Global Flow Measurement Workshop
25 - 27 October 2022

Technical Paper

Machine Learning based gross error estimation for allocation systems

Daniel Dobos[1], Tien Thanh Nguyen[1], John McCall[1], Allan Wilson[2], Helen Corbett[2], Phil Stockton[2]
[1] National Subsea Centre, Robert Gordon University, Aberdeen, UK
[2]Accord ESL, Aberdeen, UK

1 INTRODUCTION

Gross error detection (GED) is an integral step in operating all allocation systems to ensure unbiased allocation results, enable early detection of instrument faults, and reduce or avoid costly maintenance or remediation work. Many systems apply simple rules such as checking for stuck metered values or using mass balances and reconciliation factors. Statistical gross error (GE) tests that employ physical conservation laws, knowledge of the processing facility, and measurement uncertainties are more rigorous alternatives to detect GEs. Such conventional methods however can only be as effective as the mathematical models they are based on, make assumptions about the distribution of statistical noise, and typically do not employ historical information through a time series of measurement data. These approaches can also lead to excessive false positive rates generating unnecessary work and potentially discontinued use.

This paper demonstrates the effectiveness of a variety of supervised machine learning (ML) models on synthetic data to estimate the GE. A ML method learns the correlation between different state variables through a diverse dataset, containing possible GE types such as biases and leaks, and after training, the model can estimate and correct live measurements. Performance improvements, of reduced false positive and increased error identification rates, are demonstrated by running ML methods side-by-side with conventional statistical GE tests on multiple benchmark systems, as well as on a case study using realistic measurement data from an offshore processing facility.

This paper is organised as follows. Section 2 introduces the main concepts for GED and gives a short introduction to supervised ML algorithms. Section 3 discusses the potential ways of generating synthetic data to train a ML method. Section 4 demonstrates the possibility of using an ML-based GE detector, using a simple linear system. In this section, we rank the different ML algorithms based on their performances to get the top 5 methods. In section 5, we apply these top 5 methods to a dataset corresponding to a realistic system. Section 6 provides some conclusions.

2 CONCEPTS

2.1 Random Errors and Gross Errors

Random errors are present in every measurement. They can be caused by fluctuations in measurement conditions like temperature changes, electrical noise, or mechanical shocks. Most of the time this random error manifests as Gaussian noise about the unknown true value. The mean and standard deviation of the sensor measurements can be determined from relevant applicable standards and manufacturer's data sheets, calculated from a set of steady-state measurements, or by using sound engineering judgment.

In the context of allocation systems, random errors in measurements cause slight imbalances in physical quantities like mass and energy conservation. Data reconciliation is an established mathematical and statistical technique to obtain the best estimates of the true measurement values subject to physical conservation laws and has been the subject of several papers at this workshop [1],[2],[3].

Data reconciliation and therefore allocation systems only provide valid results when no GE is present in the input measured data. GEs can occur when a meter is for example poorly calibrated or maintained, damaged, or degrades over time, for example through fouling. In the production environment, GE can be seen as a sudden change in the measurement, a slight drift over time, or a constant bias in the measurement. Operators monitor measurement quality through regular inspections, measurement audits, and online diagnostics checks, but all this effort can't guarantee the quality of the data.

Examples of GEs are shown in Figure 1, where open dots represent true values and filled dots represent actual measured values [4].

Figure 1 - Examples of different GE types.

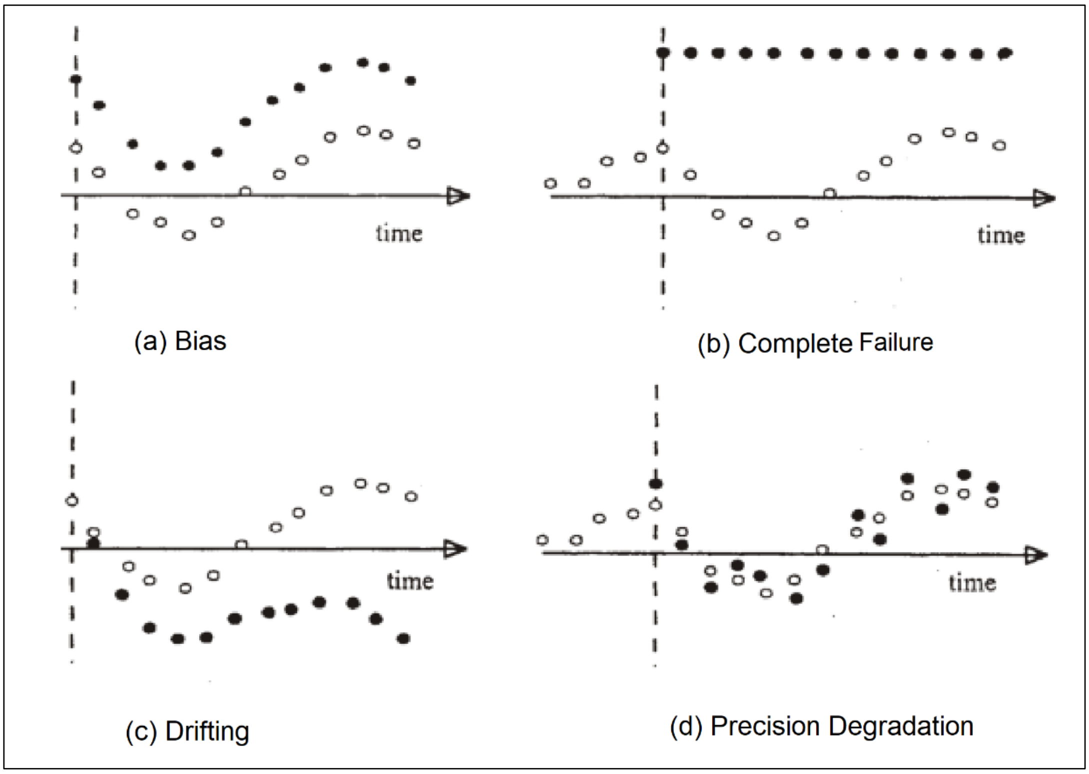


Figure 1(a) shows the effect of a measurement bias. That means the sensor has an offset or bias, so it systematically over- or under-reads the true value. This can be corrected if the value of the offset can be established.

Another type of GE is the complete failure of a meter as shown in Figure 1 (b). In this case, the sensor reports a fixed value for the measurement and the measurement is unusable. This can be identified by the system operator.

Drifting, where the sensor measurement slowly deviates from the true value over time is depicted in Figure 1 (c). This may be dependent on another variable or variables rather than being simply time-related.

If the sensor is not maintained or used for a long time the precision of it can degrade compared to the original causing GE, and this is represented in Figure 1 (d).

2.2 Statistical Gross Error Tests

Two statistical tests will be used for comparison to the ML methods in section 4.

- Global Test;
- Generalised Likelihood Ratio (GLR) Test.

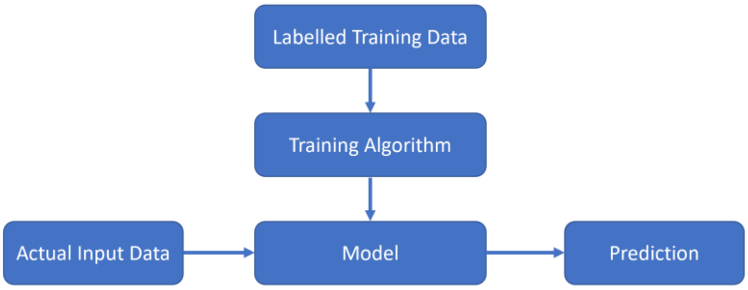
Both tests have been widely studied in data reconciliation literature [4],[5]. The Global test is capable of testing for the presence of a GE but does not indicate the location or the magnitude of the error without further effort. In addition to identifying the presence of GE, the GLR test can in theory identify the erroneous measurement and estimate the bias.

GED tests mentioned here are applicable to systems subject to linear constraints in the process variables like mass-balance or energy conservation laws and can be used when the system operates in a steady-state condition.

2.3 Supervised Machine Learning

ML has already found numerous applications in many different fields like software engineering and medical imaging, and chemical engineering. Supervised learning is a subset of ML, where a labelled dataset called training set is given. Our goal is to exploit the relationships between the data samples and their labels in the training set for decision-making and predictions. On that basis, we apply the model derived from the training set to obtain accurate predictions for unseen data samples. In supervised learning, we set apart two modes namely regression and classification. Regression is the task of predicting a continuous quantity, while classification methods aim to predict discrete class labels. In this paper, we investigate the application of regression methods for the GED problem. Concerning the training set, it is important to note that the variable that we want to predict is called the dependent variable. The variable we use to predict the other variable's value is called the independent variable.

Figure 2 – Process diagram of supervised learning principle



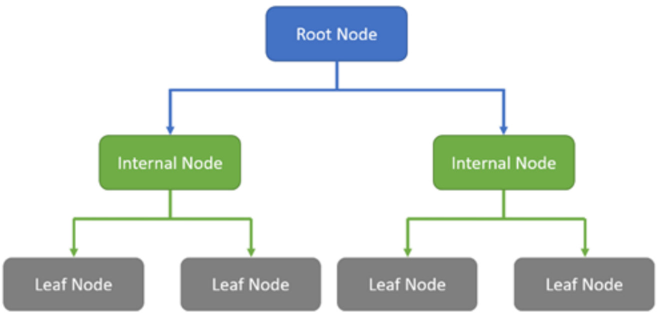
2.4 Linear Regression

Linear Regression is one of the simplest ML methods which provides easy-to-understand mathematical formulas for prediction. Linear regression is a well-established statistical procedure used in many fields effectively, such as biological environmental sciences, business, and social sciences.[6] . Linear regression is a linear approach for modelling the relationship between a dependent variable and one or more explanatory independent variables. The parameters of the linear function are estimated by using a fitting strategy on the training data. Linear regression is one of the most popular methods in both study and practical applications because of its easy fitting strategy and its easy determination of the statistical properties of the resulting estimators.

2.5 Tree Based Regression

Tree-based regression methods are well-established ML algorithms known for their simplicity and efficiency. The algorithm creates a tree structure that contains a root, multiple branches, and multiple leaf nodes based on the training data using a splitting strategy such as Gini Index or Entropy [7]. The decision tree model generates a set of simple rules and uses them to predict through the repetitive process of splitting. An unseen sample will travel on the tree by referencing the rule set until it reaches a leaf node. The prediction is conducted based on the information of training samples on that leaf node.

Figure 3 – Diagram of a decision tree



2.6 Ensemble learning regression

It is recognised that applying different ML algorithms to a dataset could produce different results. There is no single algorithm that performs the best on all datasets. Besides, each algorithm uses a different approach to understand the relationship between samples and their labels in the training data. Thus, combining several algorithms in an ensemble can obtain better results than using a single algorithm as a result of diverse learning approaches. In the past years, there were a number of ensemble methods introduced to solve the regression problem. Several state-of-the-art methods can be mentioned like Bagging [8], Random Forest [9], and Boosting [10].

2.7 Performance metrics

The performance of different methods can be compared by using a number of performance metrics. In this study, we used two common performance metrics proposed by Narasimhan and Mah [5]:

- Overall Power (OP)
- Average number of type I errors (AVTI) also known as the false alarm rate

They are defined as the following:

$$OP = \frac{Number\ of\ gross\ errors\ correctly\ identified}{Number\ of\ gross\ errors\ simulated} \qquad 2\text{-}1$$

$$AVTI = \frac{Number\ of\ gross\ errors\ wrongly\ identified}{Number\ of\ simulation\ trials} \qquad 2\text{-}2$$

Estimation of GEs (gross errors) was also considered for positive and negative biases. The estimates were evaluated using mean squared error (MSE) as a performance metric, which is also commonly used for regression methods [11].

$$MSE = \frac{1}{N}\sum_{i=0}^{N}\left(\delta - \hat{\delta}\right)^2 \qquad 2\text{-}3$$

Where δ is the ground truth for the stream bias and $\hat{\delta}$ is the estimated bias prediction.

3 SYNTHETIC DATA GENERATION

In this study, we aim to investigate the application of regression methods to detect GEs. That means both the measurement data and its associated ground truth information of GEs are required to train a regression model. In this section, we introduced a procedure to generate training data and testing data for a system. The data generation procedure is given as follows:

- Step 1: Define the system steady state condition, with a set of mean flow values, and variance which satisfies the mass balance equation.

- Step 2: Random noise is added to the mean flow measurement value. The random noise is obtained by sampling a normal distribution with a known mean and variance. This step is run N times to create N records. These records are called base cases i.e. no GE is present in each case.

- Step 3: GE is introduced in a stream by adding or subtracting a random amount from the base case. That random amount is obtained by sampling a uniform distribution between -γ% to $+\gamma$%. There are N records generated in this way.

- Step 4: Repeating step 3 for all the streams, and saving all the generated data into a training set. By using this way, the number of records in the training data will be $N + m \times N$ in which m is the number of streams in the system.

For the base case we used the following equation:

$$x = \bar{x} + N(\mu , \sigma^2) \quad \text{3-1}$$

in which x is the measurement value in a base case, $N(\mu , \sigma^2)$ is the value of random error with mean μ and variance σ^2. The GE is added to the base case value by:

$$x_n = x + x * U(-\gamma\%, +\gamma\%) \quad \text{3-2}$$

4 FEASIBILITY WITH SIMPLIFIED LINEAR ALLOCATION SYSTEM

4.1 Introduction

A simplified system proposed by Edson Cordeiro do Valle was used for testing out the different regression methods [12]. The benchmark introduced contains a realistic measurement uncertainty with a mean value to simulate one steady state condition. That necessary information will be used to generate training and testing datasets for the different ML methods. The generated dataset will be used for comparing ML methods with different statistical tests for GED according to the performance metrics introduced in the second section.

4.2 System and Datasets

We chose a linear process system (see Figure 4) from the benchmark tests for GED [12]. This system was taken from an industrial process flow diagram of water balance measurement [13] in which the flow rates were collected from measurements, mass balance calculations, datasheets of the plant, and empirical opinions of the plant operators. The necessary mean values of the streams for simulating a steady state condition are known in which the flow measurements are given in [kg/s]. The standard deviation for each stream is also given. The true flow rates and standard deviations are presented in Table 1.

Figure 4 – Process flow diagram of the water treatment process [12].

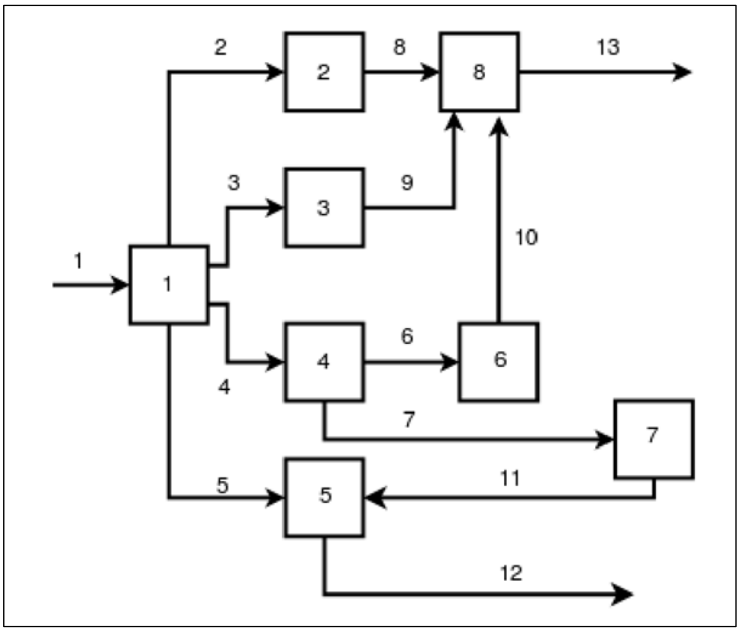


Table 1 – True flow rate and standard deviations associated with 13 streams of the water balance measurements

Stream	True Flow Rate ($kg\ s^{-1}$)	Standard Deviation
1	28.000	0.275
2	5.000	0.050
3	5.000	0.172
4	7.000	0.145
5	11.000	0.372
6	4.000	0.127
7	3.000	0.136
8	5.000	0.045
9	5.000	0.095
10	4.000	0.073
11	3.000	0.064
12	14.000	0.147
13	14.000	0.131

4.3 Model Training

As mentioned before, we need to generate training and testing data for the experimental system to train and evaluate the experimental ML methods. Based on the synthetic data generation procedure mentioned in Section 3, we first created N=1000 non-GE records i.e. only containing random errors with normal distribution with zero mean and a corresponding standard deviation taken from Table 1. We generate a GE by sampling from a uniform distribution between - γ %=-25% and + γ %=+25% and then add them to each stream. The iteration was run through all streams to create a training set of 1000+13*1000 records in which the first 1000 records have no GE while 13000 next records contain one GE on a unique stream.

To generate the testing data, we followed the studies in the literature in which a fixed magnitude of GE is added to the base cases. The fixed magnitude of GE will support directly comparing the performance of regression methods to those of other existing methods in the literature. Three testing datasets were created with a fixed percentage of the GE on one stream of positive 10% GE on one stream, positive 15% GE on one stream, and negative 10% gross on one stream. We used the 10% and 15% GE as suggested by Rendy and Mavrovouniotis [14], as well as a -10% to compare how the algorithms perform when the sensors are under reading. The testing set also has 1000+13*1000 records in which the first 1000 records have no GE while 13000 records have one GE on a stream.

Both training and testing data were generated using MATLAB 2021b using the random numbers function with normal distribution. The dataset for the training can be found in the supplementary material. It is noted that we only generated scenarios where the GE was introduced to only one stream in this experiment. The scenarios where two or more streams have GEs will be conducted in our future study.

4.4 ML methods

We selected 10 well-known regression methods to investigate their performance on the problem of GED (see Table 2). To train these methods, we used the scikit-learn library with default parameters. Scikit-learn is an open-source library written in Python programming language for solving ML problems (https://scikit-learn.org). The training times and the size of the trained model were also recorded to get a better understanding of the viability of the methods. The training was done on a PC with an Intel Core i7-11800H, 8 cores processor at 2.3GHz, and 16 GB of RAM.

It is recognised that Bagging and Random Forest require much higher training time because each of them is an ensemble model including 100 regression trees that take time for training. Meanwhile, Bayesian Ridge and K Nearest Neighbors are the two fastest models for training. For model size, Bagging and Random Forest require more than 600MB to store the trained models while the trained models of Linear Regression and Bayesian Ridge are only 6KB and 26KB, respectively.

Table 2 – List of ML methods evaluated in the Industrial Water Plant analysis

	Model Name	Training Time (s)	Model Size [KB]
1	Linear Regression (Ordinary least squares)	0.72	6
2	Bayesian ridge regression	0.06	26
3	K Nearest Neighbors Regression	0.44	41000
4	Decision Tree Regression	1.35	1900
5	Random Forest Regressor	419.3	611000
6	Extra Tree Regressor	64.18	1140000
7	Bagging Regressor	422.98	611000
8	Ada Boost Regressor	28.84	3300
9	Gradient Boosting Regressor	114.11	7400
10	Histogram-based Gradient Boosting Regression	5.36	3800

4.5 Gross Error Detection in ML models

A regression method estimates GE for each stream while conventional methods usually estimate a binary response to the null hypothesis i.e. no GE is present on the measurement [4]. To get a comparable result to the statistical tests, we test the null hypothesis by using $Z - score$ (also called a standard score) which is given as follows [15]:

$$Z = \frac{x_i - \mu}{\sigma} \tag{4-1}$$

where Z is the $Z - score$, x_i is the measurement, μ is the mean value of the measurement, which could be regarded as the real value, and σ is the standard deviation of the measurement (see Table 1). However, the real value is not available for new measurements. For the calculation, we assumed that the predicted GE ($\hat{\delta}_i$) is correct so we could calculate the mean value of the measurement with the following equation:

$$\mu = x_i - \hat{\delta}_i \tag{4-2}$$

Changing the Z-score equation by substituting μ

$$Z = \frac{x_i - (x_i - \hat{\delta}_i)}{\sigma} \rightarrow Z = \frac{\hat{\delta}_i}{\sigma} \tag{4-3}$$

If the absolute value of the Z-score is higher than a critical score associated with a confidence level, we reject the null hypothesis and GE is detected. In this study, we used two critical Z-values:

- $Z_{critical}$ for 95% confidence interval $\cong$ 1.9600
- $Z_{critical}$ for 99% confidence interval $\cong$ 2.5759

4.6 Results

Figure 5 shows the AVTI of the 10 regression models and 2 statistical tests on the experimental data. In this study, the AVTI was calculated with a dataset containing 1000 containing no GE, and 13000 records where GE was present on one stream. It is noted that the ideal GE test result would have an AVTI of zero and OP of 1. With 95% of confidence level, AdaBoost is the poorest in which 636 samples get incorrect GE detection by this method. GLR, Linear Regression, and Bayesian Ridge also obtained high false alarm rates with the values of 49.7%, 42.8%, and 42.8% respectively. K Nearest Neighbors achieved the best result of AVTI with only 3.5%, which means only 35 samples got incorrect predictions among 1000 non-GE samples.

When the confidence level increases to 99%, the AVTIs of all methods reduce by about a half compared to the AVTI of 95% of the confidence level. K Nearest Neighbors still is the best method for AVTI in which only 22 non-GE samples get incorrect GE detection.

Table A1-A6 in the appendix show the OP of all methods on 3 test datasets with 10%, 15%, and -10% of bias with 95% and 99% of confidence levels. With the stream bias of 10%, the GLR test obtained the best result of OP (92.24%), followed by AdaBoost (92.1%) and Decision Tree (91.53%). K Nearest Neighbors which is the best concerning AVTI now is the second poorest method concerning OP (77.89%). All of the other regression methods obtained more than 80% of OP.

When the confidence level increases to 99%, 7 regression methods obtained 84%-85% of OP which is better than the GLR test. Global Test is the worst method in the experiment for both 95% and 99% of confidence levels. All methods obtained high performance with more than 94% of OP except Global Test (87.15%) for 15% bias and 8 methods except Global Test, Linear Regression, Bayesian Ridge, and K Nearest Neighbors obtain OP of more than 90%. It is noted that some regression models outperform GLR and Global Test with 99% of confidence level. The outperformance of regression methods in the experiment demonstrates the potential of this approach for the problem of GE detection.

Figure 5 – AVTI results

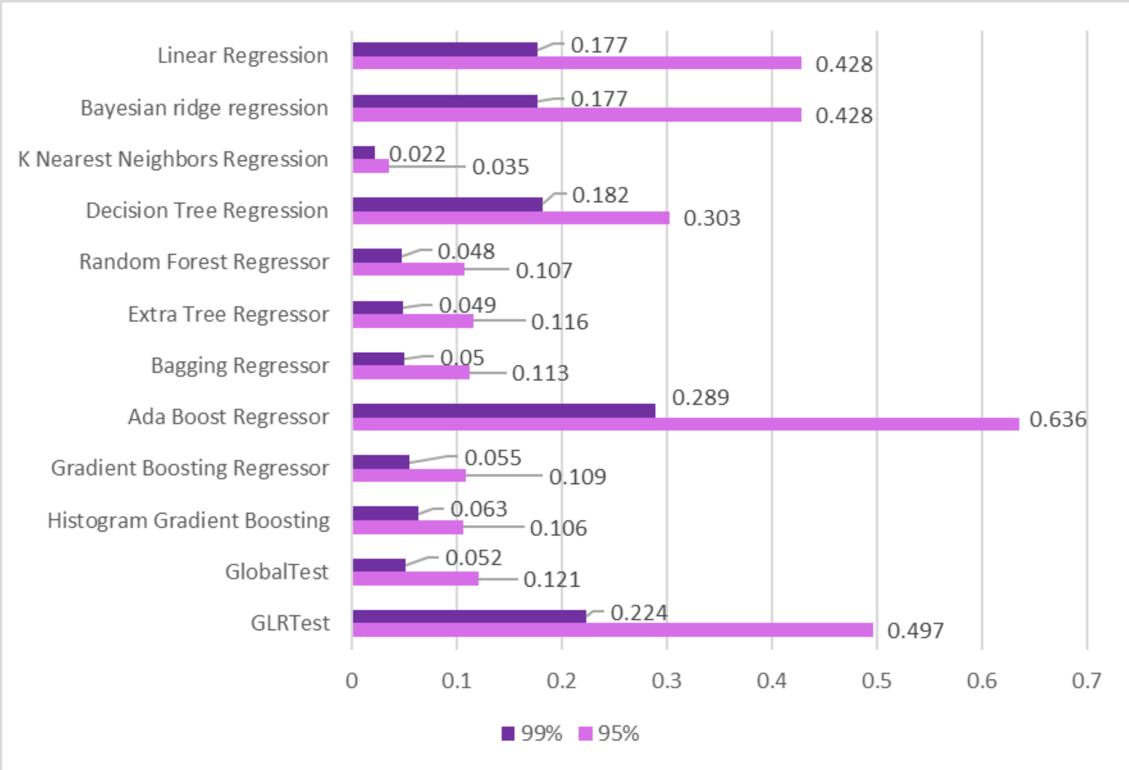
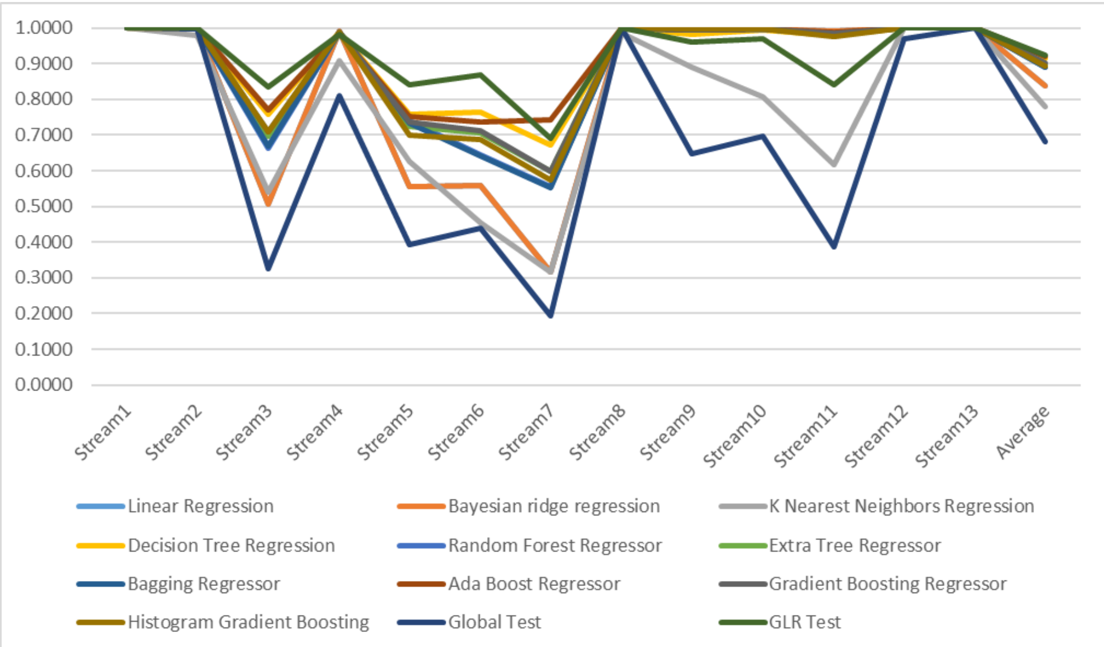


Figure 6 – OP – Stream bias of 10% with, 95% confidence level



4.7 Method ranking

We compared the MSE of 10 regression methods on each stream in Figure 7. While OP and AVTI are suitable performance metrics for the comparison with statistical tests, MSE provides an effective metric to evaluate the GE magnitude estimation of a regression method. For simplicity, we only showed the MSE of all methods on the dataset with 10% of bias. K Nearest Neighbors obtained the smallest value of MSE on all streams except the 5th stream where Extra Tree obtained the smallest MSE with 0.0249. On this stream, Decision Tree achieved the highest value of MSE.

Table 3 ranks the different ML methods based on the average MSE performance. It is observed that K Nearest Neighbors achieved the best result among all methods while Linear Regression obtained the poorest result. We chose the top five methods namely K Nearest Neighbors, Bagging, Random Forest, Extra Trees, and Gradient Boosting to conduct the experiments on a realistic system.

Figure 7 – MSE values of different ML methods on the data with 10% overread on a single stream

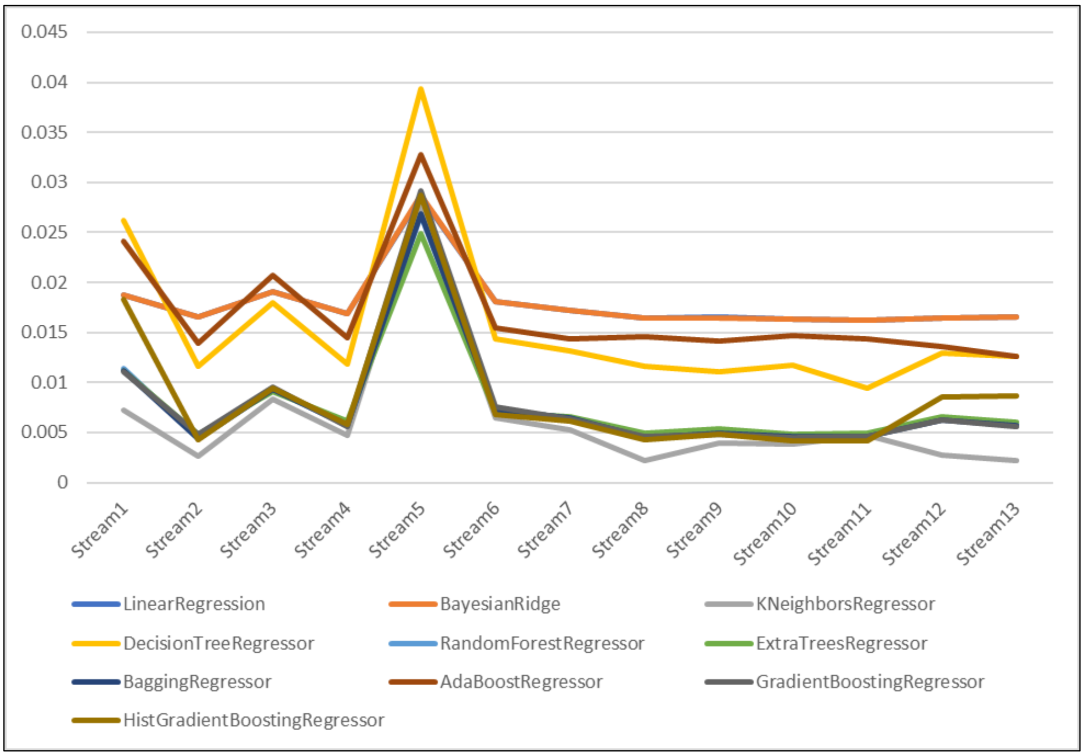


Table 3 – Average MSE values of different ML methods in ascending order

Method	Average MSE
K Nearest Neighbors Regression	0.006309
Bagging Regressor	0.007843
Random Forest Regressor	0.007846
Extra Tree Regressor	0.007884
Gradient Boosting Regressor	0.008030
Histogram Gradient Boosting	0.008773
Decision Tree Regression	0.015683
Ada Boost Regressor	0.016905
Bayesian ridge regression	0.017977
Linear Regression	0.017982

5 REALISTIC SYSTEM

5.1 System Description

We experimented with a realistic system to further understand the effectiveness of ML methods for GED. The experimental system is based on a realistic hydrocarbon processing system presented in Figure 8. For the system described in this section, we used realistic values for the feed streams, while the rest of the flow measurements are calculated from these feed streams by using a simulation package. Thus, the flow values represent a true steady state condition.

Table 4 shows the true flow rate and standard deviations associated with 35 streams in the realistic system. These values will be used to simulate training data and testing data for the experiment. It is observed that the true flow rates and standard deviations of this system are much higher than those of the water treatment process in Section 4.

5.2 Gross error tests and results

We generated training and testing data corresponding to the real system to train and evaluate the 5 selected ML methods. For training data, we generate GEs by sampling from a uniform distribution between - γ %=-25% and + γ %=+25% and then add them to each stream. For testing data, 4 datasets were created in which each dataset contains 1000 records. The test datasets were created as follows:

- 10% overreading on Camelot separator oil
- 10% overreading on Merlin lift gas
- 10% overreading on Merlin separator gas
- No GE

Figure 8 – Flow diagram of the realistic system

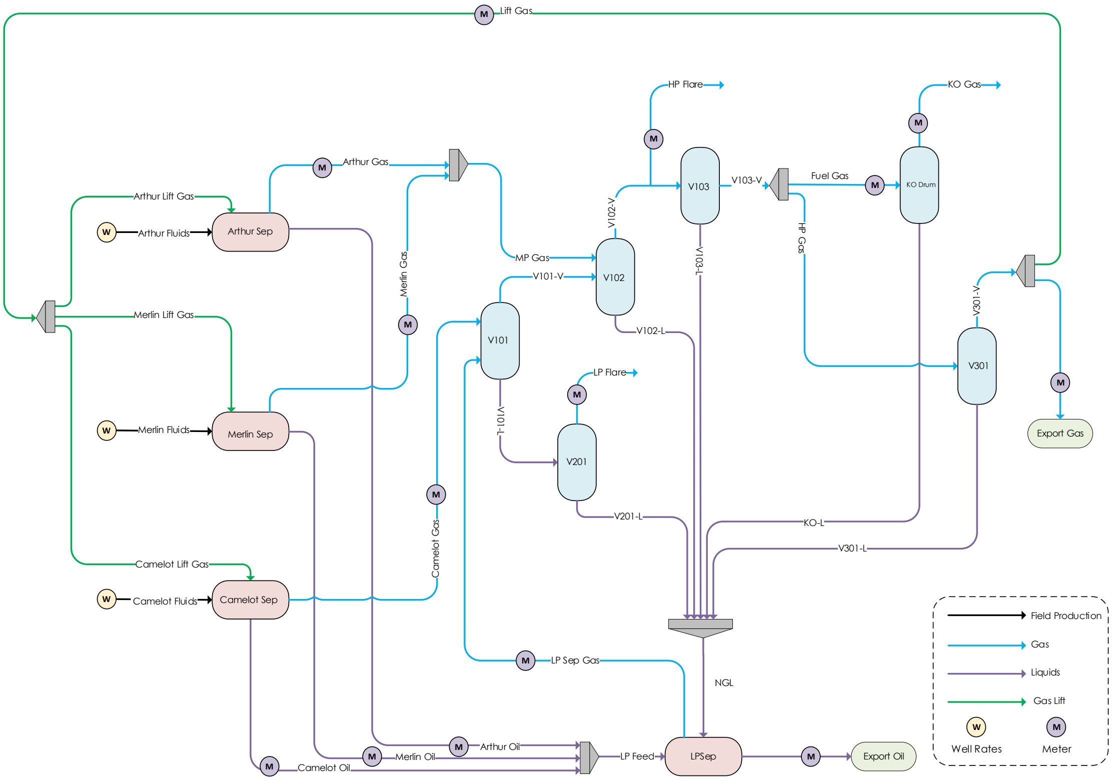


Table 4 – The true flow rates and standard deviations associated with the measured streams in the realistic system

Stream	True Flow Rate (tonnes/day)	Standard Deviation	Stream	True Flow Rate (tonnes/day)	Standard Deviation
Merlin Fluids	708.67	18.28	NGL	277.11	7.15
Merlin Lift Gas	500.23	12.91	HP Flare	38.67	1.00
Merlin Gas	556.09	14.35	Arthur Fluids	3486.52	89.95
Merlin Oil	652.82	16.84	Arthur Lift Gas	145.78	3.76
Camelot Fluids	625.74	16.14	Arthur Gas	461.92	11.92
Camelot Lift Gas	193.00	4.98	Arthur Oil	3170.38	81.80
Camelot Gas	255.61	6.59	Lift Gas	839.01	20.98
Camelot Oil	563.14	14.53	LP Flare Gas	1.27	0.03
Export Gas	206.14	5.32	LP Sep Feed	4386.34	113.17
Export Oil	4454.13	114.92	LP Sep Gas	209.32	5.40
Fuel Gas	120.72	3.11	KO Gas	120.72	3.11

Table 5 – OP and AVTI results of the realistic system

Method	OP						AVTI	
	Camelot Oil		Merlin Lift Gas		Merlin Gas			
	$(1-\alpha) = 95\%$	$(1-\alpha) = 99\%$	$(1-\alpha) = 95\%$	$(1-\alpha) = 99\%$	$(1-\alpha) = 95\%$	$(1-\alpha) = 99\%$	$(1-\alpha) = 95\%$	$(1-\alpha) = 99\%$
K Nearest Neighbors Regression	0.2010	0.2010	0.1610	0.1610	0.1940	0.1940	0.6290	0.6955
Extra Tree Regressor	0.9230	0.8830	0.8750	0.7840	0.8970	0.8710	0.1280	0.1413
Random Forest Regressor	0.9200	0.8890	0.8700	0.7930	0.8910	0.8770	0.1358	0.1400
Bagging Regressor	0.9230	0.8860	0.8650	0.7930	0.8940	0.8750	0.1345	0.1413
Gradient Boosting Regressor	0.9170	0.8670	0.8710	0.8060	0.8950	0.8730	0.1315	0.1420
Global Test	0.2630	0.0920	0.1280	0.0240	0.3170	0.1310	0.5815	0.6898
GLR Test	0.8700	0.5720	0.7550	0.3300	0.9060	0.6290	0.2640	0.4090

Table 5 shows the OP and AVTI of the selected ML methods and the two statistical tests on 3 streams namely Camelot Oil, Merlin Lift Gas, and Merlin Gas with 95% and 99% levels of confidence. It is recognised that K Nearest Neighbors and Global test perform poorly on the data as their OPs vary from about 13% to 30% dependent on stream with 95% level of confidence. Meanwhile, the other methods obtained high values of OP, from about 87% (Merlin Lift Gas stream) to 92% (Camelot Oil stream) at 95% confidence level. The other 4 ML methods perform better than the GLR test on Camelot Oil (92% vs. 87%) and Merlin Lift Gas (87% vs. 75%) stream, and slightly worse on Merlin Gas stream (89% vs. 90%).

When the confidence level increase to 99%, the Global test performs the worst among all methods, followed by K Nearest Neighbors. It is noted that the performance of K Nearest Neighbors did not change when we increased the confidence level. The OPs of other ML methods are from 79% to 88% which are 30-40% higher than that of GLR.

For AVTI, only 130-140 non-GE samples were wrongly classified by Extra Tree, Random Forest, Bagging, and Gradient Boosting among 1000 non-GE samples while more than 620 non-GE samples were wrongly classified by K Nearest Neighbors.

Figure 9 – MSE results of the realistic system

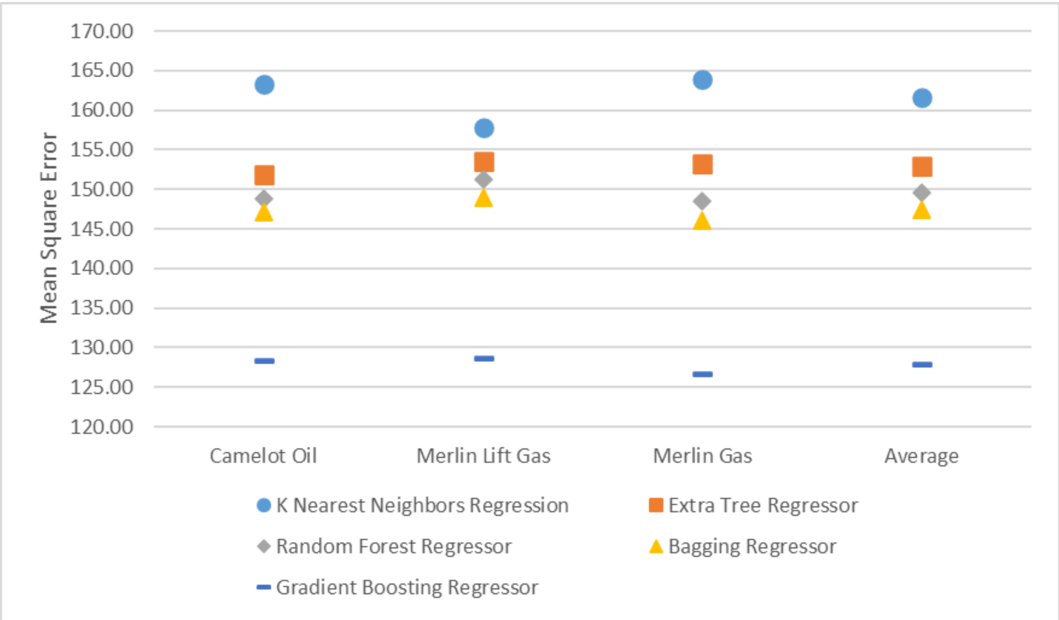


Table 6 – Statistics computed from the estimated bias of five ML methods

	Camelot Oil – true bias	K Nearest Neighbors Regression	Extra Tree Regressor	Random Forest Regressor	Bagging Regressor	Gradient Boosting Regressor
Average [tonnes/day]	56.35111	12.32763	53.88335	53.97006	53.9343	53.76136
Standard Deviation	1.38282	17.25364	19.80732	20.52005	20.67319	21.30253

Figure 9 compared the MSE of the 5 ML methods on each stream. It can be seen that Gradient Boosting obtained the smallest values of MSE on all streams among all methods. On average, the MSE of Gradient Boosting is only 127, which is much smaller than the Bagging (147.5), Random Forest (150), Extra Tree (153), and K Nearest Neighbors (161.5). Particularly, on the Camelot Separator Oil stream, the averaged bias estimations of Bagging, Random Forest, Extra Tree, and Gradient Boosting are about 53.8 which is close to the averaged value of generated bias (56.35111l). The averaged bias estimation of K Nearest Neighbors meanwhile is only 12.327683 which is much smaller than the average value of generated bias.

With the more complex dataset, the K Nearest Neighbors had a higher MSE, which means the estimations it made are less accurate, while Extra Tree, Random Forest, Bagging, and Gradient Boosting were performing similarly.

6 CONCLUSION

In this study, we have introduced an application of ML methods for GE estimation on flow measurement data. We generated training and testing data with different magnitudes of GE based on a simple water treatment process. The 10 selected ML methods were trained on the training data and then applied to the test data to estimate the GE. We used 3 performance metrics OP, AVTI, and MSE to report the performances. ML methods were compared to two well-known statistical tests namely Global Test and GLR Test. Experimental results showed that ML methods like Random Forest and Bagging achieved better results than the two statistical tests. The top 5 ML methods in this experiment namely Extra Tree, Random Forest, Gradient Boosting, Bagging, and K Nearest Neighbors were selected for another experiment on a realistic allocation system in order to further evaluate the effectiveness of the ML methods for the GED. It is observed that ML methods can obtain similar or better results compared to conventional tests for the GED when historical data (training data) is available, or reliable synthetic data can be generated.

For future work, the effect of the different sizes of training datasets on the performance of each ML method will be explored. In addition, we would like to extend the comparative study of ML methods to dynamic systems.

7 NOTATION

AVTI	Average Type I Error rate
GE	Gross Error
GED	Gross Error Detection
GLR	Generalised Likelihood Ratio
ML	Machine Learning
MSE	Mean Squared Error
OP	Overall Power
$U(\pm\gamma)$	Uniform distribution over the range of $-\gamma$ to $+\gamma$
Z	Standard Score
N	Number of Records
x_i	Measurement

Greek

α	Confidence level
σ	Variance
μ	Measurement mean value
δ	Measurement Bias
$\hat{\delta}$	Predicted Bias

8 REFERENCES

[1] P. Stockton and A. Spence, Experiences in the Use of Uncertainty Based Allocation in a North Sea Offshore Oil Allocation System, Production and Upstream Flow Measurement Workshop, 12-14 February 2008.

[2] N. Corbett, J. Johnston, R. Sibbald, P. Stockton and A. Wilson, Allocation in an Uncertain World. Maximising the Use of Data with UBA on Global Producer III, 31st North Sea Flow Measurement Workshop, 22-25 October 2013.

[3] D. Dobos, T.T. Nguyen, J. McCall, A. Wilson, P. Stockton, H. Corbett, Weighted ensemble of gross error detection methods based on particle swarm optimization. GECCO 2021 Companion - Proceedings of the 2021 Genetic and Evolutionary Computation Conference Companion, 307–308. 2021.

[4] S. Narasimhan and C. Jordache, Data Reconciliation and Gross Error Detection, An Intelligent Use of Process Data, published in 2000 by Gulf Publishing Company, Houston Texas, ISBN 0-88415-255-3.

[5] S. Narasimhan, and R. S. H. Mah. Generalised likelihood ratio method for gross error identification. AIChEJ. 33, 1514-1521, 1987.

[6] Lai, T. L_etc, Herbert Robbins, and C. Zi Wei. "Strong consistency of least squares estimates in multiple regression II." Journal of Multivariate Analysis 9.3 (1979): 343-361.

[7] J. Fürnkranz, Decision Tree. In: Sammut, C., Webb, G.I. (eds) Encyclopedia of Machine Learning. Springer, Boston, MA, 2011.

[8] L. Breiman, Bagging predictors, Machine Learning, 24(2), 123-140, 1996.

[9] L. Breiman, Random Forests, Machine Learning, 45(1), 5-32, 2001.

[10] J.H. Friedman, Greedy function approximation: A gradient boosting machine. Annals of Statistics, 29, 1189-1232, 2001.

[11] X. Jiang, P. Liu, & Z. Li, Data reconciliation and gross error detection for operational data in power plants. Energy, 75, 14–23, 2014.

[12] E. C. do Valle, R. de A. Kalid, A. R. Secchi, and A. Kiperstok, Collection of benchmark test problems for data reconciliation and gross error detection and identification. Computers & Chemical Engineering, 111, 134–148, 2018.

[13] M. A. F. Martins, C.A. Amaro, L.S. Souza, R.A. Kalid, and A. Kiperstok, New objective function for data reconciliation in water balance from industrial processes. Journal of Cleaner Production, 18(12), 1184–1189, 2010.

[14] V. N. Reddy, and M.L. Mavrovouniotis, An input-training neural network approach for gross error detection and sensor replacement. Chemical Engineering Research and Design, 76(4 A4), 478–489, 1998.

[15] J. A. Romagnoli and M. Cristina Sánchez, Data processing and reconciliation for chemical process operations, Computers & Chemical Engineering, 2000.

APPENDIX

Table A1 – OP – Stream bias of 10% with 95% confidence level

Method	Overall Power						
	Stream1	Stream2	Stream3	Stream4	Stream5	Stream6	Stream7
Linear Regression	1.0000	1.0000	0.5060	0.9900	0.5560	0.5590	0.3170
Bayesian ridge regression	1.0000	1.0000	0.5060	0.9900	0.5550	0.5590	0.3170
K Nearest Neighbors Regression	1.0000	0.9790	0.5410	0.9100	0.6270	0.4550	0.3160
Decision Tree Regression	1.0000	0.9990	0.7590	0.9880	0.7570	0.7650	0.6730
Random Forest Regressor	1.0000	1.0000	0.6640	0.9870	0.7400	0.6450	0.5550
Extra Tree Regressor	1.0000	1.0000	0.6960	0.9860	0.7240	0.7070	0.5980
Bagging Regressor	1.0000	1.0000	0.6690	0.9870	0.7330	0.6420	0.5540
Ada Boost Regressor	1.0000	1.0000	0.7700	0.9860	0.7530	0.7380	0.7420
Gradient Boosting Regressor	1.0000	1.0000	0.7090	0.9880	0.7370	0.7130	0.5980
Histogram Gradient Boosting	1.0000	1.0000	0.7080	0.9870	0.7010	0.6890	0.5750
Global Test	1.0000	0.9990	0.3260	0.8110	0.3930	0.4380	0.1920
GLR Test	1.0000	1.0000	0.8350	0.9830	0.8410	0.8700	0.6920

Method	Overall Power						
	Stream8	Stream9	Stream10	Stream11	Stream12	Stream13	Average
Linear Regression	1.0000	0.9960	0.9980	0.9900	1.0000	1.0000	**0.8394**
Bayesian ridge regression	1.0000	0.9960	0.9980	0.9900	1.0000	1.0000	**0.8393**
K Nearest Neighbors Regression	0.9840	0.8890	0.8070	0.6180	1.0000	1.0000	**0.7789**
Decision Tree Regression	1.0000	0.9830	0.9950	0.9800	1.0000	1.0000	**0.9153**
Random Forest Regressor	1.0000	0.9960	0.9970	0.9810	1.0000	1.0000	**0.8896**
Extra Tree Regressor	1.0000	0.9960	0.9970	0.9820	1.0000	1.0000	**0.8989**
Bagging Regressor	1.0000	0.9960	0.9970	0.9810	1.0000	1.0000	**0.8892**
Ada Boost Regressor	1.0000	0.9960	0.9990	0.9890	1.0000	1.0000	**0.9210**
Gradient Boosting Regressor	1.0000	0.9960	0.9980	0.9850	1.0000	1.0000	**0.9018**
Histogram Gradient Boosting	1.0000	0.9970	0.9980	0.9750	1.0000	1.0000	**0.8946**
Global Test	1.0000	0.6470	0.6960	0.3880	0.9690	1.0000	**0.6815**
GLR Test	1.0000	0.9600	0.9690	0.8410	1.0000	1.0000	**0.9224**

Table A2 – OP – Stream bias of 10% with 99% confidence level

Method	Overall Power						
	Stream1	Stream2	Stream3	Stream4	Stream5	Stream6	Stream7
Linear Regression	1.0000	1.0000	0.1500	0.9480	0.1770	0.1700	0.0470
Bayesian ridge regression	1.0000	1.0000	0.1490	0.9480	0.1760	0.1670	0.0460
K Nearest Neighbors Regression	1.0000	0.9700	0.4430	0.8770	0.5200	0.3670	0.2200
Decision Tree Regression	1.0000	0.9990	0.6290	0.9570	0.6340	0.6160	0.4910
Random Forest Regressor	1.0000	1.0000	0.6000	0.9800	0.5650	0.5520	0.4230
Extra Tree Regressor	1.0000	1.0000	0.5900	0.9710	0.5540	0.5660	0.3910
Bagging Regressor	1.0000	1.0000	0.5970	0.9790	0.5640	0.5510	0.4270
Ada Boost Regressor	1.0000	1.0000	0.6020	0.8970	0.5590	0.5530	0.3750
Gradient Boosting Regressor	1.0000	1.0000	0.6100	0.9750	0.5770	0.5900	0.4120
Histogram Gradient Boosting	1.0000	1.0000	0.5990	0.9830	0.5760	0.5810	0.4480
Global Test	1.0000	0.9990	0.1280	0.6230	0.1750	0.2080	0.0650
GLR Test	1.0000	1.0000	0.5130	0.9190	0.5460	0.6180	0.3330

Method	Overall Power						
	Stream8	Stream9	Stream10	Stream11	Stream12	Stream13	Average
Linear Regression	1.0000	0.9810	0.9910	0.9470	1.0000	1.0000	**0.7239**
Bayesian ridge regression	1.0000	0.9810	0.9910	0.9470	1.0000	1.0000	**0.7235**
K Nearest Neighbors Regression	0.9760	0.8410	0.7330	0.5010	1.0000	1.0000	**0.7268**
Decision Tree Regression	1.0000	0.9740	0.9530	0.9430	1.0000	1.0000	**0.8612**
Random Forest Regressor	1.0000	0.9940	0.9950	0.9770	1.0000	1.0000	**0.8528**
Extra Tree Regressor	1.0000	0.9930	0.9950	0.9710	1.0000	1.0000	**0.8485**
Bagging Regressor	1.0000	0.9940	0.9950	0.9770	1.0000	1.0000	**0.8526**
Ada Boost Regressor	1.0000	0.9800	0.9830	0.9780	1.0000	1.0000	**0.8405**
Gradient Boosting Regressor	1.0000	0.9940	0.9950	0.9730	1.0000	1.0000	**0.8558**
Histogram Gradient Boosting	1.0000	0.9900	0.9970	0.9410	1.0000	1.0000	**0.8550**
Global Test	1.0000	0.4320	0.4540	0.2000	0.8930	1.0000	**0.5521**
GLR Test	1.0000	0.8150	0.8430	0.5660	0.9960	1.0000	**0.7807**

Table A3 – OP – Stream bias of 15% with 95% confidence level

Method	Overall Power						
	Stream1	Stream2	Stream3	Stream4	Stream5	Stream6	Stream7
Linear Regression	1.0000	1.0000	0.8710	1.0000	0.8840	0.9010	0.5980
Bayesian ridge regression	1.0000	1.0000	0.8710	1.0000	0.8840	0.9010	0.5970
K Nearest Neighbors Regression	1.0000	1.0000	0.8990	1.0000	0.9450	0.8440	0.6740
Decision Tree Regression	1.0000	1.0000	0.9640	1.0000	0.9630	0.9590	0.8890
Random Forest Regressor	1.0000	1.0000	0.9550	1.0000	0.9710	0.9380	0.8550
Extra Tree Regressor	1.0000	1.0000	0.9570	1.0000	0.9720	0.9610	0.8790
Bagging Regressor	1.0000	1.0000	0.9540	1.0000	0.9710	0.9390	0.8540
Ada Boost Regressor	1.0000	1.0000	0.9630	1.0000	0.9580	0.9630	0.9150
Gradient Boosting Regressor	1.0000	1.0000	0.9640	1.0000	0.9660	0.9610	0.8750
Histogram Gradient Boosting	1.0000	1.0000	0.9620	1.0000	0.9600	0.9510	0.8670
Global Test	1.0000	1.0000	0.7000	0.9970	0.7400	0.7830	0.4240
GLR Test	1.0000	1.0000	0.9750	1.0000	0.9760	0.9830	0.8800

Method	Overall Power						
	Stream8	Stream9	Stream10	Stream11	Stream12	Stream13	Average
Linear Regression	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	**0.9426**
Bayesian ridge regression	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	**0.9425**
K Nearest Neighbors Regression	1.0000	0.9990	0.9970	0.9690	1.0000	1.0000	**0.9482**
Decision Tree Regression	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	**0.9827**
Random Forest Regressor	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	**0.9784**
Extra Tree Regressor	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	**0.9822**
Bagging Regressor	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	**0.9783**
Ada Boost Regressor	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	**0.9845**
Gradient Boosting Regressor	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	**0.9820**
Histogram Gradient Boosting	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	**0.9800**
Global Test	1.0000	0.9760	0.9770	0.7320	1.0000	1.0000	**0.8715**
GLR Test	1.0000	1.0000	0.9990	0.9690	1.0000	1.0000	**0.9832**

Table A4 – OP – Stream bias of 15% with 99% confidence level

Method	Overall Power						
	Stream1	Stream2	Stream3	Stream4	Stream5	Stream6	Stream7
Linear Regression	1.0000	1.0000	0.5350	1.0000	0.6110	0.6160	0.2120
Bayesian ridge regression	1.0000	1.0000	0.5350	1.0000	0.6110	0.6140	0.2110
K Nearest Neighbors Regression	1.0000	1.0000	0.8450	0.9970	0.8990	0.7860	0.5720
Decision Tree Regression	1.0000	1.0000	0.9010	1.0000	0.9170	0.9030	0.7450
Random Forest Regressor	1.0000	1.0000	0.9380	1.0000	0.9250	0.9110	0.7620
Extra Tree Regressor	1.0000	1.0000	0.9370	1.0000	0.9180	0.9180	0.7360
Bagging Regressor	1.0000	1.0000	0.9390	1.0000	0.9230	0.9100	0.7640
Ada Boost Regressor	1.0000	1.0000	0.9430	1.0000	0.9260	0.9130	0.7270
Gradient Boosting Regressor	1.0000	1.0000	0.9410	1.0000	0.9250	0.9270	0.7460
Histogram Gradient Boosting	1.0000	1.0000	0.9410	1.0000	0.9210	0.9140	0.7780
Global Test	1.0000	1.0000	0.4640	0.9780	0.4990	0.5490	0.2030
GLR Test	1.0000	1.0000	0.8690	1.0000	0.8900	0.9140	0.6200
Method	Overall Power						
	Stream8	Stream9	Stream10	Stream11	Stream12	Stream13	Average
Linear Regression	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	**0.8442**
Bayesian ridge regression	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	**0.8439**
K Nearest Neighbors Regression	1.0000	0.9980	0.9960	0.9470	1.0000	1.0000	**0.9262**
Decision Tree Regression	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	**0.9589**
Random Forest Regressor	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	**0.9643**
Extra Tree Regressor	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	**0.9622**
Bagging Regressor	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	**0.9643**
Ada Boost Regressor	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	**0.9622**
Gradient Boosting Regressor	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	**0.9645**
Histogram Gradient Boosting	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	**0.9657**
Global Test	1.0000	0.8930	0.9270	0.5080	1.0000	1.0000	**0.7708**
GLR Test	1.0000	0.9960	0.9960	0.8670	1.0000	1.0000	**0.9348**

Table A5– OP – Stream bias of -10% with 95% confidence level

Method	Overall Power						
	Stream1	Stream2	Stream3	Stream4	Stream5	Stream6	Stream7
Linear Regression	1.0000	1.0000	0.4900	0.9990	0.5390	0.5970	0.3110
Bayesian ridge regression	1.0000	1.0000	0.4900	0.9990	0.5380	0.5960	0.3110
K Nearest Neighbors Regression	1.0000	0.9760	0.4630	0.9480	0.6140	0.5190	0.3000
Decision Tree Regression	1.0000	1.0000	0.7840	0.9960	0.7830	0.8060	0.6870
Random Forest Regressor	1.0000	1.0000	0.6350	0.9970	0.7390	0.7590	0.6240
Extra Tree Regressor	1.0000	1.0000	0.6590	0.9960	0.7420	0.7670	0.6290
Bagging Regressor	1.0000	1.0000	0.6390	0.9960	0.7330	0.7630	0.6350
Ada Boost Regressor	1.0000	1.0000	0.7120	0.9980	0.7390	0.8750	0.7260
Gradient Boosting Regressor	1.0000	1.0000	0.6850	0.9970	0.7550	0.7880	0.6550
Histogram Gradient Boosting	1.0000	1.0000	0.6970	0.9910	0.7120	0.7420	0.6210
Global Test	1.0000	1.0000	0.5140	0.9110	0.5640	0.5860	0.3380
GLR Test	1.0000	1.0000	0.9060	0.9960	0.9180	0.9320	0.7800

Method	Overall Power						
	Stream8	Stream9	Stream10	Stream11	Stream12	Stream13	Average
Linear Regression	1.0000	1.0000	1.0000	0.9980	1.0000	1.0000	**0.8411**
Bayesian ridge regression	1.0000	1.0000	1.0000	0.9980	1.0000	1.0000	**0.8409**
K Nearest Neighbors Regression	0.9890	0.9130	0.8610	0.6810	1.0000	1.0000	**0.7895**
Decision Tree Regression	1.0000	0.9980	0.9990	0.9900	1.0000	1.0000	**0.9264**
Random Forest Regressor	1.0000	1.0000	1.0000	0.9950	1.0000	1.0000	**0.9038**
Extra Tree Regressor	1.0000	1.0000	1.0000	0.9950	1.0000	1.0000	**0.9068**
Bagging Regressor	1.0000	1.0000	1.0000	0.9950	1.0000	1.0000	**0.9047**
Ada Boost Regressor	1.0000	0.9990	1.0000	1.0000	1.0000	1.0000	**0.9268**
Gradient Boosting Regressor	1.0000	1.0000	1.0000	0.9890	1.0000	1.0000	**0.9130**
Histogram Gradient Boosting	1.0000	0.9990	0.9980	0.9920	1.0000	1.0000	**0.9040**
Global Test	1.0000	0.7250	0.7820	0.4160	0.9850	1.0000	**0.7555**
GLR Test	1.0000	0.9690	0.9860	0.8640	0.9990	1.0000	**0.9500**

Table A6– OP – Stream bias of -10% with 99% confidence level

Method	Overall Power						
	Stream1	Stream2	Stream3	Stream4	Stream5	Stream6	Stream7
Linear Regression	1.0000	1.0000	0.0890	0.9720	0.1340	0.1520	0.0470
Bayesian ridge regression	1.0000	1.0000	0.0890	0.9720	0.1330	0.1500	0.0470
K Nearest Neighbors Regression	1.0000	0.9660	0.3500	0.9130	0.4690	0.4040	0.1890
Decision Tree Regression	1.0000	1.0000	0.6680	0.9730	0.6470	0.7070	0.5260
Random Forest Regressor	1.0000	1.0000	0.5520	0.9940	0.5360	0.5700	0.3760
Extra Tree Regressor	1.0000	1.0000	0.5330	0.9910	0.5540	0.6170	0.3610
Bagging Regressor	1.0000	1.0000	0.5530	0.9930	0.5380	0.5770	0.3770
Ada Boost Regressor	1.0000	1.0000	0.5530	0.9880	0.5940	0.5880	0.3840
Gradient Boosting Regressor	1.0000	1.0000	0.5810	0.9870	0.6030	0.6570	0.4060
Histogram Gradient Boosting	1.0000	1.0000	0.5270	0.9780	0.5780	0.6000	0.4030
Global Test	1.0000	1.0000	0.2820	0.7620	0.3320	0.3610	0.1500
GLR Test	1.0000	1.0000	0.6730	0.9690	0.7340	0.7600	0.4670

Method	Overall Power						
	Stream8	Stream9	Stream10	Stream11	Stream12	Stream13	Average
Linear Regression	1.0000	0.9950	1.0000	0.9630	1.0000	1.0000	**0.7194**
Bayesian ridge regression	1.0000	0.9950	1.0000	0.9630	1.0000	1.0000	**0.7192**
K Nearest Neighbors Regression	0.9840	0.8610	0.8040	0.5640	1.0000	1.0000	**0.7311**
Decision Tree Regression	1.0000	0.9950	0.9980	0.9410	1.0000	1.0000	**0.8812**
Random Forest Regressor	1.0000	1.0000	1.0000	0.9780	1.0000	1.0000	**0.8466**
Extra Tree Regressor	1.0000	1.0000	1.0000	0.9770	1.0000	1.0000	**0.8487**
Bagging Regressor	1.0000	1.0000	1.0000	0.9780	1.0000	1.0000	**0.8474**
Ada Boost Regressor	1.0000	0.9960	1.0000	0.9110	1.0000	1.0000	**0.8472**
Gradient Boosting Regressor	1.0000	0.9980	0.9990	0.9790	1.0000	1.0000	**0.8623**
Histogram Gradient Boosting	1.0000	0.9970	0.9950	0.9760	1.0000	1.0000	**0.8503**
Global Test	0.9990	0.5060	0.5690	0.1870	0.9340	1.0000	**0.6217**
GLR Test	1.0000	0.8760	0.9140	0.6020	0.9980	1.0000	**0.8456**