Please fill in the name of the event you are preparing this manuscript for.	SPE Asia Pacific Oil & Gas Conference and Exhibition 2022
Please fill in your 6-digit SPE manuscript number.	SPE-210769-MS
Please fill in your manuscript title.	Applying Data Analytics & Machine Learning Methods for Recovery Factor Prediction and Uncertainty Modelling

Please fill in your author name(s) and company affiliation.

Given Name	Middle Name	Surname	Company
Munish		Kumar	ERCE (Singapore)
Kanna		Swaminathan	ERCE (Singapore)
Aizat		Rusli	ERCE (Malaysia)
Abel		Thomas-Hy	ERCE (Australia)

Abstract

The estimation of recoverable hydrocarbons, or field recovery factor (RF), is a critical process for Oil and Gas (O&G) companies to plan and optimise field development, manage ongoing production and identify profitable investments amongst other technical and commercial decisions. However, RF remains one of the greatest uncertainties in O&G projects.

The difficulty in RF prediction arises due to the number of variables affecting the recovery from a reservoir. These includes variables that are both uncertain and beyond the control of O&G operators, such as fluid flow in microscopic pores, as a function of fluid and rock properties, and variables which are engineering design based, such as completion methods, secondary and tertiary recovery mechanisms. In early field life, insufficient production data coupled with subsurface uncertainty makes RF prediction uncertain, and it is often the experience of the operator combined with analogue studies that is used to determine RF. However, there may be instances where operators may have insufficient data from analogue fields to properly capture the uncertainty in the RF range.

Utilising techniques of big data manipulation and machine learning (ML), two open-source, United States based data sets are (a) deconstructed to identify the key variables impacting the ultimate recovery of a field, and (b) used to create a ML model to predict the RF based on these key variables. These two datasets (the onshore Tertiary Oil Recovery System (TORIS), and the offshore Gulf of Mexico (GOM)) consist of over 1,000,000 real world data points.

Employing a low code environment, we test the predictive ability of 20 different ML algorithms by comparing predictive error. Decision tree type models (Random Forest and Category Boosting) show the best results. The paper shows comparison to a distance based (K Neighbour) model as well.

The work aims to show that not all variables influence RF equally and that any ML model should therefore be built with variables that have the greatest influence on RF yet have the lowest pairwise correlation. The influence of these input variables differs, depending on the implemented ML model.

The paper demonstrates the predictive ability of ML models is strongly dependent on the input dataset. Predicting the recovery factor of fields within the TORIS and GOM databases, the R^2 values are 0.81 and 0.88 respectively. Testing the algorithm on three additional fields outside of the two datasets, and in different geological provinces showed errors of up to 10-15%.

Introduction

Recovery Factor (RF) is one the most critical inputs towards determination of field resources. Despite its significance, there is no clear approach to calculate or estimate RFs, as a variety of factors govern its value. While empirical correlations exist, they fail to adequately consider the plethora of variables that can impact the final RFs. Therefore, some operators choose to evaluate RF via a combination of technical parameters, analog databases or industry standard empirical correlations. This challenge of evaluating a suitable RF is exacerbated in early field life when there is insufficient production data and significant subsurface uncertainty. In early field life, there is often no alternative other than analogs and operators' experience to predict RFs. It is therefore unsurprising that the prediction of RF has considerable uncertainty, with wider ranges of RF utilised.

In actuality, a good RF estimate would be one that considers both qualitative and quantitative parameters. With the advent of big data, cheap memory options, and fast computing, applying a machine learning (ML) and artificial intelligence (AI) methodology to predict RF seems plausible. Indeed, such an approach would allow one to use to not only consider "hard" engineering data like connate water saturation (Swc) or permeability (k) but also "soft" geologic descriptors like trap type and lithologies as well. A working hypothesis would be that the more data is considered early on, the better the RF prediction would be. Another advantage of a ML/AI based method would be the ability to probe and investigate the sensitivities that can impact RF when variables are varied or modified. Finally, an ML/AI model is adaptive because it can be rebuilt quite quickly based on the availability of new data. This flexibility is sometimes more valuable than rigid empirical models, especially when unique field types are encountered.

Given this understanding, this paper will demonstrate a workflow where we successfully apply ML techniques to predict the RF of an oil reservoir, and where, in general, usage of a ML model in RF prediction outperforms conventional solutions obtained from empirical reservoir engineering methods.

We postulated that different machine learning models would treat input variables differently, and with varying importance. Since we do not know apriori which variables are of greatest importance to the different ML algorithms, we needed a rapid prototyping approach, and we finally opted for the use of "low-code" libraries which aided not only the latter but also simplified deployment of the final model.

We will also highlight the importance of the training data to the predictive power of the model, by using 2 unique training data sets with different variables. We learn that while both sets (and therefore models) appear plausible, applying experience and domain knowledge informs us that the RF results in some cases are inaccurate. Use of a more robust training dataset would address and mitigate this inaccuracy.

Literature Review

The empirical prediction of RF has been a long-term challenge in reservoir engineering since the 1940s. Early studies often used empirical methods to constrain the bounds of RF. Once such study by Guthrie and Greenberger (1955) was based on 73 water flooded, sandstone reservoirs, and developed empirical correlations that linked RF to fundamental rock and fluid properties as given by equation (1):

$$RF = 0.272\log(k) + 0.256S_{wc} - 0.136\log(\mu_{oi}) - 1.538\emptyset - 0.0003h + 0.114 \qquad (1)$$

where k is permeability (mD), S_{wc} is connate water saturation (frac.), μ_{oi} is oil viscosity (cP) at initial conditions, $\emptyset$ is porosity (frac.) and h is formation thickness (ft). Arps et al. (1967) also developed correlations based on 312 reservoirs, considering different drive mechanisms and utililised some additional parameters and as outlined in equation (2):

$$RF = 0.549\left[\frac{\emptyset(1-S_{wc})}{B_{oi}}^{A}\left(\frac{k\mu_w}{\mu_{oi}}\right)^{B}(S_{wc})^{C}\left(\frac{P_i}{P_a}\right)^{D}\right] \qquad (2)$$

where B_{oi} is $o$il formation volume factor (stb/rbl), k is permeability (D), μ_w is water viscosity (cP), P_i is initial reservoir pressure (Psia) and P_a is abandonment pressure (Psia). The constants A, B, C and D vary depending on the drive mechanism; in water flooded reservoirs, A = 0.0422, B = 0.077, C = 0.1903 and D= -0.2159.

In 1965, Gordon E Moore predicted that the number of transistors on microchips would double every two years, and yet the cost of computing itself would half. Moore's law (as its colloquially known) is now a proven tenet in modern computing (Moore & Gordon, 1965), and it had greatly benefited the domain of ML. ML solutions were first introduced to the petroleum industry in the early 2000s. Sharma et al. (2010) had performed a study where open-source datasets were used to predict RF. Sharma performed clustering analysis followed by linear regression models within the identified clusters to predict the recovery factor. However, only 24 rows of data/ reservoirs were used after data cleaning was performed on the dataset. This was in view of keeping as many features as possible. Due to the scarcity of the data points used, the model did not achieve a good fit on the test data set but outperformed conventional correlations such as Arps. Also, the model was not accessed on a blind dataset. Ahmed et al. (2019) developed a Neural Network model to predict recovery factors for oil sandstone reservoirs with water drive on a proprietary dataset. The dataset was trained on 130 datapoints (reservoirs) using 10 features which were all numerical with no geological feature and was tested on 38 datapoints. The model achieved good results with a correlation coefficient, (R^2) of 0.94 on the dataset and outperformed the conventional empirical correlations. Makhotin et al. (2021) developed regression models based on decision trees using a combined dataset that included a public dataset and an internal proprietary dataset for fields located worldwide. First, the dataset was split into pre-, and post-production groups and models were developed for both groups. Then, a clustering analysis was performed before developing regression models within the clusters similar to Sharma et al. (2010). Results achieved for the pre-production group was less accurate compared to the results for the post-production group as the input parameters for the post-production group is much larger.

We aim to expand on the work done by the above authors by (a) looking at larger, more varied data sets and using more parameters (including geological parameters) (b) utilizing "low-code" techniques to rapidly prototype, test and depot multiple ML models at once, (c) applying boosting and bagging to see if this improves ML model performance and (d) understand how ML codes perform against analytical solutions such as Arps.

Datasets

Two open-source datasets were selected, the Tertiary Oil Recovery System (TORIS) database (US Department of Energy, 1995) and the Gulf of Mexico (GOM) database (Bureau of Ocean Energy Management, 2019).

The TORIS dataset was initially developed by the National Petroleum Council (NPC) in 1984 to assess the U.S. EOR (Enhanced Oil Recovery) potential. The database has been continuously updated and the version used here is as of 1995 which was updated by the U.S. Department of Energy to evaluate technical and economic recovery potential of specific crude oil reservoirs. These reservoirs are located onshore U.S. and they include approximately 2,500 crude oil fields containing roughly 65% of discovered oil onshore U.S. Each oil field has 69 parameters consisting of both numerical and categorical.

The GOM database was initially collated by the Bureau of Ocean Energy Management (US) in 1999. It consists of Oil and Gas accumulations at sand level located in the US waters of the Gulf of Mexico and is updated on a yearly basis. The version used here is as of 2019 and consists of 13,395 sands from 1,316 fields. 860 fields out of 1,319 fields have been abandoned thus giving good certainty in the reported recovery factor.

Methodology

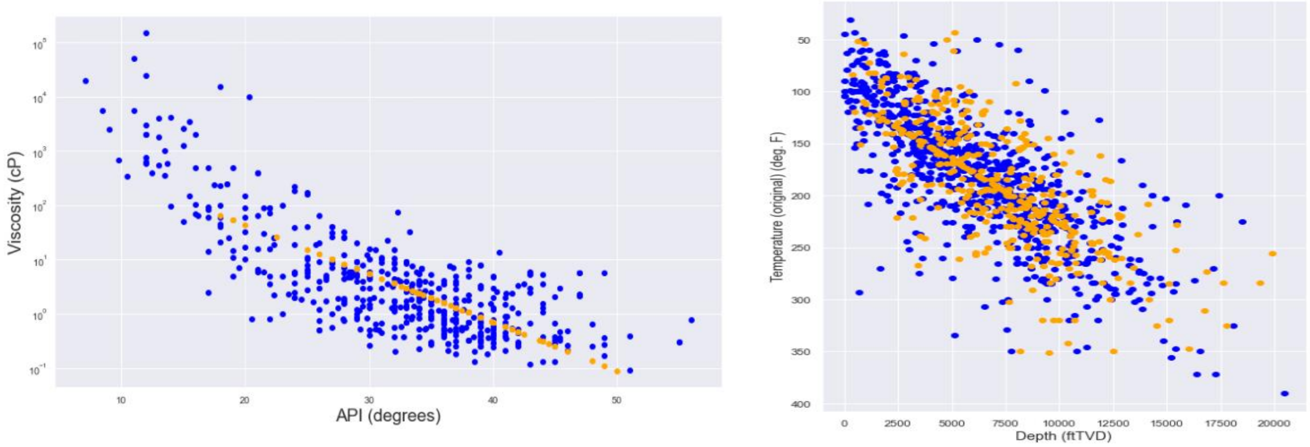
Our machine learning approach will be described in this section. The approach is divided into 5 stages: (A) data collection and data preparation, (B) machine learning model selection (C) model training and hyperparameter optimization, (D) model combination and finally (E) testing, deployment, and evaluation of model against our blind datasets (we will discuss (E) in the “Results and Discussion” section of this paper).

A. Data Collection and Preparation

Data collection and preparation can be group together under “Exploratory data analysis”. Exploratory data analysis is the initial investigation on the dataset to determine patterns and outliers through statistics and data visualization. Firstly, the attributes of the variables in dataset are defined. Statistics and data visualization are then obtained via univariate, bivariate and multivariate analysis. The previous step allows the identification of missing values, aberrant and outliers.

Data cleaning involves removing duplicates, fixing indexes, replacing incorrect characters, normalization of names, etc. Unlike conventional data science methodologies that only considers statistics, we have chosen to fill our missing values with values that consider correlations conventionally applied in industry. In this way, the created pseudo-data is grounded in a physical basis, rather than in a statistical domain. In fact, we are of the opinion that blending of data science with expert domain knowledge from learnings in reservoir engineering and petrophysics provides better results than just data science techniques alone.

. Examples of such correlations are pressure-depth or API-viscosity, where missing data can be filled by observing the behaviour from data which is present, shown Figure 1.



(a) (b)

Figure 1: Example to illustrate how missing data is filled using experience from domain knowledge (a) viscosity-API regression (b) Depth-Temperature regression

A correlation matrix is generated to provide information on any collinearity between the variables. If such pair-wise correlations exist, we remove one of each pair. This is done because no single predictor variable should excessively dominate the model, as this can lead to a potential model overfit or a highly unstable end model. Rather, changes in one of variable should not cause major fluctuations in the model.

A good machine learning models needs to be “generalized” somewhat, a model that varies significantly makes it difficult to distinguish between variables that have true significance in predictions.

Following initial data preparation, the final input variables used by the machine learning model are shown in Table 1 and The final GOM data set comprises of 2 categorical data types, and 13 numerical data types. The total data size is 3945 values per column, for a total of 59,175 data points (~5.5% of the original data base).

Table 2.

B. Machine Leaning Model Selection

The choice of the ML algorithm is not straightforward. Considerations include model interpretability, quantity of data point, required features, data format, data linearity, time taken to train and predict and finally the memory required. It is not a simple task to balance all the competing requirements (Masood et al., 2019).

A simpler way to address this is via the use of automated machine learning (Waring, Lindyall & Umerton, 2020) or low code machine learning libraries (Silipo, 2021). This significantly reduces the complexity of the problem by removing a lot of the front-end decision making, leaving application and data analysis as the key focus areas for O&G professionals.

We realized that the prediction of RF is fundamentally a regression type problem. This neatly falls under a supervised learning approach, where raw inputs are split into "train-validate-holdout" sets in a 70% - 20% - 10% ratio split. The "train" set would be used to build/train the model while the "validate" set would be used test the trained model. The holdout set is kept separate as a final blind test of the model.

C. Model Training and Hyperparameter Optimization

We initially utilized 20 different algorithms but finally opted for 3 based on the ranked results of the mean absolute error (MAE), the mean squared error (MSE), the root mean squared error (RMSE) and the coefficient of determination (R^2):

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |y_i - x_i| \tag{3}$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n} (y_i - x_i)^2 \tag{4}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (y_i - x_i)^2} \tag{5}$$

$$R^2 = 1 - \frac{\sum (y_i - x_i)^2}{\sum (x_i - \bar{x})^2} \tag{6}$$

where y_i is the predicted value, x_i is the true value, $\bar{x}$ is the mean of x_i and n is the total number of data points. Note that a good result is one that has a low MAE, MSE and RMSE, but a high R^2 (since this is essentially a measure of "goodness of fit").

The models were then tuned by varying "hyperparameters". A hyperparameter is a characteristic of a model that is external to the model and whose value cannot be estimated from data. The hyperparameters are optimized via a search algorithm with the goal of minimize the overall error metric.

To account for idiosyncrasies in the data (noise, patterns, outliers, etc.), k-fold cross-validation was run to validate the stability of the model.

D. Model Combination

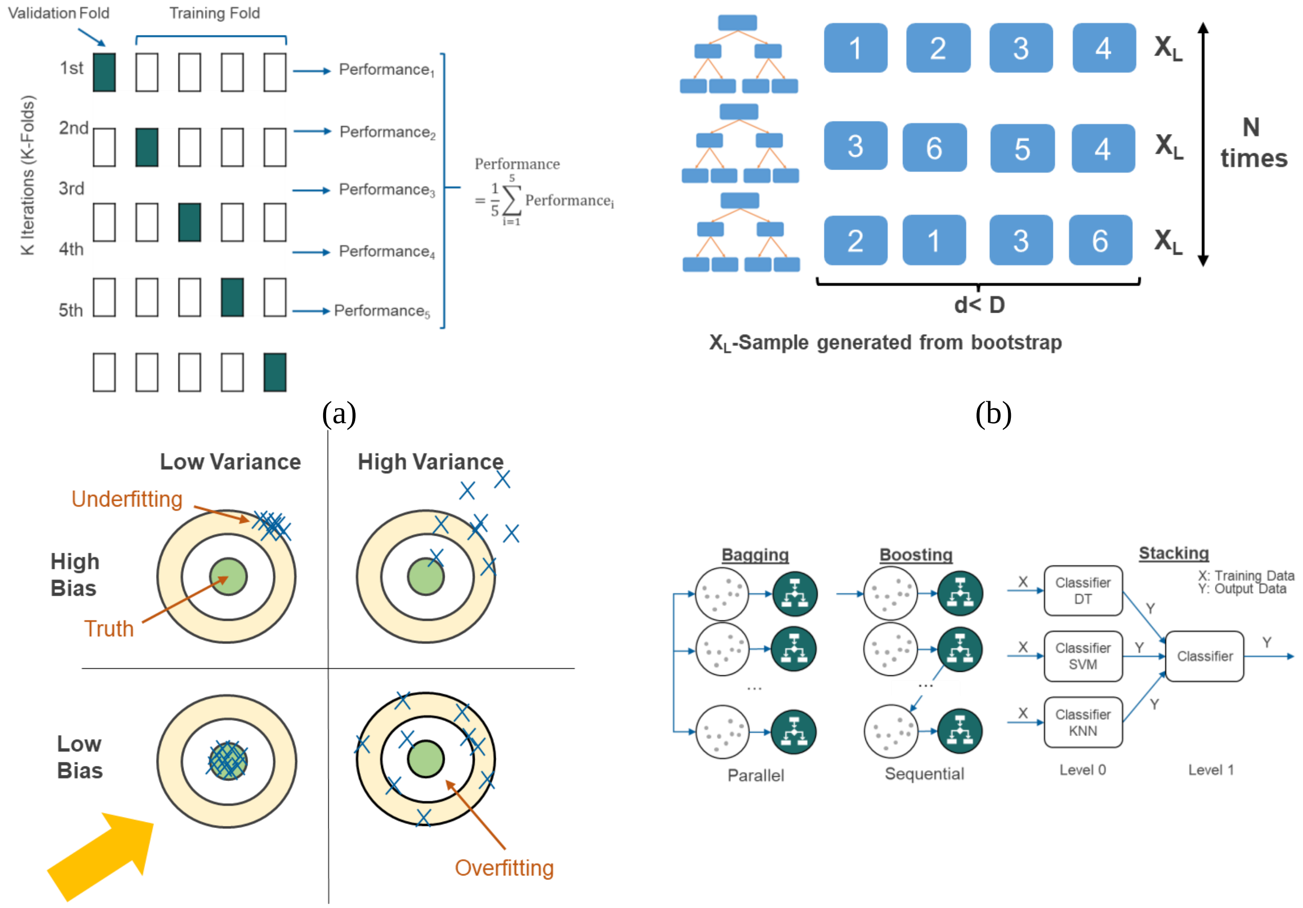
Aside from the utilisation of standard ML algorithms over the entire data set, we improved on the results using "bootstrapping" in our solutions, where we split our training data into numerous small sets (schematically illustrated in Figure 2a). For each of these k-folds, we create a machine learning model for the data subset (in our schematic, we use the example of a decision tree), giving N learned models (X$_L$), which we can then combine to form an improved model that is robust to overfitting (Ying, 2019) and outliers (Figure 2b).

This process of combining different interpreted models is known as “ensemble averaging”, with the end goal of creating a model that is a “strong learner” and which has the lowest variance and/or bias (Figure 2c).

Dependent on the algorithm class, the combination can be via “bootstrap aggregating” (bagging), “boosting” or “stacking” (Figure 2d). In bagging, (homogeneous multiple models known as) “weak learners” of the same type are trained to solve the same problem, with the aim of reducing the variance. In this case, each “weak learner” is independent of one another and is making predictions with the same variables, but on a new a random subset from the full data set, with the result being averaged deterministically to create the robust model (aka the strong learner).

Boosting follows a similar approach to bagging, with the key difference being that boosting aims to reduce bias, by having “weak learners” learn sequentially from its predecessors. We used a technique known as “adaptive boosting” (Adaboost) to update the weights attached to each new training dataset observation, based on observations from previous models that performed poorly.

The final method applied is stacking. Unlike bagging and boosting, however, stacking combines different (heterogeneous) “weak learner” base models to create a novel meta-model for predicting outputs. As an example, a decision tree could be combined with a random forest and a K-nearest neighbor regressor to create a hybrid (meta-model) of the 3 beforementioned base models. Each base model output is then fed as an input into the meta-model, and a prediction is generated.



(a) (b)

(c) (d)

Figure 2: Schematic illustration of the process of (a) K-fold cross validation, (b) bootstrapping, (c) bias and variance and (d) bagging, boosting and stacking.

E. Testing, Deployment, and Evaluation of model against our blind datasets

The testing and deployment of the model would be on the “hold-out” set. This third set of data would be data never-before been seen by the ML algorithm. To further test the predicative capabilities of our model, we additionally insert information for 3 fields, with known properties based on our own in-house interpretation, done from first principles. These 3 fields we treat as a “double-blind” test for the predictive power of the model. We additionally test the ML results against those obtained from conventional reservoir engineering principles; namely via the Arps and Guthrie empirical relationships.

Results

Final TORIS and GOM Data Sets

As expressed in the “Methodology” section, the two databases’ TORIS and GOM were first analysed and cleaned. Both datasets were tested for (multi) collinearity and the evaluated correlation matrices are shown in Figure 3 (TORIS, left; GOM, right). For purposes of this work, we define highly correlated values as having coefficients larger than 0.7.

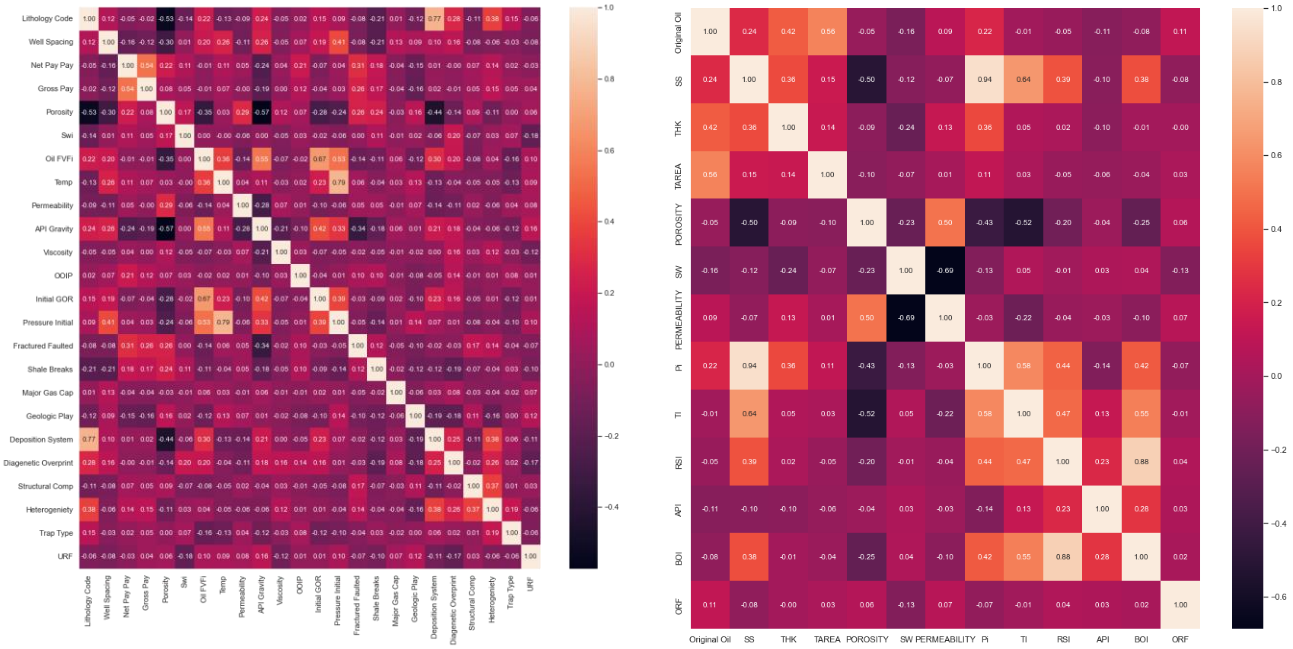


Figure 3: Correlation Matrix – Pairwise Correlation (TORIS, left; GOM – right)

We observed that for the TORIS database, “lithology code” and “depositional system” show collinearity, as did the “Temperature-Pressure” pair. For the GOM database, “Pressure-Depth” and “Gas Oil Ratio-Formation Volume Factor” were collinear. The relationship between these variable pairs aligns with conventional reservoir engineering concepts. As keeping both parameters will not add additional information to the predictive model and may potentially result in an overfit within the model, a single element from each of the variable pairs is eliminated to allow for a more stable model.

The final input variables were determined and summarized in Table 1 for the TORIS dataset and The final GOM data set comprises of 2 categorical data types, and 13 numerical data types. The total data size is 3945 values per column, for a total of 59,175 data points (~5.5% of the original data base).

Table 2 for the GOM dataset.

The TORIS dataset comprises 10 categorical data types, and 14 numerical data types. The total data size is 389 values per column, for a total of 9,336 data points which was approximately ~9.6% of the original data base.

Table 1: Statistics of the TORIS Database Input Variables

	count	mean	std	min	0.25 percentile	0.5 percentile	0.75 percentile	max
Lithology Code	389	1	1	1	1	1	2	3
Well Spacing	389	36	48	1	10	20	40	640
Net Pay	389	105	169	5	24	50	122	2300
Gross Pay	389	266	356	10	50	150	300	2300
Porosity	389	19	8	3	12	17.6	25	51
Swi	389	31	10	10	25	30	36	68
Oil FVF	389	1	0	1	1.099	1.2	1.33	2.127
Temp	389	138	44	63	105	130	164	266
Permeability	389	401	1507	0.1	10	52	300	26816.5
API Gravity	389	32	9	6	27	34	38	52
Viscosity	389	387	10468	0.07	0.81	2	7	200000
OOIP	389	294	1210	21	48	884	210	22000
Initial GOR	389	516	472	5	200	421	687	4000
Initial Pressure	389	2215	1368	200	1250	1850	2900	9500
Fractured Faulted	389	0	0	0	0	0	1	1
Shale Breaks	389	1	0	0	0	1	1	1
Major Gas Cap	389	3	46	0	0	0	0	680
Geological Play	389	623	713	6	44	414	830	2417
Deposition System	389	184	38	110	152	181	222	270
Diagenetic Overprint	389	2	2	1	1	1	3	9
Structural Complexity	389	14	9	10	10	10	10	50
Heterogeneity	389	1	1	1	1	1	2	3
Trap Type	389	2	1	1	2	2	3	3
URF	389	0	0	0.024	0.25	0.311	0.4	0.5073

The final GOM data set comprises of 2 categorical data types, and 13 numerical data types. The total data size is 3945 values per column, for a total of 59,175 data points (~5.5% of the original data base).

Table 2: Statistics of the GOM Database Input Variables

	count	mean	std	min	0.25 percentile	0.5 percentile	0.75 percentile	max
Chronozone	3945	-	-	-	-	-	-	-
Original Oil in Place	3945	4835198.0	17000000.0	24.0	221599.0	888893.0	3274200.0	560000000.0
Depth Sub Sea	3945	9895.0	4166.2	1350.0	7017.0	9368.0	11700.0	30800.0
Pay Thickness	3945	25.2	22.7	1.0	12.0	19.1	30.4	325.2
Area	3945	692.6	1215.9	1.0	110.0	300.0	765.0	20219.0
Drive Mechanism	3945	-	-	-	-	-	-	-
Porosity	3945	0.29	0.03	0.10	0.27	0.29	0.31	0.38
Water Saturation	3945	0.28	0.09	0.10	0.22	0.27	0.33	0.68
Permeability	3945	428.3	467.9	1.0	133.0	267.0	547.0	3898.0
Initial Pressure	3945	5815.8	3319.5	643.0	3577.0	4954.0	6986.0	21609.0
Initial Temperature	3945	182.7	37.7	47.3	156.3	182.3	207.3	305.3
GOR	3945	1073.1	628.7	100.0	681.0	921.0	1265.0	5000.0
API	3945	33.3	4.6	11.0	30.0	34.0	36.0	60.0
BOI	3945	1.5	0.3	1.0	1.3	1.4	1.6	3.3
ORF	3945	0.3	0.1	0.0	0.2	0.3	0.4	0.5

Initial Machine Leaning Model

When ranking the 20 models using MAE, MSE, RMSE and R^2, the best 3 approaches found were Category Boosting (CatBoost), the Random Forest (RFR) and the K Neighbours (KNN). The results for each model are tabulated in Table 3.

The CatBoost algorithm (Freund & Schapire, 1996) can handle categorical as well as numeric data, doing so without conversion of categorical to dummy variables. The RFR algorithm (Breiman, 2001) is insensitive to outliers, able to work on large number of variables and is robust against overfitting. However, it does require a lot of CPU memory. Both CatBoost and RFR are, different classes of decision trees (DT). As we wanted an alternative solution that did not depend on a DT type approach, our evaluation lead to the KNN regressor (Atlman, 1992), whose results are distance based The algorithm predicts outcomes based on how closely they match points in the training set. The distance methods considered include the Euclidian and Manhattan distances (for points that are continuous) or the Hamming distance (for categorical data).

With reference to Table 3, RFR achieved the lowest error values for the TORIS data set, while the CatBoost was the lowest for the GOM data set. In all cases, KNN produced the highest values of error.

Table 3: Results of first pass ML models

Regressor Model	TORIS Dataset				GOM Dataset			
	MAE	MSE	RMSE	R^2	MAE	MSE	RMSE	R^2
Random Forest	0.0791	0.0096	0.0976	0.0949	0.0790	0.0096	0.0978	0.5198
Category Boost	0.0821	0.0104	0.1015	0.0201	0.0708	0.0082	0.0906	0.5281
K Neighbours	0.0830	0.0109	0.1037	-0.0153	0.1091	0.0179	0.1339	0.1022

A ranking of the top 10 variables of importance using the TORIS model as an example is provided in Figure 4, to illustrate how different algorithms rank different parametric inputs. We also show the results of the variable ranking generated by a gradient boost regressor (GBR), and an “extra tree” (ET) algorithm, which are other examples of DT based algorithms.

First, we note that the RFR, CatBoost and GBR algorithms consistently rank API gravity, permeability, and initial water saturation as the top 3 categories. This is in-line with conventional reservoir engineering intuition in RF determination. The ET algorithm, however, shows a totally different and unique ranking system. What was interesting in our investigation, however, was that categoric data was shown to be of lesser smaller importance (it was outside the top 10 features of importance), even if using the CatBoost algorithm. This might be the case of the unique properties of the TORIS and GOM data sets, where the model need not rely heavily on categorial features to form a final prediction.

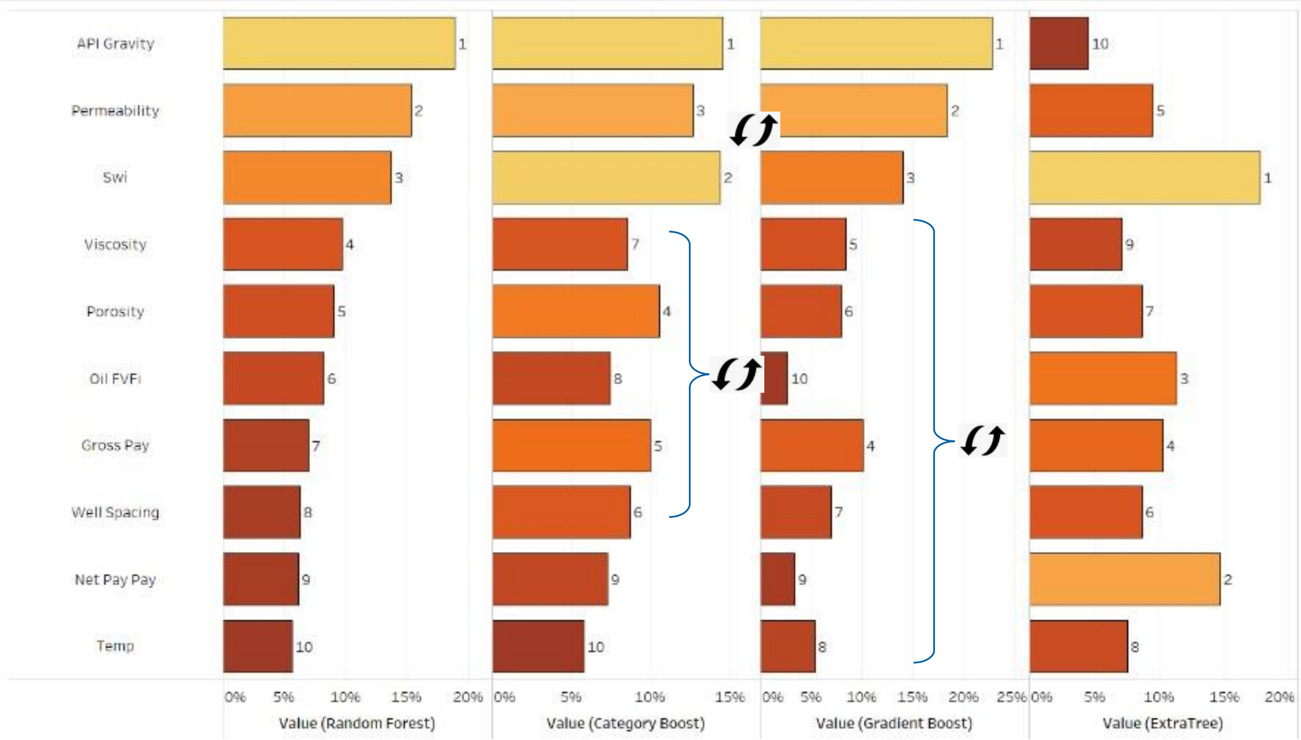


Figure 4: Ranks of Top 10 Features – TORIS Model

Tuning and Optimizing the Model

Having narrowed the choices of ML models down to CatBoost, RFR and KNN, the next step involved improving the results of the first pass modelling by using the techniques described in the previous section. Bootstrapping, via k-fold cross-validation, followed by bagging, boosting and/or stacking were used to help reduce the error.

K-fold cross-validation was tested with 5, 10 and 15 folds at the default hyperparameter values. Our experiments showed that RFR performs better at 5 K-folds while the other models perform better at 10 K-folds. However, the improvement was marginal (< 1%). For simplicity, we opted to use 10 folds for all models. When tuning the model, it was observed that optimization plateaus and more runs or more cross-validation will not reduce the error further.

Once the number of folds were determined, the hyperparameters were investigated. Each hyperparameter is given a range of values, and over a certain number of iterations (n_{iter}), different combinations of these values are used to train the model. For instance, n_{iter} = 10 means the hyperparameters are randomly varied 10 times per model, before stopping and reporting the result. It is tempting therefore to increase n_{iter} to as high a value as possible.

However, our experiments revealed that solutions tend to converge rapidly at $n_{iter} = 50$, beyond which no further uplift in results is observed. We performed experiments at $n_{iter} = 10$, $n_{iter} = 50$, $n_{iter} = 100$ and $n_{iter} = 1000$. We observed that while more iterations improve MAE (which is the metric we were using for optimisation) from ~9.12 to ~9.07., our run-time for these experiments increased by ~1900% (from 180s to 3600s), as shown in Figure 5. We finally concluded that $n_{iter} = 50$ achieves that sweet spot of good error minimization that avoids the local minimum, while ensuring that run times remain reasonable. The outcome of this approach are the "tuned" CatBoost, RFR and KNN models. We finally apply bagging, boosting and stacking to each of the tuned models to see if we can further improve the error scores.

A summary of the finalized range of inputs is given in Table 4 for the TORIS data set as an example; we only show the results of what inputs needed to be varied in the model to obtain the most optimized R^2. Table 5 demonstrates the impact of tuning on the CatBoost model, with an error comparison Pre- and Post-Tuning.

Our final outputs are 4 different optimised models for both TORIS and GOM data sets (i.e. total of 8 model). For TORIS, the 4 models are (a) Tuned and Bagged KNN (TBKNN), (b) Blended 5 K-fold (B5K), (c) Random Forest (RFR) and (d) Tuned & Bagged CatBoost (TBC). For the GOM data set, the 4 models are (a) CatBoost (CB), (b) Tuned Bagged CB (TBCB), (c) Stacked 10 K-fold (S10K) and (d) RFR.

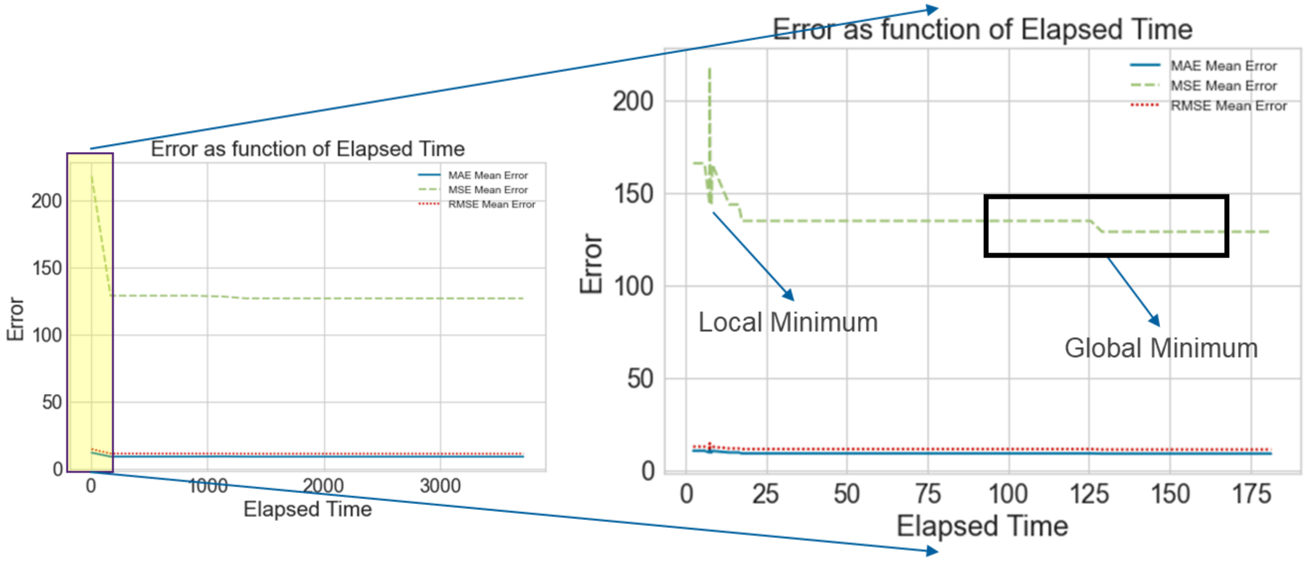


Figure 5: Impact of n_{iter} on error minimization and run-time (L) zoom out showing minimum impact at long time scales (R) zoom in showing convergence after t~125-150s

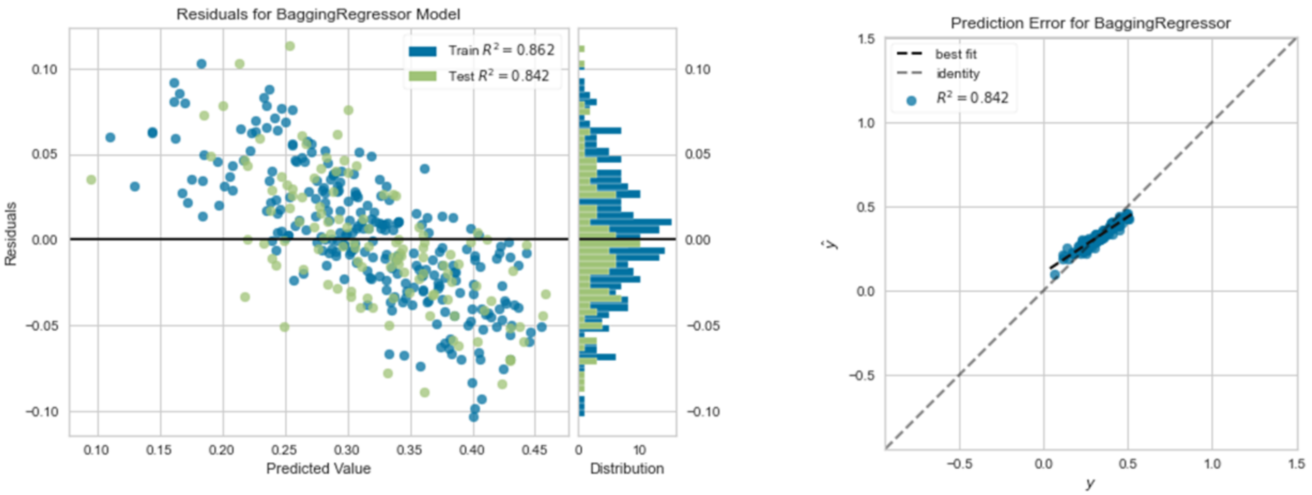
Table 4: Summary of Tuned inputs to minimize error metric

Parameters Tuned		Optimised Solution - TORIS	Optimised Solution - GOM
Model	TBKNN, B5K, RFR, TBC		RFR
Error Metric to minimize	MAE, MSE, RMSE, R^2	R^2	R^2
K-folds	5, 10, 15	10	10
Search Algorithm	Random Grid Search, Bayesian, Optuna, Hyperopt	Random Grid Search	Random Grid Search
Iterations	0-1000	50	50
Bagging	True/ False	True	False
Boosting	True/ False	False	False
Stacking	True/ False	True	False

Table 5: Error Comparison between different Regression Models Pre- and Post-Tuning on the Training Set only

Regressor Model	TORIS			
	MAE	MSE	RMSE	R^2
CatBoost	0.0821	0.0104	0.1015	0.0201
Bagged CatBoost	0.0741	0.0087	0.0934	0.0829
Boosted CatBoost	0.0761	0.0091	0.0954	0.0426
Stacked-10	0.0791	0.0096	0.0976	0.0949

As observed from Table 5, the error values (MAE, MSE and RMSE) obtained post hyperparameter tuning has decreased, while the R^2 has increased. The tuning of hyperparameters is therefore required in order to obtain a more accurate model. As illustrated in Figure 6, the R^2 value on the test set was 0.842 for the TORIS dataset (upper), and 0.722 for the GOM dataset (lower).



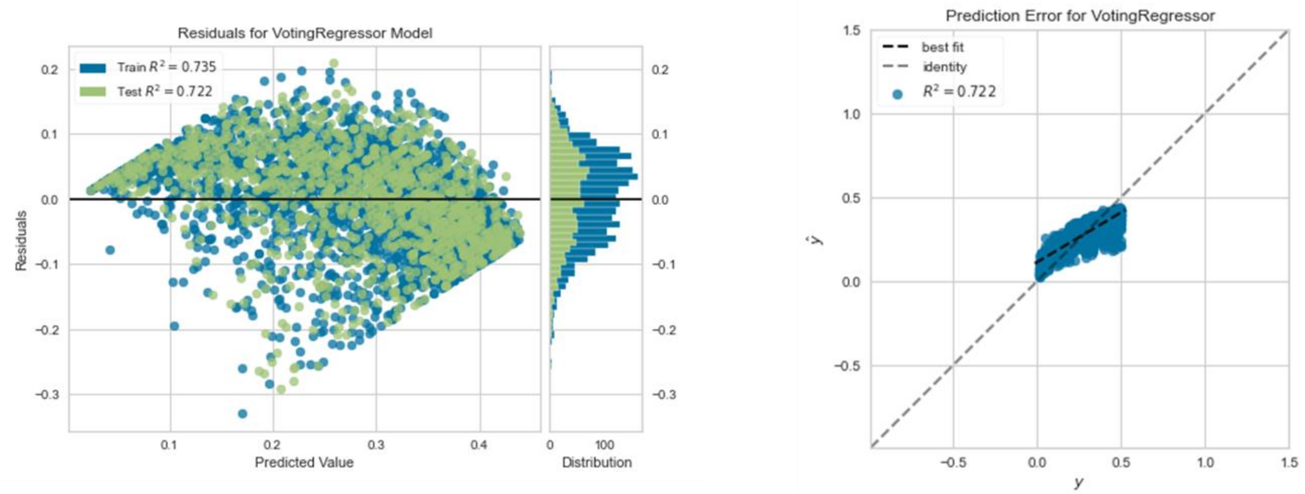


Figure 6: Prediction errors for TORIS test set (upper) and GOM test set (lower)

Evaluation against the "Hold-out" Data and the "Double Blind" Set

The final check into the predictive capability of the model is when it is applied on a "hold-out" data set. This dataset is made up of a random cull of 10% of the TORIS and GOM datasets that had been excluded from the train-validate set early on. Further to this, we add 3 additional fields from ERCE's analogue dataset, in which the recovery factor has been independently interpreted from first principles i.e. from geological mapping, petrophysical analysis, PVT and DST interpretation and in some cases decline curve analysis of the production data. These 3 fields are essentially a "double blind" test, with high confidence in interpreted "true" RF. The 3 fields are chosen to have a deliberate geographic spread, with variable rock, fluid and geological trap types.

The first field is comprising a field from the former Soviet-Union. It is an onshore field with an average permeability across the reservoir of between 10 to 100 mD. Reservoir Pressure is ~3800 psi. The 2nd field is located in Asia-Pacific region with an average reservoir permeability of 100-1000 mD, and initial reservoir pressure of ~2100 psi. The last field is offshore, located just to the south of the Gulf of Mexico. It has an average reservoir permeability of <100 mD and initial reservoir pressure is ~4700 psi

Figure 7 shows the prediction of different ML models (open circles) and the average of all models (filled circles) for the TORIS (upper) and GOM (lower) databases. Larger red points represent the fields which were analyzed independently as part of our "double blind" exercise.

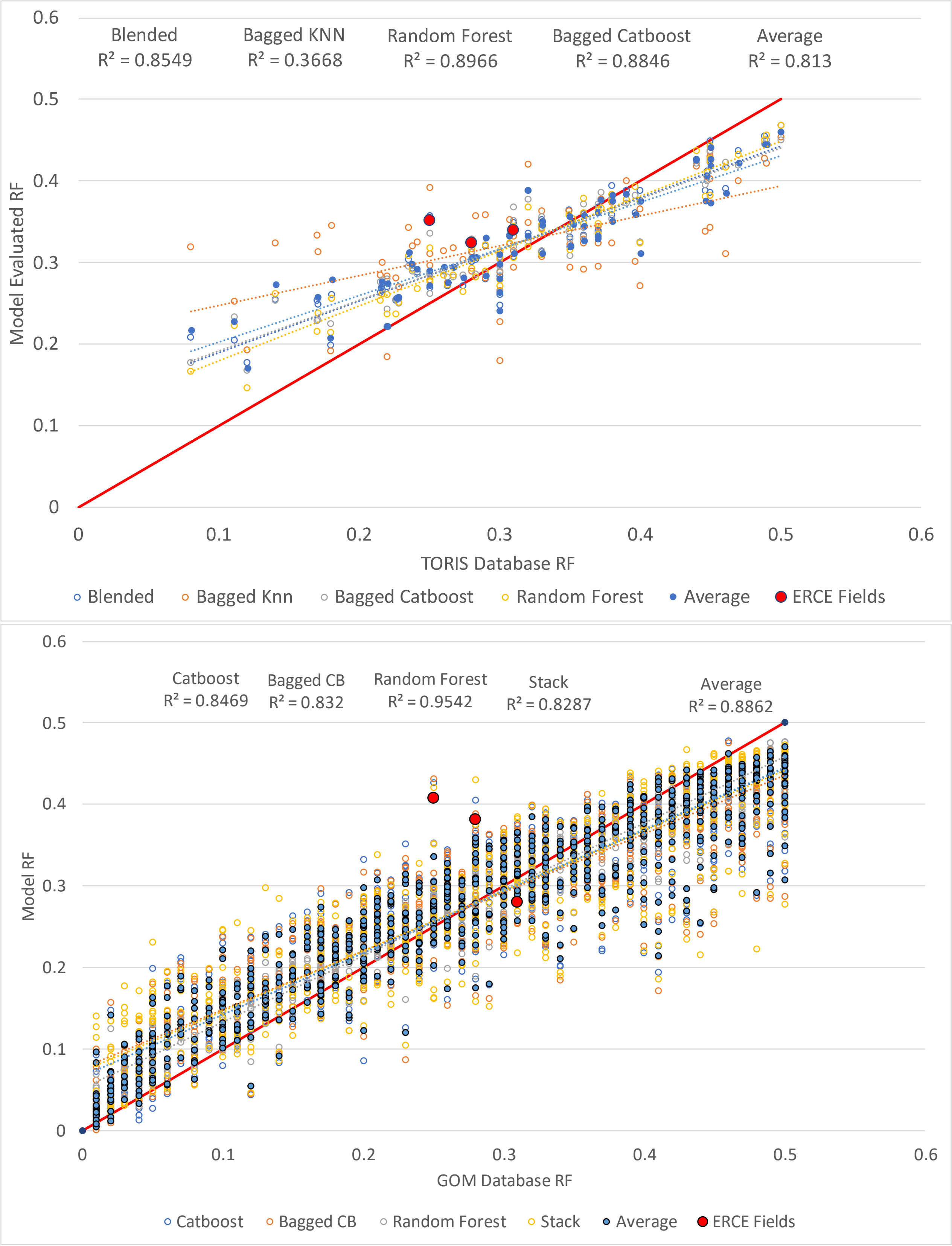


Figure 7: ML Model RF prediction on Blind Dataset (TORIS, Upper; GOM, Lower)

The average of all models shows our predicted RF to be close to the actual RF values, with R^2 of 0.813 for the TORIS dataset, and 0.886 for the GOM dataset. In both datasets, the Random Forest model showed the best results, with a margin of error of roughly +-10%.

We additionally compared the results of the machine learning algorithms to both Arps et al. and Guthrie and Greenberger's correlations, with the results shown in Figure 8 (TORIS, upper; GOM, lower).

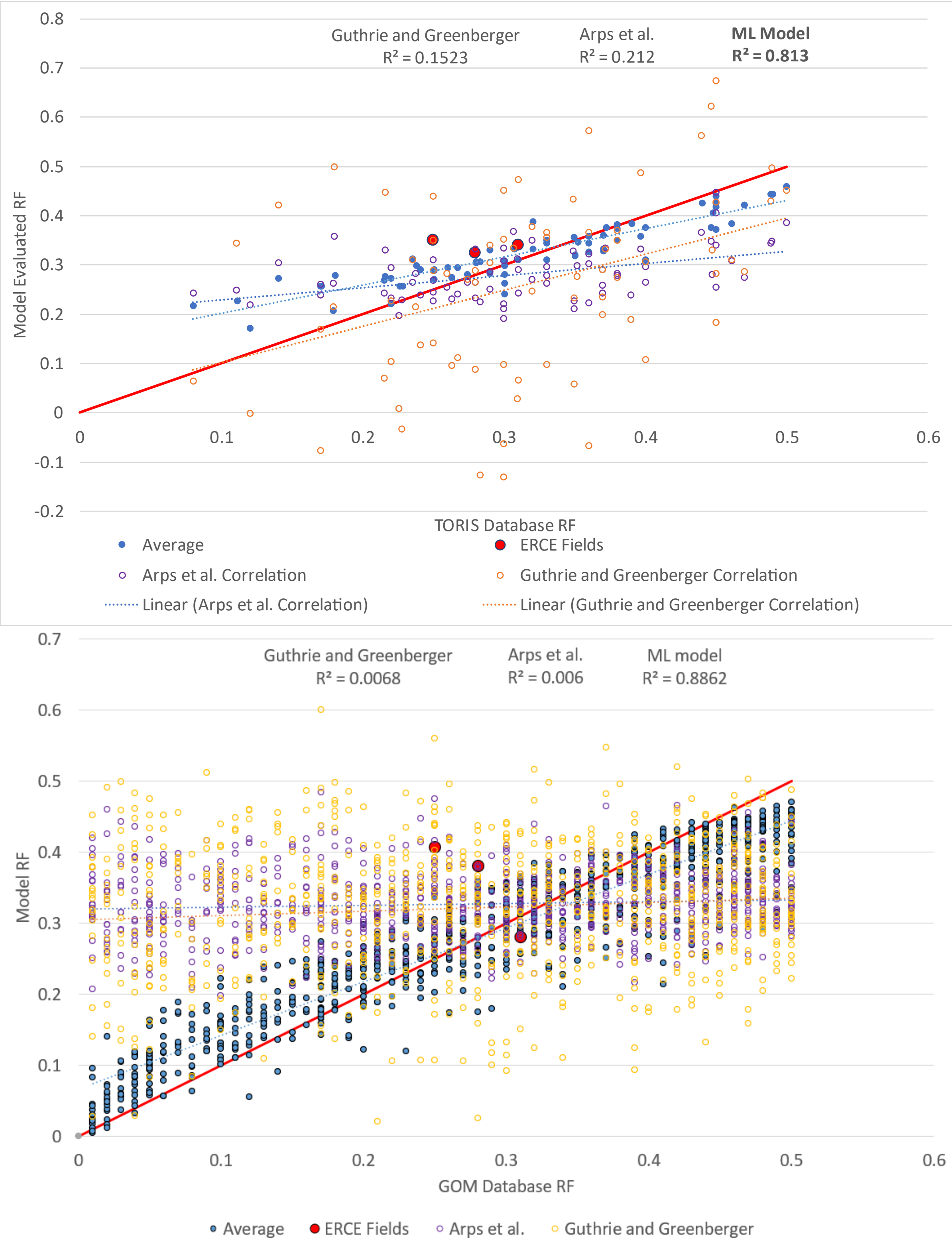


Figure 8: Machine Learning Model RF comparison to Arps et al. and Guthrie and Greenberger Correlation RF's (TORIS, upper; GOM, lower)

The RF of the three independent fields appears to perform consistently with estimates generated by the Arps et al. and Guthrie and Greenberger correlations (Table 6). This is true in all cases except the Asia-Pacific field, where only Arps is seen to be close to the independent interpretation.

However, application of the same correlations to the rest of the blind data set shows that the ML model is a much better predictor of recovery factor, if based on R^2 values (Table 7) and from observations of the results in Figure 8. The ML model data spread is narrower, proving therefore that the ML approach results in a much lower range of uncertainty.

Table 6: ML model RF compared to conventional correlations on select interpreted fields

Field	Recovery Factor (V/V)				
	Independent Interpretation	TORIS ML Model	GOM ML Model	Arps et al.	Gurthrie and Greenberger
Former SU	0.31	0.33	0.28	0.29	0.47
GOM	0.28	0.32	0.38	0.33	0.29
Asia Pacific	0.25	0.35	0.40	0.27	0.35

Table 7: R^2 values of ML model and Arps et al. and Guthrie and Greenberger correlations

Database	R^2 Value		
	ML Model	Arps et al.	Gurthrie and Greenberger
TORIS	0.81	0.21	0.15
GOM	0.88	0.006	0.007

Discussion

Below we summarise our key findings and learnings in this of developing the ML model for RF prediction.

Data Preparation

When performing the initial data-analysis, it is essential to combine both data analysis techniques with domain knowledge. In both datasets, many values are missing. This can be dealt with by dropping missing variables. However, a better alternative would be to use domain knowledge to try to supplement the missing data. The use of domain knowledge and experience is critical because correlation and causality are not the same. Taking the example of "STOIIP vs Porosity" and "Gross Pay vs Porosity", a correlation relationship is geologically reasonable in the former, but not in the latter.

Optimization

In an optimalization problem, the end goal is the minimization of a metric/ cost function. In our case, as we are dealing with regressors, the minimization must be applied to error functions like MAE, MSE and RMSE. The solution is best achieved via a gradient descent algorithm where the minimum of the cost function is achieved iteratively, largely because as the negative of the gradient is followed over time, it would theoretically reach a point where it will no longer be possible to decrease the cost function any further. In other words, when the number of iterations increases, the solution moves towards the minima which is defined by an optimal input hyperparameter set. The challenge is than finding that sweet spot of iterations and hyperparameters, and accepting the tradeoffs that occur, especially as it related to training time.

Use of Multiple ML models

When running the machine learning model on the blind dataset, most models showed results that were generally in line with the "actual" RF values and with empirical predictions, to within a margin of error of ±10%. It turns out that our use of multiple models had an unseen advantage, in that we could define an error range dependent on the class of model employed (i.e. DT vs distance based). We realized that the error range the models gave could be very useful and telling of the field. For instance, in large clearly commercial fields, a 10% uncertainty may not have a critical impact on final recoverable volumes, as contrasted to a smaller sub-marginal field, where variations in the RF value can help frame discussions around investment decisions. Fundamentally, where and how the ML outputs are used really depends on project maturity. For early-stage prospects, exploration plays, and sub-marginal fields, having a handle on the range of uncertainty would certainly add more value than having a single value.

We are far from advocating ML models as the "be all and end all" of RF prediction. In fact, we view these ML models as complementary to solutions derived from conventional empirical equations, production-based decline models or simulation models, with more weightages being given to established methods the more mature or "economically sensitive" the asset is.

Garbage In = Garbage Out

The importance of the training dataset is highlighted when considering the three, independently interpreted fields. Our ML solution was obtained from datasets that were USA focused; TORIS was a data set that had input data at the field scale, while GOM had input data that was focused at the reservoir scale. For our "double-blind" test, we applied the model to fields that were spread geographically and has as diverse a set of properties as we could determine. The TORIS machine learning model shows relatively good results for the former Soviet Union field (~2% error) and the Gulf of Mexico field (~4%) error, but a larger error in the Asia Pacific field (~10% error). This is further exemplified in the GOM model, which shows an error of up to 15% in the Asia Pacific field (3% error in former SU field and 10% error in GOM field). In fact, we discovered that the Asia Pacific field had the largest error margins by far of our 3 "double blind" data sets. We postulate that the margins of error may be reduced by using a more robust dataset that comprise of fields located worldwide.

Parametric Feature Importance

When looking at the ranked feature importance, we observed that categoric data was not considered to be critical in any of the selected ML algorithms. The effects of categoric data, like trapping type, lithology or diagenetic overprint might be important characteristics in certain reservoirs. We think that the underrepresentation is due to the plethora of possible individual values. For example, in TORIS, there are 54 unique depositional environments (eolian, lacustrine, shelf, reefs, pinnacles etc). To the ML model, there are insufficient training instance of each to observe a strong trending behaviour. To increase the importance of categoric data, the categories should be grouped and simplified. A recommendation is for a skilled geologist or geophysicist with some background in data science to look at helping to simplify some of the categoric input parameters such that only 3 to 5 unique instances present.

Conventional vs ML

The developed ML models outperformed conventional empirical correlations such as Arps et al and Guthrie and Greenberger. The spread of datapoints was much wider with using both the Arps et al. (R^2=0.21 and 0.006 in the TORIS and GOM model respectively) and Guthrie and Greenberger correlations (R^2=0.15 and 0.007 in the TORIS and GOM model respectively). Further work would be to investigate the machine learning model and its comparative performance on other fields.

Ease of Implementation

The use of automated machine learning and low code machine learning libraries significantly boosted our productivity and efficiency when it came to testing, troubleshooting and implementation of our ML solutions. The amount of time we took to test 20 models was equivalent to the amount of time it took to test one model if the ML code was built from scratch without the use of said libraries. The main challenge was making sense of the documentation and understanding the inner mechanism and interconnectedness within and between modules within the ML libraries. However, these libraries are relatively nascent, and as they gain wider acceptance by the ML community, we are confident this will prove less of an issue over time.

Non-linearity

Complementary to this work is the use of Artificial Neural Networks (ANN), which are a type of deep learning network complementary to ML. The key benefit of ANN is the ability to better handle non-linearities. Future work may investigate developing ANN networks to study the criticality of domain expertise and just how different RF predictions are if ANN is utilized instead of just ML.

Conclusion

In this work, we have demonstrated that ML models form a good basis for estimating RF; however, applying general domain knowledge and sense-checking results is still very important. The use of any ML model is dependent on the purpose of the RF estimation and should be complementary (rather than contrasting) to conventional techniques (whether they be using analogs or empirical methods). Any ML model should not replace the need for decline based, or simulation-based estimates in fields with extensive production history. In such instances, ML can be used to determine ultimate RF potential. For early-stage RF estimate, ML models might perform better that Arps et al. correlation and Guthrie and Greenberger correlation, especially when data is sparse.

We further show the utility of a low-code environment for rapid testing of multiple ML models, allowing for a combined, more accurate approach. However, a robust dataset of geographical spread with features determined by domain knowledge should be used for training the model as this would improve the performance and remove any potential bias.

Reference

[1] Ahmed, A.M., Salaheldin, E., Weiqing, C. and Abdulazeez, A. 2019., Estimation of Oil Recovery Factor for Water Drive Sandy Reservoirs through Applications of Artificial Intelligence. Energies. 2019 12(19), 3671. https://doi.org/10.3390/en12193671

[2] Altman, N.S. 1992 An Introduction to Kernel and Nearest Neighbor Nonparametric Regression The American Statistician Vol. 46, No. 3 (Aug., 1992), pp. 175-185 (11 pages) Published By: Taylor & Francis, Ltd. https://doi.org/10.2307/2685209

[3] Arps, J. J., Brons, Folkert, van Everdingen, A. F., Buchwald, R. W. and Smith, A. E. 1967. A Statistical Study of Recovery Efficiency, Bull. D14, API (Oct., 1967)

[4] Breiman, L. 2001. Random Forests. Machine Learning 45, 5–32 (2001). https://doi.org/10.1023/A:1010933404324

[5] Bureau of Ocean Energy Management (BOEM). 2019. Atlas of Gulf of Mexico Gas and Oil Sands Data. https://www.data.boem.gov/Main/GandG.aspx

[6] Freund, Y. and Schapire, R.E. 1996 Experiments with a New Boosting Algorithm. International Conference on Machine Learning, Bari, 3-6 July 1996, 148-156.

[7] Guthrie, R. K. and Greenberger, M. H. 1995. The Use of Multiple Correlation Analysis for Interpreting Petroleum Engineering Data, Drill. and Prod. Prac., API

[8] Waring, J., Lindvall, C., Umeton, R. 2020, Automated machine learning: Review of the state-of-the-art and opportunities for healthcare, Artificial Intelligence in Medicine, Volume 104, 2020, 101822, ISSN 0933-3657

[9] Makhotin, I., Orlov, D., Koroteev, D. et al., 2021., Machine learning for recovery factor estimation of an oil reservoir: A tool for derisking at a hydrocarbon asset level. SI: Computational Petroleum Engineering., Vol 8., Issue 2, June 2022, Pages 278 – 290., https://doi.org/10.1016/j.petlm.2021.11.005

[10] Moore, G. 1965. Cramming More Components onto Integrated Circuits. Electronics

Magazine Vol. 38, No. 8 (April 19, 1965).

[11] Sharma, A., Srinivasan, S., and Lake L.W. 2010. Classification of Oil and Gas Reservoirs Based on Recovery Factor: A Data-Mining Approach. SPE 130257-MS. https://doi.org/10.2118/130257-MS

[12] Silipo, R. 2021. Low Code Data Science Is Not the Same as Automated Machine Learning, https://www.knime.com/blog/low-code-analytics-platform. (Accessed May 2022)

[13] US Department of Energy. 1995. TORIS (Tertiary Oil Recovery Information System)., https://edx.netl.doe.gov/dataset/2006-oil-and-gas-industry-software (Accessed May 2022)

[14] Ying, X. 2019. An Overview of Overfitting and its Solutions. Phys.: Conf. Ser. 1168 022022.