Formation Lithology Classification: Evaluation of Machine Learning Methods

PEGN 499: Introduction to Data Science

Mohamed Ibrahim Mohamed

PhD Candidate Petroleum Engineering Department

November 21, 2018

1. Summary

Facies classification using well logs using machine learning has been a subject of intense speculation for the past few years. The idea of using automated algorithms to determine the facies is not new, however further research should be done to take the supervised classification task and the methodological limits of such problem. This project demonstrates training a machine learning algorithm to classify and predict the geological faces using well logs data. The data set used in this project was obtained from a class exercise from The University of Kansas.

2. Introduction

Facies are used by petroleum engineers and geologists to group together rocks with similar characteristics. This characterization helps identifying formations that yield hydrocarbons and water. This classification is rather subjective as it depends on the attributes we choose used for the classification. Biological properties for instance can be used to classify the rock. Rock granulometry and mineralogy are important characteristics that can be used for the classification. For oil and gas exploration and production, porosity and permeability are two critical properties to determine as they reflect the potential volume of hydrocarbons and how easy the fluid will move during production. Well logs are the main source of information for defining the facies and their properties. By measuring the electrical responses as well as the nuclear radiations of the drilled section, we can infer properties about its rock matrix and fluid content.

Classifying field data into groups is one of the main branch of the popular field of machine learning. Among the vas panel of methods, this project is mainly focusing on the supervised learning. Learning from already labeled data, those algorithms are able to discover abstract representations and to understand the data and predict.

3. Dataset Description

The data set used in this project was obtained from a class exercise from The University of Kansas. The data set is a well log for nine wells for a large gas field in the north America, the Hugoton and Panoma Fields. More information about the data can be found in Bohling and Dubois (2003) and Dubois et al. (2007). The well logs have been labeled with the corresponding labels based on core observations. The data set was first loaded to the flowers database using SQL commands as follows:

1. Creating the table:

   ```
   CREATE TABLE welllog (Facies numeric, Formation varchar(7), Well_Name text, Depth numeric,
   GR numeric, ILD_log10 numeric,DeltaPHI numeric, PHIND numeric, PE numeric, NM_M numeric,
   RELPOS numeric)
   ```

2. Copy dataset to table:

   ```
   Copy welllog(Facies,Formation,Well_Name,Depth,GR,ILD_log10,DeltaPHI,PHIND,PE,NM_M, RELPOS)
   from ’C:\ Users\ mohamed1 \ Desktop \ Training_Dataa.csv’delimiter’,’CSV;
   ```

3. Accessing the SQL database from jupyterhub:

   ```
   dbinfo = ’host’:’flowers.mines.edu’,’user’ :  ’mohamed1’,’password’:  ’****’, ’database’
   :’csci403’
   Try:  db = psycopg2.connect(**dbinfo)
   cursor = db.cursor()
   except psycopg2.Error as e:
   print(e)
   ```

4. The data set then were loaded to a dataframe using panda library as shown in the following line:

   ```
   data1=pd.read_sql_query("select*from welllog order by well_name, Depth", db)
   ```

First, we explored the data set by plotting the data from one well and by creating cross plots to look at the variation with the data. Data visualization plays an important role in the data exploration part. Thus, I focused on enhancing the visualization of the well logs and creating an interactive plots. Bokeh is a visualization library that provide interactive tools as pan, drag, click, scroll, pinch and hover tool. The listed tool make exploring the well logs much easier compared to the conventional excel plots and the matplotlib library. As shown in Figure 1, the seven logs versus depth were plotted using Bokeh library. The data consists of five wireline measurements and two indicator logs and finally the facies label track.

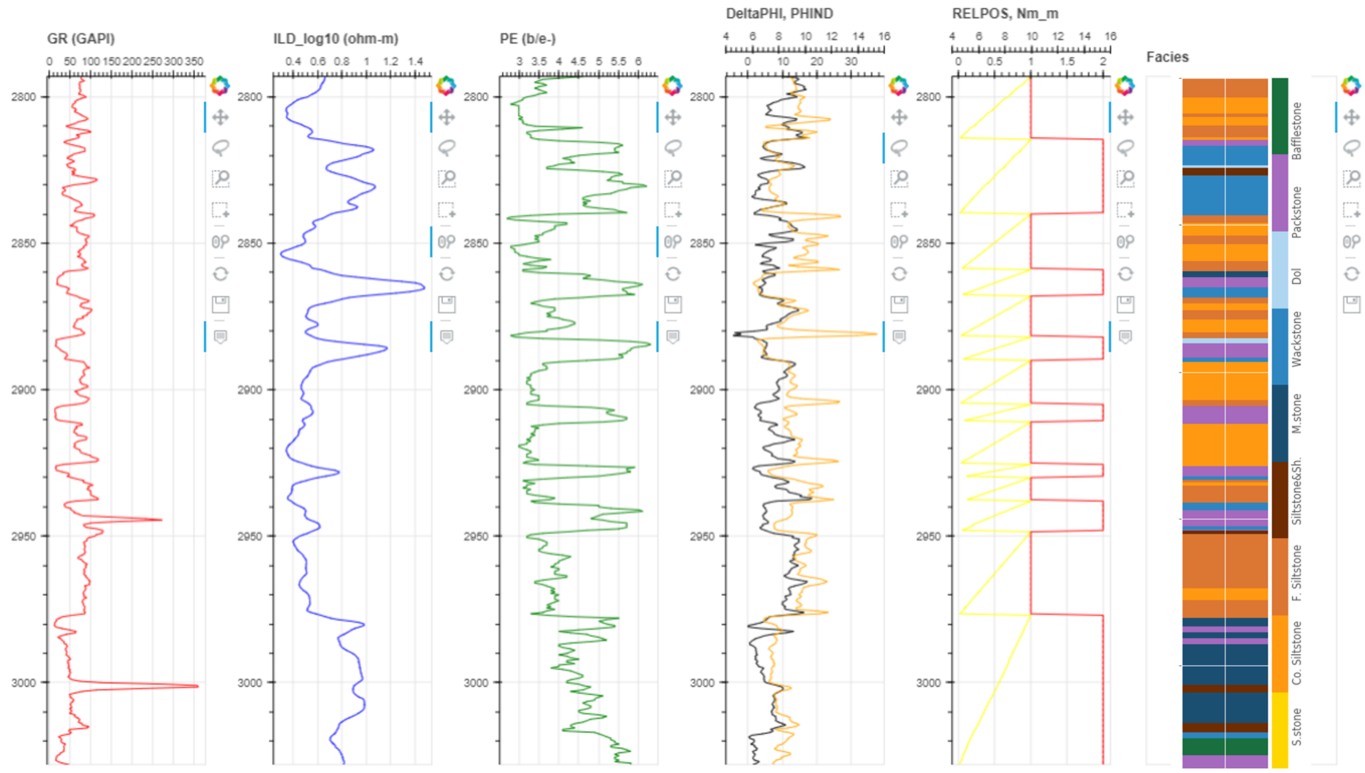


Figure 1: Well logs visulaization for one of the wells using Bokeh

Figure 2b shows the box selection and zoom tool. This tool can provide insights at specific depth intervals for better interpretation. Figure 2c shows the wheel zoom option. Hover tool is an important tool that provide a tabular where each row contains a label and its associated value. For instance, the tooltip shown in Figure 2 was created to show the depth and the resistivity log values.

Each log measurement is a feature vector that is used as feature to classify the facies type they are also named as predictor variables, while the rock facies are the labels. Nonetheless two out of the seven logs are geologic constrain variables that can be used in the algorithm.

The seven predictor variable are:

1. Five wire line log curves include gamma ray (GR), resistivity logging (ILD_log10), photoelectric effect (PE), neutron-density porosity difference and average neutron-density porosity (DeltaPHI and PHIND).

2. Two geologic constraining variables: nonmarine-marine indicator (NM_M) and relative position (RELPOS)

The statistical distribution of the input variables is shown in Figure 3. In this data sets it seems that there is no anomaly or out layers. In figure4 we show the distribution of training data by facies. This is used to count the number of times each facies appear in the dataset. It can be observed that dolomite yields the lowest occurrence and coarse siltstone has the most occurrence.

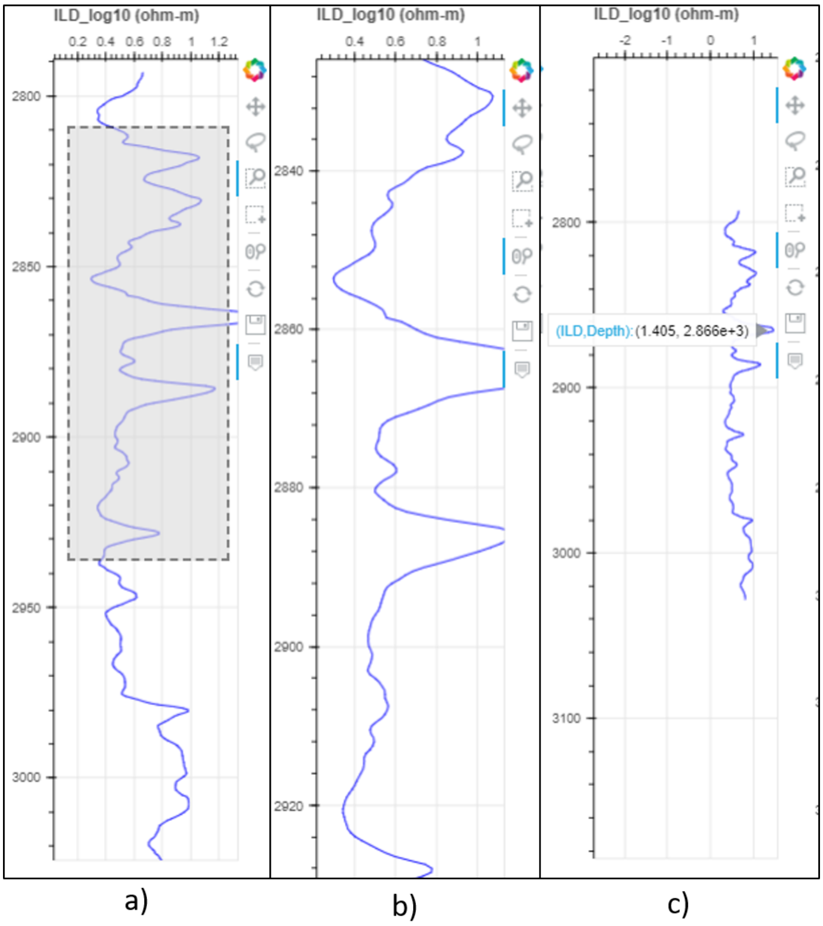
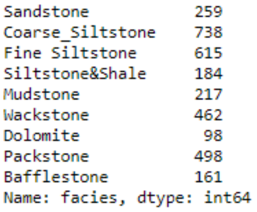


Figure 2: Bokeh BoxZoom Features and Scroll Zoom



```
Sandstone           259
Coarse_Siltstone    738
Fine Siltstone      615
Siltstone&Shale     184
Mudstone            217
Wackstone           462
Dolomite             98
Packstone           498
Bafflestone         161
Name: facies, dtype: int64
```

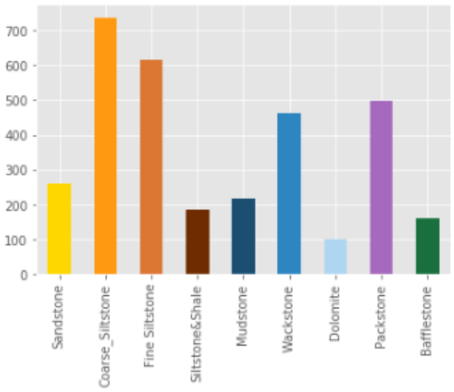


Figure 3: Statistical distribution of input variables

Scatter matrix cross plot was drawn to visualize the change of the features with the facies as shown in Figure 4. Each pane in the plot shows the relation between two features and the color change corresponds to each facies.

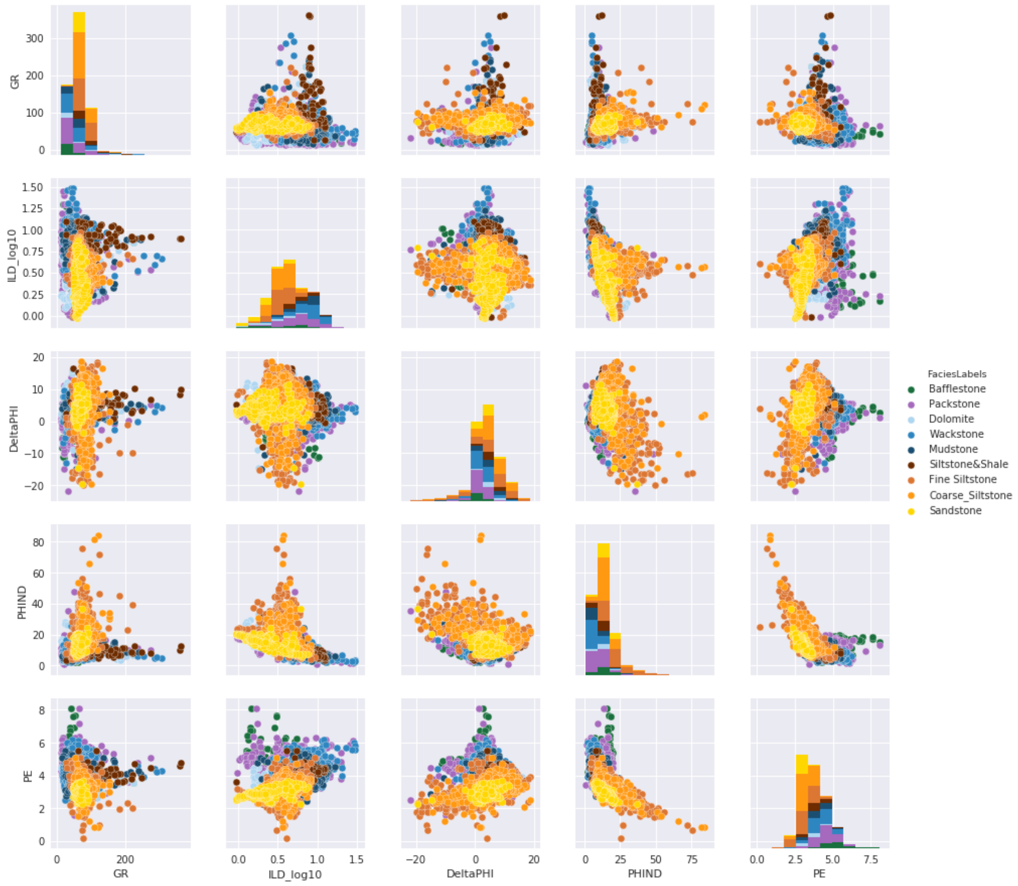


Figure 4: Scatter matrix cross plot

There are a total of nine discrete rock facies, in this project we color mapped each layer for better visualization of the data as shown in track 6 Figure 1.

- Sandstone
- Coarse siltstone
- Fine siltstone
- Siltstone and shale
- Mudstone
- Wakestone
- Dolomite
- Packstone
- Bafflestone

There are total of eight wells, where we used seven wells as training wells and one well will be used as test or blind well for evaluating the model. To standardize the data the `sklearn.preprocessing` where used to transform the raw features vectors into a representation that is more suitable. This grants that the features are standard normally distributed; Gaussian with zero mean and unit variance. We also used `train_test_split` to split training data into training and test data. The test set is used to evaluate the algorithm and are not used for training the network.

4. Methdology

After cleaning and processing the data set, the next step is to create the facies classifier. In this project several Machine Learning Algorithms such as Random Forest, Support Vector Machines, and K-Nearest Neighbors, were used to create facies classifier all of these are classified as supervised learning methods for classification.. Each model was evaluated using the test data. Model parameter selection study was done for both SVM and KNN methods.

Training using the KNN Classifier

KNN classifier is a neighbors-based classification. The classification is computed from a simple majority vote of the nearest neighbors of each point. The `KNeighborsClassifier` used in this project implements learning based on the k nearest neighbors of each point. Where k is an integer value specified before. In this project we tested the model for different k values to obtain the optimal f1-score. A nested loop were done to test all k values and the f scores for Training and the Validation data were plotted as shown in Figure 5. A k values of 1 was choosen.

The model was then evaluated using the testing data. We used the confusion matrix to obtain the f1-score, precision and recal. Using the k equals to 1 the f1-score was 69%. Also, the accuracy of the facies classification which is the number of correct classifications divided by the numer of classifications, was calculated and was found to be 69%. Noting that there is no definite boundaries between the layers it's possible that the adjacent layers, thus it's important to calculate the accuracy within the adjacent layers. In this algorithm the accuracy was found to be 85%.

The model was finally evalued with the blind test data. The f1-score obtained was 46%, the new Facies accuracy is 48% and the adjacent facies accuracy is 89%.

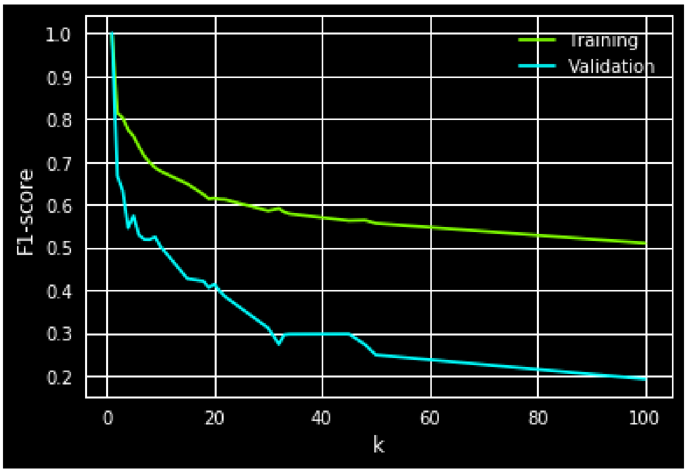


Figure 5: F1-Scores for Different K values

Training using SVM Classifier

SVM is capable of performing multi-class classification on a dataset. This requires two parameters namely; Gamma and C. The parameter C is a regularization factor, and tells the classifier how much we want to avoid misclassifying training examples. A large value of C will try to correctly classify more examples from the training set, but if C is too large it may 'overfit' the data and fail to generalize when classifying new data. If C is too small then the model will not be good at fitting outliers and will have a large error on the training set.

The gamma parameter describes the size of the radial basis functions, which is how far away two vectors in the feature space need to be to be considered close.

SVM classifier take as input two arrays: an array X of size `[n_samples, n_features]` holding the training samples, and an array y of class labels. After fitting the model, it can then be used to predict the new values.

The model was then evaluated using the testing data. We used the confusion matrix to obtain the f1-score, precision and recal. Using the default parameters the f1-score was 44%. Also, the accuracy of the facies classification which is the number of correct classifications divided by the numer of classifications, was calculated and was found to be 49%. Noting that there is no definite boundaries between the layers it's possible that the adjacent layers, thus it's important to calculate the accuracy within the adjacent layers. In this algorithm the acuuracy was found to be 85%.

Next we focused on the model parameter to obtain the optimum accuracy and f1 score. In this part we train the model with different values for the gamma and the C. The results from the loop are shown in Figure 6. We picked the parameters with the highest CV error. In this project the optimal C and gamma were found to be 50 and 1 respectively. The model was evaluted with the new parameters and f1-score of 71% was obtained. The new Facies accuracy is 71% and the adjacent facies accuracy is 89%. The model was finally evalued with the blind test data. The f1-score obtained was 45%, the new Facies accuracy is 48% and the adjacent facies accuracy is 89%.

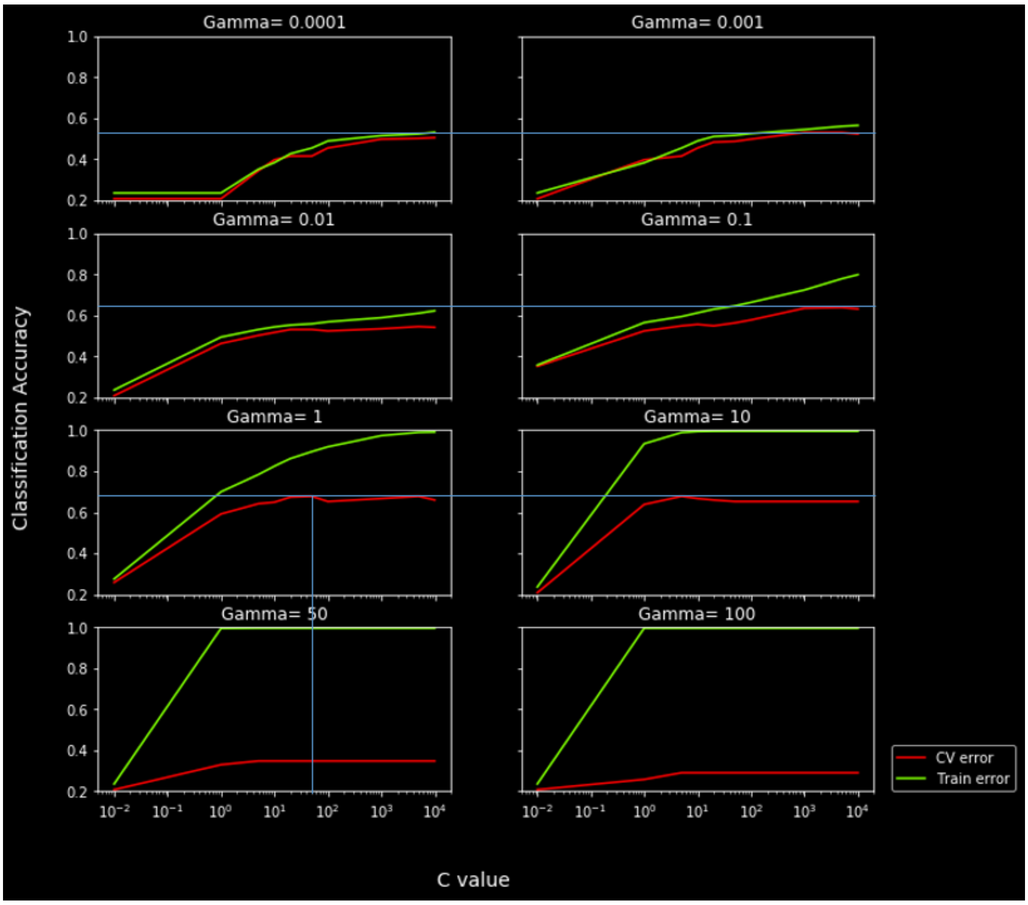


Figure 6: SVM Model Parameter Selection

Training using the RandomForrest classifier

Random forest classifier is a meta estimator that fits numer of decision tree classifiers on various sub-samples of dataset and uses averaging to imporve the predicitve accuracy and control over-fitting. Similar to the previous two classifier, `RandomForest` takes an input of `n_estimators` which represent the number of trees in the forest. The optimal n_estimators was found to be 8 where the F1-score is 68%. The new Facies accuracy is 64% and the adjacent facies accuracy is 85%. The model was finally evalued with the blind test data. The f1-score obtained was 42%, the new Facies accuracy is 42% and the adjacent facies accuracy is 86%.

4. Conclusion and Results

The preforamance of the three classifiers were compared as shown in Table 1. It can be observed that SVM yeild the best performance and the highest F1-Scores at Gamma=1 and C=50. The model was then used to predict the facies for the blind well. The results of the Facies are shown in Figure 7. The Predicted facies are shown in track 9 and the actual facies from the cores are shown in track 8.

Table 1: Summary of the Classifiers Preformance

Classifier		KNN	SVM	RandomForrest
	Optimal Parameter	K=1	Gamma=1, C=50	n_estimators=8
CV_Test Well	F1-Score	69%	71%	68%
	Facies Accuracy	69%	71%	64%
	Adj. Facies Accuracy	85%	89%	85%
Blind Well	F1-Score	46%	46$	42%
	Facies Accuracy	48%	48%	42%
	Adj. Facies Accuracy	89%	89%	86%

References

Cat, Antoine, et al. ”Machine learning as a tool for geologists.” The Leading Edge 36.3 (2017): 215-219.

Niccoli, Matteo. ”Machine Learning in Geoscience V: Introduction to Classification with SVMs.”

Seth Willis Bassett, May 23, 2017 Teaching a Computer Geology: Automated Lithostratigraphic Classification Using Machine Learning Algorithms DIGITAL MAPPING TECHNIQUES 2017

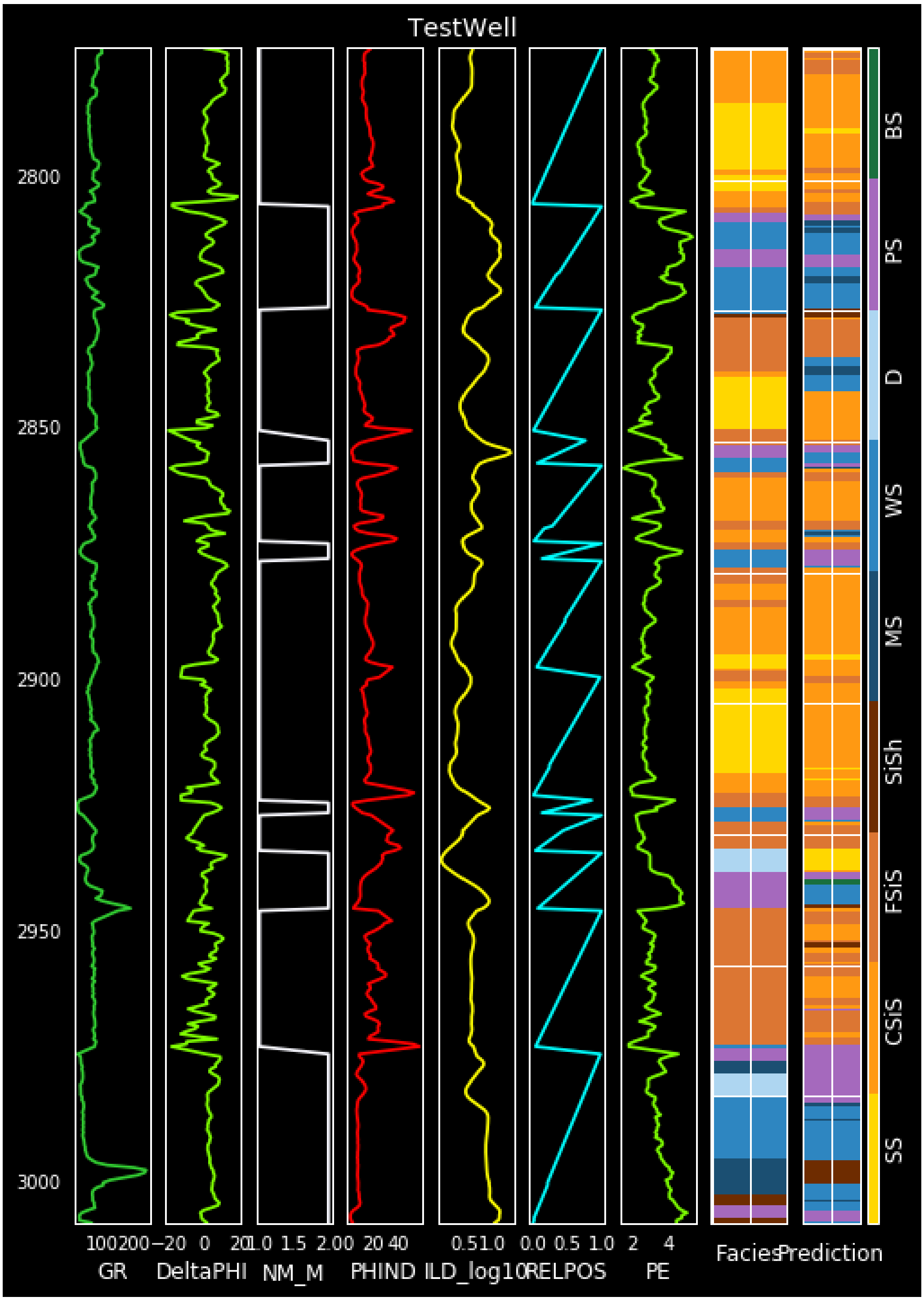


Figure 7: Predicted Facies (Track9) and Actual Facies (Track8) for the Blind Test Well