ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

RAPPORT DE PROJET PRÉSENTÉ À
L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

COMME EXIGENCE PARTIELLE À L'OBTENTION DE
LA MAÎTRISE EN GÉNIE LOGICIEL

Par

Perside GBÈHOUNOU

APPRENTISSAGE AUTOMATIQUE POUR LA PRÉDICTION DES DÉFAUTS DE PRODUCTION DE PIÈCES MANUFACTURÉES

MONTRÉAL, LE 2019/03/26

PRÉSENTATION DU JURY

CE RAPPORT DE PROJET A ÉTÉ ÉVALUÉ

PAR UN JURY COMPOSÉ DE :

Professeur Alain APRIL directeur de projet
Département de Génie logiciel et TI à l'École de technologie supérieure

Professeur Abdelaoued GHERBI jury
Département de Génie logiciel et TI à l'École de technologie supérieure

REMERCIEMENTS

Je tiens à remercier toutes les personnes qui ont contribué à ce projet.

Je remercie en particulier le Professeur Alain April qui m'a fait confiance et m'a proposé ce projet dans le cadre de ma maîtrise. Il a été disponible le long de ce travail en donnant des conseils et en corrigeant la plupart du travail de rédaction qui lui était envoyé.

Je tiens aussi à remercier mon compagnon Lalèyè José-Marie, dont la compréhension du domaine manufacturier m'a permis de développer certaines de mes idées sur l'utilisation des données.

Je remercie aussi ma sœur Syntyche Gbèhounou, dont l'expertise en intelligence artificielle m'a beaucoup aidée grâce à ses conseils.

APPRENTISSAGE AUTOMATIQUE POUR LA PRÉDICTION DES DÉFAUTS DE PRODUCTION DE PIÈCES MANUFACTURÉES

Perside GBÈHOUNOU

RÉSUMÉ

L'objectif de ce projet de maîtrise 15 crédits est d'utiliser deux techniques d'apprentissage machine pour faire la prédiction des défauts de fabrication des pièces automobiles. Les données utilisées proviennent de machines à mesurer tridimensionnelles qui sont utilisées pour vérifier la conformité des pièces.

La première étape dans la réalisation du projet est de faire une revue de la littérature pour connaître les méthodes d'apprentissage machine utilisée pour la prédiction des défauts de fabrication manufacturière.

La deuxième étape est l'analyse des données fournies, qui combinée à la première étape du projet, va permettre de choisir deux techniques parmi les plus utilisées et les plus performantes pour faire les expérimentations.

L'étape suivante qui est cruciale en apprentissage machine est le traitement des données avant les étapes d'entraînement et l'évaluation des modèles sur les données. Il faut labelliser les données, les harmoniser puis choisir aléatoirement celles qui vont servir pour les phases d'entraînement et de tests.

La dernière étape de ce projet est l'évaluation et la discussion des résultats obtenus. Cette étape permettra de proposer des pistes d'amélioration des diverses techniques expérimentées.

Mots-clés : Apprentissage machine, analyse de données, prédiction de défauts.

MACHINE LEARNING FOR THE PREDICTION OF MANUFACTURED PARTS PRODUCTION DEFECTS

Perside GBÈHOUNOU

ABSTRACT

The purpose of this 15 credits master project is to use two machine-learning algorithms to predict manufacturing defects of manufactured automotive parts. The data used for the training of the algorithm was taken from coordinate-measuring machines that are used to verify the conformity of the manufactured parts.

The first step in the realization of the project was a review of the literature to learn about the various algorithms used for the prediction of manufacturing defects.

The second stage was the analysis of the data, which used knowledge acquired during the first stage of the project, in order to choose two algorithms to be experimented.

The next central part of this research is data processing, before the training steps, and evaluating the performance of the prediction of each algorithm on this data. We have found that it is important to label the data, harmonize it and then randomly choose the data fields that are to be used for the training and testing phases. This gave us the best results.

The final stage of this project was the evaluation, and the discussion of the results obtained. This is followed by suggestions for improving on the two algorithms tested.

Keywords: machine learning, data analysis, fault prediction.

TABLE DES MATIÈRES

Page

INTRODUCTION 1

CHAPITRE 1 REVUE BIBLIOGRAPHIQUE 5
1.1 Types de défauts et leurs origines 5
1.2 Enjeux du contrôle qualité 7
1.3 L'apprentissage machine 7
1.3.1 Algorithmes utilisés par l'industrie manufacturière 8
1.3.2 Résultats de l'utilisation d'algorithmes d'apprentissage machine 9
1.4 CONCLUSION 11

CHAPITRE 2 ANALYSE ET TRAITEMENT DES DONNÉES 13
2.1 Analyse des données 13
2.1.1 Présentation des données 13
2.1.2 Problèmes détectés 15
2.2 Prétraitement et labellisation des données 18
2.3 Choix des algorithmes 21

CHAPITRE 3 APPRENTISSAGE MACHINE 23
3.1 Configuration de l'environnement de développement 23
3.2 Évaluation de la performance 23
3.3 Tâches d'apprentissage 24
3.4 Taxonomies des algorithmes sélectionnés 27
3.4.1 Perceptron multicouche 27
3.4.2 Forêts aléatoires 28

CHAPITRE 4 RÉSULTATS 29
4.1 Base de tests 29
4.1.1 Résultats du Perceptron multicouche 30
4.1.2 Résultats des forêts aléatoires 32
4.2 Discussion 34

CONCLUSION 37

RECOMMANDATIONS 39

ANNEXE I EXEMPLES DE FICHIERS DE CONTRÔLE 40

ANNEXE II LISTE DES CONTRÔLES ET LE NOMBRE DE FICHIERS ASSOCIÉS 43

LISTE DE RÉFÉRENCES BIBLIOGRAPHIQUES 47

BIBLIOGRAPHIE..49

LISTE DES TABLEAUX

Page

Tableau 2.1 : Liste des différents types de caractéristiques contrôlées par les CMM 14

Tableau 2.2 : Nombre de mesures hors tolérance par type de caractéristique 17

Tableau 2.3 : Répartition des classes sur l'ensemble de la base de données 18

Tableau 2.4 : Caractéristiques sélectionnées en fonction de leur domaine de définition 21

Tableau 3.1 : Répartition des données dans les deux bases 25

Tableau 4.1 : Pièces contrôlées 29

Tableau 4.2 : Répartition des classes 30

Tableau 4.4 : résultats sur la base de tests 33

Tableau 4.5 : importance des caractéristiques dans la prise de décision des arbres 34

LISTE DES FIGURES

Page

Figure 3.1 : Organisation des tâches d'apprentissage 26

Figure 4.1 : Matrice de confusion de la base de tests avec Softsign comme fonction d'activation 31

Figure 4.2 : Forêts aléatoires avec 1 arbre 32

Figure 4.3 : Forêts aléatoires avec 5 arbres.....32

Figure 4.4 : Forêts aléatoires avec 25 arbres.....33

Figure 4.5 : Forêts aléatoires avec 100 arbres.....33

LISTE DES ABRÉVIATIONS, SIGLES ET ACRONYMES

Abréviations, sigles et acronymes	**Définition**
CMM	Cordinate-measuring Machine/machine à mesure tridimensionnelle
CSV	Comma-separated values
SVM	Support Vector Machine/Séparateur à vaste marge)

INTRODUCTION

Le contrôle qualité est une étape importante d'un processus de fabrication de pièces manufacturées dans l'industrie, peu importe le secteur d'activités. Bien qu'il soit coûteux pour les industriels, il est incontournable, car les clients désirent un produit qui rencontre leurs exigences. En effet, il permet de détecter les défauts de fabrication présents sur des pièces manufacturées est préjudiciables pour la vente directe et lors de l'assemblage d'un autre produit. Ainsi il est incontournable que les spécifications fournies par les clients soient vérifiées et respectées. Les pièces qui présentent des défauts sont écartées de la suite du processus pour être réparées ou recyclées. Un nombre important de pièces défectueuses crée un impact négatif sur le rendement et la productivité d'une usine de fabrication.

Dans certains secteurs d'activités, comme l'industrie automobile, cette activité est obligatoire. En effet, plusieurs facteurs doivent être pris en compte ici :

- La nature dangereuse d'un défaut dans une pièce maîtresse d'une automobile peut mettre en danger la vie du conducteur,
- Les véhicules sont des assemblages de plusieurs pièces différentes qui elles-mêmes proviennent de sources diverses. Le risque de présence de défauts est donc augmenté et il est donc impératif de vérifier chaque pièce.

Les industriels utilisent plusieurs techniques de contrôle qui varient selon les pièces et les types de vérification à faire. On peut citer :

- Les tests de stress et d'usure qui permettent de vérifier la qualité du matériel et sa résistance dans des conditions d'utilisation extrêmes,
- L'inspection de pièces qui permet de vérifier que les pièces sont conformes aux caractéristiques définies et ne présentent pas des défauts physiques,
- L'essai et le réglage des machines et outils de fabrication pour s'assurer que les calibrages sont bien faits et que les machines fonctionnent correctement,

- Le contrôle 3D des pièces pour comparer les dimensions des objets par rapport à leurs modèles numériques.

Il existe une multitude de machines et outils utilisés pour réaliser les tâches de contrôle qualité citées ci-dessus. Les industriels ont la possibilité d'utiliser des techniques avancées d'imagerie, des capteurs 3D d'une extrême précision ou encore des machines à mesurer tridimensionnelles (CMM). Les machines à mesurer tridimensionnelles permettent de contrôler les dimensions de pièces à la suite de leur fabrication. Elles permettent de récupérer un grand nombre d'informations sur les pièces testées afin de les accepter ou de les rejeter. Pour chaque caractéristique à mesurer, la machine connaît la valeur nominale attendue ainsi que les valeurs de l'intervalle de tolérance spécifié par le client. La CMM enregistre la valeur mesurée et un logiciel spécialisé calcule plusieurs valeurs comme la différence entre la valeur mesurée et la valeur nominale, ou la différence entre la valeur mesurée et la borne la plus proche de l'intervalle de tolérance. Toutes les mesures faites sont utilisées pour accepter ou refuser les pièces contrôlées et sont ensuite stockées dans le système de l'entreprise. Ces données stockées constituent une source importante d'information qui peut être utilisée plus tard. C'est là qu'intervient l'apprentissage machine.

L'apprentissage machine consiste à utiliser des algorithmes pour permettre aux ordinateurs d'apprendre à reconnaître des motifs, comprendre un processus ou un phénomène, faire des prédictions en utilisant des données, etc. On peut par exemple grâce à l'apprentissage machine apprendre à un ordinateur à reconnaître les traits du visage, ce qui lui permettra ensuite de détecter les visages dans une image par exemple. Les avancées dans le domaine sont très importantes, car les algorithmes sont constamment améliorés et de plus en plus performants.

Les entreprises manufacturières utilisent de plus en plus l'intelligence artificielle pour trouver des solutions aux problèmes récurrents dans leur organisation, améliorer leurs offres pour mieux correspondre aux attentes des clients, etc. Tout ceci en utilisant les données accumulées au fil des années et qui n'étaient jusque-là pas valorisées. Dans le secteur

automobile, le grand nombre de données provenant des différents outils utilisés pour le contrôle qualité peut donc être utilisé. Elles peuvent permettre de trouver des moyens d'améliorer les processus de contrôle et aider à atteindre un objectif « zéro défaut » de fabrication grâce à l'apprentissage machine.

En effet, malgré tous les contrôles, certaines anomalies peuvent passer entre les mailles du contrôle qualité et se retrouver dans les automobiles vendues. C'est donc un enjeu important sur le plan économique, sur le plan de la sécurité et de la fiabilité des modèles de véhicules, mais aussi pour la réputation des entreprises du secteur.

L'objectif de ce projet de recherche appliquée est d'expérimenter avec deux techniques d'apprentissage machine pour tenter de prédire les défauts de fabrication des pièces automobiles, en utilisant des données provenant de machines à mesurer tridimensionnelles. Le projet de recherche suivra les étapes suivantes :

- Effectuer une revue littéraire sur l'utilisation de l'apprentissage machine pour la prédiction des défauts de fabrication manufacturière,
- Analyser des données provenant d'une CMM,
- Choisir deux algorithmes d'apprentissage machine pour l'à expérimentation,
- Échantillonner et labelliser les données en préparation pour l'apprentissage machine,
- Entrainer les modèles prédictifs,
- Évaluer les résultats des modèles,
- Discussion des résultats obtenus et travaux futurs.

CHAPITRE 1

REVUE BIBLIOGRAPHIQUE

L'objectif de cette revue littéraire est d'explorer les publications existantes qui permettront de répondre à la problématique de recherche. Ces articles doivent tenter de résoudre un problème lié à la prédiction de défauts dans l'industrie manufacturière grâce à des techniques d'apprentissage machine. Cette revue permettra d'identifier quelques types de défauts auxquels l'industrie manufacturière est confrontée et comment elle tente de les corriger grâce à l'apprentissage machine. Elle permettra d'identifier des méthodes utilisées et de comparer les résultats des plus anciennes avec les plus récentes. Cette comparaison permettra de guider le choix de méthodes à bon potentiel pour cette recherche.

1.1 Types de défauts et leurs origines

Les pièces et produits manufacturés dans l'industrie varient en taille, forme, matériaux de composition, utilisation, couleur, etc. Elles se retrouvent par milliers sur le marché, directement accessible au consommateur ou sont réutilisés pour fabriquer d'autres produits. Toutes ces pièces passent par un contrôle qualité plus ou moins rigoureux en fonction du secteur. Les défauts rencontrés sont rarement visibles à l'œil nu et nécessitent l'utilisation d'appareils de haute précision pour être détectés.

Dans l'industrie électronique, lors de la fabrication des circuits intégrés, il est très important de vérifier les défauts de positionnement des composants sur les plaquettes ainsi que de faire des tests d'utilisation intensive pour vérifier la résistance au stress et à l'usure des composants. (Lee-Ing Tong, Wei-I Lee, & Chao-Ton Su, 1997) et (Apte, Weiss, & Grout, 1993) se sont penchés sur ce problème dans leurs travaux de recherche pour trouver un moyen de rendre le processus de contrôle moins coûteux ou moins long.

Tel que présenté par (Wang, 2013) dans son étude, l'entreprise LEGO prend très au sérieux l'inspection de ses pièces. Il faut que les dimensions des composants de jouets soient conformes à leurs spécifications, surtout lorsqu'ils doivent être combinés avec d'autres composants. L'enjeu est non seulement économique, mais il en va aussi de la réputation de l'entreprise.

Les défauts sont de plusieurs types. Parfois ils peuvent apparaître comme des fissures visibles ou non à l'œil nu, des défauts dus à l'usure qu'on remarque surtout les machines de production. Les pièces composées de métal ou de plastique sont sujettes à des déformations telles que décrites par (Teti, 2015) et (Dib, Ribeiro, & Prates, 2018). Lorsque des moules sont utilisés, des défauts de moulures peuvent aussi être présents. La recherche de (Lin, Yao, Ma, & Wang, 2018) porte sur ce même type de défauts et la difficulté de les détecter avec des outils de contrôle utilisant les rayons X. Enfin, on doit noter qu'il est aussi possible qu'un défaut ait été identifié, mais que la cause est une erreur de mesure provenant des machines à mesurer la qualité. En effet, il faut tenir compte des erreurs de mesure pouvant provenir de ces outils et ainsi s'assurer de bien les calibrer comme il est décrit dans l'article de (Laaouina, Nafi, & Mouchtachi, 2016).

En résumé, les pièces manufacturées peuvent présenter une multitude de défauts relatifs à leur taille, le matériau de fabrication, le taux d'usure de la machine de production, etc. Ces défauts sont causés par plusieurs facteurs :

- L'état des machines auquel s'est intéressé (Teti, 2015). En fonction de leur usure, de leur capacité à résister à une utilisation intensive, les machines peuvent produire des pièces défectueuses,
- Les contraintes liées à l'environnement de production comme la température par exemple,
- Les propriétés mécaniques des matériaux à transformer, qui font qu'ils sont plus ou moins sensibles à une méthode de transformation ou une autre
- Les erreurs humaines lors de la conception ou du paramétrage des machines,

- Les erreurs des outils de contrôle qui n'arrivent pas à détecter certaines erreurs ou qui détectent de fausses erreurs.

1.2 Enjeux du contrôle qualité

L'objectif des manufacturiers est de se rapprocher le plus possible d'une production avec zéro-défauts. Il existe ainsi de nombreuses initiatives à ce sujet et l'utilisation des nouvelles technologies occupe une place de plus en plus importante. Les articles de (Wang, 2013) et (Teti, 2015) présentent les enjeux du « Zéro-defect Manufacturing (ZDM) » et le constat principal est la nécessité pour les entreprises d'améliorer leurs processus de fabrication pour produire des articles de meilleure qualité et ainsi augmenter le rendement de leurs unités de production. On retrouve ces mêmes préoccupations dans les travaux de (Lee-Ing Tong, Wei-I Lee, & Chao-Ton Su, 1997) et (Apte, Weiss, & Grout, 1993) qui visent à trouver des solutions pour prédire les défauts de fabrication et augmenter le rendement d'usines de fabrication de composants électroniques.

Une autre préoccupation des entreprises est la réduction des coûts liés à leurs processus de contrôle qualité. Si ce processus est trop coûteux, on peut envisager utiliser l'apprentissage machine pour voir dans quelle mesure il peut être amélioré, tel que présenté par (Apte, Weiss, & Grout, 1993), en utilisant les données provenant des outils de contrôle.
Finalement, les manufacturiers veulent aussi sécuriser et rendre plus fiables leurs systèmes de production et de contrôle qualité. Les principaux objectifs étant de pouvoir à long terme :

- Réduire les coûts,
- Améliorer la qualité des produits,
- Garder ou obtenir une bonne réputation,
- Respecter les normes imposées.

1.3 L'apprentissage machine

Pour atteindre les objectifs décrits à la section précédente, l'utilisation de l'apprentissage machine peut être une avenue intéressante. En effet, les machines de contrôle produisent un nombre important de données qui peuvent être utilisées par ces algorithmes pour la détection

ou la prédiction de défauts. Les techniques d'apprentissage machine sont divisées en deux catégories :

- L'apprentissage supervisé : Il est nécessaire ici d'utiliser des données labellisées pour entraîner son modèle. Les résultats dépendent de la qualité des labels,

- L'apprentissage non supervisé : il n'est pas nécessaire de labelliser ses données. Les algorithmes sont capables d'inférer des comportements récurrents.

1.3.1 Algorithmes utilisés par l'industrie manufacturière

Les techniques d'apprentissage machine le plus utilisées dans la littérature en ce qui concerne le domaine de la fabrication de pièces sont des méthodes d'apprentissage supervisé. Les réseaux de neurones artificiels sont les méthodes les plus utilisées dans les publications récentes. Par exemple (Lin, Yao, Ma, & Wang, 2018) proposent l'utilisation de la technique de « deep learning » et ont développé un algorithme basé sur les réseaux de convolutions qui sont une variante assez performante des réseaux de neurones. Un autre exemple provient de (Dib, Ribeiro, & Prates, 2018) qui présentent l'utilisation d'algorithme du Perceptron multicouche. En plus des réseaux de neurones, on retrouve plusieurs autres méthodes d'apprentissage machine dans les publications de ce domaine :

- Les arbres de décision : (Wang, 2013) (Apte, Weiss, & Grout, 1993) (Dib, Ribeiro, & Prates, 2018),
- La méthode des K plus proches voisins : (Apte, Weiss, & Grout, 1993),
- Les Séparateurs à Vaste Marge (SVM) : (Wang, 2013) (Dib, Ribeiro, & Prates, 2018),
- Les forêts aléatoires : (Dib, Ribeiro, & Prates, 2018),
- Le classifieur naïf bayésien : (Askarian, Benítez, Graells, & Zarghami, 2016), (Dib, Ribeiro, & Prates, 2018).

Certains auteurs ont choisi de comparer les résultats de plusieurs différentes méthodes pour tenter d'identifier laquelle des techniques est la meilleure dans le contexte de leur étude. Ainsi, l'utilisation d'un algorithme d'apprentissage machine, implique la sélection de caractéristiques à partir desquelles on souhaite décrire les observations. On peut les sélectionner manuellement ou en utilisant des algorithmes d'extraction comme l'analyse en

composantes principales ou l'analyse linéaire prédictive. (Teti, 2015) a utilisé les deux méthodes ci-dessus pour extraire les caractéristiques de modèles, car il utilisait une combinaison de données provenant de 7 capteurs différents. (Yin, Gao, Qiu, & Kaynak, 2017) ont quant à eux uniquement utilisé l'analyse en composantes principales pour faire une comparaison avec leur propre méthode de détection de fautes.

1.3.2 Résultats de l'utilisation d'algorithmes d'apprentissage machine

À la suite de l'observation des résultats des différentes méthodes présentées ci-dessus, il est observé qu'ils sont plus ou moins efficaces selon l'algorithme utilisé. De plus, les articles les plus récents semblent obtenir de meilleurs résultats, ce qui est potentiellement dû à l'amélioration des techniques et des outils d'apprentissage machine au cours des dernières années. (Apte, Weiss, & Grout, 1993) ont choisi de comparer 5 méthodes différentes d'apprentissage machine à l'aide de données réelles. Voici les taux d'erreur de classification qu'ils ont obtenus pour chacun des algorithmes expérimentés :

- Les arbres de décision : 41 %,
- La méthode des K plus proches voisins : 48 %,
- Les réseaux de neurones classiques : 38,5 %,
- Règle d'induction : 39 %,
- Discrimination linéaire : 40 %.

Le constat est que ces résultats de classification ne sont pas idéaux. Ces expérimentations ont quand même réussi à proposer un nouveau scénario de contrôle qualité afin de réduire les temps de contrôle. (Wang, 2013) a aussi comparé plusieurs techniques. Il a obtenu de meilleurs résultats de son côté :

- Réseaux de neurones : 90,30 %,
- Arbres de décision : 91,60 %,
- SVM : 95,8 %.

Il est tentant de vouloir comparer les résultats de ces deux études. En effet, il y a une différence significative au niveau des résultats obtenus en utilisant les mêmes techniques. Il faut donc faire ressortir que ces travaux ne sont pas liés au même domaine en ce qui concerne

la détection de défauts et le plus important facteur de différence est que les données analysées ne sont pas les mêmes. C'est un facteur très important à prendre en compte, car la qualité, le profil, le nombre de données accessibles et le type de données ont un impact important sur les résultats. L'évolution et l'amélioration des algorithmes, au cours des années, ont sûrement aussi un impact sur la qualité des résultats obtenus.

Dans l'étude de (Dib, Ribeiro, & Prates, 2018), les algorithmes de SVM's ont donné les meilleurs résultats avec 95,8 % de classifications correctes. Les auteurs ont eux aussi comparé plusieurs techniques. Néanmoins contrairement aux deux études précédentes, ils ont utilisé des données obtenues à partir de modèles numériques. Voici les meilleurs résultats de classifications correctes obtenues par technique :

- Arbres de décision : 93,98 %,
- Perceptron à plusieurs couches (réseaux de neurones) : 91,01 %,
- SVM : 92,01 %,
- Forêts aléatoires : 96,42 %,
- Classifieur naïf de Bayes : 94,12 %.

En utilisant aussi des données de simulation numérique (Askarian, Benítez, Graells, & Zarghami, 2016) ont réussi à montrer que leur méthode itérative, utilisant un classificateur naïf bayésien, permettait d'améliorer la précision du système de détection de fautes en corrigeant les labellisations erronées des opérateurs humains. La précision de détection est passée d'environ 70 % à plus de 90 % après 4 itérations. Leur expérience avec des données réelles a été moins concluante avec environ 62 % de précision due à la difficulté d'obtenir des données sur des cas de fautes réelles.

L'utilisation d'approches basées sur les réseaux de neurones donne aussi généralement de bons résultats. Par exemple (Wang, 2013) et (Dib, Ribeiro, & Prates, 2018) présentent leurs essais. Il en est de même pour les articles suivants :

- Avec leur algorithme de réseaux de convolutions (Lin, Yao, Ma, & Wang, 2018) ont réussi à augmenter la précision de détection des défauts de moulures. Le taux d'erreur de détection de leur méthode est inférieur à 4 %.

- (Lee-Ing Tong, Wei-I Lee, & Chao-Ton Su, 1997) ont utilisé les réseaux de neurones pour créer des clusters de défauts localisés au même endroit et les traiter comme un seul défaut. Cela leur a permis d'augmenter la précision des modèles de calcul de rendement utilisés jusqu'à 99 %.
- (Teti, 2015) a aussi obtenu de bons résultats avec sur l'ensemble de ses 3 cas d'étude qui portaient sur des données provenant de situations réelles. Le taux de succès était au-delà de 85 % dans tous les cas atteignant même 99 %. Ce qui est d'autant plus intéressant, c'est le fait que les 3 cas portaient sur des pièces et outils de forme et taille très différentes.

L'avantage de l'utilisation d'une approche à l'aide des réseaux de neurones est qu'on peut aussi les utiliser en apprentissage non supervisé, ce qui n'est pas le cas pour la plupart des autres algorithmes qui requièrent des données labellisées. Néanmoins, les SVM offrent aussi de bonnes possibilités et seront considérés aussi dans le cadre de ce projet. Il faut notre ici aussi que les algorithmes d'apprentissage machine peuvent être aussi intégrés directement dans les logiciels et qu'il existe une multitude d'outils permettant d'expérimenter ces différents algorithmes. (Dib, Ribeiro, & Prates, 2018) ont utilisé la librairie « scikit-learn » programmée en langage Python. Ce sont les seuls auteurs qui ont présenté les outils et librairies utilisés. Pour notre étude il sera aussi possible d'utiliser la librairie « TensorFlow » de Google qui est assez populaire actuellement pour faire de l'apprentissage machine. De plus, elle est disponible non seulement en Python, mais aussi dans d'autres langages comme Java, C++ ou encore JavaScript.

1.4 CONCLUSION

Cette revue de la littérature a permis de présenter que l'utilisation d'algorithmes d'apprentissage pour prédire où détecter les défauts sur les pièces manufacturées du domaine de l'industrie automobile est débutée. Il existe en effet quelques travaux de recherche portant sur ce sujet et, ce qui est intéressant, c'est que la plupart des défauts étudiés peuvent se retrouver sur des pièces automobiles. Les résultats de prédiction obtenus à l'aide de ces algorithmes sont plus ou moins encourageants compte tenu de la technique utilisée par les

chercheurs. L'équipe de (Askarian, Benítez, Graells, & Zarghami, 2016) ont obtenu de bons résultats avec des données provenant de modèles numériques, mais les résultats obtenus avec des données provenant de l'industrie étaient toujours mitigés. Cette revue a aussi permis d'identifier quelques outils à utiliser pour expérimenter des algorithmes d'apprentissage machine pour ce projet.

CHAPITRE 2

ANALYSE ET TRAITEMENT DES DONNÉES

2.1 Analyse des données

Pour faire de l'apprentissage machine, il faut avoir accès à des données qui représentent le problème à résoudre. La première étape est de comprendre ces données et la façon dont il faut les organiser pour extraire le maximum de caractéristiques pertinentes pour les algorithmes d'apprentissage.

Le but de cette phase d'analyse est de comprendre les mesures faites par les machines pour trouver une relation entre les données qui permet de leur associer une étiquette : **non conforme** et **conforme**.

Cette phase permettra aussi de détecter les problèmes liés aux fichiers et qui pourraient empêcher l'accès aux données et leur correction.

2.1.1 Présentation des données

Dans le cadre de ce projet, les données proviennent de machines à mesurer tridimensionnelles. Chaque pièce fabriquée passe un test de vérification de ses dimensions et les valeurs mesurées par la machine sont enregistrées dans un fichier CSV. Un fichier CSV représentant une pièce est présenté en Annexe 1.

Il y a en tout plus de 9433 fichiers de mesures faites par les machines. Chaque fichier représente les caractéristiques mesurées par une CMM pour une pièce qui a fait l'objet d'un contrôle.

Les fichiers concernent en tout 97 pièces différentes qui sont présentées en Annexe 2.

Pour chaque caractéristique à vérifier, la machine peut effectuer une ou plusieurs mesures. Voici ci-dessous, les informations associées à chaque caractéristique mesurée et qu'on peut retrouver dans les fichiers :

- **Feat. Type** : le type de la caractéristique à vérifier. Cela peut être un contrôle de caractéristique géométrique ou une caractéristique dimensionnelle.

  Suivant les pièces, les CMM donnent des informations de mesures sur ces deux types de caractéristiques.

  - Les caractéristiques géométriques mesurées peuvent concerner la forme de la pièce, l'orientation de certains de ses composants ou encore leur position.
  - Les caractéristiques dimensionnelles sont relatives aux tailles et proportions des pièces et leurs composants : Les distances et angles entre composants ou la taille des arcs.

  Le tableau ci-dessous contient les mesures que l'on peut retrouver dans les fichiers pour chaque type de caractéristique. Il y a en tout 17 sortes de mesures qui peuvent être réalisées sur les pièces à disposition pour le projet.

Tableau 2.1 : Liste des différents types de caractéristiques contrôlées par les CMM

Mesures dimensionnelles	Caractéristiques géométriques
Slot	Cone
Section	Sphere
Distance	Point
	Perpend.
	Profile of Surface
	Pos. Tol.
	Paral.
	Profile of Line
	Coaxiality
	Circle
	Flatness
	Surf. Pnt Angle

- **Feat. Name** : le nom de la caractéristique

- **Value** : la mesure associée à la caractéristique. Pour un Feat. Name, on peut avoir plusieurs mesures à vérifier
- **Actual** : la valeur mesurée par la machine. C'est un nombre réel positif ou négatif.
- **Nominal** : la valeur attendue. C'est un nombre réel.
- **Dev.** : la déviation entre la valeur attendue et la valeur mesurée. C'est un nombre réel donné par la formule : $\text{Dev.} = \text{Actual} - \text{Nominal}$.
- **Tol-** : C'est la tolérance minimale autorisée pour la mesure. C'est un nombre réel.
- **Tol+** : C'est la tolérance maximale autorisée pour la mesure. C'est un nombre réel.
- Si $\text{Actual} < \text{Nominal} - \text{Tol}_-$ alors, la mesure sera considérée hors tolérance.
- Si $\text{Actual} > \text{Nominal} + \text{Tol}_+$ alors, la mesure sera considérée hors tolérance.
- Si $\text{Nominal} - \text{Tol}_{-\leq \text{Actual} \leq \text{Nominal}+\text{Tol}_+}$ alors, la mesure ne sera pas considérée hors tolérance.
- **Out of Tol.** : la déviation par rapport à l'intervalle de tolérance
- Si $\text{Actual} < \text{Nominal} - \text{Tol}_-$, alors $\text{OutofTol.} = \text{Dev.} - \text{Tol}_-$
- Si $\text{Actual} > \text{Nominal} + \text{Tol}_+$, alors $\text{OutofTol.} = \text{Dev.} - \text{Tol}_+$
- Si $\text{Nominal} - \text{Tol}_{-\leq \text{Actual} \leq \text{Nominal}+\text{Tol}_+}$, alors le champ est vide.
- **Comment** : C'est un commentaire relatif à la mesure. Il est facultatif, car il n'y en a pas pour toutes les mesures.

2.1.2 Problèmes détectés

L'analyse des données a permis de voir dans quelle mesure il est possible de travailler avec les données. Le projet comporte en effet une partie d'implémentation qui nécessite l'accès aux données. L'objectif est donc de réussir à charger les fichiers et d'essayer de détecter les problèmes.

Il faut d'abord vérifier que les fichiers ne sont pas corrompus et qu'ils sont tous écrits dans le format CSV. Le langage de programmation utilisé est Python avec les librairies « Pandas » et « Numpy » qui permet de lire les fichiers CSV et de les transformer en matrices.
Pour cette première étape, la présence de fichiers vides a été détectée ainsi que des fichiers mal formatés en CSV. Pour ce dernier cas, des modifications manuelles ont été faites puisque les fichiers concernés sont peu nombreux.

Il faut aussi extraire une ou plusieurs parties des données avant de faire l'apprentissage et les tests. Il faut donc analyser le contenu des fichiers et supprimer les informations non nécessaires. Ainsi, plusieurs mesures enregistrées sont utilisées dans le calcul des valeurs d'autres mesures et ne comportent aucune information sur l'intervalle de tolérance à respecter. Dans ces cas-là, les lignes correspondantes sont tout simplement supprimées avant l'apprentissage et les tests, car considérées comme du bruit.

Lorsque les mesures ne sont pas hors tolérance, les champs de la colonne Out of tol. sont vides. À l'extraction, ces champs doivent être tous mis à 0,0 pour qu'ils soient utilisés.

Comme le problème à résoudre sera traité comme un problème de classification, il faut aussi vérifier le nombre de données pour les classes. Il y a exactement deux classes à trouver : une mesure hors tolérance et une bonne mesure.

- Une mesure est *non conforme* lorsqu'elle est hors tolérance.
- Une mesure est *conforme* lorsqu'elle est dans l'intervalle de tolérance définie.

Le fait de travailler par mesure va permettre ensuite d'introduire une généralisation permettant de traiter un fichier de mesures d'une pièce. Dans ce cas-ci, une pièce sera catégorisée défectueuse si au moins une mesure associée est hors tolérance.
Dans le cadre de ce travail d'analyse, il est intéressant de regarder dans les données quelles sont les caractéristiques qui sont le plus souvent hors tolérance de même que les pièces qui sont jugées le plus souvent défectueuses après les contrôles.

La caractéristique mesurée qui est le plus souvent hors tolérance est « Distance » avec 22 866 lignes mesurées hors des intervalles de tolérances. La pièce la plus affectée est GM HF FWD Gen II High Hood 31xx avec 3172 erreurs associées.

Dans le tableau 2 ci-dessous, chaque type de caractéristique mesurée est associée au nombre de fois où la mesure de contrôle était hors tolérance pour l'ensemble des pièces de la base de données.

Tableau 2.2 : Nombre de mesures hors tolérance par type de caractéristique

Caractéristique	**Nombre de mesures hors tolérance**
Distance	22 866
Circle	7934
Pos. Tol.	6093
Profile of Line	1045
Flatness	875
Paral.	240
Section	157
Coaxiality	36
Perpend.	21
Cone	15
Surf. Pnt	8
Sphere	3
Slot	2
Point	1
Profile of Surface	0
Angle	0

Pour faire la classification des données, il est important d'avoir une bonne répartition des données entre les classes à prédire. Or pour ce projet, il y a une très grande disparité entre les données de la classe « *Non conforme* » et la classe « *Conforme* ». Comme le présente le

tableau 2.3, il y a environ 40 000 lignes de fichiers de la classe « *Conforme* » alors qu'il y a en tout plus de 800 000 lignes à traiter.

Cette disparité sera prise en compte lors de la phase d'apprentissage. En effet, comme les algorithmes apprennent à partir d'exemples, la classe « *Conforme* » sera favorisée si cet aspect de la base est négligé.

Tableau 2.3 : Répartition des classes sur l'ensemble de la base de données

Classe	**Nombre de données associées**
1 (Non conforme)	39 303
-1 (Conforme)	830 797

2.2 Prétraitement et labellisation des données

L'objectif de cette partie de l'analyse des données est de normaliser et de labelliser les données.

Les mesures faites par les machines n'ont pas les mêmes dimensions, les mêmes unités, ou encore les mêmes grandeurs : il peut en effet avoir des mesures d'angle, de distance, de position, etc. dans un même fichier.

Même si le problème était traité par type de mesure, les grandeurs seront différentes en fonction de la pièce contrôlée. Il faut donc prendre en compte ces facteurs qui ont une forte influence sur les résultats d'apprentissage et de tests avant de continuer.

Pour harmoniser les données, la norme L_2 sera utilisée pour normaliser chaque ligne d'un fichier. Cela va permettre de remettre les données à la même échelle, sans altérer leur contenu. Cette étape va permettre d'obtenir un vecteur d'entrée de norme égale à 1. Ce traitement est fait avant la labellisation des données. Si on considère X comme le vecteur représentant une mesure de contrôle, le vecteur $X_{\text{normalized}}$ est donné par la formule 1.1 ci-dessous.

$$X_{normalized} = \frac{X}{|X|} \text{avec} |X| = \text{norme}L_2(X) = \sqrt{\left(\sum_{i=1}^{n} X_i^2\right)} \quad (2.1)$$

Avant de labelliser les données, il faut déterminer quelles sont les caractéristiques qui seront utilisées comme entrées de l'algorithme d'apprentissage.

Pour choisir ces caractéristiques, les colonnes des fichiers seront utilisées. Elles sont exactement les mêmes pour tous les fichiers.

Les colonnes « Feat. Type », « Feat. Name » et « Comment », qui ne contiennent pas de valeurs numériques sont éliminées. Les colonnes restantes sont : « Actual », « Nominal », « Dev. », « Tol- », « Tol+ » et « Out of Tol. ». Le tableau 2.4 présente un récapitulatif des caractéristiques qui seront prises en compte dans les algorithmes.

Parmi ces dernières, les caractéristiques choisies pour évaluer un contrôle sont : *« Actual »*, *« Nominal »*, *« Dev. »*, *« Tol- »*, *« Tol+ »*. La colonne « Out of Tol. » permettra de définir la sortie de l'algorithme d'apprentissage.

Pour faire de la classification, il faut définir les classes qui doivent être prédites. Dans le cadre du projet, voici les classes qui seront utilisées :

- *Non Conforme* (1) : pour une pièce dont au moins une mesure est jugée hors tolérance par la machine. Cela veut dire que la colonne « Out of Tol. » a une valeur différente de 0.
- *Conforme* (-1) : pour une pièce dont aucune mesure n'est hors tolérance. Cela veut dire que la colonne « Out of Tol. » est vide ou égale à 0.

Utiliser (-1) ou (1) va permettre de classer les données plus facilement à partir de la sortie de l'algorithme. Les valeurs prédites qui sont négatives seront associées à la classe « *Conforme* » et les valeurs prédites qui sont positives seront associées à la classe « *Non conforme* ».

Comme ces classes ne sont pas directement présentes dans les fichiers, il a fallu modifier les données avant les phases d'apprentissage et de tests. C'est la colonne « Out of Tol. » qui est

utilisée pour réaliser cette étape ici. Une colonne ***Target*** a été ajoutée dans la matrice extraite de chaque fichier et son contenu est défini avec la méthode ci-dessous :

- Si $\text{Out of Tol.} \neq 0$, alors $\text{Target} = 1$
- Si $\text{Out of Tol.} = 0$, alors $\text{Target} = -1$

Tableau 2.4 : Caractéristiques sélectionnées en fonction de leur domaine de définition

Données pour une mesure	**Domaine de définition**	**Caractéristiques sélectionnées**
Feat. Type	Chaîne de caractère	Rejeté
Feat. Name	Chaîne de caractère	Rejeté
Value	Chaîne de caractère	Rejeté
Actual	Valeur réelle	Sélectionné
Nominal	Valeur réelle	Sélectionné
Dev.	Valeur réelle	Sélectionné
Tol-	Valeur réelle	Sélectionné
Tol+	Valeur réelle	Sélectionné
Out of Tol.	Valeur réelle	Sélectionné
Comment	Chaîne de caractère	Rejeté

2.3 Choix des algorithmes

Il existe de nombreux algorithmes d'apprentissage machine qui peuvent être utilisés dans le cadre du projet. Néanmoins, l'approche choisie est de travailler uniquement avec les algorithmes de réseaux de neurones. Ce sont les plus utilisés dans l'industrie actuellement pour faire l'apprentissage profond. La revue de littérature a d'ailleurs révélé que les travaux utilisant ces méthodes avaient généralement de bons résultats.

Ce sont les algorithmes de la famille des réseaux de neurones qui seront explorés. Dans la littérature, ils permettent d'obtenir de très bons résultats sur les problèmes d'apprentissage profond. Les deux principaux algorithmes qui seront regardés sont : le Perceptron multicouche et les réseaux de neurones récurrents.

Le Perceptron multicouche va nous permettre de comprendre dans quelle mesure les réseaux de neurones sont adaptés à notre problème. Ce sera notre algorithme de base et il fera l'objet de plusieurs tests pour trouver les meilleurs paramètres.

Les réseaux de neurones récurrents sont envisagés pour explorer la possibilité de traiter chaque fichier comme une entrée du réseau de neurones. Comme les fichiers sont de tailles différentes, ce sont les algorithmes les plus adaptés. Ils sont d'ailleurs utilisés pour la classification de chaînes de caractères de tailles différentes en utilisant les lettres.

Le troisième algorithme qui sera expérimenté est l'algorithme des forêts aléatoires. C'est un algorithme qui a aussi fait ses preuves dans la littérature pour la prédiction de défauts et semble être une très bonne approche dans le cadre de ce projet.

CHAPITRE 3

APPRENTISSAGE MACHINE

3.1 Configuration de l'environnement de développement

Le développement se fera avec le logiciel Visual Studio 2017 qui permet de programmer en Python.

Le choix a été fait de travailler principalement avec deux librairies :

- Scikit-Learn qui est la librairie utilisée dans l'étude (Dib, Ribeiro, & Prates, 2018). Cette librairie propose une gamme assez importante de fonctions et d'algorithmes communément utilisés dans le domaine de l'apprentissage machine. Une documentation est fournie avec plusieurs exemples pour comprendre les paramètres utiles de chaque méthode. C'est une librairie très accessible surtout pour les débutants.
- Pytorch est au même titre que Tensorflow une des librairies les plus utilisées pour faire de l'apprentissage profond. Il propose une gamme importante d'algorithmes et de fonctions pour implémenter divers types de réseaux de neurones. Il y a une documentation significative sur l'ensemble des méthodes, mais demande un petit temps d'adaptation pour maîtriser les fonctionnalités de base.

Scikit sera utilisé pour les forêts aléatoires et Pytorch pour les algorithmes de réseaux de neurones.

3.2 Évaluation de la performance

Les mesures utilisées pour évaluer la performance des algorithmes de classification utilisés proviennent pour la plupart de la littérature et des standards dans le domaine de l'apprentissage machine.

- L'exactitude ***E*** va nous permettre de savoir à quel point l'algorithme est performant

$$\mathrm{E} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FN} + \mathrm{FP}} \tag{3.1}$$

- La précision ***P*** va nous permettre de savoir dans quelle mesure l'algorithme fait de bonnes prédictions de présence de défauts

$$\mathrm{P} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{3.2}$$

- Le rappel ***R*** va nous donner des informations sur la sensibilité de l'algorithme à prédire les défauts correctement

$$\mathrm{R} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{3.3}$$

- Le ***score F1*** qui sera très intéressant puisqu'il y a très peu de cas où les mesures sont hors tolérance (classe = 1). Cette mesure de performance sera utile pour savoir si l'algorithme prédit toujours la classe 1 ou si au contraire, il sait reconnaître les mesures hors tolérance.

$$\mathrm{Score\ F1} = \frac{2 \cdot \mathrm{TP}}{2 \cdot \mathrm{TP} + \mathrm{FP} + \mathrm{FN}} \tag{3.4}$$

TP = nombre de cas positifs correctement classés
TN = nombre de cas négatifs classés négatifs
FP = nombre de cas négatifs classés positifs
FN = nombre de cas positifs classés négatifs

Ces mesures sont les plus adaptées pour les problèmes de classification. Les valeurs TP, TN, FP et FN seront présentées dans une matrice de confusion à partir de laquelle les valeurs de performance seront calculées.

Ces mesures seront par ailleurs aussi utilisées pour évaluer comment l'algorithme évolue lors de l'apprentissage et détecter les cas de surapprentissage ou de sous-apprentissage.

3.3 Tâches d'apprentissage

Dans un premier temps, il faut extraire les données de la base de données puis les normaliser avec la norme L2. Comme indiqué dans la phase de l'analyse des données, cela permettra de réduire l'impact de la différence d'échelle entre les mesures.

Ensuite, les données seront labellisées -1 ou 1 selon qu'elles sont jugées conformes ou non conformes avant de les répartir aléatoirement dans les bases d'apprentissage et de tests. La base d'apprentissage contiendra 70 % des données et la base de validation 30 %.

Tableau 3.1 : Répartition des données dans les deux bases

Base	**Taille**	**Classe** ***Conforme***	**Classe** ***Non conforme***
Apprentissage	609070	27536 (4.5 %)	581534 (95.5 %)
Validation	261030	11767 (4.5 %)	249263 (95.5 %)

La répartition des données dans les deux bases est exactement la même.

Il y a un autre jeu de données plus récentes qui sera utilisé pour faire les tests une fois les modèles et les paramètres validés. La base de validation va aussi nous permettre de détecter les problèmes relatifs à l'apprentissage comme le surapprentissage ou le sous-apprentissage.

Une fois les données réparties dans deux groupes distincts, l'entraînement sera fait sur la base d'apprentissage puis évalué sur la base de validation. Suivant les résultats obtenus, les paramètres peuvent être modifiés pour refaire un nouvel entraînement ou le modèle sera directement évalué sur la base de tests.

La figure ci-dessous représente un schéma de l'organisation présentée ci-dessus en ce qui concerne les tâches d'apprentissage qui seront réalisées pour mener à bien l'expérimentation.

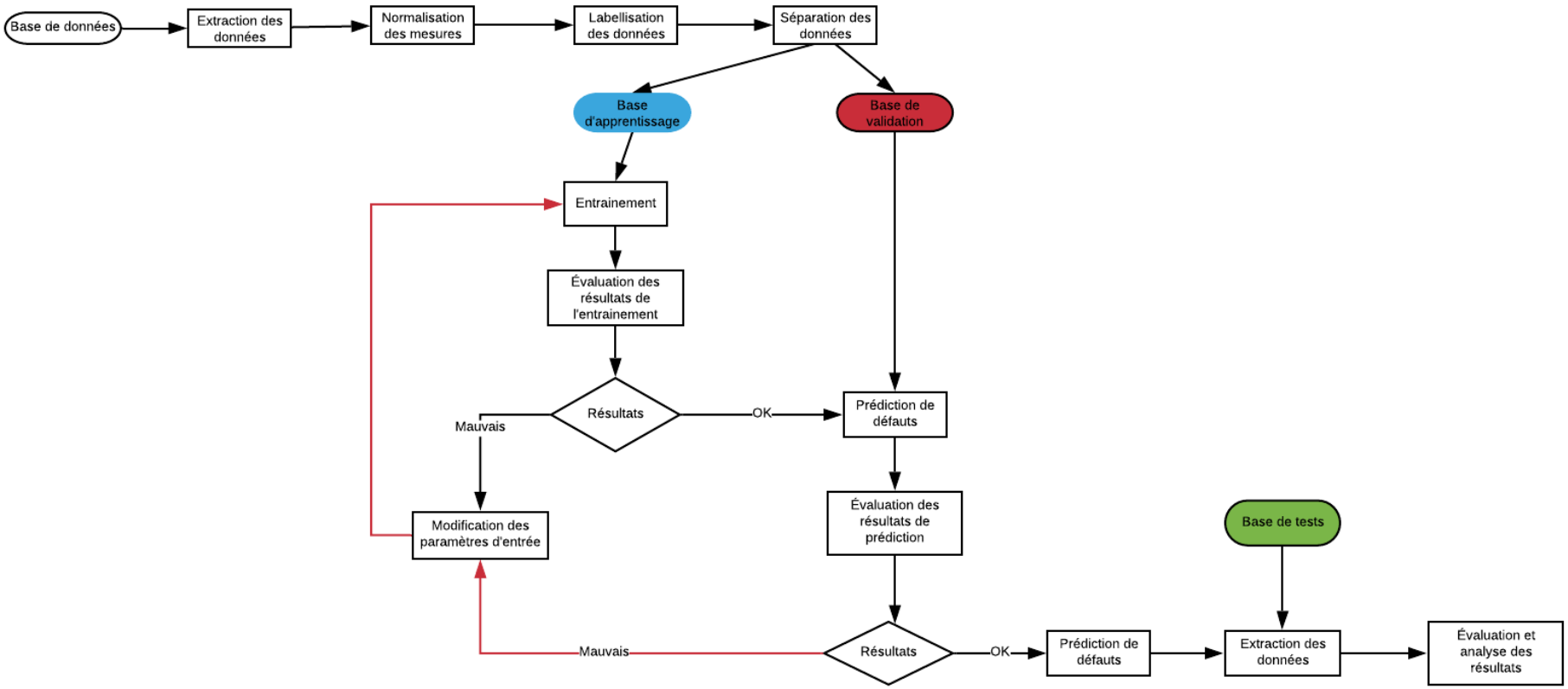

Figure 3.1 : Organisation des tâches d'apprentissage

3.4 Taxonomies des algorithmes sélectionnés

Dans cette partie, un rappel rapide sur la taxonomie des algorithmes qui seront utilisés sera fait. Ceci dans le but de mieux comprendre l'importance leurs paramètres de même que la façon dont les données du projet seront utilisées.

3.4.1 Perceptron multicouche

Le Perceptron est le modèle de base d'un réseau de neurones artificiels. L'objectif est de trouver une sortie (classification ou régression) à partir d'une entrée. Les neurones de la couche de sortie sont activés par une fonction qui permet d'estimer la valeur de la sortie ou de faire une classification.

Dans l'algorithme du Perceptron multicouche, on utilise une suite de perceptrons. La sortie du Perceptron de la couche précédente est l'entrée de la couche suivante et ainsi de suite.

La rétro propagation du gradient est utilisée pour faire la mise à jour des poids dans le réseau. Pour cela il faut utiliser une fonction de coût permettant d'évaluer la prédiction faite par le réseau par rapport à la sortie attendue. Comme c'est un problème de classification, la fonction choisie est la fonction Cross-Entropy définie par la formule 3.7.

Dans le cadre de ce projet, l'entrée est un vecteur de taille 4 avec les caractéristiques suivantes : $X = [x_1, x_2, x_3, x_4] = [Actual, Nominal, Tol_+, Tol_-]$

Les fonctions d'activation choisies sont la tangente hyperbolique et le Softsign. Les définitions de leurs fonctions sont données par les équations 3.5 et 3.6.

$$tanh(\mathrm{z}) = \frac{\mathrm{e}^{\mathrm{z}} - \mathrm{e}^{-\mathrm{z}}}{\mathrm{e}^{\mathrm{z}} + \mathrm{e}^{-\mathrm{z}}} \text{ avec } \mathrm{f(z)} \in (-1,1) \tag{3.5}$$

$$softsign(\mathrm{z}) = \frac{\mathrm{z}}{1 + |\mathrm{z}|} \text{ avec } \mathrm{f(z)} \in (-1,1) \tag{3.6}$$

$$cross - entropy(\mathrm{z},\mathrm{y}) = -[y\ \log(z) + (1-y)\log(1-z)] \qquad (3.7)$$

Les deux fonctions d'activation permettent d'obtenir des résultats en sortie dans l'intervalle (-1, 1) donc elles sont adaptées, car les sorties à prédire sont -1 ou 1.

Lors des expérimentations, il faut donc faire varier les fonctions d'activation, le nombre de neurones dans les couches cachées ainsi que le nombre de ces dernières. Les tests pour le Perceptron multicouche seront faits avec la librairie Pytorch. Toutes les fonctions présentées sont implémentées dans la librairie.

3.4.2 Forêts aléatoires

Les forêts aléatoires utilisent des arbres de décision construits en sélectionnant de manière aléatoire les caractéristiques des vecteurs d'entrées. Plusieurs chemins sont explorés pour permettre d'obtenir une classification finale. C'est une combinaison des décisions de tous les arbres qui constituent la forêt.
Cette méthode de classification permet d'éviter de tomber dans les problèmes de surapprentissage même si elle peut s'avérer assez lente lors de la prédiction, car il faut passer par tous les arbres de la forêt. Un autre avantage intéressant des forêts aléatoires est qu'elles permettent de réduire la variance des résultats et simple à mettre en œuvre.

Dans le cadre de ce projet, c'est une méthode qui peut s'avérer très efficace, car intuitivement, c'est une approche de prise de décision qui peut être utilisée par l'humain pour déterminer si une pièce est défectueuse ou non.

Plusieurs paramètres seront testés notamment le nombre d'arbres dans la forêt ainsi que leur profondeur. L'importance des caractéristiques dans la prise de décision peut être aussi déterminée à la fin de l'entraînement. La librairie Scikit-learn offre toutes les fonctions nécessaires pour réaliser l'entraînement et extraire les métriques nécessaires.

CHAPITRE 4

RÉSULTATS

Dans cette partie, les résultats des algorithmes utilisés seront présentés et analysés. Ces résultats correspondent aux résultats obtenus sur la base de tests.

4.1 Base de tests

La base de tests est un fichier Excel contenant les mesures de contrôle issues des CMM pour plusieurs pièces différentes. Ce sont des mesures effectuées en 2019. Le fichier a au total 15656 lignes de mesures. Les pièces contrôlées sont toutes des pièces contrôlées présentes dans les bases de tests et de validation.

Tableau 4.1 : Pièces contrôlées

Listes des pièces contrôlées
Corvette LT1
D-35
Ford Transmission Pan 10R80 Moule 2
GM HF FWD GEN II C1xx
Mercury Marine Great White V8
Mercury Marine TigerShark
Ford Nano MY2018
GM HF FWD GEN II 31xx
Mercury Marine
V10 6.8L Triton
D-37
Ford Transmission Pan 10R80 Transit
GM HF FWD GEN II E2xx
D-33 RWD
Denso 507F
Ford Nano 2.7L

GMI 700 Moule 3
Ford 2.3L Maverick RWD
Ford 2L Maverick FWD
Mercury Marine Great White V6
Denso 640F
GM HF RWD Gen II
I4 Moule 2
Ford Transmission Pan 10R80
TBA 2 V

Tableau 4.2 : Répartition des classes

Classe	**Nombre de mesures**
Conforme	9388 (96.5 %)
Non Conforme	344 (3.5 %)

Ci-dessus, la répartition des classes dans la base de tests dans la base de tests ainsi que le nombre de pièces concernées. 5922 lignes ont été supprimées, car elles correspondent aux lignes considérées comme du bruit lors de la phase d'analyse. Le tableau 4.2 montre bien que la répartition des données est quasiment la même que dans la base de données qui a servi pour l'entraînement et la validation.

4.1.1 Résultats du Perceptron multicouche

Les résultats obtenus sont en apparence bons parce que ce sont des valeurs au-dessus de 90 % pour les mesures de performance, mais comme le montre la figure 4.1 les mesures de la classe « non conforme » sont mal classées par l'algorithme. Seule la classe « conforme » est correctement apprise.

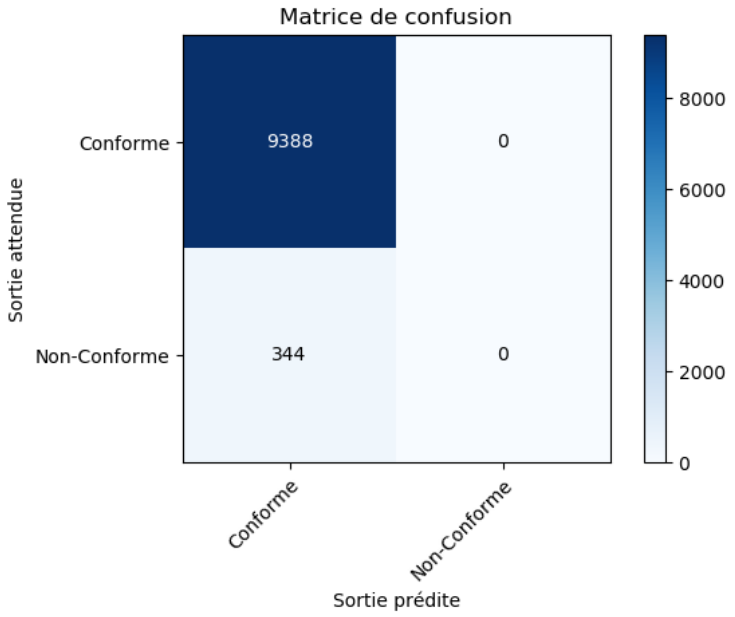

Figure 4.1 : Matrice de confusion de la base de tests avec Softsign comme fonction d'activation

L'algorithme prédit sur la base de tests correctement les mesures qui ne sont pas hors tolérance. Il y a 100% de reconnaissance sur les mesures de cette classe et 0% sur les mesures de la classe non conforme. Il y a donc un problème de surapprentissage de la classe « conforme » dû au fait qu'il y a 95% de données provenant de cette classe dans la base.

Le meilleur score obtenu sur les mesures de la classe « non conforme » sans modifier la composition de la base d'apprentissage est de 60%. Mais il a fallu pour cela faire un entrainement environ 100000 époques. La configuration du perceptron multicouche qui a permis d'obtenir ce résultat est la suivante :

Coefficient d'apprentissage	Fonction d'activation	Nombre de couches cachées	Nombre de neurones de la couche cachée
0.003	Tanh ou Softsign	1	6

En ce qui concerne la durée de l'apprentissage, plus il y a de neurones dans une couche, plus l'algorithme est lent. C'est le même constat pour le nombre de couches cachées. Plus il y en a, plus il faudra de temps à l'algorithme pour terminer.

4.1.2 Résultats des forêts aléatoires

Dans l'ensemble, les résultats obtenus avec les forêts aléatoires sembles très bons. En effet même avec un seul arbre dans la forêt, les classes des mesures sont bien prédites avec un score de prédiction correcte bien au-dessus de 99 %. Sur la base de validation, les résultats sont similaires. On peut constater que déjà avec 5 arbres on a de très bons résultats et un temps d'apprentissage peu élevé. Au-delà de 25 arbres, les résultats ne changent plus, mais le temps d'apprentissage augmente nettement. Les figures 4.1 à 4.4 présentent les matrices de confusion des résultats obtenus en faisant varier le nombre d'arbres dans les forêts aléatoires.

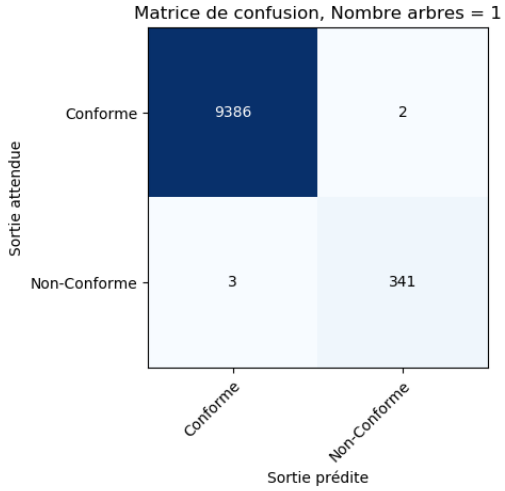

Figure 4.2 : Forêts aléatoires avec 1 arbre

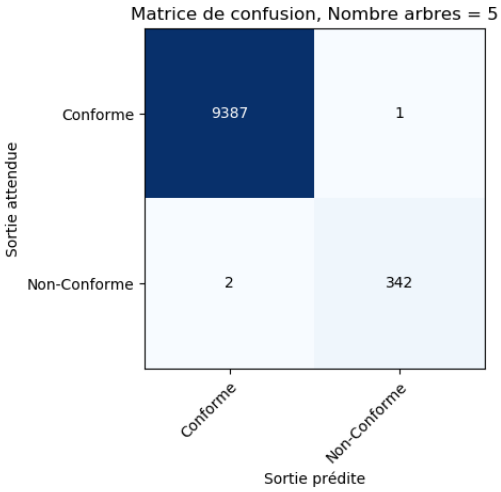

Figure 4.3 : Forêts aléatoires avec 5 arbres

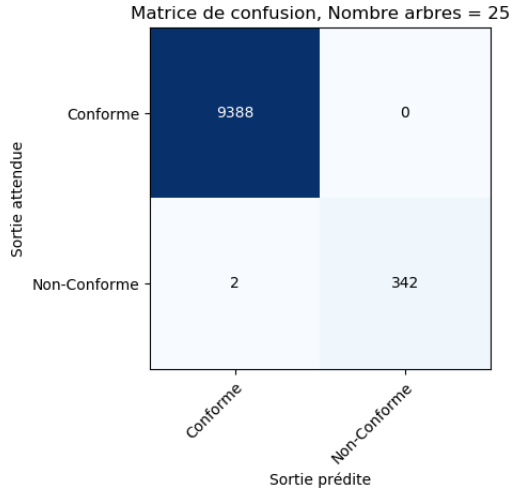

Figure 4.4 : Forêts aléatoires avec 25 arbres

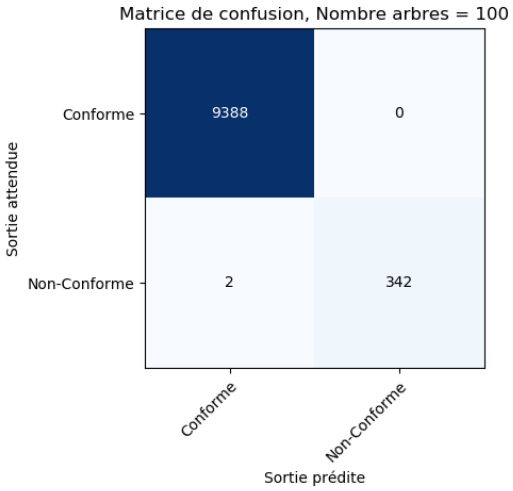

Figure 4.5 : Forêts aléatoires avec 100 arbres

Tableau 4.3 : résultats sur la base de tests

Nombre d'arbres	**Exactitude**	**Précision**	**Rappel**	**Score F1**	**Temps d'apprentissage**
1	99.95 %	99.95 %	99.95 %	99.95 %	1.74 s
5	99.97 %	99.97 %	99.97 %	99.97 %	8.57 s
25	99.98 %	99.98 %	99.98 %	99.98 %	46.26 s
100	99.98 %	99.98 %	99.98 %	99.98 %	186.61 s

Le tableau 4.5 présente les valeurs de l'importance de chaque caractéristique sélectionnée pour faire la classification. La caractéristique « Actual » est celle qui a le plus d'influence, quel que soit le nombre d'arbres dans la forêt suivie par « Tol+ ». Cela veut dire que les

mesures « non conformes » sont le plus souvent hors de l'intervalle de tolérance parce qu'ils sont plus grands que « Tol+ ». Pour finir, les deux autres caractéristiques ont quasiment la même importance.

Tableau 4.4 : importance des caractéristiques dans la prise de décision des arbres

	Importance			
Nombre d'arbres	**Actual**	**Nominal**	**Tol+**	**Tol-**
1	0.35	0.24	0.19	0.23
5	0.37	0.19	0.26	0.17
25	0.4	0.18	0.25	0.17
100	0.39	0.2	0.24	0.17

4.2 Discussion

Les réseaux de neurones fonctionnent très bien dans les articles de la littérature. Néanmoins dans ce projet, le peu de caractéristiques en entrée a sûrement été handicapant lors de l'utilisation de ces méthodes. Il faut avoir un certain nombre de paramètres dans un réseau de neurones pour espérer avoir de bons résultats.

Dans le cadre de cette étude, le réseau de perceptrons a donc obtenu de moins bons résultats malgré une quantité de données importante. Même en augmentant drastiquement le nombre de neurones dans les couches cachées, les résultats ne sont pas plus intéressants et le temps d'apprentissage augmente. L'algorithme ne savait quasiment prédire que des mesures conformes, sûrement du fait de la répartition des classes qui est largement en faveur de la classe « conforme ». Il y a très peu de mesures de la classe « non conforme » et le réseau de Perceptron a du mal à correctement apprendre à reconnaître ce type de mesures.

Dans les couches cachées, les fonctions d'activation sont des fonctions linéaires. Seule la couche de sortie a une fonction d'activation non linéaire et c'est aussi une raison d'échec qui pourrait être investiguée dans les futurs travaux. Pour espérer avoir de meilleurs résultats avec les réseaux de neurones, il faudrait utiliser une autre combinaison de facteurs pour faire l'apprentissage et vérifier si les résultats sont meilleurs.

L’algorithme est aussi très lent et il faut avoir un ordinateur assez puissant pour obtenir rapidement des résultats.

Les forêts aléatoires permettent d’obtenir de bons scores de prédiction sur les deux classes, malgré la disparité de représentation entre elles dans la base d’apprentissage, même avec un arbre.

Il faut dire que les arbres prennent leurs décisions avec un système de vote qui implique l’ensemble des arbres qui ont été utilisés. De plus en fonction de la profondeur de l’arbre le processus de prise de décision se complexifie permettant à l’arbre d’explorer plusieurs chemins lors de l’apprentissage. La profondeur des arbres est une étude à effectuer car elle n’a pas été prise en compte.

Plus il y a d’arbres dans la forêt plus l’apprentissage est long. Il faut donc trouver le juste compromis qui va permettre d’entraîner dans un temps acceptable. Si l’objectif est d’obtenir des résultats en temps réel et de favoriser l’apprentissage continu, il est très important de bien choisir et calibrer l’algorithme qui sera utilisé.

CONCLUSION

Cette recherche avait pour but de m'initier à l'utilisation des algorithmes d'apprentissage machine. Avant de commencer l'expérimentation proprement dite, la première étape de revue bibliographique a permis de trouver des travaux sur le domaine de la prédiction de défauts en industrie puis de choisir les algorithmes avec lesquels travailler en se basant sur les résultats obtenus. Les deux algorithmes suivants ont été sélectionnés :

- Les réseaux de neurones;
- Les forêts aléatoires.

Ces deux méthodes ont été choisies pour faire une comparaison entre une méthode classique et une méthode assez répandue dans le domaine de l'apprentissage profond. Pour effectuer le travail, deux librairies principales ont été utilisées pour la programmation en langage python : Pytorch pour les réseaux de neurones et Scikit-learn pour les forêts aléatoires.

Le travail d'expérimentation a débuté par une analyse qui a permis de mieux comprendre les données et corriger quelques problèmes rencontrés. Cela a permis de retenir des caractéristiques utilisées comme entrées des différents algorithmes. Les données ont ensuite été normalisées puis labellisées en fonction de la présence ou non d'une valeur dans la colonne « Out of Tol. » des fichiers.

Avec les caractéristiques utilisées, l'algorithme de forêts aléatoires semble prédire des défauts mais en fait il identifie simplement que Tol - < Valeur actuelle < Tol +. C'est pour cela que la précision résultante est aussi élevée. En enlevant tous les autres paramètres et ne conservant que la valeur actuelle et les tolérances, on aurait obtenu 100% de précision. La seule raison pour laquelle nous n'avons pas obtenu 100% c'est à cause du bruit statistique introduit par tous les autres paramètres. Ainsi cet essai démontre que les caractéristiques choisies ne permettent pas de prédire les défauts des pièces manufacturées. La performance de l'algorithme de réseau de perceptrons est aussi due au même phénomène et au fait qu'il y a une mauvaise répartition des données entre les classes et aussi les paramètres exploités.

RECOMMANDATIONS

Étant donné que les caractéristiques essayées n'ont pas permis d'obtenir des résultats concluants, à l'avenir, pour tester ces algorithmes d'apprentissage machine, il faudra enlever la valeur actuelle dans les prochains essais et ensuite effectuer l'encodage pour considérer les attributs en chaînes de caractères. Il est prévisible qu'avec cette approche la précision devrait chuter mais représenter plus fidèlement la réalité de ce jeu de données spécifique.

Les bases de données utilisées et les codes sources sont disponibles sur la page GitHub ci-contre : https://github.com/persideETS/PFE. Cela permettra de répéter l'expérience de cette recherche et aussi de tenter d'améliorer les résultats obtenus dans ce projet. Les références [2], [7], [8] et [9] ont été utilisées pour le code source.

ANNEXE I
EXEMPLES DE FICHIERS DE CONTRÔLE

Exemple de fichier de contrôle pour une pièce non défectueuse

Feat. Type	Feat. Name	Value	Actual	Nominal	Dev.	Tol-	Tol+	Out of Tol.	Comment
Text/Value	VAR_PROGRAM	VAL							N:\Metrolog XG\Programmes\Corvette LT1\Corvette LT1 Free State Boss Height (Module 44). gm2'
Text/Value	VAR_EVENEMENT	VAL							Smed '
Text/Value	VAR_COMMENTAIRES	VAL							328A'
Flatness	FLAT_DATUM_A	FLTN	0.403	0.000	0.403	0.000	2,000		PL_DATUM_A
Flatness	FLAT_DATUM_B	FLTN	0.384	0.000	0.384	0.000	2,000		PL_DATUM_B
Text/Value	H_PLASTIC_1	VAL	145,506,669	145,620,000	-0.113331	-0.375000	0.375000		
Text/Value	H_PLASTIC_2	VAL	145,461,036	145,620,000	-0.158964	-0.375000	0.375000		
Text/Value	H_PLASTIC_3	VAL	145,576,504	145,620,000	-0.043496	-0.375000	0.375000		
Text/Value	H_PLASTIC_4	VAL	145,468,831	145,620,000	-0.151169	-0.375000	0.375000		
Text/Value	H_PLASTIC_5	VAL	145,529,808	145,620,000	-0.090192	-0.375000	0.375000		
Text/Value	H_PLASTIC_6	VAL	145,361,488	145,620,000	-0.258512	-0.375000	0.375000		
Text/Value	H_PLASTIC_7	VAL	145,359,698	145,620,000	-0.260302	-0.375000	0.375000		
Text/Value	H_PLASTIC_8	VAL	145,448,787	145,620,000	-0.171213	-0.375000	0.375000		
Text/Value	H_PLASTIC_9	VAL	145,312,581	145,620,000	-0.307419	-0.375000	0.375000		
Text/Value	H_PLASTIC_10	VAL	145,543,152	145,620,000	-0.076848	-0.375000	0.375000		

Exemple de fichier de contrôle pour une pièce défectueuse

Feat. Type	Feat. Name	Value	Actual	Nominal	Dev.	Tol-	Tol+	Out of Tol.	Comment
Text/Value	VAR_PROGRAM	VAL							N:\Metrolog XG\Programmes\GMX\GMX Complete Module 01.gm2'
Text/Value	VAR_EVENEMENT	VAL							FIP-8586 chagement parametre soudeuse'
Text/Value	VAR_COMMENTAIRES	VAL							334 C'
Circle	CIR_A	DIAM	76,530	76,600	-0.070	-0.500	0.500		Criterion: Least Square (From 12 Pts On PL_PLAST_A)
Circle	CIR_A	X	0.000	0.000	0.000				
Circle	CIR_A	Y	0.000	0.000	0.000				
Circle	CIR_A	Z	0.000	0.000	0.000				
Pos. Tol.	POS_A	PTOL	0.000	0.000	0.000	0.000	1,000		CIR_A/NOMINAL - CS_ABC_BF - C-Zone - XY - Nominal
Circle	CIR_B	DIAM	16,238	16,200	0.038	-0.200	0.200		Criterion: Least Square (From 10 Pts On PL_PLAST_B)
Circle	CIR_B	X	-96,373	-95,900	-0.473				
Circle	CIR_B	Y	405,958	405,600	0.358				
Circle	CIR_B	Z	-59,500	-59,500	0.000				
Pos. Tol.	POS_B	PTOL	1.186	0.000	1.186	0.000	1.000	0.186	CIR_B/NOMINAL - CS_ABC_BF - C-Zone - XY - Nominal
Circle	CIR_C	DIAM	16,341	16,200	0.141	-0.200	0.200		Criterion: Least Square (From 10 Pts On PL_PLAST_C)
Circle	CIR_C	X	272,217	271,600	0.617				
Circle	CIR_C	Y	413,474	413,100	0.374				
Circle	CIR_C	Z	-16,000	-16,000	0.000				
Pos. Tol.	POS_C	PTOL	1,444	0.000	1,444	0.000	1,500		CIR_C/NOMINAL - CS_ABC_BF - C-Zone - XY - Nominal
Circle	CIR_D	X	260,329	261,100	-0.771				Inters. PL_PLAST_D - CYL_D.
Circle	CIR_D	Y	62,346	61,500	0.846				

Circle	CIR_D	Z	-51,815	-50,300	-1,515				
Pos. Tol.	POS_D	PTOL	3,797	0.000	3,797	0.000	4,000		CIR_D/NOMINAL - CS_ABC_BF - C-Zone - XYZ - Nominal
Circle	CIR_E	X	-,168,633	-,169,000	0.367				Criterion: Least Square (From 10 Pts On PL_PLAST_E)
Circle	CIR_E	Y	221,845	220,000	1,845				
Pos. Tol.	POS_E	PTOL	3.763	0.000	3.763	0.000	3.000	0.763	CIR_E/NOMINAL - CS_ABC_BF - C-Zone - XY - Nominal
Circle	CIR_OP	DIAM	74,309	74,000	0.309	-0.500	0.500		Proj. CIR_OP/PL_PLAST_OP.
Circle	CIR_OP	X	252,654	253,029	-0.375				
Circle	CIR_OP	Y	-32,536	-33,016	0.480				
Circle	CIR_OP	Z	-27,923	-27,367	-0.556				
Pos. Tol.	POS_OP	PTOL	1.649	0.000	1.649	0.000	5.000		CIR_OP/NOMINAL - CS_ABC_BF - C-Zone - XYZ - Nominal

ANNEXE II
LISTE DES CONTRÔLES ET LE NOMBRE DE FICHIERS ASSOCIÉS

Nom de la pièce	Nombre de fichiers associés
GM HF FWD Gen II C1xx Complete (Module 64). gm2	571
Denso 640F Complet (Module 51). gm2	444
GM HF FWD Gen II High Hood 31xx Complete (Module 64). gm2	421
Corvette LT1 Complete (Module 51). gm2	389
GM HF RWD Gen II Complete (Module 64). gm2	362
Ford 3	347
Corvette LT1 Free State Boss Height (Module 44). gm2	328
V10 6.8L Triton Complete Module 51.gm2	314
D33 RWD Complete (Module 44). gm2	262
Ford Transmission Pan 10R Complete (Module_51). gm2	243
GM HF FWD Gen II E2xx Complete (Module 64). gm2	238
Mercury Marine TigerShark Complete (Module 51). gm2	226
V10 6.8L Triton Stage-1 Module 51.gm2	217
Denso 507F Complet (Module 51). gm2	196
D33 RWD Crush Rib (Module 44). gm2	188
GM HF RWD Gen II Free State Flatness Flange (Module 64). gm2	186
D-35 Hauteur Plastic Crush Rib Moule 2 (Module 44). gm2	171
D-37 with TMAP Complete (Module 44). gm2	163
GDTI Gap Entretoises Module_50.gm2	163
D-37 Hauteur Plastic Crush Rib Module 44.gm2	162
D-35 Moule 2 Complete with TMAP (Module 44). gm2	157
GDTI Complete Module_18.gm2	154
Mercury Marine Complete (Module 51). gm2	152
Ford 2L GTDI Complete (Module 51). gm2	142
Ford Nano MY2018 Complete (Module_51). gm2	138
Ford Nano 2.7L Complete (Module_51). gm2	134

Ford Transmission Pan 10R80 Free State Cavité 1 (Module_44). gm2	133
Ford Transmission Pan 10R80 Complete Cavité 1 (Module_51). gm2	129
Ford Transmission Pan 10R60 Free State (Module_44). gm2	115
Denso 507F Stage-1 Module 51.gm2	104
GMI 700 Moule 1 Complete Module 51.gm2	97
Ford Transmission Pan 10R60 Complete Cavité 1 (Module_51). gm2	96
Ford Transmission Pan 10R60 Complete Cavité 2 (Module_51). gm2	92
D-35 Flatness Free State Moule 2 (Module 44). gm2	90
D-35 Hauteur Plastic Crush Rib Moule 1 (Module 44). gm2	90
Ford Transmission Pan 10R80 Free State Cavité 2 (Module_44). gm2	89
Ford Transmission Pan 10R80 Complete Cavité 2 (Module_51). gm2	83
Ford 2L GTDI Complete Moule 2 (Module 51). gm2	82
D-35 Moule 1 Complete with TMAP (Module 44). gm2	82
Denso 640F Stage-1 (Module 51). gm2	78
I4 Complete Moule 2 (Module 44). gm2	75
Ford Transmission Pan 10R60 Free State Cavité 1 (Module_44). gm2	75
Ford Nano 2.7L Free State (Module 51). gm2	73
Ford Nano MY2018 Free State (Module 51). gm2	71
D37 Mustang Complete (Module 44). gm2	68
D37 Mustang Hauteur Plastique Crush Rib (Module 44). gm2	67
Ford Transmission Pan 10R80 Complete Cavité 4 (Module_51). gm2	63
GMI 700 Moule 3 Complete (Module 51). gm2	61
GM Twin Turbo Complete (Module 51). gm2	60
GM Twin Turbo Free State Retaining Ring Groove (Module 67 & 44). gm2	56
GM Twin Turbo Boss Height Free State (Module 51). gm2	54
I4 Complete Moule 1 (Module 44). gm2	52
D-35 Flatness Free State Moule 1 (Module 44). gm2	44
GM Impala Validation Stage-1 (Module 51). gm2	38
Ford Transmission Pan 10R80 Complete Cavité 3 (Module_51). gm2	36

Ford Transmission Pan 10R Free State (Module_44). gm2	33
D-37 Validation Stage-1 (Module 44). gm2	31
Ford 2L Maverick FWD Complete (Module 44). gm2	31
D37 Mustang Validation Stage-1 (Module 44). gm2	29
TBA 2V Complete Module 51.gm2	25
Ford Transmission Pan 10R80 Free State Cavité 3 (Module_44). gm2	25
Ford Transmission Pan 10R80 Free State Cavité 4 (Module_44). gm2	25
TBA 3V Platine (Module 51). gm2	24
TBA 3V Complete (Module 51). gm2	24
GM Impala Complete Module 18.gm2	23
Ford Transmission Pan 10R60 Free State Cavité 2 (Module_44). gm2	23
GM Impala Boss Height Module 18.gm2	22
Ford Transmission Pan 10R80 MHT Free State (Module_44). gm2	22
GMI 700 Moule 3 Hauteur Crush Rib (Module 51). gm2	22
Mercury Marine Great White V6 Plenum (Module 51). gm2	21
Corvette LS7 Complete (Module 18). gm2	20
Dual Plenum Complete (Module 18). gm2	18
GM HF RWD Continental Complete Module 51.gm2	18
Road Runner Complete Module 51.gm2	17
Mercury Marine Great White V8 Plenum (Module 51). gm2	17
Ford Transmission Pan 10R80 Transit Free State (Module_44). gm2	17
Ford 2.3L Maverick RWD Complete (Module 51). gm2	16
P-415 Complete (Module_16). gm2	15
Mercury Marine Great White V8 Runner Pack LH (Module 51). gm2	14
Mercury Marine Great White V6 Runner Pack RH (Module 51). gm2	13
Mercury Marine Great White V8 Runner Pack RH (Module 51). gm2	12
Ford Nano MY2018 Program EGR (Module_51). gm2	12
Mercury Marine Great White V6 Runner Pack LH (Module 51). gm2	12
D33 MY2020 Crush Rib (Module 44). gm2	12

GMX Complete Module 01.gm2	11
GMI 700 Moule 2 Complete Module 51.gm2	9
GMI 700 Moule 1 Hauteur Crush Rib (Module 51). gm2	8
Ford Transmission Pan 10R80 Free State Cavité 3 (Module_44). gm2	6
Ford Transmission Pan 10R80 Free State Cavité 4 (Module_44). gm2	6
GM HF FWD Gen II E2xx Cover ASM (Module 51). gm2	4
Mercury Marine Great White V8 Runner Pack LH--TAB (Module 51). gm2	4
GMI 700 Moule 4 Hauteur Crush Rib (Module 51). gm2	3
Mercury Marine Great White V8 Runner Pack RH--NO TAB (Module 51). gm2	3
Viper Cover Air Cleaner Complet (Module 51). gm2	2
Toyota 930N Complet (Module 44). gm2	1
Ford Transmission Pan 10R80 Complete Cavité 3 (Module_51). gm2	1
GMI 700 Moule 4 Complete Module 51.gm2	1

LISTE DE RÉFÉRENCES BIBLIOGRAPHIQUES

[1] Apte, C., Weiss, S., & Grout, G. (1993). Predicting defects in disk drive manufacturing: A case study in high-dimensional classification. In Proceedings of 9th IEEE Conference on Artificial Intelligence for Applications (p. 212-218). Orlando, FL, USA: IEEE Comput. Soc. Press. https://doi.org/10.1109/CAIA.1993.366608

[2] Askarian, M., Benítez, R., Graells, M., & Zarghami, R. (2016). Data-based fault detection in chemical processes: Managing records with operator intervention and uncertain labels. Expert Systems with Applications, 63, 35-48. https://doi.org/10.1016/j.eswa.2016.06.040

[3] Dib, M., Ribeiro, B., & Prates, P. (2018). Model Prediction of Defects in Sheet Metal Forming Processes. In E. Pimenidis & C. Jayne (Éd.), Engineering Applications of Neural Networks (Vol. 893, p. 169-180). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-98204-5_14

[4] Laaouina, L., Nafi, A., & Mouchtachi, A. (2016). Application of CMM separation method for identifying absolute values of probe errors and machine errors. In 2016 International Conference on Engineering & MIS (ICEMIS) (p. 1-7). Agadir, Morocco: IEEE. https://doi.org/10.1109/ICEMIS.2016.7745333

[5] Lee-Ing Tong, Wei-I Lee, & Chao-Ton Su. (1997). Using a neural network-based approach to predict the wafer yield in integrated circuit manufacturing. IEEE Transactions on Components, Packaging, and Manufacturing Technology: Part C, 20 (4), 288-294. https://doi.org/10.1109/3476.650960

[6] Lin, J., Yao, Y., Ma, L., & Wang, Y. (2018). Detection of a casting defect tracked by deep convolution neural network. The International Journal of Advanced Manufacturing Technology, 97 (1-4), 573-581. https://doi.org/10.1007/s00170-018-1894-0

[7] Teti, R. (2015). Advanced IT Methods of Signal Processing and Decision Making for Zero Defect Manufacturing in Machining. Procedia CIRP, 28, 3-15. https://doi.org/10.1016/j.procir.2015.04.003

[8] Wang, K.-S. (2013). Towards zero-defect manufacturing (ZDM)—a data mining approach. Advances in Manufacturing, 1 (1), 62-74. https://doi.org/10.1007/s40436-013-0010-9

[9] Yin, S., Gao, H., Qiu, J., & Kaynak, O. (2017). Fault Detection for Nonlinear Process With Deterministic Disturbances: A Just-In-Time Learning Based Data Driven Method. IEEE Transactions on Cybernetics, 47 (11), 3649-3657. https://doi.org/10.1109/TCYB.2016.2574754

BIBLIOGRAPHIE

[1] Classifying Names with a Character-Level RNN—PyTorch Tutorials 1.0.0.dev20190112 documentation. (n.d.). Retrieved January 14, 2019, from https://pytorch.org/tutorials/intermediate/char_rnn_classification_tutorial.html

[2] Confusion matrix — scikit-learn 0.20.3 documentation. (s. d.). Consulté 18 février 2019, à l'adresse https://scikit-learn.org/stable/auto_examples/model_selection/plot_confusion_matrix.html#sphx-glr-download-auto-examples-model-selection-plot-confusion-matrix-py

[3] Deep Learning. (n.d.). Retrieved January 21, 2019, from http://www.deeplearningbook.org/

[4] Nielsen, M. A. (2015). Neural Networks and Deep Learning. Retrieved from http://neuralnetworksanddeeplearning.com

[5] The Unreasonable Effectiveness of Recurrent Neural Networks. (n.d.). Retrieved January 14, 2019, from https://karpathy.github.io/2015/05/21/rnn-effectiveness/

[6] Tolérance géométrique — Wikipédia. (n.d.). Retrieved January 16, 2019, from https://fr.wikipedia.org/wiki/Tol%C3%A9rance_g%C3%A9om%C3%A9trique

[7] 3.2.4.3.1. sklearn.ensemble.RandomForestClassifier — scikit-learn 0.20.3 documentation. (s. d.). Consulté 22 janvier 2019, à l'adresse https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html#sklearn.ensemble.RandomForestClassifier

[8] torch.nn — PyTorch master documentation. (s. d.). Consulté 15 janvier 2019, à l'adresse https://pytorch.org/docs/stable/nn.html#bcewithlogitsloss

[9] What is torch.nn really? — PyTorch Tutorials 1.0.0.dev20190318 documentation. (s. d.). Consulté 22 Décembre 2018, à l'adresse https://pytorch.org/tutorials/beginner/nn_tutorial.html#nn-sequential