SPE-202977-MS

Machine Learning Assisted Petrophysical Logs Quality Control, Editing and Reconstruction

Maniesh Singh, ADNOC Onshore; Gennady Makarychev and Hussein Mustapha, Schlumberger; Deepak Voleti, ADNOC Onshore; Ridvan Akkurt, Schlumberger; Khadija Al Daghar, Arwa Ahmed Mawlod, Khalid Al Marzouqi, and Sami Shehab, ADNOC Onshore; Alaa Maarouf, Obeida El Jundi, and Ali Razouki, Schlumberger

This paper was prepared for presentation at the Abu Dhabi International Petroleum Exhibition & Conference to be held in Abu Dhabi, UAE, 9 – 12 November 2020. Due to COVID-19 the physical event was changed to a virtual event. The official proceedings were published online on 9 November 2020.

Abstract

Mature field operators collect log data for tens of years. Collection of log dataset include various generation and multiple vintages of logging tool from multiple vendors. Standard approach is to correct the logs for various artefacts and normalize the logs over a field scale. Manually conducting this routine is time consuming and subjective. The objective of the study was to create a machine learning (ML) assisted tool for logs in a giant Lower Cretaceous Carbonate Onshore field in Abu Dhabi, UAE to automatically perform data QC, bad data identification and log reconstruction (correcting for borehole effects, filling gaps, cleaning spikes, etc.) of Quad Combo well logs.

The study targets Quad Combo logs acquired since mid-60's. Machine learning algorithm was trained on 50 vertical wells, spread throughout the structure of the field.

The workflow solution consists of several advanced algorithms guided by domain knowledge and physics based well logs correlation, all embedded in an ML-data-driven environment. The methodology consists of the following steps:

- Outliers detection and complete data clustering.
- Supervised ML to map outliers to clusters.
- Random Forest based ML training by clusters, by logs combination on complete data.
- Saved models are applied back to the whole data including outliers and sections with one or several logs missing.
- Validation and Blind test of results.
- Models can be stored and re-used for prediction on new data.

The ML tool demonstrated its effectiveness while correcting logs for outliers' like Depth Offsets between logs, identifying Erroneous readings, logs prediction for absent data and Synthetic logs corrections. The tool has a tendency to harmonize logs. First test demonstrated robustness of the selected algorithm for outliers' detection. It cleaned data from most of contamination, while keeping good but statistically underrepresented logs readings.

Clustering algorithm was enhanced to supplement cluster assignment by extraction of the corresponding probabilities that were used as a cut-off value and utilized for a mixture of different ML models results. This application made results more realistic in the intervals where clustering was problematic and at the transition between different clusters.

Several intervals of bad and depth shifted logs corrections were noticed. Outliers' corrections for these logs was performed the way that at Neutron-Density or Neutron-Sonic cross-plots points were moved towards expected lithology lines. Algorithm could pick-up hidden outliers (such as synthetic logs) and edited the logs to make it look intuitively natural to a human analyst.

The work successfully demonstrated effectiveness of ML tool for log editing in a complex environment working on a big dataset that was subject of manual editing and has number of hidden outliers. This strong log quality assurance further assisted in building Rock Typing based Static Model in complex and diagenetically altered Carbonates.

Introduction

A significant advancements has been made towards developing several approaches and methods for estimating and interpreting log measurements. The outcome of log analysis is dependent on various factors including but not limited to quality of original wireline and logging-while-drilling (LWD) log data, which with advancements in acquisition and corrections are assumed to be correct. This assumption on the quality of modern log data acquisition is generally valid, but often fails when the condition within wellbore degrades to the point of falling outside the physical measurement limitation of tools, or when mixing of vintage/older data from different tool design with modern data. Unless problems are encountered during the data mining, synthesis and integration, the problem or log data outliers may not be observed and resolved in subsequent log data evaluation.

Generally logs are corrected for environmental and borehole geometry effects but in extreme cases of wellbore conditons the log data degrades and requires proper way to correct or reconstruct, if possible. Data preconditioning prior to processing and interpretation is often assumed but seldom implemented. Environmental corrections are assumed to have been applied during data acquisition, and further corrections are not required. Due to these assumptions, data from vendors are often processed with little consideration of assured base quality and or with source of information, occasionally resulting in erroneous log analysis.

When discrepancies or outlier on log data are detected, the traditional approach is to apply manual and time consuming fit-for-purpose or shortcuts to correct or recreate dataset without taking into account nearby well log data from various wells, regional and geological petrophysical trends and distribution. Log data editing/ reconstruction can take many forms including regression of one type of measurement into other, generation of pseudo-logs using offset well log data and using translation application of regional trends. Methods involved vary from empirical algorithms to geological trends analysis to statistical methods to artificial intelligence (AI) based neural network (NN) to advanced form of machine learning (ML). Success of any of these methods requires data that are representative of reservoirs of interest when acquired under optimal conditions to assure their appropriate use in defining consistent and correct representation of the desired log data. Log data editing/ reconstruction requires a thorough understanding of the quality of the acquired data (calibration, accuracy, vintage etc.), the wellbore condition, tool configuration, tool type and acquisition methodology when the data is acquired and field processed.

Artificial Intellidgence (AI) is a technique of data nalysis that learns from data, identify patterns and makes predictions with minimal human interventions. It has many advantages including but not limited to that it doesn't require prior knowledge of the petrophysical response equations is self-calibrating driven by data, avoids the problems of 'rubbish in, rubbish out' by ignoring noise and outliers, little user intervention, works with unlimited number of electrical logs, core and other data, don't 'fall-over' if some of those inputs are missing and finally is not a black box as it provides insight into how it makes prediction (Cuddy, 2020).

AI programs currently developed includes ones where their machine code evolves, using similar rules used by life's DNA code. It is advisable that the AI program development should include a risk assessment.

This manuscript highlights a case study from a giant Lower Cretaceous Carbonate Onshore field in Abu Dhabi, UAE where artificial intelligence (AI) based machine learning (ML) assisted tools and workflow solution is designed with advanced algorithms guided by regional and domain knowledge to automatically perform data QC, identify outlier and edit/ reconstruct quad-combo well logs using widely represented training datasets. Quad-combo logs from same well were scanned and compared with similar logs from nearby wells and a local trend is generated for individual logs, similarly, all the logs present in the entire field were scanned for generating field trends. Individual well true sub-sea depth, and well zones were also input to generate zonal trends. It was observed, pre-picked zonal trends also had various discrepancies, so a second pass litho-stratigraphic trends were generated using all the existing dataset. This zone based, local and regional trends were robust enough to provide outlier coefficient to individual sample of every well log. The details on objectives, system architecture and workflow, validation and blind test procedure and results are discussed below.

Machine Learning Objectives

The artifical intelligent driven machine learning based algorithms and models were build to meet the following objectives:

a. To automatically and rapidly quality control and enhance reservoir imaging from well log data with minimum user interference.
b. To automatically perform data QC, bad data identification and log editing and reconstruction (correcting for borehole effects, filling gaps, cleaning spikes, etc.) on quad-combo well logs (only density, neutron, resistivity, sonic) using widely field represented 50 training vertical wells dataset.
c. To provide confidence classifications on log correction results for validation.
d. To provide the corrected well logs accompanied by outlier flags on the original well logs and uncertainty estimates on the corrected well log predictions.
e. To apply multi-well workflow to perform traditional volumetric workflow for further integration into the wider reservoir management workflow.
f. To apply on vertical wells, with the inclusion of 4 deviated wells and 4 LWD wells to assess the impact of geometry and acquisition technique on results.
g. To apply and assess spatially independent approach on results.
h. To build ML model using lithostratigraphic zones to assess improvement in results.
i. To assess potential time saving on providing corrected logs than would be required for human-led interpretations

System Architecture and Workflow

Figure 1 presents a simplified scheme of the system architecture and workflows consists of several automated and interactive steps as outlined below.

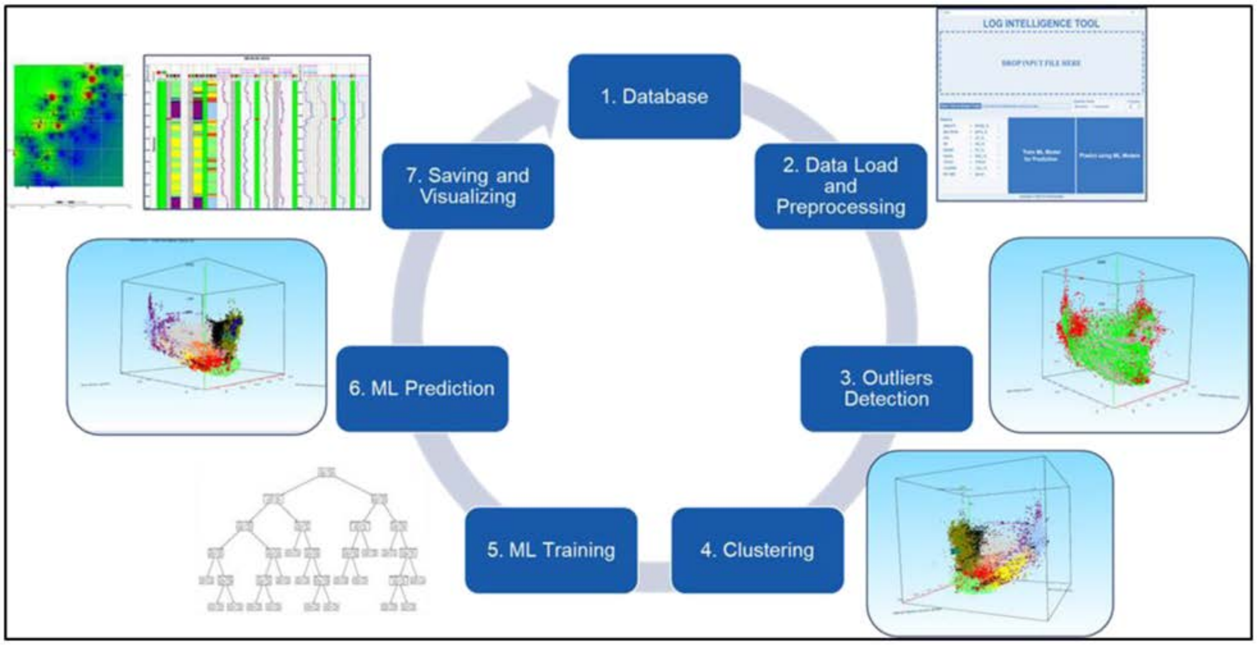


Figure 1—System Architecture and Workflow

Database Preparation

Log data from a carbonate field in UAE was collected and harmonized to ensure same mnemonics and same units for all logs. Data was also limited to the zone of interest that included several carbonate reservoirs separated by tight layers and some more shaly sections.

Density, Neutron, Compressional Slowness, Gamma Ray, Shallow and Deep Resistivity logs, Caliper and True Vertical Depth Sub Sea reference were selected for Machine Learning assisted logs editing algorithms testing.

Data Load and Preprocessing

Before starting with Machine Learning data is preprocessed. Preprocessing currently includes identification of bad hole intervals from Caliper readings and identification of log intervals with constant values (typically at the top and bottom of the logged interval). Identified intervals are flagged, logs replaced by missing values and removed from training step. Resistivity logs are linearized to be compatible with the rest of the data for outliers detection and clustering steps. After first data analysis we also created two new variables: Neutron-Density separation (difference between neutron and density porosity at limestone matrix) and Neutron-Density angle (as an angle between a given point at Neutron-Density cross plot and zero limestone porosity point).

Outliers Detection

In several applications, it is often crucial to identify anomalies in the data. Density-Based Local Outlier Factor (LOF) which is an unsupervised approach is one such method for the outliers' detection (Breunig et al., 2000). LOF measures the density variation of a data point relative to its neighbors and gives it an anomaly score called LOF score. A sample is considered an outlier if it has a high anomaly score or in other words, if its local density is notably lower than that of its neighboring samples (J. Han et al., 2011). Unlike other outliers' detection algorithms, LOF assigns a degree of anomaly to each sample instead of just assigning an inlier/outlier label to each sample. This degree of anomaly is referred to as Local Outlier Factor. LOF is based on k-nearest neighbors where local density is drawn from the distance between neighboring data points. In cases where the dataset is density-based and cluster-based by nature, it makes more sense to identify local outliers rather than global ones because there could be several density variations within the same dataset. LOF accommodates well for density deviations within the same dataset (Aggarwal, 2015).

One of the main advantages of LOF is being unsupervised and not taking prior assumptions about the data distribution. LOF is also flexible and can scale to different data types by defining a convenient distance measure. The main drawback however is the computational complexity which is O(N2). This is because the process of finding the nearest neighbors of a given data point requires computing the distance to all other samples. In addition, defining the distance measure becomes a challenge when the data type is complex like in the case of sequences (Chandola et al., 2009).

Various extensions of LOF were proposed in the literature. For instance, SLOM (Spatial Local Outlier Measure) captures spatial outliers and can scale to big data (Chawla & Sun, 2006; Sun & Chawla, 2004). A variant of LOF was introduced which can handle categorical data by defining a similarity score rather than distance measures (Yu et al., 2006). Another variant of LOF was introduced which dynamically detects anomalies in data streams (Pokrajac et al., 2007). Bai et al. introduced a distributed system which runs LOF in parallel (Bai et al., 2016). The technique divides the data into several grids which are allocated to the distributed system. The approach is especially useful on big data and requires minimal communication resources. LOF proved to be efficient in several applications like spam detection. You et al. proposed an LOF based approach which detects spam online reviews that deceive customers (You et al., 2020). The work was tested on TripAdvisor and showed significant efficiency in identifying spammers.

We utilized LOF to detect anomalies in our dataset. Figure 2 illustrates LOF results when ran on a sample of 600 data points from the data. Since the data is multi-dimensional, we utilized Principal Component Analysis (Abdi & Williams, 2010) to convert the data into two dimensions for the purpose of visualizations. The black points are data samples and the red circles represent the outlier scores. The bigger the diameter of a circle, the larger the outlier score, meaning that the sample is more likely to be an outlier. The figure clearly depicts that nearby samples within the same cluster are assigned by LOF as inliers while faraway samples with low local density are reported as outliers. We can also notice that the outlier score (radius of the red circle) increases as the samples become farther from clusters with decreased local density.

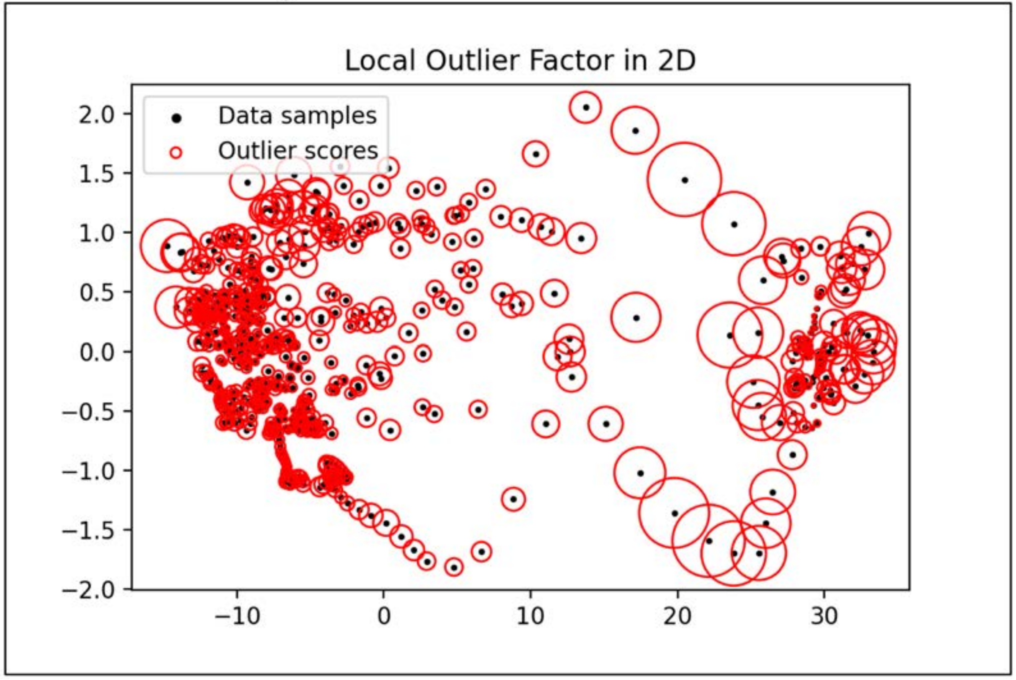


Figure 2—LOF applied on a subset of 600 samples from the dataset

An outlier detection step is required to remove from the training set any erroneous readings. The classical outliers detection algorithms use all features (logs in our case) and flags the whole sample (a depth) as an outlier. In our case it would mean that such intervals could not be edited because no information of any input log remains for prediction. Working with real data, however, more often one or couple of logs are affected either by borehole conditions or have some processing artefacts. That is why we developed a step wise outliers detection approach. The full set of input logs is used for outliers detection at a first step. At next steps inputs are removed one by one and if a given sample is not identified as outlier anymore, then a removed log is flagged as an outlier and replaced by missing value, whereas the remaining data declared as inliers and kept for training. If sample is still identified as outlier the procedure repeated with different pair of logs, then with a three logs etc. till logs causing the sample to be an outlier are not detected. Figure 3 shows the results of outliers detection as a Scatter Plot – Matrix of input logs. For all cross plots of pair of input logs outliers are detected at the edges of the main population, whereas projecting to a different cross plot same points can occur in a middle of the common cloud as it can be seen at the histograms. It is important to highlight that outliers detection algorithm parameters tuning should be performed case by case to preserve underrepresented but true readings located far from the main population (i.e. shale section that can be clearly seen at a Neutron-Density cross plot).

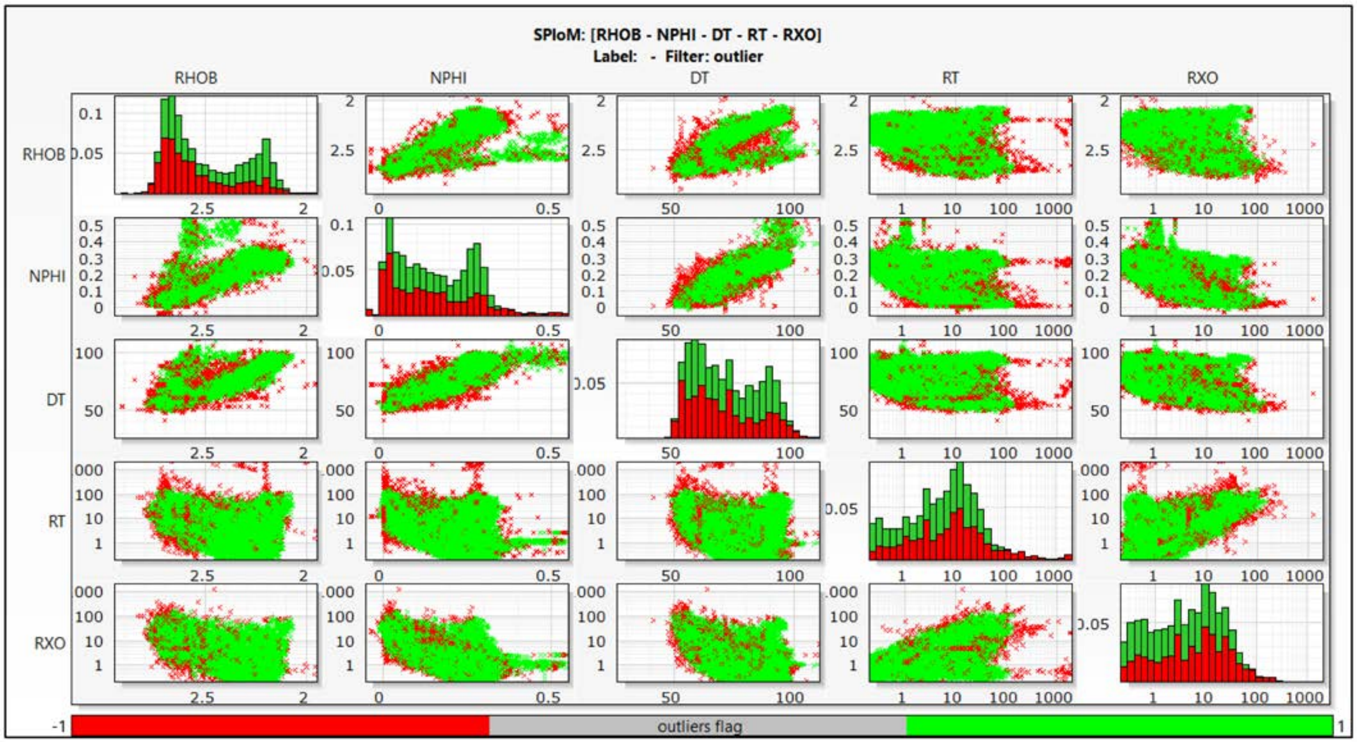


Figure 3—Scatter Plot – Matrix with outliers detection results. Outliers (red points) and inliers (green points) projected to all dimensions in a form of cross plots. Red points at all cross plots are located at the edges of main population.

Clustering

Probabilistically, one can presume that data to be clustered can be drawn from several probability distributions (e.g., t-distribution or Gaussian) or from an identical distribution parametrized differently. Mixture models clustering aims at obtaining the parameters of the data distribution (McLachlan & Basford, 1988). Gaussian Mixture Models (GMM) (Rasmussen, 2000) suggest that data follows multivariate Gaussian distributions where data that belongs to the same cluster has the same Gaussian parameters. A Gaussian mixture model can be formulated as a weighted sum of k Gaussian densities as following:

$$p(x|\lambda)=\sum_{i=1}^{k} w_i g(x|\mu_i \sum_i) \qquad \text{Equation 1}$$

where x is the data vector, w_i is the mixture weight, and $g(x|\mu_i \Sigma_i)$ is the Gaussian densities defined as following:

$$g(x|\mu_i \sum_i)=\frac{1}{(2\pi)^{D/2}\left|\sum_i\right|^{1/2}}\exp\left\{-\frac{1}{2}(x-\mu_i)\sum_i^{-1}(x-\mu_i)\right\} \qquad \text{Equation 2}$$

where μ_i is the mean vector and $\sum_i$ is the covariance matrix. GMM is parametrized by μ_i, $\sum_i$ and w_i, which are iteratively estimated by Expectation Maximization (EM) (Dempster et al., 1977). GMM is popular within the research community due to its speed to learn mixture models. On the other hand, the number of clusters is one of its requirements. However, (Yang et al., 2012) developed a robust EM that is able to automatically compute the optimal number of clusters.

Clustering is introduced in the workflow after data review and consideration that despite of the pretty uniform lithology of the formation the relationship between different logs can be quite different between different facies. Beside that and because data covers reservoirs with different saturation character and includes wet and HC saturated zones as well as water swept intervals Resistivity response might change a lot comparing to other logs variations. Figure 4 shows 3D Neutron-Density-RT cross plot. In the area of the high porosity reservoirs Resistivity input provides key information. Clusters are defined based on RT variation and hidden in the Neutron-Density space. The low correlation of the RT with other logs and clustering results high dependency on RT input for the dataset in case resulted in cases of wrong clustering and errors in logs editing for intervals where RT log was identified as an outlier. To overcome this issue, we developed a sequential approach of models training and logs editing and prediction.

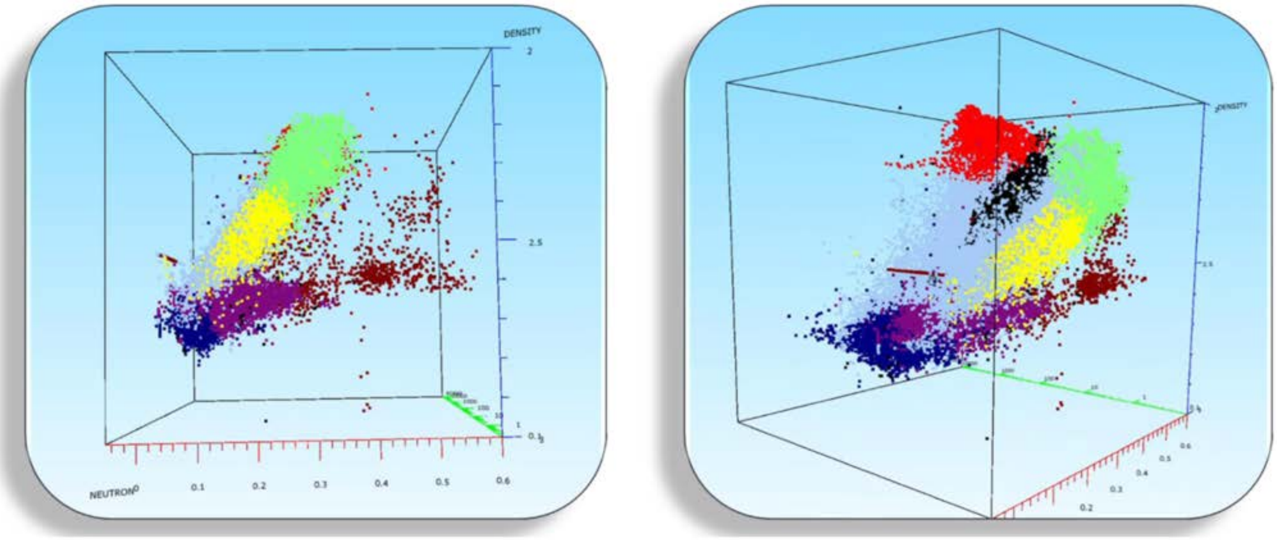


Figure 4—Clustering results in the Neutron-Density-RT space (Neutron on red horizontal axis, RT on green horizontal axis and Density on balck vertical axis)

ML Training

Originally proposed by Breiman in 2001, Random Forest (Breiman, 2001) proved its superiority over previous Machine Learning models like Support Vector Machines (Cortes & Vapnik, 1995) and Neural Networks (Hornik et al., 1989) in terms of robustness and stability (T. Han et al., 2018). Random Forest is an ensemble technique which groups several Decision Trees (Mitchell & others, 1997) and aggregates their decision for the final output. Each tree in the ensemble is constructed from a random subset of data from the training dataset and is independent from the other tree predictors. At every tree node, the best split is selected based on the input features of the corresponding tree. The final prediction is the average of the individual trees' predictions for regression and the majority label in the case of classification.

Random Forest provides superior performance over other algorithms in several aspects (Horning, 2013). The main advantage is the generalization capability encompassed by the aggregation of several estimators. This ensemble approach allows for less overfitting compared to Decision Trees for example. Furthermore, Random Forest is less affected by anomalies in the data relative to individual Trees. Random Forest holds the advantages of Decision Trees with better accuracy because it aggregates decisions of several estimators which decreases the impact of errors of the individual trees (Breiman, 2001). Moreover, Random Forest proved to be efficient with both discrete and continuous data types (Ali et al., 2012). It is also robust to increasing data dimensionality and complexity of the input data structures (Qi, 2012). Random Forest proved to be efficient in several applications including intrusion detection systems (Resende & Drummond, 2018).

Several variants of Random Forest were introduced over the years. On-line Random Forest for instance can handle real-time data by dynamically removing and building new trees in addition to evaluating real-time errors (Saffari et al., 2009). Bernard et al. proposed Dynamic Random Forest (DRF) which is inspired from boosting techniques and randomization approaches (Bernard et al., 2012). Utilizing adaptative tree induction, DRF creates new estimators that enhance the performance of the existing trees. In addition, several work aimed to scale Random Forest to handling Big Data (Genuer et al., 2017).

Given the notable performance of Random Forest, we utilized it to predict logs. Figure 5 highlights one of the tree estimators of Random Forest fitted on our dataset to predict Density using the input properties TVDSS, Neutron, DTC, GR, RDEEP, and RBEST in one of the clusters. The color intensity depicts the extremity of values with darker color meaning higher Density value. Every tree node shows the split variable and condition along with the number of samples at each node, the mean error and the average output value.

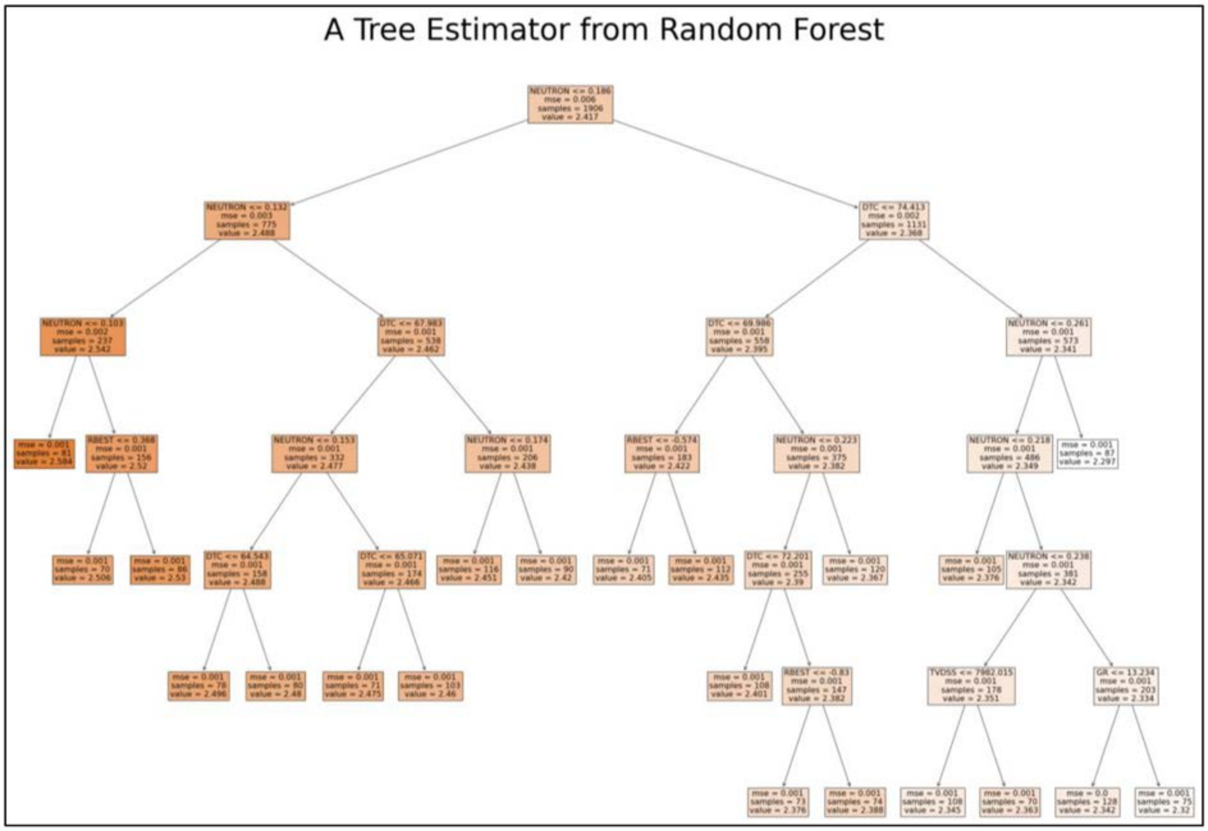

A Tree Estimator from Random Forest
NEUTRON <= 0.186 mse = 0.006 samples = 1906 value = 2.417
NEUTRON <= 0.132 mse = 0.003 samples = 775 value = 2.488
DTC <= 74.413 mse = 0.002 samples = 1131 value = 2.368
NEUTRON <= 0.103 mse = 0.002 samples = 237 value = 2.542
DTC <= 67.983 mse = 0.001 samples = 538 value = 2.462
DTC <= 69.986 mse = 0.001 samples = 558 value = 2.395
NEUTRON <= 0.261 mse = 0.001 samples = 573 value = 2.341
mse = 0.001 samples = 81 value = 2.584
RBEST <= 0.368 mse = 0.001 samples = 156 value = 2.52
NEUTRON <= 0.153 mse = 0.001 samples = 332 value = 2.477
NEUTRON <= 0.174 mse = 0.001 samples = 206 value = 2.438
RBEST <= -0.574 mse = 0.001 samples = 183 value = 2.422
NEUTRON <= 0.223 mse = 0.001 samples = 375 value = 2.382
NEUTRON <= 0.218 mse = 0.001 samples = 486 value = 2.349
mse = 0.001 samples = 87 value = 2.297
mse = 0.001 samples = 70 value = 2.506
mse = 0.001 samples = 86 value = 2.53
DTC <= 64.543 mse = 0.001 samples = 158 value = 2.488
DTC <= 65.071 mse = 0.001 samples = 174 value = 2.466
mse = 0.001 samples = 116 value = 2.451
mse = 0.001 samples = 90 value = 2.42
mse = 0.001 samples = 71 value = 2.405
mse = 0.001 samples = 112 value = 2.435
DTC <= 72.201 mse = 0.001 samples = 255 value = 2.39
mse = 0.001 samples = 120 value = 2.367
mse = 0.001 samples = 105 value = 2.376
NEUTRON <= 0.238 mse = 0.001 samples = 381 value = 2.342
mse = 0.001 samples = 78 value = 2.496
mse = 0.001 samples = 80 value = 2.48
mse = 0.001 samples = 71 value = 2.475
mse = 0.001 samples = 103 value = 2.46
mse = 0.001 samples = 108 value = 2.401
RBEST <= -0.83 mse = 0.001 samples = 147 value = 2.382
TVDSS <= 7982.015 mse = 0.001 samples = 178 value = 2.351
GR <= 13.234 mse = 0.001 samples = 203 value = 2.334
mse = 0.001 samples = 73 value = 2.376
mse = 0.001 samples = 74 value = 2.388
mse = 0.001 samples = 108 value = 2.345
mse = 0.001 samples = 70 value = 2.363
mse = 0.0 samples = 128 value = 2.342
mse = 0.001 samples = 75 value = 2.32

Figure 5—Illustration of one of the trees from Random Forest which predicts Density

Filtered by outliers detection results and separate logs prediction models are trained by clusters for all detected combination of inputs (i.e. the whole set, missing DT, missing Density and Neutron etc.). As discussed just above we had to introduce a sequential model for ML training and prediction. Since the raw logs are utilized for clustering it might happen that a wrong information provided at this stage to the Machine Learning (i.e. pseudo-logs described in the next paragraph). The wrong clustering will drive a selection of the wrong model resulting in significant errors in logs prediction and editing. We observed that RT is most sensitive in this case and introduced a sequential model training. To overcome the problem, we introduced the second step of model training using edited Neutron-Density-DT and Gamma Ray logs for clustering and prediction of RT and noticed significant improvement of RT editing results. All models are saved and can be applied to the training set or to new data coming from the same field and acquired over the same formation.

ML Prediction

Algorithm is trying to predict missing logs and correct existing for all cases where at least two input logs are present. The only constraint is created for a case if all three porosity logs (Neutron, Density and DT) are identified as ouliters or missing in the input sample. Quality of prediction depends for sure on the quality of input logs. The same two steps prediction workflow as described above for training can be applied.

Saving and Visualization

Results of the logs QC, editing and prediction are saved as .CSV file and can be visualized using any platform accepting the format. In our case we did use Techlog platform and developed metrics for quick review and control of the results. Because the developed tool works with tens and hundreds wells it is quite important to provide the user an ability to check the results quickly and highlight wells and logs that require his attention. For all edited logs two scores are computed:

- Prediction Score (applied to non-outliers):

$$Sp = \sqrt[2]{(\frac{Vinput - Vpredicted}{\sigma_{Vinput}})^{2}} \qquad \text{Equation 3}$$

Is a measure of a difference between raw and edited logs.

- Outliers Score (applied to outliers):

$$So = \sqrt[2]{(\frac{Vinput - \text{Vmedian cluster predicted}}{\sigma_{Vinput}})^{2}} \qquad \text{Equation 4}$$

Is a measure of a difference of outliers to model. Median values by well are computed. So is multiplied by $\frac{Noutliers}{Nsamplesperwell}$ to reflect number of outliers per well.

The results can be visualized as a cross plot or maps to see the wells with high scores that require special attention. Figure 6 shows an example of the application of the scoring to NPHI log editing results. Both cross plot and map show that the log has a lot of outliers and was heavily edited in Wells 84 and 79. After this first quick look a user can open a well plot or cross plot to investigate what exactly happened. For example, in case of the well 84 (Figure 7) raw NPHI log either has calibration problems or registered using a wrong matrix and shows unreasonably high values comparing to RHOB log in the clean limestone formation.

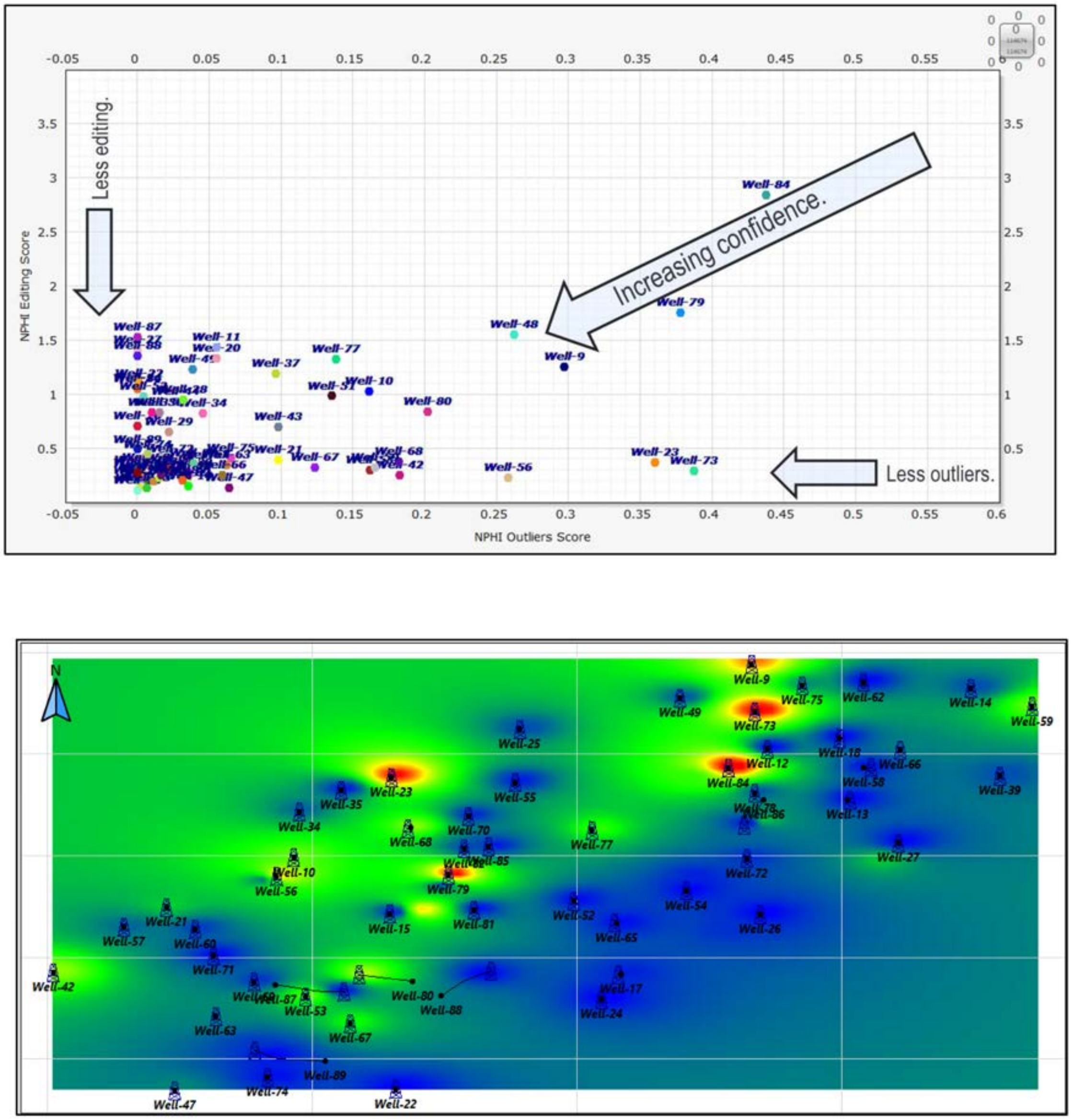


Figure 6—Machine learning logs editing results QC visualization in Techlog.

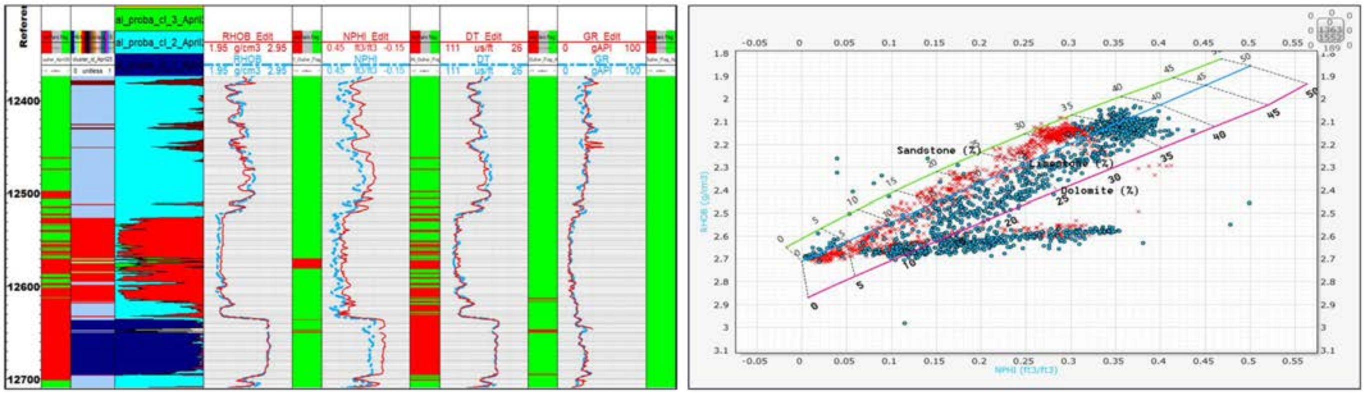


Figure 7—Well-84 log plot and Neutron-Density cross plot. Raw data is displayed with blue curves and points, edited – red.

Results

Logs QC and editing models were trained using 50 wells picked from the crest, mid-flank and flank of the field. An intensive program of the results validation and testing was planned and completed. Different scenarios of missing data and data affected by either borehole conditions or processing of the logs were selected, and trained models applied for logs from 27 wells from the same field. The process helped to fine tune the workflow and parameters of the algorithms and greatly improved final results. As of now we can see a very good performance of the trained models for Neutron, Density and DT logs editing and prediction. Resistivity editing results are good for more than 80% of cases but still challenging and would require integration of other data and local knowledge about the field for 100% success.

Following figures illustrate several cases of the Machine Learning assisted logs editing results.

- ✓ Figure 8 shows a case where in a bad hole section Neutron log was identified as outlier and successfully corrected. At the cross plots points in a bad hole are red. Edited points are moved towards an expected clean limestone lithology line.
- ✓ Figure 9 illustrates the ML logs editing behavior where DT log is affected by processing in the bottom section of the well and missing in the upper section. Edited DT log now follows Neutron and Density logs behavior and predicted with a high confidence in the upper section.
- ✓ Testing data included some wells where either Density or Neutron log is synthetic. It can be clearly seen at the cross plot where data from these wells is presented by straight lines (Figure 10). Such logs readings represent so called hidden outliers, for most cases a single sample may be realistic. However, using the ensemble of the data those pseudo logs are corrected the way to make it look intuitively natural to a human analyst.
- ✓ While analyzing results of the logs editing, we also noticed a 'side' effect of logs standardization. Logs for the field were acquired over tens of years by different vendors using different equipment. This provided significant difference of the Neutron logs acquired over the same formation (Figure 11). ML edited Neutron logs kept difference provided by difference in fluids (light oil vs. saline water) but points at the cross plot are laying uniformly and closer to an expected position in the clean limestone formation.

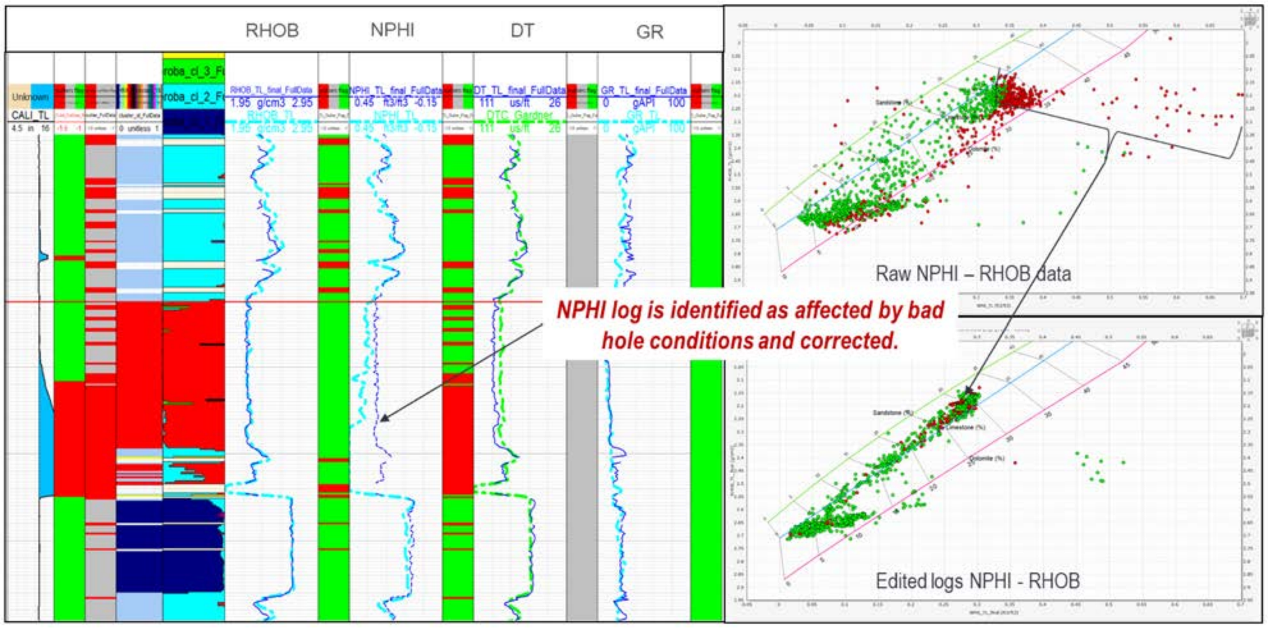
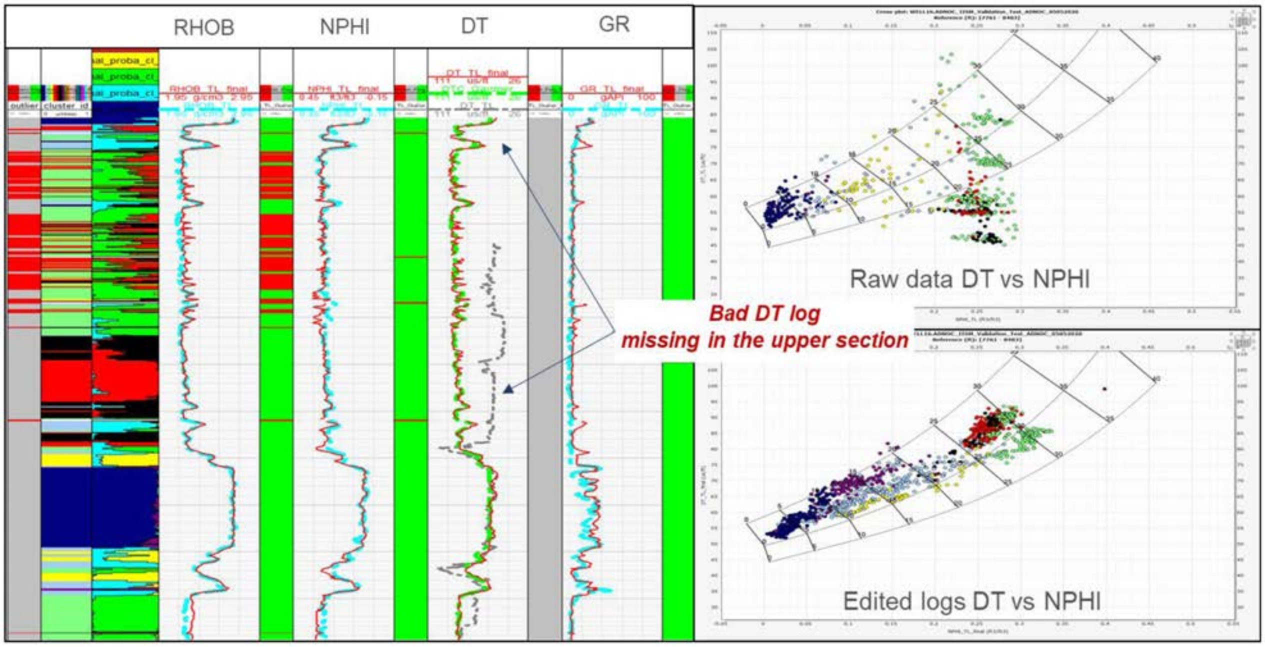


Figure 8—Correction of a Neutron log in a bad hole section.



Bad DT log missing in the upper section

Figure 9—Correction and prediction of the DT log.

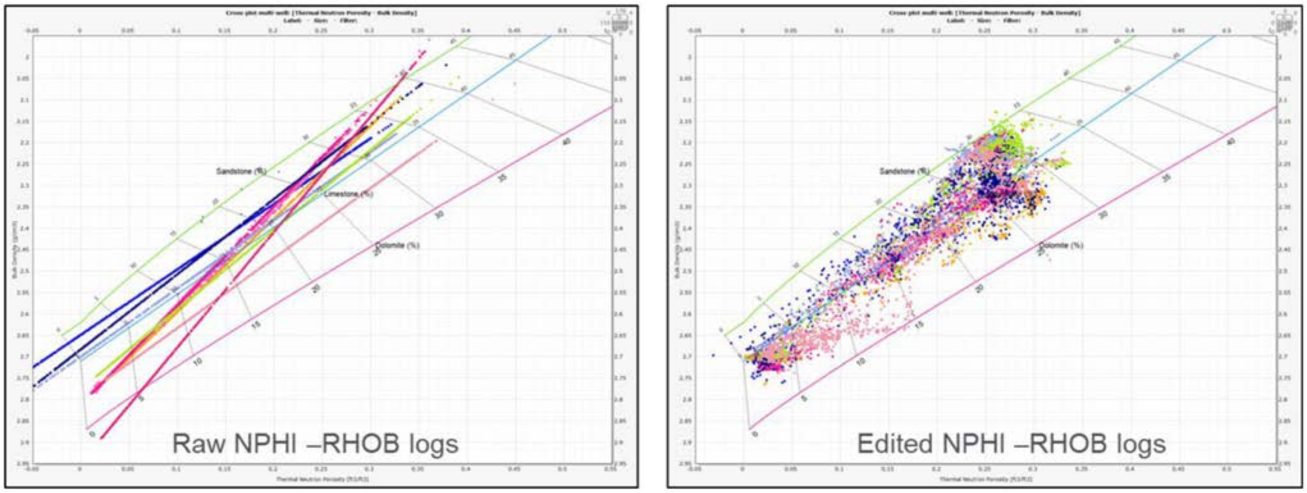


Figure 10—Neutron – Density pseudo logs editing example.

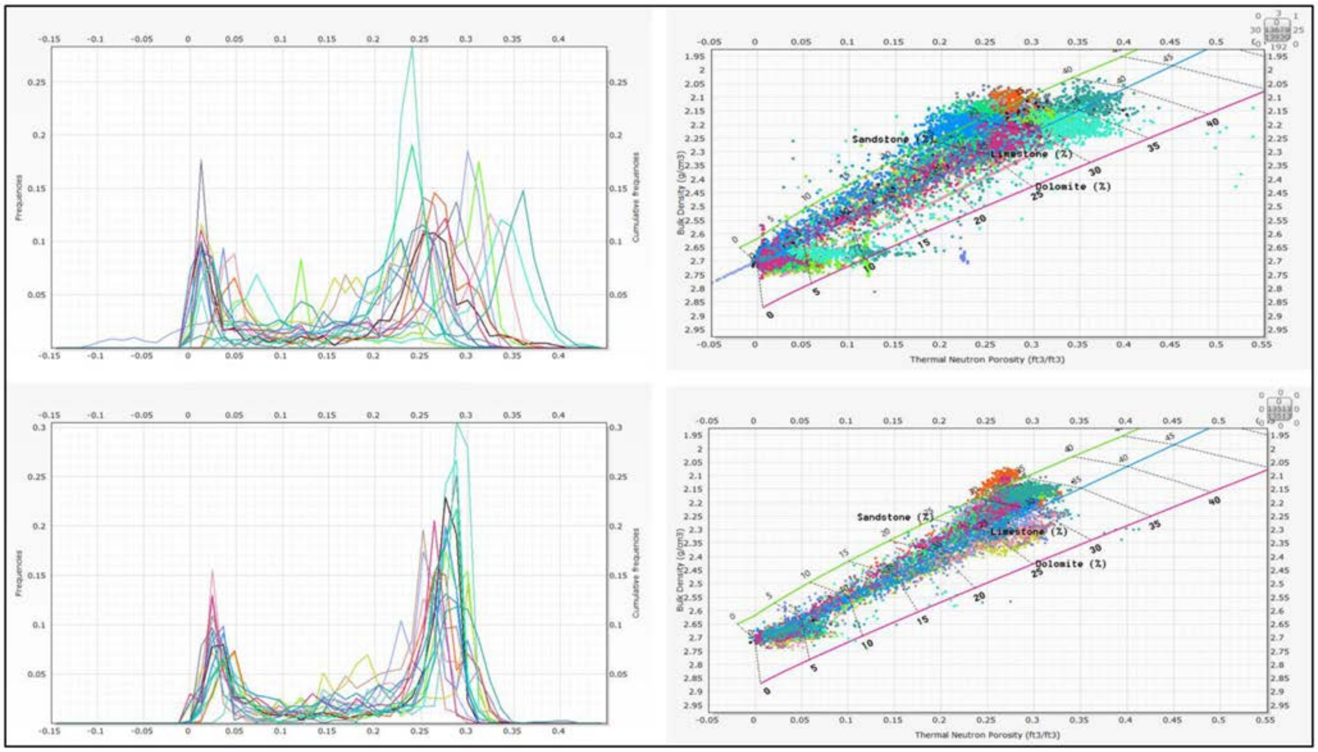


Figure 11—Illustration of the Neutron log standardization after ML assisted editing. Raw Neutron logs histogram and Neutron – Density cross plot at the top and edited logs at the bottom.

Observations & Conclusions

Efficient ML based methodology for petrophysical logs quality control and editing, where an innovative tool provided bad data identification and rapid corrections equivalent to human-led interpretation for 90–95% of Quad Combo logs in a fraction of the time to do so manually. Developed workflow includes Data Load and Preprocessing, Outliers detection using Local Outliers Factor algorithm, two steps Unsupervised and Supervised Clustering, Regression Forest ML models training and their application for logs editing end prediction. All components but visualization are embedded in the Application Programming Interface (API) with a user friendly and intuitive interface. Visualization step and QC of the results currently can be performed using any platform supporting log plots, cross plots and maps. Trained models are saved and can be applied for a new data.

The dataset combined logs of a different vintage since mid 60's acquired by different vendors and using different tools and techniques. ML models were trained on 50 wells and validated using 27 additional wells from the same field. The validation showed very good ML assisted logs editing results for editing and prediction of Density, Neutron and DT logs. Edited logs are corrected from depth offsets, borehole conditions effects and processing artefacts. Synthetic logs are corrected the way to make them look intuitively natural to a human analyst. Neutron logs are standardized towards expected lithology readings, keeping fluids effect at high porosity.

Review of the validation results did also show that trained models cannot effectively edit logs (especially resistivity data) if logs behavior is different from most of the logs used for training. Such logs and wells can be, however, quickly identified using developed QC workflow and user can take a decision about acceptance of the editing results or manual editing. If primary QC results are not satisfactory additional model(s) can be trained to cover zones or field compartments where logs behavior is different.

The full cycle of ML training, logs editing, and results review took about 2 working days (exclude developing ML algorithm and workflows) for a given dataset comparing to around 15 working days of manual well by well logs editing.

ML algorithms development confirmed that they can be successfully applied for ADNOC data to accelerate and may improve results quality of existing petrophysical synthesis workflows and their outputs for static model creation. A future development of ML and AI algorithms towards a full integration of geological, petrophysical and other domains information (i.e. spatial position of wells, lithostratigraphic and bio stratigraphic framework, depositional and diagenetic facies, wells drilling and production history etc.) is planned. This can lead to an improvement of created models and development of new workflows dedicated for more advanced petrophysical synthesis studies such as core analyses QC, logs normalization and hydrocarbon correction processing, petrophysical static rock typing, log permeability prediction etc.

Acknowledgments

The authors would like to thank the senior management and technical center of ADNOC & ADNOC Onshore for their kind support and for permission to publish the paper.

Nomenclature

AI	Artificial Intelligence
API	Application Programming Interface
CALI	Caliper
DRF	Dynamic Random Forest
DT or DTC	Compressional Slowness
HC	Hydrocarbon
LOF	Local Outlier Factor
LWD	Logging While Drilling
ML	Machine Learning
NN	Neural Network
NPHI	Neutron Porosity
RDEEP	Deep Resistivity
RHOB	Bulk Density
RT	True Resistivity
SLOM	Spatial Local Outlier Measure
QC	Quality Control
UAE	United Arab Emirates

References

Abdi, H., & Williams, L. J. (2010). Principal component analysis. *Wiley Interdisciplinary Reviews: Computational Statistics*, **2** (4), 433–459.

Aggarwal, C. C. (2015). Outlier analysis. *Data Mining*, 237–263.

Afi, J., Khan, R., Ahmad, N., & Maqsood, I. (2012). Random forests and decision trees. *International Journal of Computer Science Issues (IJCSI)*, **9** (5), 272.

Bai, M., Wang, X., Xin, J., & Wang, G. (2016). An efficient algorithm for distributed density-based outlier detection on big data. *Neurocomputing*, **181**, 19–28.

Bernard, S., Adam, S., & Heutte, L. (2012). Dynamic random forests. *Pattern Recognition Letters*, **33** (12), 1580–1586.

Breiman, L. (2001). Random forests. *Machine Learning*, **45** (1), 5–32.

Breunig, M. M., Kriegel, H.-P., Ng, R. T., & Sander, J. (2000). LOF: identifying density-based local outliers. Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, 93–104.

Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. *ACM Computing Surveys (CSUR)*, **41** (3), 1–58.

Chawla, S., & Sun, P. (2006). SLOM: a new measure for local spatial outliers. *Knowledge and Information Systems*, **9** (4), 412–429.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, **20** (3), 273–297.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, **39** (1), 1–22.

Cuddy, S., The Benefits and Dangers of using Artificial Intellingence in Petrophysics. SPWLA 61sf Annual Logging Symposium, June 24 to July 29, 2020, DOI: 10.30632/SPWLA-5066

Genuer, R., Poggi, J.-M., Tuleau-Malot, C., & Villa-Vialaneix, N. (2017). Random forests for big data. *Big Data Research*, **9**, 28–46.

Han, J., Pei, J., & Kamber, M. (2011). Data mining: concepts and techniques. Elsevier.

Han, T., Jiang, D., Zhao, Q., Wang, L., & Yin, K. (2018). Comparison of random forest, artificial neural networks and support vector machine for intelligent diagnosis of rotating machinery. *Transactions of the Institute of Measurement and Control*, **40** (8), 2681–2693.

Hornik, K., Stinchcombe, M., White, H., & others. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, **2** (5), 359–366.

Horning, N. (2013). Introduction to decision trees and random forests. *Am. Mus. Nat. Hist*, *2*, 1–27.

McLachlan, G. J., & Basford, K. E. (1988). Mixture models. Inference and applications to clustering. *Mmia*.

Mitchell, T. M., & others. (1997). Machine learning. 1997. *Burr Ridge, IL: McGraw Hill*, **45** (37), 870–877.

Pokrajac, D., Lazarevic, A., & Latecki, L. J. (2007). Incremental local outlier detection for data streams. 2007 IEEE Symposium on Computational Intelligence and Data Mining, 504–515.

Qi, Y. (2012). Random forest for bioinformatics. *In Ensemble machine learning* (pp. 307–323). Springer.

Rasmussen, C. E. (2000). The infinite Gaussian mixture model. *Advances in Neural Information Processing Systems*, 554–560.

Resende, P. A. A., & Drummond, A. C. (2018). A survey of random forest based methods for intrusion detection systems. *ACM Computing Surveys (CSUR)*, **51**(3), 1–36.

Saffari, A., Leistner, C., Santner, J., Godec, M., & Bischof, H. (2009). On-line random forests. 2009 Ieee 12th International Conference on Computer Vision Workshops, kcv Workshops, 1393–1400.

Sun, P., & Chawla, S. (2004). On local spatial outliers. Fourth IEEE International Conference on Data Mining (ICDM'04), 209–216.

Yang, M.-S., Lai, C. -Y., & Lin, C. -Y. (2012). A robust EM clustering algorithm for Gaussian mixture models. *Pattern Recognition*, **45** (11), 3950–3961.

You, L., Peng, Q., Xiong, Z., He, D., Qiu, M., & Mang, X. (2020). Integrating aspect analysis and local outlier factor for intelligent review spam detection. *Future Generation Computer Systems*, **102**, 163–172.

Yu, J. X., Qian, W., Lu, H., & Zhou, A. (2006). Finding centric local outliers in categorical/numerical spaces. *Knowledge and Information Systems*, **9** (3), 309–338.