Frederik Harder^{*1,2}, **Kamil Adamczewski**^{*1,3}, **Mijung Park**^{1,2}

Supplementary Material: Differentially Private Random Feature Mean Embeddings for Synthetic Data Generation

A Background on distance measures for DP data generation

Many recent papers on DP data generation have utilized the generative adversarial networks (GAN) [11] framework, where a discriminator and a generator play a min-max form of game to optimize for the *Jensen-Shannon divergence* between the true and synthetic data distributions [20, 30, 36]. The Jensen-Shannon divergence belongs to the family of divergences, known as *Ali-Silvey distance*, *Csiszár's ϕ-divergence* [7], defined as $D_\phi(P,Q) = \int_M \phi\left(\frac{P}{Q}\right) dQ$ where M is a measurable space and P, Q are probability distributions. Depending on the form of ϕ, $D_\phi(P,Q)$ recovers popular divergences[5] such as the Kullback-Liebler (KL) divergence ($\phi(t) = t \log t$).

Another popular family of distance measure is *integral probability metrics (IPMs)*, which is defined by $D(P,Q) = \sup_{f\in\mathcal{F}} \left|\int_M f dP - \int_M f dQ\right|$ where $\mathcal{F}$ is a class of real-valued bounded measurable functions on M. Depending on the class of functions, there are several popular choices of IPMs. For instance, when $\mathcal{F} = \{f : \|f\|_L \leq 1\}$, where $\|f\|_L := \sup\{|f(x)-f(y)|/\rho(x,y) : x \neq y \in M\}$ for a metric space (M,ρ), $D(P,Q)$ yields the *Kantorovich* metric, and when M is separable, the Kantorovich metric recovers the *Wasserstein* distance, a popular choice for generative modelling such as Wasserstein-GAN and Wasserstein-VAE [3, 29]. The GAN framework with the Wasserstein distance was also used for DP data generation [35, 9].

As another example of IPMs, when $\mathcal{F} = \{f : \|f\|_{\mathcal{H}} \leq 1\}$, i.e., the function class is a unit ball in reproducing kernel Hilbert space (RKHS) $\mathcal{H}$ associated with a positive-definite kernel k, $D(P,Q)$ yields the *maximum mean discrepancy* (MMD), $MMD(P,Q) = \sup_{f\in\mathcal{F}} \left|\int_M f dP - \int_M f dQ\right|$. In this case finding a supremum is analytically tractable and the solution is represented by the difference in the mean embeddings of each probability measure: $MMD(P,Q) = \|\mu_P - \mu_Q\|_H$, where $\mu_P = \mathbb{E}_{\mathbf{x}\sim\mathbb{P}}[k(\mathbf{x},\cdot)]$ and $\mu_{\mathbb{Q}} = \mathbb{E}_{\mathbf{y}\sim\mathbb{Q}}[k(\mathbf{y},\cdot)]$. For a characteristic kernel k, the squared MMD forms a metric, i.e., $MMD^2 = 0$, if and only if $P = Q$. MMD is also a popular choice for generative modelling in the GAN frameworks [14], as MMD compares two probability measures in terms of all possible moments (no information loss due to a selection of a certain set of moments); and the MMD estimator is in closed form (eq. 2) and easy to compute by the pair-wise evaluations of a kernel function using the points drawn from P and Q.

In this work, we propose to use a particular form of MMD via *random Fourier feature* representations [22] of kernel mean embeddings for DP data generation.

B Derivation of the bound on the expected absolute error

Given the samples drawn from two probability distributions: $X_m = \{x_i\}_{i=1}^m \sim P$ and $X'_n = \{x'_i\}_{i=1}^n \sim Q$, the biased MMD estimator is given by [12]:

$$\widehat{\text{MMD}}^2(X_m, X'_n) = \tfrac{1}{m^2}\sum_{i,j=1}^{m} k(x_i,x_j) + \tfrac{1}{n^2}\sum_{i,j=1}^{n} k(x'_i,x'_j) - \tfrac{2}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n} k(x_i,x'_j). \tag{21}$$

The MMD estimator using the D-dimensional random Fourier features $\hat{\boldsymbol{\phi}}$ for the mean embeddings $\widehat{\boldsymbol{\mu}}_P = \frac{1}{m}\sum_{i=1}^m \hat{\boldsymbol{\phi}}(\mathbf{x}_i)$ and $\widehat{\boldsymbol{\mu}}_Q = \frac{1}{n}\sum_{i=1}^n \hat{\boldsymbol{\phi}}(G_{\boldsymbol{\theta}}(\mathbf{z}_i))$ is defined as

$$\widehat{\text{MMD}}^2_{rf}(P,Q) = \left\|\widehat{\boldsymbol{\mu}}_P - \widehat{\boldsymbol{\mu}}_Q\right\|_2^2. \tag{22}$$

The noisy MMD is given by

$$\widetilde{\text{MMD}}^2_{rf}(P_{\mathbf{x}}, Q_{\tilde{\mathbf{x}}_{\boldsymbol{\theta}}}) = \left\|\widetilde{\boldsymbol{\mu}}_P - \widehat{\boldsymbol{\mu}}_Q\right\|_2^2, \tag{23}$$

[5] See Table 1 in [18] for various ϕ divergences in the context of GANs.

where $\widetilde{\boldsymbol{\mu}}_P$ is given by

$$\widetilde{\boldsymbol{\mu}}_P = \widehat{\boldsymbol{\mu}}_P + \mathbf{n} \tag{24}$$

where $\mathbf{n}$ is a draw from a Gaussian distribution $\mathbf{n} \sim \mathcal{N}(0, \Delta^2_{\widehat{\boldsymbol{\mu}}_P}\sigma^2 I)$. Note that for the bounded kernels with bound 1, $\Delta_{\widehat{\boldsymbol{\mu}}_P} = \frac{2}{m}$.

Now the proposition is given as follows.

Proposition B.1. *Given samples $\mathbf{x} = \{x_i\}_{i=1}^m \sim P$ and $\tilde{\mathbf{x}} = \{\tilde{x}_j\}_{j=1}^n \sim Q$, the expected absolute error between the noisy random-feature (squared) MMD defined in eq. 7 and the squared MMD eq. 2 is bounded by*

$$\mathbb{E}_{\mathbf{n}}\mathbb{E}_{\hat{\boldsymbol{\phi}}}\left[\left|\widetilde{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}}) - \widehat{\mathrm{MMD}}^2(\mathbf{x},\tilde{\mathbf{x}})\right|\right], \tag{25}$$

$$\leq \left(\frac{4D\sigma^2}{m^2} + \frac{8\sqrt{2}\sigma}{m}\frac{\Gamma\big((D+1)/2\big)}{\Gamma\big(D/2\big)}\right) + 8\sqrt{\frac{2\pi}{D}} \tag{26}$$

where Γ is the Gamma function.

To prove this proposition, we first rewrite the absolute error in terms of two terms due to the triangle inequality:

$$\begin{aligned}
&\mathbb{E}_{\mathbf{n}}\mathbb{E}_{\hat{\boldsymbol{\phi}}}\left[\left|\widetilde{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}}) - \widehat{\mathrm{MMD}}^2(\mathbf{x},\tilde{\mathbf{x}})\right|\right] \\
&\leq \mathbb{E}_{\mathbf{n}}\mathbb{E}_{\hat{\boldsymbol{\phi}}}\left[\left|\widetilde{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}}) - \widehat{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}})\right|\right] + \mathbb{E}_{\hat{\boldsymbol{\phi}}}\left[\left|\widehat{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}}) - \widehat{\mathrm{MMD}}^2(\mathbf{x},\tilde{\mathbf{x}})\right|\right].
\end{aligned} \tag{27}$$

What follows next proves each of these terms.

B.1 Randomness due to random features

We restate the result of [Sec. 3.3 of Sutherland and Schneider 2016].

Lemma B.1 (Sec. 3.3 of Sutherland and Schneider 2016)**.** *Given samples $\mathbf{x} = \{x_i\}_{i=1}^n \sim P$ and $\tilde{\mathbf{x}} = \{\tilde{x}_j\}_{j=1}^m \sim Q$, the probabilistic bound between the approximate MMD with random features, denoted by $\widehat{\mathrm{MMD}}_{rf}(\mathbf{x},\tilde{\mathbf{x}})$ and the original MMD, denoted by $\widehat{\mathrm{MMD}}(\mathbf{x},\tilde{\mathbf{x}})$, holds*

$$\mathbb{P}\left[\left|\widehat{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}}) - \widehat{\mathrm{MMD}}^2(\mathbf{x},\tilde{\mathbf{x}})\right| \geq t_1\right] \leq 2\exp\left(-\frac{1}{128}Dt_1^2\right) := U_1, \tag{28}$$

where the randomness comes from the random features, and $\mathbb{E}_{\hat{\boldsymbol{\phi}}}[\widehat{\mathrm{MMD}}_{rf}(\mathbf{x},\tilde{\mathbf{x}})] = \mathrm{MMD}(\mathbf{x},\tilde{\mathbf{x}})$.

Proof. To prove the proposition, we first consider the mean map kernel (MMK) defined by

$$\mathrm{MMK}(\mathbf{x},\tilde{\mathbf{x}}) = \frac{1}{nm}\sum_{i=1}^n\sum_{j=1}^m k(x_i,\tilde{x}_j) \approx \mathrm{MMK}_{\hat{\boldsymbol{\phi}}}(\mathbf{x},\tilde{\mathbf{x}}) := \hat{\boldsymbol{\phi}}(\mathbf{x})^\top\hat{\boldsymbol{\phi}}(\tilde{\mathbf{x}}), \tag{29}$$

which can be approximated by the random feature representations, denoted by $\mathrm{MMK}_{\hat{\boldsymbol{\phi}}}(\mathbf{x},\tilde{\mathbf{x}})$. The random feature mean-embedding of P is denoted by $\hat{\boldsymbol{\phi}}(\mathbf{x})$. Similarly, we can define $\mathrm{MMK}(\mathbf{x},\mathbf{x})$ and $\mathrm{MMK}(\tilde{\mathbf{x}},\tilde{\mathbf{x}})$, and define MMD in terms of MMKs

$$\widehat{\mathrm{MMD}}^2(\mathbf{x},\tilde{\mathbf{x}}) = \mathrm{MMK}(\mathbf{x},\mathbf{x}) + \mathrm{MMK}(\tilde{\mathbf{x}},\tilde{\mathbf{x}}) - 2\mathrm{MMK}(\mathbf{x},\tilde{\mathbf{x}}). \tag{30}$$

Notice that when we use the cosine/sine representation of random features, changing the frequency ω_k to $\hat{\omega}_k$ causes a bounded difference in the kth coordinate of the MMK estimate, $\mathrm{MMK}_{\boldsymbol{\phi}}(\mathbf{x},\tilde{\mathbf{x}})$:

$$\left|\frac{1}{nm}\sum_{i=1}^n\sum_{j=1}^m\frac{2}{D}\left[\cos(({\omega_k}^\top(x_i-\tilde{x}_j)) - \cos(({\omega'_k}^\top(x_i-\tilde{x}_j))\right]\right| \leq \frac{4}{D}. \tag{31}$$

Due to this bounded difference in each coordinate of random feature MMK, we can compute the tail bound using the McDiarmid's inequality,

$$\Pr\left[\left|\mathrm{MMK}_{\hat{\boldsymbol{\phi}}}(\mathbf{x},\tilde{\mathbf{x}}) - \mathrm{MMK}(\mathbf{x},\tilde{\mathbf{x}})\right| \geq t_1\right] \leq 2\exp\left(-\frac{1}{8}Dt_1^2\right). \tag{32}$$

Now using the definition of MMD^2 given in eq. 30, we obtain the tail bound.

$$\Pr\left[\left|\widehat{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}}) - \widehat{\mathrm{MMD}}^2(\mathbf{x},\tilde{\mathbf{x}})\right| \geq t_1\right] \leq 2\exp\left(-\frac{1}{128}Dt_1^2\right). \tag{33}$$

□

As a result of Lemma. B.1, the expected absolute error of the random-feature MMD is bounded by

Lemma B.2 (Sec. 3.3 of Sutherland and Schneider 2016)**.** *Given samples* $\mathbf{x} = \{x_i\}_{i=1}^n \sim P$ *and* $\tilde{\mathbf{x}} = \{\tilde{x}_j\}_{j=1}^m \sim Q$*, the probabilistic bound between the approximate MMD with random features, denoted by* $\widehat{\mathrm{MMD}}_{rf}(\mathbf{x},\tilde{\mathbf{x}})$ *and the original MMD, denoted by* $\mathrm{MMD}(\mathbf{x},\tilde{\mathbf{x}})$*, holds*

$$\mathbb{E}_{\hat{\boldsymbol{\phi}}}\left[\left|\widehat{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}}) - \widehat{\mathrm{MMD}}^2(\mathbf{x},\tilde{\mathbf{x}})\right|\right] \leq 8\sqrt{2\pi/D}. \tag{34}$$

Proof. For a non-negative random variable, $\left|\widetilde{\mathrm{MMD}}^2_{rf}(P,Q) - \widehat{\mathrm{MMD}}^2(P,Q)\right|$

$$\mathbb{E}_{\hat{\boldsymbol{\phi}}}\left[\left|\widehat{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}}) - \widehat{\mathrm{MMD}}^2(\mathbf{x},\tilde{\mathbf{x}})\right|\right] = \int_0^\infty \Pr\left[\left|\widehat{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}}) - \widehat{\mathrm{MMD}}^2(\mathbf{x},\tilde{\mathbf{x}})\right| \geq t_1\right] dt_1, \tag{35}$$

$$\leq 2\int_0^\infty \exp\left(-\frac{1}{128}Dt_1^2\right) dt_1, \text{due to Lemma. B.1,} \tag{36}$$

$$= 8\sqrt{\frac{2\pi}{D}}, \text{due to the Gaussian integral.} \tag{37}$$

□

B.2 Randomness due to noise for privacy

The following remark bound the first moment of the privatized MMD proxy $\widetilde{\mathrm{MMD}}_{rf}$ and the MMD proxy $\widehat{\mathrm{MMD}}_{rf}$.

Lemma B.3. *Let* $\widetilde{\mathrm{MMD}}_{rf}(\mathbf{x},\tilde{\mathbf{x}}) := \|\widehat{\boldsymbol{\mu}}_P(\mathbf{x}) + \mathbf{n} - \widehat{\boldsymbol{\mu}}_Q(\tilde{\mathbf{x}})\|_2$*, where* $\mathbf{n} \sim \mathcal{N}(0, \sigma^2\Delta^2_{\widehat{\boldsymbol{\mu}}_P} I_D)$*. Also, let* $\widehat{\mathrm{MMD}}_{rf}(\mathbf{x},\tilde{\mathbf{x}}) := \|\widehat{\boldsymbol{\mu}}_P(\mathbf{x}) - \widehat{\boldsymbol{\mu}}_Q(\tilde{\mathbf{x}})\|_2$*. Then,*

$$\mathbb{E}_{\mathbf{n}}\mathbb{E}_{\hat{\boldsymbol{\phi}}}\Big[\big|\widetilde{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}}) - \widehat{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}})\big|\Big] \leq \frac{D\sigma^2}{m^2} + 4\sqrt{2}\sigma\frac{\Gamma\big((D+1)/2\big)}{m\Gamma\big(D/2\big)} \tag{38}$$

Proof.

$$\mathbb{E}_{\mathbf{n}}\mathbb{E}_{\hat{\boldsymbol{\phi}}}\Big[\big|\widetilde{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}}) - \widehat{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}})\big|\Big] \overset{(a)}{=} \mathbb{E}_{\hat{\boldsymbol{\phi}}}\left[\mathbb{E}_{\mathbf{n}}\left[\left|\mathbf{n}^\top\mathbf{n} + 2\mathbf{n}^\top(\widehat{\boldsymbol{\mu}}_P(\mathbf{x}) - \widehat{\boldsymbol{\mu}}_Q(\tilde{\mathbf{x}}))\right|\right]\right], \tag{39}$$

$$\overset{(b)}{\leq} \mathbb{E}_{\hat{\boldsymbol{\phi}}}\left[\mathbb{E}_{\mathbf{n}}\left[\mathbf{n}^\top\mathbf{n}\right] + 2\mathbb{E}_{\mathbf{n}}\left[\left|\mathbf{n}^\top(\widehat{\boldsymbol{\mu}}_P(\mathbf{x}) - \widehat{\boldsymbol{\mu}}_Q(\tilde{\mathbf{x}}))\right|\right]\right],$$

$$\overset{(c)}{=} D\sigma^2\Delta^2_{\widehat{\boldsymbol{\mu}}_P} + 2\sqrt{2}\mathbb{E}_{\hat{\boldsymbol{\phi}}}\left[\|\widehat{\boldsymbol{\mu}}_P(\mathbf{x}) - \widehat{\boldsymbol{\mu}}_Q(\mathbf{x})\|_2\right]\sigma\Delta_{\widehat{\boldsymbol{\mu}}_P}\frac{\Gamma\big((D+1)/2\big)}{\Gamma\big(D/2\big)}, \tag{40}$$

$$\overset{(d)}{=} \frac{D\sigma^2}{m^2} + 4\sqrt{2}\sigma\frac{\Gamma\big((D+1)/2\big)}{m\Gamma\big(D/2\big)}, \tag{41}$$

□

where (a) is by expanding two terms following their definitions: $\widetilde{\text{MMD}}^2_{rf}(\mathbf{x}, \tilde{\mathbf{x}}) - \widehat{\text{MMD}}^2_{rf}(\mathbf{x}, \tilde{\mathbf{x}}) = \mathbf{n}^\top \mathbf{n} + 2\mathbf{n}^\top (\widehat{\boldsymbol{\mu}}_P(\mathbf{x}) - \widehat{\boldsymbol{\mu}}_Q(\tilde{\mathbf{x}}))$. (b) is followed by triangle inequality. (c) is followed by the second moment of the chi-square random variable (first term) and the first moment of the chi distribution (second term). (d) is by taking the maximum over random features. Under the random feature representation we use in our paper, the L2-norm of random features is bounded by 1. Hence, $\mathbb{E}_{\hat{\boldsymbol{\phi}}} [\|\widehat{\boldsymbol{\mu}}_P(\mathbf{x}) - \widehat{\boldsymbol{\mu}}_Q(\mathbf{x})\|_2] \leq \max_{\hat{\boldsymbol{\phi}}} [\|\widehat{\boldsymbol{\mu}}_P(\mathbf{x}) - \widehat{\boldsymbol{\mu}}_Q(\mathbf{x})\|_2] \leq \max_{\hat{\boldsymbol{\phi}}} [\|\widehat{\boldsymbol{\mu}}_P(\mathbf{x})\|_2 + \|\widehat{\boldsymbol{\mu}}_Q(\mathbf{x})\|_2] \leq 1 + 1 = 2$.

C Derivation of feature maps for a product of two kernels

Under our assumption, we decompose the kernel below into two kernels:

$$\begin{aligned}
&k((\mathbf{x}, \mathbf{y}), (\mathbf{x}', \mathbf{y}')) \\
&= k_{\mathbf{x}}(\mathbf{x}, \mathbf{x}') k_{\mathbf{y}}(\mathbf{y}, \mathbf{y}'), \text{ product of two kernels} \\
&\approx \left[\hat{\boldsymbol{\phi}}(\mathbf{x}')^\top \hat{\boldsymbol{\phi}}(\mathbf{x})\right] \left[\mathbf{f}(\mathbf{y})^\top \mathbf{f}(\mathbf{y}')\right], \text{ random features for kernel } k_{\mathbf{x}} \\
&= \text{Tr}\left(\hat{\boldsymbol{\phi}}(\mathbf{x}')^\top \hat{\boldsymbol{\phi}}(\mathbf{x}) \mathbf{f}(\mathbf{y})^\top \mathbf{f}(\mathbf{y}')\right), \\
&= \text{vec}(\hat{\boldsymbol{\phi}}(\mathbf{x}') \mathbf{f}(\mathbf{y}')^\top)^\top \text{vec}(\hat{\boldsymbol{\phi}}(\mathbf{x}) \mathbf{f}(\mathbf{y})^\top) = \hat{\mathbf{f}}(\mathbf{x}', \mathbf{y}')^\top \hat{\mathbf{f}}(\mathbf{x}, \mathbf{y})
\end{aligned}$$

D Derivation of feature maps for a sum of two kernels

Under our assumption, we compose the kernel below from the sum of two kernels:

$$\begin{aligned}
&k((\mathbf{x}_{num}, \mathbf{x}_{cat}), (\mathbf{x}'_{num}, \mathbf{x}'_{cat})) \\
&= k_{num}(\mathbf{x}_{num}, \mathbf{x}'_{num}) + k_{cat}(\mathbf{x}_{cat}, \mathbf{x}'_{cat}), \\
&\approx \hat{\boldsymbol{\phi}}(\mathbf{x}_{num})^\top \hat{\boldsymbol{\phi}}(\mathbf{x}'_{num}) + \tfrac{1}{\sqrt{d_{cat}}} {\mathbf{x}_{cat}}^\top \mathbf{x}'_{cat}, \\
&= \begin{bmatrix} \hat{\boldsymbol{\phi}}(\mathbf{x}_{num}) \\ \frac{1}{\sqrt{d_{cat}}} \mathbf{x}_{cat} \end{bmatrix}^T \begin{bmatrix} \hat{\boldsymbol{\phi}}(\mathbf{x}_{num}) \\ \frac{1}{\sqrt{d_{cat}}} \mathbf{x}_{cat} \end{bmatrix} \\
&= \hat{\mathbf{h}}(\mathbf{x}_{num}, \mathbf{x}_{cat})^T \hat{\mathbf{h}}(\mathbf{x}_{num}, \mathbf{x}_{cat}).
\end{aligned}$$

E Sensitivity of class counts

Consider the vector of class counts $\mathbf{m} = [m_1, \cdots, m_C]$, where each element m_c is the number of samples with class c in the dataset. The class counts of two neighbouring datasets $\mathcal{D}$ and $\mathcal{D}' = (\mathcal{D} \setminus \{\mathbf{x}\}) \cup \{\mathbf{x}'\}$ can differ in at most two entries k, l and at most by 1 in either entry. Assuming $\mathbf{y} \neq \mathbf{y}'$, then for $\mathbf{y}_k = 1$, $m_k = m'_k + 1$ and for $\mathbf{y}'_l = 1$, $m'_l = m_l + 1$ and $m_i = m'_i$ in all other cases. If $\mathbf{y} = \mathbf{y}'$, then $\mathbf{m} = \mathbf{m}'$. Letting $\mathbf{m}$ and $\mathbf{m}'$ denote the class counts of $\mathcal{D}$ and $\mathcal{D}'$ respectively, we get the following:

$$\Delta_{\mathbf{m}} = \max_{\mathcal{D}, \mathcal{D}'} \|\mathbf{m} - \mathbf{m}'\|_2 = \max_{\mathcal{D}, \mathcal{D}'} \sqrt{\sum_{i=1}^{C} m_i - m'_i} = \sqrt{2} \tag{42}$$

F Sensitivity of $\hat{\boldsymbol{\mu}}_P$ with homogeneous data

Below, we show that the sensitivity of the data mean embedding for homogeneous labeled data is the same as for unlabeled data. In order, we first use the fact that $\mathcal{D}$ and $\mathcal{D}'$ are neighbouring, which implies that $m - 1$ of the summands on each side cancel and we are left with the only distinct datapoints, which we denote as $(\mathbf{x}, \mathbf{y})$ and $(\mathbf{x}', \mathbf{y}')$. We then apply the triangle inequality and the definition of $\mathbf{f}$. As $\mathbf{y}$ is a one-hot vector, all but one column of $\hat{\boldsymbol{\phi}}(\mathbf{x})\mathbf{y}^\top$ are 0, so we omit them in the next step and finally use that $\|\hat{\boldsymbol{\phi}}(\mathbf{x})\|_2 = 1$.

$$
\begin{align}
\Delta_{\hat{\boldsymbol{\mu}}_P} &= \max_{\mathcal{D},\mathcal{D}'} \left\| \tfrac{1}{m} \sum_{(\mathbf{x}_i,\mathbf{y}_i)\in\mathcal{D}} \hat{\mathbf{f}}(\mathbf{x}_i,\mathbf{y}_i) - \tfrac{1}{m} \sum_{(\mathbf{x}'_i,\mathbf{y}'_i)\in\mathcal{D}'} \hat{\mathbf{f}}(\mathbf{x}'_i,\mathbf{y}'_i) \right\|_F \tag{43}\\
&= \max_{(\mathbf{x},\mathbf{y}),(\mathbf{x}',\mathbf{y})} \left\| \tfrac{1}{m}\hat{\mathbf{f}}(\mathbf{x},\mathbf{y}) - \tfrac{1}{m}\hat{\mathbf{f}}(\mathbf{x}',\mathbf{y}') \right\|_F \tag{44}\\
&\leq \max_{(\mathbf{x},\mathbf{y})} \tfrac{2}{m} \left\| \hat{\mathbf{f}}(\mathbf{x},\mathbf{y}) \right\|_F \tag{45}\\
&= \max_{(\mathbf{x},\mathbf{y})} \tfrac{2}{m} \left\| \hat{\boldsymbol{\phi}}(\mathbf{x})\mathbf{y}^\top \right\|_F \tag{46}\\
&= \max_{\mathbf{x}} \tfrac{2}{m} \left\| \hat{\boldsymbol{\phi}}(\mathbf{x}) \right\|_2 \tag{47}\\
&= \frac{2}{m} \tag{48}
\end{align}
$$

G Sensitivity of $\boldsymbol{\mu}_P$ with heterogeneous data

In the case of heterogeneous data, recall that $\hat{\mathbf{h}}(\mathbf{x}^{(i)}_{num}, \mathbf{x}^{(i)}_{cat}) = \begin{bmatrix} \hat{\boldsymbol{\phi}}(\mathbf{x}^{(i)}_{num}) \\ \frac{1}{\sqrt{d_{cat}}}\mathbf{x}^{(i)}_{cat} \end{bmatrix}$ and $\boldsymbol{\mu}_P = \frac{1}{m}\sum_{(\mathbf{x}_i,\mathbf{y}_i)\in\mathcal{D}} \hat{\mathbf{h}}(\mathbf{x}_i){\mathbf{y}_i}^\top$ where $\mathbf{x}_i$ is the concatenation of $\mathbf{x}^{(i)}_{num}$ and $\mathbf{x}^{(i)}_{cat}$. Analogous to the homogeneous case, we first derive that the labeled and unlabeled embedding have the same sensitivity (in eq. 52). We apply the definition of $\hat{\mathbf{h}}$ and analyze the numerical and categorical parts separately, using the facts that $\|\hat{\boldsymbol{\phi}}(\mathbf{x})\|_2 = 1$ and, since $\mathbf{x}_{cat}$ is binary, $\|\mathbf{x}_{cat}\|_2 \leq \sqrt{d_{cat}}$.

$$
\begin{align}
\Delta_{\boldsymbol{\mu}_P} &= \max_{\mathcal{D},\mathcal{D}'} \left\| \tfrac{1}{m} \sum_{(\mathbf{x}_i,\mathbf{y}_i)\in\mathcal{D}} \hat{\mathbf{h}}(\mathbf{x}_i){\mathbf{y}_i}^\top - \tfrac{1}{m} \sum_{(\mathbf{x}'_i,\mathbf{y}'_i)\in\mathcal{D}'} \hat{\mathbf{h}}(\mathbf{x}'_i){\mathbf{y}'_i}^\top \right\|_F \tag{49}\\
&= \max_{(\mathbf{x},\mathbf{y}),(\mathbf{x}',\mathbf{y}')} \left\| \tfrac{1}{m}\hat{\mathbf{h}}(\mathbf{x}_i){\mathbf{y}_i}^\top - \tfrac{1}{m}\hat{\mathbf{h}}(\mathbf{x}'_i){\mathbf{y}'_i}^\top \right\|_F \tag{50}\\
&\leq \max_{(\mathbf{x},\mathbf{y})} \tfrac{2}{m} \left\| \hat{\mathbf{h}}(\mathbf{x})\mathbf{y}^\top \right\|_F \tag{51}\\
&= \max_{\mathbf{x}} \tfrac{2}{m} \left\| \hat{\mathbf{h}}(\mathbf{x}) \right\|_2 \tag{52}\\
&= \max_{\mathbf{x}} \tfrac{2}{m} \left\| \begin{bmatrix} \hat{\boldsymbol{\phi}}(\mathbf{x}_{num}) \\ \frac{1}{\sqrt{d_{cat}}}\mathbf{x}_{cat} \end{bmatrix} \right\|_2 \tag{53}\\
&= \max_{\mathbf{x}} \tfrac{2}{m} \sqrt{\|\hat{\boldsymbol{\phi}}(\mathbf{x}_{num})\|_2^2 + \|\tfrac{1}{\sqrt{d_{cat}}}\mathbf{x}_{cat}\|_2^2} \tag{54}\\
&= \tfrac{2}{m}\sqrt{1 + \tfrac{d_{cat}}{d_{cat}}} \tag{55}\\
&= \frac{2\sqrt{2}}{m} \tag{56}
\end{align}
$$

I Variables in heterogeneous data are not treated as independent

While the impression may arise, our method does not assume independence between the continuous and the discrete variables, but models correlations between the two types of variables implicitly. With the sum of two kernels, the embedding is a concatenation of the two: $[E_x\phi_x(x), E_y\phi_y(y)]$, where E_x means expectation wrt $p(x)$ and E_y is wrt $p(y)$. To compute $p(x)$, we need $p(y)$ with which we marginalize out y, as $p(x) = \int p(x,y)dy$. This marginalization implicitly takes into account the correlation between the two. This is less explicit than the case using the product of two kernels. However, the sum kernel is chosen for computational tractability: a sum kernel in Fourier representation has $d_x + d_y$ features while a product kernel has $d_x \cdot d_y$.

K Heterogeneous and homogenous tabular data

In this section we describe the tabular datasets we have used in our experiments with their respective sources. We include the details of data preprocessing in case it was performed on a dataset. The datasets in this form were used in all our experiments as well as the experiments on the benchmark methods.

Credit

Credit card fraud detection dataset contains the categorized information of credit card transactions which were either fraudelent or not. Ten dataset comes from a Kaggle competition and is available at the source, `https://www.kaggle.com/mlg-ulb/creditcardfraud`. The original data has 284807 examples, of which negative samples are 284315 and positive 492. The dataset has 31 categories, 30 numerical features and a binary label. We used all but the first feature (Time).

Epileptic

Epileptic dataset describes brain activity with numerical features being EEG recording at a different point in time. The dataset comes from the UCI database, `https://archive.ics.uci.edu/ml/datasets/Epileptic+Seizure+Recognition`. It contains 11500 data points, and 179 categories, 178 features and a label. The original dataset contains five different labels which we binarize into two states, seizure or no seizure. Thus, there are 9200 negative samples and 2300 positive samples.

Census

The dataset can be downloaded by means of SDGym package, `https://pypi.org/project/sdgym/`. The dataset has 199523 examples, 187141 are negative and 12382 are positive. There are 40 categories and a binary label. This dataset contains 7 numerical and 33 categorical features.

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 at `http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html`. We used the file, kddcup.data_10_percent.gz. It is a multi-class dataset with five labels describing different types of connection intrusions. The labels were first grouped into five categories and due to few examples, we restricted the data to the top four categories.

Adult

The dataset contains information about people's attributes and their respective income which has been thresholded and binarized. It has 22561 examples, and 14 features and a binary label. The dataset can be downloaded by means of SDGym package,`https://pypi.org/project/sdgym/`.

Isolet

The dataset contains sound features to predict a spoken letter of alphabet. The inputs are sound features and the output is a latter. We binaried the labels into two classes, consonants and vowels. The dataset can be found at `https://archive.ics.uci.edu/ml/datasets/isolet`

Cervical

This dataset is created with the goal to identify the risk factors associated with cervical cancer. It is the smallest dataset with 858 instances, and 35 attributes, of which The data can be found at 15 are numerical 24 are categorical (binary). The dataset can be found at `https://archive.ics.uci.edu/ml/datasets/Cervical+cancer+%28Risk+Factors%29`. The data, however, contains missing data. We followed the pre-processing suggested at `https://www.kaggle.com/saflynn/cervical-cancer-lynn` and further removed the data with the most missing values and replaced the rest with the category mean value.

Covtype

The dataset describes forest cover type from cartographic variables. The data can be found at `https://archive.ics.uci.edu/ml/datasets/covertype`. It contains 53 attributes and a multi-class label with 7 classes of forest cover types.

K.1 The training

We provide here the details of training procedure. Some of the datasets are very imbalanced, that is they contain much more examples with one label over the others. In attempt of making categories more balanced, we undersampled the class with the largest number of samples. The complexity of a dataset also determined the number of Fourier features we used. We also varied the batch size (we include the fraction of dataset used in a batch), and the number of epochs in the training. We provide the detailed parameter settings for each of the dataset in the following table.

Table 4: Parameters settings for training tabular datasets

	non-private			**private**			
	# epochs	mini-batch size	# Fourier features	# epochs	mini-batch size	# Fourier features	undersampling rate
adult	8000	0.1	50000	8000	0.1	1000	0.4
census	200	0.5	10000	2000	0.5	10000	0.4
cervical	2000	0.6	2000	200	0.5	2000	1
credit	4000	0.6	50000	4000	0.5	5000	0.005
epileptic	6000	0.5	100000	6000	0.5	80000	1
isolet	4000	0.6	100000	4000	0.5	500	1
covtype	6000	0.05	1000	6000	0.05	1000	0.03
intrusion	10000	0.03	2000	10000	0.03	2000	0.1

K.2 Detailed results for binary class dataset

In the main text we included the details for a multi-class dataset and here we also include the results across all the classification methods for a binary dataset in Table 5 and Table 6. We also include the best and average F1-score over five runs for the respective classification methods in Table 7 and Table 8. Notice that this average corresponds to the average reported in Table 1 in the main text.

Table 5: Performance comparison on Credit dataset. The highest performance in five runs.

	Real		DP-CGAN (non-priv)		DP-MERF (non-priv)		DP-CGAN $(1, 10^{-5})$-DP		**DP-MERF** $(1, 10^{-5})$-DP	
	ROC	PRC	ROC	PRC	ROC	PRC	ROC	PRC	ROC	PRC
Logistic Regression	0.95	0.91	0.83	0.37	0.92	0.79	0.74	0.52	0.78	0.61
Gaussian Naive Bayes	0.90	0.80	0.85	0.39	0.92	0.76	0.80	0.55	0.65	0.48
Bernoulli Naive Bayes	0.89	0.84	0.58	0.19	0.89	0.82	0.67	0.42	0.90	0.74
Linear SVM	0.92	0.89	0.84	0.48	0.91	0.65	0.78	0.45	0.64	0.38
Decision Tree	0.91	0.82	0.74	0.32	0.92	0.69	0.58	0.22	0.72	0.58
LDA	0.87	0.82	0.86	0.53	0.82	0.68	0.58	0.24	0.69	0.51
Adaboost	0.94	0.89	0.83	0.51	0.93	0.85	0.62	0.32	0.75	0.63
Bagging	0.91	0.84	0.79	0.42	0.91	0.79	0.57	0.21	0.74	0.61
Random Forest	0.93	0.90	0.82	0.54	0.92	0.86	0.63	0.31	0.75	0.62
GBM	0.94	0.89	0.85	0.54	0.94	0.85	0.58	0.22	0.74	0.61
Multi-layer perceptron	0.92	0.89	0.83	0.47	0.91	0.74	0.78	0.55	0.66	0.44
XGBoost	0.94	0.91	0.81	0.49	0.94	0.87	0.70	0.53	0.72	0.59
Average	0.91	0.86	0.80	0.44	0.91	0.78	0.67	0.38	0.73	0.57

Table 6: Performance comparison on Credit dataset. The average performance over five runs.

	DP-MERF		DP-MERF	
	(non-private)		(private)	
	ROC	PRC	ROC	PRC
Logistic Regression	0.919	0.808	0.796	0.665
Gaussian Naive Bayes	0.898	0.725	0.729	0.582
Bernoulli Naive Bayes	0.879	0.791	0.752	0.586
Linear SVM	0.876	0.667	0.742	0.549
Decision Tree	0.901	0.700	0.775	0.650
LDA	0.838	0.697	0.725	0.544
Adaboost	0.912	0.828	0.787	0.689
Bagging	0.909	0.805	0.811	0.709
Random Forest	0.911	0.840	0.786	0.686
GBM	0.917	0.812	0.807	0.707
Multi-layer perceptron	0.905	0.777	0.747	0.570
XGBoost	0.915	0.837	0.812	0.716
Average	0.898	0.774	0.772	0.638

Table 7: Performance comparison on Intrusion dataset. The highest performance in five runs.

	Real	DP-CGAN	DP-MERF	DP-CGAN	DP-MERF
		(non-priv)	(non-priv)	$(1, 10^{-5})$-DP	$(1, 10^{-5})$-DP
Logistic Regression	0.948	0.710	0.926	0.567	0.940
Gaussian Naive Bayes	0.757	0.503	0.804	0.215	0.736
Bernoulli Naive Bayes	0.927	0.693	0.822	0.475	0.755
Linear SVM	0.983	0.639	0.922	0.915	0.937
Decision Tree	0.999	0.496	0.862	0.153	0.952
LDA	0.990	0.224	0.910	0.652	0.950
Adaboost	0.947	0.898	0.924	0.398	0.503
Bagging	1.000	0.499	0.914	0.519	0.956
Random Forest	1.000	0.497	0.941	0.676	0.943
GBM	0.999	0.501	0.924	0.255	0.933
Multi-layer perceptron	0.997	0.923	0.933	0.733	0.957
XGBoost	0.999	0.886	0.921	0.751	0.933
Average	0.962	0.622	0.900	0.526	0.875

Table 8: Performance comparison on Intrusion dataset. The average performance as F1 score over five runs.

	DP-MERF (non-private)	DP-MERF (private)
Logistic Regression	0.891	0.928
Gaussian Naive Bayes	0.845	0.792
Bernoulli Naive Bayes	0.454	0.508
Linear SVM	0.890	0.917
Decision Tree	0.911	0.907
LDA	0.859	0.925
Adaboost	0.899	0.592
Bagging	0.926	0.922
Random Forest	0.904	0.923
GBM	0.901	0.926
Multi-layer perceptron	0.898	0.941
XGBoost	0.891	0.921
Average	0.856	0.850

L Image data

L.1 Datasets

Both digit and fashion MNIST datasets are loaded through the torchvision package and used without further preprocessing. Both datasets of size 60000 consist of samples from 10 classes, which are close to perfectly balanced. Each sample is a 28x28 pixel image and thus of significantly higher dimensionality than the tabular data we tested.

L.2 Detailed results

A detailed version of the results summarized in Table 3 of the paper are shown below, for digit MNIST is Table 9 and fashion MNIST in Table 10. All scores are the average of 5 independent runs of training a generator and evaluating the synthetic data it produced. The tables show that DP-MERF consistently outperforms the other approaches across models. The only exceptions are Gaussian Naive Bayes and XGBoost on MNIST, where GS-WGAN and DP-CGAN respectively perform slightly better.

Table 9: Test accuracy on digit MNIST data. Average over 5 runs (data generation & model training). Best scores among private models are bold.

	Real	DP-CGAN $\epsilon = 9.6$	DP-GAN $\epsilon = 9.6$	GS-WGAN $\epsilon = 10$	DP-MERF $\epsilon = \infty$	DP-MERF $\epsilon = 1$	DP-MERF $\epsilon = 0.2$
Logistic Regression	0.930	0.600	0.702	0.741	0.772	**0.769**	0.772
Random Forest	0.969	0.638	0.538	0.460	0.714	**0.685**	0.702
Gaussian Naive Bayes	0.560	0.310	0.364	**0.576**	0.527	0.545	0.539
Bernoulli Naive Bayes	0.840	0.610	0.702	0.699	0.746	**0.750**	0.780
Linear SVM	0.920	0.550	0.700	0.704	0.756	**0.746**	0.726
Decision Tree	0.880	0.340	0.255	0.326	0.443	**0.456**	0.346
LDA	0.879	0.590	0.694	0.732	0.789	**0.793**	0.753
Adaboost	0.729	0.254	0.159	0.170	0.441	**0.456**	0.362
MLP	0.978	0.564	0.652	0.744	0.807	**0.807**	0.768
Bagging	0.928	0.430	0.282	0.387	0.624	**0.602**	0.508
GBM	0.909	0.460	0.205	0.362	0.678	**0.659**	0.552
XGBoost	0.912	**0.614**	0.459	0.408	0.525	0.555	0.509
Average	0.870	0.500	0.476	0.526	0.652	**0.652**	0.610

Table 10: Test accuracy on fashion MNIST data. Average over 5 runs (data generation & model training). Best scores among private models are bold.

	Real	DP-CGAN $\epsilon = 9.6$	DP-GAN $\epsilon = 9.6$	GS-WGAN $\epsilon = 10$	DP-MERF $\epsilon = \infty$	DP-MERF $\epsilon = 1$	DP-MERF $\epsilon = 0.2$
Logistic Regression	0.844	0.461	0.626	0.674	0.725	**0.728**	0.714
Random Forest	0.875	0.482	0.573	0.498	0.657	**0.684**	0.553
Gaussian Naive Bayes	0.585	0.286	0.149	0.505	0.598	**0.575**	0.467
Bernoulli Naive Bayes	0.648	0.497	0.592	0.558	0.602	**0.604**	0.629
Linear SVM	0.839	0.389	0.613	0.639	0.685	**0.684**	0.697
Decision Tree	0.790	0.315	0.317	0.389	0.433	**0.462**	0.352
LDA	0.799	0.490	0.638	0.653	0.735	**0.733**	0.701
Adaboost	0.561	0.217	0.224	0.275	0.291	**0.359**	0.258
MLP	0.879	0.459	0.601	0.647	0.739	**0.738**	0.696
Bagging	0.841	0.309	0.410	0.413	0.576	**0.593**	0.372
GBM	0.834	0.331	0.254	0.352	0.626	**0.624**	0.429
XGBoost	0.826	0.489	0.478	0.427	0.596	**0.610**	0.445
Average	0.780	0.390	0.457	0.502	0.605	**0.616**	0.526

M Comparison with other methods

M.1 Comparison with [4].

Algorithm 2 in [4] uses the random features similar to ours, while it releases the privatized mean embedding in terms of a weighted sum of feature maps evaluated at synthetic datapoints. The challenge is that optimizing for the synthetic datapoints using the reduced-set method becomes harder in high dimensions. To illustrate this point, we took the simulated data generated from *5-dimensional* mixture of Gaussians (the dataset [4] used). Unlike [4], our method directly trains a neural-net based generator, which can effectively approximate the privatized kernel mean embedding of the data. As a result, our method reduces the distance (this metric [4] used) between between the true kernel mean embedding $\hat{\mu}_x$ and that of the released dataset as we increase the number of synthetic datapoints, as shown in Fig. 3.

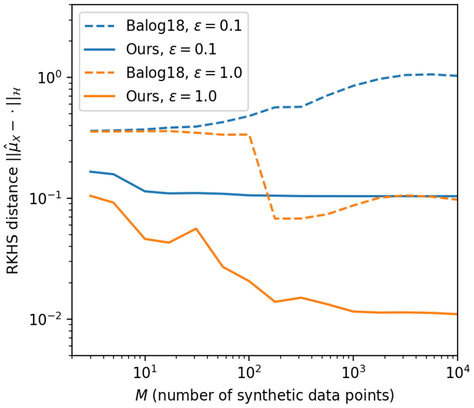

Figure 3: Comparison to [4].

M.2 Comparison with PrivBayes [38].

We compare our method to PrivBayes [38] using the published code from [15], which builds on the original code with [37] as a wrapper. We test the model on the Adult and Census datasets used in our paper by creating a version $\mathcal{D}$ of the dataset where all continuous features are discretized, and a version $\mathcal{D}^*$ where the domain of all features is reduced to a max of 15 to reduce complexity. Following [38], we measure α-way marginals for varying levels of ϵ-DP and compare them to DP-MERF at (ϵ, δ)-DP with $\delta = 10^{-5}$. Optimizing the "usefulness" parameter θ, we find, as in [38], that $\theta = 4$ is close to optimal in most settings. Results for the best θ are shown. We observe that PrivBayes performs better at $\epsilon = 1$, but is more affected by increased noise, so at $\epsilon = 0.3$ the methods are roughly tied and at $\epsilon = 0.1$ DP-MERF has lower error.

2*Adult		PrivBayes			DP-MERF			2*Census		PrivBayes			DP-MERF		
		ϵ=1	ϵ=0.3	ϵ=0.1	ϵ=1	ϵ=0.3	ϵ=0.1			ϵ=1	ϵ=0.3	ϵ=0.1	ϵ=1	ϵ=0.3	ϵ=0.1
2*$\mathcal{D}$	α=3	0.275	0.446	0.577	0.348	0.405	0.480	2*$\mathcal{D}$	α=2	0.131	0.180	0.291	0.172	0.190	0.222
	α=4	0.377	0.547	0.673	0.468	0.508	0.590		α=3	0.264	0.323	0.429	0.291	0.302	0.337
2*$\mathcal{D}^*$	α=3	0.182	0.284	0.317	0.235	0.287	0.352	2*$\mathcal{D}^*$	α=2	0.111	0.136	0.199	0.139	0.140	0.176
	α=4	0.257	0.371	0.401	0.301	0.363	0.453		α=3	0.199	0.258	0.325	0.228	0.234	0.269

It is important to stress that our approach is more general than PrivBayes in that *(i)* it does not require discretization of the data and *(ii)* scales to higher dimensionality and arbitrary domains. Bayesian network construction in PrivBayes for a

k-degree graph with d nodes (i.e. features) compares up to $\binom{d}{k}$ options on each iteration, which restricts k to small values if d is large. This means, e.g., testing PrivBayes on binarized MNIST ($d = 784$) with any $k > 2$ is infeasible.