AI-based Model to Predict Well Log Measurement: A Case Study in Volve Field, North Sea

Tobi Ore

A Project Report for Python for Oilfield Data Analytics (PNGE 691D)

Instructors:
Dr. Shahab Mohaghegh
Mr. Amir Ansari
Mr. Ayodeji Aboaba

ABSTRACT

The rapid advancement of technology as seen the application of machine learning techniques in the oil and gas industry. These techniques are used to solve problems that are somewhat difficult for traditional techniques, i.e. traditional techniques are relatively unreliable. One of these problems is in the area of prediction of well logs.

In most mature fields, millions of data are often time present. These data, some legacy, most time contain missing measurement due to several reasons, such as bad tools, blowouts or no measurement for that log altogether. To solve this problem, the missing logs have to be predicted using either traditional techniques or a more modern approach using machine learning.

In this project, a model is built to predict the Volume of Shale (Vsh) for wells. This model depends on a feed-forward neural network with 2 hidden layers that implicitly learn the relationships between the input features (known wells) and the output feature (Vsh). The model performs well with an R^2 score for the training data of 0.8840 while the score on the blind well is 0.8833. The score on the total dataset is 0.9557. This project further displays the power of a neural network, and how they can be used to model non-linear models.

TABLE OF CONTENT

LIST OF FIGURES

1.0 INTRODUCTION

In the oil and gas industry, a common step in the exploration of hydrocarbon is reservoir description. Rock properties are sought out to be understood better in terms of their spatial distribution and variability. This property ranges from porosity, permeability, lithology etc. A high-level description of the said reservoir rock is achieved by utilizing geophysical logs, seismic, cores, outcrops and well testing.

A plethora of geophysical logs are in existence and the information about the subsurface they give both as stand-alone and in combination with others is enormous. Therefore, it is common practice to acquire these logs to aid hydrocarbon resource evaluation. However, most times than often, these logs contain missing or incomplete information because of cost limitations or borehole problems. Due to the importance of these well logs, a technique as to be adopted to try to compensate for the missing logs or incomplete log readings.

Traditionally, to address this issue, people adopt empirical models and statistical relationships. For example, Gardner et al. (1974) proposed an empirically derived relationship between seismic P-wave velocities to the bulk density of the lithology in which the wave travels. Also, Greenberg and Castagna (1992) suggested a general method to predict shear-wave velocity in porous rocks from P-wave velocities (Vp). These estimations have proven to be useful, to some extent. However, the derived physical model always involves some degree of simplification, assumptions and subjective experiences, in which the quality of the synthetic well log curves cannot be guaranteed.

With the advent of machine learning and more sophisticated computing, a data-driven approach is now been adopted for the prediction of well log measurements. This data-driven approach utilizes the power of deep learning and its capability to solve complex problems without explicitly programming or assuming any conditions. These techniques are fast becoming widely accepted and are now being applied to various aspects of the petroleum industry. In the area of reservoir description, Salehi et al. (2017), Rolon et al. (2009), and Zhang et al. (2018) applied Artificial Neural Networks (ANN) in the generation of synthetic well logs. In this study, ANN will be used in the prediction of the Volume of Shale (Vsh) from conventional geophysical logs.

The Volume of Shale (Vsh), as the name implies, is the volume of shale in a given volume of rock. Generally, shale affects the response of the various logging devices and also properties such as porosity and fluid saturation. Therefore, it is imperative to have an estimation of the Vsh. The volume of shale (Vsh) is best estimated by logging measurements that respond primarily to shale,

such as, gamma-ray and spontaneous potential (SP). However, other log measurements can also be used in estimating Vsh.

1.1 AIM

The aim of this project is to build a model that has the capabilities of predicting the Volume of Shale from other geophysical logs.

1.2 OBJECTIVES

- Wrangle and process the data to the desired format;
- Carry out quick look and statistical analysis of the well log data;
- Develop a model using a neural network framework; and
- Deploy the model and investigate its performance.

1.3 LOCATION AND DATASET

The geophysical logs used in this study are from the Volve field located in the southern part of the Norwegian North Sea, at a water depth of around 80m (Figure 1). It is situated approximately 200km west of Stavanger and 8km from the Sleipner Ost field. Oil was discovered at the Volve field in 1993. The field was formed by the collapse of adjacent salt ridges during the Jurassic period. The oil is located in the middle Jurassic Hugin sandstone formations. Production of oil from the field started in February 2008. The data is made available through the open-source subsurface data published by Equinor to aid researchers and academics with real-life data. In this project, twenty-one (21) wells are available for study with a total of fifty-seven (57) geophysical logs with variable availability in the wells.

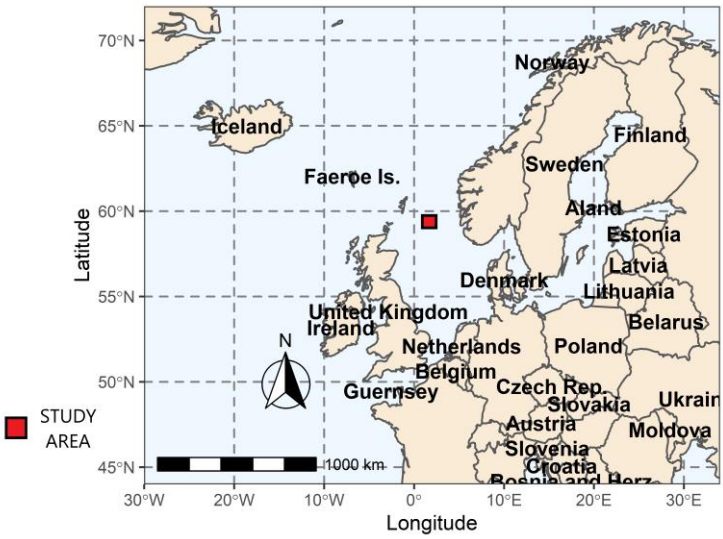


Figure 1: Map showing the study location

2.0 DATA PROCESSING

This project involves using an AI-based model to predict well log measurements. The data utilized is a fairly large dataset of well logs from the Volve field located in the North Sea. Phase 1 of the project involves wrangling the data to the desired format which is the first step of a Data Analytics project. The data is a CSV file that contains log information for different wells. The task is to read this data from the different files and merge it into one file, then clean it through steps described below. A cross-match table that shows the availability of logs for different wells and the fraction of the measured depth which have the specific log for the well will be generated and saved as a spreadsheet.

To carry out these steps, different python packages and modules will be used. These are the ***Pandas*** library, the ***Numpy*** package, and the ***OS*** module. The Pandas library makes importing, analyzing and visualizing data much easier. On the other hand, NumPy is the fundamental package for scientific computing with Python. While the OS module in python provides functions for interacting with the operating system which will come in handy to loop through all the files to be read using Pandas.

Since the data type is *.csv* the *read_csv()* function of the Pandas library is utilized to load the data. This function reads only one file at a time, but the goal is to automate the loading of the data and merge all the loaded data together. This is where the OS module which interacts with the operating system comes into play. The *os.listdir()* is used which takes the path of the directory containing the data as arguments and returns a list containing the names of the files in the directory. These names are then used in a loop to append the data into a DataFrame.

The Merging is accomplished by using the *os.path.join()* to combine the path of the directory with the file name in the list returned from the previous step. This will create a path that can be passed to the *read_csv()* function of Pandas to load that particular file. This is carried out in a for loop, which loops through the list of filenames in the directory containing the data, where the DataFrame created from loading the *.csv* file is appended using the *.append()* function to the master DataFrame called "**data**". Data is a DataFrame that is made-up of all the content of the *.csv* files in the Directory.

A summary of the data DataFrame is returned, utilizing the *.shape* attribute of a DataFrame. The *.shape attribute* returns a tuple (r,c) where r is the number of rows in the DataFrame and c is the number of columns in the DataFrame. To obtain the number of wells, the **wellName** column is

utilized which contains the unique well names for the different wells. Using the *.nunique()* function on that column will return the number of wells in the DataFrame. The number of logs in the data is the number of columns minus 4 (where 4 represents the number of columns that is not a log which is; 'wellName', 'datasetName', 'MD (M)' and 'TVD (M)').

The next step is to clean the data and remove unwanted measurements. The 'datasetName' column is removed using the *.drop()* function. To replace all the "-9999" values with NaN, the Numpy package is leveraged. Numpy has a *numpy.nan* method that returns NaN. *.replace()* function is used to change all the "-9999" values in the DataFrame to *numpy.nan*. As part of the data cleaning step, all columns with less than 10,000 measurements are to be dropped. The *.count()* function is operated on each column and any column less than 10,000 is dropped using the *.drop()* function. A summary of the new DataFrame after cleaning is then returned.

The first crossmatch is a count of valid well log measurements for each well. To obtain this, the DataFrame is grouped by the **wellName** using the *.groupby()* function and the *.count()* function is used to return the number of non NaN values for each log in the well. To create the second cross-match table which shows the percentage of valid well log measurements, the cross match 1 result which is the grouped DataFrame (grouped by **wellName**) is then divided by the **"MD (M)"** count values and multiplied by 100. To make the third cross-match table which has "X" representing logs that have at least one measurement and blank space representing logs with no measurement, conditional statements are used. *cross_match_3[cross_match_3>0]* is used to return all the values of the cross_match DataFrame that has a count greater than 0. This is then assigned to "X" and *cross_match_3[cross_match_3==0]* is assigned to an empty string (" "). The three cross-match tables created are then saved into the system's memory as a *.xls* file using the *.to_excel()* function.

2.1 DATA VISUALIZATION

To visualize the well logs, the ***matplotlib*** module in python is utilized.

The bit size (BS) and caliper (CALI) log will serve as the basis of the selection of the wells which will be visualized. To return a list of the well names which have BS and CALI log, a for loop is used which searches the cross_match_3 earlier created. Only wells in this list will have the logs plotted.

The figure generated for each well is a 1 row, 5 column image with each of the column serving as the track on which the logs will be plotted on. The first track will contain the BS and CALI log.

The Gamma Ray log will be a plot on the second track. A list is created for all other logs except BS, CALI and GR logs which are plotted already. *np.random.choice()* function is used to select three of these other logs randomly to be plotted on the last three tracks.

Swapping the x-axis from down to up: the *.twiny()* function in matplotlib is used to create a twin axes sharing the y-axis. What this function does is to create a new axes instance with an invisible y-axis and an independent x-axis positioned opposite to the original one (i.e. at the top). The y-axis autoscale setting will be inherited from the original Axes.

On the first track, two logs are plotted (BS and CALI). The limit of the x-axis for these two logs is set fixed and the same in other to carry out a qualitative assessment of their relationship. In between these logs are then filled with yellow to make their interpretation easier i.e: the relationship between these two logs. On the second track, the GR log is plotted. The GR log serves many purposes, and one is as a lithology identifier based on the radioactive signature of the rocks. A cutoff of 75 is selected in which rocks with GR > 75 are assumed to be shaly and GR < 75 is assumed to be sandy. This cutoff is, thus, used as a basis for filling the logs. The sandy rocks are colored yellow, while the shaly is colored brown. On the last three tracks, the randomly selected wells are plotted with different colors (green, blue and purple).

The images are saved in a new folder called "Log_Images". The individual images are named using the well names and saved in a *.png* format.

2.2 MODEL BUILDING

To accomplish the goal of this project of predicting the Volume of Shale, a model is developed and optimized. However, before this model is developed, final data preparation is carried out to wrangle the data into a format the algorithm used to create the model, neural networks, will work best. In this final data processing step, all the wells that have less than 16 logs are removed. Also, logs that are available for less than 16 wells and not present in at least 1 well are removed. In the data, any record with missing values is also removed.

This final processing of the data reduced it from 483,550 records and 41 logs obtained in the initial preprocessing stage to 76,039 records and 16 logs which are now well suited for the neural networks i.e. no missing values in the records.

The next step was to carry out feature reduction. Feature reduction has to do with excluding the features from the dataset which will have little to no impact of the model building. In order to select this feature, the Pearson correlation was carried out for all the 16 features (figure 2). On the

basis of correlation, PHIF (V/V) and SAND_FLAG were dropped because they were closely related to RHOB (g/cm^3) with a correlation coefficient of -0.96 and -0.80 respectively. Also, TVD and MD were dropped intuitively because the depth measurement will more features but no apparent relationship with the VSH which is sought out to be predicted.

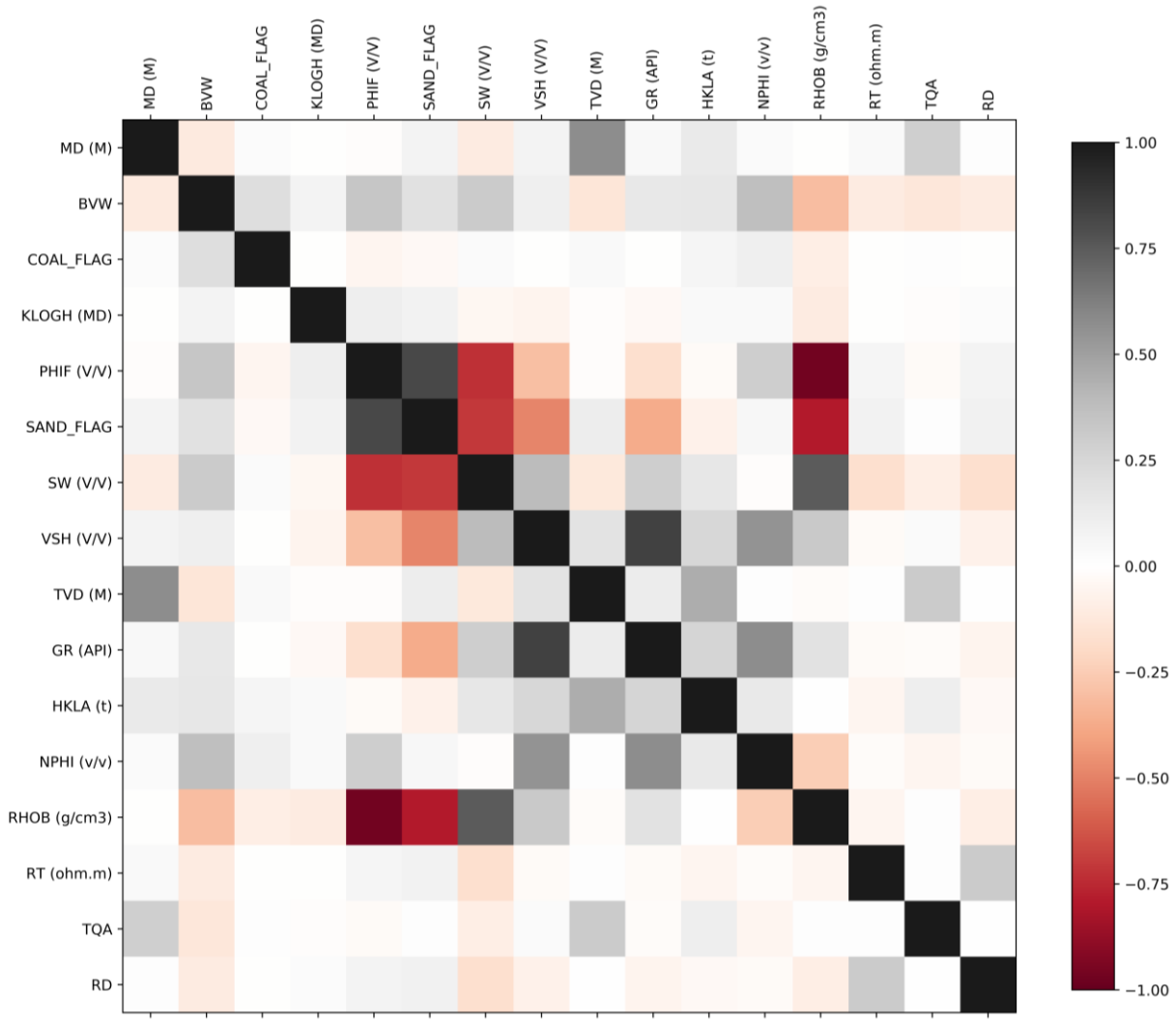


Figure 2: The relationship between the features obtained by Pearson correlation.

After all the data preparation and feature reduction prior to model building, the available data for training and verification of the model is made up of 76,039 records and 12 features. What this means is that the data is from 12 wells. 1 well data (well 15_9-F-1) is set aside from the training process for blind validation.

To build the model, SciKit-learn in Python is utilized. Scikit-learn is a library in Python that provides many unsupervised and supervised learning algorithms. The supervised learning algorithm used in this project is called Neural Networks. Neural networks are a set of algorithms,

modeled loosely after the human brain, that is designed to recognize patterns. They interpret sensory data through a kind of machine perception, labeling or clustering raw input. The patterns they recognize are numerical, contained in vectors, into which all real-world data, be it images, sound, text or time series, must be translated.

Neural networks are made up of layers which in turn are made up of nodes. Computation occurs at the node, which can be likened to what happens in a neuron in the human brain. This node combines input from the data with a set of coefficients, or weights, that either amplify or dampen that input, thereby assigning significance to inputs with regard to the task the algorithm is trying to learn; e.g. which input is most helpful is classifying data without error? These input-weight products are summed and then the sum is passed through a node's so-called activation function, to determine whether and to what extent that signal should progress further through the network to affect the ultimate outcome, say, an act of classification. If the signals pass through, the neuron has been "activated" (https://skymind.ai/wiki/neural-network). Figure 3 depicts what one node might look like.

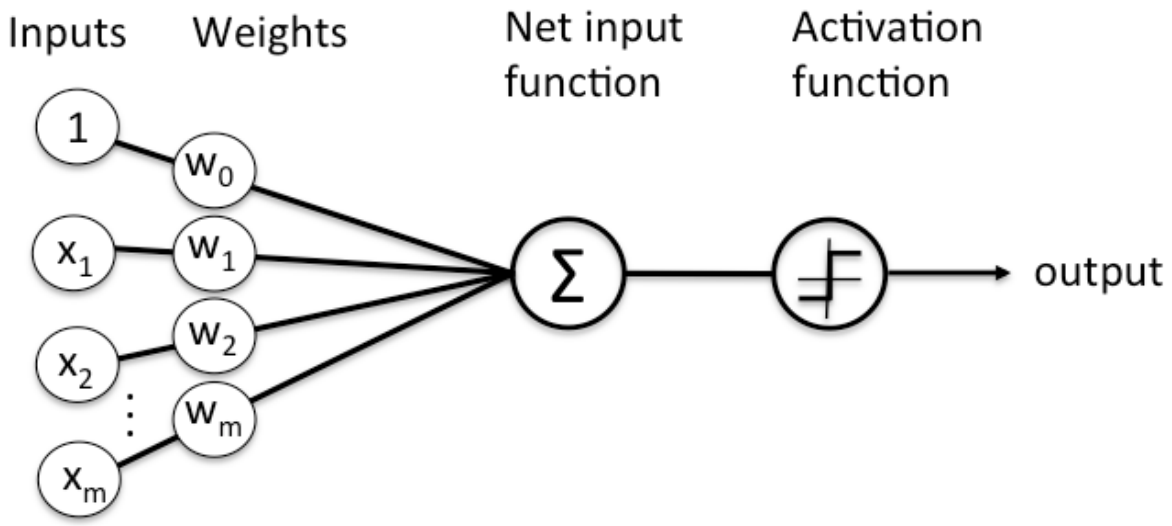


Figure 3: Representation of the various components of a node.

The module used in this project in the SciKit-learn neural networks package is called the MLPRegressor. The MLPRegressor, short for multilayer perceptron regressor, is a class of feedforward artificial neural network (ANN). The term MLP is used ambiguously, sometimes loosely to refer to any feedforward ANN, sometimes strictly to refer to networks composed of multiple layers of perceptron (with threshold activation).

An MLP consists of at least three layers of nodes: an input layer, a hidden layer, and an output layer. Except for the input nodes, each node is a neuron that uses a nonlinear activation function. MLP utilizes a supervised learning technique called backpropagation for training (Rosenblatt, 1961). Its multiple layers and non-linear activation distinguish MLP from a linear perceptron. It can distinguish data that is not linearly separable.

To account for the variance and difference of scale of the input feature, the data is scaled. The MinMaxScaler is utilized which reduces all the feature values to a minimum of 0 and a maximum of 1. To prevent redundancy in code lines, a Pipeline is used to encapsulate all the steps in the model which is to transform the input using a scaler and then build the model before inverting the output back to the initial scale.

To obtain an unbiased model with optimal performance. Some of the hyperparameters of the MLPRegressor model are tuned. A cross-validation algorithm referred to as GridSearchCV is utilized to achieve the best model with optimized hyperparameters. The hyperparameters that were tuned are hidden layer size and number of nodes, solver and activation function.

The model was trained using a cross-validation fold of 5. The best model (highest R^2 score) was saved and then deployed on blind well which was not involved in the training to access the performance of the model. The model was then deployed on all wells contained in the dataset.

2.3 RESULTS

2.3.0 Data Processing

The raw dataset comprised 21 wells that had 57 geophysical logs plus measured depth values associated with them. In total, the raw dataset has 483,550 records with 61 features. The first preprocessing step to remove the datasetName, replace the -9999 values and remove any column that has less than 10,000 measurements reduced the dataset to 483,550 records with 41 features.

The first step in the processing of data for the training and deployment of the model was to remove logs available in less than 15 wells. These resulted in 14 logs in the new dataset. The next step was to remove wells that have less than 16 logs which reduced the number of records to 409,949. The neural network requires input records that have no missing values, therefore, any incomplete rows in the dataset are removed. This resulted in the final dataset used in this project to be made up of 76,038 records from 15 wells with 14 features.

2.3.1 Data Visualization

The result of the log visualization step for Well 15_9-F-11 B is shown below (figure 4). This is important for quick-look analysis and to have a visual representation of the data. Images for all other 8 wells with both BS and CALI is in the appendix.

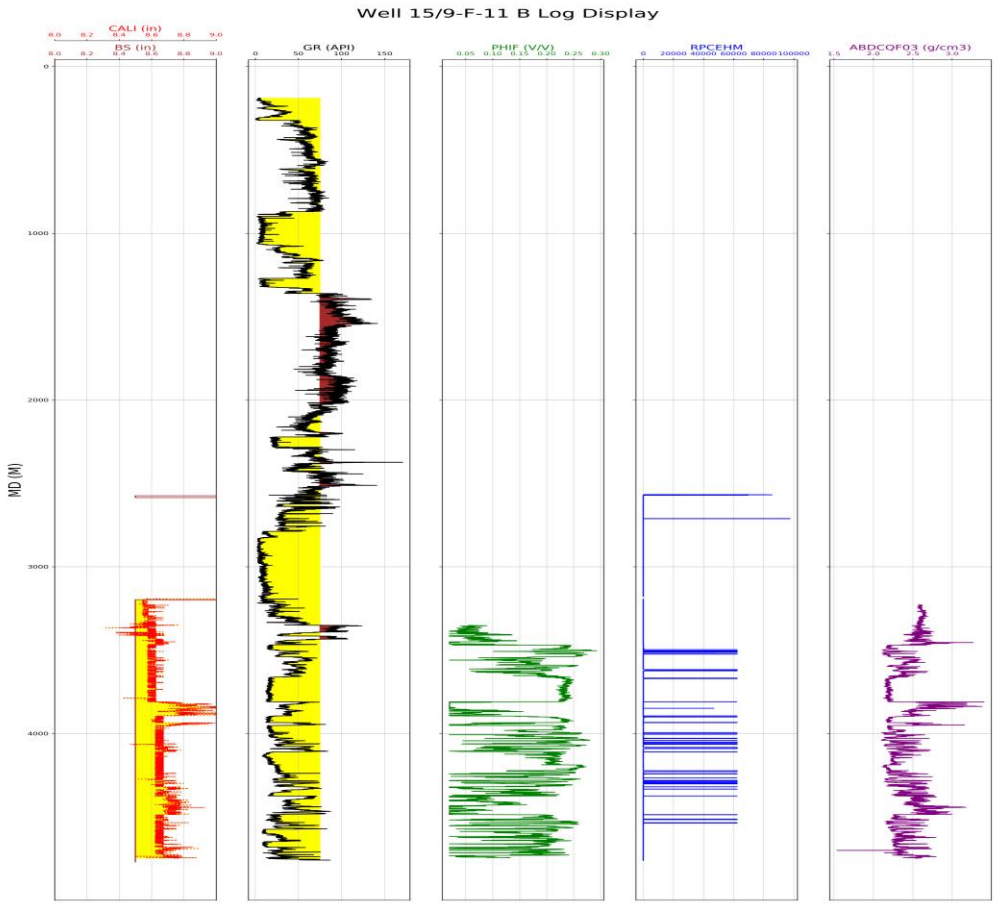


Figure 4: Log display for well 15_9-F-11 B.

2.3.2 Model Building

As described in the method, an MLPRegressor is used as the basis of building this model. This estimator has several hyperparameters. To get the best model, these hyperparameters need to be tuned. The dictionary shows the different hyperparameters that were tuned and the values that were tested by the cross-validation algorithm.

{'hidden_layer_sizes': [(50,),(100,),(150,),(200,),(50,50),(100,100),(150,150),(200,200)],
 'max_iter': [200,300,500],
 'solver':['lbfgs', 'adam']}

It took about 45 minutes to train the model with the dataset except the blind well using a cross-validation fold of 5. The best estimator has the following hyperparameters.

MLPRegressor(activation='relu', alpha=0.0001, batch_size='auto', beta_1=0.9, beta_2=0.999, early_stopping=False, epsilon=1e-08, hidden_layer_sizes=(150, 150), learning_rate='constant', learning_rate_init=0.001, max_iter=200, momentum=0.9, n_iter_no_change=10, nesterovs_momentum=True, power_t=0.5, random_state=None, shuffle=True, solver='adam', tol=0.0001,validation_fraction=0.1, verbose=False, warm_start=False)

The R^2 score for the best model is 0.8840 while the score on the blind well is 0.8833. The score on the total dataset is 0.9557.

The predicted log and the actual log is plotted as shown below (figure 5). The error (actual-predicted) is calculated and plotted on a second plot.

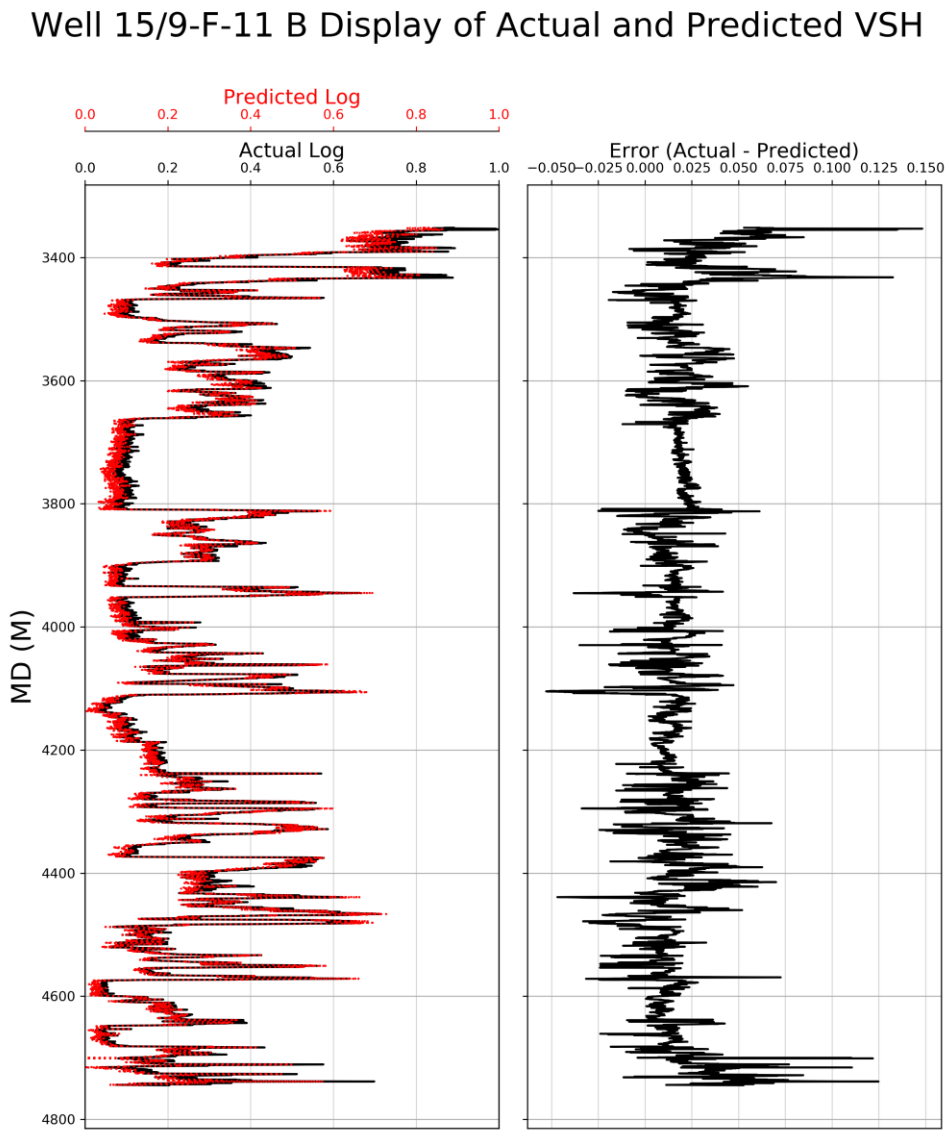


Figure 5: A plot showing the actual and predicted logs with the associated error.

3.0 CONCLUSION

This project emphasized the applicability of using machine learning techniques in the prediction and estimation of geophysical logs. The volume of Shale, a very important rock property is estimated from other wells. A typical workflow is carried out in this project, where the task starts with data cleansing and preprocessing. The relatively good R^2 scores further prove that neural networks have the capability to learn non-linear models. This is an advantage over the traditional statistical techniques of estimation where numerous assumptions and simplification is carried out to solve the problem.

4.0 REFERENCES

Gardner, G.H.F., Gardner, L.W. and Gregory, A.R., 1974. Formation velocity and density—The diagnostic basics for stratigraphic traps. Geophysics, 39(6), pp.770-780.

Greenberg, M.L. and Castagna, J.P. [1992] Shear-Wave Velocity Estimation in Porous Rocks: Theoretical Formulation, Preliminary Verification and applications. Geophysical Prospecting, 40, 195–209.

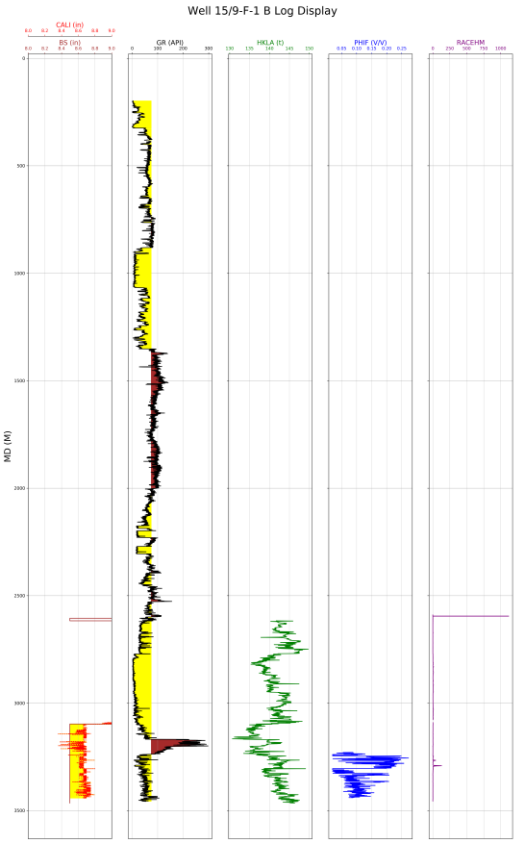
Rolon, L., Mohaghegh, S.D., Ameri, S., Gaskari, R. and McDaniel, B. [2009] Using artificial neural networks to generate synthetic well logs. Journal of Natural Gas Science and Engineering, 1(4), 118 – 133.

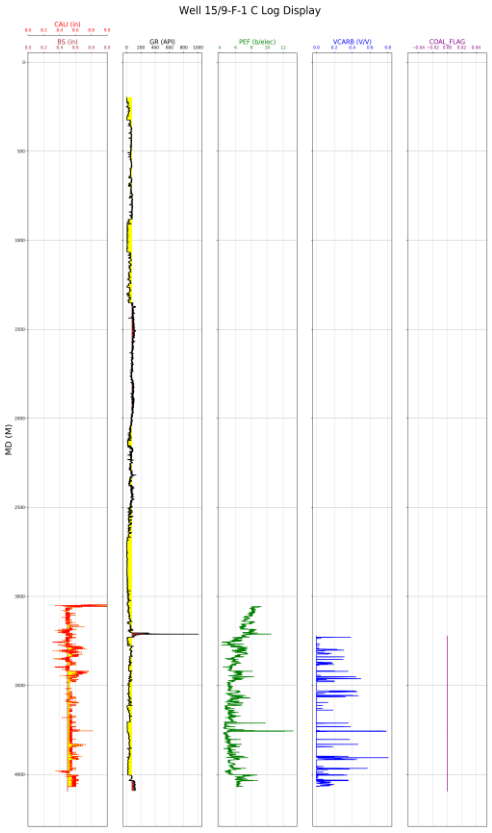
Rosenblatt, F., 1961. Principles of neurodynamics. perceptrons and the theory of brain mechanisms (No. VG-1196-G-8). Cornell Aeronautical Lab Inc Buffalo NY.

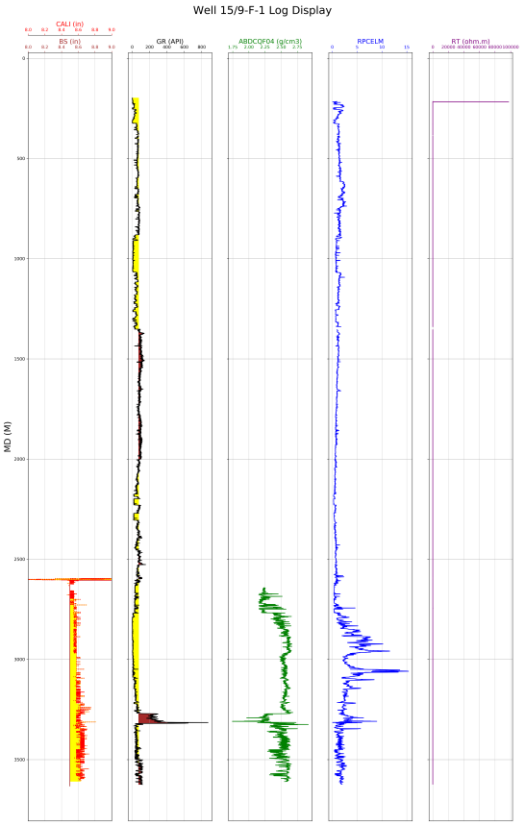
Salehi, M.M., Rahmati, M., Karimnezhad, M. and Omidvar, P. [2017] Estimation of the non records logs from existing logs using artificial neural networks. Egyptian Journal of Petroleum, 26(4), 957 – 968.

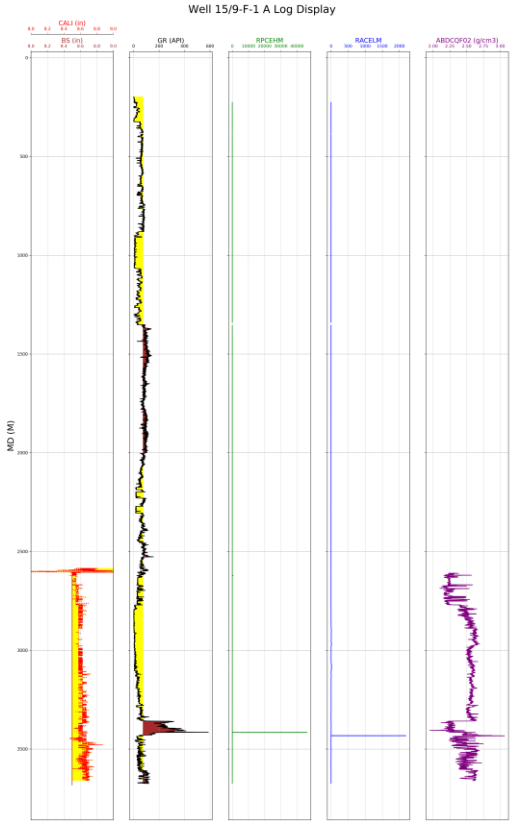
Zhang, D., Chen, Y. and Meng, J. [2018] Synthetic well logs generation via Recurrent Neural Networks. Petroleum Exploration and Development, 45(4), 629 – 639.

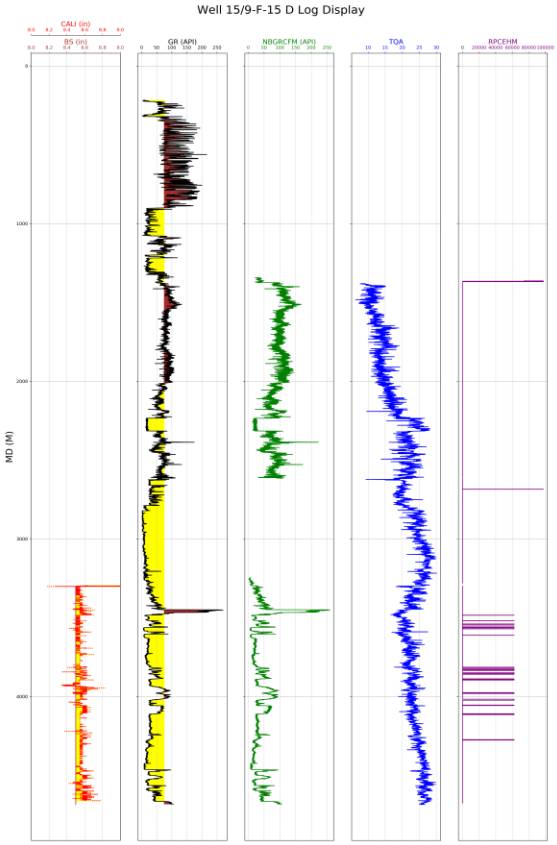
5.0 APPENDICES

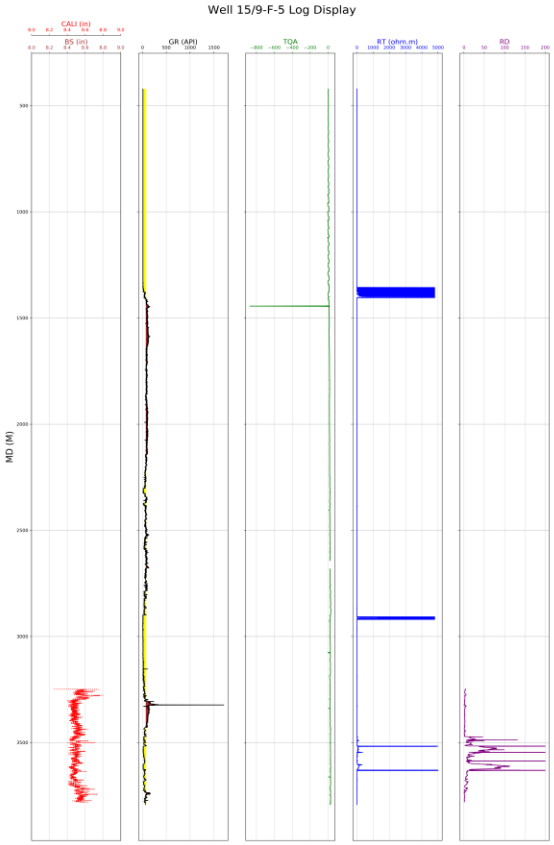


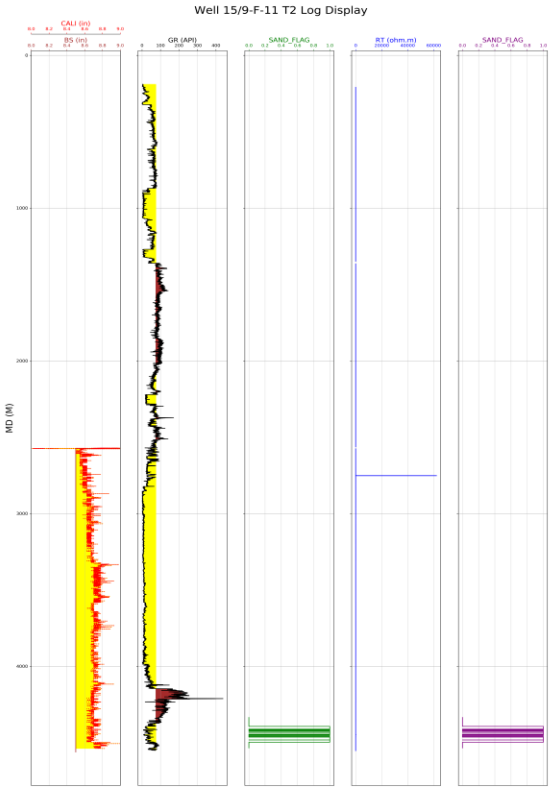


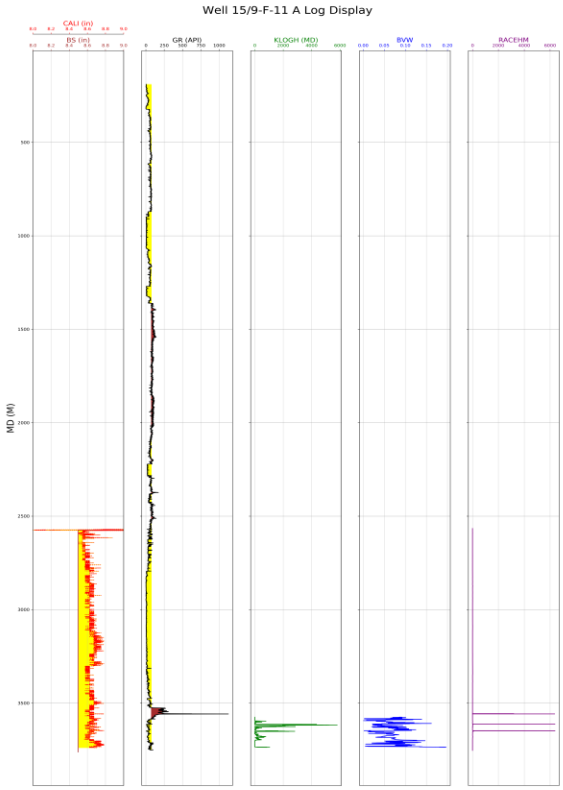




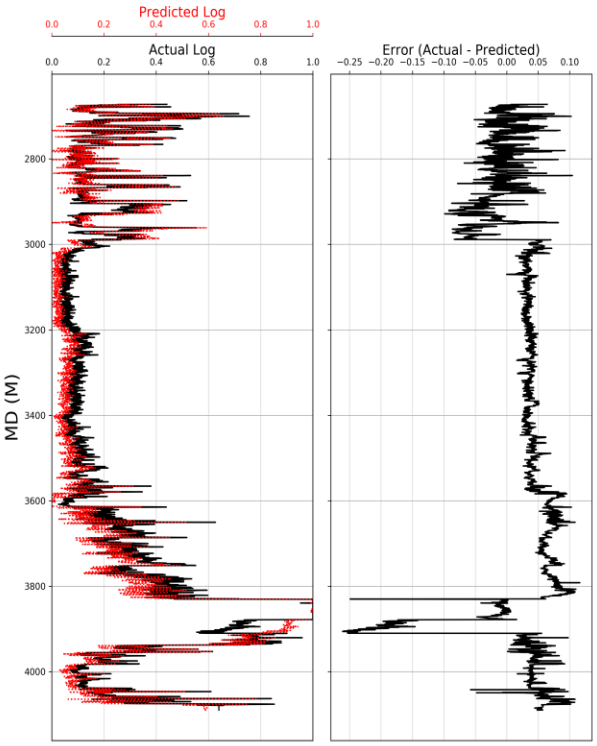

Well 15/9-F-15 D Log Display
CALI (in)
BS (in)
GR (API)
NBGRCFM (API)
TOA
RPCEHM
MD (M)


Well 15/9-F-5 Log Display
CALI (in)
BS (in)
GR (API)
TOA
RT (ohm.m)
RD
MD (M)

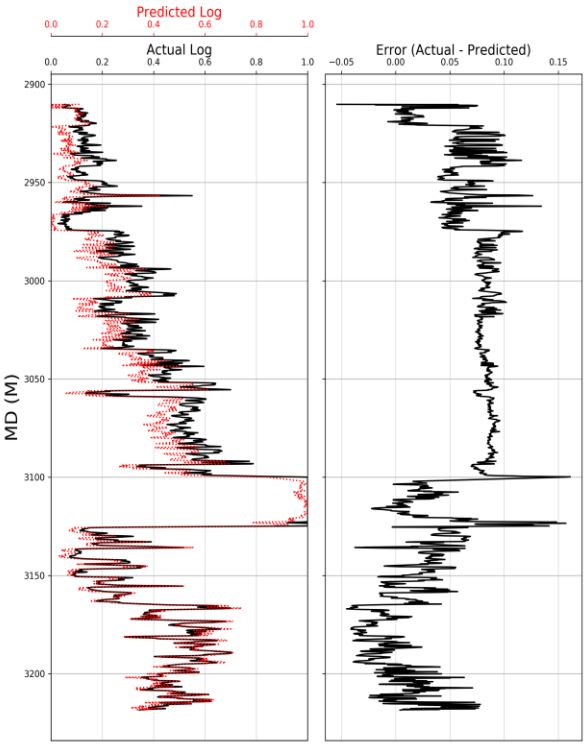

Well 15/9-F-11 T2 Log Display
CALI (in)
BS (in)
GR (API)
SAND_FLAG
RT (ohm.m)
SAND_FLAG
MD (M)


Well 15/9-F-11 A Log Display
CALI (in)
BS (in)
GR (API)
KLOGH (MD)
BVW
RACEHM
MD (M)

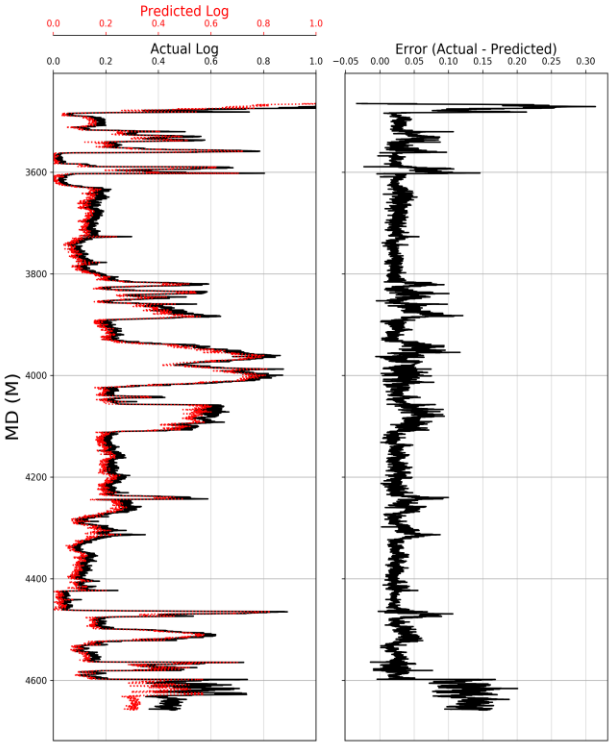
Well 15/9-F-15 Display of Actual and Predicted VSH



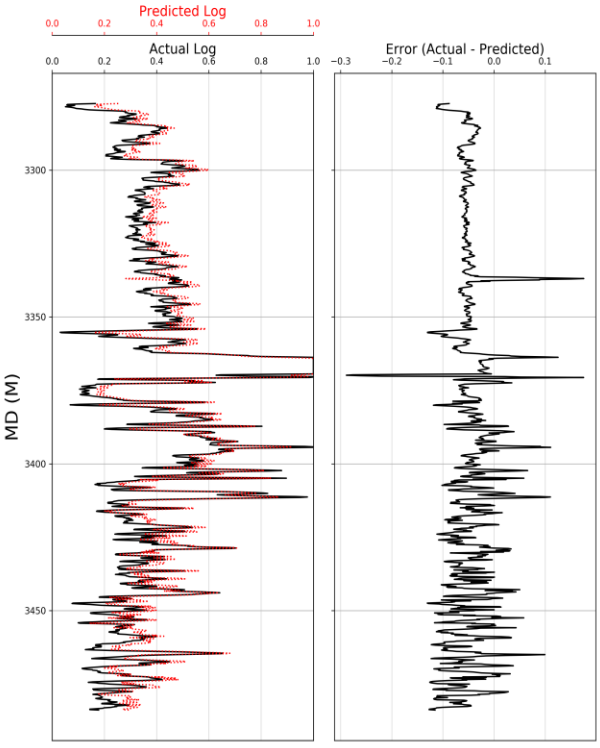
Well 15/9-F-15 C Display of Actual and Predicted VSH



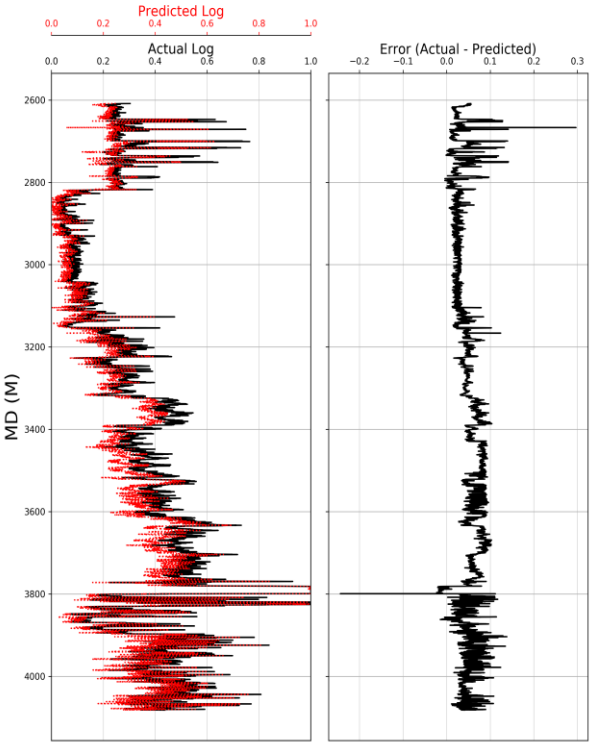
Well 15/9-F-15 D Display of Actual and Predicted VSH



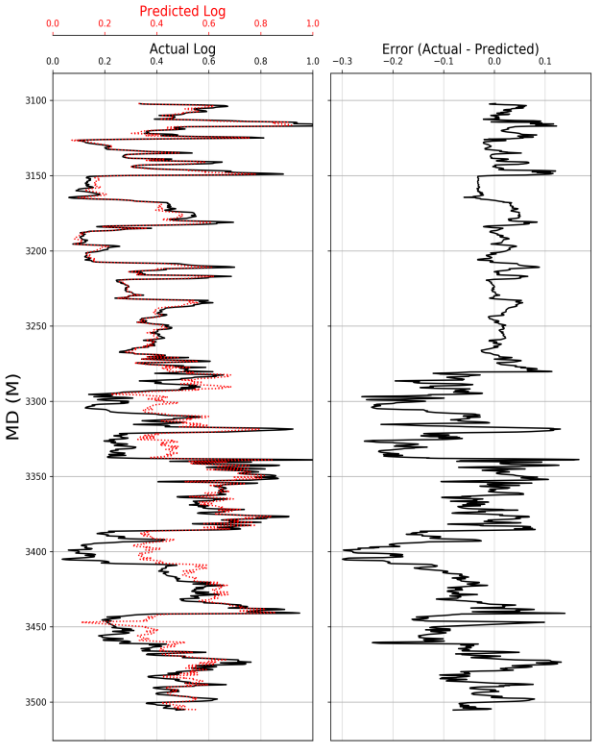
Well 15/9-F-15 B Display of Actual and Predicted VSH



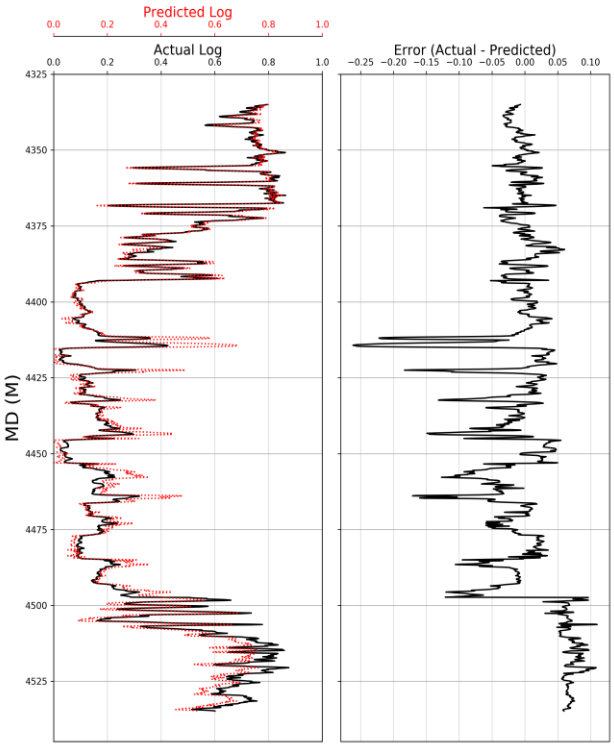
Well 15/9-F-15 A Display of Actual and Predicted VSH



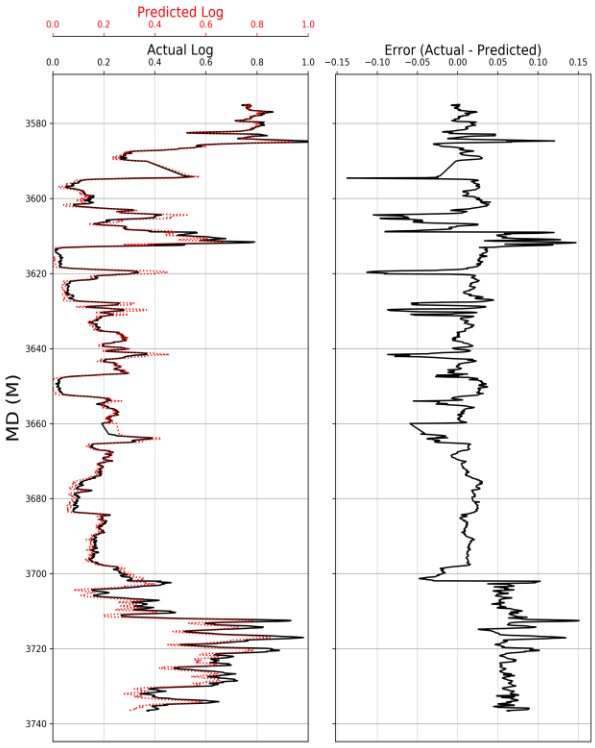
Well 15/9-F-12 Display of Actual and Predicted VSH



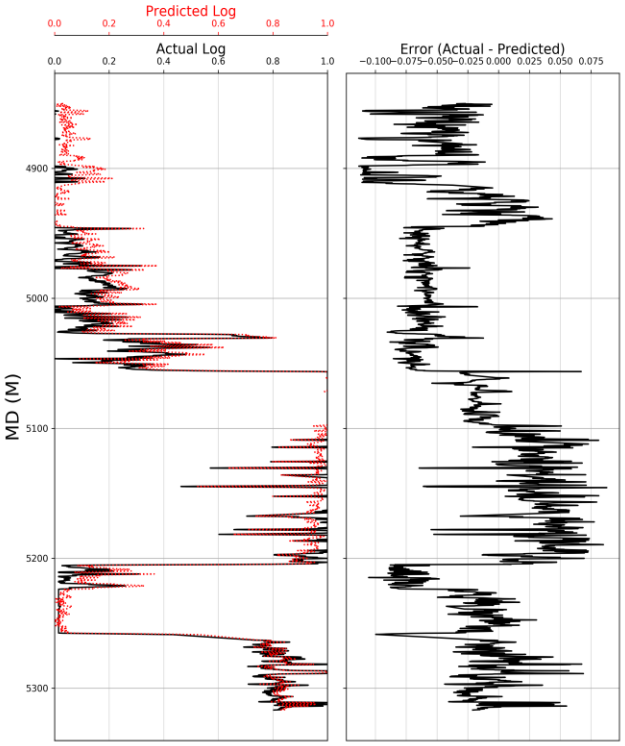
Well 15/9-F-11 T2 Display of Actual and Predicted VSH



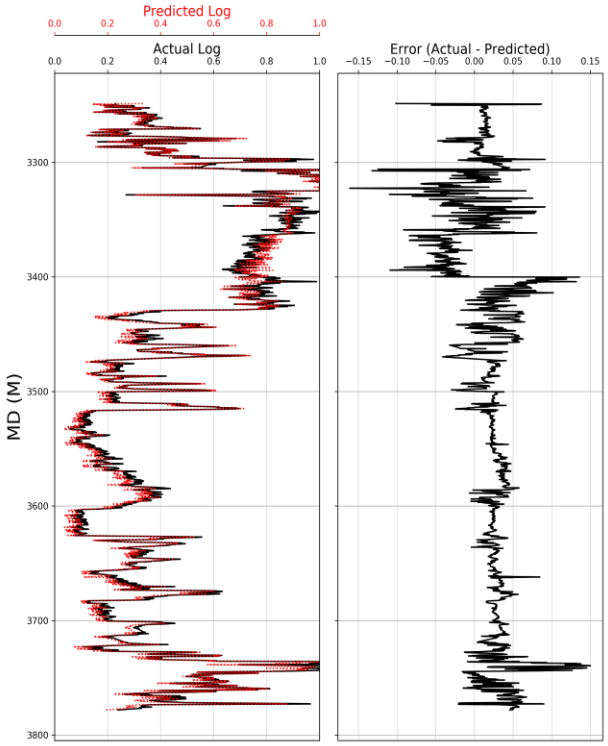
Well 15/9-F-11 A Display of Actual and Predicted VSH



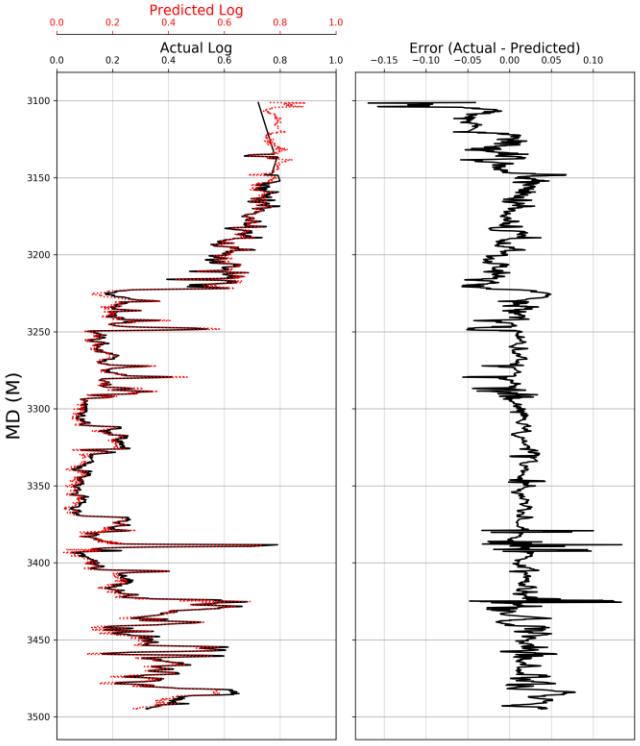
Well 15/9-F-10 Display of Actual and Predicted VSH



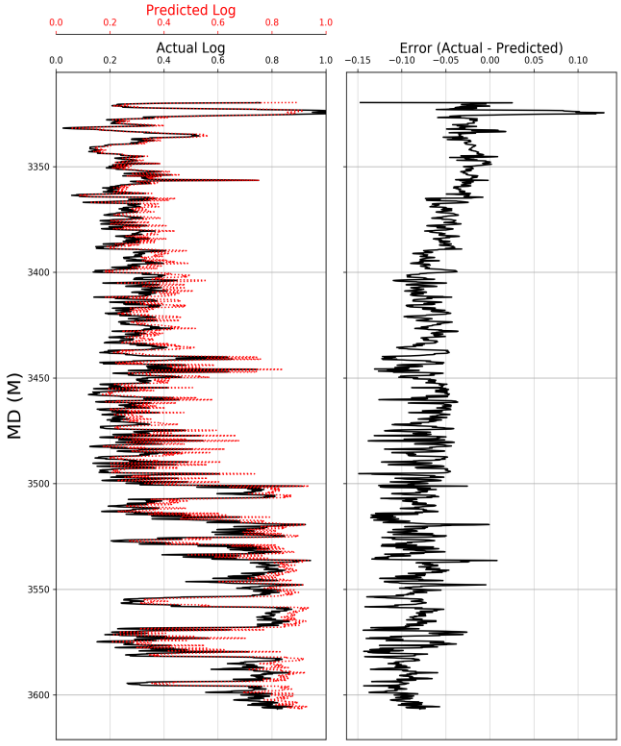
Well 15/9-F-5 Display of Actual and Predicted VSH



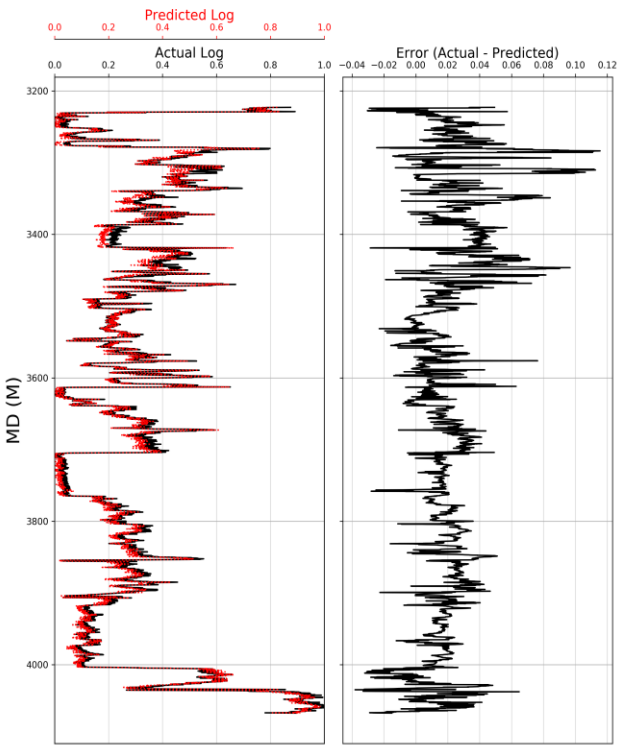
Well 15/9-F-4 Display of Actual and Predicted VSH



Well 15/9-F-1 Display of Actual and Predicted VSH



Well 15/9-F-1 C Display of Actual and Predicted VSH



Well 15/9-F-1 B Display of Actual and Predicted VSH