UiA University of Agder

Master

Federated Machine Learning: Privacy, Explainability, and Performance

HALVOR SKRIBELAND

SUPERVISOR
Christian Walter Peter Omlin

University of Agder, 2024
Faculty of Engineering and Science
Department of Information and Communication Technology

Obligatorisk gruppeerklæring

Den enkelte student er selv ansvarlig for å sette seg inn i hva som er lovlige hjelpemidler, retningslinjer for bruk av disse og regler om kildebruk. Erklæringen skal bevisstgjøre studentene på deres ansvar og hvilke konsekvenser fusk kan medføre. Manglende erklæring fritar ikke studentene fra sitt ansvar.

1.	Vi erklærer herved at vår besvarelse er vårt eget arbeid, og at vi ikke har brukt andre kilder eller har mottatt annen hjelp enn det som er nevnt i besvarelsen.	Ja
2.	**Vi erklærer videre at denne besvarelsen:** • Ikke har vært brukt til annen eksamen ved annen avdeling/universitet/høgskole innenlands eller utenlands. • Ikke refererer til andres arbeid uten at det er oppgitt. • Ikke refererer til eget tidligere arbeid uten at det er oppgitt. • Har alle referansene oppgitt i litteraturlisten. • Ikke er en kopi, duplikat eller avskrift av andres arbeid eller besvarelse.	Ja
3.	Vi er kjent med at brudd på ovennevnte er å betrakte som fusk og kan medføre annullering av eksamen og utestengelse fra universiteter og høgskoler i Norge, jf. Universitets- og høgskoleloven §§4-7 og 4-8 og Forskrift om eksamen §§ 31.	Ja
4.	Vi er kjent med at alle innleverte oppgaver kan bli plagiatkontrollert.	Ja
5.	Vi er kjent med at Universitetet i Agder vil behandle alle saker hvor det forligger mistanke om fusk etter høgskolens retningslinjer for behandling av saker om fusk.	Ja
6.	Vi har satt oss inn i regler og retningslinjer i bruk av kilder og referanser på biblioteket sine nettsider.	Ja
7.	Vi har i flertall blitt enige om at innsatsen innad i gruppen er merkbart forskjellig og ønsker dermed å vurderes individuelt. Ordinært vurderes alle deltakere i prosjektet samlet.	Ja

Publiseringsavtale

Fullmakt til elektronisk publisering av oppgaven Forfatter(ne) har opphavsrett til oppgaven. Det betyr blant annet enerett til å gjøre verket tilgjengelig for allmennheten (Åndsverkloven. §2).
Oppgaver som er unntatt offentlighet eller taushetsbelagt/konfidensiell vil ikke bli publisert.

Vi gir herved Universitetet i Agder en vederlagsfri rett til å gjøre oppgaven tilgjengelig for elektronisk publisering:	Ja
Er oppgaven båndlagt (konfidensiell)?	Nei
Er oppgaven unntatt offentlighet?	Nei

Acknowledgements

I would like to express my deepest gratitude to my supervisor, Professor Christian Walter Peter Omlin for their guidance and deep insights throughout this thesis.

Abstract

There is an increasing need for explainable and private machine learning. The European Union's AI Act is a recent legislation aimed at regulating the development and use of artificial intelligence in the European Union. Trustworthy AI is an important part of this, and some of the key requirements for trustworthy AI are data privacy and model transparency. Takagi-Sugeno-Kang fuzzy rule-based systems (TSK-FRBS) are inherently explainable, and federated learning (FL) is a way to train machine learning (ML) models while ensuring data privacy. Training an inherently explainable ML model using FL has the potential to ensure data privacy while training a transparent model.

This thesis empirically investigates the possible trade-offs between privacy and the performance of inherently explainable ML models and deep learning (DL) models. Does federated learning reduce the performance of either TSK-FRBS or deep learning models? To answer this question, a central TSK-FRBS, a federated TSK-FRBS, a central DL, and a federated DL model have been implemented on ten different datasets from different application areas.

The federated models have been trained using i.i.d data and five collaborators. The experiments show that federated learning has not significantly impacted the performance and explainability of the different models. The central models performed comparably to the federated models trained on the same datasets. The deep learning models perform slightly better than the TSK-FRBS model overall, with some exceptions. The TSK-FRBS model is explainable because it is transparent and consists of rules. Training the TSK-FRBS model using FL does not alter the rule in any way that reduces its explainability. The results suggest there is no reason not to train a TSK-FRBS using FL regarding explainability. To the researcher's knowledge, this is the only research that compares inherently explainable ML and DL to investigate the impact of FL.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The fourth industrial revolution is transforming industries through automation, digitalization, and new technologies. The goal of the revolution is to increase efficient real-time data flow in order to increase productivity, quality, and adaptability. Industrial processes can be automated using smart technology and connected systems and machines. This strive for automation presents new challenges and the need for new tools. Artificial Intelligence (AI) has proven to be a powerful tool to tackle these challenges, and advancements in AI have led to the development of capable and intelligent systems. [5]

The European Union has taken a proactive approach when it comes to regulating AI. With the aim of steering the field towards creating trustworthy AI, the European Parliament created the Artificial Intelligence Act. The Act was enacted March 13, 2024 [6], and covers all the AI systems used in the EU. The European Parliaments goal with the AI legislation is to ensure safe, transparent, traceable, non-discriminatory and environmentally friendly AI. They also state that humans should oversee AI, instead of automated systems, to prevent negative consequences [7, 8]. The AI Act use a risk level approach where there are different rules for different risk levels. Furthure, the AI Act encourages the development and use of trustworthy AI, and the European Commission has defined some guidelines for trustworthy AI [8]. According to the guidelines, an AI system should meet 7 key requirements to be deemed trustworthy. Those are **human and agency oversight**, **technical robustness and safety**, **privacy and data governance**, **transparency**, **diversity, non discrimination and fairness**, and **accountability** [9].

Creating an AI system for an application considered high-risk can be a challenge. Meeting the key requirements of trustworthy AI may lead to reduced model performance, and the models must be transparent or explainable, and at the same time keep the training data private. Systems that are not considered high risk will still have to adhere to the AI Act rules, but the rules are not as strict as the high-risk applications. Programs that do not affect safety or fundamental rights, such as ChatGPT, will not be considered high-risk. A few example application areas that are considered high risk are; aviation, cars, medical systems, employment, legal assistance, and migration. [7]

Inherently explainable machine learning (ML) models and federated learning (FL), are two

technologies that can comply with trustworthy AI's transparency, privacy, and data governance requirements. Inherently explainable machine learning models such as Takagi-Sugeno-Kang Fuzzy Rule-Based Systems (TSK-FRBS), are transparent and explainable. A rule-based system consists of a set of rules written in a similar way to natural language. It is possible to understand how a model makes its decision by analyzing an activated rule and its fuzzy membership. FL is a privacy-preserving technology that allows multiple clients to train an AI model collaboratively. Each client contributes by training a local model on their data. Furthermore, the global model is an aggregated model, and the training data never leaves the clients. The combination of TSK-FRBS and FL ensures the explainability and privacy requirements.

AI has a vast potential when it comes to solving complex problems in different fields such as e.g. the medical sector. Its application in high risk areas brings concerns about the explainability and privacy. Research towards explainable and privacy focused ML techniques is important to ensure responsible and trustworthy AI. Privacy and explainability can help us understand how a model "arrives" at its decision while ensuring the privacy of the data. Good explainable and private ML models can benefit high-risk sectors while beeing ethical and trustworthy.

1.1 Research Question

According to Cynthia Rudin [10], there is a myth commonly believed by researchers that there is a trade-off between accuracy and interpretability. Complex models are not necessarily more accurate. Nilsson et al. [11] demonstrates that the performance of FL using FedAvg, and centralized learning, are practically the same when using identically and independently distributed (i.i.d) data. Their results were achieved using neural networks. An interesting thing to evaluate is whether similar performance also applies to inherently explainable models such as TSK-FRBS.

This thesis intends to empirically investigate possible trade-offs between privacy and the performance of an inherently explainable machine learning algorithm and deep learning (DL) models. The privacy preserving technique used to answer this question is cross-silo Federated Learning. The explainability aspect is achieved by using the inherently explainable machine learning model TSK-FRBS and compare it with DL models.

The research question the thesis aims to answer is the following;
Does federated learning reduce the performance of a TSK-FRBS model or a DL model

To help answer this question a set of sub-questions have been derived.

- Can a cross-silo Federated TSK-FRBS model with i.i.d data achieve comparable performance to a central TSK-FRBS model?
- Do FL affect the explainability of a TSK-FRBS models rules?
- Can a cross-silo federated DL model with i.i.d data achieve the same performance as a central DL model?
- How does the federated TSK-FRBS models perform compared to the federated DL models

on the same datasets?

1.2 Contribution

FL is a growing field that is gaining traction because of its privacy preserving abilities. FL was designed for deep learning, which has resulted in research primarily being focused on DL. This thesis aims to address the research gap in understanding the impact of FL on inherently explainable ML algorithms. This research thesis aims to contribute the research on FL by doing an empirical study on inherently explainable ML models. The research conducted in this thesis builds upon the work of OpenFL-XAI [12]. The contributions of this thesis are to implement federated TSK-FRBS, central TSK-FRBS, federated DL and central DL models on a set of 10 datasets. The results are then compared, and the impact of federated learning on TSK-FRBS models are evaluated.

1.3 Structure

The rest of this thesis is structured as follows:

- **Section 2 Background:** This section contains background information on the main technologies used in this thesis. This includes AI Act, explainable AI, fuzzy rule-based systems, privacy-preserving ML, and Federated Learning.
- **Section 3 State of the art:** This section looks at existing research on the technologies used and what other researchers have done.
- **Section 4 Methodology:** This section present methodology. It covers the research design, model implementations, datasets, and how the results are evaluated.
- **Section 5 Results:** This section presents the results from the experiments conducted in this thesis.
- **Section 6 Discussion:** This section discusses and interpret the results. This section contains the discussion. The different results and experiments are discussed and interpreted. What do the results mean, and why are they the way they are.
- **Section 7 Conclusion:** is the conclusion. This is the final section of the thesis before the references and the conclusion of the findings.

Chapter 2

Background / Literature Review

Previous work on FL of inherently explainable models

The background section presents the technologies and areas of research covered in this thesis. These are Explainable Machine learning, privacy-preserving Machine Learning, and Artificial Neural Networks. The European Union's regulations on AI, the AI Act, will be presented before looking at Explainable Machine Learning, Privacy-Preserving Machine Learning, and Artificial Neural Networks. This will be the theoretical background for the rest of the thesis.

2.1 AI Act

To ensure safe and sound conditions for developing and using AI systems, the European Union has enacted a set of rules covering AI. The AI Act is a legal framework on AI [13]. The European Union's parliament enacted it on March 13 2024, and it is the worlds first comprehensive legal framework for AI. By enacting a set of rules for AI, the EU wants to steer the development and use of AI to benefit people. The EU Act hopes to build trust in what AI can contribute to society. [13]

The EU Act is a risk-based approach and divides AI applications into four risk categories: Unacceptable risk, High risk, limited risk, and minimal risk. The risk categories say something about what the potential consequences of the application are. A high-risk application has to follow a stricter set of rules than a minimal-risk application. The higher the application risk, the more stringent the rules to adhere to. Examples of high-risk systems include the medical sector, critical infrastructure, immigration, and legal help. Before an AI system can be put on the market, it has to follow a strict set of obligations. [13]

The EU wants its inhabitants to be able to trust AI systems. They hope to achieve this by regulating the development and use of trustworthy AI. The EU's considerations for trustworthy AI consist of seven requirements. Two of them are privacy and transparency. The privacy guidelines make sure that personal data is handled ethically and in line with the data protection regulations. Transparency allows for insight into how a system works. Understanding how a model has arrived at its decision can lead to better explainability and lower the chance of model biases. [14, 13]

Developing good AI systems for high-risk applications such as in the medical sector can be difficult. A good AI model requires a good set of training data. This is a challenge in the medical sector as patient data and hospital records are private and regulated. The models must also be explainable so a medical professional can understand the what and why a suggestion have been made.

2.2 Explainable AI (XAI)

XAI stands for Explainable AI. It is a field of research that aims to make machine learning models understandable to humans. Machine learning models can be divided into two categories: white-box and black-box. White-box models are transparent and understandable, and humans can interpret their inner workings. Black-box models are more complex but can achieve a better accuracy. XAI is an important area of research as more complex machine learning models, such as deep neural networks, are used. The results DL models can achieve are impressive, but it can be not easy to understand how they are achieved. [15, 16]

Van Lent et al. [17] introduced the word XAI in 2004. They created an AI model to control a simulation and explain its decisions. The term XAI describes the explainability part of the AI model. The growing use of deep learning models has made it clear that more explanation is needed, leading to an increased number of studies on XAI. Deep learning models are considered black-box models. They return an output, but the reasoning behind the decision is unknown. More services are using AI every day. Decisions are made without knowing the reasoning behind them. Some examples are movie recommendations on Netflix, personalized advertisements from Google, and friend suggestions on Facebook. There are also several areas where AI is limited because there is a need for better understanding and reasoning of decisions, such as in the medical and financial sectors. Explainable AI can enable the development and use of applications in high-risk sectors. Explainability can increase trust and transparency and help people better understand why an AI model makes the decisions it makes. [18, 16]

ML is a powerful technology and is used in many different sectors. It can help solve complex tasks and is an important tool for the future. When AI systems are used in applications that affect human lives, it is important to understand its decisions. [19] There are several reasons why explainability is important. Explainable AI can also lead to better-performing models. It allows researchers to better understand the model they are working on. A deeper understanding can also lead to a fairer model. With the EU's AI Act, it is not enough for a model to be accurate; it must also be explainable and fair. Increasing demand for ethical AI means that models have to be interpretable and trustworthy. For a model to comply with the AI Act, it has to be non-discriminating and every person has the right to explanation. [20, 13].

There is a general need for explainable AI models, but the level of explainability does not have to be the same across all applications. AI models used in the healthcare sector requires a good level of explainability. This is because the outcome of a decision can have a negative impact on human lives. A doctor needs to understand the decision of an AI model and its reasoning and agree that it is a good decision. For other applications, such as personalized advertisement,

the explainability does not need to be as comprehensive as with the healthcare sector. The consequences of a personalized advertisement to promote the wrong product to a person are negligible. [16]

According to [18], two terms are used a lot in papers discussing explainability in AI. Those terms are explainability and interpretability. There is no official explanation or common consensus on what explainability and interpretability in AI are. To clear this up, [18] defines the terms to help make the definitions and distinctions between them clear. Explainability helps a target audience understand or gain knowledge about a model's decision-making or function based on some explainability techniques. Interpretability means that the results and explanations make sense to a target audience. Can a doctor understand the reasoning of a model and use it to make informed decisions? [18]

2.2.1 Achieve explainability

Explainable AI can be achieved in a few different ways, either through ad-hoc or post-hoc methods. Models that are ad-hoc explainable have explainability built into the model. Ad-hoc models are designed to be explainable, and it's easier to understand their decision-making process. Transparent ML models that are explainable are called inherently explainable models. They are usually more straightforward and less complex. Decision trees, rule-based, and linear regression models are examples of inherently explainable models. Post-hoc explainability means to explain a model after it has made its prediction. Post-hoc explainability techniques can be applied to a wide range of models, both ML and DL models. As DL models are considered black-box models, post-hoc explainability can provide insight into how they work. [19] There are many different post-hoc explainability techniques. Some of the most popular ones are LIME, SHAP, and ELI5. They are applied after training and help understand why a model makes certain predictions.

Recent research in XAI has focused on DL models. This thesis will not explore the details of XAI techniques as they have not been implemented or used. The explainability of the inherently explainable model TSK-FRBS are considered by evaluating the model rules.

2.2.2 Fuzzy Rule-Based Systems

Fuzzy Rule-Based Systems (FRBS) are an inherently explainable machine learning method that uses rules and fuzzy logic to make decisions. Rule-based systems can be used to solve a wide range of problems, such as regression and classification problems. A rule in a rule-based system has the form "IF antecedent THEN consequent." This structure is called the IF-THEN rule-base form. A rule-based system consists of multiple rules, and the rules decide a model's output. FRBS can handle data uncertainty by representing knowledge using fuzzy sets and fuzzy logic. [2]

Fuzzy sets are a mathematical concept that can represent uncertainty in data. This is done using partial membership, allowing data to belong to multiple elements in a set. Imagine a fuzzy set representing temperature preference using the elements cold, comfortable, and warm.

A membership function is used to determine at which degree a temperature belongs to the categories. A temperature of 10°C can have a membership of 0.9 in cold and 0.1 in comfortable. This indicates that the temperature is mostly considered cold but also slightly comfortable. Fuzzy sets allow for a data-entry to belong to multiple sets at the same time. Figure 2.1 shows a fuzzy partition with the elements low, medium, and high. Fuzzy logic is an extension of fuzzy sets that allows logical operators to be used with degrees of membership. They are allowing for the more complex description of temperature such as "cold" AND "comfortable". [21, 22]

There are a few different types of uncertainty partitions. Crisp uncertainty (crisp) is when a value only belongs to one interval. First-order uncertainty (type-1) has overlapping intervals. The start and end of each partition are known, and the membership value is between 0 and 1. The overlap between intervals makes for smooth, specific, and defined overlap. Second-order uncertainty (type-2) is similar to first-order uncertainty except that there is uncertainty in the overlap. The start and end of the overlap are unknown, so the membership is an interval. [2]

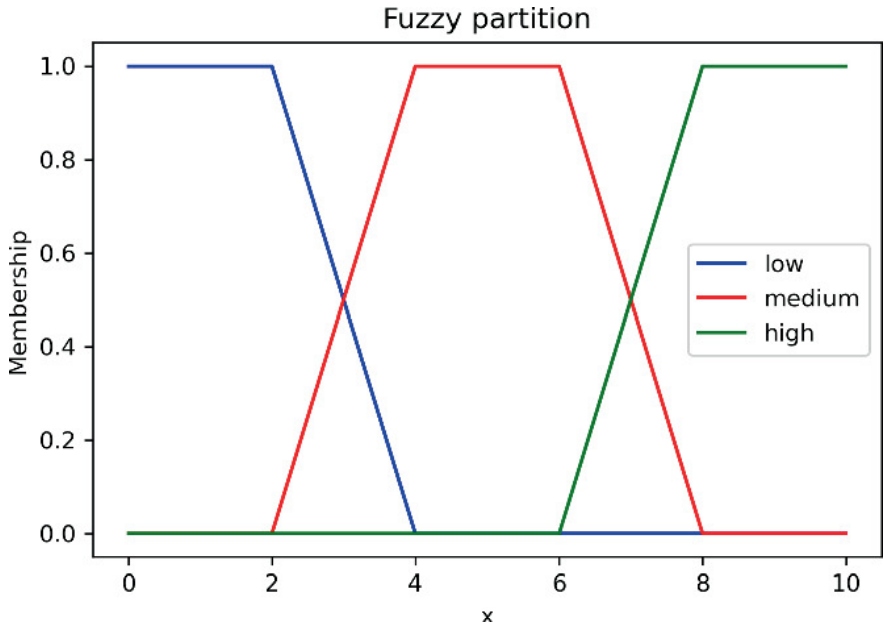

Figure 2.1: Fuzzy partitioning [1]

A FRBS model is built up by a number of rules. Each rule captures a different behavior, and, together can make decisions. An example of a rule is:
IF x1 is "medium" and x2 is "low" and x3 is "high" THEN y is "cold".
There are two main ways the rules of an FRBS are initialized. A domain expert can create the rules, or an algorithm can do it. If an expert creates the rules, they use their knowledge and experience to create well-designed rules that cover different circumstances that can happen. An example of an algorithm that can initialize and create fuzzy rules is the fuzzy c-means algorithm. The fuzzy c-means algorithm is very similar to the k-means algorithm, but instead of having each datapoint assigned to only one centroid, the datapoints belong to all the centroids at different degrees of membership. [23]

An example to understand how a rule-based system works can be to look at the heating of a

room. You have the input which is the room temperature, and the output is what power the heater should be. The fuzzy sets for this system can be hot, warm and cold for the temperature in the room, low, medium and high for the power of the heater. X is the temperature and Y is the output power to the heater. The rules for this system can be the following: R1: IF X is hot THEN Y is low R2: IF X is warm THEN Y is medium R3: IF X is cold THEN Y is high When a temperature is given to the model it will first decide to what degree of the fuzzy sets it belongs to. for example a temperature of 4 degrees Celsius and 14 degrees Celsius might both be considered cold, but the 14 degrees temperature belongs more to the warm category than the 4 degrees one.

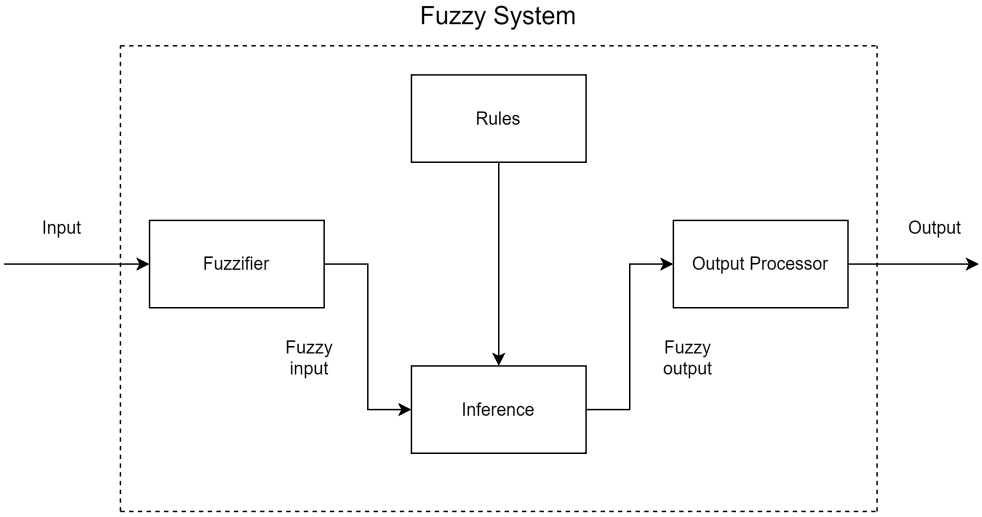

Figure 2.2: Fuzzy System [2]

A fuzzy system consists of four parts: rules, fuzzifier, inference, and output processor. Rules are the central part of an FRBS and are either extracted from some data or created by domain experts. If the fuzzy sets used for the rules are type-1, the fuzzy system is called a type-1 fuzzy system. The next part of a fuzzy system is the fuzzifier. The fuzzifier turns the input data into fuzzy values. The next part of a fuzzy system is the inference process. The fuzzified inputs are compared to the rules. The input data will match one of the rules where the fuzzy values are the most similar, and that rule's consequent (output) will be the result for that input. The final part of the system is the output processor. The output from the inference step is processed to produce the type of output the system is designed for. [2]

Fuzzy Rule-based systems are considered inherently explainable. This is because a FRBS-model consists of a set of rules. By accessing the rules a model follows when making a decision, it becomes possible to understand it better. By looking at the model's rules, you can, for example, see that if you have some input data where one variable is high and another is low, the model might make a certain prediction because the data fits best to a certain rule. According to the survey by Varshney et al. [24], fuzzy rule-based systems work well on unbalanced datasets.

There are a few different versions of Fuzzy Rule-Based systems. They all follow the main principles of a fuzzy system, but with a few differences. The first person to implement an FRBS

with real inputs and outputs is Mandami [21, 25]. His implementation is called Mandami-FRBS, one of the most popular FRBS versions. It tasks fuzzy inputs and results in a fuzzy output. The final step of a Mandami FRBS is defuzzification, a step where the model output is defuzzified to return a numerical output. The rules in a Mandami system has a form like this:
$IF\ X1\ is\ A1,\ and\ ...\ and\ Xn\ is\ An\ THEN\ Y\ is\ B_y$ [21]

2.2.3 Takagi-Sugeno-Kang FRBS

Takagi-Sugeno-Kang FRBS is another popular fuzzy system. It was introduced by Takagi, Sugeno, and Kang[26, 27]. The difference between a Mandami fuzzy system and a TSK fuzzy system is the rules. In a TSK-FRBS, the antecedent of the rules consists of fuzzy variables, and the consequent is a function of the input variables [21]. The output of a TSK-FRBS model is a numerical value because the rules return numerical values rather than fuzzy sets. As a result, TSK-FRBS models do not need defuzzification. A TSK-FRBS rule looks like this:
$IF\ X1\ is\ A1,\ and\ ...\ and\ Xn\ is\ An\ THEN\ Y = p0 + p1 * X1 + ... + pn * Xn$
[21]

TSK-FRBS has advantages and disadvantages. Both advantages and disadvantages result from how the rules are structured. An advantage is that it only needs a small number of rules to describe complex non-linear problems, a lot fewer rules than the Mandami FRBS. The disadvantage is that the TSK-FRBS rules are less understandable than the Mandami rules. The consequent of the TSK-FRBS rules can be difficult to understand, and the model output also depends on crisp inputs. [28, 21]

2.3 Privacy Preserving Machine Learning

Privacy is an important factor when creating trustworthy AI systems. ML has a big potential to increase efficiency and quality in many different areas but requires good training data. A ML model needs a lot of data to train on, and the data needs to be of a high quality. Data quality is an essential factor when creating good ML models. Getting hands on a lot of high quality data can be challenging. A solution to this can be to share data between different organizations. Collaboration can potentially create better models, but it is important to consider data privacy. Data can be confidential and private, and one way to overcome this problem is by utilizing privacy-preserving ML (PPML) techniques.

Privacy-preserving ML techniques allow for training and using ML models while protecting data privacy. Different PPML techniques work in different ways and can be used alone or multiple together. Some of the most popular PPML techniques are Differential Privacy, Secure aggregation, Homomorphic Encryption, Data Minimization, and Federated Learning.

Differential Privacy is a technique that injects noise into the data during training. The noise is carefully calibrated to make it difficult to understand the training data while allowing the model to learn general patterns. Secure Aggregation is a technique for securely aggregating models that allows multiple participants to train a model collaboratively. Each client trains a model on their data before a secure party handles the aggregation. The clients' data are never shared

with any of the other collaborators. Homomorphic Encryption is a type of encryption that allows computations to be performed on encrypted data without decrypting it first. This lets ML models train on encrypted data, and the data remains secure as it is encrypted. Federated Learning (FL) is the privacy-preserving technique this thesis focuses on. It allows multiple parties to train a model collaboratively using a central server. Only the model weights are shared from the clients, meaning the data stays safe where it is. More details on FL in the next section.

2.3.1 Federated Learning (FL)

Federated Learning (FL) is a privacy-preserving machine learning technique. It works by having multiple computers or devices collaboratively train a deep learning model. A central server orchestrates the training, and each computer or device has its own dataset. The model parameters are the only information shared between the central server and the devices. This approach ensures that the training data does not leave the devices. The result is a model collaboratively trained on all the devices' data. [29, 11, 30]

Federated Learning was first introduced in 2016 by McMahan et al. [30]. They investigated training DL models on decentralized data from mobile devices and introduced the aggregation algorithm called FederatedAveraging. The paper concludes that federated learning using the federated averaging algorithm can be a practical way to train good models. With the knowledge gained from the paper, they tested federated learning in Google's Android keyboard called Gboard [31]. The testing showed good results, and Google successfully used Federated Learning to improve the Google keyboard [32]. On next-word prediction, Google trained a model from scratch using Federated Learning, outperforming an identical model trained on a server. [30, 31, 32]

Even though McMahan was the first person to put a name to the term federated learning, the idea behind it is not new. In the early 1980s, cryptographic methods were developed to analyze data distributed between multiple parties without exposing the data. This is because it has been a long-standing goal of researchers within many different fields to gain information from data with many owners without exposing the data[29].

Classic centralized machine learning works by having a dataset, a model, and one server. The server trains the model on the dataset. The model is trained on all the data stored in one central location. The result is a ML model that has been trained on the dataset. In federated learning, a set of clients collaborate to train a global model. A central server gives orders to the clients and receives model parameters back. Each client has their own data set and trains a local model on their data set before sending the model parameters back to the central server. The server aggregates the model parameters from the different clients into one global model. The aggregated global model is then sent back to the clients, who train that model on their data again. The updated local models are then returned to the central server, aggregating them into one model again. This process is repeated multiple times. The final result is a global model trained in data from multiple clients while their data remains secure and private and does not leave the clients. Figure 2.3 shows an illustrative example of how a set of hospitals can

collaboratively train a global model on private data without exposing the data. [29, 11, 30]

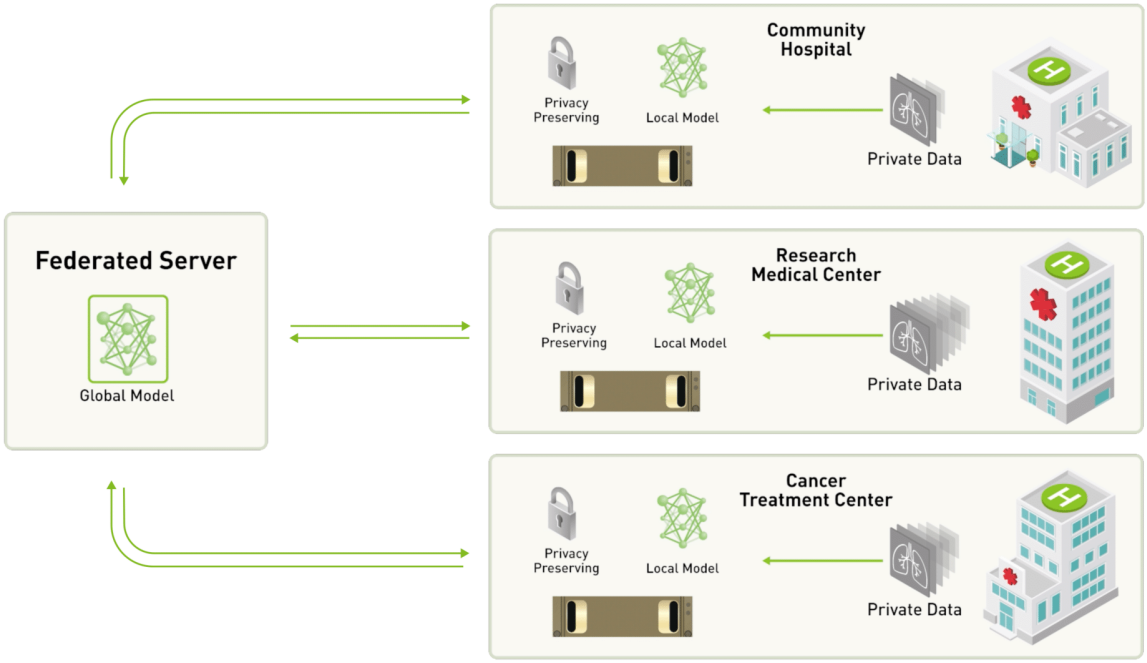

Figure 2.3: Federated Learning figure by Nvidia [3]. It shows the structure of a cross-silo FL system where different medical facilities collaborate to train a global model without sharing their data.

Here is the process of a federated learning system:

1. **Initialization**. A central server has a global model with random weights. Each client downloads the model from the central server.
2. **Local training**. Each client trains the downloaded model and trains it on their local data.
3. **Aggregation**. Each client sends the model weights from their locally trained model to the central server. The central server downloads all the weights and combines them into a single model using an aggregation technique.
4. **Model update**. The central server updates its global model to be the aggregated weights from all the local models. The central server then sends the updated global model to the clients, and the process is repeated until convergence or other stopping criteria.

Federated learning is a privacy-preserving ML technique that provides many advantages. There is enhanced data privacy, and the data owners have more control over their data. Collaboration between multiple clients or organizations provides access to a more extensive and diverse dataset. A more general dataset has the potential to improve model generalization. As McMahan et al. [30] demonstrated in their deployment of FL for Gboard, it is a scalable system. Interest in FL is greater today, and more companies use FL in their products, such as Apple [33]. [30]

FL was first introduced to run on a large number of mobile devices. With the increase in popularity and research, FL has been applied to many different applications, introducing two terms: cross-silo FL and cross-device FL. Cross-silo FL refers to a system with a few clients or

devices, and cross-device FL refers to a system with many clients. The paper by Kairouz et al. [29] introduced the terms cross-device and cross-silo to label the different types of FL systems. A system where a few hospitals collaborate on a model can be viewed as cross-silo, and a system that uses many clients, such as the Gboard, is considered as cross-device. An FL system uses multiple clients to train a global model, with no maximum number of clients. [29]

Other important terms in FL are i.i.d and non-i.i.d data distributions. i.i.d stands for independent identically distributed, and refers to the clients data distribution. A FL system where the clients have the same type of data and features are i.i.d, and systems where the clients have different features, data size, and other differences are non-i.i.d systems. non-i.i.d data is the most challenging to work with, but

An essential part of the FL process is model aggregation. This part combines local models into one global model. There are several different aggregation methods designed to face various challenges with FL. Some aggregation techniques improve the model accuracy, some tackle data security and privacy, and some reduce the data sent between clients and central servers [34]. The aggregation technique introduced by McMahan et al. [30] in 2016 is called federated averaging (FedAvg). It is one of the most common and popular FL aggregation techniques. It works by getting the parameters (model weights) from each client, adding a weight to them, and then getting the average parameters. The weight of the parameters is proportional to the client data size, and the global aggregated model consists of the weighted average parameters[34]. Other aggregation techniques such as FedAvg, FedProx, FedNova, Scaffold, Moon, Zeno, and Per-FedAvg are discussed in the survey by Qi et al. [34]. There are many different aggregation techniques available with the possibility to create custom aggregation techniques as well. According to the paper by Nilsson et al. [11], a model using fedavg achieves a higher accuracy compared to other FL aggregation algorithms. When comparing centralized learning with FL they achieved a similar accuracy when using i.i.d data. When using non-i.i.d data the centralized approach outperformed FedAvg. [11]

FL can be implemented using different frameworks that facilitate the communication, clients, central server and aggregation. Due to the increasing popularity of FL, several frameworks have emerged over the last years. The task of a FL framework is to handle the communication between the clients and the aggregation of the local models into a global model. The frameworks used in this thesis are OpenFL [35] and Flower [36].

As with any system, FL has some vulnerabilities. The goal of a FL system is to collaboratively train a model while keeping the training data private. The data has to be kept private from the different clients, and there can also be external actors that want to attack or steal information. Nuria Rodríguez-Barroso et al. [37] created a survey in 2023 covering the threats to FL.

2.4 Deep Learning

Deep learning is a sub-field of machine learning. Deep learning models are inspired by the neurons and structure of the human brain, and can be used to solve complex problems that more traditional ML models struggle with. Deep learning models consists of interconnected

nodes that create artificial neural networks. The artificial neural networks mimic the interconnected neurons in a brain. The nodes in an artificial neural network are placed in different layers of varying sizes. The input layer and the output layers are visible, but the layers in between are called hidden-layers. Each layer receives some data, perform a mathematical operation on it before sending it to the next layer. The more layers the data passes trough, the more features are extracted from the data. A DL model is trained on a set of data. As more data passes trough the network, it learns to recognize patterns and relationships in the data. The final layer of a model use the patterns learned from the data to make predictions. [38, 39, 40]

Training a DL model can be divided into three different categories. Supervised, semi-supervised and unsupervised -learning. Supervised learning means that the model trains using labelled data. Unsupervised learning means that the model trains on unlabelled data, and learns to find patterns and relationships in the data. Semi-supervised learning is a mix of the two. A model is trained on both labelled and unlabelled data. [41]

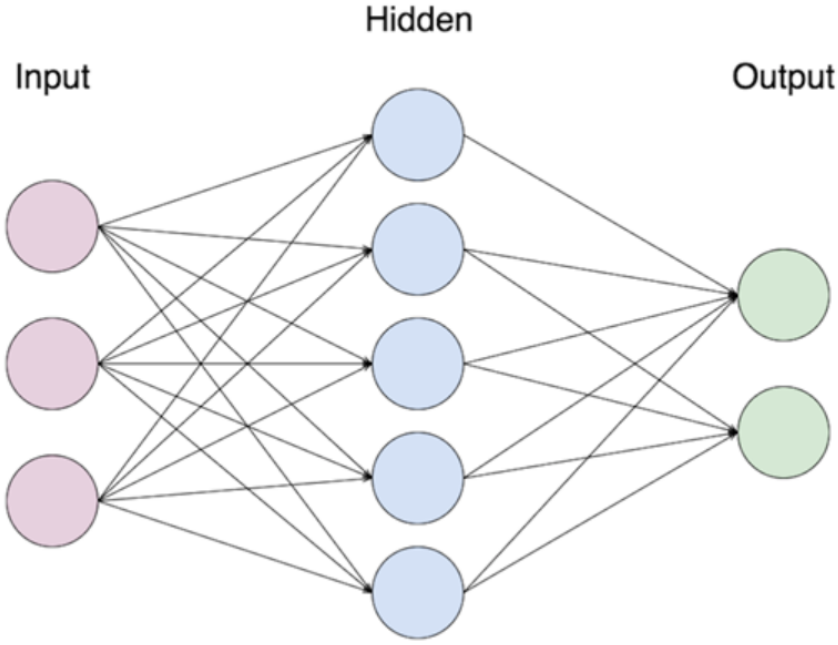

Figure 2.4: Example of a simple neural network [4]

Figure 2.4 shows a simple neural network. It consists of an input layer, a hidden layer and an output layer. The neurons are represented by the circles, and each neuron is connected to the neurons in the neighbouring layers. The connection between the neurons are weighted. This means that some of the neurons have more influence on the connected nodes than others. The hidden layers and output layers use activation functions to learn complex relationships in the data. The neurons in the hidden layer and output layers apply an activation function on the weighted input from the previous nodes.

Activation functions help a neural network to learn non-linear relationships and complex patterns. It does this by deciding whether a neuron should be activated or not. This is done by taking the weighted input from a previous node, convert it according to a function, and then

pass it to the next node. Some examples of activation functions are Relu and Sigmoid. Relu stands for rectified linear unit, and is a popular activation function in DL. Relu takes a single real number as an input. If the input is a positive number, it will output the input number. If the input number is negative or zero, it will return zero. The sigmoid activation function in a different way than relu. It takes a number as an input, and then gives an output in a range from 0 to 1. A high input number will return a value closer to 1, and a low input will return a number closer to 0. This makes the Signmoid function suitable to output a likelihood. [38, 40]

The learning process of a DL model consists of tree steps. A forward pass, error calculation and back-propagation. These steps are repeated multiple times to train a model. A forward pass is the process of data going trough a neural network. The data travels between the nodes, and activation functions calculate the weighted data from the previous node. After going trough all the layers, the output layer generates an output. The second step is the error calculation. This is where the output from the forward pass is checked. An error function is used to calculate the difference between the output value and the true value. This results in an error that says how wrong the model was at a certain data point. The third step is back-propagation. This is the step where the model is improved by adjusting the weights of the node connections. It starts from the output layer, and goes over the weights back towards the input layer. At each layer, the weight is adjusted. If a weight contributed significantly to the error, it is modified to reduce the error. The optimization process is done by an optimization algorithm. The weights of the model are adjusted a little at a time over multiple data points. [40]

Chapter 3

State of the art

Explainable AI and federated learning are two popular technologies that are receiving much attention. Federated learning, in combination with explainable AI, is not a big field of research. While there is a lot of research on FL and XAI on their own, very few papers and research cover the technologies used together. The aim of this thesis is to investigate the impact of FL on an inherently explainable ML model. This section will examine current knowledge and research regarding explainable federated machine learning.

The literature in the field of FL and XAI can be divided into two main areas. Papers using FL, DL models, and post-hoc explainability, and Papers using FL and inherently explainable models. Papers using FL, DL models, and post-hoc explainability are the most popular. This is because the majority of FL research is focused on DL. Fewer papers cover the use of FL and inherently explainable ML models. Federated learning was designed and developed for DL models. As a result, most of the research and available literature in the area uses and focuses on DL rather than simpler ML models.

3.1 XAI and FL

López-Blanco et al. [42] wrote a review on federated learning of explainable AI in 2023. The review looks at FED-XAI, a combination of FL and XAI. When reviewing articles on explainable AI and FL, they found that few papers have been published on this new trend. The main contributions to the FED-XAI field are from 2022 and 2023, and they expect the number of papers to increase with time. In their review they found seven notable articles with the term FED-XAI. Some of them discuss and review the status of FED-XAI, two mention FED-XAI without contributing anything, and two have implemented an explainable model or an explainability framework on a FL architecture. The review concludes that the main problem of FED-XAI is to achieve results comparable to those achieved using a centralized machine learning approach. [42]

One of the most prominent papers mentioned in the review by Raúl López-Blanco et al. [42] is the paper by Alessio Bechini et al. [43]. The paper describes the most recent research done by the Artificial Intelligence R&D (AI-RD) group of the Department of Information Engineering

of the University of Pisa. They mention a project called HEXA-X, where they hope to deploy FED-XAI on 6G networks. One of the requirements of the HEXA-X project is to fulfill the requirements of trustworthy AI. They aim to develop a framework for FED-XAI where user agents collaboratively train an XAI model. This has resulted in a large part of the available research on FED-XAI to originate at the University of Pisa.

The paper by Ducange et al. [44] from 2022 presents a proof of concept for FED-XAI. The proof of concept is between a set of Tele-operated cars and a central server. To drive cars remotely the video stream needs to be high quality. They trained a model to predict when the video quality would get bad using Fed-XAI.

The researcher José Luis Corcuera Bárcena from the University of Pisa has written and co-authored multiple papers discussing and implementing a TSK-FRBS model in a federated fashion. In [45] discusses frameworks that use post-hoc explainability on DL models. It then mentions that for FL frameworks that use interpretable by design models, the TSK-FRBS model looks most promising for XAI to be trained using FL. In [46] Bárcena et al. implements a TSK-FRBS model, and trains it using FL. They then compare the results with a state of the art non-FL TSK-FRBS delivered by PyFUME. The results showed that their implementation achieved a better performance compared to PyFUME's TSK-FRBS.

In 2023, Daole et al. [12] introduced a framework called OpenFL-XAI. It is an extension of the OpenFL framework [35], enabling it to train rule-based models in a federated approach. The model they use is TSK-FRBS, and the implementation of the model is the one from the paper mentioned earlier by Bárcena et al. [46].

Other papers such as [47] by Zhu et al. and [48] by Wilbik et al. implement and train a TSK-FRBS model in a federated fashion. Experiments are conducted on different datasets, and they conclude that federated FRBS can achieve high-quality results on real-world problems and FL-trained models can outperform central models.

Another term for FL and explainable ML is fXAI. Kusiak proposed it in 2023 [49], and it covers the same area as FED-XAI. He also proposes a rule-based algorithm called XRule to create a rule-based algorithm. Training the XRule algorithm in a federated fashion is the paper's contribution to the field of federated explainable AI.

The paper by Tomisin Awosika et al. [50] presents a banking fraud detection system using Federated Learning, Deep Learning and Explainability. They train a model using Federated Learning on banking data and then utilize SHAP to make the resulting model explainable. They used this dataset [51], which contains [50, 51]

Federated learning is a relatively new technology. As a result, the research on FL and XAI is relatively new. There are a few different terms used, such as FL-XAI, and fXAI. The papers investigating post-hoc explainability such as [52, 53, 50], use methods such as SHAP and LIME to explain the data after training. The OpenFL-XAI framework has made it possible to compare federated rule-based models with federated DL models.

Chapter 4

Methodology

The methodology section of the thesis outlines the methodology used to explore the research question. This thesis aims to investigate how FL impacts the performance of a TSK-FRBS or DL model. If FL has a negative impact on performance, or if FL impacts the TSK-FRBS rules. To answer these questions, a few TSK-FRBS models and deep learning models are implemented on 10 different datasets. The models are trained in a federated and central fashion, allowing a comparison between the results. This section will describe the details of the implementation and why the different decisions were made.

4.1 Research design

This thesis uses an empirical research approach, which involves gathering real-world data to test hypotheses and research questions. The research is designed to evaluate FL's impacts on the results of an inherently explainable model. This is done by comparing results on TSK-FRBS models with DL on ten datasets.

The research approach is designed in six steps.

1. **Read papers on the topic**. Investigating and exploring what others have done earlier in this field and how the different technologies work makes it easier to know how to approach the research question.

2. **Select model**. A better understanding of the field and different technologies helps select a model for the inherently explainable ML. The research will be conducted on these models.

3. **Find datasets to use**. After choosing a model, different datasets are explored and selected. It's important to include multiple datasets from different application areas.

4. **Train model on the datasets**. The central and federated ML models are then trained on the selected datasets. Results are saved for future evaluation.

5. **Implement DL models on the same datasets**. DL models are implemented on the same datasets as ML models and trained in a federated and central fashion. They are considered black box models and are more complex than simpler ML models. The results are stored

and used for evaluation and comparison with the ML models.

6. **Analyze results**. The final step of the process is to evaluate all the collected results. What are the answers to the research questions, and do FL have a big impact on performance.

4.2 Implementation

The implementation covers stage 2 to 5 in the research approach. Four model types have been implemented on ten datasets: A federated TSK-FRBS model, a central TSK-FRBS, a federated artificial neural network, and a central artificial neural network. The central and federated TSK-FRBS models have been implemented using the OpenFL-XAI framework introduced by Daole et al [12] in 2023. It is an extension of the FL framework called OpenFL [35]. The extension adds the functionality of training fuzzy rule-based systems in a federated manner. The OpenFL-XAI framework also includes an implementation of the Takagi-Sugeno-Kang FRBS algorithm. PyTorch [54] has been used to develop the DL models. It is an open source framework for developing and training deep learning models. For the federated DL models the Flower framework [36]has been chosen. OpenFL and Flower are both FL frameworks. Their task is to facilitate the communication and training of federated models while preserving data privacy. They handle the communication and collaboration between clients and a central server. Both frameworks perform the same tasks but function in different ways.

This project has chosen the OpenFL-XAI framework and the TSK-FRBS model as the inherently explainable ML model. OpenFL-XAI is a convenient tool that allows one to compare different models, such as the performance of an inherently explainable model with that of a deep learning model. The code for the OpenFL-XAI framework is publicly available, and it supports both FRBS and deep learning models. Most FL frameworks focus on deep learning models, and the OpenFL-XAI framework is the only framework to the researcher's knowledge that supports fuzzy rule-based systems. The TSK-FRBS model have been chosen because others have used previously [45, 46, 47]. A comparison can be made to check if they achieved very different results. The PyTorch framework has been chosen to develop the DL models. The reason for this is the researcher has previous experience working with it. The Flower FL framework has been used to train the federated DL models. The OpenFL-XAI was originally intended to be used for the DL models, but the researcher had problems implementing a DL model using it. The Flower framework was chosen instead as the researcher found it easier to use and managed to implement the DL models.

This project was set up by forking the GitHub repository of OpenFL-XAI. The OpenFL-XAI framework includes an implementation of a TSK-FRBS model which is the one used in this thesis. The GitHub repository contains all the code used in this project and can be found here: `https://github.com/Halvorte/FL-Explainability`.

4.2.1 Central and federated TSK-FRBS

TSK-FRBS is the inherently explainable ML model. It has been implemented on ten datasets in a central and federated fashion. The model implementation of the model comes from the

OpenFL-XAI framework. The TSK-FRBS model implemented is a one-shot training model. This means the model trains in one run and not over multiple iterations. A TSK-FRBS consist of a set of rules using fuzzy logic. The model in this thesis is a first-order TSK-FRBS model as the fuzzy partitioning is type-1.

The OpenFL-XAI framework enables federated training of explainable ML models. The openFL-XAI has two components: aggregator and collaborator nodes. The aggregator node is the central server that orchestrates the training and aggregates the final model. The collaborator is the clients that do the training and send their local models to the aggregator. The collaborators retrieve a list of tasks by sending a remote procedure call to the aggregator using OpenFL's TCP client. After training and performing its task, the collaborators send their model updates to the aggregator using TCP. OpenFL uses docker to run the different clients and aggregator. They communicate with each other over TCP. The FL process is configured trough a YAML file that contains the aggregator IP, FL process parameters, and the ML parameters. [12]

The FL process using OpenFL-XAI works as follows: First, the aggregator node is started. It is initialized by parsing the FL plan. After it has been initialized, it waits for collaborators to contact it and perform a remote procedure call. Then, the collaborators are started. They are also initialized by parsing the FL plan. The collaborators connect to the aggregator and receive a list of tasks. The different collaborators are the only ones with access to their local data. The collaborators perform their tasks until they receive an end-of-federation (EOF) command from the aggregator. After receiving the EOF command, the collaborators shut down. The aggregator does not consider a client finished with its task until it has checked that all its tasks are finished. When all the collaborators' tasks are completed, the aggregator aggregates the local contributions to a global model according to the specified aggregation technique. [12]

The federated TSK-FRBS models are trained using five clients. The central TSK-FRBS models are trained similarly to the federated ones but with one client instead of five. As a result, a single client with access to all the data trains a model without collaboration.

The fuzzy sets utilized in the TSK-FRBS model are "LOW", "MEDIUM", and "HIGH". They range between 0 and 1 on both fuzzy values and membership value. Figure 4.1 shows the fuzzy sets. The input data is normalized to a range of 0 to 1 in order to work with the fuzzy sets.

The TSK-FRBS model consists of a few different stages.

- **Data pre-processing**. The first step for all of the models is the data pre-processing. The fuzzification step requires data to be normalized in a range between 0 and 1. The data pre-processing is described more in section 4.3.
- **Rule base generation**. The rule base generation is the first step of a collaborator. This is where the initial rules are created. These can be created by a domain expert or trough algorithms. In this thesis, the initial rule generation is done through fuzzy c-means clustering. Each client is set to create 30 rules in the plan yaml file. The fuzzy c-means

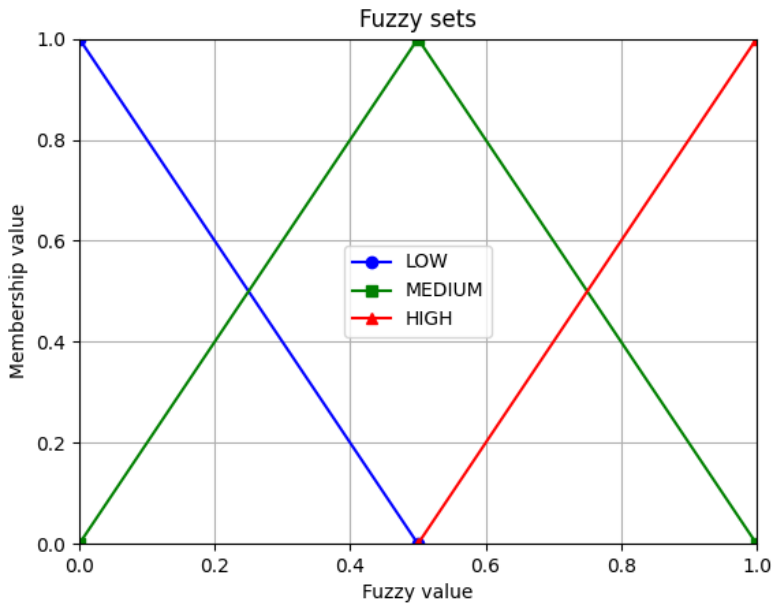

Figure 4.1: Fuzzy sets used

algorithm works by grouping the input data into 30 clusters. It is similar to the k-means algorithm, but each data point belongs to a cluster with a varying degree. Similar data is grouped together, and the fuzzy c-means algorithm returns 30 centroids. These centroids form the base of the rules. The fuzzy c-means algorithm used is imported PyFume, a python package for fuzzy model estimation.

- **Rule antecedent generation**. The next step is to create the IF part of the rules. This is done by iterating trough the 30 centroids. Each centroid consist of values for the different features of the dataset. Each feature in the centroids are checked against the fuzzy sets to see which set they belong to. For example a centroid with four features that looks like this: $0.0, 0.4, 0.8, 0, 6$, will look like this after checking against the fuzzy sets: $LOW, MEDIUM, HIGH, MEDIUM$. This results in 30 sets of fuzzy elements representing the 30 rules. The next step is to check for and remove duplicate rules. This is quickly done by comparing the 30 sets with each other and removing duplicates.

- **Rule consequent generation**. The consequent are created after the antecedents. The consequent consist of the input features with a weight. The consequent are calculated based on the firing strength of each rule on the training data. The firing strength says how much the features in each rule are activated by the input data. The firing strengths are then used to estimate the optimal weights for the different rules using a consequent estimator imported from PyFume. The consequent estimator looks at the input data, firing strengths and the target values to estimates the consequent. It uses least squares with gradient descent in order to minimize the squared distance between estimated and target values. The result is a consequent for each antecedent, consisting of the consequent weights for each feature.

- **Rule weight calculation**. With both the antecedents and consequents the next step is

to calculate the rule weights. The rule weights says something about how important or how much influence the different rules have in the final model. This is done by making a prediction with each rule on each data point. The predicted values and the true values are then compared. The error in the prediction is calculated and normalized in a range of 0 to 1. The prediction is given a fuzzy quality meaning the prediction can be LOW, MEDIUM or HIGH quality. Each rule is then given a confidence and support score. the confidence score is calculated by the mean quality across all data points weighted by the firing strengths. The support score is calculated by the mean prediction quality across all the data points. The final rule weight for each rule is a combination of quality and confidence score, and indicates the importance of that rule in the model.

- **Model**. The final model for each collaborator consists of antecedents, consequents and rule weights. The rules are saved on the client, and sent to the aggregator. In federated DL, the model weights are usually transferred between collaborators and aggregator. In OpenFL-XAI the complete rules are sent to the aggregator.

- **Model aggregation (federated)**. The aggregation part combines multiple models into one. Both the central and federated TSK-FRBS models are created using the same framework and structure. The difference between them is that the central TSK-FRBSwhilels have 1 client, while the federated TSK-FRBS has 5 clients. The central models skip the aggregation part as the aggregator only receives one set of rules. The federated version consists of 5 models that need to be aggregated into a single model. The aggregation method is FedAvg. The version used in the TSK-FRBS models is provided by the OpenFL-XAI framework. It enables the aggregation of fuzzy rules. It works by combining the rules from all the models into one big set of rues. Duplicate rules are then removed, and the mean consequent weights of the duplicate rules are computed. The aggregated model consists of a set of unique rules where the weight of the rules that appeared more than once have been averaged.

- **Model validation**. The models are evaluated by comparing the predictions with the real values. The TSK-FRBS model makes predictions by first calculating the firing strengths of all the rules. The firing strengths and the consequent rules are calculated to determine the rules' contribution to the final output. The contributions of the rules are then combined using the weighted averages into a single output. The result is a single prediction by the model based on all the activated rules.

  After the model have made a prediction on the validation data, it is compared with the real values. The evaluation methods are presented in section 4.5.

The implemented TSK-FRBS model is a regression model. It was modified to be able to do binary classification. A simple threshold function was added on to the output of the binary classification models.

The TSK-FRBS model is a regression model. This means that it predicts a real value. Some of the datasets used in this project are binary classification datasets. To modify the TSK-FRBS to work as a binary classification model, a threshold is put on the output of the model. The model works

the same on both regression and binary classification datasets, but for the binary classification, the results are set to either 0 or 1 based on a threshold.

4.2.2 Central and federated deep learning

DL models have been implemented to strengthen and gain a more comprehensive understanding of the federated ML results. A DL model has been developed for each dataset and trained in a central and federated fashion using PyTorch. The model architecture differs for each dataset, consisting of linear layers and activation functions. The model architectures and parameters can be found in the Appendix. The deep learning models were implemented on one dataset at a time. The central DL model was implemented before the federated DL model. This was done to find a model architecture that worked with the data before training it using FL. The model architectures were developed through trial and error and manual hyperparameter tuning.

After the central DL models are created they are trained using FL. The Flower FL framework was used to do this. It simulates the clients and central server and handles the communication between them. The Flower federated framework was selected because the researcher has some previous experience using it. After struggling to implement custom DL models using the OpenFL framework, the Flower framework was chosen instead.

Central DL

The central DL process loads the data using a custom dataloader script. The script cleans and pre-processes the data. Ten-fold cross-validation is then used to get the generalized performance of the models. The data is split into ten folds, and the model is trained on nine folds and tested on the last fold. The pre-processed data is then converted into Pytorch tensors, which represent the data as matrices for deep learning calculations. The data is then turned into Pytorch datasets before being turned into Pytorch dataloaders. This is done to help with the training process as it can shuffle and batch data and help with memory efficiency.

The training process iterates through multiple epochs. For each epoch a batch of data is collected from the dataloader. The model makes a prediction on the collected data, and the loss function calculates the prediction error. A loss function calculates the difference between predicted and true values. Backpropagation then traverses the model backward and adjusts the weights based on the optimizer. The optimizer adjusts the model weights in order to minimize the model loss. This is how the models learn the patterns and trends in the data. The validation then checks how the model performs on unseen data. Its important to evaluate the model in order to avoid overfitting. The model performance is stored after the final epoch, and then the whole process is repeated with a new set of folds. The final results is an average of ten model on the same dataset with different data partitions.

Federated DL

The federated DL models are trained using the Flower framework. The model architectures are the same as the central DL models and can be found in the appendix. The first steps are the same as for the central model, with the addition of splitting the data into partitions for the clients..

4.3 Datasets

The third step in the research approach for this thesis is to gather and select datasets on which to test the models. A few criteria have been made to ensure the datasets are a good fit for this project. The datasets need to be either a regression or binary classification dataset to be compatible with the TSK-FRBS models. The datasets must also represent different application areas where data privacy and security can be important. Data privacy and security can be for situations where the data itself is private, or some company wants to keep data private as it's a key asset for them. Examples of areas are healthcare, finance, energy, aviation, and more.

Ten datasets have been chosen for this project. They all follow the criteria set and vary in size and application areas. Ten datasets were selected to increase generalizability and have more results to compare with. The selected datasets are openly available on the internet and were discovered by searching online using Google and Kaggle. The implementation of the TSK-FRBS model for this project is a regression model, which means that it takes columns of data to predict an exact value for the output. An example is to predict the exact age of life expectancy based on a set of features. not all the datasets are regression sets, some are for binary classification. The TSK-FRBS model has been modified to enable binary classification by using a simple threshold.

4.3.1 Data splitting/division

When training a model using FL, each client has its own unique dataset. Each dataset is divided into ten as a result of the ten-fold cross-validation, and the nine parts used for training is divided to the clients. The clients' data is then divided into an 80-20 train test split. The validation the global model is evaluated on the part of data left out in the ten-fold cross-validation. In the OpenFL-XAI framework the data for each client is stored in zip files. The clients access these zip files when training a model. Figure 4.2 represent how all the datasets are split up to train five clients using FL. The entire dataset is first divided into a train test set by the ten-fold cross-validation. The training set divided into five new datasets with a random selection for the clienst. The five new datasets are then divided again into train and test set with an 80-20 train test split. The final model is tested using the one part of the ten-fold cross-validation saved for testing.

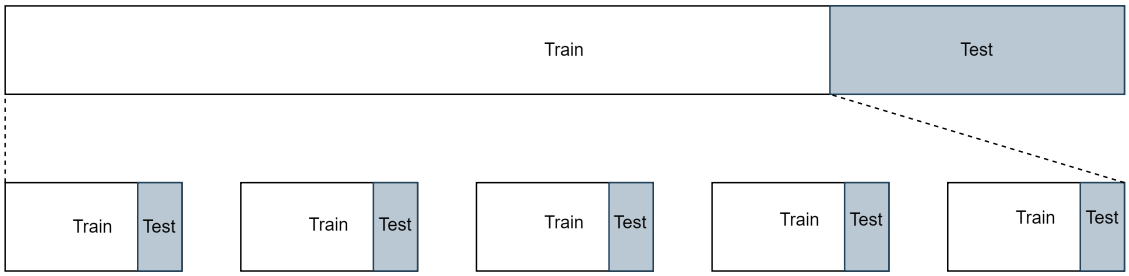

Figure 4.2: Dataset split

4.3.2 Data pre-processing

When working with several datasets from different applications, the data will vary. Therefore, all the datasets have been pre-processed. The pre-processing is different for each dataset, but they go through the same steps. The result after the pre-processing is normalized data, where one column is the prediction value and the rest are the features. The pre-processing steps are the following:

1. **Read data**. The first step of the pre-processing is to download the dataset, add the file to the project, and read it into a Python file. This step is important to be able to work on the data. Some datasets are stored as csv-files, some as txt-files, and some as artiff-files. The variety in file types adds complexity and difference in how the processing is handled.

2. **Clean headers**. The datasets have varying formats. Some have whitespaces in the headers, and some don't. This step takes care of that and makes sure that there are no whitespaces in any of the headers. Whitespaces in headers can make it harder to access specific columns when working with the data, so this is just a simple step to make it more convenient to work with the data.

3. **Remove empty rows**. The different models can not train on data with missing values. There are several different ways to handle missing values in datasets. The missing values technique used in this thesis is to remove rows with empty values. This was not an issue as most datasets had no missing values. Some did have a few missing values here and there, but removing just a few rows does not have a significant impact if the dataset is big enough.

4. **Data conversion**. The TSK-FRBS models are not able to process strings, and needs the inputs to be numbers. This is for the fuzzy logic to work. Some of the columns in the datasets contain categorical data or strings. The values in these columns are converted to numbers using Sklearns label encoder. It converts strings such as countries to numbers, years into ascending numbers, and other categorical string values into numbers. This is very helpful when datasets contain values such as "developing" or "developed" about an entry, and the label encoder converts the strings into 0 and 1.

5. **Normalization**. For the TSK-FRBS model implementation used in this thesis to work properly, the data needs to be normalized between the values of 0 and 1. To help with the normalization the Sklearn minmax scaler was used. It normalizes the values to a range

between 0 and 1. All the columns are normalized except the target column and categorical columns such as country or year. After the normalization the data is ready to be split up according to the ten-fold cross-validation.

4.3.3 Life expectancy

The Life expectancy dataset is a regression dataset. It is made of data from The Global Health Observatory under the World Health Organization (WHO). The dataset consists of 22 columns and 2938 rows, and it is made up of health status factors, financial, and economic data. This includes data such as country, year, adult mortality, alcohol consumption, and more. The data in the dataset is from between 2000 and 2015 for 193 countries. The goal of the dataset is to predict the life expectancy of a person living in a country based on the features provided. [55] The life expectancy dataset is available on Kaggle, and it can be downloaded as a CSV-file. [55]

4.3.4 Pima Indians Diabetes dataset

The Pima Indians Diabetes dataset is a binary classification datast. It is made by the National Institute of Diabetes and Digestive and Kidney Diseases and is used to predict if a patient has diabetes or not. The dataset consists of 9 columns and 768 rows made up of different diagnostic measurements such as glucose level, blood pressure, insulin, and more. The data for the dataset is from 768 Pima Indian women. It is a relatively small dataset and is unbalanced. [56]

4.3.5 Credit Card Fraud Detection

The credit card fraud detection dataset is a binary classification dataset. It is made by credit card transaction data and is used to classify fraudulent transactions. It consists of 31 columns and 285,000 rows of data. The dataset consists of transactions made by European cardholders in September 2013. The transactions are from two days, and 492 out of 284 807 transactions are fraudulent. This means that the dataset is very unbalanced. Because of confidentiality issues, the features of the dataset are labelled with names such as V1, V2, and so on. The features are also the resulting principal components of a PCA. [57]

4.3.6 Smart Grid Stability

The smart grid stability dataset is a binary classification dataset. It contains different measurements from the electrical grid and can be used to predict whether the electrical grid is stable or not. It consists of 14 columns and 60,000 rows and is an augmented version of an original dataset. The original dataset is called the "Electrical Grid Stability Simulated Dataset," created by Vadim Arzamasov and donated to the University of California (UCI) Machine Learning Repository. [58]

4.3.7 NSL-KDD, Network security binary classification

The NSL-KDD dataset is a binary classification dataset used for intrusion detection. It is a modified version of the KDD 99 dataset that removes redundant records. It is sized so that it is easy to use for training without splitting it into subsections. The dataset is stored as both an arff file and a txt file. This thesis uses the txt file to read the data. It consists of 42 features and 125,971 rows of data. [59]

4.3.8 Aileron

The Aileron dataset is a regression dataset made to predict the position of the wing aileorns on an F-16 fighter jet. The ailerons control an aeroplane's rolling or banking. Each wing has an aileron toward the tip that can move up or down to control it. The dataset was downloaded from the KEEL (Knowledge Extraction based on Evolutionary Learning) website. KEEL is an open-source tool for gaining knowledge from data through different approaches. The KEEL site for this data set states that the ailerons dataset is not an original KEEL dataset, but has been obtained from the LLIACC repository. The downloaded dataset consists 40 features, and 13,750 entries stored in a .dat file. [60]

4.3.9 Red Wine Quality

The red wine quality dataset is a regression dataset where the goal is to predict a wine's quality score in a range between 0 and 10. The dataset was first introduced in a paper by Cortez et al. [61] from 2009, where they modeled wine preferences using different data mining approaches. The dataset consists of 11 features, one target column, and 1599 rows. The features consist of different wine measurements such as acidity, citric acid, pH, alcohol, and more. The dataset is not balanced, and there are more normal winesthano excellent and poor wines. The dataset is available onKagglee [62, **?**] where it has been published by the UCI machine learning repository. The dataset is stored in a .csv file. [62, 61]

4.3.10 Combined Cycle Power Plant

The Combined Cycle Power Plant (CCPP) dataset is a regression dataset whose goal is to predict the hourly energy output in MW. A CCPP consists of gas turbines, steam turbines, and a heat recovery system. The combination of gas and steam turbines makes it possible to generate more electricity from the same amount of fuel. The dataset introduced in the paper by Tüfekci et al. [63] from 2014. It consists of four features, one target column, and 9568 rows. The features are temperature, ambient pressure, relative humidity,y and exhaust vacuum. The four features are used to predict the electricity output. The dataset can be downloaded on the UCI machine learning repository [64].

4.3.11 Condition Based Maintenance of Naval Propulsion Plants

The Condition Based Maintenance of Naval Propulsion Plants dataset consists of data from simulations of the gas turbine propulsion plant aboard a Frigate. The goal of the dataset is

to predict the degradation of the propulsion plant. This can then be used to decide when the propulsion plant requires service or part replacement. Running a system until failure can often be a lot more expensive than servicing the system before it fails. To be able to do this effectively, it is important to understand the system's status. The dataset is made by Coraddu et al. [65]. It consists of 16 features, two target values, and over 11934 rows of data. The two target values are compressor decay and turbine decay. This means it is possible to predict the decay on both, but for this project, the turbine decay is used for the prediction. The features consists of ship speed, gas turbine rpm, fuel flow, different temperatures, pressures and more. It can be downloaded from the UCI machine learning repository [66], and consists of a .txt file. [65, 66]

4.3.12 Superconductivity Data

The superconductivity dataset is a regression dataset containing data from superconductors. Its goal is to predict the critical temperature of the superconductors based on their features. A superconductor is a material that can conduct current with zero resistance. It was introduced in the paper by Hamidieh et al. [67]. It consists of 81 features and 21,263 rows of data. It has one target variable, which is the critical temperature. The dataset can be downloaded from the UCI machine learning repository [68]. [67, 68]

4.4 Test-bench

All of the algorithms and models have been run using a Dell XPS 13 9380 Laptop. It has an Intel Core i7-8565U Cpu and 16GB of ram.

4.5 Model results and evaluation

Both regression and binary classification datasets are used in this thesis. There are different ways to evaluate the performance of a ML model with different types of datasets. The regression datasets' models are evaluated using R2, MSE, and MAE. The evaluation criteria for the binary classification datasets are accuracy, precision, and recall.

The models are trained multiple times using ten-fold cross-validation. This is done to get a more reliable performance estimation. Each model is trained 10 times with different data divisions. The results are represented as the mean performance with a 90% confidence interval. Numpy's mean function has been used to calculate the mean scores. The margin of error for the 90% confidence interval has been calculated by multiplying the z-score of 1.645 with the standard deviation of the results. The k-fold cross validation used in this project is Sklearn's KFold. For the FL implementations, the split is done on the global dataset before the splitting for the clients is done. This approach for the FL models is because the global model's test set always evaluates the models. So, the test set is slightly different depending on the k-fold for each model's ten runs.

R2. R2 or R-squared is an accuracy measurement used for regression datasets. It tells how well a model fit the data. How likely is it that a new datapoint will be predicted by the model. R2 alone

is not enough to evaluate a regression model, which is why R2 is used together with MSE and MAE. Sometimes the MSE and MAE values are not enough in themselves to get a full picture of the performance of a model. The MSE and MAE values can be different depending on the dataset and the values used. The R2 implementaion in this thesis is the r2_score function from sklearn. R2 can have a score between 0 and 1. A score of 1 means that the predicted values follow the real values 100%, and a score of 0.4 means that the relationship between the prediciton and real values are 40% [69, 70, 71]

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{4.1}$$

MSE. MSE stands for Mean Squared Error and measures the average squared difference between predicted and real values. A smaller MSE means that the predictions are closer to the real values. MSE is used in conjunction with R2 as it can be sensitive to outliers and is scale-dependent. The size of the MSE prediction depends on the scale of the target. The MSE function used in this thesis is from sklearn. [72, 71]

$$MSE(y, \hat{y}) = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} (y_i - \hat{y}_i)^2 \tag{4.2}$$

MAE. MAE stands for Mean Absolute Error and measures the mean absolute difference between predicted and actual values. MAE is less sensitive to outliers than MSE as they contribute a smaller amount to the MAE distance. The MAE calculation is performed by Sklearns MAE function. [71]

$$MAE(y, \hat{y}) = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} |y_i - \hat{y}_i| \tag{4.3}$$

For the binary classification datasets the model results are measured using a different set of metrics. The classification metrics used are accuracy, F1, precision and recall.

Accuracy. Accuracy is the metric that measures the overall accuracy of the model prediction. It measures how many out of the total predictions are correct. Accuracy alone is not a very solid measurement, as it can make a result seem better than it is. If a dataset is very unbalanced, and the model only predicts the same every time, the accuracy can be high even though the model is bad at making predictions. The accuracy is calculated by Sklearns accuracy function.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{4.4}$$

Precision. Precision measures the number of correct positive predictions out of all the positive predictions. This is a good measurement of how well the model predicts the positive values. Implemented using sklearn. [73]

$$Precision = \frac{TP}{TP + FP} \tag{4.5}$$

Recall. Recall measures the number of correct positive predictions out of all the correct positive predictions that could have been made. This is a good measurement to see what proportion

of positive predictions were correctly predicted by the model. Implemented using sklearn. [73]

$$Recall = \frac{TP}{TP + FN} \tag{4.6}$$

It can be difficult to interpret the explainability of a TSK-FRBS model. The explainability depends on multiple factors, such as the number of rules, the number of features, and the number of linguistic terms. In this thesis, the rules are evaluated visually. Some of the rules from each model have been looked at to see that they have the same format.

Chapter 5

Results

The results from this project are a result of experiments where models have been trained on models. The experiments have been conducted in four different stages. The first stage was to train the federated TSK-FRBS models on the ten datasets. Each model consists of five clients, each set to create 30 rules. The training was conducted using ten-fold cross-validation to evaluate the models' performance. The data for each client is i.i.d, and each client has an 80-20 train test split. The model performance is evaluated using the global test set.

The second stage was to train the central TSK-FRBS models. The central approach is the same as the federated, but with one client instead of five. This means all the data is given to one client acting as a single model. The central TSK-FRBS models have also been trained using ten-fold cross-validation. One model has all the data and is set to create 30 rules.

The third stage was to train federated DL models. Each DL model has a unique architecture and was created specifically for each dataset. They all have 5-clients, but the number of epochs, local epochs, learning rates, and batch sizes are different. The data is i.i.d, and they were trained using ten-fold cross-validation.

The fourth stage was to train the central DL models. The model structure for each dataset is the same as the federated models, but trained in a classic central manner. The hyperparameters are different for each model. The models were trained using ten-fold cross-validation.

The results from the different models are represented through tables and some example rules. Four tables contain different performance metrics. Table 5.1 shows the results from the TSK-FRBS models on binary classification datasets. Table 5.2 shows the results from the DL models on the binary classification datasets. Table 5.3 shows the results from the TSK-FRBS models on the regression datasets. Table 5.4 shows the results from the DL models on the regression datasets.

The binary classification datasets and the regression datasets have different results metrics. This is because they are predicting different types of results. The binary classification models predict either a 0 or a 1, and the regression models predict an exact value. The binary classification results are measured using accuracy, precision, and recall. The regression results are harder to evaluate, and the results are represented using R-squared (R2), Mean Squared

Error (MSE), and Mean Absolute Error (MAE). The performance of a regression model is decided by how close to the real its predictions are.

Some rules from the TSK-FRBS models are also shown in the results. TSK-FRBS models are inherently explainable and use a set of rules to make their predictions. An important part of using the TSK-FRBS models is their explainability. It is important to check if the models are as explainable using federated as central training. A few rules from different models are shown in section 5.2 to evaluate them. The rules are IF-THEN rules, and their length depends on the number of features in the dataset.

5.1 Dataset Results

Application	Dataset	TSK-FRBS Binary Classification					
		Central			Federated		
		Accuracy	Precision	Reall	Accuracy	Precision	Recall
Health	Pima Indians Diabetes	76.1% ± 9.5%	73.2% ± 14.4%	48.2% ± 20.5%	74.7% ± 10.8%	68.5% ± 17.2%	51% ± 20.2%
Finance	Credit Card Fraud Detection	99.9% ± 0.02%	85.8% ± 7%	69.7% ± 13.3%	99.9% ± 0.01%	85.7% ± 0.7%	67.4% ± 0.5%
Energy	Smart grid stability	96.2% ± 0.4%	96.1% ± 0.6%	98% ± 0.4%	96.2% ± 0.4%	96.1% ± 0.6%	98.1% ± 0.4%
Cybersecurity	NSL-KDD Network Intrusion Detection	95.6% ± 0.5%	95.8% ± 0.9%	94.7% ± 0.8%	95.7% ± 0.7%	96% ± 1%	94.8% ± 1.1%

Table 5.1: Results from TSK-FRBS on binary classification datasets

Application	Dataset	Deep Learning Binary Classification					
		Central			Federated		
		Accuracy	Precision	Reall	Accuracy	Precision	Recall
Health	Pima Indians Diabetes	75.8% ± 16.2%	78.3% ± 52%	44.5% ± 34%	85.3% ± 8.8%	77.4% ± 18.5%	77.3% ± 19%
Finance	Credit Card Fraud Detection	99.8% ± 0.03%	0.0% ± 0.0%	0.0% ± 0.0%	99.9% ± 0.03%	7.7% ± 0.9%	7.7% ± 0.8%
Energy	Smart grid stability	100% ± 0%	100% ± 0%	100% ± 0%	99.6% ± 0.1%	99.5% ± 0.2%	99.8% ± 0.1%
Cybersecurity	NSL-KDD Network Intrusion Detection	96.9% ± 4.4%	98.1% ± 6.1%	95% ± 8.3%	98.9% ± 0.2%	99% ± 0.6%	98.7% ± 0.8%

Table 5.2: Results from DL models on binary classification datasets

Application	Dataset	TSK-FRBS Regression					
		Central			Federated		
		R2	MSE	MAE	R2	MSE	MAE
Health	Life Expectancy	$86\% \pm 3.3\%$	10.66 ± 2.53	2.31 ± 0.25	$-14.3\% \pm 118\%$	85.4 ± 77.1	5.28 ± 1.08
Aviation	Aileron	$72.8\% \pm 4.5\%$	$4.49e^{-8} \pm 3.85e^{-8}$	$0.0001 \pm 9.79e^{-6}$	$69.4\% \pm 5.6\%$	$5.06e^{-8} \pm$	$0.00016 \pm 1.31e^{-5}$
Agriculture	Red Wine Quality	$32.5\% \pm 15.9\%$	0.43 ± 0.09	0.51 ± 0.06	$27.2\% \pm 11.1\%$	0.47 ± 0.07	0.53 ± 0.04
Power	CC Power Plant	$93.1\% \pm 0.9\%$	19.91 ± 2.61	3.53 ± 0.12	$93.1\% \pm 0.9\%$	19.9 ± 2.74	3.55 ± 0.14
Mechanical	Maintenance NP Plant	$97.3\% \pm 0.3\%$	$1.51e^{-6} \pm 2.38e^{-7}$	$0.0007 \pm 5.62e^{-5}$	$95.8\% \pm 0.9\%$	$2.35e^{-6} \pm 5.67e^{-7}$	0.0010 ± 0.0001
Semiconductor	Superconductivity	-4.06 ± 3.36	5943.58 ± 4007.31	28.72 ± 4.79	-917.56 ± 2424.59	$1087415.28 \pm 2894230.74$	96.28 ± 50.47

Table 5.3: Results from TSK-FRBS on regression datasets

Application	Dataset	Deep Learning Regression					
		Central			Federated		
		R2	MSE	MAE	R2	MSE	MAE
Health	Life Expectancy	$88.5\% \pm 12.6\%$	6.8 ± 6.2	2.1 ± 1.0	$54.9\% \pm 43.4\%$	1.27 ± 1.21	1.75 ± 0.82
Aviation	Aileron	$74.8\% \pm 2.7\%$	$3.54e^{-08} \pm 2.25e^{-08}$	$0.0001 \pm 4.88e^{-05}$	$82.1\% \pm 2.7\%$	$2.21e^{-10} \pm 4.65e^{-11}$	$2.94e^{-05} \pm 4.17e^{-06}$
Agriculture	Red Wine Quality	$23.2\% \pm 27.3\%$	0.47 ± 0.3	0.51 ± 0.10	$29.6\% \pm 15.1\%$	0.01 ± 0.003	0.22 ± 0.05
Power	CC Power Plant	$91.5\% \pm 4.5\%$	21.96 ± 6.81	3.82 ± 0.62	$90.8\% \pm 6.7\%$	0.64 ± 0.05	0.23 ± 0.09
Mechanical	Maintenance NP Plant	$71.8\% \pm 81\%$	$1.05e^{-6} \pm 1.20e^{-06}$	0.0008 ± 0.0004	$89.1\% \pm 13.2\%$	$1.87e^{-7} \pm 2.26e^{-7}$	$7.17e^{-5} \pm 4.61e^{-5}$
Semiconductor	Superconductivity	$38\% \pm 97.2\%$	46.3 ± 63.07	4.40 ± 2.40	$56.2\% \pm 3\%$	12.32 ± 0.82	0.44 ± 0.07

Table 5.4: Results from the DL models on regression datasets

5.2 Resulting rules

TSK-FRBS is a rule-based model. Each model consists of a set of rules. The number of rules depends on how many duplicates the fuzzy c-means algorithm makes, and the aggregation function. All the rules in a FRBS model has the same format and structure. The TSK-FRBS models consist of rules on the format of IF antecedent THEN consequent. Below are a sample of a few rules from some of the model implementations.

Example rule from a federated TSK-FRBS model trained on the Combined Cycle Power Plant dataset:

R6: IF (AT IS HIGH) AND (V IS HIGH) AND (AP IS MEDIUM) AND (RH IS MEDIUM) THEN: 5.00e+02 - (7.20e+01 * AT) - (8.24e+00 * V) + (1.62e+00 * AP) - (8.64e+00 * RH)

Example rule from a central TSK-FRBS (non-fl) trained on the Combined Cycle Power Plant dataset:

R4: IF (AT IS HIGH) AND (V IS HIGH) AND (AP IS MEDIUM) AND (RH IS MEDIUM) THEN: 5.01e+02 - (7.24e+01 * AT) - (7.90e+00 * V) + (1.20e+00 * AP) - (8.83e+00 * RH)

Example rule from federated TSK-FRBS trained on life expectancy. The dataset has many features, making the resulting rules very long.

R11: IF (Country IS LOW) AND ... AND (Schooling IS MEDIUM) THEN: 2.93e+01 + (5.13e-02 * Country) - ... - (1.21e+01 * Schooling)

Example rule from central TSK-FRBS (non-fl) model trained on life expectancy Non-FL.

R0: IF (Country IS LOW) AND ... AND (Schooling IS HIGH) THEN: 5.52e+01 + (2.57e-03 * Country) + ... + (9.84e-01 * Schooling)

Example of rule from federated TSK-FRBS on Pima Indians Diabetes dataset. **R9: IF (Pregnancies IS MEDIUM) AND (Glucose IS MEDIUM) AND ... AND (Age IS LOW) THEN: -9.24e-01 + (2.93e-01 * Pregnancies) + (1.35e+00 * Glucose) - ... - (1.79e-01 * Age)**

Example of rules from central TSK-FRBS (non-FL) on Pima Indians Diabetes dataset.

R1: IF (Pregnancies IS LOW) AND (Glucose IS MEDIUM) AND ... AND (Age IS LOW) THEN: -6.21e-01 + (3.85e-02 * Pregnancies) + (9.06e-01 * Glucose) - ... + (3.34e-01 * Age)

5.3 Explanation of the results

The Federated TSK-FRBS models achieve a similar or or slightly worse performance than the central TSK-FRBS models. The results are acceptable on all the datasets except for the life expectancy and the superconductivity datasets. The R2 scores are very low with a high variation. The same can be said about the MSE which is high with a big variance.

The Central TSK-FRBS models perform well and match or beat the federated TSK-FRBS models. The results are good on all the datasets except the superconductivity dataset. The R2 is low with a high variation, and the MSE is very big with a high variation.

The federated DL models also perform well, and its results are better than most of the TSK-FRBS models results. There are exceptions where the DL models perform badly, or the TSK-FRBS models get better results. The datasets where this occurs are the credit card fraud detection, Pima Indians Diabetes, and combined cycle power plant.

The central DL models generally perform the best. They outperform the federated DL models or have similar scores on all the datasets except the Maintenance Naval Propulsion Plants and the Superconductivity datasets.

The rules are all in the same format. The antecedent consists of the different features of the dataset and a fuzzy label. The consequents consist of the same features from the data, ut with a numerical value to multiply the data w

Chapter 6

Discussion

This thesis set out to answer the research question along with four sub questions. The research question is: **Does federated learning reduce the performance of a TSK-FRBS model or a DL model**. The sub questions are: Can a cross-silo Federated TSK-FRBS model with i.i.d data achieve comparable performance to a central TSK-FRBS model? Do FL affect the explainability of a TSK-FRBS models rules? Can a cross-silo federated DL model with i.i.d data achieve the same performance as a central DL model? How does the federated TSK-FRBS models perform compared to the federated DL models on the same datasets?

6.1 Key findings

The key findings are that the performance differences between central and federated learning are comparable with a few exceptions. Both the federated TSK-FRBS model and federated DL models perform similarly to their central models . The performance difference between the inherently explainable TSK-FRBS and DL is also not very big. The DL models outperform the TSK-FRBS models with some exceptions.

The TSK-FRBS models are rule-based, meaning each model consists of a set of rules. The rules have the form IF THEN, and can be understood by a person. The rules presented in the results section 5 are examples from different datasets trained with and without FL. Training a TSK-FRBS model using FL has not altered the rules structure or their explainability. They have the same format trained centrally as with federated learning. The rules are just as explainable with FL as without.

The experiments conducted in this thesis suggest that federated learning does not reduce the performance of TSK-FRBS models or DL models. In most cases, the cross-silo federated TSK-FRBS model with i.i.d data achieves performance comparable to that of a central TSK-FRBS model. Training a TSK-FRBS model in a federated manner does not impact the explainability of the rules in this case. The cross-silo federated DL models using i.i.d data achieve comparable results to the central DL models, even outperforming them on a few occasions. In general, the TSK-FRBS models perform very similarly to the DL models, with some examples of DL achieving the best results and some with TSK-FRBS achieving the best

results.

6.2 Interpretations

The cross-silo federated TSK-FRBS models with i.i.d data achieved performance comparable to that of the central TSK-FRBS models. In general, the performance of the binary classification datasets is closer together than that of the regression datasets. The regression results are also harder to evaluate as they consist of how well the predictions follow the data and how far away from the data the predictions are.

The life expectancy and superconductivity datasets are outliers regarding the performance difference between the central and federated TSK-FRBS models. The central TSK-FRBS model achieved a good result in the life expectancy dataset, whereas the federated TSK-FRBS model did not. The central TSK-FRBS model achieved an R2 of 86% and an MSE of 10.66. The federated TSK-FRBS model achieved an R2 of -14.3%, and a MSE of 85.4 with a high variability in the results. The very small life expectancy dataset can be a reason for the bad performance. Each client gets a small part of the dataset, indicating that each client might not have been able to create good local models, resulting in a bad global model. The central model has access to the complete data and has more data to work with. The k-fold cross-validation might also impact the performance. It can lead to model performance variations due to excluding important data in some folds.

The superconductivity dataset saw an unsatisfactory performance from the central and federated TSK-FRBS models. A low R2 score and a high MSE from both models show that they could not follow the data trend, and the predictions are far from the actual values. This might be because the dataset consists of many features, making it too complex. The superconductivity dataset has 81 features. The models might not be able to capture the complexity of the data without increasing the number of rules. A solution to this might be to use feature extraction techniques.

FL has not affected the explainability of the TSK-FRBS rules. The format of the rules stays the same with or without FL. A TSK-FRBS model consists of rules, and training it using FL does not alter the final model's structure or format. FL means that more clients are contributing rules they have created from their dataset, which can lead to a wider range of rules. The OpenFL-XAI implementation of a TSK-FRBS model sends all the rules from the collaborators to the aggregator. This allows the aggregator to access all the collaborators' rules just as they created them.

The cross-silo federated DL models were able to achieve performance comparable to that of the central DL models. The results on the binary classification datasets are similar, and on the regression datasets, the federated DL models outperformed the central DL models on MSE distance. The expectation was that a central DL model with access to the same data as a federated DL system would achieve a higher performance. In the paper by Zhu et al. [47], they implement a TSK-FRBS on a few different datasets with i.i.d data. Their results show that the centrally trained model mostly outperforms the federated. The difference between federated

and centralized training in his results is small, which is comparable with the results achieved in this thesis.

The TSK-FRBS models perform on par with the DL models but with a slightly lower accuracy. One exception to this is with the credit card fraud detection dataset. The TSK-FRBS models achieved high accuracy, precision, and recall. The DL models, achieved a high accuracy but low precision and recall. The central DL model achieved an accuracy of 99.8%, but a precision and recall of 0.0%. This indicates that the models are predicting wrong on fraudulent transactions. The credit card fraud detection dataset is heavily imbalanced, and the results suggest that the DL models predict the same thing every time. The DL models predict a non-fraudulent transaction every time, and the accuracy stays high since there are so few fraudulent transactions. The TSK-FRBS models perform a lot better on the same dataset. The central TSK-FRBS model achieves an accuracy of 99.9%, a precision of 85.8%, and a recall of 69.7%. This suggests that the TSK-FRBS models managed to pick up on the few fraudulent transactions and learned to recognize them better. DL models learn from patterns in the data, and when the pattern shows that most of the data is non-fraudulent, that's what it will learn. This indicates that the TSK-FRBS models are better at handling heavily unbalanced datasets than the DL models.

The answer to the research question is no. FL does not reduce the performance of TSK-FRBS or DL models on i.i.d datasets. The answer is based on the results and sub-questions. The results also uncovered some challenges regarding the TSK-FRBS model. Those are datasets with few entries and high-dimensional data. These factors had a negative impact on the models and should be avoided. This implies that the downsides in performance using FL for TSK-FRBS models are small, and it should be considered when privacy and explainability are important. The FL DL models did not outperform the federated TSK-FRBS models by a big margin, which shows that federated TSK-FRBS is a good choice where privacy and explainability is important. There is no reason not to choose FL for TSK-FRBS models in regards to explainability.

6.3 Limitations

This section discusses this study's limitations. Five key limitations affect the results.

The experiments in this thesis were conducted on different models using i.i.d data. While this is convenient for performing simulated tests, it does not show the real-world impact of FL. The different clients in a FL system will most likely have unique data in varying size and variety. A FL model might also achieve the better performance as it has access to a bigger set of data.

Another limitation is with the central TSK-FRBS model. This was tested by training it the same way as the federated TSK-FRBS model but with one client instead of five. There should not be any difference in how it works, but using a different implementation of TSK-FRBS without FL could show some impact not discovered in this research.

The federated TSK-FRBS models were trained using the OpenFL-XAI framework, and the federated DL models were trained using the Flower framework. These are two different FL

frameworks. The DL models should have been trained using the same framework as the TSK-FRBS model. Due to the challenges with implementing the DL models on OpenFL-XAI, the Flower framework was used instead. There might be some inconsistencies in the training between the different frameworks that can impact the results.

A change in the data pre-processing and further model optimization could be done. While the pre-processing is similar across all the datasets, there is potential for improvement. This could be by refining the current pre-processing or exploring more advanced methods. A more thorough and careful model and hyperparameter tuning could lead to better models. It is possible that with more data pre-processing and model tuning the results could be different.

The experiments were conducted with a specific number of clients. A different number of clients would change the amount of data each client has access to and could lead to a The impact of FL on the TSK-FRBS and DL models might vary depending on the number of clients.

6.4 Recommendation

Most of the research within the field of FL is primarily focused on DL. This thesis has explored the impact of FL on the inherently explainable ML model TSK-FRBS. Some recommendations for implementation are the data and frameworks. When implementing federated systems, data and data division should be considered. It's important to ensure all clients have enough and different data. While i.i.d data was used in this thesis, real-world scenarios will often have non-i.i.d data. This increased amount of data and diversity is positive, but TSK-FRBS models can struggle with high-dimensional data. There are many frameworks available for training federated models. Its important to explore the different options and see which ones fit your needs and previous experience.

Chapter 7

Conclusion and future work

Privacy and explainability are important for trustworthy AI systems. Implementing both TSK-FRBS and federated learning is a way to ensure a model's privacy and explainability. A set of different models, such as central TSK-FRBS, federated TSK-FRBS, central DL, and federated DL, have been implemented. These have been used to evaluate the impact of FL on an inherently explainable ML model. The first-order TSK-FRBS models have been implemented using the OpenFL-XAI framework. The impact of FL on the TSK-FRBS models has been assessed by comparing them with central TSK-FRBS and DL models. This thesis concludes that FL has a low impact on TSK-FRBS in terms of performance and explainability. The results show that performance is comparable to not using FL, and the rules explainability is the same with and without FL. TSK-FRBS using FL should be considered when privacy and explainability are important. To the best of the researchers' knowledge, this is the only research that investigates possible trade-offs between central and federated learning for inherently explainable ML and compares the results with DL.

7.1 Future work

Inherently explainable models such as TSK-FRBS trained using FL have shown promising results. Explainability and privacy will become more important in the future with the passing of the AI Act. This research explores the impact of FL on i.i.d data. For future work, it would be interesting to see what the impact is with non-i.i.d data and a different number of clients. This research used five clients, and experiments using larger datasets and more clients could lead to new findings. In this project, the explainability has only been analyzed by comparing the rules before and after FL. Implementing explainability algorithms on the different models and compare the results could also be a direction for future work. Another direction for future research and development is to enable and implement the training of other transparent ML models using FL, e.g. regression models or decision trees. How can the extra explainability of a system using these techniques be used to make a better application. Future research can explore when it is recommended to use a federated TSK-FRBS over DL. In which scenarios and at what risk levels does the accuracy gain of a DL model justify using it over federated TSK-FRBS.

Bibliography

[1] C. Marsala and D. Petturiti, "Splitting rules for monotone fuzzy decision trees," in *Conference of the European Society for Fuzzy Logic and Technology*. Springer, 2023, pp. 161–173.

[2] J. M. Mendel, "Uncertain rule-based fuzzy systems," *Introduction and new directions*, vol. 684, 2017.

[3] N. Rieke, "What is federated learning?" Oct. 2019. [Online]. Available: https://blogs.nvidia.com/blog/what-is-federated-learning/

[4] M. Peixeiro, "Step-by-step guide to building your own neural network from scratch," 2019. [Online]. Available: https://towardsdatascience.com/step-by-step-guide-to-building-your-own-neural-network-from-scratch-df64b1c5ab6e

[5] K. Schwab, *The fourth industrial revolution*. Crown Currency, 2017.

[6] "Artificial intelligence act: Meps adopt landmark law." [Online]. Available: https://www.europarl.europa.eu/news/en/press-room/20240308IPR19015/artificial-intelligence-act-meps-adopt-landmark-law

[7] "Eu ai act: first regulation on artificial intelligence." [Online]. Available: https://www.europarl.europa.eu/topics/en/article/20230601STO93804/eu-ai-act-first-regulation-on-artificial-intelligence

[8] T. Madiega, "Artificial intelligence act," *European Parliament: European Parliamentary Research Service*, 2021.

[9] (2019) Ethics guidelines for trustworthy ai. [Online]. Available: https://digital-strategy.ec.europa.eu/en/library/ethics-guidelines-trustworthy-ai

[10] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nature machine intelligence*, vol. 1, no. 5, pp. 206–215, 2019.

[11] A. Nilsson, S. Smith, G. Ulm, E. Gustavsson, and M. Jirstrand, "A performance evaluation of federated learning algorithms," in *Proceedings of the second workshop on distributed infrastructures for deep learning*, 2018, pp. 1–8.

[12] M. Daole, A. Schiavo, J. L. C. Bárcena, P. Ducange, F. Marcelloni, and A. Renda, "Openfl-xai: Federated learning of explainable artificial intelligence models in python," *SoftwareX*, vol. 23, p. 101505, 2023.

[13] "Ai act: Shaping europe's digital future." [Online]. Available: https://digital-strategy.ec.europa.eu/en/policies/regulatory-framework-ai

[14] E. D. P. SUPERVISOR, "Data protection." [Online]. Available: https://www.edps.europa.eu/data-protection/data-protection_en

[15] S. Ali, T. Abuhmed, S. El-Sappagh, K. Muhammad, J. M. Alonso-Moral, R. Confalonieri, R. Guidotti, J. Del Ser, N. Díaz-Rodríguez, and F. Herrera, "Explainable artificial intelligence (xai): What we know and what is left to attain trustworthy artificial intelligence," *Information fusion*, vol. 99, p. 101805, 2023.

[16] A. Adadi and M. Berrada, "Peeking inside the black-box: a survey on explainable artificial intelligence (xai)," *IEEE access*, vol. 6, pp. 52 138–52 160, 2018.

[17] M. Van Lent, W. Fisher, and M. Mancuso, "An explainable artificial intelligence system for small-unit tactical behavior," in *Proceedings of the national conference on artificial intelligence*. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2004, pp. 900–907.

[18] W. Saeed and C. Omlin, "Explainable ai (xai): A systematic meta-survey of current challenges and future opportunities," *Knowledge-Based Systems*, vol. 263, p. 110273, 2023.

[19] A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins *et al.*, "Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai," *Information fusion*, vol. 58, pp. 82–115, 2020.

[20] B. Goodman and S. Flaxman, "European union regulations on algorithmic decision-making and a "right to explanation"," *AI magazine*, vol. 38, no. 3, pp. 50–57, 2017.

[21] L. Magdalena, *Fuzzy Rule-Based Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 203–218. [Online]. Available: https://doi.org/10.1007/978-3-662-43505-2_13

[22] J. M. Mendel, "Introduction to rule-based fuzzy logic systems," https://ewh.ieee.org/cmte/cis/mtsc/ieeecis/Intro_to_Rule_Based_FLSs.pdf.

[23] J. C. Bezdek, R. Ehrlich, and W. Full, "Fcm: The fuzzy c-means clustering algorithm," *Computers & geosciences*, vol. 10, no. 2-3, pp. 191–203, 1984.

[24] A. K. Varshney and V. Torra, "Literature review of various fuzzy rule based systems," *arXiv preprint arXiv:2209.07175*, 2022.

[25] E. H. Mamdani, "Application of fuzzy algorithms for control of simple dynamic plant," in *Proceedings of the institution of electrical engineers*, vol. 121, no. 12. IET, 1974, pp. 1585–1588.

[26] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," *IEEE transactions on systems, man, and cybernetics*, no. 1, pp. 116–132, 1985.

[27] M. Sugeno and G. Kang, "Structure identification of fuzzy model," *Fuzzy sets and systems*, vol. 28, no. 1, pp. 15–33, 1988.

[28] D. Kukolj, "Design of adaptive takagi–sugeno–kang fuzzy models," *Applied Soft Computing*, vol. 2, no. 2, pp. 89–103, 2002.

[29] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings *et al.*, "Advances and open problems in federated learning," *Foundations and Trends® in Machine Learning*, vol. 14, no. 1–2, pp. 1–210, 2021.

[30] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[31] "Federated learning: Collaborative machine learning without centralized training data." [Online]. Available: https://blog.research.google/2017/04/federated-learning-collaborative.html

[32] A. Hard, C. M. Kiddon, D. Ramage, F. Beaufays, H. Eichner, K. Rao, R. Mathews, and S. Augenstein, "Federated learning for mobile keyboard prediction," 2018. [Online]. Available: https://arxiv.org/abs/1811.03604

[33] A. D. P. Team, "Learning with privacy at scale," *Apple Machine Learning Research*, 2017.

[34] P. Qi, D. Chiaro, A. Guzzo, M. Ianni, G. Fortino, and F. Piccialli, "Model aggregation techniques in federated learning: A comprehensive survey," *Future Generation Computer Systems*, 2023.

[35] P. Foley, M. J. Sheller, B. Edwards, S. Pati, W. Riviera, M. Sharma, P. N. Moorthy, S.-h. Wang, J. Martin, P. Mirhaji, P. Shah, and S. Bakas, "Openfl: the open federated learning library," *Physics in Medicine & Biology*, 2022. [Online]. Available: http://iopscience.iop.org/article/10.1088/1361-6560/ac97d9

[36] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, J. Fernandez-Marques, Y. Gao, L. Sani, K. H. Li, T. Parcollet, P. P. B. de Gusmão *et al.*, "Flower: A friendly federated learning research framework," *arXiv preprint arXiv:2007.14390*, 2020.

[37] N. Rodríguez-Barroso, D. Jiménez-López, M. V. Luzón, F. Herrera, and E. Martínez-Cámara, "Survey on federated learning threats: Concepts, taxonomy on attacks and defences, experimental study and challenges," *Information Fusion*, vol. 90, pp. 148–173, 2023.

[38] S. Sharma, S. Sharma, and A. Athaiya, "Activation functions in neural networks," *Towards Data Sci*, vol. 6, no. 12, pp. 310–316, 2017.

[39] I. H. Sarker, "Deep learning: a comprehensive overview on techniques, taxonomy, applications and research directions," *SN Computer Science*, vol. 2, no. 6, p. 420, 2021.

[40] A. Oppermann, "What is deep learning and how does it work?" 2022. [Online]. Available: https://builtin.com/machine-learning/deep-learning

[41] saumyasaxena2730, "Introduction to deep learning," 2018. [Online]. Available: https://www.geeksforgeeks.org/introduction-deep-learning/

[42] R. López-Blanco, R. S. Alonso, A. González-Arrieta, P. Chamoso, and J. Prieto, "Federated learning of explainable artificial intelligence (fed-xai): A review," in *International Symposium on Distributed Computing and Artificial Intelligence*. Springer, 2023, pp. 318–326.

[43] A. Bechini, A. Bondielli, P. Ducange, F. Marcelloni, A. Renda *et al.*, "Responsible artificial intelligence as a driver of innovation in society and industry," in *Atti del Secondo Convegno Nazionale CINI sull'Intelligenza Artificiale*, 2022.

[44] P. Ducange, F. Marcelloni, D. Micheli, G. Nardini, A. Renda, D. Sabella, G. Stea, and A. Virdis, "Trustworthy ai for next generation networks: the fed-xai innovative paradigm from the hexa-x eu flagship project," 2022.

[45] J. L. C. Bárcena, M. Daole, P. Ducange, F. Marcelloni, A. Renda, F. Ruffini, and A. Schiavo, "Fed-xai: Federated learning of explainable artificial intelligence models," in *3rd Italian Workshop on Explainable Artificial Intelligence (XAI. it 2022)*, 2022.

[46] J. L. C. Bárcena, P. Ducange, A. Ercolani, F. Marcelloni, and A. Renda, "An approach to federated learning of explainable fuzzy regression models," in *2022 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE, 2022, pp. 1–8.

[47] X. Zhu, D. Wang, W. Pedrycz, and Z. Li, "Horizontal federated learning of takagi–sugeno fuzzy rule-based models," *IEEE Transactions on Fuzzy Systems*, vol. 30, no. 9, pp. 3537–3547, 2021.

[48] A. Wilbik and P. Grefen, "Towards a federated fuzzy learning system," in *2021 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE, 2021, pp. 1–6.

[49] A. Kusiak, "Federated explainable artificial intelligence (fxai): a digital manufacturing perspective," *International journal of production research*, vol. 62, no. 1-2, pp. 171–182, 2024.

[50] T. Awosika, R. M. Shukla, and B. Pranggono, "Transparency and privacy: The role of explainable ai and federated learning in financial fraud detection," *arXiv preprint arXiv:2312.13334*, 2023.

[51] S. Jesus, J. Pombal, D. Alves, A. Cruz, P. Saleiro, R. Ribeiro, J. Gama, and P. Bizarro, "Turning the tables: Biased, imbalanced, dynamic tabular datasets for ml evaluation," *Advances in Neural Information Processing Systems*, vol. 35, pp. 33 563–33 575, 2022.

[52] P. Chen, X. Du, Z. Lu, J. Wu, and P. C. Hung, "Evfl: An explainable vertical federated learning for data-oriented artificial intelligence systems," *Journal of Systems Architecture*, vol. 126, p. 102474, 2022.

[53] J. Fiosina, "Interpretable privacy-preserving collaborative deep learning for taxi trip duration forecasting," in *International Conference on Vehicle Technology and Intelligent Transport Systems*. Springer, 2021, pp. 392–411.

[54] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

[55] K. Rajarshi, "Life expectancy (who)," 2018. [Online]. Available: https://www.kaggle.com/datasets/kumarajarshi/life-expectancy-who/data

[56] "Life expectancy (who)," UCI Machine Learning, 2018. [Online]. Available: https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database

[57] "Credit card fraud detection," UCI Machine Learning, 2013. [Online]. Available: https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud

[58] "Smart grid stability," UCI Machine Learning, 2019. [Online]. Available: https://www.kaggle.com/datasets/pcbreviglieri/smart-grid-stability/data

[59] "Nsl-kdd," NSL-KDD, 2019. [Online]. Available: https://www.kaggle.com/datasets/hassan06/nslkdd

[60] "Ailerons data set," https://sci2s.ugr.es/keel/dataset.php?cod=93#sub2, KEEL: Knowledge Extraction and Evaluation Lab, accessed: 04.03.2024.

[61] P. Cortez, A. Cerdeira, F. Almeida, T. Matos, and J. Reis, "Modeling wine preferences by data mining from physicochemical properties," *Decision support systems*, vol. 47, no. 4, pp. 547–553, 2009.

[62] "Red wine quality," UCI machine learning repository, 2009. [Online]. Available: https://www.kaggle.com/datasets/uciml/red-wine-quality-cortez-et-al-2009/data

[63] P. Tüfekci, "Prediction of full load electrical power output of a base load operated combined cycle power plant using machine learning methods," *International Journal of Electrical Power & Energy Systems*, vol. 60, pp. 126–140, 2014. [Online]. Available: https://api.semanticscholar.org/CorpusID:111365542

[64] P. Tfekci and H. Kaya, "Combined Cycle Power Plant," UCI Machine Learning Repository, 2014, DOI: https://doi.org/10.24432/C5002N.

[65] A. Coraddu, L. Oneto, A. Ghio, S. Savio, D. Anguita, and M. Figari, "Machine learning approaches for improving condition-based maintenance of naval propulsion plants," *Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment*, vol. 230, no. 1, pp. 136–153, 2016.

[66] A. Coraddu, h. L. Oneto, A. Ghio, S. Savio, D. Anguita, and M. Figari, "Condition Based Maintenance of Naval Propulsion Plants," UCI Machine Learning Repository, 2014, DOI: https://doi.org/10.24432/C5K31K.

[67] K. Hamidieh, “A data-driven statistical model for predicting the critical temperature of a superconductor,” *Computational Materials Science*, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:55069173

[68] K. Hamidieh, “Superconductivty Data,” UCI Machine Learning Repository, 2018, DOI: https://doi.org/10.24432/C53P47.

[69] J. Starmer, “R-squared, clearly explained!!!” November 2022, StatQuest with Josh Starmer. [Online]. Available: https://youtu.be/bMccdk8EdGo?si=YdhPDbIE2Qddu84C

[70] [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.r2_score.html

[71] nandakishorereddy, “Regression metrics,” 2023. [Online]. Available: https://www.geeksforgeeks.org/regression-metrics/

[72] J. Frost. [Online]. Available: https://statisticsbyjim.com/regression/mean-squared-error-mse/

[73] J. Brownlee, “How to calculate precision, recall, and f-measure for imbalanced classification.” [Online]. Available: https://machinelearningmastery.com/precision-recall-and-f-measure-for-imbalanced-classification/

Appendix A

Code

The implemented code is available on GitHub.
`https://github.com/Halvorte/FL-Explainability`

Appendix B

DL model architectures

Life expectancy:

```
class Net(nn.Module):
def __init__(self, in_features: int, out_features: int) -> None:
    super(Net, self).__init__()
    self.lin1 = nn.Linear(in_features, 128)
    self.lin2 = nn.Linear(128, 64)
    self.lin3 = nn.Linear(64, out_features)
    self.rel = nn.ReLU()
    self.dropout = nn.Dropout(0.25)

def forward(self, x):
    x = self.lin1(x)
    x = self.rel(x)
    x = self.lin2(x)
    x = self.rel(x)
    x = self.lin3(x)
    return x
```

Model parameters	Central	Federated
Learning rate	0.001	0.0001
Epochs	900	40
Batch size	32	32
Optimizer	Adam	Adam
Loss function	MSELoss	MSELoss
Clients		5
Client epochs		400

Table B.1: Life expectancy DL parameters

Pima Indians Diabetes:

```
class Net(nn.Module):
```

```
    def __init__(self, in_features: int, out_features: int) -> None:
        super(Net, self).__init__()
        self.lin1 = nn.Linear(in_features, 16)
        self.lin2 = nn.Linear(16, out_features)
        self.rel = nn.ReLU(inplace=True)
        self.drop = nn.Dropout(0.2)
        #self.sigmoid = F.sigmoid()

    def forward(self, x):
        x = self.lin1(x)
        x = self.rel(x)
        x = self.drop(x)
        x = self.lin2(x)
        x = F.sigmoid(x)
        return x
```

Model parameters	Central	Federated
Learning rate	0.001	0.001
Epochs	100	50
Batch size	32	32
Optimizer	Adam	Adam
Loss function	MSELoss	MSELoss
Clients		5
Client epochs		55

Table B.2: Pima Indians diabetes DL parameters

Credit Card Fraud Detection:

```
    class Net(nn.Module):
    def __init__(self, in_features: int, out_features: int) -> None:
        super(Net, self).__init__()
        self.lin1 = nn.Linear(in_features, 64)
        self.lin2 = nn.Linear(64, 32)
        self.lin3 = nn.Linear(32, out_features)
        self.rel = nn.ReLU(inplace=True)
        self.drop = nn.Dropout(0.2)
        self.sigmoid = nn.Sigmoid()

    def forward(self, x):
        x = self.lin1(x)
        x = self.rel(x)
        x = self.drop(x)
        x = self.lin2(x)
        x = self.rel(x)
        x = self.drop(x)
        x = self.lin3(x)
```

```
        x = self.sigmoid(x)
        return x
```

Model parameters	Central	Federated
Learning rate	0.01	0.001
Epochs	30	10
Batch size	10000	32
Optimizer	Adam	Adam
Loss function	MSELoss	MSELoss
Clients		5
Client epochs		10

Table B.3: Credit card fraud detection DL parameters

Smart Grid Stability:

```
    class Net(nn.Module):
    def __init__(self, in_features: int, out_features: int) -> None:
        super(Net, self).__init__()
        self.lin1 = nn.Linear(in_features, 64)
        self.lin2 = nn.Linear(64, out_features)
        self.rel = nn.ReLU(inplace=True)
        self.drop = nn.Dropout(0.2)
        self.sigmoid = nn.Sigmoid()

    def forward(self, x):
        x = self.lin1(x)
        x = self.rel(x)
        x = self.drop(x)
        x = self.lin2(x)
        x = self.sigmoid(x)
        return x
```

Model parameters	Central	Federated
Learning rate	0.0001	0.0001
Epochs	30	10
Batch size	32	32
Optimizer	Adam	Adam
Loss function	MSELoss	MSELoss
Clients		5
Client epochs		15

Table B.4: Smart grid stability DL parameters

Network Security Binary Classification:

```
    class Net(nn.Module):
    def __init__(self, in_features: int, out_features: int) -> None:
        super(Net, self).__init__()
        self.lin1 = nn.Linear(in_features, 128)
        self.lin2 = nn.Linear(128, 64)
        self.lin3 = nn.Linear(64, out_features)
        self.rel = nn.ReLU(inplace=True)
        self.drop = nn.Dropout(0.2)
        self.sigmoid = nn.Sigmoid()

    def forward(self, x):
        x = self.lin1(x)
        x = self.rel(x)
        x = self.drop(x)
        x = self.lin2(x)
        x = self.rel(x)
        x = self.drop(x)
        x = self.lin3(x)
        x = self.sigmoid(x)
        return x
```

Model parameters	Central	Federated
Learning rate	0.0001	0.001
Epochs	30	10
Batch size	32	32
Optimizer	Adam	Adam
Loss function	MSELoss	MSELoss
Clients		5
Client epochs		10

Table B.5: Network security DL parameters

Aileron model

```
class Net(nn.Module):
    def __init__(self, in_features: int, out_features: int) -> None:
        super(Net, self).__init__()
        self.lin1 = nn.Linear(in_features, 128)
        self.lin2 = nn.Linear(128, 64)
        self.lin3 = nn.Linear(64, out_features)
        self.rel = nn.ReLU()
        self.dropout = nn.Dropout(0.2)

    def forward(self, x):
        x = self.lin1(x)
        x = self.rel(x)
        x = self.dropout(x)
```

```
        x = self.lin2(x)
        x = self.rel(x)
        x = self.dropout(x)
        x = self.lin3(x)
        return x
```

Model parameters	Central	Federated
Learning rate	0.0001	0.0001
Epochs	150	20
Batch size	32	128
Optimizer	Adam	Adam
Loss function	MSELoss	MSELoss
Clients		5
Client epochs		20

Table B.6: Aileron DL parameters

Red Wine Quality:

```
    class Net(nn.Module):
    def __init__(self, in_features: int, out_features: int) -> None:
        super(Net, self).__init__()
        self.lin1 = nn.Linear(in_features, 128)
        self.lin2 = nn.Linear(128, 64)
        self.lin3 = nn.Linear(64, out_features)
        self.rel = nn.ReLU()
        self.dropout = nn.Dropout(0.2)
        self.batch_norm1 = nn.BatchNorm1d(128)
        self.batch_norm2 = nn.BatchNorm1d(64)

    def forward(self, x):
        x = self.lin1(x)
        x = self.batch_norm1(x)
        x = self.rel(x)
        x = self.dropout(x)
        x = self.lin2(x)
        x = self.batch_norm2(x)
        x = self.rel(x)
        x = self.dropout(x)
        x = self.lin3(x)
        return x
```

Model parameters	Central	Federated
Learning rate	0.0001	0.0001
Epochs	400	10

Batch size	32	32
Optimizer	Adam	Adam
Loss function	MSELoss	MSELoss
Clients		5
Client epochs		150

Table B.7: Red wine quality DL parameters

Cobined Cycle Power Plant:

```
class Net(nn.Module):
def __init__(self, in_features: int, out_features: int) -> None:
    super(Net, self).__init__()
    self.lin1 = nn.Linear(in_features, 8)
    self.lin2 = nn.Linear(8, out_features)
    self.rel = nn.ReLU()

def forward(self, x):
    x = self.lin1(x)
    x = self.rel(x)
    x = self.lin2(x)
    return x
```

Model parameters	Central	Federated
Learning rate	0.0001	0.0001
Epochs	30	10
Batch size	32	32
Optimizer	SGD	SGD
Loss function	MSELoss	MSELoss
Clients		5
Client epochs		100

Table B.8: Combined cycle power plant DL parameters

Condition based maintenance of naval propulsion plants:

```
class Net(nn.Module):
def __init__(self, in_features: int, out_features: int) -> None:
    super(Net, self).__init__()
    self.lin1 = nn.Linear(in_features, 64)
    self.lin2 = nn.Linear(64, out_features)
    self.rel = nn.ReLU()

def forward(self, x):
    x = self.lin1(x)
```

```
        x = self.rel(x)
        x = self.lin2(x)
        return x
```

Model parameters	Central	Federated
Learning rate	0.0001	0.0001
Epochs	250	20
Batch size	32	32
Optimizer	Adam	Adam
Loss function	MSELoss	MSELoss
Clients		5
Client epochs		150

Table B.9: Maintenance naval propulsion plants DL parameters

Superconductivity data:

```
class Net(nn.Module):
    def __init__(self, in_features: int, out_features: int) -> None:
        super(Net, self).__init__()
        self.lin1 = nn.Linear(in_features, 256)
        self.lin2 = nn.Linear(256, 128)
        self.lin3 = nn.Linear(128, out_features)
        self.rel = nn.ReLU()
        self.dropout = nn.Dropout(0.2)

    def forward(self, x):
        x = self.lin1(x)
        x = self.rel(x)
        x = self.dropout(x)
        x = self.lin2(x)
        x = self.rel(x)
        x = self.dropout(x)
        x = self.lin3(x)
        return x
```

Model parameters	Central	Federated
Learning rate	0.0001	0.0001
Epochs	20	40
Batch size	32	32
Optimizer	Adam	Adam
Loss function	MSELoss	MSELoss
Clients		5

Client epochs		15

Table B.10: Superconductivity DL parameters