Explainable and Local Correction of Classification Models Using Decision Trees

Hirofumi Suzuki,[1] **Hiroaki Iwashita,**[1] **Takuya Takagi,**[1]
Keisuke Goto,[1] **Yuta Fujishige,**[1] **Satoshi Hara** [2]

[1] Fujitsu Limited,
[2] Osaka University
{suzuki-hirofumi, iwashita.hiroak, takagi.takuya, goto.keisuke, fujishige.yuta}@fujitsu.com,
satohara@ar.sanken.osaka-u.ac.jp

Abstract

In practical machine learning, models are frequently updated, or *corrected*, to adapt to new datasets. In this study, we pose two challenges to model correction. First, the effects of corrections to the end-users need to be described explicitly, similar to standard software where the corrections are described as release notes. Second, the amount of corrections need to be small so that the corrected models perform similarly to the old models. In this study, we propose the first model correction method for classification models that resolves these two challenges. Our idea is to use an additional decision tree to correct the output of the old models. Thanks to the explainability of decision trees, the corrections are describable to the end-users, which resolves the first challenge. We resolve the second challenge by incorporating the amount of corrections when training the additional decision tree so that the effects of corrections to be small. Experiments on real data confirm the effectiveness of the proposed method compared to existing correction methods.

Introduction

When the domain of test data changes over time, the prediction accuracy of machine learning models gets worse. In such a case, it is necessary to update, or *correct*, the models to adapt to new datasets. The standard model correction scenario is as follows. We have a classification model trained on old data. After the training, we obtain a new additional dataset. The model correction task is to correct the old model to make accurate predictions on this additional data. There are several approaches for model correction in the literature, such as model retraining, transfer learning, and domain adaptation. The majority of these methods assume that we can access the old training data. However, this assumption is sometimes unrealistic because of privacy and security reasons. For example, the General Data Protection Regulation (GDPR) requires that personal data may not be retained longer than is necessary to achieve the prescribed purpose (Voigt and Bussche 2017). In this paper, we consider the more restricted and practical problem setting for model correction where we have the old model but have no access to the old data.

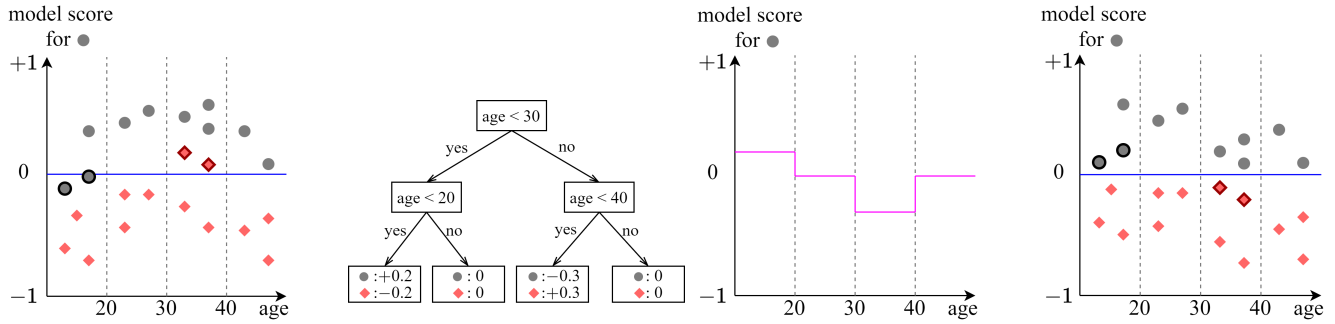
When correcting the model to adapt to new data, some forms of consistency are expected between the models before and after the correction. Suppose a machine learning model is a part of a complex system or software, and there are several processes that depend on the prediction results. A drastic correction of the model will change the operation of the entire system significantly. Such a drastic update of the system is not favorable; We do not want to release a completely different system from the previous version in order not to lose the user's experience and trust in the system. Therefore, we want to make as few corrections as possible for machine learning models embedded in complex systems. In this study, to avoid drastic updates of the entire system, we consider two requirements for model correction:

1. The specification of the correction should be explicit.
2. The number of samples to which the correction applies should be as small as possible.

The first requirement is that the user can check the changes in the model and its specifications. In other words, we would like it to be clear how the predictions are corrected so that the corrections to be explainable to the users. Explainability of correction is required, e.g., if the developer has to explain to users that the correction of the model is not intended to cause, for example, the encouragement of discrimination. The second requirement is necessary to maintain consistency between the model before and after the correction. In the analogy of a software update, this requirement corresponds to applying the minimum number of patches. The corrected model satisfying this requirement will not lead to drastic updates of the entire system.

In this paper, we propose *Local Correction Tree (LCT)*, a decision tree-based model correction method that satisfies the two requirements above so that the correction is to be explainable and to be small. The concept of LCT is shown in Figure 1. LCT is a decision tree that outputs the amount of prediction corrections to be added to the output of the old model for the given input. With LCT, the old model is corrected by adding the correction score written in the leaves to the predicted score of the model.

LCT has two notable properties. First, the decision tree itself can be regarded as a specification of the correction because each root-to-leaf path indicates a rule of how the samples will be corrected. Hence, LCT fulfills the first require-



(a) Old model scores (b) Corrections of model scores with decision tree (c) Corrected model scores

Figure 1: Concept of the proposed Local Correction Tree (LCT). (a) Old model scores for samples with two class labels, black circle and red diamond, where the score for one class is a sign-reversed version of the score for the other class. The blue horizontal line indicates the decision boundary, and there are some misclassified samples marked with boundary in the input space age < 20 and $30 \leq$ age < 40. (b) LCT is a decision tree whose leaves have the amount of score corrections for samples reaching the leaf. LCT corrects only local samples because some leaves have zero corrections. (c) Corrected model scores, which is the summation of the old model scores and the amount of corrections determined by LCT. If a sample is in age < 20 or $30 \leq$ age < 40, its old model score is corrected by $+0.2$ or -0.3, respectively, otherwise unchanged.

ment for model correction. Second, LCT is designed so that the majority of samples belong to leaves whose correction is zero. This property of LCT ensures the second requirement for model correction.

Our Contributions. We propose the first model correction method LCT that satisfies the two requirements. Using LCT, we can correct models in such a way that the specification of correction is explainable and the amount of correction to be small. Moreover, LCT is generic and easy to implement, as shown below. Specifically:

1. LCT is model agnostic, i.e., it can be applied to any classification model.
2. LCT can be implemented with a simple modification of existing decision tree construction algorithms.
3. The user can control the amount of correction in an ad-hoc manner, e.g., by adjusting the depth of the tree or by trimming branches.

Our experimental results on real data confirm the effectiveness of LCT compared to existing model correction methods in terms of its explainability and the amount of correction.

Related Work

Various methods have been proposed to correct models, such as domain adaptation and transfer learning. Domain adaptation and transfer learning aim to adapt to new tasks by transferring knowledge from models that have already been trained on another task (Pan and Yang 2010; Parisi et al. 2019; Zhuang et al. 2021). Most of these correction methods, such as density ratio estimation (Sugiyama, Suzuki, and Kanamori 2012), kernel mapping functions (Daumé III 2007), and TrAdaBoost (Dai et al. 2007), require source data that used to train the old model. Therefore, these methods cannot be used in the restricted setting where only the trained model is given but not the old data.

Fine-tuning, such as incremental learning for decision tree (Utgoff 1988; Chao and Wong 2009), random forest (Ristin et al. 2016), and gradient boosting method (Zhang et al. 2019), is one of the transfer learning methods that does not require old data. These methods allow old models to be adapted to new data. However, this may lead to a correction of the entire old model. Hence, all the samples can be affected by the correction, which conflicts with our second requirement. In addition, learning to stabilize model updates has been proposed for ranking learning using neural networks (Li et al. 2020b). However, it is not applicable to arbitrary models or tasks other than ranking.

Another line of work related to LCT is interpretable companion models that combine black-box models and white-box models (Wang 2019; Pan, Wang, and Hara 2020; Li et al. 2020a; Rafique et al. 2020). Interpretable companion models are designed to improve the interpretability of predictions by passing some of the inputs to a white-box model, and processing only remaining inputs by the black-box model. LCT is similar to interpretable companion models in the sense that it is designed to cooperate with the existing (possibly black-box) models. However, LCT and interpretable companion models differ in their objective: LCT aims at improving the accuracy of the corrected model, while interpretable companion models aim at improving interpretability at a cost of sacrificing accuracy.

Preliminaries

Notation. For a positive integer $n \in N$, we write $[n] := \{1, 2, \ldots, n\}$. In this study, we focus on classification problem: let $X = (x_1, x_2, \ldots, x_n) \in \mathbb{R}^{n \times d}$ be an input of n samples and d features and $y \in [c]^n$ be an output of class categories for each sample where c is the number of classes.

Decision Tree. A decision tree is a binary tree $T = (V, E)$ with a node set V, a root node $r \in V$, and a directed edge

set E. Each internal node $v \in V$ has a *splitter* $j(v) \in [d]$, a *cut point* $p(v) \in \mathbb{R}$, and two child nodes v_L and v_R, i.e., $(v, v_L), (v, v_R) \in E$. Given $x \in \mathbb{R}^d$, we start on descending T from r. We descend from v to v_L if $x_{j(v)} < p(v)$, otherwise to v_R. Finally, any sample reaches a leaf node that determines output.

Regularized Dual Averaging Method. We here introduce the regularized dual averaging (RDA) method (Xiao 2010) which we use for constructing LCT in the next section. RDA is a variant of stochastic gradient descent (SGD) methods and has an advantage over other SGD in realizing regularization of the following optimization problem:

$$\underset{w \in \mathbb{R}^c}{\text{minimize}} \{f(w) + \Psi(w)\}, \tag{1}$$

where f and Ψ are a differentiable function and a regularization term, respectively.

Local Correction Tree

We propose *Local Correction Tree* (LCT) as a method to correct predictions of classification models. The advantages of LCT are (i) LCT is a model agnostic method accepting any old classification model, and (ii) LCT is easy to implement and tune because LCT is a simple modification of the standard decision tree.

Outline. Let $X \in \mathbb{R}^{n \times d}$, $y \in [c]^n$ be the new data we want to adapt with. Moreover, let M be the old model and $S \in \mathbb{R}^{n \times c}$ be the prediction scores[1]: S_i is computed for the input $i \in [n]$ by using M which might use different data than X, where i is classified into $\arg\max_{k \in [c]} S_{i,k}$ by M.

LCT is a decision tree T that, for any sample $x \in \mathbb{R}^d$, outputs *correction amount* $T(x) \in \mathbb{R}^c$ for prediction scores. If x has the prediction score $s \in \mathbb{R}^c$ of M, the corrected model using LCT classifies x into $\arg\max_{k \in [c]} \{s_k + T(x)_k\}$. In LCT, we store a correction amount $w(v) \in \mathbb{R}^c$ to each node $v \in V$ so that $T(x) = w(v)$ to hold if x reaches a leaf v.

LCT aims to correct predictions locally, i.e., the number of samples affected by LCT is small. To accomplish this, LCT minimizes the following objective function g represented by the sum of the cross-entropy loss and the regularization term of L2-norm:

$$g(T) = \sum_{i \in [n]} (-\log a(S_i + T(X_i))_{y_i} + \lambda \|T(X_i)\|_2). \tag{2}$$

Here, $\lambda \in \mathbb{R}_+$ is a regularization parameter and a is the Softmax function: for a vector $v \in \mathbb{R}^c$,

$$a(v)_k = \frac{\exp(v_k)}{\sum_{k' \in [c]} \exp(v_{k'})}. \tag{3}$$

Thanks to the group-sparse regularization effect of the L2-norm (Hastie, Tibshirani, and Wainwright 2015), correction amounts in some nodes become zero. Thus, the objective function g can control the trade-off of accuracy and the number of samples affected by LCT by tuning λ.

[1]In scikit-learn-like API, we can use predict_proba method to obtain prediction scores. On deep learning models, we can use outputs of the last layer (before applying softmax) as prediction scores.

Construction Algorithm. The proposed LCT construction algorithm starts with a root node and greedily splits nodes to child nodes to decrease the objective value of $g(T)$, similar to standard tree construction algorithms (Quinlan 1993; Breiman et al. 2017). For the greedy node splitting, we consider the objective function $\hat{g}$ on the subset of samples $I \subseteq [n]$:

$$\hat{g}(I, w) = -\sum_{i \in I} \log a(S_i + w)_{y_i} + \lambda |I| \|w\|_2 \tag{4}$$

where $w \in \mathbb{R}^c$. Let $I(v) \subseteq [n]$ be the subset of samples reaching a node v. Then we can write g as

$$g(T) = \sum_{v \in \{\text{Leaves of } T\}} \hat{g}(I(v), w(v)). \tag{5}$$

When growing a leaf node v, the algorithm computes a *split* (i.e., a pair of a splitter $j(v) \in [d]$ and a cut point $p(v) \in \mathbb{R}$) such that correction amounts $w(v_L)$ and $w(v_R)$ minimizes the sum of objective values

$$\hat{g}(I(v_L), w(v_L)) + \hat{g}(I(v_R), w(v_R)). \tag{6}$$

For any split, we can use RDA to optimize $\hat{g}$ by setting

$$f(w) = -\sum_{i \in I} \log a(S_i + w)_{y_i} \text{ and } \Psi(w) = \lambda |I| \|w\|_2.$$

Algorithms 1 and 2 conclude the LCT construction algorithm. We here show a best-first construction using a heap which sorts leaves in descending order of the objective value $\hat{g}(I(v), w(v))$. We remark that we can use several stopping criteria (e.g., maximum depth of the tree, maximum number of leaves, and minimum number of samples in leaves) the same way as standard tree construction algorithms.

Pruning Strategies. After the tree construction, we can apply some pruning strategies to remove leaves with too little or too many corrections, leaves with inaccurate corrections, and redundant leaves. We show two main strategies below.

The first pruning strategy removes leaves with too little or too many corrections and leaves with inaccurate corrections by considering the prediction changes on X, y, and S. For any leaf node v of LCT, let $\hat{I}(v) \subseteq I(v)$ be the subset of samples whose predictions are changed by v; $\arg\max_{k \in [c]} S_{i,k} \neq \arg\max_{k \in [c]} \{S_{i,k} + T(X_i)_k\}$ for any $i \in \hat{I}(v)$. Similarly, let $\hat{I}_{\text{ci}}(v) \subseteq \hat{I}(v)$ be the subset of samples whose predictions are changed to incorrect classes by v; $\arg\max_{k \in [c]} \{S_{i,k} + T(X_i)_k\} \neq y_i$ for any $i \in \hat{I}_{\text{ci}}(v)$. We then apply the following rules.

- Let $n_{\text{prune}} \in \mathbb{N}$ determine the minimum number of samples required to prune leaves. If $|I(v)| < n_{\text{prune}}$ holds, we do not prune v.
- Let $\eta_{\min}, \eta_{\max} \in [0, 1]$ ($\eta_{\min} < \eta_{\max}$) determine the number of samples allowed to change their prediction. If $|\hat{I}(v)| < \eta_{\min} |I(v)|$ or $|\hat{I}(v)| > \eta_{max} |I(v)|$ holds, we set 0 to $w(v)$.

Algorithm 1: LCT Construction Algorithm

Create a decision tree T with root node r
$I(r) \leftarrow [n]$, $w(r) \leftarrow \arg\min_w \hat{g}([n], w)$
Create an empty heap H and push r to H
while H is not empty **do**
  Pop the top element v of H
  if v does not satisfy any stopping criterion **then**
    $v_L, v_R \leftarrow \mathrm{Split}(v)$
    Push v_L and v_R to H
return T

Algorithm 2: $\mathrm{Split}(v)$

Create child nodes v_L, v_R of v
$o_{\min} \leftarrow \infty$
for $j \in [d]$ **do**
  Generate cut points P of $\{X_{i,j} \mid i \in I(v)\}$
  for $p \in P$ **do**
    $I_L \leftarrow \{i \in I(v) \mid X_{i,j} < p\}$
    $I_R \leftarrow \{i \in I(v) \mid X_{i,j} \geq p\}$
    $w_L \leftarrow \arg\min_w \hat{g}(I_L, w)$
    $w_R \leftarrow \arg\min_w \hat{g}(I_R, w)$
    $o \leftarrow \hat{g}(I_L, w_L) + \hat{g}(I_R, w_R)$
    if $o_{\min} > o$ **then**
      $j(v), p(v) \leftarrow j, p$
      $I(v_L), I(v_R) \leftarrow I_L, I_R$
      $w(v_L), w(v_R) \leftarrow w_L, w_R$
      $o_{\min} \leftarrow o$
return v_L, v_R

- Let $\eta_{\text{fail}} \in [0, 1]$ determine the number of samples allowed to change their prediction to incorrect classes. If $|\hat{I}_{\text{ci}}(v)| > \eta_{\text{fail}}|\hat{I}(v)|$ holds, we set 0 to $w(v)$.

The second pruning strategy removes redundant leaves. If a node v has children such that $w(v_L) = 0$ and $w(v_R) = 0$ holds, we can delete v_L and v_R, and then we set 0 to $w(v)$, because this operation does not change the output of T. By repeating this process as much as possible, we can obtain a less redundant and more readable LCT.

Experiments

We demonstrate the performance of LCT, in comparison to other correction methods, in the restricted setting with no access to old data. All the codes were implemented by using Python 3 and ran on Ubuntu 18.04.5 LTS with Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40GHz and 128GB RAM.

Scenarios. We assumed the following two scenarios.

1. In the first scenario, we obtain new data of different distributions from old data. We then aim to create a new model that adapts to new data by correcting the old model.
2. In the second scenario, we obtain tabular data from the new domain (e.g., new side information) that was not previously available. We then aim to create a new model that incorporates the information from the new domain by correcting the old model (while also using the inputs from the original domain such as image).

Datasets. We used four datasets: Adult[2], Bank Marketing[3] (Bank in short), Cardiotocography[4] (CTG in short) from the UCI Machine Learning repository for the first scenario, and 7-point[5] (Kawahara et al. 2019) for the second scenario. Here, we split each dataset into old data to train old models and new data to train and test new models.

• **Adult** This dataset contains only tabular data consisting of census data. The task is to classify each individual whether the individual's income exceeds \$50K/year. We preprocessed categorical features by one-hot encoding and ignored entries with missing values. The resulting number of samples and features are 48,842 and 105, respectively. Finally, we split the data into old data (24,421 samples including 5,202 positive samples) and new data (24,421 samples including 6,485 positive samples) with a bias towards a feature "age": we sampled the old data with probability proportional to $\frac{1}{\text{age}}$. Thus, the old data and the new data contain many young and middle-aged individuals, respectively. The objective of model correction is to adapt to changes in the distribution affected by ages.

• **Bank** This dataset contains only tabular data of client's personal information. The task is to predict if the client will subscribe to a term deposit. We preprocessed categorical features by one-hot encoding and ignored the feature "duration", which is obtained after the correct label has been determined in the actual setting. The resulting number of samples and features are 41,188 and 63, respectively. Finally, we split the data into old data (20,594 samples including 2,350 positive samples) and new data (20,594 samples including 2,290 positive samples) with a bias towards a feature "age": we sampled the old data with probability proportional to $\frac{1}{\text{age}^2}$. Thus, the old data and the new data contain many young and old-aged individuals, respectively. The objective of model correction is to adapt to changes in the distribution affected by ages.

• **CTG** This dataset contains only tabular data of diagnostic numerical features measured from fetal cardiotocogram. The task is to predict whether the fetal state is normal, suspicious, or pathological. The number of samples and features are 2,126 and 22, respectively. For each class, there are 1,655 normal samples, 295 suspicious samples, and 176 pathological samples. We split the data evenly into old data and new data with a bias towards a feature "LB" which means "beats per minute". We sampled the old data with probability proportional as follows: the weight of each sample is 1 if "LB$''$ $<$ 120 or "LB$''$ $>$ 150 (they are a minority, but not outlier) and 5 otherwise. Thus, most minority samples do not appear in the old data. Here, the bias of the correct labels changed only slightly between old and new data. The objective of model correction is to adapt to newly obtained minority samples.

• **7-point** This dataset contains image data of skin lesions

[2]https://archive.ics.uci.edu/ml/datasets/adult

[3]https://archive.ics.uci.edu/ml/datasets/bank+marketing (bank-additional-full.csv)

[4]https://archive.ics.uci.edu/ml/datasets/cardiotocography

[5]https://derm.cs.sfu.ca/Welcome.html

and a tabular data of clinical checklist called 7-point checklist (Mackie and Doherty 1991). Each sample has two images, a clinical image and a dermoscopy color image. The task is the multi-class classification of skin lesion malignancies. We preprocessed the tabular data by encoding categorical features to one-hot except for level-of-diagnostic-difficulty. We converted level-of-diagnostic-difficulty into 0 if it is low, 0.5 if it is medium, and 1.0 if it is high. The resulting number of samples and the number of features are 1,011 and 44, respectively. The original class labels are BCC (42 samples), NEV (575 samples), MEL (252 samples), MISK (97 samples), and SK (45 samples). We combined BCC, MISK, and SK into one class OTHERS (184 samples) because their sample sizes were too small. Then we obtained the three-class (NEV, MEL, and OTHERS) classification task. Finally, we split the data into old data (505 samples) and new data (506 samples) by the stratified splitting. Here, we consider the scenario where the old model was trained using the image data only, and we obtained the new tabular data in addition to the images afterward. The objective of model correction is to improve the prediction power of the model by incorporating the newly obtained tabular data for prediction.

Old Models. We trained LightGBM (Ke et al. 2017) for Adult, Bank, and CTG dataset and deep neural network (DNN) for 7-point dataset as old models.

• **LightGBM for Adult, Bank, and CTG** We used LGBMClassifier[6] as the old model. We tuned learning_rate, min_split_gain, reg_alpha, and reg_lambda within the range [1e-8, 1] on the old data by using Optuna[7] (Akiba et al. 2019) with 100 trials. We set class_weight to balanced and used default values for the remaining parameters. We used the averaged F1 score of cross-validation on the old data for the optimization criteria to deal with data imbalance. Note that, only for CTG dataset, F1 score is computed with macro average on three class labels. The fold numbers in cross-validation are 3 for Adult and Bank dataset, and 5 for CTG dataset.

• **DNN for 7-point** We used ResNet50 (He et al. 2016) pretrained on ImageNet (Deng et al. 2009) via PyTorch (Paszke et al. 2019). We fed each of the two images (a clinical image and a dermoscopy color image) to ResNet50 and extracted a 2048 dimensional feature vector from the layer before the last fully connected layer. We then concatenated the feature vectors into a 4096 dimensional feature vector and input them to the dropout of 50% and the new fully connected layer that outputs a score for each of the three classes. We trained the new fully connected layer using SGD of batch size 64, learning rate 0.001, momentum 0.9, and 50 epoch over cross-entropy loss with balanced class weight. Moreover, we augmented image data by randomly applying flipping, affine transformation, and color jittering. Here, we split the old data to training data of 80% and validation data of 20% by the stratified splitting. We tried ten times random data splitting and selected the model of maximum macro averaged recall on validation data as the old model to deal with data imbalance in the multi-class classification task.

[6]https://lightgbm.readthedocs.io/en/latest/index.html.

[7]https://optuna.readthedocs.io/en/stable/.

Competitors. We compared our LCT, some basic classifiers, and some standard correction methods using scikit-learn (Pedregosa et al. 2011) (logistic regression, decision tree, and random forest) and LightGBM (gradient boosting decision tree). The methods except for our LCT use the prediction scores of the old model as new features.

We here split the new data into training data and test data in the manner of 5-fold cross-validation. For Adult and Bank dataset, we used the total F1 score of 3-fold cross-validation on training data for tuning hyperparameters to deal with data imbalance in the binary classification task. For CTG dataset, we used the total macro-averaged F1 score of 5-fold cross-validation on training data for tuning hyperparameters to deal with data imbalance in the multi-class classification task. For 7-point dataset, we used the total macro averaged recall of 5-fold cross-validation on training data to deal with data imbalance in the multi-class classification task. The details of competitors and the hyperparameter tuning procedures are listed as follows:

• **Ours.** We trained LCT by tuning the parameter $\lambda \in \{0.05, 0.1, \ldots, 0.5\}$ for Adult, Bank, and TCG dataset, and $\lambda \in \{0.01, 0.02, \ldots, 0.1\}$ for 7-point dataset. Here, to avoid the risk of significant changes from the old models, the candidates of λ are selected from the range where the amount of correction is small enough. We fixed the other parameters $n_{\mathrm{prune}}, \eta_{\min}, \eta_{\max}, \eta_{\mathrm{fail}}$, and some scikit-learn-like stopping criteria max_depth and min_samples_leaf. For any dataset, $n_{\mathrm{prune}} = 1$, $\eta_{\min} = 0.01$, $\eta_{\mathrm{fail}} = 0.499$, and max_depth $= 5$. For Adult and Bank dataset, min_samples_leaf $= 64$ and $\eta_{\max} = 0.5$. For CTG dataset, min_samples_leaf $= 8$ and $\eta_{\max} = 1.0$. For 7-point dataset, min_samples_leaf $= 16$ and $\eta_{\max} = 0.5$. In addition, for Adult, Bank, and TCG dataset, because there are many cut point candidates incurring much computation time for LCT construction, we computed restricted candidates in every call of Algorithm 2 by using quantile cut of 10 bins for any non-binary features.

• **L1-LR, L2-LR, DT, RF, and LGBM.** As the naive baselines, we trained some basic classifiers of scikit-learn and LightGBM using the new data only. (i) L1-LR and L2-LR are logistic regression trained via LogisticRegressionCV with L1- and L2-regularization, respectively. (ii) DT is decision tree with hyperparameters optimized by Optuna with 100 trials: we forced the parameter class_weight to balanced, and optimized the four parameters max_depth $\in \{5, 6, \ldots, 10\}$, min_samples_leaf $\in [0.0001, 0.1]$, max_features $\in [0.1, 1.0]$, and min_impurity_decrease $\in$ [1e-8, 1.0]. (iii) RF is random forest optimized similarly to DT except we set class_weight to balanced_subsample. (iv) LGBM is LGBMClassifier optimized in the same way as the old model for Adult, Bank, and CTG dataset.

• **L1-LR+, L2-LR+, DT+, RF+, and LGBM+.** As simple examples of model correction, we trained some basic classifiers using the prediction scores of the old models: we concatenated the prediction scores to the input vector as new

features. We used the same classifiers and hyperparameter optimization procedures as above.

• **LGBM+C.** For Adult, Bank, and CTG dataset, we incrementally trained the old LightGBM model by using the new data. Note that we here did NOT use additional features such as prediction scores. We used the same hyperparameter optimization procedures as above.

• **LGBM+D.** For 7-point dataset, we additionally trained LightGBM using intermediate features and prediction scores of the old DNN model. We extracted 4096 intermediate features of ResNet50 from the layer before the last fully connected layer. We then concatenated the intermediate features as well as the prediction scores to the input vector as new features. We used the same hyperparameter optimization procedures as above.

Evaluation Criteria. We first evaluated the prediction powers of each method: the test accuracy, precision, recall, and F1 scores of 5-fold evaluation with stratified-splitting. For CTG and 7-point dataset, note that precision, recall, and F1 scores were computed with macro average of multiple class labels. We then analyzed the trends of corrections: the number of samples whose predictions changed in terms of correct, incorrect, and distribution over prediction score of the old model. Finally, we scanned the constructed LCT and observed some interesting correction rules.

Result: Prediction Powers. We conclude the results for prediction powers in Figure 2: (a)–(d), (e)–(h), (i)–(l), and (m)–(p) show the results on Adult, Bank, CTG, and 7-point dataset, respectively.

On Adult dataset, except for LGBM, basic classifiers did not improve any metric. This indicates that using prediction scores of the old model is useful for adapting to the new data. As an important trend, all the new models decreased the recall from the old model. This is because the old model tended to predict too many samples to be positive on the new data, while the new models tended to suppress these optimistic predictions. Focusing on our LCT, it achieved the highest accuracy and precision, lowest recall, and comparable F1 score. Thus, our LCT tended to strongly suppress the optimistic predictions of the old model on the new data.

On Bank dataset, all the correction methods except LR-based ones succeeded to correct the old model in terms of F1 score. This indicates that it is difficult to directly predict the errors of prediction scores in Bank dataset. Here, the old model tends to be the same as that of Adult dataset, i.e. the model makes many optimistic predictions. Thus, tree-based methods suppressed the optimistic predictions of the old model on the new data more accurately than LR-based methods. Especially, looking at the results of tree-based methods in detail, our LCT achieved higher accuracy and F1 score the same as RF+, although our LCT is clearly better than RF+ in terms of simplicity of the model.

On CTG dataset, only our LCT improved F1 score clearly and made a negligible change in recall score. Moreover, our LCT achieved the highest accuracy and precision score. This suggests that only our LCT has succeeded to adapt the newly obtained minority data.

On 7-point dataset, although most of the new models improved the prediction power, using prediction scores of the old model contributed further improvements. Our LCT improved accuracy, precision, F1 score comparable or superior to the other methods. This result is favorable because (i) LCT has a more simple structure than RF+, LGBM+, and LGBM+D, (ii) LCT can output readable rules and concrete amount of corrections for prediction changes compared to L1-LR+, L2-LR+, and DT+. Although our LCT scored relatively low recall, it was still better than the old model and was comparable to L1-LR and L2-LR.

Result: Trends of Corrections. We conclude the results for trends of corrections in Figure 3. Figures 3 (a)–(b), (c)–(d), (e)–(f), and (g)–(h) show the results on Adult, Bank, CTG, and 7-point dataset, respectively. Note that the sizes of test data are 4884, 4119, 213, and 101 on Adult, Bank, CTG, and 7-point dataset, respectively.

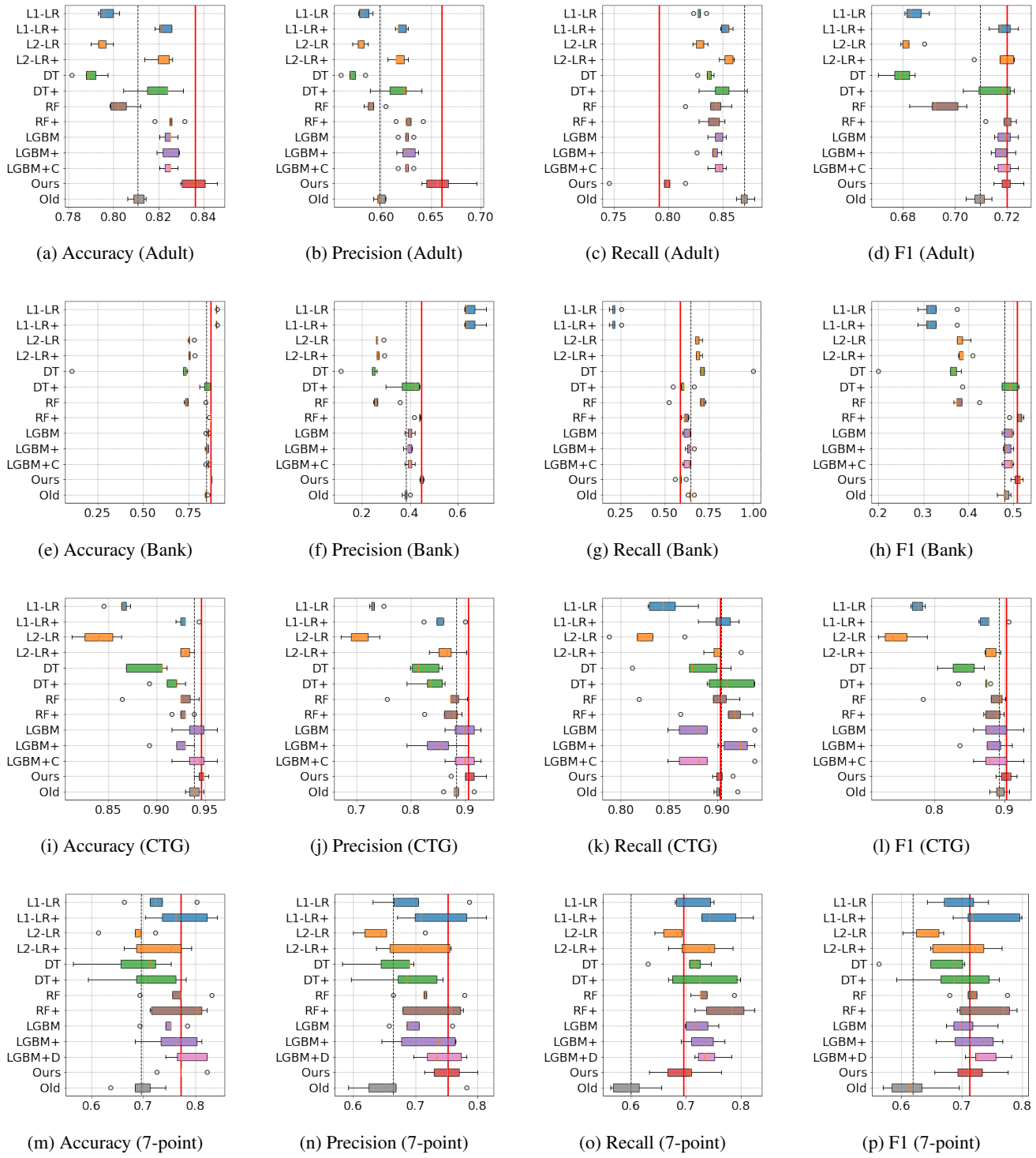
On Adult dataset, from Figure 3 (a), while both of our LCT and RF+ achieved better ratio of correct changes and incorrect changes, our LCT had more correction than RF+. Here, note that our trade-off is only between the accuracy and the number of corrections, not between the ratio and the number of corrections. In addition, all the new models achieved small enough prediction changes (within 10% of total) from the old model. Thus, since LCT achieved higher accuracy rather than RF+, our LCT could correct the old model better than any other methods with small prediction changes on Adult dataset.

On Bank dataset, from Figure 3 (c), tree-based and LR-based methods (except for DT+) made small (within 5% of total) and large (over 10% of total) prediction changes, respectively. In terms of the ratio of correct changes and incorrect changes, our LCT and RF+ were better than other tree-based methods. However, from Figure 3 (d), our LCT corrects more samples in middle prediction scores $[0.6, 0.7]$ than RF+. This is an advantage of our LCT to adapt to the new data with a broader view.

On CTG dataset, from Figure 3 (e), only our LCT achieved the high ratio of correct changes and incorrect changes. Looking at Figure 3 (f), almost all the methods except for our LCT made the many prediction changes, which might be incorrect, in high prediction scores $[0.9, 1.0]$. This indicates an advantage of our regularization method that suppresses the prediction change in high prediction scores.

On 7-point dataset, from Figure 3 (g), while our LCT changed about 15% predictions from the old model, the other new models changed more than 25% predictions. This result indicates that our LCT is superior to the other new models in the sense that it induced only small changes in predictions. Moreover, our LCT achieved the highest ratio of correct changes and incorrect changes. Thus, our LCT could correct the old models better than any other methods with small prediction changes on 7-point dataset.

As another interesting result, from Figure 3 (b), while LGBM+C changed many predictions of samples with high old scores (high confidence on the old model), our LCT changed only predictions of samples with low or middle old scores (low or middle confidence on the old model). Simi-



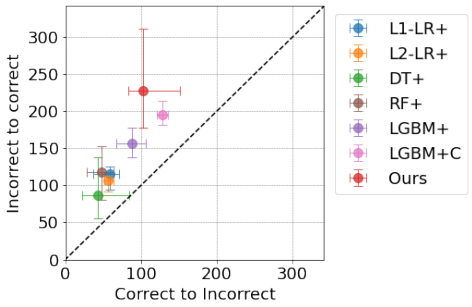
(a) Accuracy (Adult) (b) Precision (Adult) (c) Recall (Adult) (d) F1 (Adult)

(e) Accuracy (Bank) (f) Precision (Bank) (g) Recall (Bank) (h) F1 (Bank)

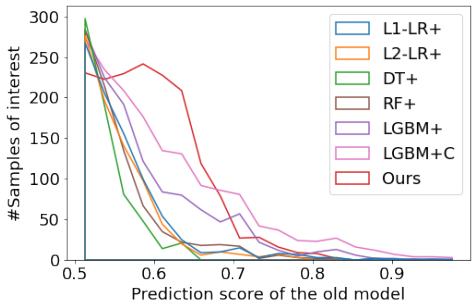
(i) Accuracy (CTG) (j) Precision (CTG) (k) Recall (CTG) (l) F1 (CTG)

(m) Accuracy (7-point) (n) Precision (7-point) (o) Recall (7-point) (p) F1 (7-point)

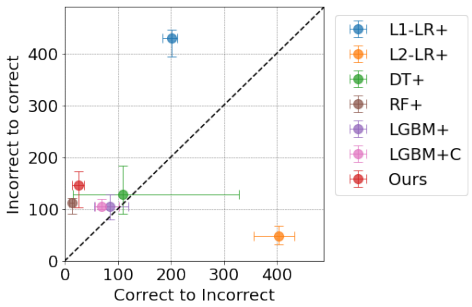
Figure 2: Test accuracy, precision, recall, and F1 scores of each model. Note that, for CTG and 7-point dataset, we used the macro averaged precision, recall, and F1 scores for multiple class labels. "Black" dotted vertical lines indicate the mean values on the old models. "Red" vertical lines indicate the mean values on our LCT. Some of the boxes are very small and are covered by the red vertical line as in (e), (f), (g), and (m).



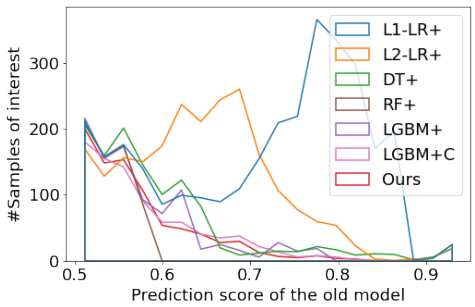
(a) The number of prediction changes (Adult)



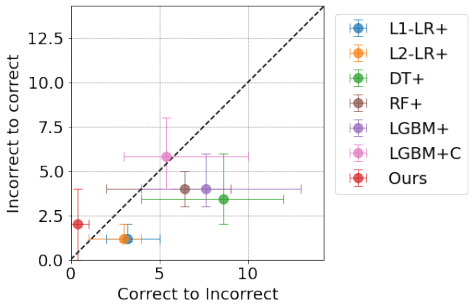
(b) Distribution of prediction changes (Adult)



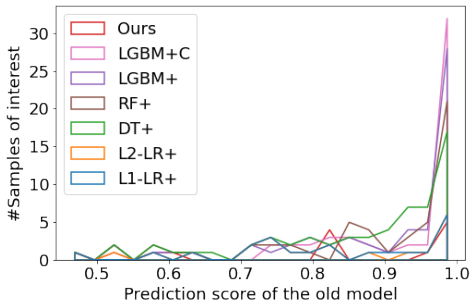
(c) The number of prediction changes (Bank)



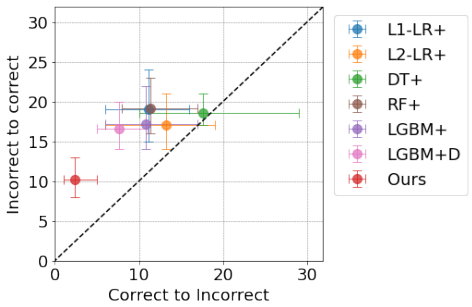
(d) Distribution of prediction changes (Bank)



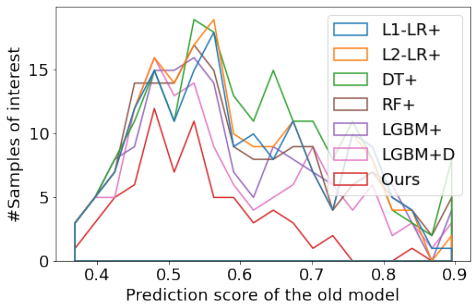
(e) The number of prediction changes (CTG)



(f) Distribution of prediction changes (CTG)

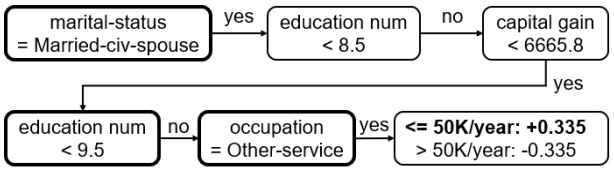


(g) The number of prediction changes (7-point)

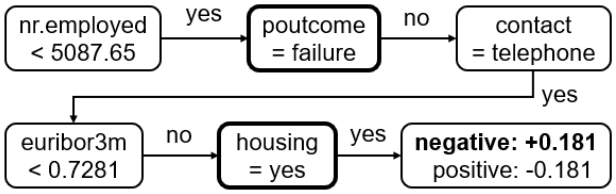


(h) Distribution of prediction changes (7-point)

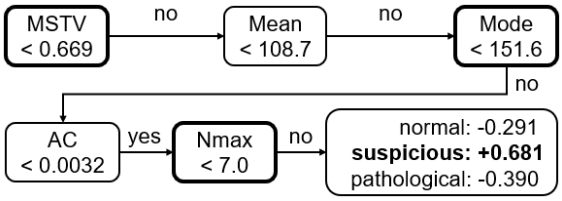
Figure 3: (a), (c), (e), and (g): The horizontal axis indicates the number of samples whose predictions changed from correct to incorrect. The vertical axis indicates the number of samples whose predictions changed from incorrect to correct. (b), (d), (f), and (h): The number of samples, per prediction score of the old model, whose predictions changed.



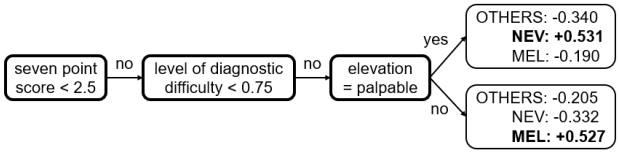
(a) A rule on Adult dataset



(b) A rule on Bank dataset



(c) A rule on CTG dataset



(d) A rule on 7-point dataset

Figure 4: Examples of interesting rules found on our LCT.

larly, from Figure 3 (h), while the other new models changed many predictions of samples with high old scores, our LCT mainly changed predictions of samples with low or middle old scores. We consider that this result is due to the effect of our regularization not allowing large amount of correction. Moreover, we consider this is one of the reasons why the corrections of our LCT performed better.

Result: Concrete Examples of Interesting Rules on LCT. We show four concrete examples of interesting rules found in our LCT in Figure 4. Figures 4 (a), (b), (c), and (d) show the found rules for Adult, Bank, CTG, and 7-point dataset, respectively.

On Adult dataset, we found an interesting rule including the conditions marital-status=Married-civ-spouse, education-num$\geq$9.5, and occupation=Other-service. In this dataset, individuals with a spouse or enough education number tend to have over \$50K/year income. However, the found rule indicates that individuals in the service industry tend to deviate from such a trend more than the old model predicts. We consider this rule is difficult to obtain from the old data consisting of many young individuals with a high likelihood of not having a spouse. Our LCT could successfully find this useful trend missed by the old model.

On Bank dataset, we found an interesting rule including the conditions poutcome$\neq$failure and housing=yes. The condition poutcome$\neq$failure means that the individual has applied for previous campaigns or never had the opportunity to campaign before. Thus, there is a good chance that the individuals will subscribe to the current campaign. However, the found rule indicates that they may be negative individuals if they have housing loan. Because the condition housing=yes tends to be found in old-aged individuals, we consider it is difficult to predict the negative individuals with housing loan on the old model trained on many young-aged individuals. Our LCT could also successfully find such useful trends missed by the old model on another dataset.

On CTG dataset, we found an interesting rule including the conditions MSTV$\geq$0.669, Mode$\geq$151.6, and Nmax$\geq$7. The condition MSTV$\geq$0.669 means that the fetus has moderate short term variability. Although such fetuses tend to be normal, the rule indicates that the fetuses may be suspicious if Mode$\geq$151.6 and Nmax$\geq$7. Here Nmax$\geq$7 means that the fetus will often have maximum beats per minute for them. We consider Nmax$\geq$7 is a suspicious sign for fetuses tending to have high beats per minute. Since the old model had little information of such minority fetuses, our LCT could successfully find a minority trend missed by the old model.

On 7-point dataset, we found an interesting rule including the conditions seven-point-score$\geq$2.5 (i.e., high seven-point-score) and level-of-diagnostic-difficulty$\geq$0.75 (i.e., too difficult to diagnose). Here, a high seven-point-score indicates that such patients are likely to be classified as MEL based on clinical knowledge. The found rule provides an insight for the information that the old model was missing for the high-stake patients with high diagnostic difficulty: the rule indicates that the state of elevation may be something important that is missed by the old model. If elevation=palpable, the patient could be classified into NEV, and MEL otherwise. We consider the state of elevation was difficult to distinguish for the old model that uses image data only. Our LCT could successfully identify the information missed by the old model, and incorporated that information from the new data appropriately for model correction.

Conclusion

We proposed the model correction method LCT that can be applied to any classification model without access to old training data. LCT provides the following advantages: (i) because LCT is a variety of decision trees, the specification of the correction is explicit, and (ii) thanks to the regularization effect, the number of samples to which the correction applies is small.

Experiments showed that LCT could correct the old models better than other standard methods with small prediction changes. Moreover, in constructed LCTs, we observed some interesting rules which seem to be missed by the old models.

References

Akiba, T.; Sano, S.; Yanase, T.; Ohta, T.; and Koyama, M. 2019. Optuna: A Next-Generation Hyperparameter Optimization Framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2623–2631.

Breiman, L.; Friedman, J. H.; Olshen, R. A.; and Stone, C. J. 2017. *Classification and regression trees*. Routledge.

Chao, S.; and Wong, F. 2009. An incremental decision tree learning methodology regarding attributes in medical data mining. In *Proceedings of the 2009 International Conference on Machine Learning and Cybernetics*, 1694–1699.

Dai, W.; Yang, Q.; Xue, G.; and Yu, Y. 2007. Boosting for Transfer Learning. In *Proceedings of the 24th International Conference on Machine Learning*, 193–200.

Daumé III, H. 2007. Frustratingly Easy Domain Adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, 256–263.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *Proceedings of the 2009 IEEE conference on computer vision and pattern recognition*, 248–255.

Hastie, T.; Tibshirani, R.; and Wainwright, M. 2015. *Statistical Learning with Sparsity: The Lasso and Generalizations*. Chapman and Hall/CRC.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.

Kawahara, J.; Daneshvar, S.; Argenziano, G.; and Hamarneh, G. 2019. Seven-Point Checklist and Skin Lesion Classification Using Multitask Multimodal Neural Nets. *IEEE Journal of Biomedical and Health Informatics*, 23(2): 538–546.

Ke, G.; Meng, Q.; Finley, T.; Wang, T.; Chen, W.; Ma, W.; Ye, Q.; and Liu, T.-Y. 2017. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 3149–3157.

Li, M.; Kuang, K.; Zhu, Q.; Chen, X.; Guo, Q.; and Wu, F. 2020a. IB-M: A Flexible Framework to Align an Interpretable Model and a Black-box Model. In *Proceedings of the 2020 IEEE International Conference on Bioinformatics and Biomedicine*, 643–649.

Li, R.; Qin, Z.; Wang, X.; Chen, S. J.; and Metzler, D. 2020b. Stabilizing Neural Search Ranking Models. In *Proceedings of The Web Conference 2020*, 2725–2732. Association for Computing Machinery.

Mackie, R.; and Doherty, V. 1991. Seven-point checklist for melanoma. *Clinical and Experimental Dermatology*, 16(2): 151–152.

Pan, D.; Wang, T.; and Hara, S. 2020. Interpretable Companions for Black-Box Models. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, 2444–2454.

Pan, S. J.; and Yang, Q. 2010. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10): 1345–1359.

Parisi, G. I.; Kemker, R.; Part, J. L.; Kanan, C.; and Wermter, S. 2019. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113: 54–71.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Kopf, A.; Yang, E.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; and Chintala, S. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc.

Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; and Duchesnay, E. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12: 2825–2830.

Quinlan, J. R. 1993. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc.

Rafique, H.; Wang, T.; Lin, Q.; and Singhani, A. 2020. Transparency Promotion with Model-Agnostic Linear Competitors. In *Proceedings of the 37th International Conference on Machine Learning*, 7898–7908.

Ristin, M.; Guillaumin, M.; Gall, J.; and Van Gool, L. 2016. Incremental Learning of Random Forests for Large-Scale Image Classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(3): 490–503.

Sugiyama, M.; Suzuki, T.; and Kanamori, T. 2012. *Density Ratio Estimation in Machine Learning*. USA: Cambridge University Press, 1st edition.

Utgoff, P. E. 1988. ID5: An Incremental ID3. In *Proceedings of the 5th International Conference on Machine Learning*, 107–120.

Voigt, P.; and Bussche, A. v. d. 2017. *The EU General Data Protection Regulation (GDPR): A Practical Guide*. Springer Publishing Company, Incorporated, 1st edition.

Wang, T. 2019. Gaining Free or Low-Cost Interpretability with Interpretable Partial Substitute. In *Proceedings of the 36th International Conference on Machine Learning*, 6505–6514.

Xiao, L. 2010. Dual Averaging Methods for Regularized Stochastic Learning and Online Optimization. *Journal of Machine Learning Research*, 11(88): 2543–2596.

Zhang, C.; Zhang, Y.; Shi, X.; Almpanidis, G.; Fan, G.; and Shen, X. 2019. On Incremental Learning for Gradient Boosting Decision Trees. *Neural Processing Letters*, 50: 957–987.

Zhuang, F.; Qi, Z.; Duan, K.; Xi, D.; Zhu, Y.; Zhu, H.; Xiong, H.; and He, Q. 2021. A Comprehensive Survey on Transfer Learning. *Proceedings of the IEEE*, 109(1): 43–76.