APPLIED MACHINE LEARNING AND DEEP LEARNING TO PREDICT OIL AND GAS PRODUCTION

A PREPRINT

Luong Khanh Loc
Department of Petroleum Engineerig
PetroVietnam University
723 CMT8, Long Toan, Baria
loclk04@pvu.edu.vn

Nguyen Van Hung
Head of Department of Petroleum Engineering
PetroVietnam University
723 CMT8, Long Toan, Baria
hungnv@pvu.edu.vn

May 27, 2019

ABSTRACT

Recently, artificial intelligent (AI) has gained a great deal of attention in many areas, especially oil and gas industry. There was a significant development of AI that enables oil and gas companies to reduce a huge amount of cost and make more accurate decisions. In this paper, a new method based on Artificial Intelligence including machine learning and deep learning techniques has applied to petroleum engineering to predict the oil and gas production.
Bottom hole pressure and flow rate are predicted using various algorithms based on AI technique. This study proposed a novel approach using Regression algorithms and Ensemble methods to leverage the results. A typical Recurrent Neural Networks include Long Short Term Memory (LSTM), Gated Recurrent Unit (GRU) and Simple RNN has been developed to predict oil ans gas production.
This research focuses on figuring out the potential and efficiency of machine learning and deep learning algorithms to predict production in the future. This study explored ten different machine/deep learning algorithms and their performance evaluation.Python is choosen for this purposes because of its useful and robust machine/deep learning libraries.

***K*eywords** machine learning · deep learning · production forecasting · artificial intelligent · volve dataset

1 Introduction

Artificial Intelligence (AI) is transforming the oil and gas industry. A study from McKinsey (2016) in 2016 mentioned that digital organization, which covers AI and Machine Learning, is one of the five big ideas that will reshape the future of the industry. Digitizing and automating work will result in better safety and productivity as fewer people will be at risk and the risk of human error is reduced. Moreover, in the low oil price environment, oil companies often have no choice but to cut their operational budget, reduce the headcount, and/or fully optimize their remaining assets. This further encourages the adoption of artificial intelligence in the industry, with the hope to simplify and/or automate inefficient processes.[2].

There have been some implementations of AI in the field of oil and gas production. Boomer (1995) [3]used Neural Network to forecast oil production in Vacuum Field, Texas and the model outperformed the professionals 93% of the time. Cao et al. (2016) found that Neural Network performed better than conventional decline curve methods (Arps, Duong, and Stretched Exponential Production Decline). Suhag et al. (2017) [4] implemented Neural Network to predict oil rate in Bakken shale wells and yielded better results than Duong's method. J. Sun, X. Ma, and M. Kazi [5] have recently propose a novel method of using Recurrent Neural Networks for production forecast and it give outperformed results compared to conventional methods like DCA, SEPD and Duong.

In this paper, we have develop a workflow of using machine learning and deep learning techique to predict future production rate. Ensemble methods have been built to gain better result. We also used three kind of Recurrent Neural Network for prediction.

2 Artificial Intelligence

Artificial intelligence is a branch of computer science that aims to create intelligent machines. It has become an essential part of the technology industry. Artificial intelligence (AI) is the simulation of human intelligence processes by machines, especially computer systems. These processes include learning (the acquisition of information and rules for using the information), reasoning (using rules to reach approximate or definite conclusions) and self-correction. Particular applications of AI include expert systems, speech recognition and machine vision.
Machine learning and deep learning are core part of AI. Learning without any kind of supervision requires an ability to identify patterns in streams of inputs, whereas learning with adequate supervision involves classification and numerical regressions. Classification determines the category an object belongs to and regression deals with obtaining a set of numerical input or output examples, thereby discovering functions enabling the generation of suitable outputs from respective inputs.

2.1 Machine learning

Machine learning is a statistical technology to get computers output expected results without being explicitly programmed. Machine learning is based on big-data without explicitly programming, therefore, it saves programming time and avoids various ideal scenarios restrictions, and it also improves the output prediction accuracy. Machine learning method such as Artificial Neural Network (ANN) is applied in oil and gas industry for production forecasting [1](Cao et al., 2016).
The main objective of every machine learning algorithms is to find a optimal set of parameters $\precapprox$ tailoring of the hypothesis to a specific set of data. In this paper, we have used 12 different machine learning algorithms for prediction. A brief overview of 12 algorithms we have used will be introduced below.

2.1.1 Linear Regression

The simplest linear model for regression is one that involves a linear combination of the input variables[7]

$$y(\mathbf{x}, \mathbf{w}) = w_0 w_1 x_1 + \ldots + w_D x_D \tag{1}$$

where $x = (x_1, \ldots, x_D)^T$.The key property of this model is that it is a linear function of the parameters $w_0, \ldots, w_D$. It is also, however, a linear function of the input variables xi, and this imposes significant limitations on the model. We therefore extend the class of models by considering linear combinations of fixed nonlinear functions of the input variables, of the form:

$$y(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(x) \tag{2}$$

where $\phi_j(x)$are known as basis functions.
In this research, we just applied linear regression as the simplest model of machine learning algotithms for prediction.

2.1.2 Ridge

Rigde is a Linear least squares with L2 regularization. Ridge regression addresses some of the problems of Ordinary Least Squares by imposing a penalty on the size of coefficients. The ridge coefficients minimize a penalized residual sum of squares.[16]

$$\min_w = \parallel Xw - y \parallel_2^2 + \alpha \parallel w \parallel_2^2 \tag{3}$$

Here,$\alpha \geq 0$ is a complexity parameter that controls the amount of shrinkage: the larger the value of α, the greater the amount of shrinkage and thus the coefficients become more robust to collinearity.

2.1.3 KNeighbors Regression

K-Nearest Neighbors uses local neighborhood points to make a prediction. In more detail, distance (Euclidean distance in this study) is calculated from a query point $x^{(i)}$ in the test set to each data point $x^{'(i)}$ in the training time:

$$D(x, x^{'}) = \sqrt{\sum_{i=1}^{N}(x^{(i)} - x^{'(i)})^2} \tag{4}$$

Continuousky, sorting the distance and pick k closest data points. Define the k points closest points in the training set that are closest to x the set G. The prediction during testing time is calculated by local interpolation of the nearest neighbors in G [2].

$$\hat{y}^i = \frac{1}{K}\sum_{j \in G}^{N} y_j \tag{5}$$

where $\hat{y}^i$ is the output of KNeighbors regressor and k is the number of nearest neighbors points.

2.1.4 Support Vector Regression

The method of Support Vector Regression is extended from SVM algorithms to solve the regression problems. Given training vectors $x_i \in \mathbb{R}^p, i = 1, \ldots, n$ and a vector $y \in \mathbb{R}^n$, SVR solves the following primal problem:

$$\min_{w,b,\zeta,\zeta^*} \frac{1}{2}w^T w + C\sum_{i=1}^{n}(\zeta_i + \zeta_i^*) \tag{6}$$

$$\begin{aligned} subject\,to\,: y_i - w^T\phi(x_i) - b \leq \varepsilon + \zeta_i^*, \\ w^T\phi(x_i) + b - y_i \leq \varepsilon + \zeta_i^*, \\ \zeta_i, \zeta_i^* \geq 0, i = 1, \ldots, n. \end{aligned}$$

2.1.5 Decision Tree Regression

Decision Tree [8] (also called Regression Tree in regression problems) is a supervised learning model that creates branches of decision nodes. Decision Tree learns a dataset by making the best split of features in the training set into subsets based on the homogeneity of each new subset. The process goes on until a stopping criterion (e.g. tree size or number of observation per node) is reached (Breiman et al. 1984).
Data $y \in \mathbb{R}^n, X \in \mathbb{R}^{n \times p}$ and observation $(y_i, x_i) \in \mathbb{R}^{p+1}, i = 1, \ldots, n$. Suppose we have a partition of R^p into M regions $R_1, \ldots, R_m$.We predict the response using a constant on each R_i:

$$f(x) = \sum_{i=1}^{m} c_i \cdot \mathbf{1}_{x_i \in \mathbb{R}_i} \tag{7}$$

In order to optimize $\sum_{i=1}^{n}(y_i - f(x_i))^2$, one needs to choose: $\hat{c}_i = ave(y_j : x_j \in \mathbb{R}_i)$.
Consider a splitting variable $j \in 1, \ldots, p$and splitting point $s \in \mathbb{R}$. Define the two half-plane:

$$R_1(j, s) := \{x \in \mathbb{R}^p : x_j \leq s\} \quad R_2(j, s) := \{x \in \mathbb{R}^p : x_j > s\}$$

Choose j,s to minimize:

$$\min_{j,s}\left[\min_{c1 \in \mathbb{R}} \sum_{x_i \in R_1(j,s)} (y_i - c_1)^2 + \min_{c2 \in \mathbb{R}} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2\right] \tag{8}$$

2.1.6 Random Forest Regression

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees[9].
A random forest is a meta estimator that fits a number of classifying decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. The generic formula is shown:

$$\hat{f}(X) = \frac{1}{B}\sum_{b=1}^{B} f_b(X) \tag{9}$$

where $\hat{f}(X)$ is the output of random forest mapping function, $f_b(X)$ is the output of individual tree function, and B is the number of classifier or estimator.[13]

2.1.7 Extra Trees Regression

The Extra-Tree method (standing for extremely randomized trees) was proposed in [GEW06], with the main objective of further randomizing tree building in the context of numerical input features, where the choice of the optimal cut-point is responsible for a large proportion of the variance of the induced tree.
The Extra-Trees algorithm builds an ensemble of unpruned decision or regression trees according to the classical top-down procedure. Its two main differences with other tree- based ensemble methods are that it splits nodes by choosing cut-points fully at random and that it uses the whole learning sample (rather than a bootstrap replica) to grow the trees. With respect to random forests, the method drops the idea of using bootstrap copies of the learning sample, and instead of trying to find an optimal cut-point for each one of the K randomly chosen features at each node, it selects a cut-point at random. A detailed explanation of ExtraTrees algorithms has been show in work of Pierre Geurts, Damien Ernst and Louis Wehenkel[14].

2.1.8 Ada Boost Regression

AdaBoost creates T weak learners ht (in this study trees) and boosts these weak learners to stronger ones. It does so by assigning more weight to data points whose error is high. In this study, we used linear loss function to update the weights. In the next iteration, the algorithm will try to avoid mispredicting these data points as it would greatly increase the penalty/error. Similar to Random Forest, the final prediction H is computed as a weighted average of the weak learners' prediction.[2]

$$H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x)) \tag{10}$$

where $H(x)$ is the output of Adaboost mapping function, α_t is the weight assigned to each classifier, $h_t(x)$ is the ouput of the weak classifier or estimator.[13]

2.1.9 Gradient Boosting Regression

Similar to AdaBoost, Gradient Boosting (or Gradient Boosting Machine) uses an ensemble of weak prediction classifiers (decision trees, in this study). Each individual tree is trained as a decision tree estimator. The model trains the tree one at a time and calculates mean squared error loss. The next step is gradient descent. But the unique step here is instead of updating parameter value, the gradient descent updates a tree function by modifying its parameters so that it moves to the right gradient[15].

$$F(x) = \sum_{m=1}^{M} \gamma_m h_m(x) \tag{11}$$

where $h_m(x)$ are the basis functions which are usually called weak learners in the context of boosting. Gradient Tree Boosting uses decision trees of fixed size as weak learners. Decision trees have a number of abilities that make them valuable for boosting, namely the ability to handle data of mixed type and the ability to model complex functions.[16]

2.1.10 Extreme Gradient Boosting Regression

XGBoost is one of the most popular and efficient implementations of the Gradient Boosted Trees algorithm, a supervised learning method that is based on function approximation by optimizing specific loss functions as well as applying several regularization techniques.
XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. It implements machine learning algorithms under the Gradient Boosting framework. XGBoost provides a parallel tree boosting (also known as GBDT, GBM) that solve many data science problems in a fast and accurate way. The same code runs on major distributed environment (Hadoop, SGE, MPI) and can solve problems beyond billions of examples.[17] The objective function (loss function and regularization) at iteration t that we need to minimize is the following:

$$Obj = \sum_{i=1}^{n} \mathcal{L}(y_i, \hat{y_i}) + \sum_{i=1}^{K} \Omega(f_k) \tag{12}$$

$$\Omega(f_t) = \gamma \mathbb{T} + \frac{1}{2} \lambda \sum_{j=1}^{T} w_j^2$$

$$f_t(x) = w_{q(x)}, w \in \mathbb{R}^T, q \in \mathbb{R}^d \rightarrow 1, 2, \ldots, T$$

where $\mathcal{L}(y_i, \hat{y_i})$ is training loss, $\Omega(f_k)$ is regularization, measures the complexity of trees and w is the leaf weight of the tree.

2.1.11 Light Gradient Boosting Regression

LightGBM is a gradient boosting framework developed by Microsoft that uses tree based learning algorithms. It is designed to be distributed and efficient with the advantages: faster training speed and higher efficiency, lower memory usage, better accuracy, support of parallel and GPU learning, capable of handling large-scale data.[18]
Light GBM grows tree vertically while other algorithm grows trees horizontally meaning that light GBM grows tree leaf-wise while other algorithm grows level-wise. It will choose the leaf with max delta loss to grow. When growing the same leaf, Leaf-wise algorithm can reduce more loss than a level-wise algorithm.
Both LightGBM and XGBoost are widely used and provide highly optimized, scalable and fast implementations of gradient boosted machines (GBMs).
In terms of LightGBM specifically, a detailed overview of the LightGBM algorithm and its innovations is given in the NIPS paper [18].

2.2 Deep learning

First introduced in the early 1980s, neural networks are a programming paradigm within deep learning that enable computers to learn from observational data. They leverage the physiology and architecture of the human brain [10].
Simple architecture of neural networks is show below:

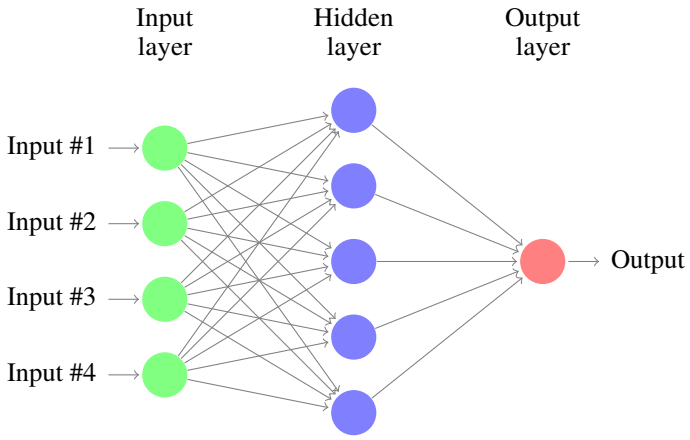


Figure 1: Shallow neural networks with one hidden layer

ANN has drawn attention from well testing researchers since the 1990s. An early application in well testing was introduced by Al-Kaabi and Lee (1990) to identify the interpretation models from derivative plots. Juniardi and Ershaghi (1993) studied the complexities of using ANN with a focus on well test analysis of faulted reservoirs, and discussed some of the shortcomings of the ANN method. Around the same time, Ershaghi et al. (1993) proposed an enhanced approach based on the work of Al-Kaabi and Lee (1990), by training multiple ANNs where each neural net was designed to learn the patterns for a specific reservoir model. In 1995, Athichanagorn and Horne combined ANN with sequential predictive probability method to improve the accuracy of interpretation. ANN was used to generate initial parameter estimates of the candidate reservoir models that it identified. The candidate models and initial estimates were then passed to the sequential predictive probability method for final evaluations.

2.2.1 Deep feed-foward networks

Deep feedforward networks, also called feedforward neural networks, or multilayer perceptron (MLPS), are the quintessential learning models. The goal of a feedforward networks is to approximate some function f. For example, for a classifier, $y = f(x)$ maps an in put $\mathbf{x}$ to a category y. A feedforward networks defines a mapping $\mathbf{y} = f(\mathbf{x}, \theta)$ and learns the values of parameters θ that the result in the best function approximation.[12]

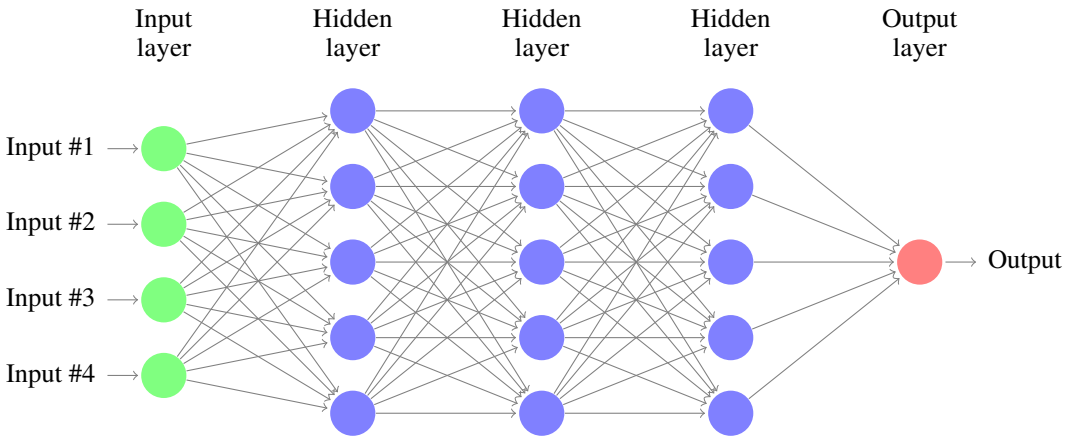


Figure 2: Deep feed forward neural networks

Recently, deep feed forward neural networks (or deep neural networks) has been applied widely in oil and gas industry. A new approach to reservoir characterization using deep neural netwokrs has been introduced by M. Korjani, Andrei Popa, Eli Grijalva, and Steve Cassidy [23]; wellbore stability prediction [24]; Deep recurrent neural network for surface pressure respone during the hydraulics fracturing process [25]; Deep neural network based prediction of Leak-Off Pressure in Offshore Norway [26]; New method for extracting the work status in Shipyard using Deep Neural Networks [27]; diagnostics of rod pumps with use of deep neural networks [29]; Automatic salt-body classification using a deep convolutional neural network [**?**]; seismic problems has been issued by deep neural networks in various papers such as: deep neural network architectures arising in seismic inverse problems[28]; Application of convolutional and deep neural networks for GPU based seismic interpretations [31]. Beside, deep convolutional neural networks has also been built for geology problems: Deep Learning Convolutional Neural Networks to Predict Porous Media Properties [32], Visual explanations from convolutional neural networks for fault detection[33].

2.2.2 Simple Recurrent neural networks

Recurrent neural networks (RNNs) exploit characteristics of a feedback network, allowing the model to construct a sequential representation of input data. However, simple RNNs have been notoriously difficult to train, due to their iterative nature and the highly volatile relationship between the parameters and the hidden states [11].
The network is given a sequence of inputs. Then, the network computes a sequence of hidden states and a sequence of predictions for each step in time. The sequences are computed by iterating through a series of equations that updates the model's weight matrices, hidden layers, and the output units. The RNN utilizes predefined vector-valued functions that contain a computable Jacobian and are typically non-linear and applied coordinate-wise[19].

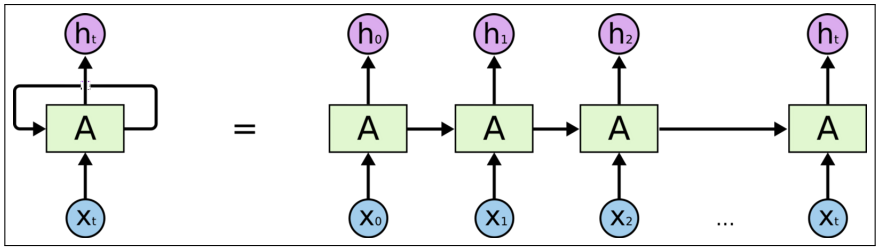


Figure 3: Recurrent neural network simple architechture[20]

2.2.3 Long short term memory

Long short-term memory (LSTM) units (or blocks) are a building unit for layers of a recurrent neural network (RNN). A RNN composed of LSTM units is often called an LSTM network. A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. The cell is responsible for "remembering" values over arbitrary time intervals; hence the word "memory" in LSTM. Each of the three gates can be thought of as a "conventional" artificial neuron, as in a multi-layer (or feedforward) neural network: that is, they compute an activation (using an activation function) of a weighted sum. Intuitively, they can be thought as regulators of the flow of values that goes through the

connections of the LSTM; hence the denotation gate and there are connections between these gates and the cell. A detailed explanation has been introduced by Colah in his blog about long short term memory[20].

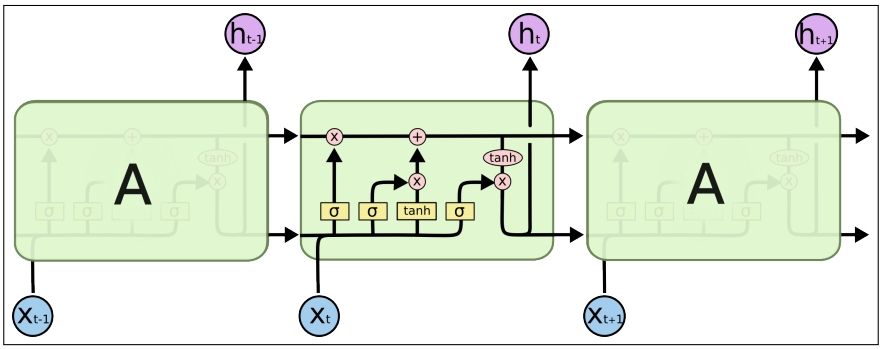


Figure 4: Simple stretch of long short term memory[20]

2.2.4 Gated Recurrent Unit

In simple words, the GRU unit does not have to use a memory unit to control the flow of information like the LSTM unit. It can directly makes use of the all hidden states without any control. GRUs have fewer parameters and thus may train a bit faster or need less data to generalize. But, with large data, the LSTMs with higher expressiveness may lead to better results.
They are almost similar to LSTMs except that they have two gates: reset gate and update gate. Reset gate determines how to combine new input to previous memory and update gate determines how much of the previous state to keep. Update gate in GRU is what input gate and forget gate were in LSTM. We don't have the second non linearity in GRU before calculating the outpu, neither they have the output gate.

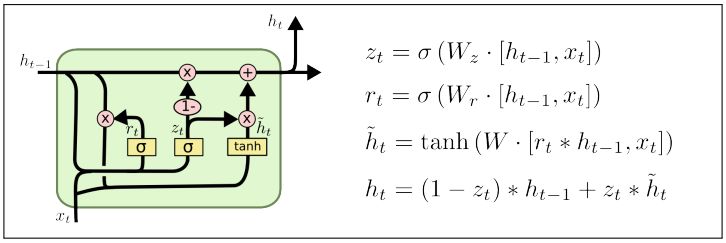


Figure 5: Simple stretch of Gated Recurrent Unit[20]

3 Methodology

3.1 Input data

The production dataset used in this paper is from Volve field on Norwegian continental shelf with around 8 years history. Daily oil, gas and water production data are all recorded consist with pressure, temperature and choke size as operation constraint. These production data are the main input variables for the machine learning and deep learning model, and it gives multi-phase production predictions in the future as model output.
Starting in February 2008, the Volve production lasted for about eight years. At peak, the field produced 56,000 barrels per day, and a total of 63 million barrels of oil were produced before the field was shut down in 2016. The field was developed when the oil price was low, and an unconventional concept was chosen to recover the resources in an easy and profitable manner. The field data will now have a new life. The Volve licensees were ExxonMobil and Bayerngas. Information about geology, geophysics, drilling, static models, dynamic simulations and other information has provied by Equinor company in 2018[21].
In this research, bottom hole pressure (BHP), tubing head pressure (THP), bottom hole temperature (BHT), well head temperature (WHT), different pressure in casing (DP), choke size (CS) in percentage is the input for training machine learning and deep learning algorithms.

3.2 Work-flow

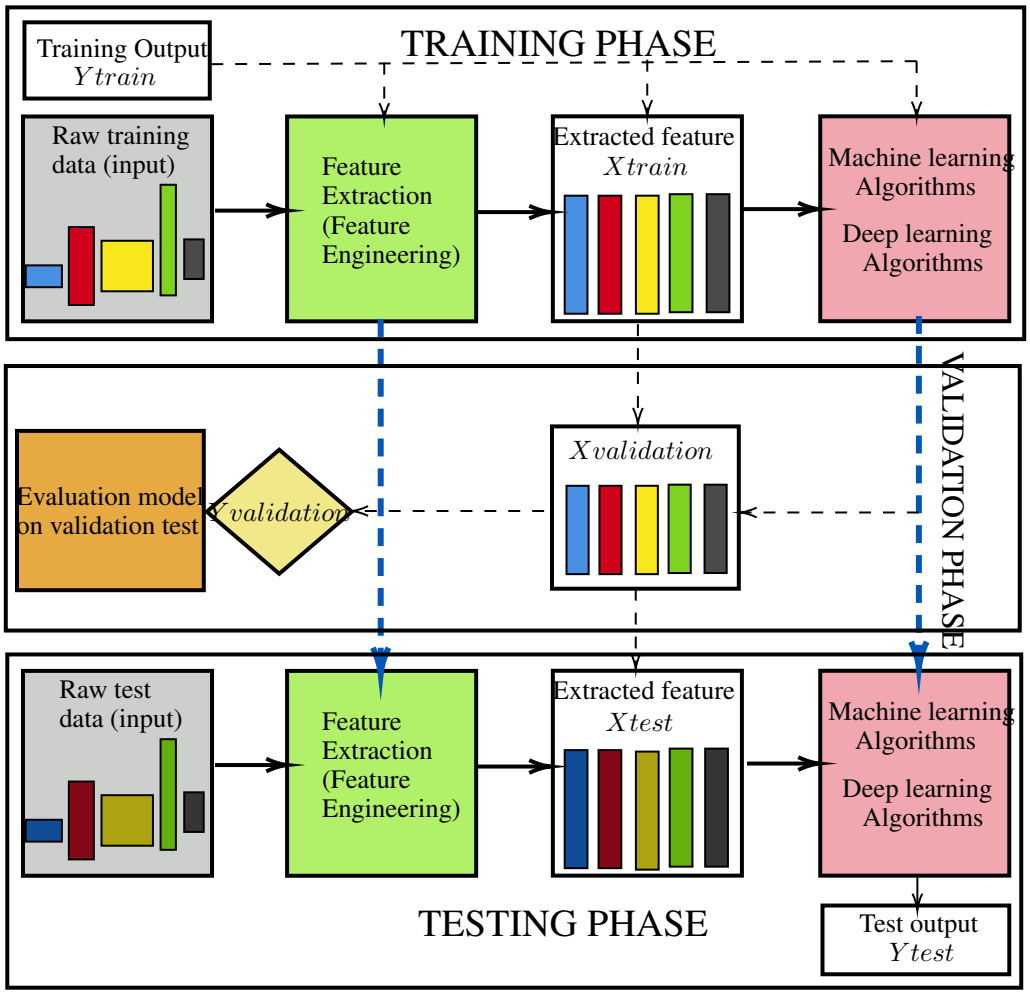


Figure 6: Work flow for using machine and deep learning algorithms

Almost every machine learning and deep learning algorithm has followed this work-flow above. Firslt, all raw data need to be collected altogether. Feature engineering means he process of using domain knowledge of the data to create

features that make machine learning algorithms work. Feature engineering is an informal topic, but it is considered essential in applied machine learning[22]. From Andrew Ng, a famous Professor in AI :"Applied machine learning is basically feature engineering".

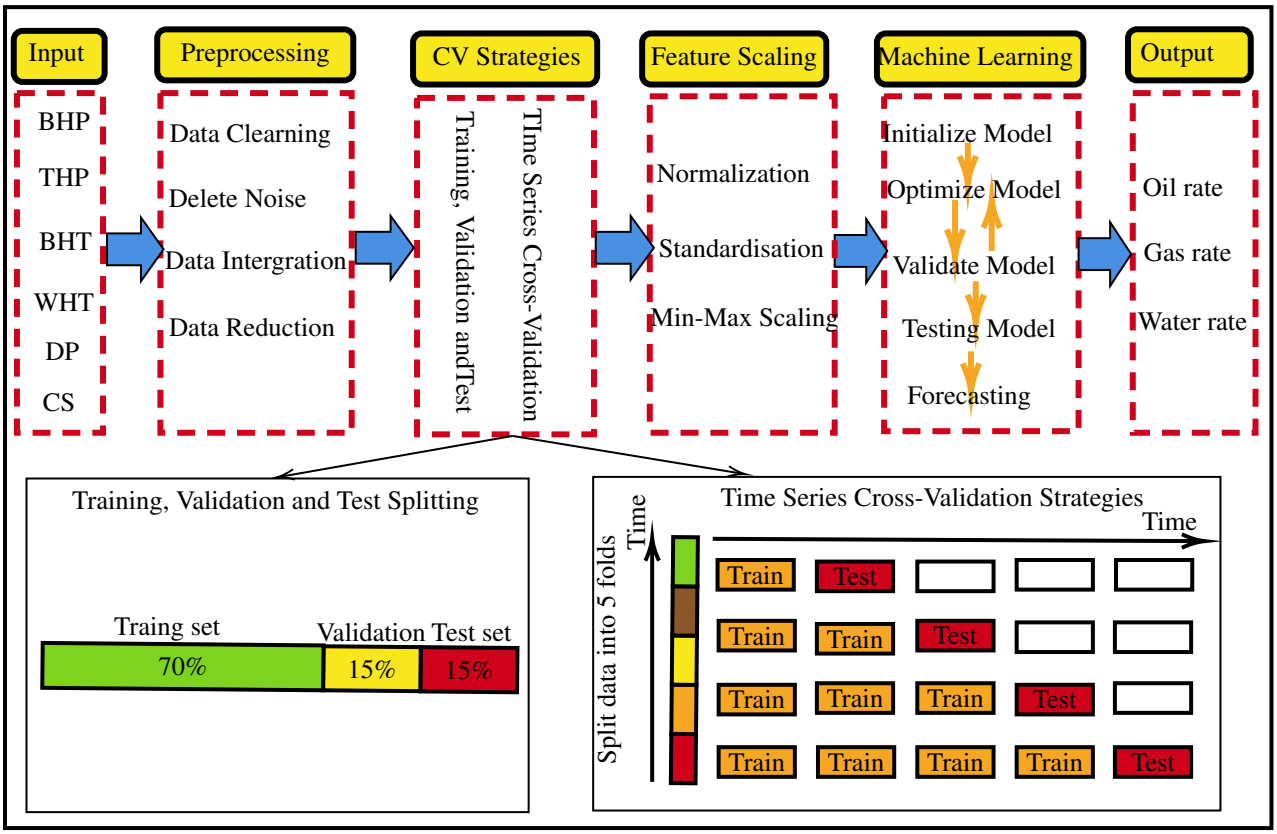


Figure 7: Detailed work flow for using machine leanring and deep learning algorithms in this paper

There are two scenarios of cross-validation (CV) strategies have been built in this reasearch. In first scenario, data is divided in three parts: training data (70%), validation data (15%) and test data (15%) with suffering all (figure7). In the second scenario, data is separated in to 5 different folds (figure 8) and algorithms will train and test alternatively in training and testing data in very loop.
Feature engieering in this research consists of three step from preprecessing to cross-validation (CV) strategies and feature scaling. Machine learning models are implementaed with Scikit-learn library, and deep learning models are also implemented with Keras library, two of these very useful and popular libraries for working with ML/DL algorithms. During training process, we have use grid search for finding the best combination of hyperparameters of algorithms.

4 Result and Discussion

4.1 Machine learning

The the table bellow we show the result of every algorithm we have used. Coefficient of determination (R^2) and mearn squared error (MSE) have been used to evaluation algorithsm.

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

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

Table 1: Score and error of machine learning algorithms on oil

No	Model	Validation Score	Validation Error	Test Score	Test Error
1	Linear Regression	0.82132	0.00598	0.80275	0.00609
2	Ridge	0.8257	0.00583	0.78125	0.00676
3	KNeighbors Regression	0.9218	0.00262	0.95795	0.00130
4	Support Vector Regression	0.8556	0.00483	0.84564	0.00477
5	Decision Tree Regression	0.94392	0.001876	0.95973	0.00124
6	Random Forest Regression	0.9687	0.00138	0.9585	0.00128
7	Extra Tree Regression	0.9569	0.00144	0.96031	0.00123
8	AdaBoost Regression	0.90031	0.00334	0.90247	0.00301
9	Gradient Boosting Regression	0.9554	0.00149	0.9597	0.00124
10	XgBoost Regression	0.92087	0.00265	0.93438	0.00203
11	LightGBM Regression	0.89392	0.00355	0.93392	0.00204

After comparing the result on valdation and test set, we just picked up five algorithms that yield the best score on test set. Score and error of gas and water prediction can be found in appendix (table 10 and 11). In order to evaluate how possible the algorithms can applied to forecast a new set of data, we use the pearson correlation for evaluation. The table below show the pearson correlation for the best five algorithms.

Table 2: Pearson correlation of best five algorithms

No	Model	Pearson correlation (oil)	Pearson correlation (gas)	Pearson correlation (water)
1	Extreme Gradient Boosting	0.96667	0.96895	0.8988
2	Decision Tree Regression	0.9798	0.9596	0.8970
3	Random Forest Regression	0.97927	0.9757	0.9161
4	Gradient Boosting Machine	0.96689	0.9745	0.92
5	Extra Tree Regression	0.98065	0.98205	0.9351

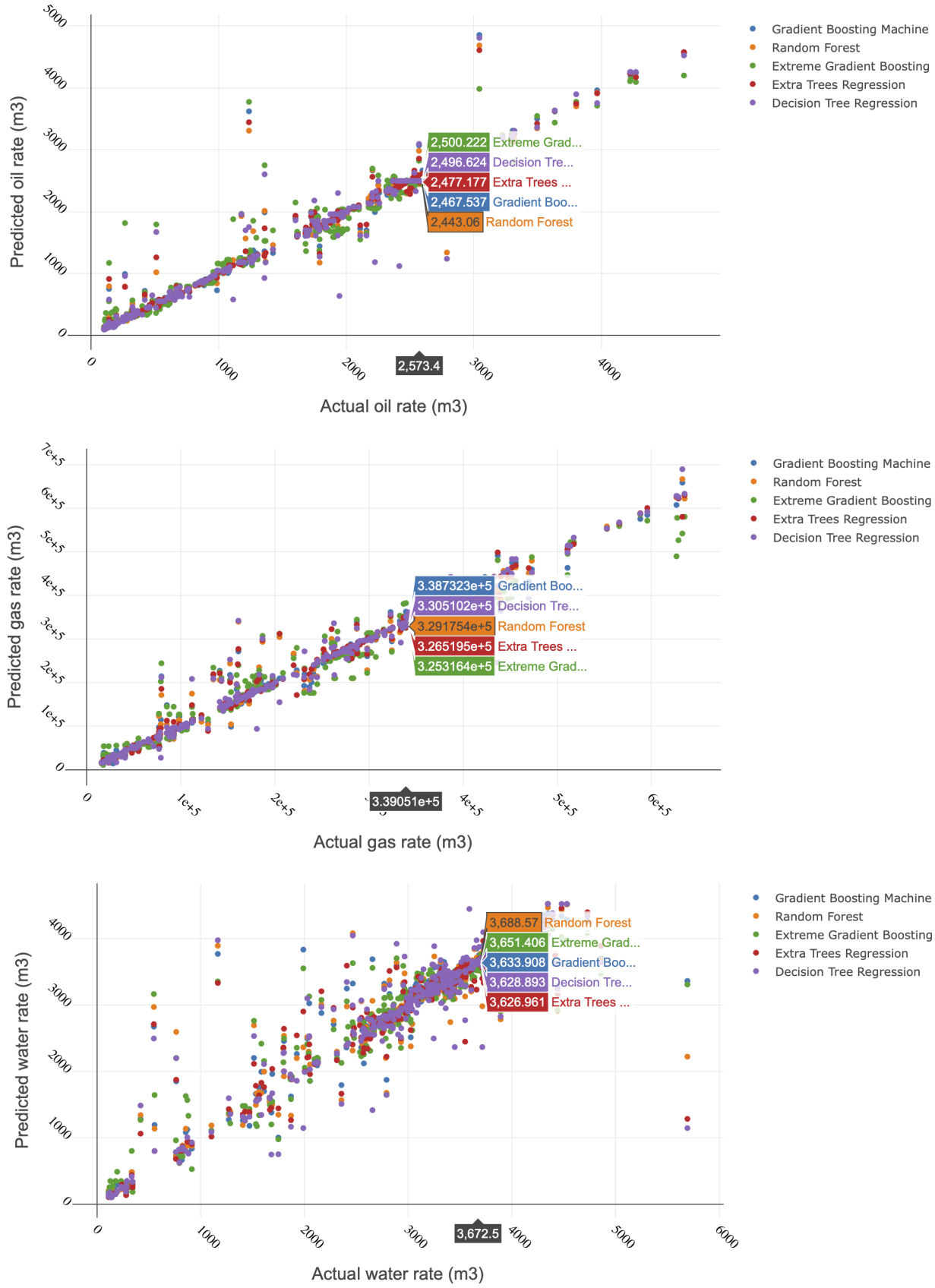
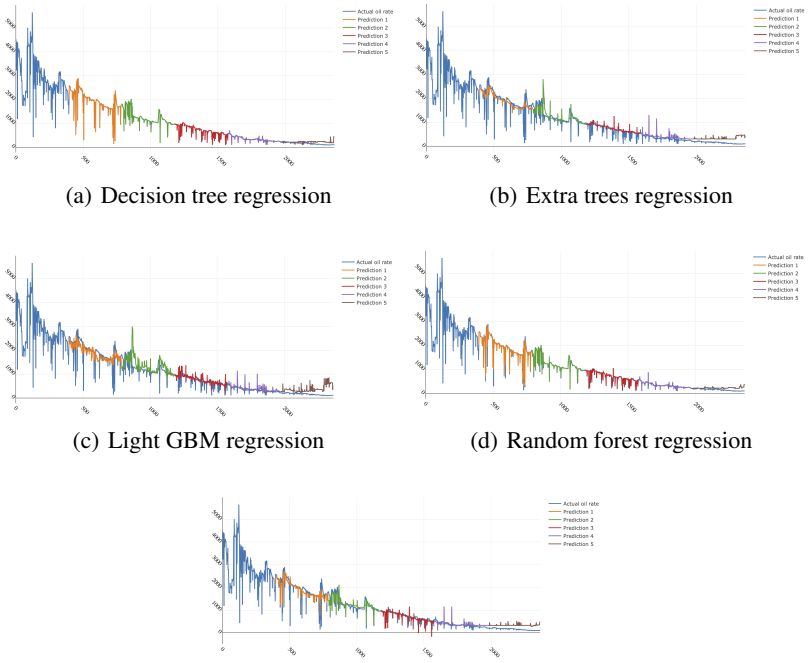


Figure 8: Production rate prediction verus actual values



(a) Decision tree regression

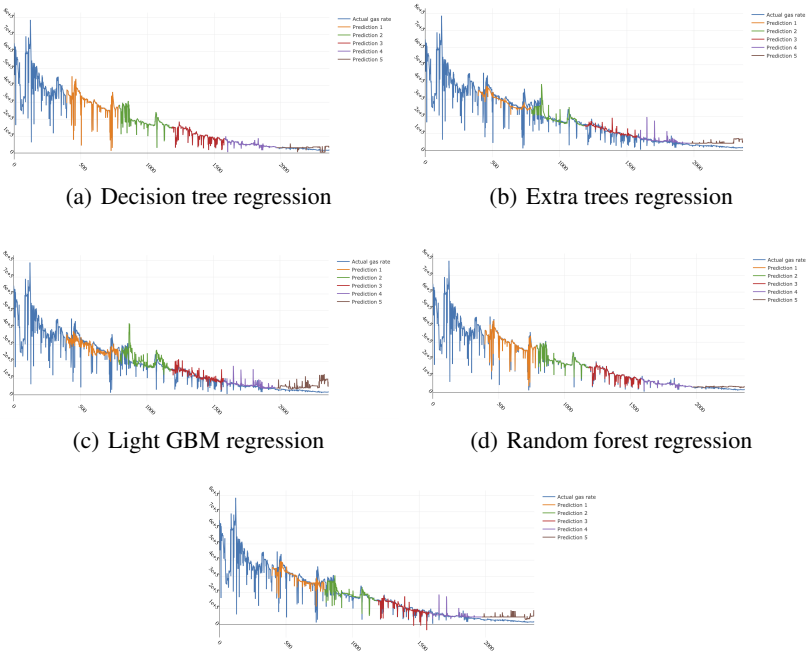
(b) Extra trees regression

(c) Light GBM regression

(d) Random forest regression

(e) Extreme gradient boosting regression

Figure 9: Oil prediction using time series cross-valdaion method.



(a) Decision tree regression

(b) Extra trees regression

(c) Light GBM regression

(d) Random forest regression

(e) Extreme gradient boosting regression

Figure 10: Gas prediction using time series cross-valdaion method.

While all machine learning algorithms we have used, data need to be shuffled before training, the dependence of production rate followd time have been vanished. Therefore, we continutes to use time series cross validation strategy on the same data set. Instead of using gradient boosting machine regressor, authors used LGBM and XGB algorithms because these algorithms have better result compare with GBM. Figure 9 and 10 have shown result of using time series cross validation method on oil and water. Despite there is a gap between actual values and prediction in the final year, other four-year period lines (orange, green, red and purple line) have give good matching with actual line. Decision tree and random forest regression give better result compare with other algorithms.

4.2 Deep learning

4.2.1 Neural networks

Table 3: Score and error on oil prediction

No	Model	Validation Score	Validation Error	Test Score	Test Error
1	Shallow neural networks	0.8563	0.0048	0.8426	0.00486
2	Deep neural networks	0.9492	0.001568	0.94	0.00199

Table 4: Score and error on gas prediction

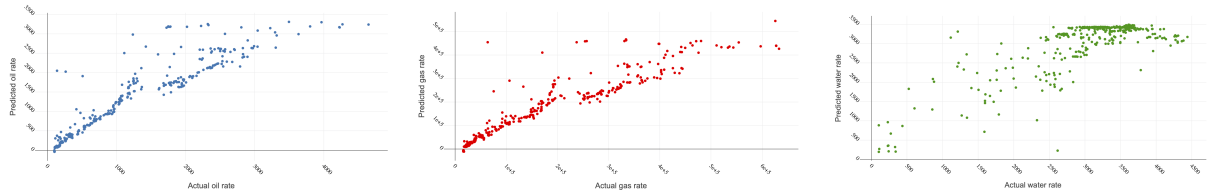
No	Model	Validation Score	Validation Error	Test Score	Test Error
1	Shallow Neural Networks	0.6826	0.0101	0.6068	0.00948
2	Deep Neural Networks	0.7824	0.00551	0.7767	0.00574

Table 5: Score and error on water prediction

No	Model	Validation Score	Validation Error	Test Score	Test Error
1	Shallow Neural Networks	0.6826	0.0101	0.6068	0.00948
2	Deep Neural Networks	0.7824	0.00551	0.7767	0.00574

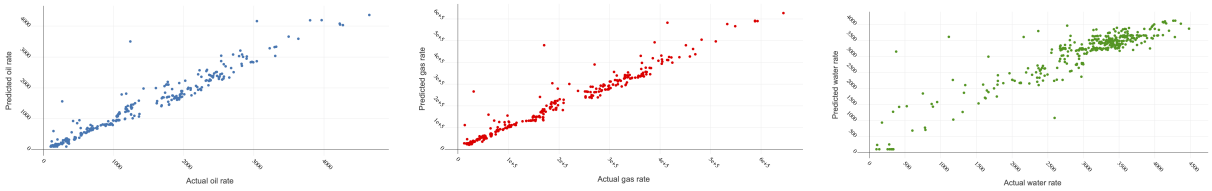
Table 6: Pearson correlation of neural networks

No	Model	Pearson correlation (oil)	Pearson correlation (gas)	Pearson correlation (water)
1	Shallow Neural Networks	0.9225	0.9372	0.8314
2	Deep Neural Networks	0.9798	0.9711	0.9014



(a) Oil prediction (b) Gas prediction (c) Water prediction

Figure 11: Production rate prediction using shallow neural networks.



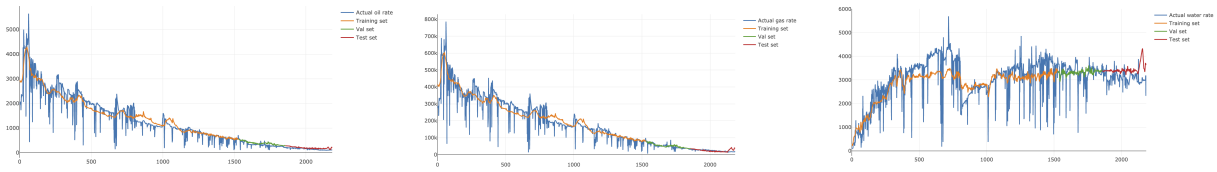
(a) Oil prediction (b) Gas prediction (c) Water prediction

Figure 12: Production rate prediction using deep neural networks.

Neural networks have been used widely in oil and gas industry from the time it is created. Compare with other machine learning algorithms, NNs have give better or worse result depend the process of traing. While shallow neural networks with one hidden layer (25 nodes) gives worse score and error is bigger, deep neural networks with three hidden layers give the increase the score and reduce error.
In this research, we have tuned hyperparametes of neural networks (table 8) to find the best comnination of them for giving good result on validation test. Moreover, we have use some very novel technique in AI, dropout for example and ReLU activation function, to train our neural networks. From the table above, since the increse of hidden layers may improve our result, but like other machine learning algorithms, data also need to be shuffled before training. That lead to the loss of temporal characteristic. This problem can be solves by using recurrent neural network.

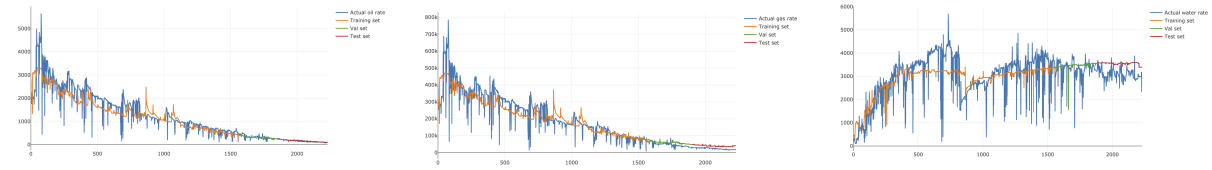
4.2.2 Recurrent Neural Networks

Recurrent neural networks algorithms are very powerful with time series data. We splited data set in three part train, validation and final year (15%) is the test set. Compare with machine learning models, RNNs have improve significantly when predicted in the last year. From figure 13, 14 and 15 above when using Recurrent neural networks (LSTM, GRU, SimpleRNN), a very close matching between prediction and actual values. Since GRU and LSTM gives best result on oil prediction, SimpleRNN is better with water forecasting.



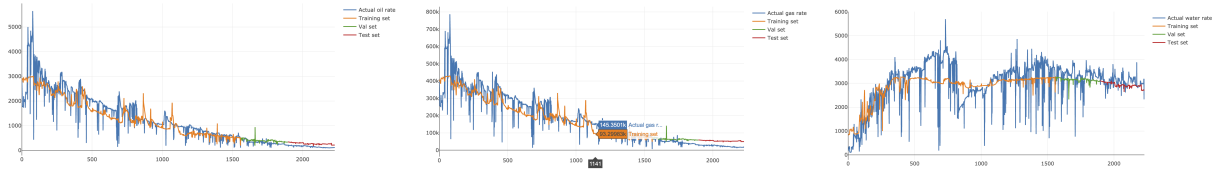
(a) Oil prediction (b) Gas prediction (c) Water prediction

Figure 13: Production rate prediction using long short term memory.



(a) Oil prediction (b) Gas prediction (c) Water prediction

Figure 14: Production rate prediction using gated recurrent unit.



(a) Oil prediction (b) Gas prediction (c) Water prediction

Figure 15: Production rate prediction using simple reccurent neural networks.

5 Conclusion

In this paper, we have applied some AI technique include machine and deep learning algotithms to predict the production rate of oil well in diffrent scenarios. Different algorithms have been used and comparion between them have showed that while machine learning gain good result in the early stage of production phase, recurrent neural networks have improved the prediction in the final stage of production phase (final year).
Without domain knowlegde in oil and gas industry and complicated physics based modeling, authors have prove that Artificial Intelligent can help us in production forecasting in effective way.AI has demonstrated their popential and could be a useful tools in oil and gas intrustry.
Futhermore, in the next study we can predict gas oil ratio (GOR), watercut, water gas ratio (WGR) and bottom hole pressure, or even make comparion between reservoir modeling and Artificial Intelligent.

References

[1] Cao, Q., Banerjee, R., Gupta, S., Li, J., Zhou, W., & Jeyachandra, B. 2016. Data Driven Production Forecasting Using Machine Learning Presented at SPE Argentina Exploration and Production of Unconventional Resources Symposium, Buenos Aires, Argentina, 1-3 June.

[2] Tita Ristanto. Machine Learning applied To Multiphase Production Problems. MS Thesis, Standford University, 2018.

[3] Boomer, R.J. (1995). Predicting Production Using a Neural Network (Artificial Intelligence Beats Human Intelligence). Society of Petroleum Engineers (SPE 30202).

[4] Suhag, A., Ranjith, R., and Aminzadeh, F. (2017). Comparison of Shale Oil Production Fore- casting using Empirical Methods and Artificial Neural Networks. University of Southern Cali- fornia. SPE ATCE (SPE-187112-MS).

[5] J. Sun, X. Ma, and M. Kazi (2018),CSE ICON. Comparison of Decline Curve Analysis DCA with Recursive Neural Networks RNN for Production Forecast of Multiple Wells SPE-190104-MS doi: SPE-190104-MS

[6] Al-Kaabi, A. U. and Lee, W.J. 1990. Using Artificial Neural Networks to Identify the Well Test Interpretation Model. Presented at the Petroleum Computer Conference, Denver, Colorado, 25-28 June. SPE-20332.

[7] Christopher M. Bishop Pattern Recognition and Machine Learning. Linear Model page 138-139

[8] Dominique Guillot Introduction to Data Mining and Analysis Decision trees. April 6, 2016.

[9] Ho, Tin Kam (1995). Random Decision Forests (PDF). Proceedings of the 3rd International Conference on Document Analysis and Recognition, Montreal, QC, 14–16 August 1995. pp. 278–282.

[10] D. O. Hebb. The Organization of Behavior: A Neuropsychological Approach. John Wiley & Sons, 1949.

[11] Y.Bengio, P.Simard, and P.Frasconi. Learning long-term dependencies with gradient descent is difficult. Neural Networks, IEEE Transactions on 5, no. 2 (1994): 157-166.

[12] Ian Goodfellow and Yoshua Bengio and Aaron Courville. Deeplearning text book. An MIT press book.

[13] Hastie, T. et al., (2001). The Elements of Statistical Learning. Springer New York Inc., New York, NY, USA.

[14] Pierre Geurts and Damien Ernst and Louis Wehenkel. Extremely randomized trees

[15] Friedman, J.H. (2011). Greedy Function Approximation: A Gradient Boosting Machine. Annals of Statistics. JSTOR.

[16] Sklearn Library. Machine learning in Python Journal of machine learning research, volume 12, 2825–2830, 2011.

[17] Chen, Tianqi and Guestrin, Carlos. XGBoost: A Scalable Tree Boosting System. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining 2016.

[18] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, Tie-Yan Liu. "LightGBM: A Highly Efficient Gradient Boosting Decision Tree". Advances in Neural Information Processing Systems 30 (NIPS 2017), pp. 3149-3157.

[19] Martens, James, and IlyaSutskever. Learning recurrent neuralnetworks with hessian-free optimization. In Proceedings of the 28th International Conference on Machine Learning (ICML-11), pp. 1033-1040. 2011.

[20] Understanding LSTM Networks

[21] Volve dataset by Equinor company

[22] Wikipedia feature engineering

[23] M. Korjani, Andrei Popa, Eli Grijalva, Steve Cassidy and I. Ershaghi. A New Approach to Reservoir Characterization Using Deep Learning Neural Networks.

[24] E. E. Okpo, A. Dosunmu, and B. S. Odagme, University of Port Harcourt Artificial Neural Network Model for Predicting Wellbore Instability.

[25] Srinath Madasu and Keshava P. Rangarajan, Halliburton Deep Recurrent Neural Network DRNN Model for Real-Time Multistage Pumping Data.

[26] Jung Chan Choi, Elin Skurtveit, and Lars Grande, NGI Deep Neural Network Based Prediction of Leak-Off Pressure in Offshore Norway.

[27] Tanaka, Takashi, Shinoda, Takeshi Kyushu University A Method for Extracting the Work Status in Shipyard Using Deep Neural Networks.

[28] Maarten V. de Hoop, Rice University Deep neural network architectures arising in seismic inverse problems.

[29] A.G. Mihajlov, S.S. Shubin, A.V. Alferov, R.N. Imashev, V.U. Yamaliev Improvement of efficiency of diagnostics of rod pumps with use of deep neural networks.

[30] Yunzhi Shi, Xinming Wu and Sergey Fome. Automatic salt-body classification using a deep convolutional neural network.

[31] Sarblund Haroon, Sergey Alyamkin, and Ramachandra Shenoy, AlphaX Decision Sciences LLC Application of Convolutional and Deep Neural Networks for GPU Based Seismic Interpretations.

[32] Naif Alqahtani, Ryan T.Armstrong, and Peyman Mostaghimi.
Deep Learning Convolutional Neural Networks to Predict Porous Media Properties.

[33] Zhining Liu, Chenyung Song, Bin She, Kunhong Li, Xingmiao Yao, Guangmin Hu. Visual explanations from convolutional neural networks for fault detection.

Nomenclature

$\mathbf{w}$	The learning paremeters for machine learning algorithms
$\mathbf{X}$	The input data for machine learning algorithms
y	Actual value of production rate
AI	Artificial Intelligent
BHP	Bottom Hole Pressure
BHT	Bottom Hole Temperature
CS	Choke Size (Percentage)
DCA	Decline Curve Analysis
DP	Different Pressure in casing
GRU	Gated Recurrent Unit
LGBM	Light Gradient Boosting Machine
LSTM	Long Short Term Memory
MLP	Multi Layers Perceptron
RNN	Recurrent Neural Networks
SEPD	Stretched Exponential Production Decline
THP	Tubing Head Pressure
WHT	Well Head Temperature
XGB	Extreme Gradient Boosting

Appendices

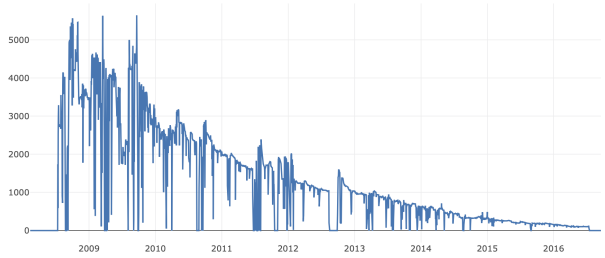


Figure 16: Data before feature engineering

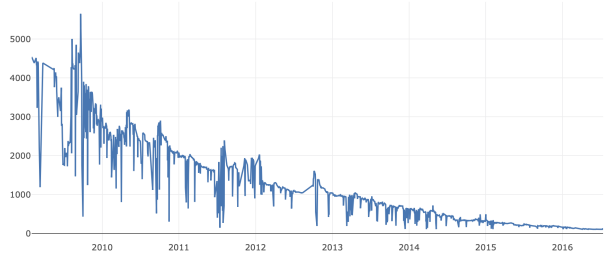


Figure 17: Data after feature engineering

Table 7: Hyperparametes to optimize machine learning algorithms

No	Model	Hyperparametes
1	Linear Regression	Nomalization
2	Ridge	alpha (0.05), solver('auto')
3	KNeighbors Regression	n_neighbors (3),
4	Support Vector Regression	kernel ('rbf), C (1e1) , gamma = 10
5	Decision Tree Regression	max_depth (10)
6	Random Forest Regression	n_estimators = 200, max_depth (8),,max_features('sqrt')
7	Extra Tree Regression	n_estimators=200, min_samples_split(25), min_samples_leaf(35)
8	AdaBoost Regression	n_estimators = 50
9	Gradient Boosting Regression	n_estimators(300), learning_rate (0.005),,max_depth(4)
10	XgBoost Regression	n_estimators(220), learning_rate(0.05), max_depth(3)
11	LightGBM Regression	n_estimators(720), learning_rate (0.05)

Table 8: Hyperparametes to optimize deep learning algorithms

No	Hyperparametes	Type
1	Activation function	tanh, ReLU, sigmoid, linear
2	Dropout (for DNN)	0.1 - 0.9, 0.4 , 0.5
3	Hidden layers (in NNs)	0 - 4
4	Hidden layer nodes	10 - 500, 8 ,20, 50
5	Loss function	MSE, MAE, RMSE
6	Metrics	MAE, MSE
7	Optimizer	SGD, LBFGS, Adagrad, Adam, RMSProp
8	Momentum	None, regular, Nesterov
9	Learning rate	0.0001, 0.001, 0.005, 0.01, 0.03, 0.3
10	Learning rate decay	0 - 1e-3, 1e-6
11	Regularization	None, L1, L2, EarlyStopping
12	Hyperparameter search	GridSearch, GridsearchCV

Table 9: Sample data for applying machine learning and deep learning

DATE	BHP	BHT	DP	CS(%)	WHP	WHT	OIL	GAS	WAT
1/30/09	257.44	105.34	163.29	35.30	94.15	61.05	4535.43	649388.07	298.19
2/11/09	261.48	105.36	164.35	34.70	97.13	65.80	4379.88	629307.34	143.54
2/20/09	264.39	105.41	166.21	34.78	98.17	64.99	4509.07	638750.17	108.74
2/22/09	266.71	105.40	166.27	34.05	100.44	67.33	4319.02	612912.62	106.60
2/23/09	266.67	105.41	166.51	34.40	100.15	66.99	4417.66	625514.01	117.37
2/25/09	269.71	105.36	166.93	31.05	102.78	69.95	3226.61	460948.01	118.99
2/26/09	268.34	105.43	167.28	34.31	101.06	67.85	4411.90	628668.27	134.19
2/27/09	268.75	105.44	167.40	34.31	101.35	68.19	4376.91	625510.25	152.76
2/28/09	269.14	105.44	167.85	34.28	101.29	68.16	4417.91	626562.21	106.72
3/1/09	269.56	105.45	167.93	34.32	101.63	68.48	4396.74	628354.12	155.97
3/2/09	270.00	105.46	168.02	34.24	101.98	68.79	4381.09	623678.16	163.39
3/7/09	281.30	105.22	166.24	24.94	115.06	82.89	2208.96	316638.28	104.01
3/11/09	273.08	105.14	166.64	25.03	106.44	73.73	1185.05	165437.35	174.74
3/24/09	242.68	105.20	159.26	36.77	83.42	50.54	4379.47	611263.11	207.66
5/14/09	222.23	105.19	153.78	41.27	68.44	35.29	4211.25	576830.95	104.06
5/16/09	223.00	105.22	154.26	41.29	68.74	35.58	4237.94	583860.51	104.19
5/17/09	223.10	105.23	154.59	41.33	68.51	35.36	4244.23	585468.54	104.30
5/18/09	223.29	105.25	154.64	41.33	68.65	35.50	3759.11	519316.88	127.10
5/19/09	223.37	105.27	154.79	41.32	68.58	35.44	3967.32	548750.04	110.41
5/20/09	223.58	105.29	155.00	41.31	68.58	35.44	3972.97	548982.34	148.53
5/21/09	223.79	105.30	155.31	41.33	68.48	35.32	3994.95	550388.29	153.09
…	…	…	…	…	…	…	…	…	…
…	…	…	…	…	…	…	…	…	…
6/17/16	269.73	100.07	238.84	44.94	30.89	5.37	103.29	16456.74	2979.58
6/18/16	269.68	100.08	238.86	44.99	30.82	5.31	103.44	16551.02	2972.75
6/19/16	269.65	100.09	238.87	45.02	30.78	5.27	103.55	16524.52	2984.83
6/20/16	269.60	99.81	238.88	44.90	30.73	5.20	103.75	16525.42	2978.42
6/21/16	269.61	100.11	238.88	44.92	30.73	5.20	103.44	16508.09	2992.62
6/22/16	269.58	100.12	238.96	44.95	30.63	5.12	103.21	16511.03	2993.41
6/23/16	269.55	100.13	238.94	44.93	30.61	5.11	103.27	16566.18	3022.90
6/24/16	269.57	100.14	238.85	44.80	30.72	5.19	103.37	16499.88	2980.02
6/25/16	269.52	100.15	238.85	44.81	30.67	5.15	102.57	16434.67	2981.04
6/26/16	269.48	100.16	238.84	44.86	30.65	5.11	102.85	16501.91	2983.48
6/27/16	269.76	100.17	238.90	45.11	30.86	5.34	101.46	16274.31	2976.42
6/28/16	269.88	100.17	238.91	45.13	30.97	5.45	102.26	16444.37	2973.66
6/29/16	269.83	100.19	238.94	45.20	30.89	5.37	102.38	16458.56	2980.12
6/30/16	269.83	100.20	238.89	45.16	30.93	5.40	101.47	16397.18	2985.45
7/1/16	269.89	100.20	238.91	45.16	30.98	5.32	102.89	16369.86	2979.13
7/2/16	269.79	100.22	238.87	45.18	30.93	5.39	101.58	16365.91	2992.00
7/3/16	269.79	100.23	238.83	45.11	30.96	5.44	102.43	16269.74	2976.87
7/4/16	269.78	100.24	238.82	45.08	30.96	5.44	100.67	16263.02	2990.10
7/5/16	269.77	100.25	238.80	45.08	30.97	5.43	101.88	16284.48	2974.30
7/7/16	266.20	100.31	238.44	93.42	27.76	2.19	106.19	17427.78	3172.96
7/8/16	266.04	100.33	238.47	100.00	27.56	2.00	106.30	17541.20	3187.95
7/9/16	268.81	100.30	239.08	82.19	29.73	4.11	102.09	16681.29	2326.24
7/10/16	265.92	100.34	238.40	100.00	27.52	1.96	113.38	18753.12	3185.47
7/11/16	267.77	100.32	238.64	91.16	29.13	3.41	108.84	17979.28	3056.29
7/12/16	266.00	100.35	238.27	100.00	27.73	1.94	113.84	18543.76	3148.91

Table 10: Description of data set

	BHP	BHT	DP	CS	WHP	WHT	OIL	GAS	WATER
Count	2365.0	2365.0	2365.0	2365.0	2365.0	2365.0	2365.0	2365.0	2365.0
Mean	249.09	102.09	209.74	78.48	39.36	82.10	1217.04	180380.41	2968.53
Std	14.47	3.35	22.44	24.82	12.61	2.23	1052.86	149356.18	934.56
Min	135.63	54.64	103.14	20.30	27.19	72.14	100.33	5928.54	100.16
25%	241.72	99.73	197.39	55.85	31.28	84.48	322.41	49848.46	2694.0
50%	247.56	101.14	208.75	97.92	32.84	87.20	950.38	151088.07	3203.3
75%	261.78	105.05	230.24	100.0	42.23	88.63	1876.97	285954.06	3541.83
Max	281.31	106.77	239.84	100.0	115.06	93.51	5644.37	786328.36	5691.77

Table 11: Score and error of machine learning algorithms on gas production

No	Model	Validation Score	Validation Error	Test Score	Test Error
1	Linear Regression	0.8539	0.00577	0.8011	0.00599
2	Ridge	0.8187	0.00619	0.8477	0.0057
3	KNeighbors Regression	0.9323	0.00231	0.9741	0.000967
4	Support Vector Regression	0.8610	0.00474	0.8817	0.00442
5	Decision Tree Regression	0.9452	0.00187	0.9625	0.0014
6	Random Forest Regression	0.9652	0.00155	0.9545	0.0013
7	Extra Tree Regression	0.9506	0.00168	0.9821	0.00068
8	AdaBoost Regression	0.9056	0.00322	0.9195	0.00301
9	Gradient Boosting Regression	0.9685	0.00112	0.9474	0.00188
10	XgBoost Regression	0.9256	0.00253	0.955	0.00168
11	LightGBM Regression	0.8980	0.00355	0.9496	0.00204

Table 12: Score and error of machine learning algorithms on water production

No	Model	Validation Score	Validation Error	Test Score	Test Error
1	Linear Regression	0.6133	0.0101	0.5228	0.013
2	Ridge	0.599	0.0105	0.523	0.013
3	KNeighbors Regression	0.9032	0.00254	0.8889	0.00303
4	Support Vector Regression	0.8501	0.00394	0.8387	0.00477
5	Decision Tree Regression	0.881	0.00313	0.7822	0.00595
6	Random Forest Regression	0.9197	0.00211	0.8904	0.00299
7	Extra Tree Regression	0.9057	0.00248	0.9015	0.00268
8	AdaBoost Regression	0.7418	0.00679	0.7473	0.0069
9	Gradient Boosting Regression	0.9071	0.00244	0.9124	0.00239
10	XgBoost Regression	0.8819	0.0031	0.8856	0.00312
11	LightGBM Regression	0.8481	0.00399	0.8591	0.00385

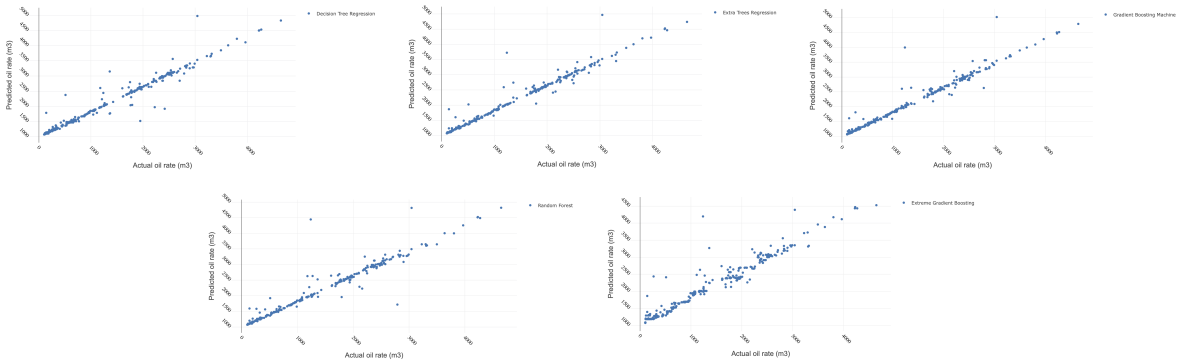


Figure 18: Best five model on oil prediction

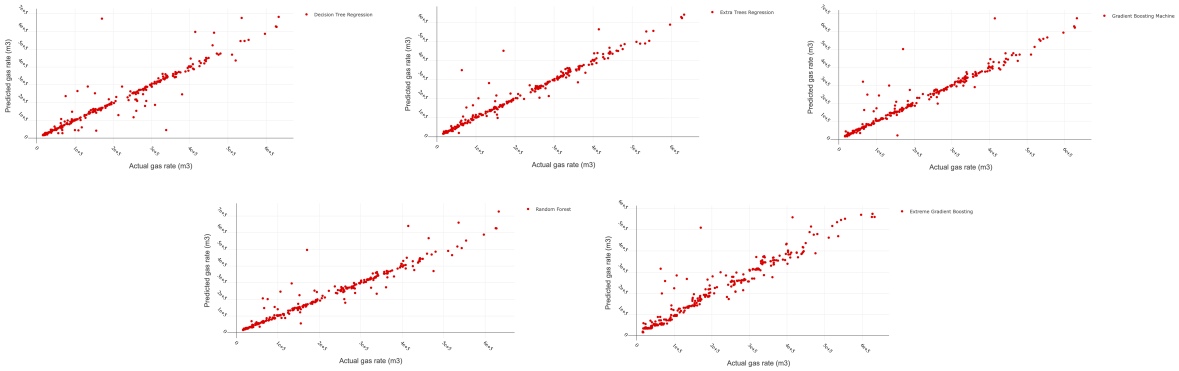


Figure 19: Best five model on gas prediction

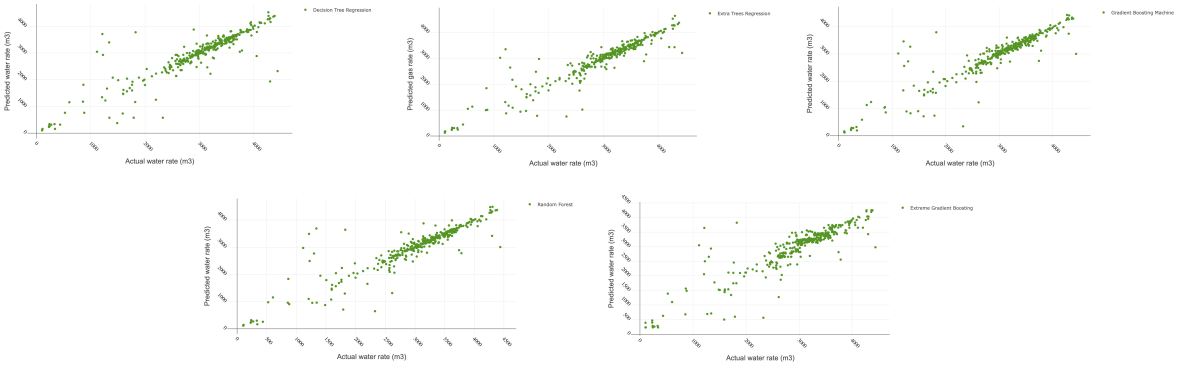
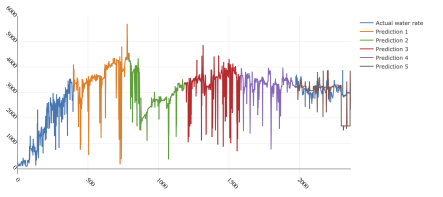
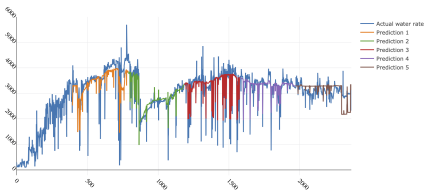


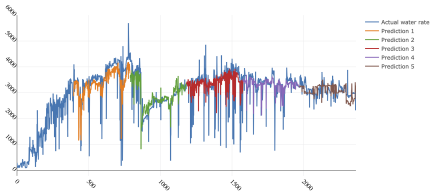
Figure 20: Best five model on water predict



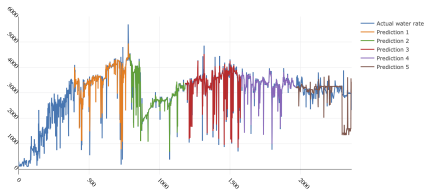
(a) Decision tree regression



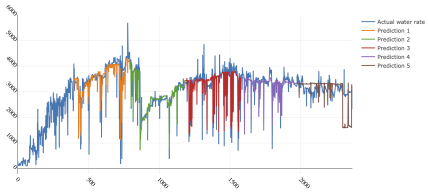
(b) Extra trees regression



(c) Light GBM regression

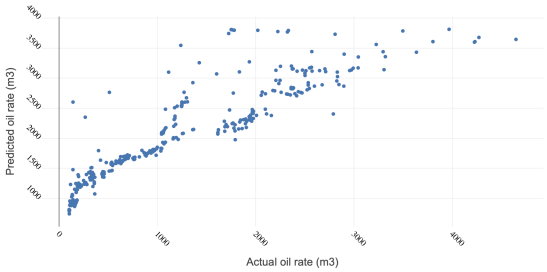


(d) Random forest regression

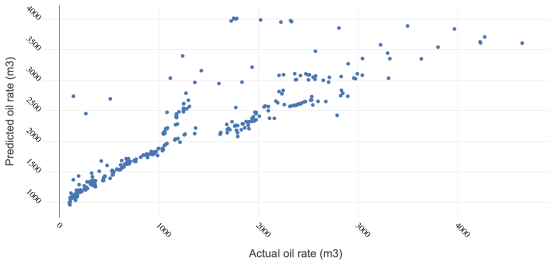


(e) Extreme gradient boosting regression

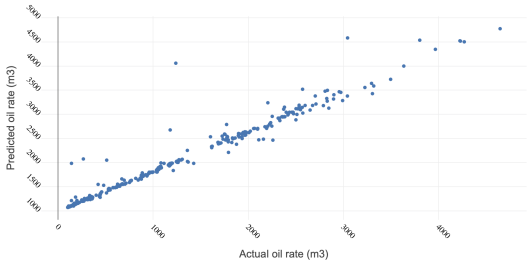
Figure 21: Water prediction using time series cross-valdaion method.



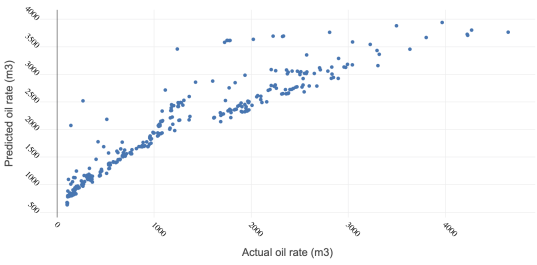
(a) Oil prediction using linear regression



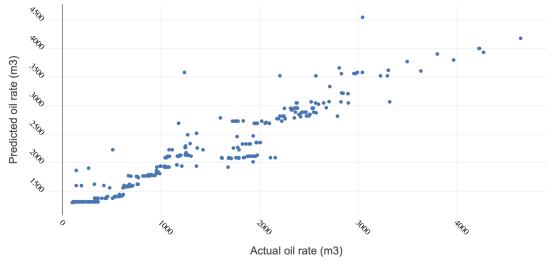
(b) Oil prediction using ridge



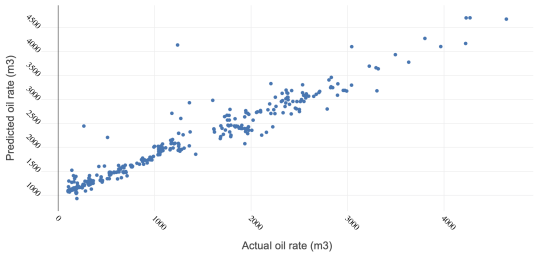
(c) Oil prediction using Kneighbors regression



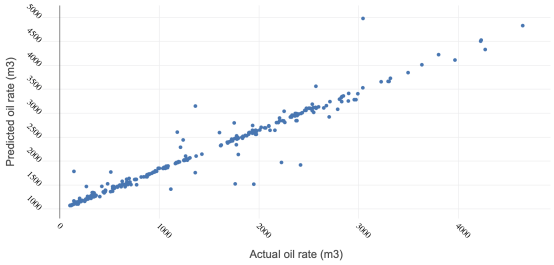
(d) Oil prediction using support vector regression



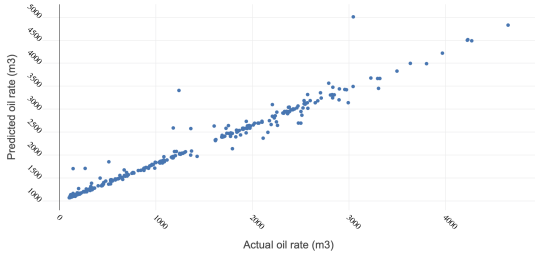
(e) Oil prediction using Ada boost regression



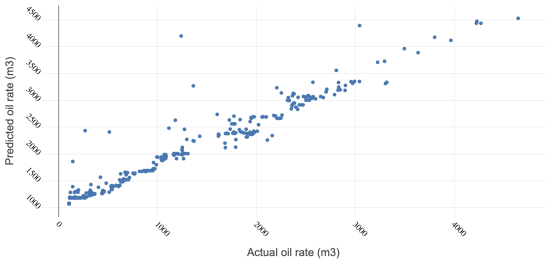
(f) Oil prediction using light gradient boosting machine



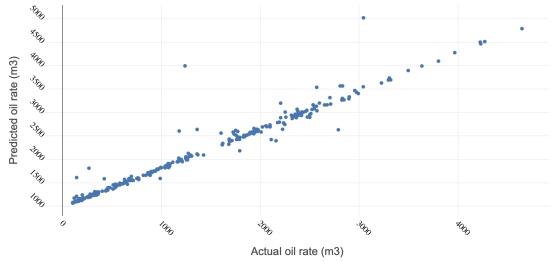
(g) Oil prediction using decision tree regression



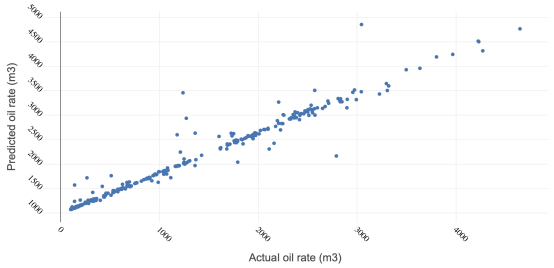
(h) Oil prediction using extra trees regression



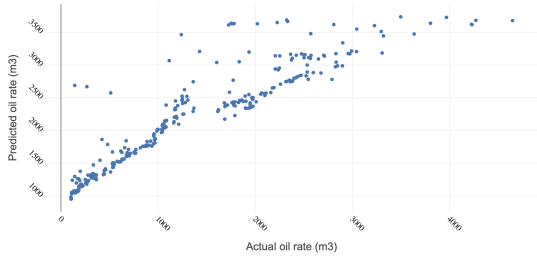
(i) Oil prediction using extreme gradient boosting machine



(j) Oil prediction using gradient boosting machine



(k) Oil prediction using random forest regression



(l) Oil prediction using multi-layers perceptron