Journal of the American Statistical Association

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/uasa20

Hierarchical Neyman-Pearson Classification for Prioritizing Severe Disease Categories in COVID-19 Patient Data

Lijia Wang, Y. X. Rachel Wang, Jingyi Jessica Li & Xin Tong

To cite this article: Lijia Wang, Y. X. Rachel Wang, Jingyi Jessica Li & Xin Tong (17 Oct 2023): Hierarchical Neyman-Pearson Classification for Prioritizing Severe Disease Categories in COVID-19 Patient Data, Journal of the American Statistical Association, DOI: 10.1080/01621459.2023.2270657

To link to this article: https://doi.org/10.1080/01621459.2023.2270657

Hierarchical Neyman-Pearson Classification for Prioritizing Severe Disease Categories in COVID-19 Patient Data

Lijia Wang[a,*], Y. X. Rachel Wang[b,*,†], Jingyi Jessica Li[c], Xin Tong[d]

[a]School of Data Science, City University of Hong Kong

[b]School of Mathematics and Statistics, University of Sydney

[c]Department of Statistics, University of California, Los Angeles

[d]Department of Data Sciences and Operations, University of Southern California

*Equal contribution

†Correspondence should be addressed to Y.X. Rachel Wang (rachel.wang@sydney.edu.au)

Abstract

COVID-19 has a spectrum of disease severity, ranging from asymptomatic to requiring hospitalization. Understanding the mechanisms driving disease severity is crucial for developing effective treatments and reducing mortality rates. One way to gain such understanding is using a multi-class classification framework, in which patients' biological features are used to predict patients' severity classes. In this severity classification problem, it is beneficial to prioritize the identification of more severe classes and control the "under-classification" errors, in which patients are misclassified into less severe categories. The Neyman-Pearson (NP) classification paradigm has been developed to prioritize the designated type of error. However, current NP procedures are either for binary classification or do not provide high probability controls on the prioritized errors in multi-class classification. Here, we propose a hierarchical NP (H-NP) framework and an umbrella algorithm that generally adapts to popular classification methods and controls the under-classification errors with high probability. On an integrated collection of single-cell RNA-seq (scRNA-seq) datasets for 864 patients, we explore ways of featurization and demonstrate the efficacy of the H-NP algorithm in controlling the under-classification errors regardless of featurization. Beyond COVID-19 severity classification, the H-NP algorithm generally applies to multi-

class classification problems, where classes have a priority order.

1 Introduction

The COVID-19 pandemic has infected over 767 million people and caused 6.94 million deaths (27 June 2023) (World Health Organization, 2023), prompting collective efforts from statistics and other communities to address data-driven challenges. Many statistical works have modeled epidemic dynamics (Betensky and Feng, 2020; Quick et al., 2021), forecasted the case growth rates and outbreak locations (Brooks et al., 2020; Tang et al., 2021; McDonald et al., 2021), and analyzed and predicted the mortality rates (James et al., 2021; Kramlinger et al., 2022). Classification problems, such as diagnosis (positive/negative) (Wu et al., 2020; Li et al., 2020; Zhang et al., 2021) and severity prediction (Yan et al., 2020; Sun et al., 2020; Zhao et al., 2020; Ortiz et al., 2022), have been tackled by machine learning approaches (e.g., logistic regression, support vector machine (SVM), random forest, boosting, and neural networks; see Alballa and Al-Turaiki (2021) for a review).

In the existing COVID-19 classification works, the commonly used data types are CT images, routine blood tests, and other clinical data including age, blood pressure and medical history (Meraihi et al., 2022). In comparison, multiomics data are harder to acquire but can provide better insights into the molecular features driving patient responses (Overmyer et al., 2021). Recently, the increasing availability of single-cell RNA-seq (scRNA-seq) data offers the opportunity to understand transcriptional responses to COVID-19 severity at the cellular level (Wilk et al., 2020; Stephenson et al., 2021; Ren et al., 2021).

More generally, genome-wide gene expression measurements have been routinely used in classification settings to characterize and distinguish disease subtypes, both in bulk-sample (Aibar et al., 2015) and, more recently, single-cell level (Arvaniti and Claassen, 2017; Hu et al., 2019). While such genome-wide data can be costly, they provide a comprehensive view of the transcriptome and can unveil significant gene expression patterns for diseases with complex pathophysiology, where multiple genes and pathways are involved. Furthermore, as the patient-level measurements

continue to grow in dimension and complexity (e.g., from a single bulk sample to thousands-to-millions of cells per patient), a supervised learning setting enables us to better establish the connection between patient-level features and their associated disease states, paving the way towards personalized treatment.

In this study, we focus on patient severity classification using an integrated collection of multi-patient scRNA-seq datasets. Based on the WHO guidelines (World Health Organization, 2020), COVID-19 patients have at least three severity categories: healthy, mild/moderate, and severe. The classical classification paradigm aims at minimizing the overall classification error. However, prioritizing the identification of more severe patients may provide important insights into the biological mechanisms underlying disease progression and severity, and facilitate the discovery of potential biomarkers for clinical diagnosis and therapeutic intervention. Consequently, it is important to prioritize the control of "under-classification" errors, in which patients are misclassified into less severe categories.

Motivated by the gap in existing classification algorithms for severity classification (Section 1.1), we propose a hierarchical Neyman-Pearson (H-NP) classification framework that prioritizes the under-classification error control in the following sense. Suppose there are $\mathcal{I}$ classes with class labels $[\mathcal{I}] = \{1, 2, \ldots, \mathcal{I}\}$ ordered in decreasing severity. For $i \in [\mathcal{I} - 1]$, the *i*-th under-classification error is the probability of misclassifying an individual in class *i* into any class *j* with *j* > *i*. We develop an H-NP umbrella algorithm that controls the *i*-th under-classification error below a user-specified level $\alpha_i \in (0,1)$ with high probability while minimizing a weighted sum of the remaining classification errors. Similar in spirit to the NP umbrella algorithm for binary classification in Tong et al. (2018), the H-NP umbrella algorithm adapts to popular scoring-type multi-class classification methods (e.g., logistic regression, random forest, and SVM). To our knowledge, the algorithm is the first to achieve asymmetric error control with high probability in multi-class classification.

Another contribution of this study is the exploration of appropriate ways to featurize multi-patient scRNA-seq data. Following the workflow in Lin et al. (2022), we integrate 20 publicly available scRNA-seq datasets to form a sample of 864 patients with three levels of severity. For each patient, scRNA-seq data were collected from

peripheral blood mononuclear cells (PBMCs) and processed into a sparse expression matrix, which consists of tens of thousands of genes in rows and thousands of cells in columns. We propose four ways of extracting a feature vector from each of these 864 matrices. Then we evaluate the performance of each featurization way in combination with multiple classification methods under both the classical and H-NP classification paradigms. We note that our H-NP umbrella algorithm is applicable to other featurizations of scRNA-seq data, other forms of patient data, and more general disease classification problems with a severity ordering.

Below we review the NP paradigm and featurization of multi-patient scRNA-seq data as the background of our work.

1.1 Neyman-Pearson paradigm and multi-class classification

Classical binary classification focuses on minimizing the overall classification error, i.e., a weighted sum of type I and II errors, where the weights are the marginal probabilities of the two classes. However, the class priorities are not reflected by the class weights in many applications, especially disease severity classification, where the severe class is the minor class and has a smaller weight (e.g., HIV (Meyer and Pauker, 1987) and cancer (Dettling and Bühlmann, 2003)). One class of methods that addresses this error asymmetry is cost-sensitive learning (Elkan, 2001; Margineantu, 2002), which assigns different costs to type I and type II errors. However, such weights may not be easy to choose in practice, especially in a multi-class setting; nor do these methods provide high probability controls on the prioritized errors. The NP classification paradigm (Cannon et al., 2002; Scott and Nowak, 2005; Rigollet and Tong, 2011) was developed as an alternative framework to enforce class priorities: it finds a classifier that controls the population type I error (the prioritized error, e.g., misclassifying diseased patients as healthy) under a user-specified level α while minimizing the type II error (the error with less priority, e.g., misdiagnosing healthy people as sick). Practically, using an order statistics approach, Tong et al. (2018) proposed an NP umbrella algorithm that adapts all scoring-type classification methods (e.g., logistic regression) to the NP paradigm for classifier construction. The resulting classifier has the population type I error under α

with high probability. Besides disease severity classification, the NP classification paradigm has found diverse applications, including social media text classification (Xia et al., 2021) and crisis risk control (Feng et al., 2021). Nevertheless, the original NP paradigm is for binary classification only.

Although several works aimed to control prioritized errors in multi-class classification (Landgrebe and Duin, 2005; Xiong et al., 2006; Tian and Feng, 2021), they did not provide high probability control. That is, if they are applied to severe disease classification, there is a non-trivial chance that their under-classification errors exceed the desired levels.

1.2 ScRNA-seq data featurization

In multi-patient scRNA-seq data, every patient has a gene-by-cell expression matrix; genes are matched across patients, but cells are not. For learning tasks with patients as instances, featurization is a necessary step to ensure that all patients have feautures in the same space. A common featurization approach is to assign every patient's cells into cell types, which are comparable across patients, by clustering (Stanley et al., 2020; Ganio et al., 2020) and/or manual annotation (Han et al., 2019). Then, each patient's gene-by-cell expression matrix can be converted into a gene-by-cell-type expression matrix using a summary statistic (e.g., every gene's mean expression in a cell type), so all patients have gene-by-cell-type expression matrices with the same dimensions. We note here that most of the previous multi-patient single-cell studies with a reasonably large cohort used CyTOF data (Davis et al., 2017), which typically measures 50–100 protein markers, whereas scRNA-seq data have a much higher feature dimension, containing expression values of $\sim 10^4$ genes. Thus further featurization is necessary to convert each patient's gene-by-cell-type expression matrix into a feature vector for classification.

Following the data processing workflow in Lin et al. (2022), we obtain 864 patients' cell-type-by-gene expression matrices, which include 18 cell types and 3,000 genes (after filtering). We propose and compare four ways of featurizing these matrices into vectors, which differ in their treatments of 0 values and approaches to dimension reduction. Note that we perform featurization as a separate step before classification

so that all classification methods are applicable. Separating the featurization step also allows us to investigate whether a featurization way maintains robust performance across classification methods.

The rest of the paper is organized as follows. In Section 2, we introduce the H-NP classification framework and propose an umbrella algorithm to control the under-classification errors with high probability. Next, we conduct extensive simulation studies to evaluate the performance of the umbrella algorithm. In Section 3, we describe four ways of featurizing the COVID-19 multi-patient scRNA-seq data and show that the H-NP umbrella algorithm consistently controls the under-classification errors in COVID-19 severity classification across all featurization ways and classification methods. Furthermore, we demonstrate that utilizing the scRNA-seq data allows us to gain biological insights into the mechanism and immune response of severe patients at both the cell-type and gene levels. Supplementary Materials contain technical derivations, proofs and additional numerical results.

2 Hierarchical Neyman-Pearson (H-NP) classification

2.1 Under-classification errors in H-NP classification

We first introduce the formulation of H-NP classification and define the under-classification errors, which are the probabilities of individuals being misclassified to less severe (more generally, less important) classes. In an H-NP problem with $\mathcal{I} \geq 2$ classes, the class labels $i \in [\mathcal{I}] := \{1, 2, \ldots, \mathcal{I}\}$ are ranked in a decreasing order of importance, i.e., class i is more important than class j if $i < j$. Let (X, Y) be a random pair, where $X \in \mathcal{X} \subset \mathrm{R}^d$ represents a vector of features, and $Y \in [\mathcal{I}]$ denotes the class label. A classifier $\phi : \mathcal{X} \to [\mathcal{I}]$ maps a feature vector X to a predicted class label. In the following discussion, we abbreviate $\mathrm{P}(\cdot | Y = i)$ as $P_i(\cdot)$. Our H-NP framework aims to control the under-classification errors at the population level in the sense that

$$R_{i^\star}(\phi) = P_i(\phi(X) \in \{i+1, \ldots, \mathcal{I}\}) \leq \alpha_i \quad \text{for} \quad i \in [\mathcal{I} - 1], \qquad (1)$$

where $\alpha_i \in (0,1)$ is the desired control level for the i-th under-classification error $R_{i\star}(\phi)$. Simultaneously, our H-NP framework minimizes the weighted sum of the remaining errors, which can be expressed as

$$R^c(\phi) = \mathrm{P}(\phi(X) \neq Y) - \sum_{i=1}^{\mathcal{I}-1} \pi_i R_{i\star}(\phi), \quad \text{where} \quad \pi_i = \mathrm{P}(Y = i). \qquad (2)$$

We note that when $\mathcal{I} = 2$, this H-NP formulation is equivalent to the binary NP classification (prioritizing class 1 over class 2), with $R_{1\star}(\phi)$ being the population type I error.

For COVID-19 severity classification with three levels, severe patients labeled as Y = 1 have the top priority, and we want to control the probability of severe patients not being identified, which is $R_{1\star}(\phi)$. The secondary priority is for moderate patients labeled as Y = 2; $R_{2\star}(\phi)$ is the probability of moderate patients being classified as healthy. Healthy patients that do not need medical care are labeled as Y = 3. Note that $R_{i\star}(\cdot)$ and $R^c(\cdot)$ are population-level quantities as they depend on the intrinsic distribution of (X, Y), and it is hard to control the $R_{i\star}(\cdot)$'s almost surely due to the randomness of the classifier.

2.2 H-NP algorithm with high probability control

In this section, we construct an H-NP umbrella algorithm that controls the population under-classification errors in the sense that $\mathrm{P}(R_{i\star}(\hat{\phi}) > \alpha_i) \leq \delta_i$ for $i \in [\mathcal{I} - 1]$, where $(\delta_1, \ldots, \delta_{\mathcal{I}-1})$ is a vector of tolerance parameters, and $\hat{\phi}$ is a scoring-type classifier to be defined below.

Roughly speaking, we employ a sample-splitting strategy, which uses some data subsets to train the scoring functions from a base classification method and other data subsets to select appropriate thresholds on the scores to achieve population-level error controls. Here, the scoring functions refer to the scores assigned to each possible class label for a given input observation and include examples such as the output from the softmax transformation in multinomial logistic regression. For $i \in [\mathcal{I}]$, let $\mathcal{S}_i = \{X_j^i\}_{j=1}^{N_i}$ denote N_i independent observations from class i, where N_i is the size

of the class. In the following discussion, the superscript on X is dropped for brevity when it is clear which class the observation comes from. Our procedure randomly splits the class-i observations into (up to) three parts: $\mathcal{S}_{is}$ ($i \in [\mathcal{I}]$) for obtaining scoring functions, $\mathcal{S}_{it}$ ($i \in [\mathcal{I}-1]$) for selecting thresholds, and $\mathcal{S}_{ie}$ ($i = 2, \ldots, \mathcal{I}$) for computing empirical errors. As will be made clear later, our procedure does not require $\mathcal{S}_{1e}$ or $\mathcal{S}_{\mathcal{I}t}$ and splits class 1 and class $\mathcal{I}$ into two parts only. After splitting, we use the combination $\mathcal{S}_s = \bigcup_{i \in [\mathcal{I}]} \mathcal{S}_{is}$ to train the scoring functions.

We consider a classifier that relies on $\mathcal{I}-1$ scoring functions $T_1, T_2, \ldots, T_{\mathcal{I}-1} : \mathcal{X} \to \mathrm{R}$, where the class decision is made sequentially with each $T_i(X)$ determining whether the observation belongs to class i or one of the less prioritized classes $(i+1), \ldots, \mathcal{I}$. Thus at each step i, the decision is binary, allowing us to use the NP Lemma to motivate the construction of our scoring functions. Note that $\mathrm{P}(Y = i \mid X = x) / \mathrm{P}(Y \in \{i+1, \ldots, \mathcal{I}\} \mid X = x) \propto f_i(x) / f_{>i}(x)$, where $f_{>i}(x)$ and $f_i(x)$ represent the density function of X when $Y > i$ and $Y = i$, respectively, and the density ratio is the statistic that leads to the most powerful test with a given level of control on one of the errors by the NP Lemma. Given a typical scoring-type classification method (e.g., logistic regression, random forest, SVM, and neural network) that provides the probability estimates $\mathrm{P}(Y = i \mid X)$ for $i \in [\mathcal{I}]$, we can construct our scores using these estimates by defining

$$T_1(X) = \mathrm{P}(Y = 1 \mid X), \quad \text{and} \quad T_i(X) = \frac{\mathrm{P}(Y = i \mid X)}{\sum_{j=i+1}^{\mathcal{I}} \mathrm{P}(Y = j \mid X)} \quad \text{for} \quad 1 < i < \mathcal{I} - 1.$$

Given thresholds $(t_1, t_2, \ldots, t_{\mathcal{I}-1})$, we consider an H-NP classifier of the form

$$\hat{\phi}(X) = \begin{cases} 1, & T_1(X) \geq t_1; \\ 2, & T_2(X) \geq t_2 \quad \text{and} \quad T_1(X) < t_1; \\ \cdots & \\ \mathcal{I}-1, & T_{\mathcal{I}-1}(X) \geq t_{\mathcal{I}-1} \quad \text{and} \quad T_1(X) < t_1, \ldots, T_{\mathcal{I}-2}(X) < t_{\mathcal{I}-2}; \\ \mathcal{I}, & \text{otherwise.} \end{cases} \tag{3}$$

Then the i-th under-classification error for this classifier can be written as

$$R_{i^*}(\hat{\phi}) = P_i\left(\hat{\phi}(X) \in \{i+1, \ldots, \mathcal{I}\}\right) = P_i\left(T_1(X) < t_1, \ldots, T_i(X) < t_i\right), \quad (4)$$

where X is a new observation from the i-th class independent of the data used for score training and threshold selection. The thresholds $(t_1, t_2, \ldots, t_{\mathcal{I}-1})$ are selected using the observations in $\mathcal{S}_{1t}, \ldots, \mathcal{S}_{(\mathcal{I}-1)t}$, and they are chosen to satisfy $\mathrm{P}(R_{i^*}(\hat{\phi}) > \alpha_i) \leq \delta_i$ for all $i \in [\mathcal{I}-1]$. In what follows, we will develop our arguments conditional on the data $\mathcal{S}_s$ for training the scoring functions so that T_is can be viewed as fixed functions.

According to Eq (3), the first under-classification error $R_{1^*}(\hat{\phi}) = P_1\left(T_1(X) < t_1\right)$ only depends on t_1, while the other under-classification errors $R_{i^*}(\hat{\phi})$ depend on $t_1, \ldots, t_i$. To achieve the high probability controls with $\mathrm{P}(R_{i^*}(\hat{\phi}) > \alpha_i) \leq \delta_i$ for all $i \in [\mathcal{I}-1]$, we select $t_1, \ldots t_{\mathcal{I}-1}$ sequentially using an order statistics approach. We start with the selection of t_1, which is covered by the following general proposition. The proof is a modification of Proposition 1 in Tong et al. (2018) and can be found in Supplementary Section B.1.

Proposition 1. *For any $i \in [\mathcal{I}]$, denote $\mathcal{T}_i = \{T_i(X) \mid X \in \mathcal{S}_{it}\}$, and let $t_{i(k)}$ be the corresponding k-th order statistic. Further denote the cardinality of $\mathcal{T}_i$ as n_i. Assuming that the data used to train the scoring functions and the left-out data are independent, then given a control level α, for another independent observation X from class i,*

$$\mathrm{P}\left(P_i\left[T_i(X) < t_{i(k)} \mid t_{i(k)}\right] > \alpha\right) \leq v(k, n_i, \alpha) := \sum_{j=0}^{k-1} \binom{n_i}{j} (\alpha)^j (1-\alpha)^{n_i - j}. \quad (5)$$

We remark that similar to Proposition 1 in Tong et al. (2018), if T_i is a continuous random variable, the bound in Eq (5) is tight.

Algorithm 1: DeltaSearch(n, α, δ)

Input : size: n; level: α; tolerance: δ.

1 $k = 0$, $v_k = 0$

2 while $v_k \leq \delta$ do

3 $v_k = v_k + \binom{n}{k} (\alpha)^k (1-\alpha)^{n-k}$

4 $k = k + 1$

5 end

Output: k

Let $k_i = \max\{k \mid v(k, n_i, \alpha_i) \leq \delta_i\}$, which can be computed using Algorithm 1. Then Proposition 1 and Eq (4) imply

$$\mathrm{P}\left(R_{i^*}(\hat{\phi}) > \alpha_i\right) \leq \mathrm{P}\left(P_i\left[T_i(X) < t_i \mid t_{i(k_i)}\right] > \alpha_i\right) \leq \delta_i \quad \text{for all} \quad t_i \leq t_{i(k_i)}. \qquad (6)$$

We note that to have a solution for $v(k, n_i, \alpha_i) \leq \delta_i$ among $k \in [n_i]$, we need $n_i \geq \log \delta_i / \log(1-\alpha_i)$, the minimum sample size required for the class $\mathcal{S}_{it}$. When $i = 1$, the first inequality in Eq (6) becomes equality, so $t_{1(k_1)}$ is an effective upper bound on t_1 when we later minimize the empirical counterpart of $R^c(\cdot)$ in Eq (2) with respect to different feasible threshold choices. On the other hand, for $i > 1$, the inequality is mostly strict, which means that the bound $t_{i(k_i)}$ on t_i is expected to be loose and can be improved. To this end, we note that Eq (4) can be decomposed as

$$R_{i^*}(\hat{\phi}) = P_i\left(T_i(X) < t_i \middle| T_1(X) < t_1, \ldots, T_{i-1}(X) < t_{i-1}\right) \cdot P_i\left(T_1(X) < t_1, \ldots, T_{i-1}(X) < t_{i-1}\right) \qquad (7)$$

leading to the following theorem that upper bounds t_i given the previous thresholds.

Theorem 1. *Given the previous thresholds $t_1, \ldots, t_{i-1}$, consider all the scores T_i on the left-out class $\mathcal{S}_{it}$, $\mathcal{T}_i = \{T_i(X) \mid X \in \mathcal{S}_{it}\}$, and a subset of these scores depending on the previous thresholds, defined as $\mathcal{T}'_i = \{T_i(X) \mid X \in \mathcal{S}_{it}, T_1(X) < t_1, \ldots, T_{i-1}(X) < t_{i-1}\}$. We*

use $t_{i(k)}$ *and* $t'_{i(k)}$ *to denote the k-th order statistic of* $\mathcal{T}_i$ *and* $\mathcal{T}'_i$*, respectively. Let* n_i *and* n_i' *be the cardinality of* $\mathcal{T}_i$ *and* $\mathcal{T}'_i$*, respectively, and* α_i *and* δ_i *be the prespecified control level and violation tolerance for the i-th under-classification error* $R_{i\star}(\cdot)$*. We set*

$$\hat{p}_i = \frac{n_i'}{n_i}, p_i = \hat{p}_i + c(n_i), \alpha_i' = \frac{\alpha_i}{p_i}, \delta_i' = \delta_i - \exp\{-2n_i c^2(n_i)\}, \quad (8)$$

where $c(n) = \mathcal{O}(1/\sqrt{n})$*. Let*

$$\bar{t}_i = \begin{cases} t'_{i(k'_i)}, & \text{if } n_i' \geq \log \delta_i' / \log(1-\alpha_i') \quad \text{and} \quad \alpha_i' < 1; \\ t_{i(k_i)}, & \text{otherwise,} \end{cases} \quad (9)$$

where $k_i = \max\{k \in [n_i] \mid v(k, n_i, \alpha_i) \leq \delta_i\}$ and $k'_i = \max\{k \in [n_i'] \mid v(k, n'_i, \alpha'_i) \leq \delta_i'\}$. *Then,*

$$\mathrm{P}(R_{i\star}(\hat{\phi}) > \alpha_i) = \mathrm{P}\left(P_i\left[T_1(X) < t_1, \ldots T_i(X) < t_i \mid \bar{t}_i\right] > \alpha_i\right) \leq \delta_i \quad \text{for all} \quad t_i \leq \bar{t}_i. \quad (10)$$

In other words, if the cardinality of $\mathcal{T}'_i$ exceeds a threshold, we can refine the choice of the upper bound according to Eq (9); otherwise, the bound in Proposition 1 always applies. The proof of the theorem is provided in Supplementary Section B.2; the computation of the upper bound $\bar{t}_i$ is summarized in Algorithm 2. $\bar{t}_i$ guarantees the required high probability control on the *i*-th under-classification error, while providing a tighter bound compared with Eq (4). We make two additional remarks as follows.

Remark 1.

a) *The minimum sample size requirement for* $\mathcal{S}_{it}$ *is still* $n_i \geq \log \delta_i / \log(1-\alpha_i)$ *because* $t_{i(k_i)}$ *in Eq (9) always exists when this inequality holds. For instance, if* $\alpha_i = 0.05$ *and* $\delta_i = 0.05$*, then* $n_i \geq 59$*.*
b) *The choice of c(n) involves a trade-off between* α_i' *and* δ_i'*, although under the constraint* $c(n) = \mathcal{O}(1/\sqrt{n})$*, any changes in both quantities are small in magnitude for large n. For example, a larger c(n) leads to a smaller* α_i' *and a larger* δ_i'*, thus a looser tolerance level comes at the cost of a stricter error*

control level. In practice, larger α_i' *and larger* δ_i' *values are desired since they lead to a wider region for* t_i*. We set* $c(n) = 2/\sqrt{n}$ *throughout the rest of the paper. Then by Eq (8),* α_i' *increases as n increases, and* $\delta_i' = \delta_i - e^{-4}$*, so the difference between* δ_i' *and the prespecified* δ_i *is sufficiently small.*

c) *Eq (10) has two cases, as Eq (9) indicates. When* $\bar{t}_i = t_{i(k_i)}$*, the bound remains the same as Eq (6), which is not tight for i > 1. When* $\bar{t}_i = t'_{i(k'_i)}$*, Eq (10) provides a tighter bound through the decomposition in Eq (7), where the first part is bounded by a concentration argument, and the second part achieves a tight bound the same way as Proposition 1.*

With the set of upper bounds on the thresholds chosen according to Theorem 1, the next step is to find an optimal set of thresholds $(t_1, t_2, \ldots, t_{\mathcal{I}-1})$ satisfying these upper bounds while minimizing the empirical version of $R^c(\hat{\phi})$, which is calculated using observations in $\mathcal{S}_e = \bigcup_{i=2}^{\mathcal{I}} \mathcal{S}_{ie}$ (since class-1 observations are not needed in $R^c(\hat{\phi})$). For brevity, we denote all the empirical errors as $\tilde{R}$, e.g., $\tilde{R}^c$. In Section 2.4, we will show numerically that Theorem 1 provides a wider search region for the threshold t_i compared to Proposition 1, which benefits the minimization of R^c.

As our COVID-19 data has three severity levels, in the next section, we will focus on the three-class H-NP umbrella algorithm and describe in more details how the above procedures can be combined to select the optimal thresholds in the final classifier.

Algorithm 2: UpperBound $(\mathcal{S}_{it}, \alpha_i, \delta_i, (T_1, \ldots, T_i), (t_1, \ldots, t_{i-1}))$

Input : The left-out class-*i* samples: $\mathcal{S}_{it}$; level: α_i; tolerance: δ_i; score functions: $(T_1, \ldots, T_i)$; thresholds: $(t_1, \ldots, t_{i-1})$.

1 $n_i \leftarrow |\mathcal{S}_{it}|$

2 $\{t_{i(1)}, \ldots, t_{i(n_i)}\} \leftarrow$ sort $\mathcal{T}_i = \{T_i(X) \mid X \in \mathcal{S}_{it}\}$

3 $k_i \leftarrow \mathrm{DeltaSearch}(n_i, \alpha_i, \delta_i)$; // i.e., Algorithm 1

4 $\bar{t}_i \leftarrow t_{i(k_i)}$

5 if *i > 1* then

6 $\mathcal{T}'_i \leftarrow \{t'_{i(1)}, \ldots, t'_{i(n_i')}\} = \mathrm{sort}\{T_i(X) \mid X \in \mathcal{S}_{it}, T_1(X) < t_1, \ldots, T_{i-1}(X) < t_{i-1}\}$; // Note that n_i' is random

7 $\hat{p}_i \leftarrow \frac{n_i'}{n_i}$, $p_i \leftarrow \hat{p}_i + c(n_i)$, $\alpha_i' \leftarrow \alpha_i / p_i$, $\delta_i' \leftarrow \delta_i - e^{-2n_i c^2(n_i)}$; // e.g., $c(n) = \frac{2}{\sqrt{n}}$

8 if $n_i' \geq \log \delta_i' / \log(1 - \alpha_i')$ and $\alpha_i' < 1$ then

9 $k'_i \leftarrow \mathrm{DeltaSearch}(n'_i, \alpha'_i, \delta'_i)$

10 $\bar{t}_i \leftarrow t'_{i(k'_i)}$

11 end

12 end

Output: $\bar{t}_i$

2.3 H-NP umbrella algorithm for three classes

Since our COVID-19 data groups patients into three severity categories, we introduce our H-NP umbrella algorithm for $\mathcal{I} = 3$. In this case, there are two under-classification errors $R_{1\star}(\phi) = P_1(\phi(X) \in \{2,3\})$ and $R_{2\star}(\phi) = P_2(\phi(X) = 3)$, which need to be controlled at prespecified levels α_1, α_2 with tolerance levels δ_1, δ_2, respectively. In addition, we wish to minimize the weighted sum of errors

$$R^c(\phi) = \mathrm{P}(\phi(X) \neq Y) - \pi_1 R_{1\star}(\phi) - \pi_2 R_{2\star}(\phi)$$
$$= \pi_2 P_2(\phi(X) = 1) + \pi_3[P_3(\phi(X) = 1) + P_3(\phi(X) = 2)]. \quad (11)$$

When $\mathcal{I} = 3$, our H-NP umbrella algorithm relies on two scoring functions $T_1, T_2 : \mathcal{X} \to \mathrm{R}$, which can be constructed by Eq (3) using the estimates $\mathrm{P}(Y = i \mid X)$ from any scoring-type classification method:

$$T_1(X) = \mathrm{P}(Y = 1 \mid X) \quad \text{and} \quad T_2(X) = \frac{\mathrm{P}(Y = 2 \mid X)}{\mathrm{P}(Y = 3 \mid X)}. \quad (12)$$

The H-NP classifier then takes the form

$$\hat{\phi}(X) = \begin{cases} 1, & T_1(X) \geq t_1; \\ 2, & T_2(X) \geq t_2 \quad \text{and} \quad T_1(X) < t_1; \\ 3, & \text{otherwise.} \end{cases} \quad (13)$$

Here T_2 determines whether an observation belongs to class 2 or class 3, with a larger value indicating a higher probability for class 2. Applying Algorithm 2, we can find $\bar{t}_1$ such that any threshold $t_1 \leq \bar{t}_1$ will satisfy the high probability control on the first under-classification error, that is $\mathrm{P}(R_{1\star}(\hat{\phi}) > \alpha_1) = \mathrm{P}\left(P_1\left[T_1(X) < t_1 \mid \bar{t}_1\right] > \alpha_1\right) \leq \delta_1$. Recall that the computation of $\bar{t}_2$ (and consequently t_2) depends on the choice of t_1. Given a fixed t_1, the high probability control on the second under-classification errors is $\mathrm{P}(R_{2\star}(\hat{\phi}) > \alpha_2) = \mathrm{P}\left(P_2\left[T_1(X) < t_1, T_2(X) < t_2 \mid \bar{t}_2\right] > \alpha_2\right) \leq \delta_2$, where $\bar{t}_2$ is computed by Algorithm 2 so that any $t_2 \leq \bar{t}_2$ satisfies the constraint.

The interaction between t_1 and t_2 comes into play when minimizing the remaining errors in $R^c(\hat{\phi})$. First note that using Eq (11) and (13), the other types of errors in $R^c(\hat{\phi})$ are

$$\begin{aligned} &P_2\left(\hat{\phi}(X) = 1\right) = P_2\left(T_1(X) \geq t_1\right), P_3\left(\hat{\phi}(X) = 1\right) = P_3\left(T_1(X) \geq t_1\right), \\ &P_3\left(\hat{\phi}(X) = 2\right) = P_3\left(T_1(X) < t_1, T_2(X) \geq t_2\right). \end{aligned} \quad (14)$$

To simplify the notation, let $\hat{Y}$ denote $\hat{\phi}(X)$ in the following discussion. For a fixed t_1, decreasing t_2 leads to an increase in $P_3(\hat{Y} = 2)$ and has no effect on the other errors in (14), which means that $t_2 = \bar{t}_2$ minimizes $R^c(\hat{\phi})$. However, the selection of t_1 is not as straightforward as t_2. Figure 1(a) illustrates how the set

$\mathcal{T}'_2 = \{T_2(X) \mid X \in \mathcal{S}_{2t}, T_1(X) < t_1\}$ (as appeared in Theorem 1) is constructed for a given t_1, where the elements are ordered by their T_2 values. Clearly, more elements are removed from $\mathcal{T}'_2$ as t_1 decreases, leading to a smaller n_2'. Consider an element in the set $\mathcal{T}'_2$ which has rank k in the ordered list (colored yellow in Figure 1(a)). Then k, n_2', α_2', and consequently $v(k, n_2', \alpha_2')$, will all be affected by decreasing t_1, but the change is not monotonic as shown in Figure 1(b). Decreasing t_1 could remove elements (dashed circles in Figure 1(b)) either to the left side (case 1) or right side (case 2) of the yellow element, depending on the values of the scores T_1. In case 1, $v(k, n_2', \alpha_2')$ decreases, resulting in a larger $\bar{t}_2$ and a smaller $P_3(\hat{Y} = 2)$ error, whereas the reverse can happen in case 2. The details of how $v(k, n_2', \alpha_2')$ changes can be found in Supplementary Section B.3, with additional simulations in Supplementary Figure S13. In view of the above, minimizing the empirical error $\tilde{R}^c$ requires a grid search over t_1, for which we use the set $\mathcal{T}_1 = \{T_1(X) \mid X \in \mathcal{S}_{1t}\}$, and the overall algorithm for finding the optimal thresholds and the resulting classifier is described in Algorithm 3, which we name as the H-NP umbrella algorithm. The algorithm for the general case with $\mathcal{I} > 3$ can be found in Supplementary Section E.

Algorithm 3: H-NP umbrella algorithm for $\mathcal{I} = 3$

Input : Sample: $\mathcal{S} = \mathcal{S}_1 \cup \mathcal{S}_2 \cup \mathcal{S}_3$; levels: (α_1, α_2); tolerances: (δ_1, δ_2); grid set: A_1 (e.g., $\mathcal{T}_1$).

$\hat{\pi}_2 = |\mathcal{S}_2| / |\mathcal{S}|$; $\hat{\pi}_3 = |\mathcal{S}_3| / |\mathcal{S}|$

$\mathcal{S}_{1s}, \mathcal{S}_{1t}, \leftarrow$ Random split $\mathcal{S}_1$; $\mathcal{S}_{2s}, \mathcal{S}_{2t}, \mathcal{S}_{2e} \leftarrow$ Random split $\mathcal{S}_2$; $\mathcal{S}_{3s}, \mathcal{S}_{3e} \leftarrow$ Random split $\mathcal{S}_3$

$\mathcal{S}_s = \mathcal{S}_{1s} \cup \mathcal{S}_{2s} \cup \mathcal{S}_{3s}$

$T_1, T_2 \leftarrow$ A base classification method($\mathcal{S}_s$) ; // c.f. Eq (12)

$\bar{t}_1 \leftarrow$ UpperBound($\mathcal{S}_{1t}, \alpha_1, \delta_1, (T_1), \text{NULL}$) ; // i.e., Algorithm 2

$\tilde{R}^c = 1$

for $t_1 \in A_1 \cap (-\infty, \bar{t}_1]$ **do**

$t_2 \leftarrow$ UpperBound($\mathcal{S}_{2t}, \alpha_2, \delta_2, (T_1, T_2), (t_1)$)

$\hat{\phi} \leftarrow$ a classifier with respect to t_1, t_2

$e_{21} = \sum_{X \in \mathcal{S}_{2e}} 1\{\hat{\phi}(X) = 1\} / |\mathcal{S}_{2e}|, e_3 = \sum_{X \in \mathcal{S}_{3e}} 1\{\hat{\phi}(X) \in \{1,2\}\} / |\mathcal{S}_{3e}|$

10

11 $\tilde{R}^c_{\text{new}} = \hat{\pi}_2 e_{21} + \hat{\pi}_3 e_3$

12 if $\tilde{R}^c_{\text{new}} < \tilde{R}^c$ then

13 $\tilde{R}^c \leftarrow \tilde{R}^c_{\text{new}}, \hat{\phi}^* \leftarrow \hat{\phi}$

14 end

15 end

Output: $\hat{\phi}^*$

2.4 Simulation studies

We first examine the validity of our H-NP umbrella algorithm using simulated data from a setting denoted **T1.1**, where $\mathcal{I} = 3$, and the feature vectors in class *i* are generated as $(X^i)^\top \sim N(\mu_i, I)$, where $\mu_1 = (0,-1)^\top$, $\mu_2 = (-1,1)^\top$, $\mu_3 = (1,0)^\top$ and *I* is the 2 × 2 identity matrix. For each simulated dataset, we generate the feature vectors and labels with 500 observations in each of the three classes. The observations are randomly separated into parts for score training, threshold selection and computing empirical errors: $\mathcal{S}_1$ is split into 50%, 50% for $\mathcal{S}_{1s}, \mathcal{S}_{1t}$; $\mathcal{S}_2$ is split into 45%, 50% and 5% for $\mathcal{S}_{2s}, \mathcal{S}_{2t}$ and $\mathcal{S}_{2e}$; $\mathcal{S}_3$ is split into 95%, 5% for $\mathcal{S}_{3s}, \mathcal{S}_{3e}$, respectively. All the results in this section are based on $1,000$ repetitions from a given setting. We set $\alpha_1 = \alpha_2 = 0.05$ and $\delta_1 = \delta_2 = 0.05$. To approximate and evaluate the true population errors $R_{1^\star}, R_{2^\star}$ and R^c, we additionally generate $20,000$ observations for each class and refer to them as the test set.

First, we demonstrate that Algorithm 3 outputs an H-NP classifier with the desired high probability controls. More specifically, we show that any $t_1 \le \bar{t}_1$ and $t_2 = \bar{t}_2$ ($\bar{t}_1, \bar{t}_2$ are computed by Algorithm 2) will lead to a valid threshold pair (t_1, t_2) satisfying $\mathrm{P}(R_{1^\star}(\hat{\phi}) > \alpha_1) \le \delta_1$ and $\mathrm{P}(R_{2^\star}(\hat{\phi}) > \alpha_2) \le \delta_2$, where $R_{1^\star}$ and $R_{2^\star}$ are approximated using the test set in each round of simulation. Here, we use multinomial logistic regression to construct the scoring functions T_1 and T_2, the inputs of Algorithm 3. Figure 2 displays the boxplots of various approximate errors with t_1 chosen as the *k*-

th largest element in $\mathcal{T}_1 \cap (-\infty, \bar{t}_1]$ as k changes. In Figure 2(a) and 2(b), where the blue diamonds mark the 95% quantiles, we can see that the violation rate of the required error bounds (red dashed lines, representing α_1 and α_2) is about 5% or less, suggesting our procedure provides effective controls on the errors of concerns. In this case, in most simulation rounds, $\bar{t}_1$ minimizes the empirical error $\tilde{R}^c$ computed on $\mathcal{S}_{2e}$ and $\mathcal{S}_{3e}$, and $t_1 = \bar{t}_1$ is chosen as the optimal threshold by Algorithm 3 in the final classifier. We can see this coincide with Figure 2(c), which shows that the largest element in $\mathcal{T}_1 \cap (-\infty, \bar{t}_1]$ (i.e., $\bar{t}_1$) minimizes the approximate error R^c on the test set. We note here that the results from other splitting ratios can be found in Supplementary Section C.2, where we observe that once the sample size for threshold selection reaches about twice the minimum sample size requirement, there are little observable differences in the results. In Supplementary Section C.3, we also compare with variations in computing the scoring functions to examine the effect of score normalization and calibration, showing that our current scoring functions are ideal for our purpose.

Next, we check whether indeed Theorem 1 gives a better upper bound on t_2 than Proposition 1 for overall error minimization. Recall the two upper bounds in Eq (6) ($t_{2(k_2)}$) and Eq (9) ($\bar{t}_2$). For each base classification algorithm (e.g., logistic regression), we set $t_1 = \bar{t}_1$ and t_2 equal to these two upper bounds respectively, resulting in two classifiers with different t_2 thresholds. We compare their performance by evaluating the approximate errors of $R_{2^*}(\hat{\phi})$ and $P_3(\hat{Y} = 2)$ since, as discussed in Section 2.3, the threshold t_2 only influences these two errors for a fixed t_1. Figure 3 shows the distributions of the errors and also their averages for three different base classification algorithms. Under each algorithm, both choices of t_2 effectively control $R_{2^*}(\hat{\phi})$, but the upper bound from Proposition 1 is overly conservative compared with that of Theorem 1, which results in a notable increase in $P_3(\hat{Y} = 2)$. This is undesirable since $P_3(\hat{Y} = 2)$ is one component in $R^c(\hat{\phi})$, and the goal is to minimize $R^c(\hat{\phi})$ under appropriate error controls.

Now we consider comparing our H-NP classifier against alternative approaches. We construct an example of “approximate” error control using the empirical ROC curve

approach. In this case, each class of observations is split into two parts: one for training the base classification method, the other for threshold selection using the ROC curve. Under the setting **T1.1**, using similar splitting ratios as before, we separate $\mathcal{S}_i$ into 50% and 50% for $\mathcal{S}_{is}$ and $\mathcal{S}_{it}$ for i = 1, 2, 3. The same test set is used. We re-compute the scoring functions (T_1 and T_2) corresponding to the new split. t_1 is selected using the ROC curve generated by T_1 aiming to distinguish between class 1 (samples in $\mathcal{S}_{1t}$) and class $2'$ (samples in $\mathcal{S}_{2t} \cup \mathcal{S}_{3t}$) merging classes 2 and 3, with specificity calculated as the rate of misclassifying a class-1 observation into class $2'$. Similarly, t_2 is selected using T_2 dividing samples in $\mathcal{S}_{2t} \cup \mathcal{S}_{3t}$ into class 2 and class 3, with specificity defined as the rate of misclassifying a class-2 observation into class 3. More specifically, in Eq (13) we use

$$t_1 = \sup\left\{ t : \frac{\sum_{X \in \mathcal{S}_{1t}} \mathbf{1}\{T_1(X) < t\}}{|\mathcal{S}_{1t}|} \le \alpha_1 \right\} \quad \text{and} \quad t_2 = \sup\left\{ t : \frac{\sum_{X \in \mathcal{S}_{2t}} \mathbf{1}\{T_2(X) < t\}}{|\mathcal{S}_{2t}|} \le \alpha_2 \right\}$$

to obtain the classifier for the ROC curve approach.

The comparison between our H-NP classifier and the ROC curve approach is summarized in Figure 4. Recalling α_i and δ_i are both 0.05, we mark the 95% quantiles of the under-classification errors by solid black lines and the target error control levels by dotted red lines. First we observe that the 95% quantiles of $R_{1\star}$ using the ROC curve approach well exceed the target level control, with their averages centering around the target. We also see the influence of t_1 on the $R_{2\star}$ – without suitably adjusting t_2 based on t_1, the control on $R_{2\star}(\hat{\phi})$ in the ROC curve approach is overly conservative despite it being an approximate error control method, which in turn leads to inflation in error $P_3(\hat{Y} = 2)$. In view of this, we further consider a simulation setting where the influence of t_1 on t_2 is smaller. The setting **T2.1** moves samples in class 1 further away from classes 2 and 3 by having $\mu_1 = (0, -3)^\top$, while the other parts remain the same as in the setting **T1.1**. α_i, δ_i are still 0.05. As shown in Figure 5, the ROC curve approach does not provide the required level of control for $R_{1\star}$ or $R_{2\star}$.

In Supplementary Sections C.4-C.6, we include more comparisons with alternative methods with different overall approaches to the problem, including weight-adjusted classification, cost-sensitive learning, and ordinal regression, and show that our H-NP framework is more ideal for our problem of interest.

3 Application to COVID-19 severity classification

3.1 ScRNA-seq data and featurization

We integrate 20 publicly available scRNA-seq datasets to form a total of 864 COVID-19 patients with three severity levels marked as “Severe/Critical” (318 patients), “Mild/Moderate” (353 patients), and “Healthy” (193 patients). The detail of each dataset and patient composition can be found in Supplementary Table S1. The severe, moderate and healthy patients are labeled as class 1, 2 and 3, respectively.

For each patient, PBMC scRNA-seq data is available in the form of a matrix recording the expression levels of genes in hundreds to thousands of cells. Following the workflow in Lin et al. (2022), we first perform data integration including cell type annotation and batch effect removal, before selecting $3,000$ highly variable genes and constructing their pseudo-bulk expression profiles under each cell type, where each gene’s expression is averaged across the cells of this type in every patient. The resulting processed data for each patient j is a matrix $A^{(j)} \in \mathrm{R}^{n_g \times n_c}$, where $n_c = 18$ is the number of cell types, and $n_g = 3,000$ is the number of genes for analysis. More details of the integration process can be found in Supplementary Section A. Supplementary Figure S1 shows the distribution of the sparsity levels, i.e., the proportion of genes with zero values, under each cell type across all the patients. Several cell types, despite having a significant proportion of zeros, have varying sparsity across the three severity classes (Supplementary Figure S3), suggesting their activity level might be informative for classification. Since age information is available (although in different forms, see Supplementary Table S4) in most of the datasets we integrate, we include it as an additional clinical variable for classification. The details of processing the age variable are deferred to Supplementary Section A.

Since classical classification methods typically use feature vectors as input, appropriate featurization that transforms the expression matrices into vectors is needed. We propose four ways of featurization that differ in their considerations of the following aspects.

- As we observe the sparsity level in some cell types changes across the severity classes, we expect different treatments of zeros will influence the classification performance. Three approaches are proposed: 1) no special treatment (M.1); 2) remove individual zeros but keeping all cell types (M.4); 3) remove cell types with significant amount of zeros across all three classes (M.2 and M.3).
- Dimension reduction is commonly used to project the information in a matrix onto a vector. We consider performing dimension reduction along different directions, namely row projections, which take combinations of genes (M.2), and column projections, which combine cell types with appropriate weights (M.3 and M.4). We aim to compare choices of projection direction, so we focus on principal component analysis (PCA) as our dimension reduction method.
- We consider two approaches to generate the PCA loadings: 1) overall PCA loadings (M.2 and M.4), where we perform PCA on the whole data to output a loading vector for all patients; 2) patient-specific PCA loadings (M.3), where PCA is performed for each matrix $A^{(j)}$ to get an individual-specific loading vector.

The details of each featurization method are as follows.

M.1 **Simple feature screening:** we consider each element $A_{uv}^{(j)}$ (gene *u* under cell type *v*) as a possible feature for patient *j* and use its standard deviation across all patients, denoted as SD_{uv}, to screen the features. Elements that hardly vary across the patients are likely to have a low discriminative power for classification. Let $SD_{(i)}$ be the *i*-th largest element in $\{SD_{uv} \mid u \in [n_g], v \in [n_c]\}$. The feature vector for each patient consists of the

entries in $\{A_{uv}^{(j)} \mid SD_{uv} \geq SD_{(n_f)}\}$, where n_f is the number of features desired and set to $3,000$.

M.2 Overall gene combination: removing cell types with mostly zero expression values across all patients (details in Supplementary Section A), we select 17 cell types to construct $\tilde{A}^{(j)} \in \mathrm{R}^{n_g \times 17}$ that only preserves columns in $A^{(j)}$ corresponding to the selected cell types. Then, $\tilde{A}^{(1)}, \ldots, \tilde{A}^{(N)}$ are concatenated column-wise to get $\tilde{A}^{\text{all}} \in \mathrm{R}^{n_g \times (N \times 17)}$, where $N = 864$. Let $\tilde{w} \in \mathrm{R}^{n_g \times 1}$ denote the first principle component loadings of $(\tilde{A}^{\text{all}})^{\top}$, and the feature vector for patient j is given by $X_j = \tilde{w}^{\top} \tilde{A}^{(j)}$.

M.3 Individual-specific cell type combination: for patient j, the loading vector $\tilde{w}_j \in \mathrm{R}^{1 \times 17}$ is taken as the absolute values of first principle component loadings for $\tilde{A}^{(j)}$, the matrix with selected 17 cell types in M.2 (details in Supplementary Section A). The principle component loading vector $\tilde{w}_j$ that produces $X_j = (\tilde{A}^{(j)} \tilde{w}_j)^{\top}$ is patient-specific, intending to reflect different cell type compositions in different individuals.

M.4 Common cell type combination: we compute an expression matrix $\bar{A}$ averaged over all patients defined as

$$\bar{A}_{uv} = \frac{\sum_{j \in [N]} A_{uv}^{(j)}}{|\{j \in [N] \mid A_{uv}^{(j)} \neq 0\}|},$$

where $|\cdot|$ is the cardinality function. Let $w \in \mathrm{R}^{n_c \times 1}$ denote the first principle component loadings of $\bar{A}$, then the feature vector for the j-th patient is $X_j = (A^{(j)} w)^{\top}$.

We next evaluate the performance of these featurizations when applied as input to different base classification methods for H-NP classification.

3.2 Results of H-NP classification

After obtaining the feature vectors and applying a suitable base classification method, we apply Algorithm 3 to control the under-classification errors. Recall that Y = 1, 2, 3 represent the severe, moderate and healthy categories, respectively, and the goal is to control $R_{1\star}(\hat{\phi})$ and $R_{2\star}(\hat{\phi})$. In this section, we evaluate the performance of the H-NP classifier applied to each combination of featurization method in Section 3.1 and base classification method (logistic regression, random forest, SVM (linear)), which is used to train the scores (T_1 and T_2). In each class, we leave out 30% of the data as the test set and split the rest 70% as follows for training the H-NP classifier: 35% and 35% of $\mathcal{S}_1$ form $\mathcal{S}_{1s}$ and $\mathcal{S}_{1t}$; 35%, 25% and 10% of $\mathcal{S}_2$ form $\mathcal{S}_{2s}$, $\mathcal{S}_{2t}$ and $\mathcal{S}_{1e}$; 35% and 35% of $\mathcal{S}_3$ form $\mathcal{S}_{3s}$ and $\mathcal{S}_{3e}$. For each combination of featurization and base classification method, we perform random splitting of the observations for 50 times to produce the results in this section.

In Figure 6, the yellow halves of the violin plots show the distributions of different approximate errors from the classical classification methods; Supplementary Table S7 records the averages of these errors. In all the cases, the average of the approximate $R_{1\star}$ error is greater than 20%, in many cases greater than 40%. On the other hand, the approximate $R_{2\star}$ error under the classical paradigm is already relatively low, with the averages around 10%. Under the H-NP paradigm, we set $\alpha_1, \alpha_2 = 0.2$ and $\delta_1, \delta_2 = 0.2$, i.e., we want to control each under-classification error under 20% at a 20% tolerance level.

With the prespecified $\alpha_1, \alpha_2, \delta_1, \delta_2$, for a given base classification method Algorithm 3 outputs an H-NP classifier that controls the under-classification errors while minimizing the weighted sum of the other empirical errors. The blue half violin plots in Figure 6 show the resulting approximate errors after H-NP adjustment. We observe that the common cell type combination feature M.4 consistently leads to smaller errors under both the classical and H-NP classifiers, especially for linear classification models (logistic regression and SVM). We have also implemented a neural network classifier. However, as the training sample size is relatively small, its

performance is not as good as the linear classification models, and the results are deferred to Supplementary Figure S14.

In each plot of Figure 6, the two leftmost plots are the distributions of the two approximate under-classification errors $R_{1\star}$ and $R_{2\star}$. We mark the 80% quantiles of $R_{1\star}$ and $R_{2\star}$ by short black lines (since $\delta_1, \delta_2 = 0.2$), and the desired control levels ($\alpha_1, \alpha_2 = 0.2$) by red dashed lines. The four rightmost plots show the approximate errors for the overall risk and the three components in $R^c(\hat{\phi})$ as discussed in Eq (14). For all the featurization and base classification methods, the under-classification errors are controlled at the desired levels with a slight increase in the overall error, which is much smaller than the reduction in under-classification errors. This demonstrates consistency of our method and indicates its general applicability to various base classification algorithms chosen by users.

Another interesting phenomenon is that when a classical classification method is conservative for specified α_i and δ_i, our algorithm will increase the corresponding threshold t_i, which relaxes the decision boundary for classes less prioritized than i. As a result, the relaxation will benefit some components in $R^c(\hat{\phi})$. In Figure 6(d), in many cases the classifier produces an approximate error $R_{2\star}$ less than 0.2 under the classical paradigm, which means it is conservative for the control level $\alpha_2 = 0.2$ at the tolerance level $\delta_2 = 0.2$. In this case, the NP classifier adjusts the threshold t_2 to lower the requirement for class 3, thus notably decreasing the approximate error of $P_3(\hat{Y} = 2)$.

3.3 Identifying genomic features associated with severity

Finally, we show that using this integrated scRNA-seq data in a classification setting enables us to identify genomic features associated with disease severity in patients at both the cell-type and gene levels. First, by combining logistic regression with an appropriate featurization, we generate a ranked list of features (i.e., cell types or genes) that are important in predicting severity. At the cell type level, we utilize logistic regression with the featurization M.2, which compresses the expression matrix for each patient into a cell-type-length vector, and rank the cell types based

on their coefficients from the log odds ratios of the severe category relative to the healthy category. Supplementary Table S8 shows the top-ranked cell types are CD 14^+ monocytes, NK cells, CD 8^+ effector T cells, and neutrophils, all with significant p-values. This is consistent with known involvement of these cell types in the immune response of severe patients (Lucas et al., 2020; Liu et al., 2020; Rajamanickam et al., 2021).

At the gene level, we utilize logistic regression with the featurization M.4, which has the best overall classification performance, and compresses each patient's expression matrix into a gene-length vector. Similar to the above analysis at the cell-type level, we generate a ranked gene list which leads to the identification of pathways associated with the severe condition. By performing the pathway enrichment analysis on the ranked gene list, we find that the top-ranked genes are significantly enriched in pathways involved in viral defense and leukocyte-mediated immune response (Supplementary Table S9).

Next, we perform further analysis to directly demonstrate the benefits of the H-NP classification results without relying on feature ranking. Based on the featurization M.4, we construct a gene co-expression network and identify modules with groups of genes that are potentially co-regulated and functionally related. By comparing the predicted severity labels from the H-NP classifier and the classical approach, we show that the H-NP labels are better correlated with the eigengenes from these functional modules, suggesting that the H-NP labels better capture the underlying signals in the data related to disease mechanism and immune response (Supplementary Figures S15-S17). Then, we compare the gene ontology enrichment of the functional modules constructed for the severe and healthy patients separately, using the predicted H-NP labels. We find strong evidence of immune response to the virus among severe patients, while no such evidence is observed in the healthy group (Supplementary Tables S10 and S11). Finally, we note that compared with the results from the severe patients as labeled by the classical paradigm, the H-NP paradigm shows more significantly enriched modules with specific references to important cell types, including T cells, and subtypes of T cells (Supplementary Tables S10 and S12). Together, these results demonstrate that by prioritizing the

severe category in our H-NP framework, we can uncover stronger biological signals in the data related to immune response.

More detailed descriptions of the methods used and analysis of results can be found in Supplementary Sections D.4 and D.5.

4 Discussion

In general disease severity classification, under-classification errors are more consequential as they can increase the risk of patients receiving insufficient medical care. By assuming the classes have a prioritized ordering, we propose an H-NP classification framework and its associated algorithm (Algorithm 3) capable of controlling under-classification errors at desired levels with high probability. The algorithm performs post hoc adjustment on scoring-type classification methods and thus can be applied in conjunction with most methods preferred by users. The idea of choosing thresholds on the scoring functions based on a held-out set bears resemblance to conformal splitting methods (Lei, 2014; Wang and Qiao, 2022). However, our approach differs in that we assign only one label to each observation, while maintaining high probability error controls. Additionally, our approach prioritizes certain misclassification errors, unlike conformal prediction which treats all classes equally.

Through simulations and the case study of COVID-19 severity classification, we demonstrate the efficacy of our algorithm in achieving the desired error controls. We have also compared different ways of constructing interpretable feature vectors from the multi-patient scRNA-seq data and shown that the common cell type PCA featurization overall achieves better performance under various classification settings. By performing extensive gene ontology enrichment analysis, we illustrate that the use of scRNA-seq data has allowed us to gain biological insights into the disease mechanism and immune response of severe patients. We note here that although parts of our analysis rely on a ranked feature list obtained from logistic regression, there exist tools to perform such a feature selection step for all the other base classification methods used in this paper, including neural networks, which can utilize saliency maps and other feature selection procedures (Adebayo

et al., 2018; Novakovsky et al., 2023). We have chosen logistic regression in our illustrative analysis based on its stable classification performance and ease of interpretation. In addition, if the main objective is to build a classifier for triage diagnostics using other clinical variables, one can easily apply our method to other forms of patient-level COVID-19 data with other base classification methods.

Even though our case study has three classes, the framework and algorithm developed are general. Increasing the number of classes has no effect on the minimum size requirement of the left-out part of each class for threshold selection since it suffices for each class i to satisfy $n_i \geq \log \delta_i / (1 - \alpha_i)$. We also note that the notion of prioritized classes can be defined in a context-specific way. For example, in some diseases like Alzheimer's disease, the transitional stage is considered to be the most important (Xiong et al., 2006).

There are several interesting directions for future work. For small data problems where the minimum sample size requirement is not full-filled, we might consider adopting a parametric model, under which we can not only develop a new algorithm without minimum sample size requirement, but also study the oracle type properties of the classifiers. In terms of featurizing multi-patient scRNA-seq data, we have chosen PCA as the dimension reduction method to focus on other aspects of comparison; more dimension reduction methods can be explored in future work. It is also conceivable that the class labels in the case study are noisy with possibly biased diagnosis. Accounting for label noise with a realistic noise model and extending the work of Yao et al. (2022) to a multi-class NP classification setting will be another interesting direction to pursue.

Acknowledgements

The authors would like to thank the Editor, Associate Editor, and two anonymous reviewers for their valuable comments, which have led to a much improved version of this paper. The authors would also like to thank Dr Yingxin Lin and the Sydney Precision Data Science Centre for their generous help with curating and processing the COVID-19 scRNA-seq data. The authors gratefully acknowledge: the UT Austin

Harrington Faculty Fellowship to Y.X.R.W. and NSF DMS-2113754 to J.J.L. and X.T. The authors report there are no competing interests to declare.

References

World Health Organization. COVID-19 dashboard, 2023. URL https://covid19.who.int/. Accessed: April 23, 2023.

Rebecca A Betensky and Yang Feng. Accounting for incomplete testing in the estimation of epidemic parameters. *Int J Epidemiol*, 49(5):1419–1426, 2020.

Corbin Quick, Rounak Dey, and Xihong Lin. Regression models for understanding covid-19 epidemic dynamics with incomplete data. *JASA*, 116(536):1561–1577, 2021.

Logan C Brooks, Evan L Ray, et al. Comparing ensemble approaches for short-term probabilistic covid-19 forecasts in the us. *International Institute of Forecasters*, 2020.

Francesca Tang, Yang Feng, et al. The interplay of demographic variables and social distancing scores in deep prediction of us covid-19 cases. *JASA*, 116(534):492–506, 2021.

Daniel J McDonald, Jacob Bien, et al. Can auxiliary indicators improve covid-19 forecasting and hotspot prediction? *PNAS*, 118(51), 2021.

Nick James, Max Menzies, and Peter Radchenko. Covid-19 second wave mortality in europe and the united states. *Chaos*, 31(3):031105, 2021.

Peter Kramlinger, Tatyana Krivobokova, and Stefan Sperlich. Marginal and conditional multiple inference for linear mixed model predictors. *JASA*, 0(ja):1–31, 2022. doi: 10.1080/01621459. 2022.2044826. URL https://doi.org/10.1080/01621459.2022.2044826.

Jiangpeng Wu, Pengyi Zhang, et al. Rapid and accurate identification of covid-19 infection through machine learning based on clinical available blood test results. *MedRxiv*, 2020.

Wei Tse Li, Jiayan Ma, et al. Using machine learning of clinical data to diagnose covid-19: a systematic review and meta-analysis. *BMC Med Inform Decis Mak*, 20(1):1–13, 2020.

Jiawei Zhang, Jie Ding, and Yuhong Yang. Is a classification procedure good enough?—a goodness-of-fit assessment tool for classification learning. *JASA*, pages 1–11, 2021.

Li Yan, Hai-Tao Zhang, et al. Prediction of criticality in patients with severe covid-19 infection using three clinical features: a machine learning-based prognostic model with clinical data in wuhan. *MedRxiv*, 27:2020, 2020.

Liping Sun, Fengxiang Song, et al. Combination of four clinical indicators predicts the severe/critical symptom of patients infected covid-19. *J. Clin. Virol*, 128:104431, 2020.

Zirun Zhao, Anne Chen, et al. Prediction model and risk scores of icu admission and mortality in covid-19. *PloS one*, 15(7):e0236618, 2020.

Anthony Ortiz, Anusua Trivedi, et al. Effective deep learning approaches for predicting covid-19 outcomes from chest computed tomography volumes. *Sci Rep*, 12(1):1–10, 2022.

Norah Alballa and Isra Al-Turaiki. Machine learning approaches in covid-19 diagnosis, mortality, and severity risk prediction: A review. *IMU*, 24:100564, 2021.

Yassine Meraihi, Asma Benmessaoud Gabis, et al. Machine learning-based research for covid-19 detection, diagnosis, and prediction: A survey. *SN computer science*, 3(4):286, 2022.

Katherine A Overmyer, Evgenia Shishkova, et al. Large-scale multi-omic analysis of covid-19 severity. *Cell Syst.*, 12(1):23–40, 2021.

Aaron J Wilk, Arjun Rustagi, et al. A single-cell atlas of the peripheral immune response in patients with severe covid-19. *Nat. Med.*, 26(7):1070–1076, 2020.

Emily Stephenson, Gary Reynolds, et al. Single-cell multi-omics analysis of the immune response in covid-19. *Nat. Med.*, 27(5):904–916, 2021.

Xianwen Ren, Wen Wen, et al. Covid-19 immune features revealed by a large-scale single-cell transcriptome atlas. *Cell*, 184(7):1895–1913, 2021.

Sara Aibar, Celia Fontanillo, et al. Analyse multiple disease subtypes and build associated gene networks using genome-wide expression profiles. *BMC genomics*, 16:1–10, 2015.

Eirini Arvaniti and Manfred Claassen. Sensitive detection of rare disease-associated cell subsets via representation learning. *Nat. Commun.*, 8(1):1–10, 2017.

Zicheng Hu, Benjamin S Glicksberg, and Atul J Butte. Robust prediction of clinical outcomes using cytometry data. *Bioinformatics*, 35(7):1197–1203, 2019.

World Health Organization. Who r&d blueprint novel coronavirus covid-19 therapeutic trial synopsis. *World Health Organization*, pages 1–9, 2020.

Xin Tong, Yang Feng, and Jingyi Jessica Li. Neyman-pearson classification algorithms and np receiver operating characteristics. *Sci. Adv.*, 4(2):eaao1659, 2018.

Yingxin Lin, Lipin Loo, et al. Scalable workflow for characterization of cell-cell communication in covid-19 patients. *PLoS Comp Biol*, 18(10):e1010495, 2022.

Klemens B Meyer and Stephen G Pauker. Screening for hiv: can we afford the false positive rate?, 1987.

Marcel Dettling and Peter Bühlmann. Boosting for tumor classification with gene expression data. *Bioinformatics*, 19(9):1061–1069, 2003.

Charles Elkan. The foundations of cost-sensitive learning. In *International joint conference on artificial intelligence*, volume 17, pages 973–978. Lawrence Erlbaum Associates Ltd, 2001.

Dragos D Margineantu. Class probability estimation and cost-sensitive classification decisions. In *Machine Learning: ECML 2002: 13th European Conference on*

Machine Learning Helsinki, Finland, August 19–23, 2002 Proceedings 13, pages 270–281. Springer, 2002.

Adam Cannon, James Howse, et al. Learning with the neyman-pearson and min-max criteria. *Los Alamos National Laboratory, Tech. Rep. LA-UR*, pages 02–2951, 2002.

Clayton Scott and Robert Nowak. A neyman-pearson approach to statistical learning. *IEEE Trans. Inf. Theory*, 51(11):3806–3819, 2005.

Philippe Rigollet and Xin Tong. Neyman-pearson classification, convexity and stochastic constraints. *JMLR*, 2011.

Lucy Xia, Richard Zhao, et al. Intentional control of type i error over unconscious data distortion: A neyman–pearson approach to text classification. *JASA*, 116(533):68–81, 2021.

Yang Feng, Xin Tong, and Weining Xin. Targeted crisis risk control: A neyman-pearson approach. *Available at SSRN 3945980*, 2021.

Thomas Landgrebe and R Duin. On neyman-pearson optimisation for multiclass classifiers. In *Proceedings 16th Annual Symposium of the Pattern Recognition Association of South Africa. PRASA*, pages 165–170, 2005.

Chengjie Xiong, Gerald van Belle, et al. Measuring and estimating diagnostic accuracy when there are three ordinal diagnostic groups. *Statistics in Medicine*, 25(7):1251–1273, 2006.

Ye Tian and Yang Feng. Neyman-pearson multi-class classification via cost-sensitive learning. *arXiv preprint arXiv:2111.04597*, 2021.

Natalie Stanley, Ina A Stelzer, et al. Vopo leverages cellular heterogeneity for predictive modeling of single-cell data. *Nat. Commun.*, 11(1):1–9, 2020.

Edward A Ganio, Natalie Stanley, et al. Preferential inhibition of adaptive immune system dynamics by glucocorticoids in patients after acute surgical trauma. *Nat. Commun.*, 11(1):1–12, 2020.

Xiaoyuan Han, Mohammad S Ghaemi, et al. Differential dynamics of the maternal immune system in healthy pregnancy and preeclampsia. *Front Immunol.*, page 1305, 2019.

Mark M Davis, Cristina M Tato, and David Furman. Systems immunology: just getting started. *Nat. Immunol.*, 18(7):725–732, 2017.

Carolina Lucas, Patrick Wong, et al. Longitudinal analyses reveal immunological misfiring in severe covid-19. *Nature*, 584(7821):463–469, 2020.

Jing Liu, Sumeng Li, et al. Longitudinal characteristics of lymphocyte responses and cytokine profiles in the peripheral blood of sars-cov-2 infected patients. *EBioMedicine*, 55:102763, 2020.

Anuradha Rajamanickam, Nathella Pavan Kumar, et al. Dynamic alterations in monocyte numbers, subset frequencies and activation markers in acute and convalescent covid-19 individuals. *Sci. Rep.*, 11(1):20254, 2021.

Jing Lei. Classification with confidence. *Biometrika*, 101(4):755–769, 2014.

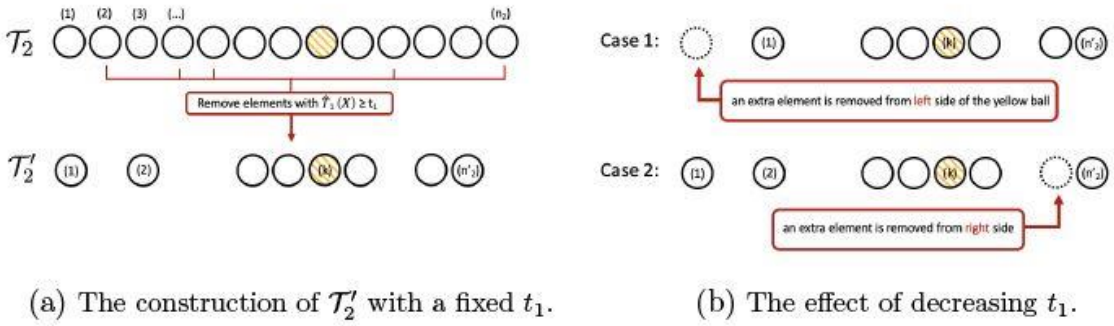
Wenbo Wang and Xingye Qiao. Set-valued support vector machine with bounded error rates. *JASA*, pages 1–13, 2022.

Julius Adebayo, Justin Gilmer, et al. Sanity checks for saliency maps. *NeurIPS*, 31, 2018.

Pengyi Yang, Hao Huang, et al. Feature selection revisited in the single-cell era. *Genome Biol.*, 22:1–17, 2021.

Gherman Novakovsky, Nick Dexter, et al. Obtaining genetics insights from deep learning via explainable artificial intelligence. *Nat. Rev. Genet.*, 24(2):125–137, 2023.

Shunan Yao, Bradley Rava, et al. Asymmetric error control under imperfect supervision: A label-noise-adjusted neyman–pearson umbrella algorithm. *JASA*, pages 1–13, 2022.



(a) The construction of $\mathcal{T}_2'$ with a fixed t_1. (b) The effect of decreasing t_1.

Fig. 1 The influence of t_1 on the error $P_3\left(\hat{Y} = 2\right)$.

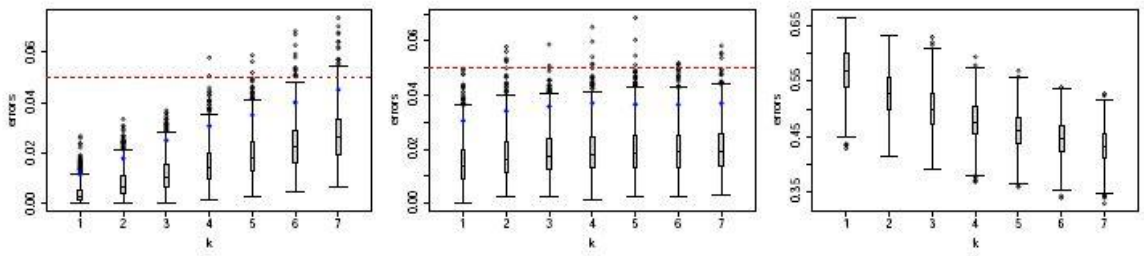
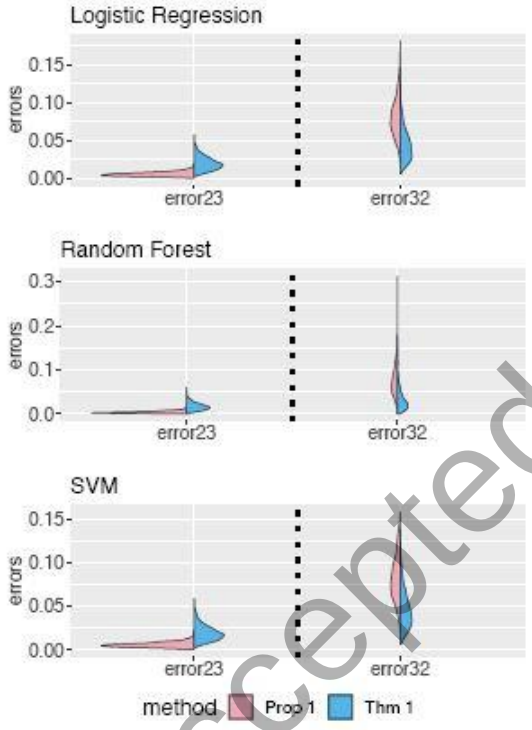
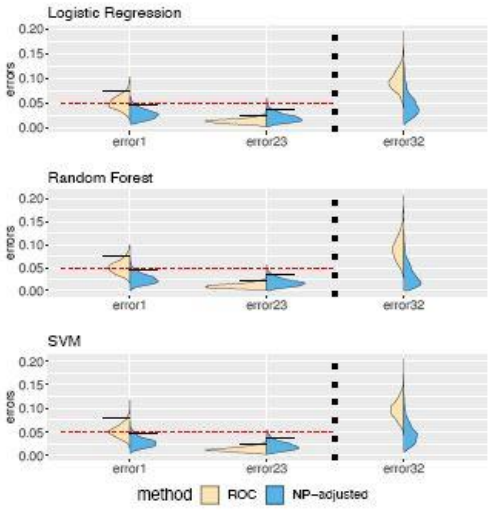


Fig. 2 The distribution of approximate errors on the test set when t_1 is the k-th largest element in $\mathcal{T}_1 \cap (-\infty, \bar{t}_1)$. The 95% quantiles of $R_{1\star}$ and $R_{2\star}$ are marked by blue diamonds. The target control levels for $R_{1\star}(\hat{\phi})$ and $R_{2\star}(\hat{\phi})$ ($\alpha_1 = \alpha_2 = 0.05$) are plotted as red dashed lines.



Logistic Regression		
Method	Error23	Error32
Prop 1	0.006	0.082
Thm 1	0.020	0.046
Random Forest		
Method	Error23	Error32
Prop 1	0.004	0.077
Thm 1	0.017	0.033
SVM		
Method	Error23	Error32
Prop 1	0.006	0.083
Thm 1	0.020	0.047

Fig. 3 The distribution and averages of approximate errors on the test set under the setting **T1.1**. "error23" and "error32" correspond to $R_{2\star}(\hat{\phi})$ and $P_3(\hat{Y} = 2)$, respectively.



Logistic Regression			
Method	Error1 (95% quantile)	Error23 (95% quantile)	Error32 (mean)
ROC	0.074	0.023	0.096
H-NP	0.045	0.036	0.047
Random Forest			
Method	Error1 (95% quantile)	Error23 (95% quantile)	Error32 (mean)
ROC	0.077	0.020	0.093
H-NP	0.047	0.034	0.032
SVM			
Method	Error1 (95% quantile)	Error23 (95% quantile)	Error32 (mean)
ROC	0.078	0.023	0.098
H-NP	0.048	0.037	0.047

Fig. 4 The distributions of approximate errors on the test set under setting **T1.1**. " error1", "error23" and "error32" correspond to $R_{1^*}(\hat{\phi}), R_{2^*}(\hat{\phi})$ and $P_3(\hat{Y}=2)$, respectively.

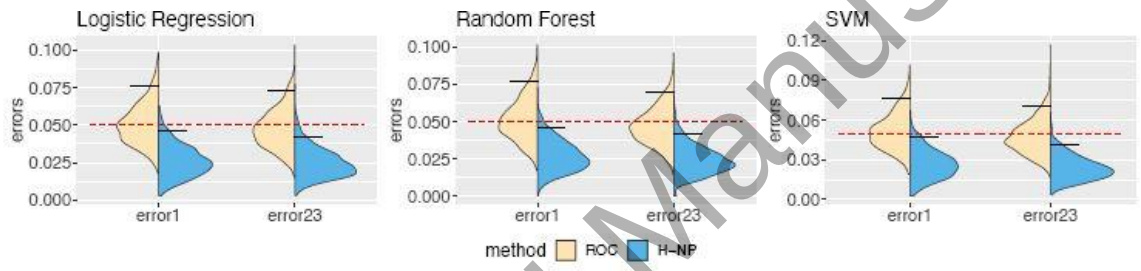


Fig. 5 The distributions of approximate errors on the test set under setting **T2.1**. " error1" and "error23" correspond to the errors $R_{1^*}(\hat{\phi})$ and $R_{2^*}(\hat{\phi})$, respectively.

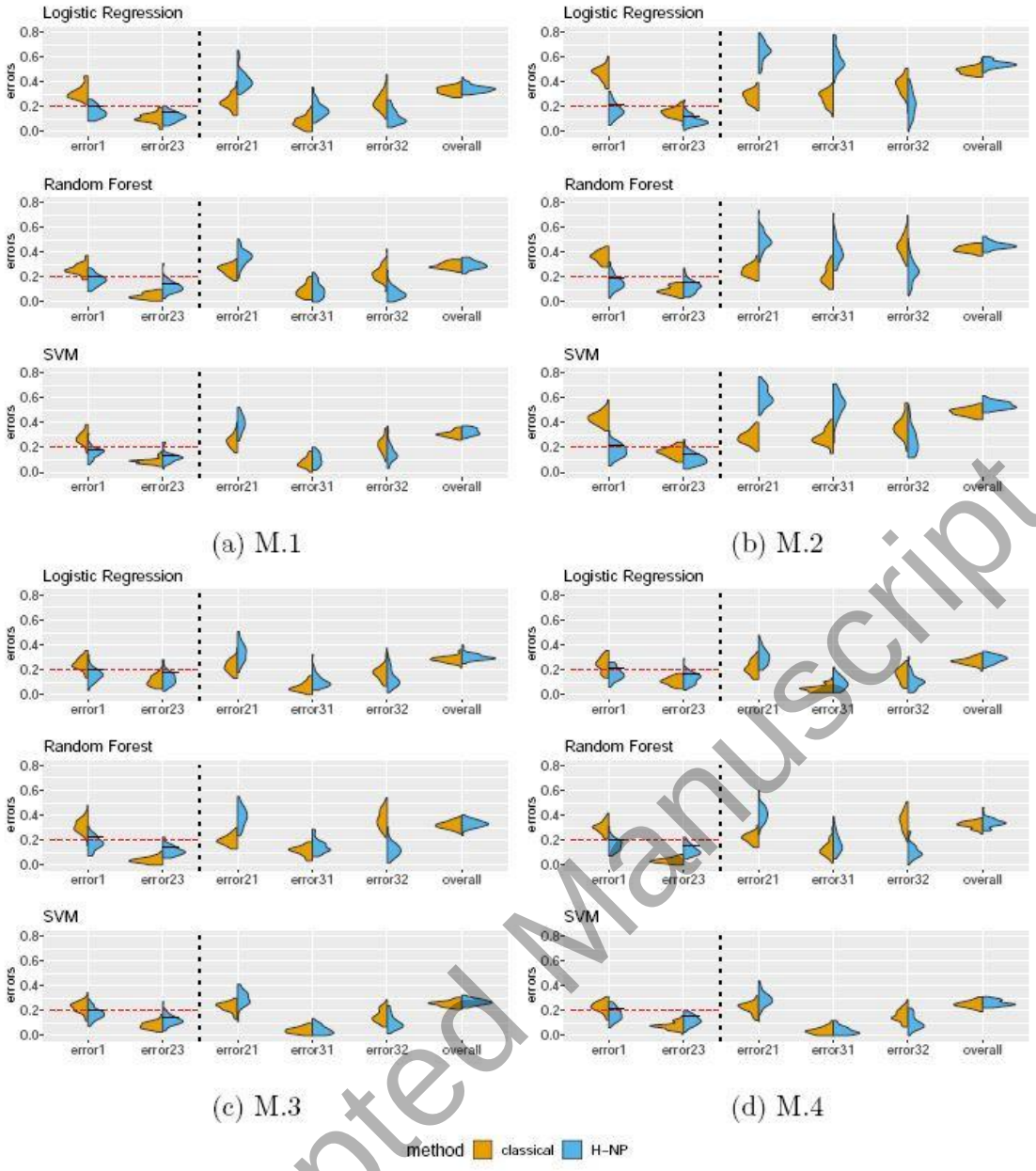


Fig. 6 The distribution of approximate errors for each combination of featurization method and base classification method. “error1”, “error23”, “error21”, “error31”, “error32”, “overall” correspond to $R_{1^\star}(\hat{\phi})$, $R_{2^\star}(\hat{\phi})$, $P_2(\hat{Y}=1)$, $P_3(\hat{Y}=1)$, $P_3(\hat{Y}=2)$ and $P(\hat{Y}\neq Y)$, respectively.