A New Ensemble Machine-Learning Framework for Searching Sweet Spots in Shale Reservoirs

Jizhou Tang*, State Key Laboratory of Marine Geology, Tongji University; **Bo Fan*,** Motorola Solutions; **Lizhi Xiao**** and **Shouceng Tian,** State Key Laboratory of Petroleum Resources and Prospecting, China University of Petroleum, Beijing; **Fengshou Zhang,** Key Laboratory of Geotechnical and Underground Engineering of Ministry of Education, Tongji University; and **Liyuan Zhang** and **David Weitz,** John A. Paulson School of Engineering and Applied Sciences, Harvard University

Summary

Knowing the location of sweet spots benefits the horizontal well drilling and the selection of perforation clusters. Generally, geoscientists determine sweet spots from the well-logging interpretation. In this paper, a group of prevalent classifiers [extreme gradient boosting (XGBoost), unbiased boosting with categorical features (CatBoost), and light gradient boosting machine (LightGBM)] based on gradient-boosting decision trees (GBDTs) are introduced to automatically determine sweet spots based on well-log data sets. Compared with linear support vector machines (SVMs), these robust algorithms can deal with comparative scales of features and learn nonlinear decision boundaries. Moreover, they are less influenced by the presence of outliers. Another prevailing approach, named generative adversarial networks (GANs), is implemented to augment the training data set by using a small number of training samples. An extensive application has been built for the field cases in a certain oilfield. We randomly select 73 horizontal wells for training, and 13 features are chosen from well-log data sets. Compared with conventional SVMs, the agreement rates of interpretation by XGBoost and CatBoost are significantly improved. Without special preprocessing of the input data sets and conditional tabular GAN (CTGAN) model fine tuning, the fake data set could still bring a relatively low agreement rate for all detections. Finally, we propose an ensemble-learning framework concatenating multilevels of classifiers and improve agreement rate. In this paper, we illustrate a new tool for categorizing the reservoir quality by using GBDTs and ensemble models, which further helps search and identify sweet spots automatically. This tool enables us to integrate experts' knowledge to the developed model, identify logging curves more efficiently, and cover more sweet spots during the drilling and completion treatment, which immensely decrease the cost of log interpretation.

Introduction

Rock properties spatially vary from microscale to field scale, which makes the reservoirs heterogeneous (Fanchi 2010). The microscale heterogeneity, including variations in pore size, grain sorting, and impurities distribution, gives rise to a broad range of permeability in different core samples (Satter and Iqbal 2015). In terms of the field-scale heterogeneity, it contains spatial variations of geological components (such as natural fractures, bedding layers, faults, facies, and pinchouts), which have a remarkable influence on identifying the sweet spots and assessing wells placement in the target formation (Suarez-Rivera et al. 2016; Tang et al. 2019; Zhang et al. 2019). In other words, the selection of an optimal drilling location for oil and gas production is significantly determined by the distribution of sweet spots. In the light of petroleum geoscience, sweet spots are regarded as areas of oil and gas reservoirs owning the best production potential (Hauge and Hermansen 2017). Tahmasebi et al. (2017) think that sweet spots represent areas containing high total organic carbon and rocks with high fracability. Aldrich and Seidle (2018) deem that sweet spots are usually referred to as the hydrocarbon regions accounting for critical elements of commerciality and exploration risk. Alqahtani et al. (2018) hold the opinion that sweet spots are the reservoir areas with the highest permeability and oil saturation.

On the basis of the different perspectives previously mentioned, the sweet spot can be defined/categorized into two types. One type is the "geoscience" sweet spot, which can be defined as reservoir areas with the highest permeability, oil saturation, and total organic carbon. The other is the "engineering" sweet spot, defining areas containing rocks with high fracability and brittleness. Finding sweet spots helps achieve the highest productivity indices and recovery factors. Thus, it is of great importance to determine the locations of both "geoscience" and "engineering" sweet spots accurately and effectively. Nevertheless, it is a challenging issue to ascertain their locations without human-made errors according to the lithology, physical properties, electrical properties, fluid properties, rock fracability, and brittleness.

As a critical step of making decisions for the optimum production strategy, several conventional approaches are widely used for sweet-spot delineation in the target reservoirs. All these methods are classified into two groups. Based on static properties (such as geometrical, petrophysical, and spatial data), the first group aims to calculate geo-object properties (such as permeability, porosity, and lithofacies) with consideration of geologic uncertainty or creating a productivity potential map (permeability, porosity, oil saturation, etc.), by means of using a genetic unit approach or productivity proxy function (Alqahtani et al. 2018). The other group captures the dynamic data for generating the quality map of the reservoir rock or maximizing the numerical and the field productivity indices to find optimum well placement.

With the advent of the era of big data, machine-learning techniques occupy a fast-growing field in areas such as information, public security, medical treatment, communication and transportation, agriculture, and image processing. All these approaches take advantage of the processing capacity of modern computers, and their applications also prevail in the petroleum and mining industries (Tahmasebi and Hezarkhani 2012; Jones 2018; Alkinani et al. 2019; Cheung 2020; Rahmanifard and Plaksina 2019). Preliminary results are achieved in sweet-spot detection, well-log interpretation, seismic interpretation, lithology identification, sedimentary facies division, geological modeling, well stimulation evaluation, reservoir simulation, and production forecasting (Fan et al. 2017; Ahmadi et al. 2018;

*Indicates equal contributions.
**Corresponding author; email: xiaolizhi@cup.edu.cn

This paper (SPE 204224) was revised for publication from paper IPTC-19941-Abstract, first presented at the International Petroleum Technology Conference, Dhahran, Kingdom of Saudi Arabia, 13–15 January 2020. Original manuscript received for review 6 March 2020. Revised manuscript received for review 9 August 2020. Paper peer approved 31 August 2020.

Bao et al. 2018; Bhattacharya and Mishra 2018; Kemajou et al. 2019; Liang et al. 2019; Luo et al. 2019; Pan et al. 2019; Saporetti et al. 2019; Silva et al. 2019; Tripoppoom et al. 2019; Wang and Chen 2019; Zhang et al. 2019; Zhou et al. 2020).

Because the detection of sweet spots is regarded as a supervised learning problem, machine-learning-based methodology can be applied to generate corresponding classifiers (Hauge and Hermansen 2017). On the basis of seismic, well logs, and production profiles, Ketineni et al. (2015) propose a comprehensive artificial-neural-network-based reservoir characterization tool capable of creating spatial oil maps for sweet spots. A model integrating with stepwise regression and a machine-learning-assisted tool is developed to deal with large databases (well log data, core data, X-ray diffraction data, etc.) and helps to enhance the possibility of identifying sweet spots (Tahmasebi et al. 2017). In consideration of microbes' sensitivities to vertical microseepage, Te Stroet et al. (2017) put forward a robust and reliable predictive DNA-based model for differentiating sweet spots and low productive areas in shale formations. Based on a support vector machine approach, Qian et al. (2018) establish a machine learning model to achieve a multiattribute prediction of sweet spot location. Tandon (2019) incorporates regression-based machine-learning algorithms into a hydraulic fracturing simulator to improve the assessment of sweet spots in complex reservoirs.

In this paper, we first discuss gradient boosting (GB) algorithms, such as XGBoost, LightGBM, and CatBoost. GB sequentially adds predictors and calibrates previous models. Compared with the linear SVM or random forest (RF), this robust algorithm can deal with comparative scales of the features and learn nonlinear decision boundaries via boosting. Moreover, it is less influenced by the presence of outliers. Then we introduce GANs, which have three types: traditional GANs, conditional GANs (CGANs), and CTGANs. Afterward, we use the field data sets, including approximately 10,000 samples with four different labels (each sample has 13 features from well logging) and conduct experiments to investigate the agreement rates of interpretation by different algorithms. Two types of GANs (CGANs and CTGANs) are applied to generate synthetic data sets. Besides XGBoost, LightGBM, and CatBoost, we also use SVM and multilayer perceptron (MLP) as the basic classifiers to sort the reservoir quality from the field data set. For synthetic data sets generated by GANs, we only used XGBoost for classification. Conclusively, we establish an ensemble-learning framework concatenating multilevels of classifiers to improve the agreement rate.

All these preceding approaches are taken as efficient ways of automatically detecting sweet spots, which help to assist petrophysicists to shorten the delivery time of work. If all features are selected, the agreement rate of interpretation by using CatBoost could reach the highest value of 82.50%, which exceeds 3.15 and 6.69% of those from RF and SVM algorithms, respectively. However, XGBoost ranks first with an 83.37% agreement rate as removing three well-log interpreted features (porosity, permeability, and oil saturation) from the original data set. We also attempt to use the GANs algorithm in combination with XGBoost for data augmentation and training. However, the agreement rates from using synthetic data sets generated by CGANs and CTGANs are 27.44 and 68.58%, respectively, which demonstrates that the GANs-based model may not be suitable for data augmentation on the oilfield data set and probably more parameter tuning work is required to improve the agreement rate of interpretation. Finally, the ensemble-learning framework combining a two-step classification is demonstrated as the best approach of improving the agreement rate.

Methodology

Because it is difficult for SVMs to pick the right kernel, GB reveals robustness to outliers and capability of modeling nonlinear decision boundaries. Also, GB algorithms can save more computational cost compared with SVMs. Thus, we first bring in the GB algorithms including XGBoost, unbiased boosting with categorical features (CatBoost), and LightGBM. Then a GAN is introduced to show its advantage of data augmentation by using a relatively small number of training samples.

GB. The boosting algorithm aims at converting weak learners to a strong learner, which facilitates both the model performance and accuracy (Kearns and Valiant 1994; Breiman 1996; Zhou 2012). In the case of multiclass categorization, this algorithm selects a classifier of a single feature during each iteration by converting the multiclass problem to a binary classification (Torralba et al. 2007). Breiman (1997) proposes the concept of GB because the boosting is regarded as an optimization algorithm on a loss function. The key principle behind the GB algorithm is to find a new submodel to compensate for the residual error created by the previous submodel (Breiman 1997; Mason et al. 1999; Friedman 2001). **Fig. 1** illustrates the schematic diagram of the GB algorithm. As shown in Fig. 1a, the first weak model M_1, labeled with blue color, predicts the output, which has an obvious error with the desired output y_{true}. This error term, named as the first residual γ_1, is determined by Eq. 1:

$$y_{\text{true}} = \alpha_1 M_1(\vec{x}) + \gamma_1, \quad (1)$$

where α_1 denotes the corresponding weight for the first weak model $M_1(\vec{x})$. Then the new weak models (M_2, M_3, M_4) would be generated sequentially based on the residual (γ_1, γ_2, γ_3) of the previous model. Finally, the fourth weak model M_4 is created to predict the residual of the third weak model M_3, which is equivalent to γ_3.

$$\begin{aligned} \gamma_1 &= \alpha_2 M_2(\vec{x}) + \gamma_2 \\ \gamma_2 &= \alpha_3 M_3(\vec{x}) + \gamma_3 \\ \gamma_3 &= \alpha_4 M_4(\vec{x}). \end{aligned} \quad (2)$$

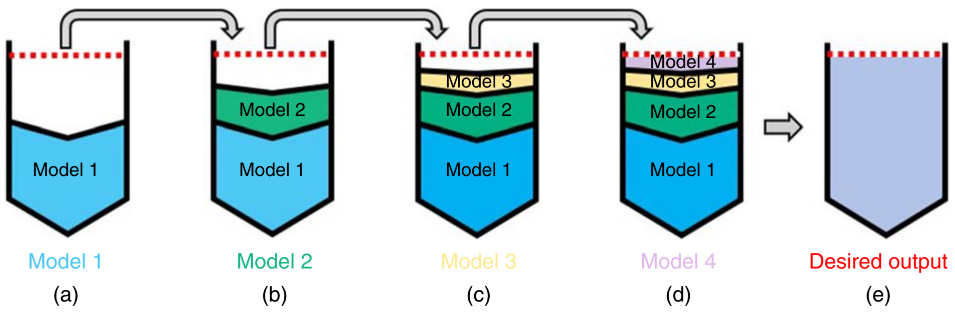


Fig. 1—Schematic diagram of GB algorithm.

Based on Eq. 2, the predicted output would be equal to the predictions of all weak models.

$$y_{\text{pred}} = \sum_{i=1}^{N} \alpha_i M_i(\vec{x}), \qquad (3)$$

where N denotes the number of weak models ($N = 4$ in this example). Then the error loss function can be written as

$$\varepsilon_l = \frac{1}{2}|y_{\text{pred}} - y_{\text{true}}|^2. \qquad (4)$$

After two decades of development, various types of algorithms, such as XGBoost, LightGBM, and CatBoost, have been derived from the original GB algorithm. More details about these algorithms are illustrated in the following subsection.

XGBoost. XGBoost was initially put forward by Chen and He (2014) for supervised learning problems. Decision-tree ensembles, as the default base learners of XGBoost, construct a set of classification and regression trees (Chen 2014; Chen and Guestrin 2016). Each tree grows one after another and presents a distinct prediction score, and the final score can be obtained by summing up the scores of all individual trees **(Fig. 2).**

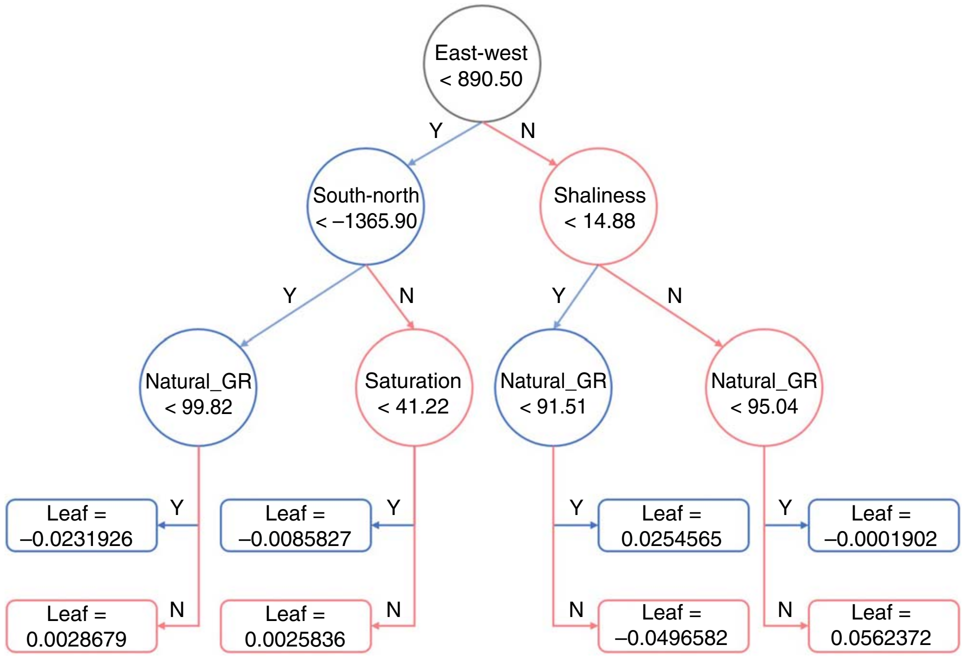


Fig. 2—Structure of a single-tree ensemble model. GR = gamma ray.

In data set D, the number of samples is set to be n. Each sample has m features, and the label is denoted as y. In XGBoost, a tree ensemble model is used by taking the summation of K additive functions, which are applied to predict the label for the ith sample x_i as

$$\hat{y}_i = \phi(x_i) = \sum_{k=1}^{K} f_k(x_i), f_k \in F, \qquad (5)$$

where $F = \{f(x) = w_{q(x)}\}$ represents the space of classification and regression trees and $q : \mathbb{R}^m \to \{1,2,3,\ldots,T\}$ denotes the mapping from the feature space to the leaf nodes' indices of trees. In XGBoost, an objective function consisting of a training loss and a regularization term is required to measure how well the predictive model fits the training data. Thus, the optimal tree structure is obtained by minimizing the objective function

$$L(\phi) = \sum_{i=1}^{n} l(\hat{y}_i, y_i) + \sum_{k=1}^{K} \Omega(f_k)$$
$$\Omega(f) = \gamma T + \frac{1}{2}\lambda\|w\|^2, \qquad (6)$$

where l is a loss function for measuring the difference between the predicted label $\hat{y}_i$ and the ground truth y_i, Ω is a regularization term to avoid overfitting, T represents the number of leaves for each tree, and $w = [w_1, w_2, \ldots, w_T]$ is defined as a vector of scores from T leaf nodes. An additive strategy is applied to fix what we have learned and add a new tree at a time. Let $\hat{y}_i^{(t)}$ be the prediction value from the ith sample under the tth iteration, and the following objective function can be optimized as

$$L^{(t)} = \sum_{i=1}^{n} l\left[y_i, \hat{y}_i^{(t-1)} + f_t(x_i)\right] + \Omega(f_t), \qquad (7)$$

where f_t is a function added greedily to the loss, which significantly improves the model. Using a second-order approximation of the Taylor expansion, the objective function at step t is simplified as

$$\tilde{L}^{(t)} = \sum_{i=1}^{n}\left[g_i f_t(x_i) + \frac{1}{2}h_i f_t^2(x_i)\right] + \Omega(f_t), \qquad (8)$$

where $g_i = \partial_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)})$ and $h_i = \partial^2_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)})$ are first-order and second-order gradients with respect to $\hat{y}^{(t-1)}$.

For problems of multiclass classification or regression, the leaf nodes are assigned with real scores, and each root node has two children. Fig. 2 reveals the learned tree ensemble model for predicting Class 1 vs. the rest based on our selected data set. Each nonleaf node (circle) includes one feature variable based on a splitting rule to decide whether it should go (left or the right branch) from top to bottom. GBDTs algorithms automatically select east-west, south-north, shaliness, natural_gamma ray, and oil saturation as key features for splitting. This figure shows eight leaf nodes (rectangles) in the given decision tree. The predicted score for each sample is normalized through the softmax function before sending it to the objective function. For example, a sample with east-west = 800, south-north = –1400, and natural_gamma ray = 96 can be predicted with the leaf node with the value of –0.023192 (the leftmost leaf node).

For multiclass classification, we adopt the one-against-all approach for XGBoost. Let C be the number of classes and B be the number of subtrees (base learners) used for classifying each class (target class's label is treated as unity; the others are treated as zero). In predicting each class's score for each instance, the scores will be accumulated through all the B subtrees responsible for that class. As shown in **Fig. 3,** we denote $K = B \times C$ trees as the total number of subtrees. The red arrow represents the path of the scores predicted from different classes and subtrees. Based on the multiclass softmax loss, the score for the ith instance of each label in the inference stage can be predicted:

$$p(\hat{y}_i = y_j | x_i) = \frac{e^{\sum_{b=1}^{B} w_{y_j,b}(x_i)}}{\sum_{j=1}^{C} e^{\sum_{b=1}^{B} w_{j,b}(x_i)}}, \qquad (9)$$

where $y_j \in \{1, 2, \ldots, C\}$ and $w_{j,k}(x_i)$ denotes the leaf value when the input is x_i for the kth subtree used for predicting label j. Finally, the predicted label is selected from the largest probability for each class:

$$\tilde{y}_i = \underset{j \in \{1,2,\ldots,C\}}{\operatorname{argmax}} \; p(\hat{y}_i = j | x_i). \qquad (10)$$

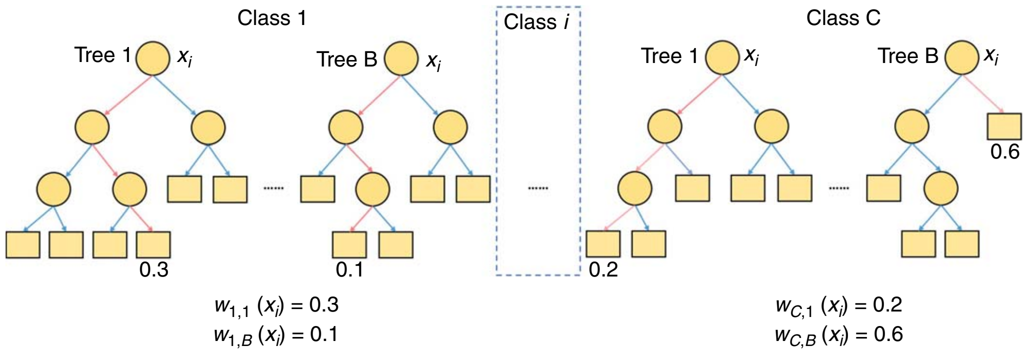


Fig. 3—Example of B × C trees structure for the multiclass classification.

CatBoost. CatBoost (Dorogush et al. 2017; Prokhorenkova et al. 2018), utilizes oblivious trees (Wang et al. 2014) that are balanced to predict labels and helps to reduce the time spent on parameter tuning. Dorogush et al. (2017) point out that CatBoost is developed for categorical features. In this paper, we mainly discuss the improvements of using CatBoost compared with the XGBoost algorithm in solving the biased gradients problem.

Let F_k be the model constructed by the first k trees and $g^k(x_i, y_i)$ be the gradient from the ith training sample with respect to the model F_k. In GBDT-based approaches, gradients $g^k(x_i, y_i)$ are estimated based on (x_i, y_i) to create the tree, which leads to unbiased gradients for the model F_k (Friedman 2001). For each sample x_i, a separate model M_k is trained and updated without using x_i. Then M_k is used to compute the scores of leaves. It can be seen from the following algorithm that M_k is updated and trained without using the data point (x_i, y_i) $(i = k)$.

LightGBM. According to Ke et al. (2017), XGBoost may not work well when the feature dimension is high and data size is huge because it is time-consuming to scan all the possible split points to estimate the information gain. Ke et al. (2017) proposed a gradient-based one-side sampling and exclusive feature bundling. Exclusive feature bundling is mainly developed to deal with sparse features. In our experiments, we do not have too many sparse features. Thus, we mainly focus on gradient-based one-side sampling.

The idea of gradient-based one-side sampling is to improve data sampling quality by reducing the number of instances with small gradients before computing information gain. Because the samples with small gradients are usually well trained to make a significant contribution to the loss, the instances with large gradients are fully sampled. To use this rule, the data points with smaller gradients are randomly sampled, but the larger ones are usually kept. However, this operation leads to the change of data distribution. Therefore, the factor $\alpha = \frac{1-a}{b}$ is used to control the balance between the smaller and larger gradients and make the data distribution unchanged in computing information gain.

GANs. As a prevailing representative of unsupervised machine learning approaches, GANs adopt an indirect training method taking the form of a downstream task over the true and the generated probability distributions, which then enforces the generated distribution closer to the true distribution by training the generative network (Goodfellow et al. 2014). The benefit of using GANs is the data-set augmentation by using a relatively small number of training samples (Goodfellow et al. 2014; Nielsen and Okoniewski 2019).

Fig. 4 illustrates the workflow of GANs. The key components of GANs are generator and discriminator. The yellow arrow represents the forward transform of input variables (blue square) to the new data, and the red arrow denotes the backpropagation of matching error to train the network. The error is defined as the distance between the true distribution (blue square) and the generated distribution (purple square). The generative network attempts to fool the discriminator and aims at maximizing the final classification error. Conversely, the discriminative network minimizes the final classification error by identifying the fake generated data. It is worth mentioning that two different neural networks can be jointly trained, and their corresponding weights are updated at each iteration. Ultimately, a classifier (red dashed curve, a discriminator only distinguishes real from fake data) is established after the training process.

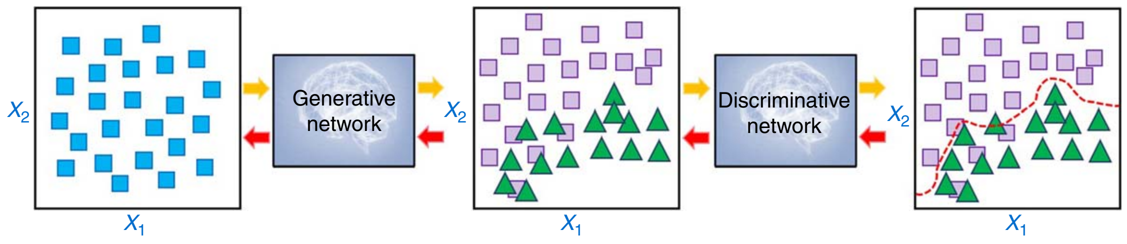


Fig. 4—Schematic diagram of the GAN.

CGANs. In GANs **(Fig. 5a),** the discriminative model D is applied to discriminate the difference between the true data and the generated data. Concurrently, the generative model G aims at fooling the discriminator by producing more and more realistic data. The error of the generator should be maximized, while the cost function of the discriminator should be minimized. Thus, it gives

$$\begin{aligned}\max_G\left[\min_D E(G,D)\right] &= \max_G\left(\min_D\left\{\frac{1}{2}\Psi_{z\sim p_N}\cdot\{1-D[G(z)]\}+\frac{1}{2}\Psi_{x\sim p_G}\cdot[D(x)]\right\}\right)\\ &= \max_G\left(\min_D\left\{\frac{1}{2}\Psi_{x\sim p_R}\cdot[1-D(x)]+\frac{1}{2}\Psi_{x\sim p_G}\cdot[D(x)]\right\}\right)\\ &= \max_G\left(\frac{1}{2}\min_D\int_{\mathbb{R}}\{p_R(x)\cdot[1-D(x)]+p_G(x)\cdot[D(x)]\}dx\right), \quad\quad (11)\end{aligned}$$

where, p_G and p_R represent the corresponding probability density of the generated data and the real data, respectively; and p_N denotes a prior noise distribution. Because the generative model G should be maximized, we have

$$\max_G\left(\int_{\mathbb{R}}\left\{p_R(x)\cdot\left[1-D_G^*(x)\right]+p_G(x)\cdot\left[D_G^*(x)\right]\right\}dx\right)=\max_G\left(\int_{\mathbb{R}}\left\{\min_D\left[p_R(x),p_G(x)\right]\right\}dx\right). \quad\quad (12)$$

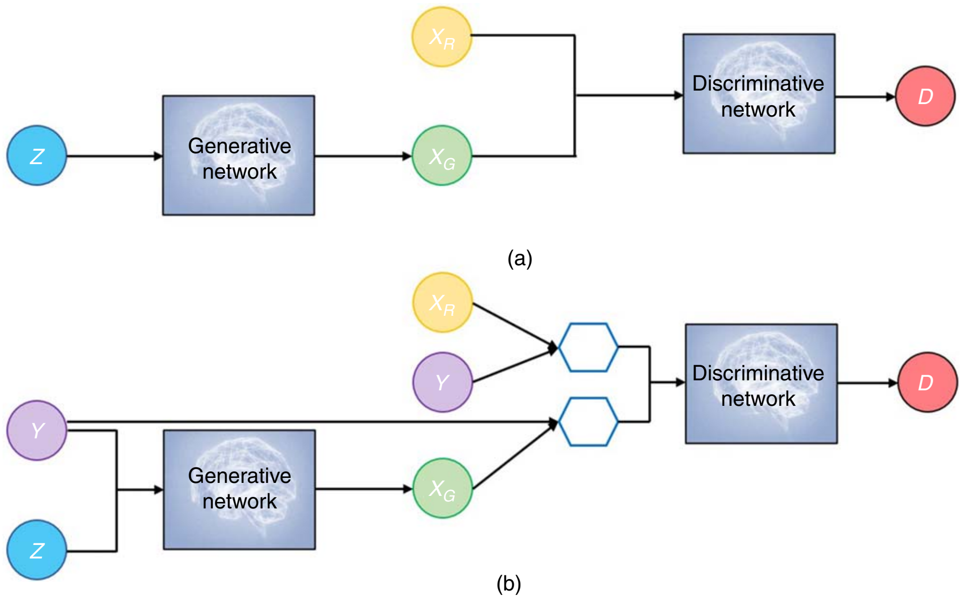


Fig. 5—(a) GANs architecture and (b) CGANs architecture.

Because the probability density $p_R(x)$ is independent of the generative model G, it should satisfy that

$$p_G(x)\geq p_R(x). \quad\quad (13)$$

Finally, the maximized G can be written as

$$\max_G \left(\frac{1}{2} \int_{\mathbb{R}} \left\{ \min_D [p_R(x), p_G(x)] \right\} dx \right) = \int_{\mathbb{R}} \left\{ \frac{\min_D [p_R(x), p_G(x)]}{p_R(x) + p_G(x)} \cdot \frac{p_R(x) + p_G(x)}{2} \right\} dx. \quad (14)$$

CGANs were introduced by Mirza and Osindero (2014) to add extra information to make the generator and discriminator conditioned. This kind of extra information can be regarded as the additional input layer (labeled as purple-colored "Y" in Fig. 5b), such as class labels or data from other modalities. We can formulate the cost function of CGANs as

$$\begin{aligned} \max_G \left[\min_D E(G,D) \right] &= \max_G \left\{ \min_D \frac{1}{2} \Psi_{x \sim p_R} \cdot [1 - D(x|y)] + \frac{1}{2} \Psi_{x \sim p_G} \cdot [D(x|y)] \right\} \\ &= \max_G \left(\frac{1}{2} \min_D \int_R \{ p_R(x) \cdot [1 - D(x|y)] + p_G(x) \cdot [D(x|y)] \} dx \right), \end{aligned} \quad (15)$$

where y denotes the conditional information, which facilitates both the generative model and discriminative model to operate in certain modes.

The specific architectures of CGAN's generator and discriminator are shown in **Fig. 6.** EMB represents an embedding layer of which each label is first converted based on the embedding layer and then concatenated with a latent variable z or the original features x. $\oplus$ is a concatenation operation, and the size of each feature vector is also included. For example, in the CGAN's generator, the dimensions of each feature vector are 10 (four from the conditional information y and six from the latent variables z), 4, 8, 16, and 13. Likewise, for its discriminator, the corresponding dimensions are 17 (4 from the label y and 13 from the original feature space), 4, 8, 8, and 1, respectively. The other operations for each layer are: fully connected layer (F), batch normalization (B), leaky Relu (L), dropout (D), and tanh (T). The combinations of different operations are represented by just concatenating their first capital letters together. For example, "FBL" means a fully connected layer followed by batch normalization and leaky Relu. "FDL" means a fully connected layer followed by drop out and leaky Relu. "FT" means a fully connected layer followed by the tanh activation layer.

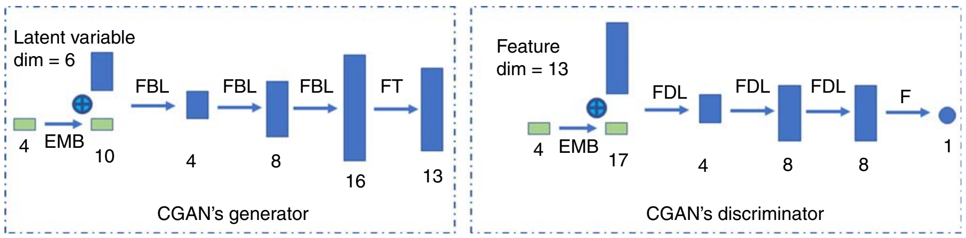


Fig. 6—Specific structures of CGAN's generator and discriminator.

CTGANs. Traditional GANs or CGANs assume continuous variables to be Gaussian, which may lead to the failure in modeling the tabular data when the continuous variables are non-Gaussian, or the data are imbalanced for discrete variables (categorical features). To deal with the data set of a mixed type including continuous and discrete variables, a CTGAN (Xu and Veeramachaneni 2019) is proposed, which is based on a CGAN and PacGAN (Lin et al. 2018). Before sending the raw table to CGAN model, the normalization should be done. A variational Gaussian mixture model is first adopted to represent the distribution of the continuous variable.

$$P_{C_i}(c_{i,j}) = \sum_{k=1}^{m_i} u^{(k)} N\left[c_{i,j}; \eta_i^{(k)}, \phi_i^{(k)}\right], \quad (16)$$

where $c_{i,j}$ is the value of the ith column and jth row from the input training table, and C_i is the vector from the ith column. $u^{(k)}$, $\eta_i^{(k)}$, and $\phi_i^{(k)}$ are the weights, means, and standard deviations from the kth component. Then a one-hot encoded vector $\beta_{i,j}$ is computed by sampling from the categorical probability mass function of k with parameters $\beta_{i,j}^{(1)}, \beta_{i,j}^{(2)}, \ldots, \beta_{i,j}^{(m_i)}$ as follows:

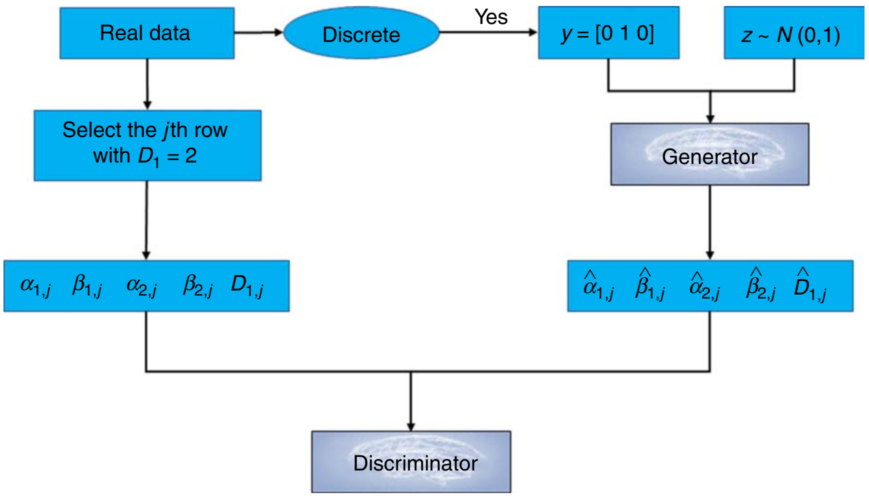
$$\text{Cat}\left[k; \beta_{i,j}^{(1)}, \beta_{i,j}^{(2)}, \ldots, \beta_{i,j}^{(m_i)}\right], \text{ where } \beta_{i,j}^{(k)} = u^{(k)} N\left[c_{i,j}; \eta_i^{(k)}, \phi_i^{(k)}\right] \Big/ P_{C_i}(c_{i,j}). \quad (17)$$

Then $c_{i,j}$ is normalized as

$$\alpha_{i,j} = \frac{c_{i,j} - \eta_i^{(k)}}{4\phi_i^{(k)}}. \quad (18)$$

One-hot encoding and concatenation are also used for multiple discrete variables' representations. For example, if there is one discrete variable D_1 with three categories, and the second category is selected as a conditional label, the corresponding one-hot encoded vector is [0 1 0]. In our problem, we only have one discrete variable, of which the labels are categorical features. The flow chart of the CTGAN model with two continuous variables and one categorical variable (label) are shown in **Fig. 7.** As shown in this figure, the CTGANs model generates fake data according to a specific label (a condition). At the same time, the real data set corresponding to that label is also selected and normalized to send to the discriminator.

The specific structures of the CTGAN generator and discriminator are shown in **Fig. 8.** The input vector of the generator includes the label (size is 4) and the hidden vector (size is 128). After the FBL layer (fully connected layer + batch normalization + leaky Relu), a feature vector of size 256 is generated and concatenated with the input feature vector as a new vector (size is 388) sent to the second FBL layer. Then the second feature map with a size of 256 is produced and concatenated with the feature vector from its input. Finally, the output vector of size 136 is generated using the fully connected layer. The output vector's size is calculated according to the number of modes for each continuous variable and the one-hot encoded vector's dimension for each discrete variable, finally consisting of α, β, and D. For the discriminator of CTGANs, its input can be the output from the generator or the real data set's α, β, and D values sampled multiple times based on the number of PAC size. After using two FLD (fully connected layer + leaky Relu + dropout) layers, one can decide whether the input is real or not. More details can be found from Xu and Veeramachaneni (2019).



Real data
Discrete
Yes
$y = [0\ 1\ 0]$
$z \sim N(0,1)$
Select the jth row with $D_1 = 2$
Generator
$\alpha_{1,j}$ $\beta_{1,j}$ $\alpha_{2,j}$ $\beta_{2,j}$ $D_{1,j}$
$\hat{\alpha}_{1,j}$ $\hat{\beta}_{1,j}$ $\hat{\alpha}_{2,j}$ $\hat{\beta}_{2,j}$ $\hat{D}_{1,j}$
Discriminator

Fig. 7—Flow chart of the CTGANs model.

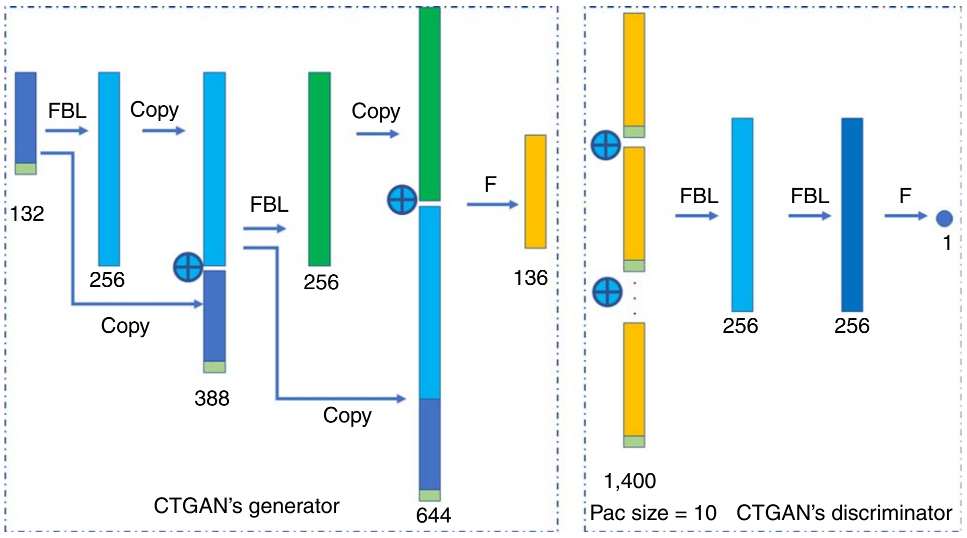


Fig. 8—Specific structures of CTGAN's generator and discriminator.

Experimental Results

Data Preparation. We randomly select 73 horizontal wells from a certain oil field for training purposes. Our selected formation has a large thickness, areal continuity, laminated siltstone, and shale formations, with primary features as extremely low permeability, low porosity, high oil saturation, strong heterogeneity, and high brittleness index. In each well, tens to hundreds of well logs are collected. As we know, well logs indicate single-point measurements of physical properties, which are aperiodic and dependent on aspects, such as mineral composition or lithology, presence of fluids, resistivity and conductivity, porosity and permeability, and cementation and compaction. In our case, each well-log data set contains 13 features, including deep resistivity, interval transit time, density, natural_ gamma ray, shaliness, orientation, deviation, depth, east-west, south-north, porosity, permeability, and oil saturation. All these features are continuous, and no categorical features (discrete values) are used in this paper. Statistics analysis (mean, standard deviation, minimum and maximum) of these features are shown in **Table 1.**

Initially, a preprocessing pipeline is performed to filter out the well logs without any value to experts or machine-guided reservoir quality classification. After data cleaning, physical parameter (i.e., interval transit time), lithological parameter (i.e., shaliness), and electrical parameter (i.e., deep resistivity) are adopted to establish a comprehensive index (CI), which can reflect the formation characteristics.

$$\mathrm{CI} = (AC - AC_L) \times (Vsh_U - Vsh) \times \log_{10}(R_T), \quad (19)$$

where AC and AC_L denote the interval transit time and its lower bound, respectively; Vsh and Vsh_U represent shaliness and its upper bound; and R_T is the deep resistivity. Then the comprehensive index, brittleness index, interval transit time, shaliness, porosity, and permeability are used to categorized the reservoir quality into four types: water formation, oil formation, dry formation, and bad formation with corresponding labels as L_0, L_1, L_2, L_3. After interpretation and labeling, we train our selected algorithm with a data set generated via interactive labeling using human expert knowledge. The data sets include almost 10,000 samples with labels. In our experiments, 80% data sets are used for training and validation, and the others for testing.

Feature Name	Mean	Standard Deviation	Minimum	Maximum
Deep_resistivity ($\Omega \cdot$ m)	63.186356	106.455652	3.44	6220.62
Interval_transit_time (μs/m)	213.783843	14.513890	98.990000	309.92
Density (g/cm^3)	2.541494	0.072679	1.68	2.9
Natural_gamma ray (API)	86.237089	19.551168	20.2	145.95
Shaliness (%)	16.291963	5.206002	3.52	42.69
Orientation (degrees)	227.245555	97.614673	0.270	362.19
Deviation (degrees)	65.454329	32.824882	0.64	92.72
Depth (m)	1856.196817	266.654981	1015.36	2209.78
East-west (m)	12.589112	406.070219	–1748.42	1436.13
South-north (m)	–75.88483	938.821799	–2567.46	2889.78
Porosity (%)	9.809155	2.785541	0.810000	25.200000
Permeability (md)	1.153005	3.313125	0.010000	55.680000
Oil_saturation (%)	42.906673	16.418168	0.100000	85.880000

Table 1—Statistics analysis of 13 features from well logging.

Data Visualization. In this section, we choose six features (permeability, porosity, oil saturation, shaliness, natural_gamma ray, sonic travel time) and do 3D visualization for each feature after normalization **(Figs. 9a through 9f).** From Fig. 9a, we can observe that the distribution of normalized permeability in the shallow layer is much larger than that in the deep area. This is because the effect of formation compaction is positively correlated with formation depth, which would make the size of pores and throats much smaller in deep areas and further reduce the permeability. Fig. 9b reveals the negative correlation between the normalized porosity and the formation depth. Shales tend to have a lower porosity compared to sands, while density, Poisson's ratio, and velocity values are much higher (Bougher 2016). In Fig. 9c, the normalized oil saturation becomes much higher in the deep layer, which indicates that the area is the productive zone. Shaliness represents the shale content in a dominantly nonshale formation. From Fig. 9d, we could observe that there is no obvious difference of shaliness in the shallow layer and deep layer. Natural_gamma ray (GR), as a measured value of gamma-ray radiation, can be used to identify the rock or sediment in the borehole. Because shale usually emits more gamma rays than other sedimentary rocks, we could find that shale content dominates in the deep formation, as illustrated in Fig. 9e.

Moreover, the value of natural_gamma ray is positively related to the content of organic matters, which could explain why the deep formation covering the productive zone owns a higher value of gamma ray. Interval transit time, as the reciprocal of wave velocity, which is measured by acoustic log, reflects the amount of time for an elastic wave to travel a certain distance. This parameter can be used for calibrating seismic data and determining the formation porosity. According to Fig. 9f, we could obtain the information that the interval transit time has a negative correlation with the formation depth. This is because much more unconsolidated shales are distributed in the shallow layer, which contributes a higher transit time of P-wave.

Parameters Tuning. We mainly use two data sets: the real data sets and the synthetic data sets generated by two types of GANs (CGANs and CTGANs) for classification. For classification on real data sets, besides XGBoost, LightGBM, and CatBoost, we also apply SVMs and MLP. For synthetic data, we only use XGBoost. In all the following experiments, we ran a grid search to find the best hyperparameters and used fivefold cross-validation to compute the average score (agreement rate of interpretation) from each set of hyperparameters. The best parameters are chosen from the model giving the highest average agreement rate on the validation sets. **Table 2** reveals the parameter settings for different methods.

For SVMs, the best agreement rate of interpretation can be obtained by using the linear kernel. We use MLP with two and three hidden layers. For the MLP with two hidden layers, the number of neurons in the first and the second hidden layers are set in the range of (3, 40) and (2, 35). For the MLP with three hidden layers, the number of neurons in the first, second, and third layers are separately set as (3, 40), (2, 28), and (2, 25). The maximum agreement rate for the MLP with two hidden layers is 64%, which is achieved when the first layer and second layer have 37 and 10 neurons. For the three hidden layers in MLP, the best agreement rate is 68% with 14, 25, and 20 neurons for each hidden layer.

There are more than 20 hyperparameters for the tree-based model. To reduce the searching space of the hyperparameters, we only tune the significant ones. Because XBGoost, LightGBM, and CatBoost have similar hyperparameters, we use the same searching spaces for them, for a fair comparison.

In XGBoost, LightGBM, and CatBoost, "max depth" determines the maximum depths the tree can grow for each model, which is set from 4 to 10 with 2 as the interval. "Gamma" is the minimum loss reduction required to split a leaf node, which is usually better than max depth in controlling overfitting. It is set in (0, 2). "Minimum child weight" defines the minimum sum of weights of all observations required in a child, and it is set between 1 and 20. "Subsample" denotes the fraction of observations that is randomly sampled from each tree, which can also be used to handle overfitting, chosen between 0.7 and 1. "Col. sample by tree" denotes the fraction of columns that is randomly sampled from each tree in the range of (0.3, 0.9). "Scale positive weight" controls the balance of positive and negative weights. Usually, it is set as sum(neg) (summation of the number of negative samples) divided by sum(pos) (summation of the number of positive samples). In this paper, they are chosen from 0.5 to 2 with 0.5 as the interval. It is worth noting that "Step Size" is the sampling interval from a given range we use in selecting parameters.

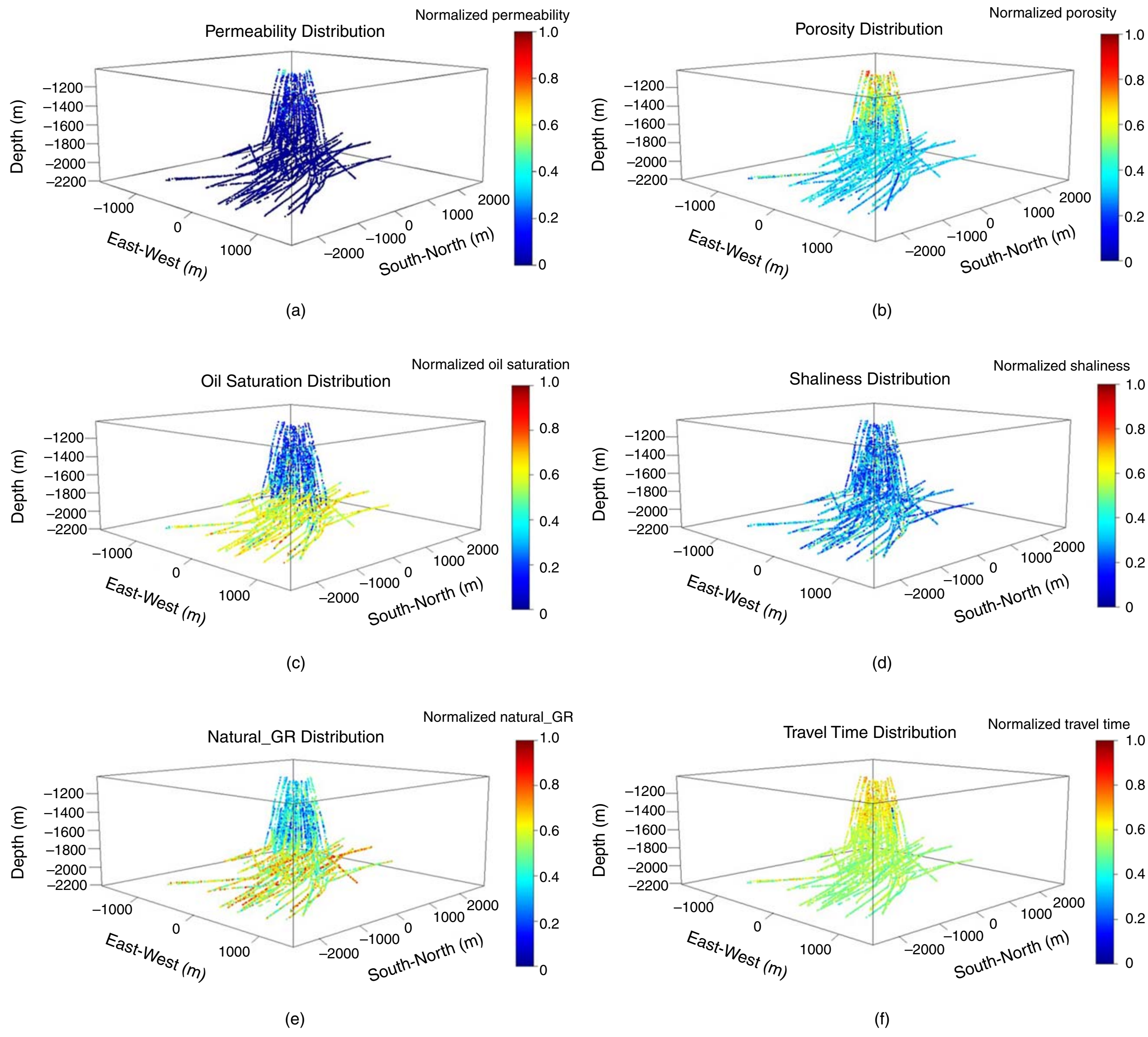


Fig. 9—Distribution of normalized (a) permeability, (b) porosity, (c) oil saturation, (d) shaliness, (e) natural_GR, and (f) interval transit time.

Method	Name of Parameters	Range of Parameters	Step Size
MLP two hidden layers	First layer's number of neurons	(3, 40)	1
	Second layer's number of neurons	(2, 35)	1
MLP three hidden layers	First layer's number of neurons	(3, 40)	1
	Second layer's number of neurons	(2, 28)	1
	Third layer's number of neurons	(2, 25)	1
XGBoost/LightGBM/CatBoost	Max depth	(4, 10)	2
	Gamma	(0, 2)	1
	Minimum child weight	(1, 20)	1
	Subsample	(0.7, 1.0)	0.1
	Column sample by tree	(0.3, 0.9)	0.1
	Scale positive weight	(0.5, 2)	0.5

Table 2—Parameter settings for different methods (MLP, XGBoost, LightGBM, CatBoost).

For both CGAN and CTGAN, we use adaptive moment optimization in training. The total number of epochs is 2,000, and the batch size is 128. We set 0.0002 as the learning rate, with hyperparameters beta1 as 0.5 and beta2 as 0.999, which are used for computing moving averages. Dropout, with a probability of 0.5, is applied to the generator and the discriminator.

It is worth noting that two types of hyperparameters have significant impacts on the experimental results. The first type of parameter aims at defining the shape of the tree model (such as max depth and number of child nodes) and overcoming the overfitting issue by using tree pruning and subsampling (such as "gamma," "minimum child weight," and "subsample"). The second type of parameter, such as learning rate and batch size, is mainly used for optimization, which helps us to learn the weights (like the scores of the leaves) of the model. Besides, the meta learner is mainly applied to control the contribution from each base learner. If the meta learner is an MLP or a tree-based model, the tuning of the hyperparameters is still similar to the base learner.

Prediction Results. The agreement rate of different categories (L_0, L_1, L_2, L_3) from different classifiers is shown in **Fig. 10 and Table 3.** In our experiments, we select all features (13 features) or partial features (10 features) for training and testing. As depicted in Fig. 10, we have nine models as XGBoost (13f_xgb), RF (13f_rf), SVM with linear kernel (13f_svm), MLP with two hidden layers (13f_mlp2), MLP with three hidden layers (13f_mlp3), LightGBM (13f_lgb), CatBoost (13f_catb), CGANs with XGBoost (cgan_xgb), and CTGANs with XGBoost (ctgan_xgb). For the remaining two models, XGBoost (10f_xgb) and SVM with the linear kernel (10f_svm), we remove highly correlated features (orientation, porosity, and depths) and keep the remaining 10 features for training. In terms of category L_0 (water formation), except CGANs and CTGANs, other eight base learners have a high agreement rate of interpretation (more than 80%), which demonstrates that GANs would have low accuracy in predicting the reservoir quality and more parameter tuning work is required to improve the agreement rate. From Fig. 10, we also observe that GB algorithms (XGBoost, CatBoost, and LightGBM) have both high agreement rate and robustness to deal with a random numbers of large-scale data sets for each label, compared with traditional SVMs and MLPs.

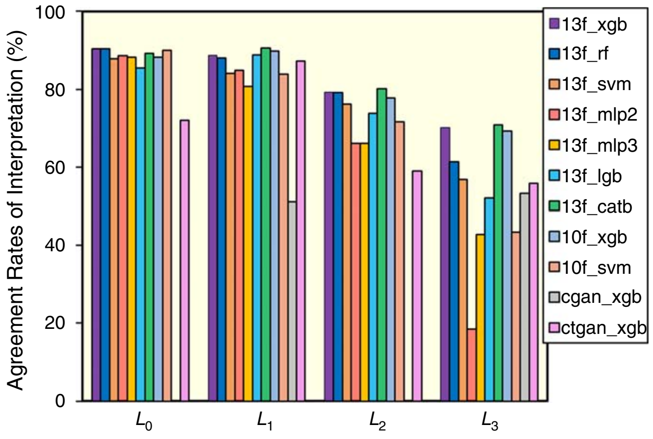


Fig. 10—Agreement rates of interpretation of different labels for different models.

Features	Model	L_0 (%)	L_1 (%)	L_2 (%)	L_3 (%)	Total Agreement Rate (%)
All features	XGBoost	90.34	88.66	79.15	70.19	81.61
All features	RF	90.34	88.09	79.15	61.27	79.35
All features	SVM linear kernel	87.93	84.12	76.15	56.81	75.81
All features	MLP (two hidden layers)	88.62	84.88	66.08	18.54	64
All features	MLP (three hidden layers)	88.28	80.72	66.08	42.72	68.42
All features	LightGBM	85.52	88.85	73.85	52.11	74.99
All features	CatBoost	89.31	90.55	80.21	70.89	82.5
10 features (remove highly correlated features)	XGBoost	88.28	89.79	77.74	69.25	80.95
10 features (remove highly correlated features)	SVMs linear kernel	90	83.93	71.55	43.43	71.51
All features (fake data from CGANs)	XGBoost	0	51.04	0	53.29	27.44
All features (fake data from CTGANs)	XGBoost	72.07	87.15	59.01	55.87	68.58

Table 3—Agreement rates of interpretation of different labels from 11 classifiers.

As shown in **Fig. 11** and Table 3, CatBoost produces the highest total agreement rate of 82.5%, and XBGoost is the second best with an agreement rate of 81.61%. However, the maximum agreement rate from LightGBM is only 74.99%, which is 6.62% lower than the results from XBGoost. The main reason may be that the number of training samples is not large enough in our problem. Empirically, one needs to use more than 10,000 samples for training LightGBM. SVM reveals better results than LightGBM by 0.82%, but it takes a longer time to train. The time complexity of SVM is between $O(n^2)$ and $O(n^3)$, where n is the number of training instances.

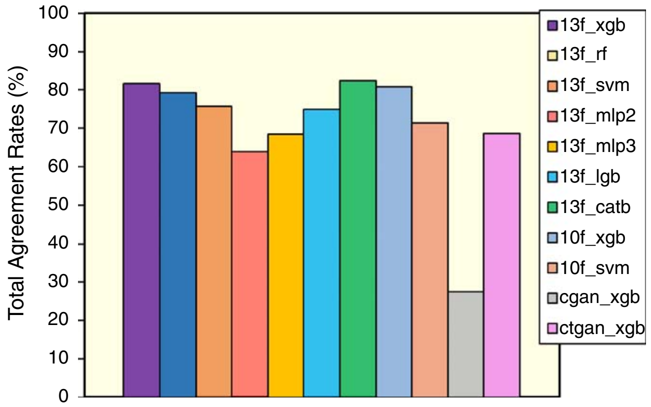


Fig. 11—Total agreement rates of interpretation for different models.

For the GAN-based model, we do data augmentation, but it turns out that this model would reduce the data quality. GANs and CGANs are not able to generate data sets similar to the input table. Many zeros appear in the Cols. of depth and shaliness, and the data range is not similar to the one from the training data. What is worse, GANs does not use the labels as conditions, so we should generate the training data for each label, which is time-consuming. Therefore, we only reported the interpretation results on the test data sets from CGANs and CTGANs. We use them to generate two fake tables with 8,500 samples for each, and the agreement rate of CGANs and CTGANs are 27.44 and 68.58%, respectively, by training on the fake data but testing on the real test data. Although the agreement rate of interpretation is low, it is much better than GANs and CGANs.

Ensemble-Learning Framework

Agreement Rates of Interpretation from Different Ensemble-Learning Approaches. The other way of improving the agreement rate of interpretation is to generate more sophisticated features and feed them to a meta learner, of which the new features can be the concatenation of the outputs from each model or just the summation of them. This type of architecture is called the ensemble-learning framework, as shown in **Fig. 12.** Because averaging and voting-based ensemble models can be divided into two steps (put averaging and voting operations in the second level), we treat them as a two-step classification just as the stacking model, which is capable of concatenating multilevels and boosting the model performance. In our experiments, we use two levels. Let $f_1, f_2, \ldots, f_B$ be the predictions from B base learners on the first level. To combine the output values from multiple base learners, we consider two types of feature combinations. The first type is to concatenate all the feature vectors, and the second is to add all features elementwise. Hence, the combined features as inputs to the second level classifier can be formulated as

$$g(f_1 \oplus f_2 \oplus ..f_B; w)\text{or}\left(\sum_{i=1}^{B} f_i, w\right), \quad \ldots\ldots \quad (20)$$

where g is a mapping from the output of base learners to the final predictions in the second level, and $\oplus$ denotes the concatenation operation. The parameter w of the stacked model is learned by training a meta learner. During the training process of our selected stacking model, we divide it into two stages. In the first level, we can use classifiers from XGBoost, LightGBM, CatBoost, and SVMs. The outputs for each classifier can be probabilities for each class or just the predicted labels, which can be added together or concatenated to send to the second level meta learner. To find the best models for each base learner, we employ the k-fold cross-validation and grid search in the first stage. By fixing the base learner's hyperparameters, we make use of the training data set to get all the predicted scores. After feature combination, they become new features and are fed to the next stage. In the second level, we use MLP with four hidden layers. The number of neurons for each layer are selected from [2, 12], [2, 11], [2, 10], and [2, 9]. The training of the second level is nearly the same as the first level except for the training data set, which are changed to the features from the first level's outputs. The predictions on the test sets for the whole stacking model are achieved from the model, which generates the highest cross-validation score.

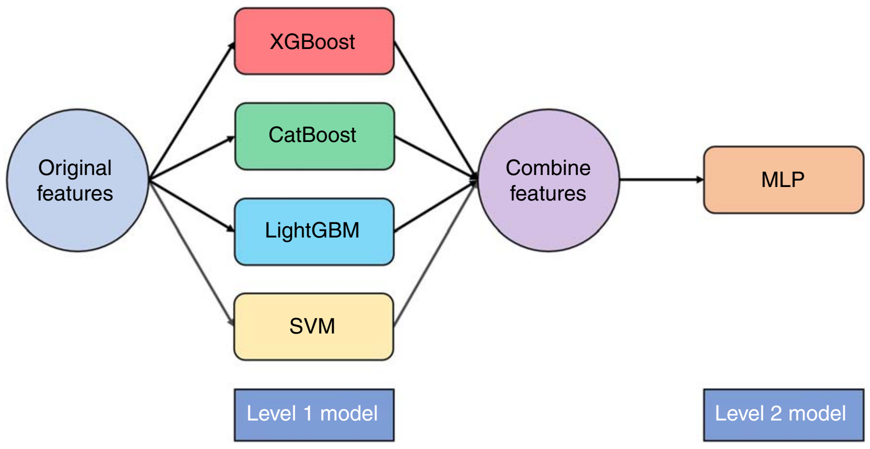


Fig. 12—Structure of an ensemble-learning framework.

Because the features with reduced dimensions do not improve the results (as shown in Table 3), we use all the features in the ensemble experiments. According to **Table 4,** for the voting strategies, no matter which base learners are combined, it cannot outperform the base learner with the highest agreement rate. For averaging, when the base learners are XGBoost and CatBoost, we can get an agreement rate of 82.61%, which is 0.11% higher than the CatBoost model. When we use MLP as the meta learner in the second level, it can mostly improve the agreement rate compared to that of only using one base learner. Specifically, when setting XGBoost, CatBoost, and SVMs as the base learners, the ensemble-learning framework improves the interpretation results by 1.77%, and the final agreement rate is 82.77%. Especially when combing XGBoost and CatBoost, we improve it by 0.66% compared to CatBoost. Correspondingly, the agreement rate of interpretation is 83.16%, which is the highest agreement rate from both ensemble modeling and single model prediction.

Features	Level 1 Model	Level 2 Features	Level 2 Model	Agreement Rate of Interpretation (%)
All	XGBoost-LightGBM-SVM	Class probabilities' sum	Average	80.95
All	XGBoost-LightGBM-SVM	Predicted labels	Voting	78.8
All	XGBoost-LightGBM-SVM	Class probabilities' sum	MLP (four layers)	81.28
All	XGBoost-LightGBM-SVM	Class probabilities' concatenation	MLP (two and three layers)	82.61
All	XGBoost-LightGBM-SVM	Class probabilities' concatenation	XGBoost	81.45
All	XGBoost-LightGBM-SVM	Class probabilities' sum	XGBoost	80.18
All	XGBoost-LightGBM-SVM	Class probabilities' sum	Averaging	82.11
All	XGBoost-LightGBM-SVM	Predicted labels	Voting	82.16
All	XGBoost-LightGBM-SVM	Class probabilities' concatenation	MLP	82.77
All	XGBoost-LightGBM-SVM	Class probabilities' sum	MLP	82.55
All	XGBoost-CatBoost	Class probabilities' sum	Averaging	82.61
All	XGBoost-CatBoost	Predicted labels	Voting	82.05
All	XGBoost-CatBoost	Class probabilities' concatenation	MLP	83.05
All	XGBoost-CatBoost	Class probabilities' sum	MLP	83.16

Table 4—Interpretation results by using ensemble-learning approach.

It is worth noting that porosity, permeability, and oil saturation (PPS) are parameters obtained from well-log interpretation. Thus, these three parameters are recommended to remove from the original data set to reduce the error margin generated by the machine-learning-based model. Then the remaining 10 parameters are used to determine the agreement rate of interpretation by different single classifiers and ensemble models. The interpretation results from both the base models and the ensemble models are shown in **Tables 5 and 6,** respectively. From the interpretation result of different base models, it can be observed that only three of them (XGBoost, CatBoost, RF) reach the agreement rates of more than 80%. The highest agreement rate of 83.37% can be obtained by using XGBoost. CatBoost ranks second with 82.36% agreement rate. Compared with the GB algorithm, MLP shows an undesired result. Thus, it can be concluded that GB (XGBoost and CatBoost) presents the best performance of interpretation. In the ensemble-learning model, XGBoost and CatBoost are adopted as base learners, and MLP is the meta learner for the second level. According to the interpretation results from ensemble model, we find that the total agreement rate can reach 84.48% by using the first concatenation approach, which exceeds 1.11% of the agreement rate from a single XGBoost model.

Features	Model	L_1 (%)	L_2 (%)	L_3 (%)	Total Agreement Rate (%)
10 features (remove PPS)	XGBoost	92.16	78.39	76.21	83.37
10 features (remove PPS)	CatBoost	91.91	76.92	74.60	82.36
10 features (remove PPS)	LightGBM	91.42	68.86	57.56	74.60
10 features (remove PPS)	MLP (two hidden layers)	78.19	68.50	51.44	67.14
10 features (remove PPS)	MLP (three hidden layers)	87.75	71.06	55.63	73.08
10 features (remove PPS)	SVM linear	88.97	80.22	58.84	77.12
10 features (remove PPS)	RF	91.18	79.85	68.49	80.95

Table 5—Agreement rates of different labels from base models after removing PPS data.

Features	Level 1 Model	Level 2 Model	L_1 (%)	L_2 (%)	L_3 (%)	Total Agreement Rate (%)
10 features (remove PPS)	XGBoost-CatBoost	MLP + concatenation (four layers)	92.89	83.52	74.28	84.48
10 features (remove PPS)	XGBoost-CatBoost	MLP + accumulation sum	93.38	84.62	72.03	84.27

Table 6—Agreement rates of different labels from ensemble models after removing PPS data.

Case Study by Using Ensemble-Learning Method. One well is selected from 73 horizontal wells for the case study. We screen out the candidate intervals with a deviation angle of approximately 90° from the original data set, and more than 5,000 samples remain for the classification procedure. For our selected wellbore, 84 samples of intervals are preserved to ensure that our studied intervals locate at the horizontal region. The measure depth of selected intervals ranges from 2100 to 3700 m. Then we investigate the detection performance of sweet spots for the horizontal intervals by using the ensemble-learning approach. **Fig. 13** elaborates profiles of log responses and corresponding prediction scores under two different types of variable selection approaches. All 13 well-log features are chosen in Fig. 13a, and only 10 well-log parameters are used in Fig. 13b. The reason why PPS are removed from the case in Fig. 13b is that these three key reservoir parameters can directly be obtained from well logging interpretation. Error margins would be generated during the calculation procedure of these variables. Therefore, PPS should be deleted, and the remaining 10 variables can be used to determine the agreement rate of interpretation by ensemble-learning framework. In Fig. 13, the abbreviations GR, AC, RT, GT, PV13, PV10 denote natural_gamma ray, interval transit time, deep resistivity, ground truth, predicted values from the ensemble-learning approach by choosing all 13 features, and predicted scores from ensemble learning by using 10 features, respectively. From both Figs. 13a and 13b, with the variation of the measured depth, a good agreement can be observed between GT and PV values. To ensure the drawing continuity, we define the nonreservoir interval as 4. Log responses of three components are regarded as a composite indicator for identifying locations of sweet spots. It is worth noting that the red and blue colors represent the interval of sweet spot and the interval without sweet spot, respectively. If the log responses of GR, AC, and RT all show the red color, the corresponding interval has a great potential to be the sweet spot. As shown in Fig. 13, several purple dashed ellipses are marked to show the locations of potential sweet spots. It can be observed that removing the PPS from original data set would not affect the determination of sweet spots. Our created profiles not only help engineers to understand the geologic and petrophysical context but also help them to find the locations of sweet spots based on the color distributions.

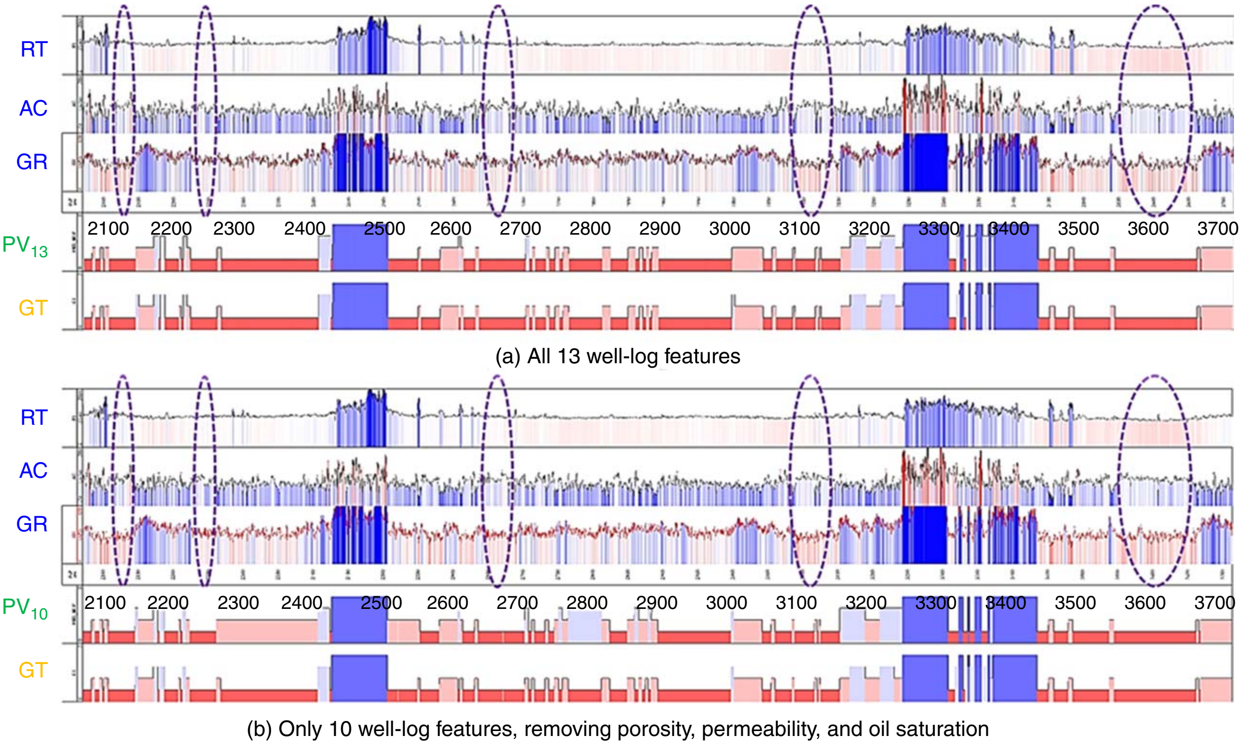


Fig. 13—Profiles showing the predicted scores by the ensemble-learning approach from selecting 13 features (a) and 10 features (b), respectively.

Although the operators in the field practice can easily identify the sweet spots by using the available well-log data from GR logs, RT logs, or logging-while-drilling (LWD) tools, our ensemble-learning framework integrates experts' knowledge to the developed model, identifies different logging curves more efficiently, decreases the cost of log interpretation, and minimizes erroneous judgment.

Discussion

Acquisition of formation properties (such as lithology, oiliness, physical properties, and electrical properties) from well logging is not simple. The signal-to-noise ratio has an obvious impact on the reality and reasonability of acquired data. Thus, how to screen out the fake data from the original data set is very challenging and also significant for the agreement rate of interpretation by different algorithms. One efficient way of picking up outliers is to analyze data consistency of its adjacent data in the same layer based on the lithology and physical properties. Moreover, it is of great importance to account for data dependency. For example, outliers can be recognized by comparing the corresponding porosity values from neutron, acoustic, and density logs.

To obtain a more accurate classification of reservoir quality, we should differentiate the oil/water layer from pure oil formation. Compared with oil formation, the oil/water layer has equivalent lithology, storage capacity, and permeability. In the oil/water layer, the margin of spontaneous potential is larger than that of the adjacent oil formation. Moreover, the resistivity of the oil/water layer is relatively low than that of its adjoining oil formation, in accordance with the logging curve of deep resistivity. Concurrently, high resistivity appears at the upper oil/water layer, and a low value occurs at its bottom. Moreover, nuclear magnetic resonance logging can also be regarded as a tool for identifying the oil-water layer based on the transverse relaxation (T_2) value. Thus, spontaneous potential, deep resistivity, and transverse relaxation T_2 become the most significant input features for our selected classifiers. Related studies would be conducted in our future work.

Among the data sorting out and meticulous classification, the selection of appropriate algorithms is also significant in improving the agreement rate of interpretation of reservoir quality classification. The GAN algorithm, as a new endeavor in the petroleum industry, enables us to create new data, which are generated from an almost equivalent distribution with the measured data but not an exact copy.

With respect to the decision tree's structures, higher agreement rates of interpretation can be achieved, once using larger search spaces of the hyperparameters from the decision trees, including max depth, gamma, minimum child weights, etc. In terms of using multiple models together, one can use majority voting, unweighted averaging to combine results from different base learners (XGBoost, CatBoost, LightGBM, SVM, etc.) to alleviate strengths for each classifier. This can also be seen from Table 3, in which the agreement rate of label 0 (L_0) from XGBoost is higher than CatBoost, but for the other three labels, CatBoost becomes better.

For testing our method on the custom data set, we suggest running the feature analysis to remove the less useful ones and then select the model via *k*-fold cross-validation, because it can reduce the bias and overcome the overfitting issue. According to the interpretation results shown in Table 3, the agreement rates of L_0 and L_1 are higher than that of L_2 and L_3. One of the reasons is probably that the corresponding number of training samples is smaller. In the future, we would oversample the data from L_2 and L_3.

Although the proposed ensemble-learning framework can improve the agreement rate of interpretation compared with the single model, the design and the configuration of the base and meta learners also take more time and effort. The search space of the hyperparameters may also increase if more base learners are included. Thus, it would be helpful to use advanced hyperparameter tuning techniques such as Bayesian optimization (Snoek et al. 2012). In our experiments, we only consider MLP or XGBoost as the meta learner. In our future work, we would attempt to use more meta learners and higher levels for model stacking (more than two levels) to boost the performance. We would also improve the GAN model in generating the quality of the fake data set and merge both the fake and real data sets in training to enhance the interpretation results.

Conclusions

In this paper, we first introduced different GB algorithms (XGBoost, CatBoost, and LightGBM) and discussed their benefits and disadvantages. Then a new algorithm, named GAN, was introduced to generate fake data sets and combine with XGBoost to predict the reservoir quality. Afterward, we collected almost 10,000 samples (each sample has 13 features and corresponding labels) from 73 horizontal wells in a certain oil field. Eleven different classifiers (such as SVMs, RF, MLP, XGBoost, LightGBM, CatBoost, CGANs/XGBoost, CTGANs/XGBoost) were implemented to predict the label of the corresponding reservoir quality. Conclusively, we established an ensemble-learning framework combining a two-step classification for improving the agreement rate of interpretation. We make the following conclusions:

1. In comparison with SVMs and MLP, GBDTs (XGBoost/CatBoost/LightGBM) not only offer high agreement rate of interpretation but deal with the arbitrary size of the data set for each label.
2. CatBoost is first applied to classify reservoir quality based on the well-log data. This algorithm produces the highest agreement rate of 82.5% among all classifiers because all features are selected for training. If the well-log interpreted parameters (porosity, permeability, and oil saturation) are removed from the original data set, the highest agreement rate of 83.37% is obtained by XGBoost. Compared with SVMs, GBDTs could save almost 70% of the time complexity during training.
3. The agreement rates of interpretation from using synthetic data sets generated by CGANs and CTGANs are 27.44 and 68.58%, respectively, which demonstrates that the GAN-based models probably show poor performance in identifying sweet spots. However, it provides a guideline for the field engineers to implement this approach by carefully tuning the relevant parameters for improving the agreement rate.
4. Among other base models (Tables 3 and 5) and ensemble models (Tables 4 and 6), the proposed ensemble-learning framework with the combination of XGBoost and CatBoost as the first level and MLP as the second level owns the highest agreement rates of 83.16 and 84.48% from selecting 13 features and 10 features, respectively. Thus, it can be regarded as an effective algorithm of identifying sweet spots based on the labeled reservoir quality, which is capable of saving the interpretation time.
5. Based on the color change of logging curves with the alteration of measured depth, the created profiles of predictions and log responses vividly reflect the geologic and petrophysical context and visualize the intervals of sweet spots. Moreover, removing the PPS parameters from the original data set would not affect the determination of sweet spots.

Acknowledgment

The authors would like to acknowledge financial support from the State Key Laboratory of Marine Geology of Tongji University, the Harvard School of Engineering and Applied Sciences–China University of Petroleum Beijing joint laboratory on petroleum science, and the 111 project (B13010).

References

Ahmadi, M. A., Zendehboudi, S., and James, L. A.2018. Hybrid Connectionist Model Determines CO_2–Oil Swelling Factor. *Petroleum Science* **15** (3): 591–604. https://doi.org/10.1007/s12182-018-0230-5.

Aldrich, J. and Seidle, J. 2018. "Sweet Spot" Identification and Optimization in Unconventional Reservoirs. Paper presented at the AAPG Annual Convention and Exhibition, Salt Lake City, Utah, USA, 20–23 May.

Alkinani, H. A., Al-Hameedi, A. T. T., Duun-Norman, S. et al. 2019. Applications of Artificial Neural Networks in the Petroleum Industry: A Review. Paper presented at the SPE Middle East Oil and Gas Show and Conference, Manama, Bahrain, 18–21 March. SPE-195072-MS. https://doi.org/10.2118/195072-MS.

Alqahtani, G. D., Alfaleh, A. A., Ali Nasser, K. et al. 2018. Methods, Systems, and Computer Medium Having Computer Programs Stored Thereon To Optimize Reservoir Management Decisions Based on Reservoir Properties. US Patent No. US20180258748A1.

Bao, A., Gildin, E., and Zalavadia, H. 2018. Development of Proxy Models for Reservoir Simulation by Sparsity Promoting Methods and Machine Learning Techniques. Paper presented at the ECMOR XVI—16th European Conference on the Mathematics of Oil Recovery, Barcelona, Spain, 3–6 September. https://doi.org/10.3997/2214-4609.201802180.

Bhattacharya, S. and Mishra, S. 2018. Application of Machine Learning for Facies and Fracture Prediction Using Bayesian Network Theory and Random Forest: Case Studies from the Appalachian Basin, USA. *J Pet Sci Eng* **170**: 1005–1017. https://doi.org/10.1016/j.petrol.2018.06.075.

Bougher, B. 2016. *Machine Learning Applications to Geophysical Data Analysis*. PhD dissertation, University of British Columbia, Vancouver, British Columbia, Canada.

Breiman, L. 1996. Bias, Variance, and Arcing Classifiers. Technical Report 460, University of California, Berkeley, Berkeley, California, USA (April 1996).

Breiman, L. 1997. Arcing the Edge. Technical Report 486, University of California, Berkeley, Berkeley, California, USA (June 1997).

Chen, T. 2014. *Introduction to Boosted Trees*. Short course presented at University of Washington, Seattle, Washington, USA, 22 October. http://docplayer.net/32894295-Introduction-to-boosted-trees-tianqi-chen-oct.html.

Chen, T. and Guestrin, C. 2016. XGBoost: A Scalable Tree Boosting System. Paper presented at the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, California, USA, 13–17 August. https://doi.org/10.1145/2939672.2939785.

Chen, T. and He, T. 2014. Higgs Boson Discovery with Boosted Trees. Paper presented at the NIPS Workshop on High-Energy Physics and Machine Learning, PMLR **42**: 69–80.

Cheung, K. 2020. 10 Applications of Machine Learning in Oil & Gas, https://algorithmxlab.com/blog/10-applications-machine-learning-oil-gas-industry/.

Dorogush, A. V., Ershov, V., and Gulin, A. 2017. CatBoost: Gradient Boosting with Categorical Features Support. https://arxiv.org/abs/1810.11363.

Fan, B., Aeron, S., Pedrycz, A. et al. 2017. On Acoustic Signal Compression for Ultrasonic Borehole Imaging. *IEEE Trans Comput Imaging* **3** (2): 330–343. https://doi.org/10.1109/TCI.2017.2670366.

Fanchi, J. 2010. *Integrated Reservoir Asset Management: Principles and Best Practices*. Amsterdam, The Netherlands: Elsevier Science. https://doi.org/10.1016/C2009-0-62240-6.

Friedman, J. 2001. Greedy Function Approximation: A Gradient Boosting Machine. *Ann Stat* **29** (5): 1189–1232. https://doi.org/10.1214/aos/1013203451.

Goodfellow, I., Pouget-Abadie, J., Mirza, M. et al. 2014. Generative Adversarial Networks. Paper presented at NIPS 2014, Montreal, Quebec, Canada, 8–13 December. arXiv:1406.2661.

Hauge, V. and Hermansen, G. 2017. Machine Learning Methods for Sweet Spot Detection: A Case Study. In *Geostatistics Valencia 2016*, ed. J. Gómez-Hernández, J. Rodrigo-Ilarri, M. Rodrigo-Clavero, E. Cassiraga, and J. Vargas-Guzmán, Vol. 19. Cham, Switzerland: Quantitative Geology and Geostatistics Series, Springer. https://doi.org/10.1007/978-3-319-46819-8_38.

Jones, V. 2018. Machine Learning To Transform Oil and Gas Industry, https://www.rigzone.com/news/machine_learning_to_transform_oil_and_gas_industry-20-sep-2018-156978-article/.

Ke, G., Meng, Q., Finley, T. et al. 2017. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. Paper presented at the 31st Conference on Neural Information Processing Systems (NIPS), Long Beach, California, USA, 4–9 December.

Kearns, M. and Valiant, L. 1994. Crytographic Limitations on Learning Boolean Formulae and Finite Automata. *Jour ACM* **41** (1): 67–95. https://doi.org/10.1145/174644.174647.

Kemajou, V. N., Bao, A., and Germain, O. 2019. Wellbore Schematics to Structured Data Using Artificial Intelligence Tools. Paper presented at the Offshore Technology Conference, Houston, Texas, USA, 6–9 May. OTC-29490-MS. https://doi.org/10.4043/29490-MS.

Ketineni, S. P., Ertekin, T., Anbarci, K. et al. 2015. Structuring an Integrative Approach for Field Development Planning Using Artificial Intelligence and Its Application to an Offshore Oilfield. Paper presented at the SPE Annual Technical Conference and Exhibition, Houston, Texas, USA, 28–30 September. SPE-174871-MS. https://doi.org/10.2118/174871-MS.

Liang, L., Le, T., Zimmermann, T. et al. 2019. A Machine Learning Framework for Automating Well Log Depth Matching. Paper presented at the SPWLA 60th Annual Logging Symposium, The Woodlands, Texas, USA, 15–19 June. SPWLA-2019-S. https://doi.org/10.30632/PJV60N5-2019a3.

Lin, Z., Khetan, A., Fanti, G. et al. 2018. PacGAN: The Power of Two Samples in Generative Adversarial Networks. Paper presented at the 32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montreal, Canada, 2–8 December. arXiv:1712.04086.

Luo, G., Tian, Y., Bychina, M. et al. 2019. Production-Strategy Insight Using Machine Learning: Application for Bakken Shale. *SPE Res Eval & Eng* **22** (3): 800–816. SPE-195681-PA. https://doi.org/10.2118/195681-PA.

Mason, L., Baxter, J., Bartlett, P. et al. 1999. Boosting Algorithm as Gradient Descent in Function Space. Paper presented at the 12th International Conference on Neural Information Processing Systems, Denver, Colorado, USA, 29 November–4 December.

Mirza, M. and Osindero, S. 2014. Conditional Generative Adversarial Nets. arXiv:1411.1784.

Nielsen, C. and Okoniewski, M. 2019. GAN Data Augmentation through Active Learning Inspired Sample Acquisition. Paper presented at the Computer Vision and Pattern Recognition Workshop (CVPR), Long Beach, California, USA, 15–21 June.

Pan, Y., Bi, R., Zhou, P. et al. 2019. An Effective Physics-Based Deep Learning Model for Enhancing Production Surveillance and Analysis in Unconventional Reservoirs. Paper presented at the SPE/AAPG/SEG Unconventional Resources Technology Conference, Denver, Colorado, USA, 22–24 July. URTEC-2019-145-MS. https://doi.org/10.15530/urtec-2019-145.

Prokhorenkova, L., Gusev, G., Vorobev, A. et al. 2018. CatBoost: Unbiased Boosting with Categorical Features. Paper presented at the 32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montreal, Canada, 2–8 December. arXiv:1706.09516.

Qian, K.-R., He, Z.-L., Liu, X.-W et al. 2018. Intelligent Prediction and Integral Analysis of Shale Oil and Gas Sweet Spots. *Petroleum Science* **15** (4): 744–755. https://doi.org/10.1007/s12182-018-0261-y.

Rahmanifard, H. and Plaksina, T. 2019. Application of Artificial Intelligence Techniques in the Petroleum Industry: A Review. *Artif Intell Rev* **52** (4): 2295–2318. https://doi.org/10.1007/s10462-018-9612-8.

Saporetti, C. M., Fonseca, L. G., and Pereira, E. 2019. A Lithology Identification Approach Based on Machine Learning with Evolutionary Parameter Tuning. *IEEE Geosci Remote Sens Lett* **16** (12): 1819–1823. https://doi.org/10.1109/LGRS.2019.2911473.

Satter, A. and Iqbal, G. 2015. *Reservoir Engineering: The Fundamentals, Simulation, and Management of Conventional and Unconventional Recoveries*. Amsterdam, The Netherlands: Elsevier Science. https://doi.org/10.1016/C2013-0-13485-X.

Silva, F., Fernandes, S., Casacao, J. et al. 2019. Machine-Learning in Oil and Gas Exploration: A New Approach to Geological Risk Assessment. Paper presented at the 81st EAGE Conference and Exhibition, London, England, UK, 3–6 June. https://doi.org/10.3997/2214-4609.201900988.

Snoek, J., Larochelle, H., and Adams, R. 2012. Practical Bayesian Optimization of Machine Learning Algorithms. Paper presented at the 26th Conference on the Neural Information Processing Systems (NeurIPS 2012), Lake Tahoe, Nevada, USA, 3–8 December. arXiv:1206.2944.

Suarez-Rivera, R., Von Gonten, W. D., Graham, J. et al. 2016. Optimizing Lateral Landing Depth for Improved Well Production. Paper presented at the SPE/AAPG/SEG Unconventional Resources Technology Conference, San Antonio, Texas, USA, 1–3 August. URTEC-2460515-MS. https://doi.org/10.15530/URTEC-2016-2460515.

Tahmasebi, P. and Hezarkhani, A. 2012. A Hybrid Neural Networks-Fuzzy Logic-Genetic Algorithm for Grade Estimation. *Comput Geosci* **42**: 18–27. https://doi.org/10.1016/j.cageo.2012.02.004.

Tahmasebi, P., Javadpour, F., and Sahimi, M. 2017. Data Mining and Machine Learning for Identifying Sweet Spots in Shale Reservoirs. *Expert Syst Appl* **88**: 435–447. https://doi.org/10.1016/j.eswa.2017.07.015.

Tandon, S. 2019. Integrating Machine Learning in Identifying Sweet Spots in Unconventional Formations. Paper presented at the SPE Western Regional Meeting, San Jose, California, USA, 23–26 April. SPE-195344-MS. https://doi.org/10.2118/195344-MS.

Tang, J., Wu, K., Zuo, L. et al. 2019. Investigation of Rupture and Slip Mechanisms of Hydraulic Fractures in Multiple-Layered Formations. *SPE J*. **24** (5): 2292–2307. SPE-197054-PA. https://doi.org/10.2118/197054-PA.

Te Stroet, C., Zwaan, J., de Jager, G. et al. 2017. Predicting Sweet Spots in Shale Plays by DNA Fingerprinting and Machine Learning. Paper presented at the SPE/AAPG/SEG Unconventional Resources Technology Conference, Austin, Texas, USA, 24–26 July. URTEC-2671117-MS. https://doi.org/10.15530/URTEC-2017-2671117.

Torralba, A., Murphy, K. P., and Freeman, W. T. 2007. Sharing Visual Features for Multiclass and Multiview Object Detection. *IEEE Trans Pattern Anal Mach Intell* **29** (5): 854–869. https://doi.org/10.1109/TPAMI.2007.1055.

Tripoppoom, S., Yu, W., Sepehrnoori, K. et al. 2019. Application of Assisted History Matching Workflow to Shale Gas Well Using EDFM and Neural Network-Markov Chain Monte Carlo Algorithm. Paper presented at the SPE/AAPG/SEG Unconventional Resources Technology Conference, Denver, Colorado, USA, 22–24 July. URTEC-2019-659-MS. https://doi.org/10.105530/urtec-2019-659.

Wang, S. and Chen, S. 2019. Insights to Fracture Stimulation Design in Unconventional Reservoirs Based on Machine Learning Modeling. *J Pet Sci Eng* **174**: 682–695. https://doi.org/10.1016/j.petrol.2018.11.076.

Wang, X. S., Nayak, K., Liu, C. et al. 2014. Oblivious Data Structures. Paper presented at the ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, Arizona, USA, 3-7 November. https://doi.org/10.1145/2660267.2660314.

Xu, L. and Veeramachaneni, K. 2019. Modeling Tabular Data Using Conditional GAN. Paper presented at the 33rd Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada. arXiv:1907.00503.

Zhang, T.-F., Tilke, P., Dupont, E et al. 2019. Generating Geologically Realistic 3D Reservoir Facies Models Using Deep Learning of Sedimentary Architecture with Generative Adversarial Networks. *Petroleum Science* **16** (3): 541–549. https://doi.org/10.1007/s12182-019-0328-4.

Zhang, F., Damjanac, B., and Maxwell, S. 2019. Investigating Hydraulic Fracturing Complexity in Naturally Fractured Rock Masses Using Fully Coupled Multiscale Numerical Modeling. *Rock Mech Rock Eng* **52** (12): 5137–5160. https://doi.org/10.1007/s00603-019-01851-3.

Zhou, K., Zhang, J., Ren, Y. et al. 2020. A Gradient Boosting Decision Tree Algorithm Combining Synthetic Minority Oversampling Technique for Lithology Identification. *Geophysics* **85** (4): WA147–WA158. https://doi.org/10.1190/geo2019-0429.1.

Zhou, Z.-H. 2012. *Ensemble Methods: Foundations and Algorithms*, first edition. Boca Raton, Florida, USA: Chapman and Hall/CRC.