PDDA Machine Learning Competition

Dr Iraklis Giannakis, Lecturer at University of Aberdeen

iraklis.giannakis@abdn.ac.uk | www.gprmax.com | Iraklis Giannakis

Introduction

The current report presents the results of our team (VELF) for the PDDA Machine Learning (ML) competition 2021. The objective of the competition is to infer the shale volume, porosity and fluid saturation, based on a set of well logs. In contrast to previous competitions, the current one differs on the fact that the dataset is incomplete with lots of missing values (see Figure 1), and instead of classification the current problem is regression. Moreover, three different outputs, correlated to each other, needed to be estimated, and major imbalances occur both within the same borehole and between boreholes. Our team has developed a novel data-driven scheme (see Figure 2) based on supervised and un-supervised machine learning in order to impute, augment, balance and filter the training data, and subsequently use them to unravel the causal relationship between well log measurements and reservoir properties.

Methodology?

Data Selection

The first thing that needs to be addressed is to choose the well log data that will be used in the regression scheme. Figure 1 outlines the missing data for both training and testing sets. It is apparent that they dataset is not complete, with many missing values. In the current scheme we decided to drop *DTC, DTS, BS* and *ROP*. The aforementioned variables have lots of missing values both in the training and testing set, and trying to impute them will compromise the overall accuracy of the scheme. Moreover, we droped all the rows with missing outputs i.e. *PHIF, SW, VSH*. Data with missing outputs are difficult to treat and it is debatable under which circumstances is beneficial to include them in the training set. From the rest of the data, we dropped any row that contains even one missing value. Doing this results to a big reduction of the dataset, nonetheless the remain data are full and coherent; and the testing data have now a small number of missing values that can be mitigated using imputation methods.

Data Pre-Processing

A simple pre-processing is applied to the remaining data. In particular *RDEP* and *RMED* are replaced by their logarithm; and *Isolation Forest* is used to all the data to identify outliers and subsequently remove them from the set.

Augmentation

In the next step we augment the data by taking the first and second derivative of all the well logs. The derivatives are calculated numerically via $df \propto f(i+1) - f(i) \forall i \in \mathbb{R}^n$, where n is the number of elements in f. Similarly for the second derivative $d^2f \propto df(i+1) - df(i)$. Then we use a KNN unsupervised learning to divide the resulting data to 25 different clusters. The number of the cluster for each point is included in the training set. This will assist supervised learning on identifying different types of data and train them accordingly.

Treating Imbalances Using SMOTE

Training the model (regardless of the methodology, pre-processing etc.) results to over-fitting when tested to unknown data from the boreholes that were included in the training. This means that generalising between different boreholes is challenging and special care should be taken such ML to be able to perform equally well to all boreholes. To do that, we need to make sure that each borehole is equally represented in the training data. To that extent, we used SMOTE which interpolates within the data of the underrepresented boreholes in order to increase their points and consequently their influence to the final model.

Scaling

A robust scaler is applied to all the data, both inputs, outputs and the augmented columns (derivatives and cluster number).

Data Imputation in Testing Set

Although we have dropped the columns with the most missing data (*DTC, DTS, BS* and *ROP*), the testing set still contains missing values that need to be imputed. In the current scheme we use a combination of KNN Inputer and Iterative Imputer that models each

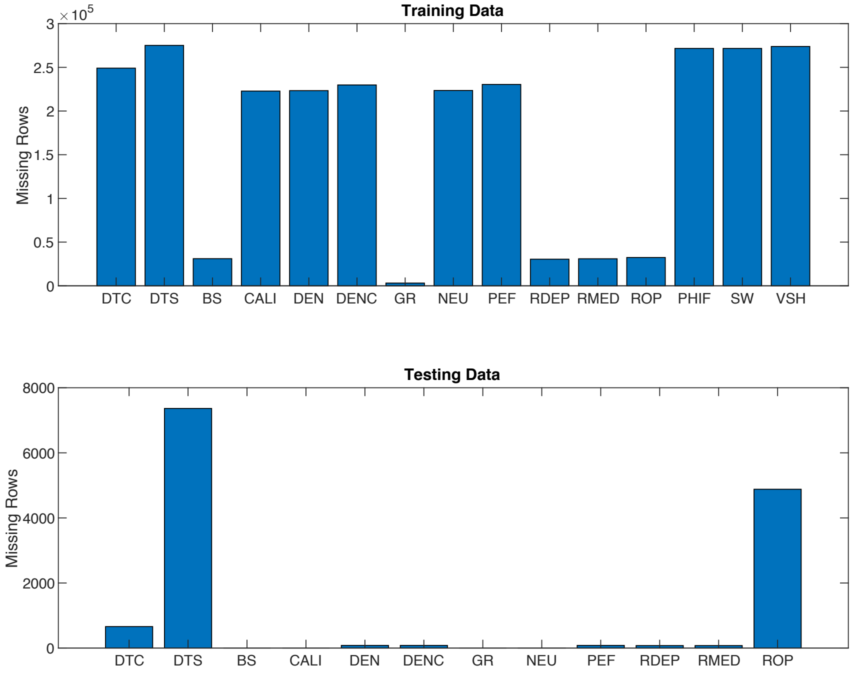

Figure 1: An outline of the missing values in training and testing data.

feature with missing values as a function of other features in a round-robin fashion using Bayesian ridge regression. The results from KNN imputer and Bayesian ridge are averaged for the final output.

Supervised Learning

The final training data are more than 100k rows with 25 input columns and three outputs. The supervised framework used here is a chain regression boosted trees with maximum depth equals 3 and 120 estimators.

Thresholding

Based on the training data, we don't expect any negative values on the outputs. Moreover, SW has a clear upper limit at 1. Consequently, the predictions are thresholded based on these bounds, i.e. if any value is negative becomes 0, and if SW is greater than 1 it equals with 1.

Results

To illustrate the effectiveness of the proposed pipeline, we applied it to unknown data from the training set. In every example a different borehole is used as a testing set to validate the generalization capabilities of the suggested scheme on inferring the reservoir properties to unknown boreholes. In all cases the overall MSE was consistently below 0.045, and for most of the wells it was bellow 0.04. The scheme shown great generalization capabilities without unnecessary complex ML schemes. SMOTE and augmenting imbalanced data greatly increased the performance of the system. More over using the first and second derivative also increased the available information and enhance ML's performance. Figure 3 illustrate the results for well 1 (MSE 0.035). It is indicative that the propose algorithm performs very well for all the investigated outputs (PHIF, SW, VSH).

Description of the submission

The folder contains one notebook (VE4F_Complete) with both training and testing scripts. The training script is under the name *Training_Script.py*, and the results can be reproduced using *Results.py*. The objects necessary for *Results.py* to run are also included in the folder, and can be reproduced running the training script (*Training_Script.py*). Both of the scripts need the test.csv and train.csv to run, that are not included in the folder due to space limitations.

Training

Dropping DTC, DTS, BS and ROP

Removing rows with NaN values

Log10(RDEP) and Log10(RMED)

Isolation Forest to remove outliers

Augment the data: first and second derivative

SMOTE between Boreholes

Robust Scaling

Gradient Boosted Trees

Inverse scaling Transform

Thresholding

Figure 2: A pseudo-flowchart of the proposed training pipeline.

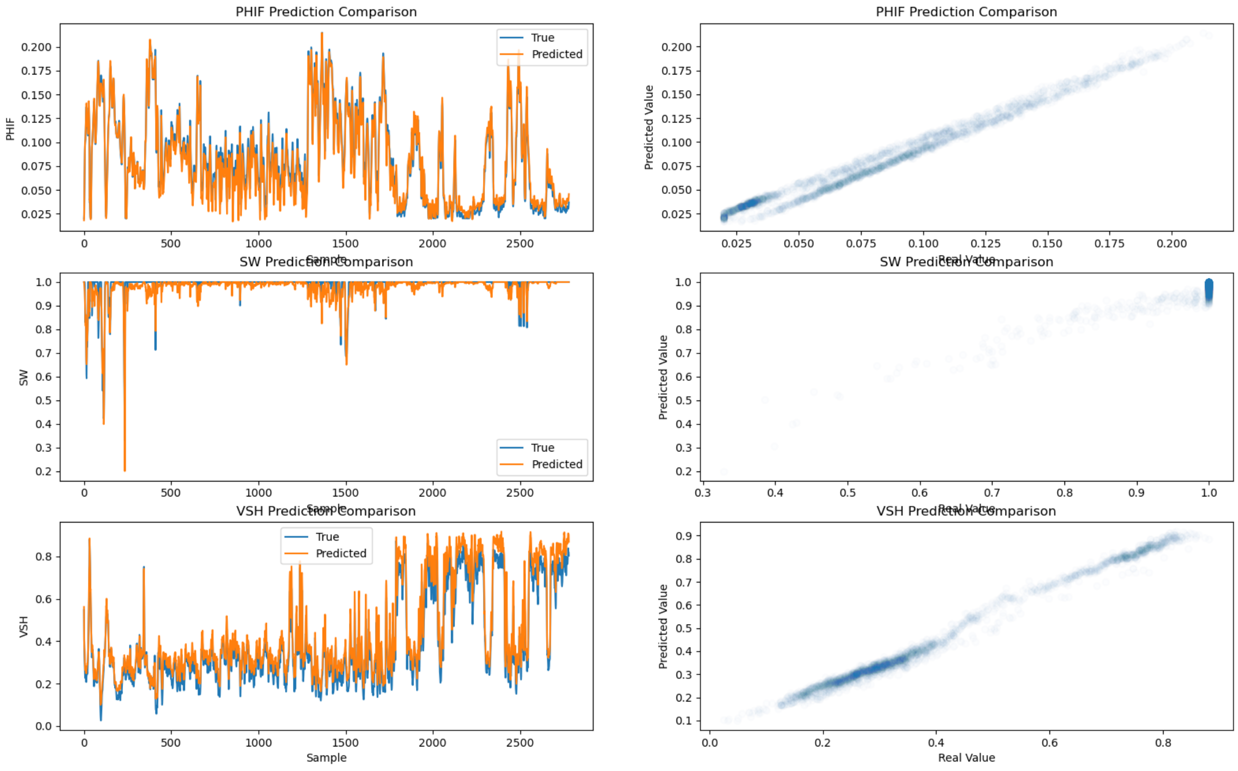

Figure 3: The results from well-1. During the training process these data were not included in the training set. Overall MSE ≈ 0.035