HELLENIC REPUBLIC

National and Kapodistrian University of Athens

— EST. 1837 —

GENERAL DEPARTMENT

PhD THESIS

Employing Zero Defects in Industrial Manufacturing by Applying Artificial Intelligence Methods

Angelos Angelopoulos

CHALKIDA 2024

PhD THESIS

Employing Zero Defects in Industrial Manufacturing by Applying Artificial Intelligence Methods

Angelos Angelopoulos

Three-member advisory committee:

1. Panagiotis Trakadas, Associate Professor, Department of Port Management and Shipping, NKUA (Supervisor).
2. Stamatis Voliotis, Professor, Department of Agricultural Development, Agrofood and Management of Natural Resources, NKUA.
3. Antonios Hatziefremidis, Professor, Department of Aerospace Science and Technology, NKUA.

Approved by the seven-member examination committee on 28/06/2024.

Seven-member examination committee:

1. Panagiotis Trakadas, Associate Professor, Department of Port Management and Shipping, NKUA.
2. Stamatis Voliotis, Professor, Department of Agricultural Development, Agrofood and Management of Natural Resources, NKUA.
3. Antonios Hatziefremidis, Professor, Department of Aerospace Science and Technology, NKUA
4. Panagiotis Gkonis, Assistant Professor, Department of Digital Industry Technologies, University of West Attica.
5. Nelly Leligou, Professor, Department of Industrial Design and Production Engineering ΠΑΔΑ.
6. Theofanis Orphanoudakis, Professor, School of Science and Technology Hellenic Open University.
7. Panagiotis Karkazis, Associate Professor, Department of Informatics and Computer Engineering, University of West Attica.

ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών

— ΙΔΡΥΘΕΝ ΤΟ 1837 —

ΓΕΝΙΚΟ ΤΜΗΜΑ

ΔΙΔΑΚΤΟΡΙΚΗ ΔΙΑΤΡΙΒΗ

Μείωση Αστοχιών στη Βιομηχανική Παραγωγή με την Εφαρμογή Μεθόδων Τεχνητής Νοημοσύνης.

Άγγελος Αγγελόπουλος

ΧΑΛΚΙΔΑ 2024

ΔΙΔΑΚΤΟΡΙΚΗ ΔΙΑΤΡΙΒΗ

Μείωση Αστοχιών στη Βιομηχανική Παραγωγή με την Εφαρμογή Μεθόδων Τεχνητής Νοημοσύνης.

Άγγελος Αγγελόπουλος

Τριμελής Επιτροπή Παρακολούθησης:

1. Παναγιώτης Τρακάδας, Αναπληρωτής Καθηγητής, Τμήμα Διαχείρισης Λιμένων και Ναυτιλίας, ΕΚΠΑ (Επιβλέπων).
2. Σταμάτης Βολιώτης, Καθηγητής, Τμήμα Αγροτικής Ανάπτυξης, Αγροδιατροφής και Διαχείρισης Φυσικών Πόρων, ΕΚΠΑ.
3. Αντώνιος Χατζηευφραιμίδης, Επίκουρος Καθηγητής, Τμήμα Αεροδιαστημικής Επιστήμης Και Τεχνολογίας, ΕΚΠΑ.

Εγκρίθηκε από την επταμελή εξεταστική επιτροπή την 28/06/2024.

Επταμελής Εξεταστική Επιτροπή:

1. Παναγιώτης Τρακάδας, Αναπληρωτής Καθηγητής, Τμήμα Διαχείρισης Λιμένων και Ναυτιλίας, ΕΚΠΑ.
2. Σταμάτης Βολιώτης, Καθηγητής, Τμήμα Αγροτικής Ανάπτυξης, Αγροδιατροφής και Διαχείρισης Φυσικών Πόρων, ΕΚΠΑ.
3. Αντώνιος Χατζηευφραιμίδης, Καθηγητής, Τμήμα Αεροδιαστημικής Επιστήμης Και Τεχνολογίας, ΕΚΠΑ.
4. Γκόνης Παναγιώτη, Επίκουρος Καθηγητής, Τμήμα Τεχνολογιών Ψηφιακής Βιομηχανίας ΕΚΠΑ.
5. Λελίγκου Ελένη-Αικατερίνη, Καθηγήτρια, Τμήμα Μηχανικών Βιομηχανικής Σχεδίασης και Παραγωγής ΠΑΔΑ.
6. Ορφανουδάκης Θεοφάνης, Καθηγητής Σχολής Θετικών Επιστημών και Τεχνολογίας ΕΑΠ.
7. Καρκαζής Παναγιώτης, Αναπληρωτής Καθηγητής, Τμήμα Μηχανικών Πληροφορικής και Υπολογιστών ΠΑΔΑ.

Αγγελόπουλος Άγγελος
Διδάκτωρ Ηλεκτρολόγος Μηχανικός Τ.Ε.

Στην οικογένεια μου...

Abstract

The new era of Industry 4.0 (I4) underscores the importance of flexible, customized production with shorter lead times and less waste of resources. In the complex environment of manufacturing, implementing a zero-defect strategy can be achieved through the use of Artificial Intelligence (AI). The primary objective of this thesis is to integrate zero-defect manufacturing (ZDM) by utilizing AI and more particularly with machine learning (ML) techniques. This PhD thesis has been composed based on a series of technical and experimental sections primarily aimed at enhancing the production process by minimizing defects, reducing costs, and waste products through the presentation and application of advanced ML-based techniques. The methods and tools employed in this thesis are of general purpose and can be utilized across a wide range of industrial operations, irrespective of the production sector.

Additionally, this work provides a comprehensive introduction to ML and Deep Learning (DL). It analyzes classification and regression techniques, and presents ensemble learning and Federated Learning (FL) methods. A method for training models, evaluation techniques, metrics, and frameworks for improving the performance of ML/DL algorithms are also presented. The influence of ML to I4 is thoroughly investigated in this work while related work in the field of I4 and ZDM is also discussed. A literature review on concept drift and defect prediction and detection is also included. Furthermore, a literature review on ML-based predictive maintenance and FL is provided, with a unique section on predictive maintenance in Shipping 4.0.

This work also includes the implementation and comparison of supervised learning algorithms on a labeled dataset from the offset printing industry to illustrate the performance of ML-assisted approaches for ZDM. This approach resulted in a beneficial machine selection policy that will aid in improving production efficiency, reducing defective products, and mitigating the company's environmental impact. Furthermore, this thesis presents and analyzes concept drift. According to the literature, concept drift is a bottleneck for ZDM. Thus, a small-scale comparison of the influence of different classifiers in DDM is conducted, namely Naïve Bayes (NB), Hoeffding Tree classifier (HT), k-Nearest Neighbors (KNN), Passive Aggressive Classifier (PAC) and Stochastic Gradient Descent (SGD).

Finally, this thesis proposes an end-to-end and general-purpose FL scheme to support Predictive Maintenance (PdM) on numerous maritime nodes as well as three practical maritime use cases. Various PdM problems are tested, including regression, classification and time series forecasting using four distinct federated learning algorithms, namely Federated Averaging (FedAvg), Federated Averaging with Momentum (FedAvgM), Federated Averaging Stochastic Gradient Descent (FedSGD) and Federated Averaging with Proximal term (FedProx).

Key-words: Zero Defect Manufacturing, Industry 4.0, Shipping 4.0, Machine Learning, Deep Learning, Concept Drift, Federated Learning, Predictive Maintenance, Defect Minimization.

Περίληψη

Η νέα εποχή του Βιομηχανίας 4.0 (I4) υπογραμμίζει τη σημασία της ευέλικτης, προσαρμοσμένης παραγωγής με μικρότερους χρόνους παράδοσης και λιγότερη σπατάλη πόρων. Στο σύνθετο περιβάλλον της βιομηχανικής παραγωγής, η εφαρμογή μιας στρατηγικής μηδενικών απωλειών μπορεί να επιτευχθεί μέσω της χρήσης Τεχνητής Νοημοσύνης (AI). Το κύριο αντικείμενο αυτής της διατριβής είναι η ενσωμάτωση της στρατηγικής μηδενικών απωλειών στη βιομηχανία (ZDM), με τη χρήση τεχνητής νοημοσύνης και πιο συγκεκριμένα, με τεχνικές μηχανικής μάθησης (ML). Αυτή η διδακτορική διατριβή έχει συνταχθεί με βάση μια σειρά τεχνικών και πειραματικών ενοτήτων που στοχεύουν κυρίως στην ενίσχυση της παραγωγικής διαδικασίας με την μείωση των απωλειών, τη μείωση του κόστους και των απορριμμάτων μέσω της παρουσίασης και εφαρμογής προηγμένων τεχνικών που βασίζονται στην ML. Οι μέθοδοι και τα εργαλεία που χρησιμοποιούνται σε αυτή τη διατριβή είναι γενικού σκοπού και μπορούν να χρησιμοποιηθούν σε ένα ευρύ φάσμα βιομηχανικών λειτουργιών, ανεξάρτητα από τον κλάδο παραγωγής.

Επιπλέον, αυτή η εργασία παρέχει μια εκτενή εισαγωγή στη Μηχανική Μάθηση (ML) και τη Βαθιά Μάθηση (DL). Αναλύονται τεχνικές ταξινόμησης και παλινδρόμησης, και παρουσιάζονται μέθοδοι συνολικής μάθησης και ομοσπονδιακής μάθησης (FL). Επίσης, παρουσιάζονται, μια μέθοδος για την εκπαίδευση μοντέλων, τεχνικές αξιολόγησης, μετρήσεις απόδοσης και σχετικές βιβλιοθήκες κώδικα μηχανικής μάθησης για τη βελτίωση της απόδοσης των ML/DL αλγορίθμων. Επίσης, συζητείται η σχετική εργασία στον τομέα του I4 και του ZDM. Περιλαμβάνεται επίσης μια βιβλιογραφική ανασκόπηση για την έννοια της μετατόπισης δεδομένων και την πρόβλεψη-ανίχνευση ελαττωμάτων. Επιπλέον, παρέχεται μια βιβλιογραφική ανασκόπηση για τη προβλεπτική συντήρηση βασισμένη σε ML και την FL, με μια μοναδική ενότητα για την Προγνωστικής Συντήρηση (PdM) στην ναυτική βιομηχανία 4.0.

Αυτή η εργασία περιλαμβάνει επίσης την υλοποίηση και σύγκριση αλγορίθμων εποπτευόμενης μάθησης σε ένα επισημασμένο σύνολο δεδομένων από τον κλάδο της offset εκτύπωσης για να απεικονίσει την απόδοση των προσεγγίσεων με τη βοήθεια ML με στόχο την ZDM. Αυτή η προσέγγιση κατέληξε σε μια ευεργετική πολιτική επιλογής μηχανών που θα συμβάλει στη βελτίωση της αποδοτικότητας της παραγωγής, στη μείωση των ελαττωματικών προϊόντων και στη μείωση των περιβαλλοντικών επιπτώσεων της εταιρείας. Επιπλέον, η παρούσα διατριβή παρουσιάζει και αναλύει την μετατόπιση δεδομένων. Σύμφωνα με τη βιβλιογραφία, η μετατόπιση δεδομένων είναι ένα εμπόδιο για την ZDM. Έτσι, διεξάγεται μια μικρής κλίμακας σύγκριση της επιρροής διαφορετικών ταξινομητών στον αλγόριθμο DDM, συγκεκριμένα Naïve Bayes (NB), Hoeffding Tree classifier (HT), k-Nearest Neighbors (KNN), Passive Aggressive Classifier (PAC) and Stochastic Gradient Descent (SGD).

Τέλος, αυτή η διατριβή προτείνει ένα από άκρο σε άκρο και γενικής χρήσης σχήμα FL για την υποστήριξη της Προγνωστικής Συντήρησης (PdM) σε πολλούς ναυτιλιακούς κόμβους καθώς και τρεις πρακτικές περιπτώσεις χρήσης. Δοκιμάζονται διάφορα προβλήματα PdM, συμπεριλαμβανομένης της παλινδρόμησης, της ταξινόμησης και της πρόβλεψης χρονοσειρών με τη χρήση τεσσάρων διακριτών αλγορίθμων ομοσπονδιακής μάθησης, συγκεκριμένα Federated Averaging (FedAvg), Federated Averaging with Momentum (FedAvgM), Federated Averaging Stochastic Gradient Descent (FedSGD) and Federated Averaging with Proximal term (FedProx).

Λέξεις-κλειδιά: Στρατηγική μηδενικών απωλειών στην βιομηχανία, Βιομηχανία 4.0, Ναυτική βιομηχανία 4.0, Μηχανική μάθηση, Βαθιά μάθηση, Μετατόπιση δεδομένων, Ομοσπονδιακή μάθηση, Προγνωστική συντήρηση, Μείωση ελαττωμάτων.

Acknowledgments

This thesis is the culmination of a long and difficult effort. I am incredibly grateful for the valuable help, support and understanding from those who are important to me, whose contribution was vital to the completion of this thesis.

First and foremost, I am extremely grateful to my supervisor, Assoc Prof. Panagiotis Trakadas for his confidence, continuous support, and patience during my PhD studies. His spiritual guidance and advice were invaluable throughout the journey of this thesis.

I would also like to express my sincere gratitude to Prof. Stamatis Voliotis and Prof. Antonios Chatzieffraimidis, for their assistance at every stage of the research project.

My gratitude extends to my friends, colleagues, lab mates, and research team for their kind help, their support and collaboration.

I would also like to express my deep gratitude to Anastasios Giannopoulos and Nicolaos Nomikos for their extensive collaboration and substantial support over the years.

Last but not the least, I would like to thank my family, my parents, my brother and most importantly my wife for their spiritual support throughout the writing of this thesis and my life in general.

List of Publications

Journal Papers

1. A. Angelopoulos, E. T. Michailidis, N. Nomikos, P. Trakadas, A. Hatziefremidis, S. Voliotis, and T. Zahariadis. Tackling Faults in the Industry 4.0 Era: A Survey of Machine-Learning Solutions and Key Aspects. Sensors, 20(1), 2020.

2. A. Angelopoulos, A. Giannopoulos, N. Nomikos, A. S. Kalafatelis, A. Hatziefremidis and P. Trakadas, "Federated Learning-Aided Prognostics in the Shipping 4.0: Principles, Workflow, and Use Cases," in IEEE Access, doi: 10.1109/ACCESS.2024.3350777.

Conference papers

3. Angelos Angelopoulos, Anastasios E. Giannopoulos, Nikolaos C. Kapsalis, Sotirios T. Spantideas, Lambros Sarakis, Stamatis Voliotis, and Panagiotis Trakadas. Impact of classifiers to drift detection method: A comparison. In Lazaros Iliadis, John Macintyre, Chrisina Jayne, and Elias Pimenidis, editors, Proceedings of the 22nd Engineering Applications of Neural Networks Conference, pages 399–410, Cham, 2021. Springer International Publishing.

4. Angelos Angelopoulos, Anastasios Giannopoulos, Sotirios Spantideas, Nikolaos Kapsalis, Chris Trochoutsos, Stamatis Voliotis, and Panagiotis Trakadas. Allocating orders to printing machines for defect minimization: A comparative machine learning approach. In Ilias Maglogiannis, Lazaros Iliadis, John Macintyre, and Paulo Cortez, editors, Artificial Intelligence Applications and Innovations, pages 79–88, Cham, 2022. Springer International Publishing.

Contribution in Journal Papers

5. Panagiotis Trakadas, Pieter Simoens, Panagiotis Gkonis, Labros Sarakis, Angelos Angelopoulos Alfonso P. Ramallo Gonzalez, Antonio Skarmeta, Christos Trochoutsos, Daniel Calvo, Tomas Pariente, Keshav Chintamani, Izaskun Fernandez, Aitor Arnaiz Irigaray, Josiane Xavier Parreira, Pierluigi Petrali, Nelly Leligou, and Panagiotis Karkazis. An artificial intelligence-based collaboration approach in industrial iot manufacturing: Key concepts, architectural extensions and potential applications. Sensors, 20(19), 2020.

Contribution in Conference papers

6. Sotirios T. Spantideas, Anastasios E. Giannopoulos, Nikolaos C. Kapsalis, Angelos Angelopoulos, Stamatis Voliotis, and Panagiotis Trakadas. Towards zero-defect manufacturing: Machine selection through unsupervised learning in the printing industry. In I-ESA Workshops, 2022.

7. Alexandros S. Kalafatelis, Chris Trochoutsos, Anastasios E. Giannopoulos, Angelos Angelopoulos, and Panagiotis Trakadas. A stacking ensemble learning model for waste prediction in offset printing. In Proceedings of the 2023 10th International Conference on Industrial Engineering and Applications, ICIEAEU ’23, page 267–272, New York, NY, USA, 2023. Association for Computing Machinery.

8. A. S. Kalafatelis, N. Nomikos, A. Angelopoulos, C. Trochoutsos, and P. Trakadas, ‘An Effective Methodology for Imbalanced Data Handling in Predictive Maintenance for Offset Printing’, 2024, pp. 89–98. doi: 10.1007/978-981-99-6523-6_7.

Table of Contents

List of Tables

List of Figures

Acronyms

AI	Artificial Intelligence	LiR	Linear Regression
AGV	Automated Guided Vehicle	LoR	Logistic Regression
ANN	Artificial Neural Network	LSTM	Long-Short Term Memory Network
AR	Augmented Reality	M2M	Machine-to-Machine
BD	Big Data	MC	Multiple Classifiers
BDA	Big Data Analytics	ML	Machine Learning
CC	Cloud Computing	MLPNN	Multilayer Perceptron Neural Network
CD	Concept Drift	NB	Naïve Bayes
CNN	Convolutional Neural Networks	NN	Neural Network
CPPS	Cyber-Physical Production Systems	PAC	Passive Aggressive Classifier
CPS	Cyber-Physical Systems	PCA	Principal Component Analysis
DDM	Drift Detection Method	PCBs	Printed Circuit Boards
DL	Deep Learning	PdM	Predictive Maintenance
DNN	Deep Neural Network	RBM	Restricted Boltzmann Machine
DT	Decission Trees	RDDM	Reactive Drift Detection Method
DiTw	Digital Twin	RF	Random Forests
DWM	Dynamic Weighted Majority	RL	Reinforcement Learning
EDDM	Early Drift Detection Method	RNN	Recurrent Neural Network
FCNN	Fully-Connected Neural Network	RUL	Remaining Useful Life
FedAvg	Federated Averaging	SGD	Stochastic Gradient Descent
FedAvgM	Federated Averaging with Momentum	SL	Supervised Learning
FedSGD	Federated Stochastic Gradient Descent	SMOTE	Synthetic Minority Over-Sampling Technique
FL	Federated Learning	StL	Stream Learning
FNN	Feedforward Neural Network	SVC	Support Vector Classification
GAN	Generative Adversarial Network	SVM	Support Vector Machines
HDDM	Hoeffding's Inequality Drift Detection Method	SVR	Support Vector Regression
HMI	Human–Machine Interaction	UL	Unsupervised Learning
HT	Hoeffding Tree	VFDRC	Very Fast Decision Rules Classifier
I4	Industry 4.0	VR	Virtual Reality
ICT	Information And Communication Technologies	WSN	Wireless Sensor Networks
IoT	Internet of Things	ZDM	Zero Defect Manufacturing
KNN	K-Nearest Neighbors		

1. Introduction

1.1 Manufacturing Evolution towards Industry 4.0

During the last three hundred years, mankind has made significant advancements in the field of industrial manufacturing. Population growth, business expansion and the development of trade, pave the way to the first industrial revolution. Mechanical innovations, utilizing steam and water, drastically increase the productivity. Furthermore, locomotive and boats, powered by steam, contribute in the further expansion of the trade and the economic growth.

In the second industrial revolution, the introduction of gasoline and electrification led manufacturing sector to a new era. Electricity-powered inventions were manufactured in large quantities, which fueled the growth of new industries. Gasoline powered machines further increase the productivity, while trade meets massive expansion. Things get faster with less effort.

Then, starting from the 1950s, the third industrial revolution adopted increased digitization using semi-conductors and more recently communication networks. New innovations also introduced that period, such as personal computer, opening new avenues to the internet era. The introduction of electronics in manufacturing paves the way to an automated production process, leading to a massive and efficient expansion of production volume.

During the last decade, Artificial Intelligence (AI) and Machine Learning (ML) have been introduced in the manufacturing sector, enabling more efficient processes sustainability with reduced waste and consumption of materials, safer working environments and increased quality and productivity. AI/ML-based manufacturing can offer various manufacturing innovations by providing fault detection and prediction, optimal use of raw materials and resources, exploiting the heterogeneous big data analysis and the interconnected manufacturing plants [1], [2], [3], [4], [5], [6].

The fourth industrial revolution (Industry 4.0) aims at providing an industrial environment for real-time, intelligent, interoperable, and autonomous manufacturing environments [7]. In order to realize this vision, Industry 4.0 (I4) is based on recent innovative information and communication technologies, such as cyber-physical systems (CPS), Internet of Things (IoT) and cloud computing (CC) [8], having attracted the research interest of the academia and the industry in a plethora of applications [9]. CPS results in the interconnection of physical elements with cyber-elements [10]. More specifically, physical elements comprise components such as sensors, operator panels and computers, collaborating and communicating to collect and provide data to cyber-elements where management, processing and decision-making takes place [11], [12]. Furthermore, IoT enables the real-time interconnection of different objects, for example, sensors, actuators, machines and robots among others, in a safe and reliable manner. At the heart of IoT lie heterogeneous communication networks, including fifth generation (5G) networks, Wi-Fi, machine-to-machine (M2M) deployments and cloud technologies (cloud, fog, edge) [13], [14], [15]. At the same time, humans should maintain an important role in the Industry 4.0 environments, empowered with smart devices, virtual and augmented reality (VR/AR), being in the loop of

the manufacturing process and taking advantage of the AI/ML-based decision-making [16]. Thus, considering these advanced technologies, Industry 4.0 promises to radically change the current industrial production processes benefiting industrial stakeholders, personnel and consumers, promoting environmental sustainability [17].

In the context of I4, a plethora of applications is envisioned, providing flexibility, competency, real-time self-optimization, and automation, as well as accomplishing complex tasks and satisfying strict quality requirements in intertwined digital and physical procedures [18]. More importantly, the relevant ubiquitous applications mainly span in the manufacturing and production development areas [19]. To evolve these applications, fault detection, prediction and prevention play a major role. With the contribution of ML algorithms, early and accurate fault detection can lead to minimum downtime, owing to the recognition of damaged and defective products or parts, in real-time. The wealth of data contributes to accurate prediction of machine condition, remaining useful life (RUL), and faults, leading to an appropriate and cost-effective maintenance schedule, thus minimizing the downtime, due to a fault occurrence [13], [20]. The human factor has also an important role in the production procedure of the Industry 4.0 ecosystem, whereas collaboration with robots and machines is a critical challenge within factory halls. Human activity recognition and decision-making algorithms increase operators' performance, leading to efficient and safe production, as well as to the minimization of ramp-up time [17].

1.2 Zero Defect Manufacturing

In the modern industrial environment, the quality of produced products with low cost and zero defective products is of utmost importance. At the same time, the protection of the environment plays a central role in the modern era. The necessity for clean and zero defective production is more crucial than ever. The increase in demand and the introduction of new products and services to meet the needs exacerbates the problem. The need to find a sustainable solution to the problem is vital. A promising solution is Zero Defect Manufacturing (ZDM) [21], [22]. ZDM is a comprehensive quality management strategy that aims to eliminate defects in the manufacturing process, in order to deliver products of impeccable quality, while minimizing the economic and environmental impact.

The first attempt to approach zero defect comes back from 1965, a guide for the department of defense:

"*For planning, implementing, and sustaining a Zero Defects-type program designed to motivate all persons directly or indirectly involved in the national defense effort to do their Jobs right the first time, every time*" [23].

Today with the contribution of emerged technologies that frame I4, we can effectively implement a strategy of ZDM. The core target of ZDM is to achieve perfection in manufacturing production process and products, by addressing every aspect of the production cycle, from design and raw material procurement to final product delivery. The focus of ZDM extends, to the continuous improvement of product and production, from every level of the organization, emphasizing prevention rather than correction. By applying the ZDM strategy, the industry can gain economic and strategic advantages by improving the quality of

manufactured products, keeping production and maintenance costs at low levels, and reducing waste, while increasing customer satisfaction [21], [22], [24].

To achieve zero defects in the complex environment of manufacturing a viable strategy is needed. To that end, [24] refers to four strategies divided into two categories named, triggering factors and actions. Triggering factors include the Detect and Predict strategies, while actions comprise the Repair and Prevent. The latest strategies, Repair and Prevent are used as actions to enchase the quality of the product when a triggering action occurs. The Detect strategy is used to detect defects, faults, and anomalies while the Predict strategy is used to predict faults, defects, and anomalies [23]. Two main approaches are basically followed by ZDM implementation, product-oriented and process-oriented approach. Both of them targeting in zeroing the defect in the end product. Product-oriented approach monitors the quality of the product at the end of the line and if a defect is detected then a repair or maintenance action is needed in the machine, otherwise the machine state is healthy. On the contrary, the Process-oriented approach monitors the machine status, and if a defect or fault is detected in the machine, a defect in the end product is triggered. **Figure 1** depicts ZDM strategies and approaches.

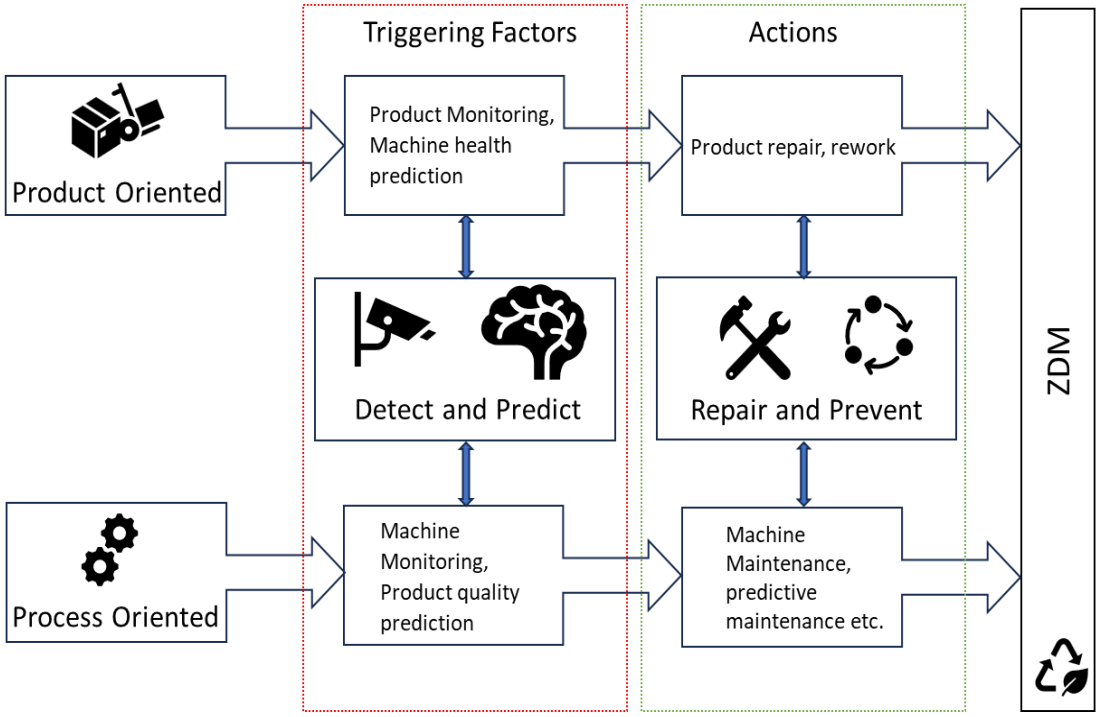

Figure 1 *ZDM Strategies and approaches*

1.2.1 Current challenges to ZDM adoption

Despite the interest in ZDM strategies that in the last few years has increased, there are still many issues that could be addressed. According to the literature, we can highlight the following shortcomings:

- **Concept drift detection:** The monitoring of process parameters, continuous quality control, and online predictive maintenance are central pillars of ZDM. Therefore, massive amounts of data are continuously gathered during the system operation. The occurrence of concept drift may result in the production of defective products. The rapid and accurate detection of sudden changes in the data stream may lead to proper corrective actions.
- **ZDM strategies extension to different industry sectors:** In general, the study of detection and prediction strategies has been extensively conducted, with a particular emphasis on specific processes. An extension to different industry sectors is required.
- **ZDM from zero:** To attain a more enhanced production with zero defects and a low cost, it is imperative to employ proficient and dependable ML and DL models for each specific process. To accomplish this objective, a substantial quantity of data is required to train the ML/DL algorithms. When a new production line is about to begin, this is not the case.
- **Holistic approach:** The literature indicates that the majority of studies focus on a single strategy, either product- or process-oriented. A comprehensive approach that encompasses both approaches is essential.

1.3 Artificial Intelligence

In the context of ZDM, artificial intelligence and more particularly ML and DL algorithms are far more important. With their contribution, they enable the fast and accurate detection of defective products, enabling fast implementation of correction and improvement actions. Also, they can provide an accurate prediction of machinery condition by indicating the RUL and enable the scheduling of effective maintenance, with reduced cost.

1.3.1 Machine Learning

The application of ML algorithms requires the existence of a vast amount of data to trigger decision-making in several industrial processes. In this regard, the implementation of novel technological paradigms, such as CPS and IoT, enables the generation of different types of data structures, as has been observed in works focusing on Big Data Analytics (BDA) [20], [25], [26]. In general, data has a specific life cycle (source, collection, storage, processing, visualization, transmission, application) [27]. However, most of the time, the gathered data that will be processed in subsequent steps, is confused with noisy data generated from the surrounding environment, making it difficult to separate the original data set from noise. On the other hand, fast-changing dynamic environments and different machine working states impose significant challenges to ML-based fault detection, prediction, and prevention. Overall, the need for reliable and accurate real-time transmission and computation arises, while security issues are becoming increasingly serious, due to the increased level of interconnection among the different subsystems [13], [17].

1.3.2 Concept Drift

Driven by the ever-growing evolution towards Industry 4.0, the zero defects manufacturing concept plays a critical role in the competitiveness of production industries [28]. According to Industry 4.0 expectations, creative technologies, new products, and novel procedures are getting involved in the production chain, generating a huge amount of data. Monitoring of process parameters, continuous quality control, and on-line predictive maintenance are central pillars of ZDM. Under those circumstances, fast and real-time prediction, based on massive streaming data processing, becomes a critical factor in the accomplishment of ZDM targets. Importantly, massive data are continuously gathered during the system operation, thus posing significant challenges in the data storage and manipulation for later analysis. ML and, in particular, stream learning (StL) has the potential to solve problems related to data analysis, such as class imbalance, noisy and/or abnormal data, mainly due to its memorization and generalization capabilities[17]. A common problem in streaming data is the concept drift which refers either to the change of the target variable while the statistical properties of the data remain unchanged - known as real drift - or the virtual drift that occurs when the data distribution changes while the target variable remains stable [29].

1.3.3 Federated Learning

The key concept of ZDM is the minimization of defects. This can be achieved by applying AI methods in the production process, such as predictive maintenance and defect detection. In order to achieve enhanced production with zero defects and low cost, well-trained and reliable ML and DL models should be applied for each respective process. To achieve this, a huge amount of data is needed in order to train the ML/DL algorithms. This is not the case when a new production line is about to start. In addition, the big data era brings in drawbacks such as data security and privacy-preserving. A promising approach to overcome these issues is federated learning (FL). Federated learning is an outstanding ML approach, that promises decentralization and collaboration among participants while preserving data security and privacy. It allows distributed training across multiple models, without the need to exchange raw data. Therefore, FL aligns with the goals of ZDM by enabling collaborative model training across distributed devices or manufacturing locations without centralizing sensitive data. ML/DL models are engaged at the very beginning, with improved performance, enabling rapid alignment with ZDM targets.

FL offers several advantages, particularly in maritime applications where training can be based on pre-existing data available at each client. Furthermore, FL ensures high levels of privacy and security since attacks can only impact individual nodes instead of compromising the entire cloud infrastructure. In maritime environments, FL-based model integration facilitates the creation of global popularity prediction models by leveraging local models [30], [31].

Even though, FL has shown numerous benefits in enabling privacy preservation, it also presents significant challenges in resource management, robustness, security, and incentive mechanisms [32]. These challenges become even more pronounced and difficult to address in

the context of industrial IoT (IIoT), where much higher levels of security, safety, and reliability are demanded in applications, such as shipping, smart ports, smart factories, smart manufacturing and smart transportation. In the area of PdM there have been various studies presenting FL-based anomaly detection with minimized computation cost for industrial control systems [33], autoencoder-based FL, using vibration sensor data from rotating machines, enabling distributed training on edge devices [34], split-learning-based PdM, facilitating FL clients to maximize available resources within their local network, and addressing orchestration, device heterogeneity and scalability issues [35], blockchain-based FL, relying on a hierarchical aggregator network, punishing and rewarding clients according to their local model quality updates [36]. Still, the integration of FL-aided prognostics in Shipping 4.0 applications has been largely missing from the literature and in this study, we aim to fill this gap by carefully tackling a variety of maritime use cases [37].

1.4 Contribution

This PhD dissertation focuses on Employing Zero Defects in Industrial Manufacturing by applying AI methods and more particularly ML and DL methods. This work mainly focuses on concept drift detection, defect minimization, and predictive maintenance.

The main goals of this research are summarized as follows:

- To provide a comprehensive overview of:
    - ZDM
    - ML in Industry 4.0
    - AI
    - Concept drift detection
    - Defect minimization
    - Federated Learning
    - supervised, unsupervised and deep learning solutions in the fields of predictive maintenance.
    - Predictive maintenance in Shipping 4.0.
- The implementation and comparison of SL algorithms in a labeled dataset from the offset printing industry, targeting to demonstrate the potency of ML-assisted methods towards ZDM by:
    - The exploitation of historical knowledge extracted by a real printing industry environment to obtain accurate and machine-specific supervised learning models.
    - The proposition of a machine selection policy targeting at minimizing the printing defects by proactive allocation of orders to machines.
    - The training, testing and comparison of multiple regression models (namely 10) to properly map the regressors to machine-specific datasets.
    - The implementation of both single-model and multi-model (or ensemble) regressors to investigate potential accuracy benefits in using combinations of multi-predictors.
    - The utilization of both classical ML and deep learning based schemes to estimate the printing accuracy of multi-feature orders.

- The comparison between different base classifiers for drift detection in streaming environments with:
    - Impact analysis of the performance across different base classifiers in terms of drift detection accuracy.
    - Performance evaluation of the concept drift detector in the presence of imbalanced/noisy data.
- To propose an end-to-end and general-purpose FL scheme for supporting PdM on multiple maritime nodes and diverse use cases by:
    - Presenting a comprehensive and end-to-end pipeline for supporting FL-based PdM capabilities in the Shipping 4.0 era.
    - Conducting a complete experimental testing of three types of PdM problems, namely regression, classification and time-series forecasting, to concretely evaluate the FL-based PdM performance in practical maritime use cases, exploiting both open and real datasets.
    - Comparing the different broadly-used FL strategies for model aggregation. By comparing four different aggregation policies (FedAvg, FedProx, FedSGD and FedAvgM).

1.5 Structure

The rest of the dissertation is organized as follows:

Chapter 2 introduces machine learning and provides a categorization of ML algorithms. Then classification and regression techniques are explained. Furthermore, ML and DL algorithms for classification and regression are demonstrated. After that, ensemble learning is presented by explaining the voting methods and algorithms such as bagging and boosting. Then, federated learning is presented and aggregation algorithms are explained. The main goal of this chapter is to enhance the performance of ML/DL algorithms. Therefore, a basic model training procedure is demonstrated, while evaluation techniques and metrics are presented. Furthermore, relevant frameworks are provided in the context of machine learning, deep learning, federated learning, and drift detection. Finally, ML contribution to I4 is extensively discussed.

Chapter 3 provides a literature review in the context of I4 and ZDM. Furthermore, it provides many papers on regard of concept drift detection and drift detection algorithms. After that, it provides a review of prediction and detection of defective products and wasted materials. Also, provides a literature review of ML-based predictive maintenance. Finally, it provides an overview of federated learning with a special section targeting predictive maintenance in Shipping 4.0.

Chapter 4 outlines the implementation and comparison of supervised learning (SL) algorithms in a labeled dataset from the offset printing industry, targeting to demonstrate the potency of ML-assisted methods towards ZDM. The purpose of the SL modeling is to link the specific characteristics of the received customer orders with the number of defects in the printing process. The accurate correlation of order features with the defective products will result in a beneficial machine selection policy, which in turn will contribute to the enhancement of production efficiency, the minimization of defective products, and the reduction of the company's environmental footprint.

In Chapter 5 concept drift is conducted. More particularly, characteristics and types of concept drift are analyzed. Also, concept drift detection algorithms and adaptation strategies are presented, while evaluation techniques are included. In addition, artificial and real-world datasets regarding concept drift detection are demonstrated. Finally, a small-scale comparison concerning the impact of different classifiers in DDM is conducted. The experimental setup and the dataset characteristics, along with the comparison of the algorithms under consideration and the evaluation procedure are presented. In addition, the simulation results of the present study are demonstrated.

Chapter 6 proposes an end-to-end and general-purpose FL scheme for supporting predictive maintenance (PdM) on multiple maritime nodes and diverse use cases. By testing three types of PdM problems, namely regression, classification, and time-series forecasting, different strategies are adopted for implementing the federation step (FedAvg, FedAvgM, FedSGD, and FedProx), as an attempt to deal with the dataset-specific intrinsic properties (e.g. data distributions, unknown inter-feature relationships) and heterogeneity between agents. Specifically, FL-based predictive maintenance is presented and evaluated for three practical maritime use cases, i.e. regression to predict the naval propulsion gas turbine (GT) measures, classification to predict the ship engine condition, and time-series regression to predict ship fuel consumption.

Chapter 7 includes a summary of the findings of this dissertation and presents future research directions.

2. Machine Learning – Deep Learning

Machine learning is an area of AI that uses data-driven algorithms to generate knowledge and accomplish difficult tasks. These algorithms serve as the cornerstone for ZDM and are utilized in a variety of applications such as defect detection, waste prediction, RUL prediction, quality inspection, and many more.

The ML algorithms can be categorized into supervised learning, unsupervised learning, reinforcement learning (RL), semi-supervised learning, and deep learning algorithms [20]. Each category is briefly described below:

- **Supervised learning** is a method where an expert inserts known outputs for specific inputs to train the algorithm and make predictions on unseen instances. It is widely used for classification and regression [20], [38], [39]. Thus, supervised ML is usually employed in scenarios with labeled data availability. Popular algorithms include artificial neural networks (ANNs) and support vector machines (SVMs) [20], [38], [40].
- **Unsupervised learning** where there is no feedback provided from anyone and the algorithm finds patterns in unknown data sets (clustering, association rules, self-organized maps) [20], [38], [40] and so, unlabeled data are used for training purposes. The most popular and well-known unsupervised algorithm is principal component analysis (PCA), mainly used for monitoring purposes [38].
- **Reinforcement learning** refers to unsupervised ML operation, examining if a chosen action resulted in a reward, for a specific performance metric [20]. RL demands sequential actions and tries their outcome, selecting those better fitting the problem at hand. So, RL significantly departs from other learning categories which are based on leveraging historical data, creating intelligence from previous decisions and rewards. It is widely used in automated guided vehicles (AGV) and articulated robots.
- **Semi-supervised learning** falls between supervised and unsupervised learning methods. Following the nature of the available training data, the model is trained in labeled and unlabeled data. This method is very advantageous when we have to deal few labeled data and large amount of unlabeled data.
- **Deep learning** where multiple layers have been employed in order to build an ANN, which is able to make intelligent decisions, handling large amounts of data with high complexity, without any human intervention [20], [41], [42]. Some DL algorithms are convolutional neural networks (CNNs), restricted Boltzmann machine (RBM) and auto-encoders (AE) [42].

In **Figure 2**, the different ML categories are summarized and their key characteristics are highlighted. It is evident that as the Industry 4.0 era is upon us, there exists an ever-increasing adoption level of ML algorithms to satisfy the needs of different aspects of industrial settings. These include process monitoring and quality control, fault detection and diagnosis, as well as machine health monitoring and predictive maintenance [43]. Moreover, the capabilities of ML, regarding the timely processing of an abundance of data are critical to safeguarding the cyber-security of the industrial IoT-enabled interconnected manufacturing environments, accurately detecting and mitigating threats [41], [44].

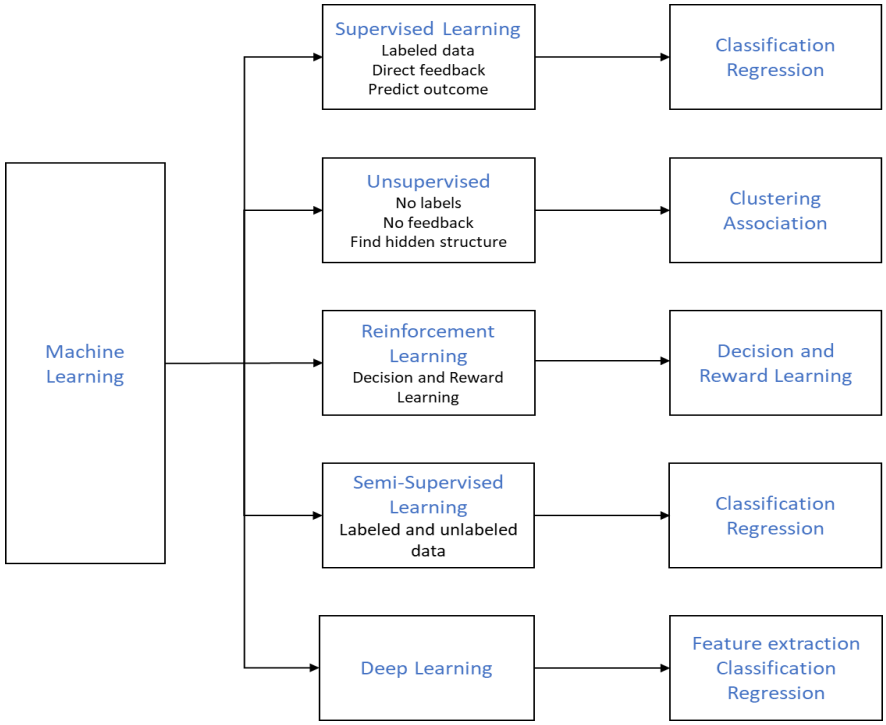

Figure 2 *Various machine learning categories and their key characteristics.*

2.1 Classification - Regression

Classification is a technique in which an algorithm is trained with a known set of features and labels (targets) to classify a set of known features into labels. Features can be in binary form, as continuous values, or as categorical values [45]. If we can choose between two classes, the classification is called binary (two-class classification), if we have more than two classes the classification is called multiclass. **Figure 3** shows the binary and multiclass classifications. The red line separating the classes is called the decision boundary and is learned from the training dataset. Following training, the algorithm can predict on which side of the boundary the target class belongs to.

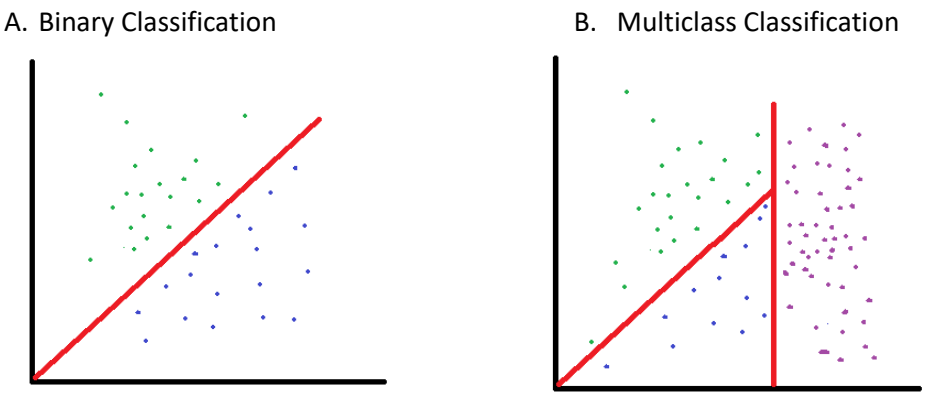

Figure 3 *Binary (A) and multiclass (B) classification*

Regression is also a supervised learning technique that is used to predict target variables from input variables. The input variables can also be binary, continuous, or categorical values. The target variable is a real or continuous value. Regression can be categorized as simple regression, multiple regression, and non-linear regression.

Simple regression is used when we only have one input variable and we try to predict a real or continuous output variable. It is defined by the following equation:

$$y = a + bx + e \tag{1}$$

where a is the intercept, b is the slope of the line, and e is the error term.

Multiple regression is used when we have multiple input variables in order to predict a real or continuous output variable. It is defined by the equation:

$$y = a + b_1 x_1 + \ldots\ldots + b_n n_n + e \tag{2}$$

where a is the intercept, b is the slope of the line, and e is the error term.

Nonlinear regression is used when a linear relationship between input variables and output variables is abscend. It is defined by the equation:

$$y = b_0 + b_1 x + b_2 x^2 + \ldots\ldots + b_n x^n + e \tag{3}$$

where $b_0, b_1, b_2, \ldots . b_n$ are the regression coefficients and x is the input variable.

Many algorithms have been proposed for regression such as linear regression, logistic regression, support vector regression, decision tree regression etc. In general regression can be used for prediction and forecasting.

2.2 Deep learning

Deep learning is the field of machine learning that uses artificial neural networks (ANNs) inspired by the learning ability of the human brain. ANNs consist of layers that are divided into input layers, hidden layers and output layers. The first layer of a neural network (NN) is always the input layer, where the features are fed into the network to be further processed in the hidden layers. The hidden layers are the place where all learning processes take place. The hidden layers feed the output layer to make the final prediction. Each layer is connected to the next layer with a certain number of nodes, also called neurons. In each layer, each input is multiplied by a weight, summed and finally transformed with an activation function [46].

Neural networks can be classified in the following types feedforward neural networks, convolutional neural networks and recurrent neural networks [46], [47].

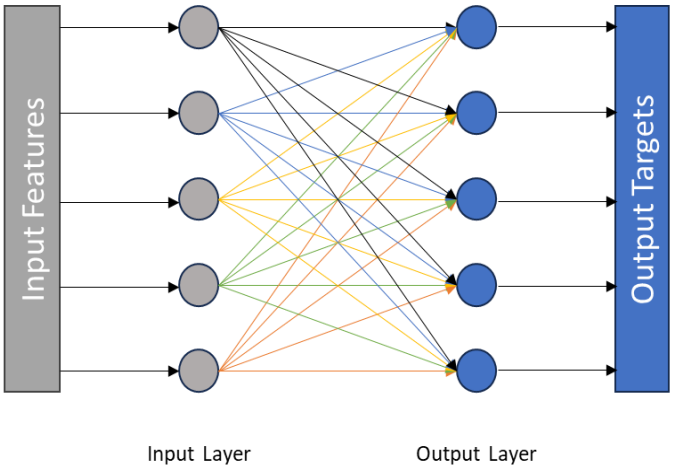

Figure 4 *Single-layer FNN*

Feedforward Neural Networks (FNNs) are the simplest form of neural networks. FNNs are characterized by the direction of information flow. The information flows in one direction only and moves from the input layer to the output layer, passing through the hidden layers. NNs without hidden layers are called single-layer neural networks and if they have one or any number of hidden layers, they are called multilayer neural networks [48]. In **Figure 4** and **Figure 5**, show a general architecture of single-layer and multi-layer neural networks, respectively. As we can see from the figures, the node of each layer is connected to all nodes of the next layer, this is called a fully connected neural network.

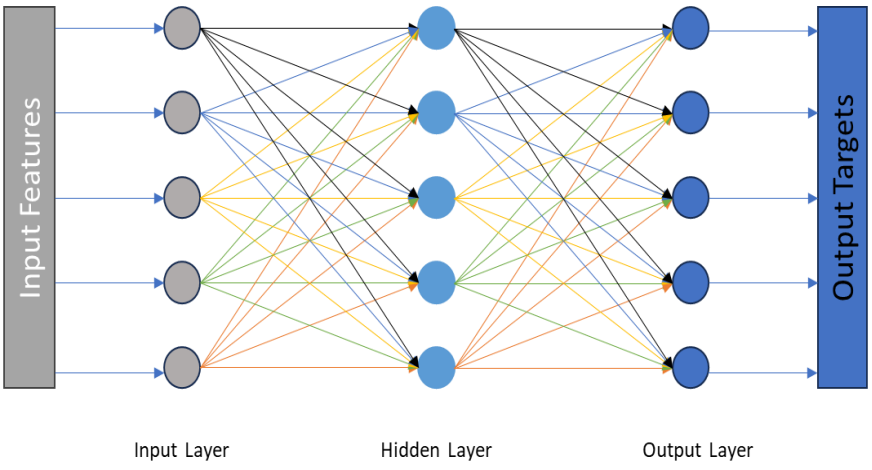

Figure 5 *Multi-layer FNN*

Convolutional Neural Network (CNN) is a type of feedforward neural network that is mainly used for classification. It consists of input layers, hidden layers and output layers. Three

main types of layers are used as hidden layers, namely convolutional layers, pooling layers and fully-connected layers. The most important layer in a CNN is the convolutional layer. Each node of the convolutional layer is connected to a set of neighboring nodes of the previous layer. Then the convolution layer convolves the connected nodes with a trained convolution kernel and then uses an activation function to provide the outputs. After the convolution layer a pooling layer is used for downsampling. Some of the widely used pooling techniques are average pooling and max pooling. Next, fully connected layers are connected to all nodes of the previous pooling layer to combine global information. Finally, an output layer with an activation function, such as softmax, is used for classification purposes [49].

Recurrent Neural Networks (RNNs) have a bidirectional information flow, they use recurrent layers that are able to preserve past knowledge and process long-term dependencies [50]. Recurrent layers pass their output as input for the next step. Nevertheless, RNNs have significant drawbacks such as vanishing and exploding gradients. For this reason, long short-term memory (LSTM) [51] was introduced to provide a practical solution to these problems. An LSTM cell consists of memory cells and gate units. The memory units are used to maintain the information storage while the gate units are used to manipulate the information flow, i.e. they decide when to keep or discard the information from the memory.

2.3 Machine learning and deep learning algorithms

According to the literature many algorithms have been proposed for classification and regression [45], [52]. In the next paragraphs the most commonly used algorithms are briefly described.

Logistic regression (LoR) is classification algorithm that uses a logistic function in order to map input data into classes. The algorithm predicts the probability distribution of features for belonging to a class and takes values between 0 and 1.

Linear regression (LiR) is a regression model that utilize a linear function in order to map feature variables to a target variable or multiple target variables. Linear Regression assumes that the target variable can be predicted through a weighted linear combination of the N independent variables. It results into a linear model with coefficients wi ($i = 1, 2, \ldots N$), graphically represented by a straight line that minimizes the residual sum of squares between the actual and predicted Accuracy values [53].

Naive Bayes (NB) [54] is a supervised learning algorithm based on probability theory. The main advantages of this classifier are its speed and its ability to require a small number of training examples and is widely used in cases of incremental learning [29].

Support Vector Machines (SVM) [55] in the training phase generate a hyperplane that is used to separate the **classes** e.g. in a two-class classification problem. The best hyperplane is recognized based on the maximum distance of the nearest data points (margin). The next figure illustrates the decision function in a two-class classification problem. The red line in the figure reflects the hyperplane that separates the two classes. The two grey lines define the margins of the largest distance between the nearest data points. Datapoints that fall on the margin lines are called support vectors. SVM can be used for classification (Support Vector Classification SVC) and regression problems (Support Vector Regression SVR). SVR uses an ε-

insensitive tube to allow for more flexibility in errors. Unlike Linear Regression, SVR penalizes only the data points that have at least ε distance from the fitting curve. The target of the SVR is to fit the best line within a threshold value (ε), reflecting the distance between the hyperplane and boundary line [53].

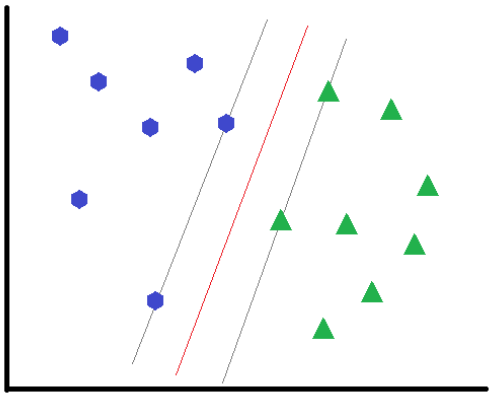

Figure 6 *SVM Decision boundaries*

Decision Tree (DT) is a supervised learning technique that uses a non-parametric method **to** learn from data. The tree consists of branches and leaves. Branches represent feature splits that lead to leaves, and leaves represent the target variables (classes). Decision trees can also be used for regression purposes. **Decision Tree (DT) regressor** is a non-parametric ML model that decides where and how to split the feature space, so as to ensure high information entropy in each decision area. As a result, a rule-based tree is constructed based on the previous separations of the feature space until no further splitting is required [53]. In general, DTs are very easy to understand and do not require much pre-processing of the dataset, while they can handle missing and noisy data.

Hoeffding Tree (HT) is an incremental tree classifier and is widely used for streaming data classification purposes. It uses Hoeffding bound, which quantifies the number of examples needed to estimate some statistics within a prescribed precision [29].

Random Forests (RF) is an ensemble method that combines the prediction of multiple decision trees by averaging them to make the final decision. RF can also be used for classification and regression purposes [53].

Stochastic Gradient Descent (SGD) regression is actually an optimization method that produces a linear model by minimizing a regularized empirical loss. The method relies on the calculation of the loss gradient of each sample, whereas the model parameters are gradually updated depending on the selected learning rate [53].

k-Nearest Neighbors (KNN) is a typical nearest neighbor classifier, whose main advantage is its simple implementation, while its major disadvantages that it can exhibit slow convergence in big datasets [29].

Passive Aggressive Classifier (PAC) is typically used for classification and regression problems and is very efficient on handling big datasets [29].

Stochastic Gradient Descent (SGD) classifier implements regularized linear models with stochastic gradient descent. Using hinge as loss function, SGD becomes an SVM classifier. Moreover, SGD allows incremental learning with the use of the partial fit method [29]. **Stochastic Gradient Descent (SGD) regression** is actually an optimization method that produces a linear model by minimizing a regularized empirical loss. The method relies on the calculation of the loss gradient of each sample, whereas the model parameters are gradually updated depending on the selected learning rate [53].

Very Fast Decision Rules Classifier (VFDRC) is an incremental rule-based classifier. VFDRC shares some common characteristics with HT method, but instead of trees it uses a collection of rules [29].

Multi-Layer Perceptron (MLP) or Artificial Neural Networks (ANNs) is a supervised learning scheme using multiple stacked layers (input, hidden and output). Each layer consists of multiple units (neurons) that compute the weighted sum of all the previous-layer neurons and then apply an activation function (e.g. softmax, ReLu, etc.). MLP properly adjusts the weights of the network so as to ensure minimization of the prediction error (or loss function) [53].

2.4 Ensemble learning

Ensemble learning is a method that uses the prediction of multiple classifiers in order to provide a combined prediction. Ensemble learning is characterized by an enchased prediction accuracy. In general, ML models are used in ensemble in two ways: Model selection and model fusion. In model selection, ML models are trained in local variables of the feature space. On the other hand, in model fusion ML models are trained on the entire dataset. Based on the literature many voting techniques have been introduced [56] :

- **Majority Voting** is a technique where all classifiers vote and the class with the most votes is the prediction of the ensemble [57].
- **Weighted Majority Voting** is used when information that some classifiers are performing better than others exist. Giving more weights to the prediction of those classifiers, the performance of the ensemble will be improved [58].
- **Borda Count** is a voting system were each classifier votes in all classes. At the end all votes are combined providing the class with the most votes.

In the last years, mane ensemble learning algorithms have been developed such as voting, bagging, boosting and stacking. **Figure 7** depicts the aforementioned techniques.

Voting is used to combine previously trained ML model (even ensemble estimators) from the same dataset in order to build a powerful collaborative model. Voting can be used for classification and regression (**Voting regressor**) problems. New predictions are derived by consulting each individual models and keeping the average output value [53].

Bagging ensemble is an algorithm that trains each classifier with different subset of the training dataset and then combines their votes with majority voting. A popular algorithm that also uses bagging is random forests [59] that uses decision trees as base classifiers.

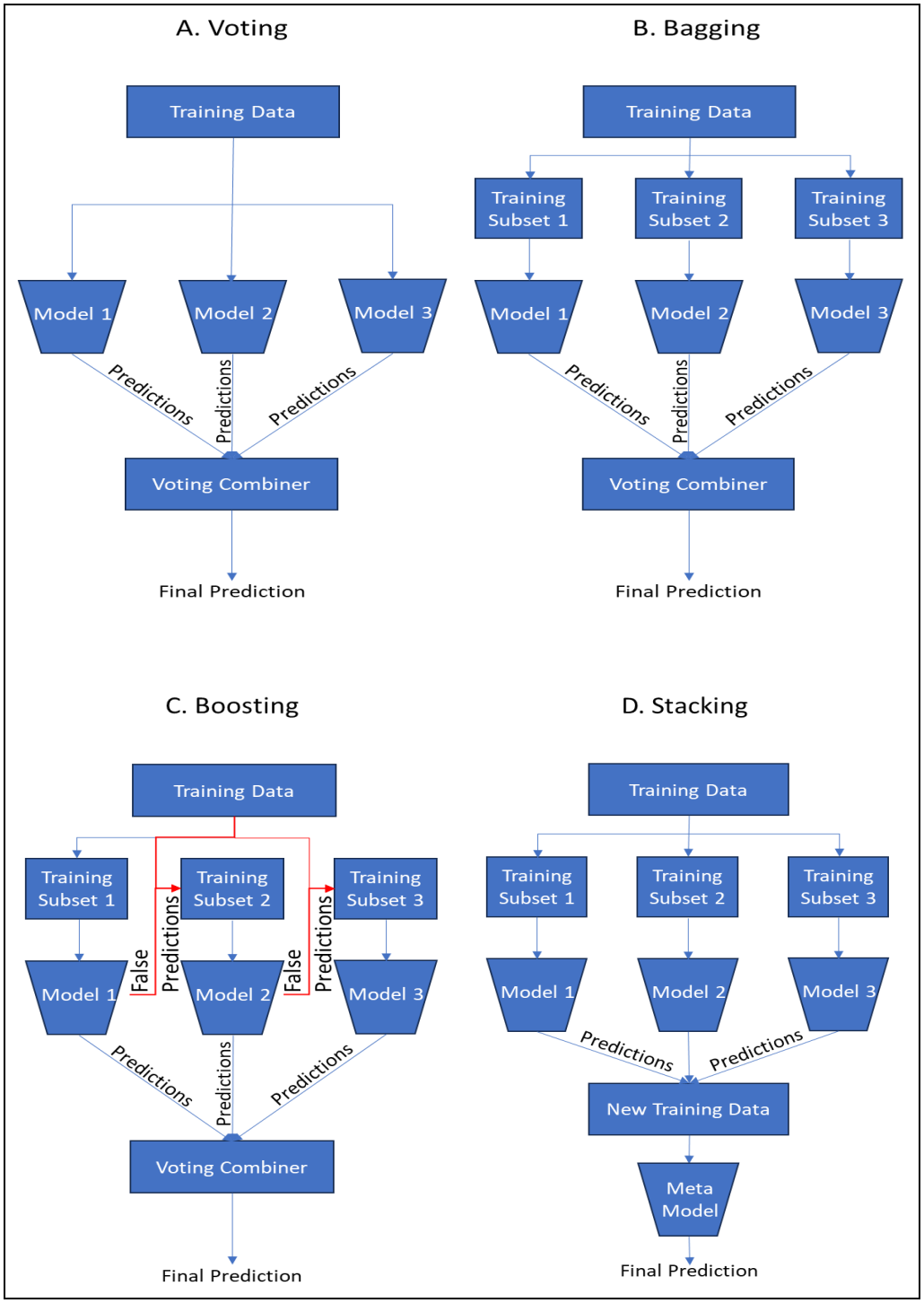

Figure 7 *Ensemble learning algorithms. A) Voting, B) Bagging, C) Boosting, D) Stacking*

Boosting [60] was firstly designed for classification. This algorithm uses three classifiers. The first classifier is trained with a random subset of training datasets. The second classifier uses two subsets, the one derived from the correct predictions of the first classifier and the second from the misclassification instances. The third classifier is trained with data that the two first classifiers disagree. For the final prediction majority voting is utilized. Many variations of boosting algorithms have been proposed through the years, such as Adaboost, AdaboostM1, AdaboostM2 and AdaBoost regression (AdaboostR) [56]. **AdaBoost regression**

is a meta-estimator that creates multiple copies of a base estimator (for instance, Decision Tree regressor already fitted on the original dataset). Each copy adjusts its weights according to the prediction error of the base estimator and new predictions can be performed by inferring the ensemble of these models [53].

Stacking [61] use a two-stage learning algorithm to make predictions. In the first stage different subset of training dataset is used to train each classifier, then, in the second stage a metaclassifier is trained in a dataset that consists of the predictions from the first stage classifiers. After that the first stage classifiers are discarded and new classifiers trained on the entire dataset are taking their place. Stacking can also be used for regression purposes. **Stacking regressor** employs the outputs of the individual selected estimators and uses them as input in an additional regression model to estimate the final output [53].

Despite the enhanced accuracy that ensemble algorithms introduce, they may also provide solutions to a variety of machine learning problems. Online learning, data fusion, concept drift, and confidence estimation is some of the areas that ensemble learning can be used with spectacular results [56].

2.5 Federated Learning

Federated learning, is a collaborative ML method that guarantees data privacy and isolation. In FL, models are trained with local data and learned parameters are aggregated to a central server that holds global model parameters. The central server after the aggregation communicates the aggregated parameters to the local model preserving data privacy and security. FL application consist of a central server (server), that holds a global model and decentralized devices (clients), that hold the local models. The main process involves the following steps: 1) Initialization Step, 2) Model parameters distribution, 3) Local model training, 4) Server Update, 5) Aggregation. At the initialization step, the global model is initialized on the central server. In the next step, the model parameters are distributed to the clients for local training. Each client trains the local model on its local data. After the training step, each model sends the updated model parameters to the server for aggregation without sharing the raw data, preserving data security and privacy. Finally, the server aggregates the model updates, from all the clients, to update the global model. This procedure is repeated iteratively to further improve the global model. **Figure 8** depicts FL general framework highlighting the steps involved in FL.

Throughout the years many aggregation strategies have been proposed such as, Federated Stochastic Gradient Descent (FedSGD**),** Federated Averaging (FedAvg), Federated Averaging with Momentum (FedAvgM) and Federated learning with Proximal term (FedProx).

FedSGD [62] is s classic approach for implementing the FL aggregation strategy, typically used as a baseline method in FL studies. According to this method, each local FL participant computes its gradient based on set of training samples and transmits the computed gradient towards the cloud server. Once all gradients from all FL participants have been collected by the cloud server, they are averaged proportionally to the number of training samples on each participant and, then, sent back to all FL participants. Note that, for each batch of training data, FL participants should compute the gradient and, once the whole training dataset has been accessed, they forward the gradients towards the cloud. Formally, the procedure unfolds

as follows: each FL participant $i = 1,2,\dots,N$ (where *N* is the total number of FL participants) selects a random data batch with size *B* of its local dataset and passes it to its local model. We let the local dataset of FL participant i having a size of Di samples, whereas the total number of samples across all FL participants is *D*. The feed-forward passing steps result in an epoch loss function computation (i.e. the error between actual and predicted values of the batch) at FL participant i, which can be notated as $Li(wi)$, where wi is the model weight matrix of FL participant i. Mathematically, the epoch loss function computed by FL participant i at time slot t can be given as:

$$L_i^t(w_i^t) = \frac{B}{D_i} \cdot \sum_{j}^{D_i/B} \sum_{k \epsilon D_i} l_{j,k}^t\left(w_i^t\right)$$

(4)

where $lj, k\left(w_i^t\right)$ denotes the loss function calculated for sample $k \epsilon D_i$ belonging in the batch j of size B using the model weights w_i^t of FL participant i. Briefly, *(4)* computes the average loss amongst all batches that are selected in a single epoch (i.e. equal to the epoch loss of FL participant i). Then, the gradient of FL participant i at time slot t can be written as:

$$g_i^t = \nabla L_i^{t(w_i^t)}$$

(5)

where $\nabla(\cdot)$ stands for the gradient calculation. Finally, the weight matrix extracted by the cloud and received by all FL participants at the next time slot is the following:

$$w^{t+1} = w^t - \alpha \sum_{i=1}^{N} \frac{D_i}{D} \; g_i^t$$

(6)

where w^t is the global model weights matrix at time slot t and α is a hyperparameter called learning rate.

FedAvg [63]: This aggregation approach is known as Federated Averaging, and is literally an immediate extension of the FedSGD to reduce to number of vertical communication rounds between the FL participants and the cloud. The idea behind FedAvg is to enforce each FL participant to locally compute the weights upon performing multiple local training rounds and, then, upload the resulting weights towards the cloud. Finally, the cloud computes a weighted mean across all local model weights and send it back to all FL participants. Using identical notations as previously, we further define E as the number of the local training epochs performed by the FL participants before they send the local model updates. We also use the symbol t (or t') to denote the epoch ID counter (or the time slot for global aggregation). Thus, at every time slot t when an epoch ends, each FL participant i calculates its local weights as:

$$w_i^{t+1} = w_i^t - \alpha \cdot g_i^t$$

(7)

where g_i^t is given by *(5)*. Upon completion of E epochs for updating the local weights, assume at time slot t', FL participants transmit their matrices $w_i^{t'}$ to the cloud, with the latter returning back to all FL participants the new global model with the following weights:

$$w^{t'+1} = \sum_{i=1}^{N} \frac{D_i}{D} w_i^{t'} \tag{8}$$

where $w^{\{t'\}}$ is the global model weights matrix at time slot t' and t' is the time slot of the aggregation step and is an integer multiplier of E (i.e. the local weights are uploaded for aggregation every E epochs).

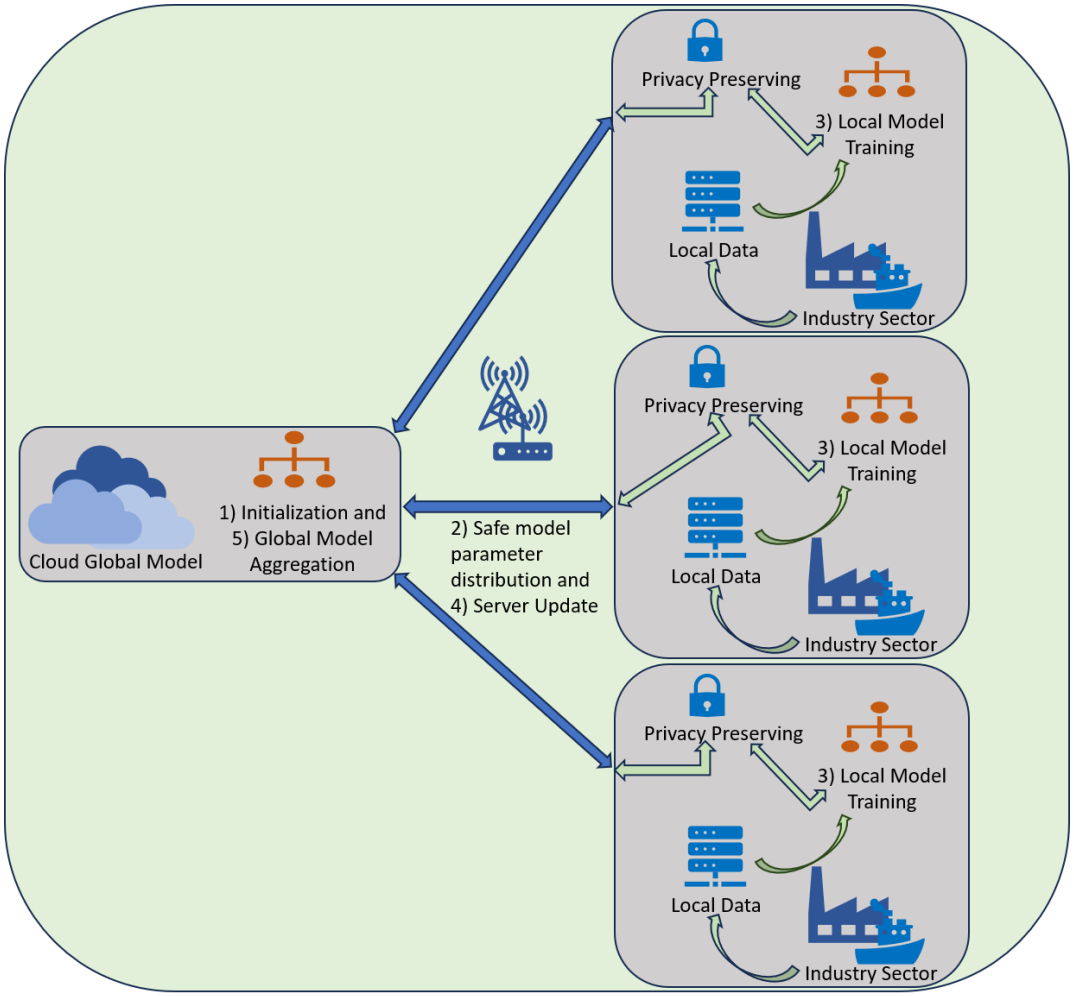

Figure 8 *Federated learning general framework.*

FedAvgM [64]: This policy deals with non-independent and identically distributed (non-IID) data across FL participants and is known as Federated Averaging with Server Momentum.

FedAvgM is appropriate in cases that distributions and statistics differ significantly amongst local datasets, as well as to guarantee convergence, and it tries to mitigate the errors introduced by the data heterogeneity via server momentum. The idea behind FedAvgM is that each FL participant i updates its local model update $w_i^{t'}$ at time slot t'. Note that local model updates follow the rule given in *(7)*. Cloud server computes the new momentum according to the following formula:

$$m^{\text{new}} = \beta m^{\text{old}} + \sum_{i=1}^{N} \frac{D_i}{D} w_i^{t'} \tag{9}$$

where m^{new} (or m^{old}) is the next (or previous) value of the momentum, and β is the momentum constant. Finally, the updated global model can be derived as:

$$w^{t'+1} = w^{t'} - m^{\text{new}} \tag{10}$$

where $w^{t'}$ is the global model weights matrix at time slot t', which updated every E local training epochs as in typical FedAvg.

FedProx [65]: This policy, termed as FL with proximal term, is widely used for implementing FL under statistical heterogeneity and non-IID data across the local FL participants. FedProx can be viewed as an empirical generalization of FedAvg, which makes some *modifications* to address heterogeneity of data. The learning is again achieved in rounds, where at each round, the server randomly selects a subset of clients and sends them the current global model. At time slot t', server sends to the selected FL participants the global model $w^{t'}$, which in turn perform E training epochs to minimize a regularized loss function, given by the formula below:

$$\min_{w_i^{t'}} \left\{ L_i^t\left(w_i^{t'}\right) - \frac{\mu}{2} \left\| w_i^{t'} - w^{t'} \right\|^2 \right\} \tag{11}$$

where $w_i^{t'}$ (or $w^{t'}$) are the local model parameters of FL participant i (or the global model parameters), and μ is the regularization constant. Noteworthy, FedProx is the same as FedAvg for $\mu = 0$. As implied by *(11)*, FedProx properly adjusts the local weights of the selected agents, so as to ensure minimization of the regularized function, which is a function of the

global weights. After performing local training rounds, the local weights are updated towards the cloud, where the global model is derived by applying *(8)*, as in FedAvg.

2.6 Model training

Training of machine learning models is the process of training a machine learning algorithm on a training dataset in order to generalize well in new unseen data. Through the training, ML algorithms learn to recognize relationships and complex patterns between features and target variables. Training is the building block of machine learning and several steps should be completed in order to build an effective ML model. In the beginning, a sufficient amount of data should be gathered for training and testing purposes. Next, is the preprocessing step where transformation and cleaning techniques are applied to the gathered data to improve the model performance. After that, the appropriate ML algorithm that best suits the problem must be selected. For instance, for a classification problem an algorithm such as SVC and logistic regression should be selected. Also, several tryouts should be performed in order to choose the best-performing algorithm against several suited ML algorithms. Then, the training step follows. The dataset collected is separated into training set, evaluation set, and testing set. The training set is used during training where the ML algorithm changes its parameters (e.g. learning rate, weights, and bias) in order to achieve a good performance against the evaluation set of data. Finally, when the model achieves a good performance based on appropriate metrics, it is tested against the testing set. A good performance in the final step indicates the success of the training process and a good derived model [66]. In **Figure 9** a basic model training procedure is depicted.

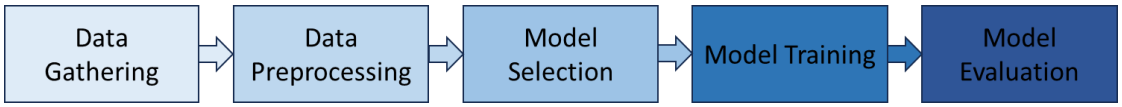

Figure 9 *Basic model training procedure.*

In the next sections more details about the ML training procedure are presented.

2.6.1 Data gathering

Data collecting is a foundation for machine learning. The goal of data collection is to acquire as much data as possible in order to sufficient train an ML algorithm. Data are collected in a production setting using monitoring devices such as sensors and cameras. After collecting, data is stored in databases or on the cloud for later processing. In order to be able to use these data for training purposes, substantial volume of quality data should be selected. Additionally, feature selection, extraction, and data labeling should be accomplished by an expert. This is a time-consuming operation that serves as a bottleneck for machine learning adaption. Thus, numerous approaches for data gathering have been proposed in the literature. Author in [67] highlights three main methods for data collection, namely data acquisition techniques, data labeling techniques for available data and using existing data or

existing models. Data acquisition focus on finding datasets in order to train a model. Three methods are generally used for data acquisition, data discovery, data augmentation and data generation[67].

- Data discovery has two steps, data sharing and data searching. In the sharing step data collected are indexed and shared in the web while in the second step interested members can consume the shared datasets from the web.
- Data augmentation of existing data with external data is another method for data collection. Adding pre-trained embeddings, entity augmentation techniques and data integration are mainly used for this purpose.
- Data generation is used to generate data in manual or automated way. Crowdsourcing uses humans for manual collection of data and dataset construction. The automated way uses algorithms to generate synthetic datasets. One such algorithm is generative adversarial network (GAN) for tabular data from [68] which is used to generate tabular data.

Next, data labeling method uses machine learning techniques in order to label a dataset. Semi supervised and unsupervised learning techniques are used for this purpose. Crowd sourcing techniques are also utilized. An ML model selects the most informative data to provide to experts for manual labeling [67]. Finaly, using existing data and model's method focuses on improving existing resources such as data and models. For data improving data cleaning and re-labeling techniques are used to improve data. For model improvement the goal is to make the model more robust to noise and bias. Also transfer learning can be utilized to enchase the model's performance [67].

2.6.2 Data preprocessing

Data preprocessing is a necessary step for effective generalization of ML algorithms. Moreover, data quality has a significant impact on ML model training. One of the main problems that should be encountered during training is noisy data. Noisy data may affect the performance of an ML algorithm forcing to misclassification. Noise data distinguished in noise on features and noise in targets. Noise data can be handled with two methods, the filters methos and the use of robust learners. Filter methods are used to detect noise data and discarded from the dataset. This method has a significant drawback of missing important information. On the other hand, training robust learners such as ensembles, Adaboost and decision trees with the utilization of appropriate splitting and pruning [69].

Outliers is another problem that should be addressed during preprocessing. Outlier detection is used to detect instances that significantly deviate from the other data. Many approaches have been proposed from the literature for outlier detection, namely density-based, statistical-based, distance-based, clustering-based, ensemble based, and learning-based approaches [70].

In real world dataset it is common to face missing values or even a large number of missing data. Missing values can certainly affect the performance of many algorithms. Missing data is a common problem in datasets especially in real world datasets. Many methods have been proposed to deal with missing data such as [71], [72], [73], [74]: 1) Deleting instances with missing values, 2) Fill with the most common feature value, 3) Fill with the mean value of the

feature, 4) Use the prediction of an ML algorithm trained in all the other features and predicting the feature with missing values, 5) Multiple imputation, this method splits the data in several partitions and calculates a mean value to fill the missing ones.

Another problem that arises is the imbalanced data. For instance, it is common in a classification problem the majority of the data to belong in the first class and only few instances to belong in the second class. This may lead to a poor generalization against the second class. Imbalanced data drives the ML algorithms to generalize well in majority class but fail in the minority. Therefore, many methods have been introduced in order to deal with imbalanced data, such as sampling methods, cost sensitive methods and kernel based methods [75]. Sampling Methods use some modifications in the original dataset in order to balance the classes. The most common methods are oversampling and undersampling. Oversampling, selects a subset of minority class and a randomly chosen subset of the majority class and then augments the data and replace the original data samples. On the other hand, undersampling randomly choose to remove data from the majority class. A widely used method is synthetic minority over-sampling technique (SMOTE) [76]. SMOTE uses minority class to create artificial data. Next, cost-sensitive methods use a cost function to represent the cost of misclassifying the minority class. Then they use this cost function in order to create weights in training instances or create cost sensitive classifiers in an ensemble. Finaly, kernel-based are methods where a kernel-based algorithm, that is robust to imbalanced data, is used. A paradigm of such an algorithm is SVM explained in section 2.2.

Furthermore, feature selection and instance selection strategies should be utilized in order to use the most informative ones and minimize the complexity of the model. Feature selection is used in order to remove irrelevant features and use relevant features that efficiently can train the ML algorithm and provide good predictions. Feature selection can be achieved with the use of ranking method. In this method the relevance of data with the target is measured with Pearson correlation coefficient or with Shannon's definition for entropy. A value that represents the correlation is calculated from one of these criteria and a threshold is used to classify the feature as relevant or irrelevant. Finally, the irrelevant features are ignored. Another method for feature selection is the wrapper method where sequential and heuristic algorithms select a subset of features to train an algorithm and measure the performance for each subset. The best performing subset is the most informative one [77].

Other preprocessing technique include discretization, data normalization and feature construction [74], [78], [79]. Data discretization is used when we have a wide range of continuous values. Training an algorithm in such cases is a time consuming and inefficient procedure. Data discretization uses algorithms to discretize data in equal spaces in a given range [78]. Next, Data normalization is used when the data have a huge difference between the minimum value and the maximum. A common data normalization algorithm called min-max normalization, is used to transform the data in a range between [0,1]. Other normalization algorithms are min-max normalization with data in range [a, b] where a and b can take any given values, z-score and unit length scaling [78]. Finaly, feature construction is used for new feature generation/transformation from the existing features. The new features can lead in more accurate ML algorithms [74]. The features can be constructed with the contribution of an expert or with the use of feature construction algorithms [80], [81], [82]

2.6.3 Model training and evaluation

In the training phase, we firstly choose the appropriate models to train for the specific problem e.g. classification, regression. Then, during evaluation we choose the best algorithm based on performance metrics. Many evaluation procedures have been proposed in the literature and they are briefly described in this section [83], [84]. Hold-out method uses a single split in the datasets to separate the training and the testing dataset. The ML algorithms are trained in the training dataset and evaluated in the testing dataset. Next, stratification also splits the dataset in training and testing with only difference that is trying to have identical distribution between the two datasets. Finaly, k-fold cross validation uses k splits of the same dataset to train the ML algorithm. Additionally, it is ensured that the whole dataset will be used as testing set arbitrary. Finally, the performance in all testing sets is aggregated. Nevertheless, the performance is dependent of the dataset. Therefore, an additional validation set should be separated from the whole dataset at the beginning and used for performance metrics in order to choose the best algorithm. The three aforementioned evaluation methods are depicted in **Figure 10**.

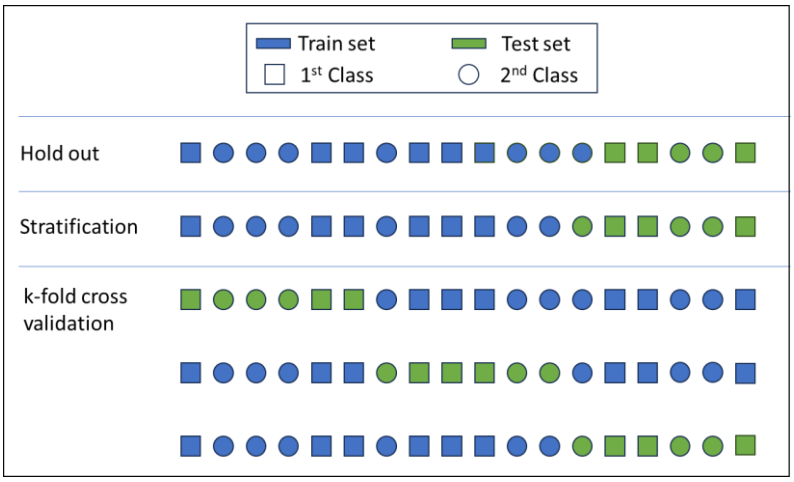

Figure 10 *Validation methods*

2.6.4 Metrics

In ML, metrics are used to measure the performance of ML algorithms and also compare them among others. Different metrics have been proposed for classification and regression problems [83], [84].

Confusion matrix, accuracy, precision, recall, F1-score, area under the receiver operating characteristic curve (ROC AUC) are some of the metrics utilized for classification. **Confusion matrix** summarizes the results from a classification problem. In binary classification confusion matrix consist of four elements divided in two rows and two columns. Rows represent the predicted values and columns the true values. **Figure 11** depicts the confusion matrix. **Accuracy** is used to measure the performance of an algorithm. Accuracy is the ratio of correct predicted instance to the total instances. **Sensitivity** (recall) is used when minimization of false

negatives is critical. Recall reveals the ability of the classifier to find all the positive samples. It can take values between [0,1] and the best possible value is one. **Specificity** measures the model performance on predicting negative instances. It can take values between [0,1] and the best possible value is one. **Positive predictive value** (PPV) is used when we want to minimize false positives. PPV also called precision. PPV can take values between [0,1]. A value near 1 indicates a good model. **Negative predictive value** (NPV) indicates the ability of the classifier to correctly classify negative samples. It can take values between [0,1] and the best possible value is one. **Balanced accuracy** (BA) mostly used with datasets with imbalanced classes. If the algorithm has a good performance BA just reveals the accuracy. **Area under the receiver operating characteristic curve** (ROC AUC) is a graphical representation of true positive rate (TPR) against false positive rate (FPR). **F1-score** is performance metric that calculates the harmonic mean of recall and precision. All metrics can be used for binary classification problems. For multiclass classification some modification should be made in order to aggregate the binary classification metrics. The most popular method is treating each class in turn as positive examples and the others as negative examples, then all measurements are aggregated [83].

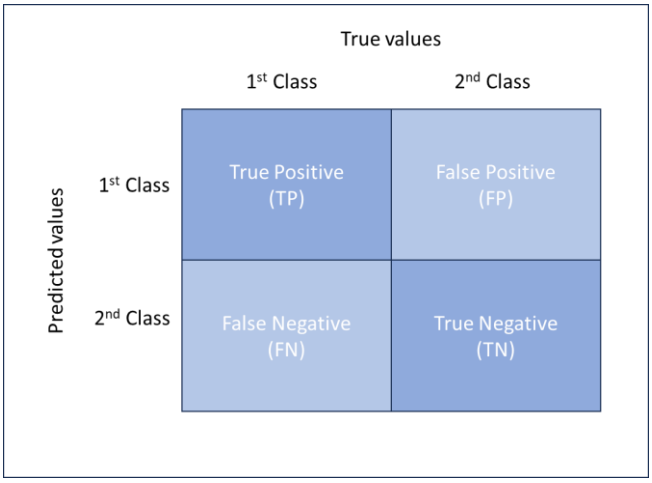

Figure 11 *Confusion matrix.* [1]

Furthermore, R2 -score, Mean Square Error (MSE), Mean Absolute Error (MAE), Root Mean Squared Error (RMSE) and Mean Absolute Percentage Error (MAPE) are mainly used as regression metrics [83], [84]. R2 -score reveals how well a regression model fits to the data. R2-score can take values between [0,1]. A value near 1 indicates a good model. MSE is the average squared difference between the estimated values and true value.it is a loss measure and can take values between [0, +inf]. Small value close to zero indicates good performance. MAE is calculated as the sum of absolute errors between real and predicted values divided by the sample size n. It ranges between [0, +inf]. As in MSE a smaller value indicates a good performance. RMSE calculates the deviation of prediction from the target variable. It is the square root of the mean of the squared errors and can take values and between [0, +inf].

[1] *True positive (TP) is the number of values that belong to the 1st class and the algorithms predict that they belong to the 1st class. True negative (TN) is the number of values that belong to the 2nd class and the algorithms predict that they belong to the 2nd class. False positive (FP) is the number of values that belong to the 2nd class and the algorithms predict that they belong to the 1st class. False negative (FN) is the number of values that belong to the 1st class and the algorithms predict that they belong to the 2nd class.*

Values near zero indicate a good performance. MAPE also called the mean absolute percentage deviation (MAPD). It measures accuracy as difference of percentage error and can take values in range [0, +inf]. Zero value indicates a perfect accuracy.

2.7 Machine learning frameworks

Machine learning frameworks are components bult in order to support machine learning models. They include prebuilt components of machine learning models, metrics and evaluation procedures. Thus, they make ML more efficient and convenient for researchers and developers. One of the most popular ML frameworks is scikit-learn [85]. Scikit-learn is implemented in python and consist of many ML algorithms for classification, regression and clustering. In its repository have preprocessing tools for dimensionality reduction, feature extraction, feature selection, scaling, discretization and normalization. Also, provides tools for artificial dataset generation. In addition, imbalanced learn is an ML framework for imbalanced data that works on top of scikit-learn. Next, TensorFlow [86] is an end-to-end framework for machine learning. It utilizes many tools, datasets, models and provides visualization and deployment tools. Keras is a deep learning framework running on top of TensorFlow. It has got a big library of deep learning models and guides that can help everyone to build their own deep learning algorithms. Scikit-multiflow [87], is a machine learning framework for stream learning, incremental learning and drift detection. Also, it provides algorithms for artificial dataset generation for stream learning and drift detection. Frouros [88] is another python library for drift detection. In recent years federated learning has gather the attention of researchers and many frameworks have been introduced such as TensorFlow Federated, Flower [89], FATE, PySyft and OpenFL [90]. In **Table 1** machine learning frameworks with a short description are listed.

Table 1 *Machine learning frameworks*

Framework	Description
Scikit-learn	ML Framework
Scikit-multiflow	ML Framework for stream learning and drift detection
TensorFlow	ML Framework
Keras	Deep learning framework
Imbalanced learn	ML framework for imbalanced data
Frouros	Drift detection framework
TensorFlow Federated	FL framework
Flower	FL framework
Fate	FL framework
PySyft	FL framework
OpenFL	FL framework

2.8 Machine Learning in Industry 4.0

As part of the Industry 4.0 paradigm, machine learning has a significant impact on the manufacturing industry. ML has the potential to improve many facets of Industry 4.0, including CPPS and IoT, as well as Cloud/Edge technologies, DT, and HMI. Machine learning methods have the capability to enhance manufacturing operations through the analysis of extensive data sets, enabling the identification of inefficiencies and the proposal of adjustments to enhance productivity and decrease expenses. The application of machine learning detection and prediction capabilities at all levels of an industrial organization can significantly improve market position and manufacturing efficiency, reduce costs and strengthen industry sustainability, increase customer satisfaction, and minimize the environmental footprint [91].

At the level of supply chain management, Machine Learning (ML) can be employed to analyze historical data and anticipate future trends. By extracting models from extensive input data over time, ML enhances demand forecasting in the supply chain. This leads to a decrease in product shortages, overstocks, and waste, while also facilitating the optimization of storage capacity, thereby streamlining inventory levels and reducing costs. Furthermore, ML can process and correlate vast volumes of data, enabling a more comprehensive understanding and prediction of the impacts of external events, which can be applied in Risk Management (RM). AI-based models can effectively be implemented on routing problems to ensure timely deliveries and enhance the transportation of products in warehouses. Finally, ML can optimally predict and suggest supplier selection, thereby improving product quality and reducing costs [92].

Also, ML has a great potential in forecasting customer behavior, which can help companies optimize their marketing strategies. By considering variables coming from customers such as their satisfaction, their needs and the way they use the products ML can contribute in the designing processes of personalized products. Thus, improve customer satisfaction [93]. Furthermore, recent developments in AI, particularly ChatGPT, can provide 24/7 customer support and enhance the overall experience by automating responses. Also, ChatGPT can provide workers with powerful tools, such as real-time training, task automation, and enhanced human-robot collaboration and human-machine interaction [94]. Thus, ML and CPS can provide a symbiotic environment that focuses on the effective collaboration and cooperation between humans and machines / collaborative robots (cobots) for handling combined task elements, while communicating over the IIoT [17], [95].

In the context of Industry 4.0, cyber-security emerges as a critical concern across various fields. ML provides safety, reliability and performance. Machine learning algorithms can analyze network traffic and detect anomalies to identify potential security threats in industrial systems [96].

In the context of Industry 4.0 and Zero Defect Manufacturing, machine learning assumes a significant role in ensuring product quality. ML has the capability to analyze real-time data from sensors and machine vision systems, enabling the detection of anomalies, defects or deviations in manufactured products. Furthermore, ML's ability to predict quality issues at early stages of production facilitates prompt implementation of corrective measures and preventive strategies. Consequently, ML contributes to minimizing product wastage, reducing costs, and mitigating environmental impact [97].

To maintain machines running at optimal performance, an efficient maintenance schedule is needed. ML-based predictive maintenance can accurately detect anomalies, faults, and defects, providing indications of machine state conditions and predictions for the remaining useful life of machines, parts and tools. This enhances prognostic knowledge and leads to effective maintenance strategies, thus reducing machine downtime, maintenance costs, and defective products [98].

3. AI-driven Zero Defect Manufacturing

3.1 Industry 4.0, Artificial Intelligence and Machine Learning

In recent times, the field of Industry 4.0 has seen numerous valuable contributions. The authors in Reference [99], provide an overview of state-of-the-art technologies involved in the evolution of manufacturing to I4 era. More particularly the present technologies in areas such as extended reality and additive manufacturing as well as BDA, IoT, and AI. In addition, they focus their research in AI involvement on different aspects of manufacturing cycle. Research gaps were tried to found out and pathway to research advancement were revealed. **Figure 12** depicts the I4 technologies researched in this section.

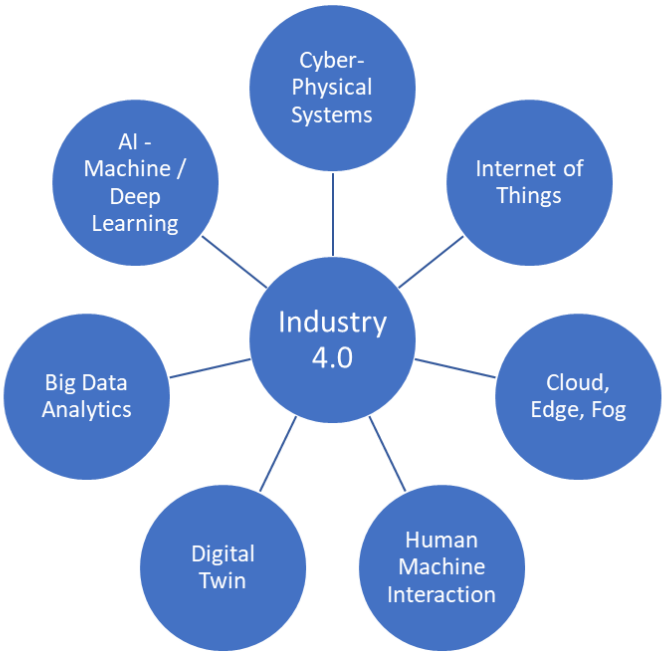

***Figure 12** Industry 4.0 technologies.*

The survey in Reference [13] presented an overview of CPS in industrial IoT settings, giving in detail a relevant architecture, enabling the control of systems and processes, while ML was presented as a promising solution for CPS. Then, the authors of Reference [25] provided a thorough analysis of the role of BDA in IoT and a taxonomy of ML algorithms in BDA-enabled applications, such as self-maintenance, self-prediction and self-configuration. Furthermore, an overview of ML for manufacturing systems was given in Reference [20], discussing implementation issues, as well as the benefits and drawbacks of different categories of ML algorithms, including supervised, unsupervised and RL algorithms, concluding that currently, supervised ML techniques are most appropriate for smart manufacturing. Next, the work in Reference [38] discussed the use of BDA in the process industry and its dependency on ML. More specifically, several supervised and unsupervised ML methods were presented,

suggesting that semi-supervised solutions have the potential to provide better trade-offs among implementation complexity, performance and data requirements. In addition, a detailed survey of big data structure and analytic techniques for CPS were examined in Reference [40]. In this context, descriptive (clustering, correlation) and predictive analytic techniques were evaluated. In the area of machining processes, the impact of ML solutions was presented in Reference [39]. Several machining cases were listed and a brief presentation of ML-based tool wear monitoring and prediction was included, outlining its potential. Regarding the field of smart manufacturing, the role of DL was investigated in Reference [42]. In greater detail, the evolution of DL and its advantages in processing heterogeneous and highly complex data, compared to conventional ML solutions, as well as various DL-based computational methods for improving manufacturing processes were highlighted. Next, the article in Reference [100] focused on three areas for AI/ML application, that is, faster convergence in environments with partial and intermediate AI/ML integration with a single optimization iteration, deep active learning towards optimal topologies through multiple iterations without neglecting the benefits of exploration-exploitation trade-offs and finally, knowledge-based assistants for improved human–machine interaction (HMI) by translating the optimal topology to a concept design, leveraging relevant metadata of historical expert decision [17].

Next, the authors of Reference [101] review the engagement of ML and IoT in the manufacturing sector. State-of-the-art tools, protocols, ML algorithms and technologies are provided and analyzed. Through their work, they were able to underline drawbacks and propose future research areas in the context of Industry 4.0. Then, authors in [102] make an access of the impact of AI in Human role in the context of Industry 4.0. Through their extensive literature investigation, they reveal benefits that AI will grand to humans such as the improved efficiency and the better decision making. On the other hand, security risks privacy concerns, and legal considerations may arise. Thus, further research directions are suggested, concerning security risks and companies' successfulness. Reference [93], provides a comprehensive literature review in Production Planning and Control (PPC) assisted by ML methods. Through their work they present the best performing ML models, techniques and tools used in the manufacturing sector for production planning and control. Also, they provide a thorough presentation of I4 attributes and conducted use cases from contemporary publications. In addition, authors in [103] investigate how I4 technologies restructure the quality of manufacturing products. In their work, they highlight production and quality issues arising through the manufacturing process. In addition, they present machine learning methods that have been utilized from resent works in different manufacturing processes and analyze the use of ML in anomaly detection. Also, they review sensors used in the manufacturing for automation and data generation purposes.

The study in Reference [104] examined several works on ML-based optimization in the areas of product quality and process improvement. The interdependencies among the used data, the amount of data, the ML algorithms, the adopted optimizers and the specific production problems were discussed, concluding that often, their correlation is not carefully investigated, thus leading to highly complex models, being trained with insufficient amounts of data, exhibiting overfitting and low interpretability. Next, the authors of Reference [105] introduced research strategies for industrial big data collection in intelligent environments, ontology modeling and deduction methods, as well as predictive diagnostic solutions for production lines, relying on deep neural network (DNN) and DL in cloud-assisted devices with self-organized reconfiguration capabilities. An overview on anomaly detection on industrial

Wireless Sensor Networks (WSN) was given in Reference [44], comparing ML methods, including K-nearest neighbor (KNN), SVM, ANNs and hybrid schemes. Next, the study in Reference [106] offered a deep insight on data mining techniques for production management, focusing on production scheduling, quality improvement, defect analysis and fault diagnosis. Still, ML-based solutions were not thoroughly presented, thus leaving a gap in the literature [17].

Then, the authors of Reference [107], provide a thorough review about artificial intelligence driven DT in the Industry 4.0 era. They discuss, general developments and AI utilization regarding different steps of manufacturing lifecycle such as management, production and material steps. Also, they provide relevant works involving ML and DL methods for production planning, production control, quality control, condition monitoring and PdM. Finally, they discuss gaps and provide future research directions in the area of AI-driven DT. Finally, authors in Reference [108] provides a comprehensive literature review in order to present the utilization of machine learning methods in manufacturing processes. They mainly focus on process monitoring quality inspection, process planning/scheduling and fault detection. In their work, they also discuss about ML utilized in different processes of the industry by discussing ML algorithms DL algorithms and ensembles for supervised and unsupervised learning problems. They conclude their work by providing the benefits that ML brings in the manufacturing field they discuss the drawbacks and propose future research directions.

Table 2 provides the summaries of the relevant surveys and possible AI/ML-based solutions that they might discuss within the Industry 4.0 area.

Table 2 *Relevant surveys and tutorials on AI/ML and Industry 4.0*

Reference	Short description	Scope of ML in Industry 4.0
Sahoo *et al*., 2022 [99]	Technologies involved in Smart manufacturing	AI, IoT, CPS in Smart manufacturing for quality and maintenance
Xu H. *et al*., 2018 [13]	CPS aspects in IIoT	IIoT, cloud/edge computing, sensing and decision-making
Rahman *et al*., 2023 [101]	ML and IoT in Industry 4.0	ML, IoT tools, protocols, algorithms
Rehman *et al*., 2018 [25]	The role of BDA in IIoT	ML-based BDA for self-organization and prediction
Wuest *et al*., 2016 [20]	ML's role in manufacturing	Focus on supervised learning for fault diagnosis
Ge *et al*., 2017 [27]	Data mining and BDA in the process industry	Supervised, unsupervised and semi-supervised learning for BDA
Kim *et al*., 2018 [39]	Overview of ML in machining processes	Brief summary on tool-wear and condition monitoring
Xu L. D. *et al*., 2018 [40]	Interaction of big data and CPS in Industry 4.0	Nothing in particular
Wang *et al*., 2018 [42]	DL for smart manufacturing	Brief overview of fault detection and prediction

Table 2 *Relevant surveys and tutorials on AI/ML and Industry 4.0 (Cont.)*

Reference	Short description	Scope of ML in Industry 4.0
Ramotsoela *et al*., 2018 [44]	Overview of anomaly detection in IIoT	Presentation of implementation aspects
Aggour *et al*., 2019 [100]	AI/ML use cases for manufacturing and inspection	Presentation of specific applications for component inspection and life prediction
Murugesan *et al*., 2023 [102]	AI impact on humans	Nothing in particular
Usuga *et al*., 2020 [93]	ML for production planning	Focus on production planning and control
Weichert *et al*., 2019 [104]	ML for process optimization	Focus on production quality, brief overview of fault detection and prediction
Md Abdul *et al*., 2022 [103]	ML, Anomaly detection, Sensors	Focus on quality inspection
Xu X. *et al*., 2017 [105]	Strategies for BDA, ontology modeling and deduction	Brief overview of DL for predictive diagnostics
Huang *et a*l., 2021 [107]	DT review	ML/DL methods for various level of Industry 4.0
Cheng *et al*., 2017 [106]	Data mining and BDA for production management	Nothing in particular
Dogan *et al*., 2021 [108]	ML for manufacturing processes	Review of ML/DL/ensembles for supervised and unsupervised learning problems

3.2 Zero Defects in Industrial Manufacturing

In the last few years zero defect manufacturing gains more and more attention, surpassing the interests of researchers. Authors in [21], provide a comprehensive evaluation of the state of the art in ZDM. The four main ZDM strategies are discussed: detection, repair, prediction, and prevention. Furthermore, the limitations of present study are thoroughly examined in light of literature. Finally, new research directions were provided. Furthermore, the authors in reference [22] present a literature overview on the most recent achievements in the field of ZDM. In addition, a review of the state of the art and important enabler technologies is provided. Furthermore, the authors underline that defect prevention method is significantly more important than defect detection, as it prevents faults in the first place and avoids rework. Finally, future research prospects and directions are discussed.

Next, the paper in reference [24] provides a literature analysis that focuses on four primary strategies: detection, repair, prediction, and prevention. In addition, a bibliometric analysis is provided to highlight the importance of ZDM. Finally, they suggest eight bottleneck issues that can be solved in the future. Furthermore, a holistic framework addressing ZDM while taking into account the critical components of a manufacturing company is provided by

[109]. A survey on current ZDM implementation is presented by [110]. Authors investigate the impact of ZDM strategies and provide current drawbacks and limitations. Furthermore, enabled technologies are depicted and future research directions are highlighted. A practical guide providing directions for integration of new or existing manufacturing system into ZDM is given by [111]. In addition, a use case in a real semiconductor manufacturing process is provided for validation. Authors in [112] provide a comprehensive framework through a ZDM platform that combines advanced information and communication technologies (ICT), AI and inspection tools.

In [113], author provides a methodology for development and implementation of Digital Twin (DiTw), in order to model a production system based on ZDM principles. The author concludes that more research should be conducted, focusing on the generalization of designed tools and methodologies. Next, [114] provides an overview of scheduling in the ZDM environment, guided by DiTw. According to the authors, more research should be conducted in this area. Next, an ontology-based DiTw tool that utilizes detect and repair strategies is presented by [115]. Authors develop an Ontology for Reliability-centered Maintenance (ORMA) tool in order to create a DiTw. According to the author, ORMA+ shows promising results but more research should be done in the decision support system.

Authors of [116], give a literature assessment on the human-centric ZDM. They categorize their findings into three primary roles: managers, engineers, and operators, and highlight the crucial function of managers, as well as the importance of future technologies, such as extended reality. Next, paper [117] conducts a systematic literature review on the human role in ZDM. By introducing a framework that emphasizes the importance of human behavior and decision making in ZDM, they propose research avenues for efficiently training humans in the ZDM mentality.

In [118], the authors suggest a cloud-enabled ZDM strategy. In order to enhance the production process and the end product in a real industrial setting, they use and analyze three primary steps: gather, analyze, and optimize the loop. Furthermore, authors of [119] provide a novel hybrid cyber-physical production system (CPPS) for ZDM. This work also provides the foundation for facilitating the smooth transition from traditional production systems to I4 and ZDM.

In Table 3 provides the summaries of the relevant surveys, implementation and key technologies solutions that they might discuss within the ZDM area.

Table 3 *ZDM design and enabled technologies*

Reference	Short Description
Psarommatis *et al., 2020* [21]	ZDM Review
Powell *et al*., 2022 [22]	ZDM Review
Caiazzo *et al., 2022* [24]	ZDM Review

Table 3 ZDM design and enabled technologies (Cont.)

Reference	Short Description
Psarommatis *et al., 2022* [109]	ZDM Framework
Fragapane *et al., 2023* [110]	ZDM Survey
Psarommatis et al., 2023 [111]	ZDM Practical guide
May *et al*., 2019 [112]	ZDM Framework, Platform
Psarommatis, *et al.,* 2021 [113]	DiTw for ZDM
Serrano *et al.,* 2023 [114]	DiTw for scheduling in ZDM
Ghedini *et al.,* 2023 [115]	Ontology-based DiTw for ZDM
Wan *et al.,* 2023 [116]	Human Factor in ZDM
Psarommatis *et al.,* 2023 [117]	Human Factor in ZDM
Leotta *et al.,* 2022 [118]	Cloud
Cacammo *et. al.,* 2021 [119]	CPPS

3.3 Concept Drift

In the era of ZDM, data play an important role. The fast-changing environment of industrial manufacturing generates a huge amount of data. In the era of Industry 4.0, historical data is used for training ML models in order to make predictions on new data. Concept drift arises when the relationships between input features utilized for predictions and the true target variable change over time, resulting in a decline in the performance of the ML model. In **Figure 13**, A) depicts the decision boundary of a well-trained model while B) depicts the decision boundary of the same model with the presence of CD. Thus, data should be monitored and changes in the concept must be detected and processed in order to minimize downtime, defective products and wasted materials. Thus, concept drift detection is a cornerstone for ZDM. Many researches focus on concept drift detection, for instance, in [120], a holistic review is presented regarding the concept drift mechanisms. The authors present a general framework consisting of three stages for concept drift handling. Furthermore, an extensive review of concept drift detection algorithms is formulated. The authors also refer to adaptation methods, evaluation techniques and datasets of different characteristics, concluding that error-rate and data distribution-based drift detectors play a dominant role in concept drift detection [29].

Another extensive review addressing the most-frequently used techniques for concept drift detection in StL is presented in [121]. In this work, the authors categorize concept drift

algorithms as statistical-based, window-based, and ensemble-based. Referring to well established algorithms such as drift DDM, DWM, EDDM, HLFR. In addition, a variety of evaluation methods and metrics is described, highlighting the need for a ground-truth framework with regards to the concept drift detection evaluation. Several drawbacks in datasets and metrics used for evaluation are also illustrated, implying that state-of-the-art drift detectors are partially outdated. Finally, the authors in [122] investigate several methods based on window, weight and ensemble-based approaches of classifiers in the handling of concept drift. Specifically, various drift detection techniques such as DDM, DWM and EDDM are documented. In this context, supervised learning methods have been widely proposed, including SVM, DT and NB [29].

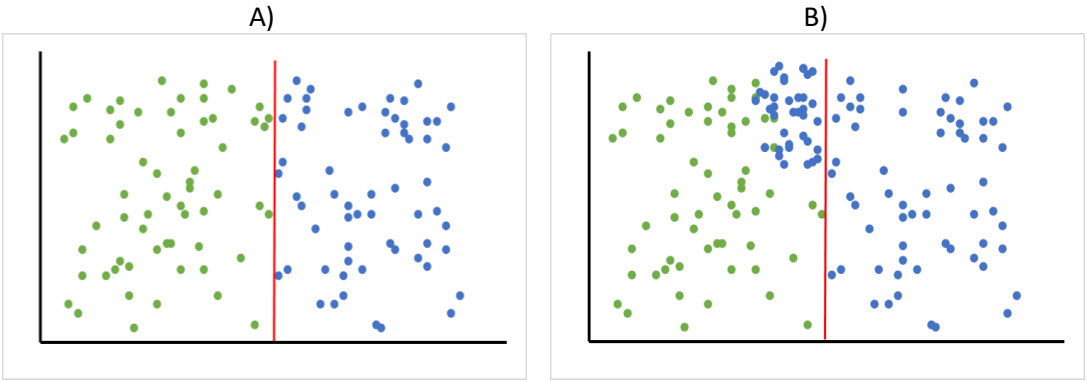

Figure 13 *Decision boundary A) Without concept drift CD, B) with CD.*

Furthermore, authors in [123] provides a literature review for concept drift detection for streaming environments. In their work they analyze the concept drift nature and analyze concept drift types. Then they provide a chronological review for CD detection algorithms and analyze CD adaptation strategies. At the end they present drawbacks and presents future research directions in the field of CD, highlighting the necessity for dealing with CD in conjunction with multidimensional and imbalanced datasets. In addition, authors in [124] provide an overview of CD for recurring concepts. Also, in this work authors provide a review for supervised learning under concept drift and suggest that more work should be done in the field of the unsupervised learning. Furthermore, they analyze meta learning on environments with recurring concepts. Despite the work done in the classification problem in the field of CD more work should be done on regression problems. Also, they indicate that more research is needed in multi-label classification and concept drift detection with data class imbalance. After that, authors in [125] provide a roadmap about CD detection. They mainly analyze concept drift and present state of the art CD detection algorithms. They also provide a section with tools developed through the years for concept drift handling and an extensive list of real-world and synthetic datasets for concept drift. Finally, they conclude that more research should be done for Semi-supervised and unsupervised approaches to drift detection and adaptation and highlight the class imbalance issue.

The study presented in [126] demonstrates a comparison among several drift detection algorithms. The performance of each considered detector was tested in various datasets, especially for sudden and gradual concept drifts. An NB model was set as the base learner. For each testing sample, the drift detector assesses the output of the base learner and creates an

alert when a drift or a warning (prior to drift alert) is detected. If a warning is triggered, a new base learner is created and trained in parallel with the first. Most importantly, when a drift is flagged, the new base learner (trained with the warning samples) replaces the existing base learner. Finally, it is noted that adaptive windowing outperforms the other algorithms in terms of runtime performance, whereas DDM presents the highest average accuracy in datasets with sudden concept drifts, while also showing enhanced performance for datasets with gradual drifts. The effect of different base classifiers in the Learn++ algorithm family was reported in [127]. For comparison purposes, five different base classifiers were evaluated in two datasets composed of imbalanced data. The results suggest that the performance of these algorithms strongly depends on the performance of the base classifier. Finally, the authors conclude that Classification and Regression Tree (CART) and Multilayer Perceptron Neural Networks (MLPNNs) are well-suited for such purposes [29].

Furthermore, an extensive and large-scale comparison of concept drift detectors is presented in [128]. In this study, various differently-configured drift detectors are compared, using seven different datasets. In specific, abrupt and gradual drifts datasets are employed, each one generating seven instance sizes. NB and Hoeffding Tree (HT) were implemented as base learners. The prequential methodology was followed and the prediction accuracy of the base classifiers was used for evaluation. As a conclusion, the authors suggest that DDM based on the Hoeffding's inequality HDDM and RDDM may serve as the optimal detectors. Finally, the comparisons in [129] include a configuration of six different ensembles with the use of 10 drift detectors, following a similar approach to [128]. Using again the NB and HT models as base learners, it is concluded that there is not an all-size-fits-all model across the considered datasets, although HT with HDDM or RDDM showed better performance [29].

Authors of [130], propose a drift detection method DDM. In order to detect a drift occurrence, authors monitor the performance of a base model and calculates the probability (pi) and the standard deviation (si) of the classifier to make false predictions. According to a predefined levels of deviation DDM triggers warnings or drifts. Next, in [131] authors extend DDM by estimating the distance between classification errors. Early drift detection method (EDDM) was compared with DDM in artificial datasets with abrupt and gradual drift. The method presents exceptional results in slow gradual drifts. Furthermore, authors of [132] use a drift detection method based on Hoefding inequality bounds (HDDM). They provide two HDDM algorithms that monitor moving averages and weighted moving averages of the observed values. They contact many simulations in artificial and real data streams and conclude providing remarkable results. Author of [133] provide an accurate drift detection method (ACDDM). This algorithm uses Hoefding bounds in order to calculate the difference of the current error rate and the minimum error rate observed. The algorithm was tested against four drift detection methods and six datasets (three artificial and three real world datasets) providing reliable results.

In [134], author presents linear four rates (LFR) algorithm for drift detection. LFR monitors the four characteristic rates of confusion probability matrix (true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN)) for significant change that will trigger a warning or a drift. Next authors in [135], propose a two-stage hierarchical hypothesis test (HLFR) for concept drift detection. In the first stage they monitor the possibility of drift existence and the second stage confirms the validity of the drift. Second stage uses LFR for drift detection. This method exhibits the ability to effectively identify concept drift, achieving

superior precision in detection, and displaying remarkable adaptability across various concepts.

In [136], authors use exponential moving average (EWMA) for drift detection. Algorithm (ECDD) observes the misclassification rate of the base classifier. Experimental results with artificial and real-world datasets shows competitive results against relevant drift detection methods. Authors of [137], propose statistical testing for drift detection (STEPD). In general, STEPD compares the accuracy in a time window (w) with the overall accuracy. A significant change in the statistics reveals the drift existence. Next, in [138] authors introduce reactive drift detection method (RDDM). Authors here use the instances beginning from the last drift, in order to detect drift with DDM. Each time a drift is detected RDDM triggers new statistics calculations. This work contacts many experiment results in artificial and real-world datasets. The findings reveal the superiority of RDDM against DDM, ECDD and STEPD. Furthermore, authors of [139] use adaptive windows (ADWIN) in order to compare the calculated average. A drift is triggered whenever the difference exceeds the predefined threshold. In [140], authors use Kolmogorov-Smirnov to compares the moving averages of null hypothesis and alternative hypothesis (drift). In their work, they compare kswin against Adwin, DDM and EDDM. Their finding suggests that kswin was the best at detecting drift in datasets from six stream generators (synthetic data) and six real-world datasets. Next, in [141] authors propose a multi-level weighted drift detection method (MWDDM). In their algorithm they use a combination of two sliding windows, a short sliding window and a sliding window with a larger length in order to deal with abrupt and gradual changes. In their windows they also use a weighted mechanism in order to append larger weights in resent instances. Following that, they propose two deviations of their method, the one uses Hoeffding inequality (MWDDM_H) and the other Mcdiarmid inequality (MWDDM_H). They use both of them in order to compare the two windows and indicate warnings and drifts. They finally compare their methods against eight state-of-the-art drift detection algorithms in four artificial datasets and three real world datasets. Their methods can achieve high accuracy and accurately detect abrupt and gradual drifts.

Authors in [142], propose Drift Detection Method with False Positive rate for multi-label classification (DDM-FP-M). In order to deal with multi-label drift, they calculate the arithmetic mean of the total false positive rate (TFPR) of all labels and feed it in DDM. Similar to DDM they use warning levels and drifts levels. They compare their algorithm with DDM by using a dataset consisted of 54 sensors that collect timestamped topology information with humidity, temperature, light and voltage from the Intel Berkeley Research lab. They split the dataset collected in multiple groups of data and test then in both algorithms. The findings suggest that the DDM-FP-M outperforms DDM in the most groups of data. In [143], authors propose Uncertainty Drift Detection (UDD) algorithm, which has the ability to detect drifts without information's about the true labels. In order to detect drifts UDD uses Monte Carlo Dropout approach to capture the uncertainty of a deep neural network (used for predictions) and feed the obtained uncertainty to ADWIN in order to detect drifts. Their method can be used for classification and regression ML problem. For evaluation purposes they test their algorithm in two synthetic and ten real world datasets against state-of-the-art algorithms such as kswin and Adwin. Finally, authors in [144] propose a student–teacher method called STUDD in order to detect drifts without the knowledge of real labels. Their method uses two ML algorithms, the first one (teacher) is trained on features and labels contained in the training dataset. Then, teacher's predictions are used as labels for training the second algorithm (student) in conjunction with the original training features. Then the discrepancy between student and

teacher is used to calculate the loss and use it as input to Page-Hinkley test for detecting drift occurrence and retrain the model.

Table 4 provides the summaries of the relevant surveys of concept drift detection and adaptation.

Table 4 *Concept drift literature*

Reference	Short Description
Jie *et al*., 2019 [120]	Review in CD detection algorithms.
Wares *et al*., 2019 [121]	Review in Window-Based CD detection.
Iwashita *et al*., 2019 [122]	Window, weight and ensemble-based approaches.
Agrahari *et al*., 2022 [123]	Literature review fin CD for data stream mining
Suarez *et al*., 2023 [124]	Survey on recurring concepts
Mahdi *et al*., 2024 [125]	Roadmap of CD adaptation
Gonçalves *et al*., 2014 [126]	Comparison between drift detection algorithms.
Liao *et al*., 2016 [127]	Compares different base classifiers in Learn++.
Maior *et al*., 2018 [128]	Large scale comparison of CD detectors.
Maior *et al*., 2019 [129]	Compares ensembles of drift detectors.
Gama *et al.,* 2004 [130]	DDM Detection method
Baena-Garcia *et al*., 2006 [131]	EDDM Detection method
Frias-Blanco *et al.,* 2015 [132]	HDDM Detection method
Yan M., 2020 [133]	ACDDM Detection method
Heng Wang et al., 2015 [134]	LFR Detection method
Yu *et al*., 2019 [135]	HLFR Detection method
Ross *et al.*, 2012[136]	ECDD Detection method
Nishida *et al.,* 2007 [137]	STEPD Detection method
Baros *et al*., 2017 [138]	RDDM Detection method
Bifet *et al.,* 2007 [139]	ADWIN Detection method
Raab *et al*., 2020 [140]	Kswin Detection method
Chen *et al.,* 2023 [141]	MWDDM Detection method
Wang et al., 2020 [142]	DDM-FP-M drift detector for multi-label classification
Baier *et al.*, 2021 [143]	UDD Detection method
Cerqueira *et al*., 2023 [144]	STUDD Detection method

3.4 Zero waste - Defect prediction and detection

Zero defect manufacturing is continuously emerging as the most critical target of the Industry 4.0 era. Minimization of the defective products in the industrial production chain can contribute towards significant improvements of the operational costs, the production efficiency and speed, as well as the environmental footprint. To that end authors in [22] focus on a thorough literature evaluation of two primary ZDM strategies: defect prevention and defect compensation. In addition, they identify gaps and potential research objectives. Authors in [145], provide a systematic literature review about defect detection with special focus on Root Cause Analysis (RCA). After that, authors of [146] give an overview of ZDM and provide a framework for integration in a defect free multistage production. Next in [147], authors develop a decision support system (DSS) in order to support decisions in the event of a defect. They test their DSS in a real-world manufacturing process on semiconductors sector. They conclude that DSS can effectively detect defects and enchase the decision making in order to achieve ZDM. After that, in [148] authors provide an overview and a bibliometric analysis for online inspection in line with ZDM.

In [149], authors apply AI in a process to a Swedish manufacturing plant. During the preprocessing, they use random forests (RF) in order to highlight the most important data and Principal component analysis (PCA) to reduce the dimensionality. Moreover, random undersampling utilized to deal with imbalanced data. Extreme gradient boosting (EGB) regressor was used to predict the defect similarity ratio. The evaluation reveals that AI can effectively support ZDM and more particularly defect detection even in the early stages of process. Next, in [150] authors provide a predict prevent sequence in order to improve the assembly quality in a real-world scenario. They introduce two LSTM networks, a deep supervised long-short term memory (SLSTM) and a long-short term memory-genetic algorithm (LSTM-GA) to predict the quality characteristics and optimize the assembly quality respectively. Furthermore, a real case study in printing industry was made by [151]. This study proposes three clustering methods (k-means, agglomerative hierarchical clustering and density-based scanning) in order to effectively reveal the features with the most impact on defective products. Finally, they use the predictions to support a selection policy and compare the results with the performance of the pre-existing selection policy. Next, in [152] authors provide a framework for waste prediction in an offset printing industry. They use a two-stage ensemble learning model. In the first stage they use a stacking ensemble that includes Support Vector Machine (SVM), Kernel Ridge Regression (KRR) and Extreme Gradient Boosting (XGBoost) and combine their predictions with Elastic Net in the second stage. During the evaluation they reveal that the proposed framework successively predicts the amount of waste that each machine produce enabling the appropriate machine selection.

In [153], authors use principal components analysis (PCA) in order to detect defects in adhesive placement and dispensing in a semiconductor production system. After the detection they use an event modeler in order to create a mapping between the input variable and the event (defective or not defective glue dispensing). Then, they use the mapping in order to create a prediction function through Gene Expression Programming (GEP). The prediction function is used for defect prediction and the results show a 99,93% accuracy rate. Furthermore in [154], authors provide a multistage algorithm to test two real world datasets in assembly of pharmaceutical devices for quality assessment. More specifically, in their algorithm they use a k-clustering algorithm for anomaly detection. Then, for assess the quality

they use an ensemble of classifiers named decision trees (DT), support vector machines (SVM) and Convolutional neural network (CNN). At the end they demonstrate that their algorithm can achieve an accuracy of 100%. An architecture for identification of defective products and improving processes to ensure defect free production is proposed by [155]. More specifically, they experiment in two use cases: 1) the casting manufacturing process and 2) mining. In the first use case, CNN models VGG-16, Inception v3, and EfficientNetB0 are trained and compared to detect casting defects. They use regression models such as K-nearest neighbors (K-NN), decision trees, Lasso regression, and linear regression to assess the quality of iron ore pulp. According to their findings, they underline that Inception v3 and decision trees are most suited for use cases 1 and 2, with accuracy 95% and R2 score 0.99, respectively.

Authors in [156], use ML methods for waste prediction in a food industry of fruit and vegetable pasteurization. Data represent the amount of wasted products and the actual amount of product yield from the production process. From their research, they conclude that the use of these ML models could improve production planning in the investigated industry, lowering economic costs and food waste. Furthermore in [157], present a framework to employ ML in laser materials processing (LMP) of e-mobility components and accessories in various industrial machinery to reduce faults and waste. An ANN was selected in order to predict the weld bead quality. When the ANN predicts bad quality, the framework employs prevention techniques to reduce the amount of material waste.

Table 5 summarizes the relevant papers for defect detection, defect prediction and waste minimization.

Table 5 *Defect detection and prevention - waste minimization*

Reference	Short description
Powell *et al.,* 2022 [22]	Review defect detection and compensation
Papageorgiou *et al.,* 2022 [145]	Review defect detection with RCA
Eger *et al*., 2018 [146]	Overview, framework for defect manipulation
Psarommatis *et al.,* 2022 [147]	DSS to handle defects
Azamfirei *et al.,* 2023 [148]	Overview, Bibliometric analysis online defect detection
Leberruyer *et al.,* 2023 [149]	EGB for defect detection and process optimization
Zhao *et al.,* 2023 [150]	LSTM for assembly quality optimization
Spantideas *et al*., 2022 [151]	Machine selection policy through unsupervised learning
Kalafatelis *et al*., 2023 [152]	Ensemble learning for waste prediction
Huang *et al.,* 2018 [153]	Defect prediction in micro-semiconductor sector
Dengler *et al*., 2021 [154]	Quality assessment in pharmaceutical assembly
Benbarrad *et al*., 2021 [155]	Defect detection and prevention
Garre *et al*., 2020 [156]	Waste prediction
Ndeda *et al*., 2023 [157]	Defect prediction

3.5 Predictive Maintenance

An important process that can greatly benefit from the application of ML, is predictive maintenance. More specifically, the availability of a wealth of data, stemming from the various processes and machine state information enables the timely and accurate prediction of when machinery requires maintenance, as well as how maintenance operation can be improved. Even when clear indications of machinery wear are not available, predictive maintenance leverages on data acquired during production to identify the characteristics of this degradation. In this way, costs and downtime duration can be reduced while increasing the production output. The application of BDA and data fusion for predictive maintenance purposes has been recently investigated within the contexts of semiconductor manufacturing and smart factories, in [158], [159], respectively [17].

Next sections focus on predictive maintenance with deep learning, supervised and unsupervised learning methods. **Figure 14** illustrates the supervised versus unsupervised methods for PdM and the applications that are used for.

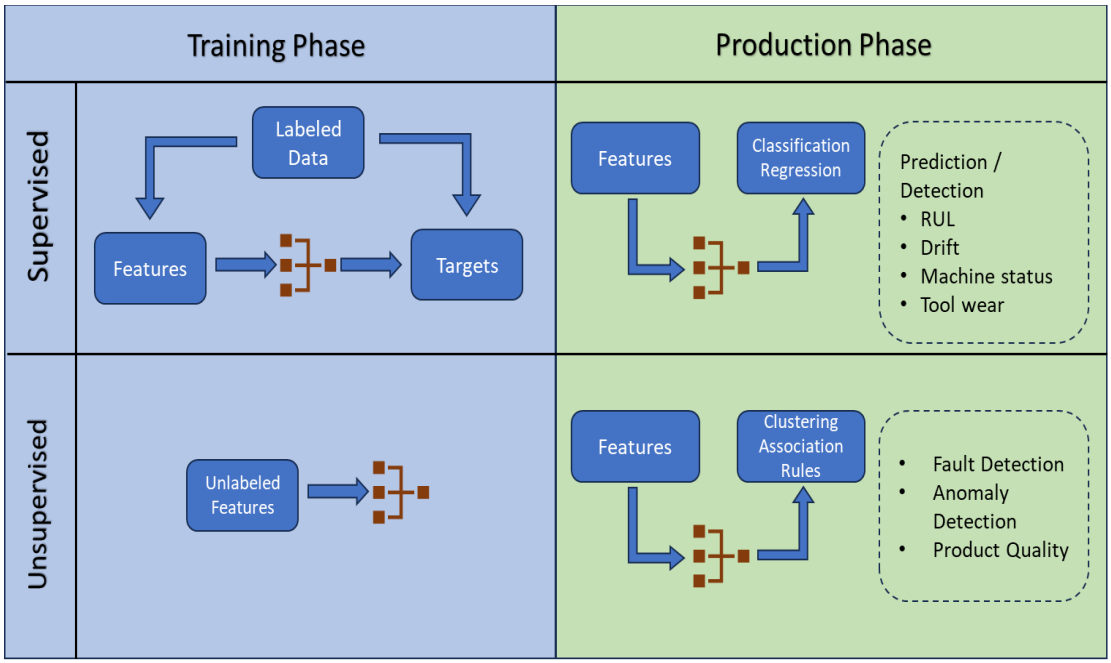

Figure 14 *Supervised vs Unsupervised learning for PdM.*

3.5.1 Supervised learning-based solutions

The first category of ML-based solutions for predictive maintenance relies on supervised learning and comprises the majority of relevant works. In [160], multiple classifiers (MC) were adopted for predictive maintenance, regarding integral type faults in semiconductor manufacturing, i.e., failures occurring due to the accumulative high usage and load on the machinery parts. Thus, the developed method aimed at addressing the issue of imbalanced data sets, usually existing in maintenance classification problems [161] and enabled

maintenance under statistical cost minimization. Towards this end, MC components operating with varying prediction horizons, improved performance targets, such as predicting frequency of unexpected breaks and unexploited lifetime, imposing maintenance decision rules in a dynamic manner and tackling high-dimensional data problems. The MC supervised ML method was implemented in a semiconductor manufacturing maintenance process, revealing that better performance can be achieved, compared to conventional preventive maintenance solutions and predictive maintenance alternatives, relying on single SVM and k-NN classifiers. Thus, the MC method was capable of providing the lowest operating costs while being robust against varying values of parameters, determining the frequency of unexpected breaks and the amount of unexploited lifetime. Another work investigated the remaining lifetime detection of key machinery components for predictive maintenance purposes by analyzing heterogeneous data that were collected through multiple sources [162]. More specifically, an ML-based solution was developed for structuring such data sets, considering the spatiotemporal property, and invisible factor modeling for reduced downtime and energy-efficient machine operation. The adopted systematic procedure comprised structuring of industrial big data, and semantic web technology for semi-structured data, as well as target recognition, detection and tracking for unstructured data. Then, multi-scale analysis was performed for structured data characterization, identifying hidden patterns, while spatial envelope analysis extracted independent subsystems and time-frequency analysis performed signal decomposition. The performance assessment of the multi-scale analysis method showed that it improved the accuracy of remaining lifetime prediction, compared to an alternative ANN method using a single source of information. More importantly, it was concluded that exploiting heterogeneous data from multiple sources enables novel methods of predictive maintenance, task scheduling and machining process optimization towards reducing the energy consumption [17].

Next, predictive maintenance based on heterogeneous data, such as cutting forces, vibration signals and acoustic emissions, from milling machinery was the main focus in [163]. The authors adopted ensemble learning RandF where each individual decision tree corresponds to a regression tree, as tool wear represents the gradual failures in the operation of cutting machinery. Also, the bagging, slipping and stopping criteria for the ML-based solutions were given and comparisons with ANNs with a single hidden layer and SVMs alternatives were presented, using a data set collected from 315 milling trials. It was observed that although RandF required longer training than ANNs and SVMs, it provided the highest accuracy in predicting tool wear. An SVM solution was proposed to address issues in shop-floors where the main production is performed, through automated machines, workers or both, in Industry 4.0 environments, millions of devices and sensors can produce a wide range of data for predictive maintenance. So, in [164], ML based on SVM was presented, modelling the conveyor belt system through the M/D/1 queue. In this setting, SVM predicted instances of abnormal operation where machine overloading or slowdowns occurred, by detecting changes in the queue parameters. For any abnormality, the situation was solved by reconfiguring the manufacturing system. This enabled a flexible system, even if an abnormal situation occurred, as illustrated through simulations where automated detection of abnormalities and self-healing were observed [17].

As small and medium-sized enterprises (SMEs) constitute a major group within the Industry 4.0, providing specific benefits in terms of innovation and willingness to adopt novel solutions, as well as limitations regarding their workforce and economic resources, the work in [165] proposed a low-complexity solution for predictive maintenance, aimed at SMEs.

Currently, existing ML-based methods rely on complex, algorithms, such as linear-discriminate analysis (LDA), decision trees and neuro-fuzzy networks. Moreover, such methods are based on significant amounts of data being collected for classification and feature extraction. For these reasons, the authors proposed a low-complexity algorithm for online feature dimensionality reduction and automated machine status detection and prediction through an NN. The proposed solution included two steps where in the first, add-on sensors collected machine status data. Then, data parsing and feature extraction were performed, prior to dimensionality reduction and feeding of the results, as input into NNs for training and prediction model creation. For verification and validation purposes, the NN-based method was implemented using real machine data from a spring manufacturing factory, employing add-on triaxial accelerometers for data collection. Comparisons with other methods, such as LDA, showed that highly attractive trade-offs between accuracy and prediction can be attained, thus facilitating the adoption of ML-based predictive maintenance by SMEs. More specifically, the proposed method achieved an overall accuracy of 93.1%, equal to LDA, but with the capability of being implemented with more simple hardware, thus avoiding the need for expensive and specialized hardware [17].

In cases where machine conditions dynamically vary in time due to concept drifting with aging, failures and repairs, predictive maintenance is a challenging process. More specifically, patterns stemming from the operational data could significantly change, thus resulting in reduced prediction accuracy. In order to tackle this issue, especially when only a single classifier type is used, ensemble learning based on MC types and diversity was employed in [166]. The basis for the proposed solution is diversity for dealing with drift (DDD) [167], enabling data diversity manipulation for constructing ensemble models, with each one comprising several base classifiers to address concept drifts. However, the multiple classifier type DDD that was developed, consisted of multiple classifier types, dynamic weight adjusting, and data-driven adaptation to concept drifts for offline learning. In this way, timely adaptation to concept drifts and high prediction accuracy can be harvested. Moreover, to relax the computational requirements, the distributed CC framework, called MapReduce was adopted [168]. From the simulations that were conducted with a concept drift data set, the efficiency of the multiple classifier type DDD was outlined, adapting to concept drifts by appropriately updating the classifier weights, leading to an accuracy of 87.99% [17].

Moreover, RUL prediction was utilized to minimize maintenance cost and process breakdown form [145]. A mixture of clustering, regression as well as ensample learning algorithms was trained in data collected from a production process of an assembly line to predict upcoming malfunctions. The outcomes suggest that the selected approach successfully restrain production downtime to the extent of 58 percent. In order to optimize the accuracy of PdM models in identifying minority classes using binary classification, authors in [169] recommend a data sampling approach for predictive maintenance algorithms for Offset Printing environments. A dataset from an Offset Printing company was used to develop the methodology, which utilized SMOTE, ADASYN and RUS techniques as well as different classification algorithms (DT, LoR, KNN, RF). The study suggests that the proposed method is effective in managing data imbalances and improving model performance, surpassing other contemporary methods.

3.5.2 Unsupervised learning-based solutions

Currently, there exist few works presenting unsupervised learning solutions for predictive maintenance. In [170], a complex system is given, performing IoT data collection, storage, processing and visualization, prior to predictive modelling. This system provides structured and unstructured data ingestion from multiple sources, as well as data management via data lakes, based on file and data base management systems, such as Hadoop and Apache Hive, among others. The joint consideration of big data and CC, during the predictive stage encompasses data quality processing and MapReduce-based distributed principal component analysis (DPCA). DPCA exploits unsupervised learning for training, using unlabeled data which is necessary in large-scale manufacturing plants, characterized by heterogeneous data. The system was subject to long-term evaluation in a real industrial environment and compared learning algorithms based on K-means, square predicted error (SPE) and DPCA-based T-squared. It was observed that SPE exhibited a more stable behavior, in terms of fault detection, while K-means and T-squared were more sensitive in instances of multi-variable value changes. Independently of the adopted ML algorithm, the system enabled real-time notification of abnormal operation, even several days prior to an actual failure [17].

Next, the work in [171] aimed at integrating real-time and historical analysis, towards self-optimization for predictive maintenance. Thus, an intelligent data analysis and real-time supervision (IDARTS) framework was presented for data collection and creation of context-aware data analysis and evaluation. IDARTS was designed according to the plug-and-produce functionality, supporting predictive maintenance with dynamic system virtualization, addressing changes at the shop-floor level. The data output from the virtual resources was used to train a K-means clustering classifier. Moreover, context-awareness and HMI allowed the system to either adopt the operation parameters towards self-reconfiguration, or guidance to an operator for safeguarding normal production conditions and product quality. From the experiments, it was revealed that IDARTS provided scalability and adaptability to production changes, in terms of shop-floor layout while performing data analysis in runtime environment, without needing additional programming, down time or redeployment.

Finally, author in [172] compare unsupervised learning models for anomaly detection on data collected from sensors of a real e-coating plant. They conduct experiments on three distinct unsupervised learning algorithms, namely Interquartile Range (IQR), Isolation Forest, and Elliptic Envelope. The collected data where preprocessed with stationarity tests and feature extraction with PCA. The algorithms are compared in terms of training time and number of anomalies detected. The findings reveal that IQR detects the most anomalies with the least training time.

3.5.3 Deep learning-based solutions

DL has emerged as a significant advancement in data-driven analysis and optimization for industrial applications, especially for those focusing on the accurate prediction of RUL. The authors in [173] presented a device electrocardiogram (DECG) framework, employing a deep denoising autoencoder (DDA) and regression operation to enhance the accuracy of RUL prediction of industrial machines. DECG avoids sensor installation and acquires temporal data

for each operation at the device or operator level, from the programmable logic controller (PLC). Also, DECG overcomes the necessity of experts in complicated production issues, greatly reducing maintenance costs. More specifically, DECG collected data relative to the machines' cycle time and through DL and a wide range of run-to-failure data, achieved automated feature extraction and accurate prediction. The proposed solution was put against a conventional factory information system and results illustrated that DECG was able to identify the behavior of operation working time and offered more fine-grained training data, facilitating DL in accurately predicting RUL. Thus, the proposed solution can lead to reduced DL overfitting and improved RUL prediction accuracy [17].

Next, RUL through deep transfer learning (DTL) was investigated in [174]. DTL addressed feature transfer in DL networks and relied on three strategies, i.e., weight transfer, feature transfer and weight update. Thus, DTL with feature transfer learned the joint features among multiple objects, improving the prediction accuracy of RUL, thus showing the advantages of DTL. In greater detail, initially, a SAE network is trained off-line with historical run-to-failure data containing RUL information of a cutting tool. Next, the trained network is transferred to an operating tool towards performing on-line RUL prediction. In this way, RUL prediction was performed without interrupting the machine tool's operation. Also, efficient RUL prediction was achieved without requiring excessive historical failure data sets. As current RUL prediction methods experience difficulties in extracting heterogeneous features from high-dimensional massive signals, the predictive maintenance performance is threatened [17].

In [175], data-driven machine health monitoring was developed, employing adaptive kernel spectral clustering (AKSC), combined with deep long short-term memory recurrent NNs (LSTM-RNN). In greater detail, a three-step approach was adopted, first including, frequency and time-frequency domain extraction from massive signals and an Euclidean distance-based algorithm for identifying the machine degradation features. Then, AKSC was presented, for detecting anomalies in the machine operation, based on multiple degradation features. In the last step, LSTM-RNN was created for updating and predicting the machine failure time. The performance of AKSC with LSTM-RNN was evaluated, using a set of test-to-failure experimental data, outlining its superiority compared to other methods, in terms of average error, root mean square error and accuracy. The improved performance of LSTM-RNN was attributed to its ability of extracting the long-term spatiotemporal dependency of different degradation parameters, towards providing short-term failure prognostics [17].

Furthermore, predictive maintenance in ultra-precision manufacturing applications has been studied in [176]. So, a DL data-driven framework was proposed, fusing multiple stacked SAEs to perform tool condition monitoring. The DL-based solutions consisted of a training model that was able to process multiple parallel feature spaces from time-, frequency- and wavelet-domains data, extracting low-level features and a feature fusion structure for learning higher-level features relevant to tool wear. In addition, feature extraction and classification performance was improved by a modified loss function. A data set representing a real manufacturing process was used to evaluate the performance of the DL-based solution. It was shown that by exploiting the feature learning ability of deep layer models and the heterogeneous feature spaces, the proposed solution successfully classified the tool wear condition with over 96% accuracy, outperforming BPNN and SVM alternatives [17].

A novel framework, presented by the authors in [177], aims to enhance the accuracy of the maintenance scheduling and fault prediction in the manufacturing sector. The framework comprises four distinct layers. The first layer involves data collection from the manufacturing

process while the second layer is responsible for the collection and management of heterogeneous data captured from the first layer. The third layer processes the collected data to derive meaningful information while the fourth layer is responsible for predicting faults, and machines state condition to arrange an appropriate maintenance schedule. For prediction purposes, they use an LSTM-GAN NN trained on real dataset collected from a typical workshop of a manufacturing enterprise. They compare LSTM-GAN against three other NN and conclude that their algorithm outperforms the rest of the NN, achieving an accuracy of 99.12% and providing a logical maintenance schedule that minimizes maintenance cost and downtime.

In [178], the authors propose an approach to address imbalanced rolling bearing fault diagnosis. They employ MsR-GAN to handle imbalanced data and train FED-CapsNet for bearing faults diagnosis. Comparing FED-CapsNet performance on a bearing dataset against three other NN, they conclude that FED-CapsNet outperforms the rest of the NN achieving better accuracy score. Finally, in [179], authors propose a bearing fault diagnosis method that utilizes Deep GRU (DGRU). Their method involves an Artificial Fish Swarm Algorithm (AFSA) that is used to obtain the optimal parameters of DGRU. Subsequently, DGRU extracts features from bearing vibration data to feed into the Extreme Learning Machine (ELM) for classification. This method was compared against SVM classifier and Softmax classifier on a locomotive bearing dataset. The results show superior performance to the other methods.

Table 6 includes the specific industrial environments and the respective ML solutions that were employed for predictive maintenance of equipment in Industry 4.0.

Table 6 *Predictive maintenance setting and respective ML solutions for Industry 4.0*

Reference	Predictive maintenance setting	ML solution
Susto *et al*., 2015 [160]	Imbalanced data sets of integral type faults	MC supervised method
Yan *et al*., 2017 [162]	Unstructured multi- source heterogeneous data	Multi-scale analysis (envelope, time-frequency)
Tasci *et al*., 2023 [180]	RUL prediction	Hybrid Solutions
Wu *et al*., 2017 [163]	Heterogeneous data	RandF
Shin *et al*., 2018 [164]	Self-healing in shop-floor	SVM
Kuo *et al*., 2017 [165]	Low-complexity operation for SMEs	NN-based for online feature dimensionality reduction and automated prediction
Lin *et al*., 2017 [166]	Bias mitigation when using a single classifier type	Ensemble learning with MC types and diversity
Kalafatelis et al., 2024 [169]	Managing class imbalance	DT, LoR, KNN, RF
Yu *et al*., 2019 [170]	Large-scale monitoring with unlabeled data	K-means, DPCA-based T-squared and SPE

Table 6 Predictive maintenance setting and respective ML solutions for Industry 4.0 (Cont.)

Reference	Predictive maintenance setting	ML solution
Belichovski, *et al*., 2022 [172]	Anomaly detection	IQR, Isolation Forest, Elliptic Envelope
Yan *et al*., 2018 [173]	Automated RUL prediction without experts' knowledge	DL-based DECG
Sun *et al*., 2019 [174]	RUL relevant feature transfer in DL network	DTL with SAE
Cheng *et al*., 2019 [175]	Heterogeneous feature extraction from massive signals	AKSC with LSTM-RNN
Shi *et al*., 2019 [176]	Tool wear identification in ultra-precision manufacturing without experts' knowledge	Feature spaces-based DL
Liu C. *et al.*, 2021 [177]	Framework for PdM and maintenance scheduling based on NN predictions	LSTM-Gan
Liu J. *et a*l., 2022 [178]	Imbalanced rolling bearing data, fault diagnosis	MsR-GAN, FED-CapsNet
Zhao *et al*., 2020 [179]	Bearing fault diagnosis	Deep GRU, ELM

3.6 Federated Learning

Federated learning is a new technology targeting at the collaborative of multiclient ML model training in order to achieve enchased performance, while preserving data privacy. FL applications are formed from a central server (server), which stores a central model, and decentralized devices (clients), that retain local models. The core procedure includes the model initialization on the server, the model parameter distribution to local clients, the local model training, the server update and finally aggregation.

Many researches have been conducting in the field of FL, for instance in [181], authors provide a survey on FL. In their work, they introduce existing FL work categorized into five aspects, namely data partitioning, privacy mechanisms, applicable ML models communication architecture and heterogeneity handling. Finally, they provide the main challenges and future research directions in the field of FL. Moreover, in the article [182] authors give an overview of FL. They provide the properties and challenges of FL against traditional machine learning. Finally, they present open issues and provide an outline for future research directions. Furthermore, the paper [183] presents a review of FL divided in three characteristic aspects s, namely design, challenges and application areas. Article [184], presents the fundamentals of FL and strongly emphasize the utilization of FL in current application trends and use cases in the domain of technology and markets. New research studies in the field of FL are also conducted.

Authors in [185], provide a review in FL with special focus on industrial engineering and computer science literature. Moreover, through their extensive literature research they provide the summarization of FL characteristics and open issues for future research opportunities. Paper [186], presents a deep overview of FD. Authors, focus on providing a

detailed overview of enabled platforms and protocols. They also describe real-world applications and use cases, as well as downsides and future research areas, with the goal of giving a walkthrough for other scientists so that they can develop better solutions.

In [187], authors provide a detailed overview of FL security vulnerabilities. They discuss potential security implications and attacks in every stage of FL. Furthermore, they provide a detailed overview of defense techniques and give future research directions for secure and robust FL environments. In a FL scenario many clients participate in the learning procedure introducing a wide range of heterogeneity in terms of data distribution and hardware configuration. Therefore, authors in [188] provide a systematically overview of state-of-the-art client selection policies. Moreover, they present the challenges that arise and give potential solutions and future research directions.

Authors in [189], conduct a comprehensive review on aggregation techniques for FL. Moreover, they provide a taxonomy of FL aggregation techniques and categorize them as synchronous, asynchronous, hierarchical, and robust aggregation. Furthermore, they explore the implementation of aggregation technique in real case scenarios and highlight drawbacks and future research opportunities. Next, in paper [190] they present FedSGD strategy. In FedSGD strategy each Local FL participant compute gradients from training samples and send them to a cloud server for aggregation. Moreover, in [63] FedAvg is presented. In FedAvg weights are computed locally after numerous rounds of training and then uploaded to the cloud for aggregation. In [65], use FedAvg with server momentum to deal with non-independent and identically distributed (non-IID) data across FL participants. After that in [64], authors present FedProx with proximal term that is commonly used to address statistical heterogeneity and non-IID data among local FL participants.

In Table 7, relevant papers in the field of FL are listed

Table 7 *Federated Learning relevant papers.*

Reference	Short description
Zhang *et al.,* 2021 [181]	Survey on FL
Li T. *et al.,* 2020 [182]	Overview on FL
Rahman *et al.*, 2021 [183]	Review on FL
Banabilah *et al*., 2022 [184]	Review and application in the field of FL
Li L. et al., 2020 [185]	Applications review of FL
Aledhari *et al*., 2020 [186]	Overview and applications in the field of FL
Bouaciada *et al*., 2021 [187]	Security concerns in FL
Fu *et al.,* 2023 [188]	Client selection policies

Qi et al., 2024 [189]	Survey on model aggregation techniques
Chen j. et al., 2016 [190]	FedSGD strategy
McMahan et al., 2017 [63]	FedAvg strategy
Li T. et al., 2018 [65]	FedProx strategy
Hsu et al., 2019 [64]	FedAvgM strategy

3.6.1 Predictive Maintenance in Shipping 4.0

In the existing literature there have been various ML-aided approaches to conduct PdM in the context of the shipping industry. **Figure 15** depicts PdM components in the context of Shipping 4.0, considering single-ship (panel A) and multi-ship (panel B) ML-aided processes. More specifically, in the single-ship case, a local model is trained on the ship by exploiting data acquisition and processing, and the definition of condition failure features. On the contrary, for multi-ship ML-aided PdM, local model parameters are fed to a central cloud which derives the global model parameters that are transmitted to the ships towards improving their local models, thus achieving collaborative ML model training. Relevant studies have provided ML-aided PdM in Shipping 4.0 for both cases [37].

The study in [191] provided a short survey of PdM in the context of maritime systems, aiming at reducing maintenance and logistics costs while increasing asset availability. Specifically, the study discussed predictive methods based on the physics of failure, as well as various issues associated with the development and implementation of these models. Challenges in this field include critical part selection, the choice between data-driven or physics-based predictive modeling, monitoring/data collection, model validation, and formulating a business case. The paper examined two use cases, related to cylinder liners in a diesel engine and Printed Circuit Boards (PCBs) in a radar system. The findings indicated that companies aiming to transition to PdM solutions must invest in effectively measuring and documenting variations in operational profiles [37].

Next, the paper in [192] presented an ML-aided approach towards enabling maritime companies to benefit from advanced anomaly detection. Several contemporary methods in the field of fault detection while addressing crucial challenges such as interpretation, scale, accuracy, and complexity, which are inherent in many anomaly detection cases. In addition, the authors conduct a comparison of different approaches, including DNNs, SVM, Gradient Boosting, and statistical terms. For the comparisons a static window of 30 days was set, denoting the corrosion, prior to each defect. The combined model, comprising a fusion of these approaches, showed promising performance, in terms of F1 score and mean absolute error (MAE) [37].

In shipping, the ability to identify evolving faults that have a detrimental impact on ship system performance and hinder energy-efficient operations is of critical importance. As a result, the work in [193] presented a data-driven methodology for fault detection in shipboard

systems by exploiting the availability of recorded voyage data for ML purposes. The proposed approach combined the advantages of expected behavior models, which involved selecting the optimal regression model, with the exponentially weighted moving average technique for fault detection, specifically tailored to ship applications. Results demonstrated that the multiple polynomial ridge regression model, achieving a testing R2 score of 0.96 is capable to detect developing faults in both the main engine cylinder exhaust gas temperature and the main engine scavenging air pressure. The early detection of these faults efficiently complements routine ship operations monitoring, facilitating proactive corrective measures [37].

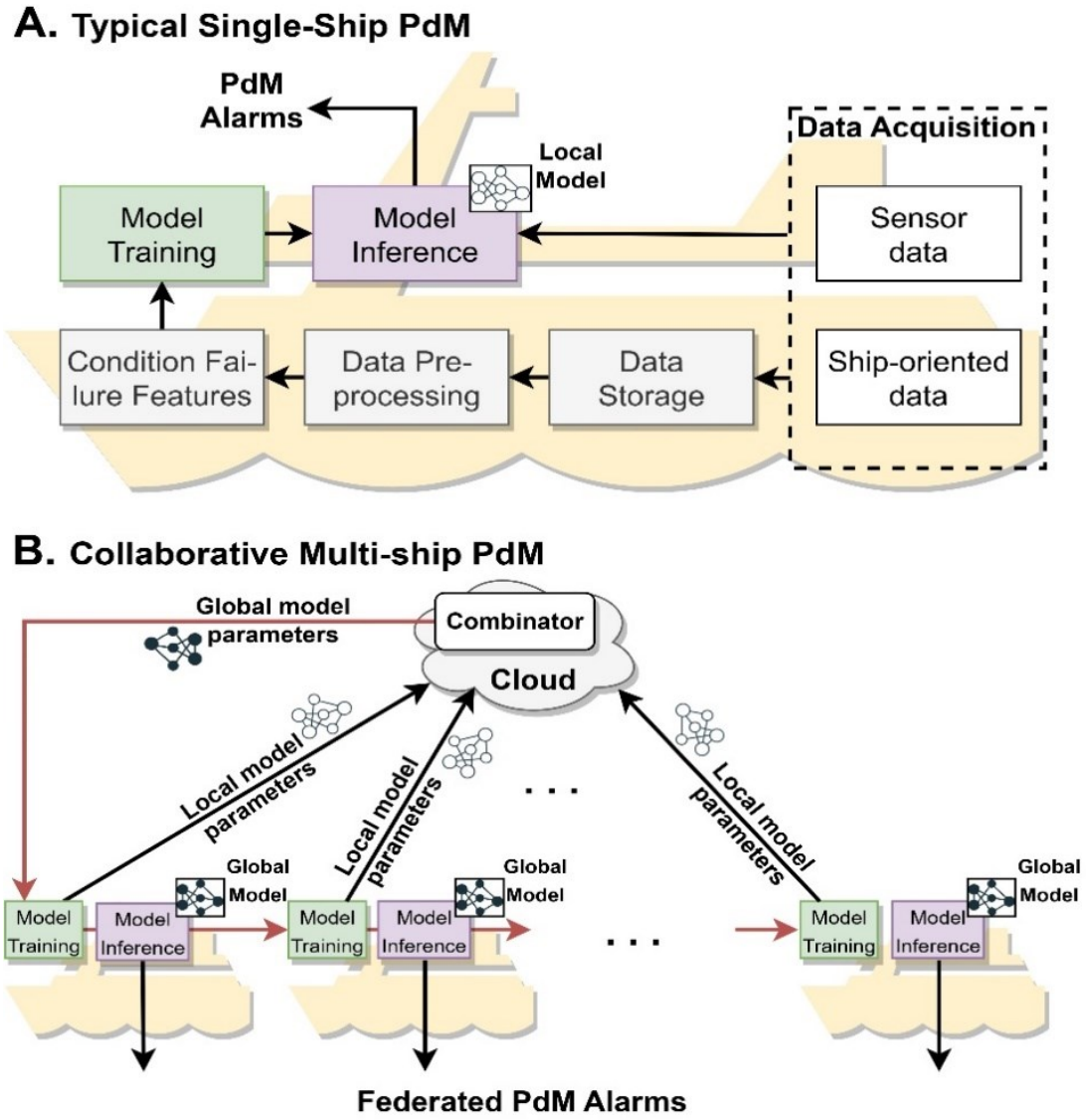

Figure 15 *Predictive Maintenance components in Smart Shipping considering single-ship (panel A) and multi-ship (panel B) learning* [37].

Then, the paper in [194] developed a weakly supervised learning-based PdM, employing balanced random forest and multiple instance learning, relying on data from event logs from ships' electric propulsion systems. Objectives encompassed, predicting the likelihood of

failure, predicting the time to failure, and providing explainability for the predictions. Towards addressing the limitations of current event-driven techniques, temporal random indexing was used to transform irregular textual logs into a consistent numerical array format, offering dimensionality reduction and accurate failure prediction. Additionally, event aggregation techniques were integrated to enable explainable PdM by tracking the sources of failures. Even without manually labeled data, the overall approach successfully discovered the unknown actual class labels of infected samples with high confidence. Most notably, the majority of actual failures were successfully predicted at least three days in advance, indicated by at least one witness sample. These results are of significance to the shipping industry as the proposed PdM can mitigate and minimize the likelihood of propulsion loss during critical maneuvers, promoting the safety of equipment and personnel [37].

The authors in [195] proposed a PdM solution, utilizing a computational AI model, based on real-time monitoring data from historical values corresponding to the health of the vessel's engines and compressors, with R software employed for data analysis. The results of the analysis highlighted the impact of key parameters on the overall condition of the vessel's components, indicating strong correlations among sensor data collected from the same equipment in the majority of cases. These findings highlighted the potential of utilizing such parameters as inputs for the development of a predictive model. However, it was noted that additional factors related to identifying failure modes, detecting potential failures, and assessing asset criticality must be addressed to increase the accuracy of the predictions [37].

Ship hull and propeller fouling have significant consequences for both fuel costs and greenhouse gas (GHG) emissions. In this context, the paper in [196] focused on developing a PdM solution to enhance energy efficiency and reduce emissions from ships. In greater detail, a two-step approach for assessing the actual propulsive ship performance by analyzing continuous onboard measurements was presented. In the first step, the onboard monitoring data underwent correction to account for the influence of wind and wave effects, utilizing fast and transparent empirical methods. Subsequently, the corrected data were filtered based on hydrodynamic criteria. Next, the processed data were subjected to mathematical analysis tools to derive an engine power-rpm curve, representing the ship's actual propulsive performance. It was shown that the proposed ML algorithms enable the macroscopic identification of abnormal operation data during engine operation, prior to conducting detailed data analysis [37].

Finally, the first work developing FL-based PdM for smart ships was [197]. The authors presented a solution, addressing the challenges of insufficient data by introducing FL, enabling multiple industrial smart ship owners to collaboratively train a DL-based model. To mitigate computation and communication costs, a control algorithm was designed, adaptively adjusting the model aggregation interval during training. Additionally, a Paillier-based communication scheme to safeguard the data resources of industrial stakeholders was employed during the training phase. The FL-based PdM was evaluated in terms of privacy, security and accuracy, demonstrating its effectiveness and potential to maintain high fault diagnosis accuracy while reducing computation and communication overheads through adaptive model aggregation intervals [37].

Table 8 provides relevant articles to predictive maintenance in Shipping.

Table 8 *Shipping articles*

Reference	Predictive maintenance setting	ML solution
Tinga *et al*., 2017 [191]	Reducing maintenance and logistics costs	Survey
Makridis *et al*., 2020 [192]	Anomaly detection	DNNs, SVM, Gradient Boosting and statistical terms
Cheliotis *et al*., 2020 [193]	Fault detection	Regression
Bakdi *et al*., 2022 [194]	Predicting the likelihood, and time of failure	Supervised learning
Jimenez *et al*., 2020 [195]	Real-time monitoring	Nothing in particular
Liu *et al*., 2022 [196]	Enhance energy efficiency, reduce emissions	Nothing in particular
Zhang *et al*., 2022 [197]	Insufficient data, fault diagnosis	FL-based PdM

4. Defect minimization

From the early eras of the industrial revolution, the problem of defective manufacturing (due to human errors, material deficiencies or manufacturing processes misconfigurations) still remains unsolved. In the framework of Industry 4.0, conventional methods for reducing defective products are gradually converted towards proactive and predictive measures in the industrial production chain, involving the digitization of the manufacturing process [198]. The emerging concept of zero defect manufacturing primarily aimed to detect the defects in the production chain and identify required corrective actions to minimize the loss of the raw materials and the associated cost, while in parallel optimizing the resource allocation and management in the production chain [199]. Importantly, the wasted resources involved in the production of defective products have additionally a significant contribution to the environmental footprint of the manufacturing industry [53].

Although the digital revolution was a crucial step towards expediting the digitalization of the manufacturing process, the emergence of AI solutions is considered the substantial breakthrough towards ZDM. AI-assisted solutions and machine learning algorithms have been developed to monitor and manage the manufacturing production, targeting to optimize the available resources and the usage of raw materials [200]. The extensive utilization of AI/ML methods has therefore boosted the manufacturing industry towards decreasing the occurrence of defective products, eliminating the wasted resources and costs of the defects and minimizing its environmental footprint [53], [199].

The majority of the ML algorithms that are typically employed in the manufacturing domain require vast amounts of historically gathered data. Through data processing and analysis, ML models are trained based on a specific dataset and are then able to provide estimations and predictions or find hidden patterns, complex correlations and irregularities [24], [28]. The ML algorithms can be categorized in: (i) Supervised Learning (SL) algorithms that require labeled data in the form of input and labeled output data pairs. SL algorithms can be used for classification or regression problems, depending on whether the output takes discrete values (classes) or continuous values; (ii) Unsupervised Learning (UL) methods that do not require labeled output dataset; instead, they use the input data to find clusters that exhibit similar features; (iii) Reinforcement Learning (RL) algorithms that are based on a trial-and-error process of an agent interacting with the environment [53], [201].

It is worth noting that although the ML deployment methods are well-established in the manufacturing domain, ML models abide by the following constraints: (i) as aforementioned, these algorithms require huge amounts of data in order to provide efficient solutions and predictions; (ii) the quality of the collected data has a direct impact on the training procedure of ML models (for instance well-known overfitting or underfitting issues [24]. In this way, each trained ML model is data-dependent and may possibly lead to inaccurate estimates in case of insufficient or noisy dataset; (iii) ML models that are trained to optimize the parameters of an individual machine or to enhance the efficiency of a specific manufacturing process cannot be easily generalized to other pieces of equipment or different processes [53].

Offset printing is a widely-used printing process for multiple products including, amongst others, newspapers, magazines, labels and books. The offset printing process consists of three phases, namely the pre-press, press and post-press. In each one of these three phases, several (raw, organic, chemical and recycled) materials are used, including paper, water, ink,

aluminum, alcohol solutions, having a direct impact on the environmental footprint of the printing process. It is estimated that over the course of a year, the production-related CO2 emissions of a regular offset printing press amount to around 7,100 metric tons for 36 million printed sheets per year, while the paper waste accounts for 230 metric tons [53], [202].

Despite the efforts spent on the formalization and standardization of the different procedures across the various offset printing processes, as well as on providing incentives to the personnel to provide products of high quality, defective products appear along the manufacturing production chain. The main issues that are currently faced within the offset printing production chain include: (i) lack of digitalization of physical processes that would enable real time decision making; (ii) plethora and diversity of production characteristics that do not allow direct process standardization; (iii) defective products are identified at the latest stages of the offset printing production chain, resulting in overdue a-posteriori manual corrective actions; (iv) the environmental footprint of the offset printing industry is considerable, since various waste materials are produced throughout the production chain [53], [203].

The present work outlines the implementation and comparison of SL algorithms in a labeled dataset from the offset printing industry, targeting to demonstrate the potency of ML-assisted methods towards ZDM. The purpose of the SL modeling is to link the specific characteristics of the received customer orders with the amount of defects in the printing process. The accurate correlation of order features with the defective products will result in beneficial machine selection policy, which in turn will contribute to the enhancement of the production efficiency, the minimization of defective products and the reduction of the company's environmental footprint [53].

4.1 Case Study

4.1.1 Available Dataset

The methodology presented in the current work is based on a labelled dataset, incorporating both historical (last 3 years) and recently collected data that will allow accurate training and verification of the developed ML tools, as well as comparison of the results with the actual historical events. Each of the collected data samples follows the press process of a particular printing order with specific characteristics. In order to maintain a balanced dataset amongst the five printing machines, 10K orders for each individual machine were selected as a representative subset of raw data, leading to a total of 50K historical samples. The order-specific characteristics include [53]:

- Associated machine ID: The ID of the machine (1–5) that the particular order was forwarded for printing.
- Quantity: The amount of paper pieces requested in this specific order. Quantity takes integer values extending to up to 1000 pieces, depending on the type of the printing assignment (e.g. newspaper, poster, etc.)
- Quality: The quality of the paper associated with the specific printing order. The Quality parameter takes discrete values (1, 2 or 3) depending on the properties of the

requested paper. In this context, class 1 stands for 'Velvet' paper quality (the most-frequently used), whereas categories 2 and 3 represent the 'Uncoated' and 'Illustration/Gloss' paper quality respectively.

- Color: This categorical variable denotes the color requirements of the particular printing assignment. The most requested color requirement is the 4-color printing (class 1), followed by 4 + 1 color printing (class 2 involving the use of gold and/or silver colors) and grayscale printing (class 3 entailing only black and white colors)
- Ink: The estimated consumed ink level per paper piece (typically 0.1 to 1 gr)
- Type: Type 1 designates that the order shall follow the 'Book' printing process, type 2 describes the 'Poster' requirements, while type 3 determines the 'Journal' specifications.
- Accuracy ratio: The amount of satisfactory printed paper pieces are reported against the required Quantity. The ratio between them indirectly reveals the percentage of defective products, ranging from 0–1.

4.1.2 Machine Selection Framework

The proposed SL modeling framework involves the training of machine-specific ML models based on the described dataset, i.e. the orders that have been historically executed in the five printing machines. For these purposes, the order characteristics (Quantity, Quality, Color, Ink and Type) are considered as the input features of the ML modeling methods, while the Accuracy ratio is the labeled output variable. Once these machine-specific ML models have been adequately trained and validated via testing data (not encountered during the training), they can be used for inference purposes on previously unseen orders. **Figure 16** illustrates the machine selection framework. A new arriving order is accompanied by specific order characteristics. Then, these order features are used as input parameters to all five ML models for each individual machine, enabling the prediction of the machine-specific Accuracy ratio for the new order. The ML model of the machine that exhibits the maximum Accuracy ratio specifies the selection of the machine that will yield the minimum number of defects during the printing process of the particular order [53].

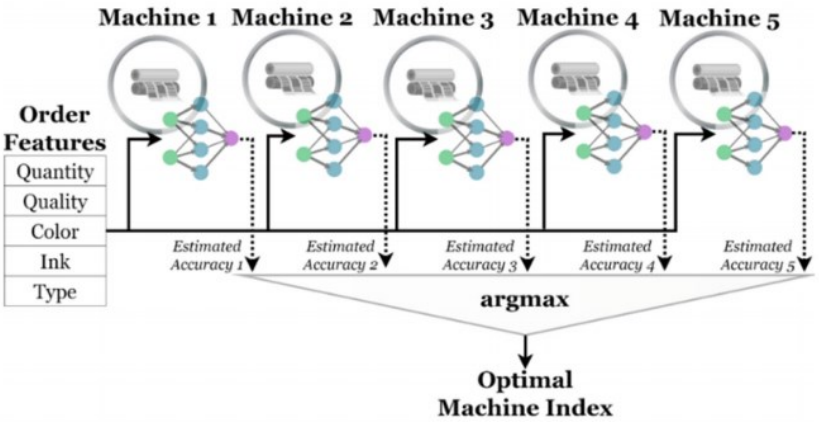

Figure 16 *ML-assisted framework for selecting the best printing machine. During the inference phase, the features of the new order pass through all machine-specific regression models and the printing machine exhibiting the best estimated Accuracy is selected* [53].

4.1.3 ML Algorithms for Accuracy Prediction

In principle, ML algorithms can be employed to create a regression model for each machine, using historical order-specific features as inputs and the respective accuracy score (good amount divided by the total amount) as labels. To find the best regressor for each machine, several ML regression algorithms were employed and tested on each machine-specific datasets [24], [29], [53], [204]:

- Linear Regression (LiR)
- Decision Tree (DT) regressor
- Support Vector Regression (SVR)
- Stochastic Gradient Descent (SGD)
- Random Forest (RF)
- AdaBoost regression
- Bagging ensemble
- Voting regressor
- Stacking regressor
- Multi-Layer Perceptron (MLP) or Artificial Neural Networks (ANNs)

More details about the aforementioned algorithms are provided in Section 2.3

4.1.4 Numerical Results

For each of the following models, a 90/10% training/test split was performed for each machine-specific dataset. Notably, to account for large variations of the features values during the training, the scalar order features (Ink and Quantity) were scaled before training so as to range between [0,1] using Min-Max scaler. All algorithmic implementations were conducted in Python 3.0 using the scikit-learn ML library [53], [205].

4.2 Training of the Multi-layer Perceptron

The performance of the MLPs is strongly influenced by the network depth (number of hidden layers) and the considered learning rate. In this section, the optimal hyper-parameters of the MLP are fine-tuned for each machine. To stabilize the values of the learning parameters, we conducted extensive simulations with varying learning rates (α = 0.1, 0.01, 0.001, 0.0001, 0.00001) and number of hidden layers ($H = 1, 2, 3, 4, 5$). The number of neurons in the ith hidden layer is $2i \times 10$. For each (α, H) combination, the validation metric was the mean squared error (MSE) between the actual and predicted Accuracy on the testing samples. The optimal (α, H) combination corresponded to the minimum convergence loss. **Figure 17** illustrates the training loss curves for the optimal (α, H) configuration for each individual machine (Figure 17 2A–E), as well as the validation MSE per machine (Figure 17 2F) [53].

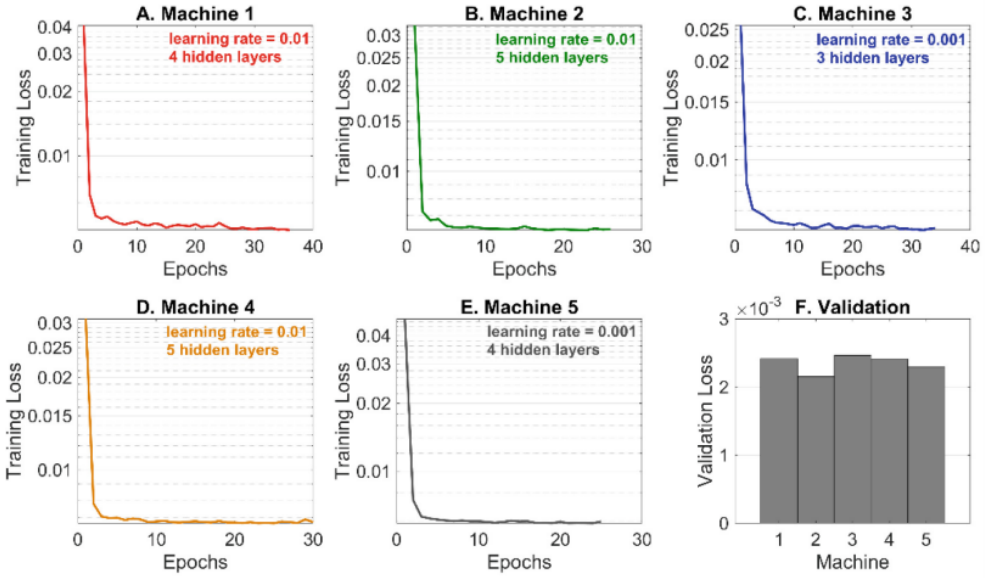

Figure 17 *Training insights of ANNs. A–E. Training loss curves for machines 1–5 employing the optimal learning rate and number of hidden layers. F. Performance of the derived models in terms of validation MSE on the testing data* [53].

4.2.1 Ensemble Learning Techniques

In this section, the convergence validation loss of the ensemble learning algorithms as a function of the number of the estimators used in the ensemble models is illustrated. Specifically, the impact of the number of estimators on the performance of RF, AdaBoost and Bagging regressors is investigated. **Figure 18** depicts the validation MSE loss for varying (10–200) number of individual estimators in each printing machine. The optimal number of estimators for (RF, AdaBoost, Bagging) was (180, 10, 180), (180, 10, 180), (80, 10, 80), (180, 10, 180) and (180, 10, 170) for machines 1–5, respectively. Generally, we noticed that RF and Bagging regressors become more accurate with increasing number of estimators, whereas AdaBoost was insensitive to the considered dataset splits [53].

4.2.2 Comparison Between ML Methods

In this section, the regression performance is compared across 10 machine-specific ML models, namely Linear Regression, Decision Tree, Support Vector Regression, Stochastic Gradient Descent, Random Forest, AdaBoost, Bagging, Voting, Stacking and Multi-Layer Perceptron (ANNs). Note that, to guarantee optimal performance in each individual model, ensemble learning schemes have been configured in their optimal number of estimators (see Section Ensemble Learning Techniques), whereas the Neural Networks have been inferred using the optimal (α, H) combination (see Section Training of the Multi-layer Perceptron). In addition, meta-estimators Voting and Stacking have been configured with the top three

regressors in terms of their prediction accuracy, namely the AdaBoost, Bagging and Random Forest regressors [53].

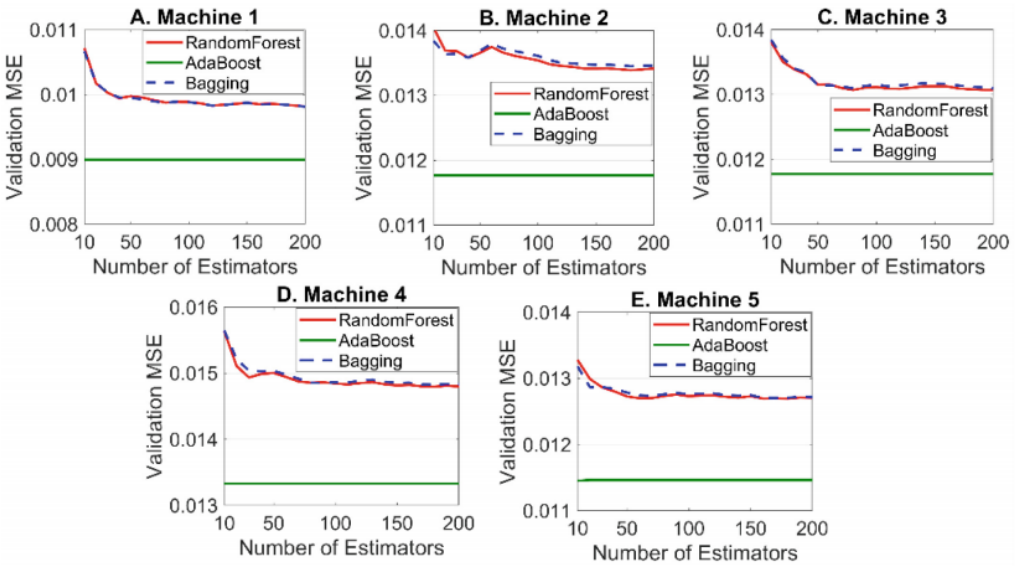

Figure 18 *Impact of the number of estimators on the validation MSE loss of the Ensemble Learning Schemes (RF, AdaBoost, Bagging regressors)* [53].

Figure 19 shows the MSE calculated between the ground-truth and model derived Accuracy scores across the 1K testing samples. Evidently, ANNs and AdaBoost constantly outperform the rest of the models in each machine, achieving a validation MSE of ~0.01. Closely to this performance were the Stacking and Voting ensemble schemes, whereas the worst performance was observed for Decision Tree regressors. Conclusively, ANNs or ensemble schemes can be used to guide the machine selection policy presented in **Figure 16**, given their excessive Accuracy fitting in the testing data [53].

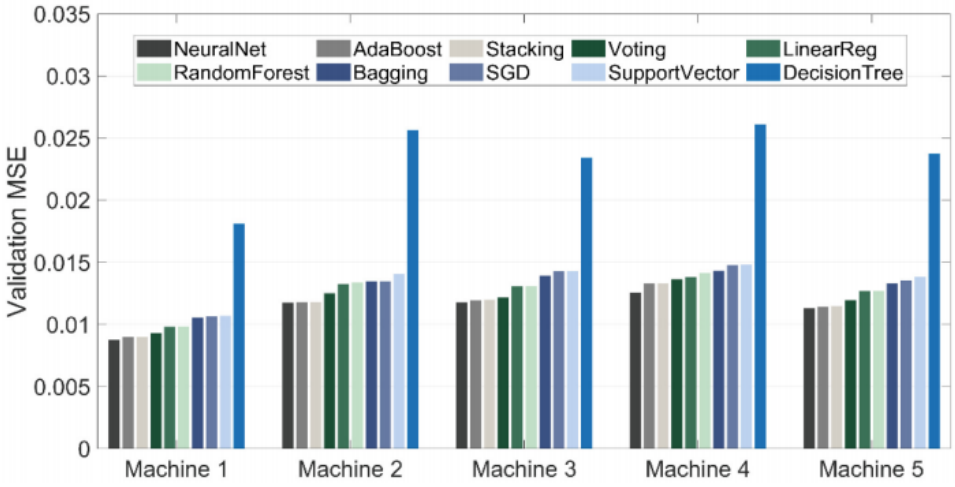

Figure 19 *Comparison of all developed ML algorithms in terms of validation MSE between actual and predicted Accuracy values* [53].

4.2.3 Evaluation of the Proposed Machine Selection Policy

Here the machine selection policy presented in **Figure 16** is evaluated. Firstly, the ANN models configured in their optimal hyperparameters were used, given their superiority in providing more precise Accuracy predictions against the other ML methods (see Section Comparison Between ML Methods). The evaluation dataset was composed by additional 100 orders, containing 20 orders for each machine (i.e. 20 × 5) that have perfectly executed at the respective machine (Accuracy values > 98.5%). In this sense, it is known a-priori that the first 20 samples have to be classified in machine 1, the second 20 samples to machine 2, and so on. The goal of the presented evaluation scheme is (i) to validate whether the proposed machine selection policy (see Figure 16) properly classifies the orders to machines and (ii) to quantify the true positive rate (i.e. number of correctly allocated orders to the respective machine) of each machine. **Figure 20** illustrates the confusion matrix of the classification scheme, with the element (i, j) reflecting the number of orders that were predicted to be executed in machine j when the optimal selection was the machine i. As evidently shown by the non-diagonal elements of each row, a misclassification rate of 3/20, 6/20, 0/20, 5/20 and 4/20 was derived for machines 1–5, respectively [53].

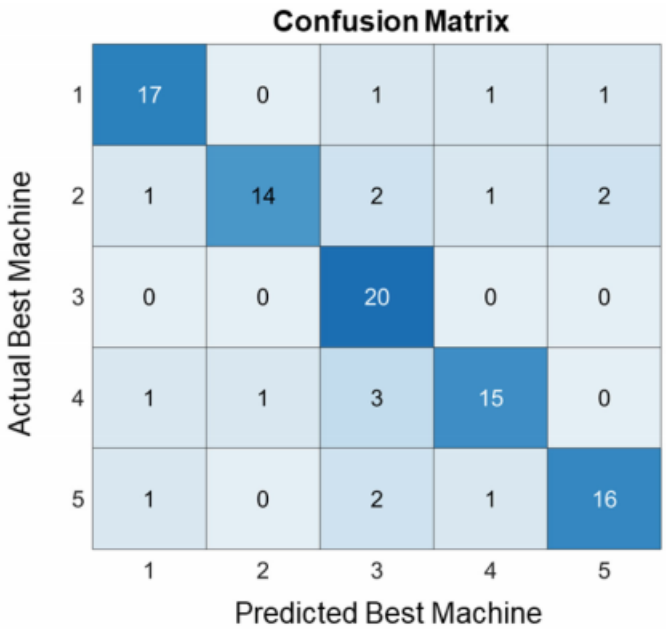

Figure 20 *Confusion Matrix of the Machine Selection policy on the evaluation dataset. Diagonal elements represent the number of orders that were correctly classified to the respective machine* [53].

4.2.4 Conclusions

In this work, an ML-based policy for assigning orders to printing machines is proposed. Using a real dataset from an offset printing company, we trained a machine-specific multi feature regression model to accurately predict the defect ratio for new orders, targeting at proactively selecting the proper machine to execute a given order. In addition, ten different

regression schemes were contrasted for comparative purposes, including classical ML, ensemble and Deep learning methods. Towards minimizing the defects, numerical outcomes confirmed that the proposed machine selection scheme can efficiently provide order accuracy predictions, with the ANN and ensemble methods outperforming the rest of the regression algorithms [53].

5. Concept Drift

Driven by the ever-growing evolution towards Industry 4.0, Zero Defects Manufacturing (ZDM) concept plays a critical role in the competitiveness of production industries [28]. According to Industry 4.0 expectations, creative technologies, new products and novel procedures are getting involved in the production chain, generating a huge amount of data. Monitoring of process parameters, continuous quality control and on-line predictive maintenance are central pillars of ZDM. Under those circumstances, the fast and real-time prediction, based on massive streaming data processing, becomes a critical factor in the accomplishment of ZDM targets. Importantly, massive data are continuously gathered during the system operation, thus posing significant challenges in the data storage and manipulation for later analysis. Machine learning and, in particular, Stream Learning (StL) has the potential to solve problems related to data analysis, such as class imbalance, noisy and/or abnormal data, mainly due to its memorization and generalization capabilities [17], [29].

The concept of drift, which can refer to either real drift, which is the change of the target variable while the statistical properties of the data remain unchanged, or virtual drift, which is the change in the data distribution while the target variable stays stable, is a common issue in streaming data. [29]. **Table 9** shows the statistical properties of a drift. Occasionally we can also observe a combination of both virtual and real drifts [120].

Table 9 *Statistical properties of a drift*

Drift	Condition	Result
Virtual Drift	$(P_t(X) \neq P_{t+1}(X))$	$(P_t(y\|X) = P_{t+1}(y\|X))$
Real Drift	$(P_t(X) = P_{t+1}(X))$	$(P_t(y\|X) \neq P_{t+1}(y\|X))$

Concept drift can lead to a severity of problems. One of the problems that occurs is the model degradation and poor prediction performance. In the context of zero defect manufacturing this situation may cause different problems such as degradation of product quality, increasing maintenance cost of a machine and increasing cost of model maintenance due to the frequent model training. Also, security concerns may arise in the absence of a drift detection mechanism, that is capable of handling a malicious gradual change in data.

Dealing with concept drift is an important area of research, and various techniques, such as online learning and adaptive algorithms, have been developed to moderate its impact on machine learning models.

5.1 Concept Drift Characteristics and Types

In industrial environments the main properties that characterize concept drift depending on the speed of changes, the duration and the recurrence.

Speed of change is the speed that the concept changes over time and can be distinguished in sudden, incremental and gradual. A sudden concept drift may occur in cases that a concept change is abrupt at a specific point in time of the control-flow process. Moreover, incremental drift refers to a smooth-and-slow change of the old concept towards the new concept, whereas, according to gradual concept drift, a new concept gradually replaces an old one over a time period [120], [206], [207].

The duration of concept drift may be either permanent, where the concept may stay at the new condition after the drift or temporary where the concept reverts to its previous condition after a random time period [120], [206], [207]. Duration also can be characterized as the speed of change. An abrupt change is a fast change while incremental and gradual have larger duration [207].

Finally, recurring concepts involve the reactivation of a previously encountered concept at a given instance in time, while non-recurring concept changes only occur once [120], [206], [207]. Recurring drifts can be distinguished in cyclical and non-cyclical and can also be characterized by the frequency of the cyclicity, the duration of the cycle and the duration of the drift[207].

We can see the characteristics of concept drift listed in **Table 10**.

Table 10 *Concept Drift Characterization*

Characteristic	Type
Speed of Change	Abrupt Incremental Gradual
Duration	Temporary Permanent
Recurrence	Recurring Concepts Non-Recurring Concepts

5.2 Detection Techniques

The phenomenon of concept drift, where the statistical properties of the target variable change over time, poses a significant challenge for machine learning models. To cope with the issues that arise from the concept drift, several drift detection mechanisms have been recently proposed, falling into one of the main three categories as mentioned below [120], namely Error Rate-Based Drift Detection, Data Distribution-Based Drift Detection and Multiple Hypothesis Test Drift Detection. This chapter delves into various concept drift detection techniques, exploring their characteristics and considerations.

5.2.1 Error Rate-Based Drift Detection

Some of the simplest yet effective ways to detect concept drift is through the monitoring of performance metrics and error rate. Accuracy, precision, recall, and F1-score provide valuable insights into the model's predictive capabilities. Monitoring error rates is a direct approach to detecting changes in the model's predictive performance. An increase in prediction errors, reflected in metrics like Mean Squared Error or Mean Absolute Error, can indicate the occurrence of drift.

One of the most popular concept drift detection methods of this category is Drift Detection Method (DDM) [130]. In this work, authors observe if the online error rate significantly change to reach the warning level or the drift level. Warning level triggers the training of a new base classifier and drift level replaces the outdated classifier with the new one. An extension of DDM is the Early Drift Detection Method (EDDM) [208]. Here authors improve DDM by calculating the average distance and the standard deviation between two errors in order to deal with gradual concept drift.

These methods are straightforward, intuitive and easy to implement making them accessible for practitioners without extensive statistical or mathematical backgrounds. Also, these methods require minimal additional complexity in model architecture and thus they consume less computational resources. Also due to the various metrics that can be used, these methods allow to choose metrics that are most relevant to specific application. Although, the effectiveness of error rate-based methods can depend on the type of concept drift. Gradual drift may require different thresholding strategies compared to sudden drift events.

5.2.2 Data Distribution-Based Drift Detection

Data distribution-based drift detection methods focus on capturing and quantifying changes in the statistical properties of the input features, the target variable, or both. These algorithms capture the statistical properties of historical data and compare the with the new data. They also have the ability to address the exact position where the change occurs. Different types of windows have been used such as: fixed windows, sliding windows, adaptive windows and combinations of the aforementioned type of windows. Although, these algorithms are more expensive computationally and need more memory to keep track of historical and new data windows [120].

5.2.3 Multiple Hypothesis Test Drift Detection

These algorithms have the ability to combine and use methods from the previous detection techniques. All previous detection techniques can be integrated in an ensemble of drift detectors and provide multiple parallel hypothesis tests in order to effectively capture different types of concept drift.

5.2.4 Drift Detection Algorithms

In this chapter some of the most cited drift detection methods are presented, namely Drift Detection Method (DDM), Early Drift Detection Method (EDDM), Hoeffding's Drift Detection Method (HDDM), Adaptive Windowing (ADWIN) and Kolmogorov-Smirnov Windowing (KSWIN).

DDM [130] monitors the performance of a base classifier and calculates the probability (p_i) and the standard deviation (s_i) of the classifier to make wrong predictions. DDM generates warnings or drifts based on predetermined levels of deviation. More details about DDM are provided in section 5.6.3.

EDDM was proposed by [131] in order to improve the proficiency of DDM in gradual drifts while maintaining excellent performance in abrupt drifts. EDDM take into account the average distance (p'_i) between the prediction errors. The distance increases when the classifier makes accurate predictions and decreases otherwise. The algorithm calculates the average distance (p'_i) and the standard deviation (s'_i) and when ($p'_i + 2 \times s'_i$) reach the maximum value it stores them as (p'_{max}) and (s'_{max}). Then, two thresholds are defined in order to indicate the warning level (α) and drift level (β). If $(p'_i + 2 \times s'_i)/(p'_{max} + 2\, s'_{max})$ is smaller than the threshold (α) or (β) warning and drift is indicated respectively.

HDDM provided by paper [132], proposes two methods for concept drift detection. The first one called $HDDM_A$ monitors the predictions and compares the moving averages of null hypothesis and alternative hypothesis (drift). Hoeffding's inequality is used in order to calculate the error bound (ε_δ) which is the maximum difference between the moving averages. By providing a significant level of (δ):

$$\varepsilon_\delta = \sqrt{\frac{1}{2n} ln \frac{1}{\delta}} \tag{12}$$

where n is the number of observations. Two significant levels are provided for warning and concept drift indication. The second method $HDDM_W$, enrich the $HDDM_A$ with McDiarmid's inequality in order to compare the weighted moving averages. In this sense, the new observations get more weight than the older ones.

ADWIN proposed by[139], uses two sub windows originated from a sliding window in order to perform statistical test for null hypothesis. ADWIN, also uses Hoeffding's inequality in order to calculate the error bound (ε_{cut}) from equation (13):

$$\varepsilon_{cut} = \sqrt{\frac{1}{2m} ln \frac{4}{\delta'}} \tag{13}$$

where m is the harmonic mean of the number of sub windows observations and $\delta' = \frac{\delta}{n}$ with (δ) the significant level and n the number of observations in the sliding window.

KSWIN proposed by [140], uses Kolmogorov-Smirnov to compares the moving averages of null **hypothesis** and alternative hypothesis (drift). It uses a small window to compare the two distributions with error bound calculated from equation (14):

$$\sqrt{\frac{ln\ a}{2r}} \tag{14}$$

where a is the confidence level and r the size of the window.

5.3 Adaptation Strategies

In order to adapt to concept drift, several strategies have been proposed. Blind strategies periodically retrain the base model with the latest data collected from a predefined window. Another subcategory of blind methods is the continuous and online training of the base model as the data arrive. A trustworthy adaptation strategy is the informative method where the model starts the training when a detection mechanism detects a change in the concept [206]. Once a drift is detected from the drift detector mechanism, the base classifier should be adapted to the change. In principle, there are three methods to deal with base classifier adaptation, as tabulated in **Table 11**.

Table 11 *Classifier Adaptation to Concept Drift* [29].

Strategy	Old Concept	Drift	New Concept
Retrain Classifier	Classifier makes predictions	After drift detection retrain the classifier.	The same classifier makes predictions.
Ensemble	Ensemble makes predictions.	After drift detection train a new classifier and replace an outdated one.	The same ensemble with new member makes predictions.
Partially Train Classifier	Classifier makes predictions.	After drift detection partially train the classifier with new data.	The updated classifier makes predictions.

The first method is based on retraining the base classifier. This is a simple method that has specific drawbacks, given that retraining the base classifier is a time-consuming task and when

the old concept reoccurs, the new base classifier fails to make accurate predictions [120]. The second method takes advantage of the ensemble learning approach, by configuring an ensemble of cognitive classifiers. This technique combines the prediction obtained by several classifiers with a voting mechanism in order to achieve enhanced predictive performance. To update the models included in the ensemble, various multi-objective algorithms, focused on different aspects of the problem, have been developed and validated [120], [121], [122]. The core idea of these algorithms includes the process of updating the models by training new classifiers in the framework of the new concept and, then, replacing the outdated ones. This method can be beneficial in the presence of recurring concepts, the replaced models can be reused and eliminate the cost of model training. The last adaptation method is to partially train the classifier using data related to the new concept. Such classifiers have the ability to be semi-trained throughout the system operation and data arrival, characterizing them as suitable tools for online learning. Furthermore, these classifiers may have access to previously-gathered data and exhibit accurate predictions in recurring concepts [29], [120].

5.4 Evaluation

Regarding the evaluation procedure, several methods have been proposed, including the holdout, prequential [120], [121], [209] and controlled permutation methods [120], [121]. The prequential method doesn't need memory for storing data for later use (training) and can get advantage of the entire data set for training. Under these terms prequential evaluation is the most suited and widely-used method for stream learning [120], [121], [126], [128], [129]. Prequential evaluation is comprised of two distinct stages: testing the classifier with each arriving data point and, then, training the classifier with the same data point [29], [120], [121], [209].

Different evaluation metrics have been proposed for the performance of the drift detection mechanisms, including the accuracy and error-rate of the predictor, RAM-Hours and Kappa-Statistic [120]. However, these measures are not accurate descriptors of the drift detection mechanism, usually failing to represent the actual drift detection performance. To that end, alternative metrics have been suggested, such as (i) the mean time to false alarms (MTFA), (ii) the mean time to detection (MTD), (iii) the false alarm rate (FAR) and (iv) the average run length (ARL) [29], [121].

5.5 Datasets for concept drift

In the literature it common to use artificial datasets. Artificial datasets have the advantage that are fully controllable in the sense that we already know when a drift occurs and we also have information about the type of the drift. Thus, artificial datasets are attractive solution for concept drift detection. The following list provides a brief description of artificial datasets that authors in [120] highlight as most cited:

- **SEA** dataset was introduced by [210]. SEA dataset contacts a binary classification problem with abrupt concept changes. It consists of three numerical attributes with range between [0,10]. Two of the attributes are relevant to the target variable. A

decision boundary is used in order to generate a drift given by $f_1 + f_2 \leq \theta$. Different value of θ gives different concept.

- **STAGGER** dataset is an artificial dataset with abrupt concept changes introduced by [211]. It has got three features that each one take three discrete values for binary classification.
- **Hyperplane** was introduced by [212]. It is a dataset for classification that can represent abrupt and gradual concept changes by changing the weights in a function that generates samples.
- **LED** dataset introduced by [213] contains 7 binary features that represent light emitting diodes digit states of a seven-segment display and 10 targets ranged [0,9]. The drift depends on the number of changed attribute values.
- **Waveform** dataset also introduced by [213]. It has got 21 features and 3 classes. Each feature is generated from three randomly selected base waveforms. The concept change is introduced with the change of the base waveforms.

Except from the artificial dataset many real-world datasets were introduced from the literature for concept drift detection. Some of the most cited according to [120] are listed below:

- **Electricity** was introduced by [214] and analyzed by [130]. The data was collected from the Australian New South Wales Electricity Market. The dataset also named ELEC2 contains 45,312 instances where 48 instances consist of data of one day. Each instance includes 5 features corresponded to the day of week, the time stamp, the New South Wales electricity demand, the Victoria electricity demand, the scheduled electricity transfer between states and the class label. The class label is a binary value relative to a moving average of the last 24 hours. The class label reveals the change of the price (if it goes UP or Down) in New South Wales.
- **Poker-hand** dataset introduced from [215] formulates a classification problem. The dataset contains 1,025,010 samples. Five cards are represented with 10 features using two attributes, namely suit and rank. The predicted class takes values between [0,9], values close to nine indicate a good hand.
- **Forest Cover Type** [216] is the actual forest cover type for a given area (30 x 30-meter cell) and was determined from US Forest Service (USFS). The dataset consists of 54 features (10 quantitative variables, 4 binary wilderness areas and 40 binary soil type variables). The target ranges between [1,7] corresponding to: (1) Spruce/Fir, (2) Lodgepole Pine, (3) Ponderosa Pine, (4) Cottonwood/Willow, (5) Aspen, (6) Douglas-fir and (7) Krummholz.

5.6 Case Study

In this section, the employed experiment setup and methodology are presented. Six classifiers 1) Naïve Bayes (NB); 2) Hoeffding Tree classifier (HT); 3) KNN; 4) PAC; 5) SGD Classifier; and 6) VFDRC are introduced in order to compare their performance and reveal their impact on DDM. Initially, a dataset with 60.000 data samples is used to provide the testing set with the first 1.000 data points employed for training purposes. After the training

phase, the classifiers are utilized to generate predictions to the entire dataset. For evaluation purposes, two classes are taken into account and the output of the classifiers is represented as a binary vector, according to the respective class. Then, the predicted class values are compared to the real class values of the dataset. This procedure results into the characterization of the classification either as true (represented by 1) or false (represented by 0). These values are then given to the drift detector to test for drift occurrence. Afterwards, accuracy and confusion matrices, number of detections, true detections, false detections and missed detections are employed to measure the performance of the base classifiers and the drift detectors. In the following subsections the dataset used, the drift detection method, the base classifiers and the evaluation methodology are presented [29]. In Figure 21 experiment setup is depicted.

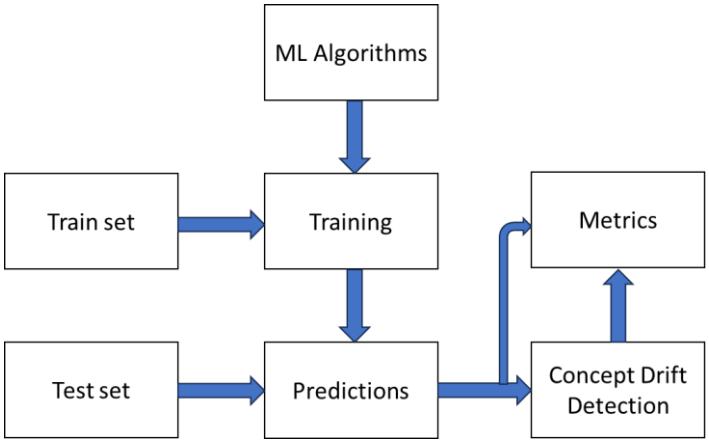

Figure 21 *Experiment setup procedure*

5.6.1 Classifiers

For comparison purposes, several classifiers are used, namely the Naive Bayes (NB) and Hoeffding Tree (HT) classifiers, since they are widely used in the field of stream learning [128], [129]. To enhance the comparison of the impact of base classifiers to the drift detector, classifiers that can deal in streaming environments and have the ability of incremental learning are additionally used, namely k-Nearest Neighbors classifier (KNN); Passive Aggressive classifier (PAC), Stochastic Gradient Descent (SGD) classifier, with hinge as loss function that results in Support Vector Machine (SVM) and, finally, Very Fast Decision Rules classifier (VFDRC). The aforementioned algorithms are implemented via scikit learn and scikit multiflow packages [87], [205]. The main characteristics of these algorithms are mentioned in Section 2.3 [29].

5.6.2 Dataset

For simulation purposes an artificial dataset of 60.000 data points was used. The above-mentioned classifiers were trained in the first 1.000 data points and were then tested against

the entire dataset. The dataset used for this experiment is an artificial dataset generated as in [210] from the Streaming Ensemble Algorithm (SEA dataset). SEA was selected because it's a widely used artificial dataset regarding concept drift detection and handling, with abrupt concept changes [120]. SEA Generator from scikit-multiflow [210] is used to generate data streams with abrupt concept changes. In specific, it produces three numerical attributes, two of which are relevant to the class value and one irrelevant in the interval [0, 10] and two classes, represented by Boolean values. As mentioned earlier, the first data class is represented as one (1) and the second class as zero (0). The sum of the two relevant attributes is compared to a threshold value. In order to incorporate concept drift occurrences, the threshold value changes from 8 to 9, then to 7 and, finally, to 9.5 every 15.000 data points. Furthermore, balance classes parameter was set to false and class imbalance was created between the number of instances that belong to the first and the second classes respectively. The data distribution of the two classes is tabulated in **Table 12**. In addition, random noise was taken into account, resulting in a 10% probability of class change. The same SEA outcome dataset is used to obtain a common denominator for all the methods included in this work, allowing for direct comparisons among the results [29].

Table 12 *Dataset Class Imbalance Table* [29]

Data		1st Class	2nd Class
Train Set (1000)}	Number of instances per class	656	344
	% of instances that belong to class	65.60%	34.40%
Test Set (60000)	Number of instances per class	36889	23100
	% of instances that belong to class	61.48%	38.52%

5.6.3 Drift Detection Method

In the present work, the DDM algorithm was selected because of its simplicity, its adaptability to any classifier and its ability to address stream learning. For each data point the algorithm calculates the probability p_i and the standard deviation s_i of the classifier to make false predictions. In each datapoint a comparison is made between p_i and the minimum value of observed till i^{th} datapoint, named as p_{min}. If $p_i < p_{min}$ then $p_{min} = p_i$. The same procedure is taking place for s_i and s_{min}. If $p_i + s_i \geq p_{min} + 2 \cdot s_{min}$, then a warning is triggered (warning level), giving information for a potential change. If $p_i + s_i \geq p_{min} + 3 \cdot s_{\min}$, the concept is changed and a drift is triggered (drift level). The warning level and drift level were introduced in [130] and can be parameterized to change the performance of the algorithm. In the proposed scikit-multiflow based implementation three parameters may vary [29]:

- Number of instances before permitting the detection of change, which for the purpose of our work has been defined equal to 30 (the default value).
- Warning level deviation that in our work holds the value of 2 (the default value).

- Drift level deviation. Many values for drift level were tested with HT to find the most appropriate for the proposed experiment. After numerous simulations, drift level deviation is set to 2.5, as it outperforms the other values. The results are in accordance with [126]. This change makes the detector more accurate to real drifts.

The same values were implemented in all tests in order to compare the impact among different classifiers in DDM [29].

5.6.4 Evaluation and Metrics

The prequential methodology was adopted for the evaluation procedure [209]. In this framework, all data points are initially provided to the base classifier to make predictions (testing phase) and then are used for training purposes (training phase). The accuracy metric is used to compare the performance of each base classifier, as depicted in **Figure 22**. Moreover, in **Table 13** the confusion matrix is used to compare the performance of each base classifier, where True Positive (TP), True Negative (TN), False Positive (FP) and False Negative predictions (FN) are tabulated. Finally, in **Table 14** the processing time of each algorithm is presented, alongside with the true, false and missed number of detections. As false detections, we define the number of times that the drift detector either triggers a drift flag without the occurrence of a real drift or triggers a real drift that has already occurred and detected. As true detections, we calculate the first detection after a real drift and missed detections are represented as “one” if no detection between two real drifts occurs [29].

5.6.5 Results

In this section, the simulation results are presented. Six base classifiers are used in order to compare their performance and reveal their impact on DDM. A dataset with 60.000 data samples is used to provide the testing set with the first 1.000 data points employed for training purposes. After the training phase, the classifiers are utilized to generate predictions to the entire dataset. The predicted class values are compared to the real class values of the dataset. The outcome of the comparison, true (represented by 1) or false (represented by 0) is fed into the drift detector to test for drift occurrence [29].

The main difference of this work is that instead of using the prequential error rate to compare the drift detector performance we use the base classifiers accuracy to address their performance impact in DDM [29].

In Figure 22, the results based on the accuracy of predicting the correct class for each classifier in the SEA dataset are depicted. Colored lines represent the drift detection accuracy over time for each base classifier. Dots in the figure represent the drifts that DDM detects for the setup with each classifier. The exact time of the drift detection for all classifiers is presented in **Table 15** [29].

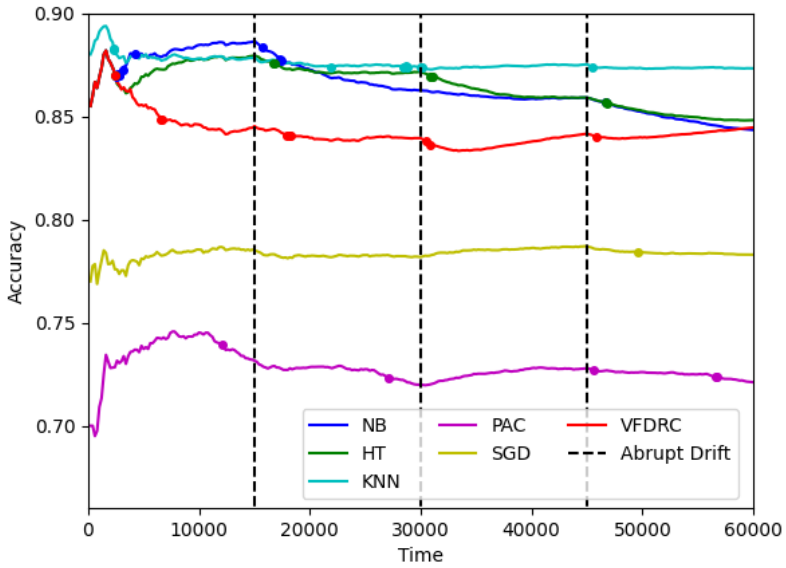

Figure 22 *Accuracy of base classifiers with DDM at SEA Dataset* [29].

As readily observed in **Figure 22**, all algorithms have a short training period (up to 1,000 data points), during which all classifiers improve their accuracy. After the initial training period, the base classifiers are tested on new data instances that have not been encountered during the training (testing phase). Evidently, this causes an abrupt fall of accuracy and DDM triggers a drift. The above condition emerges with NB, HT, KNN and VFDRC classifiers but not with SGD and PAC, mainly because these classifiers are more stable in this time period and they exhibit better accuracy based on the previous data points [29].

As already mentioned, DDM is monitoring the true or false outcomes of the base classifiers to detect concept drift. The results shown in Figure 22 indicate that the accuracy metric fails to represent the classifiers' stability performance. In specific, a sensitive algorithm, such as VFDRC, makes false predictions when concept changes occur, causing a fast deterioration of accuracy and resulting in the triggering of a concept drift. Moreover, algorithms that exhibit stable accuracy performance are misleading the drift detector, thus increasing the miss-detection rate [29].

Table 13 *Confusion Matrix* [29].

	Classifiers					
Metrics	NB	HT	KNN	PAC	SGD	VFDRC
TP	34.241	33.498	33.880	27.729	30.482	33.549
FP	6.748	5.721	4.596	7.574	6.616	5.985
TN	16.363	17.390	18.515	15.537	16.495	17.126
FN	2.648	3.391	3.009	9.160	6.407	3.340

In **Table 13**, the confusion matrix for each base classifier is depicted. For TP, the best algorithm was NB followed by KNN. KNN also performs well in TN making the most accurate predictions, followed by HT and VFDRC. The worst performance was obtained from PAC and SGD in both TP and TN [29].

As shown in **Table 13** and **Table 14**, KNN exhibits the best overall accuracy (87%) and seems to be the most stable algorithm compared to NB (86%), HT (86%) and VFDRC (84%) classifiers, which are typically used for purposes of drift detection. Overall, these algorithms show increased performance, while the noisy data have a significant impact on the model accuracy. In addition, KNN outperforms the other classifiers in terms of the false positives (lowest score) and, more importantly, exhibits higher performance in TN metric (concerning the minority class in the presence of imbalanced data). Although the SGD classifier's accuracy performance is suboptimal (78%), the algorithm exhibits a statistically equal and stable behavior. Finally, it is worth mentioning that the NB outperforms the other algorithms in terms of processing time, with VFDRC being the slowest [29].

Table 14 *Metrics* [29].

	Classifiers					
Metrics	NB	HT	KNN	PAC	SGD	VFDRC
Accuracy	0,84	0,85	0,87	0,72	0,78	0,84
Process Time (seconds)	44	96	262	341	416	463
Detections	12	8	9	5	1	9
False Detections	10	5	6	2	0	6
True Detections	2	3	3	2	1	3
Missed Detections	1	0	0	1	2	0

Regarding the drift detection performance, HT, KNN and VFDRC classifiers show enhanced performance scores detecting all real drifts, with HT performing slightly better due to reduced false detections (5 instead of 6 false detections). On the contrary, NB (10 false detections) and SGD (1 drift detection) failed to yield good results in the presented simulation scenarios [29].

The histogram of true drift detections and false drift detections are depicted in **Figure 23** and **Figure 24**, respectively. The detections of the classifiers with the four highest performances, based on the previous mentioned accuracies, are included. As observed in **Figure 23**, the NB and HT algorithms detect the first drift in close temporal proximity to the real drift, as compared to the other classifiers. Regarding the other two drifting points, this pattern is reversed with KNN showing remarkable results. Furthermore, the majority of false detections are generated by the NB classifier, followed by KNN and VFDRC, as depicted in **Figure 24**. Both histograms indicate that as the time progresses, the drift detectors exhibit enhanced performance, i.e. fewer false detections are observed and gradually, true detections are accurately recognized. These results, however, should be further complemented and verified with extensive simulation results, involving massive datasets and increased numbers of drifting points [29].

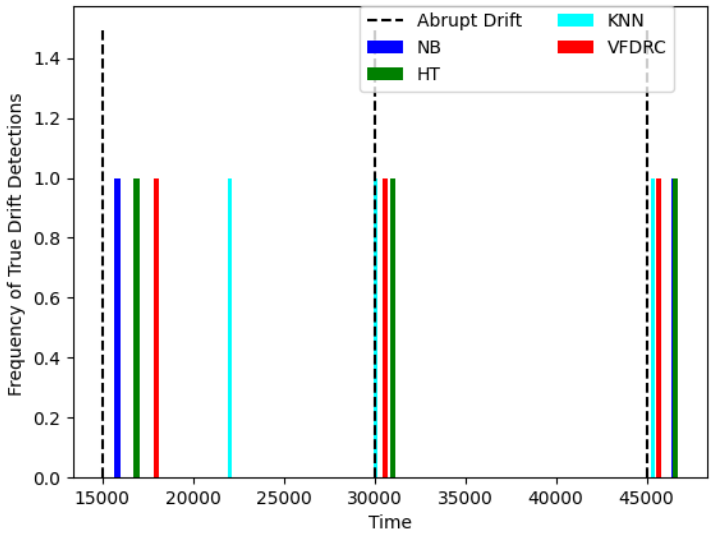

Figure 23 *Histogram of true drift detections* [29].

Comparing the drift detection time points between HT and KNN (tabulated in **Table 15**, it is evident that the maximum delay of detection from HT was 1.745 data points from the first real drift (D2), while, in contrast, KNN maximum delay was 6.925(D3). Moreover, KNN algorithm continues to make true predictions as compared to the HT classifier that makes false predictions at this point. At the other two drift detection data points (30.000 and 45.000), KNN shows enhanced performance (50 and 476 data points delay respectively) in contrast to HT (897 and 1.745 data points delay respectfully) [29].

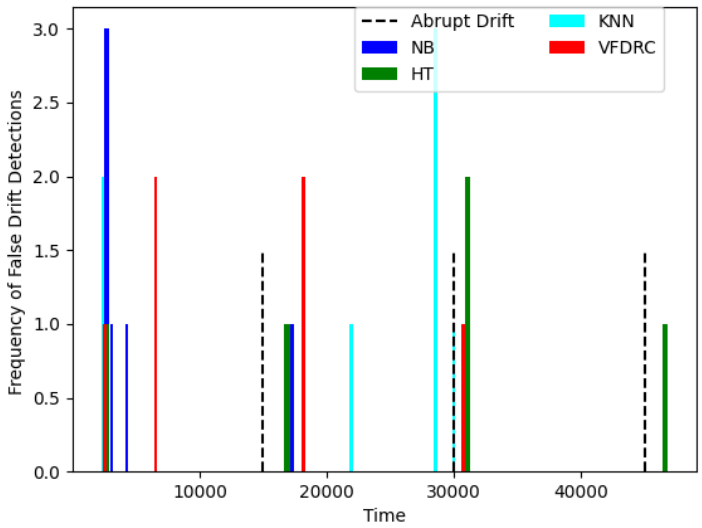

Figure 24 *Histogram of false drift detections* [29].

Table 15 *Drift Detection Time Points* [29].

D	NB	HT	KNN	PAC	SGD	VFDRC
1	2.535	2.415	2.244	12.098	49.580	2.415
2	2.614	16.745	2.277	27.150	ND	6.579
3	2.675	16.825	21.925	45.648	ND	6.614
4	2.711	30.897	28.462	56.653	ND	17.875
5	2.767	30.975	28.617	56.695	ND	18.094
6	2.815	31.088	28.731	ND	ND	18.226
7	3.096	46.745	30.050	ND	ND	30.479
8	4.191	46.856	30.128	ND	ND	30.864
9	15.679	ND	45.476	ND	ND	45.819
10	17.310	ND	ND	ND	ND	ND
11	17.469	ND	ND	ND	ND	ND
12	46.682	ND	ND	ND	ND	ND

5.6.6 Conclusions

This work is a small-scale comparison with regards to the impact of different classifiers in DDM. The findings suggest that the HT and KNN classifiers outperform the other baseline algorithms. Although the NB baseline detector exhibits higher accuracy compared to the VFDRC algorithm, it fails to yield superior performance in drift detection with DDM. Furthermore, the accuracy metric doesn't directly affect the drift detection mechanism, while being closely associated with the triggering of drift detections. To this end, DDM warning and drift level parameters may be suitably adjusted, taking into account that these modifications may increase the false detection rate and decrease true detections [29].

6. Federated Learning-Aided Prognostics in the Shipping 4.0 Era.

The maritime sector with its diverse services is a major driving force towards global economic growth. In this context, maritime transportation supports international trade activities and logistics and its efficient operation relies on intelligent fleet management, minimal downtime for maintenance tasks and reduced fuel consumption with low-carbon footprint. The integration of wireless communications and sensing technologies, following the IoT paradigm, has started to revolutionize the shipping sector by leveraging the plethora of collected, stored, and processed data [217]. Consequently, future maritime transportation systems are envisioned to employ real-time monitoring and automated optimization methods to achieve ship path optimization, energy consumption reduction, and speed optimization [218], [219]. Still, conventional optimization approaches may not be effective in this area due to the formulation of non-convex problems, necessitating the use of alternative optimization techniques, such as machine learning [37].

Over the past few years, ML algorithms have emerged as a promising solution for handling large datasets and optimizing processes. In such applications, neural networks (NNs) are trained to accurately map input variables to output vectors. However, in situations where the output variables have distinct values, a clustering strategy may be more suitable, as it significantly reduces training time compared to NNs. Generally, ML training can be accomplished through supervised learning, unsupervised learning, and reinforcement learning (RL). In supervised learning, data associations are known in advance, and the algorithm is trained using labeled data. On the contrary, unsupervised learning does not rely on predefined patterns for associating input variables with output metrics. Instead, unsupervised learning algorithms analyze a large dataset to identify any underlying patterns or associations between inputs and outputs. Lastly, in RL, an agent interacts with the operating environment and aims to determine the optimal set of policies based on rewards or penalties associated with specific actions. By accumulating a sufficient number of training samples, it becomes possible to derive the best actions for all possible states [37].

In maritime services, effective ML training involves collecting and processing data from numerous distributed sources, covering a vast geographical area with diverse communication technologies and maritime nodes [220]. This poses challenges for conventional centralized ML approaches, particularly for applications relying on fast and reliable data transmissions, where efficient data gathering over large propagation distances is crucial. Additionally, proper data manipulation techniques are necessary to mitigate the effects of varying propagation conditions. Furthermore, data privacy concerns may arise as in many instances, training data may contain sensitive information specific to a particular maritime component, making it preferable to store it locally rather than transmit it via the wireless interface. To address these issues and ensure privacy preservation, the federated learning paradigm has emerged as an alternative ML architectural approach in recent years. FL aims to accelerate ML execution times while keeping data localized [186], [221]. In FL, maritime nodes utilize shared models trained on extensive amounts of data without the need for central data storage [37], [222].

6.1 Shipping 4.0

Industry 4.0, also known as the fourth industrial revolution, aims to create an environment relying on real-time, intelligent, interoperable, and highly autonomous industrial systems. This vision is based on innovative information and communication technologies, such as CPS, the IoT, and CC [223]. CPS connects physical elements, i.e. sensors, operator panels, and computers, with cyber-elements. The physical elements collaborate and communicate to collect and provide data to the cyber-elements, where management, processing, and decision-making procedures occur. Moreover, IoT enables real-time interconnection of various objects, such as sensors, actuators, machines, and robots, in a secure and reliable manner. IoT relies on diverse communication networks, such as fifth- and sixth-generation (5G \& 6G) networks, Wi-Fi, M2M deployments, and cloud-edge technologies [37].

Meanwhile, Industry 4.0 does not ignore the role of humans in the manufacturing processes. In these smart industrial environments, humans are equipped with smart devices and have access to augmented and virtual reality technologies. In addition, they remain actively involved in the manufacturing process by leveraging AI/ML-based decision-making capabilities. By harnessing these advanced technologies, Industry 4.0 has the potential to revolutionize current industrial production processes, benefiting stakeholders, personnel, and consumers, while also promoting environmental sustainability. In the Industry 4.0 ecosystem, a wide range of applications is envisioned, offering flexibility, real-time self-optimization, automation, and the ability to handle complex tasks and meet high quality standards. To enhance these applications, the integration of advanced fault detection, prediction, and prevention technologies are crucial. The main enabler of predictive maintenance (PdM) in industrial settings is the wealth of data from industrial processes, allowing for the precise prediction of machine conditions, remaining useful life, and faults, enabling an appropriate and cost-effective maintenance schedule. As a result, ML-aided PdM algorithms play a significant role towards early and accurate fault detection, leading to minimum downtime of machinery by identifying in real-time, damaged or defective products and parts [37], [200].

The maritime industry ecosystem hosts various activities such as fishing, shipbuilding, shipping, ports, offshore energy, equipment manufacturing, tourism, financial services, and logistics and it is anticipated to undergo significant changes due to Industry 4.0 advancements. According to the "Ocean Economy in 2030" report published by the Organization for Economic Cooperation and Development (OECD), the maritime industries have the potential to double their contribution to global value creation by 2030, driven by increased demand for shipping, shipbuilding, marine equipment, and related services. However, these forecasts are heavily based on the efficient integration of technological innovations introduced by the Industry 4.0 paradigm [37].

In the future maritime ecosystem, known as Shipping 4.0, technology-based innovations will play a fundamental role. More specifically, the main CPSs in the maritime sector, referred to as ``smart ships,'' have the potential to replace conventional vessels in a fully interconnected maritime environment. Smart ships will adhere to new design criteria and operational requirements, leading to enhanced efficiency and sustainability. The development and implementation of the Industry 4.0 paradigm will form the foundation of this future maritime ecosystem, being characterized by ML-aided systems for autonomous navigation, PdM and fleet management [37].

As the shipping industry experiences increasing technological advancements and digitalization, it is undergoing a process of digital transformation, being driven by the growing demand for improved data collection, processing, and networking capabilities [224]. It is evident that the future of the shipping industry will be increasingly relying on advanced IIoT systems for real-time acquisition, transmission, storage, and analysis of large volumes of relevant data. In this sense, Shipping 4.0 is expected to offer significant benefits to the shipping industry, including reduced operational costs, increased overall revenue, and extended machine service life [37].

6.2 Methodology and Use Cases

In this section, three PdM uses cases are presented, along with the associated methodology, aiming at providing a practical overview on how Shipping 4.0 can get rid of the FL-based PdM schemes. Key motivations to promote the adoption of FL-aided PdM approach in the future smart shipping lies in the obstacles faced by the traditional centralized architecture, which requires: (i) centralization of the data collection (violating the privacy-sensitivity of the maritime data), (ii) frequent transmissions of data under coarse propagation conditions, (iii) high-dimensionality of the central model to ensure generalization of the model on the massive multi-source data (posing concerns on whether the low-capacity edge servers of vessels can execute those large models). Further, based on the performance variability presented by different FL approaches, the application of separate widely-used FL algorithms in the deployed use cases is illustrated. Note that, given the intrinsic and heterogeneous patterns of a multi-source (e.g. containing more than one vessel) dataset, the suitability of a given FL algorithm to a specific PdM optimization task cannot be *a priori* known and can be found upon extensive experimentation [37].

For completeness purposes, three different scenarios are selected, each one requiring different ML solution category (regression, classification and time-series forecasting/regression). Specifically, the following use cases are considered [37]:

- **Use Case 1** - Prediction of Naval Propulsion Gas Turbine Measures: In this scenario, a multivariate regression ML model is constructed to enable Condition-based PdM of Naval Vessels (Navy Frigate) equipped with Gas Turbines (GT). Specifically, based on an open dataset [225], [226], an FL-based scheme is developed targeting to collaboratively predict the GT performance degradation of Frigates, as GT status has serious impact on the vessel's propulsion system. The GT performance is represented by means of 2 GT parameters, namely the GT decay state coefficients of the Compressor and the Turbine, and is estimated using a 16-feature vector as input (or predictors).
- **Use Case 2** - Prediction of Ship's Main Engine Condition: This use case considers another important situation faced by the shipping crew and owners, concerning the continuous monitoring of the Primary Engine Condition (PEC). Given the strong influence of the PEC on the overall ship's fuel consumption and propulsion, here we develop an FL-based multivariate classification model, capable of providing binary alarms ('Failure' or 'Normal') about the Engine's status. This is achieved by exploiting 6 engine features as inputs and the categorical MEC parameter as desired output, according the open dataset in [227] (from IEEE Data Portal).

- **Use Case 3** - Prediction of Upcoming Main Engine Consumption: The third use case concerns the time-series forecasting of the Primary Engine Propulsion Power (PEPP, usually expressed in kW) consumption, so as to allow, in cases of excessive fuel wastes, proactive actions that could be taken by either the Engine's users or the ship captain. To that end, a dataset containing the temporal patterns related to both Engine Propulsion and Automatic Identification System (AIS) data of cargo ships was provided by a maritime enterprise [30]. The outcome of this use case is an FL-based multivariate Long-Short Term Memory Network (LSTM) that, given both ship- and weather-related information, is able to estimate the upcoming PEPP values as a direct indication of the fuel waste.

More detailed description of the datasets is provided in Section 6.3.

To give a practical overview of the proposed workflow,

Figure 25 outlines the general-purpose sequential diagram required for supporting the employed use cases, involving both Cloud and FL agents components. Components can be either physical or functional. Noteworthy, the Cloud representation could be replaced by any other central entity that is responsible for combining the local model updates, whereas the Edge components stand for any maritime node that is engaged in the FL process [37].

The PdM sequence unfolds as follows [37]:

- **Training Phase**: The use case sequence starts with the **Data Collector** gathering data for ship-related (measuring equipment in the ship) and ship-unrelated (sensors attached in the ship) parameters. The former refer to any measurement that is associated with the ship profile (position, speed, trim, heading, etc.) and engine (rpm, torque, oil level, etc.), whereas the latter concern the external/environmental parameters affecting the ship voyage (wind direction, wind intensity, weather, etc.). Upon the initiation of the FL process, raw data are sent to the **Data Preprocessor**, which includes the toolset to complete the preprocessing (cleaning, smoothing, data type transformation, dimensionality reduction, etc.). The **Model Trainer** should then initiate the hyperparameters of the local model and define the FL method that it will follow (FedProx, FedAvg, etc.). Possible tuning iterations of the local model takes place also in the Model Trainer. When the stabilization of the local model is achieved, the sequence enters the FL loop, where firstly the Model Trainer executes N steps of the Stochastic Gradient Descent (SGD) algorithm to derive the model weights. Next, local parameters from all agents are collected by the **Parameter Collector**, which then forwards them to the **Model Combiner**. When Model Combiner receives the optimizer setup (FL method used for parameter combination, e.g. FedProx, FedAvg) from the **FL Selector**, it is able to produce the global model weights. Finally, the global model is returned back to the FL agents for continuing local training rounds and, when aggregation rounds *M* are reached, the FL model is available for inference in the **Model Inferer**.
- **Inference Phase**: To enable PdM with the deployed FL models, inference data should be first gathered and preprocessed (following the same steps as for the training data) and then, PdM predictions/alarms can be obtained by calling the deployed use-case-specific model.

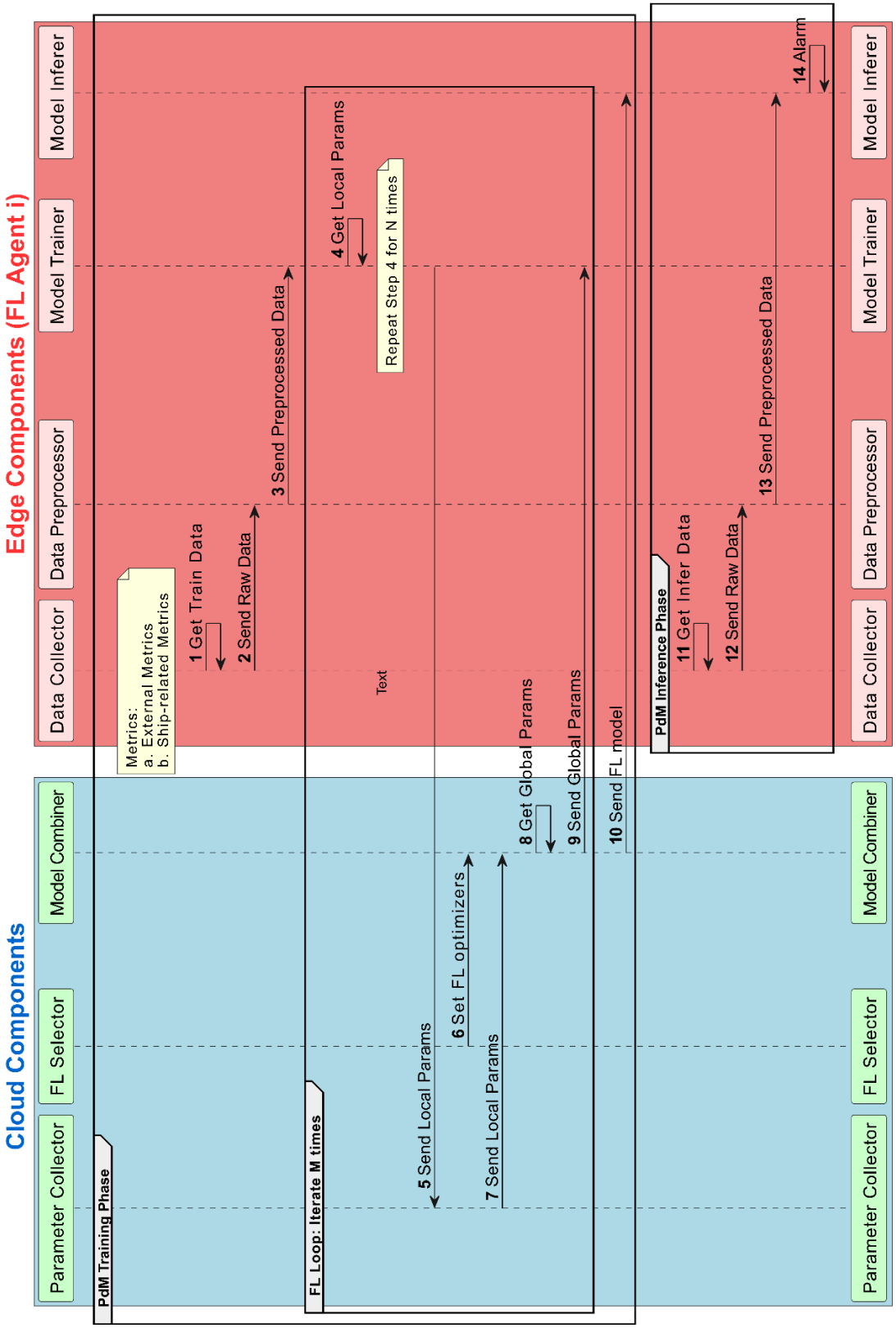

Figure 25 Sequential diagram of the training and inference phases for enabling FL-aided PdM. *'Cloud' refers to the central entity, responsible for combining the local model updates, and 'Edge' refers to the local agents considered as FL participants* [37].

6.3 Dataset Descriptions

This section tabulates the available data for implementing the considered use cases. **Table 16** summarizes the information included in the datasets described in the second paragraph of Section Methodology and Use Cases, containing the deployed local models, the features and the target variables per use case. Some of the metrics presented as features were finally excluded from the models input layer for dimensionality reduction purposes (features presented multi-collinearity with other inputs, or independence with the outputs, were ditched as redundant) [37].

Table 16 *Description of the Datasets used in the 3 use cases* [37].

Dataset	Use case	Model	Feature Metrics (Acronym) [Unit]	Target(s) (Acronym) [Unit]
1	Multivariate regression for the prediction of the Gas Turbine status	FCNN	Lever position (lp); Ship speed (v); Gas Turbine (GT) shaft torque (GTT) [kN m]; GT rate of revolutions (GTn) [rpm]; Gas Generator rate of revolutions (GGn) [rpm]; Starboard Propeller Torque (Ts) [kN]; Port Propeller Torque (Tp) [kN]; Hight Pressure (HP) Turbine exit temperature (T48) [C]; GT Compressor inlet air temperature (T1) [C]; GT Compressor outlet air temperature (T2) [C]; HP Turbine exit pressure (P48) [bar]; GT Compressor inlet air pressure (P1) [bar]; GT Compressor outlet air pressure (P2) [bar]; GT exhaust gas pressure (Pexh) [bar]; Turbine Injecton Control (TIC) [%]; Fuel flow (mf) [kg/s]	1. GT Compressor decay state coefficient (GTC coef) [0-1] 2. GT Turbine decay state coefficient (GTT coef) [0-1]
2	Multivariate classification for the prediction of the Engine Condition	FCNN	Engine rpm (rpm) [rpm]; Lub oil temperature (LOT) [C]; Coolant temperature (CT) [C]; Fuel pressure (FP) [bar]; Lub oil pressure (LOP) [bar], Coolant pressure (CP) [bar	Primary Engine Condition (PEC) [binary
3	Multivariate time-series forecasting for the prediction of the Propulsion Engine Power	LSTM	Speed over Ground (SOG) [kN]; Speed through Water (STW) [kN]; Heading (Head) [degrees]; Continuous Wind Speed (CWS) [m/s]; Discretized Wind Speed (DWS) [bft]; Wind Direction (WD) [degrees]; Draft Forward (DF) [m]; Draft Aft (DA) [m]; Trim (Trim) [m];	Primary Engine Propulsion Power (PEPP) [kW]

Here we present all the features contained in the datasets. Note that, some features have not been included in the final models, as they were rejected by running feature importance tests during preprocessing [37].

6.4 Federated Learning Algorithms

There is a growing field deploying an FL algorithmic toolset, with each algorithm presenting pros and cons depending on the optimization target and the dataset to which is applied on. For example, intrinsic patterns of the data, data distributions and relationships, or different degree of heterogeneity that exists between FL agents, can strongly affect the selection of the FL algorithm and the associated optimizers. We refer to optimizers as the functions implementing the back-propagation algorithms (e.g. SGD, Adam). To this end, four different FL algorithms were used in this study, aiming to derive the optimal FL model per use case, namely [37]:

- **FedSGD** [62]
- **FedAvg** [63]
- **FedAvgM** [64]
- **FedProx** [65]

All the aforementioned FL policies were implemented for comparison purposes [37].

Regarding the local models engaged in FL loop, deep learning models were used in all use cases, as they have proven efficacy when the relationship between inputs/outputs is unknown and non-linear. Fully-Connected Neural Networks (FCNN) were used for use cases 1 and 2, and Long-Short Term Memory Networks (LSTM) was considered in use case 3. Local training rounds targeted to the minimization between the use-case-specific loss function (error between the actual and predicted target values), as: (i) Loss of use case 1 was the Mean Squared Error (MSE) of the 2 GT measures, (ii) Loss of use case 2 was Binary Cross-Entropy (BCE) of the PEC and (iii) Loss of use case 3 was the MSE of the PEPP values. For the ease of exposure, **Figure 26** demonstrates the final structure of the ML models considered for the three use cases [37].

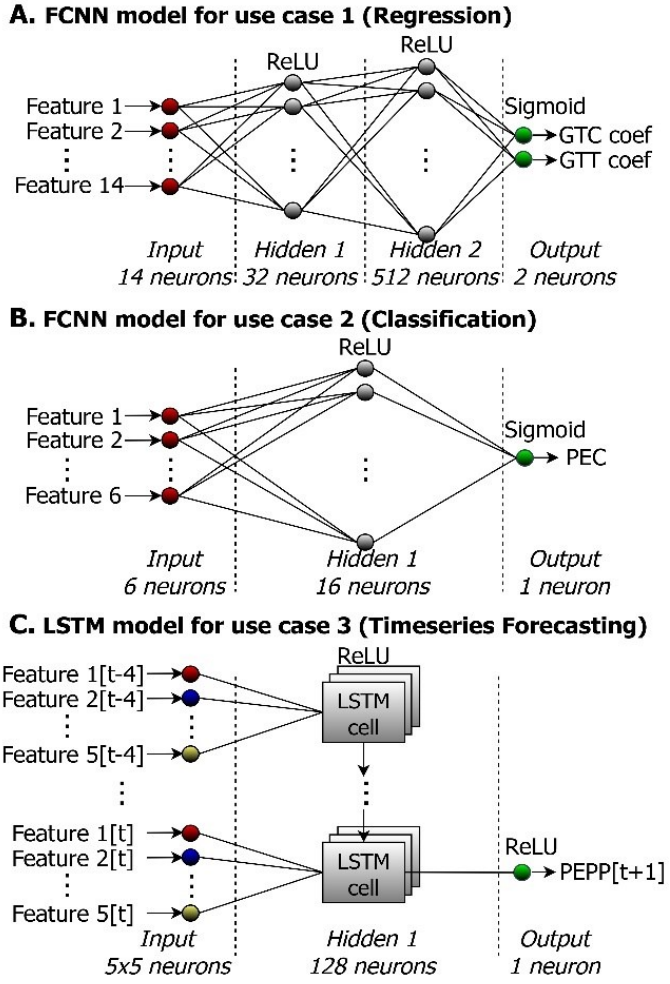

Figure 26 *ML model structure and internal architecture for use case 1 (panel A), use case 2 (panel B) and use case 3 (panel C)* [37].

6.5 Results & Performance Evaluation

The following subsections present the numerical results concerning the three use cases that were considered in this study. For each use case, there were six decentralized FL participants, together with their local datasets. The first five were considered as the FL local agents that are used to build the global FL model during the training phase, whereas the sixth was used for evaluating the FL model on unseen data. In this sense, the comparisons amongst the four FL policies (FedAvg, FedSGD, FedAvgM, FedProx) were conducted based on a dataset that was not encountered during the training for any of the FL schemes to ensure fairness and to assess the generalizability of the deployed schemes [37].

For the datasets that contained less than six agents, realistic datasets using Generative Adversarial Network (GAN) generators (tabgan) [228] was generated. A variant of the code base located in [229] was used, for purposes of dealing with uneven data distributions and creating non-IID data that could not be discriminated by the real ones. GAN-based data generations were considered only for the first and second use cases, given that there were six ship-specific real datasets in the third use case [37].

Simulations and algorithmic procedures ran on a personal PC with a processor Intel(R) Core (TM) i7-11800H, 2.30GHz, a 32 GB RAM, and a 64-bit operating system. The ML and FL programs were implemented using Python 3.9.12 and TensorFlow library, version 2.12.0, no GPU for model training acceleration [37].

6.5.1 Use Case 1: Prediction of Naval Propulsion Gas Turbine Measures

In the first use case, the Naval Vessel Condition dataset [226] was used. The dataset contains sixteen features that reveal the condition of a gas propulsion plant for Condition-based Maintenance. In the preprocessing stage, we neglected the features that showed non-significant Pearson's correlation coefficients (-0.15<r<0.15) with the target variables (GTC/GTT decay state coefficients). We then normalized all the dataset features in the range [0,1] using the standard MinMax scaler. Afterwards, fourteen numerical features were used to predict the target variables, whereas the regression problem was approached with an FCNN model (per FL participant), since the relationship between inputs and output was *a priori* unknown. The features that were excluded were the *GT Compressor inlet air temperature (T1)* and the *GT Compressor inlet air pressure (P1*), as they were fixed values (took one unique value), thus not representing any feature variance [37].

6.5.1.1 Tuning of Learning Parameters

Initially, a series of simulation experiments was conducted for each FL agent to select the optimal configuration of the FL schemes in terms of the critical hyperparameters (learning rate, momentum, proximal strength, optimizers and number of local training epochs per aggregation round). First, the optimizers considered for implementing the backpropagation step of the FCNNs were the SGD, Adam, Adagrad, RMSprop, Adadelta, Adamax and Nadam (see **Figure 27**). Note that, the backpropagation optimizer can be crucial for the final

convergence of the model, since it influences the weights adjustment of the model, which in turn affect the produced model predictions. Notably, it is impossible to know in advance which backpropagation algorithm would be the best and, thus, multiple simulations are required [37].

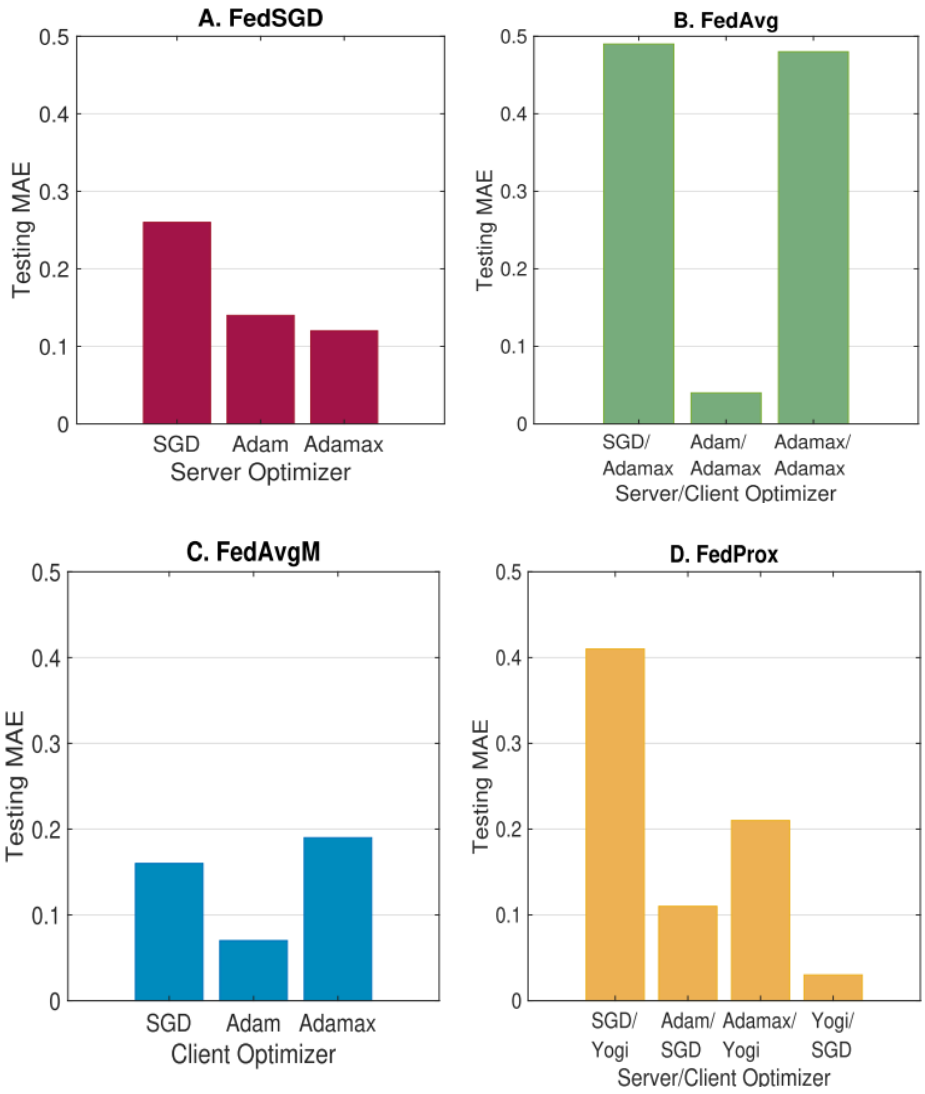

Figure 27 *Use Case 1 (Predicting the naval propulsion GT measures) predictive performance of the four Federated Learning schemes for different optimizer configuration at both server and client sites. Panels A-D correspond to FedSGD, FedAvg, FedAvgM and FedProx, respectively. The evaluation metric is the MAE loss between the actual and the predicted sample values drawn from the testing set* [37].

Regarding the FL model deployment, the TensorFlow Federated (TFF) framework [86] was used for programming decentralized predictive maintenance algorithms in the shipping. Four different FL policies (FedSGD, FedAvg, FedAvgM and FedProx) described in Section 6.4 was used for comparison purposes. In each FL algorithm, extensive simulations were carried out to fine-tune the optimizers both in cloud server side and the FL participants' side. Different values of the learning rate α were also considered for the server and client optimizers, namely α = [0.001, 0.01, 0.1, 1]. In the special cases of FedAvgM and FedProx policies, was

experimented with different momentum values β= [0.1, 0.3, 0.5, 0.7, 0.9, 0.96] and proximal strength μ= [0.1, 0.3, 0.5, 0.7, 0.9] values, respectively. The optimal values of the learning rate, momentum and proximal strength were set upon extensive simulations and are summarized in **Table 17** for each scheme [37].

Considering the optimal values for both policies (β=0.96 for FedAvgM, μ=0.1 for FedProx), **Figure 27** depicts, for each FL policy, the MAE loss as a function of different optimizers. Note that, each ship-specific dataset was divided in training (80% of the whole dataset) and testing (20% of the whole dataset) sets. Thus, to compute the MAE loss, a combined testing set was used which was comprised of the individual testing sets of each ship-specific dataset. As such, the evaluation metric for all schemes was the MAE loss calculated as the mean absolute difference between the actual and the predicted values of GTC/GTT decay state coefficients. The actual values were the ground truth target values as they are included in the combined testing set, whereas the predicted values were the output target values as they are derived by the global FL model, which was trained based on the training sets. As readily observed, the optimal optimizer for the FedSGD scheme was the Adamax which achieves a MAE of 0.12, whereas Adam was the best optimizer (MAE of 0.07) for FedAvgM. In addition, FedAvg showed the optimal accuracy (MAE of 0.03) when the cloud server uses the Adam and the FL participants use the Adamax optimizers, whereas FedProx exhibited the lowest prediction error (MAE of 0.02) for Yogi optimizer at the cloud server and SGD optimizers at the FL participants [37].

Table 17 *Hyperparameter configuration for the four FL schemes considered in the comparisons of use case 1* [37].

Parameters	**FedSGD**	**FedAvg**	**FedAvgM**	**FedProx**
Cloud/ Client optimizerr	Adamax/ SGD	Adam/ Adamax	SGD/ Adam	Yogi/ SGD
Cloud/ Client optimizerr α	0.01/ 0.001	0.1/ 0.01	1/ 0.001	0.1/ 0.001
Cloud/ Client optimizer β	1/ 1	1/ 1	0.96/ 1	1/ 0.96
Proximal strength μ	0	0	0	0.1
Local epochs per client	15	15	5	5

Using the optimal configuration of optimizers for each FL scheme, **Figure 28** shows the MAE loss (calculated as described above) as a function of the number of client epochs per round. In specific, the number of client epochs per round defines how many local training epochs are performed at each FL participant (to update the local model weights) between any two successive aggregation steps (at the cloud server). Noteworthy, this analysis allowed us to show whether an FL scheme requires frequent or rare communication rounds between the FL participants and the cloud, since, if an FL scheme shows the lowest error for low (or high) number of client epochs per round, this means that the aggregation steps should be performed frequently (or rarely) in time. Evidently from **Figure 28**, FedAvg requires 15 local training epochs per aggregation round to achieve the highest accuracy, whereas FedAvgM and FedProx exhibit the best prediction accuracy for 5 local training epochs per aggregation round.

FedSGD policy was independent of the number of local epochs per round (i.e. quasi-constant MAE loss) and, hence, we selected the highest number (15) of local training epochs per aggregation round to decrease the communication overhead between the FL clients and the cloud [37].

For the rest, each FL policy is configured with the optimal learning hyperparameters, as they were derived by the analyses of this subsection [37].

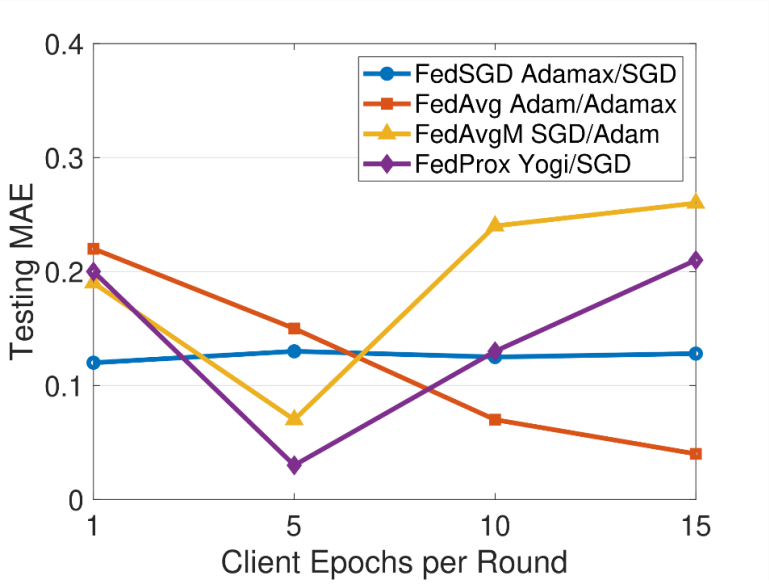

Figure 28 *The impact of the number of Epochs per Round on the Federated Learning performance in Use Case 1. The MAE of FedSGD, FedAvgM, FedAvg and FedProx schemes are depicted, respectively. All schemes are configured with their optimal optimizer, whereas MAE metrics are calculated on the testing samples* [37].

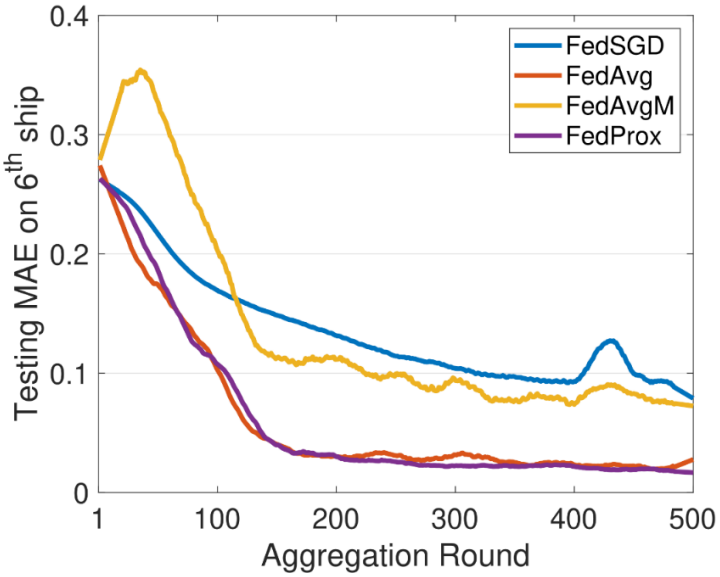

Figure 29 *Learning convergence of the four FL schemes, as a function of the training rounds. In each marker of the curves, the evaluation metric is the MAE computed over the testing set* [37].

6.5.1.2 Performance Comparison

This subsection presents the performance comparison amongst the considered FL policies in terms of the achieved accuracy on predicting the GTC/GTT decay state coefficients. The MAE loss considered for the validation comparisons was calculated as the mean absolute difference between the actual (i.e. the GTC/GTT decay state coefficients of the sixth ship dataset) and the predicted values (i.e. GTC/GTT decay state coefficients derived by inferring the global FL model that was constructed by the datasets of the first five ships). Note that, the sixth ship dataset was kept outside of the FL training phase, so as to be agnostic to each of the compared schemes of this subsection. For ease of exposure, **Table 17** tabulates the final hyperparameter configuration of each FL policy for conducting the comparisons [37].

In **Figure 29**, we demonstrate the MAE (computed over the samples of the sixth ships) as a function of the training aggregation rounds. Specifically, each point of the curves included in **Figure 29** has been derived as follows: for a given round, the FL model is inferred with the samples of the sixth ship, and the MAE of this round is equal to the mean absolute difference between the ground truth and the FL model-predicted values of the GTC/GTT decay state coefficients. Note that, local training epochs take place within two successive rounds [37].

Evidently, all FL schemes show decreased prediction errors as the number of aggregation rounds increases. FedAvg and FedProx outperform the rest of the schemes, achieving a MAE of about 0.02. Noteworthy, both methods exhibit also faster convergence than FedSGD and FedAvgM. In this use case, the similarity of FedAvg and FedProx in terms of their prediction performance had been implied by the proximal strength $\mu = 0.1$ that was found to be the optimal for FedProx. Note that, FedProx is the general case of FedAvg, with FedProx being the same as FedAvg for $\mu = 0$ (see also Equation *(11)*). We conclude that, the FedAvg is the optimal FL scheme for predicting the GTC/GTT decay state coefficients both accurately and efficiently, as it requires less local training epochs per round than FedProx, thus offering optimal performance with rare vertical communications between clients and cloud [37].

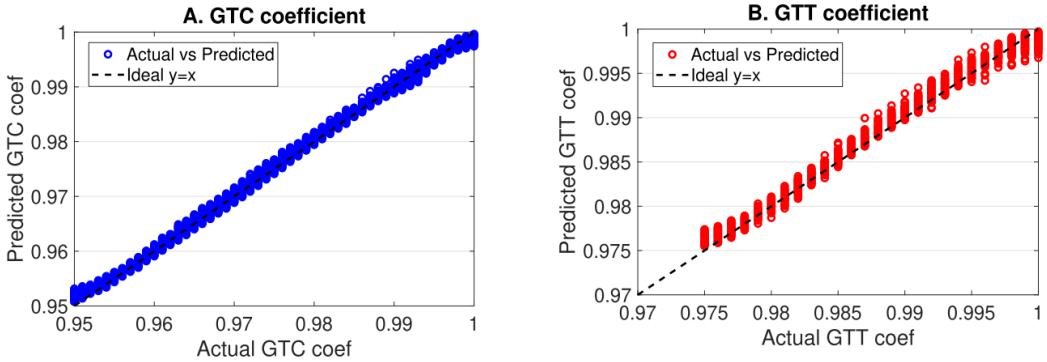

Figure 30 *Actual and predicted values of the testing samples of the GT Compressor decay state coefficient (panel A) and GT Turbine decay state coefficient (panel B) as scatter plots. Both schemes refer to the FedAvg algorithm. The dashed line represents the ideal 'actual vs predicted' matching (y=x)* [37].

To concretely quantify the performance of the best FL scheme (FedAvg), **Figure 30** depicts the actual (x-axis) and predicted (y-axis) values of the GT Compressor decay state coefficient

and GT Turbine decay state coefficient as scatter plots. Both target variables were accurately predicted with a prediction error less than 0.004 [37].

6.5.2 Use Case 2: Prediction of Ship's Main Engine Condition

In order to classify the condition of the ship's main engine, we used a data-set from [227] which contains 6 numerical engine-related features and the binary PEC variable (target). Due to the lack of correlation of features with the target, in the prepossessing step a deviation of the method proposed in [80] to transform the features was used. More specifically each numerical engine-related feature was used to train a decision tree and the prediction probability to belong in the binary target variable was used in order to transform each numerical engine-related feature to a new feature. Also, the number of tree depth was constrained in order to avoid overfitting [37].

6.5.2.1 Tuning of Learning Parameters

As previously, the four FL schemes in terms of the crucial learning hyperparameters was firstly fine-tuned. The tuning procedures were identical to those presented for use case 1. The goal was to obtain optimally-configured FL schemes (FedSGD, FedAvg, FedAvgM and FedProx) prior the comparisons by properly regulating the hyperparameters towards accurate PEC prediction [37].

Table 18 *Hyperparameter configuration for the four FL schemes considered in the comparisons of use case 2* [37].

Parameter	FedSGD	FedAvg	FedAvgM	FedProx
Cloud/ Client optimizer	Adamax/ SGD	Adam/ Adamax	SGD/ Adam	Yogi/ SGD
Cloud/ Client optimizer	0.1/ 0.001	1/ 0.01	0.1/ 0.001	0.1/ 0.001
Cloud/ Client optimizer	1/ 1	1/ 1	0.96/ 1	1/ 1
Proximal strength	0	0	0	0.1
Local epochs per client	10	15	15	15

For each FL policy, we conducted extensive simulations with different server/client optimizers, different learning rates α = [0.001,0.01,0.1,1]. The convergence performance of FedAvgM and FedProx for different momentum constants (β= 0.1, 0.3, 0.5, 0.7, 0.9, 0.96) and proximal strength μ = 0.1, 0.3, 0.5, 0.7, 0.9) was also tested. **Table 18** lists the optimal values of the learning hyperparameters [37].

The evaluation metric used for assessing the classification error was the BCE, which quantifies the ability of the models to correctly classify the samples. As depicted in **Figure 31**, there were different optimizer selections that correspond to low prediction error, depending on the FL scheme. To effectively predict the PEC testing values, FedSGD and FedAvgM require the Adam (at the cloud server side) and Adamax (at the clients side), respectively. The optimal selections for server/clients optimizers were Adamax/Adamax (for FedAvg) and Adam/Adamax (for FedProx). Note that the BCE in **Figure 31** has been computed over the combined testing set (i.e. the concatenated testing sets of the five training ships), given that each ship-specific dataset had been separated into 80%/20% training/testing subsets [37].

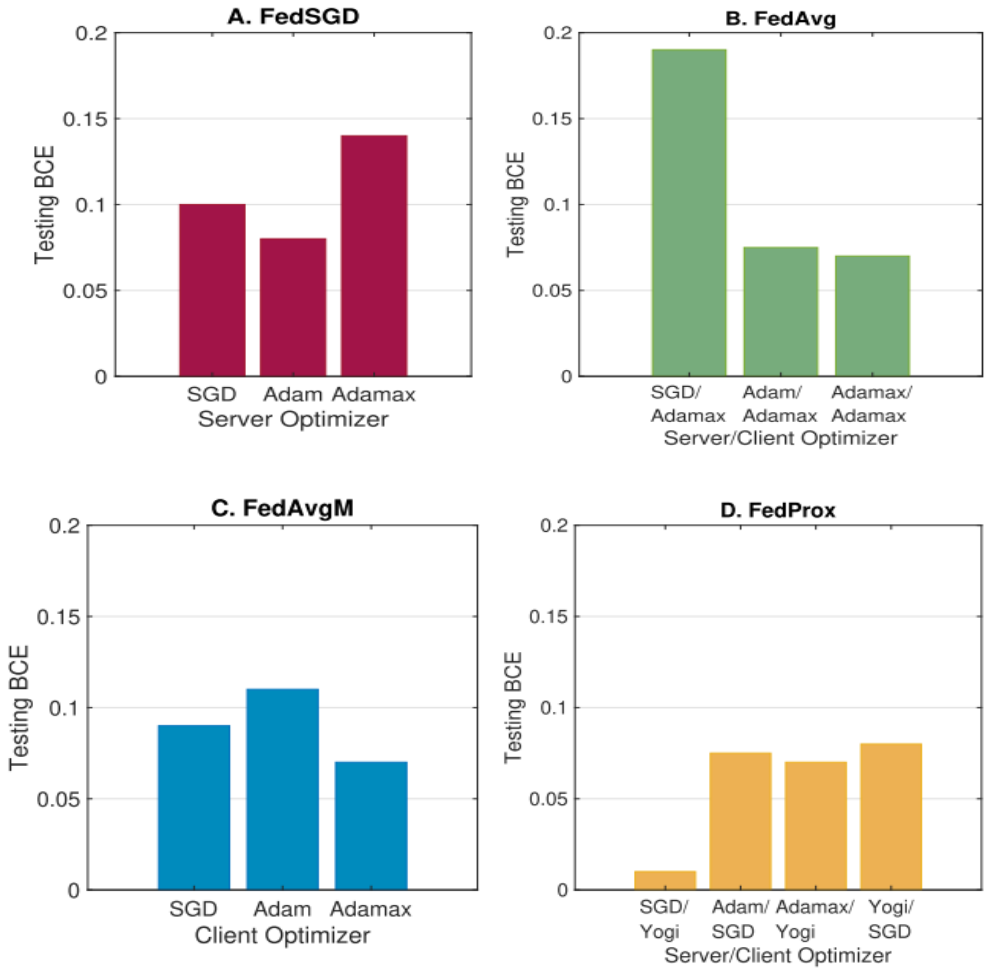

Figure 31 *Use Case 2 (Predicting the engine condition) predictive performance of the four FL schemes for different optimizer configuration at both server and client sites. Panels A-D correspond to FedSGD, FedAvg, FedAvgM and FedProx, respectively. The evaluation metric is the Binary Cross-Entropy (BCE) loss between the actual and the predicted sample classes drawn from the testing set* [37].

To investigate the impact of the number of local epochs per round, **Figure 32** shows the BCE for varying values of the local training epochs. All schemes presented optimal classification accuracy for 15 local training epochs per aggregation round, except FedSGD which showed the best performance for 10 epochs per round [37].

For the rest of the comparisons concerning use case 2, all FL schemes have been set to their optimal parameters according to **Table 18** to ensure fair comparative analysis [37].

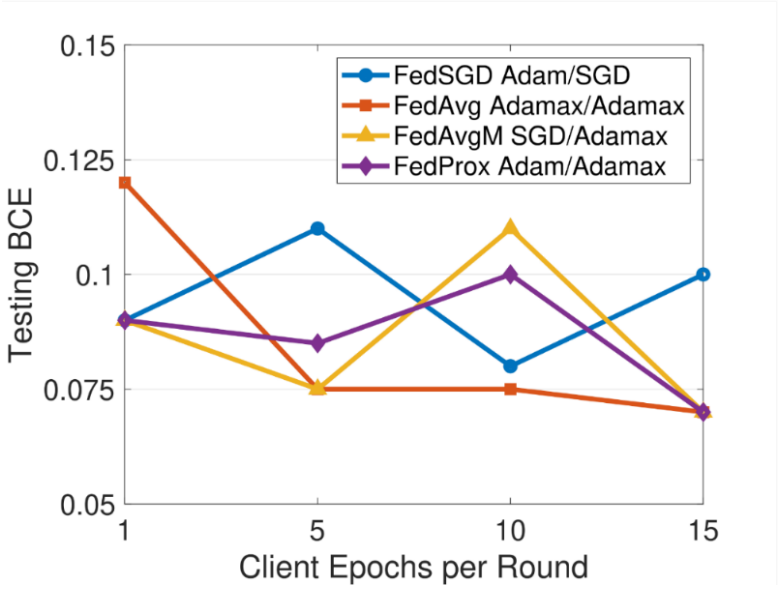

Figure 32 *The impact of the number of epochs per round on the FL performance in Use Case 2. The binary cross-entropy (BCE) loss of FedSGD, FedAvgM, FedAvg and FedProx schemes are depicted, respectively. All schemes are configured with their optimal optimizer, whereas BCE metrics are calculated on the testing samples* [37].

6.5.2.2 Performance Comparison

This subsection presents the comparison amongst the FL policies in classifying the engine condition as normal or abnormal. For the comparisons, using the parameters listed in Table **Table 18**, the classification of the FL schemes as BCE computed over the data samples of the sixth ship was measured. This means that, each FL scheme, which was trained according the federation across the five ship-specific datasets, was inferred by the samples (the 6 input features) of the sixth ship, whereas the BCE was computed based on the actual (as included in the dataset of the sixth ship) and FL model-predicted PEC values [37].

As shown in Figure 33, the learning curve of the FedAvg converges to the lowest BCE scores relative to the rest of the FL schemes. This implies that, when considering PEC prediction PdM problems in a federated or collaborative learning scenario, FedAvg would be the optimal scheme, noticing a classification BCE of 0.075 on unseen data. Noteworthy, in this use case, all methods presented also low communication-wise footprint, since the local weights are uploaded to the server quite rarely (every 15 local epochs) [37].

To further quantify the performance of the FedAvg in this use case, Figure 34 depicts the confusion matrix, which was resulted by inferring the FedAvg model with the sixth ship's input data. It is evident that 34.04% of the samples corresponding to the 'normal' PEC class were correctly classified, and 62.91% of the 'abnormal' samples were also correctly classified. This outcome results into an overall classification accuracy 96.95% for FedAvg model or, in other

words, we conclude that FedAvg exhibits a false positive/negative ratio of 3.05% in the PEC prediction [37].

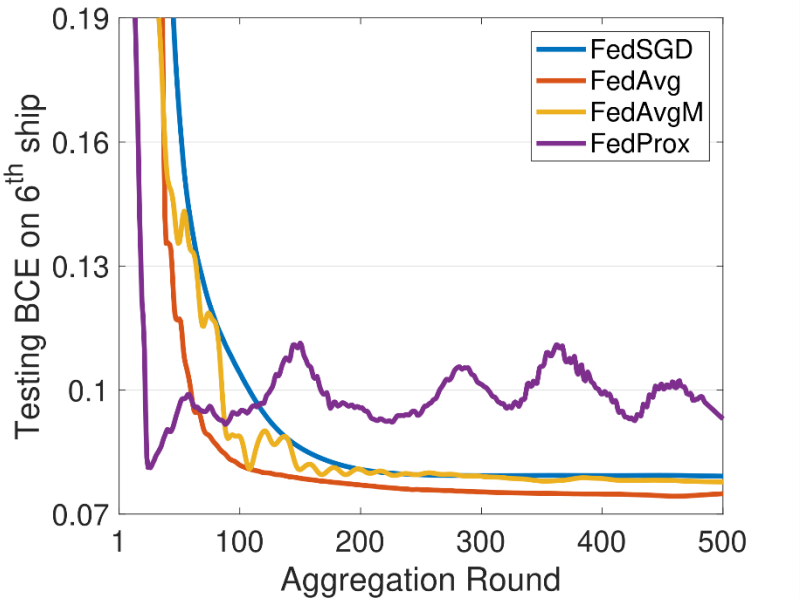

Figure 33 *Learning convergence of the four FL schemes as a function of the training rounds. In each marker of the curves, the evaluation metric is the Binary Cross-Entropy (BCE) computed over the testing set* [37].

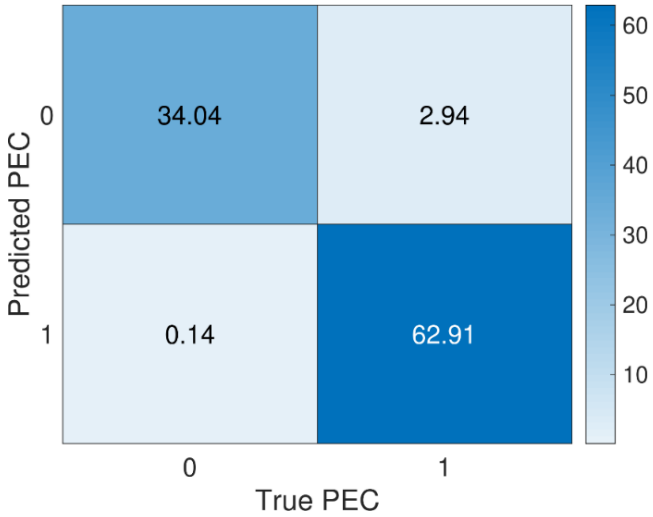

Figure 34 *Confusion matrix illustrating the classification performance of the FedAvg scheme (Server/Client optimizers are Adamax/Adamax) in the prediction of the engine condition* [37].

6.5.3 Use Case 3: Prediction of Upcoming Main Engine Consumption

In this use case, a real dataset with AIS data to define and solve a 9-variable regression problem waw used, whose solution provides forecasts of the upcoming PEPP over time, under

the principles of FL (see **Table 16** for the feature list). The dataset contained AIS data for 6 twin cargo ships, including multiple trajectories of the ships. Firstly, the number of the input features was reduced to six, based on objective criteria as: (i) heading variable was excluded since it is independent on the PEPP, (ii) DA and DF were also neglected because they are correlated with the Trim by definition (Trim = DF - DA), and (iii) from the two versions of the wind speed (DWS and CWS), we kept only DWS. As such, only six features (SOG, STW, DWS, WD, Trim) were considered as inputs and PEPP as the target output to solve the timeseries forecasting problem. Specifically, for any given time instance t, the goal of each local model was to predict the upcoming PEPP value (at $t+1$), based on the 5 previous values of each feature (i.e. $SOG_{t-4,t-3,\dots,t}$, $STW_{t-4,t-3,\dots,t}$, $DWS_{t-4,t-3,\dots,t}$, $WD_{t-4,t-3,\dots,t}$, $Trim_{t-4,t-3,\dots,t}$) (see **Figure 26**). Notably, PEPP values have been collected every 15 minutes, thus $PEPP_{t+1}$ refers to the upcoming engine's propulsion power in the next 15 minutes (after the current time instance t). Each FL participant trained an LSTM network, which was chosen because it generally can model linear and non-linear complex functions between inputs and expected outputs, as well as it can deal with timeseries with temporal dependencies. A 80%/20% splitting into training/testing sets was adopted for each local dataset [37].

6.5.3.1 Tuning of Learning Parameters

Before comparing the four FL schemes considered in this study, a learning parameter tuning procedure was followed, as in the previous subsections. By training each of the FL schemes with varying values of the learning rate α=[0.0001,0.001,0.01,0.1] and number of hidden layers $N_{layers} = [1,2,3,4]$, an optimal MAE of the predictions for α=0.001 and $N_{layers} = 1$ (128 neurons) was obtained, as shown in **Figure 26**. Note that, the simulations regarding the learning rate and hidden layers tuning are not depicted for purposes of saving figure space. Also, for this subsection, the MAE refers to the mean of squared differences between the predicted and the actual PEPP values, with the latter referring to the combined testing set of all training ships (concatenated testing sets of the training ship datasets). Specially for FedAvgM and FedProx, we ran similar simulation for tuning the momentum β = [0.1, 0.3, 0.5, 0.7, 0.9, 0.96] and the proximal strength μ = [0.1, 0.3, 0.5, 0.7, 0.9], respectively. FedAvgM presented the lowest MAE for β=0.9, whereas FedProx exhibited the lowest MAE for μ=0.7, implying that there was data heterogeneity amongst the local datasets (non-IID distributions of the features across local FL participants) [37].

Figure 35 shows the performance of each FL scheme in terms of the MAE for different server/client optimizers. The MAE was computed as the mean absolute difference between the predicted and actual PEPP values, which were taken by a combined testing set. This combined testing set was comprised of all the testing sets of the ships (each ship-specific dataset had been divided into 80%/20% training/testing sets). It is evident that FedSGD performed optimally for Adamax at the server side, FedAvg for Adamax/Adamax at the server/clients sides, FedAvgM for Adam at the client side and FedProx for Yogi/SGD at the sever/clients sides [37].

Considering the optimal optimizer per FL scheme, we also assessed the impact of the number of local epochs per round, as depicted in **Figure 36**. It is shown that, in this use case, the optimal local epochs per aggregation round was 10 for all schemes, with the FedProx outperforming the other methods noticing a MAE of 11.6 kW. Note that, the presented MAE

(11.6 kW) was considerably low with respect to the overall variance of the testing PEPP values ranging from 2000 to 13000 kW. In this sense, the MAE metrics are orders of magnitude lower than the total variance of the PEPP data, implying that the PEPP predictions are low-error. Notably, FedProx constantly outperform the FedAvg scheme, which implicitly means that the local data distribution of the features present some degree of heterogeneity across FL participants. Moreover, FedAvgM exhibited more accurate predictions than FedAvg regardless of the local epochs per round, but failed to exceed the FedProx in terms of the PEPP prediction performance [37].

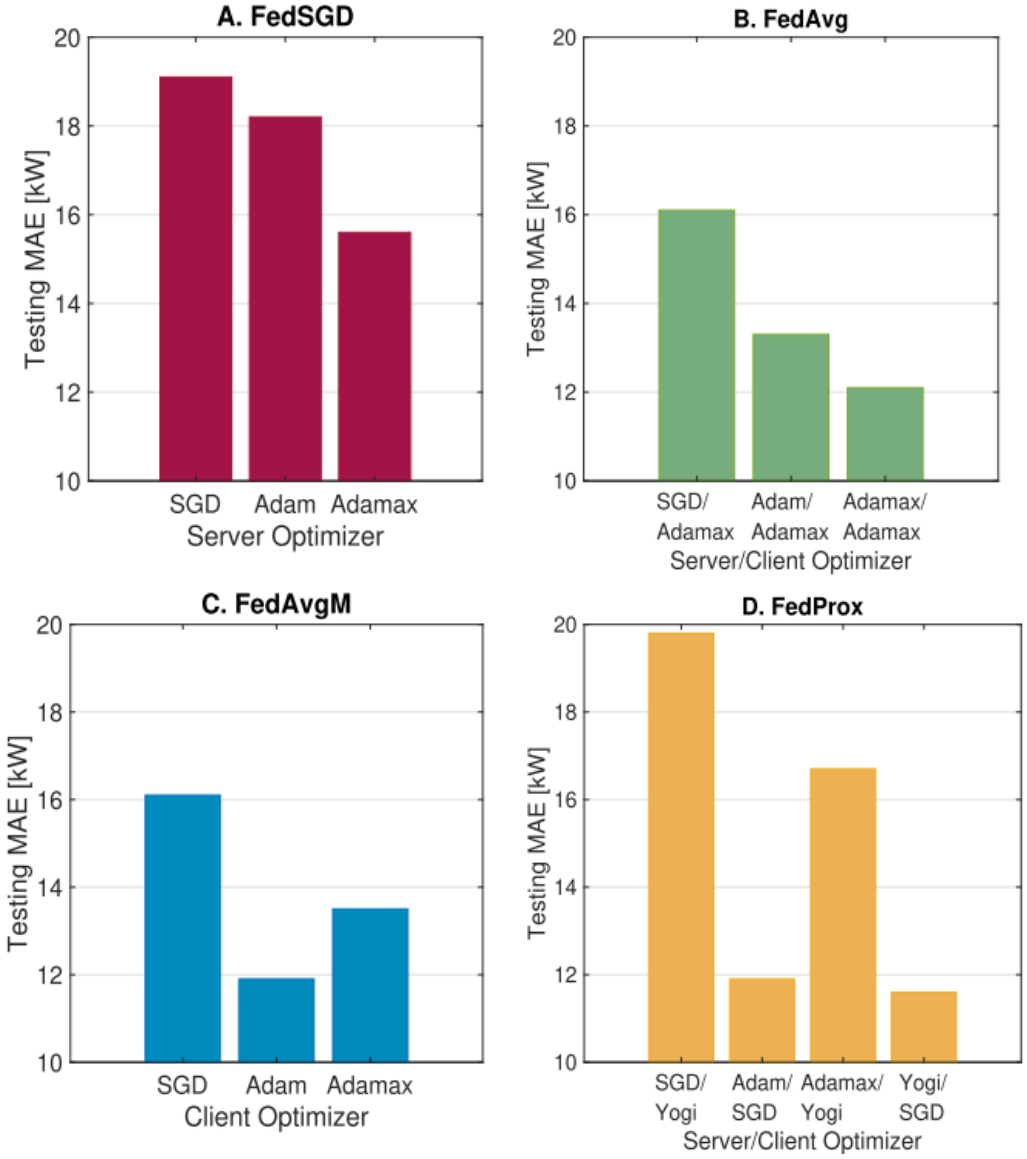

Figure 35 *Use Case 3 (Prediction of upcoming main engine consumption) predictive performance of the four FL schemes for different optimizer configuration at both server and client sites. Panels A-D correspond to FedSGD, FedAvg, FedAvgM and FedProx, respectively. The evaluation metric is the MAE loss between the actual and the predicted sample values drawn from the testing set* [37].

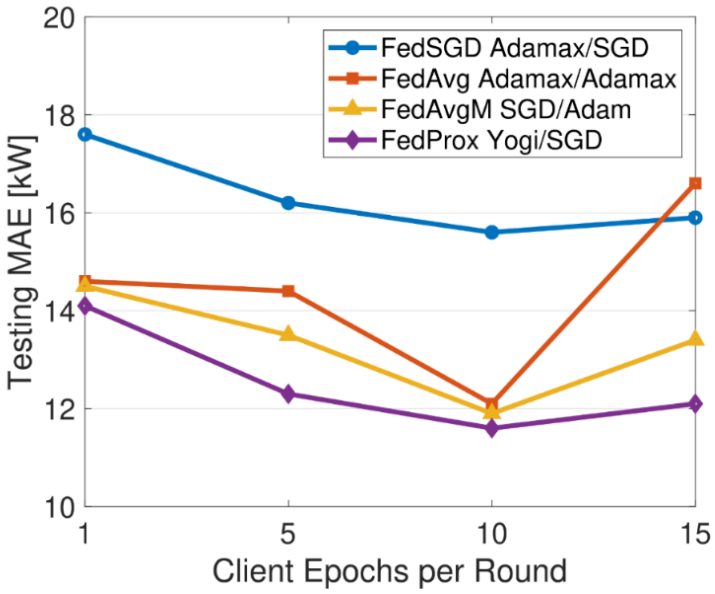

Figure 36 *The impact of the number of Epochs per Round on the FL performance in use Case 3. The Mean Absolute Error (MAE) loss of FedSGD, FedAvgM, FedAvg and FedProx schemes are depicted, respectively. All schemes are configured with their optimal optimizer, whereas MAE metrics are calculated on the testing samples* [37].

6.5.3.2 Performance Comparison

To quantitatively evaluate the FedProx performance in forecasting the PEPP values under unknown conditions, we used the data of the sixth dataset belonging to the sixth ship. This dataset had not been encountered in the FedProx model training phase, thus being able to index how well the model generalize its predictions and whether it can be reused by new ships (transfer learning) for purposes of predicting their upcoming PEPP without retraining. **Figure 37** shows the relationship between the actual and the FedProx model-predicted PEPP values of the sixth ship's dataset. Note that, the perfect relation would be the line $y = x$, which would mean that the model produces the perfectly correct predictions. Evidently, we observed that FedProx was able to accurately forecast the PEPP values of the unknown dataset, showing semi-linear relationship between predicted and ground truth PEPP values, even considering totally unknown ship dataset. Note that, if the FedProx is softly retrained using a few training samples of the sixth ship's dataset, the accuracy would be higher relative to the worst-case results that we considered in Figure 37 (no training with the sixth ship's data has been considered) [37].

Overall, the results confirm that FedProx may be the most suitable FL scheme in cases that local data distributions present heterogeneity across FL agents, while also ensuring data privacy preservation (only model parameters are communicated), low communication footprint (since it requires quite rare aggregation rounds) between the cloud and the local ship clients and reusability of the model by new ships without hard retraining. The latter is in principle crucial in the maritime applications when environmental footprint mitigation comes into play [37].

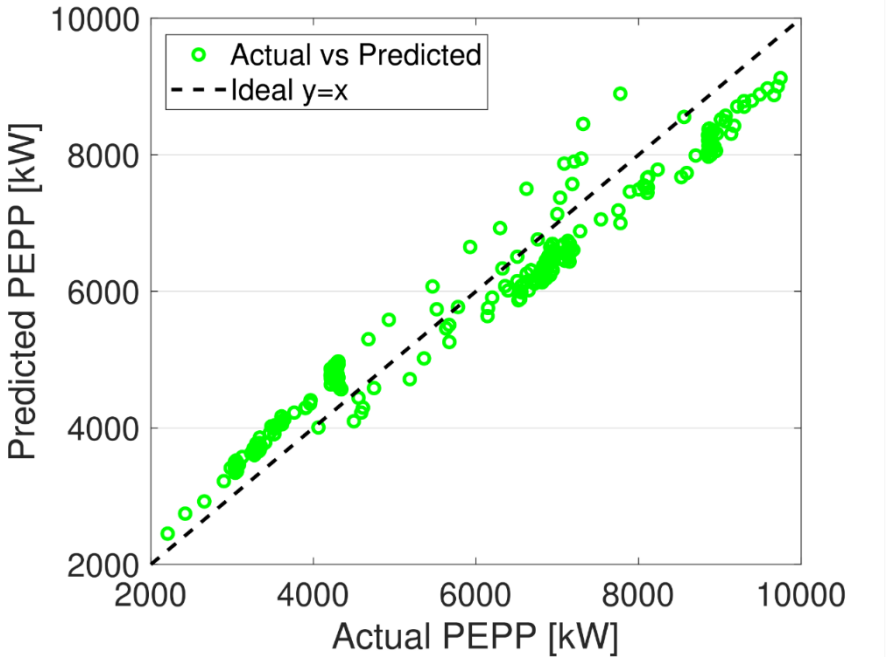

Figure 37 *Actual and predicted values [in kW] of the testing samples of the Primary Engine's Propulsion Power (PEPP) as scatter plots. Predictions refer to the FedProx algorithm. The dashed line represents the ideal 'actual vs predicted' matching (y = x)* [37].

6.6 Conclusions

Shipping 4.0 will revolutionize maritime operations by integrating advanced automation and digitization technologies. A major enabling technology in this field is ML-aided predictive maintenance, addressing inefficiencies, costs, and unexpected failures of conventional maintenance practices. This study focused on the use of FL in predictive maintenance use cases in the context of Shipping 4.0, relying on real datasets from the maritime industry. We evaluated and compared various FL algorithms across three maritime use cases were investigated, including regression for predicting naval propulsion gas turbine, classification for predicting ship engine condition, and time-series regression for predicting ship fuel consumption. The efficiency of the proposed federated learning-based predictive maintenance demonstrated its potential to enhance maintenance decision making, reduce shipping industry downtime, and improve operational efficiency for smart ships [37].

7. Summary of Findings and Future Research Directions

Zero-defect manufacturing is of the utmost importance for the future of the industrial sector. The research carried out result in the integration of zero-defect manufacturing strategy by applying ML-based techniques. The methods and tools used in this research are able to be installed in a wide range of industrial plants, regardless of the production sector. The main objectives are to improve the production process by minimizing defects, reducing costs and developing high-quality infrastructures with a low environmental footprint. Therefore, Chapter 1 highlights the importance of the ZDM strategy in the era of Industry 4.0. It also emphasizes the importance of ML in utilizing the ZDM strategy, while concept drift detection and federated learning are conducted targeting at ZDM implementation. Chapter 2 comprehensively presents ML and DL, while providing evaluation methods metrics and ML frameworks. Furthermore, the role of machine learning to I4 is extensively studied. Next, Chapter 3 provides related work in the context of Industry 4.0 and ZDM. A comprehensive literature about concept drift is provided, along with related works about zero waste, defect prediction and detection. Then, an extensive overview of supervised, unsupervised and deep learning solutions for predictive maintenance is provided. Finally, federated learning is thoroughly overviewed, while related works to predictive maintenance in Shipping 4.0 are also presented.

Minimizing defective products and improving production efficiency while keeping production costs low and reducing the environmental footprint are the goals of future industry and the building blocks for zero-defect manufacturing. To this end, an ML-based strategy for assigning orders to printing machines was proposed in Chapter 4. A machine-specific multi-feature regression model was trained using a real data set from an industrial offset printing environment. The model was used to accurately predict the defect ratio for new orders, targeting at proactively selecting the proper machine to execute a given order. In addition, classical ML, ensemble and deep learning methods were compared for comparison purposes. Numerical outcomes confirmed that the proposed machine selection scheme can efficiently provide order accuracy predictions. The artificial neural network (ANN) and ensemble methods outperformed the other regression algorithms. The accurate correlation of order features with the defective products results in a beneficial machine selection policy, which in turn contributes to the enhancement of the production efficiency, the minimization of defective products and the reduction of the company’s environmental footprint. Results can be used for further validating experiments as benchmark. While this work focusses on predicting the defect ration in order to select the proper machine to execute a given order, it can be extended by utilizing RUL prediction taking into account the degradation of defect ratio in specific orders for each machine. This approach may lead to a maintenance schedule targeting at minimization of wasted products due to machine malfunctions [53].

Monitoring of process parameters, continuous quality control and on-line predictive maintenance are central pillars of ZDM. These processes have the ability to generate huge amounts of data. Many issues develop as a result of massive quantities of data that are constantly collected during the system's online operation. A common problem in streaming data is the concept drift that refers either to the change of the target variable while the statistical properties of the data remain unchanged - known as real drift - or the virtual drift that occurs when the data distribution changes while the target variable remains stable. To that end, a comparison between different base classifiers for drift detection in streaming

environments, as well as an impact analysis of the performance across different base classifiers in terms of drift detection accuracy and a performance evaluation of the concept drift detector in the presence of imbalanced/noisy data is necessary. Considering that, Chapter 5 presents a small-scale comparison with regard to the impact of different classifiers in Drift Detection Method. Results reveal that the HT and KNN classifiers outperform the other baseline algorithms. Despite the NB baseline detector's higher accuracy compared to the VFDRC algorithm, it does not outperform it in drift detection with DDM. Additionally, the accuracy metric does not directly affect the drift detection mechanism, although it is closely associated with triggering drift detections. To achieve this goal, the DDM warning and drift level parameters can be adjusted appropriately. However, it is important to note that such modifications may increase the false detection rate and decrease the number of true detections. Additional research is required in the area of concept drift detection and handling, while more attention should be given in unsupervised learning-based DDM. A framework should be established for the evaluation of concept drift detection and additional applicable metrics should be introduced. An effective progress in the Industry 4.0 era requires considerable attention in concept drift detection combined with class imbalance, noisy data, recurring concepts and other defects. The presented framework may be easily extended and modified by incorporating additional base classifiers, alternative drift detection methods, different evaluations techniques and combination of defects. A future scope involves expanding this research by increasing the number of base classifiers and testing them with a plethora of artificial and real datasets of different sizes and from different domains of Industry 4.0. Another future research direction, is the utilization of an ML model to serve as a concept drift detection algorithm. Additionally, literature suggest that class imbalance and multi label data streams in CD scenarios need further investigation [29].

Despite the attention that ZDM attracts in the last few years, literature reveals a gap in the extension of ZDM strategies in new industrial sectors. Expanding ZDM strategies in new sectors is promising but challenging procedure, in which ML can play a crucial role with spectacular results. ML-aided predictive maintenance is a significant technology in this field, addressing inefficiencies, costs, and unexpected failures of conventional maintenance practices. Therefore, in Chapter 6, a study is presented focusing on the use of FL in predictive maintenance use cases in the context of Shipping 4.0, relying on real datasets from the maritime industry. The implementation of Shipping 4.0 will have a profound impact on maritime operations by incorporating cutting-edge automation and digitization technologies. In this study, various FL algorithms were evaluated and compared across three maritime use cases. The use cases include, regression technique for predicting naval propulsion gas turbine decay state, classification for predicting ship engine condition and time-series regression for predicting ship fuel consumption. The efficiency of the proposed federated learning-based predictive maintenance demonstrated its potential to enhance maintenance decision making, reduce shipping industry downtime, and improve operational efficiency for smart ships. It is expected that the findings will contribute to the advancement of predictive maintenance methodologies in Shipping 4.0, offering valuable insights for maritime stakeholders to embrace federated learning as a viable and privacy-preserving solution. Moreover, it will facilitate model sharing within the shipping industry and encourage collaboration among maritime stakeholders. This work can be extended in various directions, including: 1) Clustering of the federated learning agents before building the FL global model. This means that, an agent grouping (as a preliminary step) based on the features of the local agent datasets may improve the final FL-per-group performance, since agents with quasi-common

data distributions and high similarity can converge to a homogeneous high-accuracy model. 2) Evaluation of additional use cases that are envisioned in Shipping 4.0 era, exploiting real datasets concerning dense maritime areas and diverse types of maritime nodes (UAVs, buoys, underwater vehicles, port facilities) towards coordinating with the foreseen 6G communications. 3) Deployment of communication-efficient methods [230] with the aim of lowering the overhead required for model exchange rounds and reducing the energy consumption footprint of the FL schemes in the maritime sector (e.g. selection of nodes participating in the FL training, energy-efficient and bandwidth-aware methods for controlling the transmitting power of the maritime entities [37], [231], [232].

An extension of this research is the architectural design and implementation of ZDM system that incorporates ML ready techniques for process and machine oriented ZDM. That system will utilize FL, concept drift detection, process monitoring, predictive maintenance and fault detection/prediction. The main goal of this system will be to provide ready to use tools for fast alignment of a production process to ZDM strategy.

References

[1] Y. Lu, 'Industry 4.0: A survey on technologies, applications and open research issues', *J Ind Inf Integr*, vol. 6, pp. 1–10, 2017, doi: 10.1016/j.jii.2017.04.005.

[2] B. Chen, J. Wan, L. Shu, P. Li, M. Mukherjee, and B. Yin, 'Smart Factory of Industry 4.0: Key Technologies, Application Case, and Challenges', *IEEE Access*, vol. 6, pp. 6505–6519, 2018, doi: 10.1109/access.2017.2783682.

[3] P. K. Muhuri, A. K. Shukla, and A. Abraham, 'Industry 4.0: A bibliometric analysis and detailed overview', *Eng Appl Artif Intell*, vol. 78, pp. 218–235, 2019, doi: 10.1016/j.engappai.2018.11.007.

[4] B. Li, B. Hou, W. Yu, X. Lu, and C. Yang, 'Applications of artificial intelligence in intelligent manufacturing: a review', *Frontiers of Information Technology & Electronic Engineering*, vol. 18, no. 1, pp. 86–96, 2017, doi: 10.1631/fitee.1601885.

[5] L. Da Xu, E. L. Xu, and L. Li, 'Industry 4.0: state of the art and future trends', *Int J Prod Res*, vol. 56, no. 8, pp. 2941–2962, 2018, doi: 10.1080/00207543.2018.1444806.

[6] P. P. Groumpos, 'A Critical Historical and Scientific Overview of all Industrial Revolutions', *IFAC-PapersOnLine*, vol. 54, no. 13, pp. 464–471, 2021, doi: 10.1016/j.ifacol.2021.10.492.

[7] J. Qin, Y. Liu, and R. Grosvenor, 'A Categorical Framework of Manufacturing for Industry 4.0 and Beyond', *Procedia CIRP*, vol. 52, pp. 173–178, 2016, doi: https://doi.org/10.1016/j.procir.2016.08.005.

[8] R. Y. Zhong, X. Xu, E. Klotz, and S. T. Newman, 'Intelligent Manufacturing in the Context of Industry 4.0: A Review', *Engineering*, vol. 3, no. 5, pp. 616–630, 2017, doi: https://doi.org/10.1016/J.ENG.2017.05.015.

[9] E. Oztemel and S. Gursev, 'Literature review of Industry 4.0 and related technologies', *J Intell Manuf*, vol. 31, no. 1, pp. 127–182, 2020, doi: 10.1007/s10845-018-1433-8.

[10] L. Monostori *et al.*, 'Cyber-physical systems in manufacturing', *CIRP Annals*, vol. 65, no. 2, pp. 621–641, 2016, doi: https://doi.org/10.1016/j.cirp.2016.06.005.

[11] L. Da Xu, E. L. Xu, and L. Li, 'Industry 4.0: state of the art and future trends', *Int J Prod Res*, vol. 56, no. 8, pp. 2941–2962, 2018, doi: 10.1080/00207543.2018.1444806.

[12] L. Monostori *et al.*, 'Cyber-physical systems in manufacturing', *CIRP Annals*, vol. 65, no. 2, pp. 621–641, 2016, doi: 10.1016/j.cirp.2016.06.005.

[13] H. Xu, W. Yu, D. Griffith, and N. Golmie, 'A Survey on Industrial Internet of Things: A Cyber-Physical Systems Perspective', *IEEE Access*, vol. 6, p. 10.1109/access.2018.2884906, 2018, doi: 10.1109/access.2018.2884906.

[14] A. Aijaz and M. Sooriyabandara, 'The Tactile Internet for Industries: A Review', *Proceedings of the IEEE*, vol. 107, no. 2, pp. 414–435, 2019, doi: 10.1109/jproc.2018.2878265.

[15] A. J. C. Trappey, C. V Trappey, U. Hareesh Govindarajan, A. C. Chuang, and J. J. Sun, 'A review of essential standards and patent landscapes for the Internet of Things: A key enabler for Industry 4.0', *Advanced Engineering Informatics*, vol. 33, pp. 208–229, 2017, doi: 10.1016/j.aei.2016.11.007.

[16] T. Ruppert, S. Jaskó, T. Holczinger, and J. Abonyi, 'Enabling Technologies for Operator 4.0: A Survey', *Applied Sciences*, vol. 8, no. 9, p. 1650, 2018, doi: 10.3390/app8091650.

[17] A. Angelopoulos *et al.*, 'Tackling Faults in the Industry 4.0 Era—A Survey of Machine-Learning Solutions and Key Aspects', *Sensors*, vol. 20, no. 1, 2020, doi: 10.3390/s20010109.

[18] E. Oztemel and S. Gursev, 'Literature review of Industry 4.0 and related technologies', *J Intell Manuf*, vol. 31, no. 1, pp. 127–182, 2018, doi: 10.1007/s10845-018-1433-8.

[19] T. Stock and G. Seliger, 'Opportunities of Sustainable Manufacturing in Industry 4.0', *Procedia CIRP*, vol. 40, pp. 536–541, 2016, doi: 10.1016/j.procir.2016.01.129.

[20] T. Wuest, D. Weimer, C. Irgens, and K.-D. Thoben, 'Machine learning in manufacturing: advantages, challenges, and applications', *Production & Manufacturing Research*, vol. 4, no. 1, pp. 23–45, 2016, doi: 10.1080/21693277.2016.1192517.

[21] P.-A. D. Foivos Psarommatis Gökan May and D. Kiritsis, 'Zero defect manufacturing: state-of-the-art review, shortcomings and future directions in research', *Int J Prod Res*, vol. 58, no. 1, pp. 1–17, 2020, doi: 10.1080/00207543.2019.1605228.

[22] D. Powell, M. C. Magnanini, M. Colledani, and O. Myklebust, 'Advancing zero defect manufacturing: A state-of-the-art perspective and future research directions', *Comput Ind*, vol. 136, p. 103596, 2022, doi: https://doi.org/10.1016/j.compind.2021.103596.

[23] A. S. O. F. D. (MANPOWER I. A. N. D. L. W. DC, 'A Guide to Zero Defects. Quality and Reliability Assurance Handbook 4155.12-H', 1965, [Online]. Available: https://apps.dtic.mil/sti/citations/ADA950061

[24] B. Caiazzo, M. Di Nardo, T. Murino, A. Petrillo, G. Piccirillo, and S. Santini, 'Towards Zero Defect Manufacturing paradigm: A review of the state-of-the-art methods and open challenges', *Comput Ind*, vol. 134, p. 103548, 2022, doi: https://doi.org/10.1016/j.compind.2021.103548.

[25] M. H. ur Rehman, I. Yaqoob, K. Salah, M. Imran, P. P. Jayaraman, and C. Perera, 'The role of big data analytics in industrial Internet of Things', *Future Generation Computer Systems*, vol. 99, pp. 247–259, 2019, doi: 10.1016/j.future.2019.04.020.

[26] Q. Qi and F. Tao, 'Digital Twin and Big Data Towards Smart Manufacturing and Industry 4.0: 360 Degree Comparison', *IEEE Access*, vol. 6, pp. 3585–3593, 2018, doi: 10.1109/access.2018.2793265.

[27] F. Tao, Q. Qi, A. Liu, and A. Kusiak, 'Data-driven smart manufacturing', *J Manuf Syst*, vol. 48, pp. 157–169, 2018, doi: 10.1016/j.jmsy.2018.01.006.

[28] P. Trakadas *et al.*, 'An Artificial Intelligence-Based Collaboration Approach in Industrial IoT Manufacturing: Key Concepts, Architectural Extensions and Potential Applications', *Sensors*, vol. 20, no. 19, 2020, doi: 10.3390/s20195480.

[29] A. E. and K. N. C. and S. S. T. and S. L. and V. S. and T. P. A. A. and Giannopoulos, 'Impact of Classifiers to Drift Detection Method: A Comparison', in *Proceedings of the 22nd Engineering Applications of Neural Networks Conference*, J. and J. C. and P. E. I. L. and Macintyre, Ed., Springer International Publishing, 2021, pp. 399–410.

[30] N. and N. G. and S. T. and T. P. Giannopoulos Anastasios and Nomikos, 'Maritime Federated Learning for Decentralized On-Ship Intelligence', in *Artificial Intelligence Applications and Innovations*, L. and M. J. and D. M. Maglogiannis Ilias and Iliadis, Ed., Cham: Springer Nature Switzerland, 2023, pp. 195–206.

[31] K. Skianis, A. Giannopoulos, P. Gkonis, and P. Trakadas, 'Data Aging Matters: Federated Learning-Based Consumption Prediction in Smart Homes via Age-Based Model Weighting', *Electronics (Basel)*, vol. 12, no. 14, 2023, doi: 10.3390/electronics12143054.

[32] P. Boobalan *et al.*, 'Fusion of Federated Learning and Industrial Internet of Things: A survey', *Computer Networks*, vol. 212, p. 109048, 2022, doi: https://doi.org/10.1016/j.comnet.2022.109048.

[33] H. T. Truong *et al.*, 'Light-weight federated learning-based anomaly detection for time-series data in industrial control systems', *Comput Ind*, vol. 140, p. 103692, 2022, doi: https://doi.org/10.1016/j.compind.2022.103692.

[34] S. Becker, K. Styp-Rekowski, O. V. L. Stoll, and O. Kao, 'Federated Learning for Autoencoder-based Condition Monitoring in the Industrial Internet of Things', in *2022 IEEE International Conference on Big Data (Big Data)*, 2022, pp. 5424–5433. doi: 10.1109/BigData55660.2022.10020836.

[35] S. Bharti and A. Mcgibney, 'Privacy-Aware Resource Sharing in Cross-Device Federated Model Training for Collaborative Predictive Maintenance', *IEEE Access*, vol. 9, pp. 120367–120379, 2021, doi: 10.1109/ACCESS.2021.3108839.

[36] T. Ranathunga, A. McGibney, S. Rea, and S. Bharti, 'Blockchain-Based Decentralized Model Aggregation for Cross-Silo Federated Learning in Industry 4.0', *IEEE Internet Things J*, vol. 10, no. 5, pp. 4449–4461, 2023, doi: 10.1109/JIOT.2022.3218704.

[37] A. Angelopoulos, A. Giannopoulos, N. Nomikos, A. Kalafatelis, A. Hatziefremidis, and P. Trakadas, 'Federated Learning-Aided Prognostics in the Shipping 4.0: Principles, Workflow, and Use Cases', *IEEE Access*, vol. 12, pp. 6437–6454, 2024, doi: 10.1109/ACCESS.2024.3350777.

[38] Z. Ge, Z. Song, S. X. Ding, and B. Huang, 'Data Mining and Analytics in the Process Industry: The Role of Machine Learning', *IEEE Access*, vol. 5, pp. 20590–20616, 2017, doi: 10.1109/access.2017.2756872.

[39] D.-H. Kim *et al.*, 'Smart Machining Process Using Machine Learning: A Review and Perspective on Machining Industry', *International Journal of Precision Engineering and Manufacturing-Green Technology*, vol. 5, no. 4, pp. 555–568, 2018, doi: 10.1007/s40684-018-0057-y.

[40] L. Da Xu and L. Duan, 'Big data for cyber physical systems in industry 4.0: a survey', *Enterp Inf Syst*, vol. 13, no. 2, pp. 148–169, 2018, doi: 10.1080/17517575.2018.1442934.

[41] S. and van der S. P. and L. A. Sonntag Daniel and Zillner, 'Overview of the CPS for Smart Factories Project: Deep Learning, Knowledge Acquisition, Anomaly Detection and Intelligent User Interfaces', in *Industrial Internet of Things: Cybermanufacturing Systems*, C. and S. H. and R. D. B. Jeschke Sabina and Brecher, Ed., Cham: Springer International Publishing, 2017, pp. 487–504. doi: 10.1007/978-3-319-42559-7_19.

[42] J. Wang, Y. Ma, L. Zhang, R. X. Gao, and D. Wu, 'Deep learning for smart manufacturing: Methods and applications', *J Manuf Syst*, vol. 48, pp. 144–156, 2018, doi: 10.1016/j.jmsy.2018.01.003.

[43] M. Reis and G. Gins, 'Industrial Process Monitoring in the Big Data/Industry 4.0 Era: from Detection, to Diagnosis, to Prognosis', *Processes*, vol. 5, no. 4, p. 35, 2017, doi: 10.3390/pr5030035.

[44] D. Ramotsoela, A. Abu-Mahfouz, and G. Hancke, 'A Survey of Anomaly Detection in Industrial Wireless Sensor Networks with Critical Water System Infrastructure as a Case Study', *Sensors (Basel)*, vol. 18, no. 8, p. 2491, Aug. 2018, doi: 10.3390/s18082491.

[45] S. B. Kotsiantis, I. D. Zaharakis, and P. E. Pintelas, 'Machine learning: a review of classification and combining techniques', *Artif Intell Rev*, vol. 26, no. 3, pp. 159–190, Nov. 2006, doi: 10.1007/s10462-007-9052-3.

[46] A. Shrestha and A. Mahmood, 'Review of Deep Learning Algorithms and Architectures', *IEEE Access*, vol. 7, pp. 53040–53065, 2019, doi: 10.1109/ACCESS.2019.2912200.

[47] L. Alzubaidi *et al.*, 'Review of deep learning: concepts, CNN architectures, challenges, applications, future directions', *J Big Data*, vol. 8, no. 1, p. 53, Mar. 2021, doi: 10.1186/s40537-021-00444-8.

[48] S. Murat H., 'A brief review of feed-forward neural networks', *Communications Faculty Of Science University of Ankara*, vol. 50, no. 1, pp. 11–17, 2006, doi: 10.1501/commua1-2_0000000026.

[49] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, 'A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects', *IEEE Trans Neural Netw Learn Syst*, vol. 33, no. 12, pp. 6999–7019, Dec. 2022, doi: 10.1109/TNNLS.2021.3084827.

[50] A. Sherstinsky, 'Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network', *Physica D*, vol. 404, p. 132306, Mar. 2020, doi: 10.1016/j.physd.2019.132306.

[51] S. Hochreiter and J. Schmidhuber, 'Long Short-Term Memory', *Neural Comput*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[52] O. F.Y, A. J.E.T, A. O, H. J. O, O. O, and A. J, 'Supervised Machine Learning Algorithms: Classification and Comparison', *International Journal of Computer Trends and Technology*, vol. 48, no. 3, pp. 128–138, Jun. 2017, doi: 10.14445/22312803/IJCTT-V48P126.

[53] A. and S. S. and K. N. and T. C. and V. S. and T. P. A. A. and Giannopoulos, 'Allocating Orders to Printing Machines for Defect Minimization: A Comparative Machine Learning Approach', in *Artificial Intelligence Applications and Innovations*, L. and M. J. and C. P. M. I. and Iliadis, Ed., Springer International Publishing, 2022, pp. 79–88.

[54] H. Zhang, *The Optimality of Naive Bayes*, vol. 2. 2004.

[55] C. Cortes and V. Vapnik, 'Support-vector networks', *Mach Learn*, vol. 20, no. 3, pp. 273–297, Sep. 1995, doi: 10.1007/BF00994018.

[56] R. Polikar, 'Ensemble Learning', in *Ensemble Machine Learning*, New York, NY: Springer New York, 2012, pp. 1–34. doi: 10.1007/978-1-4419-9326-7_1.

[57] L. I. Kuncheva, *Combining Pattern Classifiers*. Wiley, 2004. doi: 10.1002/0471660264.

[58] N. Littlestone and M. K. Warmuth, 'The Weighted Majority Algorithm', *Inf Comput*, vol. 108, no. 2, pp. 212–261, Feb. 1994, doi: 10.1006/inco.1994.1009.

[59] L. Breiman, 'Arcing classifier (with discussion and a rejoinder by the author)', *The Annals of Statistics*, vol. 26, no. 3, Jun. 1998, doi: 10.1214/aos/1024691079.

[60] R. E. Schapire, 'The strength of weak learnability', *Mach Learn*, vol. 5, no. 2, pp. 197–227, Jun. 1990, doi: 10.1007/BF00116037.

[61] D. H. Wolpert, 'Stacked generalization', *Neural Networks*, vol. 5, no. 2, pp. 241–259, Jan. 1992, doi: 10.1016/S0893-6080(05)80023-1.

[62] M. Zinkevich, M. Weimer, L. Li, and A. Smola, 'Parallelized stochastic gradient descent', *Adv Neural Inf Process Syst*, vol. 23, 2010.

[63] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, 'Communication-Efficient Learning of Deep Networks from Decentralized Data', in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, A. Singh and J. Zhu, Eds., in Proceedings of Machine Learning Research, vol. 54. PMLR, Jan. 2017, pp. 1273–1282. [Online]. Available: https://proceedings.mlr.press/v54/mcmahan17a.html

[64] T.-M. H. Hsu, H. Qi, and M. Brown, 'Measuring the effects of non-identical data distribution for federated visual classification', *arXiv preprint arXiv:1909.06335*, 2019.

[65] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, 'Federated Optimization in Heterogeneous Networks', Dec. 2018.

[66] L. Lo Vercio *et al.*, 'Supervised machine learning tools: a tutorial for clinicians', *J Neural Eng*, vol. 17, no. 6, p. 062001, Dec. 2020, doi: 10.1088/1741-2552/abbff2.

[67] Y. Roh, G. Heo, and S. E. Whang, 'A Survey on Data Collection for Machine Learning: A Big Data - AI Integration Perspective', *IEEE Trans Knowl Data Eng*, vol. 33, no. 4, pp. 1328–1347, Apr. 2021, doi: 10.1109/TKDE.2019.2946162.

[68] L. Xu and K. Veeramachaneni, 'Synthesizing Tabular Data using Generative Adversarial Networks'. Feb. 2018.

[69] S. García, J. Luengo, and F. Herrera, 'Dealing with Noisy Data', 2015, pp. 107–145. doi: 10.1007/978-3-319-10247-4_5.

[70] H. Wang, M. J. Bah, and M. Hammad, 'Progress in Outlier Detection Techniques: A Survey', *IEEE Access*, vol. 7, pp. 107964–108000, 2019, doi: 10.1109/ACCESS.2019.2932769.

[71] K. L. Sainani, 'Dealing With Missing Data', *PM&R*, vol. 7, no. 9, pp. 990–994, Sep. 2015, doi: 10.1016/j.pmrj.2015.07.011.

[72] J. Leppink, 'Dealing with Missing Data', 2019, pp. 69–76. doi: 10.1007/978-3-030-21241-4_4.

[73] K. Lakshminarayan, S. A. Harp, and T. Samad, 'Imputation of Missing Data in Industrial Databases', *Applied Intelligence*, vol. 11, no. 3, pp. 259–275, 1999, doi: 10.1023/A:1008334909089.

[74] S. Kotsiantis, D. Kanellopoulos, and P. Pintelas, 'Data Preprocessing for Supervised Learning', *Int J Comp Sci*, vol. 1, pp. 111–117, Feb. 2006.

[75] Haibo He and E. A. Garcia, 'Learning from Imbalanced Data', *IEEE Trans Knowl Data Eng*, vol. 21, no. 9, pp. 1263–1284, Sep. 2009, doi: 10.1109/TKDE.2008.239.

[76] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, 'SMOTE: Synthetic Minority Over-sampling Technique', *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, Jun. 2002, doi: 10.1613/jair.953.

[77] G. Chandrashekar and F. Sahin, 'A survey on feature selection methods', *Computers & Electrical Engineering*, vol. 40, no. 1, pp. 16–28, Jan. 2014, doi: 10.1016/j.compeleceng.2013.11.024.

[78] S.-A. N. Alexandropoulos, S. B. Kotsiantis, and M. N. Vrahatis, 'Data preprocessing in predictive data mining', *Knowl Eng Rev*, vol. 34, p. e1, Jan. 2019, doi: 10.1017/S026988891800036X.

[79] S. García, J. Luengo, and F. Herrera, 'Tutorial on practical tips of the most influential data preprocessing algorithms in data mining', *Knowl Based Syst*, vol. 98, pp. 1–29, Apr. 2016, doi: 10.1016/j.knosys.2015.12.006.

[80] M. Moradi, T. Syeda-Mahmood, and S. Hor, 'Tree-Based Transforms for Privileged Learning', 2016, pp. 188–195. doi: 10.1007/978-3-319-47157-0_23.

[81] Y.-J. Hu and D. Kibler, 'Generation of Attributes for Learning Algorithms', Feb. 1996, pp. 806–811.

[82] S. Markovitch and D. Rosenstein, 'Feature Generation Using General Constructor Functions', *Mach Learn*, vol. 49, no. 1, pp. 59–98, 2002, doi: 10.1023/A:1014046307775.

[83] G. Varoquaux and O. Colliot, 'Evaluating Machine Learning Models and Their Diagnostic Value', 2023, pp. 601–630. doi: 10.1007/978-1-0716-3195-9_20.

[84] S. Raschka, 'Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning', Nov. 2018.

[85] F. Pedregosa, 'Scikit-learn: machine learning in python', *JMLR*, vol. 12, 2011.

[86] M. Abadi *et al.*, 'Tensorflow: Large-scale machine learning on heterogeneous distributed systems', *arXiv preprint arXiv:1603.04467*, 2016.

[87] J. Montiel, J. Read, A. Bifet, and T. Abdessalem, 'Scikit-multiflow: a multi-output streaming framework', *J. Mach. Learn. Res.*, vol. 19, 2018.

[88] J. Céspedes-Sisniega and Á. López-García, 'Frouros: A Python library for drift detection in machine learning systems', Aug. 2022.

[89] D. J. Beutel *et al.*, 'Flower: A Friendly Federated Learning Research Framework', Jul. 2020.

[90] P. Foley *et al.*, 'OpenFL: the open federated learning library', *Phys Med Biol*, vol. 67, no. 21, p. 214001, Nov. 2022, doi: 10.1088/1361-6560/ac97d9.

[91] D. Mazzei and R. Ramjattan, 'Machine Learning for Industry 4.0: A Systematic Review Using Deep Learning-Based Topic Modelling', *Sensors*, vol. 22, no. 22, p. 8641, Nov. 2022, doi: 10.3390/s22228641.

[92] Y. Riahi, T. Saikouk, A. Gunasekaran, and I. Badraoui, 'Artificial intelligence applications in supply chain: A descriptive bibliometric analysis and future research directions', *Expert Syst Appl*, vol. 173, p. 114702, Jul. 2021, doi: 10.1016/j.eswa.2021.114702.

[93] J. P. Usuga Cadavid, S. Lamouri, B. Grabot, R. Pellerin, and A. Fortin, 'Machine learning applied in production planning and control: a state-of-the-art in the era of industry 4.0', *J Intell Manuf*, vol. 31, no. 6, pp. 1531–1558, Aug. 2020, doi: 10.1007/s10845-019-01531-7.

[94] M. Javaid, A. Haleem, and R. P. Singh, 'A study on ChatGPT for Industry 4.0: Background, potentials, challenges, and eventualities', *Journal of Economy and Technology*, vol. 1, pp. 127–143, Nov. 2023, doi: 10.1016/j.ject.2023.08.001.

[95] J. Cheng, W. Chen, F. Tao, and C.-L. Lin, 'Industrial IoT in 5G environment towards smart manufacturing', *J Ind Inf Integr*, vol. 10, pp. 10–19, 2018, doi: 10.1016/j.jii.2018.04.001.

[96] M. Lezzi, M. Lazoi, and A. Corallo, 'Cybersecurity for Industry 4.0 in the current literature: A reference framework', *Comput Ind*, vol. 103, pp. 97–110, Dec. 2018, doi: 10.1016/j.compind.2018.09.004.

[97] V. Azamfirei, F. Psarommatis, and Y. Lagrosen, 'Application of automation for in-line quality inspection, a zero-defect manufacturing approach', *J Manuf Syst*, vol. 67, pp. 1–22, Apr. 2023, doi: 10.1016/j.jmsy.2022.12.010.

[98] T. Zonta, C. A. da Costa, R. da Rosa Righi, M. J. de Lima, E. S. da Trindade, and G. P. Li, 'Predictive maintenance in the Industry 4.0: A systematic literature review', *Comput Ind Eng*, vol. 150, p. 106889, Dec. 2020, doi: 10.1016/j.cie.2020.106889.

[99] S. Sahoo and C.-Y. Lo, 'Smart manufacturing powered by recent technological advancements: A review', *J Manuf Syst*, vol. 64, pp. 236–250, Jul. 2022, doi: 10.1016/j.jmsy.2022.06.008.

[100] K. S. Aggour *et al.*, 'Artificial intelligence/machine learning in manufacturing and inspection: A GE perspective', *MRS Bull*, vol. 44, no. 7, pp. 545–558, 2019, doi: 10.1557/mrs.2019.157.

[101] Md. S. Rahman, T. Ghosh, N. F. Aurna, M. S. Kaiser, M. Anannya, and A. S. M. S. Hosen, 'Machine learning and internet of things in industry 4.0: A review', *Measurement: Sensors*, vol. 28, p. 100822, Aug. 2023, doi: 10.1016/j.measen.2023.100822.

[102] U. Murugesan, P. Subramanian, S. Srivastava, and A. Dwivedi, ‘A study of Artificial Intelligence impacts on Human Resource Digitalization in Industry 4.0’, *Decision Analytics Journal*, vol. 7, p. 100249, Jun. 2023, doi: 10.1016/j.dajour.2023.100249.

[103] A. Q. Md, K. Jha, S. Haneef, A. K. Sivaraman, and K. F. Tee, ‘A Review on Data-Driven Quality Prediction in the Production Process with Machine Learning for Industry 4.0’, *Processes*, vol. 10, no. 10, p. 1966, Sep. 2022, doi: 10.3390/pr10101966.

[104] D. Weichert, P. Link, A. Stoll, S. Rüping, S. Ihlenfeldt, and S. Wrobel, ‘A review of machine learning for the optimization of production processes’, *The International Journal of Advanced Manufacturing Technology*, vol. 104, no. 5–8, pp. 1889–1902, 2019, doi: 10.1007/s00170-019-03988-5.

[105] X. Xu and Q. Hua, ‘Industrial Big Data Analysis in Smart Factory: Current Status and Research Strategies’, *IEEE Access*, vol. 5, pp. 17543–17551, 2017, doi: 10.1109/access.2017.2741105.

[106] Y. Cheng, K. Chen, H. Sun, Y. Zhang, and F. Tao, ‘Data and knowledge mining with big data towards smart production’, *J Ind Inf Integr*, vol. 9, pp. 1–13, 2018, doi: 10.1016/j.jii.2017.08.001.

[107] Z. Huang, Y. Shen, J. Li, M. Fey, and C. Brecher, ‘A Survey on AI-Driven Digital Twins in Industry 4.0: Smart Manufacturing and Advanced Robotics’, *Sensors*, vol. 21, no. 19, p. 6340, Sep. 2021, doi: 10.3390/s21196340.

[108] A. Dogan and D. Birant, ‘Machine learning and data mining in manufacturing’, *Expert Syst Appl*, vol. 166, p. 114060, Mar. 2021, doi: 10.1016/j.eswa.2020.114060.

[109] F. Psarommatis and G. Bravos, ‘A holistic approach for achieving Sustainable manufacturing using Zero Defect Manufacturing: a conceptual Framework’, *Procedia CIRP*, vol. 107, pp. 107–112, 2022, doi: https://doi.org/10.1016/j.procir.2022.04.018.

[110] G. Fragapane, R. Eleftheriadis, D. Powell, and J. Antony, ‘A global survey on the current state of practice in Zero Defect Manufacturing and its impact on production performance’, *Comput Ind*, vol. 148, p. 103879, Jun. 2023, doi: https://doi.org/10.1016/j.compind.2023.103879.

[111] F. Psarommatis and G. May, ‘A practical guide for implementing Zero Defect Manufacturing in new or existing manufacturing systems’, *Procedia Comput Sci*, vol. 217, pp. 82–90, 2023, doi: 10.1016/j.procs.2022.12.204.

[112] G. May and D. Kiritsis, ‘Zero Defect Manufacturing Strategies and Platform for Smart Factories of Industry 4.0’, 2019, pp. 142–152. doi: 10.1007/978-3-030-18180-2_11.

[113] F. Psarommatis, ‘A generic methodology and a digital twin for zero defect manufacturing (ZDM) performance mapping towards design for ZDM’, *J Manuf Syst*, vol. 59, pp. 507–521, Apr. 2021, doi: 10.1016/j.jmsy.2021.03.021.

[114] J. C. Serrano-Ruiz, J. Mula, and R. Poler, ‘Digital Twin Enabling Intelligent Scheduling in ZDM Environments: an Overview’, 2023, pp. 173–182. doi: 10.1007/978-3-031-29382-5_18.

[115] L. Ghedini, A. Polenghi, and M. Macchi, ‘An ontology to integrate process-based approach in ZDM strategies in a Digital Twin framework’, *IFAC-PapersOnLine*, vol. 56, no. 2, pp. 6370–6375, 2023, doi: 10.1016/j.ifacol.2023.10.825.

[116] P. K. Wan and T. L. Leirmo, ‘Human-centric zero-defect manufacturing: State-of-the-art review, perspectives, and challenges’, *Comput Ind*, vol. 144, p. 103792, Jan. 2023, doi: 10.1016/j.compind.2022.103792.

[117] F. Psarommatis, G. May, and V. Azamfirei, ‘The Role of Human Factors in Zero Defect Manufacturing: A Study of Training and Workplace Culture’, 2023, pp. 587–601. doi: 10.1007/978-3-031-43662-8_42.

[118] F. Leotta, J. G. Mathew, M. Mecella, and F. Monti, ‘Supporting Zero Defect Manufacturing Through Cloud Computing and Data Analytics: the Case Study of Electrospindle 4.0’, 2022, pp. 119–125. doi: 10.1007/978-3-031-07478-3_10.

[119] C. Caccamo, R. Eleftheriadis, M. C. Magnanini, D. Powell, and Odd Myklebust, ‘A Hybrid Architecture for the Deployment of a Data Quality Management (DQM) System for Zero-Defect Manufacturing in Industry 4.0’, 2021, pp. 71–77. doi: 10.1007/978-3-030-85906-0_8.

[120] L. Jie, A. Liu, F. Dong, G. Feng, J. Gama, and G. Zhang, ‘Learning under concept drift: a review’, *IEEE Trans. Knowl. Data Eng.*, vol. 31, 2019, doi: 10.1109/TKDE.2018.2876857.

[121] S. Wares, J. Isaacs, and E. Elyan, ‘Data stream mining: methods and challenges for handling concept drift’, *SN Appl. Sci.*, vol. 1, 2019, doi: 10.1007/s42452-019-1433-0.

[122] A. Iwashita and J. Papa, ‘An overview on concept drift learning’, *IEEE Access*, vol. 7, 2019, doi: 10.1109/ACCESS.2018.2886026.

[123] S. Agrahari and A. K. Singh, ‘Concept Drift Detection in Data Stream Mining : A literature review’, *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 10, pp. 9523–9540, Nov. 2022, doi: 10.1016/j.jksuci.2021.11.006.

[124] A. L. Suárez-Cetrulo, D. Quintana, and A. Cervantes, ‘A survey on machine learning for recurring concept drifting data streams’, *Expert Syst Appl*, vol. 213, p. 118934, Mar. 2023, doi: 10.1016/j.eswa.2022.118934.

[125] O. A. Mahdi, N. Ali, E. Pardede, A. Alazab, T. Al-Quraishi, and B. Das, ‘Roadmap of Concept Drift Adaptation in Data Stream Mining, Years Later’, *IEEE Access*, vol. 12, pp. 21129–21146, 2024, doi: 10.1109/ACCESS.2024.3358817.

[126] P. M. Gonçalves, S. G. Carvalho Santos, R. S. Barros, and D. C. Vieira, ‘A comparative study on concept drift detectors’, *Expert Syst. Appl.*, vol. 41, 2014, doi: 10.1016/j.eswa.2014.07.019.

[127] J. Liao, J. Zhang, and W. W. Y. Ng, ‘Effects of different base classifiers to Learn++ family algorithms for concept drifting and imbalanced pattern classification problems’, in *2016 International Conference on Machine Learning and Cybernetics (ICMLC)*, 2016, pp. 99–104. doi: 10.1109/ICMLC.2016.7860884.

[128] R. Barros, G. Silas, and C. Santos, ‘A large-scale comparison of concept drift detectors’, *Inf. Sci.*, vol. 451–452, 2018, doi: 10.1016/j.ins.2018.04.014.

[129] R. Maior, S. Barros, G. Carvalho, and Santos, 'An overview and comprehensive comparison of ensembles for concept drift', *Inf. Fus.*, vol. 52, 2019, doi: 10.1016/j.inffus.2019.03.006.

[130] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, 'Learning with drift detection', in *Advances in Artificial Intelligence–SBIA 2004: 17th Brazilian Symposium on Artificial Intelligence, Sao Luis, Maranhao, Brazil, September 29-Ocotber 1, 2004. Proceedings 17*, 2004, pp. 286–295.

[131] M. Baena-García, J. Campo-Ávila, R. Fidalgo-Merino, A. Bifet, R. Gavald, and R. Morales-Bueno, 'Early Drift Detection Method', Jan. 2006.

[132] I. Frias-Blanco, J. del Campo-Avila, G. Ramos-Jimenez, R. Morales-Bueno, A. Ortiz-Diaz, and Y. Caballero-Mota, 'Online and Non-Parametric Drift Detection Methods Based on Hoeffding's Bounds', *IEEE Trans Knowl Data Eng*, vol. 27, no. 3, pp. 810–823, Mar. 2015, doi: 10.1109/TKDE.2014.2345382.

[133] M. M. W. Yan, 'Accurate detecting concept drift in evolving data streams', *ICT Express*, vol. 6, no. 4, pp. 332–338, Dec. 2020, doi: 10.1016/j.icte.2020.05.011.

[134] Heng Wang and Z. Abraham, 'Concept drift detection for streaming data', in *2015 International Joint Conference on Neural Networks (IJCNN)*, IEEE, Jul. 2015, pp. 1–9. doi: 10.1109/IJCNN.2015.7280398.

[135] S. Yu, Z. Abraham, H. Wang, M. Shah, Y. Wei, and J. C. Príncipe, 'Concept drift detection and adaptation with hierarchical hypothesis testing', *J Franklin Inst*, vol. 356, no. 5, pp. 3187–3215, Mar. 2019, doi: 10.1016/j.jfranklin.2019.01.043.

[136] G. J. Ross, N. M. Adams, D. K. Tasoulis, and D. J. Hand, 'Exponentially weighted moving average charts for detecting concept drift', *Pattern Recognit Lett*, vol. 33, no. 2, pp. 191–198, Jan. 2012, doi: 10.1016/j.patrec.2011.08.019.

[137] K. Nishida and K. Yamauchi, 'Detecting Concept Drift Using Statistical Testing', in *Discovery Science*, Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 264–269. doi: 10.1007/978-3-540-75488-6_27.

[138] R. S. M. Barros, D. R. L. Cabral, P. M. Gonçalves, and S. G. T. C. Santos, 'RDDM: Reactive drift detection method', *Expert Syst Appl*, vol. 90, pp. 344–355, Dec. 2017, doi: 10.1016/j.eswa.2017.08.023.

[139] A. Bifet and R. Gavaldà, 'Learning from Time-Changing Data with Adaptive Windowing', in *Proceedings of the 2007 SIAM International Conference on Data Mining*, Philadelphia, PA: Society for Industrial and Applied Mathematics, Apr. 2007, pp. 443–448. doi: 10.1137/1.9781611972771.42.

[140] C. Raab, M. Heusinger, and F.-M. Schleif, 'Reactive Soft Prototype Computing for Concept Drift Streams', *Neurocomputing*, vol. 416, pp. 340–351, Nov. 2020, doi: 10.1016/j.neucom.2019.11.111.

[141] Z. Chen, M. Han, H. Wu, M. Li, and X. Zhang, 'A multi-level weighted concept drift detection method', *J Supercomput*, vol. 79, no. 5, pp. 5154–5180, Mar. 2023, doi: 10.1007/s11227-022-04864-y.

[142] P. Wang, N. Jin, and G. Fehringer, 'Concept drift detection with False Positive rate for multi-label classification in IoT data stream', in *2020 International Conference on UK-China Emerging Technologies (UCET)*, IEEE, Aug. 2020, pp. 1–4. doi: 10.1109/UCET51115.2020.9205421.

[143] L. Baier, T. Schlör, J. Schöffer, and N. Kühl, 'Detecting Concept Drift With Neural Network Model Uncertainty', Jul. 2021, [Online]. Available: http://arxiv.org/abs/2107.01873

[144] V. Cerqueira, H. M. Gomes, A. Bifet, and L. Torgo, 'STUDD: a student–teacher method for unsupervised concept drift detection', *Mach Learn*, vol. 112, no. 11, pp. 4351–4378, Nov. 2023, doi: 10.1007/s10994-022-06188-7.

[145] K. Papageorgiou *et al.*, 'A systematic review on machine learning methods for root cause analysis towards zero-defect manufacturing', *Frontiers in Manufacturing Technology*, vol. 2, Oct. 2022, doi: 10.3389/fmtec.2022.972712.

[146] F. Eger *et al.*, 'Zero Defect Manufacturing Strategies for Reduction of Scrap and Inspection Effort in Multi-stage Production Systems', *Procedia CIRP*, vol. 67, pp. 368–373, 2018, doi: 10.1016/j.procir.2017.12.228.

[147] F. Psarommatis and D. Kiritsis, 'A hybrid Decision Support System for automating decision making in the event of defects in the era of Zero Defect Manufacturing', *J Ind Inf Integr*, vol. 26, p. 100263, Mar. 2022, doi: 10.1016/j.jii.2021.100263.

[148] V. Azamfirei, F. Psarommatis, and Y. Lagrosen, 'Application of automation for in-line quality inspection, a zero-defect manufacturing approach', *J Manuf Syst*, vol. 67, pp. 1–22, Apr. 2023, doi: 10.1016/j.jmsy.2022.12.010.

[149] N. Leberruyer, J. Bruch, M. Ahlskog, and S. Afshar, 'Toward Zero Defect Manufacturing with the support of Artificial Intelligence—Insights from an industrial application', *Comput Ind*, vol. 147, p. 103877, May 2023, doi: 10.1016/j.compind.2023.103877.

[150] L.-P. Zhao, B.-H. Li, and Y.-Y. Yao, 'A novel predict-prevention quality control method of multi-stage manufacturing process towards zero defect manufacturing', *Adv Manuf*, vol. 11, no. 2, pp. 280–294, Jun. 2023, doi: 10.1007/s40436-022-00427-9.

[151] S. T. Spantideas, A. E. Giannopoulos, N. C. Kapsalis, A. Angelopoulos, S. Voliotis, and P. Trakadas, 'Towards Zero-Defect Manufacturing: Machine Selection through Unsupervised Learning in the Printing Industry', in *CEUR Workshop Proceedings*, 2022.

[152] A. S. Kalafatelis, C. Trochoutsos, A. E. Giannopoulos, A. Angelopoulos, and P. Trakadas, 'A Stacking Ensemble Learning Model for Waste Prediction in Offset Printing', in *ACM International Conference Proceeding Series*, 2023. doi: 10.1145/3587889.3588210.

[153] Z. Huang, V. C. Angadi, M. Danishvar, A. Mousavi, and M. Li, 'Zero Defect Manufacturing of Microsemiconductors – An Application of Machine Learning and Artificial Intelligence', in *2018 5th International Conference on Systems and Informatics (ICSAI)*, IEEE, Nov. 2018, pp. 449–454. doi: 10.1109/ICSAI.2018.8599292.

[154] S. Dengler, S. Lahriri, E. Trunzer, and B. Vogel-Heuser, 'Applied machine learning for a zero defect tolerance system in the automated assembly of pharmaceutical devices', *Decis Support Syst*, vol. 146, p. 113540, Jul. 2021, doi: 10.1016/j.dss.2021.113540.

[155] T. Benbarrad, M. Salhaoui, S. B. Kenitar, and M. Arioua, 'Intelligent Machine Vision Model for Defective Product Inspection Based on Machine Learning', *Journal of Sensor and Actuator Networks*, vol. 10, no. 1, p. 7, Jan. 2021, doi: 10.3390/jsan10010007.

[156] A. Garre, M. C. Ruiz, and E. Hontoria, 'Application of Machine Learning to support production planning of a food industry in the context of waste generation under uncertainty', *Operations Research Perspectives*, vol. 7, p. 100147, 2020, doi: 10.1016/j.orp.2020.100147.

[157] R. Ndeda, A. Botes, and E. O. Olakanmi, 'Framework for Incorporating Machine Learning (ML) Driven Optimisation into Laser Materials Processing (LMP) Technologies for e-Mobility Applications towards Attaining Zero-Material Waste', *Lasers in Manufacturing and Materials Processing*, Nov. 2023, doi: 10.1007/s40516-023-00227-4.

[158] J. Moyne and J. Iskandar, 'Big Data Analytics for Smart Manufacturing: Case Studies in Semiconductor Manufacturing', *Processes*, vol. 5, no. 3, p. 39, 2017, doi: 10.3390/pr5030039.

[159] A. Diez-Olivan, J. Del Ser, D. Galar, and B. Sierra, 'Data fusion and machine learning for industrial prognosis: Trends and perspectives towards Industry 4.0', *Information Fusion*, vol. 50, pp. 92–111, 2019, doi: 10.1016/j.inffus.2018.10.005.

[160] G. A. Susto, A. Schirru, S. Pampuri, S. McLoone, and A. Beghi, 'Machine Learning for Predictive Maintenance: A Multiple Classifier Approach', *IEEE Trans Industr Inform*, vol. 11, no. 3, pp. 812–820, 2015, doi: 10.1109/tii.2014.2349359.

[161] G. A. Susto, A. Beghi, and C. De Luca, 'A Predictive Maintenance System for Epitaxy Processes Based on Filtering and Prediction Techniques', *IEEE Transactions on Semiconductor Manufacturing*, vol. 25, no. 4, pp. 638–649, 2012, doi: 10.1109/tsm.2012.2209131.

[162] J. Yan, Y. Meng, L. Lu, and L. Li, 'Industrial Big Data in an Industry 4.0 Environment: Challenges, Schemes, and Applications for Predictive Maintenance', *IEEE Access*, vol. 5, pp. 23484–23491, 2017, doi: 10.1109/access.2017.2765544.

[163] D. Wu, C. Jennings, J. Terpenny, R. X. Gao, and S. Kumara, 'A Comparative Study on Machine Learning Algorithms for Smart Manufacturing: Tool Wear Prediction Using Random Forests', *J Manuf Sci Eng*, vol. 139, no. 7, 2017, doi: 10.1115/1.4036350.

[164] H.-J. Shin, K.-W. Cho, and C.-H. Oh, 'SVM-Based Dynamic Reconfiguration CPS for Manufacturing System in Industry 4.0', *Wirel Commun Mob Comput*, vol. 2018, pp. 1–13, 2018, doi: 10.1155/2018/5795037.

[165] C.-J. Kuo, K.-C. Ting, Y.-C. Chen, D.-L. Yang, and H.-M. Chen, 'Automatic machine status prediction in the era of Industry 4.0: Case study of machines in a spring factory', *Journal of Systems Architecture*, vol. 81, pp. 44–53, 2017, doi: 10.1016/j.sysarc.2017.10.007.

[166] C.-C. Lin, L. Shu, D.-J. Deng, T.-L. Yeh, Y.-H. Chen, and H.-L. Hsieh, 'A MapReduce-Based Ensemble Learning Method with Multiple Classifier Types and Diversity for Condition-Based Maintenance with Concept Drifts', *IEEE Cloud Computing*, vol. 4, no. 6, pp. 38–48, 2017, doi: 10.1109/mcc.2018.1081065.

[167] L. L. Minku and X. Yao, ‘DDD: A New Ensemble Approach for Dealing with Concept Drift’, *IEEE Trans Knowl Data Eng*, vol. 24, no. 4, pp. 619–633, 2012, doi: 10.1109/tkde.2011.58.

[168] H. Ke, P. Li, S. Guo, and M. Guo, ‘On Traffic-Aware Partition and Aggregation in MapReduce for Big Data Applications’, *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 3, pp. 818–828, 2016, doi: 10.1109/tpds.2015.2419671.

[169] A. S. Kalafatelis, N. Nomikos, A. Angelopoulos, C. Trochoutsos, and P. Trakadas, ‘An Effective Methodology for Imbalanced Data Handling in Predictive Maintenance for Offset Printing’, 2024, pp. 89–98. doi: 10.1007/978-981-99-6523-6_7.

[170] W. Yu, T. Dillon, F. Mostafa, W. Rahayu, and Y. Liu, ‘A Global Manufacturing Big Data Ecosystem for Fault Detection in Predictive Maintenance’, *IEEE Trans Industr Inform*, vol. 16, no. 1, pp. 183–192, 2020, doi: 10.1109/tii.2019.2915846.

[171] R. S. Peres, A. Dionisio Rocha, P. Leitao, and J. Barata, ‘IDARTS – Towards intelligent data analysis and real-time supervision for industry 4.0’, *Comput Ind*, vol. 101, pp. 138–146, 2018, doi: 10.1016/j.compind.2018.07.004.

[172] M. Belichovski, D. Stavrov, F. Donchevski, and G. Nadzinski, ‘Unsupervised Machine Learning Approach for Anomaly Detection in E-coating Plant’, in *2022 IEEE 17th International Conference on Control & Automation (ICCA)*, IEEE, Jun. 2022, pp. 992–997. doi: 10.1109/ICCA54724.2022.9831858.

[173] H. Yan, J. Wan, C. Zhang, S. Tang, Q. Hua, and Z. Wang, ‘Industrial Big Data Analytics for Prediction of Remaining Useful Life Based on Deep Learning’, *IEEE Access*, vol. 6, pp. 17190–17197, 2018, doi: 10.1109/access.2018.2809681.

[174] C. Sun, M. Ma, Z. Zhao, S. Tian, R. Yan, and X. Chen, ‘Deep Transfer Learning Based on Sparse Autoencoder for Remaining Useful Life Prediction of Tool in Manufacturing’, *IEEE Trans Industr Inform*, vol. 15, no. 4, pp. 2416–2425, 2019, doi: 10.1109/tii.2018.2881543.

[175] Y. Cheng, H. Zhu, J. Wu, and X. Shao, ‘Machine Health Monitoring Using Adaptive Kernel Spectral Clustering and Deep Long Short-Term Memory Recurrent Neural Networks’, *IEEE Trans Industr Inform*, vol. 15, no. 2, pp. 987–997, 2019, doi: 10.1109/tii.2018.2866549.

[176] C. Shi, G. Panoutsos, B. Luo, H. Liu, B. Li, and X. Lin, ‘Using Multiple-Feature-Spaces-Based Deep Learning for Tool Condition Monitoring in Ultraprecision Manufacturing’, *IEEE Transactions on Industrial Electronics*, vol. 66, no. 5, pp. 3794–3803, 2019, doi: 10.1109/tie.2018.2856193.

[177] C. Liu, D. Tang, H. Zhu, and Q. Nie, ‘A Novel Predictive Maintenance Method Based on Deep Adversarial Learning in the Intelligent Manufacturing System’, *IEEE Access*, vol. 9, pp. 49557–49575, 2021, doi: 10.1109/ACCESS.2021.3069256.

[178] J. Liu, C. Zhang, and X. Jiang, ‘Imbalanced fault diagnosis of rolling bearing using improved MsR-GAN and feature enhancement-driven CapsNet’, *Mech Syst Signal Process*, vol. 168, p. 108664, Apr. 2022, doi: 10.1016/j.ymssp.2021.108664.

[179] K. Zhao and H. Shao, 'Intelligent Fault Diagnosis of Rolling Bearing Using Adaptive Deep Gated Recurrent Unit', *Neural Process Lett*, vol. 51, no. 2, pp. 1165–1184, Apr. 2020, doi: 10.1007/s11063-019-10137-2.

[180] B. Taşcı, A. Omar, and S. Ayvaz, 'Remaining useful lifetime prediction for predictive maintenance in manufacturing', *Comput Ind Eng*, vol. 184, p. 109566, Oct. 2023, doi: 10.1016/j.cie.2023.109566.

[181] C. Zhang, Y. Xie, H. Bai, B. Yu, W. Li, and Y. Gao, 'A survey on federated learning', *Knowl Based Syst*, vol. 216, p. 106775, Mar. 2021, doi: 10.1016/j.knosys.2021.106775.

[182] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, 'Federated Learning: Challenges, Methods, and Future Directions', *IEEE Signal Process Mag*, vol. 37, no. 3, pp. 50–60, May 2020, doi: 10.1109/MSP.2020.2975749.

[183] K. M. J. Rahman *et al.*, 'Challenges, Applications and Design Aspects of Federated Learning: A Survey', *IEEE Access*, vol. 9, pp. 124682–124700, 2021, doi: 10.1109/ACCESS.2021.3111118.

[184] S. Banabilah, M. Aloqaily, E. Alsayed, N. Malik, and Y. Jararweh, 'Federated learning review: Fundamentals, enabling technologies, and future applications', *Inf Process Manag*, vol. 59, no. 6, p. 103061, Nov. 2022, doi: 10.1016/j.ipm.2022.103061.

[185] L. Li, Y. Fan, M. Tse, and K.-Y. Lin, 'A review of applications in federated learning', *Comput Ind Eng*, vol. 149, p. 106854, Nov. 2020, doi: 10.1016/j.cie.2020.106854.

[186] M. Aledhari, R. Razzak, R. M. Parizi, and F. Saeed, 'Federated Learning: A Survey on Enabling Technologies, Protocols, and Applications', *IEEE Access*, vol. 8, pp. 140699–140725, 2020, doi: 10.1109/ACCESS.2020.3013541.

[187] N. Bouacida and P. Mohapatra, 'Vulnerabilities in Federated Learning', *IEEE Access*, vol. 9, pp. 63229–63249, 2021, doi: 10.1109/ACCESS.2021.3075203.

[188] L. Fu, H. Zhang, G. Gao, M. Zhang, and X. Liu, 'Client Selection in Federated Learning: Principles, Challenges, and Opportunities', *IEEE Internet Things J*, vol. 10, no. 24, pp. 21811–21819, Dec. 2023, doi: 10.1109/JIOT.2023.3299573.

[189] P. Qi, D. Chiaro, A. Guzzo, M. Ianni, G. Fortino, and F. Piccialli, 'Model aggregation techniques in federated learning: A comprehensive survey', *Future Generation Computer Systems*, vol. 150, pp. 272–293, Jan. 2024, doi: 10.1016/j.future.2023.09.008.

[190] J. Chen, R. Monga, S. Bengio, and R. Józefowicz, 'Revisiting Distributed Synchronous SGD', *ArXiv*, vol. abs/1604.00981, 2016, [Online]. Available: https://api.semanticscholar.org/CorpusID:593493

[191] T. Tinga, W. W. Tiddens, F. Amoiralis, and M. Politis, 'Predictive maintenance of maritime systems: models and challenges', in *Safety & Reliability - Theory and Applications*, M. Cepin and R. Bris, Eds., United Kingdom: Taylor & Francis, Jun. 2017, pp. 421–429. doi: 10.1201/9781315210469-56.

[192] G. Makridis, D. Kyriazis, and S. Plitsos, 'Predictive maintenance leveraging machine learning for time-series forecasting in the maritime industry', in *2020 IEEE 23rd*

International Conference on Intelligent Transportation Systems (ITSC), 2020, pp. 1–8. doi: 10.1109/ITSC45102.2020.9294450.

[193] M. Cheliotis, I. Lazakis, and G. Theotokatos, 'Machine learning and data-driven fault detection for ship systems operations', *Ocean Engineering*, vol. 216, p. 107968, 2020, doi: https://doi.org/10.1016/j.oceaneng.2020.107968.

[194] A. Bakdi, N. B. Kristensen, and M. Stakkeland, 'Multiple Instance Learning With Random Forest for Event Logs Analysis and Predictive Maintenance in Ship Electric Propulsion System', *IEEE Trans Industr Inform*, vol. 18, no. 11, pp. 7718–7728, 2022, doi: 10.1109/TII.2022.3144177.

[195] V. J. Jimenez, N. Bouhmala, and A. H. Gausdal, 'Developing a predictive maintenance model for vessel machinery', *Journal of Ocean Engineering and Science*, vol. 5, no. 4, pp. 358–386, 2020, doi: https://doi.org/10.1016/j.joes.2020.03.003.

[196] S. Liu, H. Chen, B. Shang, and A. Papanikolaou, 'Supporting Predictive Maintenance of a Ship by Analysis of Onboard Measurements', *J Mar Sci Eng*, vol. 10, no. 2, 2022, doi: 10.3390/jmse10020215.

[197] Z. Zhang, C. Guan, H. Chen, X. Yang, W. Gong, and A. Yang, 'Adaptive Privacy-Preserving Federated Learning for Fault Diagnosis in Internet of Ships', *IEEE Internet Things J*, vol. 9, no. 9, pp. 6844–6854, 2022, doi: 10.1109/JIOT.2021.3115817.

[198] L. S. Dalenogare, G. B. Benitez, N. F. Ayala, and A. G. Frank, 'The expected contribution of Industry 4.0 technologies for industrial performance', *Int J Prod Econ*, vol. 204, pp. 383–394, 2018, doi: https://doi.org/10.1016/j.ijpe.2018.08.019.

[199] J. and M. J. P. and K. D. Psarommatis F. and Sousa, 'Zero-defect manufacturing the approach for higher manufacturing sustainability in the era of industry 4.0: a position paper', *Int. J. Prod. Res.*, vol. 60, 2021.

[200] A. Angelopoulos *et al.*, 'Tackling Faults in the Industry 4.0 Era—A Survey of Machine-Learning Solutions and Key Aspects', *Sensors*, vol. 20, no. 1, 2020, doi: 10.3390/s20010109.

[201] A. Kaloxylos, A. Gavras, D. Camps Mur, M. Ghoraishi, and H. Hrasnica, 'AI and ML – Enablers for Beyond 5G Networks'. Zenodo, May 2021. doi: 10.5281/zenodo.4299895.

[202] R. and P. A. S. and K. N. and S. S. and H. A. Amon-Tran I. and Anayath, 'An approach to minimize carbon footprint for an environmental friendly printing by optimizing an offset machine in a printing facility', *Procedia Soc. Behav. Sci.*, vol. 37, 2012, doi: 10.1016/j.sbspro.2012.03.316.

[203] D. and G. R. and O.-M. J. and B. M. and W. W. Villalba-Diez J. and Schmidt, 'Deep learning for industrial computer vision quality control in the printing industry 4.0', *Sensors*, vol. 19, 2019, doi: 10.3390/s19183987.

[204] S. Dengler, S. Lahriri, E. Trunzer, and B. Vogel-Heuser, 'Applied machine learning for a zero defect tolerance system in the automated assembly of pharmaceutical devices', *Decis Support Syst*, vol. 146, p. 113540, Jul. 2021, doi: 10.1016/j.dss.2021.113540.

[205] F. Pedregosa *et al.*, 'Scikit-Learn: Machine Learning in Python', *J. Mach. Learn. Res.*, vol. 12, no. null, pp. 2825–2830, Nov. 2011.

[206] J. Gama, I. Žliobaitundefined, A. Bifet, M. Pechenizkiy, and A. Bouchachia, 'A Survey on Concept Drift Adaptation', *ACM Comput. Surv.*, vol. 46, no. 4, Jan. 2014, doi: 10.1145/2523813.

[207] G. I. Webb, R. Hyde, H. Cao, and F. Nguyen Hai Long and Petitjean, 'Characterizing concept drift', *Data Min Knowl Discov*, vol. 30, no. 4, pp. 964–994, Jul. 2016.

[208] M. Baena-García, J. Campo-Ávila, R. Fidalgo-Merino, A. Bifet, R. Gavald, and R. Morales-Bueno, 'Early Drift Detection Method', ., Jan. 2006.

[209] A. P. Dawid, 'Present position and potential developments: some personal views: statistical theory: the prequential approach', *J. Roy. Stat. Soc. Ser. A (Gen.)*, vol. 147, 1984, doi: 10.2307/2981683.

[210] W. N. Street and Y. Kim, 'A Streaming Ensemble Algorithm (SEA) for Large-Scale Classification', in *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, in KDD '01. New York, NY, USA: Association for Computing Machinery, 2001, pp. 377–382. doi: 10.1145/502512.502568.

[211] J. C. Schlimmer and R. H. Granger, 'Incremental learning from noisy data', *Mach Learn*, vol. 1, no. 3, pp. 317–354, Sep. 1986, doi: 10.1007/BF00116895.

[212] G. Hulten, L. Spencer, and P. Domingos, 'Mining time-changing data streams', in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, New York, NY, USA: ACM, Aug. 2001, pp. 97–106. doi: 10.1145/502512.502529.

[213] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification And Regression Trees*. Routledge, 2017. doi: 10.1201/9781315139470.

[214] M. Harries, U. Nsw-cse-tr, and N. Wales, 'SPLICE-2 Comparative Evaluation: Electricity Pricing', Feb. 2003.

[215] R. Cattral and F. Oppacher, 'Discovering rules in the poker hand dataset', in *Proceedings of the 9th annual conference on Genetic and evolutionary computation*, New York, NY, USA: ACM, Jul. 2007, pp. 1870–1870. doi: 10.1145/1276958.1277329.

[216] J. Blackard, 'Covertype'. 1998.

[217] T. Xia, M. M. Wang, J. Zhang, and L. Wang, 'Maritime Internet of Things: Challenges and Solutions', *IEEE Wirel Commun*, vol. 27, no. 2, pp. 188–196, 2020, doi: 10.1109/MWC.001.1900322.

[218] W. Tao *et al.*, 'Coordination and Optimization Control Framework for Vessels Platooning in Inland Waterborne Transportation System', *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 12, pp. 15667–15686, 2023, doi: 10.1109/TITS.2022.3220000.

[219] H. Yu *et al.*, 'Ship Path Optimization That Accounts for Geographical Traffic Characteristics to Increase Maritime Port Safety', *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 6, pp. 5765–5776, 2022, doi: 10.1109/TITS.2021.3057907.

[220] N. Nomikos, P. K. Gkonis, P. S. Bithas, and P. Trakadas, 'A Survey on UAV-Aided Maritime Communications: Deployment Considerations, Applications, and Future Challenges', *IEEE Open Journal of the Communications Society*, vol. 4, pp. 56–78, 2023, doi: 10.1109/OJCOMS.2022.3225590.

[221] O. A. Wahab, A. Mourad, H. Otrok, and T. Taleb, 'Federated Machine Learning: Survey, Multi-Level Classification, Desirable Criteria and Future Directions in Communication and Networking Systems', *IEEE Communications Surveys & Tutorials*, vol. 23, no. 2, pp. 1342–1397, 2021, doi: 10.1109/COMST.2021.3058573.

[222] A. Giannopoulos, P. Gkonis, P. Bithas, N. Nomikos, G. Ntroulias, and P. Trakadas, 'Federated Learning for Maritime Environments:Use Cases, Experimental Results, and Open Issues', Feb. 2023, doi: 10.36227/techrxiv.22133549.v1.

[223] P. Trakadas *et al.*, 'Hybrid Clouds for Data-Intensive, 5G-Enabled IoT Applications: An Overview, Key Issues and Relevant Architecture', *Sensors (Basel)*, vol. 19, no. 16, p. 3591, Aug. 2019, doi: 10.3390/s19163591.

[224] G. Aiello, A. Giallanza, and G. Mascarella, 'Towards Shipping 4.0. A preliminary gap analysis', *Procedia Manuf*, vol. 42, pp. 24–29, 2020, doi: https://doi.org/10.1016/j.promfg.2020.02.019.

[225] F. Cipollini, L. Oneto, A. Coraddu, A. J. Murphy, and D. Anguita, 'Condition-Based Maintenance of Naval Propulsion Systems with supervised Data Analysis', *Ocean Engineering*, vol. 149, pp. 268–278, 2018, doi: https://doi.org/10.1016/j.oceaneng.2017.12.002.

[226] A. Coraddu, L. Oneto, A. Ghio, S. Savio, D. Anguita, and M. Figari, 'Condition based maintenance of naval propulsion plants data set', *UCI Machine Learning Repository*, 2014.

[227] D. Mohakul, 'Predictive Maintenance on Ship's Main Engine using AI'. IEEE Dataport, 2022. doi: 10.21227/g3za-v415.

[228] I. J. Goodfellow *et al.*, 'Generative Adversarial Networks', Jun. 2014.

[229] I. Ashrapov, 'Tabular GANs for uneven distribution', Oct. 2020.

[230] N. Nomikos, A. Giannopoulos, P. Trakadas, and G. K. Karagiannidis, 'Uplink NOMA for UAV-Aided Maritime Internet-of-Things', in *2023 19th International Conference on the Design of Reliable Communication Networks (DRCN)*, 2023, pp. 1–6. doi: 10.1109/DRCN57075.2023.10108290.

[231] C. Tsinos, S. Spantideas, A. Giannopoulos, and P. Trakadas, 'Over-the-Air Computation with Quantized CSI and Discrete Power Control Levels', *Wirel Commun Mob Comput*, vol. 2023, p. 8559701, 2023, doi: 10.1155/2023/8559701.

[232] S. T. Spantideas, A. E. Giannopoulos, N. C. Kapsalis, A. Kalafatelis, C. N. Capsalis, and P. Trakadas, 'Joint Energy-efficient and Throughput-sufficient Transmissions in 5G Cells with Deep Q-Learning', in *2021 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*, 2021, pp. 265–270. doi: 10.1109/MeditCom49071.2021.9647592.