A Model-Agnostic Approach for Explaining the Predictions on Clustered Data

Zihan Zhou, Mingxuan Sun, and Jianhua Chen
Division of Computer Science and Engineering
Louisiana State University
Email: zzhou23@lsu.edu, msun@csc.lsu.edu,edu, jianhua@csc.lsu.edu

***Abstract*—Machine learning models especially deep neural network models have shown great potential in making decisions when analyzing clustered or longitudinal data. However, lack of model transparency is a major concern in risk sensitive domains such as social science and medical diagnosis. Despite the early success of explaining machine learning models, there is a lack of explanation methods that can be applied to any predictors on *clustered data* since most of the existing models assume that all observations are independent of each other. In this paper, we address this deficiency and propose to use a linear mixed model to mimic the local behavior of any complex model on clustered data, which can also improve the fidelity of the explanation method to the complex models. We apply our method to explain several models including a deep neural network model on two tasks including movie recommendation and medical record diagnosis. Experiment results show that our model outperforms the baseline models on several metrics such as fidelity and exactness.**

***Index Terms*—Explainable machine learning; Deep neural network; Clustered data**

I. Introduction

Predictive analysis on clustered or longitudinal data is widely applied in social, behavioral, and health sciences, where data often contain clustered structures and observations within each cluster are correlated. For example, patients may have multiple hospital admissions during a period of time, and the test observations of different admissions (longitudinal data) for a patient form a cluster. Observations nested within each cluster, e.g., the ratings of movies from the same user or the length of ICU stays for the same patient, are usually correlated. Machine learning methods have been widely applied in analyzing these types of data and make decisions such as personalized recommendation, malware classification, and medical diagnosis. However, explaining the predictions by advanced learning models such as deep neural networks remains a challenging task due to the complex model architectures.

Interpreting predictions on clustered data in risk sensitive domains such as medical diagnosis is important because explanations can increase human trust in the model or help human adjust the model to make better decisions. Research studies [1], [2] have observed that explanations may enhance user trust in the system and increase the acceptance of the decision. Also accurate and reliable explanations can be the key to identifying failure models, discovering new knowledge, and avoiding unfairness issues.

It has been convincingly argued that a good explanation should be both interpretable and faithful to the original model. An explanation should make a (human)understandable connection between the input variables of the prediction model (e.g., "sneeze") and the response (e.g., "flu detected"). It is also essential to select only a few top explanatory features for better human-comprehension. Another essential criterion is fidelity; that is, the explanation should be faithful to the way the system generates prediction results.

Despite the early success of explanation models for machine learning, there is a lack of methods to explain the predictions on clustered data in a model-agnostic, interpretable, and faithful way. Some existing interpretation methods such as PLNN [3] can only explain a specific group of machine learning models. Some other model-agnostic explanation methods such as LIME [4] and LEMNA [5] can explain any black-box model, but are not able to provide explanation for clustered data with high fidelity, due to the assumption of independence of all observations. However, the independence assumption usually fails on clustered data, and therefore affects model fidelity.

In this paper, we develop a novel explanation method with high fidelity to account for the correlations existing in clustered or longitudinal data. Our method no longer assumes all data samples are independent. Specifically, we make two **key contributions** that differentiate our method from existing ones. First, when sampling from the local neighborhood of an instance, we consider the clustered nature in data and generate correlated samples. Second, we adopt a linear mixed model (LMM) to approximate the local decision boundary of a complex model such as deep neural network, to infer the important features. We argue that LMM, which integrates both cluster-level regression coefficients (i.e., random effects) and the regular regression coefficients for predictor variables (i.e., fixed effects), is a high-fidelity explanation model to accurately approximate the complex models' predictions on clustered data.

To our best knowledge, our model is the first one that can be applied to explaining the predictions of any black-box model for clustered data, with high fidelity and high exactness at the same time. Experimental results on benchmark datasets including the MovieLens and MIMIC-III datasets, demonstrate that our proposed method significantly outperforms state-of-the-art baselines in explaining the predictions on clustered data.

II. Related Work

Most of the existing explanation methods focus on generating explanations for specific prediction models. For example, EluciDebug [6], a full human-in-the-loop system, explains an already interpretable model, e.g., Naive Bayes in text mining. PLNN [3] provides consistent and exact interpretation only for the family of piecewise linear neural networks. Therefore, the explanations are restricted to specific neural network architectures and thus not model-agnostic.

Recent attempts have been made to learn simple interpretable models that can approximate the predictions of original models [7], [4] with applications in text or image classification. These methods do not require any knowledge about the classifier internals such as network architecture and parameters. They treat the original model as a "black box" and analyze the model behavior through the inputs and the corresponding outputs. The most representative system in this category is LIME [4] that can explain the predictions of any classifier in a faithful way by approximating it locally with an interpretable model such as a logistic regression or a decision tree. A recent work OpenBox [3] outperforms LIME, in the exactness and consistency of explanations for similar instances. However, OpenBox is again restricted to the family of piecewise linear neural networks, not a system that can explain any machine learning models.

A very recent model-agnostic approach LEMNA [5] proposes a high-fidelity explanation method dedicated for security applications. The method extends LIME with an important twist that it no longer assumes that features are independent. Based on fused LASSO, features are grouped together and the dependency between features is better captured. Our framework can be regarded as a complementary approach to LEMNA in that we assume observations (e.g., data samples) are no longer independent.

III. Problem Definition

A. Applications on Clustered/Longitudinal Data

Recommender System: Recommender systems such as Netflix, Pandora, Amazon aim to predict user interests and recommend items such as movies, music, and products tailored to each individual. The response variables (user ratings on items) are clustered within users since each individual user tends to have his/her own rating patterns.

Patient Health Record Analysis: Longitudinal data are clinical records summarizing clinical experience of a patient over a period of time, which can be used for evaluating and optimizing the health care delivered. The health records (e.g., ICU stay time) are clustered within patients since different patients may have different health conditions.

In general, we would like to explain any prediction model including deep neural network models for both classification and regression tasks on clustered/longitudinal data. In this paper, we focus on deep neural network models for the above two regression tasks and provide explanations for the prediction models. We then will evaluate the explanations based on metrics described in Section V-D.

B. Background of Model-Agnostic Explanation

Model-agnostic explanation methods treat a model (e.g., a deep neural network classifier) as a "black box" with no knowledge about the model details such as the architecture and parameters of the neural network. The goal is to understand the "black box" by observing the inputs and the corresponding prediction results. One of the most representative systems is LIME [4]. The system aims to identify the important features for a specific input instance to explain why the instance is classified as one of the class labels. Data is sampled from the neighborhood of the instance and a local linear model is used to approximate the local decision boundary of the target classifier in the feature space. Since a linear model is self-explained, it can provide a certain number of important features based on regression coefficient as the explanation to the target model.

Formally, a prediction model to be explained is defined as $f : \mathbf{X} \rightarrow \mathbb{R}$, which maps an input data point represented by a feature vector $x \in \mathbb{R}^p$ to the outcome variable such as the probability of a class label. An explanation is defined as a model $g \in G$, where G is a class of models that are easy to interpret. The model complexity is represented as $\Omega(g)$, such as the number of non-zero coefficients for linear models or the depth for decision tree models. The input to g is $x' \in \mathbb{R}^{p'}$, which is an interpretable representation of the original feature x, e.g., a binary vector indicating the occurrence of each feature, or a sub-vector of x. In addition, $\Pi_x(x, z)$ is used to measure the proximity between a pair of data points x and z, which defines the neighborhood around x. Finally, let $L(f, g, \Pi_x)$ measure the deviance of g from f in the neighborhood defined by x. The goal is to minimize the approximation loss $L(f, g, \Pi_x)$ and to keep the model complexity $\Omega(g)$ as low as possible so that the explanation is locally faithful and interpretable, that is:

$$\epsilon(x) = \arg\min_{g \in G} L(f, g, \pi_x) + \Omega(g). \tag{1}$$

In the framework adopted by [4], the candidate explanation model g is chosen from several models such as a sparse linear regression and a decision tree. Neighbors $z' \in \mathcal{Z}$ are sampled around x' by choosing nonzero elements of x' uniformly at random. The loss function L is the weighted mean square error loss, where the weight is a smoothing kernel defined by $\Pi_x(x', z')$.

IV. The Proposed Method

As a model-agnostic approach, we treat a target machine learning model as a black box and derive explanation through local model approximation. We consider the clustered nature of data and generate correlated data points when sampling from the neighborhood of an instance, then apply Linear Mixed Model to the sampled data for a more faithful approximation of the target model's local decision boundary.

A. Neighborhood Exploration for Clusters

Clustered data can be regarded as hierarchical or multilevel data. In the example of recommender systems, the data usually

contain two levels, where users form the first level and movies rated by the same user form the second level. Observations such as movie ratings from the same user are correlated. Similarly, patients' ICU records of different admissions (second level) are nested within the same patient (first level).

Formally, let i be the first-level index (e.g., user-ID, patient-ID) and j be the second-level index (e.g., j^{th} movie, j^{th} admission). Let $\mathbf{x_{ij}}$ be a p-dimensional feature vector for the j^{th} observation from group i. For example, features include user age, user gender, and movie genre for the j^{th} movie of user i. The outcome variable y_{ij} is the corresponding response of the j^{th} observation in group i, e.g., user i's rating for movie j and patient i's ICU stays for the j^{th} admission.

To better learn the local behavior of a black-box model for clustered data, we should consider sampling the local data in a similar clustered structure. That is, instead of simply perturbing original data by randomly selecting some features and masking their values as in [4], we should create clustered data similar to the training data. Specifically, for an instance with feature vector $\mathbf{x_{ij}}$ and response y_{ij} to explain, we first randomly perturb the features related to subject i and create a set of m virtual subjects (e.g., user ID for MovieLens data and patient ID for MIMIC-III data) as the first level of the local data. Next within each virtual subject i, following the approach similar to [4], we randomly mask out other non-zero features while keeping the subject-related features the same and repeat this process for n_i times, to get the second level of the local data. We can change the number of groups m or the group size n_i adapting to different datasets.

Formally, the local samples around neighborhood of $\mathbf{x_{ij}}$ are denoted as a set $\mathcal{Z} = \{\mathbf{z_{ij}} \in \mathbb{R}^p\}$, in which there are $i = 1, \ldots, m$ subjects (e.g., users) and $j = 1, \ldots, n_i$ observations nested within each subject. There are totally $N = \sum_{i=1}^{m} n_i$ perturbed observations in the neighbor of the original data $\mathbf{x_{ij}}$. For each observation, we get the outcome of perturbed data y_{ij} through the model to explain, i.e., $f(\mathbf{z_{ij}})$. Our goal is to fit this local data $\{\mathbf{z_{ij}}, y_{ij}\} \in \mathcal{Z}, \mathcal{Y}$ to a simple LMM with a random effect on subject ID and fixed effects on other features.

B. Linear Mixed Model (LMM)

Compared with ordinary linear regression, LMM adds a random effect component to capture the correlations among the subjects within clusters. For example, an ordinary linear regression that models the relationship between two variables assumes that the intercept and slope of the regression line are fixed for the whole population. In contrast, LMM model assumes that the intercepts and/or the slopes of different clusters may vary randomly depending on groups (e.g., subject-ID) in addition to the fixed effects (the intercept and slope coefficients returned by ordinary linear regression).

Take movie recommendation domain as an example, each individual user (depending on age and gender) may have his/her own patterns in rating movies of certain genres, say, "romance", deviating from the whole population mean in rating movies of this type. Young female users tend to give higher ratings to romance movies whereas young male users may rate in opposite way. This could be better handled in LMM by putting features "age", "gender" and movie "genres" as random effect variables.

Formally, a linear mixed model that augments the linear predictor with random effects for each subject i is:

$$y_{ij} = \mathbf{z_{ij}}^\top \beta + \mathbf{c_{ij}}^\top \gamma_\mathbf{i} + \epsilon_{ij}, \tag{2}$$

where $\mathbf{z_{ij}} \in \mathbb{R}^p$ is the feature vector of the j^{th} member of subject i, and $\beta \in \mathbb{R}^p$ is the fixed-effect coefficient vector shared by the whole population. The vector $\mathbf{c_{ij}} \in \mathbb{R}^q$, a sub-vector extracted from $\mathbf{z_{ij}}$ with $q \leq p$, is the feature vector of the j^{th} member of subject i for q random effects such as age and gender. The vector $\gamma_\mathbf{i} \in \mathbb{R}^q$ is the random effect variables/coefficients shared within each subject i, which is used to model the subject's deviation from the fixed effects. Finally, ϵ_{ij} is the residual that is not explained by the model. Both γ_i and ϵ_{ij} are assumed to follow a normal distribution with zero mean. For model simplicity, we assume a random intercept for each subject i to account for deviation of subject mean to the population mean. In such a case, γ_i is a one-dimensional random effect variable and $\mathbf{c_{ij}}$ becomes a constant (i.e., $q = 1$ and $\mathbf{c_{ij}} = 1$).

Usually, the n_i observations of subject i are stacked together and Equation (2) is rewritten in matrix notation as:

$$Y_i = \mathbf{Z_i}\beta + \mathbf{C_i}\gamma_\mathbf{i} + \epsilon_i, \qquad i = 1, \cdots, m \tag{3}$$

$$\gamma_i \sim \mathcal{N}(0, D), \qquad D \in R^{q \times q} \tag{4}$$

$$\epsilon_i \sim \mathcal{N}(0, \Sigma_i), \qquad \Sigma_i \in R^{n_i \times n_i} \tag{5}$$

where $Y_i = [y_{i1}, \ldots, y_{in_i}]^\top \in R^{n_i}$ is the response vector for subject i, matrix $\mathbf{Z_i} = [\mathbf{z_{i1}}^\top, \ldots, \mathbf{z_{in_i}}^\top]^\top$ of size $n_i \times p$ is the data (feature) matrix of fixed effects for subject i, and matrix $\mathbf{C_i} = [\mathbf{c_{i1}}^\top, \ldots, \mathbf{c_{in_i}}^\top]^\top$ of size $n_i \times q$ is the data (feature) matrix of random effects for subject i. In addition, $\gamma_\mathbf{i}$ is assumed to follow a normal distribution with zero mean and covariance D. The residual term $\epsilon_\mathbf{i}$ is assumed to follow a normal distribution with zero mean and covariance Σ_i.

Finally, we concatenate the observations of all subjects and rewrite Equation (3) in matrix notation. Let $\mathbf{Y} = [Y_1^\top, \ldots, Y_m^\top]^\top \in \mathbb{R}^N$ be the outcome vector for all observations of all subjects, where $N = \sum_1^m n_i$. Let $\mathbf{Z} = [\mathbf{Z_1}^\top, \ldots, \mathbf{Z_m}^\top]^\top \in \mathbb{R}^{N \times p}$ be the matrix of the predictor variables; Matrix $\mathbf{C}$ of size $N \times (m \times q)$ is a block diagonal matrix that concatenates each $\mathbf{C_i}$ along the block diagonal.

The matrix notation of the linear mixed model is:

$$\mathbf{Y} = \mathbf{Z}\beta + \mathbf{C}\gamma + \epsilon, \tag{6}$$

where γ is the concatenation of the random effect variables $\gamma = [\gamma_\mathbf{1}, \ldots, \gamma_\mathbf{m}]^\top$ with size $m \times q$, and $\epsilon = [\epsilon_\mathbf{1}, \ldots \epsilon_\mathbf{m}]^\top \in \mathbb{R}^N$ is the residual term. Moreover, let $\mathbf{\Lambda} = diag(D)$ of size $(m \times q) \times (m \times q)$ denote the corresponding covariance matrix for γ and $\mathbf{W} = diag(\Sigma_i) \in \mathbb{R}^{N \times N}$ be the corresponding covariance matrix for ϵ.

Algorithm 1 Algorithm for calculating explain features by LMM for any model

Input: All the training instances $\mathcal{O}^{train}$; testing instances $\mathcal{O}^{test}$ of size T; budget K
Output: a set of explain features
begin
  Train target model $f(x)$ using $\mathcal{O}^{train}$
  if $f(x)$ = *'linear'* **then**
    get a list $\mathbf{F}$ of size K important features
  end
  for $x_t \in \mathcal{O}^{test}$ **do**
    1. generate local perturbed data $\mathbf{Z}$
    2. feed local perturbed data $\mathbf{Z}$ to target model $f(x)$, to get predictions as vector $\mathbf{Y}$
    3. train LMM with input= $\mathbf{Z}$ and output= $\mathbf{Y}$ to obtain a list $\mathbf{E}$ of K important features
    4. re-fit a debiased LMM model with only K features in list $\mathbf{E}$, and get the prediction $g(x_t)$ for x_t
    5. get the prediction $f(x_t)$ for x_t by target model $f(x)$
    6. compute the absolute difference $d_t = \|f(x_t) - g(x_t)\|$
    if $f(x)$ = *'linear'* **then**
      calculate the recall, precision, and F measure of list $\mathbf{E}$ to list $\mathbf{F}$
    end
  end
  Output list $\mathbf{E}$, average prediction discrepancy ($\sum_t d_t$)/T
  if $f(x)$ = *'linear'* **then**
    Output list $\mathbf{E}$, average prediction discrepancy ($\sum_t d_t$)/T , average recall, precision, and F measure
  end
end

C. Estimation

Parameter estimation of LMM typically involves maximum joint likelihood of $\mathbf{Y}$ and γ as in the following equation:

$$\max_{\beta,\gamma,\mathbf{W},\mathbf{\Lambda}} logf(\mathbf{Y},\gamma) = \max_{\beta,\gamma,\mathbf{W},\mathbf{\Lambda}} logf(\mathbf{Y}|\gamma)\cdot f(\gamma)$$
$$= \max_{\beta,\gamma,\mathbf{W},\mathbf{\Lambda}} [-\frac{1}{2}(\mathbf{Y}-\mathbf{Z}\beta-\mathbf{C}\gamma)^\top \mathbf{W}^{-1}(\mathbf{Y}-\mathbf{Z}\beta-\mathbf{C}\gamma)-\gamma^\top \mathbf{\Lambda^{-1}}\gamma]. \tag{7}$$

By setting the partial derivatives of log-likelihood function with respect to β and γ to be 0, we can get the MLE estimators if the covariance parameters $\mathbf{W}$ and $\mathbf{\Lambda}$ are known. The matrix notation is given in Equation (8):

$$\begin{pmatrix} \mathbf{Z}'\mathbf{W^{-1}}\mathbf{Z} & \mathbf{Z}'\mathbf{W^{-1}}\gamma \\ \gamma'\mathbf{W^{-1}}\mathbf{Z} & \gamma'\mathbf{W^{-1}}\gamma + \mathbf{\Lambda^{-1}} \end{pmatrix} \begin{pmatrix} \hat{\beta} \\ \hat{\gamma} \end{pmatrix} = \begin{pmatrix} \mathbf{Z}'\mathbf{W^{-1}}\mathbf{Y} \\ \gamma\mathbf{W^{-1}}\mathbf{Y} \end{pmatrix}. \tag{8}$$

However, Equation (8) cannot be solved directly because the covariance matrices are usually unknown. We adopt restricted maximum likelihood (REML) estimation to obtain covariance matrices. Suppose the covariance matrices are only known up to a parameter θ. That is $\mathbf{W} = \mathbf{W}(\theta)$ and $\mathbf{\Lambda} = \mathbf{\Lambda}(\theta)$. Denote δ as the effect coefficients $\delta = (\beta, \gamma)$. Based on [8] and the note from Kneib[1], the algorithm simultaneously estimates δ and θ by first initializing values for $\hat{\delta} = \hat{\delta}^{(0)}$ and $\hat{\theta} = \hat{\theta}^{(0)}$ and then updating the value $\hat{\delta}^{(k+1)} = (\hat{\beta}^{(k+1)}, \hat{\gamma}^{(k+1)})$ through solving linear equations in Equation (8) iteratively. In each step, $\mathbf{Y}$ depends on $\delta^{(k)}$ and $\theta^{(k)}$ and $(\beta^{\mathbf{k+1}}, \gamma^{\mathbf{k+1}})$ is computed.

In addition, the constraint $\|\beta\|_0 \leq l$ can be added into Equation (7) so that β will not have more than l non-zeros, which is necessary for model regularization as well as controlling the explanation model complexity. Specifically, we can use L1 regularization for β. The detail of the estimation procedure is provide in [8].

D. Algorithm

Explanation System We introduce the algorithm of the whole explanation system. We first train a target predictor $f(x)$ (e.g., a linear model or a neural network) with 80% of the pre-processed data (the detail will be discussed in Section V-A). If the model is a linear regression model, which is self-explainable, we get a set of K (e.g., $K = 10$) gold features that are considered important by the target model. Then we test the explanation performance on instances from the rest 20% data. For each test instance, we sample a set of N (e.g., $N = 10,000$) perturbed data of clustered structure from the same feature space (the detail was discussed in Section IV-A). We also get the predictions of testing samples through target model f. Next we fit this local perturbed dataset to a local linear mixed model. We will get a set of K explanation features that are considered important by the explanation model g. With these selected features, we refit LMM model and get the predictions of testing samples by g. We can evaluate the explanation using metrics such as fidelity and exactness as defined in Section V-D. The detailed algorithm is shown in Algorithm 1.

V. Experiments and Results

We apply LMM to explain two types of models including linear regression and deep neural networks on two applications. The first application is to predict movie ratings using MovieLens 10M datasets and the second applications is to predict patients' ICU stay length using MIMIC-III dataset.

A. Dataset

The first one is MovieLens 10M dataset[2], which contains 10,000,054 ratings of 10,681 movies by 71,567 users with 95,580 movie and user tags. Each user has rated at least 20 movies. To reduce data sparsity, we first select 100 most commonly used tags by users and 100 most common tags associated with movies. After that, we get a subset of data, where each user or movie has at least one tag. The final dataset for our experiment contains 345,608 ratings by 1,006 users

[1]http://www.ugoe.de/de/document/download/c7cf1279ea23ebacd5084afcd5e9cdde-en.ps/remlestimation.ps

[2]http://files.grouplens.org/datasets/movielens/ml-10m-README.html

for 5,193 movies. The feature for each user-movie pair (i,j) contains the tags of user i, the tags associated with movie j, the ratings of user i for all other movies except j, and the ratings of movie j from all other users except i. For example, if a user has a tag, the feature value is the frequency of this tag being used; otherwise, the feature value is 0. The output variable is the rating of movie j by user i. Our data now include a sparse feature matrix of size (345,608, 1,209) and an outcome vector of all ratings. Then we split this dataset with 80% as training set and 20% as testing set.

The second dataset is collected from MIMIC-III v1.4 clinical database[3], which contains records of patients who stayed within an intensive care units at a medical center. This is a de-identified public dataset for research purposes. We have completed the required training course on data privacy and security and obtained access of the database. In this dataset, one patient has multiple records of being admitted to a hospital (the same hospital at different times or different hospitals). Each admission has a record described by a set of ICD-9 codes (alpha-numeric codes to classify diseases and various signs, symptoms, etc.) assigned to the patient during that particular admission, which forms the independent variables, and the length of stay in ICU (in number of days) which is the outcome variable. We find that ICU stay time is right skewed and therefore use a logarithmic transformation on the output. We select 1,000 patients with most recorded diagnosis ICD-9 codes. The final dataset contains in total 4,875 admissions of 1,000 patients and each admission is represented as a vector of size 2,952 (i.e., the number of ICD-9 codes). We split this dataset into 80% training and 20% testing sets, and test 1,000 random selected testing samples in the experiment.

B. Types of Models to Explain

For both datasets, we first use a linear regression model with LASSO regularization as the target predictor.

The second target predictor is a deep neural network, which is a black-box model for which we want to provide explanations. Since our goal is not to propose advanced deep models, we train a relatively simple deep model to compare properties of explanations. Specifically, the model has one embedding layer and one dense layer, each with 64 hidden units, and an output layer that returns a single continuous value, i.e., the predicted rating or predicted ICU stay. The model is implemented in Keras and Tensor-flow.

C. Baseline Explanation Models

We compare the performance of four explanation models: LMM (ours), LIME, LEMNA, Greedy and Random. LIME [4] is a state-of-the-art black-box method and has been used to explain image classifiers and NLP applications. We implement the method using the Python code published by its authors [4]. For a fair comparison, we also configure LIME with $N = 10,000$ artificial samples generated the same way as that of LMM, only ignoring the subject IDs. Greedy method greedily removes features that contribute the most to the predicted values until the maximum of K features is reached. LEMNA [5] fits weighted multiple linear models on local data and using the features returned by the "best" linear model as the explanation. We implement LEMNA using R package "flexmix". The algorithm fits a mixture of regression models using the EM algorithm, and then selects important features from the most "possible" components (i.e., linear model) as suggested in [5]. In our experiments, it is computationally expensive to train a mixture model on the original features. Therefore, we use Greedy method to pre-select $K+5$ features and then train LEMNA on the reduced features to further select top K features. The Random procedure randomly picks K features as an explanation. We set budget K to 5 and 10 for our experiments. We tweak parameters for all baselines to achieve their best performance. Specifically, there are several parameters, such as the number of groups m and number of samples within one group n_i for LMM model. In local sampling, we set $m = 25$ and $n_i = 400$ for MovieLens.

[3]https://mimic.physionet.org/gettingstarted/access/

D. Evaluation Metrics

We used two metrics to evaluate and compare our explanation models to the baseline models. They are fidelity and exactness. A good explanation should achieve both high faithfulness and high exactness.

Fidelity: It measures how much an explanation is faithful to the target prediction model with respect to a test set. We adopt the metrics similar to the one in the LIME paper [4]. The "gold features" (denoted as a set F) identified by a self-explainable model such as linear regression represents the most important features for global decisions. The "explanatory features" considered as important for local decision boundary identified by explanation model for each test instance, denoted as a set E, are compared with F using similarity measures. In particular, we set the same budget ($K = 5$ and $K = 10$) for the maximum number of features that a target model or an explanation model can select.

To measure the similarity between two feature sets, we use:

- $recall = \frac{|F \cap E|}{|F|}$, the fraction of F recovered by E;
- $precision = \frac{|F \cap E|}{|E|}$, the fraction of E relevant to F;
- $F - score = 2 \cdot \frac{precision \cdot recall}{precision + recall}$, the harmonic mean of precision and recall.

We compute recall, precision, and F-score for each instance in the test set but only report the average F-score for the whole test set.

Exactness: It measures how much difference between the target model and explanation model is in terms of prediction accuracy given a test set. This metric calculates the prediction from the target model of each instance in the test set, and compares it with the prediction from the explanation model. It can be tested on any target model such as sparse linear model and deep neural networks. Specifically, we calculate the prediction of each instance x_t in test set T from both the target model $f(x_t)$ and explanation model $g(x_t)$, and then

TABLE I: Fidelity (F-score) to Linear Target Model.

Data	K	Explanation Model				
		LMM	LIME	LEMNA	Greedy	Random
Movie	5	**0.6554**	0.3433	0.376	0.5567	0.3126
	10	**0.6263**	0.446	0.406	0.5777	0.1793
MIMIC	5	**0.5385**	0.3659	0.3669	0.4603	0.3575
	10	**0.3947**	0.2656	0.268	0.3337	0.2648

calculate the average of their absolute difference over all test instances by $\frac{\sum_t |f(x_t)-g(x_t)|}{|T|}$.

E. Results on Fidelity and Exactness

Fidelity Evaluation: We first measure the faithfulness of explanations on target models that are by themselves interpretable (sparse linear regression).

From Table I we can observe that LMM has the highest F-measure for both budgets for both datasets. Greedy has the second highest value. LIME and LEMNA are however, not performing well for those datasets. Random is the worst. This may be because those baseline models assume that all observations are independent, which is inefficient for handling clustered data. We can also observe that the performance of most models for MIMIC-III data is lower due to the data sparsity.

LMM achieves a higher F-measure on smaller budget set (5 v.s. 10), which suggests that LMM provides explanations good enough with a small size of features. A short explanation is beneficial to human because it is much easier for human to process small-size information. We can further reduce the budget to get a precision close to 1.0, but we do not include the results here because further reduction of budgets leads to a poor performance of the target model.

Exactness Evaluation: As discussed in section V-D, an exact explanation should have small discrepancy in the predicted value with the target model. Since Greedy and Random methods do not need the explanation model to provide predictions, we do not compare the exactness with these two baseline methods.

From Table II we can observe that, the predicted value by LMM is much closer to the sparse linear model in both datasets and budget settings. For example, with sparse linear model as the target predictor and budget setting of 5, the average of difference in the prediction on movie ratings is 0.005 for LMM, which is 35% of the number for LIME (0.014) and 11% of that for LEMNA (0.0497). With deep model as the target predictor, LMM's prediction is closer to the target model than LIME in all settings except for MovieLens dataset with budget 10. LIME outperforms LMM slightly by 0.0023. LEMNA's prediction is not as close as LMM and LIME to the target model in all settings. Note that we did a log transform to the patients' ICU stay times. Therefore, a number of 3.165e-5 is 1 day as an example. In addition, our LMM model with 5 features performs worse than the one with 10 features in terms of exactness.

TABLE II: Exactness of Explanation Models to Target Models.

Target Model	Data	K	Explanation Model		
			LMM	LIME	LEMNA
Sparse Linear	Movie	5	**0.005358**	0.01433	0.0497
		10	**0.005328**	0.02388	0.1896
	MIMIC	5	**3.165e-5**	0.04116	0.1896
		10	**5.519e-16**	0.04896	0.2010
Deep	Movie	5	**0.03454**	0.03635	0.0871
		10	0.03720	**0.03489**	0.0835
	MIMIC	5	**0.1020**	0.1372	0.2700
		10	**0.04728**	0.1309	0.2656

VI. Conclusions

In this paper, we propose a novel explanation method to explain the predictions of any model on *clustered data*. Our work is, to the best of our knowledge, the first one that uses a model-agnostic approach to explain any black-box machine learning model on *clustered data*. Specifically, we use a linear mixed model to learn the local behavior of any black-box model on clustered data application, which improves model fidelity. We calculate the fidelity and exactness of our method to target models on two applications with two budget settings. Experiment results show that our method outperforms other baseline methods.

VII. Acknowledgement

This work was supported in part by the Louisiana Board of Regents under Grant LEQSF(2017-20)-RD-A-29 and Grant LEQSF-EPS(2019)-RAP-26.

References

[1] R. Caruana, Y. Lou, J. Gehrke, P. Koch, M. Sturm, and N. Elhadad, "Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission," in *Proc. of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2015, pp. 1721–1730.

[2] J. L. Herlocker, J. A. Konstan, and J. Riedl, "Explaining collaborative filtering recommendations," in *Proc. of the ACM Conference on Computer Supported Cooperative Work (CSCW)*, 2000, pp. 241–250.

[3] L. Chu, X. Hu, J. Hu, L. Wang, and J. Pei, "Exact and consistent interpretation for piecewise linear neural networks: A closed form solution," in *Proc. of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2018, pp. 1244–1253.

[4] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should I trust you?: Explaining the predictions of any classifier," in *Proc. of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2016, pp. 1135–1144.

[5] W. Guo, D. Mu, J. Xu, P. Su, G. Wang, and X. Xing, "Lemna: Explaining deep learning based security applications," in *Proc. of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2018, pp. 364–379.

[6] T. Kulesza, M. Burnett, W.-K. Wong, and S. Stumpf, "Principles of explanatory debugging to personalize interactive machine learning," in *Proc. of the International Conference on Intelligent User Interfaces (IUI)*, 2015, pp. 126–137.

[7] D. Baehrens, T. Schroeter, S. Harmeling, M. Kawanabe, K. Hansen, and K.-R. MÃžller, "How to explain individual classification decisions," *Journal of Machine Learning Research (JMLR)*, vol. 11, pp. 1803–1831, Jun. 2010.

[8] M. J. Lindstrom and D. M. Bates, "Newton-Raphson and EM algorithms for linear mixed-effects models for repeated-measures data," *Journal of the American Statistical Association*, vol. 83, no. 404, pp. 1014–1022, 1988.