Performance of Machine Learning Algorithms on Tabular Data for Analyzing Well-Logs

Completed for the Certificate in Scientific Computation and Data Sciences
Spring 2024

Zoe Neale
BSA in Mathematics
Department of Mathematics
The University of Texas at Austin

Sergey Fomel

Sergey Fomel
Wallace E. Pratt Professor of Geophysics at the University of Texas at Austin
Department of Earth and Planetary Sciences

Abstract: *Taking place in 2020, a contest was conducted by the Society of Petrophysicists and Well Log Analysts for contestants to compete to produce the lowest root-mean squared error value when predicting compressional and shear travel-time log values. Machine learning algorithms were employed by contestants, choosing from various techniques. The absence of these values in a well log typically comes from insufficient funding or supplies, so producing an efficient prediction is crucial. The scientific goal of the project is to understand whether tree-based or deep-learning algorithms perform better for this application and what the basis is for this result.To develop this understanding, review and analyses of existing data and Python Jupyter notebooks with regression model solutions as well development of new attempts is crucial. Data is manipulated and predicted using Python packages including pandas and scikit-learn. Other factors must be considered in order to better understand the reasoning for a successful algorithm.*

I. INTRODUCTION

A. SPWLA 2020 Petrophysical Data-Driven Analytics Contest

In 2020, the Society of Petrophysicists and Well Log Analysts (SPWLA) conducted a competition to test the ability of machine learning algorithms to analyze well-log information and complete pseudo sonic log generation. In the competition, all competitors were provided an existing set of well-log data with the goal of using a chosen machine-learning algorithm to predict the Compressional Travel-Time log (DTC) and Shear Travel-Time log (DTS) for each observation. These components are oftentimes missing from some data, due to the fact that acquiring them for all wells in a field can be financially and operationally straining. Entries into the competition used different machine learning algorithms, namely some tree-based models such as randomforests and some deep learning models, being neural networks.

B. Machine Learning

According to El Naqa and Murphy, “Machine learning is an evolving branch of computational algorithms that are designed to emulate human intelligence by learning from the surrounding environment (El Naqa et al., 2015).” Machine learning is effective in predicting data that is missing from a dataset, since it uses an artificial intelligence to learn from patterns that exist within data. We can see this artificial intelligence exhibited through a number of different algorithms, but those most prevalent in this research are linear regression, neural networks, and tree-based models such as randomforests. The algorithms that are used are all supervised learning algorithms, meaning that they predict unknown values from provided, known values in the data with a labeled output.

Examples of supervised machine learning are classification and regression. For this research, I focused solely on regression models. Regression models are used to predict continuous numerical values, whereas classification models are used to predict a discrete label or categorical variable.

a. Linear Regression

A linear regression is a model that is used frequently in mathematics to predict a dependent, y, variable from an independent, x, variable (Montgomery et al., 2021). Linear regression adopts the form of $f(X) = \beta_0 + \sum_{j=1}^{p} X_j \beta_j$. In order to acquire the linear

regression equation, an intercept, β_0, and unknown coefficients, β_j, must be calculated from the existing data and X_j is a quantitative input (Akinnikawe, 2018).

b. Deep Learning and Artificial Neural Networks

Deep learning is a fragment of machine learning models that is based on artificial neural networks. An artificial neural network is, in a sense, a computer programmed brain. An artificial neural network mimics the synaptic connections in the brain. The neurons in the model are organized into input and output layers, there can also be hidden layers. A deep neural network is one that has one or more hidden layers (Janiesch et al., 2021).

c. Tree-Based Models

In machine learning, tree-based models are based on variations of decision trees. "Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression."[1] Decision trees operate from the root of the tree, following a series of child nodes, ultimately reaching a leaf node that provides an output.

i. Randomforests

The first tree-based regression model that will be used in this research is a RandomForestRegressor. Randomforest combines several randomized decision trees and proceeds to aggregate their decisions by averaging (Biau et al., 2016).

C. Importance of Accurately Analyzing Well-Logs for Various Applications in the Oil and Gas Industries

There are a number of reasons for the creation of synthetic well-logs. At the forefront of this are the demand for oil and gas suffering from a great increase, need for insights in reservoir studies, oil production, and the emerging business of carbon capture utilization and storage (Choubey, 2021). For the specific application of this research, predicting shear and compressional sonic logs, the reasoning behind the need for a synthetic well-log is because these logs can be expensive and difficult to acquire. These logs have correlations with other logs and when a successful model is used and the necessary data is provided, the problem can be resolved.

D. Existing Research on Machine Learning Algorithms Applied to Well Log Data

Synthetic Well Log Generation Using Machine Learning Techniques: In the study conducted in the Unconventional Resources Technology Conference, machine learning algorithms were used to create synthetic well logs. The synthetic well logs being predicted were photoelectric (PE) and unconfined compressive strength (UCS) logs. The existing logs used to predict PE were gamma ray, density, neutron, resistivity, and volume of clay. Those used to predict UCS were gamma ray, density, porosity, neutron, clay volume, kerogen volume, compressional slowness, shear slowness, Young's modulus, and Poison's ratio. The success of each model was measured by the least average squared error (ASE) returned from the validation data. The study included models of Linear Regression, Artificial Neural Networks (ANNs), Decision Trees, Gradient Boosting, and Random Forest. The Random Forest model computed the lowest ASE value for validation when predicting PE, whereas for UCS the lowest ASE value was computed via the neural network model (Akinnikawe 2018).

Prediction of sonic log and correlation of lithology by comparing geophysical well log data using machine learning principles: The logs included for the petrophysical

[1] https://scikit-learn.org/stable/modules/tree.html#

evaluation in this study included a majority overlap with the logs used in my research. The logs considered were Caliper (CAL), Sonic (DT), Gamma Ray (GR), Neutron (NPHI), Density (RHOB), Self-Potential (SP), Resistivity (RACEHM, RACELM, RD, RM, RPCEHM, RPCELM, RT), and others. The methodology included data cleaning, preparation, and visualization prior to using unsupervised learning via clustering for lithology prediction. A linear regression model was used to predict sonic log and a correlation matrix was generated to determine that S wave velocity had the strongest correlation to sonic log. The classification model from the k-means clustering algorithm was used for lithology prediction (Joshi et al., 2021).

Evaluation and Development of a Predictive Model for Geophysical Well Log Data Analysis and Reservoir Characterization: Machine Learning Applications to Lithology Prediction. The data used to conduct this study was composed of 40,080 data points including density porosity, neutron porosity density, gamma ray index (IGR), volume of shale in accordance with four different empirical correlations, interval transit time (delta T), resistivity, and shear and compressive wave velocities. Classification with a k-means clustering algorithm was also used in this case. The measured properties were plotted as a function of depth for the available well-logging data. The approach in this study was almost the reversed version of Joshi's study above. The linear regression model was implemented following the clustering. Two metrics were used to measure performance, root-mean-square error (RMSE, which is the standard statistical metric to measure model performance) and R-squared. The study concluded that there was "enormous potential to transform log data into cohesive, correlated categories for reliably predicting lithology and reservoir characterization" (Mishra et al., 2022).

E. Review Studies Comparing Tree-Based Models and Deep Learning Models on Tabular Data

Why do tree-based models still outperform deep learning on typical tabular data? The study considered 45 tabular datasets with various domains, each fitting the specified qualifications. Two different metrics were used to measure the performance of the models - using test set accuracy for classification and R2 score for regression. Three tree-based models were considered in the analysis: RandomForest, GradientBoostingTrees, and XGBoost. It was found that the performance gap between neural networks and tree-based models stays wide, and that the best methods on tabular data are ensemble methods with decision trees used as the weak learner. It was also found that neural networks struggle to fit irregular functions, when compared to tree-based models, since neural networks are biased to smooth solutions. Multi-layer perceptrons are also less robust to uninformative features. The conclusion stated that tree-based models more easily yield good predictions with less computational cost than the alternatives (Grinsztajn et al., 2022).

Tabular data: Deep learning is not all you need

A wide variety of datasets including both classification and regression tasks were used in this study. Typical training and preprocessing took place before fitting and predicting with the models. For a majority of the datasets, an 80:20 train/test split was used; for other data, a split provided by the authors was used 10:20:70 for validation, test, and train, respectively. XGBoost was used as the baseline model. It was determined that it is more challenging to optimize deep learning models than for XGBoost (a tree-based model). Finally, despite there being progress using deep learning models, they do not outperform XGBoost in this study (Shwartz-Ziv et al., 2022).

Comparative Analysis of Tree-Based Models and Deep Learning Architectures for Tabular Data: Performance Disparities and underlying Factors

This study analyzed the strengths and limitations for each type of machine learning model considered. The deep learning models held strength in their ability to learn relevant feature representations and ability to handle both structured and unstructured data, making them versatile. Yet, there were limitations found in the necessity for large datasets to perform well which led them to not be feasible for all applications. Deep learning models are also difficult to train and tune. Tree-based models exhibit more strengths than limitations according to the research done in this study. They generally have a successful predictive performance on a wide range of tabular datasets and they provide feature importance scores. Some models excel at categorical features and eliminate the need for extensive preprocessing. Overall, modern tree-based models are found to be robust against overfitting and can handle noisy data well.

The study concluded that tree-based models outperformed deep learning on all datasets. Tree-based models are able to learn complicated non-linear correlations between characteristics and the target variables. Tree-based models were faster on larger datasets and more robust to outliers than deep learning. The interpretability was also a consideration and it is found that the predictions are easier to understand in a tree-based model than in a deep learning approach. Ultimately, the choice of model is dependent on the data's nature, interpretability, and the task (Rana et al., 2023).

F. Key Findings and Limitations of Previous Research

When considering the previous research, the key findings encompassed that tree-based models outperformed deep learning models over a variety of domains. It is also important to consider that there is no typical benchmark for every application, so when deciding which model is best, the domain must be considered. The ease of use was also a large factor when choosing a model. The ability to understand a tree-based model was significantly greater, whereas deep learning models such as neural networks produce large arrays when attempting to understand weights and biases in a network. The majority of the studies considered covered an emphasis on classification, thus there is research lacking in terms of regression models. Classification models are key in understanding the lithology of a rock formation, but regression models must be used to predict numerical logs that are more expensive. Given the nature of the previous research, my research predicts that tree-based models will outperform deep learning models for this application.

II. MATERIALS AND METHODS

A. Software Used

In order to complete the research desired, several software programs were essential in understanding the topic as a case study and also for conducting the experiment. The provided materials existed in Google Colaboratory (also referred to commonly as "Google Colab") in the form of a Python Jupyter notebook. The competition submissions were stored as Jupyter notebooks as well, which were then adapted and ran through Google Colab as well. Google Colaboratory is used as a computing infrastructure platform to run Jupyter notebooks for data science and data modeling (Bisong, 2019).

Within each Jupyter notebook, three major Python libraries were used to manipulate data and use machine learning algorithms. These major Python libraries are pandas, matplotlib, and scikit-learn, in order of appearance in the established code. The

pandas library is commonly used when working with structured datasets. The library is a Python data analysis library that includes a large number of the functions used in this research.

The pandas functions used in the data served primarily to manipulate and pre-process the data. The functions used for indexing and iteration in the research are functions such as `DataFrame.head([n])`, which returns the first n rows of the dataframe and `DataFrame.loc` which accesses a group of rows and columns by label or boolean array. The `DataFrame.corr` and `DataFrame.describe` functions are used for descriptive statistics. These are just a few of the functions that exist in the library that are used to manipulate the input data.[2]

The matplotlib library was utilized frequently throughout the experimentation process as well. Matplotlib is a Python package that is used for data visualization (Yim et al., 2018). The pyplot module is what was used in this work and is regularly used for simplistic plotting. The library is also highly integrated with pandas, which is also being used in this research. The package is primarily utilizing functions such as `plot`, `scatter`, and `hist` to visualize relationships between attributes in the imported dataframe.

The final library that was used to facilitate this research was the scikit-learn library. scikit-learn is a machine learning library that was written in Python (Kramer, 2016). This package was essential for the research being done because it contains methods for regression that are used to provide the answers to the scientific questions and hypotheses surrounding this topic. Some of the functions used from this library include supervised learning algorithms such as linear regression which uses functions in `linear_model` such as `LinearRegression()`, `fit(X,y)`, `HuberRegressor`, and many others that will be discussed throughout the methodology and results.

B. Understanding the Dataset

The dataset provided for contestants in the competition is the same dataset being used for my research process. The data is imported as a dataframe with 30,143 entries and 9 columns. The nine columns are all different types of well logs acquired through most of the wells. The 'train.csv' dataset that is provided by the Society of Petrophysicists and Well Log Analysts (SPWLA) contains well logs that are easier to acquire such as gamma ray, resistivity, and density, among others. compressional travel-time and shear travel-time well logs are also included in the data.

All nine columns in the 'train.csv' dataset are CAL (Caliper og), CNC (Neutron log), GR (Gamma Ray log), HRD (Deep Resistivity Log), HRM (Medium Resistivity log), PE (Photo-Electric Factor log), ZDEN (Density log), DTC (Compressional Travel-time log), and DTS (Shear Travel-time log). To understand the data best, it is imperative to understand what each of these logs is. In understanding the types of logs, I will discuss each in the order they appear in our data.

Caliper logs are used to measure the size of a borehole, in the given data the measurement appears in units of inches. A borehole is a narrow shaft in the ground that is either vertical or horizontal. A caliper log is used to measure the diameter of a borehole because although the borehole width should be the outside diameter of the drillbit used, oftentimes the hole may be different in size. This difference comes from outside factors

[2] https://pandas.pydata.org/docs/reference/frame.html

such as washout, collapse, or build-up. This varies based on the type of surrounding rock. In our data, the observation with the largest borehole diameter measures at 21.064200 inches and the smallest is 5.930400. The typical range sits at 7-⅞ inches to 12 inches.

In generating a histogram of the frequency of CAL values, the distribution appears as right-skewed (Figure 1). I then chose to explore only the CAL observations in the typical range. This histogram does not display as clear of a distribution but it appears to have the greatest frequency for values between 8 and 9 (Figure 2).

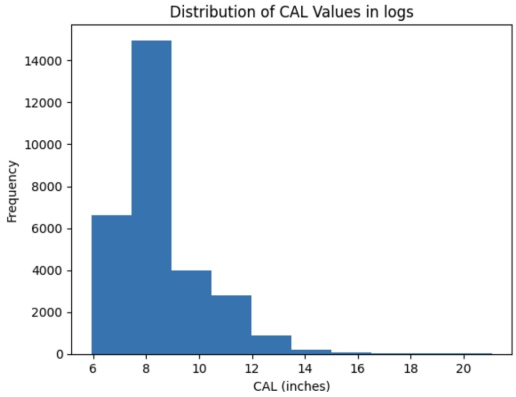

Figure 1: Distribution of CAL values in logs

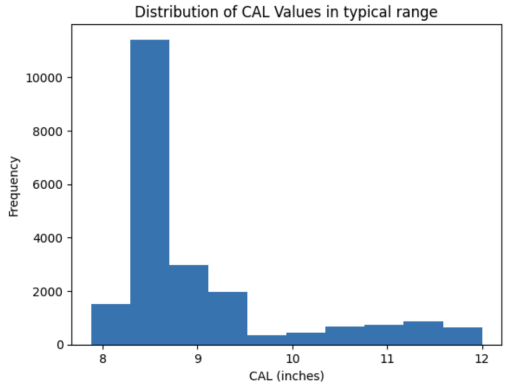

Figure 2: Distribution of CAL values in cal_typical (typical range)

The second column in the initial dataset is a neutron log, labeled as CNC. The neutron log is a porosity log that measures the hydrogen concentration in a formation. The units used in a neutron log are different from other logs, as they are not basic physical units. In this case, the unit that is provided for the data is “dec” or decimal units. Porosity of the rock is important because it is a characteristic that can be used to determine the type of rock being observed. In the data, the majority of CNC values are between 0 and 350.

Gamma ray logs are one of the “easy-to-acquire” conventional well logs as stated by the competition publisher, the Petrophysical Data-Driven Analytics Special Interest Group (PDDA SIG) within the SPWLA. Gamma ray logs are used to measure the natural radioactivity in formations (Asquith et al., 2004). Gamma rays are measured using the American Petroleum Institute (API) unit. Gamma ray logs are valuable in identifying lithologies and correlating between formations.

The two subsequent columns of the data are both measures of resistivity - HRD (Deep Resistivity log) and HRM (Medium Resistivity log). Each of the two are measured in units of Ohm per meter. The data includes a medium resistivity log and a deep resistivity log since resistivity measurements recorded at different distances into the formation can have different values. Resistivity measures the reservoir’s fluid saturation on the basis of porosity, fluid type, amount of fluid, and type of rock.

Since resistivity is a function of porosity, among other values, and we have an attribute that measures porosity (CNC), it may be useful to observe what the correlation between these values looks like (Figures 3 and 4). In observing the correlation between CNC and the different resistivity measures, there is a similar relationship between each resistivity and the porosity measure. To ensure that there is a difference between the

values of HRD and HRM, I chose to also plot the relationship between those two values. It was found that these values are different, as their relationship would exhibit a trend of x = y if each value had the same measure (Figure 5).

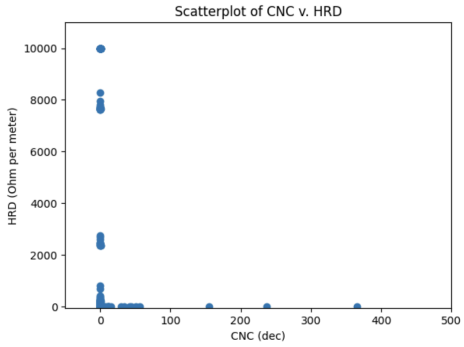

Figure 3: Scatterplot of CNC vs. HRD

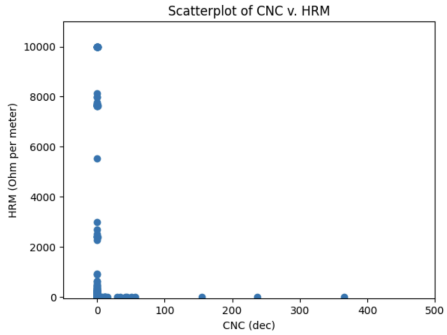

Figure 4: Scatterplot of CNC vs. HRM

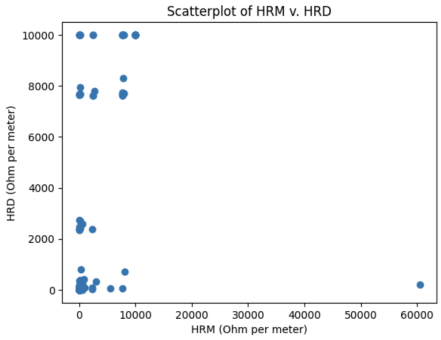

Figure 5: Scatterplot of HRD vs. HRM

The next column of our dataset to consider is PE (Photo-Electric Factor log) which is measured in Barn units, which is a metric unit of area equal to $10^{-28}m^2$. The photo-electric factor log is also referred to as the litho-density log and is a new form of the density log with added features (Mondol, 2015). Since PE is a new form of a density log, I used the pandas function pandas.corrcoef to establish whether or not PE and ZDEN (the density log) had a linear relationship, the result did not return a high positive or negative value, thus it is determined that they do not have a linear relationship. In plotting the relationship between the two, there was not an immediately appealing association between the two.

Logically, the next attribute in the columns to be explored is ZDEN (the density log). The density log is measured in grams per cubic meter. The density loh uses two separate density values - the bulk density and the matrix density. The bulk density is the density of both solid and fluid parts of the rock formation, whereas the matrix density is the density of the solid framework of the rock (Asquith et al., 2004). After plotting a scatter matrix (Figure 6) of all of the attributes in our dataset, it is determined that ZDEN is an adequate predictor of DTC and will be used frequently in exploratory data analysis.

Density data is also often combined with porosity, so the plot of this in the scatter matrix is also notable.

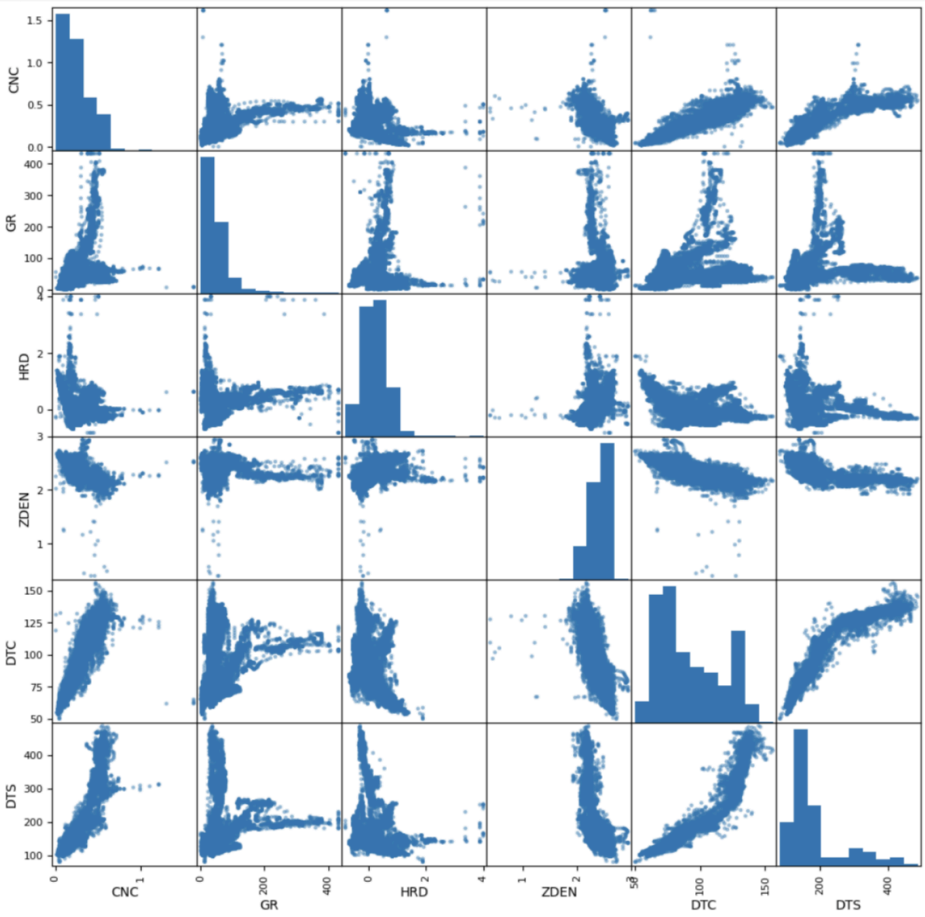

Figure 6: Scatter Matrix of Well-log Data

The two final columns in the dataframe are DTC and DTS, which are the Compressional Travel-time log and Shear Travel-time log, respectively. The units for both of these are nanosecond per foot. DTC and DTS are the two sonic logs which are established as the target of prediction from the other seven "easy-to-acquire" logs. These sonic logs are often missing or incomplete in many oil or gas wells (Yu et al., 2021). Before replacing the baseline of -999 that is used to represent a missing value with "NaN", I chose to count the number of missing values for each variable. This provided the insight that DTC was missing 4054 values and DTS was missing 4865. Thus, the aim for the competition was to use other provided data to predict these missing values.

In order to best understand which logs are the best predictors of DTC and DTS, a correlation matrix was generated over the data. In a correlation matrix a value at or close to 1 indicates a strong positive correlation, whereas a value at or close to -1 indicates a strong negative correlation. It is also true that a value at or close to 0 indicates no correlation. The correlation matrix from the logs dataframe is depicted in Table 1. Based on the values computed in the matrix, it is shown that DTC and DTS have a strong positive correlation at 0.941639. However, since both have missing values in the log, the second best correlation is between DTC and ZDEN, which is a strong negative

correlation, at -0.731694. Thus, we can use ZDEN to predict DTC and then move forward to predict DTS from DTC.

Table 1: Correlation Matrix for 'logs' dataframe

	CAL	CNC	GR	HRD	HRM	PE	ZDEN	DTC	DTS
CAL	1.000000	0.008627	0.020375	-0.009225	-0.005003	0.163847	-0.358636	0.585637	0.681271
CNC	0.008627	1.000000	-0.003242	-0.000660	-0.000410	0.017042	-0.008152	0.008636	0.048946
GR	0.020375	-0.003242	1.000000	0.031601	0.037300	-0.035093	-0.152755	0.355556	0.233226
HRD	-0.009225	-0.000660	0.031601	1.000000	0.497020	0.010584	-0.012363	-0.426659	-0.009533
HRM	-0.005003	-0.000410	0.037300	0.497020	1.000000	-0.000947	-0.004957	-0.002028	-0.005334
PE	0.163847	0.017042	-0.035093	0.010584	-0.000947	1.000000	-0.335354	0.477204	0.440085
ZDEN	-0.358636	-0.008152	-0.152755	-0.012363	-0.004957	-0.335354	1.000000	-0.731694	-0.670186
DTC	0.585637	0.008636	0.355556	-0.426659	-0.002028	0.477204	-0.731694	1.000000	0.941639
DTS	0.681271	0.048946	0.233226	-0.009533	-0.005334	0.440085	-0.670186	0.941639	1.000000

C. Pre-processing

In order to develop further insights from the provided data, the initial step in the methodology is to employ pre-processing techniques over the data. With a dataset as large as the provided 'train.csv', including 30143 observations, it is unlikely that the data is clean upon receiving. The data used for the project arrived in a tabular format and is read in from a csv file using the Python pandas library. The pandas library contains functionality for manipulation and analysis on datasets (McKinney, 2011).

I first used the `describe` function to return the count, mean, standard deviation, minimum value, quartiles, and the maximum value of the initial dataframe (Table 2). Keeping in mind that it had also been established by the SPWLA PDDA that the baseline value for a missing value would be indicated by -999, the results showed that there were missing values in each column. As discussed above, I chose to display the number of missing values in each column which provided that DTC and DTS held the majority of the data's missing values. These missing values were then replaced using the `replace` function to insert a NaN value in place of the previous placeholder.

Table 2: Statistic Description for logs dataframe

	CAL	CNC	GR	HRD	HRM	PE	ZDEN	DTC	DTS
count	29633.000000	29408.000000	29889.000000	29758.000000	29758.000000	29464.000000	29462.000000	26089.000000	25278.000000
mean	8.654281	0.683437	47.780541	16.953912	14.492077	5.173227	2.393818	91.814381	180.655730
std	1.749145	30.689679	51.377519	349.067878	445.361622	4.781088	0.196276	24.337910	81.141960
min	5.930400	-0.102800	-0.146000	0.054100	0.061600	-0.023200	-1.923800	49.970500	80.580400

25%	8.135600	0.127100	18.026100	0.740450	0.734700	0.054200	2.234800	71.357000	129.446625
50%	8.625000	0.198500	37.082200	1.662750	1.665150	5.042500	2.439600	85.237600	144.593050
75%	9.063000	0.343100	58.532800	3.180350	3.308900	7.949700	2.553000	112.112600	191.475125
max	21.064200	3490.158200	1470.253400	10000.000000	60467.761700	28.106400	3.259700	155.980300	487.438400

I then chose to develop the existing problem established by the starter Jupyter notebook provided by Dr. Fomel, 1-Regression.ipynb. The existing code opted to plot attributes ‘ZDEN’ and ‘DTC’ on a scatter plot with density on the x-axis and Compression travel-time, which can also be described as slowness, on the y-axis. It was observed that the dependence was not simple, however there was a general trend of slowness decreasing with density, which was also indicated by the correlation matrix. The dependence was then approximated using linear regression. In establishing an appropriate approximation, it was essential to pre-process the data being used by removing outliers. A large portion of the methodology used in the research was to draw comparisons between data including outliers and the data excluding outliers. This was done to reiterate the significance of preprocessing in data analysis and further establish the difference between success and failure with the algorithms used.

Prior to implementing any of the machine learning algorithms over the dataset to predict missing values, the data was randomly split into an 80:20 ratio for training and test datasets, respectively. This was done before and after removing outliers for the variables that are being used in each model.

D. Predicting DTC from ZDEN

In order to solve the problem and predict the missing values of DTC and DTS from the provided well-log data, the first step was to attempt to predict DTC using various techniques. The first technique used is a simple linear regression. The best predictor was chosen based on the scatter matrix and table of correlation coefficients generated above. As previously mentioned, the chosen variable with the greatest correlation with DTC was ZDEN.

a. Case 1: LinearRegression()

For the first case of prediction, linear regression, only the two chosen logs, DTC and ZDEN are selected. This code is adapted from the initial Jupyter notebook provided by Dr. Fomel. The two logs are selected and held in the variable ‘two’. All ‘-999’ values are replaced with a NaN value. A scatterplot is then produced to determine the locations of outliers in the data. The scatterplot visualizes the relationship between density (ZDEN) and compressional slowness (DTC).

After examining the scatterplot, it is determined that anomalous values, or outliers, are located where ZDEN is greater than 3.1 and less than 1.5. Thus, the data is then refined to exclude the outliers and held in variable name ‘two2’. This reduces the dataset to 25,442 entries. A scatterplot of the refined dataset is then produced for the purpose of comparison and to examine the dependence. The data with and without outliers are both divided into training and testing data on an 80:20 split, respectively. Training data for two and two2 are then visualized in a scatterplot.

Next, a linear regression model is created for each set of data and the model is fit from each set of training data. In fitting the model, the two parameters in the linear model are produced - the intercept and coefficient. For the data with outliers, the intercept is `300.7270359576544` and the coefficient is `[-87.89875294]`. For the data excluding outliers, the intercept is `342.2912250046268` and the coefficient is `[-105.14184477]`. The scatterplot of each is then reproduced, including linear regression lines.

To evaluate the prediction error of each linear regression model, the same metric that was designated in the SPWLA competition is employed, root-mean-square error (RMSE). Root-mean-square error is defined as $\sqrt{\frac{1}{N}\sum_{n=1}^{N}(p_n - v_n)^2}$. The RMSE evaluated for the data including outliers comes out at `16.907`, whereas the data excluding outliers produces a lower RMSE at `14.334`. This is due to the fact that linear regression attempts to predict a linear pattern for prediction, thus any outliers can affect the coefficients produced. Ultimately, this results in a less successful model.

b. Case 2: HuberRegressor()

The second prediction model is a linear model that uses the HuberRegressor() function. The Huber Regressor is an "L2-regularized linear regression model that is robust to outliers."[3] The model has its own loss function that combines L1 and L2 loss. L1 regularization (which is also known as Lasso) is the sum of absolute values and is referred to as the Manhattan distance. L2 regularization (also known as Ridge) is the sum of the squares and is referred to as the Euclidean distance. The HuberRegressor model will be used following the same methodology as in the LinearRegression case. The model will be used on the data including and excluding outliers to produce an RMSE value.

When plotting the data and linear regression lines produced through the HuberRegressor model, the scatterplots and lines produced are nearly identical to that of the linear regression models. However, the intercept, coefficient, and RMSE values vary slightly. The RMSE value for the Huber Regressor model with outliers returned an RMSE of `17.810` and the Huber Regressor model without outliers returned an RMSE value of `14.408`.

c. Case 3: Neural Networks

Next, a neural network was chosen for the prediction model. The neural network being used for this case is one adapted from one of the competition submissions by the first place "UTFE" team. The neural network initially designed by the team was made up of three layers – an input layer of 24 neurons, a second layer of 12 neurons, and finally an output layer with 2 neurons. This model was then adapted in my research to only have one neuron in the output layer. This was done because a neural network should typically have one output for a regression model.

Using this model, the weights and biases were initially explored. However, it became apparent that it was best to present the neural network model as a "blackbox" due to the fact that the weights and biases are displayed in large arrays, making them difficult to interpret. Following this, the training and testing data of the ZDEN and DTC variables

[3]https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.HuberRegressor.html

were instantiated under variables names X and y (for the data including outliers) and X2 and y2 (for the data excluding outliers).

The data was then scaled using the scikit-learn function “RobustScaler”. The RobustScaler was a better choice than the StandardScaler, as it is robust to outliers. Each scaler was fit and then transformed the data. The neural network models were then compiled using the ‘adam’ optimizer and ‘mse’ for loss. The models were then fit and used to predict. When predicting, it was crucial to inverse transform the data afterwards. Then the root-mean-square error was calculated to return the neural net prediction error for data with and without outliers. The data with outliers returned a lower RMSE value at `13.766439428246587`, whereas the data excluding outliers returned an RMSE of `13.971701017491181`.

The actual DTC and the predicted DTC from ZDEN were then visualized through a scatterplot for the data including outliers (Figure 7) and excluding outliers (Figure 8).

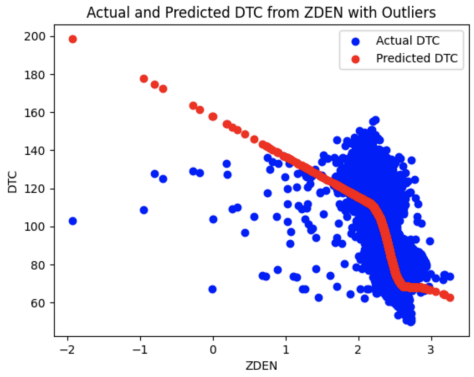

Figure 7: Actual and Predicted DTC v. ZDEN with Outliers

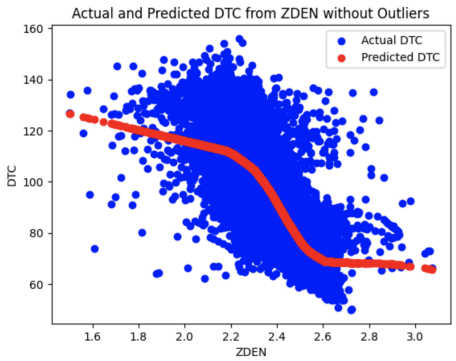

Figure 8: Actual and Predicted DTC v. ZDEN without Outliers

d. Case 4: MLPRegressor

The final model for predicting DTC from ZDEN was to use the MLP (multi-layer perceptron) regression model on the neural network. A multi-layer perceptron is a feed-forward neural network and is employed in the scikit-learn package as a regressor (Popescu, 2009). The same process used for the neural network model to scale the data was followed. The scikit-learn MLPRegressor() was then fit with the correct hidden layer sizes for the data with and without outliers. Then the model was used to predict the DTC values and the numpy expand_dims function was used to ensure the data was the correct shape for the inverse_transform function.

Then, a scatterplot was created to visualize the data including outliers, the data excluding outliers and the predicted DTC from each was added to the plot as well (Figure 9). Finally, the RMSE was calculated in order to acquire a metric to compare the prediction model. The RMSE for the data without outliers was `14.013945712754204`. The RMSE for the data excluding outliers came out as `13.731839399925402`.

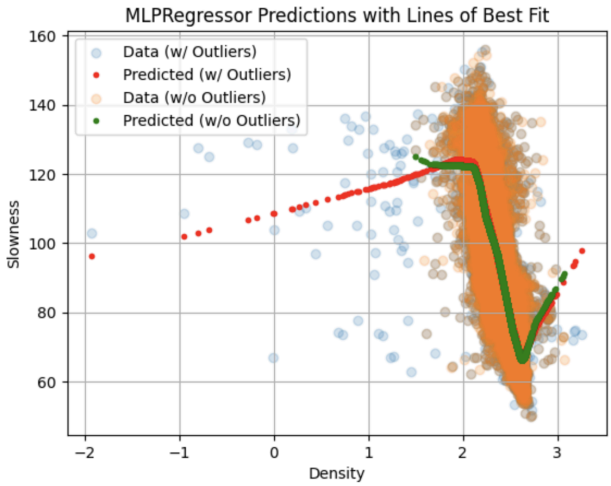

Figure 9: MLPRegressor on Data with and without Outliers

Ultimately, the neural network and MLPRegressor models resulted in data that was more interesting, as it depicted a prediction that was not linear. A non-linear model is more insightful when the two variables do not have a clear linear relationship. The most successful of the models is the MLPRegressor model over the data without outliers, resulting in the lowest RMSE when using ZDEN as a predictor for DTC.

Predicting DTS from DTC

As was determined above through the creation of a correlation matrix, DTS has a strong correlation with DTC, we know this due to the result of the correlation coefficient 0.941639. Since the value is close to positive 1, we know that the correlation between the two is close to a positive linear relationship. Thus, DTC was a good choice for a predictor for DTS. In attempting this prediction, all of the same models that were used in predicting DTC from ZDEN were repeated.

When plotting the relationship between DTC (compressional slowness) and DTS (shear slowness), there was not a clear linear relationship. There was also a lack of clear outliers in the data to remove. However, it can be observed that there are two separate linear dependencies in the data (Figure 10). Therefore, it was decided to split the data around DTC equal to 110.

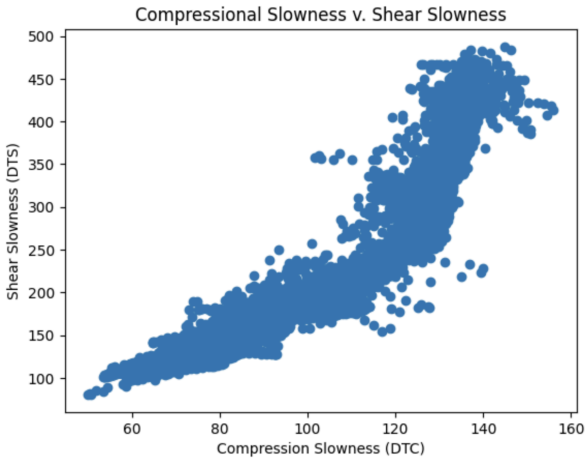

Figure 10: Scatter Plot of Compressional Slowness v. Shear Slowness

a. Case 1: LinearRegression()

For the case of linear regression, the data was referred to in three different ways - the full data (which included all of the data), the first portion of data (where DTC < 110), and the second portion of data (where DTC > 110). Each of these cases were divided into training and testing on an 80:20 split, respectively.

Then, a model was fit and trained for each set of data. From the linear model, regression intercepts and coefficients were acquired (Table 3). After the intercepts and coefficients were found, it was possible to plot the data with linear regression lines for each model. This then indicated that the second portion of the dataset could be refined further to exclude outliers. Thus, a refined dataset where DTC was between 120 and 140 was created. The intercepts and coefficients are also included in Table 3.

	Full Dataset	First Portion of Data (DTC < 110)	Second Portion of Data (DTC > 110)	Refined Second Portion of Data (120 < DTC < 140)
Intercept	-116.53110362740762	-10.043839581215593	-606.9193121999976	-836.5762160646378
Coefficient	[3.38284483]	[1.9744706]	[7.31220504]	[9.07022134]

Table 3: Regression Intercepts and Coefficients for Predicting DTS from DTC

The new linear regression lines for the first portion and refined second portion of data were then plotted over the full dataset (Figure 11). Finally, the prediction error was calculated using a metric of RMSE for the full dataset, first portion of dataset, second portion of dataset, and for the refined second portion of the dataset. The lowest of the four was the first portion of the dataset with an RMSE of 8.753, and the highest was 35.738 for the second portion of data (unrefined, where DTC > 110).

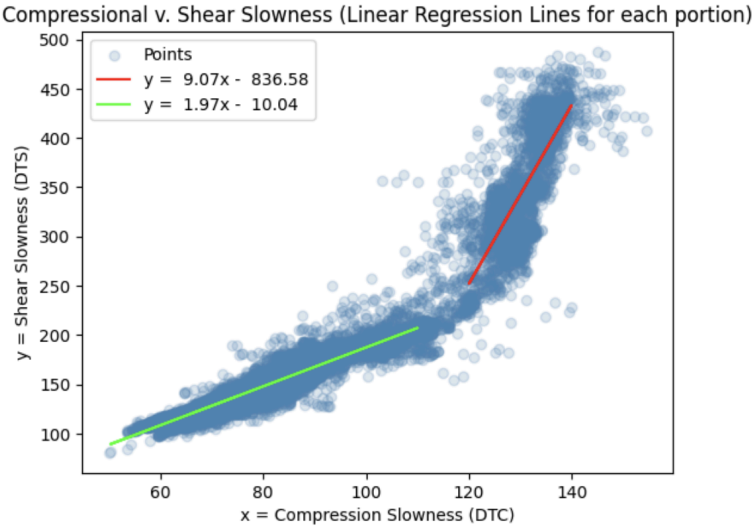

Figure 11: Compressional v. Shear Slowness Linear Regression Model

b. Case 2: HuberRegressor()

The HuberRegressor() model was then used over the three initial data sets - full, first portion, and second portion. The models were fit accordingly, coefficients were calculated using the same processes. When the linear regression lines were plotted from the huber regressor they appeared reasonably similar to the linear regression model. The

lowest RMSE came from the Huber Regressor for the first portion of the dataset at 8.729 and the highest was for the second portion at 36.839. The second portion was significantly higher, so the RMSE for the refined portion was also calculated. This returned a high value as well - 33.667.

c. Case 3: Neural Networks()

The neural network model was utilized with the same methodology as described for prediction DTC from ZDEN and resulted in an RMSE of `8.463051496776881` for the first portion and `39.733090962579475` for the second portion of data. The scatterplots produced proved to be more interesting than the linear regression model, as they were not entirely linear (Figures 12 and 13).

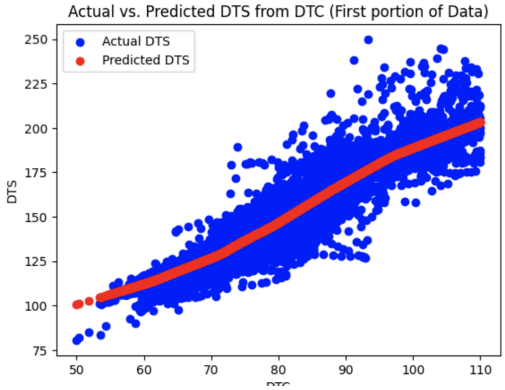

Figure 12: Neural Network Prediction for First Portion of Data

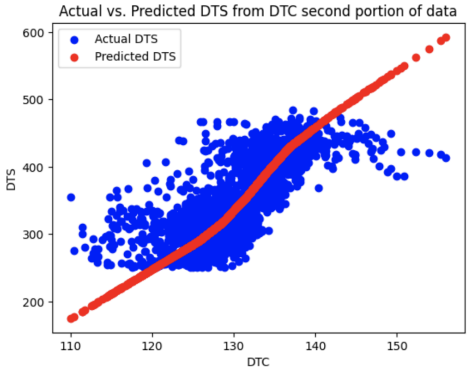

Figure 13: Neural Network Prediction for Second Portion of Data

d. Case 4: MLPRegressor()

The MLPRegressor was also used in the same fashion as for predicting DTC from ZDEN. The data was scaled, models were fit and used to predict, and the prediction error was calculated using the RMSE metric. The RMSE was calculated for the first and second portions of data with values of `8.254726596749308` and `32.26887916891879`, respectively. The actual and predicted data were plotted in Figure 14.

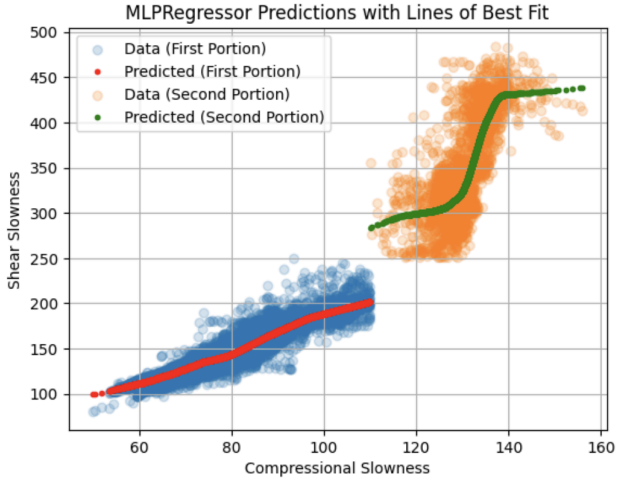

Figure 14: MLPRegressor Actual and Predicted Values of DTS

E. Tree-Based Models

In order to analyze the use of tree-based models for predicting the DTC and DTS well-logs from our existing dataset, it was chosen to analyze and understand the existing submissions for the SPWLA 2020 competition. The models created by the following teams placed in the top rankings and were selected for interpretation: "RockAbusers"[4], and "Explorum."[5] The "RockAbusers" team placed in third and used a RandomForest solution and "Explorum" used a RandomForest solution as well, leading to a twelfth place rank.

a. Case 1: "RockAbusers"

In the third place "RockAbusers" team solution, the team chose to employ a RandomForestRegressor from the scikit-learn library. Before fitting the model, the team classified the targets as DTC and DTS, as designated by the competition directions. The team likely used their existing domain knowledge to select CNS, RHOB (their name for the provided data's ZDEN variable), and PE as the features for prediction.

The model was then established and fit with the arrays of features_1 and DTC. Then an array of features was then used to predict DTC. The same was then completed for the features_2 and DTS arrays. The prediction of the whole dataset was not providing ideal values of RMSE, so the team then elected to section the data into the "top" and "bottom" zones of the well-log. The top included the section 0-2000 and the bottom included the section 2000-end.

Using the new indices, the model was then fit and used for prediction again. The error was then calculated for the DTC model and the DTS model. The RMSE results were calculated, but the GitHub repository for the competition did not include some of the files established by the team, so we lack the information to analyze the RMSE for the top and bottom sections. However, the competition domain indicated that the best RMSE produced by the team came out at 13.2136, which is one of the lowest in the code that has been examined and produced in this research.

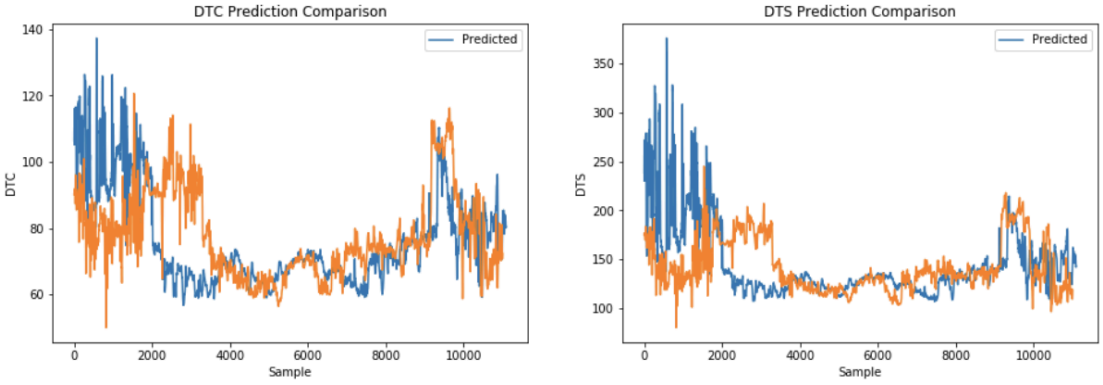

Figure 15: DTC and DTS Prediction Comparisons from "RockAbusers" Solution

b. Case 2: "Explorum"

The twelfth place team, "Explorum", also employed a RandomForestRegressor. However, the use of this function was included in a pipeline established by the team. The

[4]https://github.com/pddasig/Machine-Learning-Competition-2020/blob/master/Solutions/RockAbusers%20Solution%20Submission.zip

[5]https://github.com/pddasig/Machine-Learning-Competition-2020/blob/master/Solutions/Explorum%20solution%20Submission.zip

team created two different pipelines - one for predicting DTC and another for predicting DTS. For the DTC model, the pipeline consisted of two StackingEstimator's from tpot.builtins The StackingEstimator is established in a library called "tpot" and it is used for adding predictions and/or class probabilities as synthetic features.[6] The first StackingEstimator uses LinearSVR as the estimator and the second uses ElasticNetCV. The final part of the pipeline is a RandomForestRegressor.

The second model, the DTS model, includes a StackingEstimator with the RandomForestRegressor as an estimator and RidgeCV. Each model is then fit and used to predict the data. The data is then used to produce a comparison between the real and predicted DTC and DTS values. They are visualized on two different plots as shown in Figure 16. The best RMSE produced by the "Explorum" team was 16.70458. The error in prediction can also be seen when the predictions are visualized, since there is not a directly linear relationship between actual and predicted DTC or actual and predicted DTS.

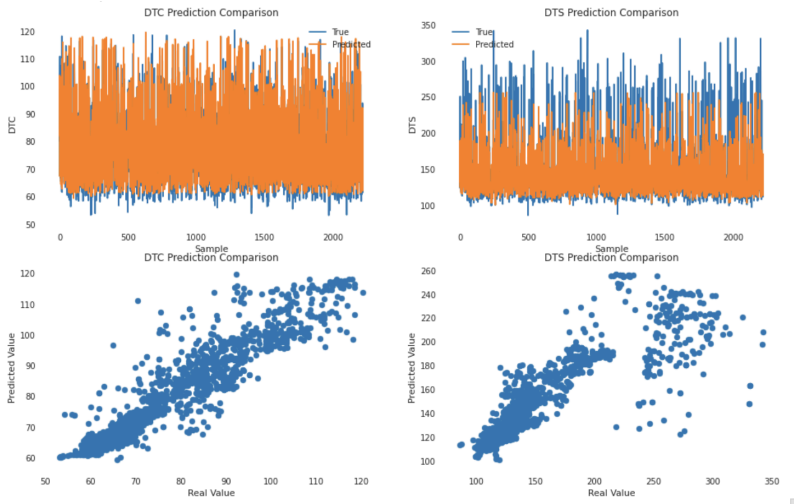

Figure 16: Actual v. Predicted Data from "Explorum"

III. RESULTS

From the research conducted, the answer to the question of whether tree-based models outperform deep learning models on tabular data for analyzing well logs was dependent on several factors. In exploring the various submissions in the machine learning competition, it became apparent that the individual's domain knowledge plays a large role in the success of a model. For example, in the first place UTFE[7] submission, it can be seen initially that the team had prior knowledge regarding rock formations. This is displayed in the decision to section the data into zones before visualizing and initiating prediction.

The use of prior domain knowledge is also observed in the solution completed by third place team, RockAbusers. The team was able to use previous knowledge to understand which well-logs were more interesting when choosing features to predict in the RandomForestRegressor. The team elected to choose CNS, ZDEN, and PE as their

[6] https://github.com/EpistasisLab/tpot/blob/master/tpot/builtins/stacking_estimator.py
[7] https://github.com/pddasig/Machine-Learning-Competition-2020/blob/master/Solutions/UTFE_Code%20.ipynb

features. This knowledge is also apparent when the team re-established their variables by the common names for the well-logs, such as initializing ZDEN as “RHOB.”

In the results produced, it is observed that the first place competition submission was in fact an instance of a neural network, which contradicts my hypothesis. However, in the competition it was apparent that there were disparities in domain knowledge between teams. In the models I produced, it was attempted to predict DTC and DTS each from one feature, to draw a comparison to the linear regression model. The neural network solutions appeared more interesting than the linear regression models, in terms of visualization of the not directly linear relationships between the respective predictors and the targets. The lowest RMSE acquired was through the use of the neural network model on the first portion of the data.

The RMSE was significantly lower than the second portion, which was exceptionally high. This is likely due to the preprocessing steps taken to exclude outliers in each region. The first portion had a more linear relationship. However, it would be interesting to consider what this might have looked like had a tree-based model been employed, since they are more robust to outliers. The results calculated from the code adaptations made reaffirm that data must be sectioned properly in order to account for outliers when using models that are not extremely robust to outliers.

IV. DISCUSSION

A. Limitations

There are still a number of steps I would have liked to take with regards to this research, had there not been a time constraint. The research completed was in an attempt to replicate a linear regression-like pattern or visualization while employing deep learning and tree-based models. For the neural network and MLPRegressor case, this was done successfully and provided insights about the models themselves. However, the next steps that I would have aimed to take would be to run the neural network and MLPRegressor models to use various features for prediction and then to use the tree-based models such as RandomForest, XGBoost, and GradientBoostingTrees using the same methodology in my predicting DTC and DTS code. Following this, I would have instantiated each of these models with an array of features for prediction and predict the targets – DTC and DTS – simultaneously. Performance would have been measured using the same metric, RMSE. Looking even further ahead, I would aim to assess these models for classification problems.

B. Conclusion

In conclusion, with consideration for my computational findings and the previous studies that I have read, it is likely that tree-based models would perform better than deep learning models in this use case. However, the computation completed is insufficient to confirm this conclusion. The 2020 competition revealed a first-place submission with a neural network model, and the first ranking to use a direct tree-based model is the “RockAbusers” third place submission.

However, studies and my research indicate that domain knowledge can play a large role in the model’s success. Through reviewing the submission code, it was apparent that the first place UTFE team had extensive domain knowledge, indicated by the zoning accomplished in the first cell of code. In the case of synthetic well-log analysis, the competition indicated that it is possible, given elaborate preprocessing and hyperparameter tuning, to create a deep learning model more successful than a tree-based

model. However, prior studies indicate that overall tree-based models are more robust to outliers and have a lower computing cost. Thus, making them the superior choice for well-log application.

V. ACKNOWLEDGEMENTS

I would like to acknowledge my supervisor for this research, Dr. Sergey Fomel. I am grateful for the opportunity to work under such an accomplished individual, and to have been provided guidance in my research. This work was supported by The Oden Institute for Computational Engineering and Sciences as well as the Department of Statistics and Data Sciences. Finally, I would like to thank the Computational Science and Engineering certificate program and the Scientific Computation and Data Sciences certificate program for the opportunity to connect with and learn from accomplished faculty and researchers at the University of Texas at Austin.

I would also like to acknowledge the Society of Petrophysicists and Well Log Analysts and their special interest group for Petrophysical Data Driven Analytics. Without access to a public dataset and competition summary, none of this research would have been possible. Competitions like these provide both an opportunity for students to practice their skills, but also for researchers to build upon the code and ideas generated through a challenge like the one structured here.

VI. Literature Cited

Akinnikawe, O., Lyne, S., & Roberts, J. (2018, July). Synthetic well log generation using machine learning techniques. In *SPE/AAPG/SEG Unconventional Resources Technology Conference* (p. D023S029R005). URTEC.

Asquith, G. B., & Krygowski, D. (2004). *Basic well log analysis*. AAPG.

Biau, G., Scornet, E. A random forest guided tour. *TEST* **25**, 197–227 (2016). https://doi.org/10.1007/s11749-016-0481-7

Bisong, E. (2019). *Building Machine Learning and deep learning models on Google Cloud Platform: A comprehensive guide for beginners*. Apress.

Choubey, S., Karmakar, G.P. Artificial intelligence techniques and their application in oil and gas industry. *Artif Intell Rev* **54**, 3665–3683 (2021). https://doi.org/10.1007/s10462-020-09935-1

El Naqa, I., Murphy, M.J. (2015). What Is Machine Learning?. In: El Naqa, I., Li, R., Murphy, M. (eds) Machine Learning in Radiation Oncology. Springer, Cham. https://doi.org/10.1007/978-3-319-18305-3_1

Grinsztajn, L., Oyallon, E., & Varoquaux, G. (2022). Why do tree-based models still outperform deep learning on typical tabular data?. *Advances in neural information processing systems*, *35*, 507-520.

Janiesch, C., Zschech, P. & Heinrich, K. Machine learning and deep learning. *Electron Markets* **31**, 685–695 (2021). https://doi.org/10.1007/s12525-021-00475-2

Joshi, D., Patidar, A. K., Mishra, A., Mishra, A., Agarwal, S., Pandey, A., ... & Choudhury, T. (2021). Prediction of sonic log and correlation of lithology by comparing geophysical well log data using machine learning principles. *GeoJournal*, 1-22.

Kramer, O. (2016). Scikit-Learn. In: Machine Learning for Evolution Strategies. Studies in Big Data, vol 20. Springer, Cham. https://doi.org/10.1007/978-3-319-33383-0_5

McKinney, W. (2011). pandas: a foundational Python library for data analysis and statistics. *Python for high performance and scientific computing*, *14*(9), 1-9.

Mishra, A., Sharma, A., & Patidar, A. K. (2022). Evaluation and development of a predictive model for geophysical well log data analysis and reservoir characterization: machine learning applications to lithology prediction. *Natural Resources Research*, *31*(6), 3195-3222.

Mondol, N.H. (2015). Well Logging: Principles, Applications and Uncertainties. In: Bjørlykke, K. (eds) Petroleum Geoscience. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-34132-8_16

Montgomery, D. C., Peck, E. A., Vining, G. G. (2021). Introduction to Linear Regression Analysis. United States: Wiley.

Popescu, M. C., Balas, V. E., Perescu-Popescu, L., & Mastorakis, N. (2009). Multilayer perceptron and neural networks. *WSEAS Transactions on Circuits and Systems*, *8*(7), 579-588.

Rana, P. S., Modi, S. K., Yadav, A. L., & Singla, S. (2023, December). Comparative Analysis of Tree-Based Models and Deep Learning Architectures for Tabular Data: Performance Disparities and Underlying Factors. In *2023 International Conference on Advanced Computing & Communication Technologies (ICACCTech)* (pp. 224-231). IEEE.

Shwartz-Ziv, R., & Armon, A. (2022). Tabular data: Deep learning is not all you need. *Information Fusion*, *81*, 84-90.

Yim, A., Chung, C., Yu, A. (2018). Matplotlib for Python Developers: Effective Techniques for Data Visualization with Python, 2nd Edition. United Kingdom: Packt Publishing.

Yu, Y., Xu, C., Misra, S., Li, W., Ashby, M., Pan, W., ... & Izadi, H. (2021). Synthetic sonic log generation with machine learning: A contest summary from five methods. *Petrophysics*, *62*(04), 393-406.

Appendix

Computer Code

https://github.com/zoeneale/SDS379R-SP2024/tree/35b87bffab9b66d6750b552c52ea0965b9caa54d

Reflection

In conducting this project, I learned both hard and soft skills. The soft skills that I gained are project management and time management. This project was the first time I have done a semester-long research project and I initially found it difficult to establish a schedule for project work. However, once I gained a full understanding of the research being done, it became easier to allocate weekly tasks for myself, with assistance from my supervisor, Dr. Fomel.

The hard skills that I learned from this project are debugging and a stronger ability to read and interpret code from other students. This is a skill I will be able to carry with me throughout my career and was very helpful. I also learned a considerable amount about geology and rock formations. I learned more about machine learning algorithms that I had not used before, such as XGBoost and MLPRegressor.

The project was a collaboration between myself and my project supervisor, Sergey Fomel. The initial code, '1-Regression.ipynb', was provided by Fomel and I began to adapt it. The predictor for DTC was established and then I reconfirmed this using computational methods. I also created an adapted Jupyter notebook for predicting DTS. I would have liked to further research the tree-based models in a separate Jupyter notebook, but given the length of the semester and deadline for the project, this was not feasible. I also conducted all research on previous studies individually, with guidance from Fomel.