

A Machine Learning Models as MIPs

In the following we provide an overview on how to formulate off-the-shelf *pre-trained* machine learning models as mixed-integer programs (MIP) into constraint learning formulations.

A.1 ReLU Neural Networks

We consider feed-forward neural networks composed of fully connected layers with ReLU activations, which can be exactly represented using MIP [9, 10]. Let x denote the input variables to the network and y its output variables. All other internal quantities (preactivations, activations, and indicator variables) are distinct auxiliary decision variables introduced solely for encoding the network structure. Therefore, each neuron computes a transformation of the form:

$$a = \max(0, w^\top s + b), \quad (7)$$

where s are the activations from the previous layer, and (w, b) are fixed, trained parameters. The ReLU nonlinearity is encoded via the so called *big-M formulation*:

$$\begin{aligned} a &\geq w^\top s + b \\ a &\leq w^\top s + b - (1 - \delta)L \\ a &\leq \delta U \\ \delta &\in \{0, 1\} \end{aligned} \quad (8)$$

where a is the activation output, δ is a binary indicator for whether the neuron is active, and L, U are tight lower and upper bounds on the pre-activation $w^\top s + b$. The full network is encoded layer by layer starting from the inputs x , recursively defining internal pre-activations and activations, until reaching the output layer y .

For further details and alternatives on MIP reformulations of ReLU neural networks, we refer readers to a recent review by Huchette et al. [69]. While multiple equivalent encodings exist, they yield the same global solution [53], differing only in computational efficiency, and thus do not affect our discussion on ground-truth feasibility.

A.2 Tree Ensembles and Linear-Model Decision Trees

We encode trained tree ensembles (including random forests and gradient-boosted trees) as well as linear-model decision trees using a unified MIP formulation based on leaf selection [11, 16, 17]. Let \mathcal{T} denote the set of decision trees in the ensemble, and \mathcal{L}_t the set of leaf nodes in tree $t \in \mathcal{T}$. Each tree partitions the input space into disjoint regions (leaves), each associated with a constant prediction value. We introduce binary variables $r_{t,\ell} \in \{0, 1\}$ indicating whether leaf $\ell \in \mathcal{L}_t$ of tree t is selected. A valid configuration activates exactly one leaf per tree:

$$\sum_{\ell \in \mathcal{L}_t} r_{t,\ell} = 1 \quad \forall t \in \mathcal{T} \quad (9)$$

Each internal node in a tree performs a threshold split of the form $x_i < v_{i,j}$, where $v_{i,j}$ is a threshold value for input feature x_i . We introduce binary variables $w_{i,j} \in \{0, 1\}$ to model these comparisons, where $w_{i,j} = 1$ if, and only if, the condition is satisfied. Monotonicity constraints $w_{i,j} \leq w_{i,j+1}$ ensure consistency across thresholds, reflecting the ordered structure of decision splits.

Each leaf ℓ in tree t is reachable only if all the splitting conditions on the path to that leaf are satisfied. For each split node s in tree t , let $i(s)$ be the split variable and $j(s)$ the corresponding threshold index. Let $\text{Left}_{t,s}$ and $\text{Right}_{t,s}$ denote the sets of leaf nodes in the left and right subtrees of s . We enforce:

$$\sum_{\ell \in \text{Left}_{t,s}} r_{t,\ell} \leq w_{i(s),j(s)}, \quad \sum_{\ell \in \text{Right}_{t,s}} r_{t,\ell} \leq 1 - w_{i(s),j(s)} \quad \forall t \in \mathcal{T}, s \in \text{Splits}(t) \quad (10)$$

The final prediction is computed as a weighted sum of the selected leaves: the weights are stage-dependent for gradient-boosted trees [11], uniform across trees for random forests [16], or equal to a linear function of the input features within the selected leaf for linear-model decision trees [17].

B Proof of Main Results

B.1 Mondrian Conformal Prediction

Given calibration data $\{(X_1, Y_1), \dots, (X_N, Y_N)\}$, Mondrian conformal prediction is a general framework that extends full conformal prediction by enforcing *conditional coverage guarantees* across a pre-defined set of *finite* groups defined in terms of both the features variables X and the outcome variable Y [57, 60]. Formally, for a group-indicator function $g(X, Y) \in \{1, \dots, K\}$, Mondrian conformal prediction construct conformal sets $\mathcal{C}_\alpha(X_{N+1})$ such that

$$\mathbb{P}(Y_{N+1} \in \mathcal{C}_\alpha(X_{N+1}) \mid g(X_{N+1}, Y_{N+1}) = k) \geq 1 - \alpha,$$

for all groups $k \in [K] = \{1, \dots, K\}$.

The core insight of Mondrian conformal prediction lies in computing different conformal quantiles for each group $k \in [K]$. Then, for a test point X_{N+1} , we generate *hypothetical* points (X_{N+1}, y) for each possible value of the outcome variable y , and compare the conformal score $s(X_{N+1}, y)$ to the conformal scores $\{s(X_i, Y_i)\}_{i \in \mathcal{I}_{g(X_{N+1}, y)}}$ for all calibration data points *in the same group as* (X_{N+1}, y) . That is, the group-specific comparison is performed for $i \in \mathcal{I}_{g(X_{N+1}, y)} = \{(i \in [N] : g(X_i, Y_i) = g(X_{N+1}, y))\}$.

This approach yields the following *Mondrian conformal sets*

$$\mathcal{C}_\alpha(X_{N+1}) = \{y : s(X_{N+1}, y) \leq \hat{q}_{1-\alpha}^y\}, \quad (11)$$

where $\hat{q}_{1-\alpha}^y$ is the group-specific quantile

$$\hat{q}_{1-\alpha}^y = \text{Quantile} \left(\{s(X_i, Y_i)\}_{i \in \mathcal{I}_{g(X_{N+1}, y)}}; (1 - \alpha)(1 + 1/|\mathcal{I}_{g(X_{N+1}, y)}|) \right).$$

The conditional coverage guarantee is formalized in the following theorem:

Theorem B.1 (Theorem 4.11 in [57]). *Assume that $(X_1, Y_1), \dots, (X_{N+1}, Y_{N+1})$ are exchangeable random variables and s is a symmetric score function. Then, the Mondrian conformal set $\mathcal{C}_\alpha(X_{N+1})$ from eq. (11) satisfies the conditional coverage guarantee*

$$\mathbb{P}(Y_{N+1} \in \mathcal{C}_\alpha(X_{N+1}) \mid g(X_{N+1}, Y_{N+1}) = k) \geq 1 - \alpha \quad (12)$$

for all groups $k \in \{1, \dots, K\}$ with $\mathbb{P}(g(X_{N+1}, Y_{N+1}) = k) > 0$, for any user-specified coverage level $\alpha \in (0, 1)$.

Thus, Lemma 3.1 directly applies Theorem B.1 using the group identifier function $g(x, y) = \mathbb{1}(y \in \mathcal{Y})$, which separates the data according to the ground-truth feasibility condition. Note that the condition $\mathbb{P}(g(X_{N+1}, Y_{N+1}) = k) > 0$ from Theorem B.1 assumes that both feasible and infeasible points can be sampled from the underlying distribution \mathcal{P}_{XY} , which is given by construction of the constraint learning optimization problem.

B.2 Proof of Theorem 4.1

Proof. We first show that Lemma 3.1 and Assumption 4.1 imply that we have appropriate coverage for the feasible points of the C-MICL problem $\mathcal{F}_N = \{(x, z) \in \mathcal{X} : g(x, z) \leq 0, \mathcal{C}_\alpha(x) \subseteq \mathcal{Y}\}$.

By the law of total probability,

$$\begin{aligned} \mathbb{P}(h(x) \in \mathcal{C}_\alpha(x) \mid (x, z) \in \mathcal{F}_N) &= \mathbb{P}(h(x) \in \mathcal{C}_\alpha(x) \mid (x, z) \in \mathcal{F}_N, h(x) \in \mathcal{Y}) \mathbb{P}(h(x) \in \mathcal{Y} \mid (x, z) \in \mathcal{F}_N) \\ &\quad + \mathbb{P}(h(x) \in \mathcal{C}_\alpha(x) \mid (x, z) \in \mathcal{F}_N, h(x) \notin \mathcal{Y}) \mathbb{P}(h(x) \notin \mathcal{Y} \mid (x, z) \in \mathcal{F}_N) \\ &= \mathbb{P}(h(x) \in \mathcal{C}_\alpha(x) \mid h(x) \in \mathcal{Y}) \mathbb{P}(h(x) \in \mathcal{Y} \mid (x, z) \in \mathcal{F}_N) \\ &\quad + \mathbb{P}(h(x) \in \mathcal{C}_\alpha(x) \mid h(x) \notin \mathcal{Y}) \mathbb{P}(h(x) \notin \mathcal{Y} \mid (x, z) \in \mathcal{F}_N) \\ &\geq (1 - \alpha) \mathbb{P}(h(x) \in \mathcal{Y} \mid (x, z) \in \mathcal{F}_N) + (1 - \alpha) \mathbb{P}(h(x) \notin \mathcal{Y} \mid (x, z) \in \mathcal{F}_N) \\ &= (1 - \alpha) [\mathbb{P}(h(x) \in \mathcal{Y} \mid (x, z) \in \mathcal{F}_N) + \mathbb{P}(h(x) \notin \mathcal{Y} \mid (x, z) \in \mathcal{F}_N)] \\ &= (1 - \alpha), \end{aligned}$$

where the second equality follows from the conditional independence assumption 4.1

$$\begin{aligned} \mathbb{P}(h(x) \in \mathcal{C}_\alpha(x) \mid (x, z) \in \mathcal{F}_N, h(x) \in \mathcal{Y}) &= \mathbb{P}(h(x) \in \mathcal{C}_\alpha(x) \mid h(x) \in \mathcal{Y}), \\ \mathbb{P}(h(x) \in \mathcal{C}_\alpha(x) \mid (x, z) \in \mathcal{F}_N, h(x) \notin \mathcal{Y}) &= \mathbb{P}(h(x) \in \mathcal{C}_\alpha(x) \mid h(x) \notin \mathcal{Y}). \end{aligned}$$

Moreover, the first inequality follows from the ground-truth feasibility conformal coverage guarantee in Lemma 3.1 at level α

$$\min (\mathbb{P}(h(x) \in \mathcal{C}_\alpha(x) \mid h(x) \in \mathcal{Y}), \mathbb{P}(h(x) \in \mathcal{C}_\alpha(x) \mid h(x) \notin \mathcal{Y})) \geq 1 - \alpha,$$

where the probabilities are taken with respect to both the calibration data \mathcal{D}_{cal} and the test point.

The previous result says that when a solution $(x', z') \in \mathcal{F}_N$ is feasible, its true function value $h(x')$ falls within $\mathcal{C}_\alpha(x')$ with probability at least $1 - \alpha$. Since $\mathcal{C}_\alpha(x')$ is a subset of \mathcal{Y} for feasible solutions to the C-MICL problem, whenever $h(x')$ is in $\mathcal{C}_\alpha(x')$, it must also be in \mathcal{Y} . Therefore,

$$\mathbb{P}(h(x') \in \mathcal{Y} \mid (x', z') \in \mathcal{F}_N) \geq \mathbb{P}(h(x') \in \mathcal{C}_\alpha(x') \mid (x', z') \in \mathcal{F}_N) \geq 1 - \alpha,$$

which proves the ground-truth feasibility guarantee. □

B.3 Discussion on the Conditional Independence of Feasibility and Coverage Assumption

Assumption 4.1 states that, conditional on ground-truth feasibility ($h(x) \in \mathcal{Y}$ or $h(x) \notin \mathcal{Y}$), the event of C-MICL feasibility $((x, z) \in \mathcal{F}_N = \{(x, z) \in \mathcal{X} : g(x, z) \leq 0, \mathcal{C}_\alpha(x) \subseteq \mathcal{Y}\})$ is independent of whether the conformal set contains the true function value ($h(x) \in \mathcal{C}_\alpha(x)$). To motivate Assumption 4.1 and clarify when it is plausible, we first emphasize that the ground-truth feasibility (GTF) conditional coverage guarantee from Lemma 3.1 can be achieved in a fully data-driven way (e.g., using Mondrian conformal prediction or other label-conditional conformal methods):

$$\begin{aligned} \mathbb{P}(h(x) \in \mathcal{C}_\alpha(x) \mid h(x) \in \mathcal{Y}) &\geq 1 - \alpha \\ \mathbb{P}(h(x) \in \mathcal{C}_\alpha(x) \mid h(x) \notin \mathcal{Y}) &\geq 1 - \alpha \end{aligned}$$

However, in C-MICL we aim to guarantee coverage over the feasible region of the optimization problem $\mathcal{F}_N = \{(x, z) \in \mathcal{X} : g(x, z) \leq 0, \mathcal{C}_\alpha(x) \subseteq \mathcal{Y}\}$, i.e.,

$$\mathbb{P}(h(x) \in \mathcal{C}_\alpha(x) \mid (x, z) \in \mathcal{F}_N) \geq 1 - \alpha$$

If the predictive model $\hat{h}(x)$ were perfect (i.e., $\hat{h}(x) = h(x)$), then the regions \mathcal{F}_N and the ground-truth feasible region \mathcal{F} would coincide, and the coverage guarantee would transfer directly. However, since we are interested in the more realistic case where $\hat{h}(x)$ is imperfect, \mathcal{F}_N and \mathcal{F} differ in a data-dependent way. In this case, since the feasible region \mathcal{F}_N is implicitly shaped by the calibration set (via $\mathcal{C}_\alpha(x)$), there is a natural dependency between the feasible solutions of the C-MICL problem and the calibration data, which invalidates standard conformal guarantees relying on exchangeable calibration and test data. Assumption 4.1 precisely seeks to decouple this dependency: it allows us to approximate conformal coverage within \mathcal{F}_N by assuming that feasibility does not systematically bias conformal validity, once conditioned on ground-truth feasibility.

To build intuition, consider partitioning \mathcal{F}_N into two disjoint subsets: $\mathcal{F}_N \cap \mathcal{F}$ and $\mathcal{F}_N \cap \mathcal{F}^c$. Then, Assumption 4.1 implies that $\mathcal{F}_N \cap \mathcal{F}$ (respectively $\mathcal{F}_N \cap \mathcal{F}^c$) is not systematically biased towards a region of \mathcal{F} (\mathcal{F}^c) that is miscalibrated. Mathematically,

$$\begin{aligned} \mathbb{P}(h(x) \in \mathcal{C}_\alpha(x) \mid (x, z) \in \mathcal{F}_N \cap \mathcal{F}) &= \mathbb{P}(h(x) \in \mathcal{C}_\alpha(x) \mid (x, z) \in \mathcal{F}_N, h(x) \in \mathcal{Y}) \\ &\approx \mathbb{P}(h(x) \in \mathcal{C}_\alpha(x) \mid h(x) \in \mathcal{Y}) \\ \mathbb{P}(h(x) \in \mathcal{C}_\alpha(x) \mid (x, z) \in \mathcal{F}_N \cap \mathcal{F}^c) &= \mathbb{P}(h(x) \in \mathcal{C}_\alpha(x) \mid (x, z) \in \mathcal{F}_N, h(x) \notin \mathcal{Y}) \\ &\approx \mathbb{P}(h(x) \in \mathcal{C}_\alpha(x) \mid h(x) \notin \mathcal{Y}) \end{aligned}$$

These enable us to translate the conformal coverage guarantees from the ground-truth feasible region \mathcal{F} to the feasible set \mathcal{F}_N used in the optimization. Assumption 4.1 is therefore reasonable when the calibration data adequately covers the parts of the input space that intersect the feasible region \mathcal{F}_N , both within the ground-truth feasible region \mathcal{F} and its complement \mathcal{F}^c . In this sense, it aligns with standard generalization assumptions that require the training and calibration data to be representative of the regions where predictions are deployed. In our experimental settings, we observe good empirical alignment between target and achieved coverage (Appendix E), suggesting that Assumption 4.1 holds reasonably well in practice in realistic data scenarios.

Alternatively, Assumption 4.1 can be approximated using more granular conditional conformal methods, by partitioning the optimization region into finer subregions and enforcing local coverage guarantees within each, which can then be translated to the feasible set \mathcal{F}_N . However, the assumption may break down if the feasible region \mathcal{F}_N is heavily concentrated in areas where the calibration set is sparse or systematically miscalibrated.

C Conformal Set MIP Reformulations

The reformulation of the conformal set in the regression setting is straightforward, as the conformal prediction interval is already algebraic in nature and can be directly expressed as:

$$\mathcal{C}_\alpha(x) \subseteq \mathcal{Y} \iff [\hat{h}(x) \pm \hat{q}_{1-\alpha} \cdot \hat{u}(x)] \subseteq [\underline{y}, \bar{y}] \quad (13)$$

$$\iff \begin{cases} \hat{h}(x) + \hat{q}_{1-\alpha} \cdot \hat{u}(x) \leq \bar{y}, \\ \hat{h}(x) - \hat{q}_{1-\alpha} \cdot \hat{u}(x) \geq \underline{y} \end{cases} \quad (14)$$

Here, we include only the two non-dominated constraints i.e., those constraints not already implied by the others, to avoid redundancy. Note that these are linear constraints in terms of the estimated $\hat{h}(x)$ and $\hat{u}(x)$, which are assumed to be MIP-representable, and a constant $\hat{q}_{1-\alpha}$ computed offline during the conformalization procedure.

The classification setting introduces additional modeling complexity, as the inclusion of classes in the prediction set must be encoded using mixed-integer constraints. Specifically, the condition

$$\mathbb{1}\{-\hat{h}(x)^k \leq \hat{q}_{1-\alpha}\} \quad \forall k \in \mathcal{K}, \quad (15)$$

represents an indicator that is activated (i.e., equals one) if and only if class k is included in the prediction set. To enforce correct classification behavior, we require that at least one desired class $k \in \mathcal{K}^{\text{des}}$ is predicted (i.e., its indicator is activated), and no undesired class $k \in \mathcal{K}^{\text{und}}$ is allowed in the prediction set. To model this, we introduce a binary decision variable for each class $k \in \mathcal{K}$:

$$w_k = \begin{cases} 1 & \text{if class } k \text{ is included in the prediction set.} \\ 0 & \text{otherwise.} \end{cases} \quad \forall k \in \mathcal{K} \quad (16)$$

Using these variables, the desired classification logic can be encoded via the following mixed-integer constraints:

$$\mathcal{C}_\alpha(x) \subseteq \mathcal{Y} \iff -\hat{h}(x)^k > \hat{q}_{1-\alpha} \quad \forall k \in \mathcal{K}^{\text{und}} \quad (17)$$

$$\iff \begin{cases} -\hat{h}(x)^k - \hat{q}_{1-\alpha} \leq M(1 - w_k) & \forall k \in \mathcal{K} \\ \hat{h}(x)^k + \hat{q}_{1-\alpha} + \epsilon \leq Mw_k & \forall k \in \mathcal{K} \\ \sum_{k \in \mathcal{K}^{\text{des}}} w_k \geq 1 \\ w_k = 0 & \forall k \in \mathcal{K}^{\text{und}} \end{cases} \quad (18)$$

Here, ϵ is a small numerical tolerance (e.g., 10^{-6}) to ensure strict inequality, and M is a large positive constant used in the big-M reformulation. A practical choice for M is the maximum absolute value of the predicted logits observed in the calibration dataset, scaled by a safety factor (e.g., 4):

$$M = 4 \cdot \max_i |\hat{h}(x_i)| \quad (19)$$

For detailed discussion on selecting valid big-M values and their impact on computational performance, we refer the reader to Trespalacios and Grossmann [70].

Note that the conformal prediction sets $\mathcal{C}_\alpha(x)$ used in our formulation of C-MICL have predictable, well-structured forms that are always MIP-representable, making the containment constraint $\mathcal{C}_\alpha(x) \subseteq \mathcal{Y}$ tractable regardless of the underlying prediction model.

D Implementation Details

All computations were performed on a Linux machine running Ubuntu, equipped with eight Intel®, Xeon®, Gold 6234 CPUs (3.30 GHz) and 1 TB of RAM, utilizing a total of eight hardware threads.

D.1 Regression

This case study focuses on the optimal design and operation of a membrane reactor system used for the direct aromatization of methane. This integrated unit enables the conversion of methane into hydrogen and benzene, achieving simultaneous chemical reaction and product separation. By selectively removing hydrogen through a membrane, the reactor leverages Le Chatelier’s principle to drive the equilibrium forward, resulting in higher methane conversion rates [66].

The optimization problem involves five key decision variables: the inlet volumetric flow rate of methane (v_0), the inlet flow rate of sweep gas (v_{He}), the operating temperature (T), the tube diameter (d_t), and the reactor length (L). These variables were sampled uniformly within physically reasonable bounds, as shown below:

- $v_0 \sim \mathcal{U}(450, 1500) \text{ cm}^3/\text{h}$
- $v_{\text{He}} \sim \mathcal{U}(450, 1500) \text{ cm}^3/\text{h}$
- $d_t \sim \mathcal{U}(0.5, 2.0) \text{ cm}$
- $L \sim \mathcal{U}(10, 100) \text{ cm}$
- $T \sim \mathcal{U}(997.18, 1348.12) \text{ K}$

The resulting samples were used as initial conditions for solving the system of ordinary differential equations governing the reactor, enabling the computation of the outlet benzene flow ($F_{\text{C}_6\text{H}_6}$). The numerical integration was performed using code available in the [opyrability](#) repository [71]. To simulate measurement uncertainty, Gaussian noise was added to the computed benzene flows.

Table 1 shows an illustrative subset of the 1,000 data-points generated:

v_0 (cm ³ /h)	v_{He} (cm ³ /h)	T (K)	d_t (cm)	L (cm)	$F_{\text{C}_6\text{H}_6}$ (mol/h)
1157.75	845.25	1272.93	0.57	30.67	37.45
773.89	1319.69	998.56	0.75	88.10	42.57
1484.77	476.68	1201.21	1.64	77.38	40.70
817.68	536.65	1123.28	0.72	86.07	28.26
1162.73	1262.03	1256.59	1.95	92.29	36.58

All hyperparameters used across the single-model baselines, wrapped approaches, and our proposed methods were kept consistent to ensure a fair comparison. For the the Linear-Model Decision Tree, we set the maximum depth to five, the minimum number of samples required to split an internal node to ten, and the number of bins used for discretization to forty. For the Random Forest, we used fifteen estimators, a maximum depth of five, a minimum samples split of three, and considered sixty percent of the features when looking for the best split. The Gradient Boosting Tree model was configured with fifteen estimators, a learning rate of 0.2, a maximum depth of five, a minimum of five samples per split, and sixty percent of the features considered at each split. Lastly, the ReLU Neural Network was set up with two hidden layers of 32 units each, an L2 regularization strength of 0.01, and trained for 2000 epochs using Adam optimizer.

All models were cross-validated using a 5-fold split of their respective datasets. For the single base models, the complete 1,000 point dataset was used for training and evaluation. In the case of the ensemble methods, bootstrapping the 1,000 data-point dataset was employed to generate the required 500 data-point (half sized) subsets for training following [7]. For our conformal method, a split of the training data (800 data points) was used, ensuring that the calibration data was kept separate (200 data points). The Table 2 presents the average Mean Squared Error (MSE) across all folds for each of the methods, considering that the variance of the output for the 1,000 data-points is 1.2871.

Table 2: Average MSE across folds for all Methods for reactor model.

Predictive Method		
Model	Approach	MSE
RandomForest	MICL	0.1748
ReLU NN	MICL	0.0597
GradientBoosting	MICL	0.1192
LinearModelDT	MICL	0.0634
RandomForest	W-MICL(5)	0.1712
ReLU NN	W-MICL(5)	0.070
GradientBoosting	W-MICL(5)	0.1298
LinearModelDT	W-MICL(5)	0.0594
RandomForest	W-MICL(10)	0.1711
ReLU NN	W-MICL(10)	0.0699
GradientBoosting	W-MICL(10)	0.1239
LinearModelDT	W-MICL(10)	0.0568
RandomForest	W-MICL(25)	0.1696
ReLU NN	W-MICL(25)	0.0689
GradientBoosting	W-MICL(25)	0.1217
LinearModelDT	W-MICL(25)	0.0568
RandomForest	W-MICL(50)	0.1665
ReLU NN	W-MICL(50)	0.0685
GradientBoosting	W-MICL(50)	0.1194
LinearModelDT	W-MICL(50)	0.0559
RandomForest	C-MICL	0.1847
ReLU NN	C-MICL	0.0841
GradientBoosting	C-MICL	0.1402
LinearModelDT	C-MICL	0.0696

For all uncertainty models $\hat{u}(x)$, we employed ReLU-based Neural Networks with a shared architecture and regularization setup to maintain consistency across approaches. Specifically, each model used two hidden layers with 32 units each and an L_2 regularization (weight decay) coefficient of 0.001. The only hyperparameter that varied during cross-validation was the number of training epochs, which does not influence the MIP optimization outcomes but can affect the quality of the uncertainty estimates. The optimal number of training epochs identified through cross-validation were as follows: 1000 epochs for the model paired with Gradient Boosting, 3000 epochs for the one used with the ReLU Neural Network (using Adam optimizer), and 2000 epochs for both the Random Forest and Linear-Model Decision Tree variants. Table 3 displays the average MSE for the uncertainty model of each base predictor across all folds.

Table 3: Uncertainty average MSE for each base model.

Base Model	$\hat{u}(x)$ MSE
GradientBoosting	0.0142
ReLU NN	0.0247
RandomForest	0.0408
LinearModelDT	0.0225

Tables 4, 5, and 6 present the sets, parameters and decision variables of the problem, respectively.

Table 4: Sets used in the reactor optimization model.

Symbol	Description
\mathcal{I}	Set of decision variables (e.g., $\mathcal{I} = \{v_0, v_{\text{He}}, T, dt, L\}$)

Table 5: Parameters used in the reactor optimization model.

Symbol	Description
c_i	Operational or design cost coefficient associated with variable $i \in \mathcal{I}$

Table 6: Decision variables in the reactor design model.

Symbol	Description
x_i	Value of design variable $i \in \mathcal{I}$, where $\mathcal{I} = \{v0, v_{\text{He}}, T, dt, L\}$
y	Predicted outlet flow of C_6H_6 from the surrogate model

Formulation.

$$\min_{\{x_i\}_{i \in \mathcal{I}}, y} \sum_{i \in \mathcal{I}} c_i x_i \quad (\text{D.1a})$$

$$\text{s.t. } 10 \leq \frac{x_L}{x_{dt}} \leq 150 \quad (\text{D.1b})$$

$$0.75 \leq \frac{x_{v0}}{x_{v_{\text{He}}}} \leq 3.0 \quad (\text{D.1c})$$

$$20 \leq \frac{x_{v0}}{x_L} \leq 120 \quad (\text{D.1d})$$

$$x_{v0} \leq 1.1 \cdot x_T \quad (\text{D.1e})$$

$$\hat{h}(\mathbf{x}) = y \quad (\text{D.1f})$$

$$y \geq 50 \quad (\text{D.1g})$$

$$x_i \geq 0 \quad \forall i \in \mathcal{I} \quad (\text{D.1h})$$

Explanation of Constraints.

- **(D.1a)**: Minimize the operating cost as a linear function of the decision variables.
- **(D.1b)**: Enforce physical bounds on the tube length-to-diameter ratio.
- **(D.1c)**: Maintain a suitable gas feed ratio between CH_4 and He.
- **(D.1d)**: Control the residence time via the CH_4 flow and reactor length.
- **(D.1e)**: Limits methane flow rate based on temperature to ensure safe and stable operation.
- **(D.1f)**: Enforces that the surrogate model output y , predicting C_6H_6 flow, is computed from the design and operation variables \mathbf{x} .
- **(D.1g)**: Requires that the predicted C_6H_6 outlet flow (i.e., product quality) meets or exceeds the target value of 50.
- **(D.1h)**: Enforces nonnegativity for all design variables.

D.2 Classification

We now examine the food basket optimization problem introduced earlier and grounded in the work of Peters et al. [68]. Our analysis is based on the model developed by Fajemisin et al. [1], which aims to minimize the cost of assembling a basket consisting of 25 different commodities while satisfying nutritional requirements across 12 key nutrients. An additional constraint is placed on the palatability of the basket, which reflects how acceptable or appealing the food is to the target population. In line with the case study by Maragno et al. [7], we require a minimum palatability score of $t = 0.5$ to ensure that the resulting food baskets are not only affordable and nutritionally adequate but also culturally and socially acceptable. The dataset used in this analysis is publicly available at and published by Maragno et al. [7] [here](#).

The palatability scores initially range continuously from 0 to 1. To transform this into a classification problem, we discretize these scores into four distinct categories: *bad*, *regular*, *good*, and *very good*. This discretization is achieved by setting thresholds at 0.25, 0.5, and 0.75. Consequently, the problem

becomes a multi-class classification task, where each food basket is assigned one of these categorical labels based on its palatability score. We restrict the optimization problem to include only food baskets that fall within the *good* and *very good* categories. This approach is inspired by the work of Maragno et al. [7], where a similar threshold of $t = 0.5$ was used, but applied in a categorical context. Table 7 displays a sample of the dataset available.

Table 7: Sample of food basket dataset with commodity quantities and palatability score. Full dataset available [here](#).

Beans	Bulgur	Cheese	Fish	Meat	CSB	Dates	DSM	...	Palatability	Class
0.7226	0	0	0	0	0	0	0.5987	...	0.199	Bad
0.7860	0	0	0	0	0.0419	0	0.3705	...	0.8049	Very Good
0.4856	0	0	0	0	0	0	0.2696	...	0.6517	Good
0	0	0.5734	0	0	0	0	0.0025	...	0.3220	Regular

In this case, we exclusively trained ReLU-based neural networks across all methods (single-model baseline, wrapped approach, and our proposed method) ensuring that all hyperparameters remained identical for a fair comparison. For the predictors, the network architecture consisted of three hidden layers with 64 units each, trained for 500 epochs with a weight decay (L2 regularization) of 0.01. Additionally, we trained an oracle model using all available data. This oracle network was configured with five hidden layers of 256 units each, trained for 1000 epochs, also with a weight decay of 0.01.

Table 8 displays the average accuracy across validation folds on the data available for each approach.

Table 8: Average accuracy across folds for all methods for the basket model.

Predictive method		
Model	Approach	Accuracy [%]
ReLU NN	MICL	84.32
ReLU NN	W-MICL(5)	78.24
ReLU NN	W-MICL(10)	76.88
ReLU NN	C-MICL	83.30

Tables 9, 10, and 11 present the sets, parameters and decision variables of the problem, respectively.

Table 9: Sets used in the basket model.

Symbol	Description
\mathcal{M}	Set of commodities (e.g., $\mathcal{M} = \{\text{rice, beans, salt, sugar, ...}\}$)
\mathcal{L}	Set of nutrients (e.g., $\mathcal{L} = \{\text{protein, iron, calories, ...}\}$)
\mathcal{K}^{des}	Set of desired categories (e.g., $\mathcal{K}^{\text{des}} = \{\text{good, very good}\}$)
\mathcal{K}^{und}	Set of undesired categories (e.g., $\mathcal{K}^{\text{und}} = \{\text{bad, regular}\}$)

Table 10: Parameters used in the basket model.

Symbol	Description
Nutreq_l	Nutritional requirement for nutrient $l \in \mathcal{L}$ (grams/person/day)
Nutval_{ml}	Nutrient content of commodity m for nutrient l (grams per gram)
p_m	Procurement cost of commodity m (in \$/metric ton)
M	Large positive constant for big-M reformulation.

Table 11: Decision variables in the basket model.

Symbol	Description
x_m	Quantity of commodity $m \in \mathcal{M}$ in the food basket (grams)
y	Palatability level of the food basket
w_k	Binary indicator for predicting desired category $k \in \mathcal{K}^{\text{des}}$ ($\{0,1\}$)

Formulation.

$$\min_{\{x_m\}_{m \in \mathcal{M}}, y} \sum_{m \in \mathcal{M}} p_m x_m \quad (\text{D.2a})$$

$$\text{s.t.} \quad \sum_{m \in \mathcal{M}} \text{Nutval}_{ml} \cdot x_m \geq \text{Nutreq}_l, \quad \forall l \in \mathcal{L} \quad (\text{D.2b})$$

$$x_{\text{salt}} = 5 \quad (\text{D.2c})$$

$$x_{\text{sugar}} = 20 \quad (\text{D.2d})$$

$$\hat{h}(\mathbf{x}) = y \quad (\text{D.2e})$$

$$y_{k'} - y_k \leq M(1 - w_k) \quad \forall k \in \mathcal{K}^{\text{des}}, k' \in \mathcal{K}^{\text{und}} \quad (\text{D.2f})$$

$$\sum_{k \in \mathcal{K}^{\text{des}}} w_k \geq 1 \quad (\text{D.2g})$$

$$x_m \geq 0 \quad \forall m \in \mathcal{M} \quad (\text{D.2h})$$

Explanation of Constraints.

- **(D.2a)**: Objective function minimizing total procurement cost of the food basket.
- **(D.2b)**: Nutritional constraints to ensure daily nutrient requirements are met.
- **(D.2c)–(D.2d)**: Fixed amounts of salt and sugar imposed (e.g., due to guidelines).
- **(D.2e)**: Palatability is computed as a function $\hat{h}(\mathbf{x})$ of the selected commodity quantities.
- **(D.2f)**: Big-M constraint that activates an indicator if the logit of a desired class $k \in \mathcal{K}^{\text{des}}$ is higher than the logit of all undesired classes $k' \in \mathcal{K}^{\text{und}}$, representing desired prediction. M is calculated using the largest value in magnitude observed in calibration data times a enlarging safety factor (e.g., 4) as $M = 4 \cdot \max_i |\hat{h}(x_i)|$. For more information on how to calculate valid M values and their impacts in optimization we refer the readers to Trespalacios and Grossmann [70].
- **(D.2g)**: Enforce at least one desired class to have larger logit than both undesired classes.
- **(D.2h)**: Ensures quantity nonnegativity.

E Complementary Results

To quantify uncertainty and support the statistical reliability of our results, we report 95% confidence intervals (CIs) for all key metrics: empirical feasibility rates, optimization times, relative differences in objective value, and true coverage. All uncertainty comes from 100 problem instances of optimization problems.

For feasibility rates and true coverage plots intervals are computed as the proportion of feasible solutions over meaning that we model each as a Bernoulli random variable and apply the standard error formula for proportions:

$$\text{SEM} = \sqrt{\frac{\bar{p}(1 - \bar{p})}{n}},$$

where \bar{p} is the sample mean and n is the number of problem instances. The corresponding confidence interval is calculated using the Student's t-distribution:

$$\text{CI}_{95\%} = \bar{p} \pm t_{n-1, 0.975} \cdot \text{SEM}$$

For optimization times and relative differences in objective values, we compute the sample mean, standard deviation, and standard error of the mean (SEM) for each method across instances. The 95% confidence intervals are then given by:

$$\text{CI}_{95\%} = \bar{x} \pm t_{n-1, 0.975} \cdot \frac{s}{\sqrt{n}}$$

where s is the sample standard deviation and $n = 100$ is the number of observations. This method assumes approximate normality of the sample means, which is reasonable due to the Central Limit Theorem given the sample size.

All error bars shown in plots correspond to these 95% confidence intervals. The factors of variability captured are due to the random generation of optimization instances and the resulting performance metrics across different methods. These intervals are reported directly in the results section and supporting figures to substantiate claims about statistical significance and performance differences.

E.1 Regression

The following results summarize performance on the reactor case study at $\alpha = 10\%$, evaluated after solving 100 optimization instances for each method. More specifically, Figure 7 reports the average computational time required to obtain an optimal solution for each method at $\alpha = 10\%$, while Figure 8 presents the relative difference in optimal objective value between baseline methods and our proposed C-MICL at the same confidence level. Lighter bars in both figures denote methods that failed to achieve the target empirical feasibility rate, as established in Figure 1. Additionally, Figure 9 shows the distribution of true constraint values $h(x)$ for each methods for $\alpha = 10\%$. Finally, Figure 10 reports the empirical coverage over 1,000 out-of-sample data points, stratified by deciles of the true output variable y . For each decile, we report the proportion of instances where the true value lies within the predicted interval, demonstrating strong coverage across the output space for all baseline models. Notably, our ReLU NN-based conformal sets empirically satisfy the ground-truth feasibility coverage guarantee from Lemma 3.1, achieving coverage rates of 89.31% and 95.38% on ground-truth infeasible and feasible samples, respectively.

We report here the full results for the reactor case study at $\alpha = 5\%$, based on evaluations conducted over 100 solved optimization instances per method. Specifically, Figure 11 reports the empirical ground-truth feasibility rate achieved by each MICL approach. Figure 12 shows the average computational time needed to solve each method at $\alpha = 5\%$, and Figure 13 displays the relative difference in optimal objective value between baseline methods and our C-MICL approach. Furthermore, Figure 14 presents the distribution of true constraint values $h(x)$ for each method at $\alpha = 5\%$. Finally, Figure 15 reports the empirical coverage of the models over 1,000 out-of-sample data points, grouped by deciles of the true output y . Coverage is measured as the proportion of points for which the true value of y falls within the predicted interval at a target level of $\alpha = 5\%$. Our ReLU NN-based conformal sets demonstrate empirical compliance with the ground-truth feasibility coverage guarantee established in Lemma 3.1, yielding coverage rates of 96.92% for ground-truth feasible instances and 96.09% for ground-truth infeasible ones.

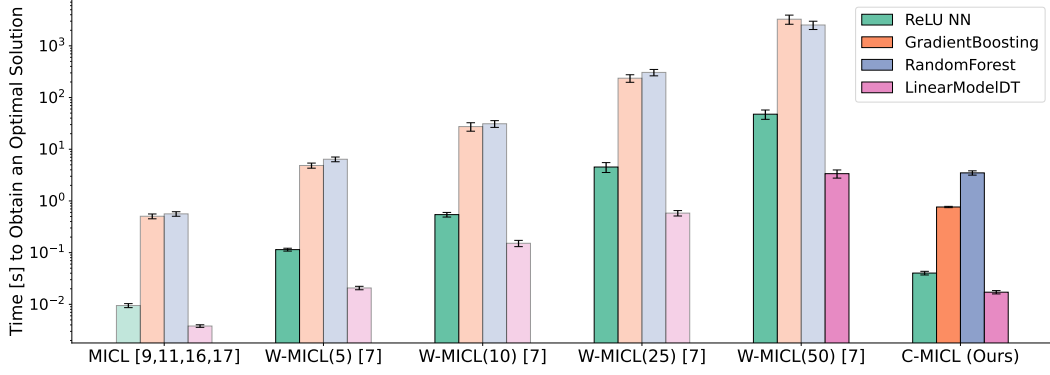


Figure 7: Average computational time to obtain an optimal solution for all methods at $\alpha = 10\%$. Our proposed C-M1CL approach is comparable to single-model baselines while being orders of magnitude faster than ensemble heuristics using the same underlying base model. Lighter bars indicate methods that failed to achieve the target empirical feasibility rate.

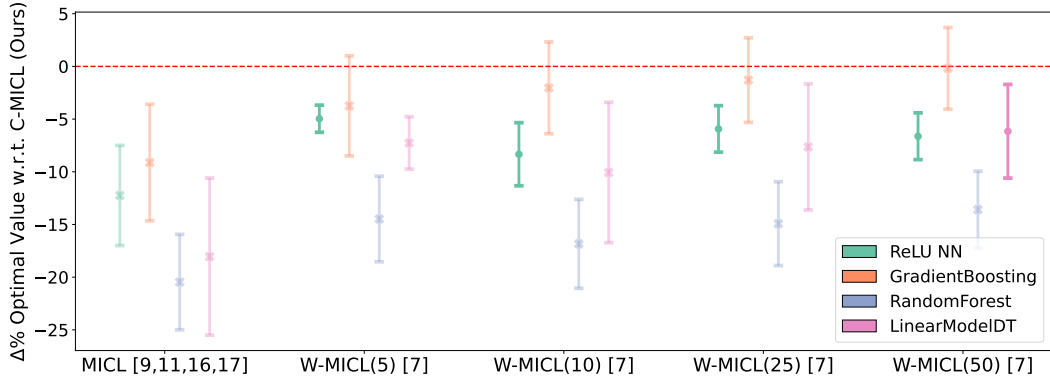


Figure 8: Relative difference in optimal objective value of baseline methods compared to our proposed C-M1CL at $\alpha = 10\%$. Our approach exhibits a small difference of around 8% compared to the method that achieved empirical coverage, indicating that C-M1CL attains comparable solution quality relative to valid approaches. Lighter bars marked with an “x” denote methods that failed to meet the target empirical feasibility rate.

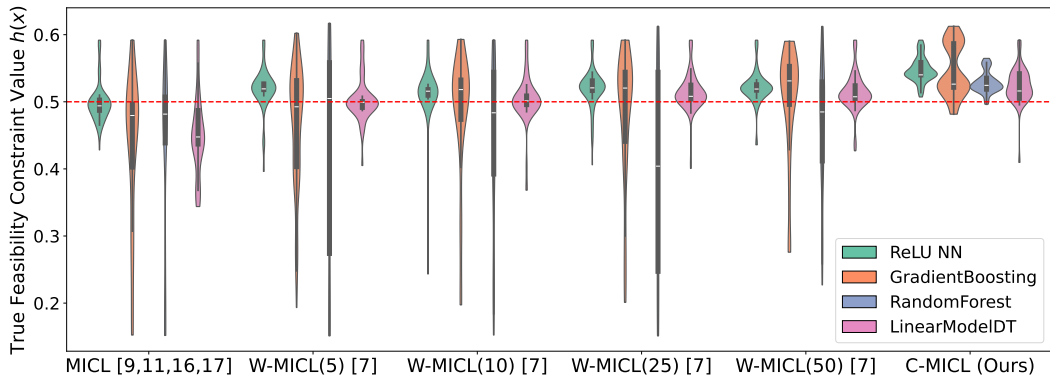


Figure 9: Distribution of true constraint values $h(x)$ for each method for $\alpha = 10\%$. The dotted line at 0.5 marks the lower bound imposed in the oracle constraint values below this line correspond to true infeasibilities. C-M1CL not only yields the fewest violations but also achieves the smallest violation magnitudes when they occur.

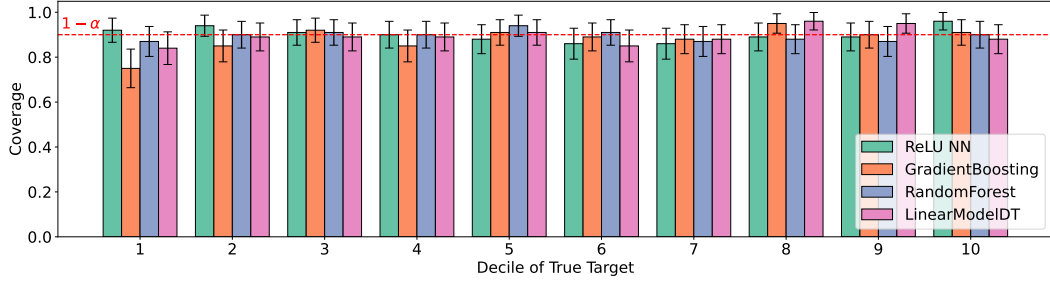


Figure 10: Empirical out-of-sample coverage across predictive models at $\alpha = 10\%$, evaluated on 1,000 test points and stratified by deciles of the true output y . Our conformal sets achieve strong target coverage. Crucially, the ground-truth feasibility threshold ($y \geq 50\%$) lies within the 9th and 10th deciles, where all methods attain valid empirical coverage, empirically supporting the assumptions of Theorem 4.1.

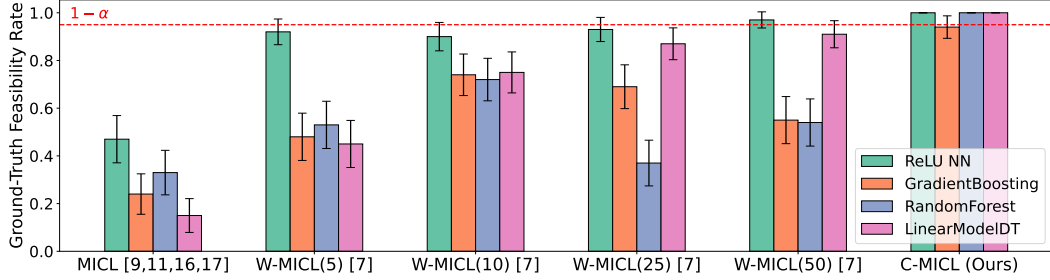


Figure 11: Empirical ground-truth feasibility rate of optimal solutions across MICL methods on 100 optimization problem instances for $\alpha = 5\%$. Our Conformal MICL approach (rightmost bars) consistently meets the target feasibility rate of $\geq 95\%$ regardless of the underlying base model. In contrast, existing methods show inconsistent performance and often fall short of the theoretical guarantee.

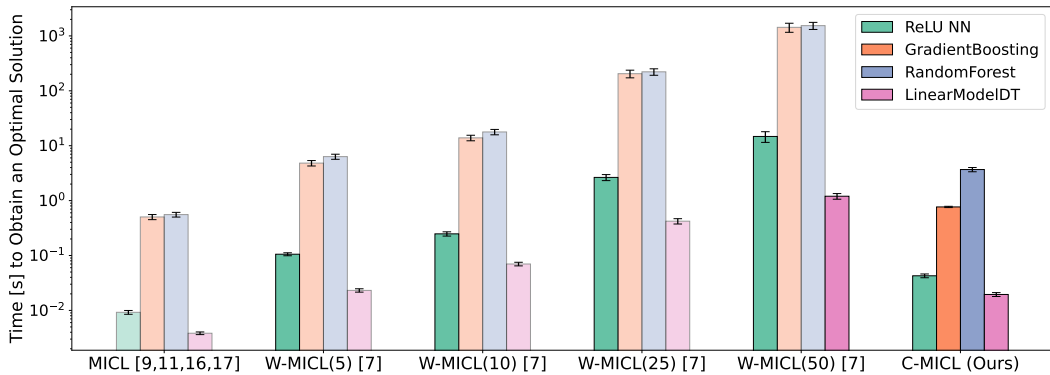


Figure 12: Average computational time to solve 100 optimization instances at $\alpha = 5\%$. Our proposed C-MICL method matches the speed of single-model baselines and is orders of magnitude faster than ensemble-based heuristics using the same base model. Lighter bars denote methods that did not reach the target empirical feasibility threshold.

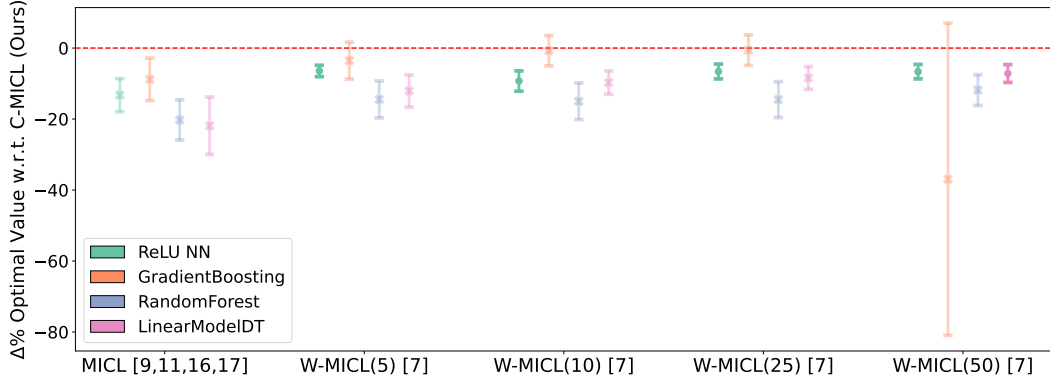


Figure 13: Relative difference in optimal objective value between baseline methods and our proposed C-MICL at $\alpha = 5\%$ across 100 optimization problems. Our approach shows a modest 8% difference compared to the method that achieved empirical coverage, demonstrating that C-MICL provides solution quality on par with other valid approaches. Lighter bars with an “x” indicate methods that did not meet the empirical feasibility target. Five outliers were removed for the W-MICL(50) Gradient Boosted Tree model, though the comparison remains statistically insignificant with or without them.

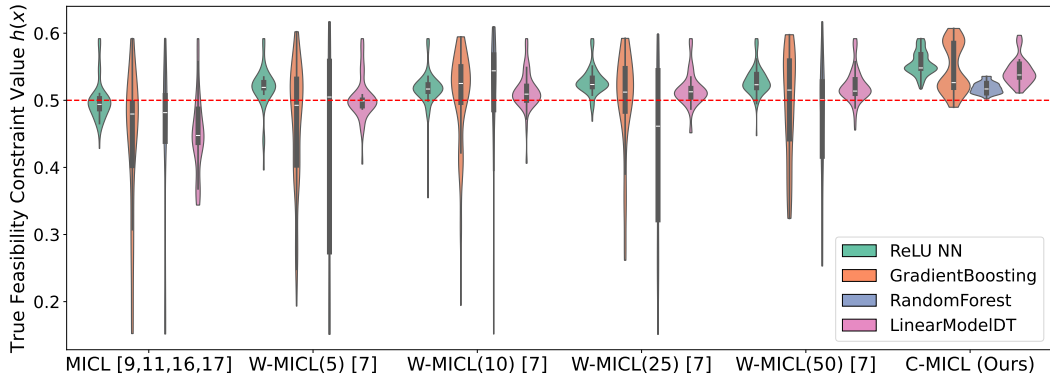


Figure 14: Violin plots of the true constraint values $h(x)$ for each method at $\alpha = 5\%$. The dotted line at 0.5 denotes the lower bound enforced in the oracle constraint where values falling below this line indicate true infeasibilities. At this stricter confidence level, C-MICL maintains the highest reliability, producing the fewest and smallest violations across all methods.

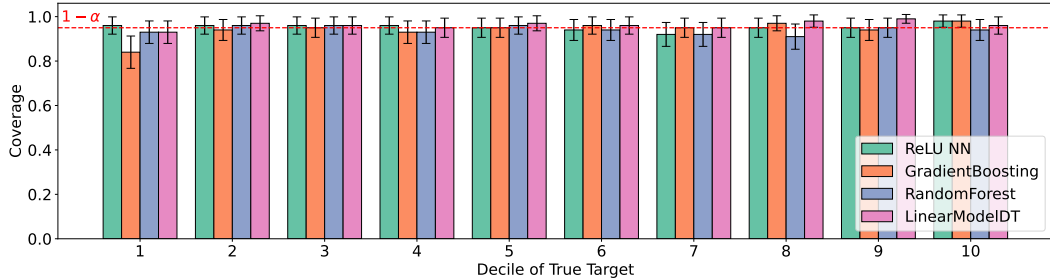


Figure 15: Empirical out-of-sample coverage at $\alpha = 5\%$ for all predictive models, evaluated on 1,000 test points and stratified by deciles of the true output y . Our conformal prediction sets achieve the desired coverage level across deciles. Notably, the ground-truth feasibility threshold ($y \geq 50$) lies in the 9th and 10th deciles, where all methods achieve valid empirical coverage, providing empirical support for the assumptions in Theorem 4.1.

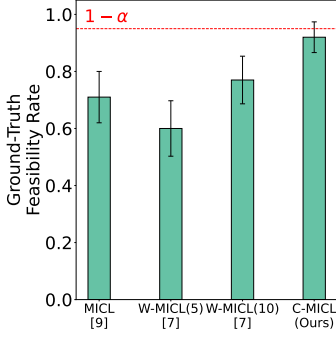


Figure 16: Empirical ground-truth feasibility rates across MICL methods on 100 optimization problem instances at $\alpha = 5\%$. C-MICL (rightmost bar) is the only method that consistently achieves the target feasibility threshold of at least 95%, in line with theoretical guarantees. All baseline methods fall short of this benchmark.

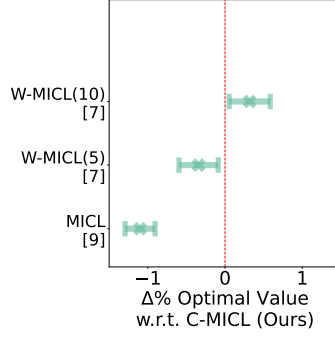


Figure 17: Relative difference in optimal objective value between baseline MICL methods and C-MICL across 100 optimization problem instances at $\alpha = 5\%$. The differences average about 1%, with all methods achieving statistically similar solution quality. However, none of the baselines produce implementable solutions, as indicated by the lighter bars, which represent methods that failed to meet the required empirical coverage guarantee. Only our method satisfies this guarantee.

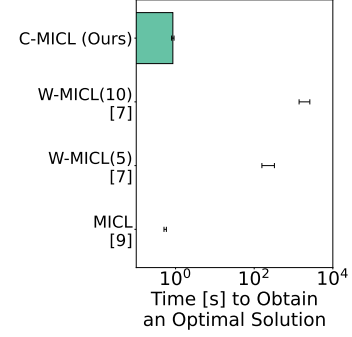


Figure 18: Average time to compute an optimal solution for each MICL method on 100 optimization instances at $\alpha = 5\%$. C-MICL matches the runtime of single-model MICL and outperforms ensemble-based methods by a wide margin. Lighter bars represent approaches that did not achieve the target feasibility level.

E.2 Classification

The following three figures present the performance of different MICL methods applied to the food basket design problem under a classification setting, with $\alpha = 5\%$. These figures assess the feasibility, objective value, and computational efficiency of the methods across 100 problem instances, highlighting the strengths of C-MICL in comparison to baseline approaches. Specifically, Figure 16 illustrates the empirical ground-truth feasibility rate, Figure 17 shows the relative difference in optimal objective value, and Figure 18 reports the average computational time required to obtain an optimal solution. Finally, Figures 19 and 20 present the empirical coverage achieved by each model across 1,000 out-of-sample points, grouped by the label y category, for $\alpha = 10\%$ and $\alpha = 5\%$, respectively. These figures report the proportion of instances in which the true value of y falls within the predicted conformal set. Our conformal sets empirically satisfy the ground-truth feasibility coverage guarantee from Lemma 3.1, achieving coverage rates of 88.22% (feasible) and 96.19% (infeasible) for $\alpha = 10\%$, and 93.41% (feasible) and 98.79% (infeasible) for $\alpha = 5\%$.

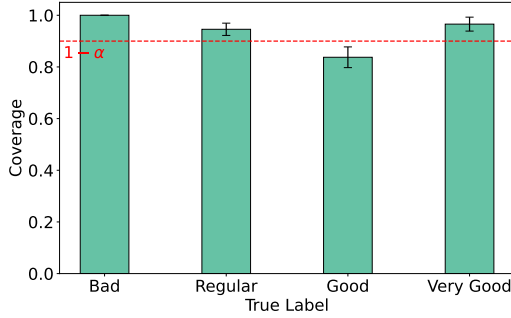


Figure 19: Empirical out-of-sample coverage, evaluated over 1,000 test points and stratified by the true categorical label of y at $\alpha = 10\%$. Across all categories, the predicted conformal sets capture the true label at the desired coverage level. Notably, for the "Good" and "Very Good" categories, (which correspond to the ground-truth feasibility region) the models approximately achieve valid empirical coverage, lending support to the assumptions in Theorem 4.1.

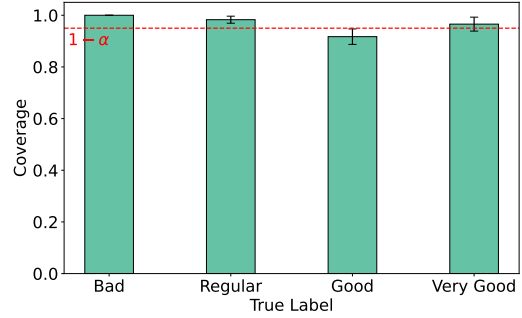


Figure 20: Evaluation of empirical coverage with ReLU NNs over 1,000 test samples, grouped by the true y label, for a target level of at $\alpha = 5\%$. The model produces conformal prediction sets that contain the true class label at the expected rate. In particular, the "Good" and "Very Good" categories, those relevant to enforcing feasibility, demonstrate approximate empirical coverage, providing evidence consistent with Theorem 4.1.