EasyChair Preprint

№ 14652

Assessment of Data Driven Techniques for Flow Rate Estimation in Sub Sea Oil Production

Neville D'Souza, Carlos Pfeiffer and Gaurav Mirlekar

September 2, 2024

Assessment of Data-Driven Techniques for Flow Rate Predictions in Sub-sea Oil Production ⋆

Neville Aloysius D'Souza, * **Carlos Pfeiffer,** *
Gaurav Mirlekar *

* *Department of Electrical Engineering, Information Technology and Cybernetics, University of South-eastern Norway (USN), Porsgrunn 3918, Norway (e-mail: gaurav.mirlekar@usn.no).*

Abstract: Accurate measurement of flow rate of the multiphase flow of oil, gas and water from the oil wells, is an important part of the oil and gas industry. This enables the safe operation and proper optimization of the production. With the increasing availability of process data, machine learning algorithms are used to create models for various applications. The application of these algorithms for flow rate estimation provides a more accurate representation of the oil and gas production process. In this paper, two oil wells and ten machine learning algorithms are evaluated. Long Short-Term Memory (LSTM) provides the best results with Mean Absolute Percentage Error of 1.96% for Well 1 and 1.56% for Well 2. In addition, the effects of noise on the models are explored. Median filter with window size of three provides good noise reduction. The uncertainty of the predictions are quantified using 95% confidence intervals in XGBoost model.

Keywords: Machine learning techniques, Data-driven estimations, Uncertainty quantification, Measurement noise

1. INTRODUCTION

The production of oil and gas requires measurements of various process data. This process data is used to ensure optimal production of oil and gas and the safe operation of the production system. One of the most important variables necessary for this is the accurate measurement of oil, gas, and liquid flow rates from the oil wells. Since there is multiphase flow from the oil wells, it is a challenge to obtain the individual flow rates of oil and gas. Typically, a separator is used as shown in Fig 1 to obtain an accurate flow rate of oil, gas and water. Here to measure the individual phases the multiphase mixture are separated physically with a separators. Phase flow meters are used to obtain accurate flow measurements, Bikmukhametov and Jäschke (2020). This process requires a steady state flow from the given oil well. In addition to this, the other oil wells have to be shut down to avoid interference with the results. This is a costly and time consuming process.

To estimate the flow rates without use of separators, Multiphase flow meters (MPFMs) can be used. While MPFMs has many advantages in measurement of multiphase flow, they are very expensive. Also, the accuracy of them gets degraded over time. In addition maintenance of these sensors are important to ensure good working conditions. Oil and gas production systems will already have many sensors installed which monitor certain physical

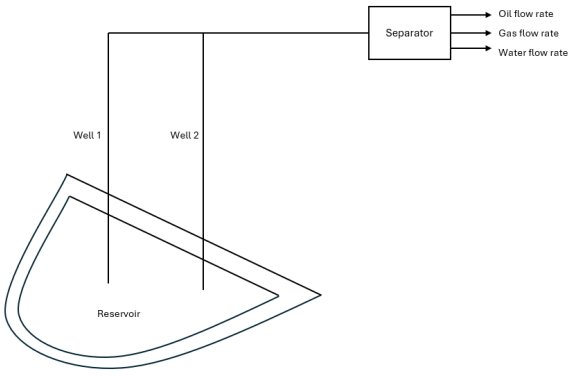

Fig. 1. Example of sub-sea oil production

quantities. The use of data driven modelling (also called machine learning modelling) in the oil and gas industry has been increasing with the availability of and storage of process data. As early as 1993, Qiu and Toral (1993) have used neural networks to estimate the flow rates of multiphase flow. Considerable research is ongoing to improve the application of machine learning models in the oil and gas industry. AL-Qutami et al. (2018) also uses neural networks to estimate phase flow. They used feed forward neural networks with k fold cross validation and early stopping. A more advanced method using LSTM has been used by Andrianov (2018) to forecast forecast the rates for a series of future time instants in addition to reliably estimating the multiphase rates at the current time. For VFM the process data usually collected are:

⋆ We gratefully acknowledge the support from the Research Council of Norway and Equinor ASA through Research Council project "308817 - Digital wells for optimal production and drainage" (Digi-Well). Corresponding author's e-mail: gaurav.mirlekar@usn.no.

- Bottomhole pressure and temperature.
- Wellhead pressure and temperature upstream of the choke.
- Wellhead pressure and temperature downstream of the choke.
- Choke opening values.

1.1 Objective

The main objective here is to assess the machine learning algorithms for flow rate predictions in a sub sea oil production system. The sub-tasks in this are 1) Data collection and preprocessing, 2)Predictions of flow rates of oil, gas using machine learning, 3) Evaluation of measurement noise on machine learning algorithms performance and 4) Quantification of the uncertainty in the predictions.

2. METHODS

2.1 Process Description

For the oil well, the simplified schematic is shown in Fig 2. Through the gas lift choke valve, high-pressurized natural gas is continually injected into the wells annulus in this system, which is mostly utilised to extract lighter crude oils. The injected gas finds its way into tubing at some points located at proper depths and mixes with the multiphase fluid from the reservoir. As a result of this mixing, the density of the fluid in the tubing will be reduced, which means that the flowing pressure losses in the tubing reduce. Consequently, the reservoir pressure will be able to overcome the flowing resistance in the well and push the reservoir fluid to the surface. Each well has its own inflow characteristics. A graphical representation of this project is shown in Fig 3.

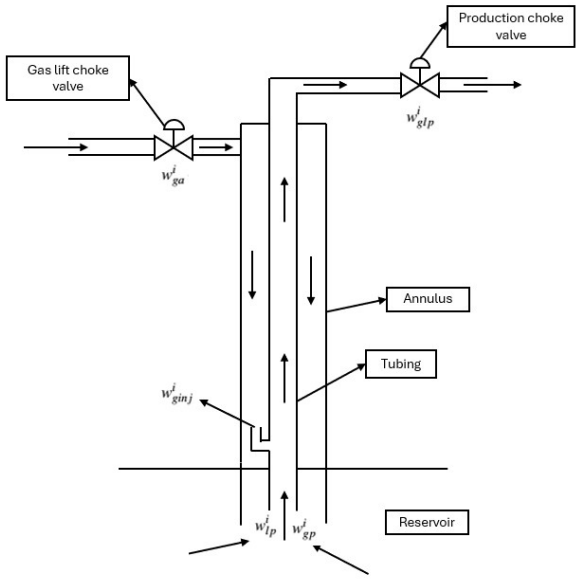

Fig. 2. Single oil well schematic

2.2 Machine learning algorithms

A brief description of the machine learning algoirthms used in this paper are describe here

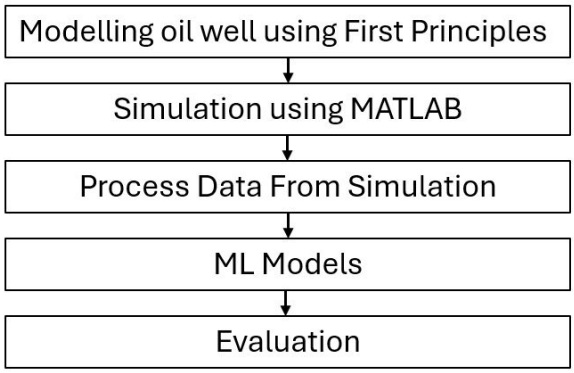

Fig. 3. Process Flow of the Project

Multivariate Linear Regression Linear Regression is the simplest machine learning algorithm. It makes a prediction by simply computing a weighted sum of the input features, plus a constant called the bias term, as shown in equation (1).

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n \tag{1}$$

Where,
$\hat{y}$ - predicted value,
n - number of features,
x_i - ith feature value,
θ_j - jth model parameter,
θ_0 - bias term.

This can be modified to output multiple $\hat{y}$ values. Multivariate linear regression is a statistical technique that models the linear relationship between multiple independent variables and a single dependent variable. It extends simple linear regression by allowing for the inclusion of more than one predictor variable. The goal is to find the linear equation that best predicts the dependent variable based on the independent variables Zangl et al. (2014).

k-Nearest Neighbors Regression The k-nearest neighbors (kNN) algorithm is a non-parametric, supervised learning method used for classification and regression tasks. It works by identifying the k closest training examples to a given data point and assigning a class or value based on the majority vote or average of those neighbors. kNN is a versatile algorithm that can handle both numerical and categorical data without making assumptions about the underlying data distribution. It is commonly used in applications like recommendation systems, pattern recognition, and anomaly detection. The choice of k is important, as lower values can lead to overfitting while higher values may cause underfitting, Géron (2023).

Support Vector Regression Support Vector Regression (SVR) is a non parametric technique that uses kernel functions to estimate a function from a set of training data. The goal is to find a function f(x) that deviates from the target values y by no more than ε, while being as flat as possible. This is achieved by solving a convex optimization problem that minimizes the norm of w, subject to the constraint that the regression errors are within ε. SVR can handle high-dimensional data and

nonlinear relationships by implicitly mapping the input data into a higher-dimensional feature space using kernel functions. Unlike other regression models that try to minimize the error between the real and predicted values, SVR tries to fit the best line within a threshold value (distance between the hyperplane and boundary line). The data points on either side of the hyperplane that are closest to the hyperplane are called Support Vectors, which are used to predict the output Xu et al. (2012). SVR has several advantages, such as being robust to outliers, having excellent generalization capability, and easy implementation. However, it is not suitable for large datasets, and its performance may degrade when the number of features exceeds the number of training samples Smola and Schölkopf (2004).

Decision Tree Regression A decision tree algorithm is a supervised machine learning technique used for both classification and regression tasks. It constructs a tree-like model of decisions based on the data's attributes. The process starts at the root node and splits the data into subsets using the most significant attribute based on selection criteria like information gain or Gini impurity. Each internal node of the tree represents a "test" on an attribute, each branch represents the outcome of that test, and each leaf node represents a class label or a continuous outcome. The paths from root to leaf represent classification rules or regression paths. Decision trees handle both numerical and categorical data and are intuitive, as they mimic human decision-making processes. They are particularly useful in scenarios where relationships between parameters are non-linear or complex. However, decision trees can suffer from overfitting, especially with very complex trees. Techniques such as pruning are used to remove parts of the tree that do not provide additional power in order to reduce overfitting and improve the model's generalizability. Decision trees are foundational elements in more complex algorithms like Random Forests and boosting methods, enhancing their stability and accuracy Dayev et al. (2023).

Gradient Boosting Regression Gradient Boosting Regression is a powerful machine learning algorithm that combines multiple weak models to form a strong learner. It is particularly effective for regression problems where the goal is to predict continuous values. The algorithm works by iteratively training decision trees on the residuals of previous predictions, which are the differences between the actual and predicted values. Each tree is trained to minimize the error of the previous tree, and the learning rate determines the contribution of each tree to the final prediction. The process begins with an initial guess, typically the mean of the target variable. Then, at each iteration, a new tree is trained to predict the residuals from the previous tree. The residuals are the differences between the actual and predicted values. The new tree is added to the previous trees, and the process is repeated until a stopping criterion is reached, such as a maximum number of trees or a minimum improvement in the model's performance. The final prediction is the sum of the predictions from all the trees, weighted by their learning rates. This approach allows the algorithm to capture complex relationships between the input variables and the target variable, making it highly effective for regression problems Kniazev et al. (2023).

XGBoost Regression XGBoost is a powerful algorithm for building supervised regression models. It was developed by Chen and Guestrin (2016). It is an implementation of gradient boosting that is designed to be highly efficient and scalable. The algorithm is particularly effective for regression problems where the goal is to predict continuous or real values. XGBoost is based on the concept of ensemble learning, where multiple base learners are trained and combined to produce a single prediction. The core components of XGBoost for regression include the objective function, base learners, and regularization. The objective function is responsible for defining the loss function and the regularization term. The base learners are the individual models that are trained and combined to produce the final prediction. Regularization is used to prevent overfitting by penalizing complex models. XGBoost uses a unique approach to building regression trees. Each tree starts with a single leaf and all residuals go into that leaf. The algorithm then calculates a similarity score for this leaf based on the residuals. The similarity score is used to determine how to split the data into two groups. This process is repeated recursively until a stopping criterion is reached. XGBoost is widely used in various applications due to its high accuracy and efficiency. It is particularly effective for large datasets and can be easily integrated with other tools and packages such as scikit-learn and Apache Spark.

PC Regression Principal component regression (PCR) is a regression analysis technique that combines principal component analysis (PCA) and linear regression. The key idea behind PCR is to first perform PCA on the predictor variables to obtain a set of uncorrelated principal components, and then use these principal components as the new predictors in a linear regression model, instead of the original variables. The advantages of PCR are that it can help address issues like multicollinearity and high dimensionality in the predictor variables. By using a subset of the principal components, PCR can reduce the number of predictors in the regression model, which can improve the model's interpretability and generalization performance. However, PCR does not perform feature selection, as each principal component is a linear combination of all the original predictors. While PCR can be a useful technique, it has some limitations. It relies on the assumption that the directions of maximum variance in the predictor variables are also the most predictive of the response variable, which is not always the case. Additionally, PCR can result in information loss, as it discards some of the principal components during the regression step Bello et al. (2014).

PLS Regression PLS regression is a powerful statistical technique that is particularly useful for analyzing high-dimensional data with many predictor variables. The key idea behind PLS regression is to find a set of latent components (linear combinations of the original predictors) that maximize the covariance between the predictors and the response variable. Unlike traditional linear regression, PLS does not require the predictors to be orthogonal or the number of predictors to be less than the number of observations. PLS regression works by iteratively extracting latent components that explain as much of the covariance between the predictors and response as possible. The resulting PLS model provides both dimension reduction and

regression coefficients, allowing for accurate prediction of the response variable from the original high-dimensional predictors. PLS regression has several advantages over other regression methods, including its ability to handle multicollinearity, its robustness to noise, and its suitability for datasets with more predictors than observations. As a result, PLS is a widely used technique in fields such as chemometrics, bioinformatics, and marketing research Boulesteix and Strimmer (2006).

MLP Neural network A Multilayer Perceptron Neural Network (MLPNN) is a type of artificial neural network that consists of multiple layers of interconnected nodes, or neurons. Unlike a single-layer perceptron, which can only learn linearly separable patterns, an MLP can learn more complex, non-linear relationships in data. The key components of an MLP are the input layer, one or more hidden layers, and an output layer. The input layer receives the data, which is then passed through the hidden layers, where the network learns to represent the data in a more abstract way. Each hidden layer applies a non-linear activation function to the weighted sum of its inputs, allowing the network to learn complex patterns. The final output layer produces the predicted result.MLPs are trained using a supervised learning algorithm, typically back propagation, which adjusts the weights of the connections between neurons to minimize the error between the predicted and actual outputs. This iterative process allows the MLP to learn the underlying structure of the data and make accurate predictions on new, unseen data.

LSTM Long Short-Term Memory (LSTM) is a type of recurrent neural network designed to address the vanishing gradient problem in traditional RNNs. The key feature of LSTMs is their memory cell, which can selectively retain or discard information as it flows through the network. LSTMs have three gates that control the flow of information: the input gate, the forget gate, and the output gate. The input gate decides what new information from the current input and previous output should be added to the memory cell. The forget gate determines what information from the previous memory cell should be retained or forgotten. The output gate controls what information from the current memory cell and input should be used to produce the output. This gating mechanism allows LSTMs to learn long-term dependencies in sequential data, making them well-suited for tasks like language modeling, machine translation, speech recognition, and time series forecasting. LSTMs have been widely adopted and have significantly advanced the state-of-the-art in many sequence-to-sequence learning problems.

Data driven VFM (also called machine learning VFM) is the method where a model of the oil and gas production system is created using the available sensor data. Here in depth domain knowledge about the process is not necessary to create a model. A typical schematic for a sub-sea oil and gas production systems which used data driven VFM is shown in Fig 4. Broadly, the steps involved are as follows:

(1) Data collection.
(2) Data pre processing.
(3) Model development.

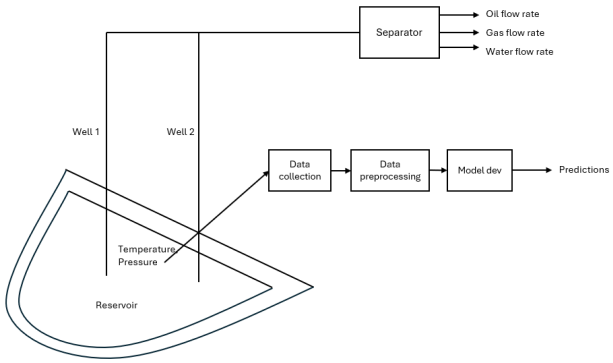

Fig. 4. Data driven VFM

(4) Predictions of flow rates.
(5) Data reconciliation.

2.3 Data collection

The first step to creating a data driven model is the collection of relevant data. In Virtual Flow Metering systems, information is transmitted from wells and processing facilities and this includes sensor readings. This data may be wireless transmitted using IoT systems or through physical communication wires. It can involve different communication protocols to ensure proper transmission of data. Historical data from the same or analogous fields may also be used as a calibrating data set for fine-tuning the model. Generally, the data collected tends to be unclean, contaminated, and may have missing values, outliers and redundant inputs. Here the data is obtained from the oil well model described by Janatian et al. (2022). Using the equations described in the paper, an open loop simulation of the oil well is obtained. For the oil well 1 and 2, 5762 samples are obtained. These are split in 70% train and 30% test sets.

2.4 Data pre processing

Data filtering, where the removal of noise from raw data is performed is part of this step. There exist many filters that can be deployed to clean the raw data. In addition outlier detection, correcting missing values can be included. Preprocessing can also involve data transformation, which might yield new insights about the information the data contains. Feature engineering is the common term for this technique. Since the oil and gas production process is time dependent, care should be taken when splitting the data for training. Time series split from Sci-kit learn library is used for this. Two splits are created as shown in Fig 5. Using this, data leakage in the training stage can be avoided. A standard scaler is used on the inputs of the train dataset. It involves re-scaling each feature such that it has a standard deviation of 1 and a mean of 0. This is necessary since the inputs have different ranges. It also help to reduce training time in certain algorithms like SVR and neural networks.

2.5 Model development

In order to create a model, an algorithm that can map input features to output (target) variables must be developed. The mapping process, also known as training or

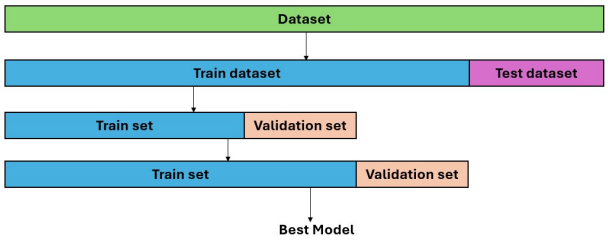

Fig. 5. Time series splitting

learning, involves the algorithm modifying the parameters so that it can precisely estimate the desired variables. Depending on the algorithm being used, the parameters must be changed. The weights that connect the neurons in a neural network, for example, are the parameters. In regression trees, on the other hand, the parameter may be the tree depth. Reducing the difference between the algorithm's predicted values and the actual (measured) values to minimise a cost function, is how training is accomplished. Mean squared error (MSE) is usually used as a cost function to solve regression problems such as Virtual Flow Metering. Here ten algorithms are used to create models for Well 1 and Well 2. LSTM Regression, Multivariate Linear Regression, KNN Regression, Decision Tree Regression, Gradient Boosting Regression, XGBoost Regression, Principal Component Regression, Partial Least Squares Regression, and MLP Neural Network Regression are used.

2.6 Predictions of flow rates

Once the training and validation for the model is completed, the model is tested on unseen data. New predictions from this data are noted and the effectiveness of model can be determined. For oil and gas flow rate predictions the commonly used performance metric is the Mean absolute percentage error (MAPE). With this the performance across various algorithms can be compared. It is easy to interpret and can be used across different input data scales. MAPE can be found by:

$$\text{MAPE}(y, \hat{y}) = \frac{1}{n_{\text{samples}}} \sum_{i=0}^{n_{\text{samples}}-1} \frac{|y_i - \hat{y}_i|}{\max(\epsilon, |y_i|)} .100 \quad (2)$$

where $\hat{y}$ is the predicted value of the ith sample, y_i is is the corresponding true value, n_{samples} is number of samples, ϵ is an arbitrary small yet strictly positive number to avoid undefined results when y is zero.

2.7 Data reconciliation

An optimization algorithm adjusts the model parameters, for instance, flow rates, choke discharge coefficient, gas and water fractions, and friction and heat transfer coefficients such that the model outputs match the validated measured data being constrained to process conditions, for instance, the material balances. The reconciliation procedure in virtual flow metering systems is frequently expressed in the constrained least squares form.

3. RESULTS AND DISCUSSIONS

3.1 Predictions using algorithms

For Well 1 the Figures 6 shows the results of each algorithm. Figure 7 shows the predictions for Well 2.

For LSTM model early stopping is used to prevent overfitting. This is implemented in Tensorflow. For well 1, 32 memory cells are used. For well 2, 40 memory cells were used. The 'adam' optimizer with loss function of mean squared error is used for training both models. A linear activation unit is used in the output layer.

In the Multi-layer perceptron neural network model, for well 1, 2 hidden layers are used, 'identity' activation function, an optimizer in the family of quasi-Newton methods (lbfgs solver) is used, L2 regularization of 0.0005 and the maximum number of iterations is 1000. For well 2 only 'logistic' activation function is changed and other hyper-parameters are same as well 1.

In the Multi variate linear regression, for well 1 and well 2 the intercept is calculated. The copy_X parameter is true, which means the input features are copied, not overwritten.

In the Support vector regression model, for well 1, radial basis function is used as kernel, with kernel coefficient of 0.1, and regularization of 1 is used. For well 2, radial basis function is used as kernel, with kernel coefficient of 0.01, and regularization of 10 is used.

In the K nearest neighbors model, for well 1 and well 2, 8 neighbors were used. For well 1, the power parameter for the Minkowski metric is used. For well 2, the power parameter for the euclidean distance metric is used. These metrics are used in distance computation.

For the Decision tree model, for well 1, the maximum depth of the tree is 1, the minimum number of samples required to be at a leaf node is 1, the minimum number of samples required to split an internal node is 2 and the number of features to consider when looking for the best split is the sqrt of the number of features. For well 2, the maximum depth of the tree is 10, the minimum number of samples required to be at a leaf node is 1, the minimum number of samples required to split an internal node is 10 and the number of features to consider when looking for the best split is the log2 of the number of features.

For Gradient boosting model, for well 1 and well 2, the loss function used is squared error for regression. The maximum depth of the individual regression estimators is 3. For well 1, the learning rate is 0.1, and for well 2 the learning rate is 0.05. With this algorithm it is possible to find the importance of each input feature. Here P_{wh}, the wellhead pressure shows the maximum effect.

A more efficient and faster implementation of gradient boost, XGBoost model Zheng et al. (2022) is developed next. Here for well 1, the squared error is used as objective function, 300 estimators, maximum depth of 7 and learning rate of 0.01 is used. For well 2, the number of estimators are reduced to 100, maximum depth of 3 and learning rate of 0.1.

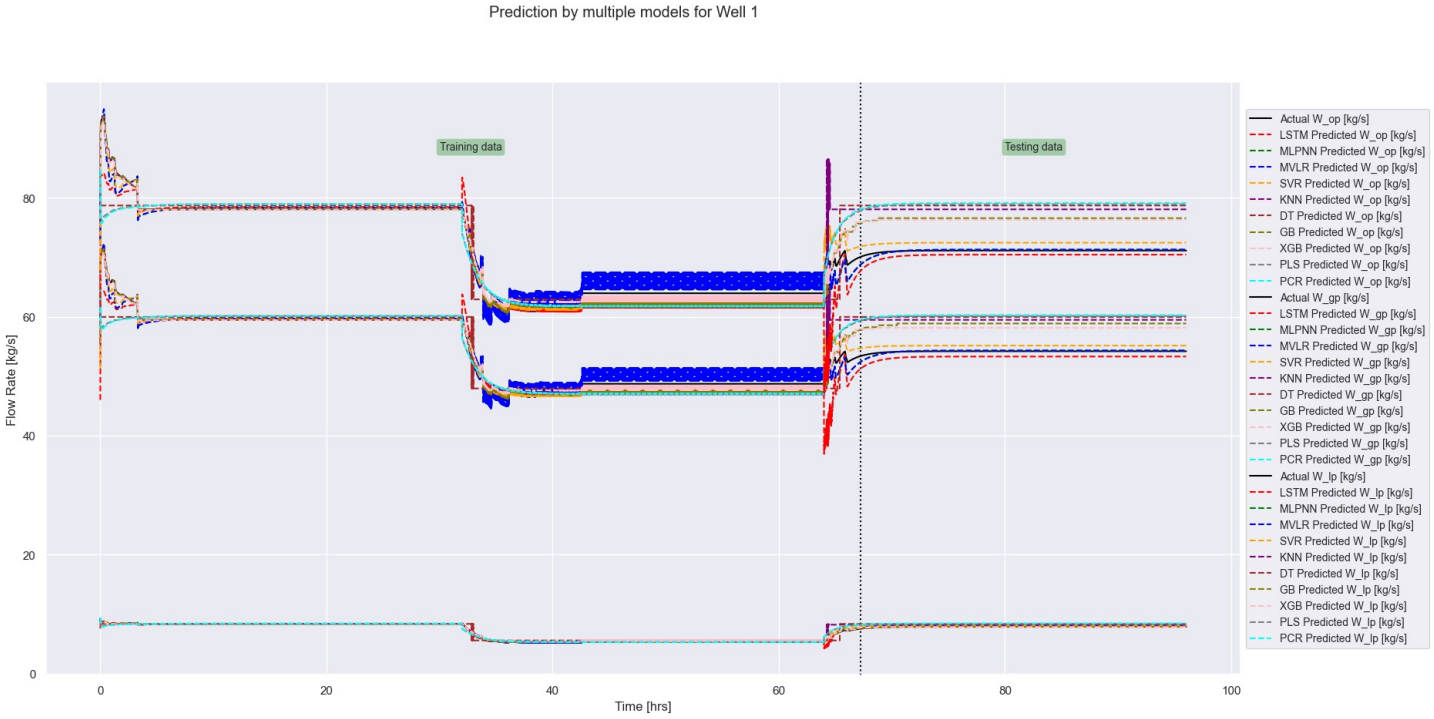

Fig. 6. Flow rate Predictions for Well 1

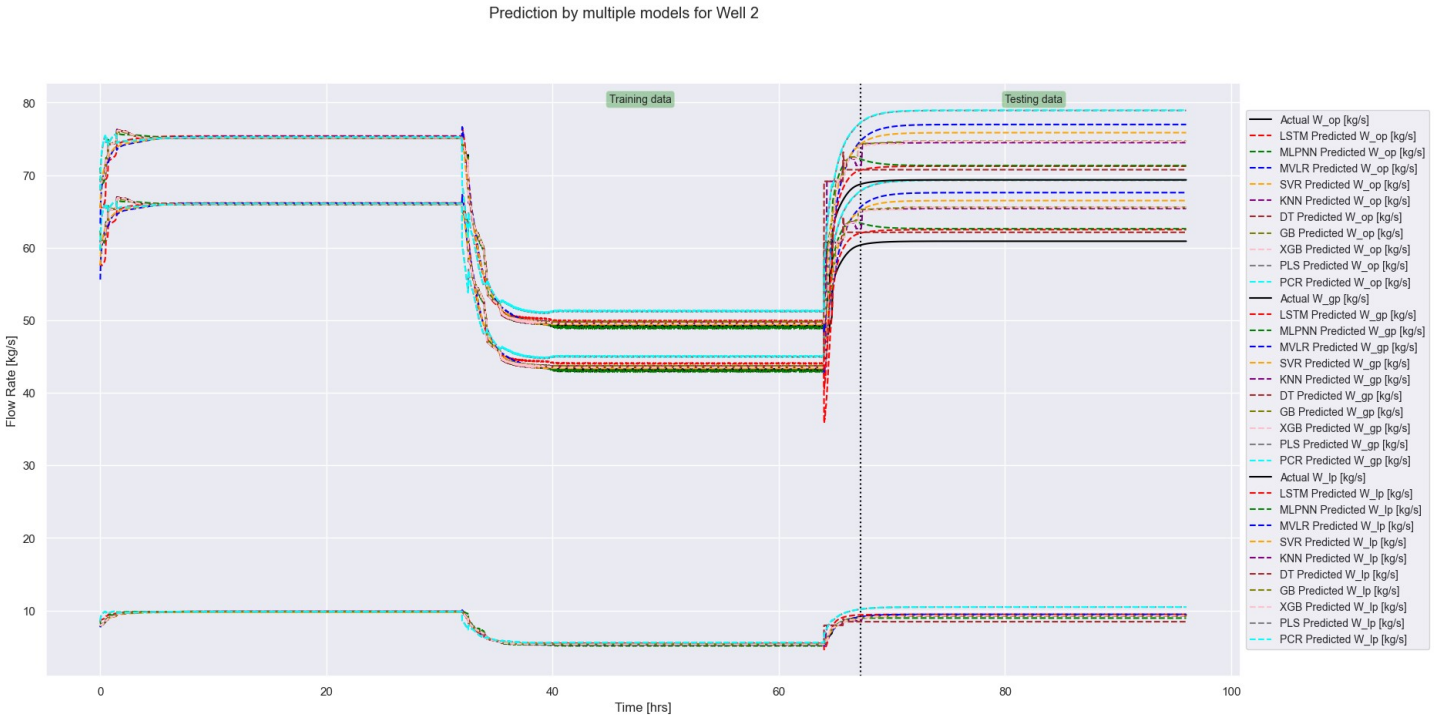

Fig. 7. Flow rate Predictions for Well 2

For the PLS and PCR models, the number of component used in 1 in both wells. These models are useful when dimension reduction of input feature space is needed.

The MAPE for each model is shown in Table 1

Table 1. MAPE for Well 1 and Well 2

Algorithm	Well 1 (%)	Well 2 (%)
LSTM	1.96	1.53
MLP NN	2.43	5.49
MV Linear Regression	2.14	7.57
SVR	5.04	4.31
KNN	8.05	5.41
Decision Tree	9.26	5.43
Gradient Boost	4.95	5.55
XGBoost	4.23	5.56
PLS	9.54	7.57
PCR	9.52	16.69

The LSTM model produces the best results. The disadvantages of using this is the training time is longer. Also to find the proper parameters is a time consuming process. It is observed that for each well the hyper-parameters has to be tuned. GridSearchCV helps with this, but it is still a complicated process. For the algorithms that are generally used for classification tasks like SVM, kNN, some modification is required to enabling its use for regression. Many of these algorithms including linear regression, and tree based, require modification to predict multiple outputs. With modifications it is possible to get the results, but the downside is the hyperparameter tuning becomes more complex. Neural networks and the LSTM model can be made more complex, giving better results. This takes more time and computation power. For finding the best hyper-parameters multiple runs are required. Since the programs were executed on a laptop, these take more time. For decrease in computation time a sample size of 5762 was used. If more samples were used in the modelling the results would probably be much better.

3.2 Effects of noise

The effect of random errors is tested on three machine learning models: XGBoost, MLP NN and LSTM. Impulse noise introduces sudden jumps or falls in the data values, simulating real-world data with occasional spikes at random locations. First a noise sample of 3% is created. The values in the sample are uniformly distributed between 20% of the minimum value of the column and 30% of the maximum value of the column. This ensures that the noise added is relative to the range of the data in the column. The noise is randomly distributed across the column and added to the 3 input features. The 3 algorithms are trained and tested. Here the figures are shown of only Well 1, since the effects are can be similarly observed in Well 2. Figures 8 and 9 show the effect of impulse noise on LSTM, MLP NN and XGBoost models respectively. To solve the problem of impulse noise, there are many filters that can be used. For example Median filter, Order statistic filters, and so on. Here the Median Filter is used to reduce the impulse noise. SciPy is used, which has a median filter function that is well-suited for removing impulse noise, as it replaces each data point with the median of the neighboring data points within a specified window size. The results of the

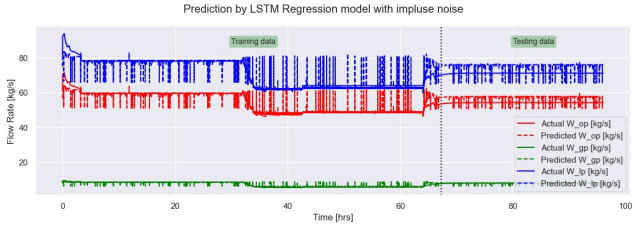

Fig. 8. Effect of Impulse noise on LSTM model

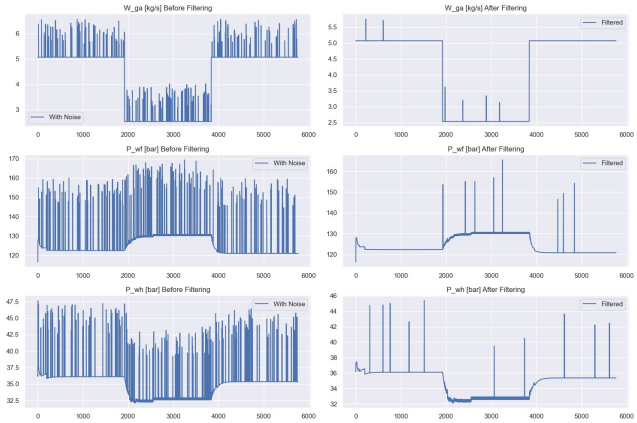

Fig. 9. Median noise filter

median filter is shown in Fig 9. A window size of 3 is used. Each data point is replaced with the median of itself and its two neighbors. Most of the impulses are filtered out. The prediction accuracy of the 3 models is improved. Tables 2 and 3 shows the effect of impulse noise on MAPE of the 3 models.

Table 2. Impulse noise effects

Well no.	LSTM (%)	MLP NN (%)	XGBoost (%)
Well 1	5.98	8.76	7.51
Well 2	4.67	5.13	5.77

Table 3. Median Filter effects

Well no.	LSTM (%)	MLP NN (%)	XGBoost (%)
Well 1	1.87	4.97	6.47
Well 2	2.86	5.51	5.29

3.3 Uncertainty quantification

Uncertainty refers to a state of limited knowledge or information, where it is impossible to precisely describe an existing state, future outcome, or multiple possible outcomes. There are two main types of uncertainty: Aleatory and Epistemic Uncertainty Pelz et al. (2021). There are many methods to quantify the uncertainty in predictions for machine learning models. Some of them are: Confidence intervals, Quantile regression, Bootstrapping, Ensemble methods and Bayesian optimization.

Using XGBoost the confidence intervals can be added. For other algorithms like LSTM it is a more complicated process. Fig 10 and Fig 11 shows the confidence intervals of 95% for XGBoost model for well 1 and well 2.

Fig. 10. Confidence intervals for XGBoost model (Well 1)

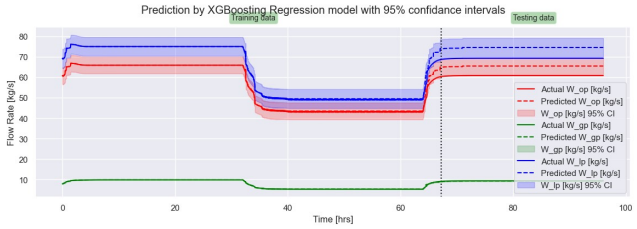

Fig. 11. Confidence intervals for XGBoost model (Well 2)

4. CONCLUSION

Applying machine learning for flow rate estimation in oil and gas productions is a complex process. From the data collection to uncertainty quantification, considerable work has to be done to obtain useful results. The applicability of the results depends on the situation. Long short-term memory (LSTM) provides the best results with Mean absolute percentage error of 1.96% for Well 1 and 1.56% for Well 2. It may be best to use the predictions from the models as a backup for more robust systems. Since each well has its own characteristics, they must be modeled individually. In addition more process data would probably improve the accuracy of the flow rate predictions.

More filters can be tested to remove measurement noise. Different methods of uncertainty quantification can also be tested. The outlier detection and correction can be added in future. Unsupervised techniques like Local Outlier Factor, Isolation Forest, Kernel Density Estimation can be tested. Data reconciliation can also be added. Here the process flow diagram is necessary, the constraints of the each well are also needed.

REFERENCES

AL-Qutami, T.A., Ibrahim, R., Ismail, I., and Ishak, M.A. (2018). Virtual multiphase flow metering using diverse neural network ensemble and adaptive simulated annealing. *Expert Systems with Applications*. doi:10.1016/j.eswa.2017.10.014.

Andrianov, N. (2018). A machine learning approach for virtual flow metering and forecasting. *IFAC-PapersOnLine*. doi:10.1016/j.ifacol.2018.06.376.

Bello, O., Ade-Jacob, S., and Yuan, K. (2014). Development of hybrid intelligent system for virtual flow metering in production wells. In *All Days*, SPE–167880–MS. SPE. doi:10.2118/167880-MS.

Bikmukhametov, T. and Jäschke, J. (2020). First principles and machine learning virtual flow metering: A literature review. *Journal of Petroleum Science and Engineering*, 184, 106487. doi:10.1016/j.petrol.2019.106487.

Boulesteix, A.L. and Strimmer, K. (2006). Partial least squares: a versatile tool for the analysis of high-dimensional genomic data. *Briefings in Bioinformatics*, 8(1), 32–44. doi:10.1093/bib/bbl016.

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*. doi:10.1145/2939672.2939785. URL `https://arxiv.org/abs/1603.02754`.

Dayev, Z., Yetilmezsoy, K., Sihag, P., Bahramian, M., and Kıyan, E. (2023). Modeling of the mass flow rate of natural gas flow stream using genetic/decision tree/kernel-based data-intelligent approaches. *Flow Measurement and Instrumentation*, 90, 102331. doi:10.1016/j.flowmeasinst.2023.102331.

Géron, A. (2023). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: concepts, tools, and techniques to build intelligent systems*. O'Reilly, Beijing Boston Farnham Sebastopol Tokyo, third edition edition.

Janatian, N., Jayamanne, K., and Sharma, R. (2022). Model based control and analysis of gas lifted oil field for optimal operation. In *The First SIMS EUROSIM Conference on Modelling and Simulation, SIMS EUROSIM 2021, and 62nd International Conference of Scandinavian Simulation Society, SIMS 2021, September 21-23, Virtual Conference, Finland*, 241–246. doi:10.3384/ecp21185241.

Kniazev, V., Erofeev, A., Demidov, A., Orlov, D., and Koroteev, D. (2023). Advanced well stimulation selection with gradient boosting. *Geoenergy Science and Engineering*, 228, 212026.

Pelz, P.F., Pfetsch, M.E., Kersting, S., Kohler, M., Matei, A., Melz, T., Platz, R., Schaeffner, M., and Ulbrich, S. (2021). *Mastering Uncertainty in Mechanical Engineering*. Springer International Publishing. doi:10.1007/978-3-030-78354-9_2. URL `https://doi.org/10.1007/978-3-030-78354-9_2`.

Qiu, J. and Toral, H. (1993). Three-phase flow-rate measurement by pressure transducers. In *All Days*, SPE–26567–MS. SPE. doi:10.2118/26567-MS.

Smola, A.J. and Schölkopf, B. (2004). A tutorial on support vector regression. *Statistics and Computing*, 14(3), 199–222. doi:10.1023/B:STCO.0000035301.49549.88.

Xu, L., Zhou, W., and Li, X. (2012). Wet gas flow modeling for a vertically mounted venturi meter. *Measurement Science and Technology*, 23(4), 045301. doi:10.1088/0957-0233/23/4/045301.

Zangl, G., Hermann, R., and Schweiger, C. (2014). Comparison of methods for stochastic multiphase flow rate estimation. In *All Days*, SPE–170866–MS. doi:10.2118/170866-MS.

Zheng, H., Mirlekar, G., and Nord, L.O. (2022). Agent-based and stochastic optimization incorporated with machine learning for simulation of postcombustion co2 capture process. *Processes*, 10(12). doi:10.3390/pr10122727. URL `https://www.mdpi.com/2227-9717/10/12/2727`.

Appendix A. PROGRAM CODES

The Matlab codes for the simulator and the python machine learning code can be accessed here:
https://github.com/dsouzaneville/FMH606-1-Masters-Thesis