Polynomial magic!
Hermite polynomials for private data generation

Mijung Park\textsuperscript{1,2} Margarita Vinaroz\textsuperscript{1,2}
Mohammad-Amin Charusaie\textsuperscript{1,2}
Frederik Harder\textsuperscript{1,2}
\textsuperscript{1}Max Planck Institute for Intelligent Systems, Tübingen, Germany
\textsuperscript{2}Department of Computer Science, University of Tübingen, Tübingen, Germany
{mpark|mvinaroz|fharder|mcharusaie}@tuebingen.mpg.de

Abstract

Kernel mean embedding is a useful tool to compare probability measures. Despite its usefulness, kernel mean embedding considers infinite-dimensional features, which are challenging to handle in the context of differentially private data generation. A recent work \cite{13} proposes to approximate the kernel mean embedding of data distribution using finite-dimensional random features, where the sensitivity of the features becomes analytically tractable. More importantly, this approach significantly reduces the privacy cost, compared to other known privatization methods (e.g., DP-SGD), as the approximate kernel mean embedding of the data distribution is privatized only once and can then be repeatedly used during training of a generator without incurring any further privacy cost. However, the required number of random features is excessively high, often ten thousand to a hundred thousand, which worsens the sensitivity of the approximate kernel mean embedding. To improve the sensitivity, we propose to replace random features with Hermite polynomial features. Unlike the random features, the Hermite polynomial features are ordered, where the features at the low orders contain more information on the distribution than those at the high orders. Hence, a relatively low order of Hermite polynomial features can more accurately approximate the mean embedding of the data distribution compared to a significantly higher number of random features. As a result, using the Hermite polynomial features, we significantly improve the privacy-accuracy trade-off, reflected in the high quality and diversity of the generated data, when tested on several heterogeneous tabular datasets, as well as several image benchmark datasets.

1 Introduction

One of the popular distance metrics for generative modelling is Maximum Mean Discrepancy (MMD) \cite{17}. MMD computes the average distance between the realizations of two distributions mapped to a reproducing kernel Hilbert space (RKHS). Its popularity is due to several facts: (a) MMD can compare two probability measures in terms of all possible moments (i.e., infinite-dimensional features), resulting in no information loss due to a particular selection of moments; and (b) estimating MMD does not require the knowledge of the probability density functions. Rather, MMD estimators are in closed form, which can be computed by pair-wise evaluations of a kernel function using the points drawn from two distributions.

Preprint. Under review.
However, using the MMD estimators for training a generator is not well suited when differential privacy (DP) of the generated samples is taken into consideration. In fact, the generated points are updated in every training step and the pair-wise evaluations of the kernel function on generated and true data points require accessing data multiple times. One of the key properties of DP is composability that implies each access of data causes privacy loss. Hence, privatizing the MMD estimator in every training step—which is necessary to ensure the resulting generated samples are differentially private—incurs a large privacy loss.

A recent work [13] uses a particular form of MMD via a random Fourier feature representation [23] of kernel mean embeddings for DP data generation. Under this representation, we can rewrite the approximate MMD in terms of two finite-dimensional mean embeddings (as in eq. [1]), where the approximate mean embedding of the true data distribution (data-dependent) is detached from that of the synthetic data distribution (data-independent). Thus, we can privatize the data-dependent term only once and use it repeatedly during training of a generator.

Building on [13], we propose to replace the random feature representation of the kernel mean embedding with the Hermite polynomial representation. We observe that using Hermite polynomial features yields a more accurate approximation to the Gaussian kernel, compared to the random features. As illustrated in our experiments, the required order of Hermite polynomial features is significantly lower than the required number of random features, for the similar quality of the kernel approximation. This is useful in reducing the effective sensitivity of the data mean embedding, specifically in the case of small datasets in which sensitivity is likely to be large. We extend this observation to the high-dimensional data setting, where we propose a sum of Gaussian kernels and each Gaussian kernel is defined on each data dimension. The choice of the sum of Gaussian kernels is owing to the computational tractability of the approximate kernel mean embedding. We then further approximate each Gaussian kernel via Hermite polynomials. This provides us with a linear number of features in terms of data dimension, which leads to a small effective sensitivity. Our contributions are summarized below:

• We propose to use the Hermite polynomial representation of the kernel mean embedding to improve the effective sensitivity of the data mean embedding.
• For high-dimensional data, we propose a new kernel, a sum of Gaussian kernels (each Gaussian kernel defined on each coordinate of the input variable), for computational tractability of the kernel mean embedding.
• As a result, we improve the privacy-accuracy trade-off, where accuracy is measured by the classification accuracy evaluated on 12 downstream tasks, as shown in Table 2 for tabular datasets and Fig. 4 for image datasets.

2 Background

In the following, we describe the background on kernel mean embeddings and differential privacy.

2.1 Maximum Mean Discrepancy

Given a positive definite kernel $k: \mathcal{X} \times \mathcal{X}$, the MMD between two distributions $P, Q$ is defined as [12]: $\text{MMD}^2(P, Q) = \mathbb{E}_{x,x' \sim P}k(x, x') + \mathbb{E}_{y,y' \sim Q}k(y, y') - 2\mathbb{E}_{x \sim P}\mathbb{E}_{y \sim Q}k(x, y)$. According to the Moore–Aronszajn theorem, there exists a unique Hilbert space $\mathcal{H}$ on which $k$ defines an inner product. Hence, we can find a feature map, $\phi: \mathcal{X} \rightarrow \mathcal{H}$ such that $k(x, y) = \langle \phi(x), \phi(y) \rangle_{\mathcal{H}}$, where $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ denotes the inner product on $\mathcal{H}$. Using this, we can rewrite the MMD [12]

$$\text{MMD}^2(P, Q) = \|\mathbb{E}_{x \sim P}\phi(x) - \mathbb{E}_{y \sim Q}\phi(y)\|^2_{\mathcal{H}},$$

(1)

where $\mathbb{E}_{x \sim P}\phi(x) \in \mathcal{H}$ is known as the (kernel) mean embedding of $P$, and exists if $\mathbb{E}_{x \sim P}\sqrt{k(x, x)} < \infty$ [28]. The MMD can be interpreted as the distance between the mean embeddings of the two distributions. If $k$ is a characteristic kernel [31], then $P \mapsto \mathbb{E}_{x \sim P}\phi(x)$ is injective, and MMD forms a metric, implying that $\text{MMD}(P, Q) = 0$, if and only if $P = Q$.

1Although the sensitivity is $\frac{1}{m}$ in both cases, for the number of samples $m$ (see Sec. 3), adding noise to a vector of longer length (when using random features) has a worse signal to noise ratio, as opposed to adding noise to a vector of shorter length (when using Hermite polynomial features).
Given the samples drawn from two probability distributions: \( X_m = \{x_i\}_{i=1}^m \sim P \) and \( X'_n = \{x'_i\}_{i=1}^n \sim Q \), we can estimate\(^2\) the MMD by sample averages\(^2\):

\[
\hat{\text{MMD}}^2 (X_m, X'_n) = \frac{1}{m^2} \sum_{i,j=1}^m k(x_i, x_j) + \frac{1}{n^2} \sum_{i,j=1}^n k(x'_i, x'_j) - \frac{2}{mn} \sum_{i=1}^m \sum_{j=1}^n k(x_i, x'_j). \tag{2}
\]

However, at \( O(mn) \) the computational cost of \( \hat{\text{MMD}}(X_m, X'_n) \) is prohibitive for large-scale datasets.

### 2.2 Kernel approximation

The above mentioned MMD estimator can be cheaply computed by approximating the kernel function \( k(x, x') \) with an inner product of finite dimensional feature vectors, i.e., \( k(x, x') \approx \hat{\phi}(x)\top \hat{\phi}(x') \) where \( \hat{\phi}(x) \in \mathbb{R}^A \) and \( A \) is the number of features. The resulting approximation of the MMD estimator given in eq.\(^2\) can be computed in \( O(m + n) \), i.e., linear in the sample size:

\[
\hat{\text{MMD}}^2 (P, Q) = \left\| \frac{1}{m} \sum_{i=1}^m \hat{\phi}(x_i) - \frac{1}{n} \sum_{i=1}^n \hat{\phi}(x'_i) \right\|^2_2. \tag{3}
\]

The approximation is beneficial for computational tractability for large datasets. More importantly, in terms of privacy, it is also beneficial: if we treat \( P \) as a data distribution and \( Q \) as a synthetic data distribution, we can summarize data distribution in terms of its mean embedding (i.e., the first term on the right-hand side of eq.\(^3\)), which can be privatized only once and used repeatedly during training of the generator which produces samples from \( Q \).

**Random Fourier features.** As an example of \( \hat{\phi}(\cdot) \), the random Fourier features\(^2\) are derived from the following. Bochner’s theorem\(^2\) states that for any translation invariant kernel, the kernel can be written as \( k(x, x') = \hat{k}(x - x') = \mathbb{E}_{\omega \sim \Lambda} \cos(\omega \top (x - x')) \). By drawing random frequencies \( \{\omega_i\}_{i=1}^A \sim \Lambda \), where \( \Lambda \) depends on the kernel, (e.g., a Gaussian kernel \( k \) corresponds to normal distribution \( \Lambda \)), \( k(x - x') \) can be approximated with a Monte Carlo average. The resulting vector of random Fourier features (of length \( A \)) is given by

\[
\hat{\phi}_{RF}(x) = (\hat{\phi}_1(x), \ldots, \hat{\phi}_A(x))\top \tag{4}
\]

where \( \hat{\phi}_j(x) = \sqrt{2/A} \cos(\omega_j\top x), \hat{\phi}_j + A/2(x) = \sqrt{2/A} \sin(\omega_j\top x), \) for \( j = 1, \ldots, A/2 \). DPMERF\(^2\) uses this very representation of the feature map given in eq.\(^4\) and minimize eq.\(^3\) with a privatized data mean embedding to train a generator.

**Hermite polynomial features.** For another example of \( \hat{\phi}(\cdot) \), one could also start with the **Mercer’s theorem** (See Supplementary Sec.\(^A\)), which allows us to express a positive definite kernel \( k \) in terms of the eigen-values \( \lambda_i \) and eigen-functions \( f_i: k(x, x') = \sum_{i=1}^\infty \lambda_i f_i(x)f_i\top(x') \), where \( \lambda_i > 0 \) and complex conjugate is denoted by \( * \). The resulting **finite-dimensional** feature vector is simply \( \hat{\phi}(x) = \hat{\phi}_{HF}(x) = [\sqrt{\lambda_0} f_0(x), \sqrt{\lambda_1} f_1(x), \ldots, \sqrt{\lambda_C} f_C(x)] \) where the cut-off is made at the \( C \)-th eigen-value and eigen-function. For the commonly-used Gaussian kernel, \( k(x, x') = \exp\left(-\frac{1}{2\sigma^2} (x - x')^2\right) \), where \( l \) is the length scale parameter, an analytic form of eigen-values and eigen-functions are available, where the eigen-functions are represented with Hermite polynomials (See Sec.\(^3\) for definition). This is the approximation we will use in our method.

### 2.3 Differential privacy

Given privacy parameters \( \epsilon \geq 0 \) and \( \delta \geq 0 \), a mechanism \( \mathcal{M} \) is \((\epsilon, \delta)\)-DP if the following equation holds: \( \Pr[\mathcal{M}(\mathcal{D}) \in S] \leq e^\epsilon \cdot \Pr[\mathcal{M}(\mathcal{D}') \in S] + \delta \), for all possible sets of the mechanism’s outputs \( S \) and all neighbouring datasets \( \mathcal{D}, \mathcal{D}' \) differing by a single entry. A DP mechanism guarantees a limit on the amount of information revealed about any single individual’s participation in the dataset. Conventionally, \( \delta \leq 1/n \) and \( \epsilon < 2 \) are considered reasonable choices because larger values already allow for significant privacy loss, as illustrated in\(^3\).

In this paper, we use the **Gaussian mechanism** to ensure the output of our algorithm is DP. Consider a function \( h: \mathcal{D} \mapsto \mathbb{R}^p \), where we add noise for privacy and the level of noise is calibrated to the global\(^2\)Note that this particular MMD estimator is biased.
we can define a

where the
datasets we often deal with are multi-
value

dimensional. While one can uniformly approximate the MMD based on multivariate Gaussian kernel

3.2 Handling multivariate inputs using a sum of Gaussian kernels

The Mehler formula holds for 1-dimensional data, while the datasets we often deal with are multi-
dimensional. While one can uniformly approximate the MMD based on multivariate Gaussian kernel

3 Our method: DP-HP

Based on the Mercer’s theorem, the Gaussian kernel can be represented as a weighted sum of Hermite polynomial features. This approximation is also described as the Mehler formula below.

3.1 Approximating the Gaussian kernel using Hermite polynomials (HP)

Using the Mehler formula, for $|\rho| < 1$, we can write down the Gaussian kernel as

$$\exp\left(-\frac{\rho}{1-\rho^2}(x-y)^2\right) = \sum_{c=0}^{\infty} \lambda_c f_c(x) f_c(y)$$

(5)

where the $c$-th eigen-value is $\lambda_c = (1-\rho)^{2c}$ and the $c$-th eigen-function is defined by $f_c$, where $f_c(x) = \frac{1}{\sqrt{N_c}} H_c(x) \exp\left(-\frac{\rho}{1-\rho^2}x^2\right)$, and $N_c = 2^c c! \sqrt{\frac{1-\rho^2}{1+\rho^2}}$. Here, $H_c(x) = (-1)^c \exp(x^2) \frac{d^c}{dx^c} \exp(-x^2)$ is the $c$-th order Hermite polynomial. As a result of the Mehler formula, we can define a $C$-th order Hermite polynomial features as a feature map (a vector of length $C+1$):

$$\hat{\phi}_H^{(C)}(x) = \left[\sqrt{\lambda_0} f_0(x), \sqrt{\lambda_1} f_1(x), \cdots, \sqrt{\lambda_C} f_C(x)\right],$$

(6)

and approximate the Gaussian kernel via

$$\exp\left(-\frac{\rho}{1-\rho^2}(x-y)^2\right) \approx \hat{\phi}_H^{(C)}(x)^T \hat{\phi}_H^{(C)}(y).$$

This feature map provides us with a uniform approximation to the MMD in eq. 1 for every pair of distributions $P$ and $Q$ (see Theorem A.1 and Lemma A.1 in Supplementary Sec. A). We compare the accuracy of this approximation with random features in Fig. 1, where we fix the length scale to the median heuristic value 3 in both cases. We also show the comparison between HP and random features in Fig. 2.

Computing the Hermite polynomial features. Hermite polynomials follow the recursive definition: $H_{c+1}(x) = 2xH_c(x) - 2cH_{c-1}(x)$. At high orders, the polynomials take on large values, leading to numerical instability. So we compute the re-scaled term $\phi_c = \sqrt{\lambda_c} f_c$ iteratively using a similar recursive expression given in Supplementary Sec. D.

3.2 Handling multivariate inputs using a sum of Gaussian kernels

The Mehler formula holds for 1-dimensional data, while the datasets we often deal with are multi-
dimensional. While one can uniformly approximate the MMD based on multivariate Gaussian kernel

Fig. 1: HP features VS. RF features. Dataset $X$ contains $N = 100$ samples drawn from $\mathcal{N}(0,1)$ and $X'$ contains $N = 100$ samples drawn from $\mathcal{N}(1,1)$. We define the error by the mean absolute difference: $\frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{N} |k(x_i, x_j') - \phi(x_i)^T \phi(x_j')|$ where $\phi$ is either HP or RF features. Top: The error decays fast when using HP features (eq. 3). Bottom: The error decays slowly when using RF features (eq. 4). The best error (black dotted line) using 500 RF features coincides with the error using HP features with order 2.

sensitivity $\delta_h$, defined by the maximum difference in terms of $L_2$-norm $||h(D) - h(D')||_2$, for neighbouring $D$ and $D'$ (i.e. $D$ and $D'$ have one sample difference by replacement), where the output is denoted by $h(D) = h(D) + \mathcal{N}(0, \sigma^2 \Delta^2 I_n)$. The perturbed function $h(D)$ is $(\epsilon, \delta)$-DP, where $\sigma$ is a function of $\epsilon$ and $\delta$ and can be computed using the auto-dp package by [34]
which is manageable compared to the size (where the features map is the size of (C + 1)^D, where D is the input dimension of the data and C is the chosen order of the Hermite polynomials (See Supplementary Sec. [3] for a detailed description). This is prohibitive for the datasets we often deal with, e.g., for MNIST (D = 784) with a relatively small order (say C = 10), the size of feature map is 11^{784}, impossible to fit in a typical size of memory. Hence, to handle high-dimensional data, we need to construct a new kernel for computationally-tractable feature maps.

Our proposal: a sum of Gaussian kernels. For D-dimensional inputs x, x’ ∈ R^D where x = [x_1, · · · , x_D] and x’ = [x’_1, · · · , x’_D], we define a kernel on χ_1 × · · · × χ_D, where x_d, x’_d ∈ χ_d for d = 1, · · · , D, by a sum of Gaussian kernels:

\[ k(x, x’) = \frac{1}{D} [k_{X_1}(x_1, x’_1) + k_{X_2}(x_2, x’_2) + \cdots + k_{X_D}(x_D, x’_D)] = \frac{1}{D} \sum_{d=1}^{D} k_{X_d}(x_d, x’_d), \quad (7) \]

where we again represent each Gaussian kernel by the Hermite Polynomials given in eq. 6

\[ k_{X_d}(x_d, x’_d) \approx \hat{\phi}_{HP,d}(x_d) \mathcal{H}^C(\hat{\phi}_{HP,d}(x_d)). \]

The corresponding feature map is represented by

\[ \hat{\phi}(x) = \begin{bmatrix} \hat{\phi}_{HP,1}(x_1)/\sqrt{D} \\ \hat{\phi}_{HP,2}(x_2)/\sqrt{D} \\ \vdots \\ \hat{\phi}_{HP,D}(x_D)/\sqrt{D} \end{bmatrix} \in \mathbb{R}^{((C + 1) \times D) \times 1}, \quad (8) \]

where the features map is the size of (C + 1)D. For the MNIST digit data (D = 784), with a relatively small order, say C = 10, the size of the feature map is 11 \times 784 = 8624 dimensional, which is manageable compared to the size (11^{784}) of the feature map under the generalized Hermite polynomials. For the computational tractability, in this paper we choose to use this sum kernel. The sum kernel does not approximate the Gaussian kernel defined on the joint distribution over all the input coordinates. Also, the sum kernel is not characteristic. However, the assigned Gaussian kernel on each coordinate is characteristic. Further, for a pair of distributions P and Q, the approximated MMD of sum kernel - based on Hermite polynomial features - being small is equivalent to the small value of Gaussian MMD for marginal distributions P_i and Q_i. The Lemma C. [1] in Supplementary

---

*Figure 2: Comparison between HP and random features at a different length scale value. Different color indicates a different datapoint, where four datapoints are drawn from N(0, 1). Left: With length scale l = 0.71 (relatively small compared to 1), random features (top) at the four datapoints exhibit large variability while the Hermite polynomial features (bottom) at those datapoints decay at around order ≤ 20. Right: With l = 7.07 (large compared to 1), random features (top) exhibit less variability, while it is not clear how many features are necessary to consider. On the other hand, the Hermite polynomial features (bottom) decay fast at around order ≤ 5 and we can make a cut-off at that order without losing much information.*

---

*Here, we let each coordinate’s Hermite Polynomials \( \hat{\phi}_{HP,d}(x_d) \) take different values of \( \rho \), which determine a different level of fall-offs of the eigen-values and a different range of values of the eigen-functions. Note that one can also impose a different cut-off \( C \) for each coordinate.*
Sec. C shows that by minimizing the approximate MMD between the training data and generated data distributions based on feature maps given in eq. 8, we assure that the marginal probability distributions of the generated data converges to those of the training data.

Comparing two input distributions in terms of the product of the marginals could be seen as restrictive. While using the sum kernel we do not explicitly learn the dependencies among different dimensions of the input, when data exhibits a strong dependencies, e.g., in image data where neighbouring pixels are more correlated than others that are far from each other, one can choose to use a generator such as a CNN to induce such dependencies in the generated inputs.

## 3.3 Approximate MMD for classification

For classification tasks with labeled data, we rewrite the approximate MMD as:

\[
\hat{\text{MMD}}^2_H(P, Q) = \left\| \mu_P(D) - \mu_Q(D') \right\|^2_2, 
\]

and define the mean embedding of the data distribution by \( \hat{\mu}_{P_{\omega,y}}(D) = \frac{1}{m} \sum_{i=1}^{m} h(x_i, y_i) \) and \( \hat{\mu}_{Q_{\omega,y}}(D') = \frac{1}{m} \sum_{i=1}^{m} h(x'_i(\theta), y'_i(\theta)) \), where \( \theta \) denotes the parameters of a synthetic data generator. Here we apply the outer product kernel on the joint of the labels and the corresponding inputs, while we apply the sum kernel given in eq. 7 on the inputs. This results in the feature map: \( h(x_i, y_i) = \phi(x_i) f(y_i)^T \in \mathbb{R}^{((C+1)D) \times L} \), where \( L \) is the number of classes and \( f(y_i) \) is an one-hot-encoded label. In other words, the approximate mean embedding \( \hat{\mu} \) is a concatenation of columns, where each column consists of the mean embedding of the input distribution corresponding to each class. Minimizing eq. 9 ensures that all synthetic class distributions match their real data counterpart in terms of the approximate mean embedding.

## 3.4 Privatizing the approximate mean embedding of the data distribution

We privatize \( \hat{\mu}_P(D) \) using the Gaussian mechanism, where we compute \( \sigma \) as a function of \( (\epsilon, \delta) \) using the RDP composition in [34] and the sensitivity \( S_{\mu_P} \), which we derive in Supplementary Sec. 3.

\[
\hat{\mu}_{P_{\omega,y}}(D) = \mu_P(D) + \mathcal{N}(0, (C+1)D) \times L, \quad \sigma^2 S_{\mu_P} I_{((C+1)D) \times L},
\]

### Imbalanced data

Building on Algorithm 1 in [13], when classes are highly imbalanced, we consider the following privatized mean embedding:

\[
\tilde{\mu}_{P_{\omega,y}} = \left[ \frac{m_1}{m} \tilde{u}_1, \ldots, \frac{m_L}{m} \tilde{u}_L \right],
\]

where we denote the \( l \)-th class’s count by \( m_l \) (with \( m = \sum_{l=1}^{L} m_l \)) and the corresponding mean embedding of the inputs by \( u_l = \frac{1}{m} \sum_{x_i \in X^{(y)}} \phi(x_i) \). We privatize the mean embedding by two steps. First, we privatize the mean embedding by eq. 10. Secondly, we privatize the vector of class counts \( \mathbf{m} = [m_1, \ldots, m_C] \) by

\[
\tilde{\mathbf{m}} = \mathbf{m} + \mathcal{N}(0, \Delta_m^2 \sigma^2 \mathbf{I}_C),
\]

where the sensitivity is simply \( \Delta_m = \sqrt{2} \) as changing a datapoint affects at most two class counts. The procedure is summarized in Algorithm 1.

## 4 Related Work

Approaches to differentially private data release can be broadly sorted into three categories. One line of prior work with background in learning theory aims to provide theoretical guarantees on the

\footnote{We consider this particular mean embedding as for rare classes \( m \) can be significantly larger than the sum of the corresponding column. In the re-weighted mean embedding each class-wise embedding \( \frac{m_l}{m} \tilde{u}_l \) has a similar norm, and equally contributes to the objective loss. This ensures that infrequent classes are also modelled accurately. Note that we arrive at this expression of mean embedding if we change the kernel on the labels to a weighted one, i.e., \( k_{\psi}(y, y') = \sum_{l=1}^{L} \tilde{m}_l y_i y'_l \).}
**Algorithm 1 DP-HP for private data generation**

Require: Dataset $\mathcal{D}$, and a privacy level $(\epsilon, \delta)$

Ensure: $(\epsilon, \delta)$-DP input and output pairs.

Step 1. Given $(\epsilon, \delta)$, compute the privacy parameter $\sigma$ by the RDP composition in [34] for the two uses of the Gaussian mechanism in steps 2 and 3.

Step 2. Release the mean embedding $\tilde{\mu}_{p_{x,y}}$ by eq. 10

Step 3. Release the class counts $\tilde{m}$ by eq. 12.

Step 4. Create the weighted mean embedding $\tilde{\mu}^*_{p_{x,y}}$ following eq. 11.

Step 5. Train the generator by minimizing $\text{MMD}_{H^{2}}(p_{x,y}, q_{x',y'}) = \left\| \tilde{\mu}^*_{p_{x,y}} - \mu_{q_{x',y'}} \right\|^2_F$.

utility of released data [40, 19, 36, 14, 42]. This usually requires strong constraints on the type of data and the intended use of the released data, which may not hold in practice. As our approach aims for practical applicability, these methods significantly differ from ours.

A second line of work focuses on the sub-problem of discrete data with limited domain size, which is relevant to tabular datasets [39, 22, 7, 40]. Such approaches typically approximate the structure in the data by identifying small sub-sets of features with high correlation and releasing these lower order marginals in a private way. Some of these methods have also been successful in the recent NIST 2018 Differential Privacy Synthetic Data Challenge [1], while these methods often require discretization of the data and do not scale to higher dimensionality in arbitrary domains.

In contrast, our work aims for broad applicability without constraints on the type of data or the kind of downstream tasks to be used and is thus most related to a number of recent works, which attempt to leverage the modeling power of deep generative models such as GANs [11] or VAEs [15] in the private setting. While work on VAEs exists [8], GANs are currently the most popular model for this purpose [37, 32, 10, 38, 6], on which we focus in our comparison. A crucial advantage of GANs is that the generator model can be trained without direct access to training data and thus requires no privatization, as long as the discriminator is made private. DP-GAN [37], DP-CGAN [32] and GS-WGAN [6] all utilize a version of DP-SGD [2] to accomplish this training. GS-WGAN improves on the other methods by using multiple discriminator networks trained on separate parts of the dataset to amplify privacy by subsampling and removes the need for gradient clipping through an adaptive loss function. PATE-GAN takes a different approach to training, which is based on the private aggregation of teacher ensembles (PATE) [20]. While DP-CGAN and GS-WGAN generate labeled data, DP-GAN and PATE-GAN do not generate labels. As a result the latter two models must be trained separately for each label subset of the data in order to generate a labeled dataset.

The closest prior work to the proposed method is [13], where kernel mean embeddings are approximated with random Fourier features [23] instead of Hermite polynomials. While random feature approximations of MMD have previously been used with DP [5, 26], to our knowledge, ours is the first work using Hermite polynomials to approximate MMD in the context of differential privacy.

### 5 Experiments

In this section we show the results of our method over different real world datasets. In each case, we train DP-HP and other generative models under privacy constraints. Since we do not know the true data distribution, we evaluate the quality of the (private and non-private) generated samples from these models using the common approach of measuring performance on downstream tasks, following [6, 32, 38, 13]. We train 12 different commonly used classifier models using generated samples and then evaluate the classifiers on a test set containing real data samples. Each setup is averaged over 5 random seeds. The test accuracy indicates how well the models generalize from the synthetic to the real data distribution and thus, the utility of using private data samples instead of the real ones. Details on the 12 models can be found in Table 8.

As comparison models in our experiments, we tested DP-CGAN [32], DP-GAN [37] and DP-MERF [13]. For image datasets we also trained GS-WGAN [6]. While DP-CGAN uses the RDP accountant [19] to account for privacy loss during training, the remaining methods all use the
analytical moments accountant \cite{34}. Using the auto-dp package\footnote{auto-dp is released here: \url{https://github.com/yuxiangw/autodp} (Apache License 2.0)} this allows us to easily compute the corresponding privacy parameter $\sigma$ for the Gaussian mechanism. Our experiments were implemented in PyTorch \cite{21} and run using Nvidia Kepler20 and Kepler80 GPUs. Our code is available at \url{https://github.com/mvinaroz/dp-hp}.

**Tabular data.** First, we explore the performance of DP-HP algorithm on eight different imbalanced tabular datasets with both numerical and categorical input features. The numerical features on those tabular datasets can be either discrete (e.g. age in years) or continuous (e.g. height) and the categorical ones may be binary (e.g. drug vs placebo group) or multi-class (e.g. nationality). The datasets are described in detail in Supplementary Sec.\footnote{For the Cervical dataset, the non-privately generated samples by DP-MERF and DP-HP give better results than the baseline trained with real data. This may be due to the fact that the dataset is relatively small which can lead to overfitting. The generating samples by DP-MERF and DP-HP could bring a regularizing effect, which improves the performance as a consequence.} 2.

As an evaluation metric, we use ROC (area under the receiver characteristics curve) and PRC (area under the precision recall curve) for datasets with binary labels; and F1 score for dataset with multi-class labels. Table 2 shows the average over the 12 classifiers trained on the generated samples (also averaged over 5 independent seeds), where over-all DP-HP outperforms the other methods in both the private and non-private settings, followed by DP-MERF. As a baseline, we consider the classifiers trained on real data samples.

In Table 1 our method needs a significantly lower number of features compared to DP-MERF for a better accuracy. Strikingly, in the private settings, we often needed only an order of HP less than 10, which greatly reduced the effective sensitivity of the feature map.

**Image data.** We follow previous work in testing our method on image datasets MNIST \cite{16} (license: CC BY-SA 3.0) and FashionMNIST \cite{35} (license: MIT). Both datasets contain 60000 images from 10 different balanced classes. We test both fully connected and convolutional generator networks and find that the former works better for MNIST, while the latter model achieves better scores on FashionMNIST. For the experimental setup of DP-HP on the image datasets see Table 1 in Supplementary Sec.\footnote{auto-dp is released here: \url{https://github.com/yuxiangw/autodp} (Apache License 2.0)} 2. A qualitative sample of the generated images for DP-HP and comparison methods is shown in Fig. 3. Comparing DP-HP plots for the two datasets highlights the smoothness bias induced by the CNN generator on FashionMNIST. Based on qualitative inspection we observe that GS-WGAN produces the cleanest samples. However, we see below that DP-HP outperforms GS-WGAN on downstream tasks. This can be explained either by a lack of sample diversity in GS-WGAN or by a mismatch between what features humans find important on visual inspection and the features that are actually relevant in a discrimination task.

In Fig. 4 we compare the test accuracy on real image data based on private synthetic samples from DP-GAN, DP-CGAN, GS-WGAN, DP-MERF and DP-HP generators. As additional baselines we include performance of real data and of full MMD, a non-private generator, which is trained with the MMD estimator in eq. 2 in a mini-batch fashion. DP-HP gives the best accuracy over the other methods.
considered methods followed by DP-MERF but with a considerable difference especially on the MNIST dataset. For GAN-based methods, we use the same weak privacy constraints given in the original papers, because they do not produce meaningful samples at $\epsilon = 1$. Nonetheless, the accuracy these models achieve remains relatively low. Results for individual models for both image datasets can be found in Supplementary Sec. G.

Finally, we show the downstream accuracy for smaller generated datasets down to 60 samples (or 0.1\% of original dataset) in Fig. 4. The points, at which additional generated data does not lead to improved performance, gives us a sense of the redundancy present in the generated data. We observe that all generative models except full MMD see little increase in performance as we increase the number of synthetic data samples to train the classifiers. This indicates that the effective dataset size these methods produce lies only at about 5\% (3k) to 10\% (6k) of the original data. For DP-GAN and DP-CGAN this effect is even more pronounced, showing little to no gain in accuracy after the first 300 to 600 samples respectively on FashionMNIST.

6 Summary and Discussion

We propose a DP data generation framework that improves the privacy-accuracy trade-off using the Hermite polynomials features by reducing the effective sensitivity of the feature map thanks to the orderedness of the polynomial features. In our framework, the sum of Gaussian kernels is chosen for computational tractability in handling high-dimensional data. Our experiments show that the quality of generated data by our method is significantly higher than that by other state-of-the-art methods, in terms of the evaluation on the 12 downstream tasks on 8 tabular datasets and 2 image datasets.

An intriguing future direction would be building a computationally tractable form of the generalized Hermite polynomials. Subsampling input dimensions in every a few learning steps and constructing the feature map on the subsampled input dimensions could be a computationally-tractable alternative, which can capture the dependencies between subsampled input dimensions. However, under this scenario, we would need to pay more privacy cost on releasing the feature map as now we have to construct the feature map multiple times (each time we consider a different subset of input dimension) during the course of training. Finding a good privacy-accuracy trade-off in this context would be a future direction that is worth exploring.
Acknowledgments

The authors thank Wittawat Jitkrittum for helpful discussions. All the authors are supported by the Max Planck Society, the Gibbs Schüle Foundation and the Institutional Strategy of the University of Tübingen (ZUK63) and the German Federal Ministry of Education and Research (BMBF): Tübingen AI Center, FKZ: 01IS18039B. M. Vinaroz, M.Charusaie and F. Harder thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for its support.
References

[1] Nist 2018 differential privacy synthetic data challenge. https://www.nist.gov/ctl/pscr/open-innovation-prize-challenges/past-prize-challenges/2018-differential-privacy-synthetic.

[2] Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, CCS ’16, page 308–318, New York, NY, USA, 2016. Association for Computing Machinery.

[3] Gergely Acs, Luca Melis, Claude Castelluccia, and Emiliano De Cristofaro. Differentially private mixture of generative neural networks. IEEE Transactions on Knowledge and Data Engineering, 31(6):1109–1121, 2018.

[4] Francis Bach. Polynomial magic III : Hermite Polynomials. https://francisbach.com/hermite-polynomials/ 2020.

[5] Matej Balog, Ilya Tolstikhin, and Bernhard Schölkopf. Differentially private database release via kernel mean embeddings. In Proceedings of the 35th International Conference on Machine Learning (ICML), volume 80 of Proceedings of Machine Learning Research, pages 423–431. PMLR, July 2018.

[6] Dingfan Chen, Tribhuvanesh Orekondy, and Mario Fritz. Gs-wgan: A gradient-sanitized approach for learning differentially private generators. In Advances in Neural Information Processing Systems 33, 2020.

[7] Rui Chen, Qian Xiao, Yu Zhang, and Jianliang Xu. Differentially private high-dimensional data publication via sampling-based inference. In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 129–138, 2015.

[8] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[9] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In Eurocrypt, volume 4004, pages 486–503. Springer, 2006.

[10] Lorenzo Frigerio, Anderson Santana de Oliveira, Laurent Gomez, and Patrick Duverger. Differentially private generative adversarial networks for time series, continuous, and discrete open data. In ICT Systems Security and Privacy Protection - 34th IFIP TC 11 International Conference, SEC 2019, Lisbon, Portugal, June 25-27, 2019, Proceedings, pages 151–164, 2019.

[11] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. arXiv preprint arXiv:1406.2661, 2014.

[12] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. Journal of Machine Learning Research, 13(Mar):723–773, 2012.

[13] Frederik Harder, Kamil Adamczewski, and Mijung Park. DP-MERF: Differentially private mean embeddings with random features for practical privacy-preserving data generation. In Arindam Banerjee and Kenji Fukumizu, editors, Proceedings of The 24th International Conference on Artificial Intelligence and Statistics, volume 130 of Proceedings of Machine Learning Research, pages 1819–1827. PMLR, 13–15 Apr 2021.

[14] Moritz Hardt, Katrina Ligett, and Frank Mcsherry. A simple and practical algorithm for differentially private data release. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, Advances in Neural Information Processing Systems 25, pages 2339–2347. Curran Associates, Inc., 2012.

[15] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114, 2013.

[16] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist, 2, 2010.

[17] Chun-Liang Li, Wei-Cheng Chang, Yu Cheng, Yiming Yang, and Barnabas Poczos. Mmd gan: Towards deeper understanding of moment matching network. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, Advances in Neural Information Processing Systems 30, pages 2203–2213. Curran Associates, Inc., 2017.
[18] Ilya Mironov. Rényi differential privacy. In 2017 IEEE 30th Computer Security Foundations Symposium (CSF), pages 263–275. IEEE, 2017.

[19] Noman Mohammed, Rui Chen, Benjamin C.M. Fung, and Philip S. Yu. Differentially private data release for data mining. In Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’11, pages 493–501, New York, NY, USA, 2011. ACM.

[20] Nicolas Papernot, Martin Abadi, Úlfar Erlingsson, Ian Goodfellow, and Kunal Talwar. Semi-supervised Knowledge Transfer for Deep Learning from Private Training Data. In Proceedings of the International Conference on Learning Representations (ICLR), April 2017.

[21] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett, editors, Advances in Neural Information Processing Systems 32, pages 8024–8035. Curran Associates, Inc., 2019.

[22] Wahbeh Qardaji, Weining Yang, and Ninghui Li. Priview: practical differentially private release of marginal contingency tables. In Proceedings of the 2014 ACM SIGMOD international conference on Management of data, pages 1435–1446, 2014.

[23] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In Advances in neural information processing systems, pages 1177–1184, 2008.

[24] Carl Edward Rasmussen and Christopher K. I. Williams. Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning). The MIT Press, 2005.

[25] Walter Rudin. Fourier Analysis on Groups: Interscience Tracts in Pure and Applied Mathematics, No. 12. Literary Licensing, LLC, 2013.

[26] Kanthi Sarpatwar, Karthikkeyan Shanmugam, Venkata Sitaramagiridharganesh Ganapavarapu, Ashish Jagmohan, and Roman Vaculin. Differentially private distributed data summarization under covariate shift. In Advances in Neural Information Processing Systems, pages 14432–14442, 2019.

[27] David Slepian. On the symmetrized kronnecker power of a matrix and extensions of mehler’s formula for hermite polynomials. SIAM Journal on Mathematical Analysis, 3(4):606–616, 1972.

[28] A. Smola, A. Gretton, L. Song, and B. Schölkopf. A Hilbert space embedding for distributions. In ALT, pages 13–31, 2007.

[29] Alex J Smola and Bernhard Schölkopf. Learning with kernels, volume 4. Citeseer, 1998.

[30] Joshua Snoke and Aleksandra Slavković. pmse mechanism: differentially private synthetic data with maximal distributional similarity. In International Conference on Privacy in Statistical Databases, pages 138–159. Springer, 2018.

[31] Bharath K Sriperumbudur, Kenji Fukumizu, and Gert RG Lanckriet. Universality, characteristic kernels and rkhs embedding of measures. Journal of Machine Learning Research, 12(7), 2011.

[32] Reihaneh Torkzadehmahani, Peter Kairouz, and Benedict Paten. Dp-cgan: Differentially private synthetic data and label generation. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, June 2019.

[33] Aleksei Triastcyn and Boi Faltings. Bayesian differential privacy for machine learning. In International Conference on Machine Learning, pages 9583–9592. PMLR, 2020.

[34] Yu-Xiang Wang, Borja Balle, and Shiva Prasad Kasiviswanathan. Subsampled rényi differential privacy and analytical moments accountant. PMLR, 2019.

[35] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint arXiv:1708.07747, 2017.

[36] Yonghui Xiao, Li Xiong, and Chun Yuan. Differentially private data release through multidimensional partitioning. In Willem Jonker and Milan Petković, editors, Secure Data Management, pages 150–168, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.
[37] Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. Differentially private generative adversarial network. *CoRR*, abs/1802.06739, 2018.

[38] Jinsung Yoon, James Jordon, and Mihaela van der Schaar. PATE-GAN: Generating synthetic data with differential privacy guarantees. In *International Conference on Learning Representations*, 2019.

[39] Jun Zhang, Graham Cormode, Cecilia M Procopiuc, Divesh Srivastava, and Xiaokui Xiao. Privbayes: Private data release via bayesian networks. *ACM Transactions on Database Systems (TODS)*, 42(4):1–41, 2017.

[40] Zhikun Zhang, Tianhao Wang, Ninghui Li, Jean Honorio, Michael Backes, Shibo He, Jiming Chen, and Yang Zhang. Privsyn: Differentially private data synthesis. In *30th {USENIX} Security Symposium ({USENIX} Security 21)*, 2021.

[41] Huaiyu Zhu, Christopher KI Williams, Richard Rohwer, and Michal Morciniec. Gaussian regression and optimal finite dimensional linear models. 1997.

[42] T. Zhu, G. Li, W. Zhou, and P. S. Yu. Differentially private data publishing and analysis: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 29(8):1619–1638, August 2017.
Supplementary Material: 
Hermite Polynomials for private data generation

A Mercer’s theorem and the generalized Hermite polynomials

We first review Mercer’s theorem, which is a fundamental theorem on how can we find the approximation of a kernel via finite-dimensional feature maps.

**Theorem A.1 ([29] Theorem 2.10 and Proposition 2.11).** Suppose \( k \in L_\infty(X^2) \), is a symmetric real-valued function, for a non-empty set \( X \), such that the integral operator \( T_k f(x) = \int_X k(x,x') f(x') d\mu(x') \) is positive definite. Let \( \psi_j \in L_2(X) \) be the normalized orthogonal eigenfunctions of \( T_k \) associated with the eigenvalues \( \lambda_j > 0 \), sorted in non-increasing order, then

1. \( (\lambda_j)_j \in \ell_1 \),

2. \( k(x,x') = \sum_{j=1}^{N} \lambda_j \psi_j(x)\psi_j(x') \) holds for almost all \( (x,x') \). Either \( N_H \in \mathbb{N} \), or \( N_H = \infty \); in the latter case, the series converge absolutely and uniformly for almost all \( (x,x') \).

Furthermore, for every \( \epsilon > 0 \), there exists \( n \) such that

\[
|k(x,x') - \sum_{j=1}^{n} \lambda_j \psi_j(x)\psi_j(x')| < \epsilon, \tag{13}
\]

for almost all \( x,x' \in X \).

This theorem states that one can define a feature map

\[
\Phi_n(x) = \begin{bmatrix} \sqrt{\lambda_1}\psi_1(x), \ldots, \sqrt{\lambda_n}\psi_n(x) \end{bmatrix}^T, \tag{14}
\]

such that the Euclidean inner product \( \langle \Phi(x), \Phi(x') \rangle \) approximates \( k(x,x') \) up to an arbitrarily small factor \( \epsilon \).

By means of uniform convergence in Mercer’s theorem, we can prove the convergence of the approximated MMD using the following lemma.

**Lemma A.1.** Let \( \mathcal{H} \) be an RKHS that is generated by the kernel \( k(\cdot,\cdot) \), and let \( \tilde{\mathcal{H}}_n \) be an RKHS with a kernel \( k_n(x,y) \) that can uniformly approximate \( k(x,y) \). Then, for a positive real value \( \epsilon \), there exists \( n \), such that for every pair of distributions \( P,Q \), we have

\[
|\text{MMD}_\mathcal{H}^2(P,Q) - \text{MMD}_{\tilde{\mathcal{H}}_n}^2(P,Q)| < \epsilon. \tag{15}
\]

**Proof.** Firstly, using Theorem [A.1] we can find \( n \) such that \( |k(x,y) - \langle \Phi_n(x),\Phi_n(y) \rangle| < \frac{\epsilon}{4} \).

We define the RKHS \( \tilde{\mathcal{H}}_n \) as the space of functions spanned by \( \Phi_n(\cdot) \). Next, we rewrite \( \text{MMD}_\mathcal{H}^2(P,Q) - \text{MMD}_{\tilde{\mathcal{H}}_n}^2(P,Q) \), using the definition of MMD in Section 2.1 as

\[
\text{MMD}_\mathcal{H}^2(P,Q) - \text{MMD}_{\tilde{\mathcal{H}}_n}^2(P,Q) = \mathbb{E}_{x,x' \sim P}[k(x,x')] + \mathbb{E}_{y,y' \sim Q}[k(y,y')] - 2\mathbb{E}_{x \sim P,y \sim Q}[k(x,y)]
- \mathbb{E}_{x,x' \sim P}[\langle \Phi_n(x),\Phi_n(x') \rangle] + \mathbb{E}_{y,y' \sim Q}[\langle \Phi_n(y),\Phi_n(y') \rangle] - 2\mathbb{E}_{x \sim P,y \sim Q}[\langle \Phi_n(x),\Phi_n(y) \rangle]. \tag{16}
\]
Therefore, we can bound \(|\text{MMD}_{\mathcal{H}}^2(P, Q) - \text{MMD}_{\mathcal{H}_n}^2(P, Q)|\) as

\[
|\text{MMD}_{\mathcal{H}}^2(P, Q) - \text{MMD}_{\mathcal{H}_n}^2(P, Q)| \leq |\mathbb{E}_{x,x' \sim P} [k(x, x')] - \mathbb{E}_{x,x' \sim P} \left( \langle \Phi_n(x), \Phi_n(x') \rangle \right)|
\]

\[
+ |\mathbb{E}_{y,y' \sim Q} [k(y, y')] - \mathbb{E}_{y,y' \sim P} \left( \langle \Phi_n(y), \Phi_n(y') \rangle \right)|
\]

\[
+ 2|\mathbb{E}_{x,y \sim P,Q} [k(x, y)] - \mathbb{E}_{x,y \sim P,Q} \left( \langle \Phi_n(x), \Phi_n(y) \rangle \right)|
\]

\[
\leq \mathbb{E}_{x,x' \sim P} \left| k(x, x') - \langle \Phi_n(x), \Phi_n(x') \rangle \right| + \mathbb{E}_{y,y' \sim Q} \left| k(y, y') - \langle \Phi_n(y), \Phi_n(y') \rangle \right|
\]

\[
+ 2\mathbb{E}_{x,y \sim P,Q} \left| k(x, y) - \langle \Phi_n(x), \Phi_n(y) \rangle \right|
\]

\[
\leq \mathbb{E}_{x,x' \sim P} \left[ \frac{\epsilon}{4} + \mathbb{E}_{y,y' \sim Q} \left[ \frac{\epsilon}{4} \right] + 2\mathbb{E}_{x,y \sim P,Q} \left[ \frac{\epsilon}{4} \right] = \epsilon, \right. \tag{17}
\]

where (a) holds because of triangle inequality, (b) is followed by Tonelli’s theorem and Jensen’s inequality for absolute value function, and (c) is correct because of the choice of \(n\) as mentioned earlier in the proof.

As a result of the above theorems, we can approximate the MMD in RKHS \(\mathcal{H}_k\) for a kernel \(k(\cdot, \cdot)\) via MMD in RKHS \(\mathcal{H}_n \subseteq \mathbb{R}^n\) that is spanned by the first \(n\) eigenfunctions weighted by square roots of eigenvalues of the kernel \(k(\cdot, \cdot)\). Therefore, in the following section, we focus on finding the eigenfunctions/eigenvalues of a multivariate Gaussian kernel.

### A.1 Generalized Mehler’s approximation

As we have already seen in eq. \([5\) Mehler’s theorem provides us with an approximation of a one-dimensional Gaussian kernel in terms of Hermite polynomials. To generalize Mehler’s theorem to a uniform convergence regime (that enables us to approximate MMD via such feature maps as shown in Lemma \([A.1\), and for a multivariate Gaussian kernel, we make use of the following theorem.

**Theorem A.2** (\([27\), Section 6). Let the joint Gaussian density kernel \(k(x, y, C) : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}\) be

\[
k(x, y, C) = \frac{1}{(2\pi)^{n/2} |C|^{1/2}} \exp \left( - \frac{1}{2} [x, -y]C^{-1}[x, -y]^T \right), \tag{18}
\]

where \(C\) is a positive-definite matrix as

\[
C = \begin{bmatrix} C_{11} & C_{12} \\ C_{12}^T & C_{22} \end{bmatrix}, \tag{19}
\]

in which \(C_{ij} \in \mathbb{R}^{n \times n}\) for \(i, j \in \{1, 2\}\), and \(C_{11} = C_{22}\). Further, let the integral operator be defined with respect to a measure with density

\[
w(x) = \frac{1}{\int k(x, y, C) dy}.
\]

Then, the orthonormal eigenfunctions and eigenvalues for such kernel are

\[
\psi_k(x) = \sum_{1||k||_1 = ||k||_1} \left( \sigma_{||k||_1} (P)^{-1} \right) \phi_1(x; C_{11}) \sqrt{\prod_{i=1}^{||k||_1} k_i!}, \tag{21}
\]

and

\[
\lambda_k = \prod_{i=1}^{n} \frac{\sigma_i^{k_i/2}}{k_i!}. \tag{22}
\]
Here, \( \sigma_p(A) \) is symmetrized Kronecker power of a matrix \( A \), defined as

\[
(\sigma_{||k||_1}(A))_{kl} = \left( \prod_{i=1}^{n} k_i! \right)^{-1} \sum_{M \in \mathbb{R}^{n \times n} : M_{kl} = k_i, l_k = m} \prod_{ij} A_{ij}^{M_{ij}} \prod_{ij} M_{ij}^{-1},
\]

for two \( n \)-dimensional vectors \( k \) and \( l \) with \( ||k||_1 = ||l||_1 \), the vector \( e \) (the matrix \( P \) ) is formed by eigenvalues (eigenvectors) of \( C_{11}^{-1} C_{12} \), and \( \varphi_1(\mathbf{x}, A) \) is generalized Hermite functions defined as

\[
\varphi_1(\mathbf{x}, A) = \frac{1}{(2\pi)^{n/2}|A|^{1/2}} \partial_{x_1}^{\alpha_1} \cdots \partial_{x_n}^{\alpha_n} \exp \left( -\frac{1}{2} \mathbf{x}^T A^{-1} \mathbf{x} \right).
\]

The above theorem provides us with eigenfunctions/eigenvalues of a joint Gaussian density function. We utilize this theorem to approximate Mahalanobis kernels (i.e., a generalization of Gaussian radial basis kernels where \( A = cI_n \)) via Hermite polynomials as follow.

**Proposition A.3.** A Mahalanobis kernel \( k(\mathbf{x}, y, A) : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R} \) defined as

\[
k(\mathbf{x}, y, A) = \exp \left( - (\mathbf{x} - \mathbf{y})^T A(\mathbf{x} - \mathbf{y})^T \right)
\]

can be uniformly approximated as

\[
k(\mathbf{x}, y, A) \simeq \left\langle \Phi_N \left( \sqrt{\frac{\alpha^2 - 1}{\alpha}} \sqrt{A} \mathbf{x} \right), \Phi_N \left( \sqrt{\frac{\alpha^2 - 1}{\alpha}} \sqrt{A} \mathbf{y} \right) \right\rangle,
\]

where \( \Phi(x) \in N^D \) is defined as a tensor product

\[
\Phi_N(x) = \bigotimes_{i=1}^{n} [\varphi_{\alpha_i}(x_i)]_{k_i = 1}^{N_i},
\]

where

\[
\varphi_{\alpha_i}(x_i) = \left( \frac{(\alpha^2 - 1)\alpha^{-k_i}}{\alpha^2 k_i!} \right)^{1/4} \exp \left( -\alpha \frac{x_i^2}{\alpha + 1} \right) H_{k_i}(x_i).
\]

**Remark 1.** Using Proposition A.3 and Lemma A.7 we can show that the MMD based on the tensor feature map in eq. 26 and between any two distributions approximates the real MMD based on Gaussian kernel with Mahalanobis norm.

**Proof of Proposition A.3** Let \( C = \begin{bmatrix} \frac{1}{\alpha} I_n & \frac{1}{\alpha} I_n \\ \frac{1}{\alpha} I_n & \frac{1}{\alpha} I_n \end{bmatrix} \), or equivalently \( C^{-1} = \begin{bmatrix} \frac{2\alpha^2}{\alpha^2 - 1} I_n & -\frac{2\alpha^2}{\alpha^2 - 1} I_n \\ -\frac{2\alpha^2}{\alpha^2 - 1} I_n & \frac{2\alpha^2}{\alpha^2 - 1} I_n \end{bmatrix} \), for \( \alpha \in [1, \infty) \). Since \( C \) is positive-definite, we can define a Gaussian kernel density as

\[
k(\mathbf{x}, y, C) = \frac{1}{|(2\pi)^{n/2}|} \exp \left( -\frac{\alpha^2}{\alpha^2 - 1} \|\mathbf{x}\|^2 - \frac{\alpha^2}{\alpha^2 - 1} \|\mathbf{y}\|^2 + \frac{2\alpha}{\alpha^2 - 1} \mathbf{x} \cdot \mathbf{y} \right).
\]

Moreover, we can calculate the integration over all values of \( y \) as

\[
\int k(\mathbf{x}, y, C)dy = \int \exp \left( -\frac{\|\mathbf{x}\|^2}{(2\pi)^{n/2}} \right) \exp \left( -\frac{\|\mathbf{y} - \mathbf{x}\|^2}{(\alpha^2 - 1)} \right) dy
\]

\[
= \exp \left( -\frac{\|\mathbf{x}\|^2}{(\alpha^2 - 1)} \right).
\]

Next, by setting \( w(x) = \int k(\mathbf{x}, y, C)dy \) and using Theorem A.2 we have

\[
\int \frac{1}{|(2\pi)^{n/2} \alpha_i^{n/2}|} \psi_k(\mathbf{x}) \exp \left( -\frac{\|\mathbf{y} - \mathbf{x}\|^2}{\alpha^2 - 1} \right) dx = \lambda_k \psi_k(\mathbf{y}).
\]

16
Now to find the eigenfunctions of the Gaussian kernel $k'(x, y) = \exp\left(-\frac{\alpha||x - y||^2}{\alpha^2 - 1}\right)$, we let $\psi_k(x) = \psi_k(x) \exp\left(\frac{\alpha}{\alpha^2 - 1}||x||^2\right)$ and let the weight function be $w'(x) = (\pi)^{n/2} \exp\left(-\frac{(\alpha-1)}{\alpha+1}||x||^2\right)$. As a result of such assumptions, we see that

$$\int \psi_k'(x) k'(x, y) w'(x) dx = \int (\pi)^{n/2} \psi_k(x) \exp\left(-\frac{1}{\alpha^2 - 1}||x||^2 - \frac{\alpha}{\alpha^2 - 1}||y||^2 + \frac{2\alpha}{\alpha^2 - 1} x \cdot y^T\right) dx$$

$$= (\pi)^{n/2} \exp\left(\frac{\alpha}{\alpha + 1}||y||^2\right) \int \psi_k(x) \exp\left(-\frac{||\alpha y - x||^2}{\alpha^2 - 1}\right) dx$$

$$= (\pi)^{n/2} \exp\left(\frac{\alpha}{\alpha + 1}||y||^2\right) \sqrt{\lambda_k} \psi_k(y) \left(\frac{\pi (\alpha^2 - 1)}{\alpha^2}\right)^{n/2}$$

$$= (\pi)^n \left(\frac{\alpha - 1}{\alpha^2}\right)^{n/2} \lambda_k \psi_k'(y),$$

where (a) holds because of eq. [31] and (b) is followed by the definition of $\psi_k'(y)$. As a result, $\psi_k(x)$ is an eigenfunction of the integral operator with kernel $k'(x, y)$ and with weight function $w'(x)$.

Equation eq. [35] shows that the eigenvalue of $k'(x, y)$ corresponding to $\psi_k(x)$ is as

$$\lambda_k' = (\pi)^n \left(\frac{\alpha^2 - 1}{\alpha^2}\right)^{n/2} \lambda_k$$

Now we show that such eigenfunctions are orthonormal. Deploying the idea in eq. [35], for two eigenfunctions $\psi_k(x)$ and $\psi_l(x)$ for fixed vectors $k, l \in \mathbb{N}^n$, we have

$$\int \psi_k'(y) \psi_l'(y) w'(y) dy = \int \psi_k(y) \psi_l(y) \frac{\pi^{n/2}}{\exp\left(-||x||^2\right)} dy$$

$$= \int \psi_k(y) \psi_l(y) w(y) = \delta[l - k]$$

where (a) is followed by the definition of eigenfunctions $\psi_k'(\cdot), \psi_l'(\cdot)$ and the definition of weight function $w'(x)$, (b) is due to the definition of $w(x)$ and eq. [30] and (c) holds because of orthonormality of $\psi_k$s as a result of Theorem [A.2].

Further, in this case we have $C_{11}^{-1} C_{12} = \frac{1}{\alpha} I_n$, or equivalently $P = I_n$ and $e = \frac{1}{\alpha} e_n$. Hence, firstly using eq. [22], one can see that

$$\lambda_k = \alpha^{-\|k\|/2}.$$  

(39)

Secondly, in finding symmetrized Kronecker power $\sigma_{\|k\|, l}(P)$ in eq. [23] for non-diagonal matrices $M$, the term $\prod_{ij} P_{ij}^{M_{ij}} = 0$. Further, for a diagonal matrix $M$, we have $M I_n = I_n M$. This induces the fact that

$$\sigma_{\|k\|, l}(P) = \begin{cases} 0 & k \neq 1, \\ 1 & k = 1. \end{cases}$$

(40)

This shows that

$$\psi_1(x) = \frac{\varphi_1(x)}{\sqrt{\prod_{i=1}^{n} l_i}}$$

(41)

To find the formulation of eigenfunction $\psi_k(x)$, we can rewrite the term $\varphi_1(x, C_{11})$ in eq. [21] for $C_{11} = \frac{1}{2} I_n$ as

$$\varphi_1(x, I) = \frac{1}{(\pi)^{n/2}} \frac{\partial^{\|l\|}_1}{\partial x_1^{l_1} \ldots \partial x_n^{l_n}} \exp\left(-\sum_{i=1}^{n} x_i^2\right).$$

(42)
We note that the exponential function can be written as the product of functions that are only dependent on one variable $x_i$ for $i \in [n]$. Hence, we can rephrase eq. 42 as a product of the derivative of each function as

$$\varphi_I(x, I) = \prod_{i=1}^{n} \frac{1}{\sqrt{\pi}} \frac{\partial^{i}}{\partial^{i} x_i} \exp \left(-x_i^2\right). \tag{43}$$

As a result of this equation and the definition of Hermite functions in one dimension, we have

$$\varphi_I(x, I) = \exp\left(-\|x\|^2\right) \prod_{i=1}^{n} H_{k_i}(x_i) \tag{44}$$

Hence, we can calculate $\psi^\prime_k(x)$ as

$$\psi^\prime_k(x) = \frac{1}{\sqrt{\pi}^n \prod_{i=1}^{n} k_i!} \exp\left(-\|x\|^2\right) \prod_{i=1}^{n} H_{k_i}(x_i). \tag{45}$$

Using above discussion, we see that $k$-th element $[\Phi_N(x)]_k$ of the tensor $\Phi_N(x)$, which is defined in the proposition statement, is equal to

$$[\Phi_N(x)]_k = \sqrt{\lambda_k} \psi^\prime_k(x). \tag{46}$$

This fact and Theorem A.1 concludes that we can uniformly approximate $k(x, y)$ as

$$k(x, y) = \langle \Phi_N(x), \Phi_N(y) \rangle. \tag{47}$$

Further, for any positive-definite matrix $A$, since the singular values of $\sqrt{\alpha - 1} \sqrt{A}$ are bounded, one can uniformly approximate $k''(x, y) := \exp \left(- (x - y)A(x - y)^T \right) = \sqrt{\alpha - 1} \sqrt{A}x, \sqrt{\alpha - 1} \sqrt{A}y)$ as

$$k''(x, y) \simeq \langle \Phi_N(\sqrt{\frac{\alpha - 1}{\alpha}} \sqrt{A}x), \Phi_N(\sqrt{\frac{\alpha - 1}{\alpha}} \sqrt{A}y) \rangle \tag{48}$$

\[ \square \]

### B Challenges in using the outer product kernel

Even though Proposition A.3 promises to retrieve almost all information regarding a probability measure via a finite-dimensional mean embedding, using such mean embedding induces several issues that we mention as follows:

- Firstly, finding such mean embedding in high dimension consumes huge amount of memory. For $D$-dimensional vector $x$, to find tensor feature map defined in Proposition A.3 of order $T$ for every dimension, we need $T^D$ slices of memory. Such amount of memory is cumbersome for the datasets in which we train the models. As an instance, having the tensor mean embedding for MNIST dataset with order 2 in each dimension costs $2^{784} \approx 10^{236}$ slices of memory.

- Another problem which is also induced by memory consumption of generalized Hermite polynomials is regarding the privacy-accuracy trade-off. Formally, to have the Frobenius norm of tensor mean embedding be less than 1, it is meaningful to have the variance of perturbation less than 1 as well. Using this fact and bounded sensitivity of $\frac{1}{m}$ for $m$ number of samples, it is easy to show that we have $\frac{m}{T^D} > O\left(\log \frac{1}{\delta} \right)$ that implies a slight increase of $D$ lower-bounds $\epsilon$ with a very large value. In conclusion, preserving privacy and having a mean embedding with norm less than 1 is impossible in slightly high dimensions.

- The curse of dimensionality leads to the under-fitting, even in the cases that memory is affordable and privacy is preserved to a certain point. As an instance, in Table 3 two generative models based on generalized Hermite polynomials and DP-HP are compared. One can see that DP-HP mostly gives much better accuracy on various downstream tasks.
### Table 3: Performance comparison on Adult dataset for various downstream tasks. The average over five independent runs.

| Model                        | Generalized Hermite (non-priv) | DP-HP (non-priv) |
|------------------------------|--------------------------------|-----------------|
|                              | ROC | PRC | ROC | PRC |
| Logistic Regression          | 0.628 | 0.587 | 0.626 | 0.583 |
| Gaussian Naive Bayes         | 0.482 | 0.498 | 0.517 | 0.524 |
| Bernoulli Naive Bayes        | 0.541 | 0.529 | 0.713 | 0.651 |
| LinearSVC                    | 0.643 | 0.596 | 0.621 | 0.580 |
| Decision Tree                | 0.620 | 0.578 | 0.683 | 0.629 |
| LDA                          | 0.618 | 0.578 | 0.608 | 0.569 |
| Bagging                      | 0.618 | 0.575 | 0.676 | 0.621 |
| Adaboost                     | 0.648 | 0.596 | 0.681 | 0.625 |
| Gradient Boosting            | 0.634 | 0.585 | 0.673 | 0.617 |
| Random Forest                | 0.635 | 0.586 | 0.705 | 0.646 |
| MLP                          | 0.705 | 0.652 | 0.667 | 0.616 |
| XGB                          | 0.638 | 0.589 | 0.685 | 0.629 |
| Average                      | 0.618 | 0.579 | 0.655 | 0.608 |

### C Sum-kernel upper-bound

Instead of using Generalized Hermite mean embedding which takes a huge amount of memory, one could use an upper bound to the joint Gaussian kernel. We use the inequality of arithmetic and geometric means to prove that.

\[
k(x, y) = \exp \left( -\frac{1}{2\ell^2} (x - y)^T (x - y) \right) = \exp \left( -\frac{1}{2\ell^2} \sum_{d=1}^{D} (x_d - y_d)^2 \right) \quad (49)
\]

\[
= \prod_{d=1}^{D} \exp \left( -\frac{1}{2\ell^2} (x_d - y_d)^2 \right) \quad (50)
\]

\[
\leq \left(\frac{a}{D}\right) \sum_{d=1}^{D} \exp \left( -\frac{D}{2\ell^2} (x_d - y_d)^2 \right) \quad (51)
\]

\[
= \frac{1}{D} \sum_{d=1}^{D} k_{X_d}(x_d, y_d), \quad (52)
\]

where \((a)\) holds due to inequality of arithmetic and geometric means (AM-GM), and \(k_{X_d}(\cdot, \cdot)\) is defined as

\[
k_{X_d}(x_d, y_d) := \exp \left( -\frac{D}{2\ell^2} (x_d - y_d)^2 \right). \quad (53)
\]

Next, we approximate such kernel via an inner-product of the feature maps

\[
\phi_C(x) = \begin{bmatrix}
\phi_{HP,1}^{(C)}(x_1)/\sqrt{D} \\
\phi_{HP,2}^{(C)}(x_2)/\sqrt{D} \\
\vdots \\
\phi_{HP,D}^{(C)}(x_D)/\sqrt{D}
\end{bmatrix} \in \mathbb{R}^{((C+1) \cdot D) \times 1}. \quad (54)
\]

Although such feature maps are not designed to catch correlation among dimensions, they provide us with a guarantee on marginal distributions as follows.

**Lemma C.1.** Define \(k_{X_i}(\cdot, \cdot)\) as in eq. (53) and define \(\phi_C(x)\) as in eq. (54). For \(\epsilon \in \mathbb{R}^+\), there exists \(N\) such that for \(C \geq N\) we have

\[
\|E_{x \sim P} [\phi_C(x)] - E_{y \sim Q} [\phi_C(y)]\|_2 \leq \epsilon \Rightarrow \text{MMD}_{k_{X_i} \cdot P_i, Q_i} \leq \sqrt{D + 1}\epsilon \text{ for every } i \in \{1, \ldots, D\}, \text{ and}
\]
• MMD_{k,X}(P_i, Q_i) \leq \epsilon \text{ for every } i \in \{1, \ldots, D\} \implies \\
\|E_{x \sim P}[\phi_C(x)] - E_{y \sim Q}[\phi_C(y)]\| \leq \sqrt{2\epsilon},

where \( P_i \) and \( Q_i \) are marginal probability distributions corresponding to \( P \) and \( Q \), respectively.

**Proof.** Since \( \phi^{(C)}_{HP,i}(x_i) \) has the certain form as in Theorem A.1, then Lemma A.1 shows that we can use such feature maps to uniformly approximate the MMD in an RKHS based on the kernel \( k_i(x_i, y_i) = \exp (-\frac{1}{\sigma^2}(x_i - y_i)^2) \). As a result, there exists \( N \) such that for \( C \geq N \), we have

\[
\|E_{x_i \sim P_i}[\phi^{(C)}_{HP,i}(x_i)] - E_{y_i \sim Q_i}[\phi^{(C)}_{HP,i}(y_i)]\|^2_2 - \text{MMD}^2_{k,X}(P_i, Q_i) \leq D^2. \tag{55}
\]

Now we prove the first part. Knowing

\[
\|E_{x_i \sim P}[\phi_C(x)] - E_{y \sim Q}[\phi_C(y)]\|_2 \leq \epsilon, \tag{56}
\]

and by the definition of \( \phi_C(\cdot) \), we deduce that

\[
\|E_{x_i \sim P_i}[\phi^{(C)}_{HP,i}(x_i)] - E_{y_i \sim Q_i}[\phi^{(C)}_{HP,i}(y_i)]\|^2_2 \leq \epsilon^2. \tag{57}
\]

Using this and eq. 55, we can prove the first part.

Inversely, by setting MMD_{k,X}(P_i, Q_i) \leq \epsilon and eq. 55, one sees that

\[
\|E_{x_i \sim P_i}[\phi^{(C)}_{HP,i}(x_i)] - E_{y_i \sim Q_i}[\phi^{(C)}_{HP,i}(y_i)]\|_2 \leq \sqrt{2\epsilon}. \tag{58}
\]

This coupled with the definition of \( \Phi_C \) completes the second part of lemma. □

**D \ \ \phi \ \ \text{Recursion}**

\[
\phi_{k+1}(x) = ((1+\rho)(1-\rho))^{\frac{1}{2}} \frac{\rho^{k+1}}{\sqrt{2^k(k+1)!}} H_{k+1}(x) \exp \left(-\frac{\rho}{\rho+1} x^2 \right), \text{ by definition}
\]

\[
= ((1+\rho)(1-\rho))^{\frac{1}{2}} \frac{\rho^{k+1}}{\sqrt{2^k(k+1)!}} [2x H_k(x) - 2k H_{k-1}(x)] \exp \left(-\frac{\rho}{\rho+1} x^2 \right),
\]

\[
= \frac{\sqrt{\rho}}{\sqrt{2(k+1)}} 2x \phi_k(x) - \frac{\rho}{\sqrt{k(k+1)}} \phi_{k-1}(x). \tag{59}
\]

**E \ \ \text{Sensitivity of mean embedding}**

Here we derive the sensitivity of the mean embedding under the combination of the sum and product kernels.

\[
S_{\mu_P} = \max_{D,D'} \|\mu_P(D) - \mu_P(D')\|_F = \max_{D,D'} \| \frac{1}{m} \sum_{i=1}^m \phi(x_i)f(y_i)^T - \frac{1}{m} \sum_{i=1}^m \phi(x'_i)f(y'_i)^T \|_F
\]

where \( \| \cdot \|_F \) represents the Frobenius norm. Since \( D \) and \( D' \) are neighbouring, then \( m - 1 \) of the summands on each side cancel and we are left with the only distinct datapoints, which we denote as \((x, y)\) and \((x', y')\). We then apply the triangle inequality and the definition of \( f \). As \( y \) is a one-hot vector, all but one column of \( \phi(x)f(y)^T \) are 0, so we omit them in the next step:

\[
S_{\mu_P} = \max_{(x,y),(x',y')} \| \frac{1}{m} \phi(x)f(y)^T - \frac{1}{m} \phi(x')f(y')^T \|_F
\]

\[
\leq \max_{(x,y)} \frac{2}{m} \| \phi(x)f(y)^T \|_F = \max_{x} \frac{\phi(x)}{m} = \frac{\| \phi(x) \|_2}{m}. \tag{60}
\]

We recall the definition of the feature map given in eq. 8.

\[
\| \phi(x) \|_2 = \frac{1}{\sqrt{D}} \left( \sum_{d=1}^D \| \phi^{(C)}_{HP,d}(x_d) \|_2^2 \right)^{\frac{1}{2}}. \tag{61}
\]
To bound $\|\phi(x)\|_2$, and therefore sensitivity $S_{\mu_P}$, we first prove that $\|\phi^{(C)}_{HP,d}(x_d)\|_2^2 \leq 1$. Using Mehler’s formula (see eq. 5), and by plugging in $y = x_d$, one can show that

$$1 = \exp \left( -\frac{\rho}{1 - \rho^2} (x_d - x_d)^2 \right) = \sum_{c=0}^{\infty} \lambda_c f_c(x_d)^2. \quad (62)$$

Using this, we rewrite the infinite sum in terms of the $C$th-order approximation and the rest of summands to show that

$$1 = \sum_{c=0}^{\infty} \lambda_c f_c^2(x_d) = \|\phi^{(C)}_{HP,d}(x_d)\|_2^2 + \sum_{c=C+1}^{\infty} \lambda_c f_c^2(x) \geq \|\phi^{(C)}_{HP,d}(x_d)\|_2^2, \quad (63)$$

where (a) holds because of the definition of $\phi^{(C)}_{HP,d}(x_d)$ in eq. 6; $\|\phi^{(C)}_{HP,d}(x_d)\|_2^2 = \sum_{c=0}^{C} \lambda_c f_c^2(x_d)$, and (b) holds, because $\lambda_c$ and $f_c^2(x)$ are non-negative scalars.

Finally, deploying eq. 60, eq. 61 and eq. 63, we bound the sensitivity as

$$S_{\mu_P} \leq \max_{x} \frac{2}{m} \|\phi(x)\|_2 \leq \frac{2}{m \sqrt{D}} \sqrt{D} = \frac{2}{m}. \quad (64)$$

F Descriptions on the tabular datasets

In this section we give more detailed information about the tabular datasets used in our experiments. Unless otherwise stated, the datasets were obtained from the UCI machine learning repository [8].

Adult

Adult dataset contains personal attributes like age, gender, education, marital status or race from the different dataset participants and their respective income as the label (binarized by a threshold set to 50K). The dataset is publicly available at the UCI machine learning repository at the following link: [https://archive.ics.uci.edu/ml/datasets/adult](https://archive.ics.uci.edu/ml/datasets/adult).

Census

The Census dataset is also a public dataset that can be downloaded via the SDGym package[10]. This is a clear example of an imbalanced dataset since only 12382 of the samples are considered positive out of a total of 199523 samples.

Cervical

The Cervical cancer dataset comprises demographic information, habits, and historic medical records of 858 patients and was created with the goal to identify the cervical cancer risk factors. The original dataset can be found at the following link: [https://archive.ics.uci.edu/ml/datasets/Cervical+cancer+%28Risk+Factors%29](https://archive.ics.uci.edu/ml/datasets/Cervical+cancer+%28Risk+Factors%29).

Covtype

This dataset contains cartographic variables from four wilderness areas located in the Roosevelt National Forest of northern Colorado and the goal is to predict forest cover type from the 7 possible classes. The data is publicly available at [https://archive.ics.uci.edu/ml/datasets/covertype](https://archive.ics.uci.edu/ml/datasets/covertype).

Credit

The Credit Card Fraud Detection dataset contains the categorized information of credit card transactions made by European cardholders during September 2013 and the corresponding label indicating if the transaction was fraudulent or not. The dataset can be found at: [https://www.kaggle.com/mlg-ubl/creditcardfraud](https://www.kaggle.com/mlg-ubl/creditcardfraud). The original dataset has a total number of 284807 samples where only 492 of them are frauds. In our experiments, we descared the feature related to the time the transaction was done. The data is released under a Database Contents License (DbCL) v1.0.

Epileptic

[10]SDGym package website: [https://pypi.org/project/sdgym/](https://pypi.org/project/sdgym/)
The Epileptic Seizure Recognition dataset contains the brain activity measured in terms of the EEG across time. The dataset can be found [here](https://archive.ics.uci.edu/ml/datasets/Epileptic+Seizure+Recognition). The original dataset contains 5 different labels that we binarized into two: seizure (2300 samples) or not seizure (9200 samples).

**Intrusion**

The dataset was used for The Third International Knowledge Discovery and Data Mining Tools Competition held at the Conference on Knowledge Discovery and Data Mining, 1999, and can be found [here](http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html). We used the file named "kddcup.data10percent.gz" that contains the 10% of the original dataset. The goal is to distinguish between intrusion/attack and normal connections categorized in 5 different labels.

**Isolet**

The Isolet dataset contains the features sounds as spectral coefficients, contour features, sonorant features, pre-sonorant features, and post-sonorant features of the different letters on the alphabet as inputs and the corresponding letter as the label. The original dataset can be found [here](https://archive.ics.uci.edu/ml/datasets/isolet). However, in our experiments we considered this dataset as a binary classification task as we only considered the labels as constants or vowels.

Table 4 summarizes the number of samples, labeled classes and type of different inputs (numerical, ordinal or categorical) for each tabular dataset used in our experiments. The content of the table reflects the results after pre-processing or binarizing the corresponding datasets.

| dataset | # samps | # classes | # features |
|---------|---------|-----------|------------|
| isolet  | 4366    | 2         | 617 num    |
| covtype | 406698  | 7         | 10 num, 44 cat |
| epileptic | 11500  | 2         | 178 num    |
| credit  | 284807  | 2         | 29 num     |
| cervical | 753    | 2         | 11 num, 24 cat |
| census  | 199523  | 2         | 7 num, 33 cat |
| adult   | 48842   | 2         | 6 num, 8 cat |
| intrusion | 394021 | 5         | 8 cat, 6 ord, 26 num |

### F.1 Training DP-HP generator

Here we provide the details of the DP-HP training procedure we used on the tabular data experiments. Table 5 shows the Hermite polynomial order, the fraction of dataset used in a batch, the number of epochs and the undersampling rate we used during training for each tabular dataset.

| dataset | hermite order | mini-batch rate | epochs | undersampling rate |
|---------|---------------|-----------------|--------|--------------------|
| adult   | 20            | 0.1             | 100    | 0.3                |
| census  | 10            | 0.1             | 400    | 0.2                |
| cervical | 5            | 1.0             | 80     | 0.4                |
| covtype | 100           | 0.05            | 100    | 0.02               |
| credit  | 20            | 0.5             | 1400   | 0.001              |
| epileptic | 100        | 0.5             | 800    | 0.8                |
| intrusion | 8            | 0.01            | 400    | 0.3                |
| isolet  | 10            | 0.85            | 1400   | 0.35               |

### F.2 DP-HP and DP-MERF comparison

In the main text we compared the Hermite polynomial order used in DP-HP and the number of random features used in DP-MERF for the private escenario for each tabular dataset. Here we add the complementary comparison between Hermite orders and number of features for the respective
methods for the non-private case in Table 6. Note that in the non-private scenario our method also needs a smaller Hermite order compared to the used random features on DP-MERF.

Table 6: Comparison between the order of Hermite Polynomials and the number of Fourier features considered in each tabular dataset for non-private scenario.

|          | DP-MERF | DP-HP |
|----------|---------|-------|
|          | # features | Hermite order |
| adult    | 50000    | 100   |
| census   | 10000    | 100   |
| cervical | 2000     | 100   |
| credit   | 50000    | 100   |
| epileptic| 100000   | 100   |
| islet    | 100000   | 100   |
| covtype  | 1000     | 100   |
| intrusion| 2000     | 100   |

G Image data

G.1 Results by model

In the following we provide a more detailed description of the downstream model’s accuracy over the different methods considered in the image datasets.

Figure 5: We compare the real data test accuracy of models trained on synthetic data for various models: DP-HP, DP-MERF, GS-WGAN and DP-CGAN. As baselines we also include results for real training data and a generator, which is non-privately trained with MMD, listed as "full MMD". We show accuracy sorted by downstream classifier and the mean accuracy across classifiers on the right. Each score is the average of 5 independent runs.

G.2 MNIST and FashionMNIST hyper-parameter settings

Here we give a detailed hyper-parameter setup and the architectures used for generating synthetic samples via DP-HP for MNIST and FashionMNIST datasets in Table 7. Table 8 summarizes the 12 predictive models hyper-parameters setup for the image datasets trained on the generated samples via DP-HP.
Table 7: Hyperparameter settings for image data experiments. All parameters not listed here are used with their default values.

| Parameter       | MNIST    | FashionMNIST |
|-----------------|----------|--------------|
| Hermite order   | 100      | 100          |
| Kernel length   | 0.005    | 0.15         |
| Mini-batch size | 200      | 200          |
| Epochs          | 10       | 10           |
| Learning rate   | 0.01     | 0.01         |
| Architecture    | fully connected | CNN + bilinear upsampling |

Table 8: Hyperparameter settings for downstream models used in image data experiments. Models are taken from the scikit-learn 0.24.2 and xgboost 0.90 python packages and hyperparameters have been set to achieve reasonable accuracies while limiting runtimes. Parameters not listed are kept at their default values.

| Model             | Parameters                                                                 |
|-------------------|-----------------------------------------------------------------------------|
| Logistic Regression| solver: lbfgs, max_iter: 5000, multi_class: auto                            |
| Gaussian Naive Bayes| -                                                                           |
| Bernoulli Naive Bayes| binarize: 0.5                                                               |
| LinearSVC         | max_iter: 10000, tol: 1e-8, loss: hinge                                     |
| Decision Tree     | class_weight: balanced                                                      |
| LDA               | solver: eigen, n_components: 9, tol: 1e-8, shrinkage: 0.5                   |
| Adaboost          | n_estimators: 1000, learning_rate: 0.7, algorithm: SAMME.R                  |
| Bagging           | max_samples: 0.1, n_estimators: 20                                           |
| Random Forest     | n_estimators: 100, class_weight: balanced                                    |
| Gradient Boosting | subsample: 0.1, n_estimators: 50                                            |
| MLP               | -                                                                           |
| XGB               | colsample_bytree: 0.1, objective: multi:softprob, n_estimators: 50          |