Cross Validation for Correlated Data in Classification Models

Oren Yuval
Tel Aviv university

Saharon Rosset
Tel Aviv university

Abstract

We present a methodology for model evaluation and selection in binary classification models with the presence of correlations in the data, where the sampling mechanism violates the i.i.d. assumption. Our methodology involves a formulation of the bias term between the standard Cross-Validation (CV) estimator and the mean generalization error, and practical data-based procedures to estimate this term. Consequently, we present the bias-corrected CV estimator, which is the standard CV estimate supplemented by an estimate of the bias term. This concept was introduced in the literature only in the context of a linear model with squared error loss as the criterion for prediction performance. Our suggested bias-corrected CV estimator can be applied to any learning model, including deep neural networks, and to standard criteria for prediction performance for classification tasks, including misclassification rate, cross-entropy and hinge loss. We demonstrate the applicability of the proposed methodology in various scenarios where the data contains complex correlation structures (such as clustered and spatial relationships) with synthetic data and real-world datasets, providing evidence that the bias-corrected CV estimator is better than the standard CV estimator.

1 INTRODUCTION

Datasets with complex correlation structures are a staple of modern data analysis. The correlation between observations can stem from spatio-temporal relationships within the samples, as is common in neuroscience and ecology (Roberts et al., 2017). It can also capture clustering of observations into groups, as is common for example in electronic medical records (same hospital, same city) and in computer vision (images of same person). In predictive modeling scenarios, such correlation structures can have major effects on various aspects of the modeling process. They can affect the way models are built, leading to traditional approaches like linear mixed models, generalized linear mixed models (GLMM, Stroup (2012)) and Gaussian process regression (GPR, Williams and Rasmussen (1995)) and also to more modern variants integrating them into methods like deep neural networks (DNNs) and trees (Simchoni and Rosset, 2023; Rabinowicz and Rosset, 2022b). However, they also have implications for model evaluation and model selection, specifically for the use of cross-validation for these purposes (Rabinowicz and Rosset, 2022a; Roberts et al., 2017; Anderson et al., 2019). From an intuitive perspective, when the correlation structure within the training data does not match that between the training data and the prediction points, standard cross-validation (CV) can lead to a misleading assessment of the generalization error. The generalization error refers to the expected loss of a learning algorithm at new, unseen prediction points. This discrepancy could impact both the assessment and the selection of models.

Traditionally, efforts to address this challenge of CV when dealing with changing correlation structures have focused on designing sampling approaches that would allow cross-validation to give valid scores for model evaluation and selection, exactly or approximately, such as the leave cluster out (LCO) approach (Rice and Silverman, 1991), or leave observation from each cluster out (LOFCO, Wu and Zhang (2002)). However, these methods are only applicable to specific correlation structures that involve clusters, and necessitate a balanced data design. In addition, the number of folds in the CV process is determined by the data structure, which includes the number of clusters and their sizes. An approach pertinent to data exhibiting spatial correlation is the h-blocking CV (Burman et al., 1994), in which data are divided into separate blocks (or folds) spaced apart by a certain spatial dis-

Proceedings of the 28[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2025, Mai Khao, Thailand. PMLR: Volume 258.

tance. This technique can be flawed due to the artificial creation of blocks differing from the natural data sampling process, potentially leading to a biased estimate of the prediction error relative to the real generalization error. Recent work by Rabinowicz and Rosset (2022a), introduced an innovative method designed to tackle these challenges by quantifying and correcting the inherent bias in regular CV. The method suggested by Rabinowicz and Rosset (2022a) is versatile and applicable to various types of data sets and correlation scenarios without the limitations associated with the traditional techniques described above. However, its applicability is restricted to linear regression models employing squared loss.

In this work, we take on the challenge of correcting CV in classification tasks in the presence of correlation structures. We derive a surprisingly general expression for the bias of CV, related to the generalization error, for a class of prediction quality scores, and we demonstrate that it can serve to explicitly describe this bias when using the most common measures of classification prediction error: 0-1 loss, cross entropy (Bernoulli log-likelihood) and hinge loss. The expressions we derive are simple and completely general, in that they do not depend on the true model or the modeling approach used for building the models. However, to quantify these general expressions in any specific settings requires knowledge about model form and/or model parameters. One approach we present for estimating this bias in practice is based on the parametric bootstrap and is widely applicable; however, it can be computationally prohibitive in realistic settings. Therefore, we also propose several efficient approximations that apply to specific model settings, in particular when the true model is a GLMM.

To make our overall findings both clear and concise, we concentrate on two main scenarios involving correlated data. The first scenario examines the multiple cluster structure with random intercepts model, which is discussed in Gelman (2007). For instance, repeatedly sampling the same entities on different days results in two observations being correlated if they come from the same entity or the same day. In this scenario, we might be interested in predicting future observations of an already observed entity (partial correlation), or in predicting observations from new, unsampled entities (zero correlation). The second scenario (Random Fields) involves spatial correlation, which occurs when data are collected from a geographical surface and the correlation between two observations is influenced by their spatial distance. For example, data are collected on real estate properties situated in particular geographic locations, and we are interested in predictions on properties in different locations that have a weaker correlation to the collected data.

1.1 Notations and assumptions

We consider a statistical learning process that uses the training set $T = (X, Y) = \{x_i, y_i\}_{i=1}^n$ in order to fit a predictor $\hat{y}(x_{te}; T)$ for a new covariate variable $x_{te} \in \mathbb{R}^p$ to accurately predict the binary outcome $y_{te} \in \{0, 1\}$, where $X \in \mathbb{R}^{n \times p}$ denotes the matrix of fixed effect covariates. We assume that for any $i = 1 \cdots n$, the covariate variable x_i, is an i.i.d. sample from P_x and $y_i \in \{0, 1\}$ is an independent sample from the Bernoulli distribution with the following success rate:

$$P(y_i = 1 | x_i, \delta_T) = g\left(f(x_i) + \delta_{T,i}\right),$$

where $\delta_T \in \mathbb{R}^n$ is the *random effect* vector that depends on the sampling mechanism and the realization of the latent variable that induces correlation across the $\delta_{T,i}$'s. The set of parameters of the distribution P_δ, is denoted by γ_r. The *fixed effect* function $f : \mathbb{R}^p \to \mathbb{R}$ is a nontrivial function for which DNNs and trees are suitable, with parameters γ_c. The link function $g : \mathbb{R} \to (0, 1)$ is a monotonically increasing mapping between the mixed effect (fixed + random) and the expected value of the outcome, such as the Sigmoid or the Probit functions. We also assume that the covariate vector x_{te} is sampled from P_x, and the response y_{te} is a Bernoulli random variable with success rate:

$$P(y_{te} = 1 | x_{te}, \delta_{te}) = g\left(f(x_{te}) + \delta_{te}\right),$$

where $\delta_{te} \in \mathbb{R}$ might depend on the entries of δ_T differently from how they depend on each other, due to the sampling mechanisms of T and (x_{te}, y_{te}). The outlined model covers the GLMM framework by taking $f(x) = x^t\beta$, where $\beta \in \mathbb{R}^p$ denotes an unknown vector of fixed coefficients. It also includes more complex mixed-effects models presented recently (Simchoni and Rosset, 2023; Xiong et al., 2019; Rabinowicz and Rosset, 2022b). However, we assume, for simplicity, that the prediction function $\hat{y}(x_{te}; T)$ is deterministic in both T and x_{te}, which means that there is no additional randomness in the learning process (like that induced by using stochastic gradient descent in DNN learning).

In the illustration of the multiple cluster structure involving q_1 entities and q_2 days, each entity $j_1 \in \{1, \ldots, q_1\}$ generates a random intercept $u_{j_1} \sim N(0, \sigma_u^2)$, while each day $j_2 \in \{1, \ldots, q_2\}$ generates a random intercept $s_{j_2} \sim N(0, \sigma_s^2)$. Assume that the outcome y_i observed for entity j_1 on day j_2, its conditional distribution is determined by the additive random effect $\delta_{tr,i} = u_{j_1} + s_{j_2}$. If the test pair (x_{te}, y_{te}) is

sampled from entity j_1 on a new day, the random effect u_{j_1}, is replicated, and we can write $\delta_{te} = u_{j_1} + s_{te}$ where s_{te} is a new realization, independent of the s_{j_2}'s. If (x_{te}, y_{te}) is sampled from an unseen entity on a new day, we have $\delta_{te} = u_{te} + s_{te}$ where u_{te} is also a new realization independent of the u_{j_1}'s.

In the case of data with spatial Gaussian random field (GRF, Kyriakidis and Journel (1999)) structure, the covariance between two observations is defined by a kernel function $\mathcal{K}(z_{i_1}, z_{i_2}) : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}^+$, which operates on pairs of coordinates. Let $K_{tot} \in \mathbb{R}^{(n+1)\times(n+1)}$ represent the matrix of the values of the kernel function between two coordinates, $K_{tot,i_1,i_2} = \mathcal{K}(z_{i_1}, z_{i_2})$, $i_1, i_2 \in \{1, \cdots, n+1\}$, and let $\delta^t_{tot} = (\delta^t_T, \delta_{te})$ denote the random effect vector of the training and test data. In GRF we assume that $\delta_{tot} \sim MN(0_{n+1}, K_{tot})$, where MN stands for multivariate Gaussian distribution. In this scenario, we consider a specific situation where training data is collected from q geographical regions, selected randomly, and the test observation (x_{te}, y_{te}) is sampled from another randomly chosen region. This typically results in stronger correlations between two observations within the same region compared to those between observations from different regions.

To assess the predictive capacity of a specific learning model, compare models, and tune hyperparameters, it is common to use the K-fold cross-validation procedure. This procedure uses a loss function $L(\cdot, \cdot) : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$, which quantifies the prediction quality. The procedure can be described as follows: The set T is randomly ordered and partitioned into K folds of size $\tilde{n} = n/K \in \mathbb{N}$, $T_1 = \{x_i, y_i\}_{i=1}^{\tilde{n}}, \cdots, T_K = \{x_i, y_i\}_{i=(K-1)\tilde{n}+1}^{n}$. For each fold $k \in \{1, \cdots, K\}$, the model is trained on the set $T_{-k} = (X_{-k}, Y_{-k}) = T \setminus T_k \in \mathbb{R}^{(n-\tilde{n})\times(p+1)}$ which is the whole data without the observations of the kth fold. Then, the predicted outcome is calculated on observations in the kth fold, $i \in \{(k-1)\tilde{n}+1, \cdots, k\tilde{n}\}$, and stored in the CV vector: $\hat{y}^{cv}_i := \hat{y}(x_i; T_{-k})$. The CV estimator is calculated by averaging the loss over CV predictions:

$$\text{CV} = \frac{1}{n}\sum_{i=1}^{n} L\left(y_i, \hat{y}^{cv}_i\right).$$

In what follows, we use the notation T_{-i} instead of T_{-k} to denote the subset of T, of size $n - \tilde{n}$, without the observations from the fold of the ith observation. We note that due to the randomness in the splitting process, any fold T_{-i} shares the same distribution. Moreover, since any covariate vector from (X, x_{te}) is an i.i.d sample from P_x, for any $i \in \{1, \cdots, n\}$, the pair (T_{-i}, x_i) shares the same distribution as (T_{-1}, x_{te}).

The goal of the CV process is to estimate the generalization error (GenErr) of a modeling procedure (like a DNN with specific structure), which is the mean out-of-sample loss of this modeling approach applied to an arbitrary training set of size $n - \tilde{n}$, denoted by T_{-1}:

$$\text{GenErr} = \mathbb{E}_{\delta_{tot}, T, x_{te}, y_{te}} L\left(y_{te}, \hat{y}(x_{te}; T_{-1})\right). \quad (1)$$

Using similar notation and setting, the analysis in Rabinowicz and Rosset (2022a) found that the CV estimator may be biased estimator of GenErr when the correlations between y_{te} and T_{-1} differ compared to the correlation between y_1 and T_{-1}. However, the discussion in Rabinowicz and Rosset (2022a) was limited to regression settings, where the loss function L is the squared difference between y_{te} and $\hat{y}(x_{te}; T_{-1})$. In the next Section, we define a general form of L, which covers standard loss functions in the classification framework, including misclassification rate, cross entropy and hinge loss, under which we can explicitly express the bias of the CV estimator. In Section 3, we suggest an empirical evaluation of the adjusted CV version by employing a parametric bootstrap technique suitable for complex non-linear models.

2 FORMULATION OF CV BIAS IN BINARY CLASSIFICATION

In a classification task, the loss function $L(y, \hat{y}) : \{0, 1\} \times \mathbb{R} \to \mathbb{R}$ assesses the discrepancy between the actual response variable $y \in \{0, 1\}$ and the predicted value $\hat{y}$. We point to the fact that any such function $L(y, \hat{y})$ can be decomposed as follows:

$$\begin{aligned} L(y, \hat{y}) &= y \cdot L(1, \hat{y}) + (1 - y)L(0, \hat{y}) \\ &= \underbrace{L(0, \hat{y})}_{L_1(\hat{y})} - \underbrace{[L(0, \hat{y}) - L(1, \hat{y})]}_{L_2(\hat{y})} \cdot y. \end{aligned} \quad (2)$$

This can be illustrated on standard classification settings:

1. The output $\hat{y} \in (0, 1)$ aims for the probability that $y = 1$, and the performance is measured by the cross entropy loss. In this case, we can write:

$$\begin{aligned} L(y, \hat{y}) &= -y \cdot log(\hat{y}) - (1 - y) \cdot log(1 - \hat{y}) \\ &= \underbrace{-log(1 - \hat{y})}_{L_1(\hat{y})} - \underbrace{log\left(\frac{\hat{y}}{1 - \hat{y}}\right)}_{L_2(\hat{y})} \cdot y. \end{aligned}$$

2. The output $\hat{y} \in \{0, 1\}$ is the best guess for the actual value of y, and the performance is measured by the zero-one loss. We can express this as follows:

$$L(y, \hat{y}) = y \cdot (1 - \hat{y}) + (1 - y) \cdot \hat{y} = \underbrace{\hat{y}}_{L_1(\hat{y})} - \underbrace{(2\hat{y} - 1)}_{L_2(\hat{y})} \cdot y.$$

3. The output $\hat{y} \in \{-\infty, \infty\}$ is some score in favor of $y = 1$, and the performance is measured by the Hinge loss. In this case, the penalty is zero if $y = 1$ and $\hat{y} \geq 1$, or $y = 0$ and $\hat{y} \leq -1$, and increases linearly in $\hat{y}$ otherwise. This can expressed as follows:

$$\begin{aligned} L(y, \hat{y}) &= y \cdot \mathrm{Re}(1 - \hat{y}) + (1 - y) \cdot \mathrm{Re}(1 + \hat{y}) \\ &= \underbrace{\mathrm{Re}(1 + \hat{y})}_{L_1(\hat{y})} - \underbrace{[\mathrm{Re}(1 + \hat{y}) - \mathrm{Re}(1 - \hat{y})]}_{L_2(\hat{y})} \cdot y, \end{aligned}$$

where Re stands for the rectified linear unit (ReLU), $\mathrm{Re}(x) = max\{0, x\}$.

The main idea in correcting the CV estimator, as presented in Rabinowicz and Rosset (2022a), is to identify the bias of the estimator with respect to the generalization error, and denote it by w_{cv}. The correlation-corrected CV estimator is subsequently defined as: $\mathrm{CV}_c = \mathrm{CV} + w_{cv}$. Using the definition of w_{cv}, and the general structure of $L(y, \hat{y}) = L_1(\hat{y}) - yL_2(\hat{y})$, we derive the following explicit expression of w_{cv} in binary classification:

$$\begin{aligned} w_{cv} &= \mathrm{GenErr} - \mathbb{E}_{\delta_T, X, Y}[\mathrm{CV}] \\ &= \mathbb{E}\, L\left(y_{te}, \hat{y}(x_{te}; T_{-1})\right) - \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}\, L\left(y_i, \hat{y}_i^{cv}\right) \\ &= \mathbb{E}\left[L_1\left(\hat{y}(x_{te}; T_{-1})\right) - L_2\left(\hat{y}(x_{te}; T_{-1})\right) \cdot y_{te}\right] \\ &\quad - \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}\left[L_1\left(\hat{y}_i^{cv}\right) - L_2\left(\hat{y}_i^{cv}\right) \cdot y_i\right], \end{aligned}$$

and since (x_i, T_{-i}) have the same distribution as (x_{te}, T_{-1}), we conclude that $\hat{y}_i^{cv}$ and $\hat{y}(x_{te}; T_{-1})$ have the same marginal distribution, and therefore the expectations $\mathbb{E}\left[L_1(\hat{y}(x_{te}; T_{-1}))\right]$ and $\mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} L_1(\hat{y}_i^{cv})\right]$ cancel out in the above expression. Moreover, we can decompose the expectation $\mathbb{E}\left[L_2\left(\hat{y}_i^{cv}\right) \cdot y_i\right]$ into $\mathbb{E}\left[L_2\left(\hat{y}_i^{cv}\right)\right] \cdot \mathbb{E}\left[y_i\right] + \mathbb{C}ov(L_2\left(\hat{y}_i^{cv}\right), y_i)$, and similarly the expectation $\mathbb{E}\left[L_2\left(\hat{y}(x_{te}; T_{-1})\right) \cdot y_{te}\right]$. Since the y_i and y_{te} have the same marginal distribution, they have the same expectation, and again from the fact that $\hat{y}_i^{cv}$ and $\hat{y}(x_{te}; T_{-1})$ have the same marginal distribution, we get that $\mathbb{E}\left[L_2(\hat{y}(x_{te}; T_{-1}))\right] = \mathbb{E}\left[L_2(\hat{y}_i^{cv})\right]$. Therefore:

$$\begin{aligned} w_{cv} = &\frac{1}{n}\sum_{i=1}^{n} \mathbb{C}ov\left(L_2\left(\hat{y}_i^{cv}\right), y_i\right) \\ &- \mathbb{C}ov\left(L_2\left(\hat{y}(x_{te}; T_{-1})\right), y_{te}\right). \end{aligned}$$

We note that the pair (X_{-i}, x_i) and (X_{-1}, x_{te}) are identically distributed, and for any realization $(\widetilde{X}, \tilde{x})$, the equations $\mathbb{E}[y_{te}|x_{te} = \tilde{x}] = \mathbb{E}[y_i|x_i = \tilde{x}]$ and $\mathbb{E}\left[L_2\left(\hat{y}(x_{te}; T_{-1})\right) | (X_{-1}, x_{te}) = (\widetilde{X}, \tilde{x})\right] = \mathbb{E}\left[L_2\left(\hat{y}_i^{cv}\right) | (X_{-i}, x_i) = (\widetilde{X}, \tilde{x})\right]$ hold true. Therefore, when we apply the law of total covariance, the term $\mathbb{C}ov\left(\mathbb{E}\left[L_2\left(\hat{y}_i^{cv}\right)|X\right], \mathbb{E}\left[y_i|x_i\right]\right)$ is cancel out with the term $\mathbb{C}ov\left(\mathbb{E}\left[L_2\left(\hat{y}(x_{te}; T_{-1})\right)|X_{-1}, x_{te}\right], \mathbb{E}\left[y_{te}|x_{te}\right]\right)$, and we derive the final result:

$$\begin{aligned} w_{cv} = &\frac{1}{n}\sum_{i=1}^{n} \mathbb{E}_X\left[\mathbb{C}ov_{\delta_T, Y}\left(L_2\left(\hat{y}_i^{cv}\right), y_i|X\right)\right] \\ &- \mathbb{E}_{X_{-1}, x_{te}}\left[\mathbb{C}ov\left(L_2\left(\hat{y}(x_{te}; T_{-1})\right), y_{te}|X_{-1}, x_{te}\right)\right]. \end{aligned} \tag{3}$$

From the above expression, we conclude that, in general, the bias of the CV estimator is equivalent to the extra covariance between $L_2(\hat{y}_i^{cv})$ and y_i over the one of $L_2(\hat{y}(x_{te}; T_{tr}))$ and y_{te}.

This is our key result, and it applies to all classification loss functions mentioned above. It also applies to other loss functions, like squared loss for regression, and in that sense it generalizes the previous results by Rabinowicz and Rosset (2022a). Interestingly, for the case of regression with squared loss discussed in Rabinowicz and Rosset (2022a), it can be shown that $L_2(\hat{y}) = 2\hat{y}$, and hence their bias expression coincides with the case of zero-one loss for classification.

3 METHODS FOR ESTIMATING THE BIAS OF THE CV ESTIMATOR

We suggest a parametric bootstrap procedure in which we utilize the covariate matrix in hand, X, and estimates $\hat{\gamma}_c$ and $\hat{\gamma}_r$ of the true parameters γ_c and γ_r, that can be obtained by the training sample T. This approach does not require any prior knowledge about the distribution P_x nor any distributional assumptions on it. However, it is necessary to simplify w_{cv} from equation (3), according to the specific scenario. Our objective is to derive a formulation that excludes the expectation over x_{te}, which is not observed in the available training dataset.

In the scenario of the structure of multiple levels clusters, where the test observation is sampled from the same set of entities on a new day, let us denote by $u_{tr} \in \mathbb{R}^{q_1}$, the vector of the realizations of the random intercepts of the entities in the training data. Since the random intercept of the new day is independent of those of the training days, the following holds:

$$\mathbb{C}ov\left(L_2\left(\hat{y}(x_{te}; T_{-1})\right), y_{te}|X_{-1}, x_{te}, u_{tr}\right) = 0.$$

Moreover, when conditioning on $u = u_{tr}, (X_{-1}, x_{te}) = (\widetilde{X}, \tilde{x})$ and $(X_{-i}, x_i) = (\widetilde{X}, \tilde{x})$, we get equalities in the expectations: $\mathbb{E}[y_i] = \mathbb{E}[y_{te}]$ and also $\mathbb{E}[L_2\left(\hat{y}(x_{te}; T_{-1})\right)] = \mathbb{E}[L_2\left(\hat{y}_i^{cv}\right)]$. As a result, by applying the law of total covariance to equation (3), we obtain the appropriate formula for w_{cv} in this context:

$$w_{cv} = \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}_{X, u_{tr}}\left[\mathbb{C}ov\left(L_2\left(\hat{y}_i^{cv}\right), y_i|X, u_{tr}\right)\right].$$

The above result shows that, in a clustered correlation structure, the bias of the CV estimator is equal to the mean conditional covariance between the CV predictions, $L_2(y_i^{cv})$'s and the labels y_i's, when the conditioning yields independence between y_{te} and $\hat{y}(x_{te}; T_{-1})$.

Assuming that the covariate matrix $X = \{x_i\}_{i=1}^n$ is given in random order, and let us denote by C_i, the mean conditional covariance between $L_2(y_i^{cv})$ and y_i: $C_i = \mathbb{E}_{u_{tr}} \mathbb{C}ov\left(L_2(y_i^{cv}), y_i | u_{tr}\right)$. We note that each C_i is an unbiased estimator of w_{cv}, and we suggest estimating w_{cv} by averaging the estimates C_i for $i = 1, \cdots, n$. The estimation of C_i is done by the following parametric bootstrap procedure:

1. Evaluate estimates of the unknown parameters, denoted by $\hat{\gamma}_c$, and $\hat{\gamma}_r = (\hat{\sigma}_u^2, \hat{\sigma}_s^2)$.
2. For $b_1 = 1, \cdots, B_1$:
   (a) Draw $u_{b_1} \sim MN(0_{q_1}, \hat{\sigma}_u^2 I_{q_1})$
   (b) For $b_2 = 1, \cdots, B_2$:
      i. Draw $s_{b_2} \sim MN(0_{q_2}, \hat{\sigma}_s^2 I_{q_2})$
      ii. Draw Y_{b_1,b_2} from Bernuli distribution with mean $g\left(f_{\hat{\gamma}_c}(X) + \delta(u_{b_1}, s_{b_2})\right)$, where $f_{\hat{\gamma}_c}$ is the estimated fixed-effect function, and $\delta(u_{b_1}, s_{b_2})_i = u_{b_1,j_1} + s_{b_2,j_2}$ if observation i samples from entity j_1 on day j_2.
      iii. Apply the learning procedure to the drawn training set $T_{b_1,b_2,-i} = (X_{-i}, Y_{b_1,b_2,-i})$, and calculate $l_{b_1,b_2,i} = L_2(\hat{y}(x_i; T_{b_1,b_2,-i}))$.
   (c) Calculate the inner-sample covariance:

$$\widehat{C}_{i,b_1} = \frac{1}{B_2} \sum_{b_2=1}^{B_2} (l_{b_1,b_2,i} - \overline{l_{b_1,i}})(y_{b_1,b_2,i} - \overline{y_{b_1,i}}),$$

$$\overline{l_{b_1,i}} = \frac{1}{B_2} \sum_{b_2=1}^{B_2} l_{b_1,b_2,i} \; ; \; \overline{y_{b_1,i}} = \frac{1}{B_2} \sum_{b_2=1}^{B_2} y_{b_1,b_2,i}.$$

3. Calculate the mean: $\widehat{C_i} = \frac{1}{B_1} \sum_{b_1=1}^{B_1} \widehat{C}_{i,b_1}$.

Following the basic properties of bootstrap procedure, the result of the suggested algorithm converges to the quantity of interest, C_i, if $(\hat{\gamma}_c, \hat{\gamma}_r) = (\gamma_c, \gamma_r)$, and $B_1, B_2 \to \infty$. However, the major downside is that it requires a great deal of computing power, especially at Step 2.b.iii, where we apply the learning procedure $B_1 \cdot B_2$ times. In Section 3.1 We will show how this issue can be resolved by applying an approximate estimate of $l_{b_1,b_2,i}$, instead of the full fitting process.

In the scenario of spatial correlation, we are not able to achieve an exact simplified formula of w_{cv}. Instead we suggest to embrace a simplified formula that holds in the Linear Mixed models (LMM, Gałecki et al. (2013)) framework, as an approximate measure of w_{cv}. In LMM, the link function g is identity, and the outcome vector (Y_{-i}, y_i) is linear in the random effect vector (δ_{-i}, δ_i). Moreover, as discussed in Rabinowicz and Rosset (2022a), prediction over x_i is linear in Y_{-1}, and can be written as $\hat{y}(x_i; T_{-i}) = h_i^t Y_{-1}$ where the vector h_i depends on the covariates X_{-i} and x_i. Finally, when considering the squared loss criterion, L_2 is a linear function that can be ignored, and we can write:

$$\mathbb{C}ov\left(\hat{y}_i^{cv}, y_i | X, K_{tot}\right) = h_i^t K_{tot_{-i,i}} \; ; \; i \in \{1, \cdots n\},$$

where $K_{tot_{-i,i}}$ is the vector of the values of covariance between δ_i and each entry of δ_{-i}. We note that the whole matrix K_{tot} depends on the realization of the coordinates $Z = \{z_i\}_{i=1}^n$ and z_{te}, but not on X. Similarly, we can write:

$$\mathbb{C}ov\left(\hat{y}(x_{te}; T_{-1}), y_{te} | X_{-1}, x_{te}, K_{tot}\right) = h_{te}^t K_{tot_{-1,te}},$$

where the vector h_{te} depends on the covariates X_{-1} and x_{te}, and $K_{tot_{-1,te}}$ is the vector of the values of covariance between δ_{te} and each entry of δ_{-1}. Let us define $\tilde{k}_i = K_{tot_{-i,i}} - K_{tot_{-1,te}}$, the random vector of differences between the value of covariance between the sampling of Z and z_{te}, and note that h_i and h_{te} are identically distributed, we can express w_{cv} in the LMM scenario as follows:

$$\begin{aligned} w_{cv} &= \mathbb{E}_{X,Z}\left[h_i^t K_{tot_{-i,i}}\right] - \mathbb{E}_{X,Z,x_{te},z_{te}}\left[h_{te}^t K_{tot_{-1,te}}\right] \\ &= \mathbb{E}\left[h_i^t \tilde{k}_i\right] = \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}\left[\mathbb{C}ov\left(\hat{y}_i^{cv}, y_i | X; \widetilde{K}\right)\right], \end{aligned}$$

where the notation $\mathbb{C}ov(\cdot, \cdot; \widetilde{K})$ implies that the randomness of Y is induced by random effect vector $\delta \sim MN(0, \widetilde{K})$ and each column of $\widetilde{K}$ satisfies $\widetilde{K}_i = \tilde{k}_i$.

The above formula can be used to approximately estimate the w_{cv} term in scenario of regions-wise sampling of observations with binary outcome. We do it by defining the covariance matrix $\widetilde{K}$ as follows: $\widetilde{K}_{i_1 i_2} = 0$ if i_1 and i_2 are not from the same region, and otherwise, $\widetilde{K}_{i_1 i_2} = \widehat{\mathcal{K}}(z_{i_1}, z_{i_2}) - \overline{K}$, where $\widehat{\mathcal{K}}$ is the kernel function with the estimated parameters $\hat{\gamma}_r$, and $\overline{K}$ is the estimated mean covariance between the two observations from different regions. The algorithm designed for clustered scenarios can be modified for this context by establishing a primary loop consisting of B iterations. In each iteration $b = 1, \cdots, B$, a random effect vector δ_b is sampled from $MN(0, \widetilde{K})$. Subsequently, the label vector Y_b is sampled from a Bernoulli distribution with mean $g\left(f_{\hat{\gamma}_c}(X) + \delta_b\right)$.

3.1 Fast-Approximated estimation for GLMM

In GLMM for binary regression the assumed fixed model is $f(x) = x^t \beta$, and the fitting procedure of the

estimators $(\hat{\beta}, \hat{\gamma}_r)$, as described in Zhang et al. (2011), involves the optimization of the Bernoulli likelihood, $\mathcal{L}$ of a given training sample T_{-k}, $k \in \{1, \cdots, K\}$, with respect to (β, γ_r), and can be written as follows:

$$\mathcal{L} = \int \prod_{i=(k-1)\tilde{n}+1}^{k\tilde{n}} g_i^{y_i}(1-g_i)^{1-y_i} p(\delta; \gamma_r) d\delta \qquad (4)$$

where $g_i = g(x_i^t\beta + \delta_i)$, and g is the link function mapping between the mixed linear score to the success rate. Usually, the optimization of $\mathcal{L}$ is done in an iterative manner and requires a lot of computational resources. When it comes to prediction, we first consider the scenario in which the predicted value $\hat{y}(x_i; T_{-i})$ is a well-defined function of $x_i^t\hat{\beta}_{T_{-i}}$ (depending on the model performance criterion), and therefore the statistic $l_{b,i}$ in the bootstrap procedure is a well-defined function of $\hat{\beta}_{T_{b,-i}}$, where $\hat{\beta}_{T_{b,-i}}$ is the fitted estimator of β based on the bootstrap training sample $T_{b,-i}$. Thus, our goal is to find an efficient but informative method to fit the estimator $\hat{\beta}_{T_{b,-i}}$ inside the bootstrap loop. In scenarios involving outer and inner loops, the subscript b can denote $b = (b_1, b_2)$.

In order to fit $\hat{\beta}_{T_{b,-i}}$ efficiently, we embrace the idea of Quasi-Likelihood (QL) of the marginal likelihood of β (discussed in Wolfinger and O'connell (1993) and in Ju et al. (2020)), while using the parameter $\hat{\gamma}_r$ that was fitted once at the beginning of the algorithm using the whole data set T. We assume that $\hat{\beta}_{T_{b,-i}}$ obtained by optimizing $\mathcal{L}$, approximately satisfies the following QL equations:

$$S(\beta; T_{b,-i}) = \left[X^t D V^{-1}\right]_{-i} (Y_{b,-i} - \mu_{-i}) = 0,$$

where the subscript $-i$ stands for the elimination of rows and columns in the same fold of observation i. The matrix $V \equiv V(\beta, \hat{\gamma}_r) = \mathbb{C}ov(Y|X; \beta, \hat{\gamma}_r)$, is the covariance matrix of Y, while conditioning on X, and assuming that the parameters of the model are β and $\hat{\gamma}_r$. In particular, for any pair of indices (i_1, i_2) such that $i_1 \neq i_2$, we can write: $V_{i_1 i_2} = \mathbb{C}ov_\delta\left(g(x_{i_1}^t\beta + \delta_{i_1}), g(x_{i_2}^t\beta + \delta_{i_2})\right)$, and the diagonal elements also contain the iterative term: $\mathbb{E}_\delta\left[\mathbb{V}ar(y_i|x_i, \delta_i; \beta)\right]$. The vector $\mu \in \mathbb{R}^n$ satisfies $\mu_i = \mathbb{E}_\delta\left[g(x_i^t\beta + \delta_i)\right]$ and $D \in \mathbb{R}^{n\times n}$ is a diagonal matrix with elements $D_{ii} = \frac{\partial \mu_i}{\partial (x_i^t\beta)} = \mathbb{E}_\delta\left[g'(x_i^t\beta + \delta_i)\right]$.

Intuitively, the likelihood equations presented above aim to minimize the discrepancy between any label y and its expectation μ, while taking into account the correlation structure between observations, and averaging it over the distribution of the random effect δ. The standard assumption of the QL approach is that the solution $\breve{\beta}_{T_{b,-i}}$ of the QL equations is a good approximation of $\hat{\beta}_{T_{b,-i}}$ in terms of its correlation with $Y_{b,i}$. Nevertheless, finding an exact closed of $\breve{\beta}_{T_{b,-i}}$ is not possible, and we use Taylor expansion around the vector $\hat{\beta}$ (evaluated globally only once) to achieve the following quadratic approximation:

$$\breve{\beta}_{T_{b,-i}} \approx \hat{\beta} + \left[(X^t D V^{-1} D X)^{-1}\right]_{-i} S(\hat{\beta}; T_{b,-i}) = \tilde{\beta}_{b,-i},$$

where the entries of V, μ, and D are evaluated globally with $\hat{\beta}$, and by sampling random draws of δ from the distribution $P_{\delta;\hat{\gamma}_r}$.

The approximate solution $\tilde{\beta}_{b,-i}$ can be calculated quickly for any draw of Y_b, and $\tilde{l}_{b,i} = L_2(\hat{y}(x_i^t\tilde{\beta}_{b,-i}))$ can be obtained immediately. However in some cases, the predicted value $\hat{y}(x_i; T_{-i})$ may depend on some predictor of the specific realization of a latent random effect. Specifically, in the scenario of multiple-level clustered structure, assuming that observations i is from entity j, the predicted value $\hat{y}(x_i; T_{-i})$ is a well-defined function of $x_i^t\hat{\beta}_{T_{-i}} + r_{te}^t\hat{u}_{T_{-i},j1}$, where $\hat{u}_{T_{-1}} \in \mathbb{R}^{q_1}$ is the predictor of u_{tr}, based on the training sample T_{-i}. The estimation of $\hat{\beta}_{T_{b,-i}}$ stays consistent with the previously covered scenario, ensuring that the approximate estimator $\tilde{\beta}_{b,-i}$ is still applicable. Consequently, the task now is to develop a fast approximation approach for estimating $\hat{u}_{T_{b,-i}}$.

Usually, the predictor $\hat{\eta}^t = (\hat{u}^t, \hat{s}^t)$ is obtained by maximizing the joint likelihood of Y_{-i} and η, substituting β with $\hat{\beta}_{T_{-i}}$, which can be written as follows:

$$\mathcal{L}(\eta, Y_{-i}|X_{-i}; \hat{\beta}_{T_{-i}}, \hat{\gamma}_r) \propto P_{Y_{-i}|X_{-i},\eta;\hat{\beta}_{T_{-i}}} \cdot P_{\eta;\hat{\gamma}_r},$$

where $\hat{\gamma}_r = (\hat{\sigma}_u^2, \hat{\sigma}_s^2)$. In the Quasi-Likelihood approach, this yields the following set of equations:

$$S(\eta) = \left[Q^t D_\eta V_\eta^{-1}\right]_{-i} (Y_{b,-i} - \mu_{\eta,-i}) + V_{\hat{\gamma}_r}^{-1}\eta = 0,$$

where $Q = \partial\delta/\partial\eta \in \mathbb{R}^{n\times(q_1+q_2)}$, and the matrix $V_\eta \equiv V(\hat{\beta}_{T_{-i}}, \eta) = \mathbb{C}ov(Y|X, \eta; \hat{\beta}_{T_{-i}})$ is the covariance matrix of Y, while conditioning on X and η. Moreover, $\mu_\eta = g\left(X\hat{\beta}_{T_{-i}} + \delta(\eta)\right)$, and $D_{ii} = g'(x_i^t\hat{\beta}_{T_{-i}} + \delta_i(\eta))$. The matrix $V_{\hat{\gamma}_r} = diag((\hat{\sigma}_u^2 1_{q_1}^t, \hat{\sigma}_s^2 1_{q_2}^t))$ is the covariance matrix of η.

Using a Taylor expansion around the generated vector $\eta_b^t = (u_{b_1}^t, s_{b_2}^t)$, and substituting $\hat{\beta}_{T_{-i}}$ with $\tilde{\beta}_{b,-i}$, we achieve the following approximation of the solution $\hat{\eta}_{T_{b,-i}}$ in the bootstrap procedure:

$$\hat{\eta}_{T_{b,-i}} \approx \eta_b + \left(\frac{\partial S}{\partial \eta}|_{\eta=\eta_b}\right)^{-1} S(\eta_0) := \tilde{\eta}_{b,-i},$$

where $\partial S/\partial\eta = \left[Q^t D_\eta V_\eta^{-1} D_\eta Q\right]_{-i} + V_{\hat{\gamma}_r}^{-1}$. Now, the approximate formula for the predictor $\hat{u}_{T_{b,-i}}$ is first q_1 rows of $\tilde{\eta}_{b,-i}$: $\tilde{u}_{b,-i} = [\tilde{\eta}_{b,-i}]_{(1:q_1)}$. With the suggested formulas of $\tilde{\beta}_{b,-i}$ and $\tilde{u}_{b,-i}$, we can quickly calculate the estimator $\tilde{w}_{cv}$ of w_{cv} in this scenario. The

bias-corrected CV estimator obtained from the fast-approximated procedure is $\widetilde{\text{CV}}_c = \text{CV} + \tilde{w}_{cv}$

3.2 Adaptations to Deep Learning

As discussed in the Introduction, some recent studies (Simchoni and Rosset, 2021, 2023; Tran et al., 2020; Xiong et al., 2019) have suggested integrating the idea of mixed-effects model to deep neural networks (DNNs),where the fixed effect function f_{γ_c} can be a non-trivial neural network. The main idea (and also the main challenge) is to optimize the likelihood function $\mathcal{L}$ with respect to γ_c and γ_r, $(\hat{\gamma}_c, \hat{\gamma}_r) = \underset{\gamma_c,\gamma_r}{argmin}\{\mathcal{L}\}$, where $g_i = g(f_{\gamma_c}(x_i) + \delta_i)$ and:

$$\mathcal{L} = \int \prod g_i^{y_i}(1-g_i)^{1-y_i} p(\delta;\gamma_r) d\delta. \tag{5}$$

In the context of this paper, we consider the parameter learning algorithm to be a black box, and we implement a cross-validation procedure when handling correlated data. In this case, the methodology suggested for estimating $\hat{w}_{cv}$ outlined in this section can be applied, regardless of the complexity of f_{γ_c}. However, the computational resources required might be unbearable when f_{γ_c} is a complex DNNs trained on a large-scale dataset. We propose two methods to address this problem and estimate the bias of the CV estimator in complex deep mixed models:

Hot-start training. For each random training sample $T_{b,-i}$, start the training process with the parameters $\hat{\gamma}_c$, and $\hat{\gamma}_r$, which were already globally fitted once, based on the whole data T. Moreover, limit the number of iterations as necessary to control the required computational resources.

Last-layer analysis. Many DNN models have a fully connected layer with the appropriate activation function for the type of model as the output layer of f_{γ_c}. This layer can be viewed as a GLMM, while the rest of the network is fixed. Therefore, we can apply the fast approximate estimation outlined in Section 3.1 to the last layer, while all other parameters are treated as given. To be more precise, we write $\gamma_c = \{W, \beta\}$ where W is the set of parameters of the hidden layers of f_{γ_c} and β is the parameter vector of the last layer. The output of the model in the bootstrap loop can be approximated as $\hat{y}(x_i; T_{b,-i}) \approx g\left(\left(\tilde{f}(x_i;\widehat{W})\right)^t \tilde{\beta}_{b,-i}\right)$, where $\tilde{f}$ is the model up to the last layer, $\widehat{W}$ is the estimator of W based on the whole data T, and $\tilde{\beta}_{b,-i}$ is calculated as described in Section 3.1 while substituting X with $\tilde{f}(X;\widehat{W})$.

Intuitively, we expect that both methods will result in an underestimation of the real bias term w_{cv}, because of the fact that changes of $\hat{y}(x_i; T_{b,-i})$ in $Y_{b,-i}$ are limited, compared to the case of a full learning process from the beginning. However, constraining the variability of $\hat{y}(x_i; T_{b,-i})$ can guarantee that the adjusted estimator $\widetilde{\text{CV}}_c = \text{CV} + \tilde{w}_{cv}$ does not experience a higher variance relative to CV, while simultaneously diminishing some bias with respect to the true generalization error. The empirical study on real data in Section 4 suggests that the use of $\widetilde{\text{CV}}_c$ is beneficial as an estimator of the real generalization error in terms of bias and variance.

4 EMPIRICAL STUDY

We illustrate how the proposed methodology can be applied across three distinct learning scenarios: A simulated mixed logistic regression with two clustered correlation structure; a Convolutional Neural Network (CNN) model on celebrity images, with a clustering structure based on the celebrity's identity; and a Fully connected DNN model Airbnb rental properties, with a spatial correlation structure.

In all scenarios, we consider two different criteria for the prediction error: cross-entropy loss and zero-one loss. For each criterion, we compare the standard CV estimator with our suggested estimator $\widetilde{\text{CV}}_c$. In the subsections that follow, we present each scenario along with the primary outcomes. Comprehensive information about the simulations, such as hyper-parameters, optimization methods, computational duration, e.t.c., is included in Appendix A. Relevant datasets and code are available at `https://github.com/OrenYuv/CVc-for-General-Models`.

4.1 Mixed Logistic Regression

We perform an experiment on simulated data in which the outcome y follows the Bernoulli distribution with two-level random intercept. Formally: $y_{i,j_1,j_2} \sim Ber\left(\frac{exp\{LP_{ij_1j_2}\}}{1+exp\{LP_{ij_1j_2}\}}\right)$ for $j_1 \in \{1,\cdots,10\}$ and $j_2 \in \{1,\cdots,5\}$, where $LP_{ij_1j_2} = x^t_{ij_1j_2}\beta + u_{j_1} + s_{j_2}$, $\beta = 0.5 \cdot \mathbf{1}_p$, $p = 10$. Additionally, $u_{j_1} \sim N(0, \sigma_u^2)$, $s_{j_2} \sim N(0, \sigma_s^2)$, where $\sigma_u = 1$ and $\sigma_s = 0.5$.

The focus of this experiment is on the predictive capability of the relevant model from the GPBoost library (Sigrist, 2022, 2021), where the training sample size is $n = 110$, and a 11-Folds-CV procedure aims to estimate the performance of the model with 100 observations. The model is treated as a black box, producing an estimator $\hat{\beta}$ of β, estimates of σ_s and σ_u, and predicted random effects $\hat{s}$ and $\hat{u}$ as output.

In the described setting, we generate 2000 training samples, and calculate the 11-Folds-CV estimator for

each of them, as well as an estimation of the real Generalization Error. First, we consider the scenario where the realizations of u_{j_1}'s and s_{j_2}'s in the test set are not the same as in the training set. In Figure 1 we present the density of the statistics calculated on the 2000 training samples. We additionally examine the suitability of $\widetilde{\text{CV}}_c$ within the model selection framework, which considers three distinct models: the fully specified model incorporating all covariates in x, a model utilizing only the initial 7 entries of x, and a model employing just the first 2 entries. In figure 2, it is evident that $\widetilde{\text{CV}}_c$ ranks the models correctly according to the true cross-entropy generalization error, whereas the conventional CV estimator ranks them in the reverse order.

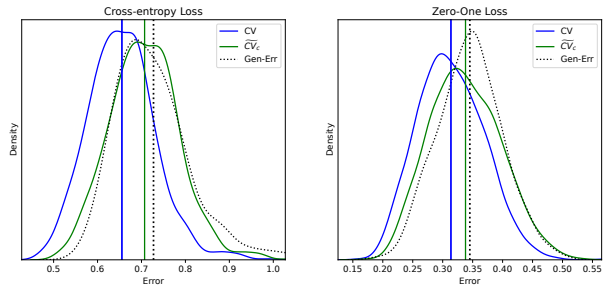

Figure 1: Densities of CV, $\widetilde{CV}_c$ and the generalization error for different Loss functions in Logistic regression model. The mean values are denoted by vertical lines.

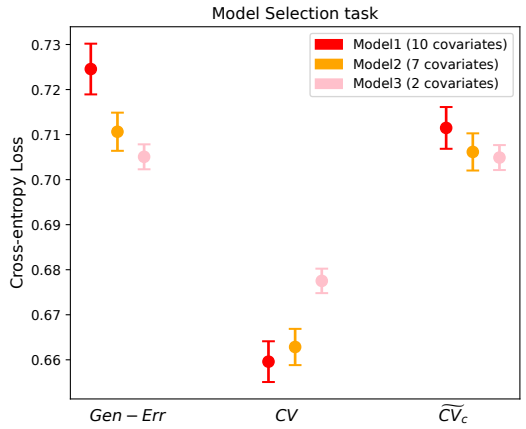

Figure 2: Mean values of CV, $\widetilde{CV}_c$ and the generalization error and their 95% CIs for the three models.

4.2 CNN model on the CelebA DataSet

We demonstrate the last-layer analysis method in CNN aimed at the task of smile detection in CelebA facial images. The CelebA dataset (Liu et al., 2018) contains 202,599 facial images from 10,177 celebrities, each of which has between 1 and 35 images, with an average of 20 images for a celebrity. Every image is labeled 1 if the celebrity smiles and 0 otherwise. We use the same mixed-CNN as in Simchoni and Rosset (2021), treating the celebrity identity as the random-effects feature, where the last layer of the CNN model is a fully connected layer with sigmoid activation and approximate binary NLL objective function, as suggested in Simchoni and Rosset (2023). For the purpose of error estimation, we isolate the last layer and view it as a mixed logistic regression model. We then apply the fast-approximated estimate of w_{cv} and calculate the estimator $\widetilde{CV}_c$. We aim to evaluate how well the model can predict outcomes for new random identities that it did not encounter during the training stage.

A training set of all images from random 200 identities was drawn and randomly splitted into 10 folds in order to evaluate CV, and $\widetilde{CV}_c$ estimators. The test error was calculated by averaging the prediction error of other random 900 identities. In Figures 3 we present the results of 200 random samples. We can see that the estimator $\tilde{w}_{cv}$ effectively captures most of the bias observed between the CV estimate and the true generalization error.

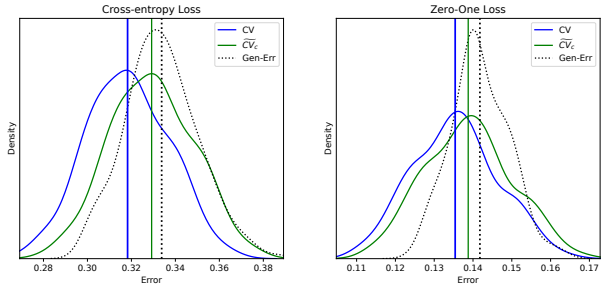

Figure 3: Densities of CV , $\widetilde{CV}_c$ and generalization in the CNN model for smile detection on CelebA dataset. The mean values are denoted by vertical lines.

4.3 DNN model on the AirBNB

We implement the Hot-start technique in a fully connected DNN model using a data set comprising 50,000 AirBNB properties, as collected and pre-processed by Rezazadeh Kalehbasti et al. (2021). The model is trained using properties situated in randomly chosen small regions, each spanning 100 meters, with labels assigned as 1 if the property features hot water. Spatial correlation is incorporated using a Gaussian kernel based on property distances, with its parameters co-optimized with the model's. The sampling and evalua-

tion mechanism is similar to Section 4.2, aiming to assess the model's ability to predict labels for new properties in different regions. In the results (Figure 4), we can see that the estimator $\widetilde{CV}_c$ is effective in reducing the bias of the CV estimator, especially in the case of zero-one loss. In the case of cross-entropy loss, GenErr is very nosy with a long tail to the right, that is not captured by the approximate estimator $\widetilde{CV}_c$.

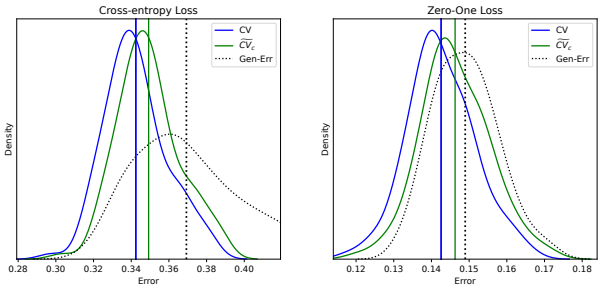

Figure 4: Densities of CV , $\widetilde{CV}_c$ and generalization in the CNN model for hot water feature detection on AirBNB dataset. The mean values are denoted by vertical lines.

5 CONCLUSION

In this paper, we have demonstrated how CV can be adapted to account for correlation structures in data, in particular when the underlying random effects distribution changes between the training data (which is used for CV) and the prediction set. Our methods are based on a careful analysis of the effect of covariance structures on the model evaluation, resulting in a clear formulation of the bias between standard CV and generalization error, and are demonstrated to be effective in simulation and on real data.

We have focused on classification using the most standard evaluation metrics (cross entropy, zero-one loss, hinge loss); however, our approach can be applied to other problem settings as well, such as the area under the ROC curve (AUC). Preliminary results suggest that CV for AUC can be corrected by directly extending our methodology, and additional extensions are detailed in Yuval and Rosset (2025).

Acknowledgements

We would like to convey our appreciation to the reviewers for their insightful comments. The authors acknowledge with gratitude *the Israeli Science Foundation grant 2180/20* and the support from *the Israel Council for Higher Education Data-Science Centers*.

References

Airbnb public dataset. *http://insideairbnb.com/get-the-data.html*, 2018.

C. Anderson, R. Hafen, O. Sofrygin, L. Ryan, and H. Community. Comparing predictive abilities of longitudinal child growth models. *Statistics in medicine*, 38(19):3555–3570, 2019.

P. Burman, E. Chow, and D. Nolan. A cross-validatory method for dependent data. *Biometrika*, 81(2):351–358, 1994.

A. Gałecki, T. Burzykowski, A. Gałecki, and T. Burzykowski. *Linear mixed-effects model*. Springer, 2013.

A. Gelman. *Data analysis using regression and multilevel/hierarchical models*. Cambridge university press, 2007.

P. Jäckel. A note on multivariate gauss-hermite quadrature. *London: ABN-Amro. Re*, 2005.

K. Ju, L. Lin, H. Chu, L.-L. Cheng, and C. Xu. Laplace approximation, penalized quasi-likelihood, and adaptive gauss–hermite quadrature for generalized linear mixed models: Towards meta-analysis of binary outcome with sparse data. *BMC medical research methodology*, 20:1–11, 2020.

P. C. Kyriakidis and A. G. Journel. Geostatistical space–time models: a review. *Mathematical geology*, 31:651–684, 1999.

Z. Liu, P. Luo, X. Wang, and X. Tang. Large-scale celebfaces attributes (celeba) dataset. *Retrieved August*, 15(2018):11, 2018.

A. Rabinowicz and S. Rosset. Cross-validation for correlated data. *Journal of the American Statistical Association*, 117(538):718–731, 2022a.

A. Rabinowicz and S. Rosset. Tree-based models for correlated data. *Journal of Machine Learning Research*, 23(258):1–31, 2022b.

P. Rezazadeh Kalehbasti, L. Nikolenko, and H. Rezaei. Airbnb price prediction using machine learning and sentiment analysis. In *international cross-domain conference for machine learning and knowledge extraction*, pages 173–184. Springer, 2021.

J. A. Rice and B. W. Silverman. Estimating the mean and covariance structure nonparametrically when the data are curves. *Journal of the Royal Statistical Society: Series B (Methodological)*, 53(1):233–243, 1991.

D. R. Roberts, V. Bahn, S. Ciuti, M. S. Boyce, J. Elith, G. Guillera-Arroita, S. Hauenstein, J. J. Lahoz-Monfort, B. Schröder, W. Thuiller, et al. Cross-validation strategies for data with temporal, spatial,

hierarchical, or phylogenetic structure. *Ecography*, 40(8):913–929, 2017.

F. Sigrist. Gpboost: Combining tree-boosting with gaussian process and mixed effects models. In *https://github.com/fabsig/GPBoost*, 2021.

F. Sigrist. Gaussian process boosting. *Journal of Machine Learning Research*, 23(232):1–46, 2022. URL `http://jmlr.org/papers/v23/20-322.html`.

G. Simchoni and S. Rosset. Using random effects to account for high-cardinality categorical features and repeated measures in deep neural networks. *Advances in Neural Information Processing Systems*, 34:25111–25122, 2021.

G. Simchoni and S. Rosset. Integrating random effects in deep neural networks. *Journal of Machine Learning Research*, 24(156):1–57, 2023.

W. W. Stroup. *Generalized linear mixed models: modern concepts, methods and applications*. CRC press, 2012.

M.-N. Tran, N. Nguyen, D. Nott, and R. Kohn. Bayesian deep net glm and glmm. *Journal of Computational and Graphical Statistics*, 29(1):97–113, 2020.

C. Williams and C. Rasmussen. Gaussian processes for regression. *Advances in neural information processing systems*, 8, 1995.

R. Wolfinger and M. O'connell. Generalized linear mixed models a pseudo-likelihood approach. *Journal of statistical Computation and Simulation*, 48(3-4): 233–243, 1993.

H. Wu and J.-T. Zhang. Local polynomial mixed-effects models for longitudinal data. *Journal of the American Statistical Association*, 97(459):883–897, 2002.

Y. Xiong, H. J. Kim, and V. Singh. Mixed effects neural networks (menets) with applications to gaze estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7743–7752, 2019.

O. Yuval and S. Rosset. Cross validation for correlated data in regression and classification models, with applications to deep learning. *arXiv preprint arXiv:2502.14808*, 2025.

H. Zhang, N. Lu, C. Feng, S. W. Thurston, Y. Xia, L. Zhu, and X. M. Tu. On fitting generalized linear mixed-effects models for binary responses using different statistical packages. *Statistics in Medicine*, 30(20):2562–2572, 2011.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Not Applicable]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Not Applicable]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]. It will be included in the supplementary material by the deadline.
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]. It will be included in the supplementary material by the deadline.
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes] It will be included in the supplementary material by the deadline.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Yes]
   (b) The license information of the assets, if applicable. [Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]
   (d) Information about consent from data providers/curators. [Not Applicable]

(e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

A COMPREHENSIVE DETAILS FOR SIMULATIONS

A.1 Mixed Logistic Regression

In the Mixed Logistic Regression experiment, the data for each random sample of size $n = 110$ are generated as follows:

- The grouping variables $j_1 \in \{1, \cdots, 10\}$ and $j_2 \in \{1, \cdots, 5\}$ assigned evenly such that 11 observations are assigned to each cluster j_1 and 22 observations are assigned to each cluster j_2, with a random order. This data is stored in a one-hot encoded matrix $Z \in \{0,1\}^{n \times (q_1, q_2)}$.
- The covariate matrix $X \in \mathbb{R}^{n \times p}$ is drawn from a multivariate Gaussian distribution with a general structure that takes into account the grouping relationships.
- We draw latent variables: $u_{tr} \sim MN(0_{q_1}, \sigma_u^2 I_{q_1})$, $s_{tr} \sim MN(0_{q_2}, \sigma_s^2 I_{q_2})$ and stored in $r_{tr}^t = (u_{tr}^t, s_{tr}^t)$.
- We draw the response vector $Y \sim Ber(g(X\beta + Zr_{tr}))$ where g is the Sigmoid function.

We utilize the GPBoost library (Sigrist, 2022, 2021) to fit a mixed logistic regression model and obtain the estimators $(\hat{\beta}, \hat{\sigma}_u^2, \hat{\sigma}_s^2)$ by executing the *GPModel.fit* function over 800 iterations in the whole data set $T = (X, Z, Y)$. For each $\tilde{p} \in \{2, 6, 10\}$, we define the data set $T_{\tilde{p}} = (\mathcal{X} = X_{[0:\tilde{p}]}, Z, Y)$, subscript $[0 : \tilde{p}]$ refers to the first $\tilde{p}$ columns of X, and $T_{\tilde{p}}$ is then divided into 11 folds $\{T_k = (\mathcal{X}_k, Z_k, Y_k)\}_{k=1}^{11}$ to assess GenErr, CV, and w_{cv} as detailed below:

- The CV estimator is calculated by applying the same learning procedure as above on each one of the training folds (T_{-k}) and averaging over $L(y_i, \hat{y}(\mathcal{X}_i; \hat{\beta}_{T_{-i}}))$.
- The GenErr is evaluated by averaging over $L(y_i^{te}, \hat{y}(x_i; \hat{\beta}_{T_{-i}}))$, where y_i^{te} is the the ith index of $Y_{te}^t = (Y_{te,1}^t, \cdots, Y_{te,11}^t)$, such that $Y_{te,k}^t \sim Ber(g(X_k\beta + Z_k r_{te,k}))$, $k \in \{1, \cdots, 11\}$, where each $r_{te,k}$ is a new realization of the latent random associated with the fold k, independent of r_{tr} and the other r_{te}'s.
- To compute the estimator w_{cv}, we initially determine the matrix V globally, and (μ, D) for each fold. This process involves generating random vectors $\tilde{\mu} = X\hat{\beta} + Z\tilde{r}$ and $\tilde{\mu}_{\tilde{p}} = \mathcal{X}\hat{\beta}_{T_{-i}} + Z\tilde{r}$ multiple times where $\tilde{r}^t = (\tilde{u}^t, \tilde{s}^t)$, $\tilde{u} \sim MN(0_{q_1}, \hat{\sigma}_u^2 I_{q_1})$, and $s_{tr} \sim MN(0_{q_2}, \hat{\sigma}_s^2 I_{q_2})$, followed by computing the following empirical expectations:

$$\mu = \widehat{\mathbb{E}}[g(\tilde{\mu}_{\tilde{p}})] \;\; ; \;\; D = diag\left(\widehat{\mathbb{E}}[g'(\tilde{\mu}_{\tilde{p}})]\right)$$

$$V = \widehat{\mathbb{C}ov}\left(\tilde{\mu}\right) + diag\left(\widehat{\mathbb{E}}[g'(\tilde{\mu})]\right).$$

  We note that generally $\mathbb{V}ar(y|x, z, r) = g(x^t\beta + z^t r)(1 - g(x^t\beta + z^t r))$, but when g is the Sigmoid function, this expression simplifies to $\mathbb{V}ar(y|x, z, r) = g'(x^t\beta + z^t r)$.
- We then employ the bootstrap method, conducting $B = 200$ iterations, using this formula for each iteration $b = 1, \ldots, B$, and $k = \{1, \cdots, 11\}$:

$$\tilde{\beta}_{b,-k} = \hat{\beta} + \left[(\mathcal{X}^t D V^{-1} D \mathcal{X})^{-1} \mathcal{X}^t D V^{-1}\right]_{-k} (Y_{b,-k} - \mu_{-k}).$$

We run the simulation on the Google Colab platform using a standard CPU. The running time for a single sample (including all three models and $B = 200$ bootstrap iterations) is approximately 6 seconds.

A.2 CNN model on the CelebA DataSet

In this experiment, we employ the same CNN architecture as in Simchoni and Rosset (2021), which contains 4 convolution layers followed by 2 fully connected layers. It is presumed that for each identity $j \in \{1, \ldots, q = 100\}$ from the training sample, there is an impact from a random effect s_j distributed as $N(0, \sigma_s^2)$, and that the link function g that specifies the model is the Sigmoid function. Under this setting, the contribution of identity j to the negative log likelihood (NLL) can be written as follows:

$$\mathbb{L}_j(\gamma_c, \sigma_s^2) = -log\left\{\frac{1}{\sqrt{\pi}} \int \zeta_j(u) e^{-u^2} du\right\},$$

where,

$$\zeta_j(u) = exp\left[\sum_{l=1}^{n_j} y_{jl}\zeta_{jl}(u) - log\left(1 - e^{\zeta_{jl}(u)}\right)\right],$$

where n_j is the number of observations of identity j, and $\zeta_{jl}(u) = f_{\gamma_c}(x_{jl}) + \sqrt{2}\sigma_s u$. To approximately minimize $\mathbb{L}_j(\gamma_c, \sigma_s^2)$ with respect to (γ_c, σ_s^2), Simchoni and Rosset (2021) proposed employing Gauss-Hermite quadrature. This approach results in the following objective function:

$$\mathbb{L}_j(\gamma_c, \sigma_s^2) \approx -log\left\{\sum_{k=1}^{K_{GH}} \frac{w_k}{\sqrt{\pi}}\zeta_j(\nu_k)\right\},$$

where ν_k is the kth zero of the Hermite polynomial of degree K_{GH}, and w_k is the associated weigh. With the stated objective function formula above, we are able to train the model using stochastic gradient descent (SGD), selecting batches based on identities. We provide an implementation of the described learning procedure in the PyTorch library.

The described approach is applied on the whole training dataset, which generally includes $q = 100$ randomly chosen identities and about 2000 images in total. We perform 80 epochs, to fit the estimators $\hat{\gamma}_c$ and $\hat{\sigma}_s^2$. Initially, $\hat{\sigma}_s^2$ is set to 1, with learning rates of 10^{-4} for $\hat{\sigma}_s^2$, and 10^{-5} for network γ_c. Subsequently, the estimator $\tilde{w}_{cv}$ is evaluated using the last-layer analysis approach, similar to the procedure detailed in the Mixed Logistic Regression simulations. The CV estimator is evaluated using a standard 10-Fold cross-validation procedure, and the GenErr is evaluated folds-wise on distinct test set of random 900 identities. We run the simulation on the Google Colab platform using a Tesla T4 GPU. The duration of processing a single sample, taking into account the CV procedure, is approximately 50 minutes.

A.3 Fully connected NN on the AirBNB DataSet

In this experiment, we employ a data preprocessing procedure similar to that described in Rezazadeh Kalehbasti et al. (2021) on the Airbnb public dataset public Airbnb dataset for New York City, with certain adjustments, leading to the creation of the "*AirBNBlistings_cleaned.csv*" dataset, which is included in the supplementary material. The empirical marginal success rate of the outcome variable, "hot water", is 0.43. Each training set T comprises 100 randomly selected geographical "blocks" each measuring 100 meters, containing on average approximately 2000 listings. Any test set, on the other hand, consists of 500 randomly chosen geographical blocks.

A fully connected network, f_{γ_c}, serves as the learning model, featuring two hidden layers with ReLU activations, containing 100 and 20 neurons, respectively. It is presumed that a listing i of the training set, featuring hot water with probability $g(f_{\gamma_c}(x_i) + \delta_{T,i})$ where g is the Sigmoid function and $\delta_T \in \mathbb{R}^n$ is the vector of random effects associated with the data. Moreover, we assume that $K_{\delta_T,i_1,i_2} = \mathcal{K}(z_{i_1}, z_{i_2}; \sigma_s^2; \tau^2) = \sigma_s^2 exp\{-\tau^2||z_{i_1} - z_{i_2}||_2^2\}$, where $K_{\delta_T} \in \mathbb{R}^{n\times n}$ is the covariance matrix of δ_T. Under this setting, the negative log likelihood (NLL) can be written as follows:

$$\mathbb{L}(\gamma_c, \sigma_s^2, \tau^2) = -log\left\{\frac{1}{\pi^{n/2}}\int_{u\in\mathbb{R}^n} \zeta(u)e^{-u^t u}du\right\},$$

where,

$$\zeta(u) = exp\left[\sum_{i=1}^{n} y_i\zeta_i(u) - log\left(1 - e^{\zeta_i(u)}\right)\right],$$

where $\zeta_i(u) = f_{\gamma_c}(x_i) + \sqrt{2}\left[K_{\delta_T}^{0.5}u\right]_i$. To achieve an approximate minimization of $\mathbb{L}$ with respect to $(\gamma_c, \sigma_s^2, \tau^2)$, we introduce a Gauss-Hermite quadrature approximation performed in two dimensions for pairs of observations. By employing this method, we derive the subsequent objective function related to the observations i_1 and i_2:

$$\mathbb{L}_{i_1,i_2}(\gamma_c, \sigma_s^2, \tau^2) \approx -log\left\{\sum_{k_1,k_2} \frac{w_{k_1}w_{k_1}}{\pi}\zeta_{i_1,i_2}(\nu_k)\right\},$$

$$\zeta_{i_1,i_2}(\nu_k) = exp\left[\sum_{l=1}^{2} y_{i_l}\zeta_{i_l}(\nu_k) - log(1 - e^{\zeta_{i_l}(\nu_k)})\right]$$

where $\nu_k^t = (\nu_{k_1}, \nu_{k_2})$ is the combinations of k_1th and k_2th zeros of the Hermite polynomial, and (w_{k_1}, w_{k_1}) are the associated weighs. Moreover, $\zeta_{i_l}(\nu_k) = f_{\gamma_c}(x_{i_l}) + \sqrt{2}\left[K_{i_1,i_2}^{0.5}\nu_k\right]_l$, where:

$$K_{i_1,i_2} = \begin{pmatrix} \sigma_s^2 & \sigma_s^2 e^{-\tau^2||z_{i_1} - z_{i_2}||_2^2} \\ \sigma_s^2 e^{-\tau^2||z_{i_1} - z_{i_2}||_2^2} & \sigma_s^2 \end{pmatrix}.$$

Utilizing the given objective function formula, we can employ SGD to optimize the model with respect to the parameters $(\gamma_c, \sigma_s^2, \tau^2)$, by randomly selecting batches of size 2. To enhance the efficiency of the procedure, we adopted the approach outlined in Jäckel (2005), considering only those combinations (k_1, k_2) for which $w_{k_1}w_{k_1}$ exceeds a specified threshold value. To maintain the stability of SGD, we first perform an initial optimization of the network parameters γ_c using the typical cross-entropy loss function, before fully incorporating the entire NLL objective function. We provide an implementation of the described learning procedure in the PyTorch library.

The described approach to learning is applied to the entire training data set for 200 epochs with the typical cross-entropy loss function (with batches of size 100), and 10 epochs of pairwise full-parameter optimization. Initially, $\hat{\sigma}_s^2$ and τ^2 are set to 1 (we note that the distance unit is scaled to 1 nautical mile), with learning rates of 10^{-5} for $(\hat{\sigma}_s^2, \tau^2)$, and 10^{-4} to fit the parameters of the network γ_c. Evaluation of the estimator $\tilde{w}_{cv}$ is performed using the Hot-Start analysis technique, employing 100 standard epochs across $B = 20$ bootstrap iterations. We run the simulation on the Google Colab platform utilizing a standard CPU. The duration of processing a single sample, taking into account the CV procedure, is approximately 35 minutes.