7. Appendix

7.1. Differentially Private Binning

While not a main contribution of this paper, for completeness (and reproducibility) we describe the differentially private binning algorithm used as a preprocessing step to DP-EBMs. Our goal is to create bins for each feature such that each bin contains roughly equal proportions of data. For example, if the goal is to end up with 10 bins per feature, we expect each bin to roughly contain $\frac{1}{10}$ of the data.

Like other DP tree implementations, we assume that the min and max of each feature are supplied by the user (Dwork et al., 2014; Jagannathan et al., 2009). The algorithm begins by uniformly dividing the feature space into *equal width* bins, without looking at any user data. If the user requests m bins in total, the binning procedure creates $2 \cdot m$ equal width bins to begin with. We then create a differentially private histogram based on those uniform bin widths, adding noise with sensitivity 1 (Dwork et al., 2006). Theorems 4 and 5 are also applied here to track cumulative privacy budget across all K features and calibrate how much noise to add. Finally, to transform the noisy equal width bins into equal density bins, the algorithm greedily post-processes the released bin definitions by collapsing small bins into their neighbors until a sufficiently large "quantile" bin is achieved. While this method can be sub-optimal on highly skewed distributions, we find that it works well in practice on most datasets, and users have some control by choosing appropriate min/max values or applying transforms to features prior to training. The full algorithm is detailed below.

Algorithm 3 Differentially Private Quantile Binning

Input: data X, target bins m, privacy parameters ε, δ
Output: Histogram $\hat{H}_k$ per feature
Target datapoints per bin: $t = \frac{|X|}{m}$
for feature $0...K$ **do**
  Equal width bins: $H_k = Hist(X[:,k], 2m)$
  Add noise to counts: $\hat{H}_k = H_k + \sigma \cdot N(0,1)$
  Postprocess:
  for each histogram bin $b_i \in \hat{H}_k$ **do**
    if $|b_i| < t$ **then**
      Greedily collapse bins: $b_{i+1} = b_{i+1} + b_i$
      Delete previous bin: $Delete(\hat{H}_k, b_i)$
    end if
  end for
  Check if final bin is sufficiently large
  if $|b_i| < t$ **then**
    Collapse into previous bin: $b_{i-1} = b_{i-1} + b_i$
    Delete final bin: $Delete(\hat{H}_k, b_i)$
  end if
end for

7.2. Adaptations to other settings

Algorithm 2 in the main body of the paper focuses on regression, which often can be adapted to other settings. It also is possible to use many alternative loss functions in DP-EBMs with no change to the privacy analysis. For example, to adapt DP-EBMs to binary classification, we might prefer residuals to be *logits*. Our proof of privacy depends on ensuring that the sensitivity of the sum of residuals is bounded by at most R at each iteration. In the regression setting, we show that the sum of residuals T can be framed as:

$$T = \eta \cdot \left(\sum_{b \in S_\ell} \sum_{x_i \in \hat{H}_k(b)} y_i \right) - Z$$

where Z is entirely public information released from previous iterations of the model. Therefore, simple transformations on Z do not affect the sensitivity of each update or the ultimate privacy guarantee of the algorithm. For binary classification, the only modification is to line 25:

$$r_i^t = y_i + \frac{1}{1+e^{\sum f_k^t(\rho(H_k, x_i))}} - 1$$

7.3. Abnormally low AUROCs

In our experiments, some models produce AUROCs substantially lower than 0.5. For example, on the credit fraud dataset for $\varepsilon = 1$, DPBoost had an average test set AUROC of 0.438. This was a bit surprising, as predicting random labels should result in AUCs near 0.5. Further investigation suggests that the noise DP adds to models increases the variance of model predictions enough to make AUCs much larger and smaller than 0.5.

For example, Figure 5 shows the distribution in AUCs for the healthcare dataset when predictions are made completely randomly (left), and also shows the distribution for the same dataset when predictions are made with DP Logistic Regression with $\varepsilon = 0.5$ (right). Both distributions have mean 0.5, but the distribution is much wider for the model with $\varepsilon = 0.5$. Similar behavior is observed for all DP algorithms, including DP-EBMs with $\varepsilon < 0.1$.

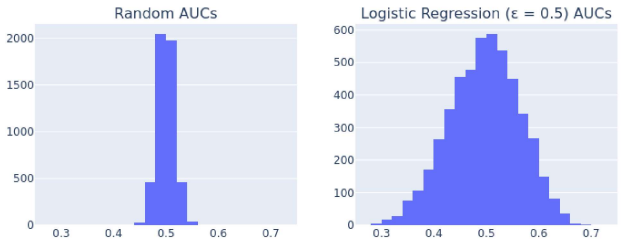

Figure 6. Distribution of AUROCs on 5000 train/test splits of healthcare dataset. Left: Randomly generated predictions. Right: DP Logistic Regression ($\varepsilon = 0.5$) test set AUROCs.