JOURNAL OF QUALITY TECHNOLOGY
https://doi.org/10.1080/00224065.2021.1963200

Robustness with respect to class imbalance in artificial intelligence classification algorithms

Jiayi Lian, Laura Freeman, Yili Hong, and Xinwei Deng

Department of Statistics, Virginia Tech, Blacksburg, Virginia

ABSTRACT

Artificial intelligence (AI) algorithms, such as deep learning and XGboost, are used in numerous applications including autonomous driving, manufacturing process optimization and medical diagnostics. The robustness of AI algorithms is of great interest as inaccurate prediction could result in safety concerns and limit the adoption of AI systems. In this paper, we propose a framework based on design of experiments to systematically investigate the robustness of AI classification algorithms. A robust classification algorithm is expected to have high accuracy and low variability under different application scenarios. The robustness can be affected by a wide range of factors such as the imbalance of class labels in the training dataset, the chosen prediction algorithm, the chosen dataset of the application, and a change of distribution in the training and test datasets. To investigate the robustness of AI classification algorithms, we conduct a comprehensive set of mixture experiments to collect prediction performance results. Then statistical analyses are conducted to understand how various factors affect the robustness of AI classification algorithms. We summarize our findings and provide suggestions to practitioners in AI applications.

KEYWORDS

AI assurance; deep learning; design of experiments; distribution shift; safety of AI systems

1. Introduction

Machine learning (ML) and deep learning (DL) algorithms are widely used in many artificial intelligence (AI) applications such as computer vision (Hashimoto et al. 2019), autonomous driving (Tian et al. 2018), and medical diagnostics (Quer et al. 2017). There is a growing interest in investigating the robustness of AI algorithms (Dietterich 2017; Hamon, Junklewitz, and Sanchez 2020; Silva and Najafirad 2020), as it is highly related to the safety of AI systems (Amodei et al. 2016). The training data can be noisy and imbalanced, resulting in performance degradation of the AI algorithm (Ning et al. 2019). Moreover, some AI algorithms could be vulnerable to data poisoning attacks, where adversarial insiders may change the data structures in certain scenarios. Therefore, there is an emerging need to understand how the data quality affects the quality assurance of the AI algorithms. Such a comprehensive understanding will pave a foundation for developing solid solutions to mitigate and remedy data quality problems in AI algorithms.

In this work, we focus on the robustness of AI classification algorithms in terms of how the performance of the algorithm, such as the classification accuracy, may change when the training and test datasets contain differences in class proportions (Xu, Caramanis, and Mannor 2012). The robustness of AI algorithms relies heavily on how the algorithm is trained and the evaluation of the algorithm relies heavily on the test dataset. We focus on investigating the robustness of AI classification algorithms in terms of the classification accuracy based on the area under the curve (AUC). The proposed framework is not restricted to using AUC for examining the robustness, and can be extended to study the robustness for AI algorithms beyond classification.

A robust classification algorithm is expected to have high accuracy and low variability under various scenarios. The robustness of the classification algorithm can be affected by a wide range of factors, such as the composition of training dataset, the change of distribution in the training and test datasets, the chosen prediction algorithm, and so on. In the training stage, AI algorithms often rely on a large number of data points. However, in many real-world applications, data for classification can be highly imbalanced in which data points from a particular class label overwhelmingly dominate data points from other class

CONTACT Xinwei Deng ✉ xdeng@vt.edu Department of Statistics, Virginia Tech, Blacksburg, VA 24061-0131, USA.

labels. It is crucial to understand how the performance of AI classification algorithms is affected by the class label imbalance in the training data. Moreover, when the distribution of test data deviates away from the distribution of the training data, such distribution changes can also largely affect the robustness of the classification algorithms (Kuleshov and Liang 2015). This lack of robustness has direct application to deployed algorithms used on data sets whose composition may differ from the original training data set.

In this paper, we use designed experiments (Wu and Hamada 2011) to systematically investigate the robustness of classification algorithms with respect to: (1) the class label imbalance in the training data; and (2) the distribution change between training and test data regarding the proportions of class labels. In particular, our key idea is to use the mixture experimental design (Cornell 2011) for the proportions of class labels in the training data, such that the robustness of the AI classification algorithms can be investigated in a systematic manner. Here the robustness of a classification algorithm is characterized by the mean and standard deviation of the areas under the receiver operating characteristic curves (AUC) for each class label (Yuan and Bar-Joseph 2019). We also study a wide range of other factors that can affect the robustness, including the distributional changes between the training data and the test data, different classification algorithms, and datasets from different applications. The convolutional neural network (CNN) (Kim 2014) and XGBoost (Chen et al. 2015) are adopted as two representative AI classification algorithms. Based on the classification performance results collected from the designed mixture experiments, we build a statistical surrogate model for the classification accuracy (i.e., AUC) as a function of the mixture proportions, and reveal some interesting findings on the robustness of AI classification algorithms. The resultant model estimation, inference, and prediction provide a set of useful tools to quantify the importance of class proportion, visualize the effect of class imbalance, and to identify conditions under which the algorithms tend to be robust, which can not be easily achieved by existing studies on class imbalance. Note that the proposed framework can be extended to include more factors for investigating the robustness of the AI algorithms, and can also be used to study other characteristics of the AI algorithms, such as the transparency of the AI systems (Hamon, Junklewitz, and Sanchez 2020).

The rest of the paper is organized as follows. Section 2 gives a literature review on related works. Section 3 describes the proposed framework, including the response variables, the experimental factors, design runs, data collection and the modeling method. Section 4 reports the analysis results of experimental data. Section 5 contains some concluding remarks.

2. Related literature

In a typical mixture experiment (Cornell 2011), the design variables under study are the proportions of mixture components in a blend, with their summation equal to unity. A common objective is to investigate how the changes in the proportions of mixture components would affect the response outcomes (Piepel and Cornell 1994). Mixture experiments have been used in many agriculture and engineering applications (Kang, Roshan Joseph, and Brenneman 2011, Kang, Salgado, and Brenneman 2016; Shen, Kang, and Deng 2020). In our work, we adopt the idea of mixture experiments to study how the proportions of class labels would affect the performance of the AI classification algorithms. Among various AI classification algorithms, we consider the convolutional neural network (CNN) and the XGboost as the two representative algorithms. The CNN is a popular deep learning algorithm for classification, where the input variables are tensors (Goodfellow et al. 2016). Generally, the CNN uses convolution in place of general matrix multiplication in its perception layers. In contrast, XGboost is a popular classification algorithm when the input variables are vectors. The XGBoost stands for the Extreme Gradient Boosting, which is a parallel tree boosting method with efficient gradient-based optimization. Both CNN and XGboost are widely used in various application areas (Parsa et al. 2020). With the consideration of mixture experiments for the proportions of class labels, our approach is to provide a comprehensive understanding on the performance of these two classification algorithms.

The robustness of AI algorithms has attracted a large amount of attention in the machine learning community (Tsipras et al. 2018). There is a broad spectrum regarding the robustness of AI algorithms, including adversarial robustness, robust learning, robust models, robustness to distributional shift, etc. In robust learning, a key interest is to redesign the learning procedure so that the algorithm is robust against malicious actions (Madry et al. 2018; Zantedeschi, Nicolae, and Rawat 2017). Robust procedures include better training procedures against adversarial examples, as well as a better mathematical foundation of the algorithms by adopting techniques from statistics and optimization, such as robust statistical inference (Huber 2004; Lozano, Jiang, and Deng 2013) and robust optimization (Ben-Tal, El Ghaoui, and Nemirovski 2009). In the area of deep learning, some research work has focused on investigating the adversarial robustness (Dvijotham et al. 2018; Gehr et al. 2018; Wong and Kolter 2018; Xiang, Tran, and Johnson 2018). It is known that neural network models are vulnerable to adversarial examples. That is, perturbing inputs that are very similar to some regular inputs could result in the output being dramatically different (Szegedy et al. 2014). For example, Tjeng, Xiao, and Tedrake (2017) proposed a mixed integer programming method to examine the vulnerability of neural networks to such adversarial examples. A useful framework for certifying the robustness of CNNs against adversarial examples is discussed in Boopathy et al. (2019). Several works considered the use of experimental designs for tuning the hyper-parameters of AI algorithms (Balestrassi et al. 2009; Packianather, Drake, and Rowlands 2000; Staelin 2003). Different from finding the optimal tuning parameters, the scope of our work is to use an experimental design framework to investigate the robustness of AI algorithms against the class label imbalance in the training data and the distribution change on proportions of class labels between the training and test data.

3. The proposed framework

3.1. Design factors and response variables

Let us consider a multi-class classification with m classes (labels). The accuracy of a classification algorithm can be measured by various performance measures, such as the misclassification error, false positive rate (FPR), F_1 score, etc. Note that the classification rule of a classification algorithm often depends on the setting of the classification threshold (Yuan and Bar-Joseph 2019). Among various performance measures, the area under the receiver operating characteristic (ROC) curve (AUC) provides an aggregate measure of prediction accuracy for all possible thresholds. The ROC curve is a graphical curve by plotting the true positive rate against the false negative rate at various classification threshold settings. There are other metrics not depending on the threshold, such as multiple cross-entropy and Brier score (Hernández-Orallo, Flach, and Ferri Ramírez 2012; Mannor, Peleg, and Rubinstein 2005). However, the multiple cross-entropy appears to be unstable among multiple replications of the same treatment, while the Brier score appears to be highly sensitive to layouts for training and test datasets. For a multi-class classification problem, one can obtain the AUC for each class label, but the multiple cross-entropy and Brier score can only measure the overall performance in all class labels. To characterize the robustness of a classification algorithm, we consider two types of performance measures based on the AUC's. The first one is the sample mean of the AUC values of m classes, and the second one is the logarithm of the standard deviation of the AUC values from m classes.

To investigate the robustness of the AI classification algorithms, we consider a mixture experiment on proportions of class label in the training dataset with several covariate variables such as different algorithms and different datasets of interest. Specifically, let $(x_1, x_2, \ldots, x_m)$ be the proportions of class labels $\{1, \ldots, m\}$ in the training dataset. Note that $x_1 + x_2 + \cdots + x_m = 1$ and $x_j \in [0, 1]$ for $j = 1, \ldots, m$. Suppose that there are h covariate variables, $z_1, \ldots, z_h$, for the mixture experiment. In our specific experimental setting, we consider two covariate variables z_1 and z_2, each with two levels. In particular, z_1 is a two-level factor that represents two different algorithms: the CNN algorithm and the XGboost algorithm. That is,

$$z_1 = \begin{cases} 1, & \text{if the XGboost algorithm is used,} \\ 0, & \text{if the CNN algorithm is used.} \end{cases}$$

The z_2 variable is a two-level factor that represents two different dataset of interests: the KEGG dataset and the Bone Marrow dataset. That is,

$$z_2 = \begin{cases} 1, & \text{if the KEGG dataset is used,} \\ 0, & \text{if the Bone Marrow dataset is used.} \end{cases}$$

The above two datasets are used in Yuan and Bar-Joseph (2019) for the classification of the single cell RNA sequencing (scRNA-seq) expression. In Yuan and Bar-Joseph (2019), the scRNA-seq data are converted into the normalized empirical probability distribution functions (NEPDF) between gene pairs as the input matrix. In particular, the gene expression levels is divided into 32 intervals. Rows represent the gene expression levels of one gene in the gene pair, columns represent the levels of the other gene in the gene pair. Then the authors count the number of gene pairs in each row-column entry, forming a 32 by 32 matrix. Because of the state-of-art ability to handle image data, the CNN is used for classification with the formed matrix as input directly. For the use of XGboost, we manipulated the matrix and created new features, such as column sums and row sums and trace of the matrix. The KEGG dataset is derived from the Kyoto Encyclopedia of Genes and Genomes

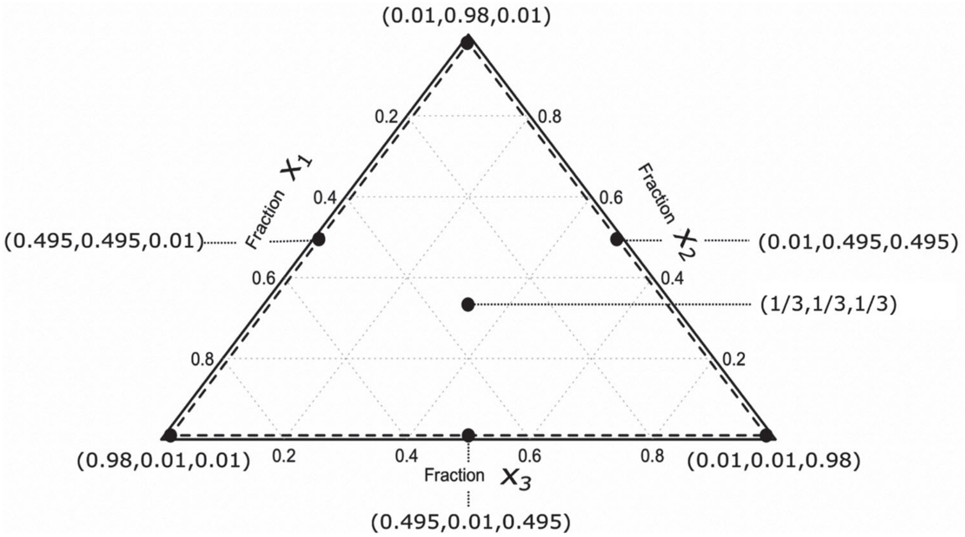

Figure 1. Illustration of seven settings of proportions of class labels in the training dataset. The dashed lines illustrate the restriction on the minimal value of a class proportion larger than 0.01, i.e., $x_j \geq 0.01$.

(KEGG) as pathway datasets of study (Wixon and Kell 2000). The Bone Marrow dataset denotes the bone marrow derived macrophage scRNA-seq as the dataset of study (Orlic et al. 2001). The KEGG dataset contains 92,472 observations and the Bone Marrow dataset has 80,253 observations. Both datasets have $m = 3$ classes with class proportions equally distributed, i.e., each class contains one third of observations. The three classes are labeled as 0, 1, 2, respectively. The class 1 represents that in each gene pair the first gene regulates the second gene pair. The class 2 represents that the second governs the first, and the class 0 represents there is no such interactions between the two genes. The details about the two datasets and their pre-processing can be found in Yuan and Bar-Joseph (2019).

3.2. Design construction and runs

Because the datasets of interest have three class labels, we consider a modified simple centroid design for mixture experiment for $x_1, x_2, \ldots, x_m$ with $m = 3$. A simple centroid design usually has $2^m - 1$ points, including m pure components (e.g., (0,0,1)), $\binom{m}{2}$ binary points (e.g., (0,1/2,1/2)) and $\binom{m}{3}$ ternary mixture (e.g., (1/3,1/3,1/3)). When $m = 3$, there are seven different settings of proportions of class labels for the training dataset, as shown in Figure 1. With the consideration of the covariate variables z_1 and z_2 with two-levels, we use the cross-array between the mixture design of x_1, x_2, x_3 and the full factorial design of z_1 and z_2. The design matrix of our proposed method is shown in Table 1.

Table 1. The 28-run mixture design with 2 covariate factors.

Run	x_1	x_2	x_3	z_1	z_2
1	0.01	0.01	0.98	1	1
2	0.01	0.98	0.01	1	1
3	0.98	0.01	0.01	1	1
4	0.01	0.495	0.495	1	1
5	0.495	0.01	0.495	1	1
6	0.495	0.495	0.01	1	1
7	1/3	1/3	1/3	1	1
8	0.01	0.01	0.98	1	0
9	0.01	0.98	0.01	1	0
10	0.98	0.01	0.01	1	0
11	0.01	0.495	0.495	1	0
12	0.495	0.01	0.495	1	0
13	0.495	0.495	0.01	1	0
14	1/3	1/3	1/3	1	0
15	0.01	0.01	0.98	0	1
16	0.01	0.98	0.01	0	1
17	0.98	0.01	0.01	0	1
18	0.01	0.495	0.495	0	1
19	0.495	0.01	0.495	0	1
20	0.495	0.495	0.01	0	1
21	1/3	1/3	1/3	0	1
22	0.01	0.01	0.98	0	0
23	0.01	0.98	0.01	0	0
24	0.01	0.01	0.98	0	0
25	0.01	0.495	0.495	0	0
26	0.495	0.01	0.495	0	0
27	0.495	0.495	0.01	0	0
28	1/3	1/3	1/3	0	0

Note that our goal is to investigate the robustness of AI classification algorithms on their prediction performance. For the experimental setting of x_1, x_2, x_3, the proportions of class labels in the training dataset, it is not practical to set the proportion of a class label to be zero, i.e., $x_j = 0$ for some j. Under this consideration, we modify the simple centroid design to restrict the minimum value of class proportion to be larger than 0.01, i.e., $x_j \geq 0.01$, which is illustrated by the dashed lines in Figure 1.

To calculate the classification accuracy, a test dataset is often needed. The proportions of class labels in the test dataset also can have an effect on the classification performance of the algorithms. Often time, we assume that the training and test datasets have the same distributions on the proportion of class labels. In practice, the distribution of proportions of class labels in the test dataset can be different from that in the training dataset. To comprehensively evaluate the robustness of the AI classification algorithm, we will

Table 2. Three scenarios of proportions of class labels for forming the test dataset.

Training	x_1	x_2	x_3	Test	x_1	x_2	x_3
			Balanced Scenario				
	0.01	0.01	0.98		1/3	1/3	1/3
	0.01	0.98	0.01		1/3	1/3	1/3
	0.98	0.01	0.01		1/3	1/3	1/3
	0.01	0.495	0.495		1/3	1/3	1/3
	0.495	0.01	0.495		1/3	1/3	1/3
	0.495	0.495	0.01		1/3	1/3	1/3
	1/3	1/3	1/3		1/3	1/3	1/3
			Consistent Scenario				
Training	x_1	x_2	x_3	Test	x_1	x_2	x_3
	0.01	0.01	0.98		0.01	0.01	0.98
	0.01	0.98	0.01		0.01	0.98	0.01
	0.98	0.01	0.01		0.98	0.01	0.01
	0.01	0.495	0.495		0.01	0.495	0.495
	0.495	0.01	0.495		0.495	0.01	0.495
	0.495	0.495	0.01		0.495	0.495	0.01
	1/3	1/3	1/3		1/3	1/3	1/3
			Reverse Scenario				
Training	x_1	x_2	x_3	Test	x_1	x_2	x_3
	0.01	0.01	0.98		0.495	0.495	0.01
	0.01	0.98	0.01		0.495	0.01	0.495
	0.98	0.01	0.01		0.01	0.495	0.495
	0.01	0.495	0.495		0.98	0.01	0.01
	0.495	0.01	0.495		0.01	0.98	0.01
	0.495	0.495	0.01		0.01	0.01	0.98
					*0.01	0.01	0.98
	1/3	1/3	1/3		*0.01	0.98	0.01
					*0.98	0.01	0.01

*Randomly select one from the three proportion settings.

alter the distribution of proportions of class labels in the test dataset to be possibly different from the distribution of class proportions in the training dataset. Specifically, three scenarios for the test dataset are considered, which are listed as follows.

- **Balanced Scenario**: The proportions of class labels in the test dataset are equal in each class, regardless of the proportions of class labels in the training dataset.
- **Consistent Scenario**: The proportions of class labels in the test dataset are the same as the proportions of class labels in the training dataset.
- **Reverse Scenario**: The proportion of class labels in the test dataset is in a reverse pattern as the proportion of class labels in the training dataset.

In the Reverse Scenario, if the proportion of a class label is high in the training dataset, then the proportion of this class label is set to be low in the test dataset. Table 2 shows the three scenarios of proportions of class labels between the training and test dataset.

Note that for all experimental runs, we keep the size of the training dataset constant at the proportion of $n_{tr} = 10\%$ *of the full dataset*, and keep the size of the test dataset constant at the proportion of $n_{ts} = 25\%$ *of the full dataset*. Here, we make the size of the test dataset larger than that of the training dataset for the following considerations. First, it allows the evaluation of classification performance at the test datasets being valid, avoiding the potential variation due to the small sample size. Second, a relatively smaller size of the training dataset could better serve the purpose of investigating the robustness of the algorithms. In the full datasets of the KEGG (of the size $92,472$ observations) and Bone Marrow (of the size $80,253$ observations), the class proportions are equally distributed, i.e., each label having one third of the full data. In each experimenting run, when forming the training dataset, we randomly sample with replacement from the full dataset in each class label based on the proportion setting (x_1, x_2, x_3) for training. The use of sampling with replacement for forming the training dataset is to simulate the possible duplicated data points in practice (Li et al. 2021). For forming the test dataset, we randomly sample without replacement from the remaining data in each class label based on the proportion setting (x_1, x_2, x_3) for test. We use the sampling without replacement to make each data point unique in the test dataset.

To illustrate how data points with its percentage for each label are composed in both training and test datasets, we show a toy example as follows. Suppose that there are $N = 10000$ observations in the full dataset with three class proportions equally distributed. Then we set the size of training dataset to be fixed at $n_{tr} = 10\% \cdot N = 1000$, while the size of test dataset is set to be $n_{ts} = 25\% \cdot N = 2500$. Assume that in one experimental run, the proportions of class labels in the training dataset is designed to be $(x_1, x_2, x_3) = (0.01, 0.01, 0.98)$, and the proportions of class labels in the test dataset is designed to be $(x_1, x_2, x_3) = (0.01, 0.01, 0.98)$. Based on the full dataset, the training dataset is composed by randomly sampling (10, 10, 980) observations from the three classes of data points, respectively. Then using the remaining data points, the test dataset is composed by randomly sampling (25, 25, 2450) observations from the three classes of the remaining data points, respectively.

3.3. Data collection and modeling method

As shown in Table 1, there are 28 treatments (i.e., runs) in the proposed design under each scenario of test data. For each treatment, we conduct three replications. All experiments were carried out on a DGX-2 machine which is a product of NVIDIA focusing on deep learning implementation. On average, each experiment takes around 30 minutes in computation. The outcome of an experiment is the AUC value of each class on the test dataset by a selected classification algorithm (CNN or XGboost), which is estimated based on the training data. Note that for the classification algorithms, the input of the CNN model is the matrix of the normalized empirical probability distribution functions (NEPDF) between gene pairs, which is an image-type input. We adopt the same hyperparameter settings of the CNN as in Yuan and Bar-Joseph (2019), which tried different settings including the number of layers and the partition of gene expression levels. More details on the setting of the hyper-parameters can be found in the Appendix of Yuan and Bar-Joseph (2019). For the XGboost, we have tried to tune hyper-parameters over a high dimensional candidate set every time we change the class proportions in the training data or the datasets in use (z_2). One could choose the setting with the highest average AUC score. Our empirical study found that, for a training dataset with a given class proportion, the XGBoost with hyper-parameters tuned under the balanced class proportions (i.e., $(1/3, 1/3, 1/3)$) produces almost the same prediction accuracy as that with hyper-parameters tuned under this given class proportion. Thus, the hyper-parameters in the XGboost are tuned under the balanced class proportions for each dataset in use (z_2).

Based on the AUC value of each class, denoted as $\eta_1, ..., \eta_m$, then the averaged AUC (mean AUC) can be obtained as

$$\bar{\eta} = \frac{1}{m}\sum_{j=1}^{m}\eta_j,$$

and the logarithm of the standard deviation (Log SD) can be obtained as

$$\text{LogSD} = \log\left(\left[\frac{1}{m-1}\sum_{j=1}^{m}(\eta_j - \bar{\eta})^2\right]^{1/2}\right) .$$

When the outcomes of designed experiments (i.e., the collected data) are obtained, our key interest is to conduct proper statistical modeling to understand how the design factors (class label proportions $x_1, ..., x_m$, the algorithms, and the datasets of interest) affect the response (the classification accuracy). Note that we have three test scenarios, the Consistent Scenario, the Balanced Scenario, and the Reverse Scenario. We will conduct the statistical analysis of experimental data separately for each scenario.

Given a given test scenario, we denote $(x_{i1}, x_{i2}, ..., x_{im})$ as the proportions of class labels of the training dataset for the ith run, and z_{ik} to be the level of covariate variable z_k for the ith run. Let y_i be the corresponding response variable, which can be the mean AUC or the Log SD. Let n be the sample size of the collected data under a given test scenario. To analyze the collected data $(y_i, x_{i1}, x_{i2}, ..., x_{im}, z_{i1}, ..., z_{ih})$, we consider the following regression model,

$$\begin{aligned} y &= \sum_{j=1}^{m}(\beta_j + \sum_{k=1}^{h}\gamma_{kj}z_k)x_j + \sum_{j<j'}\beta_{jj'}x_jx_{j'} + \sum_{k<k'}\delta_{kk'}z_kz_{k'} + \epsilon \\ &= \sum_{j=1}^{m}\beta_jx_j + \sum_{j<j'}\beta_{jj'}x_jx_{j'} + \sum_{k=1}^{h}\sum_{j=1}^{m}\gamma_{kj}z_kx_j + \sum_{k<k'}\delta_{kk'}z_kz_{k'} + \epsilon, \end{aligned} \tag{1}$$

where β_j's and $\beta_{jj'}$'s are regression coefficients for the main and interaction terms of the label proportions, respectively. The γ_{kj}'s are the regression coefficients for the interaction between class label proportions and the covariate variables, and $\delta_{kk'}$'s are the regression coefficients for the interactions of the covariate variables. The $\epsilon \sim \text{N}(0, \sigma^2)$ is the error term. Here, the model does not include the high-order interaction terms to keep the model parsimonious.

Denote $\boldsymbol{\beta}$ to be the vector of coefficients including β_0, β_j's and $\beta_{jj'}$'s. Similarly, we define $\boldsymbol{\gamma}$ and $\boldsymbol{\delta}$ to be the vectors of coefficients for γ_{kj}'s and $\delta_{kk'}$'s, respectively. For the parameter estimation, we adopt the maximum likelihood estimation, which is equivalent to the least squares estimation in our problem. That is,

$$\begin{aligned} (\hat{\boldsymbol{\beta}}, \hat{\gamma}, \hat{\boldsymbol{\delta}}) = \arg\min_{\boldsymbol{\beta}, \gamma, \boldsymbol{\delta}} \sum_{i=1}^{n}(y_i - \sum_{j=1}^{m}\beta_jx_{ij} - \sum_{j<j'}\beta_{jj'}x_{ij}x_{ij'} \\ - \sum_{k=1}^{h}\sum_{j=1}^{m}\gamma_{kj}z_kx_{ij} - \sum_{k<k'}\delta_{kk'}z_{ik}z_{ik'})^2. \end{aligned}$$

Note that the model in (1) does not include an intercept term because the class label proportions sum up to one, i.e., $x_1 + \cdots + x_m = 1$. The model in (1) also does not include the quadratic term of label proportions since it can be linearly expressed by its main effect and its interactions with other class label proportions, i.e., $x_j^2 = x_j - \sum_{j' \neq j} x_jx_{j'}$. Moreover, the model in (1) does not include the main effect of the covariate factor since it can be linearly expressed by the interactions between class label proportions and the covariate variables, i.e., $z_k = z_kx_1 + \cdots + z_kx_m$. To make proper inference on the contribution of z_k in the estimated model, one could include z_k into the regression model in (1), which leads to the following terms in the model,

$$\begin{aligned}\gamma_{k1}z_kx_1 + \gamma_{k2}z_kx_2 + \cdots + \gamma_{km}z_kx_m + \gamma_kz_k \\ = (\gamma_{k1} + \gamma_k)z_kx_1 \\ +(\gamma_{k2} + \gamma_k)z_kx_2 + \cdots + (\gamma_{km} + \gamma_k)z_kx_m.\end{aligned} \tag{2}$$

By imposing the sum-to-zero constraint (Wu and Hamada 2011) for model identifiability, we have

$$\begin{aligned}(\gamma_{k1} + \gamma_k) + \cdots + (\gamma_{km} + \gamma_k) = 0 \Rightarrow \gamma_k \\ = -\frac{1}{m}(\gamma_{k1} + \gamma_{k2} + \cdots + \gamma_{km}).\end{aligned}$$

In this sense, we make inference on z_k based on the linear combination of $\hat{\gamma}_{kj}$'s through $\frac{1}{m}\sum_{j=1}^{m}\hat{\gamma}_{kj}$. It is worth to pointing out that one could also impose the baseline constraint, such as $\gamma_k = 0$ in (2), then it becomes the original model in (1).

Moreover, based on the estimated model, it is useful to quantify the impact of the predictor variables to the response. Here we adopt the SHAP (SHapley Additive exPlanations) approach (Lundberg and Lee 2017) to quantitatively assess the impact of predictor variables to the response. The Shapley value, based on the cooperative game theory (Shapley 1953), can be applied in a wide variety of models and is not affected by the unit of measurement. The SHAP method assigns each predictor variable a Shapley value of importance for the predictive model. For the linear model in (1), the SHAP has an explicit form. The detailed explanations of the Shapely formula under the linear model can be found in the appendix. Denote $\phi_i^{(x_j)}$ as the importance of the label proportion variable x_j to the model output for individual observation i. Similarly, we can define $\phi_i^{(z_kx_j)}, \phi_i^{(x_jx_{j'})}$, $\phi_i^{(z_kz_{k'})}$. For the ith observation in our proposed model, the importance of predictor variables x_j's, $x_jx_{j'}$'s, z_kx_j's, and $z_kz_k's$ have the following forms:

$$\begin{aligned}\phi_i^{(x_j)} &= \beta_j\left(x_{ij} - \frac{1}{n}\sum_{i=1}^{n}x_{ij}\right), \quad j = 1, ..., m; \\ \phi_i^{(x_jx_{j'})} &= \beta_{jj'}\left(x_{ij}x_{ij'} - \frac{1}{n}\sum_{i=1}^{n}x_{ij}x_{ij'}\right), \\ & j < j', j = 1, ..., m, j' = 1, ..., m; \\ \phi_i^{(z_kx_j)} &= \gamma_{kj}\left(z_{ik}x_{ij'} - \frac{1}{n}\sum_{i=1}^{n}z_{ik}x_{ij}\right), \\ & k = 1, ..., h, j = 1, ..., m; \\ \phi_i^{(z_kz_{k'})} &= \delta_{kk'}\left(z_{ik}z_{ik'} - \frac{1}{n}\sum_{i=1}^{n}z_{ik}z_{ik}\right), \\ & k < k', k = 1, ..., h, j' = 1, ..., h.\end{aligned}$$

The SHAP can also be used to quantify the overall impact of each predictor variable by using the average absolute impact on model output magnitude as the evaluation metric; i.e.,

$$\phi^{(x_j)} = \frac{1}{n}\sum_{i=1}^{n}|\phi_i^{(x_j)}|, \quad j = 1, ..., m; \tag{3}$$

$$\begin{aligned}\phi^{(x_jx_{j'})} = \frac{1}{n}\sum_{i=1}^{n}|\phi_i^{(x_jx_{j'})}|, \\ j < j', j = 1, ..., m, j' = 1, ..., m;\end{aligned} \tag{4}$$

$$\phi^{(z_kx_j)} = \frac{1}{n}\sum_{i=1}^{n}|\phi_i^{(z_kx_j)}|, \quad k = 1, ..., h, j = 1, ..., m; \tag{5}$$

$$\begin{aligned}\phi^{(z_kz_{k'})} = \frac{1}{n}\sum_{i=1}^{n}|\phi_i^{(z_kz_{k'})}|, \\ k < k', k = 1, ..., h, j' = 1, ..., h.\end{aligned} \tag{6}$$

These metrics, $\phi^{(x_j)}$'s, $\phi^{(x_jx_{j'})}$'s,$\phi^{(z_kx_j)}$'s, and $\phi^{(z_kx_j)}$'s will be used to assess and compare these impacts to the response in the next section.

4. Data analysis

In this section, we conduct the modeling and data analysis when the response variable is the mean AUC and the Log SD, respectively. In our analysis with $m = 3$ class labels and $h = 2$ covariate variables, the collected data of experiments are $(y_i, x_{i1}, x_{i2}, x_{i3}, z_{1i}, z_{2i}), i = 1, ..., n$. Then the model in (1) is expressed as,

$$\begin{aligned}y_i = \beta_1x_{i1} + \beta_2x_{i2} + \beta_3x_{i3} + \beta_{12}x_{i1}x_{i2} \\ +\beta_{13}x_{i1}x_{i3} + \beta_{23}x_{i2}x_{i3} \\ +\gamma_{11}z_{i1}x_{i1} + \gamma_{12}z_{i1}x_{i2} + \gamma_{13}z_{i1}x_{i3} \\ +\gamma_{21}z_{i2}x_{i1} + \gamma_{22}z_{i2}x_{i2} + \gamma_{23}z_{i2}x_{i3} + \delta_{12}z_{i1}z_{i2} + \epsilon_i.\end{aligned} \tag{7}$$

4.1. Data visualization

We first visualize the data from the experimental results for the mean AUC response and the Log SD response, respectively. Figure 2 displays the boxplots of the mean AUC response under different levels of covariate variables and different combinations of label proportions under the three test scenarios. It is seen that for covariate variable z_1, there is a clear difference between the CNN algorithm $(z_1 = 0)$ and the XGboost algorithm $(z_1 = 1)$. For the three test scenarios, it appears that the XGboost algorithm gives a higher value of mean AUC than the CNN algorithm. Such a pattern is more evident under the Consistent Scenario in comparison with other scenarios. But for covariate variable z_2, there is not a clear difference on the mean AUC between using the KEGG dataset

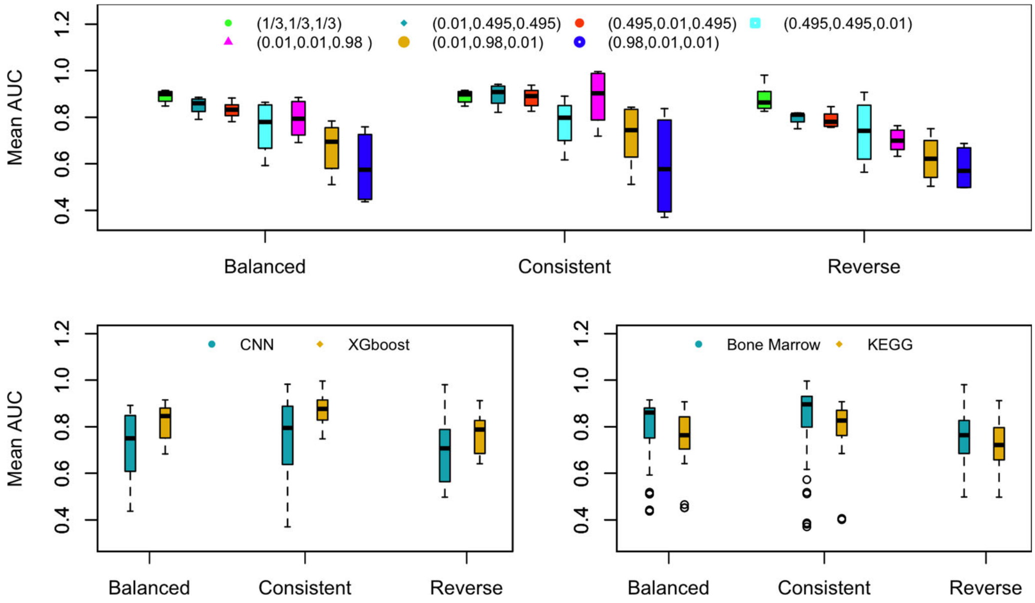

Figure 2. Boxplots of the mean AUC response under different levels of covariate variable and different combinations of label proportions for the three test scenarios.

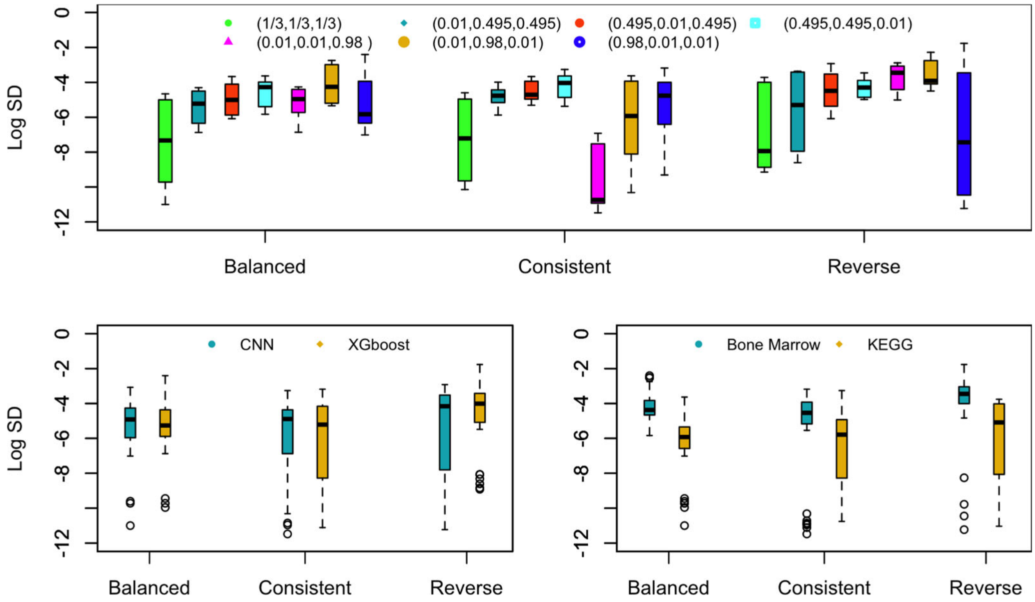

Figure 3. Boxplots of the Log SD response under different levels of covariate variable and different combinations of label proportions for the three test scenarios.

($z_2 = 0$) and using the Bone Marrow data ($z_2 = 1$), although using the KEGG dataset gives a slightly smaller Log SD than using the Bone Marrow dataset. A possible explanation is that the two datasets are of similar data characteristics with respect to the overall classification accuracy, as described in Yuan and Bar-Joseph (2019).

For the boxplots of the mean AUC response under different combinations of label proportions, it is seen that when the label proportions are balanced (i.e., $(x_1, x_2, x_3) = (1/3, 1/3, 1/3)$), the performance of mean AUC is usually better than those under other settings. It implies that the balance of class label proportions in the training dataset plays an important role for AI classification algorithms to achieve robustly good accuracy. When there are two classes dominating (e.g., $(x_1, x_2, x_3) = (0.01, 0.495, 0.495)$), it is observed that the balance between class label 2 and label 3 (x_2 and x_3) gives better performance of mean AUC than the balance between class label 1 and label

Table 3. Parameter estimation and testing statistics under the Balanced Scenario.

Mean AUC analysis					Log SD analysis				
Coef	Est	SE	*t*-value	*p*-value	Coef	Est	SE	*t*-value	*p*-value
x_1	0.4400	0.0173	25.440	<0.001	x_1	−4.631	0.565	−8.202	<0.001
x_2	0.5455	0.0173	31.546	<0.001	x_2	−2.714	0.565	−4.806	<0.001
x_3	0.8599	0.0173	49.764	<0.001	x_3	−3.937	0.565	−6.997	<0.001
x_1x_2	0.5989	0.0513	11.627	<0.001	x_1x_2	−3.228	1.677	−1.919	0.059
x_1x_3	0.6472	0.0513	12.604	<0.001	x_1x_3	−2.512	1.677	−1.498	0.139
x_2x_3	0.5512	0.0513	10.734	<0.001	x_2x_3	−6.725	1.677	−4.011	<0.001
x_1z_1	0.2532	0.0195	12.975	<0.001	x_1z_1	1.288	0.637	2.022	0.047
x_2z_1	0.1660	0.0195	8.506	<0.001	x_2z_1	−0.218	0.637	−0.343	0.733
x_3z_1	−0.0148	0.0195	−0.758	0.451	x_3z_1	0.179	0.637	0.281	0.779
x_1z_2	0.0241	0.0195	1.233	0.222	x_1z_2	−1.727	0.637	−2.711	0.008
x_2z_2	0.0744	0.0195	3.815	<0.001	x_2z_2	−1.721	0.637	−2.701	0.009
x_3z_2	−0.1186	0.0195	−6.088	<0.001	x_3z_2	−1.850	0.637	−2.909	0.005
z_1z_2	−0.0414	0.0160	−2.593	0.012	z_1z_2	−0.611	0.521	−1.172	0.245
Implied effect for z_1 and z_2									
z_1	0.1348	0.0113	11.929	<0.001	z_1	0.416	0.369	1.127	0.263
z_2	−0.007	0.0113	−0.619	0.538	z_2	−1.766	0.369	−4.785	<0.001

2 (x_1 and x_2). When there is only one class dominating (e.g., $(x_1, x_2, x_3) = (0.01, 0.01, 0.98)$), we observe that the performance under the class label 3 dominating is better than the performance under the class label 1 or label 2 dominating. This interesting pattern implies that the class label 3 plays a most substantial role among the three classes for the classification accuracy in terms of the mean AUC.

For the investigation of the Log SD as the response, note that a small value of Log SD means small variation of AUC values of three classes. That is, a smaller value of the Log SD is preferred. Similarly to Figure 2, Figure 3 displays the box plots of the Log SD response. For covariate z_1, there is not a clear difference on Log SD between the CNN algorithm ($z_1 = 0$) and the XGboost algorithm ($z_1 = 1$). This implies that the two algorithms perform similarly with respect to the variation of AUC values from three classes. Under the Balanced Scenario of the test dataset, it is seen that the Log SD is more stable (i.e., narrow range on the boxplots) than those under the other two scenarios for covariate variable z_1. But for covariate variable z_2, the Log SD under the KEGG dataset ($z_2 = 0$) is generally smaller than the Log SD under the Bone Marrow dataset ($z_2 = 1$). In contrast to z_2 being not that significant for the mean AUC, the significance of z_2 for the Log SD implies that the KEGG dataset may provide more equally amount of information for three classes than the Bone Marrow dataset.

For the boxplots of the Log SD response under different combinations of label proportions, the general pattern is similar to the pattern for the boxplots of the mean AUC response. Specifically, the setting of balanced proportions of class label in the training dataset gives smaller Log SD than other settings. It indicates that the balance among class labels in the training dataset could help algorithms to obtain a small variation of the AUC values of three classes. When the proportions are dominated by two classes (e.g., $(x_1, x_2, x_3) = (0.01, 0.495, 0.495)$), the setting gives slightly lower Log SD than the other two settings. The interesting patterns are observed when there is only one class prevailing. Under the Reverse Scenario, the range of Log SD for $(0.98, 0.01, 0.01)$ is the greatest, which may imply that such a setting could result in the most unstable classification performance. Under the Consistent Scenario, the Log SD for the setting $(0.01, 0.01, 0.98)$ is the smallest even compared to the settings under the Balanced Scenario. One possible explanation is that when the class label 3 is dominating, the variation of AUC values for three classes becomes consistently low.

4.2. Modeling results

In this section, we report analysis results of the regression model in Eq. [1] for the mean AUC and the Log SD as the response, respectively. By fitting the regression model with the response of interest, we obtain estimated coefficients and their corresponding t-statistic and p-value. The prediction performance is also investigated for different class label proportions. The impact of predictor variables is assessed by the SHAP method.

Table 3 reports the analysis results for the Balanced Scenario. For the mean AUC, among significant factors with p-values less than 0.05, the variable x_3 has the largest t statistic. It confirms our observation from the data visualization that the class label 3 play a significant role in the classification accuracy. We also see that the estimated coefficients of interactions between two proportions (i.e. $x_jx_{j'}$'s) are all positive. It implies that balance in the training dataset will increase the accuracy of the classification algorithms. For covariate

Table 4. Parameter estimation and testing statistics under the Consistent Scenario.

Mean AUC analysis					Log SD analysis				
Coef	Est	SE	*t*-value	*p*-value	Coef	Est	SE	*t*-value	*p*-value
x_1	0.4122	0.0273	17.386	<0.001	x_1	−3.649	0.836	−4.367	<0.001
x_2	0.5833	0.0273	24.604	<0.001	x_2	−5.523	0.836	−6.608	<0.001
x_3	1.0009	0.0273	42.251	<0.001	x_3	−10.448	0.836	−12.511	<0.001
x_1x_2	0.5091	0.0704	7.210	<0.001	x_1x_2	1.514	2.481	0.608	0.545
x_1x_3	0.6150	0.0704	8.737	<0.001	x_1x_3	7.079	2.481	2.853	0.006
x_2x_3	0.3838	0.0704	5.453	<0.001	x_2x_3	7.823	2.481	3.153	0.002
x_1z_1	0.3068	0.0267	11.471	<0.001	x_1z_1	−0.102	0.943	−0.109	0.914
x_2z_1	0.1576	0.0267	5.894	<0.001	x_2z_1	1.185	0.943	1.257	0.213
x_3z_1	−0.0366	0.0267	−1.371	0.175	x_3z_1	0.428	0.943	0.455	0.651
x_1z_2	0.0290	0.0267	1.083	0.282	x_1z_2	−1.922	0.943	−2.038	0.045
x_2z_2	0.1085	0.0267	4.058	<0.001	x_2z_2	−1.283	0.943	−1.361	0.178
x_3z_2	−0.1943	0.0267	−7.273	<0.001	x_3z_2	2.498	0.943	2.653	0.010
z_1z_2	−0.0189	0.0219	−0.861	0.392	z_1z_2	−2.002	0.772	−2.594	0.012
Implied effect for z_1 and z_2									
z_1	0.1426	0.0155	9.200	<0.001	z_1	0.504	0.546	0.923	0.359
z_2	−0.0189	0.0155	−1.219	0.227	z_2	−0.236	0.546	−0.432	0.667

Table 5. Parameter estimation and testing statistics under the Reverse Scenario.

Mean AUC analysis					Log SD analysis				
Coef	Est	SE	*t*-value	*p*-value	Coef	Est	SE	*t*-value	*p*-value
x_1	0.4755	0.0205	23.245	<0.001	x_1	−8.713	0.716	−12.162	<0.001
x_2	0.5211	0.0205	25.472	<0.001	x_2	−2.181	0.716	−3.044	0.003
x_3	0.7350	0.0205	35.960	<0.001	x_3	−1.873	0.716	−2.617	0.011
x_1x_2	0.6454	0.0607	10.594	<0.001	x_1x_2	1.594	2.127	0.747	0.458
x_1x_3	0.6990	0.0607	11.509	<0.001	x_1x_3	1.303	2.127	0.613	0.542
x_2x_3	0.6472	0.0607	10.656	<0.001	x_2x_3	−11.262	2.127	−5.295	<0.001
x_1z_1	0.1807	0.0231	7.831	<0.001	x_1z_1	4.950	0.808	6.125	<0.001
x_2z_1	0.1737	0.0231	7.524	<0.001	x_2z_1	−0.463	0.808	−0.573	0.568
x_3z_1	−0.0208	0.0231	−0.901	0.371	x_3z_1	−0.233	0.808	−0.288	0.774
x_1z_2	0.0105	0.0231	0.456	0.650	x_1z_2	−1.132	0.808	−1.400	0.166
x_2z_2	0.0150	0.0231	0.650	0.517	x_2z_2	−1.449	0.808	−1.792	0.077
x_3z_2	−0.0576	0.0231	−2.500	0.015	x_3z_2	−2.635	0.808	−3.265	0.002
z_1z_2	−0.0319	0.0189	−1.688	0.096	z_1z_2	−0.695	0.661	−1.051	0.297
Implied effect for z_1 and z_2									
z_1	0.1112	0.0114	9.754	<0.001	z_1	1.418	0.468	3.030	0.003
z_2	−0.012	0.0114	−1.053	0.296	z_2	−1.739	0.468	−3.716	<0.001

variable z_1 concerning two algorithms, the estimated coefficient of z_1 is positive with a statistically significant effect. It confirms that the XGboost algorithm produces a higher mean AUC in comparison with the CNN algorithm. For covariate variable z_2 concerning the two datasets (i.e., the KEGG and the Bone Marrow datasets), the corresponding p-value is $0.538 > 0.05$, which confirms data visualization that z_2 does not provide much influence on the Mean AUC.

For the Log SD analysis, only the main effects of proportions and several interaction effects are statistically significant. It is seen that covariate z_1 does not have a significant effect for the Log SD, which is consistent with the data visualization. Note that x_2x_3 is significant with the largest t-statistic, which may imply that the proportions of x_2 and x_3 are essential to minimize the dispersion of the AUC values among three classes.

Table 4 reports the analysis results for the Consistent Scenario. It is seen that almost all terms involving the class proportion x_3 are significant for both mean AUC and Log SD. This implies that the proportion of class label 3 is not only of most importance to maintain the classification accuracy, but also important for the variation of the AUC values of the three classes. The covariate variable z_1 is significant with a positive coefficient, indicating that the XGboost algorithm performs better than the CNN algorithm for the mean AUC in the Consistent Scenario. For the Log SD, the estimated coefficient for x_1x_3, which is not significant in the Balanced Scenario, becomes statistically significant. A possible explanation is that the proportion of label 1 becomes more important in the Consistent Scenario than that in the Balanced Scenario. Under the Consistent Scenario, the results in Table 4 show that both covariates z_1 and z_2 seem not having much influence on the Log SD.

Table 5 reports the analysis results for the Reverse Scenario. The significance of x_3 (i.e., the proportion of class 3) has the similar pattern as what we observed in both the Balanced Scenario and the Consistent Scenario. The covariate z_1 is significant for the mean AUC, but it is not significant for the Log SD. In contrast, covariate z_2 is significant for the log SD, but it is not significant for the mean AUC. It is interesting to note that, under the Reverse Scenario, the t-statistic for x_1 is the largest in absolute value for the Log SD. It may indicate that the proportion of label 1 becomes crucial to affect the variation of AUC values among three classes in the Reverse Scenario.

To expand the insights on how the class proportions in the train and test data along with the covariates affect the quality of AI algorithm, we combine the three scenarios of test data composition and treat it as a three-level factor. As reported in Appendix C, we consider a regression model by combining three scenarios of the test data composition as a three-level factor (denoted as a composition factor with two contrasts c_1 and c_2). The model accommodates the interaction effects between the proportions (x_1, x_2, x_3) and the composition factor (c_1, c_2), and interactions between covariate variables (z_1, z_2) and the composition factor (c_1, c_2). The analysis results in Appendix C show that several interactions between the class proportions and the composition factor data are significant, revealing the importance effects of the composition of the test data in terms of the robustness of the AI algorithm. In addition, with respect to four different combinations of covariate variables (z_1 and z_2), separated analyses are conducted to provide additional insights. The results are shown in Appendix B. One can see that the effects of the class proportions can vary under different combinations of covariate variables. Furthermore, we have considered the separate models for every level combinations of covariate variables (z_1, z_2) when the composition factor is included in the model. The results are reported in Appendix D. It is seen that the effects of the class proportions and the compositions of test data can vary under different combinations of covariate variables.

Furthermore, we evaluate the performance of the estimated model at different proportions of class labels (x_1, x_2, x_3). Specifically, given a level combination of covariate variables z_1 and z_2, we provide the triangle contour plots of prediction at (x_1, x_2, x_3). Figure 4 shows the triangle contour plots of prediction for the Mean AUC under different level combinations of the covariate variables. Based on those results in Figure 4, the prediction accuracy of the classification algorithms (i.e., the predicted mean AUC) generally achieves the best performance when the class proportions are balanced (i.e., $(x_1, x_2, x_3 = (1/3, 1/3, 1/3))$. However, comparing the XGboost ($z_1 = 1$) and the CNN ($z_1 = 0$), the XGboost algorithm appears to be more symmetric with respect to three vertex x_1, x_2, and x_3. While the prediction accuracy of the CNN algorithm appears to be more affected by the proportion of class 3. It is noted that, under the setting of the Consistent Scenario, both classification algorithms can achieve the perfect accuracy (i.e., prediction of the mean AUC being 1) under the setting of class proportion x_3 prevailing. One plausible explanation is that data points from class 3 are more important than the data points from the other two classes in terms of affecting the classification accuracy for the KEGG and the Bone Marrow datasets in this study.

Figure 5 shows the triangle contour plots of prediction for the Log SD under different level combinations of the covariate variables. The patterns of the contour plots are generally more heterogeneous across different level combinations of covariate variables z_1 and z_2. One interesting observation is that the prediction pattern in the contour plot is not symmetric with respect to three vertex x_1, x_2, and x_3. It may imply that the data points from the three classes do not equally contribute to the classification performance in terms of the variation of the AUC values from three classes.

Lastly, we examine the impact of predictor variables to the response through the Shapely values in (3)–(6). Table 6 reports the Shapley values of predictor variables in the estimated models for the mean AUC and the Log SD, respectively. From the results in the table, it is clear that the proportion of class label 3, x_3, has the largest impact on the mean AUC for all the three scenarios. While for the Log SD, the proportion of class label 1, x_1, has the largest impact to the response under the Reverse and Balanced Scenarios, and the proportion of class label 3, x_3, has the largest impact to the response under the Consistent Scenario.

4.3. Summary of findings

Based on the data visualization in Section 4.1 and the modeling results in Section 4.2, we provide a summary of the findings as follows.

- The balanced setting of the class label proportions generally gives the best classification accuracy in terms of the mean AUC for all the three test scenarios. This finding is applicable for both the CNN and XGboost algorithms. The predicted mean AUC based on the proposed model in (1) generally

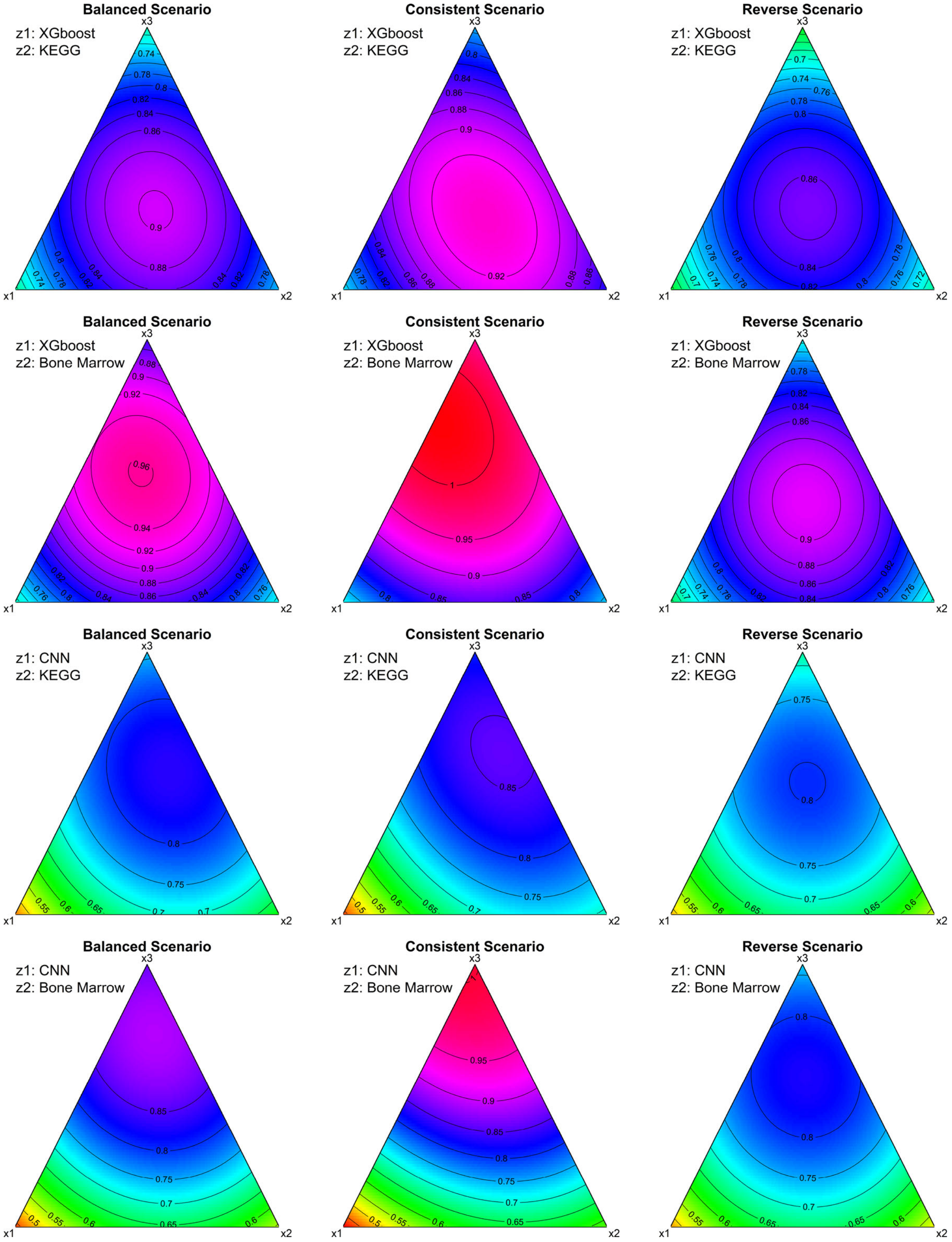

Figure 4. The triangle contour plots of prediction for the Mean AUC under three scenarios.

shows the largest accuracy under the setting of balanced class proportions in the training dataset.

- The proportions of class labels may not be of equal importance for the robustness of the AI algorithms. For the KEGG and Bone Marrow datasets studied in this paper, the proportion of class label 3 in the training data appears to be more important than the other two class labels for classification accuracy in terms of the mean AUC. This finding highlights that some datasets have unequal effects of class labels on the quality assurance of AI algorithm.

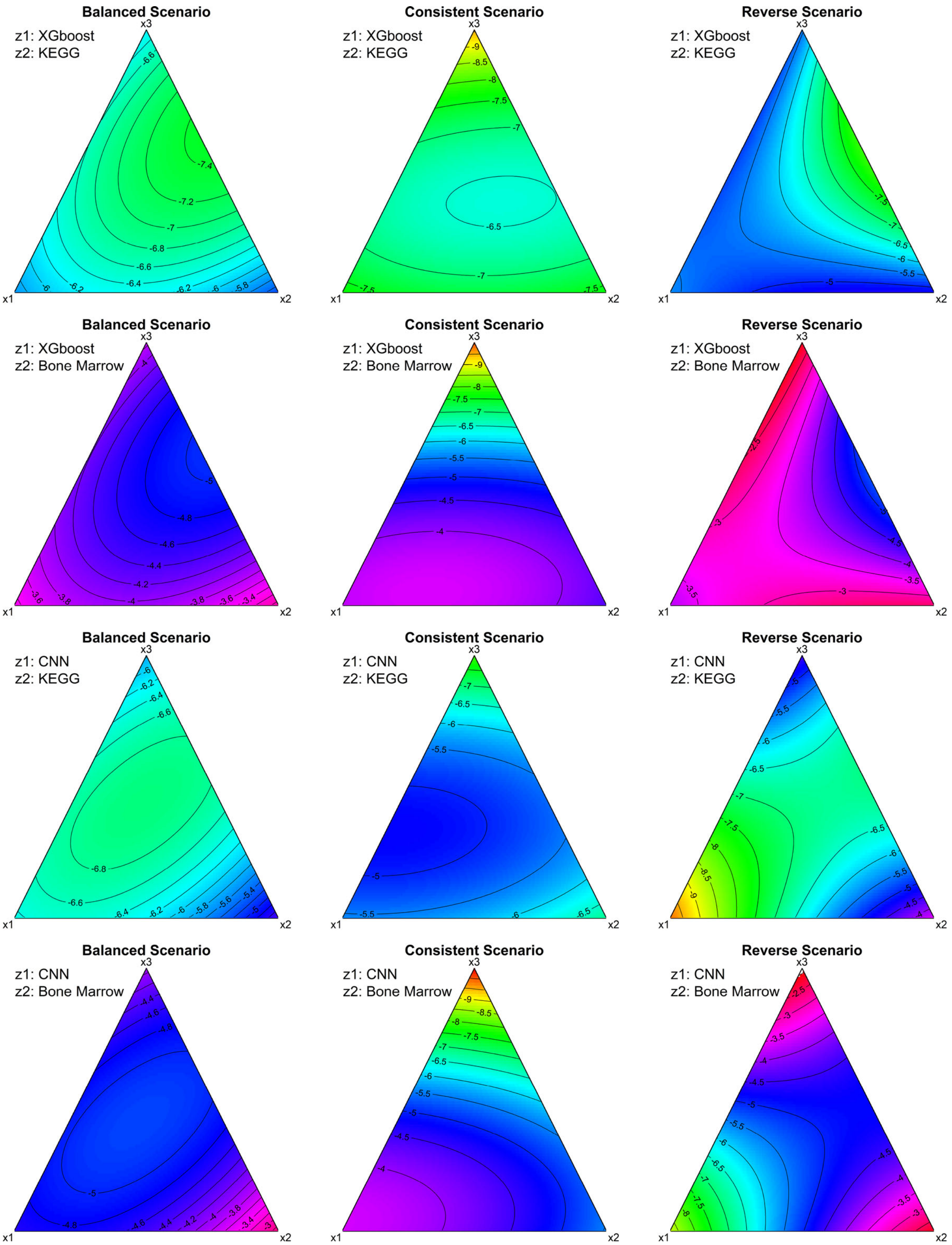

Figure 5. The triangle contour plots of prediction for the Log SD under the three scenarios.

- The magnitude of interaction effects between class proportions and the covariates (i.e., the chosen algorithm and the chosen dataset) varies under different scenarios of training and test datasets. It indicates that the impact of data quality on quality of AI algorithm are influenced by the layouts of training and testing data. The interactions between the class label proportions and the chosen algorithm are often significant for the classification performance in terms of the mean AUC and the Log SD.
- The choice of classification algorithms (i.e., covariate variable z_1) appears to have a

Table 6. The Shapley values of predictor variables in the estimated model for the mean AUC and the Log SD under the three scenarios.

Mean AUC analysis				Log SD analysis			
Coef	Balanced	Consistent	Reverse	Coef	Balanced	Consistent	Reverse
x_1	0.118	0.110	0.127	x_1	**1.239**	0.977	**2.332**
x_2	0.151	0.162	0.145	x_2	0.753	1.533	0.605
x_3	**0.250**	**0.291**	**0.213**	x_3	1.143	**3.033**	0.544
x_1x_2	0.042	0.036	0.045	x_1x_2	0.226	0.106	0.112
x_1x_3	0.046	0.043	0.049	x_1x_3	0.177	0.499	0.092
x_2x_3	0.039	0.027	0.045	x_2x_3	0.473	0.550	0.792
x_1z_1	0.059	0.072	0.042	x_1z_1	0.301	0.024	1.156
x_2z_1	0.039	0.037	0.041	x_2z_1	0.051	0.277	0.108
x_3z_1	0.003	0.009	0.005	x_3z_1	0.042	0.100	0.055
x_1z_2	0.006	0.007	0.002	x_1z_2	0.403	0.449	0.264
x_2z_2	0.017	0.025	0.004	x_2z_2	0.402	0.300	0.338
x_3z_2	0.028	0.046	0.014	x_3z_2	0.434	0.586	0.618
z_1z_2	0.016	0.007	0.012	z_1z_2	0.229	0.751	0.261

statistically significant effect on the response of the mean AUC. There is weak evidence on a significant difference of Log SD between two algorithms.

- For the response Log SD, the patterns of significant factors and predicted values are generally more heterogeneous across different level combinations of the covariate variables. It implies the complex relationship between variability of AUC values and the covariates.

5. Conclusions and directions for further research

In this work, we propose an experimental design framework to systematically investigate how the data quality, such as imbalance among class labels, and distribution shift between training data and test data, affects the quality assurance of the AI classification algorithms. It is to elaborate how the experimental design thinking is used to systematically investigate the performance characteristics of AI algorithms, which is called AI assurance in the general media. The AI assurance is all about ensuring that the AI is going to operate in a proper and risk-controllable manner over time, which broadly includes robustness and reliability, privacy and security, fairness and bias, transparency, and reproducibility. All these aspects are crucial when AI methods are applied in quality problems, such as quality data modeling, and process optimization, to help the engineers and practitioners use the AI properly. For example, it can be used to investigate the model assurance of the crystal growth process in semiconductor manufacturing (Jin et al. 2019; Sun et al. 2016), where the classification method is used for quality prediction.

Although we choose two representative AI classification algorithms, the XGboost and the CNN, with the KEGG and Bone Marrow datasets of three classes in our current study, the proposed framework can be extended for other algorithms and other datasets with multiple classes. Besides the AUC, other classification performance measures such as false positive rate (FPR) and false negative rate (FNR) can also be considered especially when the practitioners have domain knowledge of the classification threshold. When the choice of datasets K is larger than two, the corresponding covariate z_2 will have K levels with $K-1$ degrees of freedom. Then the analysis and inference need to be carefully conducted to include the interactions between class proportions and choice of datasets. It would also be interesting to extend our proposed analysis with a random effect for the choice of datasets. This research also shows the value of building a statistical model to determine not only the significant factors affecting algorithm performance, but also to provide an analytical tool for predicting the classification performance for new test sets with previously unseen proportions of class labels. Note that a well-designed experiment for collecting the data is important to ensure the valid analysis and inference. Moreover, our proposed framework is also suitable to investigate other characteristics of AI algorithms, such as the AI fairness and the AI reliability (Freeman, Medlin, and Johnson 2019; Hong, Zhang, and Meeker 2018).

There are a few directions for the future work. First, it will be interesting to study an optimal design of mixture experiments for the investigation of AI classification algorithms, especially when the number of classes is large. Note that the classical optimal design, such as the I-optimal design, is often based on certain statistical models (Goos, Jones, and Syafitri 2016; Li and Deng 2021). For the investigation of the AI classification algorithm, the design optimality will be associated with both the statistical model and AI algorithms. Second, when the dataset used in the AI classification algorithms involves many classes, it is also challenging to design a good mixture experiment. One possible solution is to consider the batch sequential experiments and active learning (Deng et al. 2009) with the focus on the proportions of class labels dominated by a small number of class labels in each stage. The analysis of such a mixture experiment with a large number of components also requires a sophisticated model such as the cubic mixture models (Piepel and Cornell 1994). Third, the conventional simulation of data generation, often based on multivariate normal, may not adequate for use to demonstrate complexity of AI algorithms. It is interesting to investigate how to properly generate the simulation data for studying the characteristics of AI algorithms. Fourth, the AI robustness can be influenced by the amount of data as well as class proportions. It is possible to extend the current model by including the amount of data as a block factor. Finally, our current analysis of the experiment outcomes is based on linear models. While the experiment outcomes can also be categorical. It will be interesting to adopt other flexible modeling methods for the analysis such as the Gaussian process modeling (Deng et al. 2017).

About the authors

Jiayi Lian is a PHD student of Statistics at Virginia Tech. His research interest is robustness of AI algorithms, interface between experimental design and machine learning, and design and analysis of computer experiments.

Laura Freeman is the director of Intelligent Systems Lab at Hume Center, Virginia Tech. She is a research associate professor of statistics at Virginia Tech. Her research interests are experimental design considerations in machine learning and artificial intelligence, cybersecurity analytics, reliability analytics, statistical engineering.

Yili Hong is Professor of Statistics at Virginia Tech. He is a member of ASQ. His email address is yilihong@vt.edu.

Xinwei Deng is an associate professor in the Department of Statistics at Virginia Tech. He received his PhD degree in industrial engineering from Georgia Tech. His research interests focus on big data analytics, design of experiments, and the interface between experimental design and machine learning.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

The authors acknowledge the Commonwealth Cyber Initiative (CCI) for providing computational resources. This research was a part of Security and Software Engineering Research Center (S2ERC) and was supported by National Science Foundation under Grant CNS-1650540 to Virginia Tech. The work by Hong was also partially supported by the Virginia Tech Institute for Critical Technology and Applied Science (ICTAS) Junior Faculty Award.

ORCID

Jiayi Lian http://orcid.org/0000-0002-6416-3402
Yili Hong http://orcid.org/0000-0003-1720-9540
Xinwei Deng http://orcid.org/0000-0002-1560-2405

References

Amodei, D., C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mane. 2016. Concrete problems in AI safety. *arXiv: 1606.06565*.

Balestrassi, P. P., E. Popova, A. d Paiva, and J. M. Lima. 2009. Design of experiments on neural network's training for nonlinear time series forecasting. *Neurocomputing* 72 (4–6):1160–78. doi: 10.1016/j.neucom.2008.02.002.

Ben-Tal, A., L. El Ghaoui, and A. Nemirovski. 2009. *Robust optimization*, vol. 28. Princeton, NJ: Princeton University Press.

Boopathy, A., T.-W. Weng, P.-Y. Chen, S. Liu, and L. Daniel. 2019. CNN-Cert: An efficient framework for certifying robustness of convolutional neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 3240–47. doi: 10.1609/aaai.v33i01.33013240.

Chen, T., T. He, M. Benesty, V. Khotilovich, and Y. Tang. 2015. Xgboost: Extreme gradient boosting. *R package version 0.4-2*, pages 1–4.

Cornell, J. A. 2011. *Experiments with mixtures: Designs, models, and the analysis of mixture data*, vol. 403. Hoboken, NJ: John Wiley & Sons.

Deng, X., V. R. Joseph, A. Sudjianto, and C. J. Wu. 2009. Active learning through sequential design, with applications to detection of money laundering. *Journal of the American Statistical Association* 104 (487):969–81. doi: 10.1198/jasa.2009.ap07625.

Deng, X., C. D. Lin, K.-W. Liu, and R. Rowe. 2017. Additive Gaussian process for computer models with qualitative and quantitative factors. *Technometrics* 59 (3): 283–92. doi: 10.1080/00401706.2016.1211554.

Dietterich, T. G. 2017. Steps toward robust artificial intelligence. *AI Magazine* 38 (3):3–24. doi: 10.1609/aimag.v38i3.2756.

Dvijotham, K., R. Stanforth, S. Gowal, T. A. Mann, and P. Kohli. 2018. A dual approach to scalable verification of deep networks. In *UAI*, vol. 1, 2.

Freeman, L. J., R. M. Medlin, and T. H. Johnson. 2019. Challenges and new methods for designing reliability experiments. *Quality Engineering* 31 (1):108–21. doi: 10.1080/08982112.2018.1546394.

Gehr, T., M. Mirman, D. Drachsler-Cohen, P. Tsankov, S. Chaudhuri, and M. Vechev. 2018. AI2: Safety and robustness certification of neural networks with abstract interpretation. In *2018 IEEE Symposium on Security and Privacy (SP)*, 3–18. IEEE. doi: 10.1109/SP.2018.00058.

Goodfellow, I., Y. Bengio, A. Courville, and Y. Bengio. 2016. *Deep learning*, vol. 1. Cambridge, MA: MIT Press Cambridge.

Goos, P., B. Jones, and U. Syafitri. 2016. I-optimal design of mixture experiments. *Journal of the American Statistical Association* 111 (514):899–911. doi: 10.1080/01621459.2015.1136632.

Hamon, R., H. Junklewitz, and I. Sanchez. 2020. Robustness and explainability of artificial intelligence. *Publications Office of the European Union*.

Hashimoto, D. A., G. Rosman, E. R. Witkowski, C. Stafford, A. J. Navarette-Welton, D. W. Rattner, K. D. Lillemoe, D. L. Rus, and O. R. Meireles. 2019. Computer vision analysis of intraoperative video: Automated recognition of operative steps in laparoscopic sleeve gastrectomy. *Annals of Surgery* 270 (3):414–21. doi: 10.1097/SLA.0000000000003460.

Hernández-Orallo, J., P. Flach, and C. Ferri Ramírez. 2012. A unified view of performance metrics: Translating threshold choice into expected classification loss. *Journal of Machine Learning Research* 13:2813–69.

Hong, Y., M. Zhang, and W. Q. Meeker. 2018. Big data and reliability applications: The complexity dimension. *Journal of Quality Technology* 50 (2):135–49. doi: 10.1080/00224065.2018.1438007.

Huber, P. J. 2004. *Robust statistics*, vol. 523. Hoboken, NJ: John Wiley & Sons.

Jin, R., X. Deng, X. Chen, L. Zhu, and J. Zhang. 2019. Dynamic quality-process model in consideration of equipment degradation. *Journal of Quality Technology* 51 (3):217–29. doi: 10.1080/00224065.2018.1541379.

Kang, L., V. Roshan Joseph, and W. A. Brenneman. 2011. Design and modeling strategies for mixture-of-mixtures experiments. *Technometrics* 53 (2):125–36. doi: 10.1198/TECH.2011.08132.

Kang, L., J. C. Salgado, and W. A. Brenneman. 2016. Comparing the slack-variable mixture model with other alternatives. *Technometrics* 58 (2):255–68. doi: 10.1080/00401706.2014.985389.

Kim, Y. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014*, October 25-29, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL, 1746–51.

Kuleshov, V., and P. S. Liang. 2015. Calibrated structured prediction. In *Advances in Neural Information Processing Systems*, 3474–82.

Li, Y., and X. Deng. 2021. An efficient algorithm for Elastic I-optimal design of generalized linear models. *Canadian Journal of Statistics* 49 (2):438–70.

Li, Y., X. Deng, S. Ba, W. R. Myers, W. A. Brenneman, S. J. Lange, R. Zink, and R. Jin. 2021. Cluster-based data filtering for manufacturing big data systems. *Journal of Quality Technology*. doi: 10.1080/00224065.2021.1889420

Lozano, A. C., H. Jiang, and X. Deng. 2013. Robust sparse estimation of multiresponse regression and inverse covariance matrix via the l2 distance. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 293–301. doi: 10.1145/2487575.2487667.

Lundberg, S. M., and S.-I. Lee. 2017. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, 4765–74.

Madry, A., A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. 2018. Towards deep learning models resistant to adversarial attacks. In *Proceeding of the 6th International Conference on Learning Representations*.

Mannor, S., D. Peleg, and R. Rubinstein. 2005. The cross entropy method for classification. In *Proceedings of the 22nd International Conference on Machine Learning*, 561–8. doi: 10.1145/1102351.1102422.

Ning, R., C. Wang, C. Xin, J. Li, L. Zhu, and H. Wu. 2019. Capjack: Capture in-browser crypto-jacking by deep capsule network through behavioral analysis. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, 1873–81. IEEE.

Orlic, D., J. Kajstura, S. Chimenti, I. Jakoniuk, S. M. Anderson, B. Li, J. Pickel, R. McKay, B. Nadal-Ginard, D. M. Bodine, et al. 2001. Bone marrow cells regenerate infarcted myocardium. *Nature* 410 (6829):701–5. doi: 10.1038/35070587.

Packianather, M., P. Drake, and H. Rowlands. 2000. Optimizing the parameters of multilayered feedforward neural networks through taguchi design of experiments. *Quality and Reliability Engineering International* 16 (6):461–73. doi: 10.1002/1099-1638(200011/12)16:6<461::AID-QRE341>3.0.CO;2-G.

Parsa, A. B., A. Movahedi, H. Taghipour, S. Derrible, and A. K. Mohammadian. 2020. Toward safer highways, application of XGBoost and SHAP for real-time accident detection and feature analysis. *Accident Analysis and Prevention* 136:105405. doi: 10.1016/j.aap.2019.105405.

Piepel, G. F., and J. A. Cornell. 1994. Mixture experiment approaches: Examples, discussion, and recommendations. *Journal of Quality Technology* 26 (3):177–96. doi: 10.1080/00224065.1994.11979525.

Quer, G., E. D. Muse, N. Nikzad, E. J. Topol, and S. R. Steinhubl. 2017. Augmenting diagnostic vision with AI. *The Lancet* 390 (10091):221. doi: 10.1016/S0140-6736(17)31764-6.

Shapley, L. S. 1953. A value for n-person games. *Contributions to the Theory of Games* 2 (28):307–17.

Shen, S., L. Kang, and X. Deng. 2020. Additive heredity model for the analysis of mixture-of-mixtures experiments. *Technometrics* 62 (2):265–76. doi: 10.1080/00401706.2019.1630010.

Silva, S. H., and P. Najafirad. 2020. Opportunities and challenges in deep learning adversarial robustness: A survey. *arXiv Preprint arXiv:2007.00753*

Staelin, C. 2003. Parameter selection for support vector machines. Hewlett-Packard Company, *Tech. Rep. HPL-2002-354R1*, 1.

Sun, H., X. Deng, K. Wang, and R. Jin. 2016. Logistic regression for crystal growth process modeling through hierarchical nonnegative garrote-based variable selection. *IIE Transactions* 48 (8):787–96. doi: 10.1080/0740817X.2016.1167286.

Szegedy, C., W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. 2014. Intriguing properties of neural networks. In *2nd International Conference on Learning Representations, ICLR 2014*, Banff, AB, Canada, April 14–16, Conference Track Proceedings.

Tian, Y., K. Pei, S. Jana, and B. Ray. 2018. Deeptest: Automated testing of deep-neural-network-driven autonomous cars. In *Proceedings of the 40th International Conference on Software Engineering*, 303–14.

Tjeng, V., K. Xiao, and R. Tedrake. 2017. Evaluating robustness of neural networks with mixed integer programming. In *Proceedings of the 7th International Conference on Learning Representations.*

Tsipras, D., S. Santurkar, L. Engstrom, A. Turner, and A. Madry. 2018. Robustness may be at odds with accuracy. In *Proceedings of the 7th International Conference on Learning Representations.*

Wixon, J., and D. Kell. 2000. The Kyoto encyclopedia of genes and genomes–KEGG. *Yeast* 17 (1):48–55.

Wong, E., and Z. Kolter. 2018. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*, 5286–95. PMLR.

Wu, C. J., and M. S. Hamada. 2011. *Experiments: Planning, analysis, and optimization*, vol. 552. Hoboken, NJ: John Wiley & Sons.

Xiang, W., H.-D. Tran, and T. T. Johnson. 2018. Output reachable set estimation and verification for multilayer neural networks. *IEEE Transactions on Neural Networks and Learning Systems* 29 (11):5777–83. doi: 10.1109/TNNLS.2018.2808470.

Xu, H., C. Caramanis, and S. Mannor. 2012. Sparse algorithms are not stable: A no-free-lunch theorem. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34 (1):187–93. doi: 10.1109/TPAMI.2011.177.

Yuan, Y., and Z. Bar-Joseph. 2019. Deep learning for inferring gene relationships from single-cell expression data. *Proceedings of the National Academy of Sciences*, 116: 27151–27158.

Zantedeschi, V., M.-I. Nicolae, and A. Rawat. 2017. Efficient defenses against adversarial attacks. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, 39–49.

Appendix A. The Shapley formula under the linear model

Suppose that one considers a linear regression model with p predictor variables $x_1, ..., x_p$. Denote the $S_{\text{all}} = \{1, ..., p\}$ to a full index set of predictor variables in the model. That is, the regression model can be written as

$$y|\boldsymbol{x} = f(x_1, ..., x_p) + \epsilon = \sum_{j \in S_{\text{all}}} \beta_j x_j + \epsilon.$$

Denote the $M \subseteq \{1, ..., p\}$ to be an index subset of variables. Let $S_{-j} = \{1, ..., j-1, j+1, ..., p\} = S_{\text{all}} \setminus \{j\}$, which is the full index set except the index j. The Shapley value is to examine the impact of the predictor variable X_k based on the idea of the cooperative game theory (Shapley 1953). According to the Shapley value, the impact of variable x_k is

$$\phi_k = \sum_{M \subseteq S_{-k}} w_M [\nu(M \cup \{k\}) - \nu(M)], \tag{8}$$

where $\nu(\cdot)$ is a metric of information gain or utility function. The w_M a weight is as $1/(p\binom{p-1}{q})$ with $q = \text{card}(M)$. In the context of regression, one popular choice is $\nu(M) = \mathbb{E}(y|\boldsymbol{x}_M)$, which is the expected output of the predictive model, conditional on the predictor values $\boldsymbol{x}_M = \{x_j : j \in M\}$ of this subset. Note that the expectation here is with respect to both y and $\boldsymbol{x}_{\bar{M}}$ where $\boldsymbol{x}_{\bar{M}} = \{x_j : j \notin M\}$. Specifically, we have

$$\nu(M) = \mathbb{E}(y|\boldsymbol{x}_M) = \mathbb{E}_{\boldsymbol{x}_{\bar{M}}}\left[\mathbb{E}_{y|\boldsymbol{x}}\left(\sum_{j \in M} \beta_j x_j + \sum_{j \in \bar{M}} \beta_j x_j + \epsilon\right)\right]$$

$$= \mathbb{E}_{\boldsymbol{x}_{\bar{M}}}\left(\sum_{j \in M} \beta_j x_j + \sum_{j \in \bar{M}} \beta_j x_j\right) = \sum_{j \in M} \beta_j x_j + \sum_{j \in \bar{M}} \beta_j \mathbb{E}(x_j).$$

Consequently, we have

$$\nu(M \cup \{k\}) - \nu(M) = \sum_{j \in M \cup \{k\}} \beta_j x_j + \sum_{j \in \bar{M} \setminus \{k\}} \beta_j \mathbb{E}(x_j)$$

$$-\left[\sum_{j \in M} \beta_j x_j + \sum_{j \in \bar{M}} \beta_j \mathbb{E}(x_j)\right] = \beta_k x_k - \beta_k \mathbb{E}(x_k).$$

Noting that the sum of the weights is 1, we thus have $\phi_k = \beta_k [x_k - \mathbb{E}(x_k)]$.

Appendix B. Separate analysis results

In the following analyses, we conduct the regression models using the experiments at a given level combination of the covariate variables, z_1 and z_2. Based on the mode in (1), the reduced regression model is

$$y = \sum_{j=1}^{m} \beta_j x_j + \sum_{j<j'} \beta_{jj'} x_j x_{j'} + \epsilon.$$

Tables B1–B4 are results of parameter estimation for $(z_1, z_2) = (1,0), (1,1), (0,0), (0,1)$, respectively.

Table B1. Parameter estimation and testing statistics for Xgboost trained on Bone Marrow data.

	Mean AUC analysis					Log SD analysis			
Coef	Est	SE	*t*-value	*p*-value	Coef	Est	SE	*t*-value	*p*-value
Balanced									
x_1	0.7427	0.0050	147.60	<0.001	x_1	−2.315	0.1022	−22.651	<0.001
x_2	0.7690	0.0050	152.85	<0.001	x_2	−2.677	0.1022	−26.184	<0.001
x_3	0.8770	0.0050	174.26	<0.001	x_3	−4.427	0.1022	−43.295	<0.001
x_1x_2	0.4438	0.0232	19.12	<0.001	x_1x_2	−7.455	0.4715	−15.810	<0.001
x_1x_3	0.2769	0.0231	11.97	<0.001	x_1x_3	−4.544	0.4701	−9.666	<0.001
x_2x_3	0.2376	0.0231	10.27	<0.001	x_2x_3	−4.200	0.4701	−8.935	<0.001
Consistent									
x_1	0.8120	0.0078	104.22	<0.001	x_1	−3.257	0.3861	−9.629	<0.001
x_2	0.8187	0.0078	105.09	<0.001	x_2	−3.718	0.3861	−9.629	<0.001
x_3	0.9962	0.0078	129.835	<0.001	x_3	−12.018	0.3861	−31.119	<0.001
x_1x_2	0.0556	0.0359	1.546	0.143	x_1x_2	−3.710	1.781	−2.083	0.054
x_1x_3	0.1031	0.0358	2.876	0.012	x_1x_3	11.695	1.776	6.587	<0.001
x_2x_3	0.1237	0.0358	3.453	0.004	x_2x_3	11.610	1.776	6.539	<0.001
Reversed									
x_1	0.6714	0.0076	89.814	<0.001	x_1	−1.656	0.1077	−15.373	<0.001
x_2	0.7296	0.0076	95.429	<0.001	x_2	−2.282	0.1077	−21.195	<0.001
x_3	0.7496	0.0076	98.024	<0.001	x_3	−3.017	0.1077	−28.008	<0.001
x_1x_2	0.7801	0.0353	22.121	<0.001	x_1x_2	−10.525	0.4968	−21.187	<0.001
x_1x_3	0.4391	0.0352	12.489	<0.001	x_1x_3	−4.877	0.4953	−9.847	<0.001
x_2x_3	0.2582	0.0352	7.343	<0.001	x_2x_3	−2.431	0.4953	−4.909	<0.001

Table B2. Parameter estimation and testing statistics for Xgboost trained on KEGG data.

	Mean AUC analysis					Log SD analysis			
Coef	Est	SE	*t*-value	*p*-value	Coef	Est	SE	*t*-value	*p*-value
Balanced									
x_1	0.6816	0.0063	107.79	<0.001	x_1	−5.626	0.7698	−7.308	<0.001
x_2	0.7280	0.0063	115.13	<0.001	x_2	−4.910	0.7698	−6.378	<0.001
x_3	0.7262	0.0063	114.81	<0.001	x_3	−4.998	0.7698	−6.490	<0.001
x_1x_2	0.5636	0.0291	19.32	<0.001	x_1x_2	−5.574	3.5513	−1.569	0.137
x_1x_3	0.5339	0.0291	18.32	<0.001	x_1x_3	−6.478	3.5404	−1.830	0.087
x_2x_3	0.5154	0.0291	17.72	<0.001	x_2x_3	−10.829	3.5404	−3.059	0.008
Consistent									
x_1	0.7556	0.0059	128.95	<0.001	x_1	−8.501	1.371	−6.202	<0.001
x_2	0.8284	0.0059	141.36	<0.001	x_2	−8.183	1.371	−5.970	<0.001
x_3	0.8138	0.0059	138.84	<0.001	x_3	−8.840	1.371	−6.447	<0.001
x_1x_2	0.3554	0.0270	13.15	<0.001	x_1x_2	6.495	6.323	1.027	0.321
x_1x_3	0.3491	0.0270	12.95	<0.001	x_1x_3	7.308	6.304	1.159	0.264
x_2x_3	0.2771	0.0270	10.28	<0.001	x_2x_3	5.387	6.304	0.855	0.406
Reversed									
x_1	0.7420	0.0197	37.750	<0.001	x_1	−2.409	0.565	−4.263	<0.001
x_2	0.7725	0.0197	39.300	<0.001	x_2	−2.892	0.565	−5.118	<0.001
x_3	0.8743	0.0197	44.468	<0.001	x_3	−6.487	0.565	−11.477	<0.001
x_1x_2	0.4265	0.0907	4.704	<0.001	x_1x_2	−7.230	2.607	−2.774	0.007
x_1x_3	0.2730	0.0904	3.020	0.004	x_1x_3	0.758	2.600	0.292	0.772
x_2x_3	0.2065	0.0904	2.284	0.026	x_2x_3	1.660	2.600	0.639	0.526

Table B3. Parameter estimation and testing statistics for CNN trained on Bone Marrow Data.

Mean AUC analysis					Log SD analysis				
Coef	Est	SE	*t*-value	*p*-value	Coef	Est	SE	*t*-value	*p*-value
Balanced									
x_1	0.4153	0.0134	30.95	<0.001	x_1	−5.869	0.565	−30.913	<0.001
x_2	0.4918	0.0134	36.64	<0.001	x_2	−3.260	0.565	−17.172	<0.001
x_3	0.8380	0.0134	62.42	<0.001	x_3	−4.324	0.565	−22.772	<0.001
x_1x_2	0.6477	0.0619	10.46	<0.001	x_1x_2	−0.881	2.607	−1.006	0.330
x_1x_3	0.8550	0.0617	13.85	<0.001	x_1x_3	4.749	2.600	5.440	<0.001
x_2x_3	0.8997	0.0617	14.58	<0.001	x_2x_3	−3.247	2.600	−3.718	0.002
Consistent									
x_1	0.3549	0.0091	30.95	<0.001	x_1	−4.485	0.9808	−4.572	<0.001
x_2	0.5172	0.0091	36.64	<0.001	x_2	−6.697	0.9808	−6.828	<0.001
x_3	0.9731	0.0091	62.42	<0.001	x_3	−13.136	0.9811	−13.389	<0.001
x_1x_2	0.7363	0.0418	10.46	<0.001	x_1x_2	1.680	4.5247	0.371	0.715
x_1x_3	0.9714	0.0418	13.85	<0.001	x_1x_3	17.963	4.5108	3.982	0.001
x_2x_3	0.7133	0.0418	14.58	<0.001	x_2x_3	20.613	4.5108	4.570	<0.001
Reversed									
x_1	0.4749	0.0322	14.739	<0.001	x_1	−11.3716	0.8998	−12.638	<0.001
x_2	0.5061	0.0322	15.705	<0.001	x_2	−3.7825	0.8998	−4.204	<0.001
x_3	0.7161	0.0322	22.216	<0.001	x_3	−2.8442	0.8998	−3.160	0.006
x_1x_2	0.4023	0.1487	2.706	0.016	x_1x_2	12.708	4.1508	3.062	0.008
x_1x_3	0.8841	0.1482	5.966	<0.001	x_1x_3	12.389	4.1381	2.994	0.009
x_2x_3	0.9118	0.1482	6.153	<0.001	x_2x_3	−3.599	4.1381	−0.870	0.398

Table B4. Parameter estimation and testing statistics for CNN trained on KEGG Data.

Mean AUC analysis					Log SD analysis				
Coef	Est	SE	*t*-value	*p*-value	Coef	Est	SE	*t*-value	*p*-value
Balanced									
x_1	0.4333	0.0082	52.95	<0.001	x_1	−6.1615	1.2392	−4.972	<0.001
x_2	0.6327	0.0082	77.32	<0.001	x_2	−4.4447	1.2392	−3.587	0.003
x_3	0.6903	0.0082	84.33	<0.001	x_3	−5.9060	1.2396	−4.765	<0.001
x_1x_2	0.7402	0.0378	19.61	<0.001	x_1x_2	0.1664	5.7166	0.029	0.977
x_1x_3	0.9229	0.0376	24.52	<0.001	x_1x_3	−4.6231	5.6991	−0.811	0.430
x_2x_3	0.5521	0.0376	14.67	<0.001	x_2x_3	−9.5196	5.6991	−1.670	0.116
Consistent									
x_1	0.3790	0.0077	49.37	<0.001	x_1	−4.448	1.287	−3.457	0.003
x_2	0.6825	0.0077	88.91	<0.001	x_2	−5.843	1.287	−4.541	<0.001
x_3	0.7398	0.0077	96.35	<0.001	x_3	−6.747	1.287	−5.242	<0.001
x_1x_2	0.8889	0.0354	25.10	<0.001	x_1x_2	1.118	5.936	0.188	0.853
x_1x_3	1.0363	0.0353	29.36	<0.001	x_1x_3	−3.806	5.917	−0.643	0.530
x_2x_3	0.4211	0.0353	11.93	<0.001	x_2x_3	−1.257	5.917	−0.212	0.835
Reversed									
x_1	0.4765	0.0130	36.80	<0.001	x_1	−11.335	0.7750	−15.013	<0.001
x_2	0.5433	0.0130	41.97	<0.001	x_2	−3.509	0.7750	−4.648	<0.001
x_3	0.6311	0.0130	48.74	<0.001	x_3	−4.574	0.7750	−6.056	<0.001
x_1x_2	0.7067	0.0597	11.83	<0.001	x_1x_2	10.690	3.483	3.069	0.007
x_1x_3	0.8667	0.0595	14.56	<0.001	x_1x_3	6.093	3.472	1.755	0.100
x_2x_3	0.7137	0.0595	11.99	<0.001	x_2x_3	−18.226	3.472	−5.249	<0.001

Appendix C. Analysis results for the composition factor

Denote a composition factor to be a three-level factor with three levels as the three scenarios of the test data composition, i.e., the Balanced Scenario, the Consistent Scenario, and the Reverse Scenario. For this three-level factor, we use one-hot encoding to create three dummy variables denoted as c_1, c_2, and c_3 each with two levels.

Based on the model in (1), the expanded model for accommodating the composition factor is

$$y = \sum_{j=1}^{m}\beta_j x_j + \sum_{j<j'}\beta_{jj'}x_jx_{j'} + \sum_{k=1}^{h}\sum_{j=1}^{m}\gamma_{kj}z_kx_j + \sum_{k<k'}\delta_{kk'}z_kz_{k'} + \sum_{l=1}^{T}\sum_{j=1}^{m}\zeta_{lj}c_lx_j + \sum_{l=1}^{T}\sum_{k=1}^{h}\eta_{kl}z_kc_l + \epsilon,$$

where ζ_{lj} is coefficient of term c_lx_j and η_{kl} is coefficient of term z_kc_l. Here $T=2$ since we choose c_1 and c_2 into the model and treat c_3 as a baseline. Note that the above model does not include the main effects of c_k because $c_k = c_kx_1 + c_kx_2 + \cdots + c_kx_m$ for $k=1, 2$. The analysis result for this model is reported in Table C1.

Table C1. Parameter estimation and testing statistics including the composition factor.

Mean AUC analysis					Log SD analysis				
Coef	Est	SE	*t*-value	*p*-value	Coef	Est	SE	*t*-value	*p*-value
x_1	0.4535	0.0169	26.831	<0.001	x_1	−7.3067	0.6862	−10.648	<0.001
x_2	0.5239	0.0169	30.996	<0.001	x_2	−3.1707	0.6862	−4.620	<0.001
x_3	0.7924	0.0169	46.930	<0.001	x_3	−4.2082	0.6855	−6.138	<0.001
x_1x_2	0.5845	0.0381	15.330	<0.001	x_1x_2	0.0466	1.548	0.030	0.976
x_1x_3	0.6537	0.0380	17.199	<0.001	x_1x_3	2.4838	1.543	1.609	0.109
x_2x_3	0.5274	0.0380	13.876	<0.001	x_2x_3	−3.0867	1.543	−2.004	0.046
x_1z_1	0.2280	0.0167	13.665	<0.001	x_1z_1	3.087	0.6775	4.556	<0.001
x_2z_1	0.1469	0.0167	8.802	<0.001	x_2z_1	1.051	0.6775	1.551	0.122
x_3z_1	−0.0429	0.0167	−2.575	0.011	x_3z_1	1.079	0.6775	1.594	0.112
x_1z_2	0.0221	0.0167	1.324	0.186	x_1z_2	−1.966	0.6775	−2.902	0.004
x_2z_2	0.0669	0.0167	4.010	<0.001	x_2z_2	−1.851	0.6775	−2.732	0.007
x_3z_2	−0.1225	0.0167	−7.354	<0.001	x_3z_2	−0.737	0.6775	−1.089	0.277
z_1z_2	−0.0307	0.0118	−2.599	0.010	z_1z_2	−1.158	0.4799	−2.413	0.017
x_1c_1	−0.0113	0.0191	−0.590	0.556	x_1c_1	1.845	0.7760	2.378	0.018
x_2c_1	0.0269	0.0191	1.409	0.160	x_2c_1	−0.016	0.7760	−0.021	0.984
x_3c_1	0.0733	0.0191	3.842	<0.001	x_3c_1	−0.831	0.7760	−1.072	0.285
x_1c_2	−0.0214	0.0191	−1.122	0.263	x_1c_2	2.626	0.7760	3.384	<0.001
x_2c_2	0.0514	0.0191	2.691	0.008	x_2c_2	−0.085	0.7760	−1.038	0.300
x_3c_2	0.1454	0.0191	7.617	<0.001	x_3c_2	−3.927	0.7760	−5.067	<0.001
z_1c_1	0.0188	0.0145	1.298	0.196	z_1c_1	−1.032	0.5878	−1.756	0.080
z_2c_1	−0.0009	0.0145	−0.060	0.953	z_2c_1	−0.021	0.5878	−0.036	0.971
z_1c_2	0.0379	0.0145	2.615	0.010	z_1c_2	−1.617	0.5878	−2.752	0.006
z_2c_2	−0.0019	0.0145	−0.130	0.896	z_2c_2	0.992	0.5878	1.689	0.093

Appendix D. Separate analysis with the composition factor

Based on experiments at a given level combination of the covariate variables, z_1 and z_2, we cam also conduct the regression model with the accommodation of the composition factor. Based on the mode in Appendix C, the reduced regression model is

$$y = \sum_{j=1}^{m}\beta_jx_j + \sum_{j<j'}\beta_{jj'}x_jx_{j'} + + \sum_{l=1}^{T}\sum_{j=1}^{m}\zeta_{lj}c_lx_j + \epsilon,$$

Tables D1–D4 report the analysis results under $(z_1, z_2) = (1,0), (1,1), (0,0), (0,1),$ respectively.

Table D1. Parameter estimation and testing statistics including the composition factor for XGboost trained on Bone Marrow data.

Mean AUC analysis					Log SD analysis				
Coef	Est	SE	*t*-value	*p*-value	Coef	Est	SE	*t*-value	*p*-value
x_1	0.7176	0.0159	45.260	<0.001	x_1	−2.3980	0.5197	−4.614	<0.001
x_2	0.7636	0.0159	48.160	<0.001	x_2	−2.8597	0.5197	−5.502	<0.001
x_3	0.7638	0.0158	48.216	<0.001	x_3	−3.8507	0.5193	−7.415	<0.001
x_1x_2	0.4265	0.0485	8.789	<0.001	x_1x_2	−7.2301	1.5908	−4.545	<0.001
x_1x_3	0.2730	0.0484	5.643	<0.001	x_1x_3	0.7582	1.5859	0.478	0.635
x_2x_3	0.2065	0.0484	4.268	<0.001	x_2x_3	1.6596	1.5859	1.046	0.300
x_1c_1	0.0265	0.0205	1.289	0.203	x_1c_1	−0.3232	0.6735	−0.480	0.633
x_2c_1	0.0098	0.0205	0.477	0.635	x_2c_1	−0.2823	0.6735	−0.419	0.677
x_3c_1	0.1161	0.0205	5.662	<0.001	x_3c_1	−1.5902	0.6723	−2.365	0.022
x_1c_2	0.0468	0.0205	2.277	0.027	x_1c_2	0.2895	0.6735	0.430	0.669
x_2c_2	0.0169	0.0205	0.823	0.415	x_2c_2	0.1848	0.6735	0.274	0.785
x_3c_2	0.2153	0.0205	10.499	<0.001	x_3c_2	−6.3190	0.6723	−9.399	<0.001

Table D2. Parameter estimation and testing statistics including the composition factor for XGboost trained on KEGG data.

Mean AUC analysis					Log SD analysis				
Coef	Est	SE	*t*-value	*p*-value	Coef	Est	SE	*t*-value	*p*-value
x_1	0.6506	0.0138	47.275	<0.001	x_1	−5.3454	0.9829	−5.438	<0.001
x_2	0.6818	0.0138	49.542	<0.001	x_2	−4.9938	0.9829	−5.081	<0.001
x_3	0.6810	0.0138	49.525	<0.001	x_3	−4.9602	0.9821	−5.051	<0.001
x_1x_2	0.5373	0.0421	12.755	<0.001	x_1x_2	−1.0774	3.0085	−0.358	0.722
x_1x_3	0.4964	0.0419	11.821	<0.001	x_1x_3	−1.745	2.9993	−0.582	0.563
x_2x_3	0.4992	0.0419	11.888	<0.001	x_2x_3	−8.9549	2.9993	−2.986	0.004
x_1c_1	0.0366	0.0178	2.051	0.045	x_1c_1	−1.0857	1.2737	−0.852	0.398
x_2c_1	0.0495	0.0178	2.777	0.008	x_2c_1	−0.4156	1.2737	−0.326	0.756
x_3c_1	0.0497	0.0178	2.793	0.007	x_3c_1	−0.5675	1.2715	−0.446	0.657
x_1c_2	0.0789	0.0178	4.424	<0.001	x_1c_2	−1.8836	1.2737	−1.479	0.145
x_2c_2	0.1124	0.0178	6.304	<0.001	x_2c_2	−1.3515	1.2737	−1.061	0.294
x_3c_2	0.1021	0.0178	5.735	<0.001	x_3c_2	−1.8684	1.2715	−1.469	0.148

Table D3. Parameter estimation and testing statistics including main and interaction effects of composites for CNN trained on Bone Marrow data.

Mean AUC analysis					Log SD analysis				
Coef	Est	SE	*t*-value	*p*-value	Coef	Est	SE	*t*-value	*p*-value
x_1	0.4546	0.0189	24.089	<0.001	x_1	−10.4340	0.8962	−11.643	<0.001
x_2	0.4953	0.0189	26.246	<0.001	x_2	−3.7947	0.8962	−4.234	<0.001
x_3	0.7238	0.0189	38.388	<0.001	x_3	−3.6605	0.8954	−4.088	<0.001
x_1x_2	0.5954	0.0578	10.309	<0.001	x_1x_2	4.5022	2.7430	1.641	0.107
x_1x_3	0.9035	0.0576	15.690	<0.001	x_1x_3	11.7004	2.7446	4.279	<0.001
x_2x_3	0.8316	0.0576	14.615	<0.001	x_2x_3	4.5889	2.7446	1.678	0.099
x_1c_1	−0.0399	0.0245	−1.631	0.109	x_1c_1	3.5757	1.1613	3.079	0.003
x_2c_1	0.0073	0.0245	0.299	0.766	x_2c_1	−0.5495	1.1613	−0.473	0.638
x_3c_1	0.1142	0.0244	4.680	<0.001	x_3c_1	−1.9253	1.1593	−1.661	0.103
x_1c_2	−0.0787	0.0245	−3.217	0.002	x_1c_2	6.0012	1.1613	5.168	<0.001
x_2c_2	0.0219	0.0245	0.895	0.375	x_2c_2	−1.8060	1.1613	−1.555	0.126
x_3c_2	0.2416	0.0244	9.895	<0.001	x_3c_2	−7.3973	1.1593	−6.381	<0.001

Table D4. Parameter estimation and testing statistics including main and interaction effects of composites for CNN trained on KEGG data.

Mean AUC analysis					Log SD analysis				
Coef	Est	SE	*t*-value	*p*-value	Coef	Est	SE	*t*-value	*p*-value
x_1	0.4607	0.0110	41.809	<0.001	x_1	−9.9670	1.0211	−9.761	<0.001
x_2	0.5517	0.0110	50.072	<0.001	x_2	−3.7920	1.0211	−3.714	<0.001
x_3	0.6392	0.0110	58.060	<0.001	x_3	−4.8357	1.0211	−4.740	<0.001
x_1x_2	0.7786	0.0337	23.087	<0.001	x_1x_2	3.9915	3.1253	1.277	0.207
x_1x_3	0.9419	0.0336	28.016	<0.001	x_1x_3	−0.7786	3.1157	−0.250	0.804
x_2x_3	0.5623	0.0336	16.725	<0.001	x_2x_3	−9.6674	3.1157	−3.103	0.003
x_1c_1	−0.0324	0.0143	−2.270	0.027	x_1c_1	3.1085	1.3231	2.349	0.023
x_2c_1	0.0769	0.0143	5.389	<0.001	x_2c_1	−0.9225	1.3231	−0.697	0.489
x_3c_1	0.0491	0.0143	3.447	0.001	x_3c_1	−1.3457	1.3208	−1.019	0.313
x_1c_2	−0.0609	0.0143	−4.262	<0.001	x_1c_2	4.8479	1.3231	3.664	<0.001
x_2c_2	0.1264	0.0143	8.856	<0.001	x_2c_2	−1.4988	1.3231	−1.133	0.263
x_3c_2	0.0946	0.0143	6.634	<0.001	x_3c_2	−1.3742	1.3208	−1.040	0.303