Title: Evaluating Different Clustering Algorithms for Prediction of Rock Types and Permeability

Authors: Munish Kumar[a] and Tan Wen Xuan[a]

Affiliations: [a]Singapore University of Social Sciences, School of Business,463 Clementi Rd, Singapore 599494

Corresponding author. Email: munishkumar001@suss.edu.sg

Abstract: Rock typing has multitudes of uses, from optimizing drilling spots, to determining perforation zones, evaluating in-place volumes using static and dynamic models, to better understanding complex flow properties that take place in oil and gas reservoirs. Studies on rock typing have mainly utilized porosity and permeability measurements converted to rock typing indices, with newer studies introducing elements of supervised machine learning as well. However, a comparison of different unsupervised machine learning algorithms for rock typing applications has not been widely studied. This paper aims to perform a comparative study of the performance and outputs from 5 different unsupervised machine learning models, which we will benchmark against a modified iterative multi-linear regression (IMLR) rock typing technique. This study will be conducted on core data comprising 2000 unique data points from the United Kingdom (UK). The 5 unsupervised machine learning model are the K-Means, Self-Organising Map (SOM) + K-Means, Density-Based Spectral Clustering of Application with Noise (DBSCAN), Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH) and Gaussian Mixture Models (GMM). This study's results show that while K-Means and BIRCH rock typing have well defined/ well-spaced clusters, they do not do as good a job at predicting permeability as DBSCAN and GMM, indicating that while machine learning metrics define uniqueness of a solution, it is ultimately the interpreter that must review the data and decide on the suitability of the model.

One-Sentence Summary: This paper aims to explore the performance of rock typing using different unsupervised machine learning algorithms against a modified IMLR rock typing method.

Keywords (minimum 6): Oil and Gas, Rock Typing, Unsupervised Machine Learning, K-Means, Density-Based Spectral Clustering of Application with Noise (DBSCAN), Self-Organising Map (SOM), Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH), Gaussian Mixture Models (GMM).

Introduction

Rock typing is based on the simple principle that not all reservoir rocks are created equal. Rocks that are similarly deposited, buried under similar conditions, and undergo similar diagenetic processes will share a set of unique characteristics [1]. In work by Rushing, 3 different classes of rock typing are identified – depositional, petrographic, and hydraulic [2]. Of the 3, hydraulic rock typing (RT) seeks to characterize reservoir rocks into similar groups based on petrophysical properties which (a) quantifies the ability of fluids to flow and (b) determines the availability of pore volume for storage [3]. The physical properties that govern fluid storage and flow are fundamentally linked to the dimension (size), geometry (shape, surface roughness, tortuosity) and distribution of the pore and pore-throat network. Such characterisation is routinely performed using routine core analysis porosity and permeability measurements [4], but can be performed with mercury injection capillary pressure (MICP) as well [5]. These unique porosity-permeability relationships and flow characteristics are integrated across multiple scales (nm to km), so that large scale geologic features can be adequately represented from small scale petrology and petrophysical properties. Often, RT at the core scale is done manually, with experienced litho-stratigraphers and experts in core analysis involved in the process of subdividing the data into families, based on depositional and geological principles.

In certain instances, the unique relationships from porosity and permeability are not readily discernable. Assuming stratigraphic continuous rock units have similar reservoir properties, and assuming a capillary tube model as defined by Kozeny-Carman [6], the concept of flow units/ flow zone index (FZI, μm) was developed to honor the geology while deconvolving the relationship between these 2 variables [7]. Of particular importance is the mean hydraulic radius of pore throats, R_{mh}, which is used to evaluate the reservoir quality index (RQI, μm), and pore volume to grain volume ratio (also known as the normalized porosity index or ϕ_z), defined as:

$$R_{\mathrm{mh}} = \sqrt{\frac{k}{\emptyset}} \quad (1)$$

$$RQI = 0.0314R_{mh} \quad (2)$$

$$\emptyset_z = \frac{\emptyset}{1-\emptyset} \quad (3)$$

$$FZI = \frac{RQI}{\emptyset_z} \quad (4)$$

where k is permeability (mD) and ϕ is porosity (V/V). For rocks with similar RQI, a graph of (log) FZI vs (log) ϕ_z will fall on a straight line with a slope of -1. Rock samples with different RQI values will fall on other parallel lines with even slope but different intercepts. The intercept of these lines at $\phi_z = 1$ defines the mean pore throat size of each linear group of sample points. Numerous authors have derived combinations of these 2 properties to generate variations of FZI (e.g., FZI*, FZIM and MFZI) either from core data or applied data from well-logs e.g. Resistivity Zone Index (RZI) (see Appendix 1 for details).

The clustering of FZI that are close in proximity will determine the number of rock types (hydraulic flow units, HFUs) present in the core dataset. Several approaches are used to cluster core data into HFUs, with iterative multi-linear regression (IMLR) and analysis by least square regression being one of the methods used [8, 9]. The many approaches, however, all aim to do something similar - using FZI to form distinct groups, for determination of the number of rock types or hydraulic flow units there are in the data.

The iterative multi-linear regression (IMLR) technique is a method that starts off with a (log) ϕ_z vs (log) RQI plot of the data. Initial guesses of straight lines with slope = 1 are plotted based on the chart. The data points are then allocated to the nearest line, and line intercepts are then recalculated using the least-square regression. Resulting values are then compared the old intercepts to see if the variance is within acceptance range. The process repeats until the variance is acceptable for all lines of the plot [9].

Using Machine Learning for RT applications

While rock typing using correlations is one way in which reservoir rocks are classified, the advent of faster computing coupled with advances in memory has encouraged geoscientists to attempt new data driven means to attempt rock classification. Some of the newer rock typing work now utilizes machine learning (ML) to complement or even supplant existing methods as the expectation is that such methods would prove more accurate in carrying out anomaly detection and correlative evaluation. Most of the work has however been via the utilization of supervised machine learning which requires expert knowledge to firstly prepare the training sets into fundamental rock types that can be later be applied to the testing data sets. What has been less researched is how unsupervised machine learning algorithms would perform in similar rock classification tasks.

Mohamed et. al [10] approached the task of rock typing using both supervised and unsupervised algorithms. They started with a data set of 128 carbonate core samples, and firstly applied supervised machine learning (Extreme Gradient Boosting or XGB) to generate an output in the form of a FZIM*. The second step of this process involved using the FZIM* output as input into a K-Means algorithm, ultimately resulting in 4 unique RTs. A few things to note about this work is that their initial data set was small, input variables other than porosity and permeability were utilized e.g. connate water saturation and data from MICP, and they required multiple ML methods to perform the classification.

Mohamed et. al. [11] similarly performed rock typing using unsupervised and supervised ML algorithms, with supervised ML classifiers such as Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Random Forest – Decision Tree and one neural network approach. They only used one unsupervised ML method, K-Means. Input variables were in the form of log data from 8 wells, but with facies already manually classified prior. They found that there were inherent differences between what was manually classified and what was classified computationally and stated that the unsupervised learning algorithm was the worst performing.

Zakyan et. al. [12] proposed a method for RT of log data with artificial neural networks (ANN) [13]. HFUs were derived using a back propagation technique based on 35 samples from routine core analysis (RCAL). The result is that there were no missed classifications using the proposed methodology on the 35 samples and it even outperformed the multi-linear regression method. In both the work by Mohamed et. al. [11] and Zakyan et. al. [12], the data was mostly log based data and did not use much core data. Additionally, their described methodology was multi step, making it difficult for general application. Mohamed et. al. additionally required that the data be labelled before applying their method.

Scope & Methods

Given the lack of information regarding unsupervised ML methods for core classification, we were interested in comparing the performance of 5 different unsupervised ML algorithms and benchmarking it against the more "conventional" RT method, namely the iterative multi-linear regression (IMLR) technique. We wanted to use a data source that was heterogeneous and would encompass a wide range of values, while also having sufficient density of data to warrant a ML technique. Ultimately, we opted to utilize open source data from the UK North Sea for this work [14, 15, 16].

The data was available in an initially unstructured format. Input data files had to filtered, collated, and refined before they could be further pre-processed. For instance, there were close to 2000 individual files which needed collation. The format of the data present within each file was not homogeneous (confirmed through random sampling). The file types themselves were not consistent, with 43% in excel format, 39% in TIFF format, 14% in PDF format and 4% in other formats including jpeg and ASCII formats. From the excel data, only 33% was relevant for use. There were also corrupt files present which needed removal. We observed that some data files had as many as 250 columns, of which a majority were NULL or empty. We found that the most complete of the columns was only the porosity and permeability data, the very minimum of the data types needed for rock typing. Collating, sorting, combining and standardizing the data manually would have been extremely time consuming and tedious, with an estimate of >3000 man-hours spent doing the latter 2 alone. Repetitive tasks like this would also likely result in human errors, so we opted for a computational solution which automatically merged the individual data files into a single "mega-merge" (MM) file.

Following the generation of this MM file, we performed a "clean-up" by removing data which was of the NULL data type, alphanumeric or non-physical data (negative numbers or zero division errors). What was remaining was computationally corrected for, using statistical methods where empty data types were filled in using the KNN imputation method. The method replaces the missing values using the nearest neighbors average values and outliers were removed to prevent data skew. The data is min-max normalized to shrink the feature to the same scale of 0 to 1 and to reduce the space between data points for distance based algorithms to work well [17] before being split into 2 separate csv files, one containing the unnormalized data file (Ori) and the final input data file containing the normalized data (Norm). We schematically illustrate this process in Figure 1.

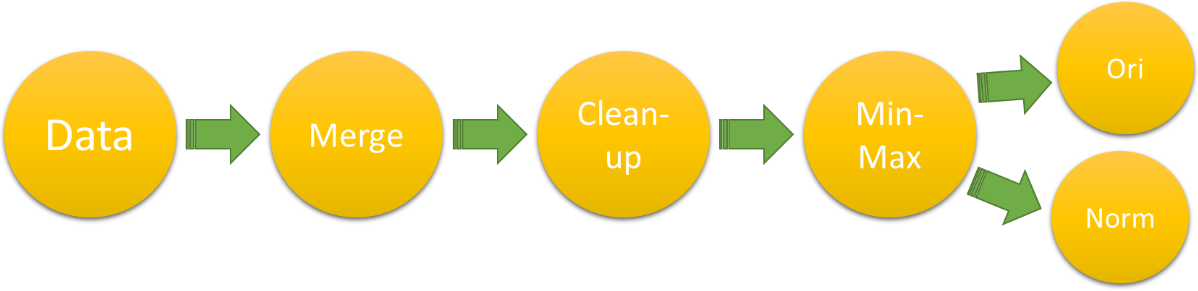


Figure 1: *Generic data preparation flowchart*

Exploratory Data Analytics and Down sampling

Our final data set was made up of >100,000 unique data points, all from sandstone reservoirs. General histograms of the porosity and permeability of this data set is given in Figure 2. This data was collected from a total of an estimated 50+ unique wells with a depth range of about 400m - 18,000m. A cross-plot of the data is shown in Figure 3, and a table of descriptive statistics is given in Table 1.

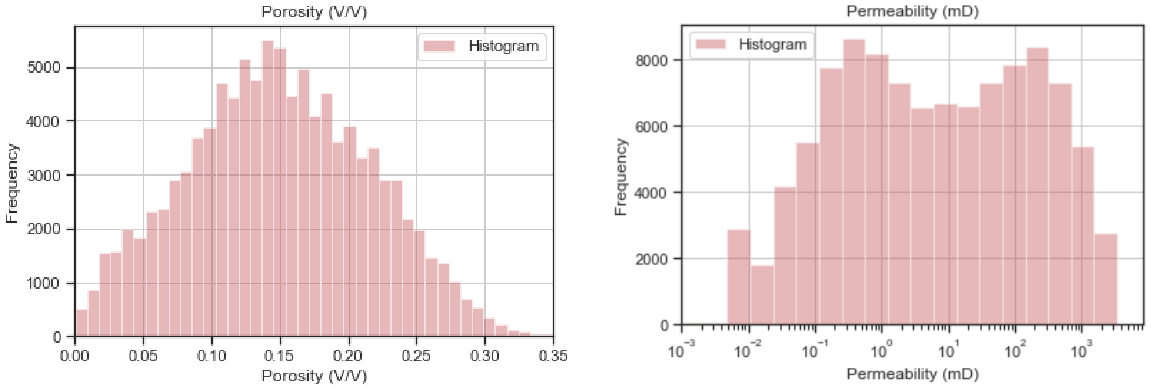


Figure 2: *Histograms depicting (Left) Porosity and (Right) Permeability of the core database*

While the data set above satisfies the requirement for sample size and heterogeneity, we are faced with a new challenge in that the data itself is too dense to properly perform core classification using a clustering process, based purely on porosity and permeability. As stated earlier, RT split by porosity-permeability would conventionally be manually done and assisted with core description and facies observation on physical core samples. However, for data sets like this, this would not be possible.

For optimal clustering using computational methods, the individual clusters would have to be distinct and separate enough so that distance-based methods (which most clustering algorithms are based on) would work reasonably well. Samples that were repeats or which had values close to one another (highly correlatable to one another) do not necessarily add value to partitioning out of the individual clusters. Another issue with very large data sets is the rapidity of the analysis; on data sets like this, running multiple clustering algorithms would be time consuming and computationally expensive.

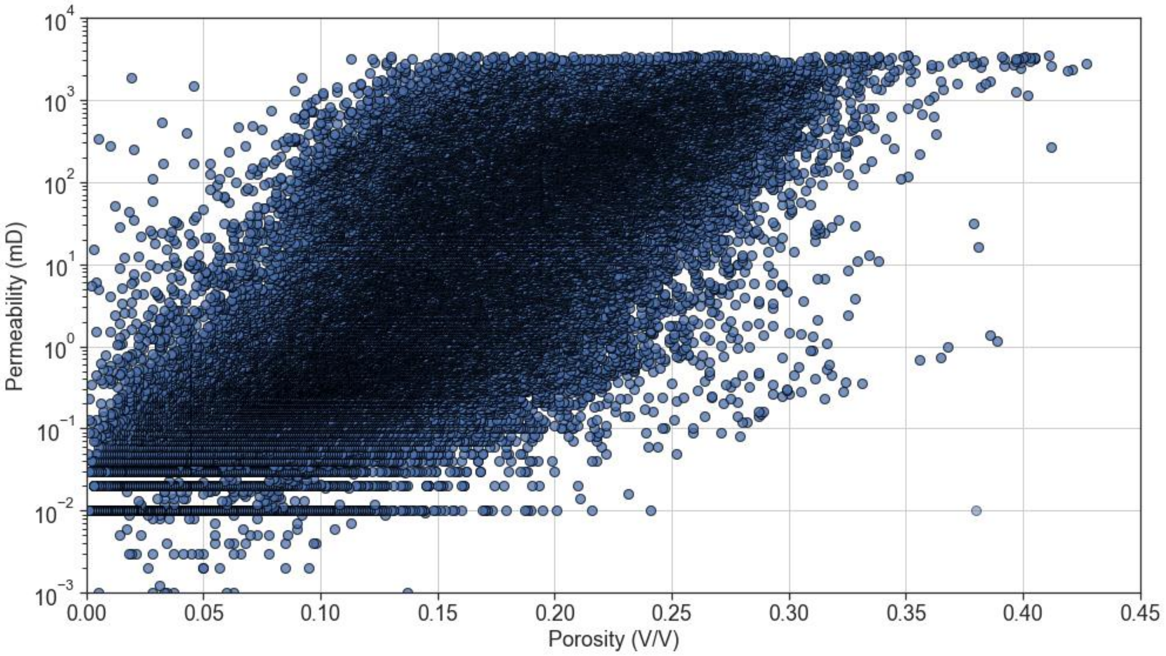


Figure 3: *Porosity and Permeability Cross-plot of Whole Dataset*

Table 1: *Descriptive Statistics for the MM Core Database*

Property	mean	std	min	25%	50%	75%
Porosity (V/V)	0.15	0.067	0.001	0.102	0.148	0.198
Permeability (mD)	176	434	0.001	0.37	5.8	113

We therefore attempted to down sample the data, being careful to optimize run time, while ensuring that no critical data was sacrificed in this process. We did this using a down sampling technique based on a relatively simple K-Means algorithm, where we tested 3 separate sized datasets of 2,000, 40,000 and 100,000+ (full data set) data points. We did not vary the algorithm hyperparameters, except for the random state, which we kept fixed in all 3 cases. We also did not optimise the number of clusters and instead arbitrarily selected 4 clusters. A table of execution time is given in Table 2. From the comparison in the table, we note a 91.5% improvement in running the downsampled data set, as compared to the full dataset.

Table 2: *Time Taken to run K-Means for the different dataset sizes*

Datasets	Time Taken (ms)				Run time Improvement
	Trial 1	Trial 2	Trial 3	Average	
2,000	108	104	106	106	91.5%
40,000	547	491	515	518	58.2%
100,000+	1290	1180	1260	1240	-

Given in Figure 4 is an overlay of a random sampling of 1000 data points from each of the 3 data sets (100,000+, 40,000 and 2,000 points), with solid filled points represents the outcome of the clustering algorithm where all 3 datasets are identical, while semi-translucent points are data points which have been reassigned to a new cluster value as a result of the down sampling. Overall, we observe <10% of the points are reassigned. We therefore consider the down sampling approach to be robust, as it preserves data integrity while saving on computational resources.

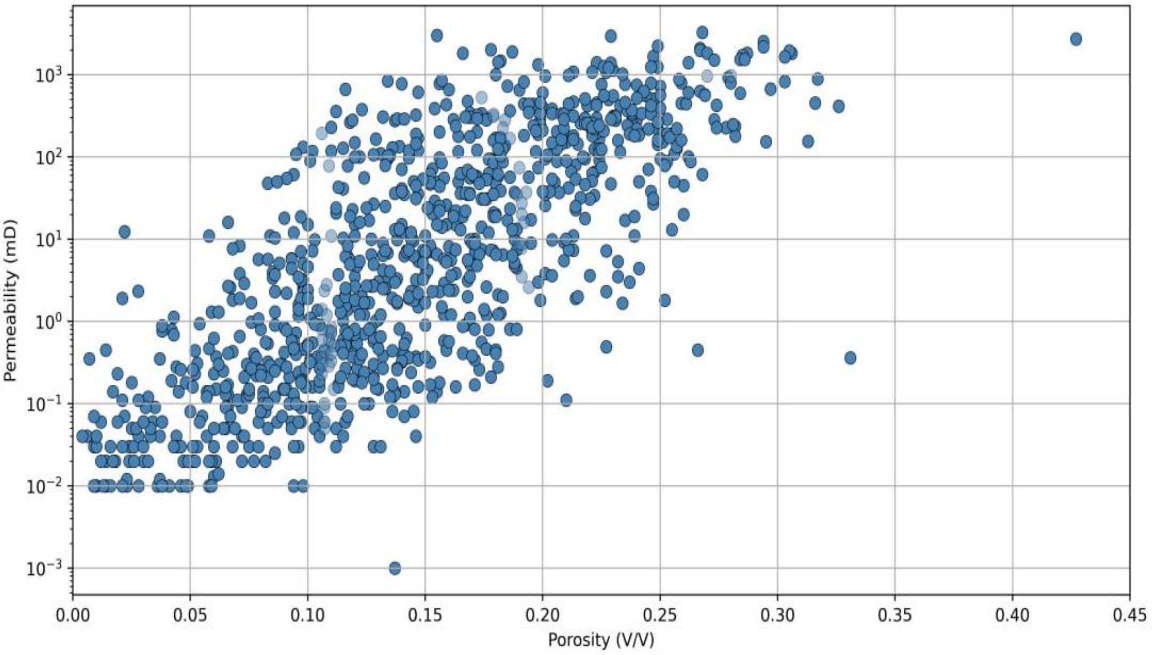


Figure 4: *Porosity and Permeability Cross-plot of the 3 Datasets Clustering Results*

ML Algorithm Choices

We select the following unsupervised ML algorithms for our work which are a combination of traditional unsupervised ML models and artificial neural networks (ANN). The ML models selected are heterogeneous, with (a) K-Means (centroid based clustering), (b) Self-Organising Map (SOM) + K-Means (ANN based clustering), (c) Density-Based Spectral Clustering of Application with Noise (DBSCAN) (density-based clustering), (d) Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH) (hierarchical clustering) and (e) Gaussian Mixture Models (GMM) (probabilistic based clustering). These algorithms were chosen due to the popularity of the algorithms as well as each being based on a different clustering modality.

(A) K-Means

K-Means is one of the most widely used clustering methods. It is a centroid-based clustering algorithm based on principles of Euclidean distance, where observations are split into k clusters, and where each observation is than attributed to the nearest mean or cluster centroid. The algorithm iterates such that the center point will shift to be the average of all the data points within that cluster. The shortcoming of the algorithm is that it must be initialized with an initial number of clusters [18]. We schematically illustrate the process of K-Means model training in Figure 5.

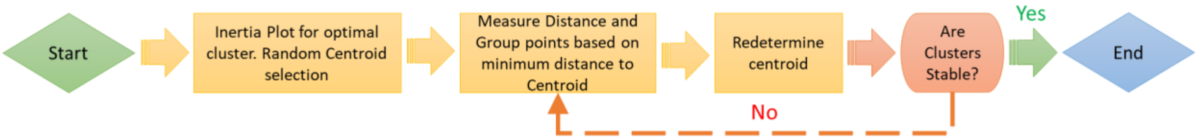

Start
Inertia Plot for optimal cluster. Random Centroid selection
Measure Distance and Group points based on minimum distance to Centroid
Redetermine centroid
Are Clusters Stable?
Yes
End
No

***Figure 5:** Process of K-Means*

(B) Self-Organising Map (SOM) + K-Means

A Self-Organizing Map (SOM) is ANN-trained using unsupervised learning, to ultimately produce a low dimension, discretized representation of the input sample. Unlike K-Means, where the centroids (nodes) are free moving and have no relationship to one another, a SOM "feature map" tries to topologically preserve the properties of the input space by "pulling" neighboring nodes along with it, making it ideal for applications in clustering. An optimal SOM is one where the feature space is distorted as minimally as possible. SOMs learn by competitively adjusting weights to neurons. The final output of a SOM map is one where all neuron positions are known and where the calculated relative Euclidean distance is retained between points, with points close to one another being mapped to similar units.

However, SOMs can sometime give large numbers of clusters, necessitating a simplification of sorts by combining these clusters, done with K-Means. The distance between neurons and data can be calculated and organized in a (dissimilarity) matrix, with this matrix as an input for K-Means [19].

For this work, we utilized the "MiniSOM" library to construct the SOM. We initialized a large SOM grid of 10X10 with default sigma and learning rates, before applying K-Means to get a smaller number of clusters. The sigma parameter refers to the area of the circle of each node [20]. The optimal cluster was decided with the inertia (elbow) plot which measures how well the dataset is clustered for different number of clusters. Shown below in Figure 6 is a simplified schematic of the process.

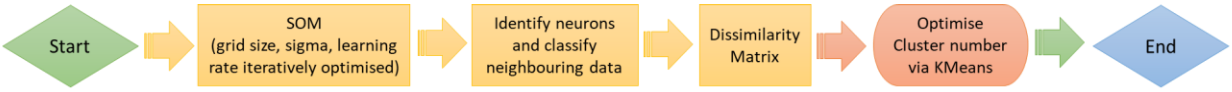


***Figure 6:** Process of SOM + K-Means*

(C) Density-Based Spatial Clustering of Application with Noise (DBSCAN)

DBSCAN is a type of density-based clustering which groups data points together based on a certain defined maximum distance between two points, known as epsilon, ε. Those that are sufficiently close together belong to a single cluster and those that aren't are treated as belonging to be separate cluster. The two parameters governing the separation of clusters are ε, defined as the maximum distance between points allowed to be considered as part of a cluster, and minimum samples (σ), defined as the minimum points that are required for a formation of a cluster. Both ε and σ are user defined, but while the choice of σ is somewhat arbitrary, ε is constrained using a knee plot (Figure 7). The maximum point of inflexion (or the knee) occurs when the average distance between a point and its σ nearest neighbors is the largest. In this work, we set $\sigma = 4$. In Figure 7, the maximum inflexion point is identified by the dashed line; therefore $\varepsilon = 0.0152$ is applied for the DBSCAN process, schematically illustrate in Figure 8 [21, 22].

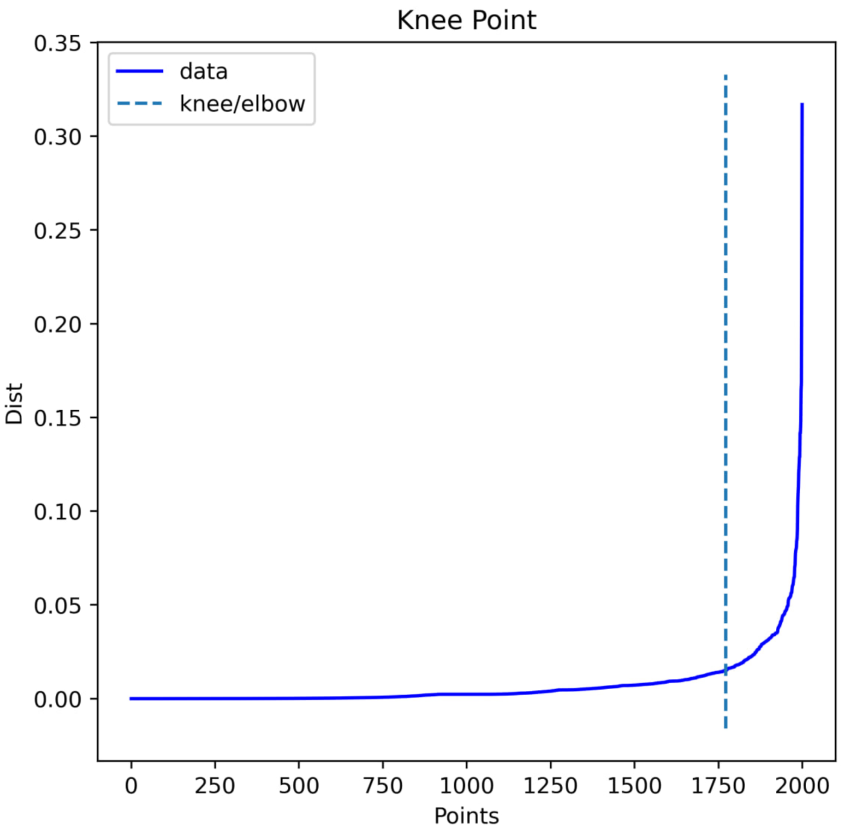


Figure 7: *Average distance of point to nearest 4 neighbors plot (ascending) with knee point*

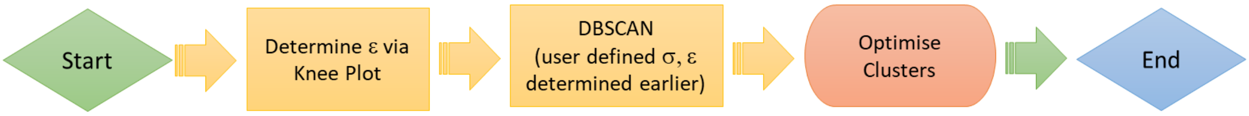

Start
Determine ε via Knee Plot
DBSCAN (user defined σ, ε determined earlier)
Optimise Clusters
End

Figure 8: *Process of DBSCAN*

(D) Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH)

BIRCH is a hierarchical clustering algorithm that builds an accurate summarized compact version of the original data for clustering. The algorithm builds a clustering feature (CF) tree where data points close in proximity forms a subcluster and outlier data points are removed. A CF tree is the compact representation of the data set, where each leaf node is a subcluster. The three parameters needed are threshold, branching factor, and n _cluster which are user-defined. The threshold represents the number of data point each leaf node can have, the branching factor represents how many sub-clusters per leaf node and the n_clusters represent how many clusters we require. The shortcoming of the algorithm is that it works only for metric attributes [23]. We schematically illustrate the process of K-means model training in Figure 9.

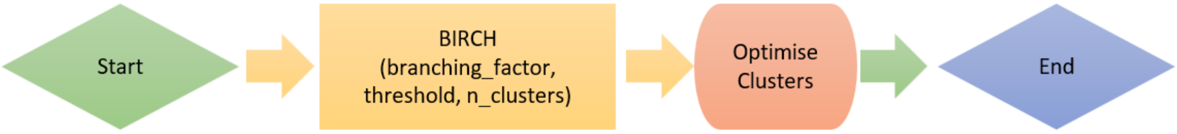


Figure 9: *Process of BIRCH*

(E) Gaussian Mixture Model (GMM)

GMMs is a probabilistic based clustering technique that clusters data into ellipsoidal shaped groups based on probabilistic methods, with each cluster distributed in a Gaussian normal distribution [24]. The parameter needed is the number of components. The number of components is determined using Bayesian Information Criterion (BIC). The shortcoming of the algorithm is the assumption of the dataset having a Gaussian distribution a priori, and that the clusters are ellipsoidal in shape. Figure 10 shows the BIC plot where like the Knee plot in Figure 7, the point of maximum inflection is of interest. We illustrate the process of Gaussian Mixture Models training in Figure 11.

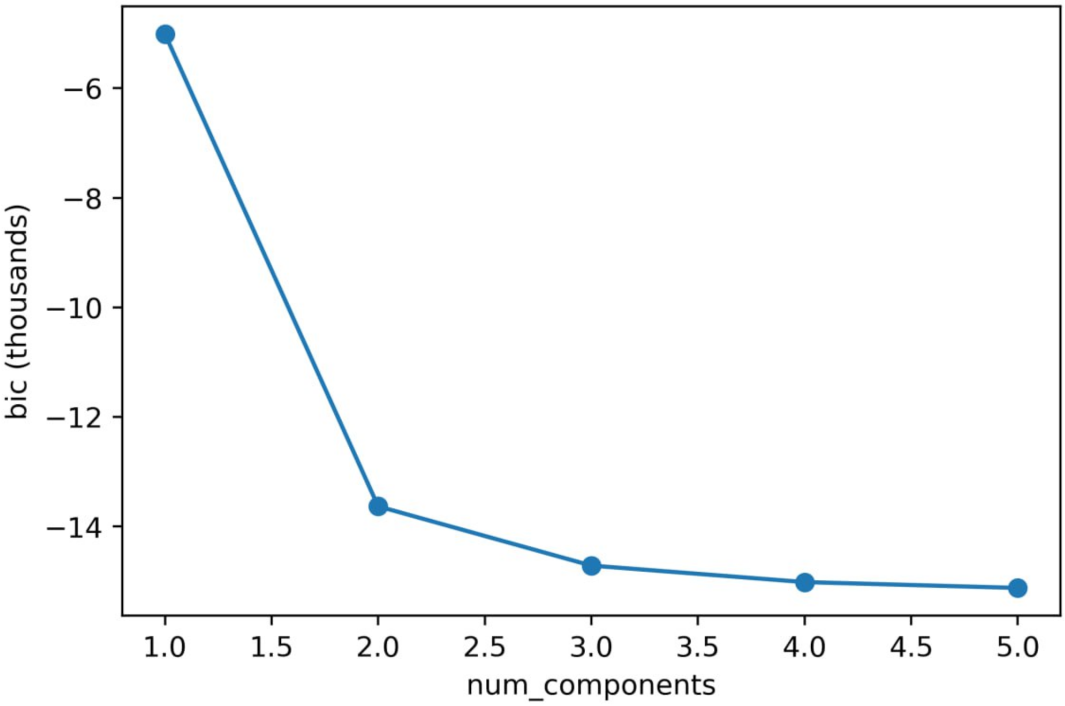


Figure 10: *BIC Plot for Number of Components Selection*

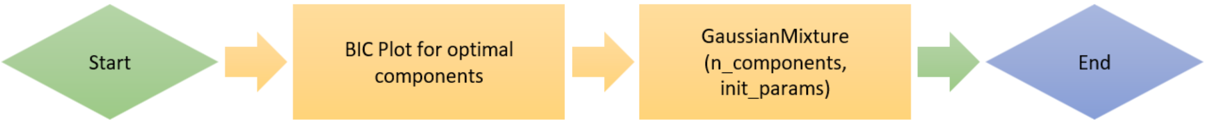

Start
BIC Plot for optimal components
GaussianMixture (n_components, init_params)
End

Figure 11: *Process of Gaussian Mixture Models*

Conventional Rock Typing using Iterative Multi-Linear Regression (IMLR)

We applied the IMLR technique as the benchmark "conventional" RT method based on a review of literature [8, 25, 26]. Figure 12 demonstrates how IMLR RTs are typically generated If manually done, it utilizes the gradient changes from a probability density plot of log FZI (Figure 12) to determine how best to segment the plot of Log $\emptyset_z$ vs Log RQI.

To perform our version of IMLR, we apply a ML method as well. We (i) first evaluate the FZI for the entire data set, (ii) sort in ascending order the data by FZI value, (iii) train, test and predict on the data a linear regression (LR) and record the value, (iv) split the sorted data into 2 subsets, (iv) train, test and predict on each subset of data a linear regression (LR) again, (v) evaluate an average r-squared (r^2) for all the subsets and (vi) repeating steps (iv) to (vi) with ever increasing subsets to a maximum value of 10. We then plot a graph of r^2 vs number of clusters, and select the optimal cluster based on the r^2 value where the rate of change of increase r^2 decreases. We then apply the optimized cluster number back to the entire data set and label the input data accordingly. Given in Figure 13 is the r^2 plot, while Figure 14 is a schematic of the IMLR method we applied. Finally, Figure 15 is the output of the classification process.

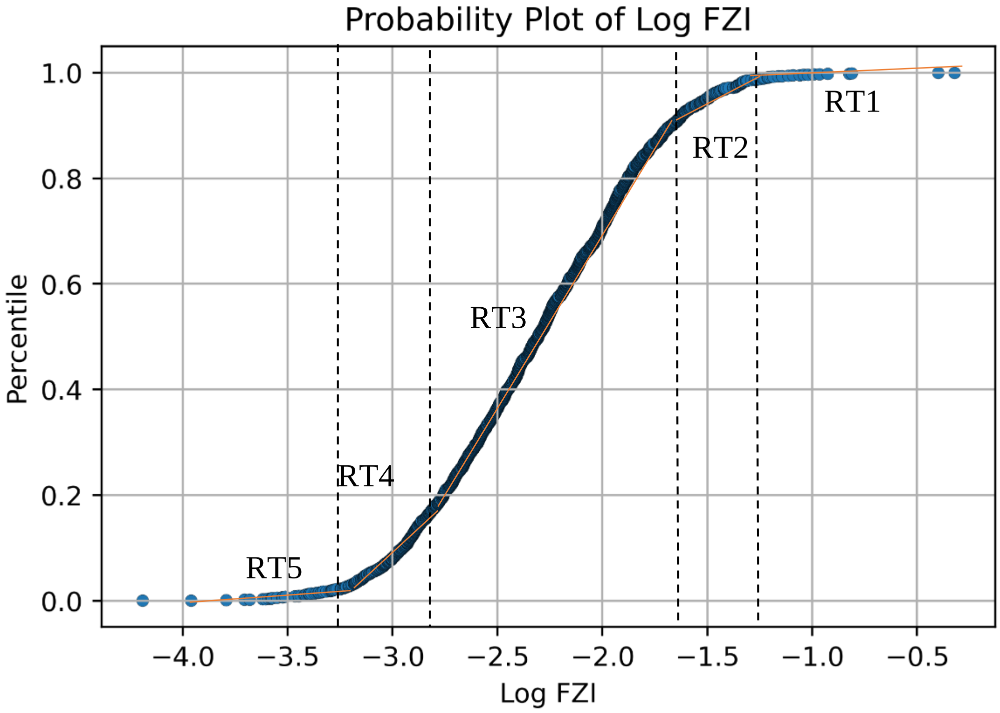


Figure 12: *(a) Graph depicting rock type class based on normal probability chart of log FZI*

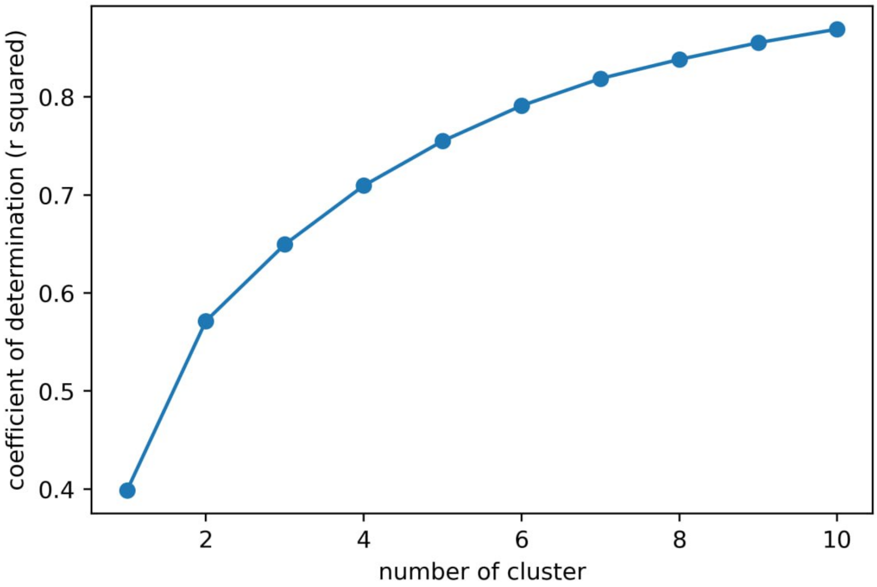


Figure 13: *r2 plot for number of cluster selection*

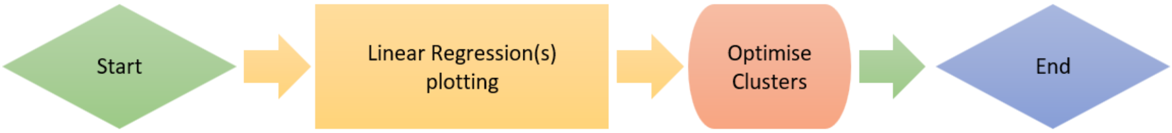


Figure 14*: Process of IMLR*

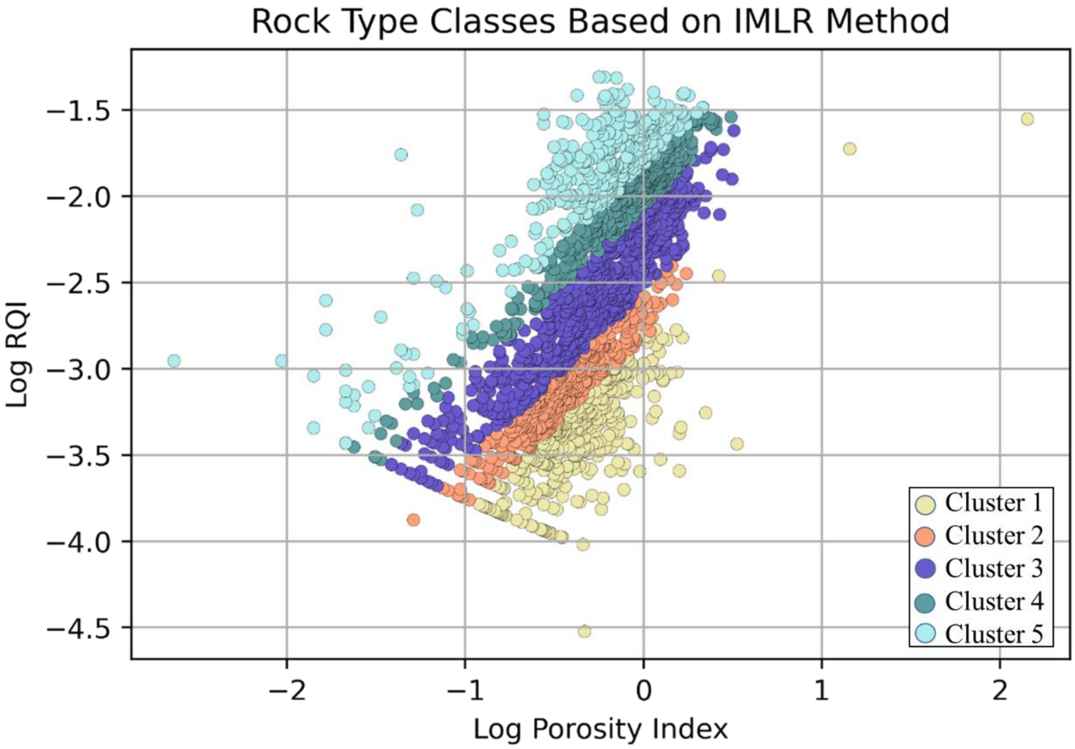


Figure 15: *The rock type classification viewed on a log RQI and log porosity index chart for IMLR*

Results

1. Comparing the Number of Hydraulic Rock Types generated.

We first compare the results of the number of unique RT generated by the ML models versus the IMLR model. Shown in Table 3 is the overview of the number of clusters for each method and the metrics used to determine optimal cluster size. Shown in Table 4 is the breakdown of the cluster distribution of the 6 methods. We consistently observe that, across the 5 unsupervised methods, 35% of the data set is labelled as cluster 1, while 2% is labelled as cluster 3. There is no commonality in points labelled as cluster 2, 4 or 5.

Shown in Figure 16 is the distribution plot of the various outputs. Right away, we note that the outputs from the various algorithms' plots look dissimilar. We noted that while the conventional RT method has 5 unique RTs, none of the other ML methods gave more than 4 unique clusters.

Table 3 shows the number of cluster and metrics used to determine optimal cluster for each method and Table 4 shows the cluster distribution breakdown for each method.

Table 3: *Number of clusters and metrics for cluster sections per method*

Method	IMLR	K-Means	SOM + K-Means	DBSCAN	BIRCH	GMM
Number of Clusters	6	3	3	2	4	3
Cluster Determined by	r2	Inertia Plot	Inertia Plot	Silhouette Score	Calinski Harabasz Score	BIC

Table 4: *Cluster distribution breakdown per method*

	Cluster 1		Cluster 2		Cluster 3		Cluster 4		Cluster 5	
Method	Num	%	Num	%	Num	%	Num	%	Num	%
IMLR	335	17	333	17	666	33.3	333	17	333	17
K-Means	1010	51	881	44	109	5				
SOM + K-Means	737	37	792	40	471	24				
DBSCAN	1439	72	561	28						
BIRCH	1052	53	696	35	184	9	68	3		
GMM	1203	60	575	29	222	11				

2. Comparing Metrics between the various Unsupervised Machine Learning Algorithms

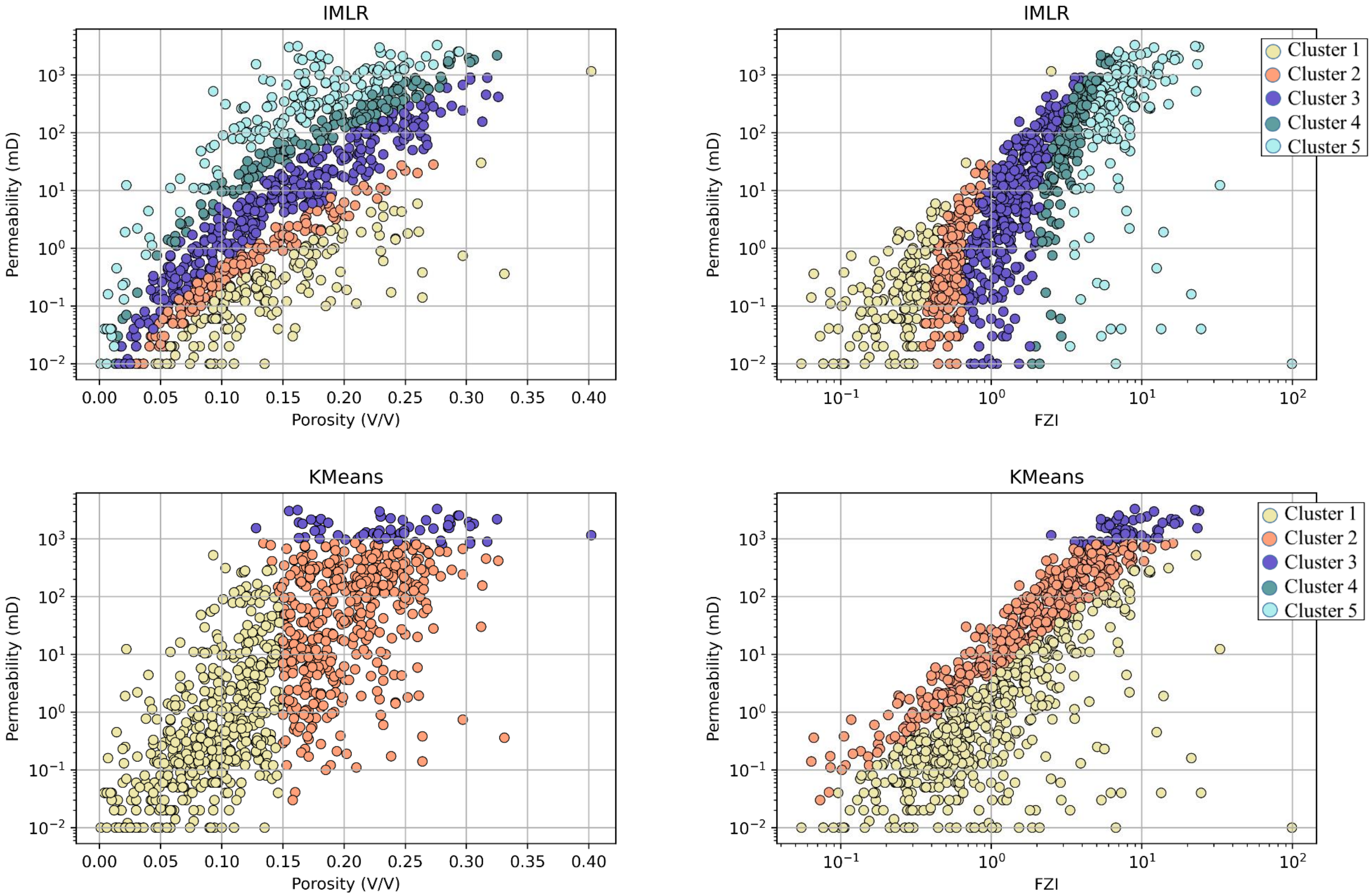
We compare the ML results using the Silhouette Coefficient (SC), Calinski-Harabaz Index (CHI), and the Davis-Bouldin Index (DBI). SC, CHI, and DBI are three commonly used metrics for evaluating the quality of clustering results. Each of these metrics measures different aspects of the clustering performance.

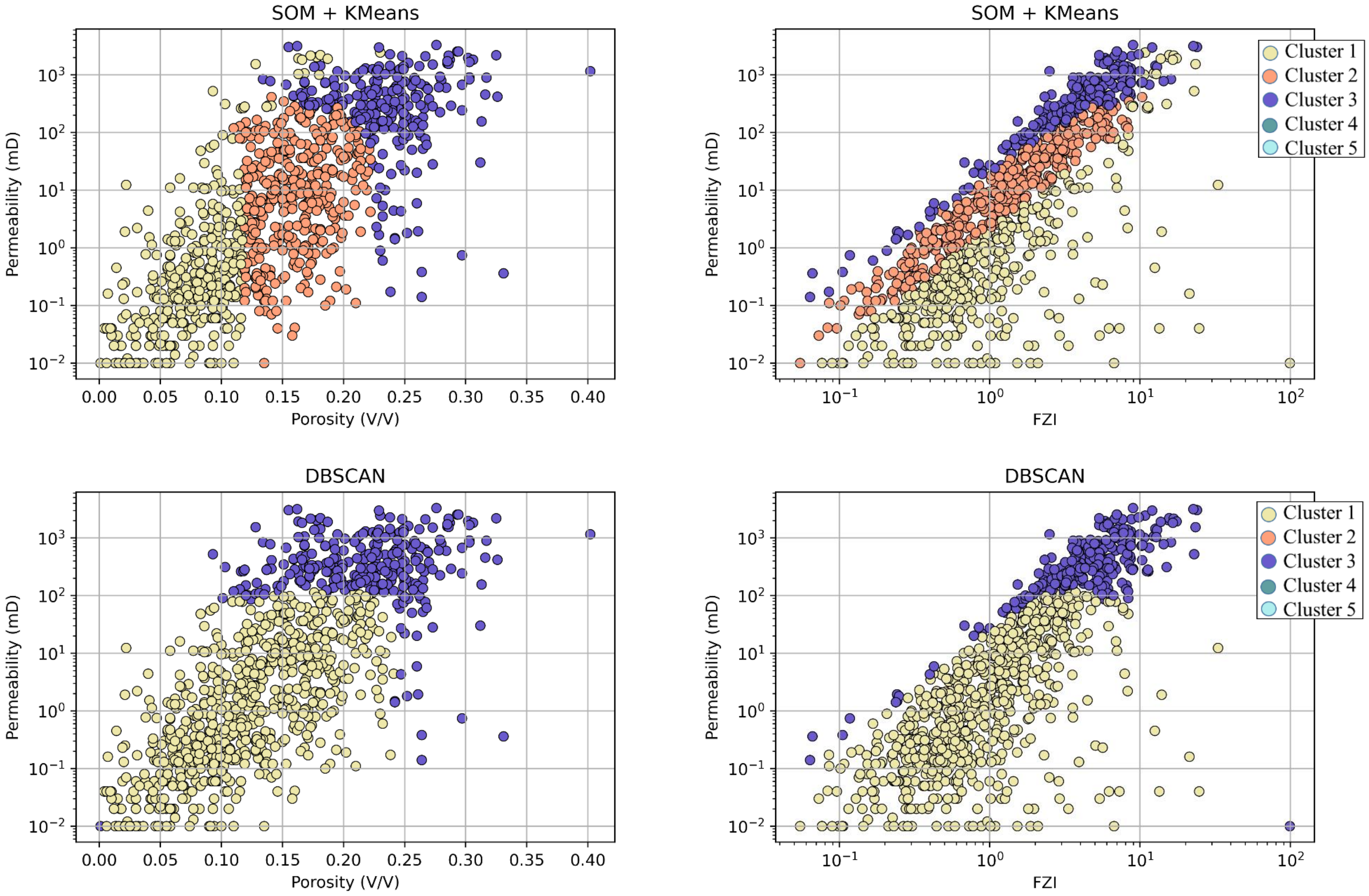
SC measures how well each data point fits into its assigned cluster, compared to other clusters. It ranges from -1 to +1, where a value of +1 indicates that the data point is well matched to its assigned cluster and poorly matched to neighboring clusters, while a value of -1 indicates the opposite. A value of 0 indicates that the data point is equally matched to its assigned cluster and a neighboring cluster. CHI measures the ratio of the between-cluster variance to the within-cluster variance. A higher CHI value indicates that the clusters are well separated and compact. This index is particularly useful when the ground truth labels are not available. Finally, DBI measures the average similarity between each cluster and its most similar cluster, compared to the distance between the cluster centers. A lower DBI value indicates better separation between the clusters. In general, a higher SC value, CHI value, or lower DBI value indicates better clustering performance.

From the results in Table 5, we note that the best performing clustering algorithm is K-Means and BIRCH. The SC range of values for K-Means and BIRCH is 0.53 and 0.47 respectively, and is higher than any of the other models, all of which have values less than 0.47. All values are greater than 0 however and positive, indicating that the data points are properly labelled and appropriately assigned to its cluster. The CHI value is equally high, at >2000, while the remaining 3 methods are all significantly lower, with GMM having the lowest score. This means that K-Means and BIRCH have well separated, compact clusters. The DBI value for both methods is also low, although the magnitude of separation of values for SOM+K-Means and BIRCH is negligible, at just 0.02. For the DBI score, DBSCAN and GMM have the highest values.

Table 5: *Summary of the 5 Clustering Models' Metrics*

Algorithm	K-Means	SOM+K-Means	DBSCAN	BIRCH	GMM
Silhouette Coefficient (SC)	0.529	0.412	0.404	0.474	0.263
Calinski-Harabaz Index (CHI)	2560.0	1510.0	1180.0	2170.0	1090.0
Davis-Bouldin Index (DBI)	0.642	0.859	1.040	0.839	1.030


IMLR
Permeability (mD)
10^3
10^2
10^1
10^0
10^{-1}
10^{-2}
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
Porosity (V/V)
IMLR
FZI
10^{-1}
10^0
10^1
10^2
Cluster 1
Cluster 2
Cluster 3
Cluster 4
Cluster 5
KMeans
Permeability (mD)
Porosity (V/V)
KMeans
FZI
Cluster 1
Cluster 2
Cluster 3
Cluster 4
Cluster 5


SOM + KMeans
Permeability (mD)
10^{-2}
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
Porosity (V/V)
SOM + KMeans
Permeability (mD)
FZI
10^{-1}
10^{0}
10^{1}
10^{2}
Cluster 1
Cluster 2
Cluster 3
Cluster 4
Cluster 5
DBSCAN
Permeability (mD)
Porosity (V/V)
DBSCAN
Permeability (mD)
FZI
Cluster 1
Cluster 2
Cluster 3
Cluster 4
Cluster 5

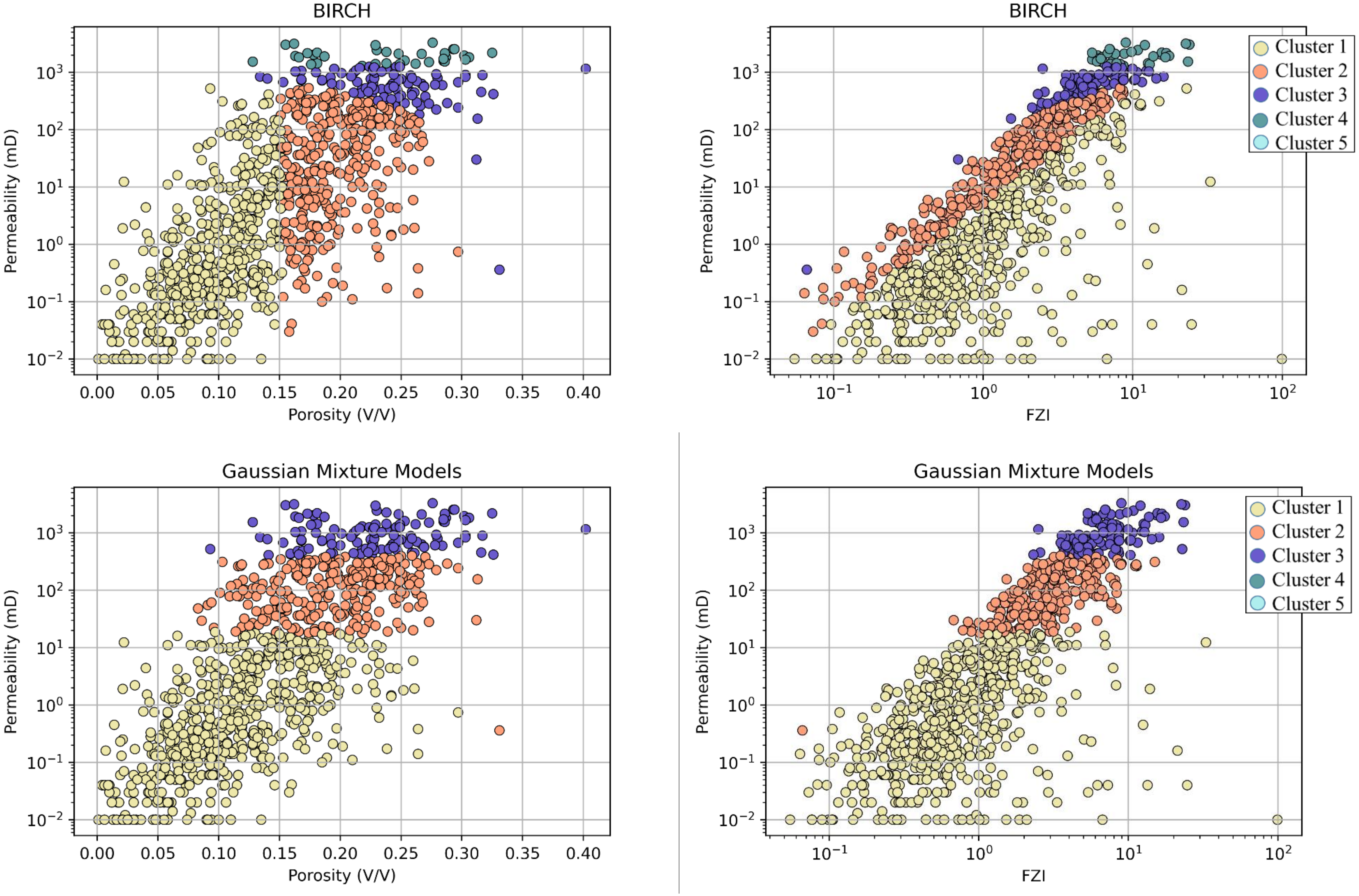


Figure 16: (a) Porosity vs Log Scaled Permeability Scatterplot Cluster Distributions; (b) Log Scaled Porosity vs Log Scaled Permeability Scatterplot Cluster Distribution

3. Permeability Prediction and comparing Residuals between Model Outcomes

Aside from classification of RT for lithology and distribution in static models, one of the main uses of RT is for the prediction of permeability. For this work, we created a blind data set comprising of 6 porosity values with values ranging from 0.05 V/V to 0.3 V/V, at 0.05 V/V increments. Using the earlier modelled clusters, we next fit linear regressions through each one and evaluated the corresponding permeability value.

Shown in Table 6 is the table comparing the residuals for all the different RT methods. Residual here refers to the absolute difference between the predicted permeability (P) and the core permeability (K). Overall, the results predicted by DBSCAN and GMM have the least residual, compared to the other ML methods. K-Means and SOM+K-Means are consistently the worse performing models, despite the latter having the best performing clustering metric.

***Table 6**: Residuals comparison between rock typing methods*

Φ	Core K	IMLR		K-Means		SOM+K-Means		DBSCAN		BIRCH		GMM	
(V/V)	(mD)	P	R	P	R	P	R	P	R	P	R	P	R
0.05	0.1	0.1	0.05	2.5	2.37	54.6	54.5	2.1	2.0	2.0	187.0	0.6	0.5
0.10	1.0	2.3	1.3	17.4	16.4	125.0	124.0	6.8	5.8	18.4	17.4	2.1	1.1
0.15	10.0	11.8	1.8	32.3	22.3	30.7	20.7	15.7	5.7	34.9	24.9	3.5	6.5
0.20	31.6	48.4	16.8	152.0	120.0	48.1	16.5	24.6	7.0	89.6	58.0	142.0	110.0
0.25	100.0	145.0	45.0	230.0	130.0	591.0	491.0	644.0	544.0	590.0	490.0	167.0	67.0
0.30	316.2	866.0	550.0	1622.0	1310.0	721.0	405.0	779.0	463.0	469.0	153.0	1310.0	994.0

Note: P refers to predicted while R refers to residual

We observe that the benchmark IMLR method has predicted permeability close to the experimental measured core data. K-Means appears to overestimate permeability, with permeabilities being overestimated by a factor of 2-3. The results of SOM+K-Means are wildly inconsistent, where in some instances, low porosity gives very high permeability and vice-versa. DBSCAN gives better results and is within expected ranges, although it does tend to be somewhat inconsistent at higher porosities (0.25 V/V and above). BIRCH shows a similar trend of results to DBSCAN but with larger residuals in almost all instances. Finally, GMM shows reasonable results, being close to experimental core data. Interestingly, excluding the results of K-Means and SOM+K-Means, most models give outputs of permeability that are within an order of magnitude of the core permeability, with very low porosities of 0.05 V/V being the hardest to predict.

Discussion – the risk of overfitting in unsupervised ML

From the analysis, we find that the results of the IMLR, DBSCAN and GMM are very similar to one another, yet the latter 2 ML methods have the lowest SC and CHI scores, as well as the highest DBI value of the 5 methods trialed. We find this result surprising, especially when we noted earlier that a higher SC value, CHI value, or lower DBI value indicate better clustering performance.

We think that algorithms that “performed better” were in fact suffering from overfitting. Overfitting in clustering occurs when a clustering model is trained to fit the specific characteristics of the training data too closely, at the expense of generalizing well to new, unseen data. Overfitting can occur when the clustering model is too complex, and it can result in poor performance when the model is applied to new data. Given that our data set is extremely dense, this is a real possibility.

One of the key determinants in unsupervised ML is the number of clusters to select. Unfortunately, this is not an easy task, with our clusters ranging from 2-4 depending on the model selected. In a ML solution that has too many clusters, the model may start by partitioning the data into too many subgroups. This can result in a model that fits the training data very well but does not generalize well to new data. Additionally, the choice of distance metric in clustering can also lead to overfitting. If the distance metric is too sensitive to noise or outliers in the data, the model may be overfitting the training data by creating clusters that are too specific to the noise in the data.

Hence, when one considers the overall results, we note that the poorer performing models, being more “general”, are able to predict permeability better.

Limitations of Study and Conclusion

In this study, we performed rock typing using our implementation of the IMLR method as well as 5 other unsupervised machine learning algorithms, namely K-Means, SOM+K-Means, DBSCAN, BIRCH, and GMM. A scaled-down core dataset with porosity and permeability was used for the analysis, consists of 2000 samples from the UK North Sea.

When comparing the cluster distributions in the porosity vs log-scaled permeability scatterplots and the cluster averages of the 6 models, we observe that none of the 5 models' cluster distributions and cluster averages resemble the IMLR results. We do note however that both K-Means and BIRCH cluster distribution and averages resemble one another closely. We also note that while K-Means and BIRCH have the best-performing metrics (SC, CHI and DBI), it is in fact DBSCAN and GMM that give the best result, permeability prediction wise.

Overall, we have demonstrated that unsupervised ML algorithms can perform permeability prediction to a reasonable level of accuracy but cannot be solely relied on without the intervention of subject matter experts (SME), who should firstly QC the data, making sure it is fit for purpose. SMEs should also be relied on to determine the efficacy and robustness of the results, as was demonstrated here, where, if purely based on ML metrics, the K-Means and BIRCH model would have been the default result, yet they both did not come close to predicting the “true” values as measured on the core data.

Our study has only used a limited dataset. Future work could be the inclusion of more data, use of more features (volume of shale, grain density etc), the use of other algorithms and/or benchmarking to other RT methods.

Bibliography

[1] G. Archie, “Introduction to Petrophysics of Reservoir Rocks,” *AAPG Bulletin,* vol. 34, pp. 943-961, 1950.

[2] J. A. Rushing, “Rock Typing — Keys to Understanding Productivity in Tight Gas Sands,” *Society of Petroleum Engineers (SPE) Paper Number 114164,* pp. 1-31, 2008.

[3] C. Hollis, V. Vahrenkamp, C. Tull, A. MookerJee, C. Taberner and Y. Huang, “Pore system characterisation in heterogeneous carbonates: An alternative approach to widely-used rock-typing methodologies,” *Marine and Petroleum Geology,* vol. 27, no. 4, pp. 772-793, 2010.

[4] P. W. M. Corbett and D. K. Potter, “Petrotyping: A basemap and atlas for navigating through permeability and porosity data for reservoir comparison and permeability prediction.,” *Paper SCA2004-30 presented at the International Symposium of the Society of Core Analysts,* vol. 5, no. 9, pp. 1-12, 2004.

[5] X. Tang, W. Y. Zhou, W. L. Yang, C. Zhang and C. M. Zhang, “An improved method in petrophysical rock typing based on mercury-injection capillary pressure data.,” *Energy Sources, Part A: Recovery, Utilization, and Environmental Effects,* pp. 1-16, 2020.

[6] W. McCabe, J. C. Smith and P. Harriot, in *Unit operations of chemical engineering,* McGraw-Hill, New York, 2005.

[7] J. O. Amaefule, A. Mehmet, T. Djebbar, D. G. Kersey and K. K. Dare, “Enhanced Reservoir Description: Using Core and Log Data to Identify Hydraulic (Flow) Units and Predict Permeability in Uncored Intervals/Wells,” *SPE Annual Technical Conference and Exhibition,* pp. 1-12, October 1993.

[8] M. Khalid, S. E. D. Desouky, M. Rashed, T. Shazly and K. Sediek, “Application of hydraulic flow units’ approach for improving reservoir characterization and predicting permeability.,” *Journal of Petroleum Exploration and Production Technology,* vol. 10, no. 2, pp. 467-479, 2020.

[9] H. Abdulelah, S. Mahmood and G. Hamada, “Hydraulic flow units for reservoir characterization: A successful application on arab-d carbonate,” in *Materials Science and Engineering 380 (2018) 012020,* 2018.

[10] E. Mohammadian, M. Kheirollahi, M. Ostadhassan and M. Sabet, “A case study of petrophysical rock typing and permeability prediction using machine learning in a heterogenous carbonate reservoir in Iran,” *Nature Portfolio,* vol. 12, no. 4505, pp. 1-16, 2022.

[11] I. M. Mohamed, S. Mohamed, I. Mazher and C. Pieprzica , “Formation Lithology Classification: Insights into Machine Learning Methods,” *SPE Annual Technical Conference and Exhibition,* pp. 1-12, 2019.

[12] H. Zakyan, A. K. Permadi and E. A. Pratama, “An Improved Method of Clay-Induced Rock Typing Derived from Log Data in Modelling Low Salinity Water Injection: A Case Study on an Oil Field in Indonesia,” *Energies,* vol. 15, no. 10, pp. 1-12, 2022.

[13] I. Juhasz, “Normalised Qv - The Key To Shaly Sand Evaluation Using The Waxman-Smits Equation In The Absence Of Core Data,” *OnePetro,* 1981.

[14] GeoProvider AS, [Online]. Available: https://www.geoprovider.no/.

[15] UK National Data Repository, [Online]. Available: https://ndr.nstauthority.co.uk/.

[16] software underground, [Online]. Available: https://dataunderground.org/.

[17] M. Kumar, K. Swaminathan, A. Rusli and A. Thomas-Hy, “Applying Data Analytics & Machine Learning Methods for Recovery Factor Prediction and Uncertainty Modelling,” *Society of Petroleum Engineers,* pp. 1-18, 2022.

[18] P. Franti and S. Sieranoja, “How much can k-means be improved by using better initialization and repeats?,” *Pattern Recognition,* vol. 93, pp. 95-112, 2019.

[19] B. Brentan, G. Meirelles, E. Luvizotto Jr. and J. Izquierdo, “Hybrid SOMþk-Means clustering to improve planning, operation and management in water distribution systems,” *Environmental Modelling & Software,* pp. 1-12, 2018.

[20] G. Vettigli, “MiniSom: minimalistic and NumPy-based implementation of the Self Organizing Map,” 2018. [Online]. Available: https://github.com/JustGlowing/minisom/.

[21] A. Ram, S. Jalal, A. S. Jalal and M. Kumar, “A Density based Algorithm for Discovering Density Varied Clusters in Large Spatial Databases,” *International Journal of Computer Applications,* 2010.

[22] A. Starczewski, P. Goetzen and J. E. Meng, “A New Method for Automatic Determining of the DBSCAN Parameters,” *Journal of Artificial Intelligence and Soft Computing Research,* pp. 209-221, 23 May 2020.

[23] T. Zhang, R. Ramakrishnan and M. Livny, "BIRCH: A New Data Clustering Algorithm and Its Applications," *Data Mining and Knowledge Discovery,* no. 1, pp. 141-182, 1997.

[24] E. Patel and D. S. Kushwaha, "Clustering Cloud Workloads: K-Means vs Gaussian Mixture Model," *Procedia Computer Science,* vol. 171, pp. 158-167, 2020.

[25] F. A. Al-Ajmi and S. A. Holditch, "Permeability estimation using hydraulic flow units in a central Arabia reservoir," *SPE Annual Technical Conference and Exhibition. OnePetro.,* pp. 1-12, 2000.

[26] S. Desouky, "A New Method for Normalization of Capillary Pressure Curves," *Oil & Gas Science and Technology – Rev. IFP,* vol. 58, no. 5, pp. 1-6, 2003.

[27] A. Mirzaei-Paiaman, M. Ostadhassan, R. Rezaee, H. Saboorian-Jooybari and Z. Chen, "A new approach in petrophysical rock," *Journal of Petroleum Science and Engineering,* no. 166, pp. 445-464, 2018.

[28] H. A. Nooruddin and M. E. Hossain, "Modified Kozeny–Carmen correlation for enhanced hydraulic flow unit characterization," *Journal of Petroleum Science and Engineering,* vol. 80, no. 1, pp. 107-115, 2011.

[29] M. Izadi and A. Ghalambor, "A New Approach in Permeability and Hydraulic-Flow-Unit Determination," *SPE Res Eval & Eng,* vol. 16, no. 03, pp. 257-264, 2013.

[30] J. ,. S. Shahat, M. I. Balaha, M. S. El-Deab and A. M. Attia, "surface grain volume," *Energy Reports,* vol. 7, pp. 711-723, 2021.

[31] S. Kolodzie, " Analysis of pore throat size and use of the Waxman–Smits equation to determine OOIP in Spindle Field, Colorado," 1980.

Nomenclature

ANN	Artificial Neural Network
BIRCH	Balanced Iterative Reducing and Clustering using Hierarchies
BIC	Bayesian Information Criteria
CHI	Calinski-Harabaz Index
m	Cementation Factor
CF	Clustering Feature
DBI	Davis-Bouldin Index
DBSCAN	Density-Based Spectral Clustering of Application with Noise
ε	Epsilon
XGB	Extreme Gradient Boosting
FZI	Flow Zone Index
R_0	Formation Resistivity when water saturation is at 100%
R_W	Formation Water Resistivity
FZI*	Modified Flow Zone Index
GMM	Gaussian Mixture Models
HFU	Hydraulic Flow Units
S_{wir}	Irreducible Water Saturation
IMLR	Iterative multi-linear Regression
KNN	K-Nearest Neighbors

ML	Machine Learning
Rmh	Mean Hydrauliuc Radius of Pore Throats
MICP	Mercury Injection Capilary Pressure
FZIM	Modified Flow Zone Index
FZIM*	Modified Flow Zone Index
MFZI	Modified Flow Zone Index
$\emptyset_z$	Normalised Porosity Index
k	Permeability
r35	Pore Throat Radius (35% Mecury Saturation)
$\emptyset$	Porosity
RQI	Reservoir Quality Index
I_r	Resistivity Index
RZI	Resistivity Zone Index
RT	Rock Typing
r^2	R-squared
SOM	Self-Organising Map
F_s	Shape Factor
SC	Silhouette Coefficient
SVM	Support Vector Machine
S_{gv}	Surface Grain Volume
R_T	True Formation Resistivity

Acknowledgement: We wish to acknowledge Ryan Larazoo for review of the work.

Appendix 1 – Rock Typing Indices Equations

Number	Indices	Equations	Authors
(1)	FZI	$FZI = \frac{RQI}{\emptyset_z}$ $RQI = 0.0314 \times \sqrt{\frac{k}{\emptyset}}$ $\emptyset_z = \frac{\emptyset}{1-\emptyset}$ Where: k is permeability (mD) $\emptyset$ is porosity (V/V) $\emptyset_z$ is normalized porosity (V/V) RQI is Rock Quality Index (mm) FZI is Flow Zone Index (mm)	Amaefule et. al. [7]
(2)	FZI*	$FZI^* = 0.0314 \times \sqrt{\frac{k}{\emptyset}}$ Where: k is permeability (mD) $\emptyset$ is porosity (V/V)	Mirzaei-Paiaman et. al. [27]
(3)	FZIM	$FZIM = \frac{RQI}{\emptyset^{m-1}\emptyset_z}$ Where: m is cementation factor (dimensionless)	Nooruddin and Hossain [28]
(4)	MFZI	$MFZI = \frac{RQI}{\emptyset_z(1-S_{wir})^{1.5}}$ Where: RQI is Rock Quality Index (mm) $\emptyset_z$ is normalized porosity (V/V) S_{wir} is irreducible water saturation (fraction)	Izadi and Ghalambor [29]
(5)	RZI	$RZI = \frac{I_r * R_w}{k * F_s * {S_{gv}}^2}$ $I_r = \frac{R_T}{R_0}$ Where: I_r is resistivity index (fraction) R_T is true formation resistivity (ohm-m) R_0 is formation resistivity when water saturation is 100% (ohm-m) R_W is formation water resistivity (ohm-m) S_{gv} is surface grain volume (µm^{-1}) k is permeability (mD) F_s is shape factor(dimensionless)	Shahat et. al. [30]
(6)	Winland	$logr35 = 0 - 0.996 + 0.588 \log k - 0.864 \log \phi$ Where: k is uncorrected air permeability (mD) ϕ is porosity (V/V) r35 is pore throat radius with 35% mercury saturation (mm)	Kolodzie [31]