*Final Project – Artificial Intelligence in Oil and Gas Industry (13012)*

# Machine Learning Approach to Facies Classification and Clustering using UP Field's Well Log Data

**Muhammad Abdul Aziz [1], Muhammad Hisyam Ilyasa [2]**

[1] Petroleum Enginnering Student, Pertamina University; mabdulaziz703@gmail.com
[2] Petroleum Enginnering Student, Pertamina University; hisyamilyasa35@gmail.com

**Abstract:** This paper discusses the comparison of several commonly used machine learning algorithms, namely K-Nearest Neighbor (K-NN), Support Vector Machine (SVM), and Random Forest Classifier in the context of facies classification, and K-Means Clustering in the context of facies clustering. The main objectives of this paper are to determine the appropriate input features to be used in facies classification prediction, to determine the best machine learning classification algorithm in facies classification prediction, to perform facies clustering with K-Means Clustering and compare the data distribution statistics per facies class, and to evaluate the prediction model that has been created. This study uses well log data from the UP Field that includes attributes relevant to facies classification. Each algorithm is then applied to the dataset and the results are evaluated using confusion metrics such as accuracy, precision, recall, R-2 score and F1-score. The results of the machine learning approach using the KNN model can provide a higher level of accuracy with an R-2 score of 0.90 and F-1 of 0.95.

**Keywords:** machine learning; facies classification; k-nearest neighbor

## 1. Introduction

Facies or lithofacies classification consists of determining the type of rock at a particular depth by interpreting a series of measurements (well logs). In the oil and gas industry, facies classification helps geoscientists interpret the complex subsurface geology and make informed decisions regarding drilling locations, reservoir modeling, and production strategies. By identifying and classifying different sedimentary facies within a reservoir, geoscientists can determine the rock's porosity, permeability, and fluid saturation, which are critical parameters for estimating reservoir productivity and optimizing recovery.

The traditional approach to facies classification involves manual assignment of lithofacies by human interpreters, which is a laborious and time-consuming process. As a result, various alternative methods for classifying facies using well data have been suggested. [1] Introduced a Geophysical Tutorial that demonstrated the application of machine learning techniques for facies classification, offering a straightforward example in this case. Specifically, [1] utilized a limited dataset consisting of seven wireline logs and their corresponding interpreted facies from ten wells located in the Hugoton gas field in southwest Kansas [2]. The objective was to use this dataset to predict the geological facies in two additional wells based on wireline measurements.

Recently, the geoscientists have been utilizing a combination of machine learning algorithms and various processing and interpretation techniques. The most popular technique for facies classification is K-Nearest Neighbor (KNN), Support Vector Machine (SVM), Random Forest Classifier (RFC), and the most popular technique for facies clustering is K-Means Clustering. The purpose of KNN algorithm is to classify new objects based on the attributes and training data. To classify objects based on the training data that has the closest distance to a new object is based on the euclidean equation formula [3]. SVM is a binary classification algorithm and implements the idea of mapping nonlinearly vectors to a very high-dimension feature space to construct a linear decision

surface (hyperplane) in this feature space. In hence, it can achieve good effect on solving separable and non-separable problems [4]. RF is an ensemble learning algorithm, which uses decision trees to train samples and aggregate prediction results through voting, thereby improving the model's prediction accuracy. Its implementation is simple, and it performs well on multi-feature data and data partially missing features, making it widely used in machine learning. K-Means clustering is an algorithm required as much input as k which divides n objects into k clusters so that the level of similarity between members in one cluster is high while the level of resemblance to the members in the other cluster is very low [5]. The level of similarity between members in a cluster is measured by the proximity of the object to the mean value on the cluster or usually referred to as the centroid cluster.

This paper focuses on comparing and evaluating the performance of various machine learning algorithms in the classification and facies clustering of UP Field well-log data. Some of the machine learning algorithms used for classification are K-Nearest Neighbor (KNN), Support Vector Machine (SVM), and Random Forest Classifier (RFC). The machine learning algorithm used for clustering is K-Means Clustering. By systematically analyzing various algorithms and considering the strengths and limitations of each, the main objectives of this paper are to determine the appropriate input features to be used in facies classification prediction, to determine the best machine learning classification algorithm in facies classification prediction, to perform facies clustering with K-Means Clustering and compare the data distribution statistics per facies class, and to evaluate the prediction model that has been created. Finally, the findings of this study have implications for various disciplines, including geology, petroleum exploration, and may pave the way for future advances in automated geological interpretation.

## 2. Materials and Methods

### *2.1. Exploratory Data Analysis*

#### 2.1.1. Data Understanding and Preprocessing

The well log data used came from recorded logging data in the UP Field (wells A, B, and C) provided by the lecturer. To see the complete data, you can go to the link in **Supplementary Materials**. This dataset consists of 15 features which are as follows.

1. DEPTH_MD, is the measured depth (ft)
2. CALI, is the caliper log reading (ft)
3. RSHA, is the shallow resistivity log reading (ohm.m)
4. RMED, is the medium resistivity log reading (ohm.m)
5. RDEP, is a deep resistivity log reading (ohm.m)
6. RHOB, is formation (bulk) density log reading (gr/cc)
7. GR, is the gamma ray log reading (API)
8. NPHI, is the porosity read from the neutron log (fraction)
9. PEF, is the log photoelectric absorption reading (barns/electron)
10. DTC, is the transit-time compressional wave reading from the sonic log (micron-s/ft)
11. SP, is the log spontaneous potential reading (mV)
12. BS, is the bit size (in) used during drilling
13. ROP, is the rate of penetration (ft per minute) read during drilling
14. DRHO, is the density correction curve (gr/cc) that indicates the quality of the bulk density reading data
15. FACIES, is the facies label. There is no detailed information on the lithology and characteristics of each facies class.

In machine learning terminology, the set of measurements at each depth interval consists of feature vectors, each of which is associated with a class (facies type). We used the pandas library to load the data into a data frame, which provides an easy-to-use data structure for working with well log data. Before processing the data, we first merge the

three well data into one with the addition of one feature in the form of "well name" to distinguish each data. Then, we can use the `.describe()` function to provide a quick overview of the statistical distribution of the dataset as can be seen in **Figure 1**. It can be seen from the Count row in Table 1 that this dataset has a total of 29871 values.

Furthermore, there are several things in preprocessing that need to be done, namely removing features that have no value (NaN) and also removing features that are not used in the training dataset. Here is an explanation of the two steps.

1. Removal of features that have no value (NaN)

   Based on **Figure 1**, it can be seen that there are some features that do not have values (marked with a smaller count row than the others). Therefore, the data points that do not have valid values are deleted using the `.notnull().values` function. Some of the features that are removed are 'CALI', 'RMED', 'RHOB', 'NPHI', and 'DTC', where the process can be seen in the python script provided.
2. Removal of unused features

   The determination of necessary or unnecessary data is based on the scientific domain of Well Logging. The features used in the training process are all data, except for 'BS', 'ROP', and 'DRHO' data, because they do not really affect the prediction of facies classification. This is because 'BS' is a tool in well logging, 'ROP' is a penetration rate value, and 'DRHO' is a density correction. These three features only provide an indirect effect as a tool and method in the well logging process, where when they are not good it will affect the value of other features. While the other features provide a direct effect in modeling because they are the main features in the form of reading log data. As for the 'RSHA' and 'SP' data, both are not used in this training process because both have no value (NaN).

The next process is to convert the 'WELL_NAME' feature into categorized data and remove one well from the dataset to be used as a Blind Dataset and the rest as a Training Dataset. In this case, researchers used Well C as the Blind Dataset and Well A and Well B as the Training Dataset. Berikut adalah cara pembuatan Blind Dataset dan Training Dataset dalam python.

```
blind_dataset = dataABC[dataABC['WELL_NAME'] == 'WELL C']
training_dataset = dataABC[dataABC['WELL_NAME'] != 'WELL C']
training_dataset['WELL_NAME'].value_counts()
```

After determining the Blind Dataset and Training Dataset, the log data plot, facies count plot, and pair plot for wireline data reading were conducted. The three plots can be seen in **Figure 2**, **Figure 3**, **Figure 4**, respectively. In addition, the correlation plot heatmap in **Figure 5** can also be seen from the dataset before the dataset training process.

2.1.3. Data Conditioning

In this process, data splitting is done into matrix features (X) and label features (y). This needs to be done before starting the dataset training process. This can be done using the following code.

```
X = training_dataset.drop(['DEPTH_MD', 'FACIES', 'WELL_NAME'], axis=1)
y = training_dataset['FACIES']

from sklearn.model_selection import train_test_split

X_train,X_test,y_train,y_test = train_test_split(X, y, test_size=0.2)
```

After splitting the data and model selection has been made for X_train, X_test, y_train, and y_test, it is necessary to process Feature Scaling with StandardScaler to normalize (make the average 0 and variance 1). When this process has been completed, the

training dataset is ready for the training process using the classification and clustering algorithms that have been determined.

*2.2. Machine Learning Methods*

2.1.1. Machine Learning Classification Algorithm

1. K Nearest Neighbor (KNN)

KNN was named as one of the ten most popular and most important algorithms. KNN is known to be very simple and easy. KNN is an example-based learning group. This algorithm is also one of the lazy learning techniques. KNN is done by searching for the group of K objects in the closest training data (similar) to objects in new data or data testing [6]. The KNN method is not affected by outliers and is suitable for classification problems with a large number of overlapping sample sets or cross-class domains [7]. KNN is formulated in **Equation 1**.

2. Support Vector Machine (SVM)

SVM is a statistical method that can be applied to linear and nonlinear regression problems. It enhances the generalization ability of the learning machine by minimizing the empirical risk and confidence range while seeking the minimum structured risk [7]. SVM extends the two-dimensional linear separable problem to multidimensional, and aims to seed the optimal classification surface, also called as optimal hyperplane. The optimal hyperplane can be defined as **Equation 2** and the relation between $\boldsymbol{x_i}$ and $\boldsymbol{f(x_i)}$ can be defined as **Equation 3**. Seeding the optimal hyperplane is equivalent to maximal the distance between the closest vectors to the hyperplane [4].

3. Random Forest Classifier (RFC)

The RFC algorithm is a combination classifier algorithm that uses a CART decision tree as the base classifier. It constructs a decision tree from the random sample sets with and without placement. The final multiple decision trees form an RFC model whose final prediction result is determined by the base classifier's vote [7]. RFC uses Gini Index [8] method for its attributes' selection which measures the impurity of an attribute with respect to its classes. For a given training set $P$, selecting a sample case randomly and to predict its class as $C_i$, the Gini index can be written as **Equation 4**.

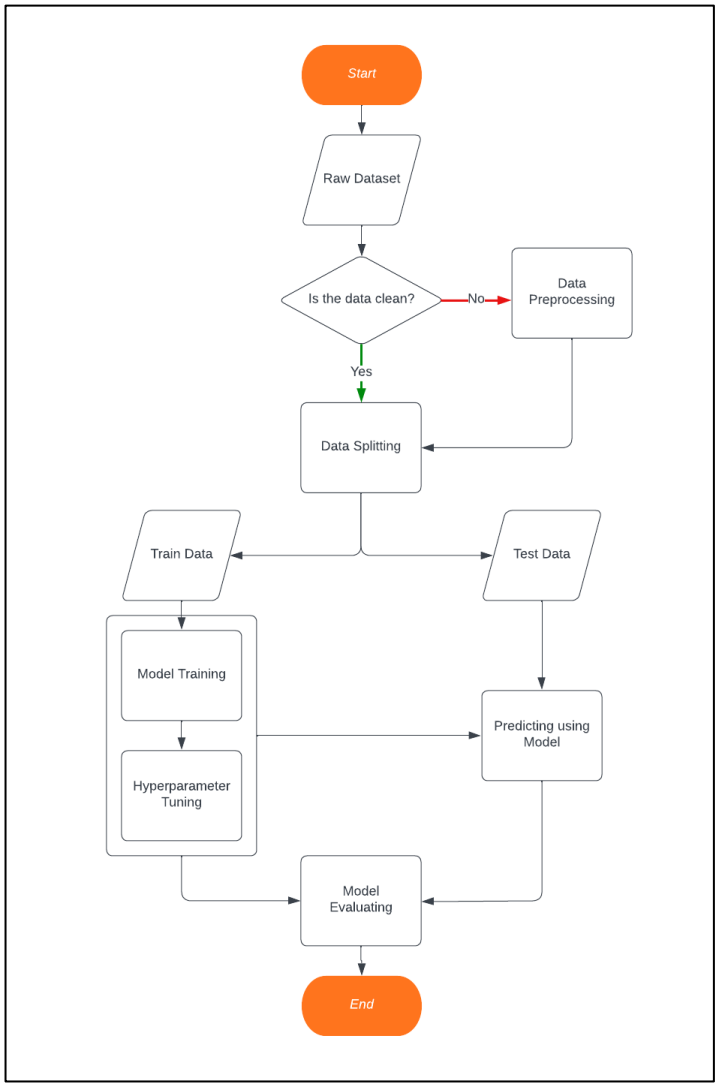
For generating a prediction model, the RFC needs the definition and insertion of two parameters, the number of classification trees desired and a number of predicting variables that are used in each node to grow the trees. For each node, the best split is done by searching selected features. Thus, RFC consists of $N$ decision trees where $N$ is user defined value about the number of trees to be grown. When new data points are to be classified, these are passed down to all those trees and then it chooses its class by maximum votes out of $N$ votes.

2.1.2. Machine Learning Clustering Algorithm (K-Means Clustering)

The K-means algorithm is a popular clustering method due to its high computational efficiency. It is commonly used for large-scale data clustering. The algorithm works by setting k as a parameter, determining the number of categories k that the data set containing n objects needs to be divided into [7]. Given a dataset of $\boldsymbol{n}$ data points $\boldsymbol{x_1}$, $\boldsymbol{x_2}$, …, $\boldsymbol{x_n}$ such that each data point is in $\boldsymbol{R^d}$, the problem of finding the minimum variance clustering of the dataset into $\boldsymbol{k}$ clusters is that of finding k points $\{\boldsymbol{m_j}\}$ ($\boldsymbol{j}$=1, 2, …, $\boldsymbol{k}$) in $\boldsymbol{R^d}$ such that it can be seen as an **Equation 5.**

The problem in **Equation 5** is to find $\boldsymbol{k}$ cluster centroids, such that the average squared Euclidean distance (mean squared error, MSE) between a data point and its nearest cluster centroid is minimized.

### *2.3. Workflow*




The following is an explanation for each stage of the workflow above:

1. Start
2. Collecting raw well log data from UP Field that will be used in facies classification, such as caliper log, resistivity log, formation bulk log, gamma ray log, neutron log, photoelectric log, sonic log, spontaneous log, and other data.
3. Cleaning data with data preprocessing to address missing data, invalid data, or outliers.
4. Splitting data into training data and testing data, where testing data uses data from one of the wells in the UP Field.
5. Training the data using several models, such as KNN, SVM, RFC, and K-Means.
6. Identifying the hyperparameter of each model in order to produce an optimal model.
7. Predicting facies of testing data using the trained model.
8. Calculating the evaluation metric of each model.
9. End

## 3. Results

### *3.1. Training Dataset & Blind Dataset*

In accordance with the previous process, the dataset used for training is a combination of well A and well B data with the aim that the data is more numerous and diverse for the training process so that the resulting model becomes better. While the dataset used as a blind dataset is well C data, where this data is not included in the training data. The features that are the focus of the training dataset and blind dataset are 'DEPT_MD', 'CALI', 'RMED', 'RDEP', 'RHOB', 'GR', 'NPHI', 'PEF', 'DTC', and 'FACIES'. Therefore, in using this model, preprocessing needs to be done first according to the data needs in running this model.

### *3.2. Model Training Result*

The results given from this training model are in the form of Confusion Matrix Display, Classification Report, and Comparison of R2-Score & F1-Score Values from each classification model.

#### 3.2.1. KNN Training Result

The Confusion Matrix Display and Classification Report training results using the KNN classification model can be seen in **Figure 6** and **Table 1**, respectively.

#### 3.2.2. SVM Training Result

The Confusion Matrix Display and Classification Report training results using the SVM classification model can be seen in **Figure 7** and **Table 2**, respectively.

#### 3.2.3. RFC Training Result

The Confusion Matrix Display and Classification Report training results using the SVM classification model can be seen in **Figure 8** and **Table 3**, respectively.

#### 3.2.4. Comparison of R2-Score & F1-Score Values (Training Dataset)

The following are the results of the comparison of the R2-Score and F1-Score values of each classification model which can be seen in **Table 4**. Based on these comparison values it can be seen that the KNN classification model has a higher value than other classification models, both from R2-Score and F1-Score so that this can be the best model in this facies classification..

### *3.3. Model Prediction Result*

The results given from this prediction model are in the form of Confusion Matrix Display, Classification Report, and Comparison of R2-Score & F1-Score Values from each prediction model.

#### 3.3.1. KNN Prediction Result

The Confusion Matrix Display, True Label VS KNN Predicted Label, and Classification Report prediction (blind) results using the KNN classification model can be seen in **Figure 9, Figure 10,** and **Table 5**, respectively.

#### 3.3.2. SVM Prediction Result

The Confusion Matrix Display, True Label VS SVM Predicted Label, and Classification Report prediction (blind) results using the SVM classification model can be seen in **Figure 11, Figure 12,** and **Table 6**, respectively.

#### 3.3.2. RFC Prediction Result

The Confusion Matrix Display, True Label VS RFC Predicted Label, and Classification Report prediction (blind) results using the RFC classification model can be seen in **Figure 13**, **Figure 14,** and **Table 7**, respectively.

3.3.2. Comparison of R2-Score & F1-Score Values (Blind Dataset)

The following are the results of the comparison of the R2-Score and F1-Score values of each classification model which can be seen in **Table 8**. In contrast to the results of the training dataset process, this blind dataset process produces the SVM model which has the highest value compared to other models.

*3.4. Facies Clustering with K-Means Clustering*

The facies clustering process is carried out using the combined data of the three wells, where before clustering, normalization is first carried out with StandardScaler, which is the same process as in the classification process. Here is the python code in doing this clustering.

```
km = KMeans(n_clusters=5, init='k-means++', n_init=100, max_iter=1000)
km.fit(clustering_data_scaled)

pd.DataFrame(scaler2.inverse_transform(pd.DataFrame(km.cluster_cen-
             ters_)), columns=dataABC.columns)
```

After running the python code above, the data tabulation of the clustering process results with K-Means Clustering can be seen in **Figure 15**. Furthermore, there are four outputs from this data clustering process, namely The Elbow Method Plot to Show the Optimal 'k', Average Silhouette Score at Various Number of Clusters, Box-Whisker Plot of Each Cluster (discussed in section 3.6 below), and Pair Plot per Cluster. The Elbow Method Plot to Show the Optimal 'k' can be seen in **Figure 16**. In accordance with the purpose of the Elbow Method is to show the most optimal value of k (number of clusters) from this facies clustering process. Based on **Figure 16**, it can be concluded that the number of clusters 20 is the maximum number of clusters. Then, for the Average Silhouette Score at Various Number of Clusters can be seen in **Figure 17**. This plot displays a measure of how close each point in one cluster is to points in the neighboring clusters and thus provides a way to assess parameters like the number of clusters visually. This measure has a range of [-1, 1]. Finally for Pair Plot per Cluster can be seen in **Figure 18**, where you can see how the relationship between the two features in the cluster and in the scatter plot section you can see the distribution of the data. Each feature has various relationships and are interrelated with each other. Based on **Figure 18** it is also known that there are clusters 0, 4, 8, 12, and 16.

*3.5. Comparison of Data Distribution Statistics per Facies Class*

Comparison of Data Distribution Statistics per Facies Class is presented in the form of a Box-Whisker Plot which can be seen in Figure 19. Based on Figure 19 it can be seen that there is a statistical distribution for each feature with the number of clusters being 19 clusters. Each Box-Whisker Plot shows quantitative data from each feature which has been grouped by cluster. Through these plots we can conclude how the minimum value, maximum value, and median of each feature, as well as early detection regarding the presence or absence of outlier data. The length of the boxes represents the level of spread of the data, where the longer it means the data is more spread out. The 'DTC' feature Box Plot shows a large number of long boxes indicating that this feature data has a good spread and few outliers. When compared with other features, it can be said that this 'DTC' feature can be one of the important features in the modeling process as will be discussed in the investigation and evaluation below.

*3.7. Investigation and Evaluation of Facies Classification Prediction Model*

After all the classification and clustering processes have been carried out, a prediction model for the classification and clustering of a rock facies will be obtained. However, this model must be investigated and evaluated again to get better prediction results along with

the use of the model in the future. Using the Random Forest algorithm, the Feature Importance process is carried out to find out how the features in the dataset affect the training process and produce a prediction model. The following is the python code to run the Feature Importance process.

```
from sklearn.ensemble import RandomForestRegressor

RF = RandomForestRegressor(n_estimators=100, random_state=0)
RF.fit(X_blind_scaled, y_blind)

sorted_idx = RF.feature_importances_.argsort()
plt.barh(X.columns[sorted_idx],RF.feature_importances_[sorted_idx])
plt.xlabel('Random Forest Feature Importance')
```

The results of Feature Importance can be seen in **Figure 20**, where it can be seen that the 'DTC' feature ranks first as the most important feature in this modeling process, followed by the 'GR' feature in the second position and so on for other features. Therefore, based on this, in the future it is necessary to consider the 'DTC' and 'GR' features for the development of the modeling process.

### *3.8. Figures, Tables and Schemes*

| | DEPTH_MD | CALI | RSHA | RMED | RDEP | RHOB | GR | NPHI | PEF | DTC | SP | BS | ROP | DRHO | FACIES |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| count | 29871.000000 | 29701.000000 | 0.0 | 29810.000000 | 29862.000000 | 28902.000000 | 29871.000000 | 29384.000000 | 28902.000000 | 29477.000000 | 0.0 | 29871.000000 | 29848.000000 | 29232.000000 | 29871.000000 |
| mean | 1477.632928 | 11.761533 | NaN | 2.090305 | 2.261846 | 2.198956 | 80.967951 | 0.373683 | 4.185141 | 124.158746 | NaN | 11.477007 | 25.597866 | 0.007022 | 2.324897 |
| std | 442.818362 | 1.607150 | NaN | 14.986359 | 16.000113 | 0.175694 | 36.150132 | 0.123706 | 1.402850 | 26.750434 | NaN | 1.503031 | 13.741399 | 0.034622 | 1.775686 |
| min | 684.270396 | 7.886668 | NaN | 0.323498 | 0.318176 | 1.556648 | 6.510116 | 0.003064 | 1.658541 | 48.621746 | NaN | 8.500000 | 0.703106 | -0.115128 | 1.000000 |
| 25% | 1096.260798 | 12.258815 | NaN | 0.844643 | 0.879092 | 2.071743 | 53.016529 | 0.271813 | 3.305484 | 101.170441 | NaN | 12.250001 | 16.146353 | -0.010219 | 1.000000 |
| 50% | 1474.659200 | 12.416596 | NaN | 1.168165 | 1.213086 | 2.158131 | 85.741882 | 0.406890 | 3.836642 | 136.910248 | NaN | 12.250001 | 25.099759 | -0.002041 | 1.000000 |
| 75% | 1852.991000 | 12.563520 | NaN | 1.582224 | 1.643712 | 2.352444 | 105.693970 | 0.469338 | 4.829510 | 144.459579 | NaN | 12.250001 | 33.263824 | 0.009434 | 4.000000 |
| max | 2361.883200 | 17.688429 | NaN | 1366.792481 | 1596.285523 | 2.789680 | 726.266113 | 0.768312 | 29.109318 | 177.843659 | NaN | 12.250001 | 299.693603 | 0.436384 | 8.000000 |

**Figure 1.** Overview of the Statistical Distribution of the Dataset

**Table 1.** Classification Report of KNN Model

| | **precision** | **recall** | **f1-score** | **support** |
|---|---|---|---|---|
| 1 | 0.97 | 0.98 | 0.97 | 2171 |
| 2 | 0.94 | 0.95 | 0.95 | 654 |
| 3 | 0.72 | 0.63 | 0.68 | 169 |
| 4 | 0.97 | 0.94 | 0.96 | 554 |
| 5 | 0.95 | 0.87 | 0.90 | 60 |
| 6 | 0.91 | 0.95 | 0.93 | 367 |
| 7 | 0.85 | 1.00 | 0.92 | 11 |
| accuracy | | | 0.95 | 3986 |
| macro avg | 0.90 | 0.90 | 0.90 | 3986 |
| weighted avg | 0.95 | 0.95 | 0.95 | 3986 |

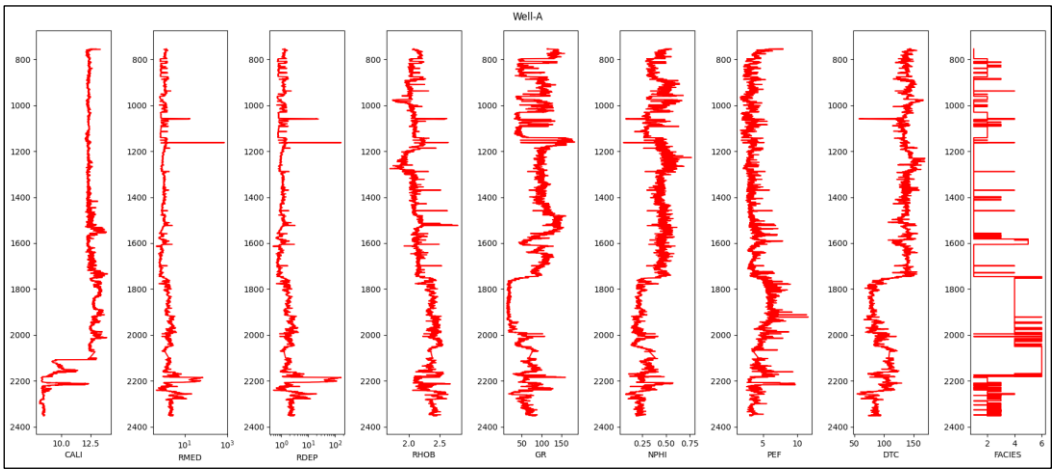



**Figure 2.** Log Data Plot of Training Dataset (ex: Well A)

**Table 2.** Classification Report of SVM Model

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 1 | 0.95 | 0.98 | 0.97 | 2171 |
| 2 | 0.89 | 0.95 | 0.92 | 654 |
| 3 | 0.65 | 0.20 | 0.30 | 169 |
| 4 | 0.90 | 0.91 | 0.91 | 554 |
| 5 | 0.90 | 0.92 | 0.91 | 60 |
| 6 | 0.85 | 0.88 | 0.87 | 367 |
| 7 | 0.92 | 1.00 | 0.96 | 11 |
| | | | | |
| accuracy | | | 0.92 | 3986 |
| macro avg | 0.87 | 0.83 | 0.83 | 3986 |
| weighted avg | 0.91 | 0.92 | 0.91 | 3986 |

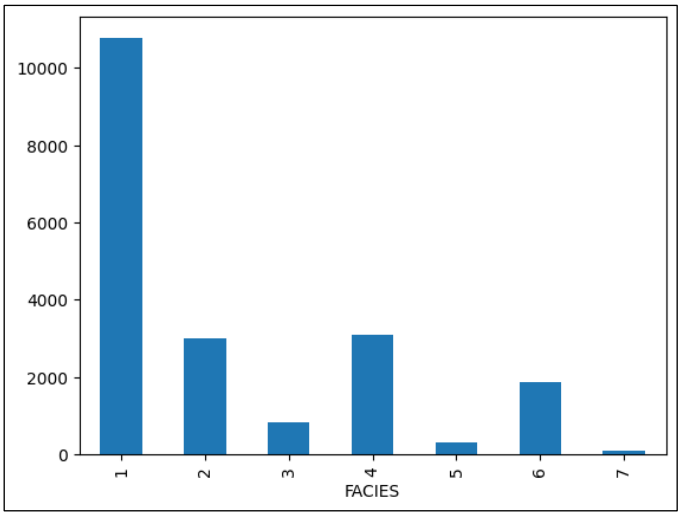



**Figure 3.** Facies Count Plot of Training Dataset

**Table 3.** Classification Report of RFC Model

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 1 | 0.95 | 0.98 | 0.96 | 2171 |
| 2 | 0.84 | 0.94 | 0.88 | 654 |

| | | | | |
|---|---|---|---|---|
| 3 | 0.60 | 0.02 | 0.03 | 169 |
| 4 | 0.88 | 0.91 | 0.90 | 554 |
| 5 | 0.91 | 0.83 | 0.87 | 60 |
| 6 | 0.85 | 0.86 | 0.85 | 367 |
| 7 | 0.92 | 1.00 | 0.96 | 11 |
| accuracy | | | 0.91 | 3896 |
| macro avg | 0.85 | 0.79 | 0.78 | 3896 |
| weighted avg | 0.90 | 0.91 | 0.89 | 3896 |

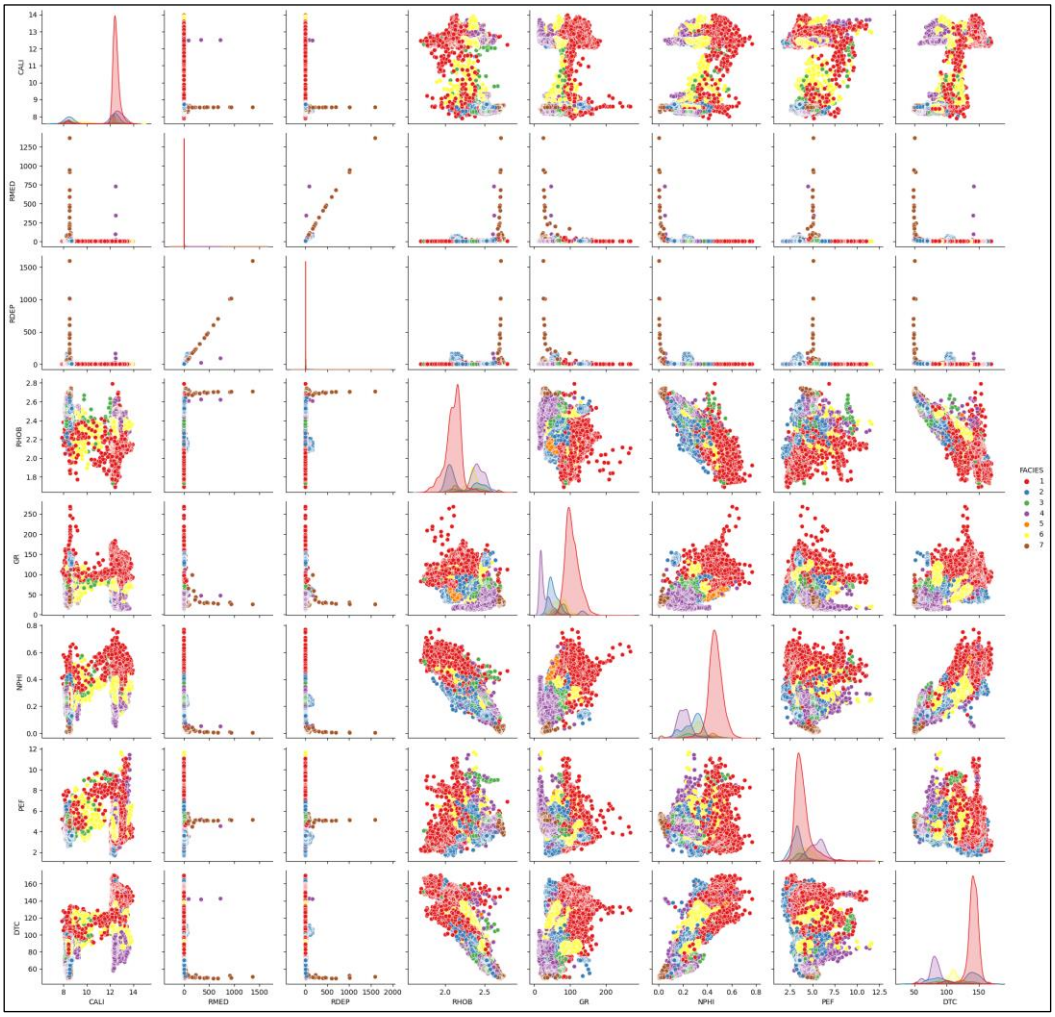



**Figure 4.** Pair Plot for Wireline Data Reading of Training Dataset

**Table 4.** Comparison of R2-Score & F1-Score Value

| Index | Model Type | R-2 Score | F1-Score |
|---|---|---|---|
| 0 | KNN | 0.90 | 0.95 |
| 1 | SVM | 0.85 | 0.91 |
| 2 | RFC | 0.82 | 0.89 |

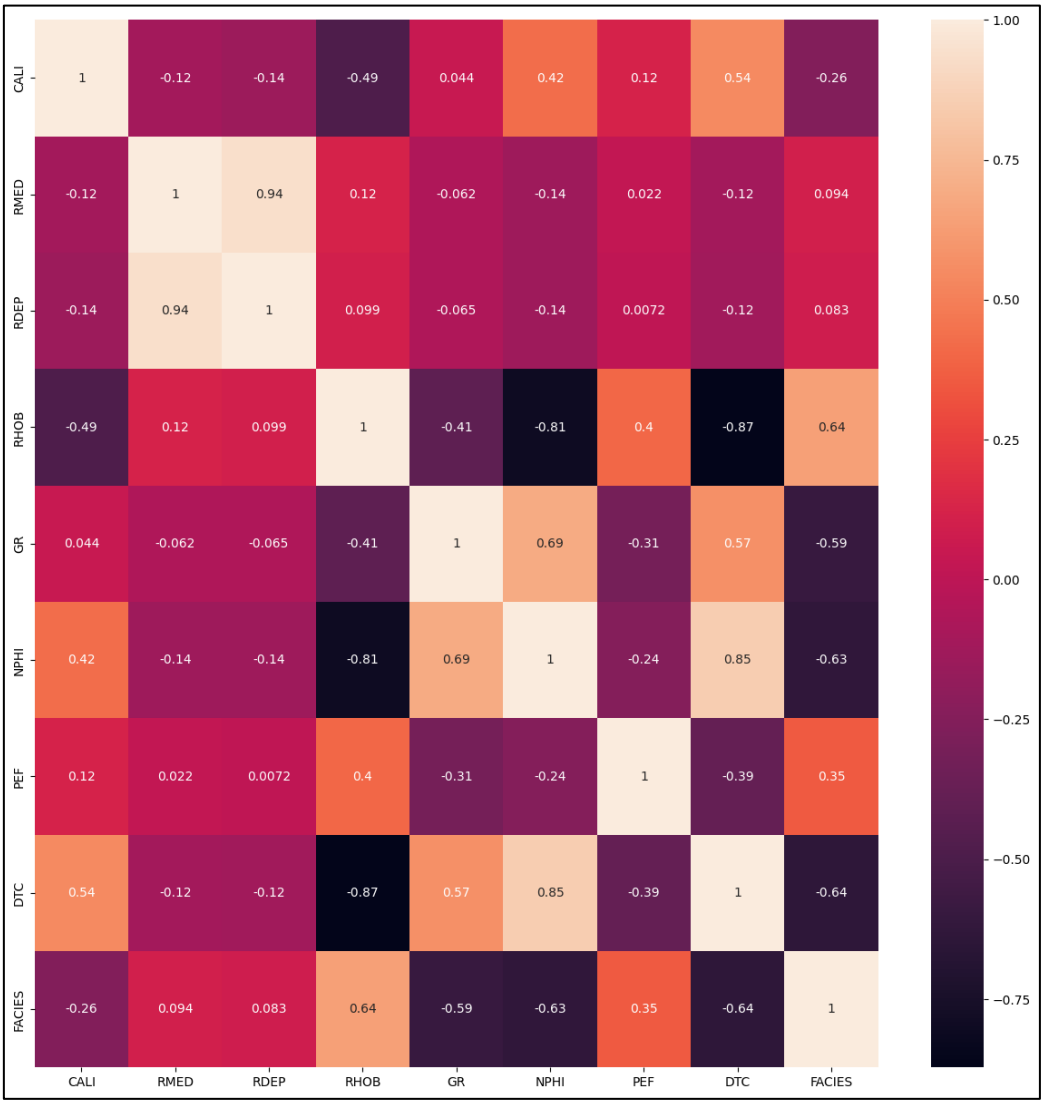



**Figure 5.** Heatmap Correlation Plot of Training Dataset

**Table 5.** Classification Report of KNN Prediction Model

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 1 | 0.94 | 0.96 | 0.95 | 4592 |
| 2 | 0.83 | 0.63 | 0.71 | 1421 |
| 3 | 0.09 | 0.08 | 0.08 | 311 |
| 4 | 0.77 | 0.80 | 0.79 | 1119 |
| 5 | 0.80 | 0.96 | 0.87 | 123 |
| 6 | 0.75 | 0.89 | 0.82 | 964 |
| 7 | 0.00 | 0.00 | 0.00 | 0 |
| 8 | 0.00 | 0.00 | 0.00 | 53 |
| accuracy | | | 0.84 | 8583 |
| macro avg | 0.52 | 0.54 | 0.53 | 8583 |
| weighted avg | 0.84 | 0.84 | 0.84 | 8583 |

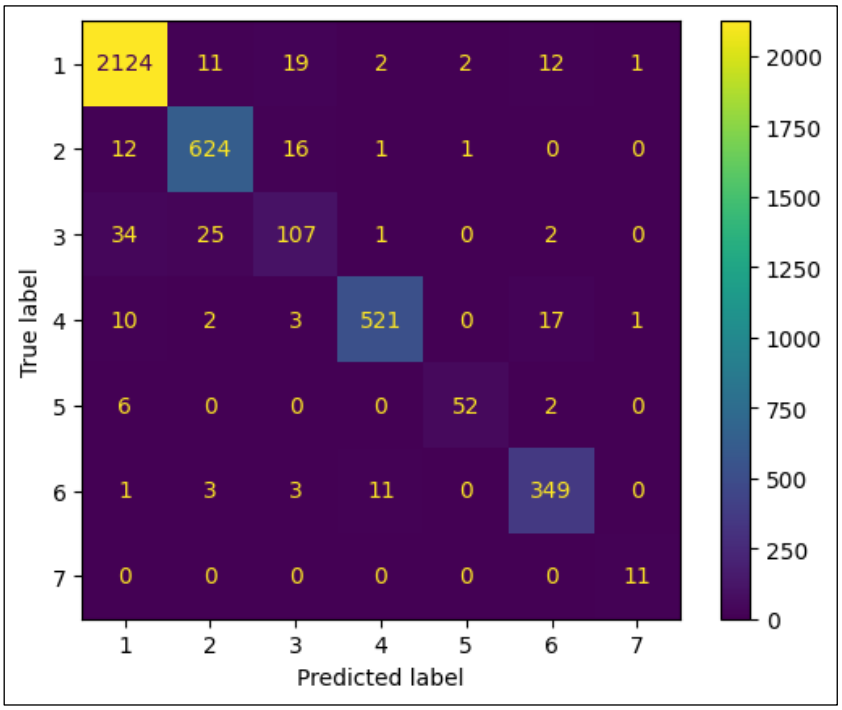



**Figure 6.** Confusion Matrix Display of KNN Classification Model

**Table 6.** Classification Report of SVM Prediction Model

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 1 | 0.94 | 0.98 | 0.96 | 4592 |
| 2 | 0.84 | 0.86 | 0.85 | 1421 |
| 3 | 0.00 | 0.00 | 0.00 | 311 |
| 4 | 0.82 | 0.84 | 0.83 | 1119 |
| 5 | 0.82 | 0.93 | 0.87 | 123 |
| 6 | 0.82 | 0.86 | 0.84 | 964 |
| 7 | 0.00 | 0.00 | 0.00 | 0 |
| 8 | 0.00 | 0.00 | 0.00 | 53 |
| accuracy | | | 0.88 | 8583 |
| macro avg | 0.53 | 0.56 | 0.54 | 8583 |
| weighted avg | 0.85 | 0.88 | 0.87 | 8583 |

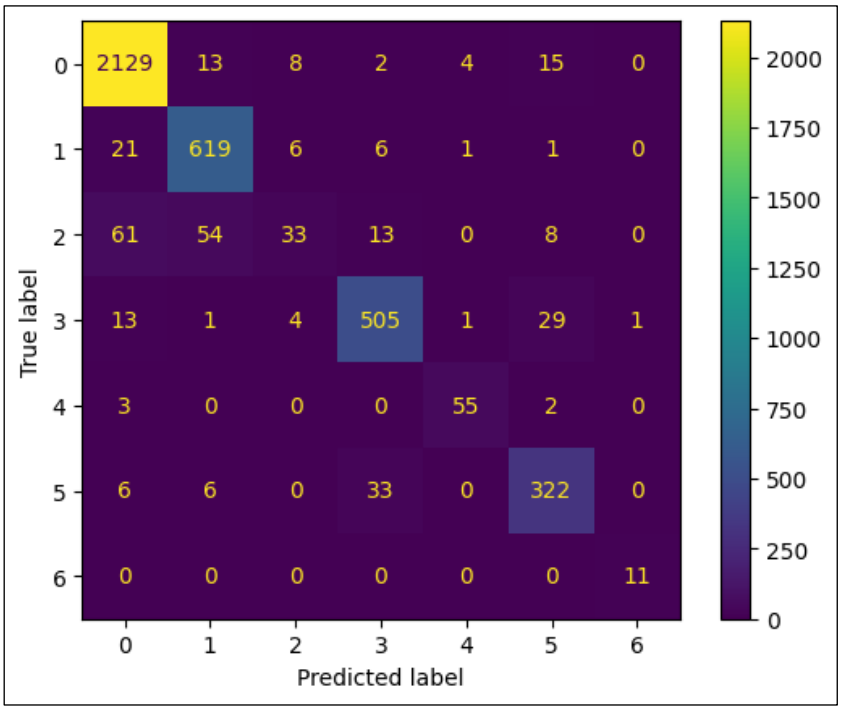



**Figure 7.** Confusion Matrix Display of SVM Classification Model

**Table 7.** Classification Report of RFC Prediction Model

| | **precision** | **recall** | **f1-score** | **support** |
|---|---|---|---|---|
| 1 | 0.94 | 0.97 | 0.95 | 4592 |
| 2 | 0.87 | 0.72 | 0.79 | 1421 |
| 3 | 0.00 | 0.00 | 0.00 | 311 |
| 4 | 0.71 | 0.86 | 0.77 | 1119 |
| 5 | 0.80 | 0.79 | 0.79 | 123 |
| 6 | 0.71 | 0.81 | 0.76 | 964 |
| 7 | 0.00 | 0.00 | 0.00 | 0 |
| 8 | 0.00 | 0.00 | 0.00 | 53 |
| accuracy | | | 0.85 | 8583 |
| macro avg | 0.50 | 0.52 | 0.51 | 8583 |
| weighted avg | 0.83 | 0.85 | 0.84 | 8583 |

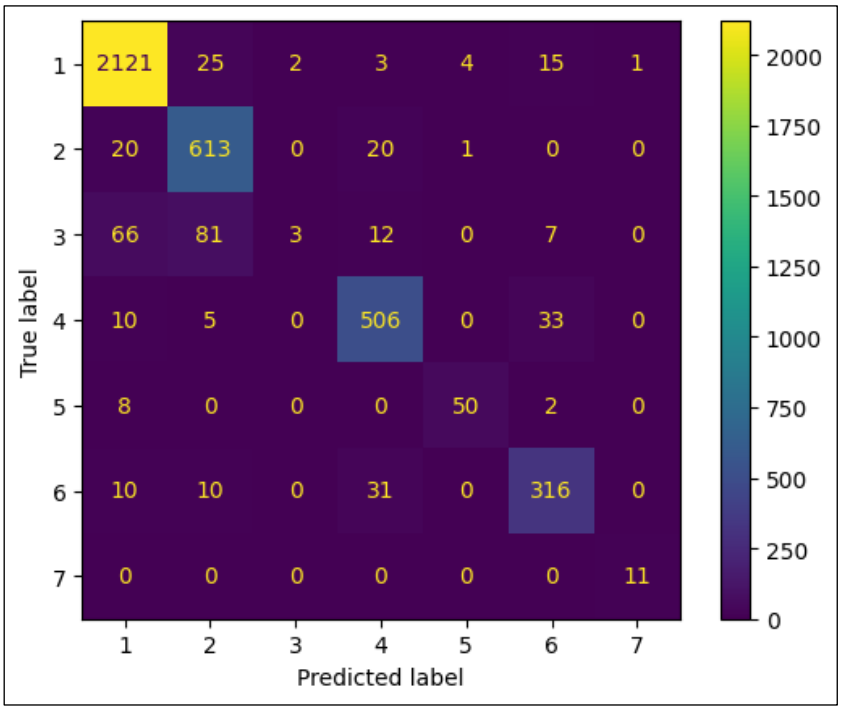



**Figure 8.** Confusion Matrix Display of RFC Classification Model

**Table 8.** Comparison of R2-Score & F1-Score Value

| Index | Model Type | R-2 Score | F1-Score |
|---|---|---|---|
| 0 | KNN | 0.66 | 0.84 |
| 1 | SVM | 0.71 | 0.87 |
| 2 | RFC | 0.63 | 0.84 |

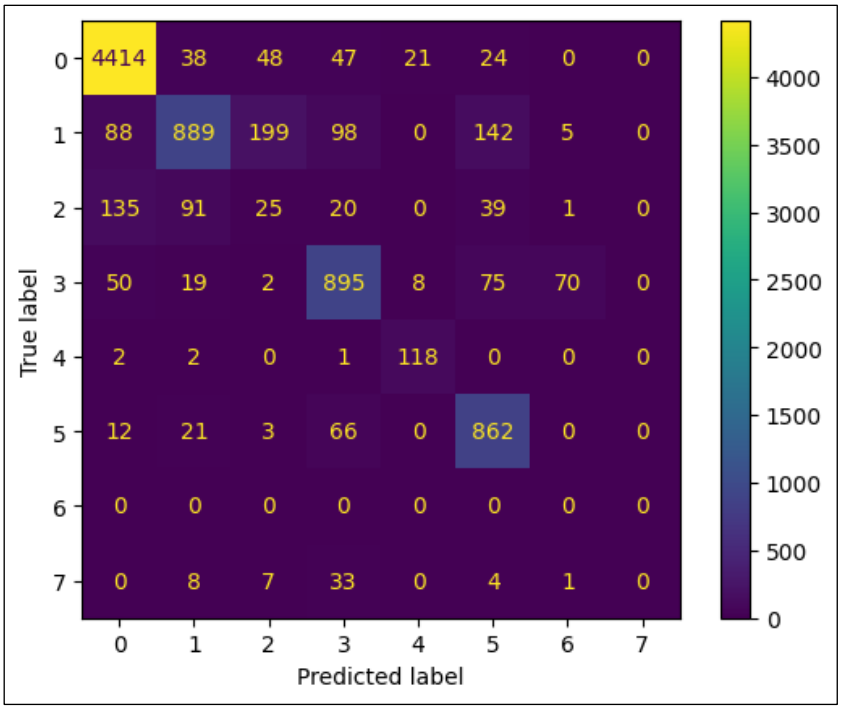



**Figure 9.** Confusion Matrix Display of KNN Prediction Model

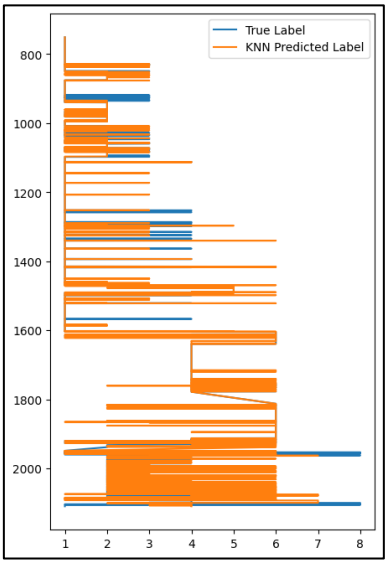



**Figure 10.** True Label VS KNN Predicted Label

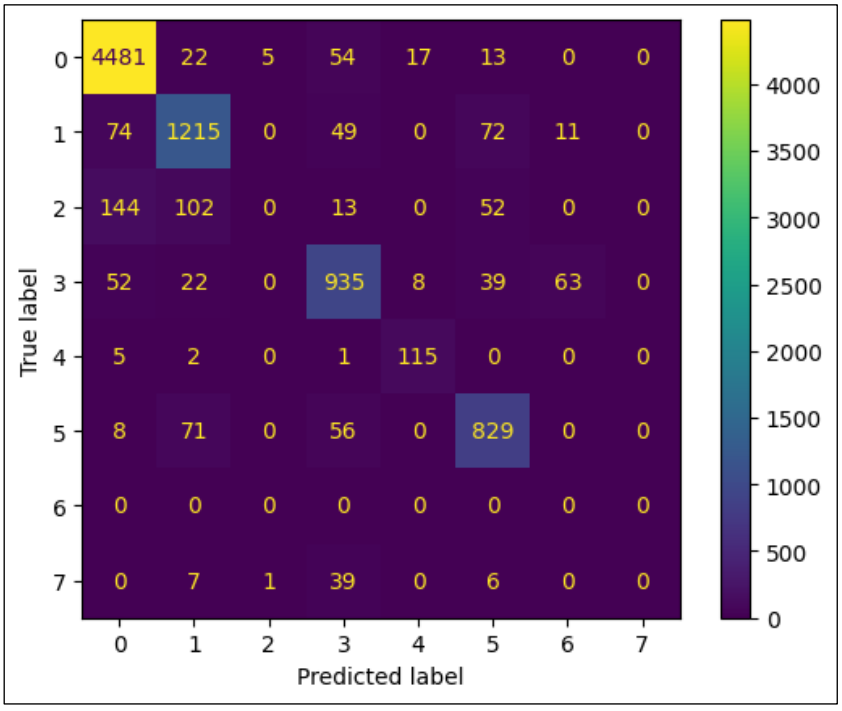



**Figure 11.** Confusion Matrix Display of SVM Prediction Model

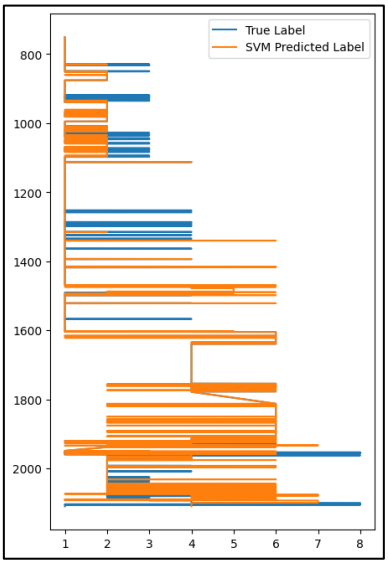



**Figure 12. True Label VS SVM Predicted Label**

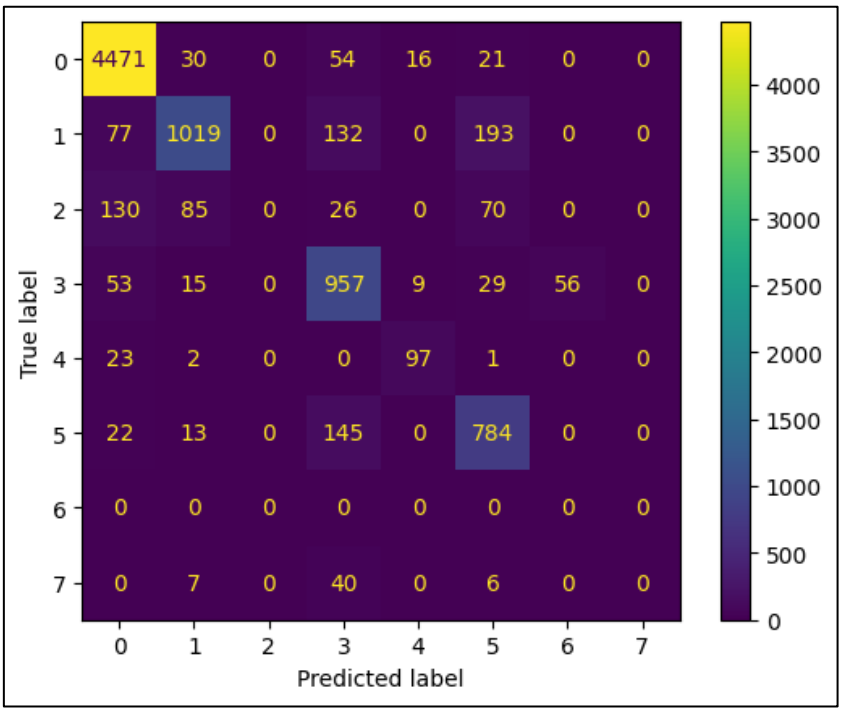



**Figure 13.** Confusion Matrix Display of RFC Prediction Model

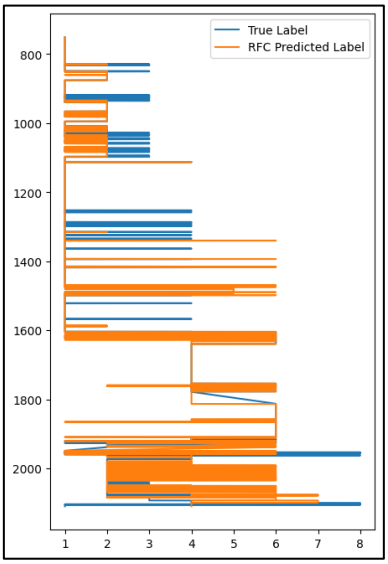



**Figure 14.** True Label VS RFC Predicted Label

| | DEPTH_MD | CALI | RMED | RDEP | RHOB | GR | NPHI | PEF | DTC |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 2070.642989 | 8.617190 | 4.735422 | 5.516940 | 2.401984 | 68.141398 | 0.254618 | 4.478600 | 92.283258 |
| 1 | 1257.145544 | 12.488280 | 1.041317 | 1.092958 | 2.097148 | 103.772405 | 0.467892 | 3.837199 | 142.010983 |
| 2 | 1816.246556 | 12.691088 | 2.090270 | 2.139485 | 2.392915 | 31.974326 | 0.216387 | 5.623295 | 88.979025 |
| 3 | 986.858613 | 12.407524 | 0.802330 | 0.854246 | 2.057857 | 52.412253 | 0.334132 | 3.014335 | 140.333710 |
| 4 | 2052.083374 | 8.982019 | 727.831516 | 706.148221 | 2.692090 | 30.031739 | 0.010981 | 5.010483 | 59.989110 |

**Figure 15.** Data Tabulation of the Clustering

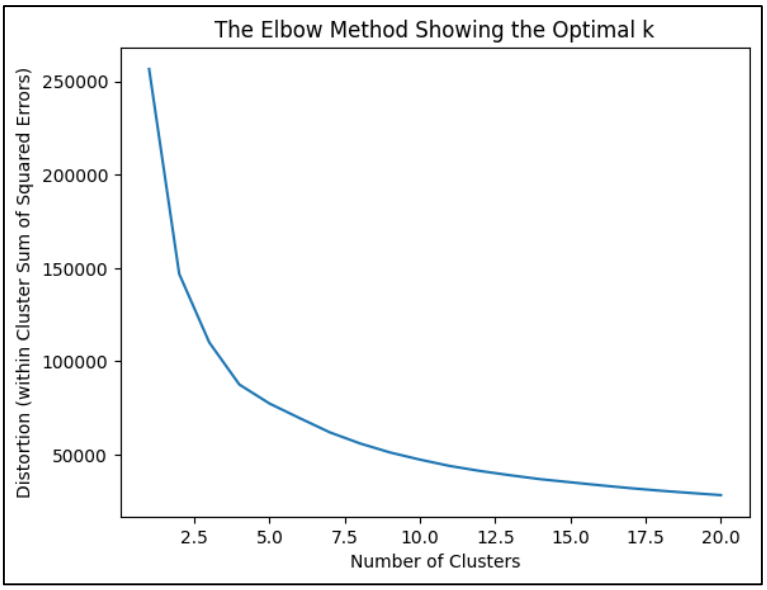



**Figure 16.** The Elbow Method Plot

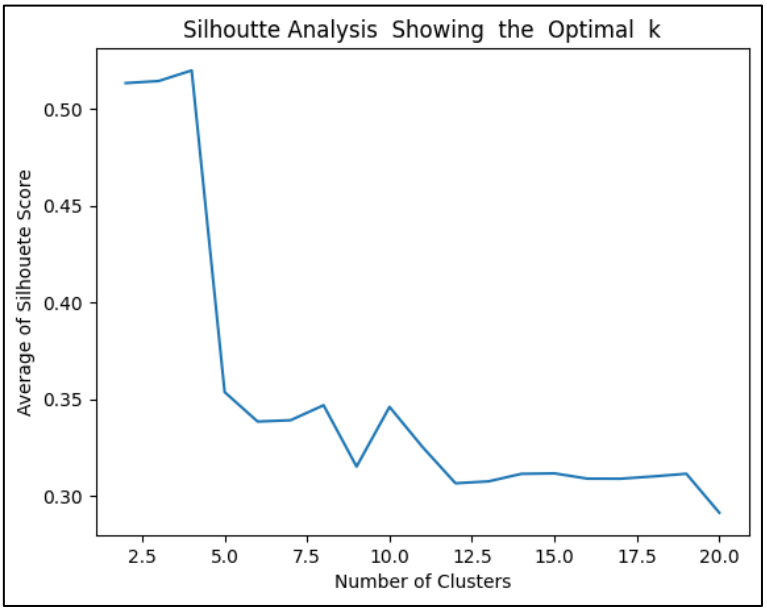



**Figure 17.** Average Silhouette Score at Various Number of Cluster

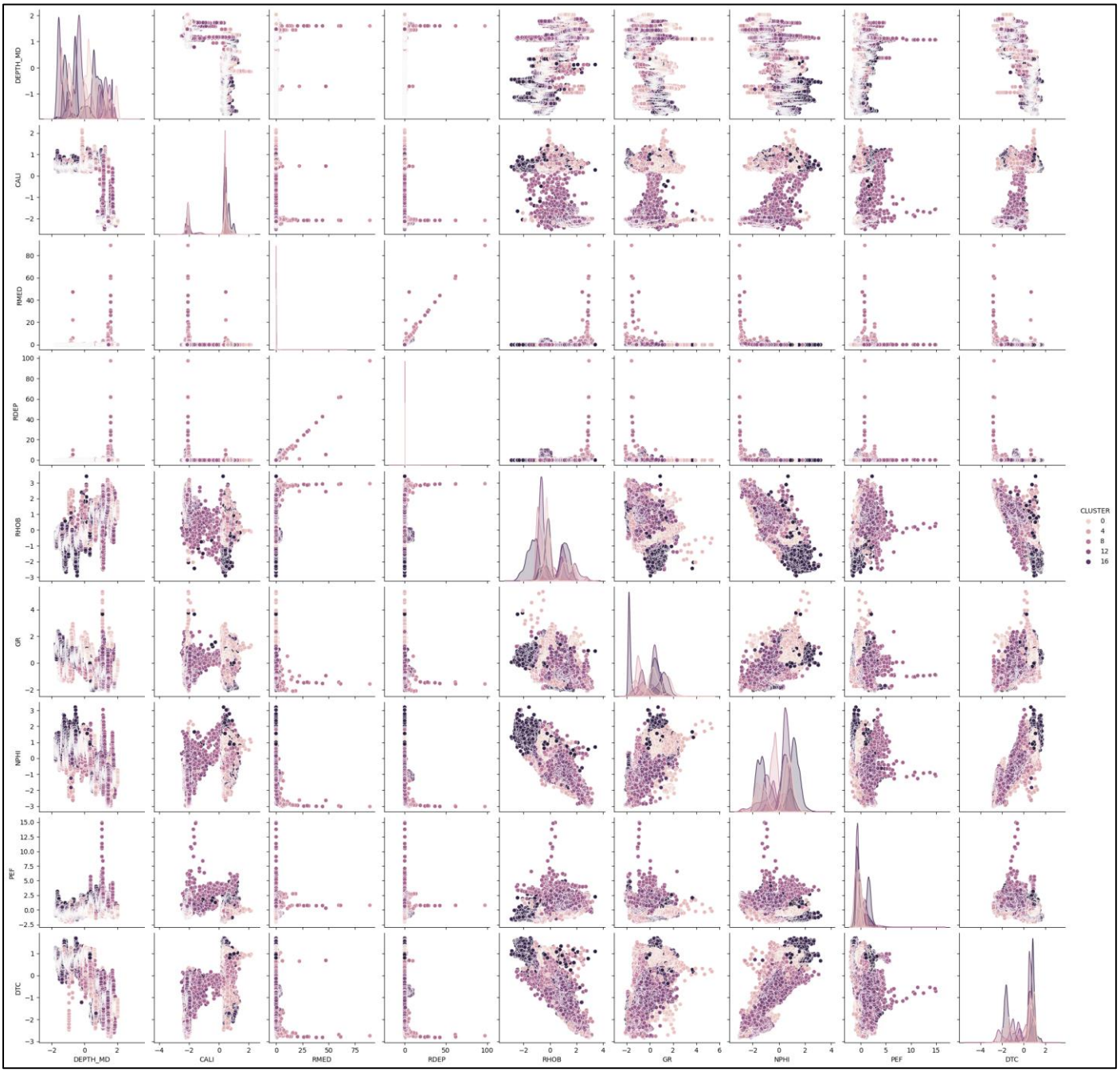



**Figure 18.** Pair Plot per Cluster

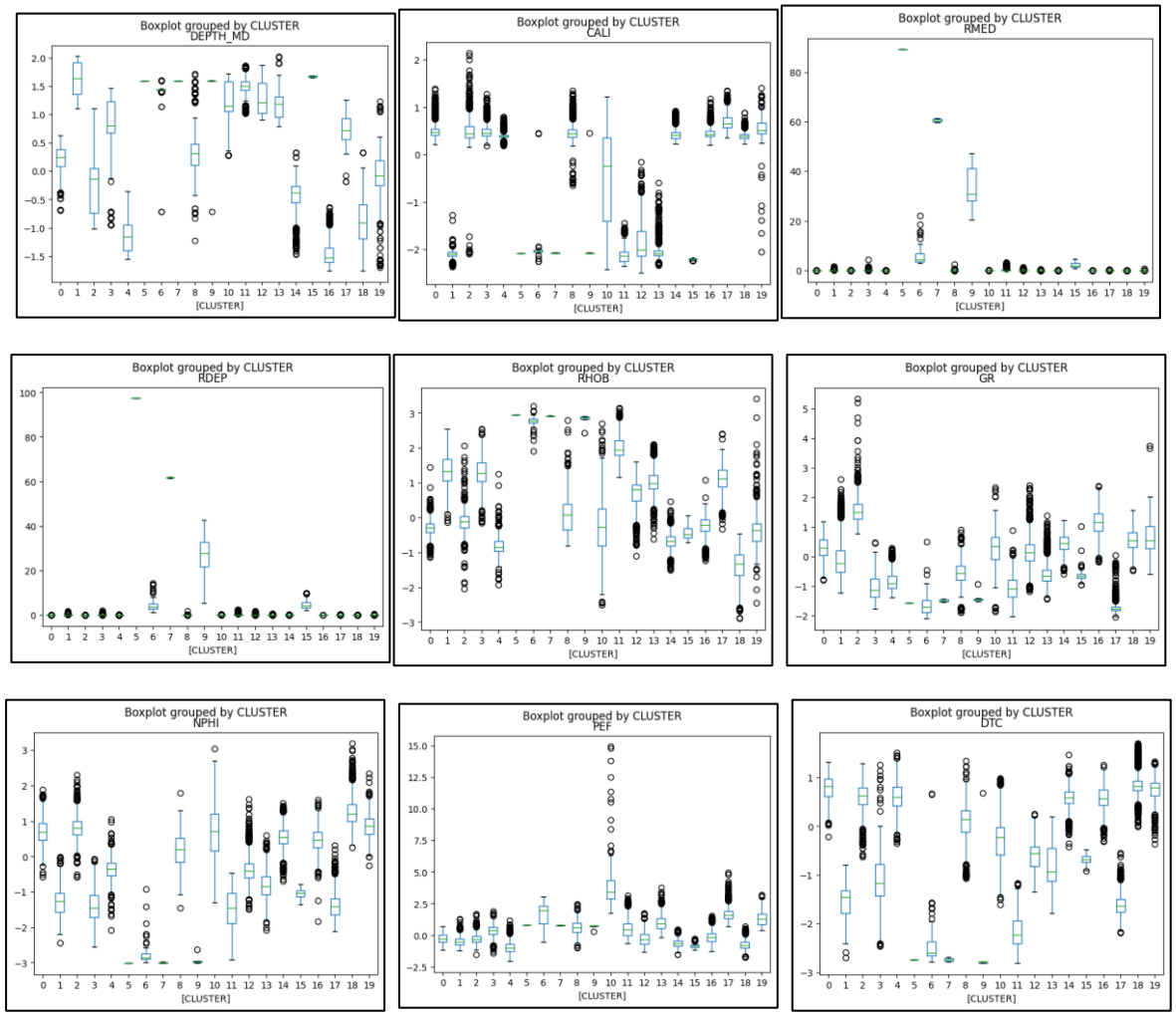



**Figure 19.** Box-Whisker Plot per Cluster

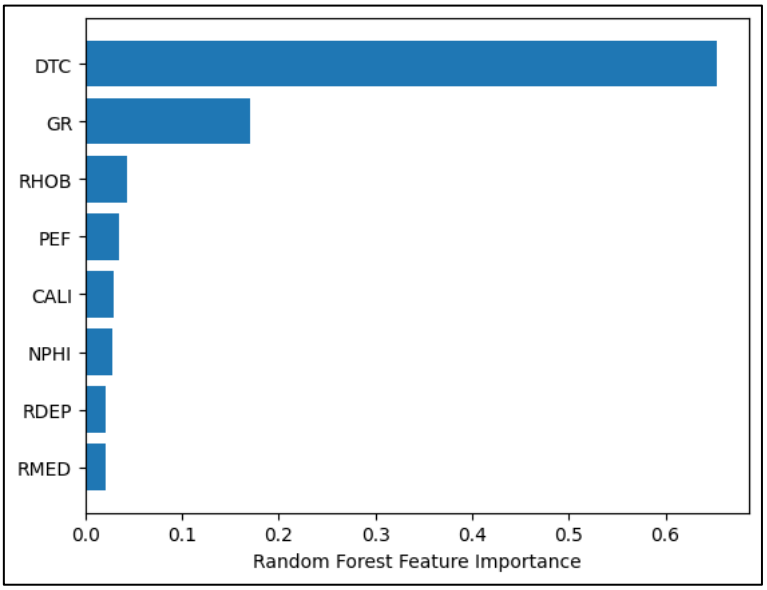



**Figure 20. Random Forest Feature Importance**

### *3.3. Formatting of Mathematical Components*

Equation 1:

$$\boldsymbol{dxy} = \sqrt{\sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} (\boldsymbol{x_i} - \boldsymbol{y_i})^2} \tag{1}$$

Equation 2:

$$\boldsymbol{\omega^T x + b = 0} \tag{2}$$

in which $\boldsymbol{\omega}$ refers to the weight vector and b is the threshold.

Equation 3:

$$\boldsymbol{f(x) = \omega^T x + b^T x + b} \tag{3}$$

Equation 4:

$$\sum \sum_{\boldsymbol{j \neq i}} (\boldsymbol{f(C_i, T)/|T|})(\boldsymbol{f(C_j, T)/|T|}) \tag{4}$$

Here, $(f(C_i, T)/|T|)$ is the probability that a selected case belongs to class $C_i$.

Equation 5:

$$\boldsymbol{\frac{1}{n} \sum_{i=1}^{n} \left[ min_j d^2(x_i, m_j) \right]} \tag{5}$$

is minimized, where $\boldsymbol{d(x_i, m_j)}$ denotes the Euclidean distance between $\boldsymbol{x_i}$ and $\boldsymbol{m_j}$. The points $\{\boldsymbol{m_j}\}$ ($\boldsymbol{j}$=1, 2, …, $\boldsymbol{k}$) are known as cluster centroids.

## 4. Discussion

There are several things that the author highlights, namely related to the use of data for training and data for blind. Both are very influential on the results of the training process and the resulting model and how the results are when blind datasets are carried out. In this paper, the author performs the process by combining the three wells into one data and then from the three data, one data is taken for the blind dataset (Well C) and the rest (Well A and Well B) for the training dataset. Based on the results obtained, when training obtained relatively high R1-Score and F1-Score values, but quite decreased when blind data sets were carried out but the accuracy value was relatively high. This can mean that the opposite can happen when different combinations are made. For example, Well A for the blind dataset and the rest (Well B and Well C) for the training dataset.

Therefore, the author also tried to consider other ways of doing this facies classification process. One way of comparison that has been done is to use only one well for training, namely Well A. While for the blind dataset, Well B is used. Based on the results obtained, initially the results of the training process obtained relatively high R2-Score and F1-Score values even close to 1. However, these values immediately dropped dramatically when blinded with the Well B dataset. Therefore, the author concludes that the determination of training datasets and blind datasets, as well as the way they are used also greatly affects the results of the facies classification prediction model created. The python script for this comparison method is also included in the **Supplementary Materials**.

In addition to the things described above, determining the features to be used as input in the training process is also the second most important thing in producing the best prediction model. This is because there are many factors that can affect it, even data that may be considered unimportant or less influential can become important data that can directly change the prediction results. Therefore, it is necessary to test several times to compare each prediction model made in the future in order to get the best facies classification prediction model.

## 5. Conclusions

This paper presents a comparison between four popular machine learning algorithms, such as K-Nearest Neighbor (KNN), Support Vector Machine (SVM), and Random Forest Classifier (RFC) in the context of classification facies, and K-Means Clustering in the context of facies clustering. In this study, experiments were conducted using a well log dataset from the UP Field that includes various attributes and information about facies. Each algorithm is then applied to the dataset and the results are evaluated using confusion metrics such as accuracy, precision, recall, and F1-score. K-Nearest Neighbor shows classification ability with R-2 score 0.90 and F-1 score 0.95. Support Vector Machine shows classification ability with R-2 score 0.85 and F-1 score 0.91. Random Forest Classifier shows classification ability with R-2 score 0.82 and F-1 score 0.89. This shows that KNN is the most recommended model in classification facies with a machine learning approach.

## 6. Patents

**Supplementary Materials:** The following Python Script can be downloaded at: https://github.com/azizpandoyo/UAS-AI/

**Author Contributions:** "Conceptualization, Muhammad Abdul Aziz and Muhammad Hisyam Ilyasa; Introduction, Muhammad Hisyam Ilyasa; coding script & materials of exploratory data analysis, Muhammad Abdul Aziz; coding script of KNN & SVM classification, Muhammad Hisyam Ilyasa; coding script of RFC classification & K-Means clustering, Muhammad Abdul Aziz; materials of machine learning methods, Muhammad Hisyam Ilyasa; training dataset process, Muhammad Hisyam Ilyasa; blind dataset process, Muhammad Abdul Aziz; feature importances for investigation and evaluation, Muhammad Abdul Aziz; comparison of data distribution per cluster, Muhammad Abdul Aziz; writing—original draft preparation, Muhammad Hisyam Ilyasa; writing—review

and editing, Muhammad Abdul Aziz; visualization, Muhammad Abdul Aziz; discussion, Muhammad Hisyam Ilyasa and Muhammad Abdul Aziz; conclusion, Muhammad Hisyam Ilyasa. All authors have read and agreed to the published version of the manuscript."

**References**

[1] B. Hall, "Facies classification using machine learning," *Leading Edge*, vol. 35, no. 10, pp. 906–909, Oct. 2016, doi: 10.1190/tle35100906.1.

[2] M. K. Dubois, G. C. Bohling, and S. Chakrabarti, "Comparison of four approaches to a rock facies classification problem," *Computers and Geosciences*, vol. 33, no. 5, pp. 599–617, May 2007, doi: 10.1016/j.cageo.2006.08.011.

[3] D. T. Larose and C. D. Larose, *Discovering knowledge in data : an introduction to data mining*.

[4] C. Cortes, V. Vapnik, and L. Saitta, "Support-Vector Networks Editor," Kluwer Academic Publishers, 1995.

[5] A. K. Dubey, U. Gupta, and S. Jain, "Comparative study of K-means and fuzzy C-means algorithms on the breast cancer data," *International Journal on Advanced Science, Engineering and Information Technology*, vol. 8, no. 1, pp. 18–29, 2018, doi: 10.18517/ijaseit.8.1.3490.

[6] R. Agrawal, "K-Nearest Neighbor for Uncertain Data," 2014.

[7] R. Liu *et al.*, "Application and Comparison of Machine Learning Methods for Mud Shale Petrographic Identification," *Processes*, vol. 11, no. 7, p. 2042, Jul. 2023, doi: 10.3390/pr11072042.

[8] C. Liu, C. Liu, M. White, and G. Newell, "Measuring the accuracy of species distribution models: a review," 2009. [Online]. Available: http://mssanz.org.au/modsim09