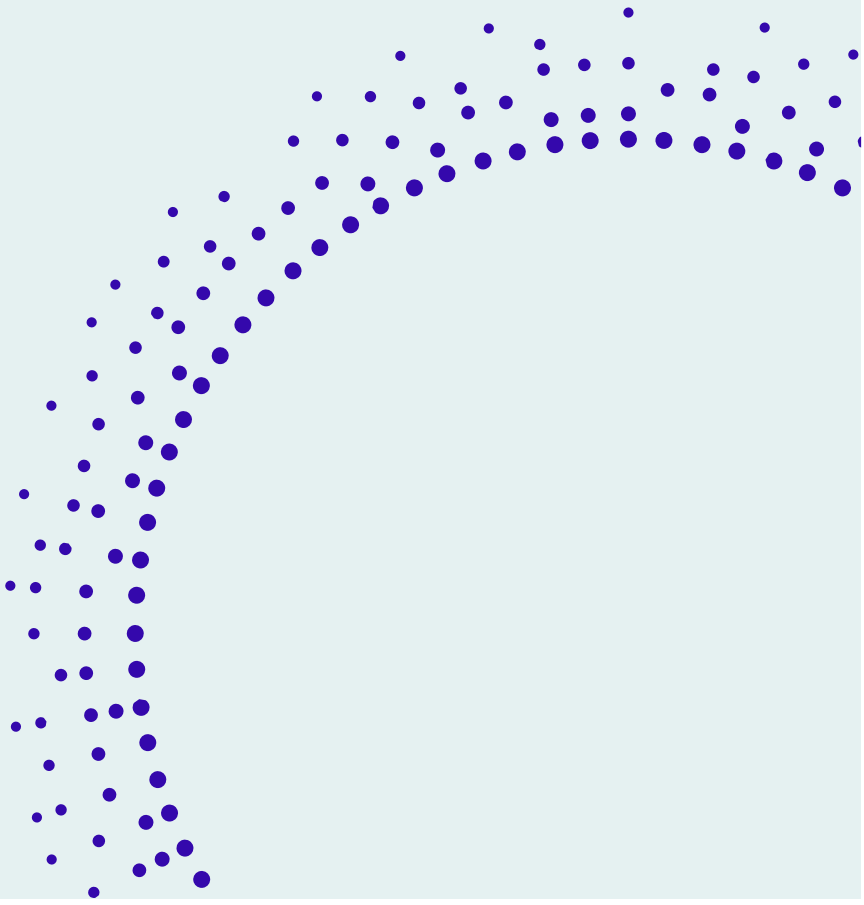
Federated Learning in Aquaculture

Exploring the potential for cross-industry collaboration in the Norwegian aquaculture sector using federated machine learning on AquaCloud data

Authors | Anders Ohma, Miles Granger, Anders D. Sleire
Report | Nr. 1-2025 – NORCE Energy and technology



Støttet av: Vestland fylkeskommune

Report title	Federated Learning in Aquaculture
Project number	NORCE 108795
Institution	NORCE
Client(s)	AquaCloud AS
Classification	Open
Report number.	1-2025 NORCE Analytics
ISBN	[Write text here]
Number of pages	24
Publishing date	[Write text here]
CC-license	CC-BY-SA
Keywords	Aquaculture, federated learning, industry collaboration, sensitive data

Disclaimer

Executive summary

The aquaculture industry faces significant challenges that require innovative solutions, particularly when dealing with sensitive data and the need for collaboration between competitors. AquaCloud AS has played a key role in standardizing data collection and facilitating data sharing across the industry. By developing standards for sensor data, fish health data, and environmental data, AquaCloud ensures that information is collected and reported consistently.

Given the sensitivity of some of the data, it is crucial to utilize AquaCloud's resources safely. Federated machine learning offers a promising approach by enabling collaborative data analysis while maintaining data privacy and security. This secure sharing of models and learning from the unique AquaCloud centralized data platform, while data remains distributed, not only maintains privacy but also provides individual fish farmers with access to a broader dataset. By having access to more comprehensive data, than governance limitations dictates, fish farmers can make better-informed decisions, optimize their operations, and improve fish health and welfare. This method would also open AquaCloud insights to benchmarking and product fit evaluation for external actors.

In this report, we prove that the method is applicable to AquaCloud and demonstrate how federated learning can improve model performance in the aquaculture industry based on AquaCloud data. The report describes the foundation for how a non-invasive platform can be implemented to yield flexible and secure collaborative model training. Implementing such infrastructure for federated learning have the potential to increase industry collaboration and generate significant value, driving advancements in fish health, operational efficiency, and overall sustainability.

Contents

1 Introduction and background

The aquaculture industry is one of the largest export industries in Norway, but faces significant challenges related to fish health and environmental impact. Addressing the complex challenges requires close collaboration among companies within the sector, as well as with government bodies and research institutions. Effective collaboration hinges on the ability to share and act upon insights derived from data in a standardized and secure manner.

This report explores how federated learning may offer a solution. Unlike conventional methods, federated learning enables organizations to collaboratively train machine learning models while keeping data decentralized and secure. Using real-world data from AquaCloud, we demonstrate how federated learning is a performant alternative to conventional model training, enhancing model performance for all participants and overcoming data-sharing barriers.

We also evaluate various technical solutions for implementing federated learning, from existing frameworks to a custom-built platform, outlining a path toward a secure, flexible, and scalable system tailored to the aquaculture industry. By harnessing this technology, stakeholders can unlock new insights, drive innovation, and build a more resilient and sustainable industry.

Below, we discuss some challenges faced by the Norwegian aquaculture industry, particularly the high mortality rates among fish populations. We emphasize the potential of federated learning to address these issues by enabling collaborative training of machine learning models while maintaining data privacy. We provide a general introduction to federated learning in section 2.

We present a quantitative analysis in section 3, demonstrating how federated learning compares to conventional learning using data from AquaCloud. We use different algorithms and strategies when we model the data to demonstrate that federated learning is both a performant and a flexible alternative to conventional training and that data sharing increases performance for all participants.

In section 4, we evaluate various technical solutions for implementing federated learning, including existing frameworks and the development of a custom platform. We discuss the advantages and challenges of different approaches and propose a flexible, secure, and non-invasive federated learning platform. Section 5 concludes the report, summarizing the key findings.

1.1 Industry challenges

The Fish Health Report 2023 from the Norwegian Veterinary Institute (Sommerset et al., 2024) disclose serious challenges in the Norwegian aquaculture sector. High mortality rates for salmon and rainbow trout are reported in all phases of production. For salmon in the sea phase, the mortality rates are the highest recorded so far. The Norwegian Veterinary Institute reports that the three most prominent health issues in 2023 are the same as in 2022: injuries from delousing operations, complex gill disease, and winter ulcers. Additionally, there is a concerning increase in several infectious diseases.

Deteriorating fish health represents a major challenge for the aquaculture industry and society at large. The underlying mechanisms are complex, and individual farmers have limited ability to combat the problem on their own. The highlighted challenges are industry-wide issues that require efforts across stakeholders and greater collaboration in data sharing and analysis. On the other hand, there are limits to the data that stakeholders can exchange without conflicting with current regulations, such as the securities trading act and competition law.

1.2 Standardization and sharing of data in AquaCloud

The aquaculture industry is characterized by a vast amount of data generated from various sources, including sensors, fish health records, and environmental monitoring. However, this data often remains siloed within individual organizations, limiting its potential to drive industry-wide improvements.

The big data initiative AquaCloud was established in 2017 as a project within the industry cluster NCE Seafood Innovation. The original purpose was to establish a secure database for storing data and analysis to identify where outbreaks of salmon lice were likely to occur. Since then, the project has evolved into a hub for various industry activities, including the development of standards for sensor and environmental data, as well as fish health.

Developing common standards for the industry and sharing insights from data is a prerequisite for stakeholders to collaborate effectively and improve in areas such as disease management, operational efficiency, and sustainable practices. At the time of writing this report, data from farms producing approximately 55% of the biomass in Norwegian aquaculture is represented in AquaCloud. By participation in the collaboration, the individual fish farmer contributes to a large industry-wide data set, containing information the individual companies may not have access to when acting on their own. This illustrates how increased utilization of the data in AquaCloud may generate large value through research and analysis for the aquaculture industry, both in Norway and internationally.

1.3 Objectives

Main Goal:

Mortality is the single biggest problem in the aquaculture sector. The main goal of the project is to explore the potential for using federated machine learning in aquaculture to combat mortality related to delousing operations.

Sub-goals:

1. Identify data for analyzing mortality related to delousing across the sector and identify new data that should be included in AquaCloud to better understand mortality.

2. Describe a federated machine learning framework for data available in AquaCloud.

2 Materials and methods

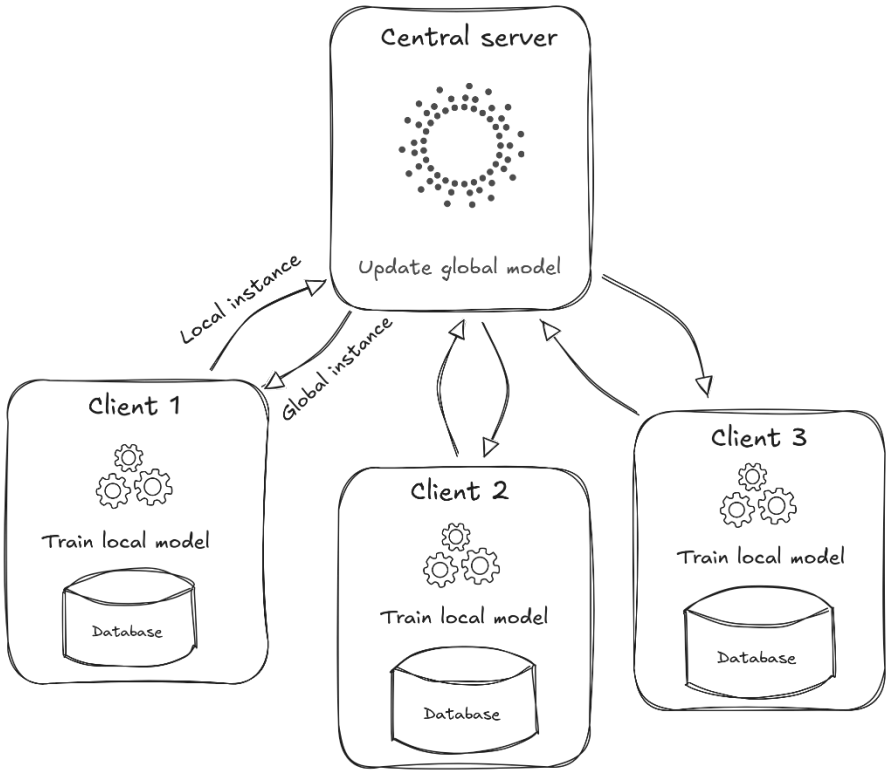
Before we dive into the quantitative and technical analyses of federated learning applied to AquaCloud data, we give a general introduction to key concepts in federated learning. While optimalization using distributed data has been studied for a long time, it is only in the last decade that federated learning has been used extensively in an applied sense.

2.1 Federated machine learning

Federated learning is an approach to collaboratively train machine learning models from decentralized data without exchanging raw data. The term federated learning was coined by McMahan et al. (2017), and refers to the federation of participating clients which, under the coordination of a central server, are used to train the models. This is fundamentally different from conventional machine learning, in which all data is gathered at the central server prior to training. Instead, model updates are computed locally, and only aggregated results are sent to the server, which are then combined from several clients to improve the global model.

The training process is illustrated in the sketch below. Initiated by the server, training starts at one, several or all clients using local data. Training can start from scratch or continue from a pretrained model. After a set number of local iterations, the local model instances are returned to the server. This could be in the form of coefficients or weights, gradients, new splits or trees, or the entire model object, depending on the algorithm being used. At the server side, the local models are used to update the global model by aggregating or appending information from the local instances. Once updated, the global model instance is distributed to new clients or returned to the original clients for further training. This is repeated for several global epochs, ending after a predefined number of iterations or by some early stopping logic. The trained model must be evaluated locally, unless some central test data is available. This could involve clients not used in training or using left-out data at each client. Regardless, different metrics are calculated at the clients and aggregated at the server, to be used both for tuning the model and for final evaluation of

the performance. Both globally aggregated and local metrics are useful when evaluating a model, since different use cases desire optimal performance globally or locally.



A key benefit of federated learning is that data remains decentralized, ensuring privacy and reducing the need for large-scale data transmission. As sharing of raw data is simply not possible in many situations, federated learning enables access to more data for model training that would otherwise be inaccessible. However, this comes with several costs. Training and evaluation become more complex, and clients have different amounts of non-independent and identically distributed (non-IID) data of different quality. Federated learning also requires direct communication between the server and the clients and sufficient computational resources at each client. While a federated setup reduces certain security risks, like an attack against a central server with raw data from different sources, it opens new risks that must be mitigated, like malicious clients and the possibility of extracting sensitive data from the trained model itself.

Broadly speaking, federated learning can be split into cross-device and cross-silo learning (Kairouz et al., 2021). Cross-device learning is related to mobile and edge device applications, characterized by a vast number of devices. An example could be a model for image classification in a mobile app. Cross-silo learning refers to situations where different organizations or companies utilize their collective data to produce better models but are for different reasons prohibited or unwilling to share raw data. This can refer to situations where organizations either have different records with the same attributes (horizontal) or have different attributes for the same records (vertical), or both. An example is banks collaborating on a model to detect money laundering attempts. Cross-device and cross-silo learning share many traits, but there are also some key differences. For example, communication is the key bottleneck in cross-device learning, and thus limiting the number

of requests between server and clients is of uttermost importance for efficient training, affecting the setup of the federated leaning applications. For cross-silo training, such considerations are less important, since the participating organizations can configure network and compute resources to minimize communication as a limiting factor, enabling a more flexible setup. On the other hand, organizations may have different types of data, use different data structures, or maintain data of varying quality. They might also have different payoff when participating in shared model training based on their available data and in-house data science expertise, which might introduce the need for incentive mechanisms based on each participants contribution (Wang et al., 2019).

FedAqua is a cross-silo project, where significant amounts of data from a limited number of fish farmers are available through AquaCloud. Since the data from all owners follows the same schema, this is a horizontal situation where the same data attributes are available for each client with similar data quality. Collaborative cross-silo training has successfully been applied in e.g. healthcare, finance and agriculture. For example, Dayan et al., (2021) showed how federated learning significantly improved future oxygen requirement predictions for COVID-19 patients compared to individual models, while Manoj et al. (2022) demonstrated how a federated model was comparable to a centralized model in predicting soybean crop yield. Two examples from the Norwegian finance sector are initiatives to detect insurance fraud (NCE Finance Innovation, 2020) and to build an anti-money laundering model (Datatilsynet, 2022), where the former project provided NORCE Analytics with firsthand exposure to federated learning in an industry-applied setting due to our leader's key involvement in that initiative.

2.2 Co-creation workshops with fish farmers

To ensure the successful implementation and relevance of our project, we prioritized end user involvement by collaborating with the data science team at Grieg Seafood. Our collaboration involved three meetings, each designed to deepen our understanding of the industry's needs and to refine our approach to using federated machine learning in this domain.

The first meeting was held with the Chief Information Officer (CIO) and a data scientist from Grieg Seafood. This introductory session aimed to establish a mutual understanding of the project's goals and to identify the key areas where Grieg Seafood could contribute. The subsequent two meetings were structured as workshops, focusing on a more detailed exploration of the project objectives and relevant data. Our meetings with Grieg Seafood were also essential to ensure that the federated machine learning framework aim to implement will meet the needs of data scientists and analysts at the fish farmers.

Say something here about data in AquaCloud and data that could/should be included in AquaCloud for better models.

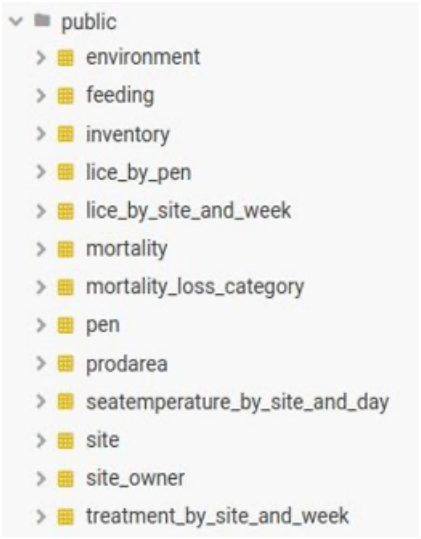
3 Quantitative analysis

To demonstrate that federated learning is applicable in the aquaculture industry, we use data provided by AquaCloud. This data originates from a variety of owners and is available

in standardized tables containing information of inventory, mortality, environment, lice occurrence, and treatment from production sites across Norway. We simulate federated (distributed) learning by treating data from each owner in isolation and compare this model to centralized and individual models. We focus on a simple binary classification problem to highlight the performance of each method, since predicting two classes is the most basic modeling task and classification metrics are normalized and easy to interpret. However, we also present a regression analysis towards the end of the section, to illustrate that federated learning also works for problems more relevant for the industry. Note that we aim to demonstrate the performance of federated training compared to other methods, not to create models with production-grade quality.

3.1 Model features and target variables

We want to model the mortality of fish after receiving lice treatment. Based on the available data, we define the following features:



- Minimum, mean and maximum water temperature
- Number of fish and the average weight
- Average mortality per day
- Days of cohort existence
- Lice occurrence (Adult female, mobile, fixed)
- Average feed amount
- Month and season
- Treatment (Method, scope, type, substance)
- Days since last treatment
- Production area

The values are based on the first day of the week as lice treatment is only provided weekly. Aggregated values use a 14-day window prior to the first day of week. The target variable is the relative mortality in the two weeks after a cohort has received treatment to remove lice. Initially we define the problem as a binary classifier to predict low or high mortality, depending on whether the percentage of dead fish is below or above the typical mortality rate. In the final regression setup, we model the mortality rate directly by using the percentage of dead fish as the target variable.

3.2 Classification with deep learning

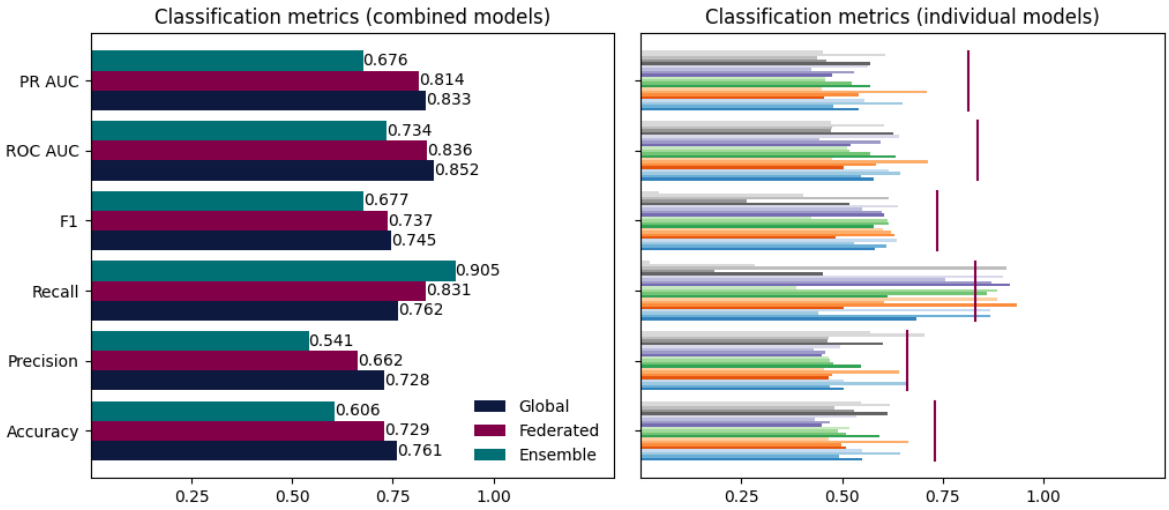
TabNet is a deep neural network tailored to tabular data (Arik & Pfister, 2021). It is both globally and locally explainable and has been shown to have state-of-the-art performance for several problems. Neural networks like TabNet are ideally suited for federated learning, as the fitted model can be represented by a set of coefficients that can easily be combined. TabNet can also be pretrained, which could be utilized to get faster convergence during training. TabNet has previously been used in a federated setting e.g. to classify obstacles, irregularities and pavement types on roads (Lindskog & Prehofer, 2022).

3.2.1 Performance of federated learning

We randomly split the data into a training set (80% of the data) and a test set (remaining data), ensuring that any single site is always in either the training set or the test set, but not in both. We then train models with different access to the data, all using the same architecture:

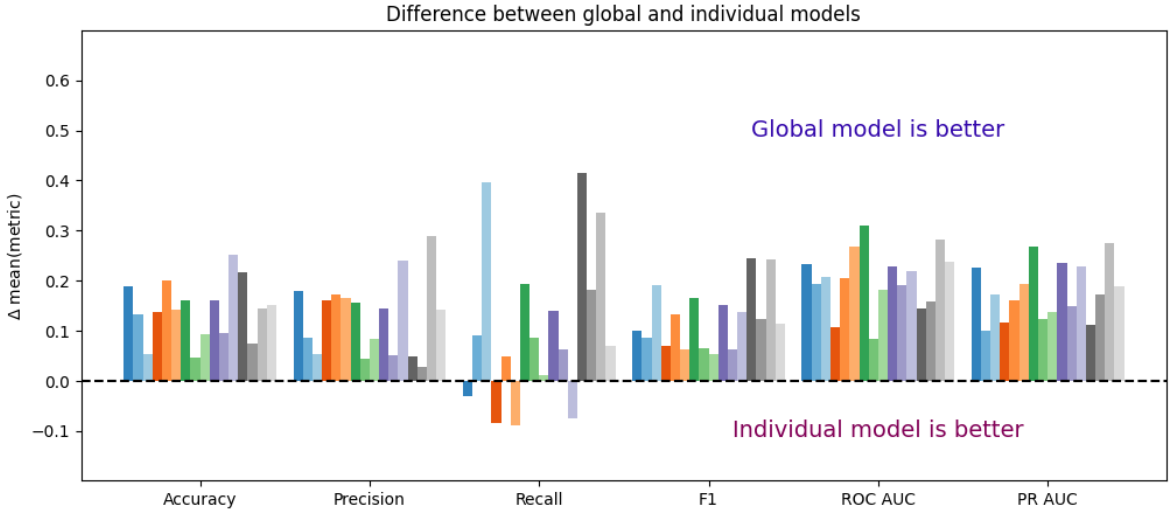
- The global model learns from the centralized training data directly (normal training). We use default settings for most hyperparameters and train for 30 epochs.
- The federated model learns from the distributed training data in isolation and then updates the main model by taking the weighted average of the model coefficients. The weights for the average are based on the number of data points offered by each client. This is called federated averaging and has been shown to be a simple yet powerful method to combine the local models (McMahan et al., 2017). The updated model is returned to the clients to train further on the distributed data. This has been repeated for 30 global epochs.
- Individual models are trained on each owner's data in isolation for 30 epochs. These individual models are also combined into an ensemble model, where the returned score is the weighted average of the individual models' scores. This is a simple form of federated leaning (combining the models that minimizes the local loss at each client in a single iteration, but it has been shown that this strategy could produce a model that is no better than training a model on a single client (Arjevani & Shamir, 2015).

The metrics obtained on the test data are shown in the figure below. The combined models are displayed in the left panel, where the global, federated and ensemble models are shown in deep blue, fuchsia and turquoise, respectively. The global model performs slightly better than the federated model, but the metrics are comparable. The metrics for the ensemble model, on the other hand, are lower. In the figure's right panel, the scores of the individual models are shown in assorted colors. The scores of the federated model are indicated by fuchsia lines for comparison. This shows that this model outperforms the individual models. While some individual models have better precision or recall, they are all worse when both are considered simultaneously (F1). We have used the default hyperparameters when constructing the models, except for a stepwise decay of the learning rate to avoid overfitting, as suggested in the original TabNet-publication. In addition, we have experimented with different hyperparameters, and the results reported in the figure seem robust to some change. However, in the limited scope of this preproject, we are unable evaluate all the model settings possible and it is not trivial to consistently compare the global and federated model using grid search and cross validation.



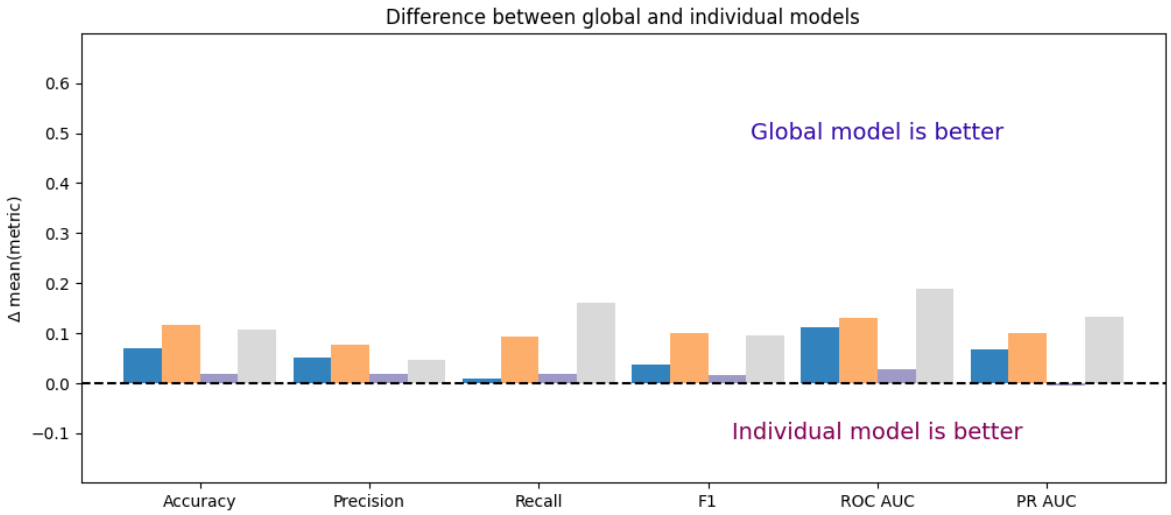
3.2.2 The effect of including more data

The above example uses a centralized dataset for testing. This demonstrates how the models perform in general, but it is more relevant for an owner how a model performs on their own data. To test that data from other owners improves predictions, we split the data into five equal parts, with similar owner distributions and with specific sites always in the same part. Only owners with data in all splits are used. We cycle through the splits, fitting models on four parts and scoring on the remaining part. This is called cross validation and is used to avoid over- or underestimation of a model's performance if a split is randomly over- or underfitted due to limited data for each owner in each split. During training, the global model uses data from all owners while the individual models use data from each owner only. We test on data from each owner separately and display the difference between the mean scores in the figure below, where each color represents one owner.



Positive values indicate that the global model using data from several owners is better, while negative numbers indicate that individual models based on just an owner's individual data is better. The figure clearly illustrates how models with more data are more performant.

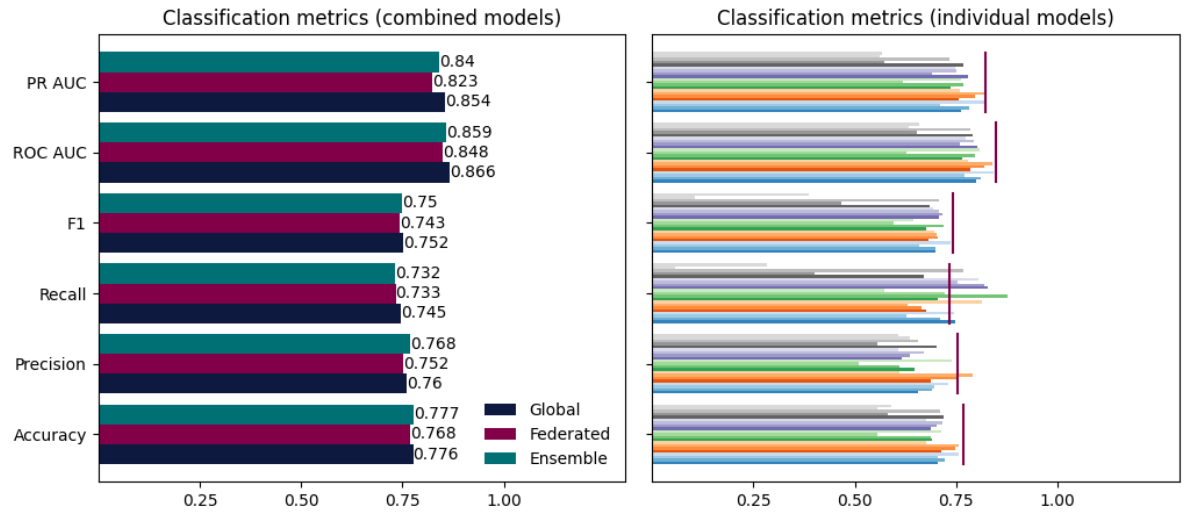
While the above analyses are based on data from all production areas, fish farmers more commonly collaborate with others in their vicinity. To verity that data collaboration also improves predictions in a more limited geographical region, we repeat the above for production area six, which have the most data for this specific problem. The results are given in the below figure and are consistent with the domestic results in that the global model utilizing data from several owners outperforms the individual models.



3.3 Classification with gradient boosting

Gradient boosting is one of the most used techniques to model tabular data and consists of an ensemble of weak learners such as decision trees. XGBoost (Chen & Guestrin, 2016) is a leading implementation of this technique and is widely used in applied models. A challenge in terms of federated learning is that there is no obvious way to combine trees. Instead, federated learning involves adding more trees every time the model is trained at a client. We have trained a global, federated and ensemble model using XGBoost as follows:

- The global model learns from the centralized training data directly. We use a learning rate of 0.1, λ of 1 and tree depth of 5. The model is trained over 50 epochs.
- The federated model cycles through the owners several times during training, using the same tree depth and regularization λ. However, the learning rate is decreased significantly to avoid too strong fitting to the first owners and iterations. We set the learning rate to 0.005. We also need to consider the amount of data available at each client. We introduce pseudo-weighting by allowing one additional tree to fit per 250 observations during local training. Since several trees are added to the model at each client, we only perform 15 global epochs. The order of the clients is randomly selected for each epoch, to avoid some taking precedence.
- Individual models are fitted for each owner in isolation using the same parameters as the global model. Again, we combine these models into an ensemble model by weighted average of their prediction output.

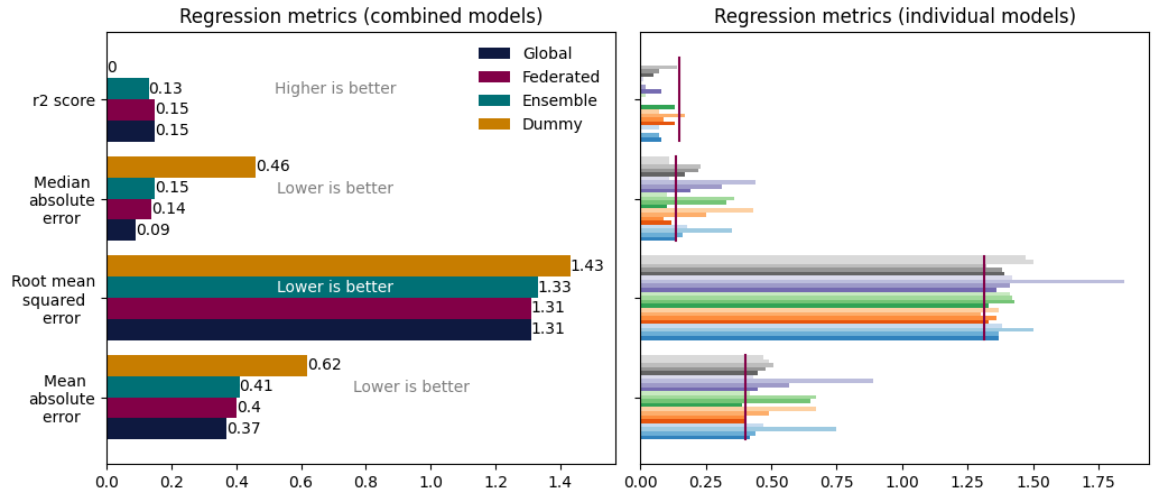


The performance of the different models is found by calculating the classification metrics on the centralized test set also used for the TabNet classifier. The results are shown in the above figure. Again, the metrics are comparable for the global and federated models, with the global model only performing slightly better. The metrics are also close to the metrics obtained by the TabNet classifier. However, using XGBoost, the ensemble model has increased performance, on par with the global model. This is a result of more performant individual models, as seen in the right panel of the figure. The increased performance is likely related to gradient boosting being a less complex method than deep learning, leading to better performance on the limited data available at each client. In addition, the combination of individual models to an ensemble model is more straightforward than the cyclic training scheme used in the federated setup. Regardless, both federated and ensemble models are applicable in distributed learning, and one could simply select the approach that produces the best results in practice.

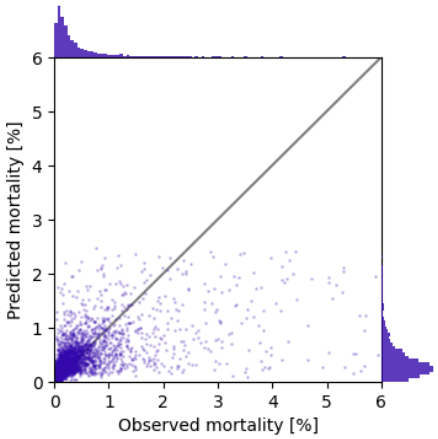
As noted above, the individual models' performances are displayed in the right panel of the figure. Despite the improved predictions for the individual models using XGBoost compared to TabNet, the federated model still outperforms the individual models.

3.4 Regression with gradient boosting

While the simple classifications presented above demonstrate that federated models perform comparably to global models, regression problems are more common in the aquaculture industry. We have therefore repeated the XGBoost experiment, but using the mortality rate directly as the target, thus making it a regression problem. The setup regarding hyperparameters and epochs is the same as in the classification. However, as most regression metrics are not normalized, we add a dummy model for comparison. The dummy model is simply the mean mortality rate of the training data and thus provides the simplest possible meaningful prediction of the mortality rate on the test data. The results are displayed in the figure below.



As seen in the figure, all the combined models beat the dummy model. The federated model is not as good as the global model but similar to the ensemble model. Compared to the individual models, the federated model is equal or better considering most metrics, but ambiguous when considering the median absolute error.



The figure to the left displays the fit of the federated model. It is clear from the figure that the fit is not particularly great, but that some main trends are captured by the model. The predicted distribution resembles the observed distribution as shown outside the main plot, but the model is not able to fit the long tail of the real distribution. This is also the case for the global model (not shown), which indicated that this is related to the lack of explanatory features and not related to the federated learning itself.

3.5 Summary quantitative analysis

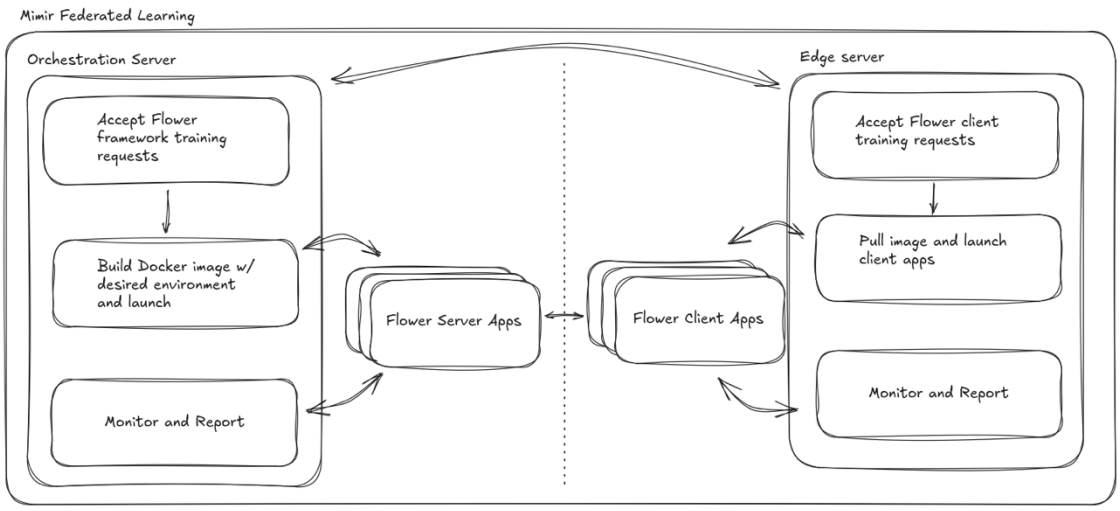
The above results demonstrate that distributed learning has comparable performance to centralized learning when applied to mortality rates in the aquaculture industry, despite differences in the amount of data provided by each owner and potential heterogeneities (non-IID data). This is the case for both federated models (training one model on distributed data) and ensemble models (averaging the prediction from individual models from each owner), and two completely different state-of-the-art algorithms yield comparable results. This gives flexibility regarding algorithms and training- and evaluation strategies. The analysis also shows that improved model predictions are expected when data from several owners are combined, compared to owners training models on their own data in isolation. It is thus highly likely that federated learning will provide value for this industry, offering a robust, flexible, secure and privacy preserving alternative to solve real-world modeling problems.

4 Technical analysis

With an increasing interest in FL, there exists a variety of solutions for training models in a federated manner. We have evaluated many of these from a technical and practical standpoint. The standout disadvantage of using an existing solution is that many of them are focused on bringing a narrowly defined application into a federated learning environment. These frameworks involve a single deployment environment for training the models when the system is in use. For example, with Flower (Beutel et al., 2020), a leading open-source framework in FL, one can deploy client and server apps within a specific environment, for a single participant in the federated context. However, in the MIMIR federated learning framework, we wish to support many participants along with each having their flexibility in determining their own machine learning methodology and libraries of choice to meet their unique needs. The service should be an enhancement and not a hinderance to existing and new processes and certainly should not confine creativity in exploring novel ways to construct such models.

4.1 Repacking an existing solution

With this in mind, we aim to either repackage a framework like Flower under the umbrella of a higher abstraction orchestration server and nanny clients to compensate, or take lessons learned from existing frameworks and services to fully leverage ownership flexibility combined with the practicality of not starting fully from scratch. At a diagram level this might look something like the following:



The positive outcomes of such a design includes benefiting from existing graphical interfaces and other niceties from a Flower deployment which is no small feature set. Add to that the composition of open-source software which has benefited from multiple users reporting bugs and features, yielding a well-rounded product. A consequence of this design is the added complexity of managing these sub-systems requiring things such as dynamic proxying to N number of Flower deployments, to the requirement in-and-of-itself, forcing participants in this project to use Flower as a mechanism for federated learning. In turn,

this would expose all involved to much higher technical debt should Flower become unmaintained or other incompatible directions. Another potential option is looking more into NVIDIA's Flare project, although similar consequences exist here.

4.2 Delivering an owned solution

While this design is tempting, we are aware of the possibility of imposing too many restrictions on the participants while exposing our system to lock-in with regards to the direction or abandonment of the Flower project. This principle applies to any other framework which might also be practically evaluated. Therefore, we find it an attractive alternative in taking implementations of frameworks such as Flower, distributed machine learning libraries like Dask-ML, and others to reconcile a novel platform and service whose direction is under everyone's full control and input. It is our understanding that maximizing flexibility in user modeling means we should not build a federated learning platform which gets in the way. There ought to be a short distance mentally and technically between a user evaluating a model of their choice on their own data, then using that same model on the federated learning platform. This objective should remain clear in our minds moving forward.

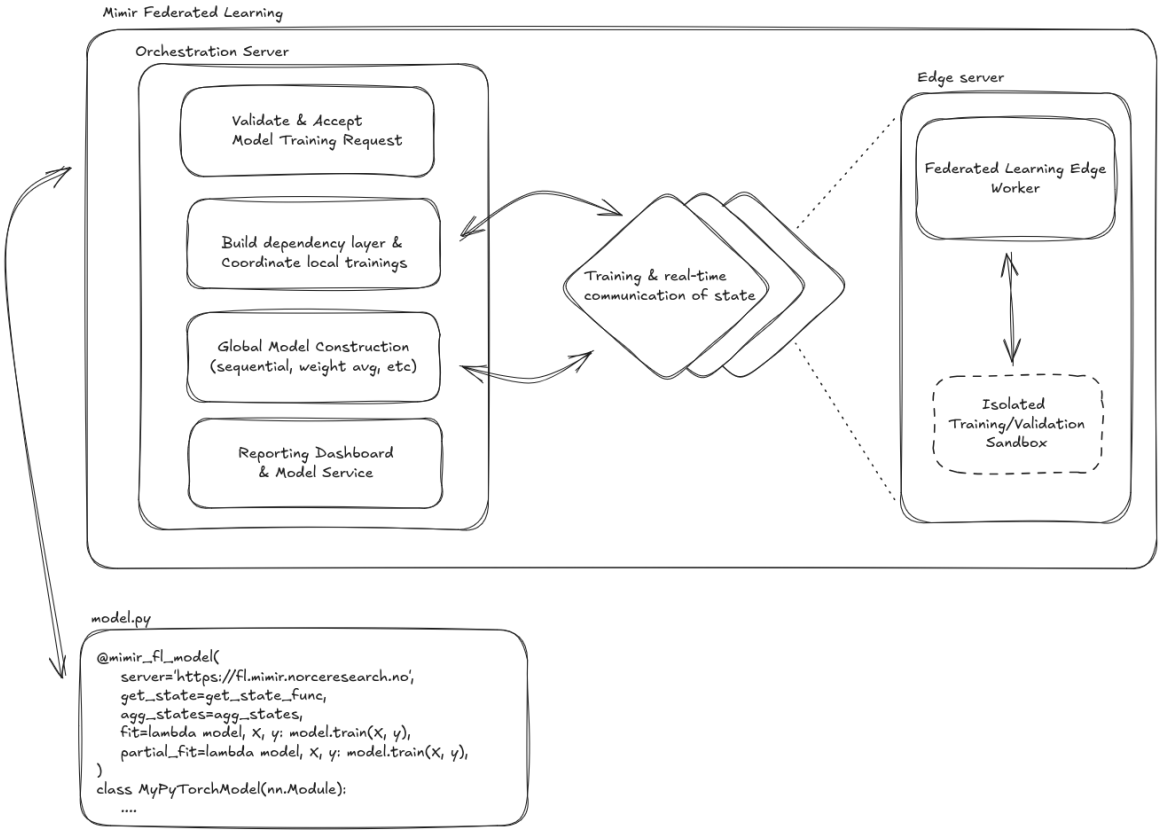
Therefore, the ideal way forward is developing the platform and SDK with a composable and plugin style mindset. If we establish mechanisms to dynamically construct model training environments with precise API definitions to most *any* model which fits into a Docker container, then we will have succeeded in this flexibility and simultaneously opened the doors to security guarantees such as no-network, sandboxed training environments on the client's servers. From this basis, we can collaboratively develop a non-invasive Python library which participants can use with their Python based models to make it compatible with the MIMIR Federated Learning Platform. We can first reuse lessons learned to support popular modeling libraries such as PyTorch, XGBoost and ScikitLearn in a federated capacity, while maintaining a policy of zero cost abstractions. That is, if we maintain the core API at the platform level being a configuration of a docker container and its input/output characteristics, then the user is technically free to implement their federated learning in nearly any language or runtime. Our aim for the Python library is to be non-invasive to existing models while providing a healthy degree of dependency injection to allow participants to be creative in how their model is trained and used. One example could be a parameter to the library for a function taking the current state and returning next training iteration parameters, to broadly defining round-robin vs parallel global training and aggregation; it is not for the platform to decide what is best for the participants but to support them with secure, flexible and reliable federated learning.

A potential option to decouple the model implementation from its federated learning conformity with the platform could look something like this:

```python
@mimir_federated_model(
    server="https://federated-learning.mimir.norceresearch.no",
    client_id=client_id,
    client_secret=client_secret,
    revision="123",
    training_strategy=Strategy.RoundRobin,
    early_stop=early_stop_determination_function,
    get_state=lambda model: model.get_state(),
    set_state=lambda model, state: model.set_state(weights),
    aggregate_states=lambda states: aggregate_states(states),
    evaluate=lambda model, X: model.evaluate(X),
    dependencies=pathlib.Path("requirements.txt"),
)
class MyPyTorchModel(nn.Module):
    ...smart code...
```

A decorator which can transitively detect if it is running on the MIMIR platform could allow easy re-use of the model for local usage in combination with compatibility at the federated learning level. Notably, this encourages participants to include the same model in current modeling processes alongside federated learning deployments of the model for eased clarity in evaluating its organizational value.

Leveraging a dependency injection API approach, we can enable the Data Scientist to have greater control over the model's training behavior in the federated context and ease innovative experimentation. We decouple the federated learning platform from the modeling capabilities while enhancing the cohesion in novel modeling methodology. Furthermore, this broadly opens the door to cross-collaboration for new and novel ways to split and combine weights on different models. If the platform is non-opinionated in its method for aggregation of model states and instead offloads this logic, then we are in a strong position to maximize flexibility and usefulness. Additionally, this results in a slightly simpler architecture with a linear decrease in servers to manage when compared to hosting N number of trainings in comparative frameworks like Flower. There now is a single central server which only needs to offload model aggregations in an ephemeral method, and the same applies for client-side training routines. Further, hosting of trained models on the platform is further simplified with flexibility such as not continuously running a server per model but having a warm caching of models to be served as one approach.



4.3 Physical data transfer, deep dive

The careful reader will have noticed mention of both an isolated sandboxed environment for training and a description of architecture allowing for modeling in any language or with any library. We see that partnership with AquaCloud in this field allows for applying their deep expertise in conforming each individual farmer's data into an agreed upon schema. How do we then physically share this data with the model securely? Here we see that dynamically launching an ingress/egress locked container evades malicious attempts at exporting the data. From there, we can share the data through modern Arrow IPC (Apache Arrow, 2025), either via mounted volumes or through a single port to the edge-server. From there, the model can pass back its state, weights, etc. to the edge-server who in turn does any necessary validations, before reporting to the central server. We would define a specification for such edge-server to local training communications. From here, the Python SDK we develop would abstract away these security aspects and models would be free to accept input data in Arrow or other commonly used format.

4.4 Operational use

We see that development and experimentation iterations should be a quick and smooth process. To move this into an operational perspective, once a satisfactory model has been developed, we aim to provide a reliable and secure interface to its use. To safeguard the

individual owner's data, we acknowledge retaining the model on our platform and exposing it through an API represents the safest way forward. This reduces the opportunity for reverse-engineering data it was trained on or otherwise encoding facets of information into the model. Serving via a REST API allows control over the exposed attack surface area while allowing for simple development of abstraction functionality. A call to a trained model's prediction method continues and upholds the same invariants of a local call. An important side effect of this design is its ability to allow the gradual adoption of federated learning by simplifying the creation of an ensemble with a locally trained model. It is recognized that the local model may benefit from proprietary data not yet available in the federated learning context. This enables incremental adoption of the federated learning platform and therefore a more stable and safe integration of its proposed advantages to each participant.

4.5 Security in model training

Recent research (Hu et al., 2024) has pointed to possibilities of privacy and security issues during training phase as well. One example is a malicious participant purposely trying to corrupt training activities. To counter this attack, all participants can establish and agree upon a base model. This model should be open to all participants and primarily designed for transparency. The base model is then trained and used regularly by the system and its results are available for analysis by those participants. Training and evaluating this model regularly can lend itself to being a proxy to determine from the platform's perspective if something is going wrong at a participant's data, either in good-faith or a bad actor.
An additional attack vector is a malicious participant trying to corrupt the system with a computationally expensive model. This model may try to allocate a lot of memory to freeze the system for example. Here is where the sandboxed environment again comes into play. We can pre-allocate a container with a set amount of memory and CPU resources, taking it offline should it exceed these. Additionally, we may implement timeouts or other restrictions it must conform with.

4.6 Security in inference

already discussed an approach for securing its privacy while models train at the edge, what about when the model is being served? A malicious user has tried to encode this data into the model itself. This is again why we want to withhold passing the trained model itself back to the participant, at least for the first iteration of this platform. Next, we see that a malicious user can try to have correspondence of the data through the limited API of the models being served. Therefore, we can include a vast security net of protocols. This can include:

- Internally verifying randomly sampled sourced data results in different outputs from the model. This reduces the exposure of training models to report "yes/no" responses on information its seen.
- Rate limiting the model being served also reduces the risk of 'streaming' any encoded information, rendering it impractical as an attack vector.

- Reloading models from scratch. Following the above point, doing a complete stateless reload at random of a model from disk prevents the malicious model from maintaining state in communicating data out from the platform.
- Ensuring any pre-condition filters before training always results in both more than a single participant, not heavily biased towards one, and/or some metric therein.
- Injecting slightly varied input data on model inference. It also remains possible to add jitter to input data and keep track of where these artificial input records are placed to remove them post inference. This method can further corrupt an attempt at communicating confidential information.
- Determining a deterministic model. If the model cannot produce the same or similar output given identical input, we can automatically flag the model and take it offline.

If we combine these and more precautions, we can develop a solid degree of trust in the system during hosting of a trained federated model.

4.7 Future opportunities

Looking even further into the opportunities this collaboration provides we see several possibilities which may come to fruition and will briefly name and expand on these here.

4.7.1 Crowd sourcing

By owning the implementation and getting the security boundaries correct, it opens the door to crowd sourcing improvements to the federated learning models. We have seen this occur with success in other areas (NINA, 2025). If we come to the stage where all participants are assured by the security and effectiveness of this project's objective, we could choose to have open competitions for outsiders to participate in creating still better models which internal participants could then choose to make use of. This is beneficial to all involved, especially those lacking either Data Science teams or even the time constraints to focus on modelling iterations.

4.7.2 Model as a service

Additionally, it becomes possible for participants to decide to open one or more of their models to be consumed by other participants as a service. If the platform is hosting the models and exposing training metadata such as cross validation scoring, effectiveness, etc., in a secure way per participant, such participant could then choose to 'open' this model for use by other participants for a cost. Again, we think this may become beneficial to organizations without a mature Data Science team by leveraging models produced by other participants. Of course, participants may have valid reasons to not open their best model, but when upgrading to another model, it could make sense to repurpose the former as another revenue stream while simultaneously helping to improve the aquaculture ecosystem.

5 Conclusions

In this report, we have explored the use of federated learning in the aquaculture domain using data from AquaCloud. The aquaculture industry faces significant challenges that require innovative solutions, particularly when dealing with sensitive data and the need for collaboration between competitors. Federated machine learning offers a promising approach by enabling collaborative data analysis while maintaining data privacy and security.

We have shown that the performance is comparable to the performance of conventional learning using centralized data. Models of similar quality can therefore be build using a federated approach to ensure privacy protection and legal compliance between organizations. We have also demonstrated how data sharing improves predictions for all participants compared to models trained in silos, showcasing the mutual benefits from collaborative analytics in the sector. While we have focused on a specific problem related to mortality rates and lice treatment, we expect similar results for other relevant use cases in the industry.

In the analysis, we used different algorithms and aggregation strategies to demonstrate that the expected flexibility is available for data scientists also in a federated setting. There are also additional measures that could be implemented to further enhance the performance of federated models, like more sophisticated aggregation methods and fine tuning for specific clients which have not been addressed in this report.

Finally, we have discussed pros and cons of using existing or custom frameworks for technical implementation of federated learning in the industry and proposed a flexible, secure, and non-invasive federated learning platform that we believe would significantly enhance the analytical capabilities and possibilities for its users.

6 References

Arik, S. Ö., & Pfister, T. (2021). TabNet: Attentive Interpretable Tabular Learning. Proceedings of the AAAI Conference on Artificial Intelligence, 35(8), 6679-6687. https://doi.org/10.1609/aaai.v35i8.16826. Code: https://github.com/dreamquark-ai/tabnet.

Chen, T. and Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, 13-17 August 2016, 785-794. https://doi.org/10.1145/2939672.2939785. Code: https://xgboost.readthedocs.io/en/stable/index.html.

Beutel, D. J., Topal, T., Mathur, A., Qiu, X., Parcollet, T., de Gusmão, P. P. B., Lane, N. D., & Van Der Smagt, P. (2020). Flower: A Friendly Federated Learning Research Framework. arXiv. https://arxiv.org/abs/2007.14390

Norwegian Institute for Nature Research, (2025). Open competition to identify and read numeric markers on seagulls in the Bergen area. https://urbpop.no/mlkonkurranse/

Apache Arrow Inter Process and Communication specification. (2025). https://arrow.apache.org/docs/format/Columnar.html#format-ipc

Kai Hu, Sheng Gong, Qi Zhang, Chaowen Seng, Min Xia & Shanshan Jiang. (2024). An overview of implementing security and privacy in federated learning https://link.springer.com/article/10.1007/s10462-024-10846-8#Sec27

McMahan, B., Moore, E., Ramage, D., Hampson, S., & y Arcas, B. A. (2017). Communication-efficient learning of deep networks from decentralized data. In Artificial intelligence and statistics. https://doi.org/10.48550/arXiv.1602.05629

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., et al (2021). Advances and open problems in federated learning. Foundations and trends® in machine learning. https://doi.org/10.48550/arXiv.1912.04977

Dayan, I., Roth, H.R., Zhong, A. et al. Federated learning for predicting clinical outcomes in patients with COVID-19. Nat Med 27, 1735–1743 (2021). https://doi.org/10.1038/s41591-021-01506-3

Manoj. T, Makkithaya, K. and Narendra, V. G (2022). A Federated Learning-Based Crop Yield Prediction for Agricultural Production Risk Management, 2022 IEEE Delhi Section Conference (DELCON). https://doi.org/10.1109/DELCON54057.2022.9752836

Wang, G., Dang, C. X. and Zhou, Z. (2019). Measure Contribution of Participants in Federated Learning, 2019 IEEE International Conference on Big Data (Big Data). https://doi.org/10.1109/BigData47090.2019.9006179

Lindskog, W. and Prehofer, C. (2022). Federated Learning for Tabular Data using TabNet: A Vehicular Use-Case, 2022 IEEE 18th International Conference on Intelligent Computer Communication and Processing (ICCP), https://doi.org/10.1109/ICCP56966.2022.10053975

Arjevani, Y., & Shamir, O. (2015). Communication complexity of distributed convex learning and optimization. Advances in neural information processing systems, 28. https://doi.org/10.48550/arXiv.1506.01900

Sommerset I, Wiik-Nielsen J, Moldal T, Oliveira VHS, Svendsen JC, Haukaas A og Brun E. (2024). Norwegian Fish Health Report 2023, Norwegian Veterinary Institute Report, series #8b/2024, published by the Norwegian Veterinary Institute. https://www.vetinst.no/rapporter-og-publikasjoner/rapporter/2024/fishhealthreport-2023

Datatilsynet. (2022). Finterai, sluttrapport: Maskinlæring uten datadeling. https://www.datatilsynet.no/regelverk-og-verktoy/sandkasse-for-kunstig-intelligens/ferdige-prosjekter-og-rapporter/finterai-sluttrapport/

NCE Finance Innovation (2020). Taking the next step in the fight against insurance fraud. https://financeinnovation.no/news/stories/insurance-fraud-project-enters-development-phase

NORCE
Norwegian Research Centre AS
Postboks 22 Nygårdstangen
5838 Bergen, Norway

E-POST post@norceresearch.no
WEB norceresearch.no
TEL. +47 56 10 70 00
ORG NO 919 408 049