Improved well log classification using semi-supervised Gaussian mixture models and a new model selection strategy

Michael W. Dunham[a,*], Alison Malcolm[a], J. Kim Welford[a]

[a]*Department of Earth Sciences,*
Memorial University of Newfoundland
St. John's, NL, A1B 3X5 Canada

Abstract

Well log classification, the process of mapping well log measurements to lithofacies identified from core samples, is a common procedure in the oil and gas industry. Manually assigning lithofacies to the wireline log measurements without core can be time consuming, and can also introduce a bias. Supervised machine learning algorithms are commonly used to automate this process, but they are prone to overfitting when the training data are scarce, which is common for well log classification problems. Semi-supervised machine learning algorithms are designed for classification problems with minimal training data, and we adopt a semi-supervised Gaussian mixture model (ssGMM) method to solve this problem. The dataset we consider for our study is from a machine learning competition held in 2016 and we simulate a semi-supervised scenario by assuming only one out of the ten wells is the labeled data. We apply ssGMM to this well log dataset and compare its performance to the supervised method that was the winner of this competition, XGBoost. We also introduce a new model selection strategy that simultaneously uses the mean and standard deviation cross-validation scores, compared to the default procedure that only utilizes the mean cross-validation scores. Our results indicate that ssGMM is able to outperform XGBoost in our semi-supervised context, which supports the suggestion that semi-supervised algorithms are more appropriate in low training data situations. We also show that our new model selection technique selects models for ssGMM that perform better on the testing data, but the performance is mixed for XGBoost.

Keywords: lithofacies, well logs, semi-supervised, classification, model selection

*Corresponding author
Email addresses: mwdunham@mun.ca (Michael W. Dunham), amalcolm@mun.ca (Alison Malcolm), kwelford@mun.ca (J. Kim Welford)

1. Introduction

In recent years, there has been an increasing demand for data science across many disciplines for two reasons. One, technological advances have improved data storage capabilities as well as how data can be obtained (e.g., real-time data). Manually interpreting data that are exponentially growing in volume has obvious management and analysis challenges. Machine learning automates pattern recognition of big data in an efficient manner. The second motivation for data science is derived from a shortcoming in inversion. In many scientific problems, we are trying to understand the model that generates the data we observe. For inversion, this relationship between data and model is conventionally expressed as $d = \mathbf{G}m$ where d is the observed dataset, m is the model, and $\mathbf{G}$ is the forward operator that contains the *explicit* physics or mathematics needed to relate the observed data to the model. Simplistically, the model can be described by taking the inverse of the forward operator, $m = \mathbf{G}^{-1}d$. Performing this operation is not trivial if $\mathbf{G}$ is complicated, or non-linear, or perhaps it is not even possible if $\mathbf{G}$ does not exist for the problem of interest. Machine learning algorithms learn an *implicit* mapping to go from d to m without the need for *explicit* relationships to be programmed, like inversion.

One scenario where an explicit mathematical relationship does not exist is well log classification. The objective for well log classification is to map well log measurements (e.g., gamma ray, density, neutron porosity, resistivity, sonic, etc.) to lithofacies identified from core samples, which cannot be achieved using standard inversion techniques. Consequently, this problem has been addressed using supervised learning (SL) techniques in recent decades. Neural networks were the first machine learning algorithms applied to well log classification (Baldwin et al., 1990; McCormack, 1991; Rogers et al., 1992), and they continued to be popular for the next two decades (Benaouda et al., 1999; Saggaf and Nebrija, 2000; Maiti et al., 2007; Al-Bulushi et al., 2009; Malvić et al., 2010; Al-Bulushi et al., 2012). Other implementations included fuzzy logic (Saggaf and Nebrija, 2003; Dubois et al., 2007) and k-nearest neighbors (Dubois et al., 2007), but in recent years there have been support vector machine (Wang et al., 2013; Hall, 2016) and ensemble method implementations (Bestagini et al., 2017; Keynejad et al., 2019).

A challenge with well log classification is the availability of labeled data. Generally, it is relatively cheap to collect unlabeled data, but obtaining labels for the unlabeled data can be difficult because this commonly requires human annotators or expensive experiments (Zhu and Goldberg, 2009). For the well log classification problem, it is relatively inexpensive to collect wire-line log data, but extracting, preserving, and storing core samples can be costly. Therefore, these classification problems commonly have a paucity of labeled data. For instance, Dubois et al. (2006) train a neural network using 14 wells with core and they use that mapping to predict the lithofacies for 1364 different wells. In this situation, the supervised classifier is trained using roughly 1% of the data and this assumes that the classifier can generalize in classifying the remaining 99%. This may be a poor assumption, in general, because it is known that training a supervised

classifier on a small training set can lead to overfitting, and complex models such as deep neural networks are even more susceptible to overfitting in these situations (Krizhevsky et al., 2012; Waldeland et al., 2018). This is analogous to underdetermined inverse problems where the number of model parameters is greater than the number of data.

Since supervised algorithms are prone to overfitting when the training dataset is small, perhaps an alternative approach would be more effective in these situations. One solution is to use *semi-supervised learning* (SSL) techniques because they utilize both the labeled *and* unlabeled data in the training process, which is predicted to achieve more accurate labels for the unlabeled data than SL techniques in the context of limited training data (Chapelle et al., 2006; Zhu and Goldberg, 2009). Returning to the inversion analogy, when an inverse problem is underdetermined, a common solution is to stabilize the objective function by adding an additional term that involves the model parameters, otherwise known as regularization (Aster et al., 2005). Regularization smooths the objective function and this prevents the predicted data from overfitting the observed data. Similarly, SSL can be thought of as SL where a regularization term including the unlabeled data is added to the objective function (Zhu and Goldberg, 2009).

Semi-supervised applications to well log classification are relatively unexplored with a limited number of publications in the literature. The only publication to our knowledge is the label propagation (LP) method coupled with self-training given by Dunham et al. (accepted). In their paper, they show that their self-training process of incrementally adding unlabeled points with the highest LP confidence to the labeled dataset can be effective, but there are some disadvantages to this technique. As they mention, a disadvantage of LP is that it is a transductive method, but their self-training process also removes the LP confidence of the unlabeled points that are added to the labeled data (i.e. by adding points to the labeled data, one assumes the predicted labels of these points are correct and probabilities of unity are assigned). The method we utilize in this paper is semi-supervised Gaussian mixture models (ssGMM). This method has similar benefits to the self-training LP technique of Dunham et al. (accepted) of being well established and easy to implement, but the unique benefit of ssGMM is that it is an inductive algorithm and the probabilities of the predicted labels for the unlabeled data are maintained. We also introduce a new model selection strategy that selects a machine learning model based on simultaneously using the mean and standard deviation scores coming from cross-validation rather than the default procedure that only relies on the mean cross-validation scores.

We apply these ideas to a lithofacies classification problem where the well log data we use are publicly available through an open competition held in 2016 (Hall, 2016; Hall and Hall, 2017). To assess the performance of the ssGMM method, we compare it to two supervised methods, and the self-training LP method from Dunham et al. (accepted). Furthermore, to evaluate the efficacy of our new model selection strategy, we show how algorithm performance (both SL and SSL methods) can vary using our new strategy versus the default approach. This paper is the first to our knowledge that uses a semi-supervised Gaussian mixture models method in a well log classification context, and we hope our new model selection strategy

will convince readers to re-evaluate how machine learning models are chosen during training.

2. Methodology

Machine learning algorithms that possess the capability of assigning classes to unlabeled data fall into two categories: supervised and semi-supervised. The objective of supervised learning (SL) is to learn a mapping, otherwise known as a classifier f, from instances and their associated classes using the training data,

$$L = (x_1, y_1), ..., (x_l, y_l). \tag{1}$$

where l is the number of labeled data. Any given SL algorithm uses the training data (L) to learn a mapping that is corrected by the error between the predicted and true labels. Ultimately, the mapping learned from this process is used to make predictions for data where the labels are unknown, i.e. the unlabeled data

$$U = x_{l+1}, ..., x_{l+u}, \tag{2}$$

where the actual predictions are given by,

$$H = y_{l+1}, ..., y_{l+u}, \tag{3}$$

and u represents the number of unlabeled data. SSL algorithms are similar to SL algorithms, but the unlabeled data (U) are incorporated into the training process. As a result, SSL algorithms train with

$$D = \{(x_1, y_1), ..., (x_l, y_l), x_{l+1}, ..., x_{l+u}\} = L \cup U \tag{4}$$

where D is the union of L and U and the objective is to still make predictions for the unlabeled data (H). From a theoretical perspective, including the unlabeled data in the training process can help SSL algorithms achieve more improved/generalized predictions for U than SL algorithms, in the context of low training data.

2.1. Semi-supervised Gaussian mixture models (ssGMM)

Many semi-supervised techniques are simply extensions of existing unsupervised or supervised methods to include additional information. For instance, semi-supervised Gaussian mixture models (ssGMM) is an algorithm that essentially combines a Naïve Bayes classifier (supervised) and Gaussian mixture models (unsupervised). While there are a few different implementations of ssGMM (Nigam et al., 2000; Zhu and

Goldberg, 2009; Xing et al., 2013), our implementation is based on the approach outlined in Yan et al. (2017). The algorithm is summarized below.

In this semi-supervised framework, the training data consist of both labeled and unlabeled data (D) and the goal for ssGMM is to employ a probabilistic approach that seeks the labels that maximize the conditional probability $p(D|\theta)$. Training amounts to finding good model parameters (θ), and the maximum likelihood estimate (MLE),

$$\hat{\theta} = \arg\max_{\theta}[p(D|\theta)] = \arg\max_{\theta}[\log p(D|\theta)] \tag{5}$$

gives the parameters under which the data likelihood is the largest. The log-likelihood yields the same maxima as the straight likelihood because log() is monotonic, and using a log-likelihood simplifies the next step considerably. Simplifying log-products into sum-logs and substituting Bayes rule into Equation 5 gives,

$$\begin{aligned} \log p(D|\theta) &= \log\left(\prod_{i=1}^{l} p(x_i, y_i|\theta)^{\beta} \prod_{i=l+1}^{l+u} p(x_i|\theta)^{1-\beta}\right) \\ &= \beta\sum_{i=1}^{l} \log\left[p(y_i|\theta)p(x_i|y_i,\theta)\right] + (1-\beta)\sum_{i=l+1}^{l+u} \log p(x_i|\theta) \end{aligned} \tag{6}$$

where the first term is the supervised likelihood and the second term is the marginal (unsupervised) likelihood. Equation 6 is the objective function in general, but if the model parameters are those that describe a multivariate Gaussian distribution for each class, then Equation 6 becomes,

$$\log p(D|\theta) = \beta\sum_{i=1}^{l} \log\left[\pi_{y_i}\mathcal{N}(x_i;\mu_{y_i},\Sigma_{y_i})\right] + (1-\beta)\sum_{i=l+1}^{l+u} \log\left(\sum_{k=l}^{K}\mathcal{N}(x_i;\mu_k,\Sigma_k)\right) \tag{7}$$

where π, μ, and Σ represent the Gaussian priors, means, and covariances respectively, and $\mathcal{N}$ represents a normal distribution. The standard implementations of ssGMM (Nigam et al., 2000; Zhu and Goldberg, 2009, Chapter 3) do not include β, but this parameter is introduced by Yan et al. (2017) to give a relative weighting ($0 < \beta < 1$) between the labeled and unlabeled portions of the log-likelihood.

To find the MLE, Equation 7 needs to be optimized. Unfortunately, the unobserved labels (H) make the log-likelihood non-concave and hard to optimize. The standard remedy is to use the Expectation-Maximization (EM) algorithm (Dempster et al., 1977), which provides an iterative solution to obtaining the MLE in the context of hidden data. For this implementation of ssGMM, the EM algorithm is as follows:

- Step 1: Set $t = 0$ and initialize the model parameters for each class (there are K classes) for a multivariate Gaussian distribution. We achieve this by training a Gaussian Naïve Bayes classifier on

the labeled data and obtaining,

$$\theta^{(0)} = \{\pi_k^{(0)}, \mu_k^{(0)}, \Sigma_k^{(0)}\}_{\forall k}. \tag{8}$$

- Step 2: The E-step. Create a matrix γ that is size $n \times K$ where $n = l + u$. For labeled instances $(i = 1, ..., l)$, define $\gamma_{ik} = 1$ if $y_i = k$, and 0 otherwise. For the unlabeled instances $(i = l+1, ..., l+u)$, calculate γ_{ik} via,

$$\gamma_{ik} = \frac{\pi_k^{(t)} \mathcal{N}(x_i; \mu_k^{(t)}, \Sigma_k^{(t)})}{\sum_j \pi_j^{(t)} \mathcal{N}(x_i; \mu_j^{(t)}, \Sigma_j^{(t)})}. \tag{9}$$

- Step 3: The M-step. Determine $\theta^{(t+1)}$ using the current γ_{ik}. For $k = 1, ..., K$ compute:

$$NL_k = \sum_{i=1}^{l} \gamma_{ik} \qquad NU_k = \sum_{i=l+1}^{l+u} \gamma_{ik} \qquad C = \beta NL_k + (1-\beta) NU_k$$

$$\begin{aligned} \pi_k^{(t+1)} &= \frac{C}{\beta l + (1-\beta)u} \\ \mu_k^{(t+1)} = \frac{\beta \sum_{i=1}^{l} \gamma_{ik} x_i}{C} &+ \frac{(1-\beta) \sum_{i=l+1}^{l+u} \gamma_{ik} x_i}{C} \\ \Sigma_k^{(t+1)} = \frac{\beta \sum_{i=1}^{l} \gamma_{ik} x_i (x_i - \mu_k^{(t+1)})(x_i - \mu_k^{(t+1)})^T}{C} &+ \frac{(1-\beta) \sum_{i=l+1}^{l+u} \gamma_{ik} x_i (x_i - \mu_k^{(t+1)})(x_i - \mu_k^{(t+1)})^T}{C} \end{aligned}$$

- Step 4: Increment the iteration step, $t = t + 1$

- Step 5: Repeat Steps 2-4 until Equation 7 converges to a tolerance defined as the percent change in the log-likelihood (e.g., $tolerance = 0.1$ is one-tenth of a percent)

The matrix γ represents the *soft labels* for the unlabeled data. Once the algorithm converges, a threshold can be applied to assign a hard classification to the unlabeled data points via,

$$y_i = \arg\max_k \gamma_{ik} \qquad for \quad i = l+1, ..., l+u. \tag{10}$$

Implementing this algorithm requires setting the hyper-parameters a priori, which are the tolerance and β.

ssGMM makes a critical assumption that the labeled data have been generated by multivariate Gaussian distributions and the unlabeled data use the same parametric model for classification. This cluster assumption states that if two points are in the same cluster, then they likely belong to the same class. However, it does not imply that each class forms an isolated cluster, but rather we should not observe data of two

distinct classes in the same cluster (Chapelle et al., 2006, Chapter 1). Similarly, for this implementation, this assumption implies that each class can only be described by one Gaussian cluster (i.e. it cannot be multinomial). If this cluster assumption is violated in any way (e.g., multinomial, non-Gaussian distributions), then semi-supervised learning could actually degrade performance (see Figures 3.2 and 3.3 in Zhu and Goldberg, 2009). Therefore, it is important to ensure that the data features for each class can be described by Gaussian distributions a priori.

An important advantage of ssGMM is that it is an inductive algorithm, which means it has the ability to predict labels on future unlabeled data not involved in the training process (i.e. a classifier). This is not the case for semi-supervised transductive algorithms, such as the label propagation technique used by Dunham et al. (accepted). In larger problems, the amount of unlabeled data could number in the millions, and including all of the unlabeled data in training, which would be required of transductive algorithms, may be computationally demanding. Inductive algorithms are advantageous in these situations because they can instead use a subset of the unlabeled data during training and then, in the case of ssGMM, the learned Gaussians can be used to classify the remaining unlabeled data.

2.2. Supervised methods

To assess if the ssGMM method can outperform supervised methods in the context of small training sets on our well log classification scenario, we need supervised methods to serve as a basis of comparison. We consider two methods, a Gaussian Naïve Bayes (GNB) classifier and a gradient boosting classifier. We will not discuss the details of these algorithms here, but we refer the interested reader to the literature for GNB (Theodoridis, 2015) and gradient boosting classifiers (Friedman, 2001). We choose GNB because it represents the fully-supervised version of ssGMM and the output of GNB is the starting condition for ssGMM (see Equation 8); comparing the performance of ssGMM to GNB will indicate if including the unlabeled data into the training process is beneficial. We use the *GaussianNB* class in `scikit-learn` and implementing this algorithm is trivial because GNB contains no hyper-parameters and no cross-validation is required. For our second supervised method, we use a particular gradient boosting algorithm, *XGBoost* (Chen and Guestrin, 2016), which is the algorithm of choice for the winners of the 2016 SEG machine learning competition (Hall and Hall, 2017). Since we are using the exact same dataset for our study, *XGBoost* is the best method to compare against ssGMM in terms of performance. Unlike GNB, *XGBoost* has several hyper-parameters that require setting, but *XGBoost* has been applied in many different disciplines and is proven to be robust (Tamayo et al., 2016; Torlay et al., 2017; Zhang et al., 2018)

2.3. Model selection strategies

Supervised and semi-supervised algorithms commonly have hyper-parameters that need to be tuned, and the process of choosing a set of hyper-parameters is called model selection. The model selection process first

consists of splitting the training data (L) into training and validation sets and then each hyper-parameter setting is evaluated on the validation data using classification metrics such as accuracy, precision, recall, F1, or the area under the curve (AUC) for ROC curves (Lever et al., 2016). If only one validation set is used, it is well-known that there is a larger risk of a machine learning model overfitting the training data, which leads to poor prediction accuracies. One remedy is to break the training data into k pieces, or folds, and train the machine learning algorithm with a given hyper-parameter setting on $k-1$ folds, evaluate the performance on the k^{th} fold, and then repeat this process for each fold; this is k-fold cross-validation (CV). For 5-fold CV, for instance, each hyper-parameter setting has a classification score on each of the 5 validation sets. If the data are shuffled and this process is repeated 5 times (i.e. 5-repeated 5-fold CV), then there are 25 classification scores for each hyper-parameter (Krstajic et al., 2014). Conventionally, all of the classification scores are averaged and the standard model selection tactic is to select the hyper-parameter combination with the largest mean CV score (Bishop, 2006, Chapter 1.3; Hastie et al., 2009, Chapter 7.10; Krstajic et al., 2014), and the same can be said for averaging AUC values in multi-class situations (Hand and Till, 2001; Fawcett, 2006). This hyper-parameter combination is then used when trying to classify the unlabeled data (U). The model selection functions in `scikit-learn` in Python use the same selection approach, and when there is a conflict (i.e. multiple hyper-parameter combinations with the same mean CV score), the least complicated model is selected.

The standard model selection strategy only uses the mean CV scores, but there are also accompanying *standard deviation* CV scores that, to the best of our knowledge, have yet to have been directly leveraged in any model selection process. What we are proposing here is a new selection strategy that *simultaneously* uses the mean and standard deviation CV scores to select a model, i.e. a simultaneous mean and standard deviation (SMSD) score,

$$SMSD_j = \alpha \left(\frac{\bar{\boldsymbol{x}}_{\boldsymbol{j}} - \bar{x}_{CVmin}}{\bar{x}_{CVmax} - \bar{x}_{CVmin}} \right) + (1-\alpha) \left(\frac{\sigma_{CVmax} - \boldsymbol{\sigma_j}}{\sigma_{CVmax} - \sigma_{CVmin}} \right) \tag{11}$$

where $\bar{\boldsymbol{x}}_{\boldsymbol{j}}$ and $\boldsymbol{\sigma_j}$ are the mean and standard CV scores for the j^{th} hyper-parameter combination, $\bar{x}_{CVmin}$ and $\bar{x}_{CVmax}$ are the minimum and maximum mean CV scores on the hyper-parameter grid, σ_{CVmin} and σ_{CVmax} are the minimum and maximum standard deviation CV scores on the hyper-parameter grid, and α gives a relative weight between the mean and standard deviation CV scores. Generally, α can vary on the interval $[0, 1]$, but allowing it to change adds a level of complexity. For the context of this paper, we fix $\alpha = 0.5$ to let the mean and standard deviation scores *equally* contribute to the decision. The model selection scheme using SMSD is simply choosing the hyper-parameter combination with the highest SMSD score, and if there is a conflict, then the combination with the least complicated model is chosen. For *XGBoost* (XGB), this is the model with smaller hyper-parameter values, and for ssGMM, this is the model with the smallest model complexity value that we define in Figure 1. This scheme is analogous to regularized inverse problems where

we not only care about data misfit, but we also care about some measure of the model norm, and we weight the contribution of both these factors.

What we are trying to address with the SMSD method is that models with the highest mean CV score may not always be optimal. For instance, if a given hyper-parameter combination has the highest mean CV score, but also has the highest standard deviation CV score, is this the optimal model? Arguably, the optimal model is one that is both accurate (i.e. high mean) and precise (i.e. low standard deviation) and our SMSD method will select a model that has a balance between the mean and standard deviation scores. It is worth mentioning that if the highest mean CV score and the lowest standard deviation CV score align, then the SMSD model choice is equivalent to the default model choice. Where the model choices from the SMSD and the traditional methods will differ is when the highest mean and the lowest standard deviation CV scores do not align, and what we try to investigate in this paper is if the SMSD models perform better on the testing data in these situations.

3. Well log dataset

The well log dataset used for this study is the same dataset that was used for an SEG machine learning competition held in 2016 (Hall, 2016; Hall and Hall, 2017), but the data were ultimately made public by the Kansas Geological Survey. The data consist of ten wells (nine are real, but one is synthetic) drilled in the Hugoton and Panoma fields of southwest Kansas and northwest Oklahoma, and we refer the interested reader to Dubois et al. (2006) for a discussion of the geology of this region. All ten wells contain wire-line log data (i.e. the instances x_i) and core samples (i.e. the associated labels y_i) recorded at half-foot increments for 4137 total data points. Dubois et al. (2006) determine that there are nine lithofacies, or classes, in this dataset and the first three (1-3) are non-marine and the remaining six classes (4-9) are marine lithofacies (see Table 1). The features, or dimensions, for each instance consist of five wire-line logs and two geologic variables. Figure 2 shows a cross-plot of each of the seven features plotted against each other for the entire dataset. Notice that the NM_M indicator is effective at distinguishing the non-marine from the marine classes, but the classes *within* the non-marine and marine categories are not linearly separable with considerable overlap. It is well-understood that rock units are not always discrete and their physical properties are not unique, and this can lead to poor class separability (Avseth et al., 2005). Misclassifications of this dataset are expected to occur as a result of this, but one could argue that if a predicted lithofacies is *close* to the true facies, then this could still be classified as correct. For the machine learning competition associated with this dataset, Hall (2016) introduces an *adjacent accuracy* metric that deems lithofacies that occur close to each other depositionally (i.e. Walthers Law) are also considered correct. These adjacent facies are indicated in Table 1 and we also include this metric.

The machine learning competition using this dataset was structured for the competitors to train a

classifier using all ten wells (L) and the competition organizers would test their classifiers on two withheld wells (U), but the public only has access to the ten wells. To properly simulate a semi-supervised situation with this dataset, we restructure the classification problem so that only one well is used as the labeled data (L) and the objective is to predict the lithofacies for the remaining nine wells (U). Dunham et al. (accepted) use the same well log dataset for their study and also restructure their classification problem in the same way. The question now becomes which well to use for L. For continuity purposes, we choose the same well as Dunham et al. (accepted), KIMZEY, and we refer the interested reader to their study for the justification of this choice. Figure 3 depicts the distribution of points for both the labeled and unlabeled data in this situation. The distributions between the labeled and unlabeled data are similar with minor differences, but what is noteworthy in Figure 3(a) is that some classes have very few points and the consequences of this are discussed in Section 5.2. Prior to any classification, these data are scaled and normalized using the *RobustScaler* class from `scikit-learn`.

4. Results

4.1. Initial ssGMM test

With ssGMM being applied to this well log dataset for the first time, our first step consists of a testing stage to investigate how the algorithm behaves. No cross-validation procedures are used and we simply train the algorithm on the one labeled well (KIMZEY) using default values ($\beta = 0.50, tolerance = log_{10}[-1.5]$). This initial test shows that the objective function for ssGMM does not converge (dashed black line in Figure 4). Recall from Section 2.1 that the inherent cluster assumption of ssGMM is subject to violation if the data cannot be represented by multivariate Gaussians. Notice in Figure 2 that most of the variables exhibit Gaussian-like distributions, but the exception is the NM_M indicator which is a binary, bimodal variable that cannot be represented by a Gaussian distribution. Algorithmically, this makes the covariances matrices for each class poorly defined and taking the inverse of these matrices (required to compute $\mathcal{N}$ in Equation 9) causes an instability.

Our remedy for this problem is to remove the NM_M indicator variable by decomposing the well log data into two separate datasets based on the NM_M indicator, identically to what is done in Dunham et al. (accepted). This decomposes the original dataset into non-marine (NM_M = 1) and marine (NM M = 2) facies datasets that correspond to classes 1-3 and 4-9, respectively (see Table 1). The ssGMM algorithm is trained again (using default hyper-parameters) on the non-marine and marine subsets of KIMZEY, and Figure 4 shows that the objective function easily converges for both. This is evidence that the NM_M variable is the underlying cause for the ssGMM algorithm not converging on the global data. Moving forward, we apply all algorithms to the separate non-marine and marine datasets.

4.2. Comparison to Dunham et al.

Based on our necessity to decompose the well log data into non-marine and marine datasets, we can make direct comparisons to results from Dunham et al. (accepted) because they performed the exact same decomposition of this dataset. They consider 3-fold CV with total accuracy as their classification metric to train all their algorithms, and we use the same scheme here to make a direct comparison. We mirror the choice of the XGB hyper-parameter grid from Bestagini et al. (2017), but we modify it slightly to achieve better performance for this situation. The hyper-parameter grid we consider for ssGMM is indicated in Figure 1 with 31 logarithmically spaced values for the *tolerance* and 17 linearly spaced values for β. We first evaluate the total accuracy of each hyper-parameter choice on the testing data for both non-marine and marine datasets, and doing so will indicate how close future ssGMM models determined via CV are to the maximum achievable accuracies (see Figure 5). A summary of the results is given in Table 2 where the last row is the self-training label propagation result taken from Table 3 in Dunham et al. (accepted). The mean and standard deviation 3-fold CV scores for ssGMM are given in Figure 6, and the selected models are indicated by circles. Using the SMSD score (Equation 11) as the model selection strategy is not the focus of this section, but Figures 6(c) and 6(f) indicate that using the SMSD score gives the same model as the default approach. In Table 2, we see ssGMM and self-training label propagation performing better than GNB and XGB. However, the rows indicating the best possible performance for XGB and ssGMM on the unlabeled data suggest that there is room for improvement for both of these methods.

4.3. Improving XGB and ssGMM performance through model selection

Only 3-fold CV is considered in Dunham et al. (accepted), but we consider a higher fold here to see if improvement can be gained for XGB and ssGMM. The highest fold we can test is 7-fold because Class 9 only has seven points (see Figure 3) and we need at least one point from each class in each fold. However, 5-fold CV is standard and that is what we test here. Table 3 (top) summarizes the results, and Figure 7 gives the mean, standard deviation, and SMSD 5-fold CV scores for ssGMM. We see a minor performance improvement for ssGMM using 5-fold rather than 3-fold CV (compare to Table 2), and the model obtained using the SMSD score gives a marginal improvement over the default model for 5-fold CV. Using 5-fold CV for XGB deteriorated the performance slightly compared to using 3-fold CV (see Table 2), but using the SMSD score selected a model for XGB that performed slightly better in the 5-fold CV case.

Using 5-fold CV only produces five classification metric scores for each hyper-parameter combination to compute the mean and standard deviation CV scores from. However, five values are arguably not enough for the computed mean and standard deviation scores to be statistically significant. A solution to this problem is N-repeated k-fold CV (discussed in Section 2.3) and we consider 5-repeated 5-fold CV so there are 25 classification scores for each hyper-parameter combination. Table 3 (bottom) summarizes these results, and Figure 8 gives the mean, standard deviation, and SMSD 5-repeated 5-fold CV scores for ssGMM. While

the default model for ssGMM on the non-marine data performs well, the default model for the marine data is quite poor and hinders the overall performance. However, the models obtained using the SMSD score vastly improve the performance on both datasets. The default models for XGB perform well, but the models obtained using the SMSD score diminish the performance slightly. Nonetheless, for both XGB and ssGMM, using 5-repeated 5-fold CV gives an improvement in performance compared to simply using 3-fold CV.

5. Discussion

5.1. Overall performance comparison

One objective of our study is to assess if ssGMM can outperform the considered supervised methods (GNB and XGB) in the context of minimal training data. Recall from Section 2.2 that we consider GNB because it represents the fully-supervised version of ssGMM. Tables 2 and 3 show that ssGMM outperforms GNB in every circumstance and this indicates, for this algorithm, that including the unlabeled data into the training process substantially improves its performance. Our results also demonstrate that ssGMM is always able to outperform XGB in terms of accuracy (1-5% for all classification metrics) and computation time (**ADD SOMETHING HERE ABOUT COMPUTATION TIME**). However, the comparison to the self-training label propagation method of Dunham et al. (accepted) in Section 4.2 indicates that even if we are able to recover the best-possible ssGMM model, this does not outperform their result (see Table 2). In terms of performance, and also computation time, the self-training label propagation technique appears to be preferred.

The second objective of our study is to determine if simultaneously using the mean and standard deviation CV scores (i.e. the SMSD score) to select models is preferred in comparison to the default approach of only using the mean CV scores. For the ssGMM method, Tables 2 and 3 indicate that the SMSD score selects models that perform the same or better than the default models, and the fact that ssGMM only has two hyper-parameters made the visualization of this quite clear (i.e. Figures 5-8). Using the SMSD score to select models for XGB gives mixed results. It seems when the default XGB model performs poorly, then the SMSD score can select a better model (Table 3 for 5-fold CV). However, if the default XGB model is already performing well, then using the SMSD score appears to select a worse model (Table 3 for 5-repeated 5-fold CV). XGB has many hyper-parameters, and so it is difficult to ascertain the cause of this phenomenon without being able to visualize its performance in the way we did for ssGMM.

5.2. Interpretation

There are a few methods for interpreting these classification results, and the first is inspecting the performance on a per-well basis rather than globally. Table 4 shows the accuracy scores on each of the nine unlabeled wells for the best recovered XGB and ssGMM models. ssGMM is able to outperform XGB

on seven of the nine unlabeled wells by a notable margin. We can also interpret the classification results by physically observing the facies predictions. The performance of XGB and ssGMM on the SHANKLE and NOLAN wells is most characteristic of their overall performance, and so we choose to show the facies predictions for these two wells in Figures 9 and 10. Notice how in both wells, the XGB prediction is visibly chaotic and this is evidence of overfitting. However, the ssGMM predictions are must less chaotic and fit the true facies better, which supports the claim in Section 1 that including unlabeled data in the training process is akin to regularization for inverse problems and this helps prevent overfitting. The maximum probabilities used to classify the unlabeled data for ssGMM (Equation 10) are given as the probability logs in Figures 9 and 10. These probabilities are a unique benefit to ssGMM and they can be useful for interpretation. For instance, the probability tends to drop at predicted lithofacies interfaces.

Another technique for visualizing and interpreting the classification performance is through confusion matrices. Figure 11 shows the confusion matrices for best recovered XGB and ssGMM models. Notice how the predictions cluster closer to the diagonal for ssGMM compared to XGB; this is reflected by the higher accuracy and adjacent accuracy as indicated by Table 3. Even though the classification ability of XGB for Classes 1 and 9 is poor (<10%), ssGMM is unable to predict for these two classes at all. Classes 1 and 9 only have seven and nine points respectively (Figure 3) and the initial covariance matrices describing each class, which are 6-dimensional, are critically defined because of this. In Table 4, the RECRUIT F9 well is a synthetic well that only contains Class 9 and this is why ssGMM has an accuracy of 0% on that well. While the training data for other classes are sufficient to define their corresponding covariance matrices, more data are certainly needed to constrain the covariance matrices for Classes 1 and 9.

6. Conclusion

The two objectives of this paper are to investigate (1) if semi-supervised Gaussian mixture models (ssGMM) can outperform a popular supervised algorithm, *XGBoost* (XGB), in the context of a well log classification example with limited training data, and (2) if our new model selection strategy that simultaneously uses the mean and standard deviation (SMSD) cross-validation scores can make better model selections than the default approach. The results first show that one of the well log data features violates the Gaussian assumption, which causes ssGMM to not converge, but our remedy for this situation is decomposing the dataset into two pieces to remove this feature. This demonstrates how important it is to perform tests prior to classification to ensure that the data are not violating any underlying assumptions. Our results demonstrate that the ssGMM method is able to outperform XGB, but not by a significant margin. However, the benefit of ssGMM is that it is a simple semi-supervised algorithm (i.e. only two hyper-parameters) that can still achieve a better performance than a complex supervised algorithm. Using our new proposed SMSD score for model selection gives promising results for ssGMM, but mixed results for XGB. However, we only

compare the performance of the default and SMSD model selection strategies using one dataset and two algorithms, and so future tests using different algorithms and/or datasets would help determine the true efficacy of the SMSD model selection strategy. Nonetheless, our visualization of the lithofacies predictions supports the claim that supervised methods are prone to overfitting when the training data are minimal, but including the unlabeled data into the training process (i.e. semi-supervised learning) mitigates this phenomenon.

7. Acknowledgements

The authors would like to thank Chevron, the Natural Sciences and Engineering Research Council of Canada, and InnovateNL for their financial support.

8. Computer Code Availability

- *Name of code*: `ssGMM`
- *Developer*: Michael W. Dunham
- *Contact*: Department of Earth Sciences, Memorial University of Newfoundland, St. Johns NL A1B 3X5, Canada; mwdunham@mun.ca
- *First release*: 2019
- *Hardware required*: tests were performed on a standard workstation with the following specifications: Intel i5-4750 (4 CPUs) 3.20 GHz processor + 16 GB RAM
- *Software required*: core Python 3 modules: `scikit-learn`, `joblib`, `numpy`, `pandas`, `matplotlib`.
- *Programming language*: the code is written in Python 3
- *How to access the source code*: the source file for the `ssGMM` program, data files, and accompanying Jupyter notebooks that reproduce results from this paper are included in the following public repository: `https://github.com/mwdunham/ssGMM`

References

Al-Bulushi, N., King, P.R., Blunt, M.J., Kraaijveld, M., 2009. Development of artificial neural network models for predicting water saturation and fluid distribution. Journal of Petroleum Science and Engineering 68, 197–208. doi:10.1016/j.petrol.2009.06.017.

Al-Bulushi, N.I., King, P.R., Blunt, M.J., Kraaijveld, M., 2012. Artificial neural networks workflow and its application in the petroleum industry. Neural Computing and Applications 21, 409–421. doi:10.1007/s00521-010-0501-6.

Aster, R.C., Borchers, B., Thurber, C.H., 2005. Parameter estimation and inverse problems. volume 90. Elsevier Academic Press.

Avseth, P., Mukerji, T., Mavko, G., 2005. Quantitative seismic interpretation: Applying rock physics tools to reduce interpretation risk. Cambridge University Press.

Baldwin, J.L., Bateman, R.M., Wheatley, C.L., 1990. Application of a neural network to the problem of mineral identification from well-logs. The Log Analyst 31, 279–293.

Benaouda, D., Wadge, G., Whitmarsh, R.B., Rothwell, R.G., MacLeod, C., 1999. Inferring the lithology of borehole rocks by applying neural network classifiers to downhole logs: an example from the Ocean Drilling Program. Geophysical Journal International 136, 477–491. doi:10.1046/j.1365-246X.1999.00746.x.

Bestagini, P., Lipari, V., Tubaro, S., 2017. A machine learning approach to facies classification using well logs, in: SEG Technical Program Expanded Abstracts 2017, Society of Exploration Geophysicists. pp. 2137–2142. doi:10.1190/segam2017-17729805.1.

Bishop, C.M., 2006. Pattern recognition and machine learning. Information Science and Statistics, Springer-Verlag.

Chapelle, O., Schölkopf, B., Zien, A., 2006. Semi-supervised learning. 1st ed., MIT Press, Cambridge, MA.

Chen, T., Guestrin, C., 2016. XGBoost: A scalable tree boosting system, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, San Francisco, California, USA. pp. 785–794. doi:10.1145/2939672.2939785.

Dempster, A.P., Laird, N.M., Rubin, D.B., 1977. Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society 39, 1–38.

Dubois, M.K., Bohling, G.C., Chakrabarti, S., 2007. Comparison of four approaches to a rock facies classification problem. Computers and Geosciences 33, 599–617. doi:10.1016/j.cageo.2006.08.011.

Dubois, M.K., Byrnes, A.P., Bohling, G.C., Doveton, J.H., 2006. Multiscale geologic and petrophysical modeling of the giant Hugoton Gas Field (Permian), Kansas and Oklahoma, U.S.A., in: Harris, P.M., Weber, L.J. (Eds.), Giant hydrocarbon reservoirs of the world: From rocks to reservoir characterization and modeling. AAPG Memoir 88/SEPM Special Publication, pp. 307–353.

Dunham, M.W., Malcolm, A., Welford, J.K., accepted. Improved well log classification using semi-supervised label propagation and self-training, with comparisons to popular supervised algorithms. Geophysics .

Fawcett, T., 2006. An introduction to roc analysis. Pattern Recognition Letters 27, 861–874. doi:10.1016/j.patrec.2005.10.010.

Friedman, J.H., 2001. Greedy function approximation: A gradient boosting machine. The Annals of Statistics 29, 1189–1232. URL: `http://www.jstor.org/stable/2699986`.

Hall, B., 2016. Facies classification using machine learning. The Leading Edge 35, 906–909. doi:10.1190/tle35100906.1.

Hall, M., Hall, B., 2017. Distributed collaborative prediction: Results of the machine learning contest. The Leading Edge 36, 267–269. doi:10.1190/tle36030267.1.

Hand, D.J., Till, R.J., 2001. A simple generalisation of the area under the ROC curve for multiple class classification problems. Machine Learning 45, 171–186. doi:10.1023/A:1010920819831.

Hastie, T., Tibshirani, R., Friedman, J., 2009. The elements of statistical learning. Springer Series in Statistics. 2 ed., Springer-Verlag. doi:10.1007/978-0-387-84858-7.

Keynejad, S., Sbar, M.L., Johnson, R.A., 2019. Assessment of machine-learning techniques in predicting lithofluid facies logs in hydrocarbon wells. Interpretation 7, SF1–SF13. doi:10.1190/INT-2018-0115.1.

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. ImageNet classification with deep convolutional neural networks, in: Pereira, F., Burges, C.J.C., Bottou, L., Weinberger, K.Q. (Eds.), Advances in Neural Information Processing Systems 25. Curran Associates, Inc., pp. 1097–1105.

Krstajic, D., Buturovic, L.J., Leahy, D.E., Thomas, S., 2014. Cross-validation pitfalls when selecting and assessing regression and classification models. Journal of Cheminformatics 6, 1–15. doi:10.1186/1758-2946-6-10.

Lever, J., Krzywinski, M., Altman, N., 2016. Point of significance: classification evaluation. Nature Methods 13, 603–604. doi:10.1038/nmeth.3945.

Maiti, S., Tiwari, R.K., Kümpel, H.J., 2007. Neural network modelling and classification of lithofacies using well log data: A case study from KTB borehole site. Geophysical Journal International 169, 733–746. doi:10.1111/j.1365-246X.2007.03342.x.

Malvić, T., Velić, J., Horváth, J., Cvetković, M., 2010. Neural networks in petroleum geology as interpretation tools. Central European Geology 53, 97–115. doi:10.1556/CEuGeol.53.2010.1.6.

McCormack, M.D., 1991. Neural computing in geophysics. The Leading Edge 10, 11–15. doi:10.1190/1.1436771.

Nigam, K., Mccallum, A.K., Thrun, S., Mitchell, T., 2000. Text classification from labeled and unlabeled documents using EM. Machine Learning 39, 103–134. doi:10.1023/A:100769271.

Rogers, S.J., Fang, J.H., Karr, C.L., Stanley, D.A., 1992. Determination of lithology from well logs using a neural network. AAPG Bulletin 76, 731–739.

Saggaf, M.M., Nebrija, E.L., 2000. Estimation of lithologies and depositional facies from wire-line logs. AAPG Bulletin 84, 1633–1646.

Saggaf, M.M., Nebrija, E.L., 2003. A fuzzy logic approach for the estimation of facies from wire-line logs. AAPG Bulletin 87, 1223–1240. doi:10.1306/02260301019.

Tamayo, D., Silburt, A., Valencia, D., Menou, K., Ali-Dib, M., Petrovich, C., Huang, C.X., Rein, H., van Laerhoven, C., Paradise, A., Obertas, A., Murray, N., 2016. A machine learns to predict the stability of tightly packed planetary systems. The Astrophysical Journal Letters 832, 1–5. doi:10.3847/2041-8205/832/2/L22.

Theodoridis, S., 2015. Machine learning: a Bayesian and optimization perspective. Academic Press Inc.

Torlay, L., Perrone-Bertolotti, M., Thomas, E., Baciu, M., 2017. Machine learning - XGBoost analysis of language networks to classify patients with epilepsy. Brain Informatics 4, 159–169. doi:10.1007/s40708-017-0065-7.

Waldeland, A.U., Jensen, A.C., Gelius, L.J., Solberg, A.H.S., 2018. Convolutional neural networks for automated seismic interpretation. The Leading Edge 37, 529–537. doi:10.1190/tle37070529.1.

Wang, G., R.Carr, T., Ju, Y., Li, C., 2013. Identifying organic-rich Marcellus Shale lithofacies by support vector machine classifier in the Appalachian Basin. Computers & Geoscience 64, 52–60. doi:10.1016/j.cageo.2013.12.002.

Xing, X., Yu, Y., Jiang, H., Du, S., 2013. A multi-manifold semi-supervised Gaussian mixture model for pattern classification. Pattern Recognition Letters 34, 2118–2125. doi:10.1016/j.patrec.2013.08.005.

Yan, H., Zhou, J., Pang, C.K., 2017. Gaussian mixture model using semisupervised learning for probabilistic fault diagnosis under new data categories. IEEE Transactions on Instrumentation and Measurement 66, 723–733. doi:10.1109/TIM.2017.2654552.

Zhang, D., Qian, L., Mao, B., Huang, C., Huang, B., Si, Y., 2018. A data-driven design for fault detection of wind turbines using random forests and XGboost. IEEE Access 6, 21020–21031. doi:10.1109/ACCESS.2018.2818678.

Zhu, X., Goldberg, A.B., 2009. Introduction to semi-supervised learning. Synthesis lectures on artificial intelligence and machine learning, Morgan & Claypool. doi:10.2200/S00196ED1V01Y200906AIM006.

Tables and Figures

Table 1: The nine lithofacies for the well log dataset and their corresponding descriptions. The Label column serves as a key for the colors associated with each facies in Figures 2, 3, 9, and 10. The classes listed in the Adjacent facies column are used to compute the adjacent accuracy metric given by (Hall, 2016).

Facies	Description	Label	Adjacent facies
Non-marine classes			
1	Sandstone	SS	2
2	Coarse siltstone	CSiS	1,3
3	Fine siltstone	FSiS	2
Marine classes			
4	Siltstone and shale	SiSh	5
5	Mudstone	MS	4,6
6	Wackestone	WS	5,7,8
7	Dolomite	D	6,8
8	Packstone-grainstone	PS	6,7,9
9	Bafflestone	BS	7,8

Table 2: Testing data performance for the supervised methods (GNB, XGB) and the semi-supervised methods (ssGMM, self-training label propagation) where 3-fold CV with total accuracy as the classification metric is used to train all algorithms. The F-1 and adjacent accuracy metrics are not used for model selection, but are merely presented for comparison. The self-train LP row is taken directly from Dunham et al. (accepted). All computations are conducted on a desktop machine (3.2 GHz Intel Core i5 processor) with 16GB RAM and all cross-validations are performed in parallel on four cores. The XGB (best) and ssGMM (best) rows give the highest achievable total accuracy on the non-marine and marine testing data (indicated by the black bulls-eyes in Figure 5). The models representing the colored cells correspond to the same-colored circles in Figures 5 and 6.

Machine learning algorithm	**Non-marine accuracy (%)**	**Marine accuracy (%)**	**Total accuracy (%)**	**F-1 score (%)**	**Total adjacent accuracy (%)**	**Total # of CV fits**	**Elapsed time**
GNB	49.21	32.26	40.64	35.91	82.59	0	0.023 s
XGB	54.40	34.67	44.43	42.33	83.88	$700 \times 3 \times 2$	TO DO
XGB (best)	56.92	38.15	47.43	44.27	84.23	N/A	N/A
ssGMM	55.17	38.09	46.54	43.07	88.37	$527 \times 3 \times 2$	TO DO
ssGMM (best)	58.39	39.59	48.89	45.24	89.21	N/A	N/A
Self-train LP	62.33	39.17	50.62	49.50	88.02	$1681 \times 3 \times 2$	9.59 s

Table 3: Testing data performance for XGB and ssGMM using 5-fold and 5-repeated 5-fold cross-validation with total accuracy as the classification metric. Model selection for both algorithms in both situations is achieved using the standard and SMSD (Equation 11) approaches. GNB is not included because it contains no hyper-parameters and its performance is the same as shown in Table 2. The models representing the colored cells correspond to the same-colored circles in Figures 5, 7, and 8. The best performing XGB and ssGMM models are denoted by the *.

Machine learning algorithm	Non-marine accuracy (%)	Marine accuracy (%)	Total accuracy (%)	F-1 score (%)	Total adjacent accuracy (%)	Total # of CV fits	Elapsed time
5-fold cross-validation							
ssGMM	54.84	39.11	46.89	43.09	89.21	$527 \times 5 \times 2$	TO DO
ssGMM (SMSD)	55.27	39.38	47.24	43.39	89.40	$527 \times 5 \times 2$	TO DO
XGB	50.30	34.56	42.34	40.06	82.86	$700 \times 5 \times 2$	TO DO
XGB (SMSD)	50.30	36.33	43.24	40.41	82.88	$700 \times 5 \times 2$	TO DO
5-repeated 5-fold cross-validation							
ssGMM	55.99	33.87	44.81	38.35	84.86	$527 \times 5 \times 5 \times 2$	TO DO
ssGMM (SMSD)*	58.28	38.20	48.13	44.53	89.29	$527 \times 5 \times 5 \times 2$	TO DO
XGB*	55.27	34.78	44.92	42.63	83.64	$700 \times 5 \times 5 \times 2$	TO DO
XGB (SMSD)	53.75	34.72	44.13	42.35	83.69	$700 \times 5 \times 5 \times 2$	TO DO

Table 4: The prediction accuracies for the best XGB and ssGMM models decomposed into individual accuracies for each of the unlabeled wells. Table 3 denotes the best models for XGB and ssGMM come from 5-repeated 5-fold CV. The total accuracies provided in the last row correspond to those given in Table 3 and are calculated by taking the weighted average of the individual accuracies for each well.

Unlabeled well name	# points	XGB accuracy	ssGMM accuracy
SHANKLE	449	44.77	52.78
CROSS H CATTLE	501	28.94	32.34
NEWBY	463	41.04	44.28
LUKE	461	60.52	64.43
CHURCHMAN BIBLE	404	42.08	40.84
ALEXANDER	466	52.36	53.65
SHIMPLIN	471	51.59	54.99
NOLAN	415	44.34	49.40
RECRUIT F9	68	7.35	0.00
Total	3698	44.92	48.13

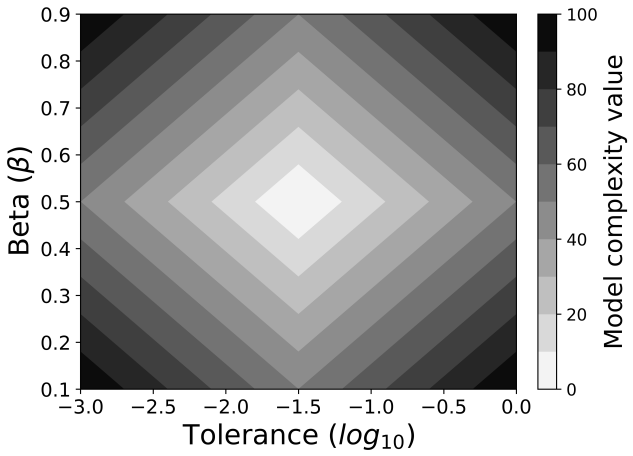

Figure 1: The assigned model complexity values for the ssGMM method. If a given model selection technique has a conflict, the hyper-parameter combination with the lowest model complexity value is chosen. This choice of model complexity is made to penalize extreme values for β and the *tolerance*, and favor those that are closer to what we deem to be default values: $(\beta, tolerance) = (0.5, log_{10}[-1.5])$.

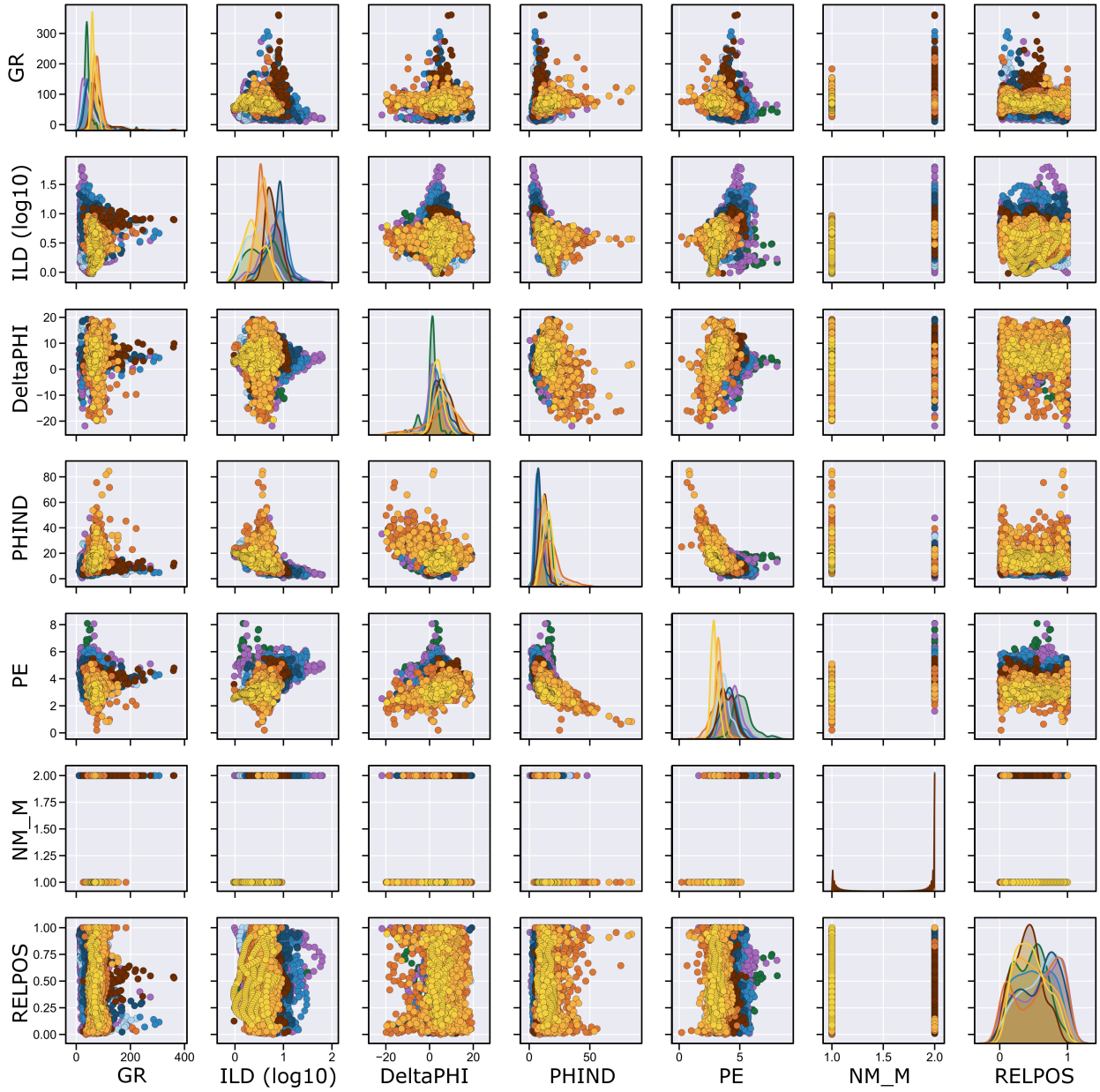

Figure 2: Cross-plot matrix showing the behavior for all seven data variables with respect to each other for the entire well log dataset. The variable abbreviations represent the following: GR = gamma ray, ILD (log_{10}) = resistivity, DeltaPHI = neutron-density porosity difference, PHIND = average neutron-density porosity, PE = photoelectric effect, NM_M = non-marine/marine indicator, and RELPOS = relative position. The first five features are log variables and the last two features are geologic variables. For the classes pertaining to each color, see Table 1. The distributions of each class for each variable are given by the diagonal elements in the matrix. The NM_M indicator is effective at separating the non-marine from the marine classes, but the classes within each category still overlap considerably.

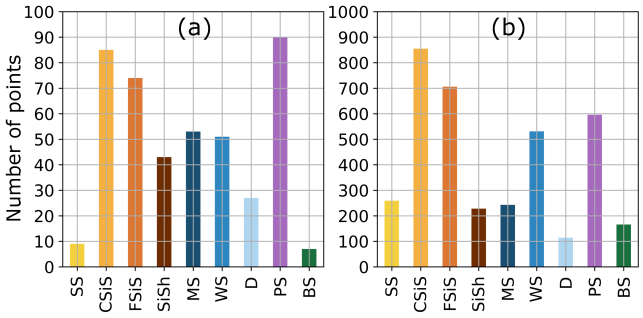

Figure 3: The distribution of points per class for the (a) one labeled well and the (b) remaining nine unlabeled wells. Classes 1 and 9 have the least number of training points with nine and seven points respectively.

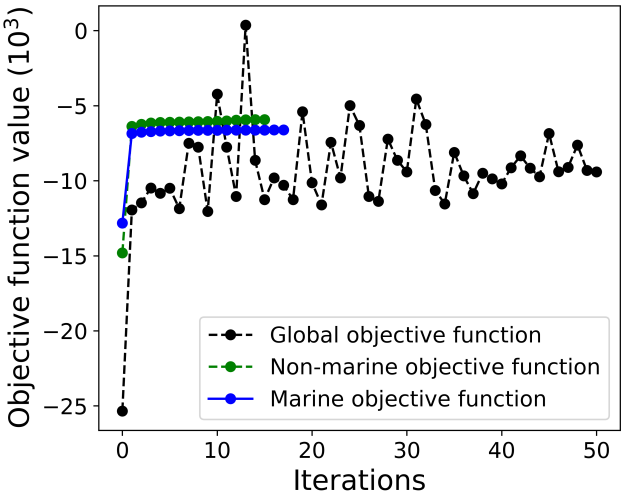

Figure 4: The behavior of the ssGMM objective function (Equation 7) with respect to the number of iterations when trained on the single labeled well using default parameter settings ($\beta = 0.50, tolerance = log_{10}[-1.5]$). ssGMM does not converge (dashed black line) because the NM_M indicator variable is not Gaussian distributed, but splitting the data into two pieces (non-marine and marine) allows the algorithm to converge.

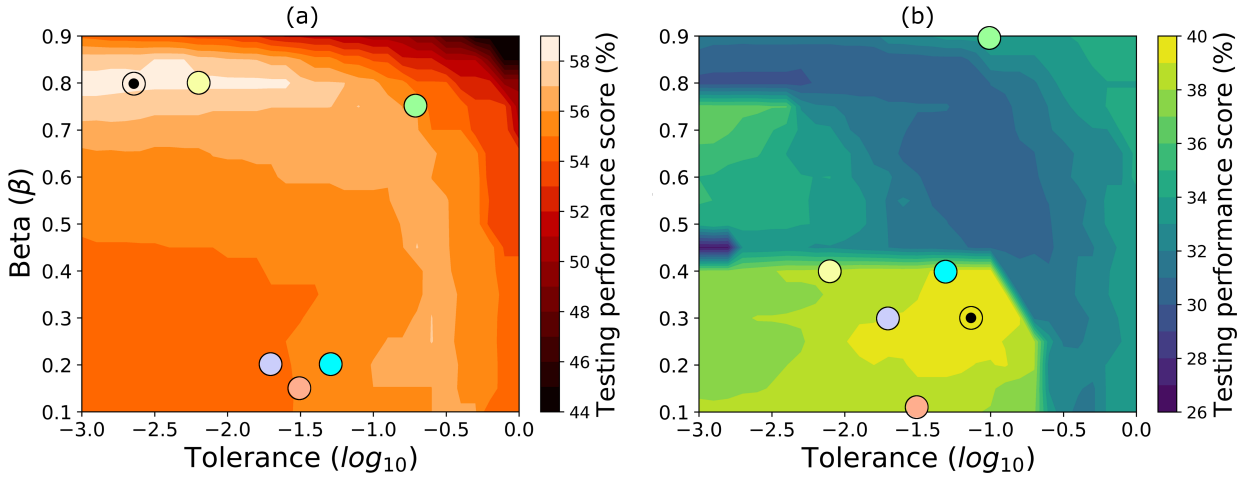

Figure 5: The total accuracy of each ssGMM hyper-parameter choice on the testing data for the (a) non-marine and (b) marine datasets. This does assume H (Equation 3) is known, which is unrealistic, but these panels help indicate how close the ssGMM models are to the highest accuracy zones. The black bulls-eyes indicate hyper-parameter choices that give the maximum achievable accuracy. The colored circles represent various ssGMM models, and their detailed numerical performances on the nine unlabeled wells are indicated in Tables 2 and 3.

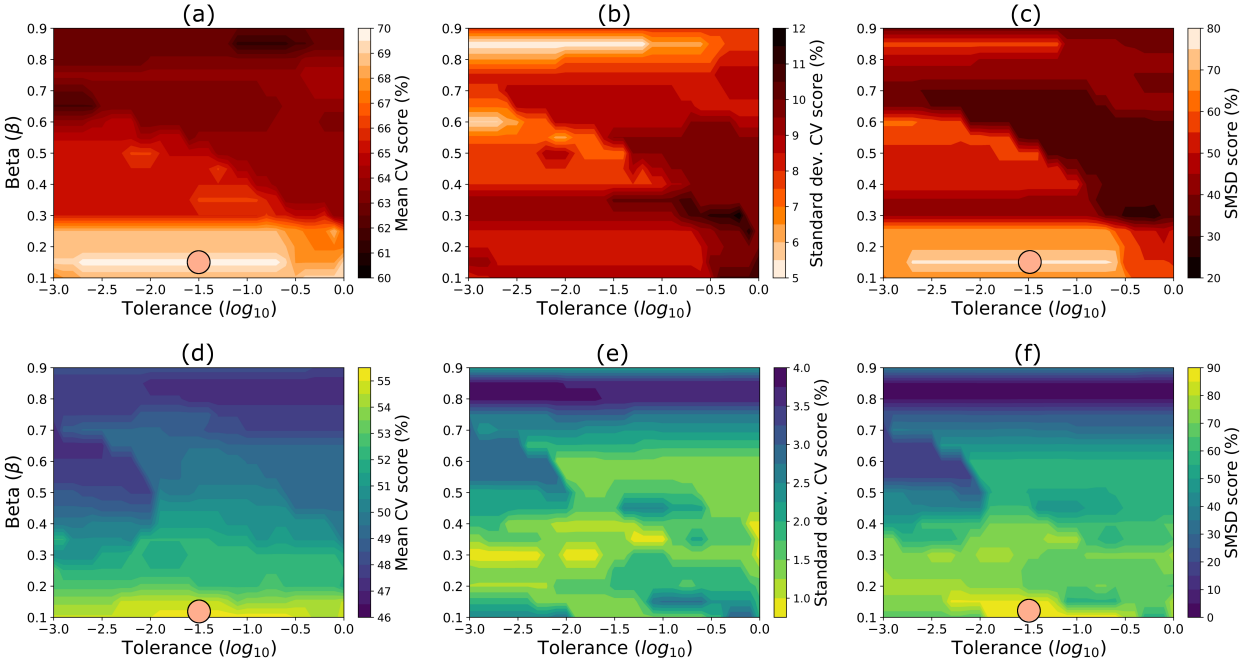

Figure 6: The mean, standard deviation, and SMSD 3-fold cross-validation scores using ssGMM for the non-marine (a, b, c) and marine (d, e, f) training datasets, respectively. Total accuracy is used as the classification metric. The standard model selection approach only utilizes the mean CV scores (a, d) and chooses models indicated by the circles in panels (a) and (d). The SMSD model selection approach uses both the mean and standard deviation CV scores (a, b, d, e) and chooses the models indicated by the circles in panels (c, f). For this situation, the chosen models are identical. See Figure 5 and Table 2 for the testing performance of the models indicated by circles.

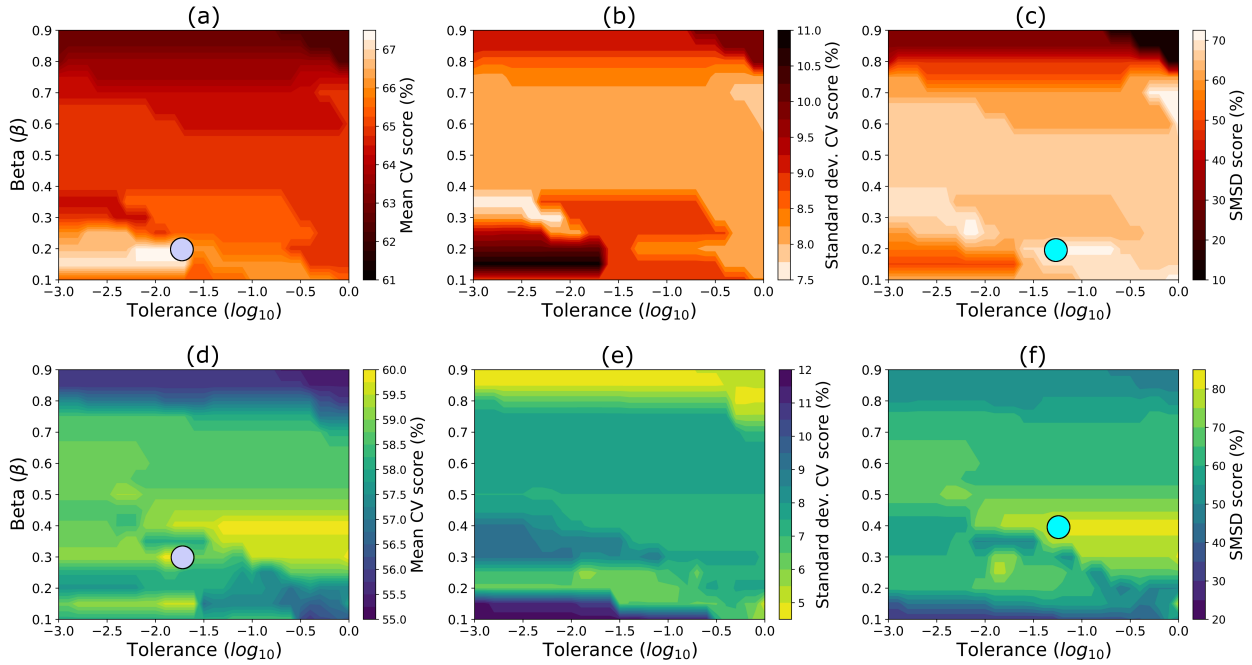

Figure 7: The mean, standard deviation, and SMSD 5-fold cross-validation scores using ssGMM for the non-marine (a, b, c) and marine (d, e, f) training datasets, respectively. Total accuracy is used as the classification metric. The default models are indicated by the purple circles in panels (a) and (d), and the SMSD models are indicated by the cyan circles in panels (c) and (f). For this situation, the SMSD models give a slightly better performance on the testing data (see Figure 5 and Table 3).

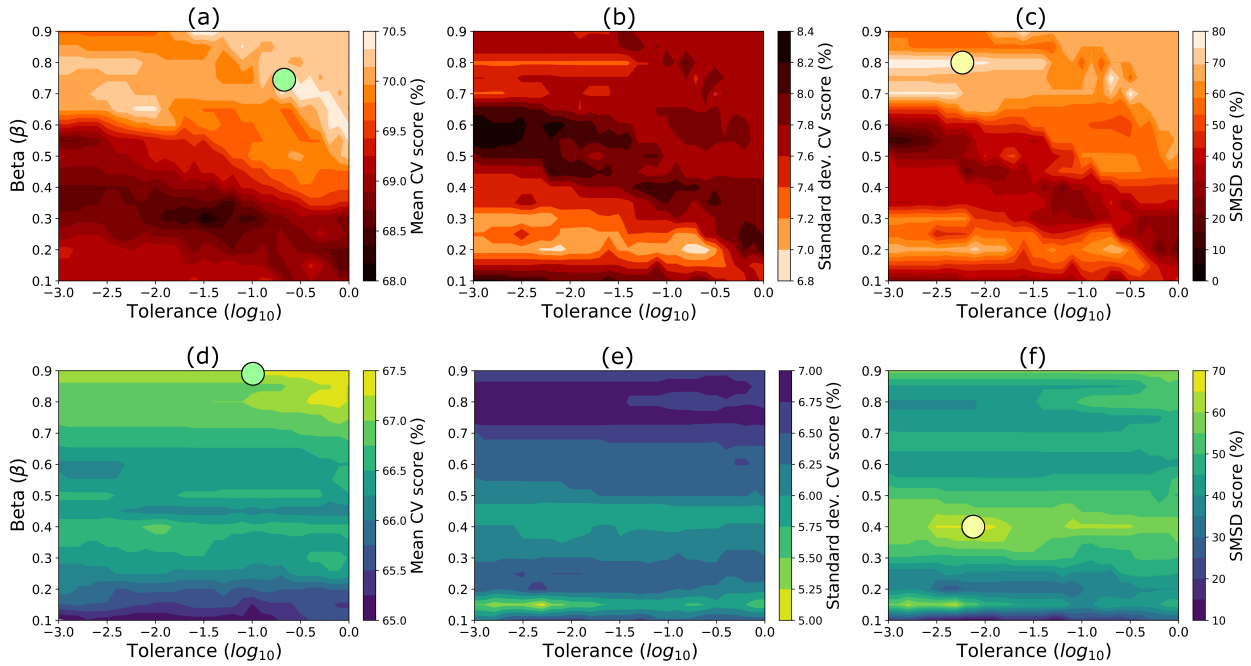

Figure 8: The mean, standard deviation, and SMSD 5-repeated 5-fold cross-validation scores using ssGMM for the non-marine (a, b, c) and marine (d, e, f) training datasets, respectively. Total accuracy is used as the classification metric. The default models are indicated by the green circles in panels (a) and (d), and the SMSD models are indicated by the yellow circles in panels (c) and (f). For this situation, the default non-marine model (a) performs well on the testing data, but the marine model (d) does not. However, the SMSD score chooses models that significantly improve the testing performance of both the non-marine and marine datasets. See Figure 5 and Table 3 for the testing performance of the models indicated by circles.

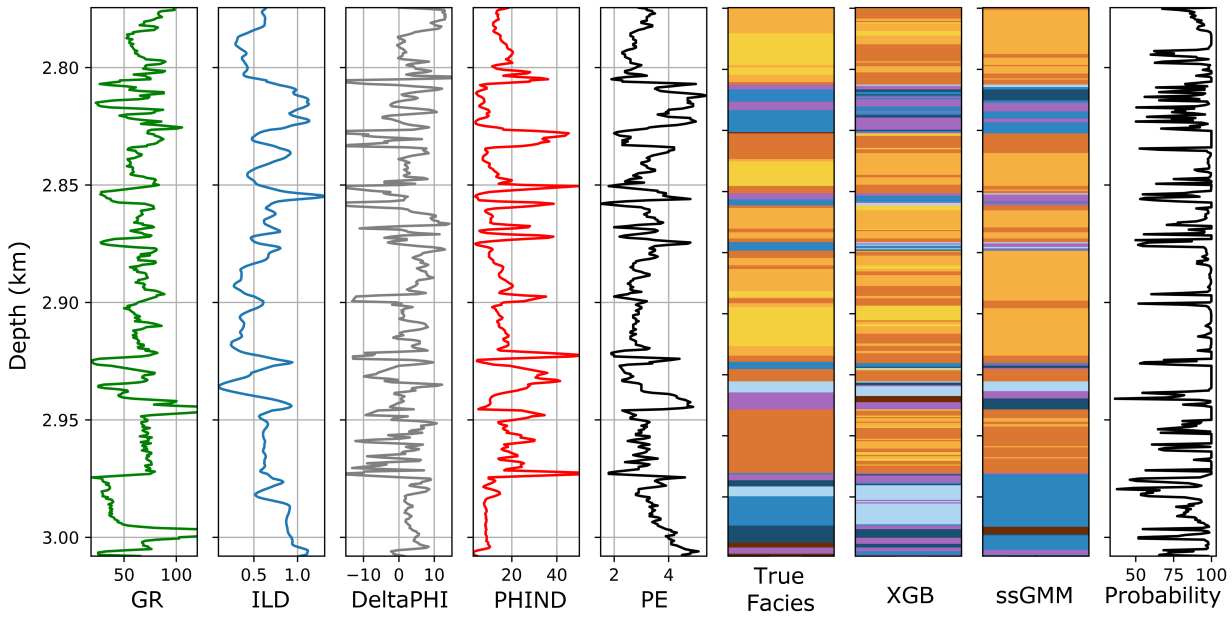

Figure 9: A comparison of the XGB and ssGMM facies predictions to the true facies for the unlabeled well, SHANKLE. For the performance of these models on SHANKLE, see Table 4. The five log variables are shown for reference. The final column gives the probabilities for the ssGMM predicted facies. See Table 1 for the facies colors key.

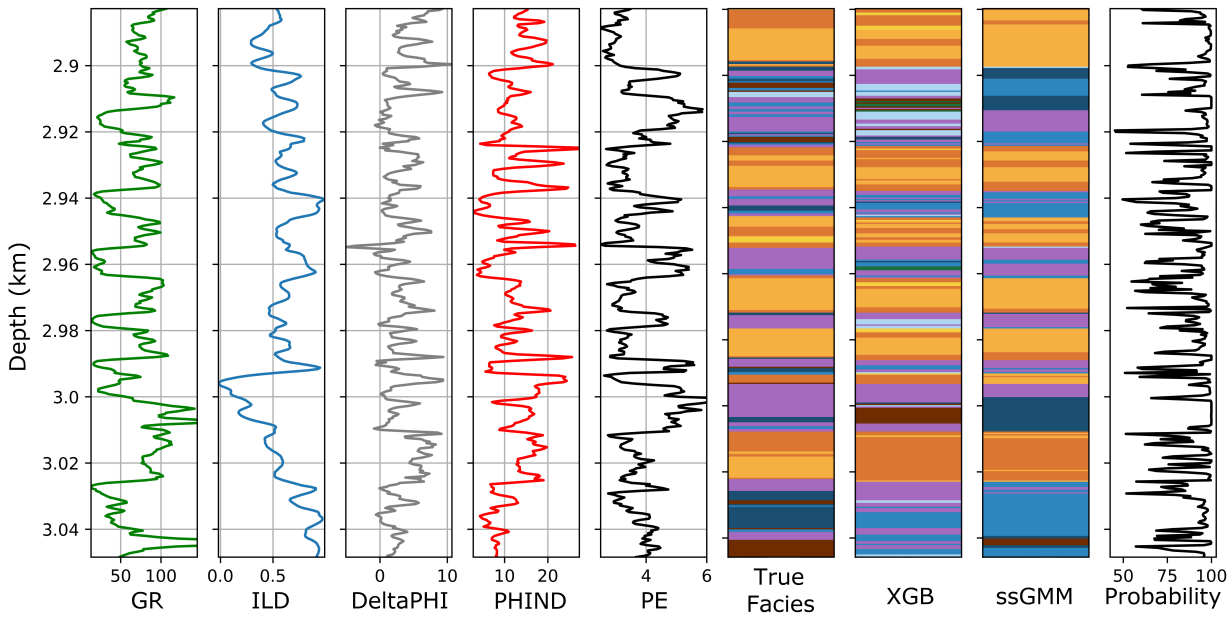

Figure 10: A comparison of the XGB and ssGMM facies predictions to the true facies for the unlabeled well, NOLAN. For the performance of these models on NOLAN, see Table 4. The five log variables are shown for reference. The final column gives the probabilities for the ssGMM predicted facies. See Table 1 for the facies colors key..

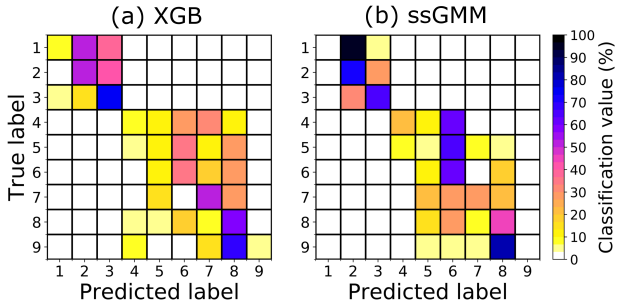

Figure 11: The normalized confusion matrices for the (a) default XGB 5-repeated 5-fold CV model and the (b) ssGMM 5-repeated 5-fold CV model selected using the SMSD score (i.e. the best models for XGB and ssGMM indicated by the * in Table 3). The predictions shown by these matrices are for the nine unlabeled wells. Diagonal cells represent correct classification, off-diagonal cells represent misclassification.