVM model went from an RMSE of 0.016 to 0.011,

remaining the best model for predicting pipeline thickness loss. This information is useful to pipeline corrosion professionals who consistently plan corrosion mitigation activities through risk-based inspections. A limitation of this research is the limited amount of data. Future work will focus on continuously collecting large amounts of data to make more accurate predictions for different pipeline locations using machine learning models.

Funding: This research was funded by NATURAL SCIENCES AND ENGINEERING RESEARCH COUNCIL OF CANADA (NSERC) AND THE QUÉBEC FONDS DE RECHERCHE NATURE ET TECHNOLOGIES (FRQNT).

Data Availability Statement: Agnico Eagle Mine Goldex Data.

Restrictions apply to the availability of these data. Data was obtained from Agnico Eagle Mine Goldex and are available from the authors with the permission of Agnico Eagle Mine Goldex.

Acknowledgments: The authors are thankful of Agnico Eagle Goldex Mine and the Quebec Metallurgy Centre for their support.

Conflicts of Interest: The authors declare no conflict of interest.

References

[1] EDDY Pump Corporation. *Slurry Pipeline Changes: What Effect Can It Have On Your Operation?*

[2] Chung, R. J.; Jiang, J.; Pang, C.; Yu, B.; Eadie, R.; Li, D. Y. Erosion-Corrosion Behaviour of Steels Used in Slurry Pipelines. *Wear* **2021**, *477*. https://doi.org/10.1016/j.wear.2021.203771.

[3] Jones, M.; Llewellyn, R. J. Erosion-Corrosion Assessment of Materials for Use in the Resources Industry. *Wear* **2009**, *267* (11). https://doi.org/10.1016/j.wear.2009.06.025.

[4] Ma, Q.; Tian, G.; Zeng, Y.; Li, R.; Song, H.; Wang, Z.; Gao, B.; Zeng, K. Pipeline In-Line Inspection Method, Instrumentation and Data Management. *Sensors*. 2021. https://doi.org/10.3390/s21113862.

[5] D'Orazio, T.; Leo, M.; Distante, A.; Guaragnella, C.; Pianese, V.; Cavaccini, G. Automatic Ultrasonic Inspection for Internal Defect Detection in Composite Materials. *NDT and E International* **2008**, *41* (2). https://doi.org/10.1016/j.ndteint.2007.08.001.

[6] Trasatti, S. P. Risk-Based Inspection and Integrity Management of Pipeline Systems. In *Lecture Notes in Civil Engineering*; 2021; Vol. 102. https://doi.org/10.1007/978-3-030-58073-5_7.

[7] Diao, Y.; Yan, L.; Gao, K. Improvement of the Machine Learning-Based Corrosion Rate Prediction Model through the Optimization of Input Features. *Mater Des* **2021**, *198*. https://doi.org/10.1016/j.matdes.2020.109326.

[8] Nesic, S.; Nordsveen, M.; Maxwell, N.; Vrhovac, M. Probabilistic Modelling of CO2 Corrosion Laboratory Data Using Neural Networks. *Corros Sci* **2001**, *43* (7). https://doi.org/10.1016/S0010-938X(00)00157-8.

[9] Nash, W. T.; Powell, C. J.; Drummond, T.; Birbilis, N. Automated Corrosion Detection Using Crowdsourced Training for Deep Learning. *Corrosion* **2020**, *76* (2). https://doi.org/10.5006/3397.

[10] Ossai, C. I. A Data-Driven Machine Learning Approach for Corrosion Risk Assessment—a Comparative Study. *Big Data and Cognitive Computing* **2019**, *3* (2). https://doi.org/10.3390/bdcc3020028.

[11] Sanchez, G.; Aperador, W.; Cerón, A. Corrosion Grade Classification: A Machine Learning Approach. *Indian Chemical Engineer* **2020**, *62* (3). https://doi.org/10.1080/00194506.2019.1675539.

[12] Yan, L.; Diao, Y.; Lang, Z.; Gao, K. Corrosion Rate Prediction and Influencing Factors Evaluation of Low-Alloy Steels in Marine Atmosphere Using Machine Learning Approach. *Sci Technol Adv Mater* **2020**, *21* (1). https://doi.org/10.1080/14686996.2020.1746196.

[13] Salami, B. A.; Rahman, S. M.; Oyehan, T. A.; Maslehuddin, M.; Al Dulaijan, S. U. Ensemble Machine Learning Model for Corrosion Initiation Time Estimation of Embedded Steel Reinforced Self-Compacting Concrete. *Measurement (Lond)* **2020**, *165*. https://doi.org/10.1016/j.measurement.2020.108141.

[14] Gong, X.; Dong, C.; Xu, J.; Wang, L.; Li, X. Machine Learning Assistance for Electrochemical Curve Simulation of Corrosion and Its Application. *Materials and Corrosion* **2020**, *71* (3). https://doi.org/10.1002/maco.201911224.

[15] Aulia, R.; Tan, H.; Sriramula, S. Prediction of Corroded Pipeline Performance Based on Dynamic Reliability Models. In *Procedia CIRP*; 2019; Vol. 80. https://doi.org/10.1016/j.procir.2019.01.093.

[16] Dia, A. K.; Ghazzali, N.; Bosca, A. G. Unsupervised Neural Network for Data-Driven Corrosion Detection of a Mining Pipeline. In *The 35th International FLAIRS Conference*; Florida, 2022. https://doi.org/10.32473/flairs.v35i.130688.

[17] Zhang, J.; Zhang, M.; Dong, B.; Ma, H. Quantitative Evaluation of Steel Corrosion Induced Deterioration in Rubber Concrete by Integrating Ultrasonic Testing, Machine Learning and Mesoscale Simulation. *Cem Concr Compos* **2022**, *128*. https://doi.org/10.1016/j.cemconcomp.2022.104426.

[18] Velázquez, J. C.; Hernández-Sánchez, E.; Terán, G.; Capula-Colindres, S.; Diaz-Cruz, M.; Cervantes-Tobón, A. Probabilistic and Statistical Techniques to Study the Impact of Localized Corrosion Defects in Oil and Gas Pipelines: A Review. *Metals*. 2022. https://doi.org/10.3390/met12040576.

[19] Wei, X.; Fu, D.; Chen, M.; Wu, W.; Wu, D.; Liu, C. Data Mining to Effect of Key Alloying Elements on Corrosion Resistance of Low Alloy Steels in Sanya Seawater EnvironmentAlloying Elements. *J Mater Sci Technol* **2021**, *64*. https://doi.org/10.1016/j.jmst.2020.01.040.

[20] Roy, A.; Taufique, M. F. N.; Khakurel, H.; Devanathan, R.; Johnson, D. D.; Balasubramanian, G. Machine-Learning-Guided Descriptor Selection for Predicting Corrosion Resistance in Multi-Principal Element Alloys. *Npj Mater Degrad* **2022**, *6* (1), 9. https://doi.org/10.1038/s41529-021-00208-y.

[21] Zhao, Z.; Chen, M.; Fan, H.; Zhang, N. Data Analysis and Knowledge Mining of Machine Learning in Soil Corrosion Factors of the Pipeline Safety. *Comput Intell Neurosci* **2022**, *2022*. https://doi.org/10.1155/2022/9523878.

[22] Aghaaminiha, M.; Mehrani, R.; Colahan, M.; Brown, B.; Singer, M.; Nesic, S.; Vargas, S. M.; Sharma, S. Machine Learning Modeling of Time-Dependent Corrosion Rates of Carbon Steel in Presence of Corrosion Inhibitors. *Corros Sci* **2021**, *193*. https://doi.org/10.1016/j.corsci.2021.109904.

[23] Peng, S.; Zhang, Z.; Liu, E.; Liu, W.; Qiao, W. A New Hybrid Algorithm Model for Prediction of Internal Corrosion Rate of Multiphase Pipeline. *J Nat Gas Sci Eng* **2021**, *85*. https://doi.org/10.1016/j.jngse.2020.103716.

[24] Adamowski, J. C.; Buiochi, F.; Tsuzuki, M.; Perez, N.; Camerini, C. S.; Patusco, C. Ultrasonic Measurement of Micrometric Wall-Thickness Loss Due to Corrosion inside Pipes. In *IEEE International Ultrasonics Symposium, IUS*; 2013. https://doi.org/10.1109/ULTSYM.2013.0479.

[25] Guyon, I.; Weston, J.; Barnhill, S.; Vapnik, V. Gene Selection for Cancer Classification Using Support Vector Machines. *Mach Learn* **2002**, *46* (1–3). https://doi.org/10.1023/A:1012487302797.

[26] Kuhn, M.; Johnson, K. *Feature Engineering and Selection: A Practical Approach for Predictive Models*; 2019. https://doi.org/10.1201/9781315108230.

[27] Svetnik, V.; Liaw, A.; Tong, C.; Christopher Culberson, J.; Sheridan, R. P.; Feuston, B. P. Random Forest: A Classification and Regression Tool for Compound Classification and QSAR Modeling. *J Chem Inf Comput Sci* **2003**, *43* (6). https://doi.org/10.1021/ci034160g.

[28] Abdi, H.; Williams, L. J. Principal Component Analysis. *Wiley Interdisciplinary Reviews: Computational Statistics*. 2010. https://doi.org/10.1002/wics.101.

[29] Cover, T. M.; Hart, P. E. Nearest Neighbor Pattern Classification. *IEEE Trans Inf Theory* **1967**, *13* (1). https://doi.org/10.1109/TIT.1967.1053964.

[30] Ho, T. *Random Decision Forests*; 1995; Vol. 1. https://doi.org/10.1109/ICDAR.1995.598994.

[31] Breiman, L. Random Forests. *Mach Learn* **2001**, *45*, 5–32. https://doi.org/10.1023/A:1010950718922.

[32] Friedman, J. H. Greedy Function Approximation: A Gradient Boosting Machine. *Ann Stat* **2001**, *29* (5). https://doi.org/10.1214/aos/1013203451.

[33] Lu, H.; Karimireddy, S.; Ponomareva, N.; Mirrokni, V. *Accelerating Gradient Boosting Machine*; 2019.

[34] Cortes, C.; Vapnik, V. Support-Vector Networks. *Mach Learn* **1995**, *20* (3), 273–297. https://doi.org/10.1007/BF00994018.

[35] Cortes, C.; Vapnik, V. Support-Vector Networks. *Chem. Biol. Drug Des.* **2009**, *297*, 273–297. https://doi.org/10.1007/%2FBF00994018.

[36] Awad, M.; Khanna, R. *Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers*; 2015. https://doi.org/10.1007/978-1-4302-5990-9.

[37] Bélanger, M.; El-Jabi, N.; Caissie, D.; Ashkar, F.; Ribi, J. M. Estimation de La Température de l'eau de Rivière En Utilisant Les Réseaux de Neurones et La Régression Linéaire Multiple. *Revue des sciences de l'eau* **2005**, *18* (3). https://doi.org/10.7202/705565ar.

[38] Stone, M. Cross-Validatory Choice and Assessment of Statistical Predictions. *Journal of the Royal Statistical Society: Series B (Methodological)* **1974**, *36* (2). https://doi.org/10.1111/j.2517-6161.1974.tb00994.x.

[39] Claesen, M.; De Moor, B. Hyperparameter Search in Machine Learning. **2015**.

[40] Feurer, M.; Hutter, F. Hyperparameter Optimization; 2019; pp 3–33. https://doi.org/10.1007/978-3-030-05318-5_1.

[41] Tang, F.; Wu, Y.; Zhou, Y. Hybridizing Grid Search and Support Vector Regression to Predict the Compressive Strength of Fly Ash Concrete. *Advances in Civil Engineering* **2022**, *2022*, 3601914. https://doi.org/10.1155/2022/3601914.

Annexe C : Code R pour les articles 1 et 2

Article 1

```
##Chargement des librairies
library(kohonen)
library(readxl)
library("clusterSim")
library("factoextra")
Rep = getwd()
setwd((Rep))
df= read_xlsx("donne_som.xlsx")
df= data.frame(df, row.names = 1)
# Creation de la grille ? utiliser pour le SOM
df_grid <- somgrid(xdim = 5, ydim = 5, topo = "hexagonal")
# Fixer la graine aleatoire
set.seed(2021)
# Utilisation de la fonction SOM
df_som <- som(X = df, grid = df_grid)
#gamme de couleurs pour les cellules de la carte
degrade.bleu <- function(n){
  return(rgb(0,0.4,1,alpha=seq(0,1,1/n)))
}
#Graphique des noeuds
plot(df_som, type="count", main="Node Counts",palette.name=degrade.bleu)
# Nombre d'instances assignés à chaque noeud
nb <- table(df_som$unit.classif)
print(nb)
# visualisation de U-matrix
plot(df_som, type="dist.neighbours", main = "SOM neighbour distances")
```

```
#Fonction de couleur pour les graphiques
coolBlueHotRed <- function(n, alpha = 1) {
  rainbow(n, end=4/6, alpha=alpha)[n:1]
}
## Tracer le heatmaps à la troisieme variable
dfsom=as.data.frame(df_som$codes)
plot(df_som,type="property",property=dfsom[,3],palette.name=coolBlueHot
ed,main=colnames(df)[3],cex=0.5)
  ##  clustering AVEC CAH
# Matrice de distance entre les cellules
dfcode=as.data.frame(df_som$codes)
dc <- dist(dfcode)
res.hc <- hclust(d = dc, method = "ward.D2")
fviz_dend(res.hc, k = 3,  cex = 0.8,  k_colors = c("#FC4E07", "#00AFBB",
"#00FF00"), color_labels_by_k = TRUE,  rect = TRUE, # Add rectangle
around groups rect_border = c("#FC4E07", "#00AFBB", "#00FF00"),
rect_fill = TRUE)
## Regroupement en  3 clusters
groupes <- cutree(res.hc,k=3)
print(groupes)
#Visualisation des clusters dans la carte
plot(df_som,type="mapping",bgcol=c("steelblue1","sienna1","yellowgreen")[gro
upes])
add.cluster.boundaries(df_som,clustering=groupes)
write.csv(df, file="base.csv")
#Assignation de chaque instance dans un cluster
ind.groupe <- groupes[df_som$unit.classif]
```

```
print(ind.groupe)

length(ind.groupe)

df$ind.cluster = ind.groupe
```

Article 2

```
### Chargement des librairies
library(readxl)
library(caret)
library(dplyr)
library(e1071)
library(ggplot2)
library(neuralnet)
Rep = getwd()
setwd((Rep))
df= read_xlsx("data_by_probe.xlsx")
df <-as.data.frame(Data_Modele)
## Fixer la graine aléatoire
set.seed(7)
## Séparation des données en données d'entrainement et données de
validation
validationIndex <- createDataPartition(df$Thickness, p=0.80, list=FALSE)
validation <- df[-validationIndex,]
dataset <- df[validationIndex,]
# Validation croisée
trainControl <- trainControl(method="repeatedcv", number=5, repeats=3)
metric <- "RMSE"
## Modeles sur l'ensemble des données
# SVM
set.seed(7)
```

```
fit.svm <- train(Thickness~., data=dataset, method="svmRadial",
metric=metric, trControl=trainControl)
# KNN
set.seed(7)
fit.knn <- train(Thickness~., data=dataset, method="knn", metric=metric,
trControl=trainControl)
##RF
set.seed(7)
fit.rf <- train(Thickness~., data=dataset, method="rf", metric=metric,
trControl=trainControl)
# GBM
set.seed(7)
fit.gbm <- train(Thickness~., data=dataset, method="gbm", metric=metric,
trControl=trainControl, verbose=FALSE)
# Compare algorithms
results <- resamples(list( SVM=fit.svm, KNN=fit.knn, RF=fit.rf,
GBM=fit.gbm))
summary(results)
dotplot(results)
statdiff= diff(results)
summary(statdiff)
##Modele final sur les données de validation
set.seed(7)
valX <- validation[,1:14]
valY <- validation[,15]
# # Prediction sur les données de validation
predictionsrf <- predict(fit.rf, newdata=valX)
```

```
plot(valY, predictionsrf)
# calculate RMSE
rmserf <- RMSE(predictionsrf, valY)
r2rf <- R2(predictionsrf, valY)
# # Prediction sur les données de validation
predictionssvm <- predict(fit.svm, newdata=valX)
# calculate RMSE
rmsesvm <- RMSE(predictionssvm, valY)
r2svm <- R2(predictionssvm, valY)
predictionsknn <- predict(fit.knn, newdata=valX)
# calculate RMSE
rmseknn <- RMSE(predictionsknn, valY)
r2knn <- R2(predictionsknn, valY)
predictionsgbm <- predict(fit.gbm, newdata=valX)
# calculate RMSE
rmsegbm <- RMSE(predictionsgbm, valY)
r2gbm <- R2(predictionsgbm, valY)
### Modèles avec selection de variable
## Selection des variables avec l'elimination recursive des caracteristiques
set.seed(7)
control <- rfeControl(functions=rfFuncs, method="repeatedcv", number=5,
repeats=3)
results <- rfe(dataset[,1:14], dataset[,15], sizes=c(1:14), rfeControl=control,
preProc=c("BoxCox"))
print(results)
predictors(results)
plot(results, type=c("g", "o"))
```

```
## Variables selections
dataset2 <- dataset %>% select("T_pulpe_Flottation" ,
"T_Km0","Temp_PS", "pour_solide_residu", "Temp_BS",
"TPH_residu_calcule", "Pression_Km0", "Tonnage","Debit_PS",
"Debit_RV", "Thickness")
# SVM
set.seed(7)
fit.svm2 <- train(Thickness~., data=dataset2, method="svmRadial",
metric=metric, preProc=c("center", "scale"), trControl=trainControl)
# KNN
set.seed(7)
fit.knn2 <- train(Thickness~., data=dataset2, method="knn", metric=metric,
preProc=c("center", "scale"), trControl=trainControl)
set.seed(7)
fit.rf2 <- train(Thickness~., data=dataset2, method="rf", metric=metric,
preProc=c("center", "scale"), trControl=trainControl)
# GBM
set.seed(7)
fit.gbm2 <- train(Thickness~., data=dataset2, method="gbm", metric=metric,
preProc=c("center", "scale"), trControl=trainControl, verbose=FALSE)
# Comparer les algorithmes
results <- resamples(list( SVM=fit.svm2, KNN=fit.knn2, RF=fit.rf2,
GBM=fit.gbm2))
summary(results)
dotplot(results)
statdiff2 = diff(results)
summary(statdiff2)
```

```
##Modele final sur les données de validation
set.seed(7)
valX <- validation %>% select("T_pulpe_Flottation" ,
"T_Km0","Temp_PS", "pour_solide_residu", "Temp_BS",
"TPH_residu_calcule", "Pression_Km0", "Tonnage","Debit_PS",
"Debit_RV", "Thickness")
valY <- validation[,15]
# Prediction sur les données de validation
predictionsrf <- predict(fit.rf2, newdata=valX)
plot(valY, predictionsrf)
# calcule RMSE
rmserf <- RMSE(predictionsrf, valY)
r2rf <- R2(predictionsrf, valY)
# Prediction sur les données de validation
predictionssvm <- predict(fit.svm, newdata=valX)
# calcule RMSE
rmsesvm <- RMSE(predictionssvm, valY)
r2svm <- R2(predictionssvm, valY)
predictionsknn <- predict(fit.knn, newdata=valX)
# calcule RMSE
rmseknn <- RMSE(predictionsknn, valY)
r2knn <- R2(predictions, valY)

predictionsgbm <- predict(fit.gbm, newdata=valX)
# calcule RMSE
rmsegbm <- RMSE(predictionsgbm, valY)
r2gbm <- R2(predictionsgbm, valY)
```

```
### Modele avec selection des variables avec ACP
df <-as.data.frame(Data_Modele)
df.pr <- prcomp(df[c(1:14)], center = TRUE, scale = TRUE)
summary(df.pr)
screeplot(df.pr, type = "l", npcs = 14, main = "Screenplot of the 14 PCs")
abline(h = 0.3, col="red", lty=5)
legend("topright", legend=c("Eigenvalue = 1"), col=c("red"), lty=5, cex=0.6)
cumpro <- cumsum(df.pr$sdev^2 / sum(df.pr$sdev^2))
plot(cumpro[0:14], xlab = "PC #", ylab = "Amount of explained variance",
main = "Cumulative variance plot")
abline(v = 8, col="blue", lty=5)
abline(h = 0.97773, col="blue", lty=5)
legend("topleft", legend=c("Cut-off @ PC8"), col=c("blue"), lty=5, cex=0.6)
plot(df.pr$x[,1],df.pr$x[,2], xlab="PC1 (41.5%)", ylab = "PC2 (22%)", main =
"PC1 / PC2 - plot")
df.pcst <- df.pr$x[,1:8]
df.pcst <- cbind(df.pcst, as.numeric(df$Thickness))
colnames(df.pcst)[9] <- "Thickness"
dataset3 = as.data.frame(df.pcst)
## Modelisation avec les composantes principales
# SVM
set.seed(7)
fit.svm3 <- train(Thickness~., data=dataset3, method="svmRadial",
metric=metric, trControl=trainControl)
# KNN
set.seed(7)
```

```
fit.knn3 <- train(Thickness~., data=dataset3, method="knn", metric=metric,
trControl=trainControl)
set.seed(7)
fit.rf3 <- train(Thickness~., data=dataset3, method="rf", metric=metric,
trControl=trainControl)
# GBM
set.seed(7)
fit.gbm3 <- train(Thickness~., data=dataset3, method="gbm", metric=metric,
trControl=trainControl, verbose=FALSE)
# Compare algorithms
results <- resamples(list( SVM=fit.svm3, KNN=fit.knn3, RF=fit.rf3,
GBM=fit.gbm3))
summary(results)
dotplot(results)
rmsesvm2 = fit.knn2$resample$Rsquared
rmsesvm3 = fit.knn3$resample$Rsquared
t.test(rmsesvm2, rmsesvm3)
##Modele final sur les données de validation
set.seed(7)
df.pr <- prcomp(validation[c(1:14)], center = TRUE, scale = TRUE)
summary(df.pr)
df.pcst <- df.pr$x[,1:8]
## Modeles sur les données de validation
valX <- as.data.frame(df.pcst)
valY <- validation[,9]
predictionsrf <- predict(fit.rf3, newdata=valX)
# calcule RMSE
```

```
rmserf <- RMSE(predictionsrf, valY)
r2rf <- R2(predictionsrf, valY)
predictionssvm <- predict(fit.svm3, newdata=valX)
# calculate RMSE
rmsesvm <- RMSE(predictionssvm, valY)
r2svm <- R2(predictionssvm, valY)
predictionsknn <- predict(fit.knn3, newdata=valX)
# calcule RMSE
rmseknn <- RMSE(predictionsknn, valY)
r2knn <- R2(predictionsknn, valY)
predictionsgbm <- predict(fit.gbm3, newdata=valX)
# calcule RMSE
rmsegbm <- RMSE(predictionsgbm, valY)
r2gbm <- R2(predictionsgbm, valY)
### Perceptron Multicouche
df <-as.data.frame(Data_Modele)
data <- as.matrix(df)
set.seed(123)
## Preparation des données en entrainement et test
ind <- sample(2, nrow(df), replace = T, prob = c(.8, .2))
training <- data[ind==1,1:14]
test <- data[ind==2, 1:14]
trainingtarget <- data[ind==1, 15]
testtarget <- data[ind==2, 15]
## Normalisation des données
m <- colMeans(training)
s <- apply(training, 2, sd)
```

```
training <- scale(training, center = m, scale = s)
test <- scale(test, center = m, scale = s)
training = as.data.frame(training)
trainingtarget = as.data.frame(trainingtarget)
training$Thickness = trainingtarget$trainingtarget
training$thickness = trainingtarget
## Perceptron Multicouche avec 10 variables selectionnées par RFE
n <- neuralnet(Thickness
~T_pulpe_Flottation+T_Km0+Temp_PS+pour_solide_residu+Temp_BS
+TPH_residu_calcule+Pression_Km0+Tonnage+Debit_PS+Debit_RV,
data = training, hidden = c(6,2), linear.output = T, lifesign = 'full', act.fct =
"tanh", rep=1)
##Perceptron Multicouche avec l'ensemble des variables
n <- neuralnet(Thickness
~Tonnage+T_pulpe_Flottation+pH_flottation+debit_de_residu
+debit_de_residu+pour_solide_residu
+TPH_residu_calcule+Pression_Km0+T_Km0+Debit_RV+Temp_RV+D
ebit_BS+Temp_BS+Temp_PS, data = training, hidden = c(6,2),
linear.output = T, lifesign = 'full', act.fct = "tanh", rep=1)
## Prediction
training2 = training[-15]
pred_nn_tmp <- predict(n, test)
stats_tmp <- postResample(pred_nn_tmp, testtarget)
stats_tmp
### Optimization des hyperparametres
### SVM
trainControl <- trainControl(method="repeatedcv", number=5, repeats=3)
```

```
metric <- "RMSE"
set.seed(7)
grid <- expand.grid(.sigma=c(0.025, 0.05, 0.5, 0.7), .C=seq(1, 10, by=1))
fit.svm <- train(Thickness~., data=dataset2, method="svmRadial",
metric=metric, tuneGrid=grid, preProc=c("center",
"scale", "BoxCox"), trControl=trainControl)
print(fit.svm)
plot(fit.svm)
## RF
control <- trainControl(method="repeatedcv", number=5, repeats=3,
search="grid")
set.seed(7)
tunegrid <- expand.grid(.mtry=c(1:15))
rf_gridsearch <- train(Thickness~., data=dataset2, method="rf",
metric=metric, tuneGrid=tunegrid, trControl=control,
preProc=c("center","scale", "BoxCox"))
print(rf_gridsearch)
plot(rf_gridsearch)
rf_gridsearch
# Recherche manuelle
control <- trainControl(method="repeatedcv", number=5, repeats=3,
search="grid")
tunegrid <- expand.grid(.mtry=c(1:15))
modellist <- list()
for (ntree in c(1000, 1500, 2000, 2500, 3000, 3500)) {
  set.seed(7)
```

```
  fit <- train(Thickness~., data=dataset2, method="rf", metric=metric,
tuneGrid=tunegrid, trControl=control, ntree=ntree, preProc=c("center",
"scale", "BoxCox"))
  key <- toString(ntree)
  modellist[[key]] <- fit
}
## Modele sur les données de validation
set.seed(7)
x <- dataset2[,1:10]
y <- dataset2[,11]
maxs <- apply(dataset2[,1:10], 2, max)
mins <- apply(dataset2[,1:10], 2, min)
x = scale(dataset2[,1:10], center = mins, scale = maxs - mins)
# Modele final
finalModel <- svm(x=x, y=y, cost=5, gamma=0.05)
summary(finalModel)
set.seed(7)
valX <- validation %>% select("T_pulpe_Flottation" ,
"T_Km0","Temp_PS", "pour_solide_residu", "Temp_BS",
                              "TPH_residu_calcule", "Pression_Km0",
"Tonnage","Debit_PS", "Debit_RV")
valY <- validation[,15]
maxs <- apply(valX, 2, max)
mins <- apply(valX, 2, min)
valX = scale(valX, center = mins, scale = maxs - mins)
# Predictions
predictions <- predict(finalModel, newdata=valX)
```

```
plot(predictions, valY)
## Graphique des Valeur observes vs valeurs predites
dataplot = data.frame(Observed = valY, Predicted = predictions)
ggplot(dataplot aes(x = Predicted, y = Observed)) + geom_point() +
  geom_abline(intercept = 0, slope = 1, color = "red", size = 2)
```