Differential Privacy for Machine Learning
Accuracy, Interpretability, and Differential Privacy via Explainable Boosting

Anonymous Authors[1]

1. Introduction

This paper explores the integration of differential privacy into Explainable Boosting Machines (EBMs). This type of model can be useful in the context of human audits and/or error corrections while working on sensitive data. In such contexts, both explainability and privacy are required. The authors propose DP-EBM, a simple adaptation of the classical EBM models for differential privacy. They demonstrate that this variation of EBM achieves state-of-the-art performances on both classification and regression on several datasets. The authors also provide theoretical guarantees on privacy under the Gaussian DP model. In this report, we will present the global framework, explain the contributions of the paper, analyze their results, and finally explain what we have implemented and how we contributed.

2. Foundations of Differential Privacy

This section presents definitions and theorems related to differential privacy.

Definition 2.1 ((ϵ, δ)-Differential Privacy (Dwork et al., 2014))**.** Let $\mathcal{M} : D \to \mathcal{R}$ be a randomized mechanism. $\mathcal{M}$ guarantees (ϵ, δ)-differential privacy if for any two neighboring databases $d, d' \in D$ differing in only one row and for any subset of outputs $S \subseteq \mathcal{R}$, it holds that

$$\Pr[\mathcal{M}(d) \in S] \leq \exp(\epsilon) \times \Pr[\mathcal{M}(d') \in S] + \delta. \quad (1)$$

The privacy parameters $\epsilon \in (0, \infty)$ and $\delta \in [0, 1]$ represent the level of privacy protection. When $\delta = 0$ and the mechanism $\mathcal{M}$ satisfies Definition 1, $\mathcal{M}$ is ϵ-DP. Smaller ϵ values provide stronger privacy protection. δ is the probability that $\mathcal{M}$ fails to provide ϵ-DP. Thus, (ϵ, δ)-DP is a relaxed version of ϵ-DP. The Gaussian mechanism adds Gaussian random noise to the result, can be applied to a query function to guarantee (ϵ, δ)-DP.

When applying the sequential composition of multiple (ϵ_i, δ_i)-DP mechanisms, the total privacy cost spent can be calculated as follows:

Theorem 2.2 (Composition Theorem for (ϵ_i, δ_i)-DP)**.** *Let $\mathcal{M} = (\mathcal{M}_1, \ldots, \mathcal{M}_n)$ be a sequential composition of mechanisms, with each $\mathcal{M}_i$ being (ϵ_i, δ_i)-DP. Then the mechanism $\mathcal{M}$ guarantees $(\sum_i \epsilon_i, \sum_i \delta_i)$-differential privacy.*

However, this theorem does not provide a tight bound on ϵ. To address this issue, J. Dong et al. proposed Gaussian Differential Privacy (GDP) (Dong et al., 2022).

Theorem 2.3 (Single Gaussian Mechanism as GDP)**.** *Let $\theta : \mathcal{D} \to \mathbb{R}$ be a statistic with sensitivity Δ, that is, for any two neighboring datasets D and D',*

$$\Delta_2 = \max_{D, D'} \|\theta(D) - \theta(D')\|_2,$$

where $\| \cdot \|_2$ denotes the l_2 norm and neighboring datasets differ in a single record.

Define the Gaussian mechanism M as

$$M(D) = \theta(D) + \xi, \quad \text{where } \xi \sim \mathcal{N}\Big(0, \frac{\Delta^2}{\mu^2}\Big).$$

Then, M is μ-GDP.

In GDP, the sensitivity Δ measures the maximum impact that a single data point can have on the statistic $\theta(D)$.

For instance, in a counting query, modifying one record changes the count by at most 1, which means the sensitivity is 1. Similarly, in a histogram, which can be seen as a vector, a single record affects only one bin by 1. This results in an l_2-sensitivity of 1. These two examples play essential roles in the DP-EBM Algorithm.

By adding Gaussian noise with variance Δ^2/μ^2, the mechanism perturbs all data points. A smaller value of μ (which means adding more noise) provides a stronger privacy guarantee. Therefore, the theorem indicates that by choosing an appropriate noise level, the mechanism M satisfies μ-GDP.

Theorem 2.4 (Composition of GDP Mechanisms)**.** *Suppose that $M_1, M_2, \ldots, M_k$ are mechanisms, where M_i is μ_i-GDP. Then, the composition*

$$M(D) = \big(M_1(D), M_2(D), \ldots, M_k(D)\big)$$

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

is $\left(\sqrt{\mu_1^2 + \mu_2^2 + \cdots + \mu_k^2}\right)$*-GDP.*

When multiple outputs are released (for example, counts in a histogram), each protected by a GDP mechanism, each mechanism incurs a privacy loss quantified by μ_i. Instead of summing these losses linearly, as done in basic composition, the GDP framework demonstrates that the overall privacy loss is determined by the square root of the sum of the squares of the individual μ_i values. This provides a tighter privacy guarantee.

Theorem 2.5 (Conversion from GDP to (ϵ, δ)-DP)**.** *A mechanism is μ-GDP if and only if it is (ϵ, δ)-DP with*

$$\delta = \Phi\left(-\frac{\epsilon}{\mu} + \frac{\mu}{2}\right) - e^{\epsilon}\,\Phi\left(-\frac{\epsilon}{\mu} - \frac{\mu}{2}\right),$$

where Φ denotes the cumulative distribution function (CDF) of the standard normal distribution.

This shows that a mechanism with a given μ-GDP guarantee satisfies a specific (ϵ, δ)-DP guarantee, and vice versa.

3. Explainable Boosting Machines

Glass-box models, such as linear regression and decision trees, provide clear interpretations of their predictions. However, these models often sacrifice accuracy for interpretability. Explainable Boosting Machines (Nori et al., 2019) overcome this trade-off by integrating Generalized Additive Models (GAM) with Gradient Boosting Decision Trees (GBDT).

EBM adopts the form of GAM as follows:

$$g(\mathbb{E}[Y]) = \beta + f_1(x_{i1}) + f_2(x_{i2}) + \cdots + f_k(x_{ik}),$$

where Y denotes the response variable, x_{ik} represents the k-th feature of the i-th data point, β is an intercept term, f_k is a shape function that captures the contribution of the k-th feature, and g is a link function that depends on the type of task (e.g., identity link for regression or logit link for classification).

Given a dataset $X = \{x_1, \ldots, x_N\}$ with K features and corresponding labels $Y = \{y_1, \ldots, y_N\}$, the prediction for an example i can be represented as:

$$\hat{y_i} = g^{-1}\left(\beta + \sum_{k=1}^{K} f_k(x_{ik})\right).$$

The goal of EBM is to find the shape functions f_k that minimize a specified loss function $L(y, \hat{y})$, ensuring both accuracy and interpretability.

The EBM Algorithm(1) uses gradient boosting to iteratively update these feature functions f_k. At each boosting round, the model focuses on reducing the current prediction errors, represented by pseudo residuals. The choice of the loss function and the corresponding pseudo residuals depend on the task. For regression tasks, the squared loss is used, defined as:

$$L(y, \hat{y}) = \frac{1}{2}(y - \hat{y})^2,$$

with pseudo residuals computed as $r_i = y_i - \hat{y}_i$.

A critical aspect of the EBM algorithm is the selection of the best splits for each feature during the training process. Each feature is divided into bins, and the algorithm evaluates different groupings of these bins (splits) based on their ability to minimize the loss.(See Fig.(1)) For regression, the optimal split is the one that minimizes the squared loss. For a candidate split S, the optimal update μ_S^* is calculated as the average of the pseudo residuals for regression:

$$\mu_S^* = \frac{\sum_{i \in \mathcal{I}(S)} r_i}{|\mathcal{I}(S)|},$$

By iteratively selecting the best splits and using cyclic gradient boosting, EBM updates each feature's contribution sequentially in a round-robin manner. In this approach, each decision tree is restricted to a single feature at a time to enforce additivity and prevent interactions between features from being learned. The process begins by training a shallow decision tree on the first feature to minimize the initial residuals. The algorithm then proceeds to the next feature, fitting a new tree to the residuals left by the previous model. This cycle repeats across all features for epochs, gradually refining the shape functions to reduce the overall loss and improve the model's accuracy.

4. DP-EBM

The algorithm for DP-EBM is outlined in Algorithm 2, featuring two differentially private phases: histogram construction and tree construction. Before constructing these, the privacy budget for these phases is pre-calculated. Given a privacy budget (ϵ, δ) and allocation ratio $\tau \in (0, 1)$, the budget is allocated as $((1-\tau)\epsilon, \delta/2)$ for histogram construction and $(\tau\epsilon, \delta/2)$ for tree construction, in accordance with Theorem 2.2. These budgets are then converted into GDP format using Theorem 2.5. We denote μ_0 and μ_1 as the converted privacy budgets from $((1-\tau)\epsilon, \delta/2)$ and $(\tau\epsilon, \delta/2)$, respectively. According to Theorem 2.4, the privacy budgets for each histogram and tree instance are:

$$\mu_{\mathcal{H}} = \frac{\mu_0}{\sqrt{K}}$$

$$\mu_{\mathcal{T}} = \frac{\mu_1}{\sqrt{E \cdot K}}$$

where $\mu_{\mathcal{H}}, \mu_{\mathcal{T}}$ represent the privacy budgets for a histogram

Algorithm 1 Explainable Boosting

Input: $X \in \mathbb{R}^{n \times K}$, $y \in \mathbb{R}^n$, $E \in \mathbb{N}$, $\eta \in \mathbb{R}^+$, $m \in \mathbb{N}$

Output: $\{f_k : H_k \to \mathbb{R}\}_{k=1}^K$

Initialization:

- For $i = 1, \ldots, n$: $r_i^0 \leftarrow y_i$.
- For each feature $k = 1, \ldots, K$:
  - Compute $a_k = \min_i X_{i,k}$ and $b_k = \max_i X_{i,k}$.
  - Partition $[a_k, b_k]$ into M_k bins H_k.
  - Set $f_k^0(b) \leftarrow 0$ for every $b \in H_k$.

Main Loop:

for $e = 1, \ldots, E$ **do**
  for $k = 1, \ldots, K$ **do**
    Select best splits $S_0, S_1, \ldots, S_m \subseteq H_k$
    for $\ell = 0, \ldots, m$ **do**
      $T \leftarrow \eta \cdot \sum_{b \in S_\ell} \sum_{i \in \mathcal{I}_k(b)} r_i^t$
      $\mu \leftarrow \dfrac{T}{\sum_{b \in S_\ell} |\mathcal{I}_k(b)|}$
      for each $b \in S_\ell$ **do**
        $f_k^t(b) \leftarrow f_k^t(b) + \mu$
      end for
    end for
  end for
  for $i = 1, \ldots, n$ **do**
    $r_i^{t+1} \leftarrow y_i - \sum_{k=1}^K f_k^t\big(\rho(H_k, X_{i,k})\big)$
  end for
end for

and a tree, respectively, K is the number of features, and E denotes the number of training epochs.

4.1. Noise Injection Strategy

In DP-EBM, noise is injected at two stages to ensure differential privacy: during the binning of data (DP binning) and per iteration while learning the values for each leaf node.

DP Binning: To privately bin data, the DP-EBM employs a differentially private histogram construction method. The histogram bins are created by adding Gaussian noise, calibrated based on the privacy budget ϵ_{bin}. For categorical data, the maximum number of bins equals the number of unique values, while for numerical data, it is set to M. This ensures that individual data points cannot be inferred from the bin counts.

Tree Construction with Noise: In the tree construction phase, DP-EBM iteratively builds trees for each feature k. For each leaf node, the values are updated by injecting Gaussian noise to the aggregated sum of residuals. Specifically, for a given leaf node v_j, the sum T is perturbed as:

$$\hat{T} = T + \sigma \cdot \eta R \cdot \mathcal{N}(0, 1)$$

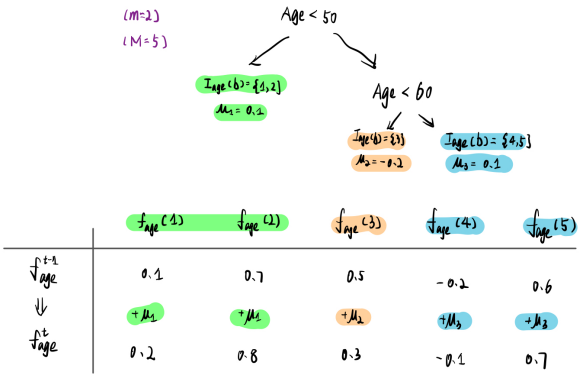

Figure 1. Illustration of gradient tree boosting updates for an age-based feature. Here, $M = 5$ bins are considered, and the maximum number of splits per tree is $m = 2$. Each bin is associated with an update μ_j, calculated from negative pseudo-residuals in the main loop of Algorithm(1). Predictions (f_{age}^t) are updated incrementally from the previous iteration's prediction (f_{age}^{t-1}) by adding the corresponding μ_j values for each bin.

Here, σ is the noise scale derived from the privacy budget, η is the learning rate, R is the sensitivity of the data, and $\mathcal{N}(0, 1)$ represents a standard Gaussian noise.

4.2. Shape Function and Tree Splitting

After constructing the differentially private histograms $\mathcal{H}_k$ for each feature k, DP-EBM builds trees $\mathcal{T}_k$ to update the shape functions f_k. The shape function is a one-dimensional lookup table that takes a bin index as input. The tree decomposes the data space into leaf nodes, each containing data points based on the histogram bins.

The location of splits within a histogram bin is randomly selected, and the maximum number of leaf nodes is set to m. The lookup table for f_k is updated per iteration using the noisy aggregate of residuals for the corresponding bin index. This iterative update ensures that the shape functions reflect the underlying data distribution while preserving privacy.

4.3. Advantages of GDP Analysis

By leveraging Gaussian Differential Privacy, DP-EBM benefits from a tighter analysis of privacy composition compared to standard differential privacy guarantees. The conversion of privacy budgets to GDP format using Theorem 2.5 allows for more efficient composition, leading to stronger privacy guarantees for the same privacy budget. This advantage becomes significant in scenarios involving multiple rounds, where composition bounds play a crucial role.

The use of GDP also simplifies the analysis of privacy trade-

Algorithm 2 Differentially Private Explainable Boosting

Input: X, y, E, η, m, R, ϵ, δ

Output: $\{f_k : H_k \to \mathbb{R}\}_{k=1}^{K}$

Initialization:

- For $i = 1, \ldots, n$: $r_i^0 \leftarrow y_i$.
- For each feature $k = 1, \ldots, K$:
  - Compute $a_k = \min_i X_{i,k}$ and $b_k = \max_i X_{i,k}$.
  - Privately bin data: $\hat{H}_k = DPBin(X[:,k], \epsilon_{\text{bin}})$
  - Set $f_k^0(b) \leftarrow 0$ for every $b \in \hat{H}_k$.

Main Loop:

for $e = 1, \ldots, E$ **do**
  for $k = 1, \ldots, K$ **do**
    Select Random splits $S_0, S_1, \ldots, S_m \subseteq H_k$
    for $\ell = 0, \ldots, m$ **do**
      $T \leftarrow \eta \cdot \sum_{b \in S_\ell} \sum_{i \in \mathcal{I}_k(b)} r_i^t$
      $\hat{T} \leftarrow T + \sigma \cdot \eta R \cdot \mathcal{N}(0,1)$
      $\mu \leftarrow \frac{\hat{T}}{\sum_{b \in S_\ell} \hat{H}_k(b)}$
      for each $b \in S_\ell$ **do**
        $f_k^t(b) \leftarrow f_k^t(b) + \mu$
      end for
    end for

    for $i = 1, \ldots, n$ **do**
      $r_i^{t+1} \leftarrow y_i - \sum_{k=1}^{K} f_k^t\big(\rho(H_k, X_{i,k})\big)$
    end for

offs by expressing privacy guarantees directly in terms of the Gaussian mechanism's noise scale, making it easier to interpret and control the impact of privacy parameters on model accuracy.

Overall, DP-EBM enhances the privacy-utility trade-off by adopting a GDP-based analysis for privacy budget allocation and noise injection. In this approach, each individual update—whether it is a leaf update in the tree or a histogram count—is protected by the Gaussian mechanism and therefore provides $1/\sigma - GDP$. Since there are E epochs and K features, the algorithm performs a total of $E \cdot K$ independent updates. By applying the theorem 2.4, the overall privacy guarantee is given by:

$$\mu = \sqrt{E \cdot K \cdot \left(\frac{1}{\sigma}\right)^2} = \frac{\sqrt{E \cdot K}}{\sigma}\text{-GDP.}$$

This result demonstrates that the cumulative privacy loss scales with the square root of the total number of updates, ensuring that even with many updates, the overall privacy guarantee remains tight.

5. Critical analysis of the results

The results presented in the article (in Table 2 and 3) are very promising. Indeed, they show that for four different datasets on both regression and classification, their model systematically outperforms DP-Boost, a state-of-the-art DP model based on gradient boosting trees and a DP linear or logistic model. They also show that if we want to apply a post-processing operation at the end, like imposing monotonicity, it can be done fairly easily and works well. But the paper, by placing itself in this context, conveniently omits the comparison with more complex models, such as deep DP models. It would have been interesting to compare the performances of DP-EBM with deep DP models on more complex datasets (the maximum number of features on the datasets presented is 49, and the EBM model cannot analyze relations between features). It is also limited to tabular data. So, in the end, the model is only relevant on tabular, non-complex datasets, which is a bit limited. Furthermore, they are emphasizing the benefits of regularization as a result of adding noise, but, as seen in Figure 5 in the article, it may have the opposite effect and smooth out subtle data patterns. This is briefly discussed at the end of the paper but quickly overlooked.

In the end, DP-EBM achieves strong results compared to similar or smaller models but is very restricted on the type and complexity of the data.

6. Experiments

This section describes the replication of experiments presented in the paper, alongside a simplified, custom-coded version of the differentially private Explainable Boosting Machine (DP-EBM) algorithm. The paper's goal is to show that an additive model trained with differential privacy can achieve strong accuracy and exact interpretability on tabular datasets. The official DP-EBM implementation uses a number of careful privacy-preserving mechanisms, including differential private binning, randomized splits and rigorous composition accounting.

In our replication, we focus on two main components:

- Empirical benchmarks comparing non-private models (Logistic Regression, Random Forest, XGBoost, APLR and EBM) on several classification datasets. (benchmark_algorithms.ipynb)

- A "toy" DP-EBM implementation that captures the spirit of Algorithm 2 but omits certain details to keep the codebase simpler and faster to prototype. (custom-DP-EBM.ipynb)

6.1. Datasets and Setup

We used four public classification datasets which were used by the authors: Adult Income, Telco Customer Churn and Credit Card Fraud, as well as a Breast Cancer dataset. For each dataset, we split into training (80%) and test (20%) .

6.2. Benchmarks with Non-Private Models

To replicate the non-private baselines, we trained five models on each dataset. The models used were: **Logistic Regression (LR)**, **Random Forest** (100 estimators), **XGBoost**, **APLR** (Additive Predictor with Linear Relationships), **EBM** (Explainable Boosting Machine, non-private)

We applied standard transformations (one-hot encoding for categorical variables, scaling for numeric features) in a scikit-learn **Pipeline** or took advantage of **EBM**'s built-in handling of mixed data types, similar to how it is done in the GitHub repository (InterpretML, 2025).

These experiments confirm that **EBMs** provide competitive performance, often near or above that of **LR** and **Random Forest**, occasionally rivaling **XGBoost** in AUC.

Telco Churn: EBM ≈ 0.853, beating LR (≈ 0.808) and RF (≈ 0.824).

Adult: EBM ≈ 0.929, slightly above LR (≈ 0.907) and RF (≈ 0.903).

Credit Fraud: EBM ≈ 0.982, close to XGBoost (≈ 0.983) and above RF (≈ 0.950).

These results fit the paper's general finding: standard (non-private) EBMs can match or exceed many classic models on tabular classification tasks.

6.3. DP-EBM Experiments (library based)

We then tested the official (library-based) DP-EBM (InterpretML, 2025) under different ϵ values and using either the "classic" or "GDP" composition method for the Adult dataset. The code iterated through $\epsilon \in \{0.1, 0.5, 2.0, 4.0, 8.0\}$. The results showed:

- At stronger privacy ($\epsilon = 0.1$), AUC dipped into the 0.77–0.81 range on Adult Income.
- As ϵ increased ($\epsilon = 4$ or 8), performance converged closer to the non-private EBM (AUC around 0.88–0.89).
- GDP composition typically gave a slight boost in accuracy relative to classic composition for the same ϵ.

Overall, we replicated the paper's observation that DP-EBM outperforms other DP baselines by incurring relatively small accuracy losses even under strong privacy guarantees.

7. Custom "Toy" DP-EBM Implementation

We additionally wrote a simplified Python script that attempts to follow Algorithm 2 by:

- **Binning each feature** into up to 32 bins,
- **Running cyclic boosting** over each feature for several epochs,
- **Adding Gaussian noise** to each residual update.

These predictions are then aggregated via an additive model. The results showed that even a simplistic or rudimentary noise-adding scheme yields reasonable AUC values (e.g., Adult test AUC ≈ 0.84, Telco test AUC ≈ 0.82, Credit Fraud test AUC ≈ 0.88).

7.1. Limitations and Omissions

Although our custom code captures a core idea (cyclic boosting + noise injection), it omits several steps from the paper's Algorithm 2.

In the paper, the binning step is itself run under differential privacy, releasing only noisy counts and bin boundaries. Our code uses conventional quantile binning without privacy checks. Additionally, the official approach picks splits randomly to avoid spending privacy budget on structure search. We rely on fixed bin edges and do not sample splits from the data.

The official method uses a differentially private mechanism to compute both the sum of residuals and the (noisy) count of points in each bin. By contrast, we compute residual means directly and add Gaussian noise afterward without a fully DP count step. Moreover, in the paper, privacy loss is tracked across multiple rounds of boosting (using either strong composition or Gaussian Differential Privacy). Our toy code applies a single global noise scale based on ϵ, δ. Lastly, The official DP-EBM uses bagging to reduce variance and edits (e.g., enforcing monotonicity). Our code runs only one model pass without any bagging or shape-function constraints.

Consequently, our implementation is not a complete replication from a privacy standpoint; it is mainly a conceptual demonstration. However, it should be noted that the omissions were made in our custom code primarily to reduce implementation complexity and runtime overhead for this classroom scale project. Each of these steps includes intricate accounting of privacy budget, additional data transformations (e.g., publishing noisy bin boundaries) or computational overhead (e.g., large-scale bagging runs). Moreover, thoroughly validating the correctness of each component (especially under multiple composition theorems) would significantly expand both the codebase and the scope of

Dataset	Domain	N	K	Task
Parkinson	Medicine	5,875	19	Reg.
Phishing	Security	11.055	30	Class.

Table 1. Statistics of datasets

this work. As a result, our simplified version offers only a conceptual demonstration of DP-EBM rather than the strict privacy guarantees or performance optimizations claimed in the paper.

8. Our contributions

As our contribution, on top of rewriting the code and checking the consistency of the results for the given datasets, we have decided to test the algorithm on new datasets. We have tested it on two new datasets: one on a regression problem and another on a classification problem. We then compared the results of our custom implementation of the algo, the two models and other baselines like linear or logistic regression and a random forest. The first dataset, for regression, is a dataset on parkinson's disease, while the second, for classification, is on phishing sites. Both datasets have been found here. The statistics of these datasets can be found in Table 1.

The results can be seen on tables 3 and 2. Because of a lack of time, we could not compare the models to DP-Boost, but instead, we compared it to a logistic regression and a random forest from the package diffprivlib for classification. As for regression, the linear model implemented by the same python library worked terribly bad (RMSE in the thousands), even for high epsilon, so we did not include it for comparison and no other model was quickly available. Notice that we do not have a standard deviation for our custom implementation; that is because we did not implement some optimization the paper did, so it was really slow to run it 20 times for each epsilon on each dataset. We thus have only one result. By doing these further experiments, we have confirmed that DP-EBM and especially the gdb version, performs well, and better than a logistic or linear regression and a DP random forest for classification.

Dataset	ϵ	custom	classic	gdb
Parkinson	0.1	29.71	30.4 $\pm$1.673	17.2 $\pm$0.749
	0.5	29.74	12.6 $\pm$0.168	11.4 $\pm$0.017
	2	29.71	10.6 $\pm$0.045	9.6 $\pm$0.044
	4	29.71	9.7 $\pm$0.119	9.4 $\pm$0.071
	8	29.71	9.4 $\pm$0.033	9.1 $\pm$0.044

Table 2. RMSE algorithm comparison on new dataset

9. Conclusion

In summary, adding differential privacy to Explainable Boosting Machines (DP-EBMs) allows us to preserve the intuitive shape function view of features while safeguarding sensitive data. Our experiments showed that, even under strong privacy budgets, DP-EBMs retain competitive performance on several tabular datasets. We further demonstrated a simplified "toy" implementation to illustrate the core mechanics of DP-EBM, confirming that cyclic boosting plus noise injection can yield acceptable results in practice. Although we intentionally omitted some privacy-preserving steps for simplicity, the official DP-EBM design highlights how a rigorous combination of Gaussian Differential Privacy and additive modeling can excel in both privacy protection and intelligibility.

How to optimally partition the privacy budget and tune parameters in differentially private models remains an open problem. For example, in DP-EBM, deciding the best allocation between histogram binning and tree updates requires balancing privacy and utility and the optimal trade-off.

10. Contributions

Aymeric: Intro, critical analysis, contributions
Xichen: Foundations, EBM, DP-EBM
Efstathios: Code and Experiment Replication, Custom DP-EBM implementation

References

Dong, J., Roth, A., and Su, W. J. Gaussian differential privacy. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 84(1):3–37, 2022.

Dwork, C., Roth, A., et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

InterpretML. interpret: Fit interpretable machine learning models. https://github.com/interpretml/interpret, 2025. Accessed: 10/03/2025.

Nori, H., Jenkins, S., Koch, P., and Caruana, R. Interpretml: A unified framework for machine learning interpretability. *arXiv preprint arXiv:1909.09223*, 2019.

Dataset	ϵ	custom	classic	gdb	logistic	rdm forest
Phishing	0.1	0.765	0.673 $\pm$0.005	0.914 $\pm$0.009	0.873 $\pm$0.008	0.757 $\pm$0.003
	0.5	0.540	0.948 $\pm$0.004	0.968 $\pm$0.003	0.888 $\pm$0.002	0.757 $\pm$0.003
	2	0.784	0.972 $\pm$0.003	0.977 $\pm$0.003	0.896 $\pm$0.005	0.757 $\pm$0.003
	4	0.943	0.975 $\pm$0.003	0.978 $\pm$0.002	0.897 $\pm$0.005	0.758 $\pm$0.003
	8	0.950	0.977 $\pm$0.003	0.979 $\pm$0.002	0.897 $\pm$0.005	0.758 $\pm$0.003

Table 3. AUROC algorithm comparison on new dataset