H2020 - INDUSTRIAL LEADERSHIP - Information and Communication Technologies (ICT)

ICT-38-2020: Artificial intelligence for manufacturing

Grant Agreement: 957362

Explainable Manufacturing Artificial Intelligence

XMANAI

MAKING AI UNDERSTANDABLE

WP1: Explainable AI Foundations Elaboration and XMANAI Concept Fusion

D1.1: State of the Art Review in XMANAI Research Domains

Deliverable Leader: POLIMI

Due Date: M6

Dissemination Level: Public

Version: F1.0

D1.1 provides an overview over the Explainable Artificial Intelligence (XAI) domain and it is structured as a collection of existing methods and algorithms, that can be used as inspiration for developing the XMANAI methodology.

It presents the state-of-the-art on XAI and machine learning from a theoretical perspective, comparing different methods and tools. Furthermore, it includes the analysis of open-source solutions for Artificial Intelligence (AI) implementation and a look to XAI applications in industry.

To provide a complete overview, the analysis is complemented by the exploration of human aspects in decision making and AI.

Further Information: www.ai4manufacturing.eu

Document Log

Contributors	Silvia Razzetti (POLIMI), Vasilis Gkolemis (ATHENA), Eleni Lavasa (ATHENA), Fenareti Lampathaki (Suite5), Evmorfia Biliri (Suite5), Ahmad Mehrbod (KBIZ), Serafeim Moustakidis (AiDEAS), Patrik Karlsson (AiDEAS), Zoumpoulia Dikopoulou (AiDEAS), David Monzó (Tyris AI), Sergi Pérez (Tyris AI)
Internal Reviewer 1	Márcio Saraiva (KBIZ)
Internal Reviewer 2	Sebastian Urbanek (FRAUNHOFER)
Type	Report
Delivery Date	M6

History

Versions	Description
D0.1	Finalized the structure of the TOC and collected all contributions for chapters 2-6, from all partners collaborating in T1.1 and T1.2. Missing Introduction, Conclusion, Executive Summary, Annex and List of abbreviation
D0.2	Added missing paragraph: Introduction, Conclusion, Executive Summary, Annex and List of abbreviation
D0.3	Internal review
R0.1	Revision of Márcio Saraiva (KBIZ)
R0.2	Revision of Sebastian Urbanek (FRAUNHOFER)
R0.3	Revision of Fenareti Lampathaki (Suite5), on top of version R0.1
D1.0	Contain KBIZ feedback in track changes mode
D1.1	Internal review by all partners incorporating feedback from KBIZ, FRAUNHOFER and Suite5
F1.0	Final version

Executive Summary

Deliverable 1.1's purpose is to provide an overview of the **state-of-the-art of artificial intelligence**, with a specific focus on eXplanaible AI (XAI), that is, models and techniques aiming at overcoming the so called "black-box" issue.

The document is the result of six months of researches conducted in parallel by "Task 1.1 - Explainable AI and Graph Machine Learning Analytics State-of-Play" and "Task 1.2 - Human Aspects in Decision Making and AI". It involved 8 partners and it required also the collaboration of the 4 pilots.

The objective is to present a document that builds the conceptual basis for the development of XMANAI platform and that can be taken as inspiration during the analysis and collection of its requirements and during the implementation of the platform itself. Hence, the approach followed in the deliverable (also according to the DoA) is to run the investigation both from a technical and a business point of view.

- Task 1.1 deals with the analysis of existing methods, components and tools related to XAI, with a specific focus on graph machine learning analytics. Researches conducted in the task are reflected in the first sections of the deliverable, where it is presented an overview of the state-of-the-art of explainable AI.

  First of all, a preliminary activity to identify the most relevant aspects of explainable AI was run, in order to orientate in front of the huge amount of papers available and to make easier the selection of those useful for the project's purposes.

  Existing literature about the XAI subject has been analysed in details in order to collect the information required to describe models developed as of now and to provide to the reader a full picture of the current ecosystem of models and solutions. More than 200 between papers, books and articles have been taken into account to write the deliverable.

  Besides of the analysis of the existing literature that provides a theoretical perspective of the subject, a further research was conducted about tools and application of XAI, to complement the overview.

  The most important available open-source tools have been investigated, analysing their pros and cons and focusing on how XMANAI can exploit each one.

  Then, existing implementations of explainable AI have been explored, to identify projects where concepts relevant to XMANAI have been already developed (or at least approached), in order to define a starting point for the projects.

  The list of tools and applications is presented in the deliverable.

- To complement the technical pictures provided by Task 1.1, Task 1.2's goals is to understand how AI can incorporate various aspects of human thinking when it comes to decision making, in order to identify the requirements of XMANAI platform from a business point of view.

  Two kinds of activities were run in Task 1.2.

  Firstly, the existing literature about decision making has been analysed, focusing on three main areas: how a decision making process works (methods and techniques adopted), and in particular, how it works in manufacturing; how AI can support humans in decision making and which are the end user's expectations in terms of trustworthiness; ethical issues related to AI adoption in decision making.

Then, key aspects identified in previous analysis were validated thanks to the pilots' contribution: with a more practical approach, four interviews were conducted with the XMANAI pilots, with the objective of collecting information about human thinking in decision making and to sketch a preliminary draft of expected outputs from the XMANAI platform. Results are summarized in last section of the deliverable, while the interview's questions are reported in the annex.

Table of Contents

Executive Summary ..... ***iii***

1 Introduction ..... ***x***

1.1 XMANAI Project Overview x

1.2 Deliverable Purpose and Scope x

1.3 Impact and Target Audiences xi

1.4 Deliverable Methodology xi

1.5 Dependencies in XMANAI and Supporting Documents xi

1.6 Document Structure xii

2 Background ***1***

2.1 XAI concepts & objectives 1

2.2 XAI methods 1

3 XAI Methods and Approaches ***3***

3.1 Explainability by design 3

3.1.1 Linear Models 3

3.1.2 Generalized Linear Models and Generalized Additive Models 4

3.1.3 Decision trees 5

3.1.4 Rule-Based models 5

3.1.5 Bayesian models 5

3.1.6 k-Nearest Neighbors 6

3.1.7 Advantages & Limitations 6

3.2 Post-hoc Explainability techniques: Model-agnostic 6

3.2.1 Explanation by simplification 6

3.2.2 Explanation of feature relevance 9

3.2.3 Explanation by visualization 12

3.2.4 Explanation by example 20

3.2.5 Advantages & Limitations 24

3.3 Post-hoc Explainability techniques: Model-specific 24

3.3.1 Machine Learning models 25

3.3.2 Deep Learning models 32

3.3.3 Advantages & Limitations 36

3.4 Graph ML techniques 36

3.4.1 Traditional ML techniques on graphs 38

3.4.2 Graph Representation Learning 39

3.4.3 Geometric Deep Learning 42

3.4.4 Using knowledge graphs to explain other models 47

3.4.5 Advantages & limitations 48
3.5 Hybrid techniques 50
3.5.1 Fusion of domain knowledge in opaque models 51
3.5.2 Coupled opaque and transparent models 51
3.5.3 Ensemble of stacked opaque and transparent models 53
3.5.4 Advantages & limitations 54
4 XAI Tools 55
4.1 Explainability by design 55
4.2 Post-hoc explainability techniques: Model-Agnostic 55
4.3 Post-hoc explainability techniques: Model-Specific 56
4.3.1 Tree Ensembles 56
4.3.2 Support Vector Machine 57
4.3.3 Deep Learning Models 57
4.4 Graph Machine Learning 58
4.5 Other tools 58
4.6 List with XAI tools 59
5 XAI in Manufacturing 61
5.1 XAI Applications in industry 61
5.1.1 Demand Planning 61
5.1.2 Product Design 61
5.1.3 Inventory/ Supply Chain Management 62
5.1.4 Production Management 62
5.1.5 Process control 63
5.1.6 Quality control 64
5.1.7 Maintainance 65
5.2 Use cases – projects 65
6 Human aspects in AI Decision Making 70
6.1 Human Centric decision-making 70
6.1.1 Internal and external factors that influence decisions 70
6.1.2 Existing methods and approaches 71
6.1.3 A focus on decision making in manufacturing 71
6.2 AI in decision-making, from a user perspective 73
6.2.1 AI on data analysis 73
6.2.2 AI on knowledge representation 74
6.2.3 AI on decision making 74
6.3 The Collaborative Intelligence Human-Machine interaction model 75
6.3.1 Human assisting Machines 76

6.3.2 Machines assisting Humans 76
6.3.3 Other methods for Industry5.0 77
6.4 Ethical Issues on AI and Human Interaction 77
6.4.1 EU ethics guidelines for trustworthy AI the seven key requirements 77
6.4.2 Ethical risks of human- AI interaction in industry 78
6.4.3 Trust, transparency and explainability in decision making 78
6.4.4 Explainability AI, responsibility and accountability 79
6.5 Most relevant results from interviews 79
7 Conclusion 82
8 References 83
List of Acronyms/Abbreviations 100
Annexes 102
Annex I - Interviews with decision makers: questions 102

List of Figures

FIGURE 3-1 EXAMPLE OF EXPLAINABILITY IN A LINEAR REGRESSION MODEL, TAKEN FROM HTTPS://LAWTOMATED.COM/EXPLAINABLE-AI-ALL-YOU-NEED-TO-KNOW-THE-WHAT-HOW-WHY-OF-EXPLAINABLE-AI/ 4

FIGURE 3-2 - EXAMPLE ILLUSTRATING THE EXPRESSIVE CAPABILITY OF A SIMPLE LINEAR MODEL AND A GAM. IMAGE TAKEN FROM HTTPS://CHRISTOPHM.GITHUB.IO/INTERPRETABLE-ML-BOOK/ 5

FIGURE 3-3-- EXPLAINABILITY OF DECISION TREES AND DERIVATE PREDICTIVE MODELS, TAKEN FROM HTTPS://DOI.ORG/10.1038/S42256-019-0142-0 5

FIGURE 3-4 - EXAMPLE ILLUSTRATING HOW LIME (LOCAL INTERPRETABLE MODEL-AGNOSTIC EXPLANATIONS) WORK 9

FIGURE 3-5 - EXAMPLE OF A PDP CURVE, AS TAKEN FROM FRIEDMAN 2001 10

FIGURE 3-6 - ICE CURVES AND PDP (AVERAGE OF THE CURVES). IMAGE TAKEN FROM GOLDSTEIN 2015 10

FIGURE 3-7-- CARTOON ILUSTRATION OF EXPLANATION MODELS WITH TREE SHAP FROM THE LUNDBERG 2018. 12

FIGURE 3-8 - HEATMAP EXPLANATIONS FOR THE TOP 2 PREDICTED CLASSES, BAGEL (LEFT) AND STRAWBERRIES (RIGTHT) MADE BY GOOGLE'S INCEPTION V3 NEURAL NETWORK USING LOCAL INTERPRETABLE MODEL-AGNOSTIC EXPLANATION. THE PROBABILITY OF BEGEL WAS 77% WHEREAS FOR STRAWBERRY WAS 4%. GREEN SIGNIFIES THE AREAS THAT INCREASED THE PROBABILITY AND RED THE AREAS THAT DECREASED IT. WE OBSERVE THAT THE EXPLANATIONS ARE VERY REASONABLE. THE MODEL PREDICTED IT IS BAGEL (EVEN THOUGH IT IS NOT SINCE THE HOLE IN THE MIDDLE IS MEASING) EMPHASIZING IN THE BREADS AND THAT IT IS NOT STRAWBERRIES BASED AGAIN IN THE BREAD AREA. 13

FIGURE 3-9 - VERTICAL BARPLOT EXPLANATIONS OF SIX INSTANCES OF THE BIKE RENTAL DATASET USING ANCHORS. EACH ROW REPRESENTS ONE EXPLANATION OR ANCHOR AND EACH BAR OUTLINES THE FEATURE PREDICATES CONTAINED BY IT. THE X-AXIS DISPLAYS A RULE'S PRECISION, AND A BAR'S TH BAR'S THICKNESS CORRESPONDS TO ITS COVERAGE 14

FIGURE 3-10 - PLOT EXPLANATIONS USING PERMUTATION FEATURE IMPORTANCE INDICATING THE IMPORTANCE OF EACH FEATURE FOR PREDICTING CERVICAL CANCER WITH A RANDOM FOREST 15

FIGURE 3-11 - BOXPLOT EXPLANATIONS USING GLOBAL SURROGATE. THE TERMINAL NODES OF A SURROGATE TREE THAT APPROXIMATES THE PREDICTIONS OF A SUPPORT VECTOR MACHINE TRAINED ON THE BIKE RENTAL DATASET 15

FIGURE 3-12 - HEATMAP MATRIX EXPLANATION USING PARTIAL DEPENDENCE PLOT ILLUSTRATING THE CANCER PROBABILITY AND THE INTERACTION OF TWO FEATURES (AGE AND NUMBER OF PREGNANCIES) 16

FIGURE 3-13 - LINE CHART EXPLANATIONS USING INDIVIDUAL CONDITIONAL EXPECTATION. EACH LINE REPRESENTS AN OBSERVATION. THE YELLOW LINE REPRESENTS THE CENTERED CURVE AT A CERTAIN POINT OF THE CURVES IN THE FEATURE; WHILE THE BLACK LINES DISPLAY THE DIFFERENCE IN THE PREDICTION TO THIS POINT. 16

FIGURE 3-14 - A COMBINATION OF SCATTERPLOT AND HEATMAP EXPLANATION USING ACCUMULATED LOCAL EFFECTS PRESENTING THE INTERACTION EFFECT OF HE 2ND ORDER EFFECT OF TWO NUMBER OF PREGNANCIES AND AGE. FOR INSTANCE, THE PLOT SHOWS AN ODD MODEL BEHAVIOR AT AGE OF 18-20 AND MORE THAN 3 PREGNANCIES (UP TO 5 PERCENTAGE POINT INCREASE IN CANCER PROBABILITY). 17

FIGURE 3-15 - BARPLOT EXPLANATION USING SHAPLEY VALUES PRESENTING THE AVERAGE CONTRIBUTION OF A FEATURE VALUE TO THE PREDICTION IN DIFFERENT COALITIONS. 18

FIGURE 3-16 - SUMMARY PLOT EXPLANATIONS USING SHAPLEY ADDITIVE EXPLANATIONS. EACH POINT ON THE SUMMARY PLOT IS A SHAPLEY VALUE FOR A FEATURE AND AN INSTANCE. THE POSITION ON THE Y-AXIS IS DETERMINED BY THE FEATURE AND ON THE X-AXIS BY THE SHAPLEY VALUE. THE COLOR REPRESENTS THE VALUE OF THE FEATURE FROM LOW TO HIGH. 18

FIGURE 3-17 - LOLLIPOP CHART EXPLANATION USING FEATURE INTERACTION THE INTERACTION STRENGTH (H-STATISTIC) FOR EACH FEATURE WITH ALL OTHER FEATURES FOR A SUPPORT VECTOR MACHINE PREDICTING BICYCLE RENTALS 19

FIGURE 3-18 - ILLUSTRATION OF (A) AN UNDIRECTED WEIGHTED GRAPH AND (B) A DIRECTED WEIGHTED GRAPH. EACH EDGE CARRIES A WEIGHT INDICATING THE STRENGTH OF INFLUENCE AMONG NODES. BLUE EDGES SPECIFY POSITIVE RELATIONS; WHILE, RED EDGES DENOTE NEGATIVE RELATIONS BETWEEN NODES. AN UNDIRECTED WEIGHTED GRAPH IS ILLUSTRATED ON THE LEFT AND A DIRECTED WEIGHTED NETWORK ON THE RIGHT. THE SIZE OF EACH NODE IS SET ACCORDING TO THE DEGREE-CENTRALITY. 20

FIGURE 3-19 - PROTOTYPES AND CRITICISMS FOR TWO DIFFERENT DOG BREEDS, LEARNT BY THE IMAGENET (KIM2016). 22

FIGURE 3-20 - COMPARING THE ACTUAL DIFFERENCE IN LOSS, CALCULATED THROUGH LEAVE-ONE-OUT TRAINING, TO THE PREDICTED ONE BY KOH & LIANG (2017). FROM LEFT TO RIGHT, RESULTS ARE PRESENTED FOR LOGISTIC REGRESSION (LINEAR), CONVOLUTIONAL NN AND RBF SVM MINIMIZING THE HINGE LOSS, WITH VARIOUS LEVELS OF SMOOTHING. 23

FIGURE 3-21 - SUMMARY OF EXPLANATIONS FOR A CORRECT PREDICTION OF A TEST FISH (TOP LEFT) BY RBF SVM (MIDDLE ROW) AND THE INCEPTION CNN (BOTTOM ROW). THE MOST INFLUENTIAL INSTANCES FOR EACH MODEL ARE INDICATIVE TO THE LEARNING PROCESS (KOH & LIANG 2017). 23

FIGURE 3-22 - THE TMC-SHAPLEY APPROXIMATION BY GIORDANI & ZHOU (2019) ENABLES DEBUGGING THE TRAINING DATA. INSTANCES WITH THE LEAST DATA SHAPLEY ATTRIBUTIONS ARE SUSPICIOUS AS MISLABELED SAMPLES (1ST TO 3RD PANEL), WHILE DEGRADATION DUE TO GAUSSIAN NOISE IS DETECTED THROUGH DIMINISHING SHAPLEY VALUES (4TH TO 6TH PANEL). 24

FIGURE 3-23 - LOCAL EXPLANATIONS CREATED BY THE TREEEXPLAINER CAN BE DISPLAYED IN RICH VISUALIZATIONS, TO PROVIDE A GLOBAL COMPREHENSION OF THE ENSEMBLE MODEL'S PREDICTIONS. IMAGE FROM (LUNDBERG, ET AL., 2019) 27

FIGURE 3-24 - SUBGROUPS OF PEOPLE WITH SIMILAR MORTALITY RISKS ARE CLUSTERED TOGETHER IN A FEATURE ATTRIBUTION EMBEDDING. IMAGE FROM (LUNDBERG, ET AL., 2019) 28

FIGURE 3-25 - GLOBAL FEATURE ATTRIBUTION ON THE MNIST DATASET FOR HANDWRITTEN DIGIT RECOGNITION. A BOOSTED TREE ENSEMBLE WAS BUILT FOR EACH DIGIT, AS A ONE-VS-REST CLASSIFIER USING CATBOOST. EACH CLASSIFIER WAS PROCESSED WITH MONOFOREST, GLOBAL (MEAN ABSOLUTE) SHAP VALUES AND MODELRELIANCE (BASED ON PERMUTATION IMPORTANCE) AND ANALYSIS RESULTS ARE DISPLAYED IN THE UPPER, MIDDLE AND BOTTOM ROW RESPECTIVELY. IMAGE FROM (KURALENOK, ET AL., 2019) 28

FIGURE 3-26 - VISUALIZATION OF SVM FOR BINARY CLASSIFICATION. LINEAR SVM ON LINEARLY SEPARABLE CLASSES IS DISPLAYED IN THE 1ST PANEL. SVM WITH LINEAR AND RBF KERNEL ARE DISPLAYED IN THE 2ND AND 3RD PANEL RESPECTIVELY, WHERE CLASSES ARE NOT LINEARLY SEPARABLE. THE SOLID LINE IS THE DECISION BOUNDARY, DASHED LINE INDICATES THE MARGIN AND SUPPORT VECTORS ARE IN CIRCLES. 30

FIGURE 3-27 - THE UNIVARIATE HISTOGRAM OF PROJECTIONS VISUALIZES THE POSITION OF TRAINING SAMPLES WITH RESPECT TO THE SVM DECISION BOUNDARY AND MARGIN (CHERKASSKY & DHAR, 2010). 31

FIGURE 3-28 - FEATURE ATTRIBUTION ON NEUROIMAGING DATA. RESULTS BY GAONKAR (2015) ARE DISPLAYED ON THE 1ST AND 2ND PANEL, THE 3RD PANEL IS PRODUCED FROM THE METHOD BY USTUN (2007), WHILE THE LAST PANEL PRESENTS THE GROUND TRUTH 31

FIGURE 3-29 - USING THE COLOR-BASED NOMOGRAM TO EXPLAIN SVM ON THE GERMAN CREDIT RISK DATASET. THE CREDITABILITY OF A PERSON ("SCORE") CAN BE ASSESSED AS THE SUM OF MAIN (UPPER ROW PANELS) AND PAIRWISE (MIDDLE PANELS) CONTRIBUTIONS. 32

FIGURE 3-30 - PROGRESSIVE ENHANCEMENT (FROM LEFT TO RIGHT) OF THE VISUAL FEATURES INSIDE THE DEEP NEURAL NETWORK. THE SIMPLE EDGES ARE COMBINED TO CREATE TEXTURES AND PATTERNS, WHICH ARE USED AS THE BUILDING BLOCKS FOR THE PARTS OF WHOLE OBJECTS. IMAGES ARE TAKEN FROM HTTPS://DISTILL.PUB/2017/FEATURE-VISUALIZATION. 34

FIGURE 3-31 - SOME EXAMPLES OF FEATURE VISUALIZATION. WE OBSERVE THAT SOME CASES THE IMAGE IS NOT INTERPRETABLE AT ALL (LEFT IMAGE) WHILE IN SOME OTHERS, IT IS MUCH CLEARER. FOR EXAMPLE, THE THIRD IMAGE CLEARLY CORRESPONDS TO A MONKEY. 34

FIGURE 3-32 - USING TEXT TO EXPLAIN WHAT THE CNN UNDERSTANDS; IN THE CURRENT EXAMPLE, THE NETWORK OUTPUTTED "A WOMAN IS THROWING A FRISBEE IN THE PARK". THE FIGURE IS TAKEN FROM (BENGIO, SIMARD AND FRASCONI, 2016) 35

FIGURE 3-33 - FEATURE RELEVANCE EXAMPLE IN THE NLP DOMAIN. STRONGER RED CORRESPONDS TO WORDS THAT CONTRIBUTED MORE TO THE PREDICTION. IMAGE IS TAKEN FROM (ARRAS ET AL.) 36

FIGURE 3-34 - STEPS OF GRAPH DATA SCIENCE (IMAGE FROM NEO4J SITE) 37

FIGURE 3-35 - SYSTEM PROPOSED BY BENNETOT 2019. 51

FIGURE 3- 36 - INTUITION BEHIND DKNN. THE IMAGE ON THE LEFT SHOWS THE ARCHITECTURE OF THE DNN. THE MIDDLE IMAGE SHOWS THE REPRESENTATION OBTAINED BY EACH LAYER. FINALLY, THE IMAGE ON THE RIGHT SHOWS THE NEAREST NEIGHBORS FOUND ON EACH LAYER. DKNN WOULD INDICATE THAT THE REAL PANDA IS COMPLIANT BUT ITS ADVERSARY PANDA IS NOT. IMAGE FROM PAPERNOT 2018. 52

FIGURE 3-37 - SYSTEM PROPOSED BY LOYOLA-GONZALEZ 2019 54

FIGURE 5-1 VARIOUS XAI METHODS WITH THEIR RELEVANCE HEATMAPS ON A TIME SERIES PREDICTION TASK (SCHLEGEL, ET AL., 2019) 61

FIGURE 5-2 HOW AN AI-CREATED DIGITAL TWIN EFFECTS PRODUCT DESIGN AND DEVELOPMENT IN PETROCHEMICAL INDUSTRY (MIN, ET AL., 2019) 62

FIGURE 5-3 SELF-THINKING SUPPLY CHAIN (CALATAYUD, ET AL., 2019) 62

FIGURE 5-4 EXAMPLE OF EXPLAINABLE AI (VARIABLE IMPORTANCE) IN DEFECTIVE PRODUCTS PREDICTION IN THE STEEL PLATES INDUSTRY (KHARAL, 2020) 63

FIGURE 3-5-5. A CYBER-PHYSICAL SYSTEM FORMED BY THE COMPONENTS OF AI AIDED, SMART MANUFACTURING (ARINEZ, ET AL., 2020). 64

FIGURE 3-5-6. THE SPECTRUM OF HUMAN-MACHINE COLLABORATION IN A SUPERVISORY CONTROL OPERATIONAL FRAMEWORK (ARINEZ, ET AL., 2020). 65

FIGURE 6-1 - RATIO OF HUMAN-MACHINE WORKING HOURS, 2018 VS. 2022 (PROJECTED) - [SOURCE: FUTURE OF JOBS SURVEY 2018, WORLD ECONOMIC FORUM] 73

FIGURE 6-2 - A SYNERGETIC RELATIONSHIP OF HUMANS AND AI TO TACKLE COMPLEX DECISION-MAKING SITUATIONS DISTINGUISHED BY THREE MAIN ASPECTS: THE UNCERTAINTY, THE COMPLEXITY AND THE EQUIVOCALITY (JARRAHI, 2018; PAGE 7). 75

List of Tables

TABLE 3-1 - NETWORK EMBEDDING ML ALGORITHMS 40

TABLE 3-2 – GRAPH NEURAL NETWORK ALGORITHMS 43

TABLE 4-1 - LIST OF OPEN-SOURCE TOOLS THAT CAN BE USED BY XMANAI 59

TABLE 5-1 - LIST AND DESCRIPTION OF THE NEWEST AI MANUFACTURING PROJECTS 66

1 Introduction

The main purpose of Chapter 1 is to provide a brief overview of the deliverable, introducing the purpose of the document, its main content and possible dependencies with other XMANAI tasks and activities.

1.1 XMANAI Project Overview

Despite the indisputable benefits that Artificial Intelligence (AI) can bring in society and in any industrial activity, humans typically have little insight about AI itself and even less concerning the knowledge on how AI systems make any decisions or predictions due to the so-called “black-box effect”. Many of the machine learning/deep learning algorithms are opaque and not possible to be examined after their execution to understand how and why a decision has been made. In this context, to increase trust in AI systems, XMANAI aims at rendering humans (especially business experts from the manufacturing domain) capable of fully understanding how decisions have been reached and what has influenced them.

Building on the latest AI advancements and technological breakthroughs, XMANAI shall focus its research activities on Explainable AI (XAI) in order to make the AI models, step-by-step understandable and actionable at multiple layers (data-model-results). The project will deliver "glass box" AI models that are explainable to a "human-in-the-loop", without greatly sacrificing AI performance. With appropriate methods and techniques to overcome data scientists’ pains such as lifecycle management, security and trusted sharing of complex AI assets (including data and AI models), XMANAI provides the tools to navigate the AI’s “transparency paradox” and therefore:

(a) accelerates business adoption addressing the problematic that “if manufacturers do not understand why/how a decision/prediction is reached, they will not adopt or enforce it”, and
(b) fosters improved human/machine intelligence collaboration in manufacturing decision making, while ensuring regulatory compliance.

XMANAI aims to design, develop and deploy a **novel Explainable AI Platform** powered by explainable AI models that inspire trust, augment human cognition and solve concrete manufacturing problems with value-based explanations. Adopting the mentality that “AI systems should think like humans, act like humans, think rationally, and act rationally”, a catalogue of **hybrid and graph AI models** is built, fine-tuned and validated in XMANAI at 2 levels: (i) baseline AI models that will be reusable to address any manufacturing problem, and (ii) trained AI models that have been fine-tuned for the different problems that the XMANAI demonstrators’ target. A bundle of **innovative manufacturing applications and services** are also built on top of the XMANAI Explainable AI Platform, leveraging the XMANAI catalogue of baseline and trained AI models.

XMANAI will validate its AI platform, its catalogue of hybrid and graph AI models and its manufacturing apps in **4 realistic, exemplary manufacturing demonstrators** with high impact in: (a) optimizing performance and manufacturing products’ and processes’ quality, (b) accurately forecasting product demand, (c) production optimization and predictive maintenance, and (d) enabling agile planning processes. Through a scalable approach towards Explainable and Trustful AI as dictated and supported in XMANAI, manufacturers will be able to develop a robust AI capability that is less artificial and more intelligent at human and corporate levels in a win-win manner.

1.2 Deliverable Purpose and Scope

This deliverable aims at providing an overview of Artificial Intelligence domain with a specific focus on Explainable Artificial Intelligence (XAI), considering existing methods and algorithms conceived to

explain the AI outcomes. The goal is to provide a guide to drive data scientists and developers throughout the current landscape of solutions, highlighting advantages and disadvantages of each one, to be of inspiration in order to eventually find the best option that matches technical and business requirements.

Hence, the document is not conceived as a report of activities run from M1 to M6 by WP1, but it is structured as a key reference point to be used in all the XMANAI WPs.

Namely, D1.1 is the result of research conducted by "Task 1.1 - Explainable AI and Graph Machine Learning Analytics State of Play" and "Task 1.2 - Human Aspects in Decision Making and AI", investigating the state-of-the-art of AI domain, exploring methods, requirements and implication of AI adoption. To be more precise:

- Task 1.1 analyzed existing methods, components and tools from a technical point of view
- Task 1.2 focused on various aspects of human thinking, such as decision making, human-machine relationship, AI ethics issues.

The deliverable's purpose is not to teach how an AI solution works but to present features and domains of application, so it is not written from a didactic point of view, but it requires pre-existing knowledge about the topic.

1.3 Impact and Target Audiences

The deliverable is mainly addressed to data scientists and AI developers who are in charge of defining requirements and implementing the XMANAI platform, choosing the most suitable solution according to different options presented in D1.1 and taking into account some initial users' expectations collected in Task 1.2.

As mentioned, Sections 2, 3 and 4 are developed from a technical point of view, so a certain technical background is required to properly understand them. Conversely, Sections 5 and 6 are presented in a more informative way and are more easily understandable also by people without an analytical acquaintance or data science background.

1.4 Deliverable Methodology

Information reported in the deliverable are derived by different sources:

- Literature, to explore the state-of-the-art of AI methods and algorithms. The list of books and papers mentioned in the deliverable is available in the "References" section;
- CORDIS website[1], to extract information about European projects dealing with the same topic;
- Stakeholder interviews, to validate theoretical assumptions and to collect requirements and expectations from the final users of the XMANAI platform. The interview's questions are reported in the Annex.

1.5 Dependencies in XMANAI and Supporting Documents

Since D1.1 aims to be a guide to be taken into account for developing the XMANAI concept and platform, the deliverable is expected to be of support for activities performed in "Task 1.3 - Platform Requirements Elicitation, Data Acquisition and AI Scenarios", "Task 1.4 - XMANAI Concept Elaboration, MVP Definition and Validation", "WP2 – Industrial Asset Management and Secure Asset Sharing Bundles", "WP3 - Core Artificial Intelligence Bundles for Algorithm Lifecycle Management", "WP4 – Novel Artificial Intelligence Algorithms for Industrial Data Insights Generation" and "WP6 - Demonstrator Setup, Operation and Business Value Exploration".

[1] https://cordis.europa.eu/projects/it

1.6 Document Structure

D1.1 consists of 5 sections, besides the Introduction (current Section 1).

Section 2 describes the background and motivations that boost the research in XAI domain and represent the basis for the XMANAI project. Namely, the increasing amount of data collected and the growing availability of computational resources are giving a boost to Artificial Intelligence models, which are required more and more to discover hidden patterns and to provide decision support to humans. AI is turning to be fundamental to augment human cognition and to enhance and speed up data analysis. Anyway, a larger adoption of AI solutions to perform tasks that so far were delegated to humans requires trustworthiness; and trustworthiness means to understand what there's behind the model.

This is why the subject of "explainable AI" is getting more and more of interest and literature about the topic provides several methods.

Our analysis of the state-of-the-art of XAI starts in **Section 3** from the description of transparent models, explainable by design. It means that they don't need further explanations because they are prone to be easily understood by the final user. Linear models and decision tree models are typical transparent models that are commonly in use. On one side these models provide transparency and easy understandability, but on the other hand, either they require a strong hypothesis that can't always be fulfilled (as linear models) or they are too simple to describe complex situations.

"Explainability by design" is often replaced with "explanation methods", that is, techniques that help to give an interpretation of the AI outcome (typically the result of a "black box" model). Hence, it means that they are applied on top of complex machine learning models and, according to their adaptability, they are classified as model agnostic and model non-agnostic techniques.

- Model agnostic techniques: they don't depend on the model they explain but are suitable to be applied to different ones. Of course, it means great flexibility and no restrictions in their choice, but on the other side, using model agnostic techniques, accuracy decreases. The result and the analysis are more superficial and generic, as they are not techniques created especially for a specific AI model.
- Model non-agnostic techniques: they are tailored on the specific AI model they explain. Of course, they have the fundamental advantage of exploiting the internal structure of the model under examination, but in contrast they are weak in flexibility.

Several examples for each category are provided, highlighting advantages and disadvantages in the adoption of one instead of another.

Due to their explainability advantages, a specific focus is provided also to Graph Machine Learning (ML) techniques used to explore and understand data shaped in a graph structure. Indeed, graph representations represent a very powerful and flexible way to visually represent a data network and graphs are frequently used in manufacturing to describe pathways among sensors and devices, to depict associations between resources, workload and production. Graph representation fits well with the research of latent features, useful for analysis or prediction purposes, and this task is often enhanced by the use of ML techniques. Graph ML (including knowledge graphs) extracts useful knowledge from relationships and structures, exploiting machine learning methods; graph representation learning techniques are applied to find a mapping of the discrete graph onto the continuous domain that algorithms function; graph embeddings transform the graph structure to allow graph ML models (usually graph neural networks) to perform further analysis.

The overview of XAI methods ends with the description of hybrid models combining black-box and transparent components, to reach a trade-off between explainability and performances. Several different models are presented in order to provide a full picture of solutions and their domain of application.

In addition, to provide a complete picture of the state-of-the-art of explainability techniques, it is of fundamental importance to identify already existing tools that support their implementation. In **Section 4**, a list of available solutions has been collected, with a specific focus on Open-Source tools, highlighting pros and cons, such as easiness in development or lack of documentation.

This highly technical analysis is complemented in **Section 5** with discussions about XAI applications in industry and in manufacturing and an overview about human aspects in decision making and AI. Exploring already existing implementations of explainable AI represents, indeed, a stimulating starting point to understand what kinds of concepts relevant to XMANAI have been developed so far: the objective is to overcome the purely theoretical analysis with a more practical approach and to understand if expected results are indeed achievable.

Furthermore, to finalize the research, an analysis about the human aspects in AI decision making is provided in **Section 6**. The objective is to understand, with the support of pilots' direct experience, how artificial intelligence may impact the decision making process and which is final user's expectation on AI solutions. Namely, four main topics are addressed:

- Decision making process, to identify how it works and which are the internal and external factors that influence decisions, with a specific focus on manufacturing domain.
- AI in decision making from a user perspective, to understand the approach, fears and level of trustworthiness of the end user toward AI models.
- Industry 5.0, to present a new concept of human and machine relationship, strongly based on collaboration between workers and artificial intelligence, where the latter must be conceived to enhance human capabilities and not to replace them.
- Ethics issues in AI adoption, to prevent or reduce potential risks deriving from its implementation and development.

2 Background

Artificial Intelligence (AI) systems and Machine Learning (ML) have become ubiquitous in many scientific fields during the last decade due to their success in solving many difficult tasks. AI models' rising success has two primary causes: the massive increase of available data and the growing availability of computational resources, especially with the gains obtained through GPU cards. In general, predictions show that this interest in AI systems will keep rising in the following years.

There are, though, some critical open challenges for AI technologies. For ML models to solve more difficult problems, they tend to become more complex and, hence, less understandable by human beings. For example, even for a simple image prediction task with a Deep Neural Network (DNN), it is impossible to automatically understand the underlying information the DNN is extracting to achieve an accurate prediction. The rising complexity of typical ML models increases the demand for supporting them with accurate explainability techniques, which is the explainable AI domain's basic responsibility.

The field of explainable AI fills the gap between the models' complexity and the models' interpretability. Explainable AI provides insights into how a complex and non-interpretable (black-box) model performs at a specific task, i.e. what type of information extracts from the input to produce the output. Explainable AI is challenging since interpreting a complex model's complicated computations and presenting them in a human-understandable structure is an intrinsically challenging task.

2.1 XAI concepts & objectives

Being able to explain how a model works is important from many different perspectives. The answer differs according to the audience that expresses the explainability concern. For example, if an insurance agent asks a model that outputs a trustability score, they want explainability for being convinced that a model is truly based on sensible information for producing the score (trustability concern). If a regulatory entity questions a classification model, the explainability concern is for certifying the model's compliance with the legislation in force. If a worried citizen questions the fairness of an automatic insurance acceptance ML system, explainability aims at answering whether a model is fair or not. If a data scientist or an ML engineer explains a model they just trained, they want explainability in order to understand the model's weaknesses and debug it. In all the aforementioned examples, we provide different perspectives on the importance of transforming a non-interpretable (black-box) model to an interpretable one (white-box).

2.2 XAI methods

There are many different approaches to classifying explainability techniques. In the present analysis, we decide to follow two basic guidelines that classify explainability techniques based on (a) "to which ML models they can be applied" and (b) "what type of information they output".

The first criterion, which is "to which AI models a specific explainability technique can be applied", dictates the structure of Section 3. Section 3 distinguishes explainability techniques into five basic categories: explainable by design, post-hoc model-agnostic techniques, post-hoc model-specific techniques, graph ML models, and hybrid models.

Explainable by design refers to models that are simple enough for understanding them without external help (i.e. an external explainability technique). The post-hoc model-agnostic category contains explainability techniques that can be applied to any underlying ML model without any restriction. The post-hoc model-specific category refers to explainability techniques designed for providing interpretability to a specific model or a specific class of models. For example, the Integrated Gradient (Sundararajan, et al., 2017) approach can be applied only to Neural Networks and, more

specifically, to Convolutional Neural Networks. Graph ML models are a special category since they are applied to a different type of input data. Hence, adding explainability to graph ML models is treated specially. Finally, hybrid models define a special class of explainability techniques where a complex (black-box) and an interpretable one (white-box) are chosen from the beginning, and they are used together during the training and the inference phase.

The second criterion concerns "the outcome of the interpretation method". In Section 3, when analyzing a specific interpretation technique, we provide this information. The different interpretation outcomes can be roughly distributed in the following four categories: feature summary statistic/visualization, model internals, interpretable proxy model and data point. Feature summary statistic refers to the explainability methods that return a piece of information that targets a specific input feature. For example, feature importance returns a number (on a specific scale) that describes how much each input feature has contributed to a specific prediction. In many cases, it is helpful to visualize such information. For example, when the input is an image, it isn't easy to interpret thousands or million numbers (one per input pixel), but it is quite easy to interpret a heatmap. Model internals refers to internal information of the model that helps the interpretation. By default, all intrinsically interpretable models fall in this category. For example, the weights of a linear regression model are the importance of each input feature. There are also black-box models in which some internal information helps understand how they work. For example, reporting the output of a specific layer of a neural network helps to understand what high-level features the particular layer captures. The interpretable proxy model describes the case where we approximate (locally) a black-box model with a simple interpretable model. Finally, the data point is the broad class of explainability techniques where we describe a model by giving examples. This is a method that humans tend to use quite frequently when they try to explain something.

3 XAI Methods and Approaches

This section will present and analyze the SotA methods in the explainable AI domain. Explainability techniques are split into subsections according to the AI models that each technique can explain, as analyzed in the introduction. In Section 3.1, we present the transparent (white-box) models that do not need further explanation. In Section 3.2, we present the model-agnostic explainability techniques, that can be applied to all AI models. In contrast, in Section 3.3, we present the model-specific techniques, which are designed to explain specific models. In Section 3.4, we present explainability techniques for graph ML models. Finally, in 3.5, we present the hybrid models, where a black-box model and a transparent one are coupled at the training and inference phase.

3.1 Explainability by design

Explainable by design or transparent models are those that are interpretable by themselves. According to (Arrieta, et al., 2020), a model that aspires to enter this category must have three properties: simulatability, decomposability and algorithmic transparency.

- **Simulatability** judges whether a models prediction procedure can be simulated by a human without the help of any external machine. This property is directly related to the complexity of the model, since a model with many complex steps for producing its output (i.e. a deep neural network) is impossible to be reproduced by a human.
- **Decomposability** concerns the ability to explain all model's parts, i.e. the parameters, the operation etc. A human must understand each part without the need for any external explanation tool. Decomposability describes a more limited explainability level than simulatability since one may understand each part of the model but cannot simulate its operations.
- **Algorithmic transparency** defines the ability of the user to understand the algorithmic process that the model follows in order to compute the output. The user should be able to describe the process using only mathematical analysis. Continuing the aforementioned structure, Algorithmic transparency define an even more limited explainability level than decomposability. A user may understand how a model operates using external explainability tools (Algorithmic transparency), but maybe unable to explain every individual part of the model (Decomposability).

3.1.1 Linear Models

Linear models compute the output as a linear combination of the input features. Mathematically, linear models are expressed by the following formula:

$$y = b_0 + \sum_{i}^{K} w_i x_i + \varepsilon$$

Where ε represents, with normal Gaussian noise, the difference between the real and the predicted output. Linear models are transparent. The computations for determining the output given the input are transparent and easily simulated by a human. Furthermore, the weights that are learned by the data during the training phase have a very intuitive explanation since they describe how much each input feature contributes in to the output.

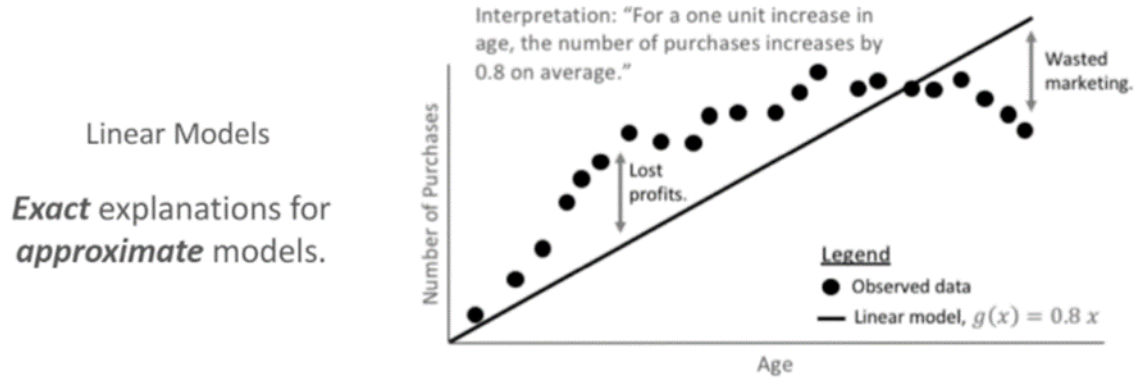

Figure 3-1 Example of explainability in a linear regression model, taken from https://lawtomated.com/explainable-ai-all-you-need-to-know-the-what-how-why-of-explainable-ai/

3.1.2 Generalized Linear Models and Generalized Additive Models

Linear models are heavily constrained by strong assumptions. The output must be a linear combination of the input features and the added noise is restricted to the normal Gaussian distribution. However, in many cases, real world data cannot be modelled under these assumptions. In these cases, linear models are not able to fit. Generalised linear models (GLM) and generalised additive models (GAM) are generalized versions of the simple linear prediction model that can be employed to model more complex input-output relationships.

Generalised linear models (Nelder 1972) go further than linear models, as they can answer non-normal observed variables in a unified way. These models are good for observed data that do not fit normal distributions. Mathematically, GLMs are a modification of the linear model formulation:

$$E_y(y|x) = g\left(b_o + \sum_{i=1}^{K} w_i\, x_i\right)$$

where g is a link function, the summation term is a linear model and E_y is the expected value of a probability distribution of the exponential family. An example of a GLM is the logistic regression function, which is obtained by applying the logit function as a link function and a Bernouilli distribution.

Generalized Additive Models (Hastie 1987) are an extension of GLMs, where the input features are firstly transformed through non-linear functions, before following the computations as in the GLM case. GAMs can be formulated mathematically as follows:

$$E_y(y|x) = g\left(b_o + \sum_{i=1}^{K} w_i\, f_i(x_i)\right)$$

Using the non-linear functions f_i, GAMs are able to capture non-linear relationships between the input and the output, as shown in Figure 3-2.

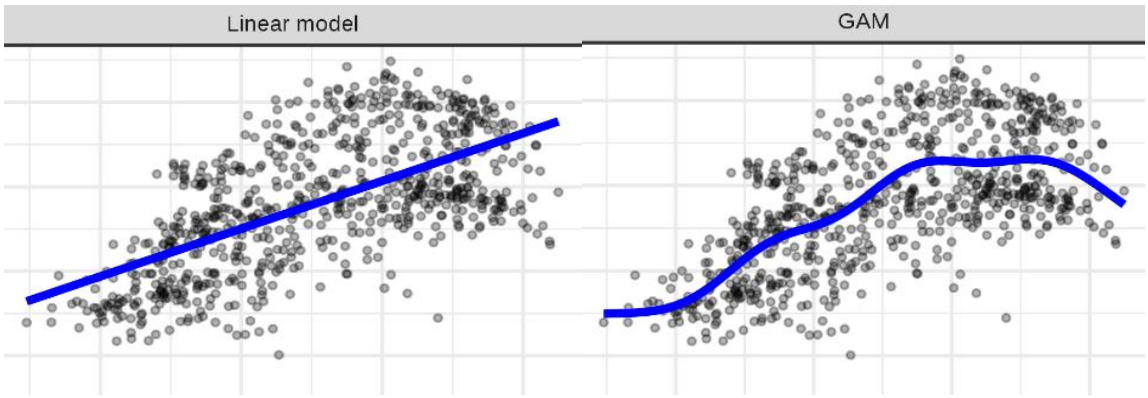

Figure 3-2 - Example illustrating the expressive capability of a simple linear model and a GAM. Image taken from https://christophm.github.io/interpretable-ml-book/

3.1.3 Decision trees

Decision trees are based on the idea that a model can be trained by splitting the data space in a recursive manner and then fitting a simple prediction model at each division obtained. Due to its transparency, the splitting can be represented in a graphical way. Depending on the problem to be addressed, decision trees are classified into two groups: classification trees (output values are assigned to a finite number of unordered values) and regression trees (output values are mapped to discrete ordered or continuous values). There is a variety of decision tree algorithms, based on the treatment of different aspects, such as the splitting type, the number of branches per split or the loss criteria adopted, among others. Among them, the mose influential ones based, on the citation rate, are (Quinlan, 1993), (Kass, 1980), (Loh & Shin, 1997) for classification purposes; (Quinlan, 1992) for regression problems and (Loh, 2002), (Breiman, 1993) for both issues.

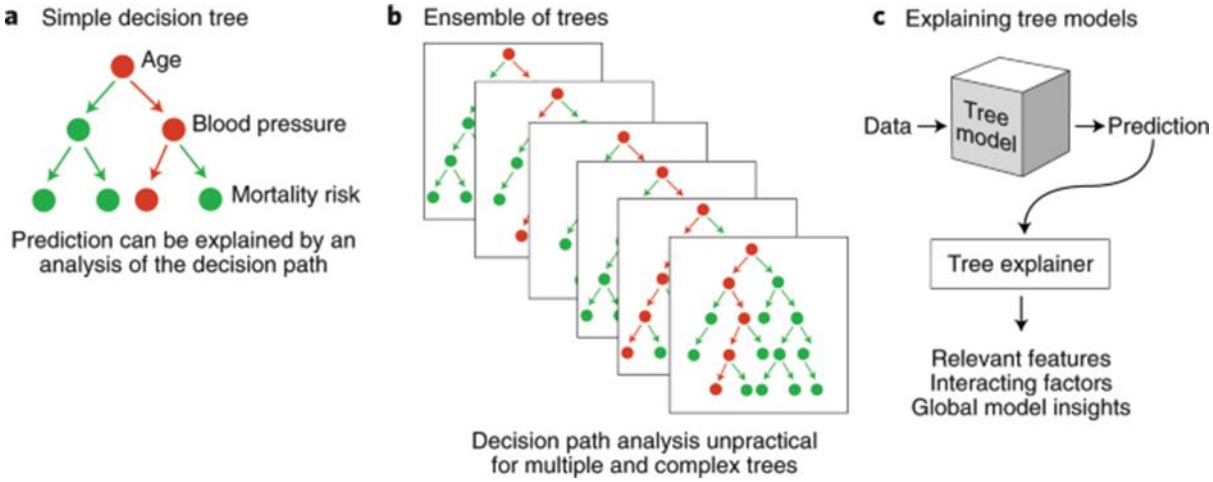

Figure 3-3-- Explainability of Decision Trees and derivate predictive models, taken from https://doi.org/10.1038/s42256-019-0142-0

3.1.4 Rule-Based models

Rule-based models identify a set of relational rules that can lead on a decision or a prediction based on the available data. These rules are of the type of an *if-then* rule or a combination of several rules of the same type. However, the greater the number of rules is, the greater the model complexity is. An evolution of these models are fuzzy rule-based models that use fuzzy rules and increase the interpretability of the system. Rule-based models, due to their transparency level, are also commonly used to explain complex models,such as Support Vector Machines (Nunez 2006).

3.1.5 Bayesian models

Bayesian modelling aims to establish a probabilistic connection between the features and the output. Usually, Bayes models derive the posterior probability as a consequence of two antecedents: a prior probability and a likelihood function derived from a statistical model of the observed data. With the help of a directed acyclic graphical model, it represents the probabilistic relationship between inputs and outputs and allows to analyse which variables contribute more to a specific outcome. These visual contributions of the input variables show that these models are inherently transparent. Bayesian models have been widely used in various fields, for example in climate forecasting (Min 2007), gaming (Synnaeve 2011) or econometrics (Koop 2007). Besides these applications, Bayesian models have also been used to explain other more complex models.

3.1.6 k-Nearest Neighbors

The idea behind k-Nearest Neighbours classification approaches is that a new data sample is classified as class C, according to the most repeated class of its k nearest neighbours. If the model addresses a regression problem, instead of voting and assigning the most frequent class, its neighbours aggregate their target values (e.g., by an averaging or summation function) to infer the new result. Thus, these models rely on the metric to calculate the distance between the new sample and its neighbours as well as the extracted feature space. Due to their interpretability, kNN models are used in a variety of areas, such as prediction of economic events (Imandoust 2013) or text categorization (Jiang 2012).

3.1.7 Advantages & Limitations

The high level of explainability provided by these models is their main advantage, allowing humans to understand predictions without the use of any post-hoc explainability techniques. This is very useful in a large number of different areas such as finance or healthcare. Despite the high level of transparency, there are several limitations that need to be taken into account before using them. The most important is the fact that, due to their simplicity, these models fail when dealing with complex problems and this is the main reason why black box models are widely used in such cases. Therefore, it is important to provide certain levels of transparency, but there is a trade-off between these levels and accuracy.

3.2 Post-hoc Explainability techniques: Model-agnostic

Model-agnostic methods refer to explainability techniques that separate the explanations from the machine learning models. As some machine learning models are not interpretable, model-agnostic methods can be applied in order to draw conclusions about their predictions. The main advantage of these interpretation methods is their flexibility, since they can be applied to any machine learning model (Ribeiro, Singh & Guestrin, 2016). Further to that, as there are several model-agnostic methods, they provide: (a) explanation flexibility, which allows machine learning developers to apply interpretation methods suitable for the particular task they are trying to solve; (b) representation flexibility, as they are able to use feature representations that are different from the model being explained.

In general, there are three main categories of model-agnostic methods that are used for explainability of machine learning models, which are: model simplification, feature relevance estimation, and visualization techniques (Barredo, Rodriguez et.al, 2019). The focus of the following paragraphs is on explanations by model simplifications, both globally and locally, and almost all of these techniques are based on rule extraction methods.

3.2.1 Explanation by simplification

Explanation by simplification is the interpretability approach where we employ a surrogate interpretable model in order to approximate the predictions of an initial black box model. Since the

surrogate model is explainable by design, we use it as a proxy in order to understand the behaviour of the initial model. The approximation can be either global, where the surrogate model approximates the initial one in the whole dataset, or local where we train the surrogate to mimic the black-box model at a specific example.

Global Surrogate Models

Global surrogate models are used to draw conclusions about the predictions of a black box model. In essence, they are interpretable models that are trained to approximate the predictions of a black box model as accurately as possible, but without requiring any information about its inner workings. Some interpretable models that are used to create surrogate models are linear regression, logistic regression, Lasso, Decision Tree, K-Nearest Neighbors and Naive Bayes Classifiers.

Since surrogate models only need the data and a prediction function (Molnar, 2021), they can be used to explain any machine learning model. The first step is to select the data, which can be either the same dataset that was used to train the black box model, a dataset with the same distribution or even a subset of the data. Then, the predictions of the black box model are used as labels for training the surrogate model. After selecting the type of the interpretable model (e.g. decision tree), the model is trained using the chosen dataset. A measurement score is then used to see how well the surrogate model replicates the predictions of the black box model. To this direction, the most common measure used to calculate how well the surrogate models replicates the black box model is the R-squared (Molnar, 2021).

In this context, G-REX (standing for Genetic-Rule Extraction) is a versatile data mining framework based on Genetic Programming (GP), that can be used to globally explain black box machine learning models by providing extracted rules in a tree-based format. Essentially G-REX uses the predictions of another black-box model as the target variable, for rule extraction. The rules extracted from G-REX explain the relationship between the input and output variables found by the black-box model, and not the topology of the black-box model (Johansson, Niklasson, Koning, 2004).

Local Surrogate Models

Local surrogate models are used to draw conclusions about individual predictions of a black box model. As the black box model's behaviour might be very complex globally, it is much easier to approximate its behaviour around the vicinity of a chosen instance. Therefore, instead of training a model to approximate all the predictions of the black box model, like in global surrogate models, local surrogate models focus on explaining individual predictions. The accuracy of how well a local surrogate model explains the predictions of the black box machine learning model is called fidelity.

Local surrogate models with interpretability constraint can be expressed with the following mathematical equation:

$$explanation(x) = argmin_{g \in G}(L(f, g, \pi_x) + \Omega(g))$$

where g is the interpretable model (from the family of possible interpretable models denoted as G) that minimizes loss L. Loss L expresses how well the interpretable model g approximates the explanation of the original black box model f. The term $\Omega(g)$ computes the surrogate model's complexity. Finally, π_x is the proximity measure that defines the distance between the chosen instance x and the selected samples. The proximity measure, which is selected by the user, specifies how many training examples, near x, will be used for computing the loss term L.

A concrete implementation of local surrogate models is found in Local interpretable model-agnostic explanations (LIME) (Ribeiro, Singh & Guestrin, 2016). LIME's goal is to test what happens to the

predictions of any machine learning model, when training data change. As a first step, the instance for which we search for an explanation (instance of interest) is chosen. LIME then generates a new dataset consisting of permuted samples along with their correlated predictions of the black box model. For text and image datasets, LIME generates the new dataset by turning on or off single words or pixels; whereas, for tabular data, LIME generates the new dataset by perturbing each feature individually based on some noise distribution (e.g. a normal distribution). In order for LIME to define the new dataset (which is also called neighborhood) around the point of interest, LIME uses an exponential smoothing kernel. A smoothing kernel is a function that has as input two data instances and as an output it yields a proximity measure. Moving on, the new data samples are assigned a weight based on their proximity to the chosen instance of interest (the closer the sample to the chosen instance, the higher the weight). Finally, an interpretable model is selected and trained on the newly generated dataset along with their corresponding predictions from the black box model. It has to be noted that interpretable models are self-explainable models, such as Decision Trees where you can easily extract decision rules. Similar to global surrogate models, the most common interpretable models to local surrogate are: Linear Regression, Logistic Regression, Decision Trees, Naïve Bayes and K-nearest Neighbors.

A visual example of how LIME operates is presented in Figure 3-4. The decision function of a machine learning black box classification model is displayed in subfigure A. The decision function is displayed by a blue line and it is not linear. The two classes that are separated by the decision function are shown by crosses and circles. As explained above, an instance is chosen where an explanation will be provided by LIME, which is shown as a green circle in subfigure B. LIME then selects samples from the dataset; in particular from mass center of the training data, instead of only around the selected instance (based on normal distribution). This increases the probability that there will be a variation of sample points (not all points will belong to the same class as the chosen instance), and therefore LIME can be properly trained. The selected samples are then assigned a weight based on their proximity to the selected instance, which is shown in subfigure C. At the end, the surrogate model learns the decision function shown by a red dotted line in subfigure D, which can explain locally (around the chosen instance) the behaviour of the black box machine learning model. As displayed, some instances, which are further away from the selected instance, are predicted as a different class using the locally trained model , than what they were originally predicted from the black box model (Figure 3-4). This behaviour occurs as the local surrogate model is only trained to explain the local predictions and not the global ones.

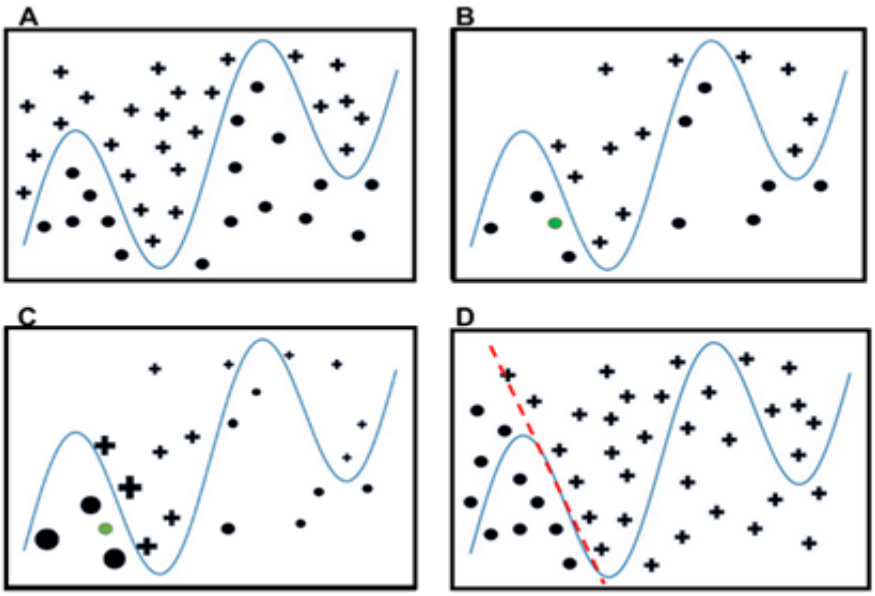

Figure 3-4 - Example illustrating how LIME (Local Interpretable Model-Agnostic Explanations) work

Conclusions

The main advantage of explanation by simplification, either with Local or Global surrogate models, is the unified approach in interpreting all black-box models. We may select a common surrogate model and use it to interpret several different black-box models, allowing easy comparisons between them. Furthermore, there is no restriction in the white-box surrogate model that will be used, hence each user can choose the one they are more comfortable with. On the other hand, the main disadvantage is that in most cases the approximation is not accurate. Therefore, if we attempt to explain a really complex model (e.g. a deep neural network) with a simple surrogate (i.e. linear model), the extracted explanation will be weak because of the inconsistent approximation.

3.2.2 Explanation of feature relevance

Feature relevance is the broad category of explainability methods that try to measure how much each input feature has contributed to the output prediction.

Partial Dependence Plot (PDP)

Partial Dependence Plot (Friedman 2001) aims to analyze the influence of a subset of input variables on the output result of a trained model. It provides insight into how input features affect the final results. For computing the partial dependence, small perturbations are applied in the input feature under examination and the effect on the output prediction is measured. This procedure is repeated in all examples of the training set and the effects are averaged. If the pertrubations are applied in a wide range of values, the Partial Dependence can be visualized in a plot, the so-called Partial Dependence Plot. PDP has been used by a large number of supervised models across different research fields in order to gain a better understanding, including voter mobilisation experiments through Bayesian Additive Regression Trees (Green 2010), forecasting criminal behavior by means Random Forests (Berk 2013) or to understand how different environmental factors contribute to the distribution of a particular freshwater eel via a Stochastic Gradient Impulse model (Elith 2008).

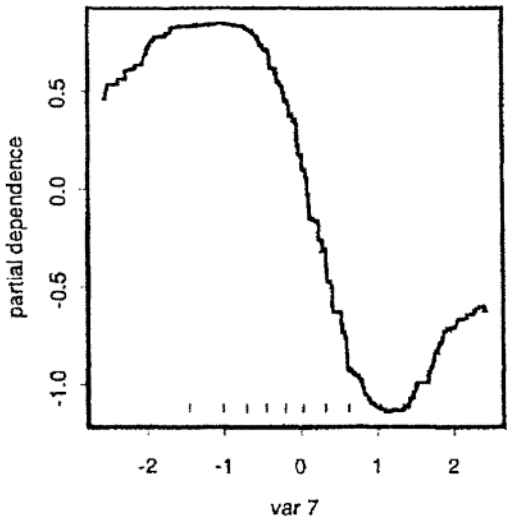

Figure 3-5 - Example of a PDP Curve, as taken from Friedman 2001

Individual Conditional Expectation (ICE)

As mentioned above, Partial Dependence Plots calculate the average of all observed output values for each specific point of the analyzed subset. Performing this averaging may lead to a misleading conclusion. Let us use a simple example. Consider the case where in half examples of the training dataset, a specific feature has positive relation with the ouput i.e. increasing the feature value, increases the output in a linear fashion. In the other half of the dataset, the relation is completely opposite. In this case, the partial dependence plot would show that the specific feature does not influence the output at all (i.e. negative and positive relations would cancel out each other). To address this problem, Individual Conditional Expectation (Goldstein 2015) mitigates this effect by taking into account and plotting each obtained curve for the feature of interest, not the average. As observed at Figure 3-6, the ICE plot reveals that the zero-effect outcome has been produced due to opposite partial dependences between the training set examples.

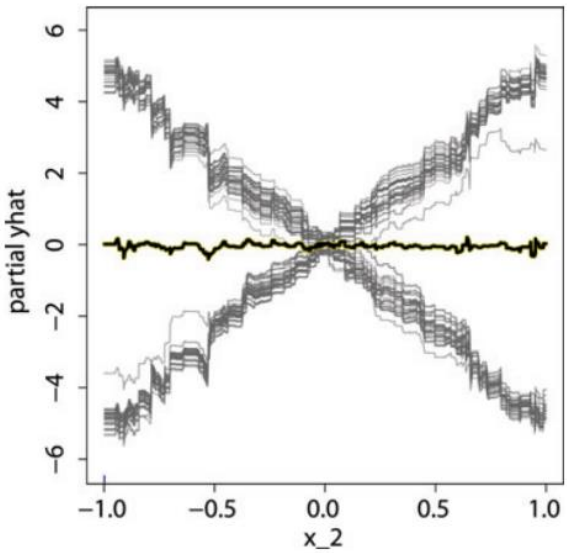

Figure 3-6 - ICE curves and PDP (average of the curves). Image taken from Goldstein 2015

Accumulated Local Effects (ALE)

Partial Dependence Plots are a good approach when the analysis features are not correlated with the other features used by the model to make predictions. When features are correlated, the PDP takes into account all possible values for the features of interest, resulting in unrealistic scenarios, where some values in the analysis grid do not have typical values for the correlated variables. To address this problem, Accumulated Local Effects were introduced by (Apley, 2019). The objective of this algorithm is to calculate the effect of the analysis variables, dividing their possible values into areas to take into account scenarios where the correlated variables have realistic values. For each area, a difference between outputs is obtained by taking the maximum and minimum values of the study variables and calculating their respective outputs. Once this is done, an average of all differences is calculated. The final result is an average difference value for each area, which indicates a higher relevance in those areas with higher values.

Feature Interaction

This type of approach aims to estimate whether there is an interaction between variables as well as a way to measure the strength of this interaction. Friedman 2008 proposes the measurement of the H statistic to address 2 cases: the interaction between two features or the interaction between one feature and the rest of them. The output values are in the range of 0 to 1, with 0 meaning no interaction and 1 meaning that all variables contribute to the final outcome only by interacting with each other. Other approaches that study the feature interaction effect are Variables Interaction Networks (Hooker 2004) or partial dependence based feature interactions (Greenwell 2018).

Permutation Feature Importance

The main objective of Permutation Feature Importance is to study the importance of each feature by analyzing the effect on the prediction error of the model if the values of the feature are permuted by other values of that feature. Breiman 2001 proposed a first version of this method specifically for the random forest model although Fisher 2018 adapted this idea to an agnostic version of the model and called this method model reliance. In this method a permutation of the data is suggested based on splitting the dataset in half and swapping these two halves rather than permuting each feature value with every other possible value in the dataset for that feature in order to not take a large computation time.

Shapley Values

Shapley Values (Shapley, 1953) is based on the idea of cooperative game theory, where features take part of the game forming a coalition in order to contribute to the payout (model prediction). The main objective of this method is to calculate, for each data instance, how each feature contributes to the prediction, compared to the average of all predictions and finally to average these contributions. Due to the number of operations required, estimating Shapley values for more than 2 features becomes unfeasible, as the required number of operations grows exponentially. (Strumbelj, 2014) proposes a method to approximate them by Monte-Carlo sampling.

SHapley Additive exPlanations (SHAP)

SHAP is a modification of the Shapley values proposed by Lundberg 2017. The idea of this method is the same: to explain the contribution of each feature to the prediction. Thus, it is based on cooperative game theory where the features are players forming a coalition and the payout corresponds to the prediction. The main difference between Shapley Values and SHAP is that the latter combines Shapley Values with LIME models. Following this idea, the value contributions are combined by a weighted sum and the explanation of Shapley Values is represented as an additive feature attribution method. Later, a method to explain feature contributions for tree-based methods (such as simple trees, XGBoost, LightGBM among others), was proposed by Lundberg 2018. The figure below shows with an example how this works with a practical case.

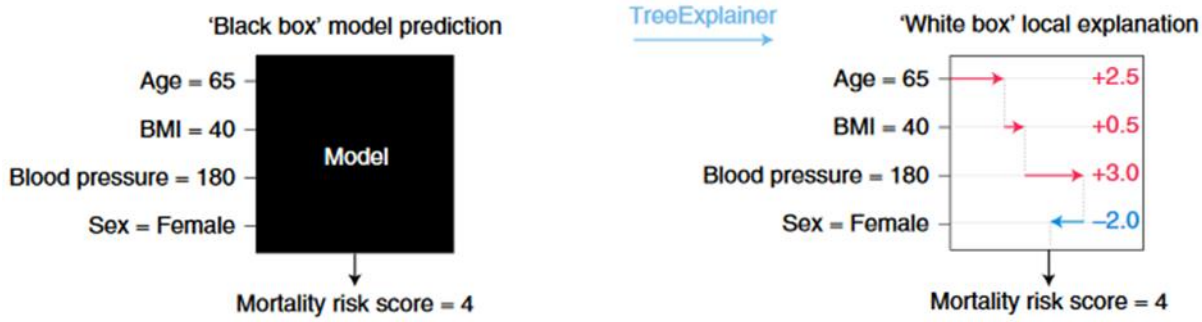

Figure 3-7-- Cartoon ilustration of explanation models with Tree SHAP from the Lundberg 2018.

3.2.3 Explanation by visualization

Visual explanation techniques attempt to provide the end user with a visually conceivable explanation of a black-box model's prediction. Often, visualizations are combined with other techniques to improve their understanding, which are deemed to be more relevant to infer complex interactions among the variables of the model (Arrieta et al., 2020). Generally, visual explanation techniques are considered as a reasonable approach to accomplish model-agnostic explanations of a black-box ML model (Murdoch et al., 2019). In addition, it is important to highlight that the pattern of these visualizations' methods should be applicable to any non-transparent machine learning model focusing on its feature inputs and output pairs and not its inner structure. For this reason, researchers have developed various types of visualizations such as: boxplots, barplots, heatmaps, line charts, scatterplots, lollipop charts, summary plots, etc. which assist to perceive what the trained model has learned (Molnar, 2019). Below[2], we present many visualization approaches[3] that can support the model-agnostic explainability approaches defined in the previous section.

In general, different visualization plots are used depending on how the information can be better described (Nussbaumer Knaflic, 2015). For instance, barplots are useful for comparing the distribution of quantitative values between the observed features. Specifically, a bar chart illustrates a distribution of data points or compares metric values across different features of the data. Accordingly, a lollipop chart illustrates the same information as a bar chart, but with different aesthetics. Particularly, when many features (or categories) participate and their values are close together then it is more convenient to use a lollipop chart. A boxplot presents how the values in the data are spread out and provide additional information about the measures of central tendency (median, mean, and mode). Since boxplots take up less space (comparing to histograms or density plots), they are useful when comparing distributions between many groups or features. Scatter plots are used to observe and show relationships between two numeric features. There are many types of relationships that can be described in many ways: positive or negative, strong or weak, linear or nonlinear. Once multiple dots are plotted, trends can be spotted and samples can be compared, depending on how many colours are featured in the chart. Nevertheless, scatter plots do not only show relationships between two features but also, they present the contribution of the features (Figure 3-13). The colour represents the value of the feature; blue colour indicates low, purple signifies the median and red means high value). Heatmaps are used to show relationships between two features, in which the axis features are divided into ranges. Every colour determines the value of the main feature in the particular cell range. Heatmaps are very useful since they allow you to easily observe the underlying trends without having to deal with numbers or compare metrics. Line charts or line graphs show trends in data over a period

[2] From Figure 3-8 until Figure 3-17
[3] Taken from https://christophm.github.io/interpretable-ml-book/

of time or a particular correlation. Each value is plotted on the chart, then the points are connected to display a trend over the compared time span.

Finally, it is important to highlight that each model-agnostic method can explain a model with different visualization charts; for example, the PDP method explains a model using barplots, line charts and heatmaps (Molnar, 2019).

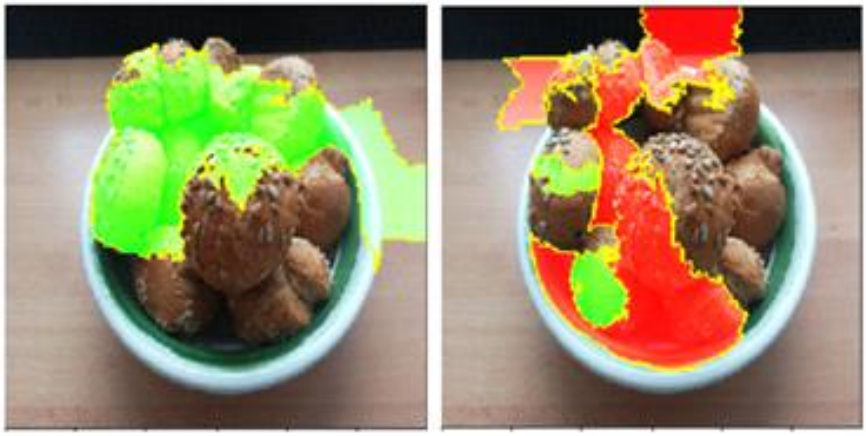

Figure 3-8 - Heatmap explanations for the top 2 predicted classes, bagel (left) and strawberries (rigtht) made by Google's Inception V3 neural network using Local interpretable model-agnostic explanation. The probability of begel was 77% whereas for strawberry was 4%. Green signifies the areas that increased the probability and red the areas that decreased it. We observe that the explanations are very reasonable. The model predicted it is bagel (even though it is not since the hole in the middle is measing) emphasizing in the breads and that it is not strawberries based again in the bread area.

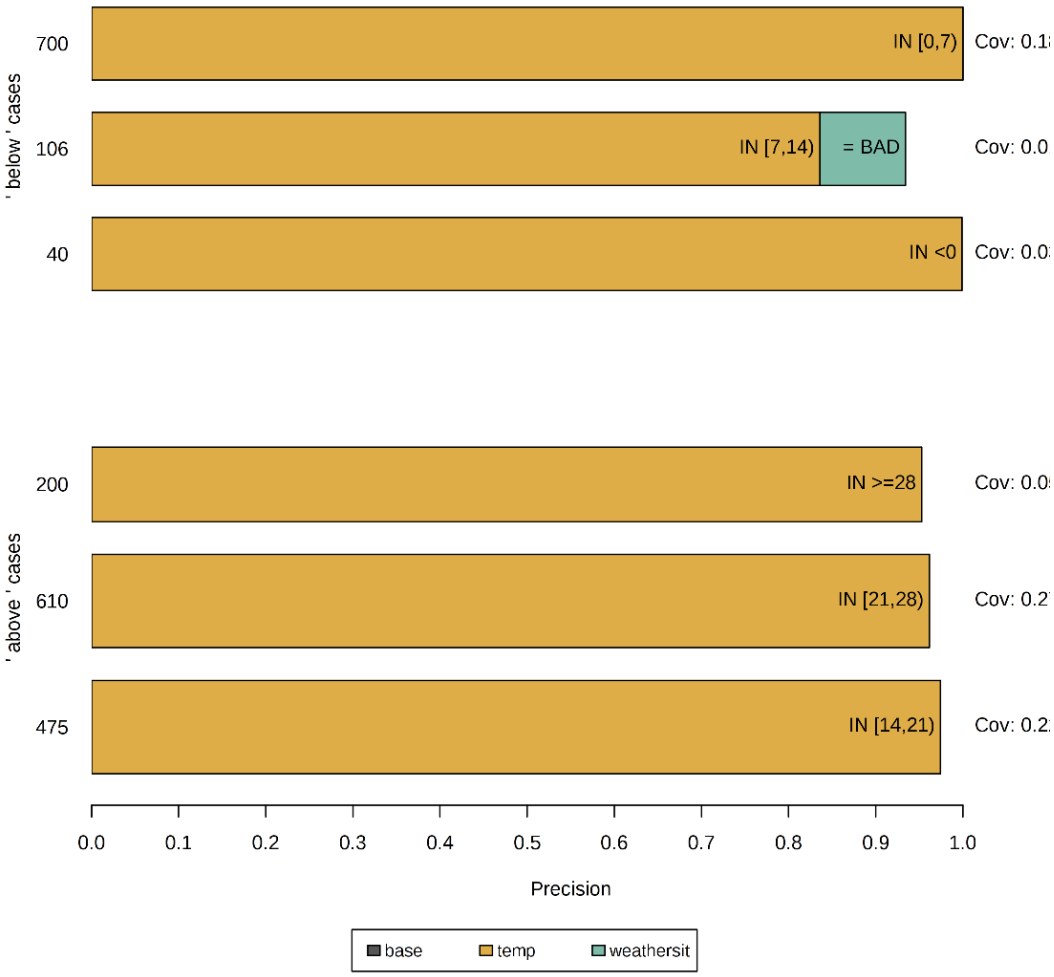

Figure 3-9 - Vertical barplot explanations of six instances of the bike rental dataset using Anchors. Each row represents one explanation or anchor and each bar outlines the feature predicates contained by it. The x-axis displays a rule's precision, and a bar's th bar's thickness corresponds to its coverage

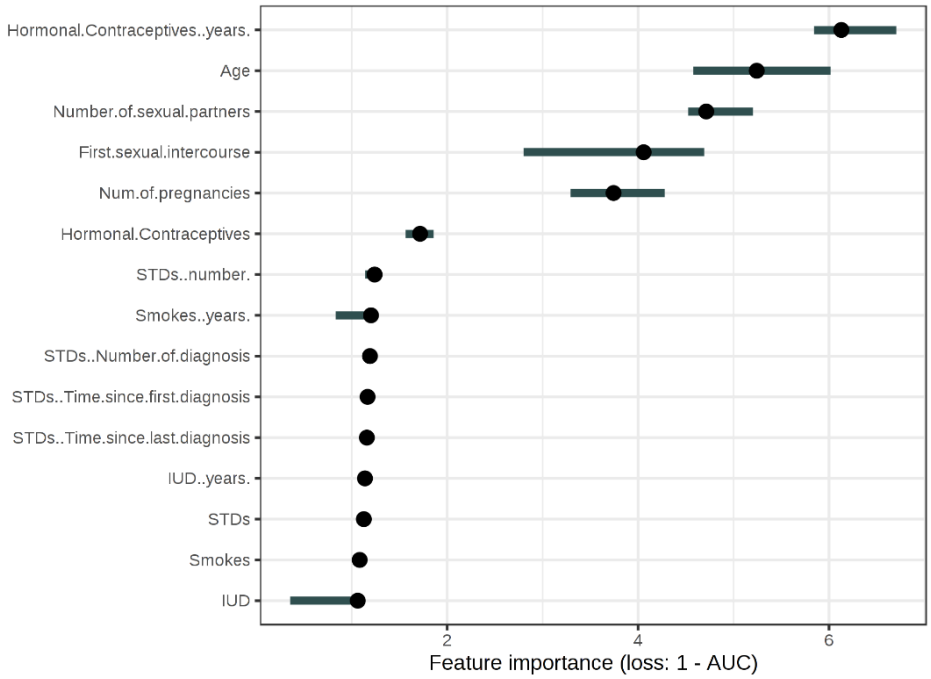

Figure 3-10 - Plot explanations using Permutation Feature Importance indicating the importance of each feature for predicting cervical cancer with a random forest

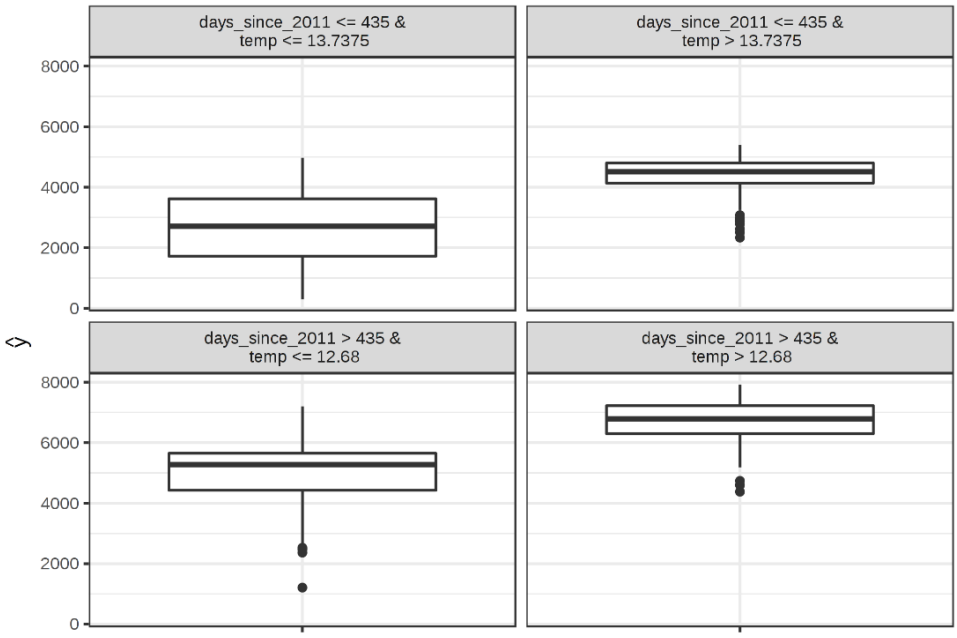

Figure 3-11 - Boxplot explanations using Global Surrogate. The terminal nodes of a surrogate tree that approximates the predictions of a support vector machine trained on the bike rental dataset

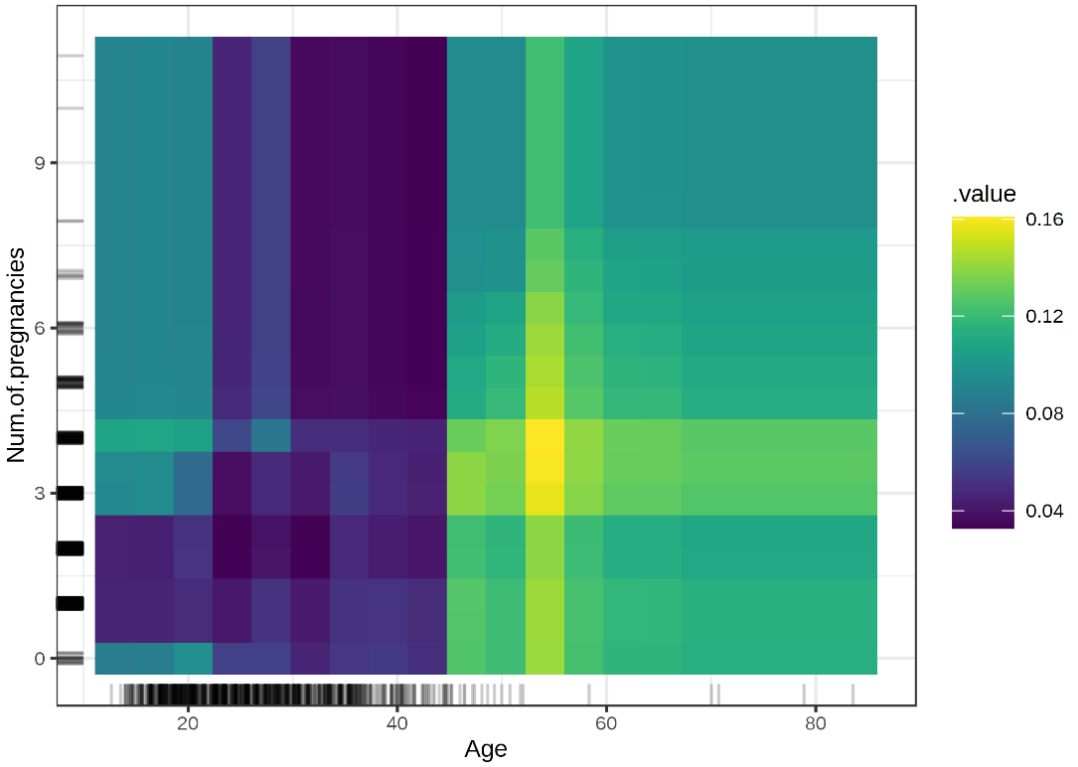

Figure 3-12 - Heatmap matrix explanation using Partial Dependence Plot illustrating the cancer probability and the interaction of two features (age and number of pregnancies)

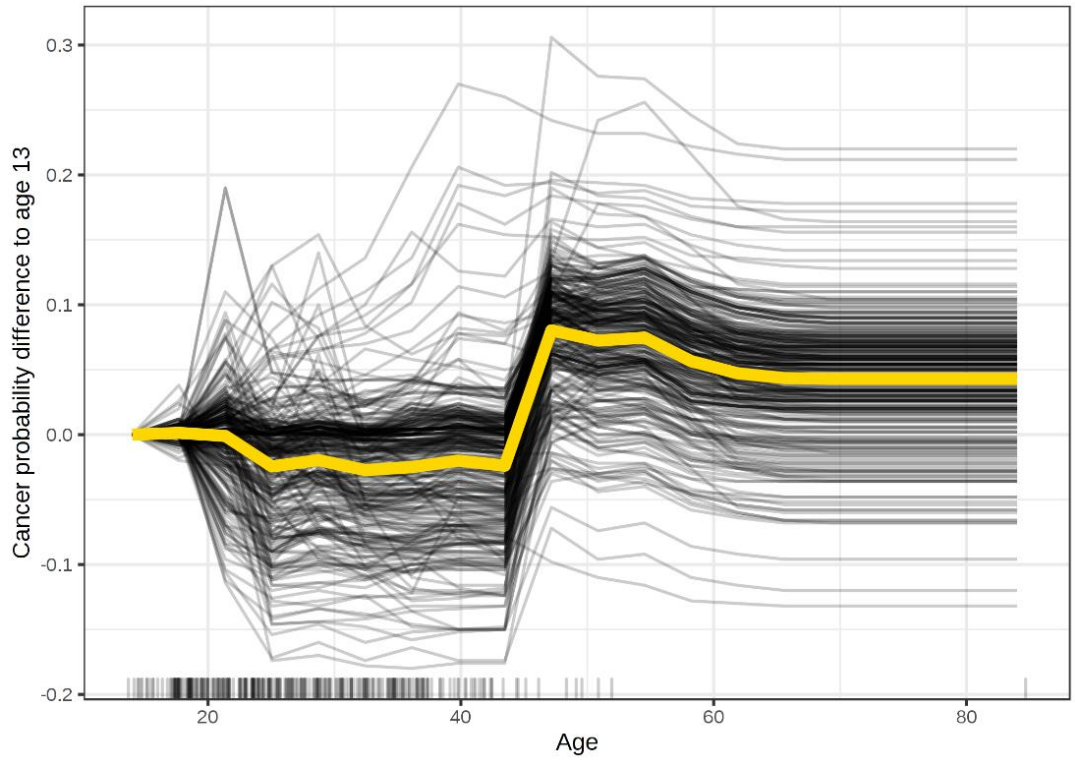

Figure 3-13 - Line chart explanations using Individual Conditional Expectation. Each line represents an observation. The yellow line represents the centered curve at a certain point of the curves in the feature; while the black lines display the difference in the prediction to this point.

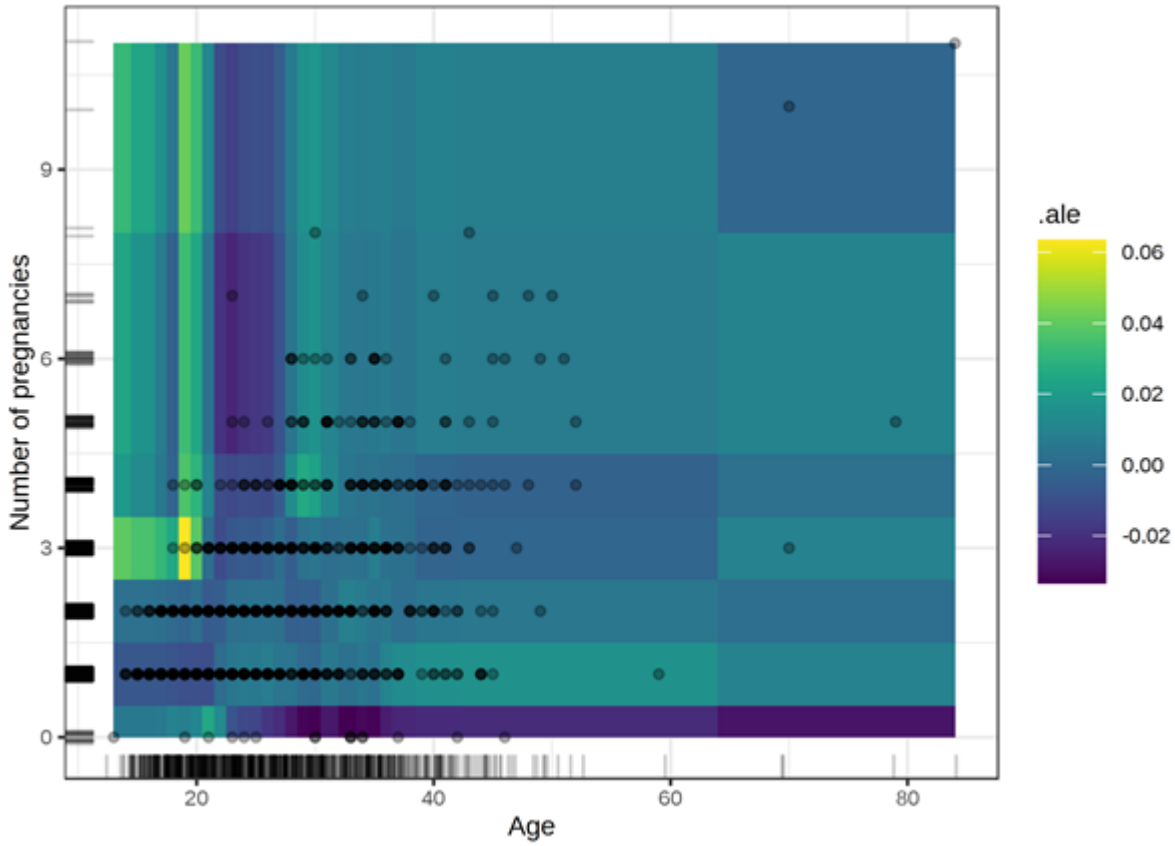

Figure 3-14 - A combination of Scatterplot and Heatmap explanation using Accumulated Local Effects presenting the interaction effect of he 2nd order effect of two Number of pregnancies and Age. For instance, the plot shows an odd model behavior at age of 18-20 and more than 3 pregnancies (up to 5 percentage point increase in cancer probability).

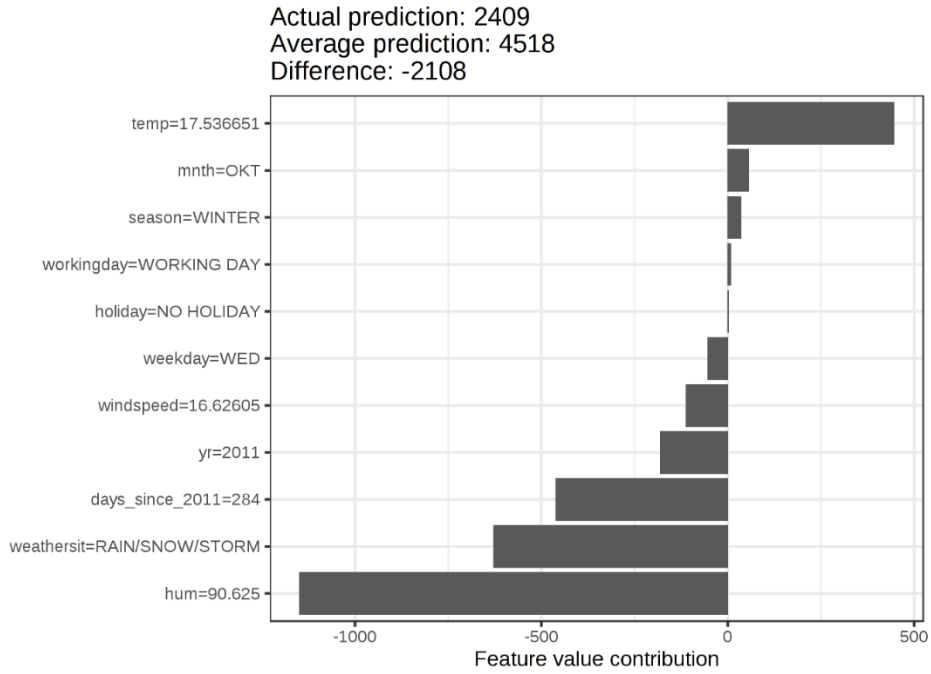

Figure 3-15 - Barplot explanation using Shapley Values presenting the average contribution of a feature value to the prediction in different coalitions.

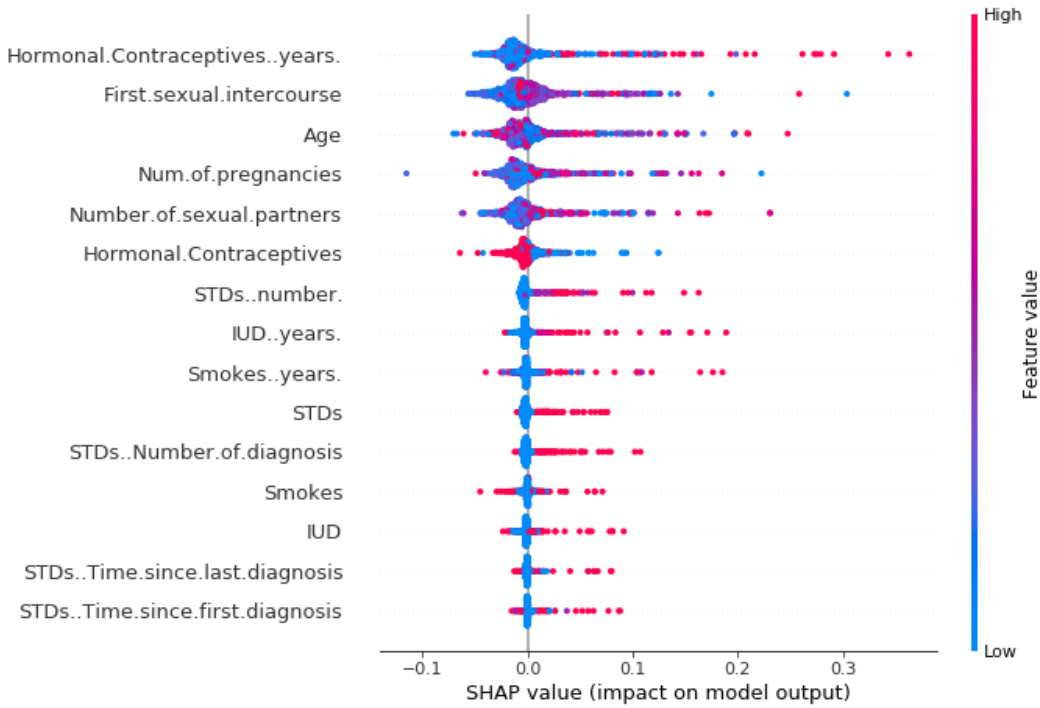

Figure 3-16 - Summary plot explanations using SHapley Additive exPlanations. Each point on the summary plot is a Shapley value for a feature and an instance. The position on the y-axis is determined by the feature and on the x-axis by the Shapley value. The color represents the value of the feature from low to high.

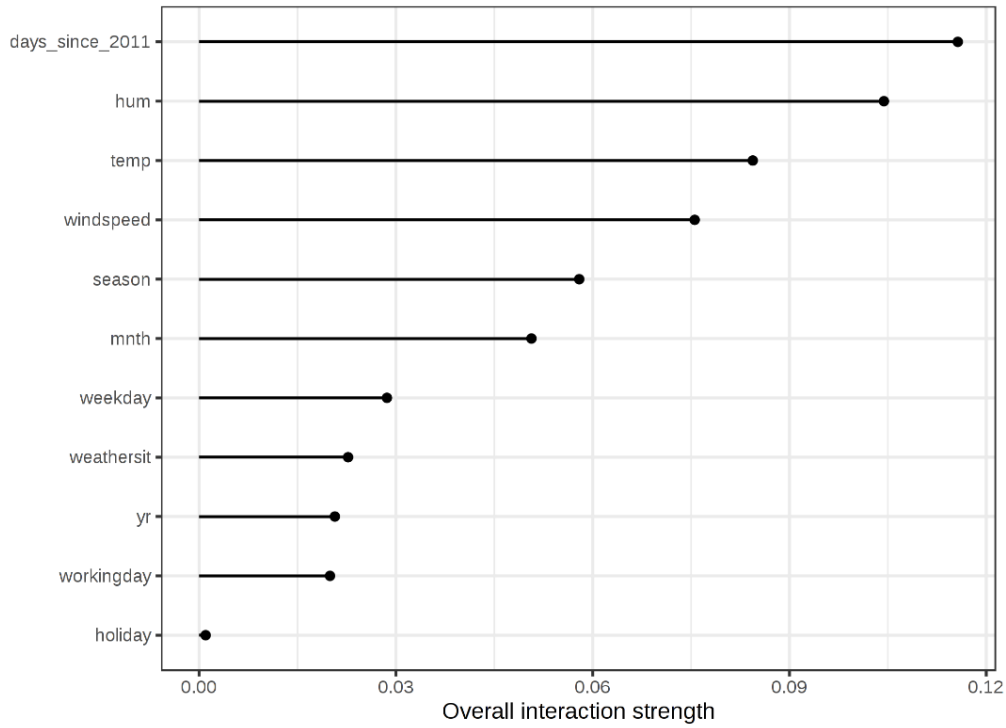

Figure 3-17 - Lollipop chart explanation using Feature Interaction The interaction strength (H-statistic) for each feature with all other features for a support vector machine predicting bicycle rentals.

Graph-Based visual explanations

In the graph, nodes or vertices represent variables/features and edges or connections indicate predictive relationships, which can be undirected drawn as simple lines or directed drawn as arrows (Newman, 2010). Moreover, graph visualization provides an initial qualitative way to understand complex data.

In the estimated graphs, not all nodes are equally important. Different measurements are necessary to infer the most significant nodes in the network (Kolaczyk, 2009). The importance of each node can be assessed by evaluating the node's centrality (Freeman, 1978; Golbeck, 2013) providing additional information about the node. For instance, centrality indices show which node directly influences many other nodes or the node with obtains the maximum number of relationships (degree), which node can reach many nodes via particularly short paths (closeness), and which node controls an especially large number of the shortest connections to an adjacent network (betweenness). Consequently, a graph can be displayed in alternative ways according to the centrality indices by using node size to display visually the significance of each node.

Figure 3-18 presents two simple graph models of five nodes (A, B, C, D and E) and six edges. Circles depict the observed variables and their size depends on the degree-centrality index. Therefore, node A is displayed as the largest node since four edges are attached on it. On the other hand, node E is connected with only one edge which is the minimum number of node connectivity comparing to the rest nodes of the graph and for this reason, it is displayed with the smallest size. Moreover, the size and the colour density at the edges varying according to the polarity and the strength of the relationships between the variables. Typically, positive relationships are coloured blue or green and negative relationships are coloured red. In Figure 3-18 (b) there are four positive connections {A, B},

{A, C}, {A, E} and {B, D} and two negative connections {A, D} and {C, D}. It is also observed that there are no interactions among some pairs of nodes, such as node {B, E}, {B, C}, {C, E} and {D, E}. Thus, the width and saturation of edges indicate the strength of the relationships (i.e., thicker edges represent higher interactions among variables).

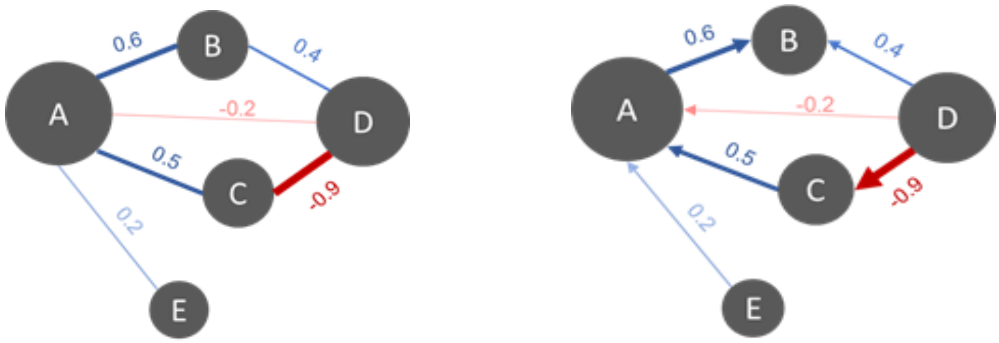

Figure 3-18 - Illustration of (a) an undirected weighted graph and (b) a directed weighted graph. Each edge carries a weight indicating the strength of influence among nodes. Blue edges specify positive relations; while, red edges denote negative relations between nodes. An undirected weighted graph is illustrated on the left and a directed weighted network on the right. The size of each node is set according to the degree-centrality.

As a final point, a graph visualization is a very important procedure since it provides valuable information concerning the visibility of each node in the space, the identification of the most significant and insignificant nodes, the monitoring of how the information flow from one node to another, the observation of potential clusters and finally, the discovery of node's neighbourhood.

3.2.4 Explanation by example

In example-based explanation techniques, we understand how a ML model works by selecting appropriate input examples. The examples can be either artificial or particular instances of the dataset. In general, this category can also contain model-specific methods i.e. a technique for extracting representative examples that is optimized for working with a special method. In order to be consistent with our general classification, we isolate here only the model-agnostic explanation by examples techniques. The model-specific techniques, that output as explanation a specific instance of the dataset, are presented in the next section. We divide the model-agnostic explanation by example techniques in four classes; counterfactual explanations, adversarial examples, protypes/criticism and influential instances.

Counterfactual Explanations

Counterfactual explanations are very intuitive as humans use them to understand real-life procedures. For example, when we want to explain why we failed in some exams (output), we tend to think of what we could have done better (counterfactual explanation) to succeed. Similarly, we use counterfactual examples to explain a black-box model. In more technical terms, a counterfactual explanation is the smallest alteration to a specific input that causes the output to change to a predefined level.

Searching for counterfactual examples can be algorithmically modelled as an optimization problem. In their work, (Wachter, Mittelstadt and Russell, 2017) set the objective function as the sum of two distances following the definition of counterfactuals. The first term ensures that the counterfactual produces the expected outcome, while the second term ensures that the counterfactual is as close as possible to the actual input. Afterwards (Dandl et al., 2020) proposed adding more criteria to the optimization problem. Apart from the two ones used in (Wachter, Mittelstadt and Russell, 2017), this work stated that a counterfactual should also (a) change the fewer possible features and (b) get feature values that are likely to happen. The latter criterion deals with a common weakness of these

explanations, which is proposing a counterfactual that is infeasible. Finally, (Mothilal, Sharma and Tan, 2020) developed the Diverse Counterfactual Explanation (DiCE) method, which exploits gradient-based techniques to solve the optimization problem more efficiently.

Adversarial Examples

Adversarial examples share the same backbone idea with counterfactual explanations. In counterfactuals, we search for a small alteration of the input which changes the output to a predefined level. In adversarial examples, we set this level to be an erroneous production. At a higher level, we can say that adversarial examples[4] explain an AI system by finding its weaknesses, i.e. possible inputs that cause the system to fail.

A simple adversarial setup was proposed by (Szegedy *et al.*, 2014), where they search for the smallest alteration that produces a wrong prediction. The optimization problem is solved using a gradient-based optimization technique. Afterwards, (Su, Vargas and Kouichi, 2017) presented the so-called 1-pixel attack, where instead of altering many pixels (features) with small variations, they search for a single-pixel (feature) perturbation. Furthermore, they solved this optimization problem with a differential evolution algorithm that makes no assumptions about the black-box model[5], making the method entirely model-agnostic. A different research line was adopted by (Brown *et al.*, 2017) and (Athalye *et al.*, 2018). In their works, they try to find a global adversarial patch, i.e. a set of pixel (feature) alterations that provokes a false prediction in (almost) all possible inputs. In this scenario, there is no (hard) constrain for the alteration to be small, but it should work in all cases. Finally, (Papernot *et al.*, 2016) proposed a method for creating adversarial examples even when the underlying black-box model is entirely unknown, i.e. we can access only its predictions through an API. They initially used a surrogate model trained to learn the black-box model's behaviour, and afterwards, they used the surrogate model to produce adversarial candidates.

Prototypes – Criticisms

Prototypes constitute a reduced representation of a whole dataset by a small subset of characteristic samples. The validity/coverage of such a representation, depends on the similarity between the distribution of prototypical instances to the data distribution. Class-specific prototypes can serve as local explanations of trained ML models' predictions. A given prediction can be explained by presenting similar examples drawn from the closest prototypes of the same class. The identification of prototypes can be addressed as a clustering task, assuming k concentrations are formed in the data as in the k-medoids method (Kaufman & Rousseeuw, 1987). The effectiveness of such an approach is limited in case of complex real-life data distributions, also by the fact that the unknown number of prototypes must be defined a priori. The work by (Kim, et al., 2016) set the proper framework to prototype explanations by stressing out the importance to additionally identify criticisms, in order to explain cases that are not represented by prototypical examples (Figure 3-19).

[4] Adversarial examples are widely adopted when the input is an image, but they are not restricted to this case.
[5] For example, there is no need for gradients to be supported.

Figure 3-19 - Prototypes and criticisms for two different dog breeds, learnt by the ImageNet (Kim2016).

Using the Maximum Mean Discrepancy (MMD) to measure the similarity of prototype distribution to that of the data, they develop MMD-critic to select a small set of prototypes that minimize MMD, extracting also criticisms as samples where the two distributions differ the most. The ProtoDASH framework by (Gurumoorthy, et al., 2019) extends MMD-critic to additionally assign importance weights to the extracted prototypes, also to apply on any positive definite similarity kernel. In a recent contribution to example-based explanations, (Van Looveren & Klaise, 2019) found the use of prototypes to assist in recovering counterfactual explanations of enhanced interpretability.

Instance attribution – Influential instances

ML models are data-driven, extracting knowledge from the available data. In this context, instance attribution methods quantify the contribution of individual training samples to the learning process and resulting predictions. Influential instances are training samples with considerable effect on the parameters learnt by an ML model and consequently on the model's predictions. In other words, the model's parameters and predictions would change significantly, if influential training samples were deleted. Training samples with high influence on a given prediction, can serve as example-based explanations for this prediction. Moreover, instance attribution assists the developer to identify mislabeled or corrupted samples, create efficient adversarial attacks, recognize bias in the data and improve the model's robustness by mitigating high dependence on a few samples. Instance attribution is typically assessed through either deletion diagnostics or influence functions.

Deletion diagnostics can be applied by deleting samples one by one and retraining the model (leave-one-out training). Each sample's influence can then be directly evaluated as the differentiation in model's overall test predictions (global influence) or in a particular prediction (local influence) when the sample is not present. This approach is straightforward and truly model-agnostic, although computationally expensive and practically unfeasible for large datasets and/or highly complex models.

Influence functions on the other hand, approximate the expected change in the model when a training sample is infinitesimally upweighted, relative to all other samples (eg (Cook & Weisberg, 1980)). Although these methods do not require retraining the model for each sample, they involve the calculation of 2nd order gradients of the loss function minimized by the ML algorithm, with respect to model parameters. As a result, these are directly applicable to parametric models minimizing twice differentiable and strictly convex losses, such as Logistic Regression, SVM or Neural Networks. Even so, the exact gradient computations can be particularly time-consuming and impractical, especially for deep models that learn a very large number of parameters.

In order to address this issue, (Koh & Liang, 2017) proposed a first order Taylor decomposition to approximate gradient computations. The effectiveness of their method in simulating leave-one-out training is experimentally investigated (Figure 3-20). The approximations are accurate in case the differentiability and convexity assumptions are satisfied (as in linear models). In the opposite case, results are consistent but less accurate for Convolutional Neural Network (CNN), while for Radial Basis Function (RBF) SVM results are inconsistent unless smoothing is applied to the Hinge loss. Point explanations are presented in Figure 3-21, for CNN (Inception v3, bottom row) and SVM with RBF

kernel (smoothing t=0.001, middle row) on a fish-versus-dog classification. The test image of a clown fish (top left) was correctly classified by both models. Fish images (green dots) were mostly helpful for the SVM to classify the image as a fish, while dog images (red dots) negatively affected the prediction. This is not the case for Inception, where the contribution of both classes is comparable, so that the 5th most influential sample was an image of a dog (top right). The differences in top two influential instances for each prediction, reflect differentiations in each model's learning process: Inception learns high-level features characteristic to the clown fish, whereas SVM learns from similarities in raw pixel values.

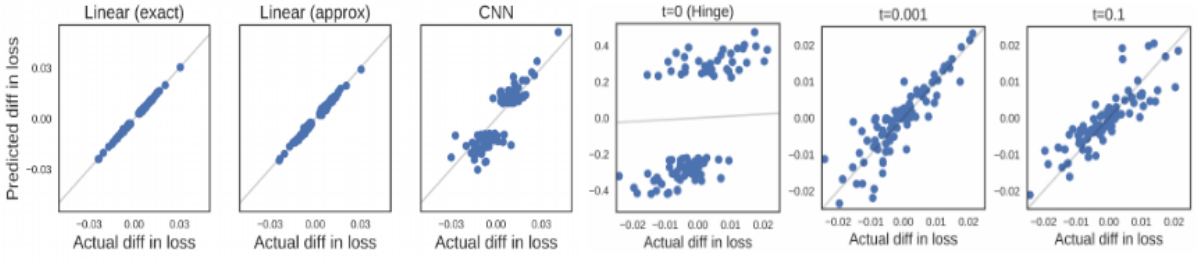

Figure 3-20 - Comparing the actual difference in loss, calculated through leave-one-out training, to the predicted one by Koh & Liang (2017). From left to right, results are presented for Logistic Regression (Linear), Convolutional NN and RBF SVM minimizing the Hinge loss, with various levels of smoothing.

This method was extended to Gradient-Boosted Decision Trees by (Sharchilev, et al., 2018), where a fast computation was developed taking advantage of the internal structure of the ensemble. Additional experiments performed by (Han, et al., 2020), support that the method by (Koh & Liang, 2017) is also applicable to Natural Language Processing (NLP) tasks such as sentiment analysis. However, experimental results by (Basu, et al., 2020) indicate that the consistency of this method on deep learning models degrades with increasing depth and complexity of model architecture and/or volume of training data, also depends on the imposed regularizations.

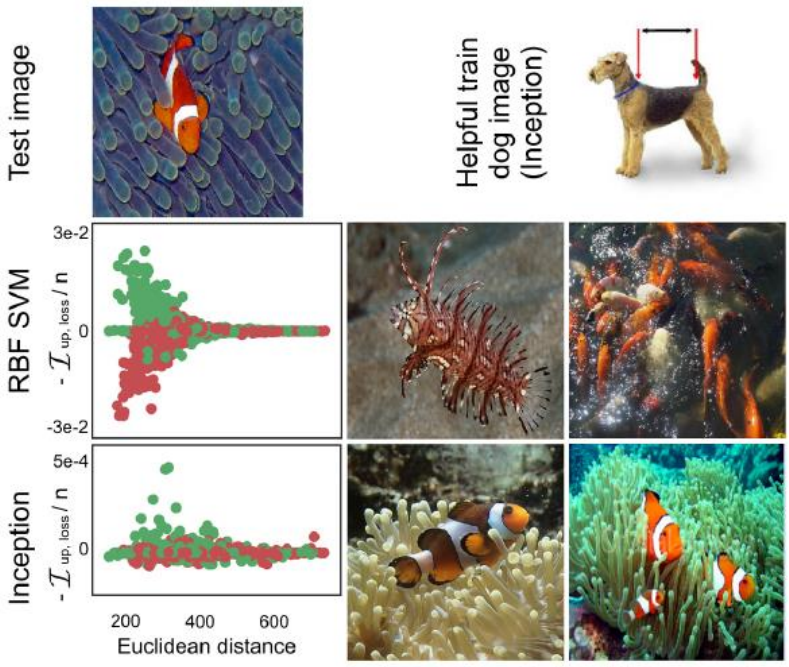

Figure 3-21 - Summary of explanations for a correct prediction of a test fish (top left) by RBF SVM (middle row) and the Inception CNN (bottom row). The most influential instances for each model are indicative to the learning process (Koh & Liang 2017).

Instance attribution based on the Shapley value is almost concurrently proposed by (Ghorbani & Zou, 2019) and (Jia, et al., n.d.). The former, introduce Data Shapley value and develop a truncated monte-carlo approximation (TMC-Shapley), to enhance computational efficiency. The latter develop a series of approximations to Shapley values for instance attribution, providing also an upper bound to the approximation error. Experimental results by (Ghorbani & Zou, 2019) showcase the validity of Shapley instance attribution, in detecting mislabeled (Figure 3-22 - 1st to 3rd panel) and noisy (Figure 3-22 - 4th to last panel) samples.

Figure 3-22 - The TMC-Shapley approximation by Giordani & Zhou (2019) enables debugging the training data. Instances with the least Data Shapley attributions are suspicious as mislabeled samples (1st to 3rd panel), while degradation due to Gaussian noise is detected through diminishing Shapley values (4th to 6th panel).

An alternative approach is proposed in the Representer Point Framework by (Yeh, et al., 2018), to decompose model predictions into representer training point contributions. Although the original method applies to CNN, the idea of breaking down model predictions into training point representations is very promising, so that a modified version for tree-based ensembles has already been developed by (Brophy & Lowd, 2020).

Finally, influential groups of instances are also investigated with the use of influence functions, under either first (Koh, et al., 2019) or second (Basu, et al., 2020) order approximation. Experimental results from both studies justify the ability of influence functions to uncover group effects on model predictions.

3.2.5 Advantages & Limitations

Model-agnostic techniques decouple the explanation technique from the machine learning model (Ribeiro, Singh and Guestrin, 2016). This approach has the great advantage of removing all restrictions when choosing which machine learning model will be used in a specific experiment. Furthermore, designing an accurate and efficient model agnostic method has great impact since its advantages will be exploited by all black-box models. Model-agnostic techniques are also attractive in cases when we want to compare explanations among different black-models i.e. we run multiple models and we want to understand what ever model has learnt. Unavoidably, this flexibility come with a cost. Model-agnostic techniques are not the best choice for explaining a specific black-box model compared to model-specific techniques. This is normal, since model-agnostic techniques do not exploit the specific features of each specific model (architecture, structure etc.). Hence, sometimes they lack in efficiency or accuracy compared to the model-specific alternatives.

3.3 Post-hoc Explainability techniques: Model-specific

Model-specific techniques refer to the methods designed to explain a specific model or a specific category of models. They are based on the particular mathematical/statistical analysis that a specific type of models is built upon to extract meaningful reasoning for its predictions. In some exceptional cases, we can find model-agnostic techniques optimized for a particular category of models, e.g. Feature Relevance for Deep Neural Networks. Such special versions of model-agnostic techniques belong to this category. In general, model-specific methods leverage the internal structure of a specific

family of ML/DL algorithms, intending to enhance the accuracy and quality of the explanations while optimizing computational efficiency. We examine model-specific, post-hoc explainability techniques for famous families of ML and DL models in what follows.

3.3.1 Machine Learning models

We identify two broad families of complex ML models anticipated to hold worth-mentioning solutions for the XMANAI project: tree-based ensembles and Support Vector Machines (SVM). Both types of models have been found to perform very well on several tasks, capable of uncovering non-linear relations in the data by learning highly non-linear decision functions. As a result, the interpretation of such models and their predictions is not straightforward. Therefore, high quality, post-hoc explanations are essential to justify the use of these models within the XMANAI framework. Post-hoc explainability techniques, specific to tree-based ensembles and SVM models, are investigated in the following sections.

Tree ensembles

Ensemble methods are based on the idea that a diverse set of weak base learners can be combined into a single strong learner that outperforms its components. The output of an ensemble model is deduced as the average or the "majority vote" of individual predictions. The application of ensemble methods has found remarkable success in additive models of Decision Trees (Lundberg, et al., 2019), especially under:

- the bootstrap aggregating technique ("bagging"), where each tree is trained on a randomly selected subset of samples and features (e.g. Random Forest, (Brieman, 2001))
- boosting techniques, where trees are created sequentially so that they can learn from previous trees' mistakes by focusing on different sub-regions of the training data where misclassified samples reside (e.g. AdaBoost (Freund & Schapire, 1997), Gradient Boosting Machine (Friedman, 2001), Extreme Gradient Boosting (Chen & Guestrin, 2016).

The improved accuracy and generalization ability, however, come at the cost of model explainability. Although base learners (decision trees) are interpretable by design, ensemble methods usually build a large number of complex trees in order to cover all subregions of the training data, so that the resulting model is considered as “black box”, no longer easy to comprehend. Post-hoc explainability techniques are therefore required, in order to interpret tree ensembles and justify their predictions.

Global explanation by simplification. The goal here is to globally approximate a tree ensemble by a single Decision Tree of reduced complexity, while maintaining the predictive performance of the ensemble. To achieve this, most methods exploit the benefits of finding a global surrogate with the same tree structure as the ensemble’s base components. Pruning can be applied either independently to each tree or to the proposed combination, aka tree-wise and ensemble-wise pruning, respectively.

For example, tree-wise pruning is applied by (Szpunar-Huk, 2006), where a weighted voting system is established to combine pruned base rulesets by regulating the contribution of partially satisfied rules. Also, in (Deng, 2014), the InterpretableTrees framework is introduced with the purpose to construct a Simplified Tree Ensemble Learner (STEL). Rules extracted by base learners are evaluated (by means of rule 'popularity', induced error and complexity), pruning is applied and the final decision ruleset is selected on the base of frequent patterns in variable interactions and complexity reduction.

On the other hand, ensemble-wise pruning is applied, for example, by (Iqbal, 2012), where rule combination and reduction are based on logic minimization, while in (Vandewiele, et al., 2017) the process is performed using a generative algorithm (GENESIS). Exploring solutions for the case of imbalanced data, (Obregon, et al., 2019) propose the RuleCOSI framework, designed to operate on boosting ensembles. Base rulesets are first ordered according to the weights assigned by the ensemble, then sequentially combined and finally pruned by means of the pessimistic error (Quinlan,

1993), proceeding to the next base ruleset in case performance is improved, and discarding the combination otherwise.

More recent studies, though, introduce novel techniques particularly focused in optimizing the size/complexity of the surrogate model. Examples include (a) Born-Again Trees (Vidal, et al., 2020), an algorithm to retrieve a minimum size decision tree from a bagging ensemble by randomly examining splits (instead of sequentially), and (b) Rectified Decision Trees (Bai, et al., 2019), where knowledge distillation has been implemented for tree-based ensembles by using a weighted average of "hard and soft" labels, corresponding respectively to the target labels and the predicted ones.

Finally, a completely diverse to the above methods, Bayesian approach is proposed by (Hara & Hayashi, 2018), where the ensemble is transformed into a probabilistic generative model. Its global surrogate decision tree is then approximated, by maximizing the expectation for the two models to have minimum divergence.

Explanation by feature relevance & visualizations. Traditional tree-specific, global feature attribution methods are based on the following measures: (a) split count, measuring how often a particular feature is used in order to perform a split, and (b) gain, the contribution of splits performed on a particular feature to the total information gain. However, (Lundberg, et al., 2018) demonstrated that both options can produce inconsistent global feature attributions, in contrast to the consistent results of model-agnostic permutation importance. In addition, built-in variable importance measures for bagging ensembles, widely used for feature selection in research fields such as bioinformatics, were pointed out as biased under circumstances by several authors and alternative solutions were proposed, as in (Strobl, et al., 2008).

At a local scale, the statistical nature of tree-ensemble models and the discontinuity of the resulting decision boundaries pose additional difficulties to the assessment of valid feature attributions on individual predictions. The straightforward application of model-agnostic techniques is considerably time consuming, while results have been found to vary with random sampling (Lundberg & Lee, 2017). Moreover, local feature attribution methods such as PDP or LIME are unable to capture high-order nonlinear variable interactions, that tree ensembles are capable to uncover.

A unified approach is proposed by (Lundberg, et al., 2018) to overcome all the above obstacles. Based on the optimality of SHAP values as both accurate and consistent local feature attributions, they developed the TreeSHAP algorithm to provide a tree-specific, fast computation of the exact solutions, as well as SHAP interaction values to account for feature interaction effects. The aggregation of local SHAP explanations is presented as an alternative global feature attribution method, consistent as permutation importance but also preserving local faithfulness to the ensemble (Figure 3-23.A).

TreeSHAP is integrated into TreeExplainer (Lundberg, et al., 2019) and explanations produced by the proposed tool are displayed in joint visualizations, allowing for the identification of influential features and feature interactions. For example, SHAP interaction values can be used to decompose the SHAP dependence plot of Mortality risk on systolic blood pressure (Figure 3-23.B) into the separate contribution of systolic blood pressure without the interaction effect of age (Figure 3-23.C) plus the variable effect of systolic blood pressure on mortality with respect to age (Figure 3-23.D). In addition, the clustering of predictions across feature attributions can be used to identify subsamples of instances that are influenced by similar features, as in Figure 3-24.

Finally, following a completely distinct approach to the global feature attribution, (Kuralenok, et al., 2019) proposed the MonoForest framework, converting any tree-based ensemble in a unique polynomial form. In this context, an additive tree ensemble model is transformed into a "forest" of mononomials and ensemble-wise pruning is applied, whereas feature relevance is assessed by the polynomial form of the ensemble, with remarkable success as evidenced in Figure 3-25.

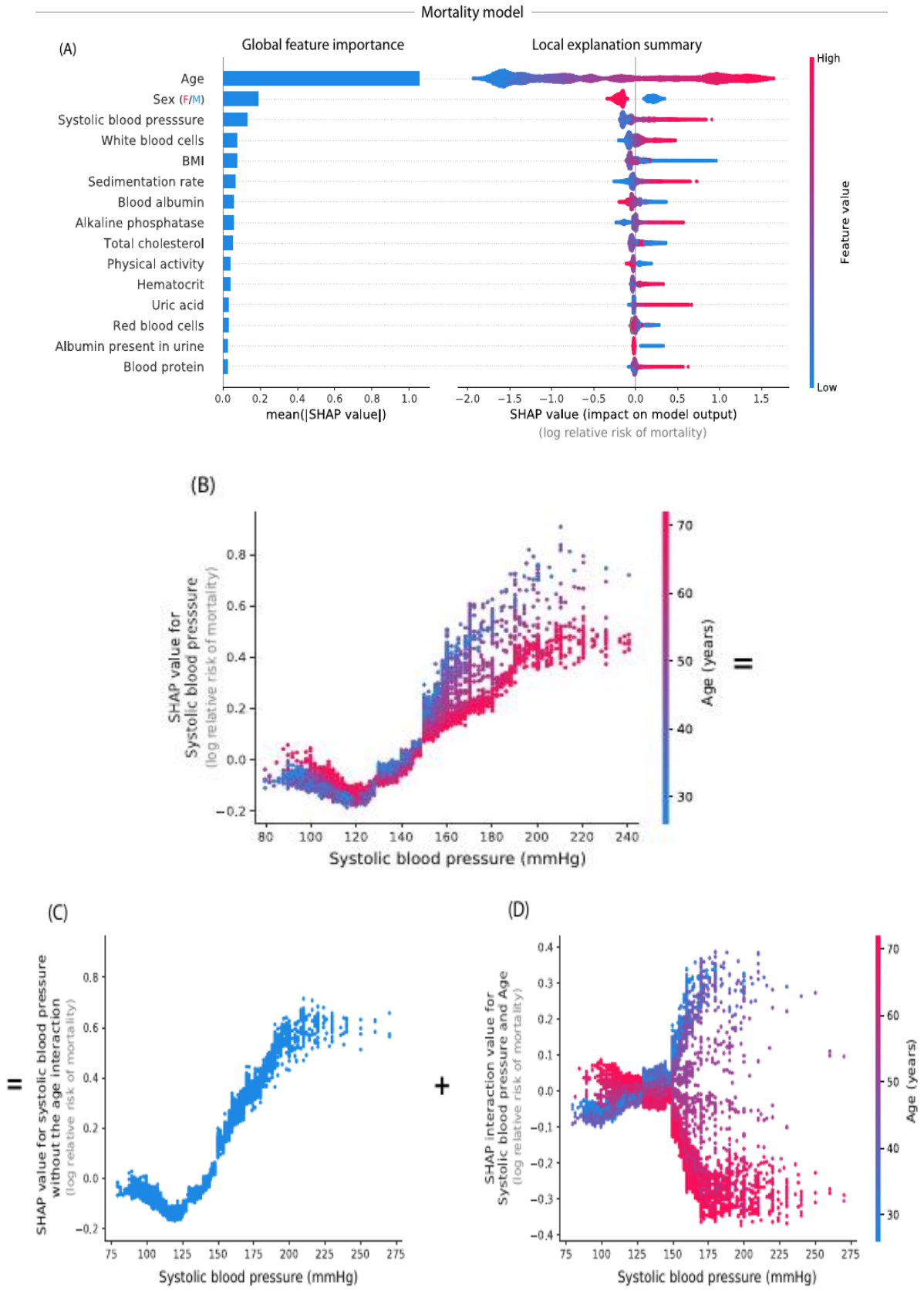

Figure 3-23 - Local explanations created by the TreeExplainer can be displayed in rich visualizations, to provide a global comprehension of the ensemble model's predictions. Image from (Lundberg, et al., 2019)

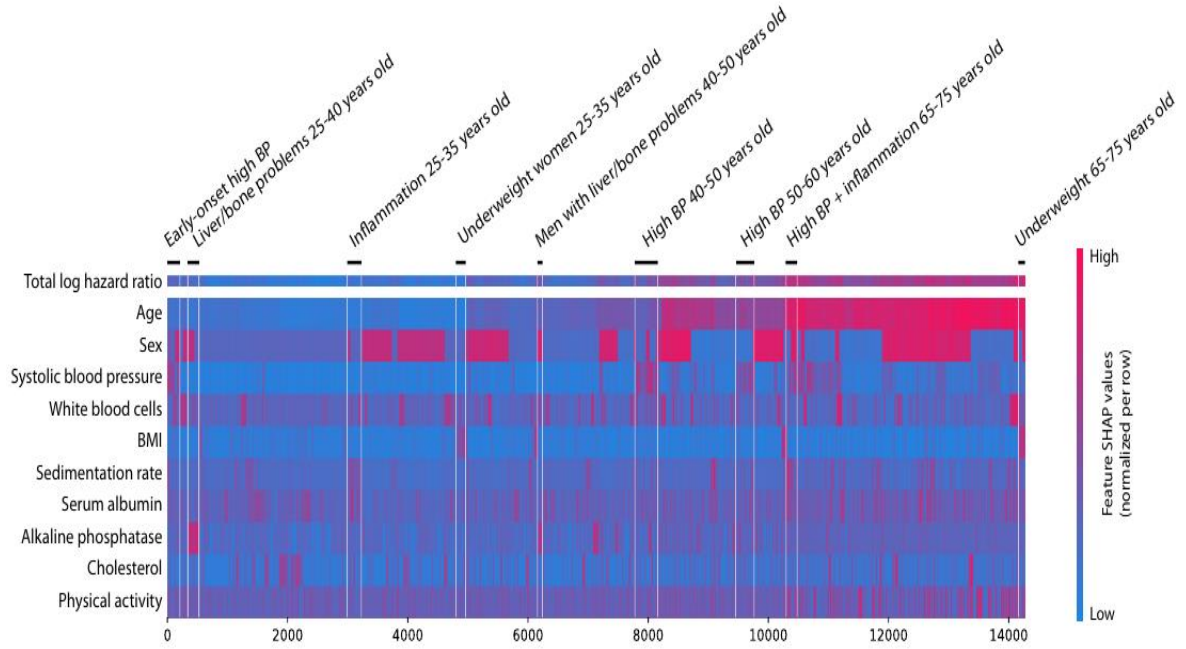

Figure 3-24 - Subgroups of people with similar mortality risks are clustered together in a feature attribution embedding. Image from (Lundberg, et al., 2019)

Local explanations by examples – counterfactuals. In (Tolomei, et al., 2017), an algorithm for tree-based ensembles was developed to provide interpretable suggestions (actionable recommendations) on how tweaking certain adjustable features can alter the outcome of a certain prediction to the opposite class, at the minimum possible effort. "Actionable feature tweaking" was extended to boosting ensembles by (Lucic, et al., 2019) and further refined in FOCUS (Flexible Optimizable Counterfactual Explanations for Tree Ensembles) by (Lucic, et al., 2019), outperforming both its pre-descendant as well as a recent contribution by (Kanamori, et al., 2020), with respect to the proximity of counterfactuals to the original instance as well as data coverage.

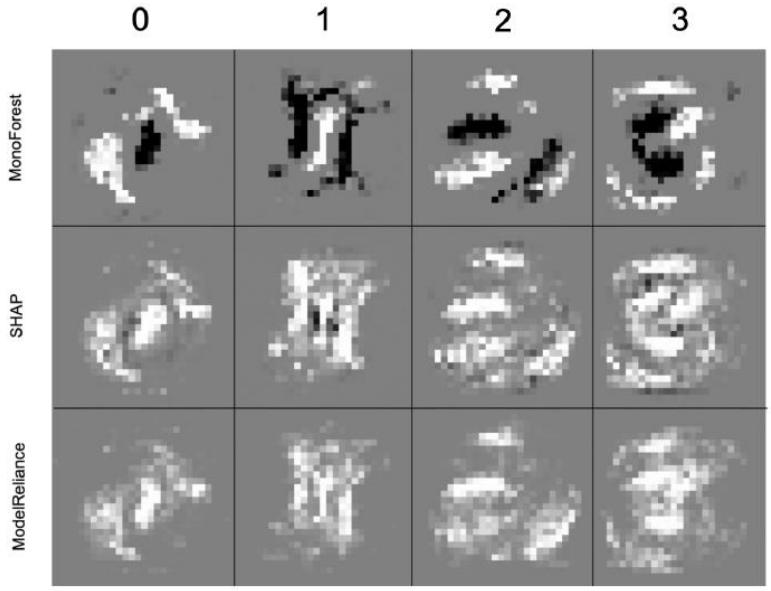

Figure 3-25 - Global feature attribution on the MNIST dataset for handwritten digit recognition. A boosted tree ensemble was built for each digit, as a one-vs-rest classifier using CatBoost. Each classifier was processed with MonoForest, global (mean

absolute) SHAP values and ModelReliance (based on permutation importance) and analysis results are displayed in the upper, middle and bottom row respectively. Image from (Kuralenok, et al., 2019)

Local explanations by examples – adversarials. Tree ensembles are particularly vulnerable to adversarial attacks and the definition of the minimum adversarial perturbation that results in unwanted change in model's prediction is a subject of great interest. A graph-based approach by (Chen, et al., 2019), manages to set tighter lower bounds to the minimum adversarial perturbation in considerably reduced computational time than the previously proposed method by (Kantchelian et al., 2016) that is based on Mixed-Integer Linear Programming. In (Zhang, et al., 2020), the authors carry out an extended comparative study of their proposed method for "adversarial attacks" to several previous ones, in terms of accuracy, robustness and execution time. The work by (Törnblom & Nadjm-Tehrani, 2020) in VoTE (Verifier of Tree Ensembles) is particularly focused on safety-critical applications, where the robustness of the adversarial explanations to noise and the plausibility of range are of essence. (Devos, et al., 2020) introduce a more complete perspective to model verification, taking into account the fairness of individual predictions. They develop VERITAS, a multitask verification tool for precise and robust adversarial attacks on tree ensembles, identification of influential features and assessment of model's fairness by examining the influence of protected features (such as race or gender). (Ranzato, et al., 2021) adopt a similar point of view and extend Meta-Silvae (Ranzato & Zanella, 2020), a genetic adversarial training tool that recovers and fixes vulnerability issues, into a Fairness-Aware Tree Training method (FATT).

Local explanations by examples – prototypes. In (Tan, et al., 2020), the authors define an adjustable number of representatives for each class, to cover any sub-region of the input feature space where a given prediction resides. Results of a conducted user study are presented, to support that local explanation with the use of prototypes is closer to human intuition, as compared to explanation by Shapley values.

Local explanations by examples – influential instances. The TREX algorithm has been developed by (Brophy & Lowd, 2020), to assess the effect of individual instances on a particular prediction. TREX utilizes a kernel similarity function designed for tree ensembles, in a representer point framework initially aimed to explain deep neural networks (Yeh, et al., 2018). The algorithm allows to debug the training dataset (locate & examine mislabeled samples), making use of the identified influential instances.

Support Vector Machines

Support Vector Machine (SVM) is a supervised learning algorithm based on the Structural Risk Minimization (SRM) principle (Cortes & Vapnik, 1995). The optimal separating hyper-plane, in the case of linearly separable classes, is the one leaving maximum possible distance to the closest points of each class (Figure 3-26, left). This distance is the functional margin. Points lying within this distance on either side of the decision boundary are the support vectors (SV), saved for later use (i.e. prediction) along with the decision boundary provided by a trained SVM. Since the decision boundary is defined on SV alone, these can be considered as the most representative/informative instances in the training data, with respect to the discrimination of classes. In the most common case of overlapping class distributions (Figure 3-26, right), the SVM "relaxes" the margin, allowing for some training instances to be misclassified, that is, be classified on the wrong side of the decision boundary but within the functional margin. As a result, SVM models show enhanced generalization ability.

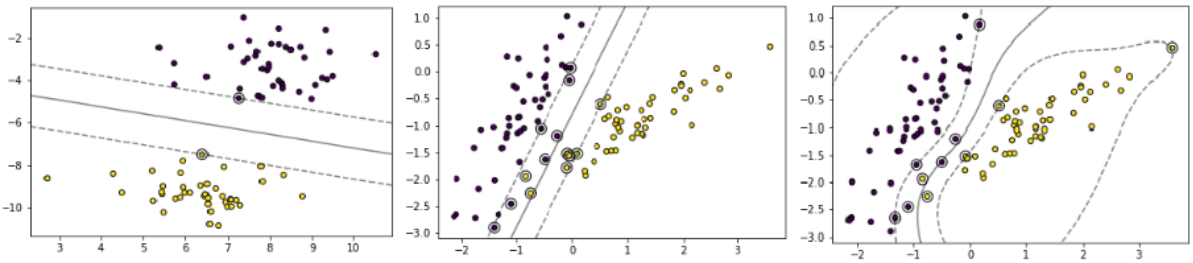

Figure 3-26 - Visualization of SVM for binary classification. Linear SVM on linearly separable classes is displayed in the 1st panel. SVM with linear and RBF kernel are displayed in the 2nd and 3rd panel respectively, where classes are not linearly separable. The solid line is the decision boundary, dashed line indicates the margin and Support Vectors are in circles.

Additionally, by utilizing the kernel trick the SVM model is capable to learn nonlinear decision boundaries in the input space (for example, learning a Radial Basis Function as in the last panel of Figure 3-26). These correspond to linear decision boundaries in a higher dimensional feature space, defined only on the SV. Therefore, the SVM can construct discriminatory features from the input variables and is often used for feature generation, selection and ranking (e.g. (Thuy, et al., 2011)). Finally, the trained SVM provides a weighted similarity matrix (kernel matrix), mapping the similarity/dissimilarity among SV in the constructed feature space.

However, both the input as well as the constructed feature space are typically of high dimension. Post-hoc techniques are consequently employed to explain SVM models, the most commonly used so far being explanation by simplification, feature attribution and visualizations.

Explanation by simplification. Methods with different levels of translucency are applied, to globally approximate a trained SVM by a Rule-based model or Decision Tree:

- Pedagogical methods involve SVM-specific implementations of model-agnostic techniques (Barakat & Bradley, 2010)). Considering the trained SVM as a black box, the global surrogate model is trained on artificially labeled samples, by substituting the original labels with the ones predicted by the SVM (e.g. (Torres & Rocco, 2005), (Huysmans, et al., 2008)).
- Decompositional methods utilize individual SVM components for rule extraction: (a) Based on the SV only. Rules can be extracted as regions in the input space, defined using the coordinates of the SVs (e.g. (Barakat & Bradley, 2007)). The extraction of Fuzzy Rules, instead of the traditional propositional rules, is proposed by similar work in order to enhance linguistic comprehensibility (e.g. Carraro 2013). (b) Based on SVs and decision boundary. Rules can be deduced via the formation of hyper-rectangles by the intersections between SVs and the separating hyper-plane (RulExSVM, (Fu, et al., 2004)). (c) Based on the SVs, decision boundary and training samples. Clustering (k-means) can be performed on the training data to find prototypes for each class, then combine SV, decision surface and prototypes to extract rules as regions formed in the input space (SVM+ by Nunez 2002). A recent contribution by (Shakerin & Gupta, 2020) synthesizes elements of the SVM model with local feature attribution, to develop SHAP-FOIL, an Inductive Logic Programming (ILP) algorithm. The authors utilize the FOIL top-down algorithm (Quinlan 1993) to deduce generic-to-specific logic clauses from support vectors, on the most influential features as indicated by SHAP values. Logic clauses induced for each SV are meant to cover similar training samples.
- Eclectic or hybrid methods comprise elements from both the above approaches, as for example in Active Learning Based Approach (ALBA) by (Martens, et al., 2009). The extracted rules are based on the SV and the size of the margin, along with SVM-labeled synthetic data close to the decision boundary, in order to focus on regions where most misclassified data reside.

Explanation by visualizations. Methods in this class follow two main approaches, in particular:

- Visualizing the decision boundary and margin, as in (Cherkassky & Dhar, 2010). The authors introduce the Univariate Histogram of Projections, of the training data on the normal direction to the SVM decision boundary (Figure 3-27). This method is very simple and particularly useful to visualize the model and its performance on high dimensional data.

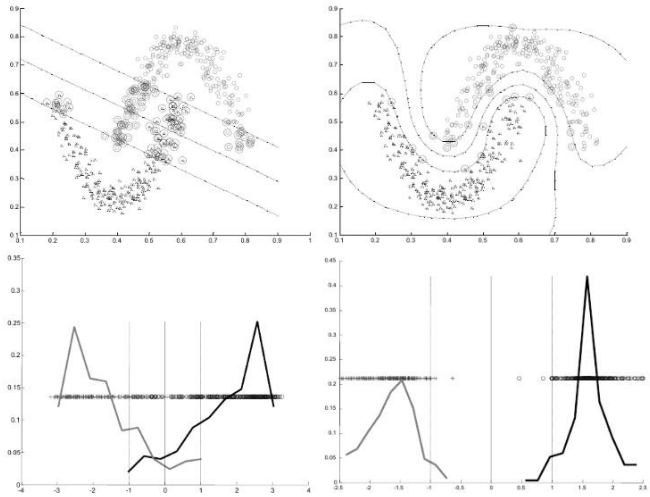

Figure 3-27 - The Univariate Histogram of Projections visualizes the position of training samples with respect to the SVM decision boundary and margin (Cherkassky & Dhar, 2010).

- Visualizing feature attribution. Model-specific feature attribution is based either directly on the weights assigned to the SVs by the trained SVM as in (Üstün, et al., 2007), or by processing this information taking also the SVM margin into account via a new statistic as in (Gaonkar, et al., 2015). Comparing results of the two methods in Figure 3-28, the latter (first two panels) seems to better capture the ground truth (last panel) than the former (3rd panel).

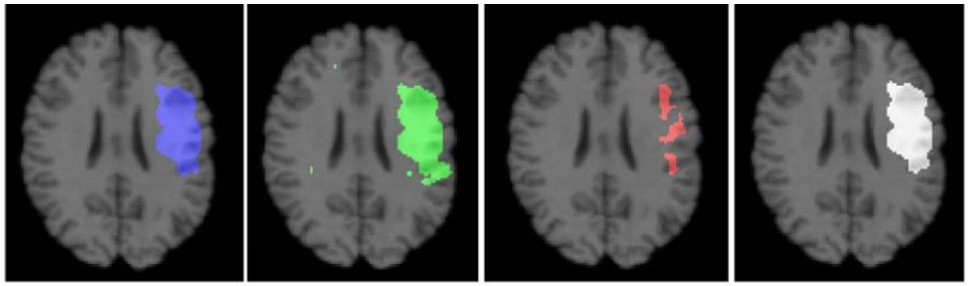

Figure 3-28 - Feature attribution on neuroimaging data. Results by Gaonkar (2015) are displayed on the 1st and 2nd panel, the 3rd panel is produced from the method by Ustun (2007), while the last panel presents the ground truth.

A different approach to feature attribution involves Taylor decomposition of the SVM kernel into main and pairwise input variable contributions plus the rest terms, as in (Van Belle, et al., 2016). Feature attributions are jointly visualized in a color-based nomogram, in case the evaluated rest terms can be ignored, as for an SVM model on the German Credit Risk Dataset (Figure 3-29). The contributions from 3 out of 20 input variables (balance, credit durability and amount of capital) are non-negligible. Main (top row) and pairwise (middle row) contributions of the 3 variables, given the specific input values for the particular person, are estimated based on the color legend. The sum of 6 contributions provides the final score for creditability (colorbar at the bottom).

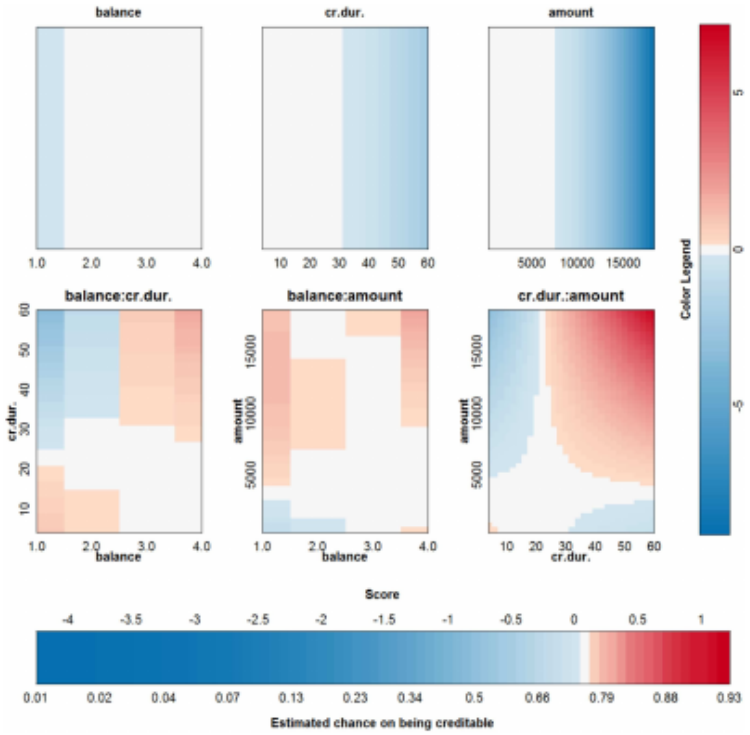

Figure 3-29 - Using the color-based nomogram to explain SVM on the German Credit Risk dataset. The creditability of a person ("score") can be assessed as the sum of main (upper row panels) and pairwise (middle panels) contributions.

Explanation by examples. Counterfactual explanations with increased robustness are provided by (Mochaourab, et al., 2021), for the case of a private SVM. The authors investigate the trade-off between SVM accuracy, privacy and explainability. Privacy mechanisms induce Gaussian noise perturbation to SVM weights, decreasing the model's predictive accuracy. In addition, the uncertainty in SVM weights compromises the robustness as well as the interpretability of the counterfactual explanations. As a result, increased privacy negatively affects the quality of explanations. The proposed method offers a tunable level of confidence that counterfactuals are indeed of the opposite class, according to the strength of imposed privacy.

Another recent study on counterfactual explanations for SVM by (Denton & Salleb-Aouissi, 2020), aims to provide weighted actionable recommendations, according to user preference on actionable features. Setting all weights to unity results in all features considered as static, allowing thus for local attribution of feature relevance.

3.3.2 Deep Learning models

Deep learning (DL) is a broad field of Machine Learning involving Artificial Neural Networks (ANN) with multiple layers ("deep") and representation learning. It is known since the seventies but started re-gaining wide attention recently due to the advances in computer hardware that led to efficient methods for ANN training. Its success is mainly related to its efficient performance on image and text data, although ANN can be used at structured data, as well. Deep learning models have been marked as entirely black-box due to their intrinsic complexity. The need to explain how DL models work has led to a significant rise in research works in the last years.

We split the Deep Learning explainability methods following the fundamental categorization of Deep Learning models (Deep Neural Networks - DNN, Convolutional Neural Networks - CNN and Recurrent Neural Networks - RNN). We have to note that the categories overlap and some explainability techniques can be applied in more than one case.

Deep Neural Networks

The most important categories of explainability techniques for Deep Neural Networks are: Feature Relevance and Model Simplification.

Feature Relevance. Feature Relevance is the broad category of explainability methods that answer the question "how much each input feature contributed to the output", producing an explanation that accompanies the prediction.

Most feature relevance methods share the common idea of applying the backpropagation technique on the values of input features to measure how much a slight change in the input affects the prediction. Though, at the technical level, there are many different alternatives on how to achieve so efficiently. An overview of such alternatives has been published by (Ancona *et al.*, 2017). This work also proposes an evaluation metric, called sensitivity-n, as a unified metric for comparing each method's accuracy. A well-known method called Deep Taylor decomposition was proposed by (Montavon *et al.*, 2017). Deep Taylor decomposition introduces a new relevance propagation technique where feature relevance is propagating from the top layer down to the input. Furthermore, (Shrikumar, Greenside and Kundaje, 2017) proposed an alteration of the gradient type used in the backpropagation, named "signed partial derivatives", leading to an improvement on the sharpness of the attribution map.

A fundamental problem that many backpropagation approaches face is the effect of vanishing (saturating) gradients. Vanishing gradients block the propagation of information from the output to the input, leading to false estimation of the feature relevance. For addressing it, (Sundararajan, Taly and Yan, 2017) proposed the method of "integrated gradients", which incorporates a reference input and computes all gradients along the line from the reference input to the actual value. In this way, the method avoids vanishing gradients, but a computational overhead is added due to the numerical approximation of the integral. Finally, another approach that tackles the vanishing gradient effect is DeepLift (Shrikumar, Greenside and Kundaje, 2017). It adopts a reference point in both the output and the input and measures differences from the reference points. Then, it incorporates these differences for identifying which input feature plays a significant role in the prediction. In this way, an efficient and accurate estimation of feature relevance is achieved.

Model Simplification. Although feature relevance has dominated the field in recent years, model simplification proposes a worth-mentioning different viewpoint. The DeepRed algorithm proposed by (Zilke, Mencía and Janssen, 2016) tries to decompose the network's decisions into a tree representation. Each layer of the network is mapped to a logical decision (i.e. if-then) and by connecting these logical decisions, we can create an equivalent tree-representation. The methods main advantage is that it provides a very comprehensible explanation, i.e. sequence of if-then decisions. On the other hand, the method has also two basic drawbacks, (a) not all layers can be mapped to logical decision (e.g. residual connections) and (b) as the network becomes deeper, the sequence of logical decisions grows larger and the tree cannot then provide a comprehensible explanation.

Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) have been quite successful in various tasks related to images such as localization, classification, segmentation etc. Interpreting CNNs involves understanding the high-level features extracted by the network units[6]. CNNs tend to learn more and more complex features hierarchically. For example, a CNN trained for locating people will start by detecting low-level visual elements (e.g. edges, angles) in the first layers and will synthesize them into more complex parts

[6] It is important to notice that, depending on the work, quite different things are described as a network's unit; a single neuron, a whole feature map or even a whole layer. In general, we can think of the entity unit as "a part of the network".

(e.g. head, feet) as we move deeper. Figure 3-30 presents a very intuitive example for understanding this process. Interpreting what a specific unit of a network is trying to learn is the fundamental task of explainability for CNNs. From a mathematical perspective, the different ways of addressing the question "what a specific unit is trying to learn" define the various explainability categories. We present these categories in the following paragraphs.

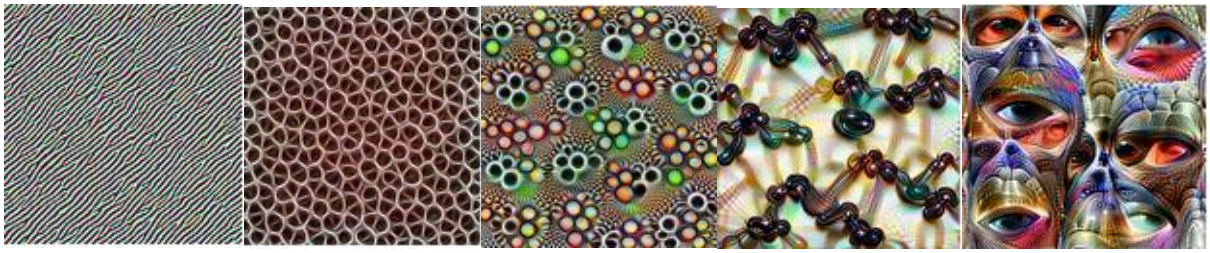

Figure 3-30 - Progressive enhancement (from left to right) of the visual features inside the deep neural network. The simple edges are combined to create textures and patterns, which are used as the building blocks for the parts of whole objects. Images are taken from https://distill.pub/2017/feature-visualization.

Feature Visualization. Feature Visualization is the most popular class of explainability methods. It explains what a unit is trying to learn by finding the input (image) that maximizes the unit's output (Olah & al., 2017). Various techniques have been proposed for achieving so. (Erhan *et al.*, 2009) recommended solving it as an optimization problem, exploiting gradient ascent and backwards-propagation. Namely, we consider the trained weights of the network as constants, and we search for the input that maximizes the unit's output. The optimization problem can be solved by starting from a random image and use gradient-ascent for iteratively moving to the optimal. (Nguyen, Yosinski and Clune, 2016) introduced a probabilistic perspective, proposing to search for the image that would maximize a specific class's probability. Unfortunately, the methods above do not ensure the creation of an interpretable input image. Conversely, if the regularization constraints are not enough, the produced image may look like random noise, as in Figure 3-31.

Another way to identify images that maximize units' output is to search in a list of real images, e.g. the training set. There are also techniques that lie in the middle of the two extremes, e.g. (Nguyen *et al.*, 2017) proposed finding the training's set image that maximizes a unit and use it as the starting point of generating an artificial one. A slightly different approach implies that a single unit may only capture a single feature. Hence, to uncover what the model is trying to learn, we have to combine such units. The idea of interactions between units' neurons was introduced by (Szegedy *et al.*, 2014). Continuing this research line, (Bau *et al.*, 2017) showed that interpretable features could be revealed by searching in the same spatial positions of different activation maps.

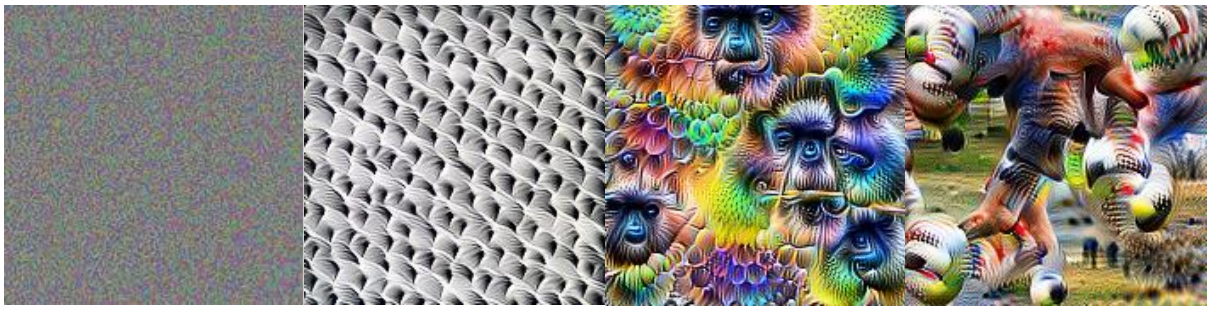

Figure 3-31 - Some examples of feature visualization. We observe that some cases the image is not interpretable at all (left image) while in some others, it is much clearer. For example, the third image clearly corresponds to a monkey.

Clustering the features. Another way of answering what a unit is trying to learn is by finding the dataset's images that lead to a similar output to the specific unit. This could be performed by applying

clustering methods in the feature space. (Papernot and McDaniel, 2018) proposed to use the common k-NN technique for clustering the outputs of all the activation maps.

Using text to explain images. A different class of explainability is generating text for describing the visual content of an image. The idea is that if a model produces natural text for explaining the visual elements that are apparent in an image, then we know the reason behind the prediction, as in Figure 3-32. For example, (Bengio, Simard and Frasconi, 2016) proposed to apply an RNN attention model on the features extracted by the CNN, for producing text explanations. Also, (Xiao *et al.*, 2015) proposed the following scheme: (a) presenting candidate patches inside the image (b) selecting relevant patches to a particular object and (c) localizing the discriminative parts.

Figure 3-32 - Using text to explain what the CNN understands; in the current example, the network outputted "A woman Is throwing a frisbee in the park". The figure is taken from (Bengio, Simard and Frasconi, 2016)

Recurrent Neural Networks (RNNs)

Equally to CNNs that have dominated the computer vision domain, RNNs have been the state-of-the-art approach in Natural Language Processing (NLP)[7]. A particular class of RNNs are the Long-Short Term Memory (LSTM) networks that have successfully captured relationships between input entities that appear far away in the sequential order[8]. Since RNNs are relatively recent models and their inner structure is very complicated, finding methods for making them interpretable still remains an open area. Feature relevance, which in the case of RNNs involves identifying the words that played a significant role in the final prediction, is the fundamental explainability technique. Figure 3-33 shows a representative example where the words that contributed more to classify the sentence expressing a positive or a negative opinion are highlighted with an intense red. In his work (Arras et al.) presents an algorithm for adjusting gradient-based sensitivity analysis (Li *et al.*, 2016) and Layer-Wise Relevance Propagation (Binder *et al.*, 2016) in the case of RNNs. Another category of interpretability techniques is called Representation Plotting (Li, et al., 2016). Here we try with proper visualizations (e.g. t-SNE, UMAP) to understand the relative position of input words and phrases in the feature space. Hence, we can visually spot the clusters of words and phrases with similar descriptors.

[7] For the shake of simplicity, we will restrict ourselves to the case where the input is a sentence written in natural language, even though RNNs can be used in all types of sequential data, e.g. time series.
[8] All the explainability techniques that are described in the RNN paragraph, can also be applied in LSTM networks.

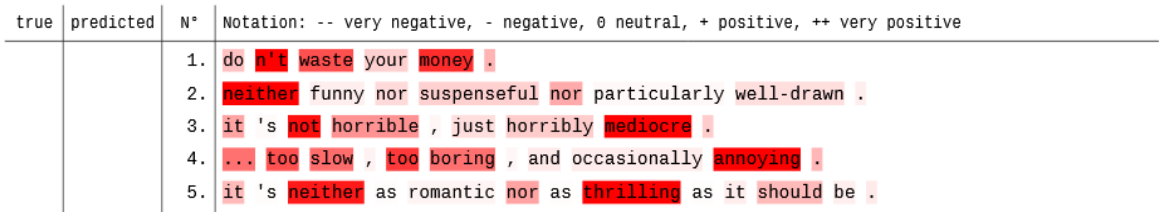

Figure 3-33 - Feature Relevance example in the NLP domain. Stronger red corresponds to words that contributed more to the prediction. Image is taken from (Arras et al.)

3.3.3 Advantages & Limitations

Model-specific interpretability techniques have the fundamental advantage of exploiting the internal structure of the model under examination, for providing accurate and efficient explanations. For example, in the case of deep learning, most interpretability methods take advantage of automatic differentiation for measuring the feature relevance. This approach leads to an important speed-up in the extraction of the explanation. Attempting a model-agnostic technique in this particular scenario would probably lead to a very slow explanation extraction procedure. Furthermore, model-specific techniques take under consideration the particularities of the domain of a machine learning model. For example, it would be pointless to measure a specific feature's relevance[9] in the case of a CNN model, since the individual feature is a single pixel. On the contrary, it is suitable to create a comparison of the relevance between different features, which can be visualized nicely by a heatmap. The main disadvantage of model-specific techniques is their applicability which is restricted to particular class of models. Hence, in order to explain many models we have to also develop many different interpretability techniques.

3.4 Graph ML techniques

Graphs are a powerful and flexible way to visually represent a data network and are frequently used to express relations and solve real-life problems. Nowadays, there is an increasing interest in graph analysis by many scientific fields that model and process this type of data, like Biology, Social Networks analysis or Chemoinformatics (Zhou, et al., 2020; Zhang, et al., 2020). In the Manufacturing sector specifically, graph networks can describe pathways of IoT devices and sensor networks (Aggarwal, et al., 2017) in the framework of predictive maintenance, or represent associations between resources, daily workload and production in decision-making and dynamic scheduling problems (Hu, et al., 2020). There are also cases less specific to manufacturing, but nonetheless applicable across a number of industries such as the customer identity and access management (Poolsappasit, et al., 2012), the supply chain support (Tan, et al., 2015) or the anomaly and fraud analysis (Akoglu, et al., 2015).

Graph data science is a sub-domain of data science devoted to the extraction of useful knowledge from relationships and structures in data. This knowledge is then leveraged, typically with the assistance of machine learning models, to answer some previously intractable questions or produce some form of prediction. To this end, the process followed can be roughly identified as follows (see Figure 3-34): The workflow starts with the creation of a knowledge graph or some other form of graph-structured dataset derived from a knowledge-base. Graphs are represented computationally using various matrices (e.g. Incidence matrix, Adjacency matrix, Degree matrix or Laplacian matrix), with each matrix providing a different type of information. The goal is to extract latent feature representations (known as graph embeddings) of the available nodes and edges through a process known as graph representation learning (or graph feature engineering). The graph embeddings is an appropriate transformation of the graph structure that allow graph ML models (usually graph neural

[9] As many model-agnostic techniques do.

networks) to perform further analysis and computations by "learning" the graph topology (graph native learning).

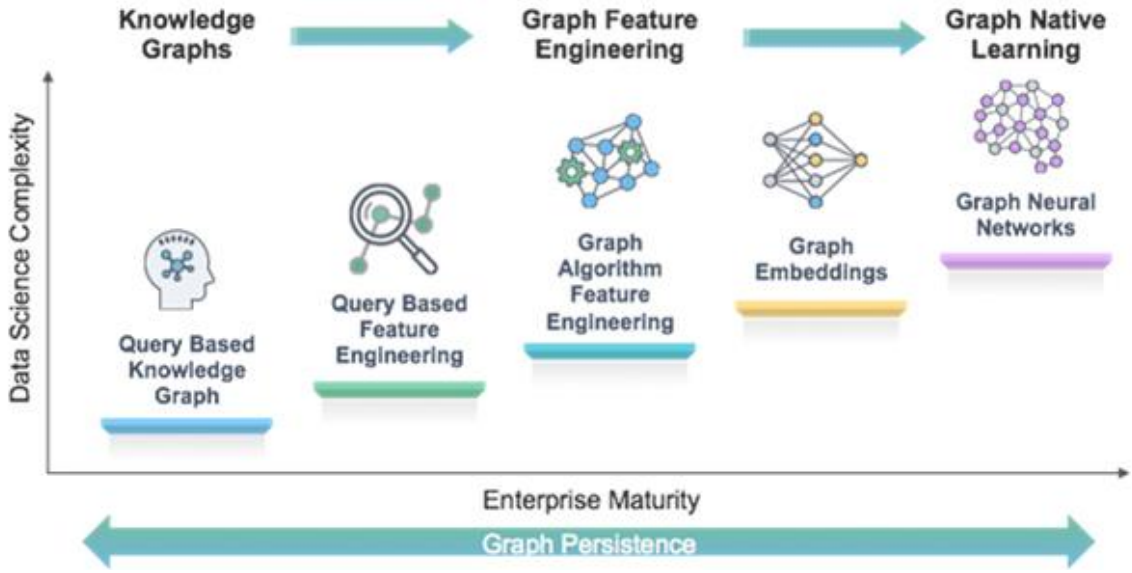

Figure 3-34 - Steps of Graph Data Science (image from neo4j site[10])

More specifically, machine learning can solve a number of graph analytics problems that can be broadly classified into the following five categories:

- Node classification (also known as node attribute inference),
- Link (edge) prediction (often referred to as recommendation),
- Node clustering (also known as community detection),
- Graph or subgraph classification, regression, and clustering
- Graph generation

Graph analysis using machine learning (Graph ML) is a quite popular topic during the last few decades with increasing interest both from industrial and academic point of view. A wide range of practical applications for graph ML has been introduced by various publications in literature (Battaglia, et al., 2018; Zhou, et al., 2020). These applications can be divided into two categories based on the data available: the first category is about data where the entities and relations can be explicitly specified, like knowledge graphs or road networks, while in the second category data, where the relational structure is not clear and must be inferred or assumed, like visual scenes or text corpora, belong.

No matter how the structure is given, in order for someone to apply analytic tasks on graphs, such as classification, or clustering, it is necessary, as already mentioned, to find a mapping or representation of the discrete graph onto the continuous domain that algorithms function. This task is called "graph representation learning" and the output of the process is called graph embedding. A number of surveys in the literature give a more analytical view on traditional ML and more advanced graph embedding techniques (Chami, et al., 2020; Kinderkhedia, 2019; Cai, et al., 2018; Goyal & Ferrara, 2018) comparing their performance and results. Graph embedding methods have generally fallen into four main groups, according to their underlying techniques. The first group, distance based approaches, aim to produce optimal embeddings by minimizing of Euclidean distances between similar nodes and are commonly used for visualization purposes. The second group is based on matrix factorization and focuses on learning network embeddings by factorizing the matrix that represents the connections between nodes. The third one, random walk-based methods, leverages random walk techniques to approximate certain properties of a graph, like node centrality and similarity. The fourth

[10] https://neo4j.com

group is related to graph neural networks and auto-encoders that aim to learn differentiable functions over discrete topologies with arbitrary structure.

During the last decade, research has shifted to obtaining more scalable methods using deep learning approaches, under the umbrella-term "Geometric Deep Learning" introduced by Bronstein in (Bronstein, et al., 2017). These techniques incorporate the graph embedding mechanism into their core functionality (e.g. as a filter in convolutional neural networks) as an unsupervised part whilst performing as ML models (supervised part) for the classification or regression task at hand. Recent works, like the ones in (Wu, et al., 2021; Zhang, et al., 2020; Cao, et al., 2020), review and compare the most recent approaches and propose a, more or less common, taxonomy for Geometric DL methods and Graph Neural Networks (GNN) based on their model architectures and training strategies. The authors also discuss some of the GNN limitations, such as the difficulty to directly apply such models on heterogeneous graphs, or to incorporate interdisciplinary knowledge or to capture dynamic spatial relations. More important, the advantage of scalability that GNNs offer, comes at the price of corrupting graph integrity.

In summary, traditional machine learning algorithms seem to face significant challenges by the complexity of graph data when applied directly. Graphs can be irregular, and often have a variable size of unordered nodes. In contrast to image processing, where the nodes of a processed image can be considered as part of a grid with each node having a fixed number of neighbours, in the graph domain the nodes may have a different amount of neighbours making convolutions difficult to apply. Furthermore, a core assumption of existing ML algorithms is that instances are independent of each other. This assumption, however, is no longer valid for graph data because each instance (node) is related to others by links of various types, such as machine interactions, sensor connections or supply chain relations.

The sections that follow explore in depth the different ML models and techniques under the prism of graph data science, from traditional methods to the more advanced and scalable Geometric deep learning approaches.

3.4.1 Traditional ML techniques on graphs

Traditional methods for graph analysis focus on different tasks of graph analysis (classification, clustering, visualization) and depend on different information retrieved from the graph. Supervised and unsupervised models have been developed, in an effort to extract knowledge from graph topologies. Community finding, clustering, network ranking, semi-supervised learning (for classification, link prediction) are some of the applications that traditional machine learning methods on graphs are used for. The typical ML techniques applied on graphs include:

- **Girvan–Newman clustering**, which is a structured-based approach that calculates the betweenness of edges and removes those with the highest values. The nodes that remain linked are considered to be close to one another than those separated (Girvan & Newman, 2002).
- **Spectral clustering**, a kernel based approach based on the eigenvector decomposition of the graph Laplacian. The K largest eigenvectors are chosen to cluster the nodes in K clusters. A 2D matrix is formed having the K eigenvectors as columns and N number or rows. For the clustering, an algorithm such as k-means is employed (Ng, et al., 2001).
- **Modularity based clustering** that is another structure-based approach in which the clusters are formed in such a way that the modularity score is maximized after each partitioning. Modularity is calculated as the subtraction of the edges connecting the nodes of two communities and the edges connecting the nodes in each community (Newman, 2006).
- **Semi-supervised learning with kernel matrix** as a kernel based approach that attempts to classify the unlabeled nodes of a directed graph according to the information provided by the labeled nodes. A kernel matrix based on dyadic links is used while directionality and global relationships are considered for the classification (Zhou, et al., 2005).

- **Semi-supervised learning with discriminative random walks (D-walks)** which an approach based on random walks. According to this approach, the nodes are classified based on a class betweenness measure that depends on the passage time during random walks performed in the input graph. This approach is less expensive than the before mentioned kernel approach and can also been used for clustering (Callut, et al., 2008).
- **Stochastic block-model (SBM)** as a model based approach that depends on block modeling. Block models map the nodes of a network to a series of clusters and the interactions between members of the clusters is expressed by a set of blocks. SBM assumes structural equivalence of the nodes and the probability of a connection between two nodes is described by the probability of the clusters they belong to (Holland, et al., 1983).
- **Latent position cluster model (LPCM)** as a distance based approach that maps each node to a position in a Euclidean latent space. Through this mapping of the nodes, a representation of the network occurs such that the nodes which are likely to be connected have similar positions. LPCM assumes that there is an embedding of the network in low-dimensional space and draws the positions from a Gaussian mixture model in the latent space, such that each Gaussian distribution corresponds to a cluster (Handcock, et al., 2007).

Implementations of the aforementioned methods on graphs can be found in different languages e.g. in R (Girvan–Newman clustering, Stochastic block-model, Latent position cluster model) and in Python either in Scikit-learn[11] or PyTorch[12] (Spectral clustering, Modularity based clustering, Semi-supervised learning with kernel matrix, Semi-supervised learning with discriminative random walks).

3.4.2 Graph Representation Learning

Graph representation learning (GRL), or Network Embedding, is a subfield of Graph ML that aims to learn a mapping function from a discrete graph to the continuous domain (Chami, et al., 2020). In other words, GRL is an approach that transforms a graph structure into a, usually, lower dimension vector space, while preserving, as much as possible, graph structure and information. That is why GRL methods are also referred to as non-linear dimensionality reduction methods. Especially on large graphs, the need for lowering the dimensionality of the input data is also an important requirement for ML methods to work. This is due to the "curse of dimensionality" but also due to the lack of computational resources. Based on the output granularity, the graph embedding output can be divided into four categories, namely node embedding, edge embedding, hybrid embedding (both edge and node) and whole-graph embedding.

The most prominent and popular techniques, which are summarized in Table 3-1, include graph kernels, matrix factorization-based methods, random-walk based algorithms and Graph Neural Networks, which can all be deemed as a subset of unsupervised machine learning. Graph embeddings can be used in a variety of tasks, which can be classified as: network compression, visualisation, clustering and graph reconstruction. If followed by an appropriate complementary algorithm (e.g. SVM), these approaches can also lead to node classification, link prediction and recommendation.

It is worth mentioning, that most of these methods are available either as custom Python or R implementations, as standalone libraries or as part of a more general framework like Scikit-Learn[11], like GRaKeL[13], Stellargraph[14] and PyTorch Geometric[12].

[11] https://scikit-learn.org/stable/
[12] https://pytorch-geometric.readthedocs.io/en/latest/#
[13] https://github.com/ysig/GraKeL
[14] https://github.com/stellargraph/stellargraph

Table 3-1 - Network Embedding ML algorithms

Algorithm Family	Algorithm Name
Distance Based	Multi-Dimensional Scaling (MDS), IsoMap, LLE, LE, t-SNE, Poincare embeddings
Factorization based	Graph factorization (GF), GraRep, HOPE
Random walk based	DeepWalk, Node2Vec, WYS , Graph2Vec, LINE, HARP, Graph Kernels
GNN and Auto-encoders	SDNE, DNGR, VGAE, DGI

Distance based

The distance-based methods rely on the minimisation of Euclidean distances between similar nodes in order to produce the optimal embeddings. Among these, there are linear embedding methods such as Multi-Dimensional Scaling (MDS) or PCA (which is a particular case of MDS), that learn low-dimensional linear projection subspaces, and non-linear methods such as Laplacian eigenmaps and Local linear embedding. More recent works, like the Poincaré embedding method and its variations, use non-Euclidean distances (e.g. in the hyperbolic space) with better results reported. All of these methods have originally been introduced for dimensionality reduction or visualization purposes, but can easily be extended to the context of graph embedding.

In detail, the distance-based algorithms include:

- **Multi-Dimensional Scaling (MDS).** This family of linear embedding techniques maps objects into a N-dimensional Cartesian space while preserving pairwise distances between objects. It is used to display the information of a distance matrix and visualize the level of similarity among objects (Kruskal, 1964).
- **Isometric Mapping (IsoMap).** It is a non-linear extension of MDS that approximates manifold distances in contrast to straigh-line Euclidean geodesics. It utilizes the KNN algorithm to construct a neighbourhood graph (Tenenbaum, et al., 2000).
- **Locally Linear Embedding (LLE).** LLE is also a non-linear dimensionality reduction method which approximates each point using a linear combination of embeddings in its local neighborhood (linear patches). These local neighborhoods are then compared globally to find the best non-linear embedding (Roweis & Saul, 2000).
- **Laplacian Eigenmaps (LE).** The LE algorithm exploits the spectral properties of the Laplacian operator (matrix) of a graph so as to provide an optimal embedding and perform a geometrically motivated dimensionality reduction. The high-level intuition for LE is that the points that are close on the graph will have similar representations, due to the "smoothness" of Laplacian's eigenvectors with small eigenvalues (Belkin & Niyogi, 2001).
- **t-distributed Stochastic Neighbor Embedding (t-SNE).** The t-SNE is a well-known dimensionality reduction algorithm that can be applied also on graph embedding. T-SNE tries to minimise the Kullback-Leibler (KL) divergence of the joint probabilities in the original space and the embedded space by grouping samples based on local structure (Van Der Maaten & Hinton, 2008).
- **Poincaré embeddings.** This algorithm belongs to the non-Euclidean methods and learns hierarchical representations of a graph by embedding them into the hyperbolic space (n-dimensional Poincaré ball). Due to the underlying geometry, both hierarchy and similarity characteristics can be captured. Embeddings are learnt based on the Riemannian optimization (Nickel & Kiela, 2017).

Factorization based

Factorization based methods factorize the matrix that represents the connections between nodes to obtain the node embedding. The approaches presented here, vary based on the matrix employed, which can be the node adjacency matrix, the Laplacian matrix, the node transition probability matrix, or the similarity matrix, among others. Most of these approaches are used for graph compression as analyzed below:

- **Graph factorization (GF).** Graph Factorization factorizes the adjacency matrix of the graph by minimizing a graph regularization loss function (Ahmed, et al., 2013).
- **Graph representation with global structure information (GraRep).** GraRep is a representation learning method which learns asymmetric embeddings. It uses the node transition probability matrix and preserves k-order proximity (Cao, et al., 2015).
- **HOPE.** HOPE is also able to learn asymmetric embeddings while preserving higher order proximity like GraRep, but it employs various similarity measures and a generalized Singular Value Decomposition (SVD) to efficiently obtain the embedding (Ou, et al., 2016).

Random walk based

Random walks is a popular technique used to approximate some of the properties of a graph including node centrality and similarity. They are especially useful when a graph is partially observed or the graph is too large to measure in its entirety and are commonly used as a pre-step for node classification and link prediction. The most popular embedding techniques that use random walks on graphs are the following:

- **DeepWalk.** Inspired by the skip-gram model approach commonly found in NLP tasks, DeepWalk can analyze a graph by learning latent representations associated with each vertex in the graph through a stream of random walks. It is scalable and parallelizable due to its online nature (Perozzi, et al., 2014).
- **Node2Vec.** The algorithm uses random walks to construct vertex neighbourhoods and learns a mapping of nodes to a low-dimensional space that maximizes the likelihood of preserving these network neighbourhoods. It combines graph exploration through breadth first search (BFS) and through depth first search (DFS) to generate biased random walks on the graph that leads to more informative embeddings (Grover & Leskovec, 2016).
- **Graph2Vec.** Similar to Node2Vec, but instead of feature representation of nodes, Graph2Vec learns entire graphs. This is accomplished in the same way word2vec works on text documents, i.e. using a "corpus" of graphs, graph2vec considers the set of all rooted subgraphs (neighbourhoods) around every node as a vocabulary to be trained with (Narayanan, et al., 2017).
- **Watch Your Step (WYS).** It is a type of Graph Attention model (GAM), improving on the ideas of DeepWalk and Node2Vec, that automates hyper-parameter tuning using a faster and more efficient mechanism. This mechanism replaces the previously fixed hyper-parameters with attention ones that get trained via backpropagation (Abu-El-Haija, et al., 2017).
- **Hierarchical representation learning for networks (HARP).** HARP is more of a pre-processing strategy that simplifies the graph in order to achieve faster training and avoid local minima. HARP uses graph coarsening to hierarchically reduce the number of nodes by grouping them together and introducing "supernodes". It can be applied to other methods, such as DeepWalk, LINE and Node2vec (Chen, et al., 2018).
- **Large scale Information Network Embedding (LINE).** LINE is a scalable and efficient graph embedding method which is also suitable for arbitrary types of graphs: directed, undirected, or weighted. It also preserves both 1st and 2nd order proximities, by defining two different loss functions and tries to minimize the combination of the two (Tang, et al., 2015).

- **Graph Kernels.** Graph kernels is a whole family of kernel methods used in measuring graph similarity and are commonly utilized for graph classification along with kernel-based machine learning algorithms like SVM (Vishwanathan, et al., 2010).

Graph Neural Networks and Auto-encoders

Deep neural networks, in general, can be regarded as an effective method for learning high level abstractions from low level inputs. This process essentially performs dimension reduction that maps data from a high dimensional space into a lower dimensional space. Unlike the (truncated) SVD-based dimension reduction method, which maps from the original representation space to a new space with a lower rank through a linear projection, deep neural networks such as stacked Auto-Encoders, can scale easily and learn projections which are highly non-linear. The most prominent standalone examples in graph representation learning are the following:

- **Structural Deep Network Embedding (SDNE).** The SDNE model consists of an unsupervised part and a supervised part. The unsupervised part includes an autoencoder that uses the adjacency matrix to find an embedding for each node which can reconstruct its neighborhood by preserving the first and second order network proximities, while the supervised part relies on Laplacian Eigenmaps that apply a penalty when similar vertices are mapped far from each other in the embedding space (Wang, et al., 2016).
- **Deep Neural Networks for Learning Graph Representations (DNGR).** The DNGR model consists of three components: random surfing, positive pointwise mutual information (PPMI) calculation and stacked autoencoders for denoising. Random surfing generates a probabilistic co-occurrence matrix analogous to the similarity matrix in HOPE (Cao, et al., 2016).
- **Variational Graph Auto-Encoders (VGAE).** This approach utilizes variational graph auto-encoders (GAEs) to encode and decode the graph structure. The input is the adjacency matrix which is fed to a graph convolutional network encoder that learns the higher order dependencies between nodes in an unsupervised manner (Kipf & Welling, 2016).
- **Deep Graph Infomax (DGI).** A general approach for learning node representations within graph-structured data in an unsupervised manner. DGI relies on maximizing mutual information between patch representations and corresponding high-level summaries of graphs, both derived using established graph convolutional network architectures (Veličković, et al., 2018).

3.4.3 Geometric Deep Learning

The term “Geometric Deep Learning” was first seen only a few years ago (Bronstein, et al., 2017) as an umbrella phrase that encompasses all approaches that aim at applying deep learning techniques to non-Euclidean data, such as graphs. These approaches can be divided into seven categories: Recurrent GNNs, Convolutional GNNs, Spatio-temporal GNNs, Graph Reinforcement Learning, Graph Adversarial Methods, Graph Attention Networks and Graph Generative Networks. The categorization of the models depends on the task they focus on and the neural network approach that is being used. It should be noted that in some cases the distinction is not obvious as many algorithms may belong in more than one category, such as in the case of NetGAN that belongs both in the Graph Adversarial Methods (due to the methodology) and in the Graph Generative Networks (due to the task, i.e. graph generation).

Although deep learning techniques, as already discussed, can be applied as standalones for representation learning, their most common and important use is when combining the learning embeddings step with a more advanced function, like predicting node or graph labels. In fact, the tasks that can be handled by these methods include graph classification, node classification, network embedding, graph generation, spatial-temporal graph forecasting, node clustering, and link prediction (Wu, et al., 2021).

The most popular deep learning models that achieve this two-phase functionality are summarized in Table 3-2 and presented in detail in the lines to follow:

Table 3-2 – Graph Neural Network Algorithms

Algorithm Family	Algorithm Name
Recurrent GNNs	Graph Neural Networks (GNNs), Gated graph sequence neural networks (GGS-NNs), Stochastic Steady-state Embedding (SSE), Graph Echo State Network (GraphESN)
Convolutional GNNs	Spectral Convolutional Neural Network (Spectral CNN), Chebyshev Spectral CNN (ChebNet), Graph Convolutional Networks (GCNs), Adaptive Graph Convolution Networks (AGCNs), Neural Network for Graphs (NN4G), GraphSage, Diffusion Convolutional Neural Network (DCNN), Dual Graph Convolutional Networks (DGCN), PATCHY-SAN, Message Passing Neural Network (MPNN), Mixture Model Network (MoNet)
Spatio-temporal GNNs	Graph Convolutional Recurrent Network (GCRN), Diffusion Convolutional Recurrent Neural Network (DCRNN), Structural-RNN, CGCN, ST-GCN, Graph WaveNet
Graph reinforcement learning	Graph convolutional policy network (GCPN), Graph transformation policy network (GTPN), Graph Attention Model (GAM)
Graph adversarial methods	GraphGAN, Adversarial network embedding (ANE), NetGAN, Nettack, Adversarial attacks on graph neural networks via meta learning
Graph attention Networks	Graph attention networks (GATs), Heterogeneous Graph Attention Network (HAN)
Graph Generative networks	Deep Generative Model of Graphs (DeepGMG), GraphRNN, Graph Variational Autoencoder (GraphVAE)

Recurrent GNNs

Recurrent neural networks are a well-known approach for modelling sequential data. Exploiting recurrent neural architectures, Recurrent GNNs aim to capture sequential and recursive patterns on graphs either on node or graph level. They assume that there is a constant exchange of information between the nodes and their neighbours until an equilibrium is reached. It should also be mentioned that recurrent GNNs apply the same set of parameters recurrently over the nodes of a graph. The most popular graph neural networks that fall under this category are:

- **Graph Neural Networks (GNNs).** In this approach graph structural information is encoded by representing each node with a low dimensional state vector. To reach a stable point, each state vector is recursively updated by exchanging information with immediate neighbors. A GNN architecture uses as inputs the adjacency matrix, the nodes' and edges' features while to ensure convergence, the recurrent function must be a contraction mapping (Scarselli, et al., 2009). A variant of this network is the Graph Echo State Network (GraphESN) extended to deal with cyclic/acyclic, directed/undirected, labeled graphs (Gallicchio & Micheli, 2010).
- **Gated graph sequence neural networks (GGS-NNs)** were proposed as an improvement over GNNs. A gated recurrent unit (GRU) is employed as the recurrent function, thus there is no need to constrain the parameters to ensure convergence. The node state is updated by its previous states and the previous neighbor states. The proposed network can also be used for sequential output problems by using several networks in sequence to produce one output each (Li, et al., 2016).

- **Stochastic Steady-state Embedding (SSE)** employs stochastic fixed-point gradient descent to speed up the training process. It samples a batch of nodes for hidden state update using local neighborhood and a batch of nodes for optimizing the parameters of the model. The hidden nodes update is performed in an asynchronous fashion, which allows the framework to handle large graphs in an effective and efficient way (Dai, et al., 2018).

Convolutional GNNs

Graph convolutional networks are currently the most popular topic in the field of geometric deep learning as they attempt to learn common structural patterns of the graphs by applying convolution similar to the way it is employed in the Euclidean space. Convolutional GNNs are divided into two categories:

- **Spectral based GNNs**, which introduce filters that perform convolution by transforming the node representation to the spectral domain using the graph Fourier transform. The difference among the various spectral based GNNs lies in the design of filters in the convolutional layers. Typical spectral-based algorithms in this category include:
    - **Spectral Convolutional Neural Network** uses the Fourier transform formula to take the product of the input in the Fourier domain first and return it to the original space with the inverse Fourier transform, which is equivalent to directly applying the convolution (Bruna, et al., 2014).
    - **Chebyshev Spectral CNN**. Chebyshev polynomial basis is used to represent the spectral CNN's filters, which simplifies a lot the filtering operation (Defferrard, et al., 2016).
    - **Graph Convolutional Networks** introduces a first-order approximation of ChebNet, which means that the convolution only considers the direct neighbors of the node (Kipf & Welling, 2017).
    - **Adaptive Graph Convolution Networks** constructs a residual graph by computing the pairwise distance between nodes, thus it can capture complement relational information and hidden structural relations (Li, et al., 2018).

- **Spatial based GNNs** that can be viewed as recurrent GNNs that perform convolutions by information propagation considering node neighborhoods. However, instead of applying the same set of parameters as recurrent GNNs do, each convolutional layer has its own set of parameters. Spatial based methods have developed rapidly due to their flexibility, efficiency and generality. Typical spatial-based GNNS include:
    - **Neural Network for Graphs** performs convolution by summing up directly the information of the node's neighbors. Each layer has independent parameters and for the convolution, the unnormalized adjacency matrix is used (Micheli, 2009).
    - **GraphSage** employs an aggregation function to define convolutions. The final state of the node is the aggregation of the neighborhood information and is invariant of the order of the nodes (Hamilton, et al., 2017).
    - **Diffusion Convolutional Neural Network** applies diffusion i.e. random walk process on the graph. The transition probability matrix of a random walk is calculated and used for the hidden node representation (Atwood & Towsley, 2015).
    - **Dual Graph Convolutional Networks** proposes the combination of two graph convolutional networks. The first network is the GCN and the second replaces the adjacency matrix with the positive pointwise mutual information matrix (PPMI) of the transition probability. The two convolutions are combined by minimizing the mean square difference of the two node representations (Zhuang & Ma, 2018).
    - **PATCHY-SAN** attempts to convert the graph-structured data to grid-structured data and use traditional CNN to learn the graph's hidden representation (Niepert, et al., 2016).

- **Message Passing Neural Network** is a general framework for spatial-based ConvGNNs. It runs message passing iterations to let the information propagate further. Essentially, each node sends messages according to its state and updates its state according to messages from the first order neighbors (Gilmer, et al., 2017).
- **Mixture Model Network** assigns weights to node's neighbors. It determines the relative position between the node and its neighbor and a weight function maps this relative position to relative weight between the two nodes. The parameters of the weight function can be shared across different locations (Monti, et al., 2017).

Spatio-temporal GNNs

Spatio-temporal GNNs have been introduced to capture the dynamicity of the graphs as many real world applications require graphs that are dynamic in terms of the graph structure and graph inputs. They model the dynamic input nodes and assume independency of the connected nodes. They capture spatial and temporal dependencies at the same time and can be used for future node values or labels forecasting, e.g. traffic speed forecasting, delay predictions. Two main categories are followed: RNN-based, and CNN-based models, as described below.

- RNN based approaches:
  - **Graph Convolutional Recurrent Network** employs graph convolutions to filter the inputs and hidden states that pass to a recurrent unit**,** in order to capture the spatio-temporal dependencies of a graph. Specifically a LSTM unit is used combined with ChebNet (Seo, et al., 2018).
  - **Diffusion Convolutional Recurrent Neural Network** suggests a diffusion graph convolutional layer to capture the spatial dependencies of the graph and a GRU network to capture the temporal dependencies. In addition, DCRNN employs an encoder-decoder framework in order to predict the K future steps of the node states (Li, et al., 2017).
  - **Structural-RNN** proposes the employment of two RNNs to predict node labels at each timestep. A node-RNN network and an edge-RNN network are used to capture the temporal information of each node and each edge respectively. The outputs of the edge-RNN are passed as inputs to the node-RNN, to take the spatial information of the graph into consideration (Jain, et al., 2016).
- CNN-based approaches:
  - **CGCN** combines 1-D convolutional layers with ChebNet or GCN layers. Specifically a 1-D convolutional layer is followed by a graph convolutional layer followed by another 1-D convolution. The input to the network is a 3-D graph matrix with time dimension. The 1-D convolutions slide over the time axis to capture the temporal information, while the graph convolution layer operates on a specific timestep of the input matrix to capture that timestep's spatial information (Yu, et al., 2017).
  - **ST-GCN** consists of an 1-D CNN layer for the temporal information aggregation followed by a partition graph convolutional layer (PGC) capturing the spatial information. PGC partitions the node's neighbors into Q groups and constructs Q different adjacency matrices. Then, GCN is applied to each group with different parameters each (Yan, et al., 2018).
  - **Graph WaveNet** learns the latent graph structures automatically by making use of the many snapshots of graph data. The graph convolutions are performed using a self-adaptive adjacency matrix, which is computed using the source and target node embeddings. The embeddings are multiplied to get the dependency weight between the nodes and thus the adjacency matrix is created (Wu, et al., 2019).

It needs to be noted that the recurrent nature of the RNN-based spatio-temporal GNNs causes time consuming iterative propagation, that slows down response to dynamic changes and issues with

explosion/vanishing of the gradients. On the other hand, CNN-based spatio-temporal neural networks require low memory and have the advantages of parallel computing, which render them suitable for dynamic applications, where responses are requested in real-time.

Graph reinforcement learning

Reinforcement learning (RL) is widely used in AI tasks that include learning from feedback and non-differentiable objectives and constraints, for example playing games. Reinforcement learning can also be applied on graphs mainly for graph generation, node prediction and knowledge graph reasoning.

- **Graph convolutional policy network (GCPN)** uses RL in order to generate molecular graphs. The graph generation is implemented as a Markov decision process of adding nodes and edges. The generative model is an RL agent, whose actions are treated like link predictions, using domain-specific and adversarial rewards. For learning the node representation of the graph, GCNs are used (You, et al., 2018).
- **Graph transformation policy network (GTPN)** attempts to predict the products of chemical reactions. An RL agent selects pairs of nodes from a molecule graph and predicts their new bonding type. Rewards are given back to the agent when the predictions are correct. A GCN is used to learn the node representations and a RNN to memorize the prediction sequence (Do, et al., 2019).

Graph Attention Model (GAM) uses RL for graph classification using random walks. The agent performs two actions: first predicts the label of the graph and following it selects the next node for the random walk. The reward is received when the graph is classified correctly (Lee, et al., 2018).

Graph adversarial methods

Adversarial networks have received a lot of interest recently in the machine learning community. They consist of two linked networks: (a) the discriminator that tries to distinguish if the samples come from the real data or not; and (b) the generator that produces fake data and tries to “fool” the discriminator. They are trained concurrently using minimax game. Graph adversarial networks are divided into two categories: the adversarial training and the adversarial attacks. The adversarial training methods are used for graph generation, embeddings enhancement, while the adversarial attacks try to “fool” the model on purpose by perturbating the input data, in order to motivate for more robust model architectures. In more detail:

- **GraphGAN** is used for enhancing the graph embeddings. The discriminator has to discriminate whether two pairs of nodes belong to the original graph or whether they are generator’s products (Wang, et al., 2018).
- **Adversarial network embedding (ANE)** is used as an additional regularization term to existing embedding networks like DeepWalk. It regards the embeddings as generator’s products and imposes a prior distribution as the real data (Dai, et al., 2018).
- **NetGAN** considers graph generation as a task of learning the distribution of biased random walks. The discriminator objective is to distinguish among random walks produced by the generator, in order to generate the graph (Bojchevski, et al., 2018).
- **Nettack** attacks node classification models (GCNs) by alternating the graph structure and the node features. Nettack tries to find the best legitimate changes that can cause the misclassification of a node. The attack occurs before training and can either influence directly the node or the surrounding nodes (Zügner, et al., 2018).
- **Adversarial attacks on graph neural networks via meta learning** tries to compromise the global performance of the model. It tries to learn the model structure by using meta-gradients, assuming that the model structure is a hyper-parameter to be learnt (Zügner & Günnemann, 2019).

Graph Attention Networks

Attention mechanisms are successfully used in sequential tasks such as machine translation and natural language processing by determining the importance of the features to the target. Graph Attention Networks, in the same way, can capture the important neighbors of a center node and adjust their weights accordingly. Graph Attention Networks that can be found in the literature include:

- **Graph attention networks (GATs)** determine the importance of each neighbor and adjust its weight accordingly, during the information aggregation of the neighborhood nodes (Velicković, et al., 2017).
- **Heterogeneous Graph Attention Network (HAN)** contains different types of nodes and edges, meaning that complex semantic information is involved, which is reflected by meta-paths. HAN attempts to learn the importance of each meta-path and at the same time selects the most informative neighbors of a center node. HAN proposes a node level attention mechanism and a semantic level attention mechanism to assign different weights to aggregating meta-path-based neighbors (Wang, et al., 2019).

Graph attention networks demonstrate potentially good interpretability for graph analysis, since they give insights about important nodes, edges and paths in the graph.

Graph Generative Networks

Graph Generative Networks (GGN) generate a new graph by learning the generative distributions of graphs. Most graph generative methods are related to specific fields such as natural language processing, knowledge graphs or molecular graph generation. Graph generative methods propose graph generation in a sequential or global manner. Sequential approaches generate graphs by proposing nodes and edges one by one, while global approaches output graphs all at once. In more detail:

- **Deep Generative Model of Graphs (DeepGMG)** iteratively adds nodes and edges to a growing graph until a specific criterion is reached. Graphs are generated by making decisions about whether to add a node or an edge, which node to add and which nodes to connect to the new node. These decisions are based on the node and graph states which are updated by a recurrent GNN (Li, et al., 2018).
- **GraphRNN** proposes the employment of two recurrent neural networks: the graph-level RNN and the edge-level RNN. The graph-level RNN adds a new node to the sequence of nodes, while the edge-level RNN generates a binary sequence indicating the connection between the newly added node to the nodes previously generated (You, et al., 2018).
- **Graph Variational Autoencoder (GraphVAE)** employs a ConvGNN as an encoder, which defines the posterior distribution and a MLP as a decoder, which defines the generative distribution. GraphVAE tries to optimize the variational lower bound to learn and output a generated graph with the adjacency matrix, node attributes and edge attributes (Simonovsky & Komodakis, 2018).

3.4.4 Using knowledge graphs to explain other models

With the growth in the use of AI models in various sectors, it is important to make them explainable and understandable to increase the confidence of the application of the model. Just as the wide application of graphs and knowledge graphs in increasing the expandability of machine learning models, other types of AI models have been gaining attention as well in order to build intelligent systems capable of explaining and exposing the solution of a problem in a human-understandable way using graphs (Lecue, 2020). AI models such as game theory (Lundberg & Lee, 2017), search and recommendation systems, Knowledge Representation and Reasoning, and Planning and Scheduling,

robotics are mentioned as the potential domains to apply XAI ((Arrieta, et al., 2020) and (Lecue, 2020)).

One of the domains that have received considerable attention in recent years is explanations in the recommender system. Most of the works regarding explainable AI in recommender systems are focused to answer the question of why a particular recommendation was taken (Confalonieri, et al., 2021). Knowledge Graphs have been shown to improve the accuracy and explainability of recommendation systems in two major ways. They not only allow to explore and discover connections between users and items (Wang, et al., 2019) that otherwise would not be possible to recognise, but also to justify these connections and hence why they are recommended.

Like other AI models, the recommendation systems can be explained in a Post-hoc approach by analysing the output of a trained recommender and make them explainable, or in a model-based approach that targets the recommendation mechanism in order to explain how the algorithm selects an item to recommend (Confalonieri, et al., 2021). Rather than common XAI approaches, there are Knowledge-based approaches that are based on exploiting external knowledge about items stored in knowledge graphs to provide explanations (Catherine, et al., s.d.). This helps in finding new user-item relationships and provides insights into how users behave in terms of their interests on different items.

Knowledge Graphs have been used to explain planning models. AI planning is to model the behaviour of an intelligent agent in choosing action sequences to complete the specific task (Blum & Furst, 1997). In this sense, graphs not are only used to explain the model itself but also to structure and organise the data used for decision making.

XAI-Plan framework presents a methodology to add explanations for the decisions made by a planner. The core idea is to allow the user to select the paths with the lowest costs among a set of alternative paths suggested by the planner. Explanations are created by allowing the user to suggest alternative actions in plans and then compare the resulting plans with the ones found by the planner (Borgo, et al., 2018). One of the first research works in this domain, Graphplan, uses graphs for increasing the explainability and understandability of the model by creating a graph to explore all the possible paths in each step of the planning (Blum & Furst, 1997). In this approach, rather than immediately start searching the actions, the algorithm instead begins by explicitly constructing a graph that encodes the planning problem and its constraints to reduce the amount of search needed.

Summarily, graph modeling is not only used for organising and structuring input data for AI models but also is being widely used to explain the underlying model as well. In the connected world of data, Knowledge Graphs are considered a new and advantageous way of increasing the explainability of AI models.

3.4.5 Advantages & limitations

In section 3.4, we examined various techniques and methods from the field of machine learning applied to Graph analysis. This analysis can be performed using traditional ML methods with satisfactory results when the requirements are low. But, as graphs have grown larger and more complex, the need for a different, reduced, representation of their structure, easily digestible by ML algorithms, have become evident. This is where graph embeddings and representation learning apply. When further analysis and predictions are required, the application of deep learning techniques and Graph Neural Networks is the state-of-the-art approach.

Although graphs can be regarded as very expressive structures, this is hardly conveyed in their computational representations, as there is a trade-off between expressiveness and efficiency for the graph embedding algorithms. Thus, the limited explainability is a common characteristic shared by most of the models described in this section and especially the more advanced non-linear ones. While the problem of explainability of machine-learning predictions has received substantial attention in

recent literature, the available methods fall short in their ability to incorporate relational information, which is the essence of graphs (Ying, et al., 2019). Some recent approaches attempt to explain GNNs by employing methods that have been applied to other neural networks and deep learning models. For instance, explainability methods designed for CNNs, such as gradient attribution, saliency maps and class activation, are extended and mapped to GNNs providing some promising results (Pope, et al., 2019; Ying, et al., 2019). Moreover, in text analysis and recommenders, Knowledge Graphs can be considered as a possible assistance in linking and explaining the results of the respective AI models. In any case, as graph explainabilty gets more popular, it is anticipated that over time more approaches will appear.

Regarding traditional machine learning and graph analysis, semi-supervised learning methods focus mostly on classification or clustering of a graph's nodes. Some of them depend on network structure such as the Girvan–Newman and Modularity Based clustering while others are based on representations of the network in latent space such as the Latent Position Cluster Model (LPCM) and Stochastic Block-Model (SBM). Latent structure models produce a representation of a graph that allow for better understanding of the data in cases where hidden structures exist (Goldenberg, et al., 2009). Besides their limited applicability, traditional ML approaches face few more issues, that require further research, at least from the machine learning perspective. Dynamic graphs have not been yet studied thoroughly, in spite of the fact that there have been some attempts to handle dynamic relationships using machine learning (Latouche & Rossi, 2015). In addition, scaling the current approaches to huge graphs is maybe the greatest challenge that the traditional methods face and one that should be tackled, as huge graphs are becoming more and more popular.

On the other hand, graph representation learning approaches aim to convert the graph data into a lower dimensional space where structural information and graph properties are highly preserved. Nodes that are "close" in the graph are embedded so as to have similar vector representations. The main difference between the various graph embedding methods lies in how they define the "closeness" between two nodes (Cai, et al., 2018). This embedding process leads to deeper understanding of what is behind the data and may provide visual insights. Additionally, the graph data are compressed and transformed into something easily digestible by task-specific ML algorithms, like classifiers or recommenders. The most common challenges encountered in this area, besides the lack of explainability, are related to scalability, the choice of dimensionality and what features to be preserved (Goyal & Ferrara, 2018). The methods examined, work well for small graphs or a small number of graphs, but most real-world networks contain a large number of nodes and edges which may pose a great challenge especially for methods that try to preserve the global properties of the graph. Additionally, finding the best dimensionality of the embedding, as well as the selection of properties the embedding should preserve, can be difficult. A higher number of dimensions may increase complexity, while a lower number may result in poor accuracy. The same happens with the selection of distance metrics and structural properties. The choice can be difficult and may affect the performance depending on the application at hand.

On the field of geometric deep learning, the approaches presented covered a wide range of the state-of-the-art algorithms. Afterall, Deep Learning on graphs has recently become one of the hottest topics in machine learning community. Since the amount of graph-structured data produced nowadays is enormous (e.g social networks), it is very tempting to try to apply deep learning techniques that have been remarkably successful in other data-rich settings. Consequently, recent publications have shown remarkable progress and impressive results in graph deep learning too.

Nevertheless, a set of open Issues has been identified in the bibliography (Cai, et al., 2018; Chami, et al., 2020) and can be summarized as follows:

- **Model depth and Scalability:** Deep neural networks rely on the existence of many subsequent layers. However, in the case of GNNs the performance of the network drops dramatically when the number of layers increase (Li, et al., 2018). In order for the GNNs to be able to scale

to large graphs, many approaches propose using sampling or clustering. In this way, however, a node might miss important neighbors and a graph might be deprived of its unique structural pattern (Wu, et al., 2021). Thus, further research is necessary.

- **Dynamic graphs:** Dynamic graphs are structures that are not static, which is very common in real life scenarios e.g. social networks graphs. On the one hand, the graph structure may evolve over time, i.e. new nodes/edges appear, others are removed, while, on the other hand, the information or attributes related to nodes/edges may be varying through time. Spatio-Temporal GNNs partially address this issue, however further research is necessary. In graph embedding, approaches that can be scalable and incremental are still evolving (Goyal, et al., 2018; Pareja, et al., 2019).
- **Information used:** Most of the current graph embedding methods rely on node and/or edge structure and usually omit patterns hidden in the global structure of a graph (e.g., paths, trees, subgraph patterns) or in the attributes that accompany the structural elements. In the literature, some works employing deep learning models can be found (Feng, et al., 2016; Shi & Weninger, 2017), but are application specific and suffer from low efficiency. Therefore, the design of non-deep learning solutions that can efficiently learn graph embeddings with a substructure and/or attribute sampling strategy, remains an open topic for research.
- **GPU support:** The graphic processing in modern GPUs involves applying operations on large matrices, so they are frequently utilized in deep learning architectures to optimize performance. This succeeds by assuming a grid-like structure of data, which can be broken down to pieces that can be processed in parallel. However, this is not the case for graphs, since graphs have arbitrary structure which cannot be regarded as a grid and cannot be directly parallelized. Thus, deep representation learning methods need to seek alternative solutions to improve processing efficiency. An example, still under research, is the Galois library[15] that supports irregular parallelism (e.g., irregular amount of work in parallel sections, irregular memory accesses and branching patterns) and claims to be suitable for graph processing.
- **Evaluation and Applications:** Most of the methods covered in the literature are typically evaluated and compared using benchmark datasets for graph embedding with sizes of up to a few thousands of nodes. Furthermore, results may vary based on dataset's training-testing split or training procedures (e.g. early stopping), as shown in (Shchur, et al., 2018). So, the need for a robust and unified evaluation procedure is evident, as is the need for using large and realistic graphs that may contain billions of nodes, which in turn will push hardware advancements in graph analysis.

3.5 Hybrid techniques

An alternative approach to overcome the trade-off between model interpretability and predictive capacity, is to synthesize both complex and transparent components under a common framework. A hybrid framework as such formulates the problem at hand as a multiple optimization task. Performance optimization is assigned to black-box components, whereas transparent components can either optimize the explanations or influence the complex model's training by transferring prior domain knowledge, to render it more comprehensible. This path holds interesting families of solutions to XAI applications, where accuracy of predictions as much as understanding of model behavior are of the essence and need to be collectively addressed from scratch, in a self-consistent manner. Hybrid methods described in this section include black-box models trained subject to logical constraints, as well as black-box and transparent models joint together into internally bound pairs or stacking ensembles.

[15] https://github.com/IntelligentSoftwareSystems/Galois

3.5.1 Fusion of domain knowledge in opaque models

The first use case of hybrid models are those methods that combine domain and background knowledge, in the form of logical statements or constraints, as Knowledge Base (KB). These types of approaches have shown improved results not only in terms of explainability but also in terms of prediction accuracy, as mentioned in (Donadello, 2017) or (Garcez, 2019). The use of these hybrid models has positive impacts on prediction tasks, for instance, by providing robustness to the learning model in the face of mislabelled training data or by offering models able to predict and reason with both symbolic and subsymbolic representations and inference, incorporating deep learning techniques by means of neural predicates in an end-to-end probabilistic logic programming language framework (Manhaeve, 2018). Another case of positive impact is presented by (Donadello, 2019), where rule-based and data-driven methods are combined to produce a dietary recommendation system.

(Doran, 2017) proposed a method in which an external KB is provided to a model and allows the system to generate its own explanations as to why it decided to predict a specific output using natural language. Since the KB is provided to the model and cannot be incorrect, an error in the output reasoning implies an error between the high-level features of the black-box model and the final output. The inclusion of the reasoning in the model eliminates the possibility of corrupting external interpretations of the outputs in order to become an explainable model. However, some adjustments were proposed by (Bennetot, 2019). The architecture of this approach can be seen in Figure 3-35 - System proposed by Bennetot 2019.. The causal links given by the KB do not directly reflect the operations that took place in the black box and, therefore, it is impossible to claim that the model outputs an outcome for the reasons extracted from the system using natural language. To mitigate this, the proposed modifications were to extract the KB directly from a first black box model to create symbolic rules and then, reflect these rules in a second black box model and constrain its learning according to the perceived properties by modifying the initialization protocols, hyperparameters and using the Confusion Loss and Confidence Loss introduced by (Burns, 2019).

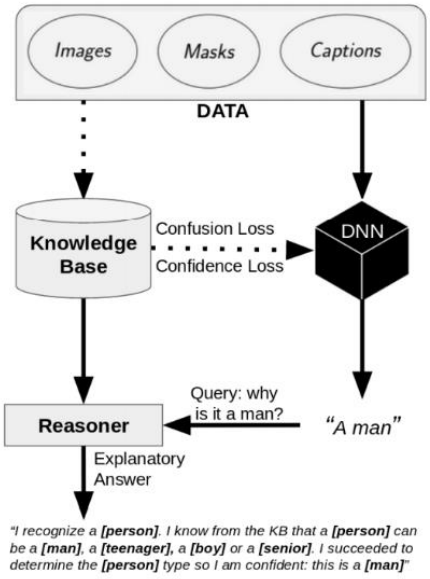

Figure 3-35 - System proposed by Bennetot 2019.

3.5.2 Coupled opaque and transparent models

Future research in the area of data fusion focuses on the deep formulation of classical ML models with high levels of transparency. These approaches provide deep models with the interpretability inherent to transparent models. Specifically, Deep k-Nearest Neighbours (DkNN), Deep Kalman Filters (DKFs), Deep Variational Bayes Filters (DVBFs) and Structured Variational AutoEncoders (SVAE) are explained in depth.

Deep Nearest Neighbors (DkNN)

Deep Learning has shown excellent results in a huge number of applications, such as malware detection (Grosse, 2017) or medical diagnosis (Caruana, 2015). Despite this, in critical or security activities, these methods remain limited due to 3 factors in terms of security: lack of reliable confidence estimates (Guo, 2017), lack of model interpretability (Lipton, 2018) and lack of robustness (Szegedy, 2014).

In order to mitigate these problems, (Papernot, 2018) proposes Deep K-Nearest Neighbours, a classification algorithm that strengthens the conformance of predictions made by a DNN, by combining this method with the inherently transparent k-Nearest Neighbors algorithm. The way this model operates is described in *Figure 3- 36*. For a test input sample, a nearest neighbour search is calculated at each layer to find the training points closest to the new sample. An analysis of the closest training points labels on each layer and the prediction label of the final output is then performed to ensure the conformity of the final output with the intermediate calculations. Thus, the purpose of DkNN models is to provide:

- Confidence. This term expresses the level of homogeneity between the nearest neighbour labels.
- Interpretability. The use of nearest neighbors at each layer provides a higher level of transparency.
- Robustness. This refers to the ability of the model to detect non-conforming predictions with input data outside the distribution, e.g., adverse images.

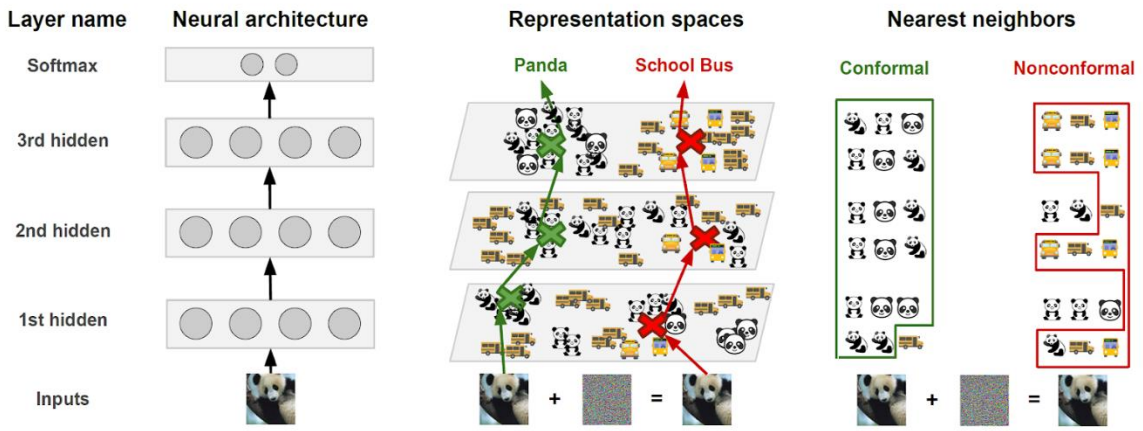

Figure 3- 36 - Intuition behind DkNN. The image on the left shows the architecture of the DNN. The middle image shows the representation obtained by each layer. Finally, the image on the right shows the nearest neighbors found on each layer. DkNN would indicate that the real panda is compliant but its adversary panda is not. Image from Papernot 2018.

Deep Kalman filters (DKFs)

Deep Kalman Filters method proposed by (Krishnan, 2015) is inspired by both recent variational methods for generative Deep Learning and Kalman Filters (Kalman, 1960). The use of Kalman Filters is widespread and they have been very successful in the last two decades, being used in a wide range of applications such as GPS, optimal control or counterfactual inference. In these filters, the latent state evolution, as well as the emission distribution and action effects, are modeled as linear functions perturbed by Gaussian noise. However, nonlinearities are present in real-world applications. That is why DKFs aim to develop a method for probabilistic generative modeling of complex observations

perturbed by nonlinear actions using Deep Neural Networks, deriving an algorithm for learning a broad class of Kalman filters and employing them for counterfactual inference and analysis of the effect of external actions.

Deep Variational Bayes Filters (DVBFs)

This method aims to overcome the limitations of classical Bayes filters, whose purpose is to infer in a state space the current latent state from the observations and the state transition. Thus, these methods estimate what the current latent state is given some observations. This estimate is the posterior distribution, called here filter distribution. Bayes filters have some weaknesses due to their strong assumptions. On the one hand, they are intractable in many highly nonlinear observations. On the other hand, the underlying process (state transition) of the state space is assumed to be known, and then the latent state is estimated by the Bayes formula on which the filter depends. Obtaining a proper state space is a difficult task that requires a significant amount of domain knowledge. (Karl, 2017) proposed a method to tackle this problem, combining Bayes filters with deep neural networks, and called that approach Deep Variational Bayes Filters. The idea behind this is to replace limiting components with deep learning methods, and extend it to accept nonlinear observations.

Structured Variational Autoencoders (SVAE)

Structured variational autoencoders are the result of combining ideas related to probabilistic graphical models and variational autoencoders (VAE). (Johnson, 2016) proposes a joint modeling and inference framework, blending these two ideas, where structured representations of probabilistic graphical models are inflexible, but VAEs are more flexible in terms of learning data representation, even though they cannot encode a probabilistic structure. SVAE aims to leverage their strengths and mitigate their weaknesses by working together. Probabilistic graphical models have demonstrated that work well on conjugate observations inference but their performance falls down when non-conjugate observations are intended to infer. To solve this problem, VAE is very useful due to it may learn how to output conjugate observations for non-conjugate entries and use those outputs as the inputs of the probabilistic graphical model. Thus, the variational generation part of the system acts as a surrogate model for non-conjugate distributions.

3.5.3 Ensemble of stacked opaque and transparent models

Another type of research focuses on combining black-box predictions, which are more accurate, with white-box predictions, which are more explainable, and stacking their predictions in some way. Some studies point in the direction of using learning representations of the hidden layer output of a convolutional neural network as input to a tree-based model and training both models in an end-to-end way, as proposed by (Biau, 2019), (Roy, 2016) or (Yang, 2018). Other research, as mentioned in (Hailesilassie, 2016), tries to extract rules for each class from the output of a CNN, establishing the behavior of one layer based on the previous layer. These techniques have been used in several areas, such as sentiment analysis (Ray, 2020) or semantic image labeling (Rota Bulo, 2014). (Kanehira, 2019) proposed a method in which 3 models were used: predictor, linguistic explainer and sample selector. The goal of this framework is to place the predictions of each object in a similarity space corresponding to the linguistic explanations to mark as valid explanations those predictions that were close in the similarity space.

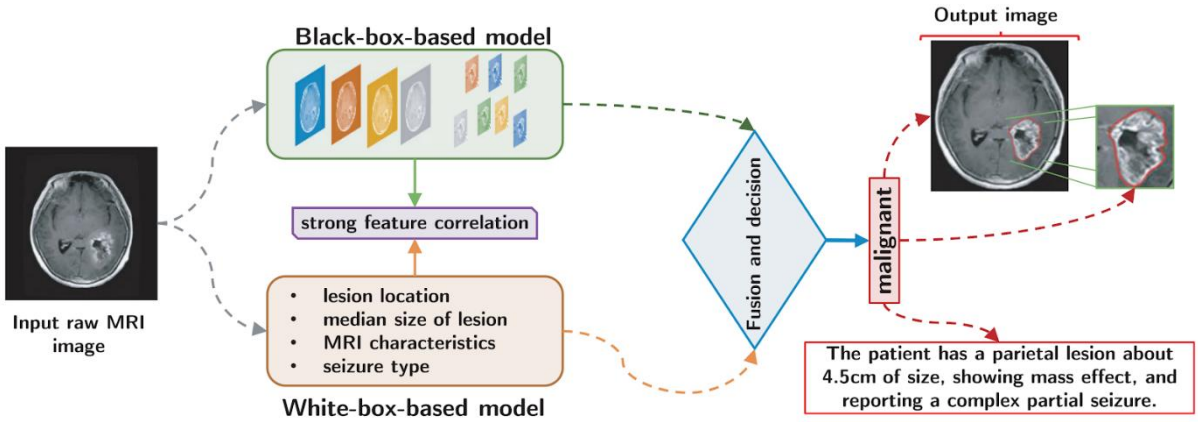

Figure 3-37 - System proposed by Loyola-Gonzalez 2019

In addition to these studies, two novelty approaches were proposed by Loyola-Gonzalez (2019). One of them (corresponding with Figure 3-37) aims to extract information (detection/prediction and relevant information in a natural language way) and feature representations from images through 2 models (one of them, black-box and one of them, white-box), both in parallel. If the feature representations of both models are correlated between them, the information provided by both models is merged, providing an output with complemented information provided by both boxes. The other approach tries to stack the information from a black box and a white box at each sequential layer, forming a new set of predictions where the stacking step deals with the difference between the outputs of both models.

3.5.4 Advantages & limitations

Hybrid models represent a middle ground between black-box and white-box models. The aim of such approaches is to mitigate the limitations of each type of model by combining their strengths. In this way, these models can be more accurate and explainable by working together rather than separately. The way they are combined leads to different families of hybrid algorithms for example those in which both approaches are internally combined or those in which each type of model works separately and then their predictions are combined to provide accuracy and explainability at the same time. Notwithstanding all these benefits, hybrid models that incorporate domain knowledge as Knowledge Base are dependent on the particular KB used in the specific area of application, being the selection of the KB the most limiting aspect of this type of models.

4 XAI Tools

In this section, we present the leading software tools that exist in the domain of explainable AI. Our focus will be given to the open-source libraries that may be used for the needs of XMANAI. For consistency, we follow the categorization of methods of the previous chapters i.e Model-specific, Model-Agnostic etc. However, we have to mention that some tools provide XAI solutions in more than one class of methods.

4.1 Explainability by design

InterpretML - https://github.com/interpretml/interpret

InterpretML (Nori *et al.*, 2019) is an open-source framework offered by Microsoft that incorporates state-of-the-art machine learning interpretability techniques under one roof. The package contains two different types of methods. Firstly, it implements a list of glass-box models by design, namely Explainable Boosting Machine (EBM), Decision Tree, Logistic Regression and Decision Rule List. These models can be trained in a dataset and provide explanations for their behaviour out-of-the-box. Furthermore, it also contains many explainability techniques, such as SHAP Kernel Explainer, SHAP Tree Explainer, LIME and Partial Dependence.

The package's community is very active and fast-evolving. Its interface is very straightforward, and it also has a wide range of examples. Though, the documentation could be a more complete with full coverage of the provided API. InterpretML can serve as a primary workhorse for the glass-box model implementation.

4.2 Post-hoc explainability techniques: Model-Agnostic

Alibi Explain - https://github.com/SeldonIO/alibi

Alibi is an open-source Python library aimed at machine learning model inspection and interpretation. The library focuses on providing high-quality implementations of black-box, white-box, local and global explanation methods for classification and regression models. Most implemented methods belong to the model-agnostic class (e.g. Accumulated Local Effects, Anchors, Counterfactuals etc.) and can be used in any black-box model. There is also a second class of explainability methods related to Deep Neural Networks (Integrated gradients, prototype counterfactuals, Contrastive Explanation Method). For the latter scenarios, the Neural Networks must be implemented in the Keras library.

Alibi is a complete open-source library covering many model-agnostic explainability methods. Its interface is very straightforward and the documentation well-written. XMANAI could widely adopt alibi.

AIX360 - https://github.com/Trusted-AI/AIX360

The AI Explainability 360 toolkit is part of IBM's Research Trusted AI project, that supports interpretability and explainability of datasets and machine learning models, including a comprehensive set of algorithms covering different dimensions of explanations and proxy explainability metrics.

The library contains a fair amount of model-agnostic explainability algorithms. One significant disadvantage is the lack of analytical documentation.

SHAP implementation - https://github.com/slundberg/shap

This library implements the SHAP method. While SHAP is a model-agnostic explainability technique, particular versions of the method are optimized for specific algorithms. For example, TreeExplainer is optimized for explaining the predictions of a tree ensemble in a very efficient way. Accordingly, DeepExplainer implements the Deep SHAP algorithm, a high-speed approximation of SHAP values in Deep Learning.

The library is very robust, and it is being used for quite a long time by the research community. This library can offer the basic underline functionality for all methods related to the SHAP approach.

LIME method implementation - https://github.com/marcotcr/lime

The library implements the LIME explainability method, which can be used for any black-box classifier. One fundamental advantage of the specific implementation is the built-in support for scikit-learn classifiers. For XMANAI, the library can be used as the backend functionality of the LIME method.

4.3 Post-hoc explainability techniques: Model-Specific

4.3.1 Tree Ensembles

InTrees – https://cran.r-project.org/package=inTrees

InTrees R package is designed to simplify tree ensembles (eg Random Forest, Regularized RF, XGBoost). The provided modules can be used to (a) extract rules from base learners, measure their quality and apply leave-one-out pruning (b) select a compact set of rules and summarize into a Simplified Tree Ensemble Learner (STEL) (c) calculate frequent feature interactions (d) extract STEL in latex format.

TreeInterpreter - https://pypi.org/project/treeinterpreter/

Library to decompose individual predictions by Random Forest models into bias and feature contributions. TreeInterpreter also allows for global feature importance attribution, by means of

Mean Decrease Impurity (MDI) and the debiased MDI, calculated on the Out-Of-Bag samples (MDI-oob).

Detailed documentation and examples. Feature contributions are considered independent (i.e. feature interactions are not accounted for).

FOCUS - https://github.com/a-lucic/focus

Python modules to produce Flexible Optimizable Counterfactual explanations for tree ensembles

TREX - https://github.com/jjbrophy47/trex

TREX (Tree-ensemble Representer-point Explanations) is a tool to identify influential training samples on individual model predictions.

4.3.2 Support Vector Machine

FERM - https://github.com/jmikko/fair_ERM

Fair Empirical Risk Minimization is an algorithm to develop fair SVM models, under the Equal Opportunity constraint (Donini 2018). The method is applicable to both linear and nonlinear kernels.

4.3.3 Deep Learning Models

Captum - https://github.com/pytorch/captum

Captum is a model interpretability and understanding library for PyTorch. It has quick integration for models built with domain-specific libraries such as Torchvision, Torchtext, and others. All implemented methods can be applied in 3 levels. In the Primary attribution level, the package evaluates contribution of each input feature to the output of a model, in the layer attribution level, the package evaluates the contribution of each neuron in a given layer to the output of the model and, finally, at the Neuron Attribution layer, the package evaluates contribution of each input feature on the activation of a particular hidden neuron. Its interface is very straightforward and easy-to-use and It has excellent documentation.

Lucid - https://github.com/tensorflow/lucid

Lucid is a collection of tools for research in neural network interpretability, compatible with Tensorflow.[16] Lucid mainly implements Feature Visualization techniques, i.e., setting up an optimization problem and searching for the input that maximizes the unit under investigation.

It ships with an extensive collection of Tutorials and Examples implemented in google Colab notebooks. These examples help understand how to use Lucid in practice, but it lacks some more official documentation. Its interface is straightforward and easy-to-use.

iNNvestigate - https://github.com/albermax/innvestigate

iNNvestigate (Alber *et al.*, 2018) provides out-of-the-box many deep learning explainability methods with a collection of examples and a clear interface. There are some compatibility issues since it is compatible with an outdated tensorflow version (1.12).

DeepExplain - https://github.com/marcoancona/DeepExplain

DeepExplain (Ancona *et al.*, 2017) is DeepExplain provides a unified framework for state-of-the-art gradient and perturbation-based attribution methods. Researchers and practitioners can use it to understand the recommended existing models better and benchmark other attribution methods. The interface seems simple, but the library is not actively updated.

4.4 Graph Machine Learning

4.5 Other tools

In this section, we present other type of tools that cannot be classified in one of the categories above.

What-if-tool - https://pair-code.github.io/what-if-tool/

What-if-tool is an online service, maintained by Google, for explaining Datasets and models. It is quite intuitive (works through an interactive GUI) and easy-to-use. It doesn't offer many explainability techniques; it mainly offers a nice interface for (a) producing pair plots, (b) perturbing a datapoint (i.e. change the value of a feature) and show the change in the prediction (definition of what-if question) and (c) some analytics on the dataset. The API can be integrated into any python model.

EUCA - https://weina.me/end-user-xai/

[16]It is only compatible with Tensorflow 1 (not Tensorflow 2 which is the current version)

End-User Centered explainable AI framework (EUCA) provides non-technical users with a suite of explanatory tools, including explanations by feature relevance (feature attributions and interactions, distribution of values), explanation by examples (similar, typical, counterfactual) as well as rule-based explanations (ruleset, Decision Tree). Supplementary information is provided on the model's overall performance, uncertainty and bias, whereas the dataset under study can be investigated both locally (i.e. by training sample, as model's input-output pair) and globally (data distribution).

Used-friendly GUI environment, detailed documentation and examples.

4.6 List with XAI tools

In the following table we present an overview of all the major open-source tools in Explainable AI landscape.

Table 4-1 - List of open-source tools that can be used by XMANAI

Tool	Link	Category	Supported Methods	Package	License
InterpretML	https://github.com/interpretml/interpret	By Design	SHAP Kernel Explainer, SHAP Tree Explainer, LIME, Partial Dependence	Python	MIT License
Alibi Explain	https://github.com/SeldonIO/alibi	Model agnostic	ALE, Anchors, CEM, Counterfactuals, Prototype Counterfactuals, Integrated Gradients, Kernel SHAP, Tree SHAP	Python	Apache License 2.0
AIX360	https://github.com/Trusted-AI/AIX360	Model Agnostic	ProtoDash, Contrastive Explanation Methods, LIME, SHAP	Python	Apache License 2.0
SHAP	https://github.com/slundberg/shap	Specific Method	SHAP	Python	MIT License
LIME	https://github.com/marcotcr/lime	Specific Method	LIME	Python	BSD 2
TreeInterpreter	https://github.com/andosa/treeinterpreter	Model specific - Trees	MDI, debiased MDI, decompose prediction	Python	BSD 3
InTree	https://cran.rproject.org/package=inTrees	Model specific - Trees	Tree Explanations	R	BSD 3
Focus	https://github.com/a-lucic/focus	Model specific - Trees	Focus	Python	-
Trex	https://github.com/jjbrophy47/trex	Model specific - Trees	TREX	Python	-
FairERM	https://github.com/jmikko/fair_ERM	Model Specific - SVM	FairERM	Python	
Captum	https://github.com/pytorch/captum	Model Specific – CNN	Integrated Gradients, Gradient SHAP, DeepLIFT, DeepLIFT SHAP, Saliency, Feature Permutation	Python/Pytorch	BSD 3
Lucid	https://github.com/tensorflow/lucid	Model Specific – CNN	Most methods described in Chapter 2.3.2.2 are implemented in Lucid	Python/Tensorflow	Apache License 2.0

Tool	Link	Category	Supported Methods	Package	License
iNNvestigate	https://github.com/albermax/innvestigate	Model Specific - Deep Learning	smoothgrad, deconvnet, DeepTaylor, LRP, integratedGradients	Python/Keras	BSD License
DeepExplain	https://github.com/marcoancona/DeepExplain	Model specific – Deep Learning	Saliency, Integrated Gradients, epsilon-LRP, DeepLift, Shapley Value sampling	Python/Keras,Tensorflow	MIT License
What-if-tool	https://pair-code.github.io/what-if-tool/	Model agnostic	-	Service	Apache License 2.0

5 XAI in Manufacturing

5.1 XAI Applications in industry

Although several AI solutions have been already applied in the manufacturing domain, the integration of reasoning components remains an open challenge. The investigation of explainable AI solutions for industrial settings is constantly gaining attention, as evidenced by the rising number of research projects on this field. In the following sections we examine XAI solutions that have been proposed to address specific tasks along the industrial workflow, including Assembling/Quality testing, Default detection, Maintenance prediction, Logistics and End-to-end chain supply.

5.1.1 Demand Planning

Demand planning is a very important aspect for manufacturing worldwide as it is closely related to warehouse management optimization. Stocking on products is an expensive business, requiring infrastructure and working force. Forecasting demand is the core AI application related to this task, which allows managers to design better strategies, make wise investments and compete effectively in the market. Demand forecasting employs machine learning algorithms to perform time-series analysis. Even though a time series analysis is hard to be interpreted, there are a few attempts in literature following this direction (Wisdom, et al., 2016; Schlegel, et al., 2019). In this line of work, XAI methods already used in image and text-domain are applied on manufacturing data, trying also to incorporate the temporal dimension.

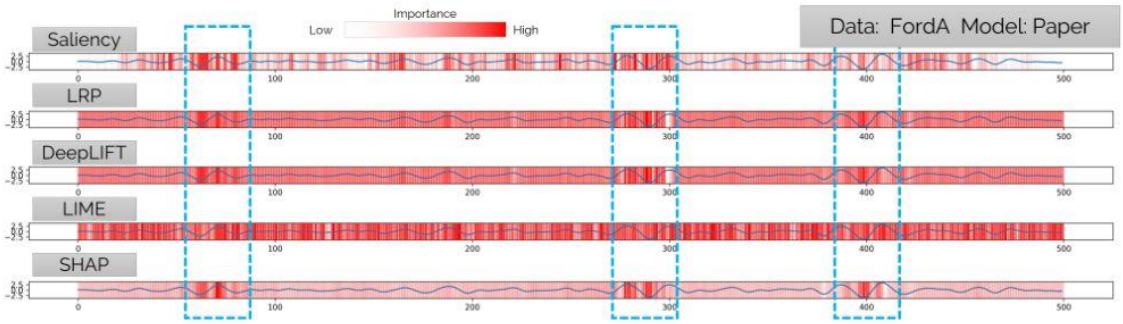

Figure 5-1 Various XAI methods with their relevance heatmaps on a time series prediction task (Schlegel, et al., 2019)

5.1.2 Product Design

AI and machine learning, nowadays, find many applications in product design and development. Well known firms, like General Motors (Danon, 2018), use generative design and state-of-the-art technology to develop better, safer and cost-effective products. In a similar fashion, industrial big data can be used to train and optimize digital twin models (Min, et al., 2019). This digital twin can be defined as an adaptive model of a complex physical system and can be the virtual testbed for various AI experiments that assist in decision making. In certain cases, it is really important to know what the basis was for making certain decisions and this is where explainable AI could play an important role (Rao & Mane, 2019). Typical explanations in this field could include feature attributions showing what factors contribute more to the final outcome, or rule-based explanations which can be especially valuable to domain experts.

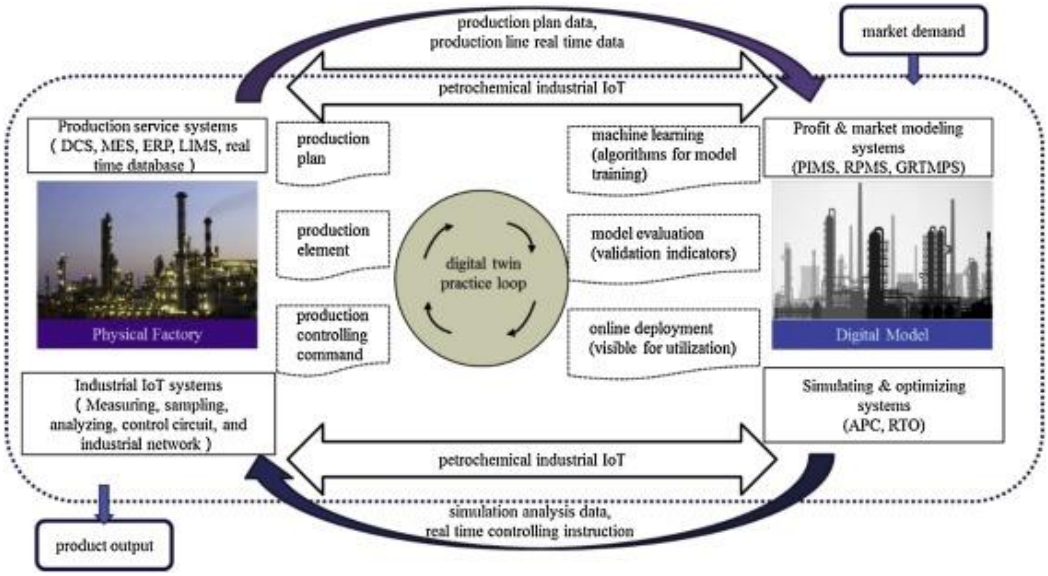

Figure 5-2 How an AI-created digital twin effects product design and development in petrochemical industry (Min, et al., 2019)

5.1.3 Inventory/ Supply Chain Management

In supply chain management, there is a wide variety of tasks that AI is already delivering unprecedented value. Some of the high impact areas in supply chain management include planning and scheduling, demand forecasting, spend analytics, and logistics network optimization (Calatayud, et al., 2019) forming a new paradigm called self-thinking supply chain (Figure 5-3 Self-thinking supply chain Figure 5-3). In addition, one can find reinforcement learning systems applied to full-inventory management (Bharti, et al., 2020) or SVM and Decision trees handling the supply chain risk management (Baryannis, et al., 2019), among others (Lingam, 2018). All of these points are closely connected to decision making, and while they demonstrate the role of AI in this field, they also underline the need for explainability and transparency in order for managers to grow trustful of AI and embrace the AI-generated solutions.

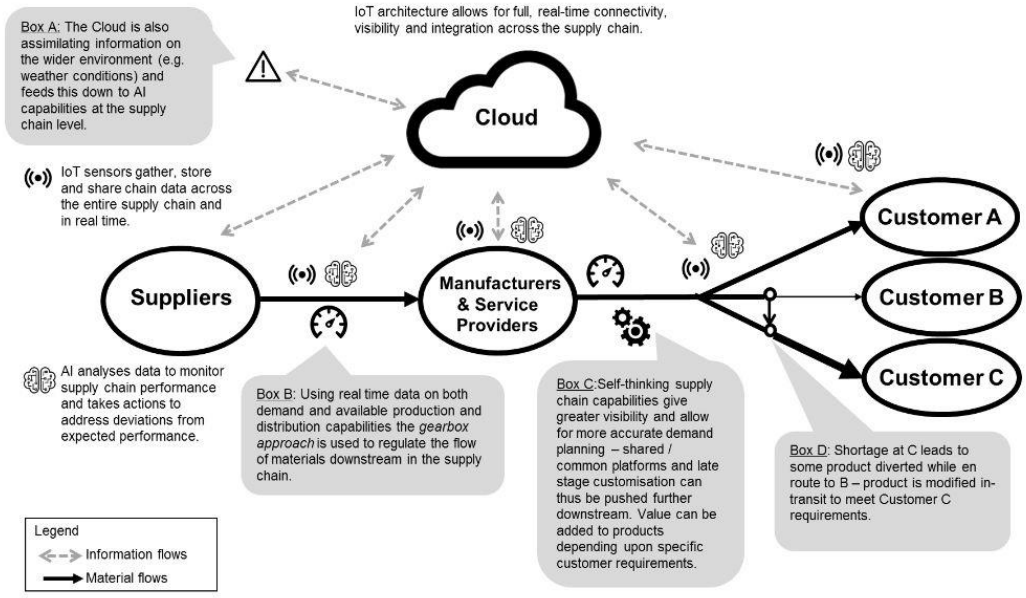

Figure 5-3 Self-thinking supply chain (Calatayud, et al., 2019)

5.1.4 Production Management

The industry today is experiencing a tremendous increase in available sensory data collected from different aspects across a whole manufacturer, which can be analyzed by various AI applications. Leveraging these new technologies at different stages of the production lifecycle, ranging from raw material management and human resources to machines' operations and facility logistics, can lead to a more efficient and "intelligent" decision making, improving not only sales, but also factory conditions or the quality of the products (Wang, et al., 2018).

In this context, explainable AI may find many applications, such as the interpretation of the predictive analytics concerning defective products (Kharal, 2020), or the anticipated downtime of the production line (Hrnjika & Softic, 2020), both as part of an improved and transparent predictive maintenance framework. Explaining industrial scheduling, which involves shared resources, route flexibility and stochastic arrivals of raw products in a dynamic environment may also fall under this category (Hu, et al., 2020).

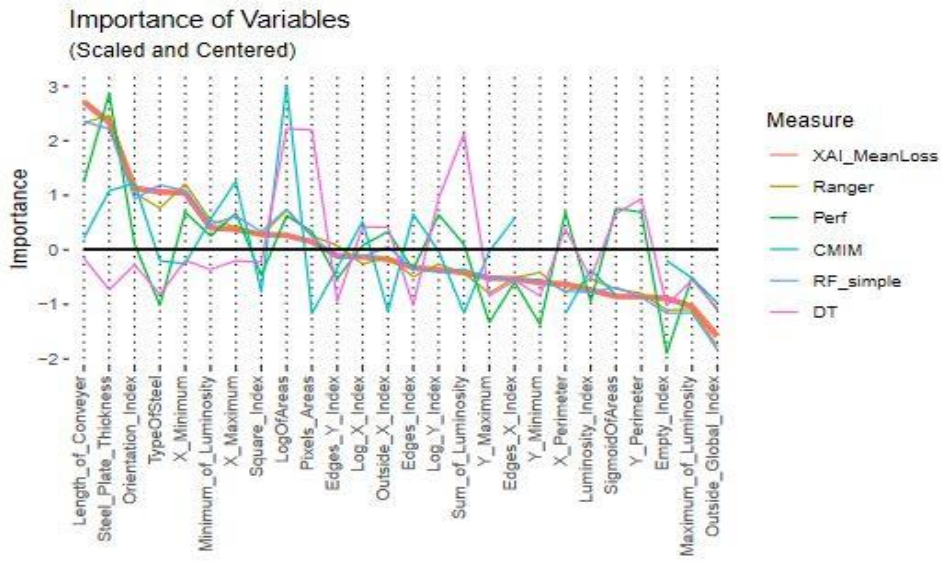

Figure 5-4 Example of explainable AI (variable importance) in defective products prediction in the Steel Plates industry (Kharal, 2020)

5.1.5 Process control

A high degree of automation and human-machine collaboration is usually employed in modern industrial settings. Processes that compose the entire length of a manufacturing production line are constantly monitored using multiple sensors and operationally controlled via multiple actuators (Figure 3-5-5) under digital twin technology (e.g. (Reimann & Sziebig, 2019)). These measurements are analyzed in real time by, among others, SVM (Doltsinis, et al., 2020), RF (Wang, et al., 2018), CNN and LSTM (Chen, et al., 2021) or reinforcement learning (Kuhnle, et al., 2021) models to assess the state of each sub-system undertaking a certain process. Given this set of current system parameters, the model predicts the next optimal state and fires the proper commands to the actuators for the system to adapt accordingly. In this way, the process is optimized with respect to quality, cost, efficiency and resource allocation (Arinez, et al., 2020). Therefore, support for human-machine interaction is crucial for AI solutions to be successfully integrated in such a complex and dynamic industrial environment. The operational framework controlling the workflow should include a friendly interface for the operator to monitor the process and comprehend the actions decided by AI components using explainability techniques, while being able to intervene manually when necessary

(Cohen & Singer, 2021). A hybrid DL and Case Based Reasoning (CBR) approach might also pose a successful solution for industrial process monitoring and control, as proposed in the DeepKAF framework (Amin, et al., 2020).

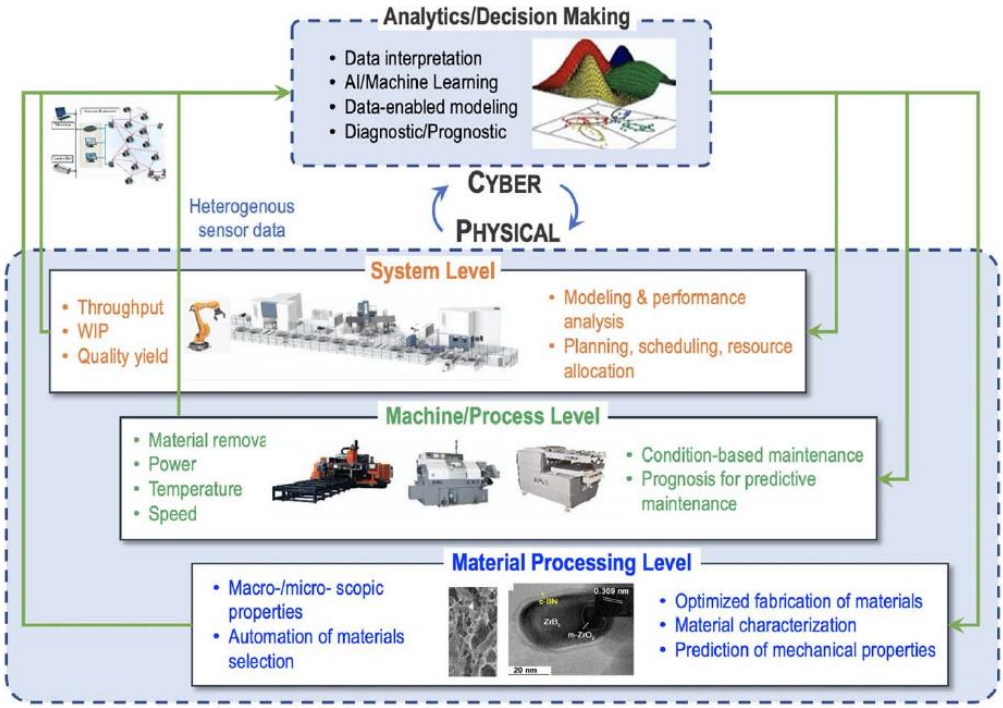

Figure 3-5-5. A cyber-physical system formed by the components of AI aided, smart manufacturing (Arinez, et al., 2020).

5.1.6 Quality control

Industrial product quality is closely inspected along the production and assembly line, from the inflow of raw material to the outflow of the final product. Deep NN models have been found to perform very well in default identification via computer vision, due to their ability to detect anomalies invisible or hardly visible to the naked eye (Wang, et al., 2018). Explanations in the form of e.g. saliency maps can enable operators to supervise quality inspection. Tree-based ensembles on the other hand have been often found to outperform other methods on sensory data, as in (Peres, et al., 2019). XGBoost and Random Forest (RF) models are applied to multiple-stage quality control in Volkswagen AutoEuropa assembly line, although the authors do not currently propose any form of explanations, to assist human operators in supervising the process. In contrast, the work by (Kharal, 2020) on steel plate manufacturing fault diagnosis exploits several techniques to explain the best performing RF model, including RF variable importance, PDP, ICE and global simplification by rule extraction. The contribution to anomaly detection in refrigerator quality control by (Carletti, et al., 2019) is also focused on explainability, as the means to root cause analysis. They develop Depth-based Isolation Forest Feature Importance (DIFFI) as a novel, model-specific feature attribution method for Isolation Forest (IF), a popular algorithm handling unsupervised outlier detection.

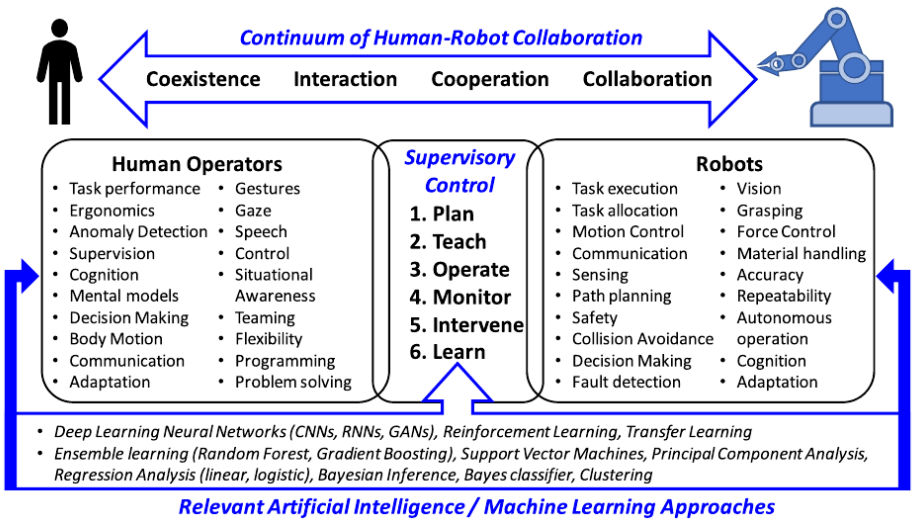

Figure 3-5-6. The spectrum of human-machine collaboration in a supervisory control operational framework (Arinez, et al., 2020).

5.1.7 Maintainance

The constant feedback on system diagnostics and machinery conditions in modern industrial settings, along with historical data on machine failures, provides manufacturers with the opportunity to operate under condition-based and predictive maintenance. As opposed to scheduled maintenance, condition-based maintenance is based on real-time assessment of tool bearing or remaining useful life (Lee, et al., 2019). Using archived data, ML models can be applied to predict machine failures before they occur (eg (Luo, et al., 2021)). This information allows engineers to act on time and avoid extended damage, minimizing thus hazard to human workers as well as unplanned machine downtime. AI solutions have been employed to this end, additionally contributing to efficient cost, energy and waste management, as Capgemini Institute researchers report (Capgemini Research Institute, 2019). The survey on industrial predictive maintenance by (Ran, et al., 2019) indicates that a wide spectrum of knowledge-based, ML and DL methods have been exploited in this application field, while reinforcement learning is also frequently utilized. The significance of including a reasoning component to an ML-driven prescriptive maintenance decision support framework is pointed out (eg (Ansari, et al., 2019), (Stohr & O'Rourke, 2021)). Explanations enable experienced operators to construct a mental view of the model and relate machine fault predictions to the underlying physics, rendering the system more reliable to the human operator (Grezmak, et al., 2020). As demonstrated by (Chen & Lee, 2020), a suite of post-hoc explainability techniques can be synthesized to explain high performing, “black box” models such as deep CNN handling machine bearing fault diagnosis. The authors justify model decisions offering attention-based explanations by gradient class activation mapping, as well as global simplification by Decision Tree, simple NN and ANFIS (Adaptive Network Fuzzy Inference System), supporting thus operators towards an informed, timely machine maintenance.

5.2 Use cases – projects

After conducting extensive research on Cordis (https://cordis.europa.eu/) and other relevant EU databases, there is no a closed EU-funded research project in which explainable AI is applied on the manufacturing sector. Together with XMANAI, a number of Horizon 2020 projects (such as STAR, COALA, TEAMING AI, MAS4AI, etc) that fall under the ICT-38-2020 topic explore different AI explainability aspects, but they have been excluded from the following analysis since liaisons and collaborations have been already established (as presented in the XMANAI Deliverable D8.1

"Dissemination, Communication and Stakeholder Engagement Plan"). Table 5-1 includes the latest projects with their short description that AI was applied in manufacturing to develop novel manufacturing techniques as well as overhauling the interaction between human workers and automated tools.

Table 5-1 - List and description of the newest AI manufacturing projects

AI manufacturing projects		
Title	**Funding scheme**	**Description**
Regions and DIHs alliance for AI-driven digital transformation of European Manufacturing SMEs (AI REGIO)	(https://cordis.europa.eu/project/id/952003) Grant agreement ID: 952003 H2020-EU.2.1.1. (Start date: 01/10/2020 End date: 30/11/2023)	The AI REGIO project aims at filling 3 major gaps currently preventing AI-driven DIHs from implementing fully effective digital transformation pathways for their Manufacturing SMEs: at policy level the Regional vs. EU gap; at technological level the Digital Manufacturing vs. Innovation Collaboration Platform gap; at business level the Innovative AI (Industry 5.0) vs Industry 4.0 gap.
Artificial Intelligence for Digitizing Industry (AI4DI)	(https://cordis.europa.eu/project/id/826060) Grant agreement ID: 826060 H2020-EU.2.1.1.7. (Start date: 01/05/2019 End date: 31/05/2022)	The goal of this project was to transfer machine learning (ML) and AI from the cloud to the digitizing industry. A seven-key-target approach was used to evaluate and improve its relevance within the industry. Finally, the project aims to connect factories, processes and devices within the digitized industry by utilizing ML and AI.
VerSatilE plug-and-play platform enabling remote pREdictive maintenance (SERENA)	(cordis.europa.eu/project/id/767561) Grant agreement ID: 767561 H2020-EU.2.1.5.1. (Start date: 01/10/2017 End date: 31/03/2021)	This project aims to explore the potential of using AI-based tools for optimizing the production process in terms of more efficient maintenance covering covered the requirements for versatility, transferability, remote monitoring and control. Moreover, a powerful platform to aid manufacturers in easing their maintenance burdens was presented which was applied in different applications. Specifically, SERENA project was focused on advancing the TRL of the existing developments into levels TRL5 to TRL7.

	Grant agreement ID: 723764	(https://cordis.europa.eu/project/id/723764)
aGent Oriented Zero Defect Multi-stage mANufacturing (GO0D MAN)	H2020-EU.2.1.5.1. (Start date: 01/10/2016 End date: 30/11/2019)	The main objective of this project was to integrate and combine process and quality control for a multi–stage manufacturing production into a distributed system architecture built on agent-based Cyber-Physical Systems (CPS) and smart inspection tools designed to support Zero-Defect Manufacturing (ZDM) strategies.
	Grant agreement ID: 835614	(https://cordis.europa.eu/project/id/835614)
Disrupting industrial robots with AI software - how new AI-driven software can be used to program industrial robots to do new tasks faster, resulting in massive productivity savings (OS for Ind robots)	H2020-EU.3. H2020-EU.2.3. H2020-EU.2.1. (Start date: 01/01/2019 End date: 30/09/2019)	The Cambrian Intelligence company specializing in AI-guided robots for manufacturing was funded for the development of a software focusing on the Ind robots project. The proposed retrofittable AI model has made robots to learn (without supervision) a new task in a simulated environment that was running in the cloud. After the training, the AI model could be run locally and guide the robots.
	Grant agreement ID: 723277	(https://cordis.europa.eu/project/id/723277) AI solutions developed through this project to enable workers to have more influence over their work, and to take greater responsibility for their own learning and skills development. Final solutions were piloted in industrial environments, and demonstrated the positive impact that AI can have on both productivity and worker well-being.
Empowering and participatory adaptation of factory automation to fit for workers (Factory2Fit)	H2020-EU.2.1.5.1. (Start date: 01/10/2016 End date: 30/09/2019)	
	Grant agreement ID: 780732	(https://cordis.europa.eu/project/id/780732) BOOST 4.0 addressed the the need for development of large scale experimentation and demonstration of data-driven “connected smart” Factories 4.0, to retain European manufacturing competitiveness by explaining in a measurable and replicable way, an open standardised and transformative shared data-driven Factory 4.0 model through 10 lighthouse factories. Moreover, this project presented ways for the European industry to build unique strategies and competitive advantages through big data across all phases of
Big Data Value Spaces for COmpetitiveness of European COnnected Smart FacTories 4.0 (Boost 4.0)	H2020-EU.2.1.1. (Start date: 01/01/2018 End date: 31/12/2020)	

		product and process lifecycle building upon the connected smart Factory 4.0 model to meet the Industry 4.0 challenges.
Digital Reality in Zero Defect Manufacturing (Qu4lity)	Grant agreement ID: 825030 H2020-EU.2.1.1. Start date: 01/01/2019 End date: 31/03/2022	(https://cordis.europa.eu/project/id/825030) QU4LITY will demonstrate, measurable, and replicable way an open, certifiable and highly standardised, SME-friendly and transformative shared data-driven ZDM product and service model for Factory 4.0 through 5 strategic ZDM plug & control lighthouse equipment pilots and 9 production lighthouse facility pilots. The main goal of this project is to build an autonomous quality model to meet the Industry 4.0 ZDM challenges.
UNIFIED PREDICTIVE MAINTENANCE SYSTEM (UPTIME)	Grant agreement ID: 768634 H2020-EU.2.1.5.1. Start date: 01/09/2017 End date: 28/12/2021	(https://cordis.europa.eu/project/id/768634) The goal of UPTIME project was to design a unified predictive maintenance framework and an associated unified information system in order to enable the predictive maintenance strategy implementation in manufacturing industries. The UPTIME predictive maintenance system extended and unified the new digital, e-maintenance services and tools and incorporated information from heterogeneous data sources, to more accurately estimate the process performances.
Scalable online machine learning for predictive analytics and real-time interactive visualization (PROTEUS)	Grant agreement ID: 687691 H2020-EU.2.1.1. (Start date: 1 December 2015 End date: 30 November 2021)	(https://cordis.europa.eu/project/id/687691) Ready-to-use scalable online machine learning algorithms and interactive visualization techniques for real-time predictive analytics to deal with extremely large data sets and data streams, aimed to cover industrial use cases, with real-world industrial validation. The main goal of the project is to reduce the gap and dependency from the US technology, empowering the EU Big Data industry through the enrichment of the EU platform Apache Flink.
Multimodal spectral sensors and orchestrated deep models for integrated process optimisation (MULTIPLE)	Grant agreement ID: 871783 H2020-EU.2.1.1. (Start date: 1 December 2019	(https://cordis.europa.eu/project/id/871783) Development of multimodal monitoring systems to deliver cost-effective spectrometers and camera cores in a broad VIS/SWIR range,

	End date: 30 November 2022)	complemented with cost effective laser-based chemometric sensors in the MWIR. Their solutions combine cloud, big data, and deep learning for agile development and orchestration of complex AI-based models to optimise production improving EU manufacturing competitiveness.
Machine learning to augment shared knowledge in federated privacy-preserving scenarios (MUSKETEER)	Grant agreement ID: 824988 H2020-EU.2.1.1. (Start date: 1 December 2018 End date: 30 November 2021)	(https://cordis.europa.eu/project/id/824988) Creation of a validated, federated, privacy-preserving machine learning platform tested on industrial data that is inter-operable, scalable and efficient enough to be deployed in real use cases. MUSKETEER aims to alleviate data sharing barriers by providing secure, scalable and privacy-preserving analytics over decentralized datasets using machine learning. Data can continue to be stored in different locations with different privacy constraints, but shared securely.

6 Human aspects in AI Decision Making

6.1 Human Centric decision-making

Since decision making is an activity performed everyday by everyone, in several different situations, during the years the literature has been enriched with a huge number of papers, studies and analysis regarding this topic.

What clearly emerges is that the perfect solution to make decisions doesn't exist, but the choice of methods, tools, techniques and strategies strongly depends on the involved actors and the situation to be faced.

In this document, we focus on "**structured decisions**", results of a complex process usually involving and impacting more than an actor, with business and economic implications. It is true that decisions are made also regarding personal and everyday life situations, where often they are driven by psychological, emotional and irrational factors but since it is far from the aim of the XMANAI project, we omit this type of analysis.

6.1.1 Internal and external factors that influence decisions

Generally speaking, it is possible to describe the decision making process according to following steps:

a) **Identification of the problem** or of the situation that presents more options to be selected.
b) **Collection of the available information and evaluation criteria development**, to deeply analyse the problem and identify constraints in applying certain solutions. Information sources may be of different nature, for instance the company database, an internal assessment, market research reports, external consultants analysis.
c) **Identification of alternatives**, evaluating expected results while adopting each of them.
d) **Choice of the most suitable alternative** and implementation deriving from their adoption.

Taking into account mainly point b), it is important to underline that according to the situation and the scenario, a bigger or smaller number of information may be available. Sometimes, for example, it may be difficult to collect all information required to proceed with a rational and scientific approach and room to a more intuitive and creative strategy must be made.

Moreover, although from a theoretical point of view the best choice is the one that maximizes the decision maker's **utility function,** the experience teaches us that the context and the situation strongly affect the decision.

Factors that may influence the final choice are several and of different nature, starting from the **emotional status** of the decision maker (which is an aspect totally unforeseeable) to **bias, experiences** and **personal opinions** that accompany each human being. For instance, during the phase of information selection, criteria applied to evaluate what is relevant and what is not may change according to the person who is making the choice. And of course, discard a relevant parameter can potentially influence the final decision.

A substantial part of the literature on decision-making has attended to cognitive biases of decision-makers. Cognitive biases are observer effects (e.g. statistical errors, social attribution errors and memory errors) common in all humans that skew the reliability of other evidence (Rahman & De Feis, 2009). Decision-makers may become more susceptible to cognitive biases in the face of **additional complexity** and **time pressure**. In (Kocher & Sutter, 2006) for instance, the authors contend that complexity and time pressure would make a different decision-making model and methods.

Complexity is often related to uncertainty and instability of potential outcomes, but also the increasing number of variables (and consequently the growing amount of interdependencies among various inputs) increases the level of complexity (Rahman & De Feis, 2009).

Time pressure could also impact decision-making**:** when time is short, the decision maker is forced to apply changes in the strategy and to neglect the most time-consuming activities, for example a proper discussion among participants or an accurate data analysis. To reduce the amount of information to be managed, usually data are further filtered, often driven by own biases (Svenson & Edland, 1993). Moreover, as it is proven in (Dhar, et al., 2000), time pressure makes emphasize the positive aspects of an alternative and reduces the weight given to disadvantages.

On the other side, (Ariely & Zakay, 2001) put in light that sometimes consequences of time-stress are not necessarily negative since it helps decision makers to overcome procrastination which can paralyse them from taking action when the choice set is large. However, the largest part of psychological studies dealing with this topic agrees that time-stress on decision making drives to a wrong judgment and evaluation.

Anyway, the combination of levels of complexity and time pressure may give rise to different methods that favour some factors with respects to others: for example, the rational model with low complexity and time pressure; the incremental model when the complexity is high, the boundedly-rational model when the time pressure is high and the garbage can model when both are high. (Rahman & De Feis, 2009).

6.1.2 Existing methods and approaches

As already mentioned in the previous section, decisions can be made using different approaches, all equally valid according to the situation.

In literature, there exist several studies aimed at classifying the typology of the decision maker, that of course may depend on the scenario. One of the most known is the "**Vroom Yetton Jago**" **decision model** ((Vroom & Yetton, 1973) and (Vroom & Jago, 1988)): to be implemented after the choice has been made, it consists of a list of questions (in a tree shape) about the nature of the problem and about decisions that have been made. The final model drives towards the identification of the leader profile in that specific situation, as autocratic, consultative or collaborative. Indeed, decisions can be made **individually** (when there is only one actor involved in the process, maybe supported by AI tools or helped by collaborators to collect information) or **in** a **group** (when the leader is assisted in decisions or decisions are made collectively).

With the second approach, different possible perspectives and experiences are evaluated, with the benefit of taking into account the needs of several people and, maybe, of reaching a **wider consensus**. But clearly, a group decision making process requires more time and a structured organization, such us **brainstorming workshops**, **meetings**, **questionnaires** or **Delphi techniques** (depending on the level of collaboration).

One of the easiest methods to choose among alternatives is evaluating the **pros and cons** of each, applying a specific weight and prioritizing them. In this context, the **decision matrices**, displaying on the rows potential options to be evaluated and on the columns factors and criteria to be taken into account, are very effective but easy to use.

Decision trees, derived directly from game theory, that allow to develop classification algorithms to support decisions, are widely spread among decision makers, as they can be easily automated and outputs are easily explainable (as also discussed in Section 3.1.3). Every node of the tree is a possible consequence of the action described in the father node and each leaf represents a classification rule.

Influence diagrams used mainly as a visual tool, supporting team decisions, may be an alternative to decision trees, that typically suffer from exponential growth. An influence diagram is built taking into account three fundamental elements: the decision to be made, uncertainty aspects of the model and at least a utility function, defined to evaluate and weight choices.

6.1.3 A focus on decision making in manufacturing

In the manufacturing domain, decisions are made for different purposes: some are common to all domains, such as those related to human resources, business and sales; instead, others are peculiar of the industrial environment, mainly when concerning machinery and equipment, warehouse management, plant fault detection. In a continuous process, it is often required to make decisions on a **weekly and daily basis**, or even **real time**, for example in case of plant breakdown.

Moreover, decisions can be classified into **strategic, tactical** and **operational**: for instance, the choice of the warehouse layout or the selection of the equipment (typically long-term decision) are classified as strategic; the definition of the resources dimension (typically medium-term decision) is classified as tactical; finally, shifts planning or batch formation (short-term) are classified as operational.

The four steps path depicted in previous paragraph (problem identification, collection of information, alternatives identification and final choice) is a general description of any decision process and so, it is valid also in the manufacturing domain. What is peculiar here are the tools used in support of the process, used for instance to collect, store and present data or to provide information required to understand the problem.

Enterprise resource planning (ERP) software are widely spread among enterprises and represent an efficient collector of information, from where the management can start to make business decisions and planning activities. Most common and modern solutions, besides integration, collection and storage tasks, cover more functions and roles, including decision making support. As (Holsapple & Sena, 2003) put in light, even if the ERP software do not make decision autonomously, they make available an integrated set of data that relieves decision makers from sorting out information, provide evidence in support of decisions, enhance communication among participant involved in a joint-decision.

Moving from planning to operational level, decisions are often made leveraging information provided by the **Manufacturing execution system (MES)**, that is a real time system to track and document the production activities, monitoring inputs, machines, and materials. The MES is designed to show in real-time which tasks have been completed and which are coming next, creating notifications and logs for issues and this represents an optimal starting point to make decisions effectively.

The **Digital Twin** is another solution emerging in manufacturing that does not play the role of decision maker but is able to enhance decision making capabilities. Since the Digital Twin is the digital representation of a physical object, it allows to test alternatives and to include simulations in the set of available information.

The adoption of a Digital Twin is not so common as the ERP and MES system, since it is a more recent solution that requires the support of various advanced technologies to be implemented (such as machine learning and distributed computing).

However, according to the World Manufacturing Forum 2020[17], the worldwide manufacturing sector investment in AI software, hardware and services increases from $2.9 billion in 2018 to $13.2 billion in 2025 (projection). Artificial Intelligence has several applications, from improving the product quality to reducing downtime of a plant (and consequently costs), from demand forecasting to efficiency improvement as also discussed in Section 5.1. Hence, directly or indirectly, also decision-making activities are impacted by the AI adoption.

As Figure 6-1 shows, in 2022 machines are expected to cover the 28% of effort required for decisions making (versus a 19% of 2018), demonstrating that manufacturing investments in AI development are also made also to provide support to decision makers.

[17] https://worldmanufacturing.org/report/report-2020/

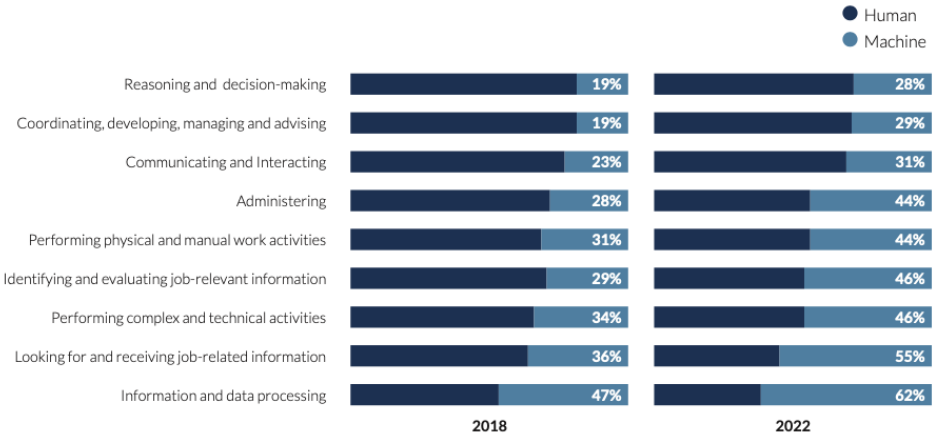

Figure 6-1 - Ratio of human-machine working hours, 2018 vs. 2022 (projected) - [Source: Future of Jobs Survey 2018, World Economic Forum]

6.2 AI in decision-making, from a user perspective

As massive volumes of data are used, decisions relying solely on human intuition is inefficient, inconsistent, fallible and limits the ability of the organization (Colson, 2019). For this reason, Artificial Intelligence (AI) systems are increasingly moving out of the laboratory affecting the real world and consequently people's lives (Guidotti et al., 2018). In general, AI involves into the workflow as a primary processor of data since i) it is less prone to human's cognitive bias, ii) it is more efficient to work with nonlinear relationships, exponential, power laws, geometric series, binomial distributions, or otherwise and iii) it can be trained to find segments in the population that best explain the variance even if they are unintuitive to our human perceptions (Colson, 2019).

However, as AI is involved in our reality, concerns are raised about how these systems make their decisions (Guidotti et al., 2018). Mainly, in the context of AI, the input-output relationships are complex indicating that they cannot be interpreted by examining the code but instead allows the system to build a model that encodes those relationships (Mohammed et al., 2017). Therefore, in such black-box systems, there is no knowledge of why any particular input produces its corresponding output. According to Peters (Peters, 2011) explanations of AI decision-making systems may also be important for user satisfaction and acceptance in general. Therefore, by leveraging both AI and humans' better decisions and knowledge representation can be taken since humans do not interact directly with data but with the possibilities produced by AI's processing of the data (Colson, 2019).

6.2.1 AI on data analysis

In recent years, incredible amounts of complex data which are characterized by a plethora of elements or variables and unstructured data such as texts, images, time series are growing exponentially (Jarrahi, 2018). Moreover, the availability of very high processing power (i.e., GPU's), incorporated with the industrial interest, has generated new AI methods, like deep and reinforcement learning, which infer meaningful information from large scale (heterogeneous) data in a very short time, solving highly complex problems and opening new perspectives. Therefore, we "traverse" the era of artificial intelligence or cognitive technologies also known as the era of Analytics 4.0 (Danenport, 2018) in which not only AI methods are used but also the automated machine learning methods are used to execute automatically the AI methods. Generally, machine learning is at the core of many approaches to artificial intelligence, and is also referred to as "predictive analytics" (Siegel, 2016).

In general, any manufacturing process that handles high-volumes of data can gain from AI (Davenport, 2018). AI with superior quantitative, computational, and analytical capabilities has outperformed humans in complex tasks (Jarrahi, 2018) providing dynamic and continuous analysis that improves

over time. As a consequence, AI can assist human decision-makers with predictive analytics as they can create new insights through probabilities and approaches to statistical inferences based on data. This signifies that AI surpasses at finding insights and patterns in large datasets that humans are not able to detect. It also does this at scale and at speed. Thus, AI has the ability to unify data across platforms signifying that it is capable not only to concentrate dynamical data into unified views but also combines data across different sources, that humans are hard to follow and comprehend.

6.2.2 AI on knowledge representation

Regardless of the scientific area, often, experts and decision-makers confront the inability or weakness to effectively and efficiently describe a data-driven problem since they are not capable to study the parameters of the features of the problem in order to make the proper decisions (Longo, et al., 2021). This problem occurs since many interconnected parameters describe the underlying system (Christoforou & Andreou, 2017) which are almost impossible to be identified from decision-makers. According to (Davis, 2015) the knowledge representation plays a central role in the artificial intelligence and it studies "… how the beliefs, intentions, and value judgments of an intelligent agent can be expressed in a transparent, symbolic notation suitable for automated reasoning", that capture information about the world that can be used to solve complex problems. However, the automatic construction of knowledge representations is a challenging process since machines find it difficult to interpret all types of knowledge.

Generally, there are many different ways for representing the knowledge or the patterns that can be detected by AI, and each one dictates the kind of technique that can be used to infer that output structure from data (Jarrahi, 2018). For instance, AI can assist human decision-makers to identify causal relationships among the observed variables/features and affirm the appropriate cause-effect connections among many possibilities through causal loops. This kind of knowledge is known as knowledge graphs, which provide a process for mimicking the implicit functions of the human brain combining it with the computing power of machines to represent meaningful flows by putting data into a framework, similar to the way humans connect pieces of information to reach a conclusion. In this way, AI facilitates decision-makers to collect effectively and act upon new sets of information that, under other conditions, humans could not be able to observe.

To sum up, knowledge representation in AI is going to be an evolving field in which researchers believe that it will provide an integrated system that has a perception and reasoning very close to humans in order to facilitate decision-makers to make better judgments/predictions.

6.2.3 AI on decision making

Another important aspect of AI concerns the decision making which is a process of evaluating optimal choices under conditions of uncertainty and attempting to mimic the reasoning of humans (Muenning, 2008). Uncertainty is characterized as a lack of information about all alternatives or their consequences, which makes interpreting a situation and taking a decision more difficult (Choo, 1991). According to Schoemaker and Russo (Schoemaker and Russo, 2018) the definition of the decision-making is determined as "… the process whereby an individual, group or organization reaches conclusions about what future actions to pursue given a set of objectives and limits on available resources. This process will be often iterative, involving issue-framing, intelligence-gathering, coming to conclusions and learning from experience". (Maria, 1997). (Craiger, et al., 1996)

As it was mentioned above, in such complex environments of high volume of data, humans face difficulties to comprehend and make proper decisions. Nevertheless, the benefit of the appearance of AI has driven the decision-making process into an outstanding level by facilitating the machine to learn from raw data itself and broaden by integrating larger data sets providing faster and higher decision quality (Jarrahi, 2018). In other words, AI-based decision-making systems do not rely on human preconceived notions and demonstrate a better representation of a problem since AI take decisions in a faster, autonomous, unbiased and rational way due to ML techniques/algorithms

(Dejoux & Léon, 2018). Another problem that decision-makers are dealing with is the equivocality which refers to the situation in which several simultaneous but divergent interpretations of a decision domain are occurred (Weick & Roberts, 1993). Equivocality often emerges because of the conflicting interests of stakeholders and decision-makers. Under this condition, the humans' decision-making is modified from an impartial and objective process into a political and subjective process that seeks to meet the conflicting needs and goals of various decisions. Therefore, it reflects a negative impact to even the most elaborate rational decision since the procedure can be stopped when the interests are influenced by the intended and unforeseen consequences of a decision. AI can provide some useful utilities that facilitate the process of decision-making to overcome equivocal conditions and handle contradictory situations for successfully making, negotiating, and implementing decisions. Even if artificial intelligence techniques can assess the best optimal outcome, the responsibility of final decision to handle the equivocality is primarily on humans' hands.

Consequently, making this process of AI technologies transparent, the confidence and the interaction of decision-makers with these technologies become more effective. Thus, human-AI synergy provides opportunities for humans to support analytical skills (Davenport & Kirby, 2016). The benefit of AI and human combination can be outstanding since AI can address the complexity of collecting and analyzing large amounts of data faster than humans; while, decision-makers can infer a more holistic and intuitive approach by processing uncertainty and equivocality in decision-making. According to Reid Hoffman, the former executive chairman of LinkedIn, the human decision-making has been improved due to AI systems which can manage large amounts of data, highlighting the most important patterns or features, that data analysts or managers can analyze in depth, using human intelligence to reach significant outcomes and take actions (Hoffman, 2016).

To conclude, the synergistic relationship between AI and decision-makers is focused on three main characteristics the uncertainty, the complexity and the equivocality (Jarrahi, 2018). Figure 6-2 presents the collaboration of the partnership of human and AI at each characteristic to deal with different aspects of decision making. Particularly, AI confronts issues of complexity utilizing analytical approaches; while, decision-makers tackle issues concerning more uncertainty and equivocality using creative and intuitive approaches. Finally, in most complex decisions, AI can provide the optimal solution but for the sake of completeness, it is highly required the human involvement to handle the uncertainty and the equivocality.

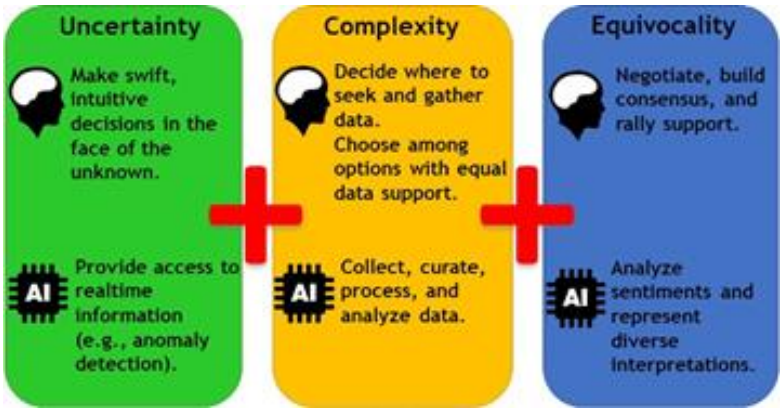

Figure 6-2 - A synergetic relationship of humans and AI to tackle complex decision-making situations distinguished by three main aspects: the uncertainty, the complexity and the equivocality (Jarrahi, 2018; page 7).

6.3 The Collaborative Intelligence Human-Machine interaction model

Exploring the process of decision-making in manufacturing domain and the possible benefits that can derive from the adoption of AI, it is of fundamental importance to dwell on the concept of "**Collaborative Intelligence**" (with a glance on Industry 5.0), that helps to define requirements for the AI-human relationship of the future.

Indeed, according to the new industrial vision, humans and machines interaction should be based on a collaborative relation where the two actors are able to help and support each other: this perspective may represent an interesting starting point also when talking of approaching decision making with AI.

Actually, as of now, the concept of human-robot collaboration in manufacturing is mainly related to **assembly activities**, that can be performed in specific environments set-up in order to allow robots (or better, cobots) to assist (and speed up) humans in their tasks. It has been proven (Wilson and Daugherty, 2018) that collaboration is more effective than running activities separately and so, great efforts have been (and are) spent to **improve communication among humans and machines**.

Communication turned out to be a crucial aspect to be taken into consideration when you are talking of activities that cannot be completely assigned to machines but that require the human contribution, such as in decision making process.

6.3.1 Human assisting Machines

Artificial intelligence, both if we are talking of "physical solutions" as cobots but also simply of software solutions as supporting models for decision making, is **not autonomous**. It requires to be **trained** to understand how to behave**, explained** to be more easily understood and accepted by humans and **monitored**, after the deployment.

Training activities are mainly run by processing even more and more updated data, taken from historical datasets and timeseries or caught from the surrounding environment. Additionally, parameters should be constantly monitored and updated if required.

Moreover, **laws and legislation may change** and accordingly AI algorithms should do. Think for example to cobots that operate in a workplace subjected to a strict national workers regulation or to algorithms that make decision about people and have to apply criteria that respect ethics norms.

Often, AI solutions are so innovative that when they are adopted for the first time, there isn't any law regulating them, but their wide spread usually drives to introduction of new norms: hence, they are required to be adapted to new legislation and compliancy must be guaranteed by human workers over time.

6.3.2 Machines assisting Humans

According to the latest paradigm of Human-Machine Collaboration (that puts the basis for the next Industry 5.0), artificial intelligence and machines in general must be conceived to **complement and augment human capabilities**, not to replace them. The goal for a perfect human-machine match is to identify which human features can't be embodied by machines, in order to enhance and concentrate on them, and to delegate those where workers can be effectively replaced. For example, humans should focus on leadership, teamwork, creativity, while machines could be easily exploited to improve speed, scalability and big data management.

Indeed, it's not a secret that **humans tackle unpredictable situations better**, since they are aware of lot of details of the environment around them (even if unconsciously), but on the other side **they are more prone to error**, mainly in case of repetitive tasks.

Amplifying is a key word to properly define the I5.0 machines' support to workers: relieved from trivial and time-consuming tasks, workers can spend more effort in specific activities that AI is not able to perform, improving performance and results.

Recently, some studies have been published presenting innovative solutions able to monitor the worker's level of stress and fatigue. In (Paredes-Astudillo, et al., 2020), for example, it is described a Cyber-Physical Production System where machines are enabled to measure the current worker cognitive fatigue, to estimate his/her following status and to change the schedule of his/her tasks, **making autonomously decisions**, in order to improve the worker's well-being. Perfectly in accordance

with I5.0 principles, shifts planning is adapted not only to increase productivity but also to reduce workload.

6.3.3 Other methods for Industry5.0

In January 2021, the European Commission published the paper "Industry 5.0 – Towards a sustainable, human-centric and resilient European Industry", where there are defined the three main features of Industry 5.0 according to European perspective, that is **sustainability, human-centricity** and **resilience**. They are three key concepts to be taken in mind while developing new solutions for industry.

Sustainability has been a flagship of European policy for a long time and, in particular, it represents the core of the Green Deal, signed in December 2019 where it is stated that Europe must do transition to a sustainable economy.

The concept of **resilience**, that got popular due to pandemic crisis of covid-19, has been identified as one of the essential features that a manufacturer must show to be competitive in the international scenario.

Last but not least, **human-centricity** is strictly connected to what stated in previous paragraphs: it means that from digital transition both companies and workers should benefit and innovative solutions must be thought also to bring advantages to workers.

An important point to be taken into account, to fully benefit from digital innovation, is the need of re-skilling and up-skilling people. According to the I5.0 approach where workers are no more "costs" but "investments", the employer should finance training activities to increase employees' digital capabilities. A Deloitte study[18] highlights that currently the European industry is strongly affected by lack of competences and specific skills related to digital innovation and AI. The approach is twofold.

- On one side, it is important to invest on workers re-skilling, providing at each level of the workforce pyramid the **basic knowledge about how AI works** and potential benefits/limitations of this technology.
- On the other side, AI solutions developers should take in mind the novel approach of Industry 5.0: **technology could be made more intuitive and user-friendly.**

The idea of a 5.0 industry is still at its definition phase, anyway the European Union is investing funds to carry on research activities where implement, test and improve the concept of Industry 5.0, in order to understand feasibility and advantages and to identify obstacles and barriers in the implementation.

For instance, in the wide scenario of Horizon 2020 projects, it's worth mentioning "CoLLaboratE" (with the purpose of developing a system of collaborative assembly capable to optimize the allocation of human and robotic resources); "EMPOWER" (to implement a platform to promote helth and well-being, reduce psychological distress, prevent common mental problems and reduce their impact in the workplace); "HUMAN" (aiming at demonstrating workplaces where automation and human workers operate in harmony to improve productivity, quality, performance and satisfaction); "AI REGIO" (to model the playground for Industry 5.0 and Collaborative Intelligence, from a process management point of view in order to provide tools and skills to DIHs).

6.4 Ethical Issues on AI and Human Interaction

6.4.1 EU ethics guidelines for trustworthy AI the seven key requirements

[18] https://www.prnewswire.com/news-releases/deloitte-research-reveals-a-generation-disrupted-growing-up-in-a-world-of-accelerated-transformation-leaves-millennials-and-gen-zs-feeling-unsettled-about-the-future-300851008.html

AI is a promising means to enhance individual and societal well-being, as well as bringing progress and innovation. However, to maximize the benefits but preventing or reducing potential risks deriving from its implementation and development, AI system must be trustworthy. Since the debate on ethical principles for AI has started ((Wiener, 1969); (Samuel, 1960)), copious numbers of frameworks, principles and value about the ethics of AI has been proposed (for a review see (Jobin, et al., 2019)) sometimes contributing to generate confusion and a customized using of ethics ((Floridi, 2019)a; (Floridi, 2019)b; (Hagendorff, 2020)). To offering a clear frame and a common benchmark to qualify trustworthy AI, the European Commission established a High-Level Expert Group on Artificial Intelligence (AI HLEG)1 composed of 52 experts from academia, civil society and industry, that on April 2019 released their Ethics Guidelines for Trustworthy AI (first release, 2018). The Guidelines identified three components as prerequisites to use and develop Trustworthy AI systems: it should be lawful, ethical, and robust. The Guidelines did not directly focus on lawful AI but instead translated some ethical principles for ethical and robust AI into seven requirements that Trustworthy AI should meet:

1. human agency and oversight: protection of fundamental rights, the interaction between humans and AI systems;
2. technical robustness and safety: resilience, accuracy, reliability of AI systems;
3. privacy and data governance: data protection, data management, privacy rights;
4. transparency: traceability, explainability, communication;
5. diversity, non-discrimination and fairness: accessibility, lawfulness;
6. environmental and societal well-being: sustainability, social and societal impact;
7. accountability: auditability, reporting, responsibility.

6.4.2 Ethical risks of human- AI interaction in industry

Even though ethical guidelines are aiming at offering a guide to incorporate ethical aspects in AI development and use, their implementation needs to be adapted to the specific context of AI application (Floridi, 2019). In the context of Industry 4.0, with the integration of AI with emerging technologies (Aghion, et al., 2017), such as industrial Internet of Things or data analytics (for a review, (Li, et al., 2017); see (Lee, et al., 2018); (Becker & Stern, 2016)), important ethical issues may concern replacing of careers; (Ford, 2015), risk to loose human skills ((Torresen, 2018); (Parasuraman, et al., 2000); (Endsley & Kiris, 1995)) and the integration of supported or automated decision-making through a data-driven approach (e.g. predictive maintenance, (Susto, et al., 2015); see also (Mpofu & Nicolaides, 2019); (Manzey, et al., 2012); (Palhares, et al., 2019)).

Specifically, algorithms widely used for automated or assisted decision-making (e.g. deep learning algorithms) pose a serious issue for transparency ((Miller, 2019); (Abdollahi & Nasraoui, 2018); (Abdul, et al., 2018)) given their black-box nature ((Fenech, et al., 2018); (Doshi-Velez & Kim, 2017); (Lepri, et al., 2018)). Generally, how these system process and analyze the input to provide specific output (e.g. recommendations or predictions), could be challenging to be understood ((Doshi-Velez & Kim, 2017); (Lepri, et al., 2018)) especially to non-computing professionals (Kizilcec, 2016). In such a context, transparency, here conceived as the "availability of information about an actor allowing other actors to monitor the workings or performance of this actor" (Meijer, 2014) is highly associated with explainability, as the ability of a system to provide precious and understandable explanations for validating a decision rationale ((Doran, et al., 2017); (Chazette & Schneider, 2020)) and represents a crucial ethical standard for Trustworthy AI.

6.4.3 Trust, transparency and explainability in decision making

In the contexts where the integration of AI technologies such as Machine Learning (ML), are used to assist decision-makers (e.g. healthcare, consumer behavior, etc), often with low level of human control ((Alpaydin, 2016); (Zerilli, et al., 2019)), the issue of transparency of AI has become a key issue ((Jobin, et al., 2019); (Burrell, 2016); (Pasquale, 2015)). Prospective transparency (i.e. how the AI system reaches decisions) and retrospective transparency (i.e. post hoc explanations and rationales;

Paal and Pauly, 2018) are considered key elements to allow decision makers to understand model' recommendations and decisions and thus to reach optimal humans- AI collaboration (Hois, et al., 2019).

The emerging field on explainable AI (XAI), ((Pynadath, et al., 2018); (Adadi & Berrada, 2018); (Miller, 2019); (Zerilli, et al., 2019)), focused on transparency, traceability interpretability ((Grzymek & Puntschuh, 2019); (Koene, et al., 2019); (Guidotti, et al., 2018); (Samek, 2019)) of decisions, predictions and actions of system as a mean to foster trust in AI and in turn its adoption ((Siau & Wang, 2018); (Holliday, et al., 2016); (Fukuyama, 1995)).

Even if the relationship between transparency and trust is not fully ascertained ((Cramer, et al., 2008); (Schmidt, et al., 2020); (Felzmann, et al., 2019); (Felzmann, et al., 2020)), a lack of transparency on the internal operation of AI system can result in either disuse due to mistrust ((Siau & Wang, 2018); (Holliday, et al., 2016)) or abuse over-reliance ((Moray & Inagaki, 2000); (Wagner & Robinette, 2015); (Parasuraman & Riley, 1997); (Cummings, 2004)) that in turn may have risky consequences on monitoring and error prevention (Parasuraman, et al., 1993). Hence a lack of transparency limits human verification over AI and the developing of appropriate reliance and management of unwarranted outcomes ((Ribeiro, et al., 2016); (Zarsky, 2016); (Kulesza, et al., 2015); (Rader, et al., 2018); (Mittelstadt, et al., 2016)).

6.4.4 Explainability AI, responsibility and accountability

Beyond the efforts to make AI reliable, there is always the possibility that something goes wrong and in these cases it is important to be able to ascribe responsibility. With the increased agency of AI in decision-making, the issue of responsibility for decisions, and especially for mistakes, is crucial ((Perc, et al., 2019); (Bostrom & Yudkowsky, 2014)).

Despite the issue of Responsible AI and moral agency have received extensive consideration in ethics of computing and robot ethics (e.g. (Sullins, 2011); (Johnson, 2006); (Dignum, 2019); (Murphy & Woods, 2009); (Floridi & Sanders, 2004); (Coeckelbergh, 2010)), there is a spread consensus that humans are the ultimate responsible for AI based decisions (e.g. (Dignum, 2019); (Zhang, et al., 2020)). For instance, according to a recent work from (Coeckelbergh, 2020), "*AI technologies do not meet traditional criteria for full moral agency (and hence preconditions for responsibility)" (*see also (Bryson, et al., 2017)) intended here as the Aristotelian' conditions (Barnes, 1984) of control (e.g. having free will) and knowledge (e.g. be aware of). Yet, despite only humans can hold responsibility for AI-based decisions, a lack of their control over the AI (i.e. responsibility gap, (Matthias, 2004)) or of knowledge (the knowledge of the instrument used to act a decision), call into question *if* and *which* humans (e.g. the problem of many hands, (Van De Poel, et al., 2015); distributed responsibility, (Taddeo & Floridi, 2018)) can actually hold such responsibility. Hence, transparency of AI system became critical not only for ascribing agency but also for responsibility as answerability, conceived as the ability to provide justifications to those who are affected by the decisions (i.e. moral patients).

Responsibility and answerability are thus about accountability (see (Kohli & Barreto, 2018) and (Olsen, 2014)) as the acknowledgement and reporting of any potential negative implications of AI system adoption, especially in domains where system reliability is crucial (e.g. healthcare, automated transportation, critical industrial application, etc).

6.5 Most relevant results from interviews

As a collateral activity of the analysis of the state of the art about AI models and decision-making process, Task 1.2 has run a series of interviews with decision makers at various level of manufacturing system, in order to validate information collected initially at theoretical level. Actors involved in the interviews were the four pilots' demonstrators of XMANAI project: FORD, WHIRLPOOL, CNH and UNIMETRIK, all involved in the development of AI solutions.

- FORD is representative of automotive sector and its goal is to implement several **recommendation** systems and a Digital Twin for simulation;
- WHIRLPOOL acts in the household appliances sector and its **forecasting** solution is addressed to the direct to consumer (D2C) sales planning department;
- CNH is leader of the tractors' sector and it is planning to develop an AI solution of real-time data analysis to **optimize** the manufacturing process and the maintenance;
- UNIMETRIK is a metrological service company, whose objective is to build an AI model to deduce the best strategy for **machine configuration,** leveraging on past configurations having similarities.

As already mentioned, **recommendation**, **forecasting**, **optimization** and **machine configurations** are all activities strongly related to the decision making process in manufacturing, meaning that an AI model acting with these purposes is capable to support the decision maker, providing useful information and suggestions.

Hence, the main objective of the interviews was to understand how the decision making process takes place and which is the current relationship between humans and AI, taking into account different manufacturing sectors and different workforce levels. Namely, the interviews have been structured in three main blocks (you can find the entire list of questions in the annex):

- **Decision making process**: to collect information about how it is conducted, which methods and techniques are usually adopted and how time pressure can influence a decision;
- **AI adoption in decision making process:** to collect information about AI adoption in general (if people make use of it or not), how AI solution are chosen, what it is expected from an AI model, which competences are required;
- **Ethical issues in AI adoption**: to collect information about possible ethical issues that may occur/have already occurred, what makes an AI model trustworthy, how to behave to face AI failure.

An average of 3 people for each pilot have been interviewed, highlighting topics of great interest for XMANAI project, to be taken into account while developing tools conceived to collaborate and to support (and to be accepted by) humans.

The approach with artificial intelligence is not homogeneous: for instance, WHIRLPOOL employees (at managerial, strategic and planning level) already experienced AI solutions and the company is pushing for its adoption besides XMANAI project; on the other side FORD, CNH and UNIMETRIK admitted that have not made use of AI in their activities and in the company in general.

In any case, whether AI has been already adopted or not, interviewees currently run their tasks without the support of any artificial intelligence, but leveraging on simple tools and mainly on their own experience. The enthusiasm toward the implementation of an automatic model, able to reveal information that humans don't catch, is indisputable: in particular, the greatest expectation concerns the increasing amount of data that will be processed by an artificial intelligence process. Having at disposal a larger number of variables and information, doesn't necessarily mean to make a better decision; but if there is a trustworthy "artificial mind" able to find links and relationships among them, decisions can be taken with a greater awareness.

For example, FORD is expecting to develop a recommendation tool for batch planning that will take into account more variables than those manually analysed today, able to provide new suggestions to increase efficiency; WHIRLPOOL is taking into account the possibility of offering customized promotions to its D2C clients (that can be planned only if having at their disposal information about their navigation and tastes); in CNH breakdowns occur daily and it is expected to create an AI support tool that helps in defining the best strategy for the plant maintenance.

Besides the well-known advantages that an AI solution can provide, there are some key aspects that can't be overlooked:

- Humans' experience can't be replaced. AI models should be conceived as support tools, that helps humans to improve the process.
- The final decision can't be made by machines but the worker's validation is always required.
- AI acceptance is not automatic, mainly among workers that have never dealt with it. Showing success stories and advantages could help.
- Data visualization (of data used by the AI model) could reduce the risk of "black box" solution, hardly accepted and considered unreliable.
- It is important to teach people what can be learnt from AI and make them aware of its potential. Moreover, people must have basic knowledge and understanding of the model to trust it.
- The development and maintenance of an AI model is not automatic and required competence may be not available in the company. This could be an obstacle in AI adoption.

7 Conclusion

The objective of D1.1 is to provide the result of the technical investigation conducted in the domain of AI complemented by a more business-oriented focus on decision making process, in order to deliver a list of concepts, methods, tools and requirements to be used as a starting point to develop the XMANAI solution.

The analysis of the state-of-the-art of techniques dealing with AI explainability (starting from the transparent models explainable by design to post-hoc explainable methods) is provided from a technical point of view. Since the purpose of the document is not to be a didactic book, but to drive the reader among the dozens of different existing solutions, algorithms are not explained in details and require a certain mathematical background to be deeply understood. Anyway, methods and techniques are all described putting in light the scenario where they can be applied, the hypothesis required and they are compared pointing out advantages and disadvantages adopting each one.

The research activity was not limited only to the theoretical aspects, as well as models that are presented in literature, but it was extended including also the investigation over existing software tools developed to implement XAI techniques and over example of XAI applications.

It was of great interest and relevance for the initial XMANAI activities to conduct interviews with decision makers with the aim of exploring which different techniques are adopted to take decisions, including the adoption of AI solutions. Since interviews were run with the XMANAI demonstrators, the activity allowed to identify which are the requirements and expectations from the XMANAI platform, what is “nice to have” and what is essential, what makes an AI algorithm reliable and how the decision making process can be enhanced by AI adoption, in order to define how models should be expressed to be comprehensible to the business user point of view.

It is not trivial to match the right ratio between efficiency and explainability and in order to manage it, it is fundamental to consider not only the technical standpoint but also the final user’s requirements.

Hence, D1.1 is a sort of guide that collects, from different perspectives, information about the AI domain, presenting methods, tools and experiences, to help to orientate among the massive amount of knowledge and to be taken into account by the XMANAI technical WPs to better match the demonstrators’ and project’s needs.

8 References

Abdollahi, B. & Nasraoui, O., 2018. Transparency in Fair Machine Learning: the Case of Explainable Recommender Systems. In: J. Zhou & F. Chen, eds. *Human and Machine Learning: Visible, Explainable, Trustworthy and Transparent.* Cham: Springer International Publishing, p. 21–35.

Abdul, A. et al., 2018. Trends and trajectories for explainable, accountable and intelligible systems: An hci research agenda.. *CHI conference on human factors in computing systems,* pp. 1-18.

Abu-El-Haija, S., Perozzi, B., Al-Rfou, R. & Alemi, A., 2017. *Watch your step: Learning node embeddings via graph attention.* s.l.:s.n.

Adadi, A. & Berrada, M., 2018. Peeking inside the black-box: a survey on explainable artificial intelligence (XAI). *IEEE access,* Volume 6, pp. 52138-52160.

Aggarwal, C. C., Bar-Noy, A. & Shamoun, S., 2017. On sensor selection in linked information networks. *Computer Networks,* Volume 126, p. 100–113.

Aghion, P., Jones, B. & Jones, C., 2017. Artificial intelligence and economic growth. *National Bureau of Economic Research.*

Ahmed, A. et al., 2013. *Distributed large-scale natural graph factorization.* s.l., s.n., p. 37–47.

Akoglu, L., Tong, H. & Koutra, D., 2015. Graph based anomaly detection and description: A survey. *Data Mining and Knowledge Discovery,* Volume 29, p. 626–688.

Alber, M., n.d. *iNNvestigate neural networks!.* s.l.:s.n.

Alpaydin, E., 2016. Machine learning: the new AI. *MIT press.*

Alvarez-Melis, D. & Jaakkola, T. S., 2018. *On the robustness of interpretability methods.* s.l.:s.n.

Amin, K. et al., 2020. *DeepKAF: A Heterogeneous CBR & Deep Learning Approach for NLP Prototyping.* s.l., s.n., pp. 1-7.

Ancona, M., Ceolini, E., Öztireli, C. & Gross, M., 2017. *Towards better understanding of gradient-based attribution methods for deep neural networks.* s.l.:arXiv.

Angeline, P. J., 1994. *Genetic programming: On the programming of computers by means of natural selection,.* s.l.:MIT Press.

Ansari, F., Glawar, R. & Nemeth, T., 2019. PriMa: a prescriptive maintenance model for cyber-physical production systems. *International Journal of Computer Integrated Manufacturing,* 32(4-5), pp. 482-503.

Apley, D. W., 2019. Visualizing the effects of predictor variables in black box supervised learning models..

Ariely, D. & Zakay, D., 2001. A timely account of the role of duration in decision making. *Acta Psychologica,* Volume 108, p. 187–207.

Arinez, J. F. et al., 2020. Artificial Intelligence in Advanced Manufacturing: Current Status and Future Outlook. *Journal of Manufacturing Science and Engineering,* August.142(11).

Arras, L., Montavon, G., Müller, K. R. & Samek, W., 2017. *Explaining recurrent neural network predictions in sentiment analysis.* s.l.:s.n.

Arrieta, A. B. et al., 2020. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion,* Volume 58, p. 82–115.

Athalye, A., Engstrom, L., Ilyas, A. & Kwok, K., 2017. *Synthesizing robust adversarial examples.* s.l., s.n., p. 449–468.

Atwood, J. & Towsley, D., 2015. *Diffusion-convolutional neural networks.* s.l.:s.n.

Bai, J. et al., 2019. Rectified Decision Trees: Towards Interpretability, Compression and Empirical Soundness. *arXiv: Learning.*

Barakat, N. & Bradley, A., 2007. Rule Extraction from Support Vector Machines: A Sequential Covering Approach. *IEEE Transactions on Knowledge and Data Engineering,* July, Volume 19, pp. 729-741.

Barakat, N. & Bradley, A. P., 2010. Rule extraction from support vector machines: A review. *Neurocomputing,* 74(Artificial Brains), pp. 178-190.

Barnes, J., 1984. The complete works of aristotle. *Princeton University Press,* Volume 2, p. 1729–1867.

Barredo Arrieta, A. et al., 2019. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *arXiv,* Volume 58, p. 82–115.

Baryannis, G., Dani, S. & Antoniou, G., 2019. Predicting supply chain risks using machine learning: The trade-off between performance and interpretability. *Future Generation Computer Systems,* Volume 101, pp. 993-1004.

Basu, S., Pope, P. & Feizi, S., 2020. Influence Functions in Deep Learning Are Fragile. *arXiv e-prints,* p. arXiv:2006.14651.

Basu, S., You, X. & Feizi, S., 2020. *On Second-Order Group Influence Functions for Black-Box Predictions.* s.l., PMLR, pp. 715-724.

Battaglia, P. W. et al., 2018. *Relational inductive biases, deep learning, and graph networks.* s.l.:s.n.

Bau, D. et al., 2017. *Network dissection: Quantifying interpretability of deep visual representations.* s.l., s.n., p. 3319–3327.

Becker, T. & Stern, H., 2016. Future trends in human work area design for cyber-physical production systems.. *Procedia Cirp,* Volume 57, pp. 404-409.

Belkin, M. & Niyogi, P., 2001. *Laplacian eigenmaps and spectral techniques for embedding and clustering.* Cambridge, MA, USA, MIT Press, pp. 585-591.

Bharti, S., Kurian, D. S. & Pillai, V. M., 2020. *Reinforcement Learning for Inventory Management.* Singapore, Springer, pp. 877-885.

Binder, A. et al., 2016. Layer-wise relevance propagation for neural networks with local renormalization layers. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics).* s.l.:s.n., p. 63–71.

Blum, A. L. & Furst, M. L., 1997. Fast planning through planning graph analysis. *Artificial Intelligence,* Volume 90, p. 281–300.

Boehmke, B., Greenwell, B., Boehmke, B. & Greenwell, B., 2020. *Interpretable Machine Learning.* s.l.:s.n.

Bojchevski, A., Shchur, O., Zügner, D. & Günnemann, S., 2018. *NetGAN: Generating graphs via random walks.* s.l., PMLR, p. 610–619.

Borgo, R., Cashmore, M. & Magazzeni, D., 2018. *Towards Providing Explanations for AI Planner Decisions.* s.l.:s.n.

Bostrom, N. & Yudkowsky, E., 2014. The ethics of artificial intelligence. *The Cambridge handbook of artificial intelligence,* Volume 1, pp. 316-334.

Bovens, M. & Goodin, R. E., n.d. *the Oxford Handbook of Public.* s.l.:s.n.

Bovens, M., Goodin, R. E., Schillemans, T. & Olsen, J. P., 2014. *Accountability and Ambiguity.* Oxford: Oxford University Press.

Breiman, L., 1993. *Classification and regression trees.* New York: Chapman & Hall.

Brieman, L., 2001. Random Forests. *Machine Learning,* Volume 45, pp. 5-32.

Bronstein, M. M. et al., 2017. Geometric Deep Learning: Going beyond Euclidean data. *IEEE Signal Processing Magazine,* Volume 34, p. 18–42.

Brophy, J. & Lowd, D., 2020. TREX: Tree-Ensemble Representer-Point Explanations. *arXiv e-prints,* September.p. arXiv:2009.05530.

Brown, T. B., n.d. *Adversarial Patch.* s.l.:s.n.

Bruna, J., Zaremba, W., Szlam, A. & Lecun, Y., 2014. *Spectral networks and locally connected networks on graphs.* s.l., CBLS.

Bryson, J. J., Diamantis, M. E. & Grant, T. D., 2017. Of, for, and by the people: the legal lacuna of synthetic persons. *Artificial Intelligence and Law,* Volume 25, p. 273–291.

Burrell, J., 2016. How the machine 'thinks': Understanding opacity in machine learning algorithms. *Big Data & Society,* Volume 3, pp. 1-12.

Cai, H., Zheng, V. W. & Chang, K. C., 2018. A Comprehensive Survey of Graph Embedding: Problems, Techniques and Applications. *IEEE Transactions on Knowledge and Data Engineering,* Volume 30, p. 1616–1637.

Calatayud, A., Mangan, J. & Christopher, M., 2019. The self-thinking supply chain. *Supply Chain Management: An International Journal,* 24(1), pp. 22-38.

Callut, J., Françoisse, K., Saerens, M. & Dupont, P., 2008. *Semi-supervised classification from discriminative random walks.* Berlin, Springer, p. 162–177.

Cao, S., Lu, W. & Xu, Q., 2015. *GraRep: Learning graph representations with global structural information.* s.l., s.n., p. 891–900.

Cao, S., Lu, W. & Xu, Q., 2016. Deep neural networks for learning graph representations. *30th AAAI Conference on Artificial Intelligence, AAAI 2016,* Volume 30, p. 1145–1152.

Cao, W., Yan, Z., He, Z. & He, Z., 2020. A Comprehensive Survey on Geometric Deep Learning. *IEEE Access,* Volume 8, p. 35929–35949.

Capgemini Research Institute, 2019. *Scaling AI in manufacturing operations,* s.l.: Capgemini Research Institute.

Carletti, M., Masiero, C., Beghi, A. & Susto, G. A., 2019. *Explainable Machine Learning in Industry 4.0: Evaluating Feature Importance in Anomaly Detection to Enable Root Cause Analysis.* s.l., s.n., pp. 21-26.

Catherine, R., Mazaitis, K., Eskenazi, M. & Cohen, W., n.d. *Explainable Entity-based Recommendations.* s.l.:s.n.

Chami, I. et al., 2020. *Machine learning on graphs: A model and comprehensive taxonomy.* s.l.:s.n.

Chazette, L. & Schneider, K., 2020. Explainability as a non-functional requirement: challenges and recommendations. *Requirements Engineering,* Volume 25, p. 493–514.

Chen, H.-Y. & Lee, C.-H., 2020. Vibration Signals Analysis by Explainable Artificial Intelligence (XAI) Approach: Application on Bearing Faults Diagnosis. *IEEE Access,* Volume 8, pp. 134246-134256.

Chen, H., Perozzi, B., Hu, Y. & Skiena, S., 2018. *HARP: Hierarchical representation learning for networks.* s.l., s.n.

Chen, H., Zhang, H., Boning, D. & Hsieh, C.-J., 2019. *Robust Decision Trees Against Adversarial Examples.* s.l., PMLR, pp. 1122-1131.

Chen, T. & Guestrin, C., 2016. *XGBoost: A Scalable Tree Boosting System.* San Francisco, California, USA, Association for Computing Machinery, p. 785–794.

Chen, X., Zhang, B. & Gao, D., 2021. Bearing fault diagnosis base on multi-scale CNN and LSTM model. *Journal of Intelligent Manufacturing,* 32(4), pp. 971-987.

Chen, X., Zhang, B. & Gao, D., 2021. Bearing fault diagnosis based on multi-scale CNN and LSTM model. *Journal of Intelligent Manufacturing,* 32(4), pp. 971-987.

Cherkassky, V. & Dhar, S., 2010. *Simple Method for Interpretation of High-Dimensional Nonlinear SVM Classification Models..* Las Vegas, Nevada, USA, CSREA Press, pp. 267-272.

Choo, C. W., 1991. Towards an information model of organizations. *The Canadian Journal of Information Science,* Volume 16, p. 32–62.

Christoforou, A. & Andreou, A. S., 2017. A framework for static and dynamic analysis of multi-layer fuzzy cognitive maps. *Neurocomputing,* Volume 232, p. 133–145.

Coeckelbergh, M., 2010. Moral appearances: emotions, robots, and human morality. *Ethics and Information Technology,* Volume 12, pp. 235-241.

Coeckelbergh, M., 2020. Artificial intelligence, responsibility attribution, and a relational justification of explainability. *Science and engineering ethics,* Volume 26, pp. 2051-2068.

Cohen, Y. & Singer, G., 2021. A smart process controller framework for Industry 4.0 settings. *Journal of Intelligent Manufacturing,* 03.Volume 32.

Confalonieri, R., Coba, L., Wagner, B. & Besold, T. R., 2021. A historical perspective of explainable Artificial Intelligence. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery,* Volume 11.

Cook, R. D. & Weisberg, S., 1980. Characterizations of an Empirical Influence Function for Detecting Influential Cases in Regression. *Technometrics,* 22(4), pp. 495-508.

Cortes, C. & Vapnik, V., 1995. Support-Vector Network. *Machine Learning,* Volume 20, pp. 273-297.

Craiger, J. P., Goodman, D. F., Weiss, R. J. & Butler, A., 1996. Modeling organizational behavior with fuzzy cognitive maps. *International Journal of Computational Intelligence and Organizations,* Volume 1, p. 120–123.

Cramer, H. et al., 2008. The effects of transparency on trust in and acceptance of a content-based art recommender. *User Modeling and User-adapted interaction,* Volume 18, p. 455.

Cummings, M. L., 2004. *Automation bias in intelligent time critical decision support systems.* s.l., s.n., p. 557–562.

Dai, H. et al., 2018. *Learning steady-states of iterative algorithms over graphs.* s.l., PMLR, pp. 1106-1114.

Dai, Q., Li, Q., Tang, J. & Wang, D., 2018. *Adversarial network embedding.* s.l., s.n.

Danon, B., 2018. *How GM and Autodesk are using generative design for vehicles of the future.* [Online] Available at: https://adsknews.autodesk.com/news/gm-autodesk-using-generative-design-vehicles-future
[Accessed 19 03 2021].

Davenport, T. H., 2018. From analytics to artificial intelligence. *Journal of Business Analytics,* Volume 1, p. 73–80.

Davenport, T. H. & Kirby, J., 2016. Just how smart are smart machines?. *MIT Sloan Management Review,* Volume 57, p. 21–25.

Davis, E., 2015. Knowledge Representation. In: *International Encyclopedia of the Social & Behavioral Sciences: Second Edition.* Second ed. s.l.:s.n., p. 98–104.

Defferrard, M., Bresson, X. & Vandergheynst, P., 2016. *Convolutional neural networks on graphs with fast localized spectral filtering.* s.l.:s.n.

Deloitte Global, 2019. *Deloitte research reveals a "generation disrupted": Growing up in a world of accelerated transformation leaves Millennials and Gen Zs feeling unsettled about the future.* s.l.:s.n.

Deng, H., 2014. Interpreting Tree Ensembles with inTrees. *arXiv e-prints,* August.p. arXiv:1408.5456.

Denton, S. M. & Salleb-Aouissi, A., 2020. A Weighted Solution to SVM Actionability and Interpretability.

Devos, L., Meert, W. & Davis, J., 2020. Versatile Verification of Tree Ensembles. *arXiv e-prints,* October.p. arXiv:2010.13880.

Dhar, R., Nowlis, S. M. & Sherman, S. J., 2000. Trying hard or hardly trying: An analysis of context effects in choice. *Journal of Consumer Psychology,* Volume 9, p. 189–200.

Dignum, V., 2019. *Responsible Artificial Intelligence: How to Develop and Use AI in a Responsible Way.* s.l.:s.n.

Do, K., Tran, T. & Venkatesh, S., 2019. *Graph transformation policy network for chemical reaction prediction.* New York, NY, USA, Association for Computing Machinery, p. 750–760.

Doltsinis, S., Krestenitis, M. & Doulgeri, Z., 2020. A Machine Learning Framework for Real-Time Identification of Successful Snap-Fit Assemblies. *IEEE Transactions on Automation Science and Engineering,* 17(1), pp. 513-523.

Doran, D., Schulz, S. & Besold, T. R., 2017. *What does explainable ai really mean? A new conceptualization of perspectives.* s.l.:s.n.

Doshi-Velez, F. & Kim, B., 2017. *Towards A Rigorous Science of Interpretable Machine Learning.* s.l.:s.n.

Doshi-Velez, F. & Kim, B., 2017. Towards A Rigorous Science of Interpretable Machine Learning.

Endsley, M. R. & Kiris, E. O., 1995. The out-of-the-loop performance problem and level of control in automation. *Human Factors,* Volume 37, p. 381–394.

Erhan, D., Bengio, Y., Courville, A. & Vincent, P., 2009. Visualizing higher-layer features of a deep network. *Bernoulli,* p. 1–13.

Felzmann, H. e. a., 2020. Towards Transparency by Design for Artificial Intelligence. *Science and Engineering Ethics,* Volume 26, p. 3333–3361.

Felzmann, H., Fosch-Villaronga, E., Lutz, C. & Tamò-Larrieux, A., 2020. Towards Transparency by Design for Artificial Intelligence. *Science and Engineering Ethics,* Volume 26, p. 3333–3361.

Felzmann, H., Villaronga, E. F., Lutz, C. & Tamò-Larrieux, A., 2019. Transparency you can trust: Transparency requirements for artificial intelligence between legal norms and contextual concerns. *Big Data and Society,* Volume 6, p. 2053951719860542.

Fenech, M., Strukelj, N. & Buston, O., 2018. Ethical, social and political challenges of artificial intelligence in health: future Advocacy report for the Wellcome Trust..

Feng, J., Huang, M., Yang, Y. & Zhu, X., 2016. *GAKE: Graph aware knowledge embedding.* s.l., s.n., p. 641–651.

Floridi, L., 2019. Establishing the rules for building trustworthy AI. *Nature Machine Intelligence,* Volume 1, p. 261–262.

Floridi, L., 2019. *The Logic of Information.* s.l.:s.n.

Floridi, L. & Sanders, J. W., 2004. On the morality of artificial agents. *Minds and machines,* Volume 14 (3), pp. 349-379.

Ford, M., 2015. *Rise of the Robots: Technology and the Threat of a Jobless Future..* New York: Basic Books.

Fout, A., Byrd, J., Shariat, B. & Ben-Hur, A., 2017. *Protein interface prediction using graph convolutional networks.* s.l.:Doctoral dissertation. Colorado State University.

Freeman, L. C., 1978. Centrality in social networks conceptual clarification. *Social Networks,* Volume 1, p. 215–239.

Freund, Y. & Schapire, R. E., 1997. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of computer and system sciences,* August, Volume 55, pp. 119-.

Friedman, J. H., 2001. Greedy function approximation: A gradient boosting machine. *Annals of Statistics,* October, 29(5), pp. 1189-1232.

Fukuyama, F., 1995. Trust: The social virtues and the creation of prosperity. *New York: Free press,* Volume 99.

Fu, X. et al., 2004. *Extracting the knowledge embedded in support vector machines.* s.l., IEEE.

Gallicchio, C. & Micheli, A., 2010. *Graph echo state networks.* s.l., IEEE, pp. 1-8.

Gaonkar, B. T., Shinohara, R. & Davatzikos, C., 2015. Interpreting support vector machine models for multivariate group wise analysis in neuroimaging. *Medical Image Analysis,* 24(1), pp. 190-204.

Garg, V. K., Jegelka, S. & Jaakkola, T., 2020. *Generalization and representational limits of graph neural networks.* s.l., s.n., p. 3419–3430.

Gellera, G. & Thompson, J. W., 2017. Nicomachean ethics. In: J. Barnes, ed. *Nicomachean Ethics.* Princeton: Princeton University Press, p. 1–97.

Ghorbani, A. & Zou, J., 2019. Data Shapley: Equitable Valuation of Data for Machine Learning. *arXiv e-prints,* p. arXiv:1904.02868.

Gilmer, J. et al., 2017. *Neural message passing for quantum chemistry.* s.l., PMLR, p. 1263–1272.

Girvan, M. & Newman, M. E. J., 2002. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences of the United States of America,* Volume 99, p. 7821–7826.

Golbeck, J., 2013. Network Visualization. *Analyzing the Social Web,* Volume 1, p. 45–62.

Goldenberg, A., Zheng, A. X., Fienberg, S. E. & Airoldi, E. M., 2009. *A survey of statistical network models.* s.l.:s.n.

Gold, N. E., 2020. *Virginia Dignum: Responsible Artificial Intelligence: How to Develop and Use AI in a Responsible Way.* s.l.:s.n.

Goyal, P. & Ferrara, E., 2018. Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems,* Volume 151, p. 78–94.

Goyal, P., Kamra, N., He, X. & Liu, Y., 2018. *DynGEM: Deep embedding method for dynamic graphs.* s.l.:s.n.

Grezmak, J. et al., 2020. Interpretable Convolutional Neural Network Through Layer-wise Relevance Propagation for Machine Fault Diagnosis. *IEEE Sensors Journal,* 20(6), pp. 3172-3181.

Grover, A. & Leskovec, J., 2016. *Node2vec: Scalable feature learning for networks.* s.l., s.n., p. 855–864.

Grzymek, V. & Puntschuh, M., 2019. What Europe knows and thinks about algorithms: Results of a representative survey. *Bertelsmann Stiftung,* p. 1–38.

Guidotti, R. et al., 2018. A survey of methods for explaining black box models. *ACM Computing Surveys,* Volume 51, pp. 1-42.

Gurumoorthy, K. S., Dhurandhar, A., Cecchi, G. A. & Aggarwal, C. C., 2019. *Efficient Data Representation by Selecting Prototypes with Importance Weights.* Beijing, China, IEEE, pp. 260-269.

Hagendorff, T., 2020. The Ethics of AI Ethics: An Evaluation of Guidelines. *Minds and Machines,* Volume 30, p. 99–120.

Hamilton, W. L., Ying, R. & Leskovec, J., 2017. *Inductive representation learning on large graphs.* Red Hook, NY, USA, Curran Associates Inc., pp. 1025-1035.

Handcock, M. S., Raftery, A. E. & Tantrum, J. M., 2007. Model-based clustering for social networks. *Journal of the Royal Statistical Society. Series A: Statistics in Society,* Volume 170, p. 301–354.

Han, X., Wallace, B. C. & Tsvetkov, Y., 2020. Explaining Black Box Predictions and Unveiling Data Artifacts through Influence Functions. *arXiv e-prints,* p. arXiv:2005.06676.

Haouari, M., Laporte, G., Monthly, M. & Trinajstic, N., 2002. E53 – Solutio problematis ad geometriam situs pertinentis. *Comment. Acad. Sci. U. Petrop,* Volume 286, p. 9–10.

Hara, S. & Hayashi, K., 2018. *Making Tree Ensembles Interpretable: A Bayesian Model Selection Approach.* s.l., PMLR, pp. 77-85.

Hevey, D., 2018. Network analysis: A brief overview and tutorial. *Health Psychology and Behavioral Medicine,* Volume 6, p. 301–328.

High-Level Independent Group on Artificial Intelligence (AI HLEG), 2019. *Ethics Guidelines for Trustworthy AI.* s.l.:s.n.

Hoffman, R., 2016. Using Artificial Intelligence to set information free. *MIT Sloan Frontiers.*

Hois, J., Theofanou-Fuelbier, D. & Junk, A. J., 2019. How to Achieve Explainability and Transparency in Human AI Interaction. In: C. Stephanidis, ed. *Communications in Computer and Information Science.* Cham: Springer International Publishing, p. 177–183.

Holland, P. W., Laskey, K. B. & Leinhardt, S., 1983. Stochastic blockmodels: First steps. *Social networks,* 5(2), p. 109–137.

Holliday, D., Wilson, S. & Stumpf, S., 2016. *User trust in intelligent systems: A journey over time.* s.l.:s.n.

Holsapple, C. W. & Sena, M. P., 2003. ERP plans and decision-support benefits. *Decision Support Systems,* Volume 38, p. 575–590.

Hou, Y. et al., 2009. Nonlinear dimensionality reduction by locally linear inlaying. *IEEE Transactions on Neural Networks,* Volume 20, p. 300–315.

Hrnjika, B. & Softic, S., 2020. *Explainable AI in Manufacturing: A Predictive Maintenance Case Study.* s.l., Springer, Cham, pp. 66-73.

Hu, L. et al., 2020. Petri-net-based dynamic scheduling of flexible manufacturing system via deep reinforcement learning with graph convolutional network. *Journal of Manufacturing Systems,* Volume 55, pp. 1-14.

Huysmans, J., Setiono, R., Baesens, B. & Vanthienen, J., 2008. Minerva: Sequential Covering for Rule Extraction. *IEEE Transactions on Systems, Man, and Cybernetics,* 38(Part B (Cybernetics)), pp. 299-309.

Iqbal, M. R. A., 2012. *Rule Extraction from Ensemble Methods Using Aggregated Decision Trees.* Berlin, Heidelberg, Springer-Verlag, p. 599–607.

Jain, A., Zamir, A. R., Savarese, S. & Saxena, A., 2016. *Structural-rnn: Deep learning on spatio-temporal graphs'.* s.l., s.n., p. 5308–5317.

James Wilson, H. & Daugherty, P. R., 2018. Collaborative intelligence: Humans and AI are joining forces. *Harvard Business Review,* Volume 2018.

Jarrahi, M. H., 2018. Artificial intelligence and the future of work: Human-AI symbiosis in organizational decision making. *Business Horizons,* Volume 61, p. 577–586.

Jia, R. et al., n.d. Towards Efficient Data Valuation Based on the Shapley Value. *arXiv e-prints,* p. arXiv:1902.10275.

Jobin, A., Ienca, M. & Vayena, E., 2019. The global landscape of AI ethics guidelines. *Nature Machine Intelligence,* Volume 1, p. 389–399.

Jobin, A., Ienca, M. & Vayena, E., 2019. The global landscape of AI ethics guidelines. *Nature Machine Intelligence,* Volume 1, p. 389–399.

Johansson, U., Niklasson, L. & König, R., 2004. *Accuracy vs. comprehensibility in data mining models.* s.l.:s.n.

Johnson, D. G., 2006. Computer systems: Moral entities but not moral agents. *Ethics and information technology,* Volume 8, pp. 195-204.

Kanamori, K., Takagi, T., Kobayashi, K. & Arimura, H., 2020. *DACE: Distribution-Aware Counterfactual Explanation by Mixed-Integer Linear Optimization.* s.l., International Joint Conferences on Artificial Intelligence Organization, pp. 2855-2862.

Kang, S., 2020. Product failure prediction with missing data using graph neural networks. In: *Neural Computing and Applications.* s.l.:s.n., p. 1–10.

Kass, G. V., 1980. An Exploratory Technique for Investigating Large Quantities of Categorical Data. *Applied Statistics,* Volume 29, p. 119.

Kaufman, L. & Rousseeuw, P., 1987. *Clustering by Means of Medoids.* Delft, Netherlands: Faculty of Mathematics and Informatics, TU Delft.

Kharal, A., 2020. *Explainable Artificial Intelligence Based Fault Diagnosis and Insight Harvesting for Steel Plates Manufacturing,* s.l.: s.n.

Kim, B., Khanna, R. & Koyejo, O. O., 2016. *Examples are not enough, learn to criticize! Criticism for Interpretability.* s.l., Curran Associates, Inc..

Kinderkhedia, M., 2019. *Learning Representations of Graph Data: a Survey.* s.l.:s.n.

Kipf, T. N. & Welling, M., 2016. *Variational Graph Auto-Encoders.* s.l.:s.n.

Kipf, T. N. & Welling, M., 2017. *Semi-supervised classification with graph convolutional networks.* s.l.:s.n.

Kizilcec, R. F., 2016. *How much information? Effects of transparency on trust in an algorithmic interface.* New York, NY, USA, Association for Computing Machinery (CHI, p. 2390–2395.

Kocher, M. & Sutter, M., 2006. Time is money - time pressure, incentives, and the quality of decision making. *Journal of Economic Behaviour & Organization,* Volume 61, pp. 375-392.

Koene, A. et al., 2019. *A governance framework for algorithmic accountability and transparency.* s.l.:s.n.

Kohli, N. & Barreto, R. a. K. J. A., 2018. Translation tutorial: a shared lexicon for research and practice in human-centered software systems. *1st Conference on Fairness, Accountability, and Transparanc,* Volume 7.

Koh, P. W. & Liang, P., 2017. *Understanding Black-box Predictions via Influence Functions.* Sydney, Australia, PMLR, pp. 1885-1894.

Koh, P. W. W., Ang, K.-S., Teo, H. & Liang, P. S., 2019. *On the Accuracy of Influence Functions for Measuring Group Effects.* s.l., Curran Associates, Inc..

Konig, R., Johansson, U. & Niklasson, L., 2008. *G-REX: A versatile framework for evolutionary data mining.* s.l., s.n., p. 971–974.

Konovsky, M. A., 1990. *The New Leadership: Managing Participation in Organizations..* Englewood Cliffs, NJ, Prentice-Hall: s.n.

Kroll, J. A. et al., 2018. *A Shared Lexicon for Research and Practice in Human-Centered Software Systems.* New York, NY, USA, s.n.

Kruskal, J. B., 1964. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika,* Volume 29, p. 1–27.

Kuhnle, A. et al., 2021. Designing an adaptive production control system using reinforcement learning. *Journal of Intelligent Manufacturing,* 32(3), pp. 855-876.

Kulesza, T., Burnett, M., Wong, W. K. & Stumpf, S., 2015. *Principles of Explanatory Debugging to personalize interactive machine learning.* New York, NY, USA, Association for Computing Machinery (IUI, p. 126–137.

Kuralenok, I., Ershov, V. & Labutin, I., 2019. *MonoForest framework for tree ensemble analysis.* s.l., Curran Associates, Inc..

Latouche, P. & Rossi, F., 2015. *Graphs in machine learning: An introduction.* s.l., s.n., p. 207–218.

Lecue, F., 2020. On the role of knowledge graphs in explainable AI. *Semantic Web,* Volume 11, p. 41–51.

Lee, J. B., Rossi, R. & Kong, X., 2018. *Graph classification using structural attention.* s.l., s.n., p. 1666–1674.

Lee, W. J. et al., 2019. Predictive Maintenance of Machine Tool Systems Using Artificial Intelligence Techniques Applied to Machine Condition Data. *Procedia CIRP,* 80(26th CIRP Conference on Life Cycle Engineering (LCE) Purdue University, West Lafayette, IN, USA May 7-9, 2019), pp. 506-511.

Lepri, B. et al., 2018. Fair, Transparent, and Accountable Algorithmic Decision-making Processes: The Premise, the Proposed Solutions, and the Open Challenges. *Philosophy and Technology,* Volume 31, p. 611–627.

Levy, O., Goldberg, Y. & Dagan, I., 2015. Improving Distributional Similarity with Lessons Learned from Word Embeddings. *Transactions of the Association for Computational Linguistics,* Volume 3, p. 211–225.

Li, B. H. et al., 2017. Applications of artificial intelligence in intelligent manufacturing: a review.. *Frontiers of Information Technology & Electronic Engineering,* Volume 18, pp. 86-96.

Li, J., Chen, X., Hovy, E. & Jurafsky, D., 2016. *Visualizing and understanding neural models in NLP.* s.l., s.n., p. 681–691.

Lingam, Y. K., 2018. The role of Artificial Intelligence (AI) in making accurate stock decisions in E-commerce industry. *International Journal of Advance Research, Ideas and Innovations in Technology,* 4(3), pp. 2281-2286.

Li, Q., Han, Z. & Wu, X. M., 2018. Deeper insights into graph convolutional networks for semi-supervised learning. *arXiv,* Volume 32.

Li, R., Wang, S., Zhu, F. & Huang, J., 2018. Adaptive graph convolutional neural networks. *arXiv,* Volume 32.

Li, Y. et al., 2018. *Learning deep generative models of graphs.* s.l.:s.n.

Li, Y., Yu, R., Shahabi, C. & Liu, Y., 2017. *Diffusion convolutional recurrent neural network: Data-driven traffic forecasting.* s.l.:s.n.

Li, Y., Zemel, R., Brockschmidt, M. & Tarlow, D., 2016. *Gated graph sequence neural networks.* s.l., s.n.

Loh, W. Y., 2002. *Regression trees with unbiased variable selection and interaction detection,* s.l.: s.n.

Loh, W. Y. & Shin, Y. S., 1997. *Split selection methods for classification trees.* s.l.:s.n.

Longo, F. et al., 2021. Fuzzy Cognitive Map-Based Knowledge Representation of Hazardous Industrial Operations. In: *Procedia Computer Science.* s.l.:s.n., p. 1042–1048.

Lucic, A., Oosterhuis, H., Haned, H. & de Rijke, M., 2019. FOCUS: Flexible Optimizable Counterfactual Explanations for Tree Ensembles. *arXiv e-prints,* November.p. arXiv:1911.12199.

Lucic, A., Oosterhuis, H., Haned, H. & Rijke, M. D., 2019. Actionable Interpretability through Optimizable Counterfactual Explanations for Tree Ensembles. *ArXiv,* Volume abs/1911.12199.

Lundberg, S. M. et al., 2019. Explainable AI for Trees: From Local Explanations to Global Understanding. *arXiv e-prints,* May.p. arXiv:1905.04610.

Lundberg, S. M., Erion, G. G. & Lee, S.-I., 2018. Consistent individualized feature attribution for tree ensembles. *arXiv e-prints,* Februaru.p. arXiv:1802.03888.

Lundberg, S. M. & Lee, S.-I., 2017. *A Unified Approach to Interpreting Model Predictions.* s.l., Curran Associates, Inc..

Lundberg, S. M. & Lee, S. I., 2017. *A unified approach to interpreting model predictions.* s.l.:s.n.

Luo, J., Huang, J. & Li, H., 2021. A case study of conditional deep convolutional generative adversarial networks in machine fault diagnosis. *Journal of Intelligent Manufacturing,* 32(2), pp. 407-425.

Manzey, D., Reichenbach, J. & Onnasch, L., 2012. Human Performance Consequences of Automated Decision Aids: The Impact of Degree of Automation and System Experience. *Journal of Cognitive Engineering and Decision Making,* Volume 6, p. 57–87.

Maria, A., 1997. *Introduction to modeling and simulation.* s.l., s.n., p. 7–13.

Martens, D., Baesens, B. & Van Gestel, T., 2009. Decompositional rule extraction from support vector machines by active learning. *IEEE Transactions on Knowledge and Data Engineering,* 21(2), pp. 178-191.

Matthias, A., 2004. The responsibility gap: Ascribing responsibility for the actions of learning automata. *Ethics and Information Technology,* Volume 6, p. 175–183.

Meijer, A., 2014. Transparency, The Oxford Handbook of Public Accountability..

Micheli, A., 2009. Neural network for graphs: A contextual constructive approach. *IEEE Transactions on Neural Networks,* Volume 20, p. 498–511.

Miller, T., 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence,* Volume 267, p. 1–38.

Min, Q. et al., 2019. Machine learning based digital twin framework for production optimization in petrochemical industry. *International Journal of Information Management,* Volume 49, pp. 502-519.

MIT Sloan Management Review, 2019. *Using Artificial Intelligence to Set Information Free.* s.l.:MIT Sloan Frontiers.

Mittelstadt, B. D. et al., 2016. The ethics of algorithms: Mapping the debate. *Big Data and Society,* Volume 3, p. 2053951716679679.

Mochaourab, R., Sinha, S., Greenstein, S. & Papapetrou, P., 2021. Robust Explanations for Private Support Vector Machines.

Molnar, C., 2019. *Interpretable Machine Learning. A Guide for Making Black Box Models Explainable..* s.l.:s.n.

Montavon, G. et al., 2017. Explaining nonlinear classification decisions with deep Taylor decomposition. *Pattern Recognition,* Volume 65, p. 211–222.

Monti, F. et al., 2017. *Geometric deep learning on graphs and manifolds using mixture model CNNs.* s.l., s.n., p. 5425–5434.

Moray, N. & Inagaki, T., 2000. Attention and complacency. *Theoretical Issues in Ergonomics Science,* Volume 1, pp. 354-365.

Mpofu, R. & Nicolaides, A., 2019. Frankenstein and the Fourth Industrial Revolution (4IR): Ethics and Human Rights Considerations. *Tourism and Leisure,* Volume 8, p. 25.

Muenning, P., 2008. Decision analytic modeling. In: *International Encyclopedia of Public Health.* s.l.:s.n., p. 71–76.

Murdoch, W. J. et al., 2019. Definitions, methods, and applications in interpretable machine learning. *Proceedings of the National Academy of Sciences of the United States of America,* Volume 116, p. 22071–22080.

Murphy, R. R. & Woods, D. D., 2009. Beyond asimov: The three laws of responsible robotics. *IEEE Intelligent Systems,* Volume 24, p. 14–20.

Narayanan, A. et al., 2017. *Graph2vec: Learning distributed representations of graphs.* s.l.:s.n.

Newman, M. E. J., 2006. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences of the United States of America,* Volume 103, p. 8577–8582.

Newman, M. E. J., Barabási, A. L. & Watts, D. J., 2011. *The structure and dynamics of networks.* s.l.:Princeton University Press.

Newman, M. E. J., n.d. *Networks: an introduction'.* Oxford(UK): Oxford University Press.

Ng, A., Jordan, M. & Weiss, Y., 2001. *On spectral clustering: Analysis and an algorithm.* Cambridge, MA, USA, MIT Press, pp. 849-856.

Nguyen, A. et al., 2017. *Plug and play generative networks: Conditional iterative generation of images in latent space.* s.l., s.n., p. 3510–3520.

Nguyen, A., Yosinski, J. & Clune, J., 2016. *Multifaceted Feature Visualization: Uncovering the Different Types of Features Learned By Each Neuron in Deep Neural Networks.* s.l.:s.n.

Nickel, M. & Kiela, D., 2017. Poincaré embeddings for learning hierarchical representations. In: *arXiv.* s.l.:s.n., p. 6338–6347.

Niepert, M., Ahmad, M. & Kutzkov, K., 2016. *Learning convolutional neural networks for graphs.* s.l., PMLR, p. 2958–2967.

Nori, H., Jenkins, S., Koch, P. & Caruana, R., 2019. *InterpretML: A unified framework for machine learning interpretability.* s.l.:s.n.

Nunes, M., 2015. *Statistical Analysis of Network Data with R.* New York, NY, USA: Springer.

Obregon, J., Kim, A. & Jung, J.-Y., 2019. RuleCOSI: Combination and simplification of production rules from boosted decision trees for imbalanced classification. *Expert Systems with Applications,* February.Volume 126.

Olah & al., e., 2017. Feature Visualization. *Distill.*

Olsen, J. P., 2014. Accountability and Ambiguity. *The Oxford Handbook of Public Accountability, edited by Mark Bovens, Robert E. Goodin, and Thomas Schillemans,* pp. 106-124.

Ou, M. et al., 2016. *Asymmetric transitivity preserving graph embedding.* s.l., s.n., p. 1105–1114.

Palhares, R. M., Yuan, Y. & Wang, Q., 2019. Artificial intelligence in industrial systems. *IEEE Transactions on Industrial Electronics,* Volume 66, p. 9636–9640.

Papernot, N. & McDaniel, P., 2018. *Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning.* s.l.:s.n.

Papernot, N. et al., 2017. *Practical black-box attacks against machine learning.* s.l., s.n., p. 506–519.

Parasuraman, R., Molloy, R. & Singh, I., 1993. Performance consequences of automation-induced complacency. *The international journey of aviation Psychology,* Volume 3, pp. 1-23.

Parasuraman, R. & Riley, V., 1997. Humans and automation: Use, misuse, disuse, abuse. *Human factors,* Volume 39, pp. 230-253.

Parasuraman, R. & Riley, V., 1997. Humans and automation: Use, misuse, disuse, abuse. *Human factors,* Volume 39, pp. 230-253.

Parasuraman, R., Sheridan, T. B. & Wickens, C. D., 2000. A model for types and levels of human interaction with automation. *IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans.,* Volume 30, p. 286–297.

Paredes-Astudillo, Y. A. et al., 2020. *Human Fatigue Aware Cyber-Physical Production System.* s.l., s.n.

Pareja, A. et al., 2019. EvolveGCN: Evolving Graph Convolutional Networks for Dynamic Graphs. *arXiv,* Volume 34, p. 5363–5370.

Pascanu, R., Mikolov, T. & Bengio, Y., 2013. On the difficulty of training recurrent neural networks. *30th International Conference on Machine Learning, ICML 2013,* Volume 5, p. 2347–2355.

Pasquale, F., 2015. *The Black Box Society: the secret algorithms that control money and information..* Cambridge: Harvard University Press.

Perc, M., Ozer, M. & Hojnik, J., 2019. Social and juristic challenges of artificial intelligence. *Palgrave Communications,* Volume 5, pp. 1-7.

Peres, R. S., Barata, J., Leitao, P. & Garcia, G., 2019. Multistage Quality Control Using Machine Learning in the Automotive Industry. *IEEE Access,* Volume 7, pp. 79908-79916.

Perozzi, B., Al-Rfou, R. & Skiena, S., 2014. *DeepWalk: Online learning of social representations.* s.l., s.n., p. 701–710.

Poolsappasit, N., Dewri, R. & Ray, I., 2012. Dynamic security risk management using Bayesian attack graphs. *IEEE Transactions on Dependable and Secure Computing,* 9(1), p. 61–74.

Pope, P. E. et al., 2019. *Explainability methods for graph convolutional neural networks.* s.l., s.n., p. 10764–10773.

Pynadath, D. V., Barnes, M. J., Wang, N. & Chen, J. Y. C., 2018. *Transparency Communication for Machine Learning in Human-Automation Interaction.* s.l.:s.n.

Quinlan, J. R., 1992. Learning with continuous classes. *Australian Joint Conference on Artificial Intelligence,* p. 343–348.

Quinlan, J. R., 1993. *C4.5: Programs for Machine Learning.* San Francisco, CA, USA: Morgan Kaufmann Publishers Inc..

Rader, E., Cotter, K. & Cho, J., 2018. *Explanations as mechanisms for supporting algorithmic transparency.* Montreal QC Canada, ACM, p. 1–13.

Rahman, M. S., 2017. Basic Graph Terminologies. In: *Basic Graph Theory', Undergraduate Topics in Computer Science.* Cham: Springer, p. 11–29.

Rahman, R. & De Feis, G., 2009. Strategic Decision-Making: Models and Methods in the Face of Complexity and Time Pressure. *Journal of General Management.*

Ran, Y. et al., 2019. A Survey of Predictive Maintenance: Systems, Purposes and Approaches. *arXiv e-prints.*

Ranzato, F., Urban, C. & Zanella, M., 2021. Fair Training of Decision Tree Classifiers. *arXiv e-prints,* January.p. arXiv:2101.00909.

Ranzato, F. & Zanella, M., 2020. Genetic Adversarial Training of Decision Trees. *arXiv e-prints,* December.p. arXiv:2012.11352.

Rao, D. J. & Mane, S., 2019. *Digital Twin approach to Clinical DSS with Explainable AI,* s.l.: s.n.

Reimann, J. & Sziebig, G., 2019. The Intelligent Factory Space – A Concept for Observing, Learning and Communicating in the Digitalized Factory. *IEEE Access,* Volume 7, pp. 70891-70900.

Ribeiro, M. T., Singh, S. & Guestrin, C., 2016. *"Why should i trust you?" Explaining the predictions of any classifier.* New York, NY, USA, Association for Computing Machinery (KDD, p. 1135–1144.

Ribeiro, M. T., Singh, S. & Guestrin, C., 2016. Model-Agnostic Interpretability of Machine Learning. In: *ICML Workshop on Human Interpretability in Machine Learning.* s.l.:s.n.

Roweis, S. T. & Saul, L. K., 2000. Nonlinear Dimensionality Reduction by Locally Linear Embedding. *Science,* 290(5500), pp. 2323-2326.

Samek, W., 2019. *Explainable AI: Interpreting, explaining and visualizing deep learning.* s.l.:Springer Nature.

Samuel, A. L., 1960. Some moral and technical consequences of automation - A refutation. *Science,* Volume 132, p. 741–742.

Scarselli, F. et al., 2009. The graph neural network model. *IEEE Transactions on Neural Networks,* 20(1), p. 61–80.

Schlegel, U. et al., 2019. *Towards a rigorous evaluation of XAI Methods on Time Series.* s.l., s.n.

Schmidt, P., Biessmann, F. & Teubner, T., 2020. Transparency and trust in artificial intelligence systems. *Journal of Decision Systems,* Volume 29, p. 260–278.

Seo, Y., Defferrard, M., Vandergheynst, P. & Bresson, X., 2018. *Structured sequence modeling with graph convolutional recurrent networks.* Cham, Springer, p. 362–373.

Shakerin, F. & Gupta, G., 2020. White-box Induction From SVM Models: Explainable AI with Logic Programming. *Theory and Practice of Logic Programming,* 20(5), pp. 656-670.

Sharchilev, B., Ustinovskiy, Y., Serdyukov, P. & de Rijke, M., 2018. *Finding Influential Training Samples for Gradient Boosted Decision Trees.* Stockholm, Sweden, PMLR, pp. 4577-4585.

Shchur, O., Mumme, M., Bojchevski, A. & Günnemann, S., 2018. *Pitfalls of graph neural network evaluation.* s.l.:s.n.

Shi, B. & Weninger, T., 2017. ProjE: Embedding projection for knowledge graph completion. *31st AAAI Conference on Artificial Intelligence, AAAI 2017,* Volume 31, p. 1236–1242.

Shrikumar, A., Greenside, P. & Kundaje, A., 2017. *Learning important features through propagating activation differences.* s.l., s.n., p. 4844–4866.

Siau, K. & Wang, W., 2018. Building trust in artificial intelligence, machine learning, and robotics. *Cutter Business Technology Journal,* Volume 31, pp. 47-53.

Simonovsky, M. & Komodakis, N., 2018. *Graphvae: Towards generation of small graphs using variational autoencoders.* Cham, Springer, p. 412–422.

Slack, D. et al., 2019. *Fooling LIME and SHAP: Adversarial Attacks on Post hoc Explanation Methods.* s.l., s.n.

Song, L., Zhang, Y., Wang, Z. & Gildea, D., n.d. *A graph-to-sequence model for AMR-to-text generation'.* s.l.:s.n.

Stohr, A. & O'Rourke, J., 2021. *Through the Cognitive Functions Lens - A Socio-Technical Analysis of Predictive Maintenance.* Essen, Germany, s.n.

Strobl, C. et al., 2008. Conditional Variable Importance for Random Forests. *BMC bioinformatics,* August.Volume 9.

Su, J., Vargas, D. V. & Sakurai, K., 2019. One Pixel Attack for Fooling Deep Neural Networks. *IEEE Transactions on Evolutionary Computation,* Volume 23, p. 828–841.

Sullins, J. P., 2011. When is a robot a moral agent. *Machine ethics,* Volume 6, pp. 151-161.

Sundararajan, M., Taly, A. & Yan, Q., 2017. *Axiomatic attribution for deep networks.* s.l.:s.n.

Susto, G. A. et al., 2015. Machine learning for predictive maintenance: A multiple classifier approach.. *IEEE Transactions on Industrial Informatics,* Volume 11, pp. 812-820.

Svenson, O. & Edland, A., 1993. *On judgment and decision making under time pressure and the control of process industries.* s.l., s.n., p. 367–373.

Szegedy, C. et al., 2014. *Intriguing properties of neural networks.* s.l., s.n.

Szpunar-Huk, E., 2006. *Classifier Building by Reduction of an Ensemble of Decision Trees to a Set of Rules.* Los Alamitos, CA, USA, IEEE Computer Society.

Taddeo, M. & Floridi, L., 2018. How AI can be a force for good. *Science,* Volume 361, p. 751–752.

Tang, J. et al., 2015. *LINE: Large-scale information network embedding.* s.l., s.n., p. 1067–1077.

Tan, K. H. et al., 2015. Harvesting big data to enhance supply chain innovation capabilities: An analytic infrastructure based on deduction graph. *International Journal of Production Economics,* Volume 165, p. 223–233.

Tan, S., Soloviev, M., Hooker, G. & Wells, M. T., 2020. *Tree Space Prototypes: Another Look at Making Tree Ensembles Interpretable.* Virtual Event, USA, Association for Computing Machinery, p. 23–34.

Tenenbaum, J. B., De Silva, V. & Langford, J. C., 2000. A global geometric framework for nonlinear dimensionality reduction. *Science,* Volume 290, p. 2319–2323.

Thuy, N., Vinh, N. & Vien, N., 2011. *Nomogram Visualization for Ranking Support Vector Machine.* Berlin, Heidelberg, Springer Berlin Heidelberg, pp. 94-102.

Tigard, D. W., 2020. Responsible AI and moral responsibility: a common appreciation. *AI and Ethics,* p. 43681–020–00009–0.

Tolomei, G., Silvestri, F., Haines, A. & Lalmas, M., 2017. Interpretable Predictions of Tree-based Ensembles via Actionable Feature Tweaking. *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining,* August.

Törnblom, J. & Nadjm-Tehrani, S., 2020. Formal verification of input-output mappings of tree ensembles. *Science of Computer Programming,* August, Volume 194, p. 102450.

Torres, D. E. D. & Rocco, C. M. S., 2005. *Extracting trees from trained SVM models using a TREPAN based approach.* Rio de Janeiro, Brazil, Institute of Electrical and Electronics Engineers.

Torresen, J., 2018. *A review of future and ethical perspectives of robotics and AI. Frontiers in Robotics and AI.* s.l.:s.n.

Üstün, B., Melssen, W. J. & Buydens, L. M. C., 2007. Visualisation and interpretation of Support Vector Regression models. *Analytica Chimica Acta,* 595(Papers presented at the 10th International Conference on Chemometrics in Analytical Chemistry), pp. 299-309.

Van Belle, V. et al., 2016. Explaining Support Vector Machines: A Color Based Nomogram. *PLOS ONE,* 11(10), pp. 1-33.

Van De Poel, I. et al., 2015. *Moral responsibility and the problem of many hands.* s.l.:Routledge Taylor & Francis Group.

Van Der Maaten, L. & Hinton, G., 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research,* Volume 9, p. 2579–2625.

Van Looveren, A. & Klaise, J., 2019. Interpretable Counterfactual Explanations Guided by Prototypes. *arXiv e-prints,* p. arXiv:1907.02584.

Vandewiele, G. et al., 2017. *A Genetic Algorithm for Interpretable Model Extraction from Decision Tree Ensembles.* s.l., s.n., pp. 104-115.

Velicković, P. et al., 2017. *Graph attention networks.* s.l.:s.n.

Veličković, P. et al., 2018. *Deep graph infomax.* s.l.:s.n.

Vidal, T., Pacheco, T. & Schiffer, M., 2020. *Born-Again Tree Ensembles.* s.l., ICML.

Vishwanathan, S. V. N., Schraudolph, N. N., Kondor, R. & Borgwardt, K. M., 2010. Graph kernels. *Journal of Machine Learning Research,* Volume 11, p. 1201–1242.

Vroom, V. H. & Yetton, P. W., 1973. *Leadership and Decision_making.* Pittsburgh: University of Pittsburgh Press.

Vroom, V. & Jago, G., 1988. *The new leadership: Managing Participation in Organizations.* Englewood Cliffs: Pretience-Hall.

Wachter, S., Mittelstadt, B. & Floridi, L., 2017. Transparent, explainable, and accountable AI for robotics. *Science Robotics,* Volume 2.

Wagner, A. R. & Robinette, P., 2015. Towards robots that trust. *Interaction Studies. Social Behaviour and Communication in Biological and Artificial SystemsInteraction Studies / Social Behaviour and Communication in Biological and Artificial SystemsInteraction Studies,* Volume 16, p. 89–117.

Wagner, A. R. & Robinette, P., n.d. *Towards robots that trust, is.16.1.05wag.* s.l.:John Benjamins Publishing Company.

Wang, D., Cui, P. & Zhu, W., 2016. *Structural deep network embedding.* s.l., s.n., p. 1225–1234.

Wang, H. et al., 2018. *GraphGAN: Graph representation learning with generative adversarial nets.* s.l., s.n., pp. 2508-2515.

Wang, J. et al., 2018. Deep learning for smart manufacturing: Methods and applications. *Journal of Manufacturing Systems,* 48(Special Issue on Smart Manufacturing), pp. 144-156.

Wang, J. et al., 2018. Deep learning for smart manufacturing: Methods and applications.. *Journal of Manufacturing Systems,* Volume 48, pp. 144-156.

Wang, S., Liu, X., Yang, T. & Wu, X., 2018. Panoramic Crack Detection for Steel Beam Based on Structured Random Forests. *IEEE Access,* Volume 6, pp. 16432-16444.

Wang, X. et al., 2019. *Heterogeneous graph attention network.* s.l., s.n., p. 2022–2032.

Wang, X. et al., 2018. Explainable reasoning over knowledge graphs for recommendation. *arXiv,* p. 5329–5336.

Weick, K. E. & Roberts, K. H., 1993. Collective Mind in Organizations: Heedful Interrelating on Flight Decks. *Administrative Science Quarterly,* Volume 38, p. 357.

Wiener, N., 1969. Some moral and technical consequences of automation. *Science,* Volume 131, p. 1355–1358.

Willamson, A., 1997. *Trust: The Social Virtues and the Creation of Prosperity.* New York: Free press.

Wisdom, S., Powers, T., Pitton, J. & Atlas, L., 2016. *Interpretable recurrent neural networks using sequential sparse recovery.* s.l., s.n.

Wu, Z. et al., 2021. A Comprehensive Survey on Graph Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems,* Volume 32, p. 4–24.

Wu, Z. et al., 2019. *Graph wavenet for deep spatialoral graph modeling.* s.l.:s.n.

Xiao, T. et al., 2015. *The application of two-level attention models in deep convolutional neural network for fine-grained image classification.* s.l., s.n., p. 842–850.

Xu, D. et al., 2020. *Inductive representation learning on temporal graphs.* s.l.:s.n.

Yanardag, P. & Vishwanathan, S. V. N., 2015. *Deep graph kernels.* s.l., s.n., p. 1365–1374.

Yan, S., Xiong, Y. & Lin, D., 2018. Spatial temporal graph convolutional networks for skeleton-based action recognition. *arXiv,* Volume 32.

Yeh, C.-K., Kim, J. S., Yen, I. E. ~. & Ravikumar, P., 2018. Representer Point Selection for Explaining Deep Neural Networks. *arXiv e-prints,* November.p. arXiv:1811.09720.

Ying, R. et al., 2019. GNNExplainer: Generating explanations for graph neural networks. *arXiv,* Volume 32, p. 9240.

Ying, R. et al., 2018. *Graph Convolutional Neural Networks for Web-Scale Recommender Systems.* s.l., s.n., p. 974–983.

You, J. et al., 2018. *Graph convolutional policy network for goal-directed molecular graph generation.* s.l.:s.n.

You, J. et al., 2018. *GraphRNN: Generating realistic graphs with deep auto-regressive models.* s.l., PMLR, p. 5708–5717.

Yu, B., Yin, H. & Zhu, Z., 2017. *Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting.* s.l.:s.n.

Zarsky, T., 2016. The Trouble with Algorithmic Decisions: An Analytic Road Map to Examine Efficiency and Fairness in Automated and Opaque Decision Making. *Science Technology and Human Values,* Volume 41, p. 118–132.

Zarsky, T., 2016. The Trouble with Algorithmic Decisions: An Analytic Road Map to Examine Efficiency and Fairness in Automated and Opaque Decision Making. *Science, Technology, & Human Values,* Volume 41, p. 118–132.

Zerilli, J., Knott, A., Maclaurin, J. & Gavaghan, C., 2019. Transparency in Algorithmic and Human Decision-Making: Is There a Double Standard?. *Philosophy and Technology,* Volume 32, p. 661–683.

Zhang, C., Zhang, H. & Hsieh, C.-J., 2020. *An Efficient Adversarial Attack for Tree Ensembles.* s.l., Curran Associates, Inc., pp. 16165-16176.

Zhang, Y., Vera Liao, Q. & Bellamy, R. K. E., 2020. *Efect of confidence and explanation on accuracy and trust calibration in AI-assisted decision making.* Barcelona Spain, ACM, p. 295–305.

Zhang, Z., Cui, P. & Zhu, W., 2020. Deep Learning on Graphs: A Survey. *IEEE Transactions on Knowledge and Data Engineering.*

Zhou, D., Schölkopf, B. & Hofmann, T., 2005. Semi-supervised learning on directed graphs. In: *Advances in Neural Information Processing Systems.* s.l.:s.n., p. 1633–1640.

Zhou, J. et al., 2020. Graph neural networks: A review of methods and applications. *AI Open,* Volume 1, pp. 57-81.

Zhuang, C. & Ma, Q., 2018. *Dual graph convolutional networks for graph-based semi-supervised classification.* s.l., s.n., p. 499–508.

Zilke, J. R., Mencía, E. L. & Janssen, F., 2016. *DeepRED – Rule extraction from deep neural networks.* s.l., Springer Verlag, p. 457–473.

Zügner, D., Akbarnejad, A. & Günnemann, S., 2018. *Adversarial attacks on neural networks for graph data.* s.l., s.n., p. 2847–2856.

Zügner, D. & Günnemann, S., 2019. *Adversarial Attacks on Graph Neural Networks Via Meta Learning.* s.l.:s.n.

List of Acronyms/Abbreviations

Acronym/ Abbreviation	Description
AI	Artificial Intelligence
ANN	Artificial Neural Network
CNN	Convolutional Neural Network
DL	Deep Learning
DNN	Deep Neural Network
DoA	Description of Action
GAM	Generalised Additive Models
GF	Graph Factorization
GLM	Generalised Linear Models
GNN	Graph Neural Networks
GP	Genetic Programming
G-REX	Genetic-Rule EXtraction
GRL	Graph Representation Learning
ILP	Inductive Logic Programming
KPI	Key Performance Indicator
LIMA	Local Interpretable Model-Agnostic
LSTM	Long Short-Term Memory
MDS	Multi-Dimensional Scaling
ML	Machine Learning
MMD	Maximum Mean Discrepancy
MVP	Minimum Viable Product
NLP	Natural Language Processing
NN	Neural Network
PDP	Partial Dependence Plot
RL	Reinforcement Learning
RNN	Recurrent Neural Network
SHAP	SHapley Additive exPlanations
SotA	State-of-the-Art
SRM	Structural Risk Minimization
STGNN	Spatio-Temporal Graph Neural Network

Acronym/ Abbreviation	Description
SV	Support Vector
SVD	Singular Value Decomposition
SVM	Support Vector Machine
WP	Work Package
XAI	eXplainable Aritical Intelligence

Annexes

Annex I - Interviews with decision makers: questions

The following questions are thought to understand how you usually manage the decision making process, to identify methods and techniques adopted and to identify your propensity to AI adoption (pros and cons, issue faced, obstacles and difficulties).

No predefined answers are provided: questions require open answers, and they represent the skeleton to put in light which are the main topics to be analysed. So, feel free to tell us your experiences, answering to questions in the order you prefer and merging them, if required.

Decision making process:

1. **DECISIONS AREA**: Which are the main situations where you are asked to make decision (business, human resource, maintenance, planning…)? Please, provide some examples

2. **DECISION MAKER STYLE**: According to literature, a decision maker can be defined as "autocratic", "consultative" and "collaborative"; decision can be made individually or in group. Do you think that the decision maker's approach may depend only on personality or may change according to the situation?

3. **TECHNIQUES**: Decision Tree, Decision Matrix, Influence matrix, Pros and Cons list, Brainstorming, structured meeting, questionnaire, are all different techniques and methods used in decision making process, considering individual or group activities. Which are the main method and techniques that you usually adopt?

4. **REMOTE DECISIONS**: Considering mainly the last period when lot of people have been forced to remote working, how much the decision making process has been affected by remote meeting?

5. **CASE STUDY (external factors)**: Time pressure → Consider the following situation: you'd like to start selling a new product, that very likely will increase revenues but that requires expensive changes to the plant/organization. Which steps do you follow to make the decision? What happens if you find out that a competitor is going to sell the same and so, you need to rush? [How does your behaviour change in case of time pressure? What are the steps that you skip?]

AI adoption in decision making process:

1. **AI vs HUMAN (benefit from AI)**: Have you identified any limitations on the current human-centric decision-making process? Do you believe that an AI system could support you in taking better decisions? Based on your knowledge, which areas / applications would be more benefited by the use of AI?

2. **AI vs HUMAN (explainable AI)**: Have you ever experienced a situation when the use of AI has made the decision more complicated? For example because the result provided by the model was counterintuitive?

3. **CASE STUDY (how to choose AI solutions)**: Assuming that your company asks our help to purchase a new AI software. There are two options. The first one is very expensive but it is more accurate and it is autonomous to make-decisions. The second one is within your budget

but requires as well as the human knowledge and/or the validation of the results. Which one would you recommend according to your expertise?

4. **AI vs HUMAN (current situation)**: Are you satisfied with your current AI software for decision-making, if any? If not, why? How much do you trust the upcoming decisions from the AI software?

5. **AI vs HUMAN (human replacement)**: In your company, how much the AI has replaced humans focusing on the decision-making process?

6. **CHANGE OF WORKFORCE SKILLS:** Do you think that required competencies and skills are changed due to the adoption of AI in decision making? If yes, which level of the workforce has been impacted the most?

Ethical issues in AI adoption:

1. **AI vs HUMAN (ethical issue AI)**: Based on your experience, which aspect of AI use, especially the most implicated in the interaction with operators, is more crucial on an ethical point of view (e.g. privacy, control, deskilling, etc). Have you identified any ethical issues on the current AI- human interaction in your working environment?

2. **AI trust (explainability AI)**: What is your general feeling of trust toward AI making a decision? According to your experience and personal point of view, how much important is that an AI system is transparent and easy to understand? In which context or under which circumstances you will not let an AI system making decisions by itself? Why?

3. **Actual experience**: Have you ever experienced a situation when the use of AI has made the decision ethically wrong according to your point of view? If yes, in which terms?

4. **What if scenario**: Let imagine an AI system that usually works effectively and efficiently but its functioning is completely secret to you. What if you should take a critical decision and the AI system provides you with a potential decision? What if the AI system works alone against decisions?

5. **What-if scenario**: AI enables predictive maintenance, real-time analysis and optimisation of output quality, and a more objective measurement of overall performance, to the level of single specific inputs and processes. AI focuses on the growing use of big data, insights and smart machines to optimise production processes and enable greater customisation of complex products. Imagine production is optimised through improved data collection. Data are processed automatically, quickly and easy providing the user with direct, personalized information about the status of the production and the individual products, supporting better operation and maintenance. Suddenly something goes wrong during a very large production and the AI system for the analysis of the performance get stuck. What will be needed when failures will require getting “under the bonnet” and taking over the work? How to ensure the user is capable of following what is happening and intervene when the machine fails?