Quantifying the Privacy Risks of Learning High-Dimensional Graphical Models

Sasi Kumar Murakonda
National University of Singapore
murakond@comp.nus.edu.sg

Reza Shokri
National University of Singapore
reza@comp.nus.edu.sg

George Theodorakopoulos
Cardiff University
TheodorakopoulosG@cardiff.ac.uk

Abstract

Models leak information about their training data. This enables attackers to infer sensitive information about their training sets, notably determine if a data sample was part of the model's training set. The existing works *empirically* show the *possibility* of these membership inference (tracing) attacks against complex deep learning models. However, the attack results are dependent on the specific training data, can be obtained *only after* the tedious process of training the model and performing the attack, and are missing any measure of the confidence and unused potential power of the attack.

In this paper, we *theoretically* analyze the maximum power of tracing attacks against high-dimensional graphical models, with the focus on Bayesian networks. We provide a tight upper bound on the power (true positive rate) of these attacks, with respect to their error (false positive rate), for a given model structure *even before* learning its parameters. As it should be, the bound is independent of the knowledge and algorithm of any specific attack. It can help in identifying which model structures leak more information, how adding new parameters to the model increases its privacy risk, and what can be gained by adding new data points to decrease the overall information leakage. It provides a measure of the potential leakage of a model given its structure, as a function of the model complexity and the size of the training set.

Proceedings of the 24[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130.

1 Introduction

How much is the privacy risk of releasing high-dimensional models which are trained on sensitive data? We focus on measuring information leakage of models about their training data, using tracing (membership inference) attacks. In a tracing attack, given the released model and a target data sample, the adversary aims at inferring whether or not the target sample was a member of the training set. We use the term tracing attack and membership inference attack interchangeably (Dwork et al., 2017; Shokri et al., 2017). The attack is evaluated based on its power (true positive rate), and its error (false positive rate), in its binary decisional task.

Tracing attacks have been extensively studied for summary statistics, where independent statistics (e.g., mean) of attributes of high-dimensional data are released. Homer et al. (2008) showed the existence of powerful tracing attacks; more recent work provided theoretical frameworks to analyze the upper bound on the power of these inference attacks (Sankararaman et al., 2009), and their robustness to noisy statistics (Dwork et al., 2015). The theoretical analysis helps explain the major causes of information leakage and assess the privacy risk even before computing the statistics. However, these analyses are limited to simple models such as product distributions.

Advanced machine learning models, such as deep neural networks, have recently been tested against tracing attacks. In the black-box setting, the attacker can only observe predictions of the model. The attack involves training inference models that can distinguish between members and non-members from the predictions that the target model produces (Shokri et al., 2017). The attacks are tested against deep neural networks as well as machine-learning-as-a-service platforms, and their accuracy is shown to be related to their generalization error (Shokri et al., 2017; Yeom et al., 2018). In the white-box setting, the attacker obtains the parameters of the model, and decides that the target sample is

a member if the gradients of the loss with respect to the model's parameters computed on the target data sample are aligned with the model's parameters (Nasr et al., 2019). Large models are empirically shown to be more vulnerable to the attack, even if they have better generalization performance. These attacks highlight the susceptibility of high-dimensional neural networks to tracing attacks. However, their analysis is limited to empirical measurements of the attack success, after training the models on particular data sets.

Contributions Using the above-mentioned existing methods, it is possible to reason theoretically about tracing attacks, yet only for simple models (independent statistics). In parallel, it is possible to perform empirical tracing attacks against complex models (deep neural networks), yet without much theoretical analysis on the maximum power of inference attacks. In this paper, we aim at addressing this gap by providing a theoretical analysis of tracing attacks against high dimensional graphical models, i.e. models with many parameters. We provide a bound on the performance of tracing attacks to quantify the privacy risk that learning a model, with max likelihood estimation (MLE), implies for the training set. Using the bound, one can determine the elements of a released model that contribute to the power of the attacker. Our focus is on *probabilistic graphical models*, which are very general statistical models that capture the correlations among data attributes, as they are among fundamental models for machine learning, and the basis of deep learning models via e.g., restricted Boltzmann machines.

We use the likelihood ratio test (LR test) as the foundation of our tracing attack (Sankararaman et al., 2009). This enables us to **design the most powerful attack** against any probabilistic model. Thus, for any given error, there exists no other attack strategy that can achieve a higher power. Our objective is not to empirically evaluate the performance of attacks (even the theoretically strongest one) on trained models. We instead **compute the maximum achievable power of tracing attacks**. This upper bound can be used as a measure to evaluate the effectiveness of different attack algorithms by comparing their achieved power to the bound for any false positive error.

Our objective is to identify the elements of a model that cause membership information leakage and measure their influence. We prove that, for a given model structure, the potential leakage of the model (the leakage that corresponds to the most powerful attack for any given error) is proportional to the square root of model's complexity (defined as the number of its independent parameters), and is inversely proportional to the square root of the size of the training set. Thus, the theoretical bound enables us to **quantify the potential leakage of a model before even learning the parameters of the model** on that structure. This can be used to efficiently compare different model structures based on their susceptibility to tracing attacks. The theoretical bound can quantify the power that the attack gains/loses if a new attribute is added/removed from the data, or when the model requires capturing/removing the correlation between certain attributes. It can also determine the size of the training set for a very high-dimensional model that leaks a similar amount of information as a small model leaks on a small set of data.

We evaluate our attack against real (sensitive) data: location check-ins, purchase history, and genome data. We empirically show that the upper bound is tight, and the power (true positive rate) of the likelihood ratio test attacker is extremely close to the bound for any error (false positive rate).

2 Probabilistic Graphical Models

Probabilistic graphical models make use of a graph-based representation to encode the dependencies and conditional independence between random variables (Koller et al., 2009). Each node X_i in a graph G is a random variable, and the edges represent the dependencies. In this paper, we focus on **Bayesian networks**. However, our attack framework is easily applicable to Markov random fields. In Bayesian networks, the model structure is a directed acyclic graph, and the model $\langle G, \theta \rangle$ enables factoring the joint probability over the random variables. Given all its parents in the graph, a random variable is conditionally independent of other random variables. Therefore, the joint distribution can be factored as follows.

$$\Pr[X_1, X_2, \cdots, X_m] = \prod_{i=1}^{m} \Pr[X_i | Pa_{X_i}^G; \theta], \quad (1)$$

where $Pa_{X_i}^G$ is the parent random variables of X_i. The parameters θ of the model encode the conditional probabilities.

We define the **complexity** $C(G)$ of a Bayesian network $\langle G, \theta \rangle$ with discrete random variables as the number of independent parameters used to define its probability distribution. Let $V(X)$ be the number of distinct values that a random variable X can take. For each conditional probability $\Pr[X_i | Pa_{X_i}^G; \theta]$, we need $|V(Pa_{X_i}^G)|(|V(X_i)|-1)$ independent parameters. Thus, the total complexity of a model is:

$$C(G) = \sum_{i=1}^{m} |V(Pa_{X_i}^G)|(|V(X_i)| - 1). \quad (2)$$

3 Problem Statement

We consider a set of n independent m-dimensional data samples from a *population*. We refer to this set as the *pool*. We do not make any assumption about the probability distribution of the data points in the general population. Given a graphical model structure G, the pool data is used to train a graphical model, i.e., to estimate the parameters $\hat{\theta}$ of the probabilistic graphical **model** $\langle G, \hat{\theta} \rangle$. The estimation can be either Max likelihood estimation (MLE) or Max a posteriori (MAP) estimate. This model is *released*. Our objective is to quantify the privacy risks of releasing such models for the members of their training data.

Let us consider an adversary who observes the released model $\langle G, \hat{\theta} \rangle$. We assume that the attacker can collect a set of independent samples from the population. We refer to this set as the *reference population*. The objective of the adversary is to perform a **tracing attack** (also known as membership inference attack) against the released model, on any target data point x: create a decision rule that determines whether x was used in the training of the parameters of $\langle G, \hat{\theta} \rangle$ or not, i.e. to classify x as being in the pool (IN) or not (OUT).

The accuracy of the tracing attack indicates the information leakage from the model about the members of its training set. We quantify the attacker's success using two metrics: the adversary's **power** (the true positive rate), and his **error** (the false positive rate). The power measures the conditional probability that the attacker classifies x as IN, given that x is indeed in the pool, i.e. $\Pr[IN|x \in \text{pool}]$. The error measures the conditional probability that the attacker classifies x as IN, given that x is not in the pool, i.e. $\Pr[IN|x \notin \text{pool}]$. The ROC curve (Receiver Operating Characteristic), which is a plot of power versus error, captures the trade-off between power and error. Thus, the area under the ROC curve (AUC) is a single metric for measuring the strength of the attack. The AUC can be interpreted as the probability that a randomly drawn data point from the pool will be assigned larger probability of being IN than a data point randomly drawn from outside the pool. So AUC = 1 implies that the attacker can correctly classify all data samples as IN or OUT.

3.1 Membership Inference Attack against Graphical Models

Given the released model, the reference population, and the target data point, the adversary aims at distinguishing between two hypothesis. Each hypothesis describes a possible world that could have resulted in the observation of the adversary, where in one world the target data was part of the training set (pool), while in the other one the target data is a random sample from the population.

- Null hypothesis (H_{OUT}): The pool is constructed by drawing n independent samples from the general population. Parameters $\hat{\theta}$ of the model $\langle G, \hat{\theta} \rangle$ are learned on the pool data. Target data x is drawn from the general population, independently from the pool.
- Alternative hypothesis (H_{IN}): The pool is constructed by drawing n independent samples from the general population. Parameters $\hat{\theta}$ of the model $\langle G, \hat{\theta} \rangle$ are learned on the pool data. Target data x is drawn from the pool.

This generalizes the hypothesis test designed by Sankararaman et al. (2009), in which the released model follows a product distribution, i.e. the random variables corresponding to data attributes are independent (equivalent to a probabilistic graphical model without any dependency edges between the nodes).

We use the Likelihood Ratio test to distinguish the two hypotheses. The goal of hypothesis testing is to find whether there is **enough evidence to reject the null hypothesis *in favor of the alternative hypothesis*,** i.e. whether the likelihood L_{IN} of the alternative hypothesis is large enough compared to the likelihood L_{OUT} of the null hypothesis. The only information we know about the pool is $\hat{\theta}$, the parameters of the released model learned using the pool data. Hence, we must calculate these *exact same parameters* under null hypothesis (i.e., learn the parameters using general population). Let θ be the result of this computation, i.e., the parameters of G trained on a large reference population. We calculate L_{IN} as the likelihood of the parameters of G taking the value $\hat{\theta}$, which is equal to $\Pr[x; \langle G, \hat{\theta} \rangle]$. Similarly, we calculate L_{OUT} as the likelihood of the parameters of G taking the value θ, which is equal to $\Pr[x; \langle G, \theta \rangle]$.

Hence, the log likelihood statistic is computed as follows.

$$L(x) = \log \left(\frac{\Pr[x; \langle G, \theta \rangle]}{\Pr[x; \langle G, \hat{\theta} \rangle]} \right) \tag{3}$$

The LR test is a comparison of the log likelihood statistic $L(x)$ with a threshold. If $L(x) \leq$ threshold, then the attacker decides in favor of H_{IN} (rejects H_{OUT}); else, he decides in favor of H_{OUT} (more precisely, he fails to reject H_{OUT} because there is not enough evidence to support this rejection in favor of H_{IN}). To determine the threshold, the attacker selects a (false positive rate) error α that he is willing to tolerate. He then empirically or theoretically estimates the distribution of $L(x)$ under the null hypothesis, using his reference

population. We denote the CDF of this distribution as F. Given α and F, the attacker computes a threshold value $F^{-1}(\alpha)$ and compares it to $L(x)$, to decide whether to reject the null hypothesis. This concludes the hypothesis test.

The *power* of the test, as defined earlier, can be expressed as $\Pr[L(x) \leq F^{-1}(\alpha)]$, computed under the alternative hypothesis, for an individual data point x randomly drawn from the pool. In other words, it is the fraction of pool data points that are correctly classified by the test. By varying α, and thus the threshold $F^{-1}(\alpha)$, we can draw the ROC curve and compute the AUC metric. It is worth emphasizing that according to the Neyman and Pearson (1933) lemma, the LR test achieves the **maximum power** among all decision rules with a given error (false positive rate). So, any other decision rule would result in a lower AUC.

4 Theoretical analysis - Bound on power of attack

Our objective is to compute the maximum power β for any false positive error α of an adversary that observes the released model $\langle G, \hat{\theta} \rangle$ which has been trained on a pool of size n. In our main result, Theorem 1, we show which combinations of α and β are possible for the attacker, and we find the major factors that determine these combinations, as a function of the model complexity and size of the dataset. To derive our main result about the best achievable power-error tradeoff, we assume that the released parameters satisfy the below conditions.

- The value of every released parameter is learned from a large enough number of samples for the central limit theorem to hold good.
- The value of every released parameter is non-trivial i.e., it is bounded away from 0 and 1 (Sankararaman et al., 2009).

These are valid assumptions to make on part of the model publisher, as it is not beneficial to publish statistically insignificant or trivial estimates. In fact, the recently published methodology of learning Bayesian Networks on Cancer Analysis System (CAS) database in the National Cancer Registration and Analysis Service (NCRAS) has similar assumptions (they use only the parameters that are learned using at least 50 samples)[1]

[1]Generating the Simulacrum A methodology overview `https://simulacrum.healthdatainsight.org.uk/wp/wp-content/uploads/2018/11/Methodology-Overview-Nov18.pdf`

Theorem 1. *Let β and α be the power and error of the LR test, for the membership inference attack, respectively. Let n be the size of the pool (model's training set), and $C(G)$ be the complexity of the released probabilistic graphical model $\langle G, \hat{\theta} \rangle$. Then, the tradeoff between power and error follows the following relation:*

$$z_\alpha + z_{1-\beta} \approx \sqrt{\frac{C(G)}{n}}, \tag{4}$$

where z_s is the quantile at level $1-s, 0 < s < 1$ of the Standard Normal distribution.

Proof sketch. To compute $\beta = \Pr_{pool}\{L(x) \leq F^{-1}(\alpha)\}$, the power of the LR test for the inference attack, for any error α, we need the distribution of $L(x)$ when x is drawn from the pool and when x is drawn from the population. Our approach to estimating the distributions of $L(x)$ is through computing its moments $\mathrm{E}(L^k), k > 0$. To approximate the distribution using its moments, we use an established statistical principle for fitting a distribution with known moments: the maximum-entropy principle. This principle states that the probability distribution which best represents the current state of knowledge is the one with largest entropy (Jaynes, 1957a,b).

To simplify the computation of the moments, we take advantage of the Bayesian decomposition to split this $L(x)$ as sum of simpler terms (one for each attribute X_i). We start by expanding (3) to give the following expression for $L(x)$:

$$\begin{aligned} L(x) &= \log\left(\frac{\prod_{i=1}^{m} \Pr[X_i | Pa_{X_i}^G; \theta]}{\prod_{i=1}^{m} \Pr[X_i | Pa_{X_i}^G; \hat{\theta}]}\right) \\ &= \sum_{i=1}^{m} \log\left(\frac{\Pr[X_i | Pa_{X_i}^G; \theta]}{\Pr[X_i | Pa_{X_i}^G; \hat{\theta}]}\right) \end{aligned} \tag{5}$$

where the X_i are the attributes of the data point x, which is now a random variable as it is drawn from the pool (or population), as just mentioned. We define L_i as the contribution of attribute X_i to the likelihood ratio L. Hence the value of L_i can be calculated as:

$$L_i = \log\left(\frac{\Pr[X_i | Pa_{X_i}^G; \theta]}{\Pr[X_i | Pa_{X_i}^G; \hat{\theta}]}\right) \tag{6}$$

We calculate the first two moments of $L(x)$ for our approximation. The mean and variance of $L(x)$ are $\mu_0 = \frac{C(G)}{2n}, \sigma_0^2 = \frac{C(G)}{n}$ under the null hypothesis and $\mu_1 = -\frac{C(G)}{2n}, \sigma_1^2 = \frac{C(G)}{n}$ under the alternative hypothesis (See proof in Appendix A). For a known mean μ and variance σ^2, the max-entropy distribution that matches the target distribution is a Gaussian $N(\mu, \sigma^2)$.

Deriving higher order moments of $L(x)$ requires information about the exact distribution that generated the data. Note that in our analysis, we do not make any assumption on the distribution from which data is generated. We just assume that the pool and the population are from the same distribution. Making assumptions on the distribution that generated the data limits the practical utility of such bounds (as we want to estimate potential leakage from a model, before touching the data). See Appendix A for details on how the data generator distribution affects the distribution of the log-likelihood ratio.

Given this approximation, and the computed mean and variance, the relationship between power β, and error α is

$$\mu_0 - z_\alpha \sigma_0 = \mu_1 - z_\beta \sigma_1 \tag{7}$$

where z_s is the quantile at level $1-s, 0<s<1$ of the standard normal distribution. This equation can be derived by equating quantiles at level β, α in the pool and population distribution respectively.

Substituting $\mu_0, \sigma_0, \mu_1, \sigma_1$ into (7), we derive the main result. □

In case of high-dimension models, the log-likelihood ratio distribution is very close to normal distribution (as it is a sum of large number of independent random variables) and assuming it follows a normal distribution is a good approximation for most practical purposes. As we will show in the latter sections (Section 5.2 and Section 4.1), the contribution of higher order moments to the estimates of privacy risk and understanding of the sources of information leakage is marginal. We illustrate this by comparing our theoretical bound with the empirically observed maximum power for any error and explaining the power of attacks using parameter estimation errors.

The intuition behind our result is that the centers ($\mu_0 = \frac{C(G)}{2n}$ and $\mu_1 = -\frac{C(G)}{2n}$) of $L(x)$ under the null and alternative hypotheses are separated by a distance of $\frac{C(G)}{n}$. The overlap between the distributions is determined by variance $\frac{C(G)}{n}$ of the statistic, and the amount of the overlap between the two distributions determines the power $\beta = \Pr_{pool}\{L(x) \leq F^{-1}(\alpha)\}$ for any error α.

Our result generalizes that of Sankararaman et al. (2009) on releasing independent marginals. In their case, the released graph has no edges and nodes are binary variables. The complexity of such a graph is equal to the number of nodes m. Hence, for independent marginals we recover Sankararaman et al's relation:

$$z_\alpha + z_{1-\beta} = \sqrt{\frac{m}{n}}. \tag{8}$$

4.1 Insights from the bound:

The bound in Theorem 1 is independent of the exact values of the data in the pool and depends only on the metadata of the model: pool size n, number of attributes m and model structure G. This implies that the analysis is robust to varying the details of the dataset, but it is expressive enough to capture and resolve questions like the following:

- Which one of many model structures has the largest/smallest leakage?
- What is the additional leakage caused by releasing one more attribute for each data point in the pool?
- How do the dependencies among a certain group of attributes affect the leakage?
- How exactly does the pool size affect leakage?

Using the bound, we can observe and quantify the effect of releasing a model in terms of its complexity $C(G)$. Releasing more parameters helps the attacker, and we also see that e.g. quadrupling $C(G)$ would double the sum $z_\alpha + z_{1-\beta}$, thus reducing the error or increasing the power or both. The amount of improvement depends on how large the sum already is and there are diminishing returns. In contrast, increasing the pool size n has the opposite effect to increasing $C(G)$: the attack performance becomes worse. This makes sense, as a larger pool is more similar to (has more overlap with) the population, so it is more difficult for the attacker to distinguish between them.

It is also possible to see whether a heuristic attack can be improved by comparing its error and power to the ones implied by the main theorem for a given complexity and pool size. From the heuristic attack's error and power, we can compute the corresponding Standard Normal quantiles and compare their sum to $\sqrt{\frac{C(G)}{n}}$. If the sum is far from the bound, then the attack can be improved. From a defender's point of view, we can quantify the maximum leakage associated with releasing various models **without** having to train each model and **without** having to perform any attack. We can reason about the ultimate/maximum power of the attacker, e.g. one with perfect knowledge about the population, so as to guide our choice of a model to release.

It is also interesting to note that the natural complexity behind the structure of a graphical model captures its privacy risk. The difference between an estimated parameter value (calculated from the pool) and the actual parameter value (calculated from the general population) is an estimation error that leaks information about the pool. Since these estimation errors are

independent across parameters of a Bayesian network, each parameter makes a separate contribution to the power of the attacker. Hence, the complexity measure defined as the number of parameters captures the potential privacy risk of the model. See Appendix G for a detailed discussion on why the estimation errors of parameters in graphical models are independent.

5 Experiments

We use two methods for performing and evaluating the attack:

1. **Theoretical:** Given a false positive rate and released model structure G, we use our main result (Theorem 1) to calculate the power, error, and AUC.
2. **Empirical:** In empirical analysis, we vary the threshold of LR test from $-\infty$ to $+\infty$ and calculate the power at each value of false positive rate. This is the maximum possible power that can be achieved. Hence we use the power and AUC values calculated here to compare with the bound presented in Theorem 1.

5.1 Data Sets

A summary of all the data sets which are used in our experiments is provided in Table 1.

Location: This is a binary data set containing the Foursquare location check-ins by individuals in Bangkok (Shokri et al., 2017). Each record corresponds to an individual and consists of binary attributes reflecting visits to different locations.

Purchase: This is a binary data set containing information about individuals and their purchases (Shokri et al., 2017). Each record corresponds to an individual and each attribute represents a product. A value of 1 at attribute j means that the individual purchased the product corresponding to attribute j.

Genome: OpenSNP[2] is an open source data sharing website, where people can share their genomic data test results. We obtained the data provided by OpenSNP and considered only the individuals sequenced by 23andme. We randomly selected 1000 SNPs on chromosome 1. Individuals with more than 2 missing values were filtered out. After this pre-processing, we were left with 2497 individuals and 1000 SNPs for each individual.

Bayesian Networks have been used to model genome sequences in (Agrahari et al., 2018; Su et al., 2013).

[2]https://opensnp.org/snps

Data Set	# Attributes	Original Size	Augmented Dataset Size
Location	446	5010	30000
Purchase	600	30000	30000
Genome	1000	2497	10000

Table 1: **Summary of Datasets used:** As the size of the original dataset is small for Location and Genome data, we augment the original dataset with synthetic data generated independently from a Bayesian network with $\eta = 3$ learned on the original dataset, where η is the maximum number of parents a node can have. We use the full augmented set as the general population. See Appendix I for complete details on data synthesis.

We use a similar approach to model the SNPs as a Bayesian Network. Since humans are diploid, at each position, we have two bases i.e. three possible values. While releasing graphical models constructed from genomic data, we only estimate the Minor and Major Allele Frequencies. To calculate the probability of any combination, we assume independence and compute it as the product of Allele Frequencies.

Data augmentation and Evaluation method: As the size of the original dataset is small for Location and Genome data, we augment the original dataset with synthetic data sampled independently from a Bayesian network with learned $\eta = 3$ (maximum number of parents per node) on the original dataset. We use the full augmented set as the general population. See Appendix I for details on data synthesis, structure learning, and parameter learning in Bayesian networks. In all the experiments, the pool and reference population are sampled independently from the general population. To evaluate the attack, all available samples from the general population are used to compute power and false positives. The pool size and reference population size for experiments with the Location and Purchase datasets are 3000 and 15000 respectively. The pool size and reference population size for experiments with the Genome dataset are 1000 and 5000 respectively. We perform each experiment with 50 different and independent splits of pool and reference population and report the average statistics. **This random splitting and averaging ensures that the results are not biased by data augmentation or by a single instance of sampling the pool.**

5.2 Validity of the theoretical bound

We first present how the complexity of released graphical model affects the power of tracing attack. Table 2 and Figure 1 (column 1) show the AUC and power respectively of the tracing attack when models of various complexities are released. In Figure 1 (columns 2, 3), we compare the observed power of tracing attack with the bound from Theorem 1. Columns 2 and 3 correspond to releasing Bayesian Networks learned with

Data set	No. of Nodes	η	No. of Edges	Complexity	AUC (Empirical)	AUC (Theoretical)
Location	446	0	0	446	0.5928	0.6074
		1	343	789	0.6337	0.6415
		2	566	1222	0.6655	0.6741
		3	757	1905	0.6998	0.7134
Purchase	600	0	0	600	0.5700	0.6241
		1	496	1096	0.6266	0.6654
		2	941	1942	0.6885	0.7153
		3	1358	3431	0.7541	0.7752
Genome	1000	0	0	1000	0.6729	0.7602
		1	729	1729	0.7875	0.8237
		2	1244	2706	0.8495	0.8776
		3	1712	4323	0.9058	0.9292

Table 2: **AUC comparison for model structures we learned on different datasets, with different complexities**. We compare the AUC values for empirical attack with the corresponding values computed using the theoretical bound. The variable η represents maximum number of parents a node can have in the graph. We can observe that the empirical values of AUC are closer to the bound and increase with increasing complexity of the model.

$\eta = 0$ and $\eta = 3$ respectively. The variable η represents maximum number of parents a node can have in the graph.

As shown in Table 2, the AUC values are comparatively smaller for the Purchase data set compared to that of the Genomes dataset. This is because, in case of Purchase data, we only have 600 attributes for a pool size of 3000. In case of Genomic data, we have 1000 attributes for a pool size of 1000. Even then, on Purchase data, if a Bayesian network with $\eta = 3$ is released, we can achieve an AUC value of approximately 0.75, compared to 0.57 when only marginals are released. The complexity of a graphical model represents the number of independent parameters in the model. Each of these parameters is learned from individuals in the pool and hence can leak more information about membership in the pool. This leakage contributes to the power of the tracing attack. Hence, we confirm that the higher the complexity of the released model, the higher the power of the tracing attack.

We can clearly observe that the empirical and theoretical power in column 2 and 3 of Figure 1 are very close to each other except for the case of $\eta = 0$ on genome data. When $\eta = 0$, the model cannot capture any dependency in the data. If the released model does not capture all the dependencies among attributes in data (under fitted), then estimation errors of parameters in released graphical model become correlated. This effectively reduces the amount of information available to perform membership inference. Overall, we can see that the theoretical bound can capture the empirically observed power very effectively. **The observed power of tracing attack is close to the value calculated from the theoretical bound demonstrating the validity and usefulness of the bound.**

6 Related Work

Homer et al. (2008) developed a statistical test based on likelihood ratio for inferring the presence of a genome sequence, given the Allele frequencies. Sankararaman et al. (2009) extend this work and provide tight bounds on the power of tracing attack for any adversary. This was further extended for continuous Gaussian variables (Micro RNA) data in (Backes et al., 2016). Similar attacks on genomic data using statistics published in association studies are performed in (Shringarpure and Bustamante, 2015; Wang et al., 2009). Dwork et al. (2015) take a different approach and provide a generic framework for tracing attacks based on distance metric, when noisy statistics are released. Im et al. (2012), the authors use a correlation statistic to perform a tracing attack against regression coefficients from quantitative phenotypes. All these works assume independence among data attributes, whereas our work addresses the case of dependent attributes.

Shokri et al. (2017) perform membership inference attacks against black-box machine learning models. The adversary constructs *shadow models* that mimic the behavior of the target model. The attack is treated as a binary classification problem and the decision rule is a machine learning model trained on data from the shadow models. Salem et al. (2018) follow a similar framework as (Shokri et al., 2017) but relax certain assumptions on the knowledge and power of adversary. Similar attacks were performed against aggregate location data (Pyrgelis et al., 2017), generative adversarial networks (Hayes et al., 2018) and in a collaborative learning setting (Melis et al., 2018; Nasr et al., 2019). These works provide empirical analysis of tracing attack on complex models. A theoretical formulation of Bayes-optimal attack for membership inference against neural networks was given in (Sablayrolles et al., 2019),

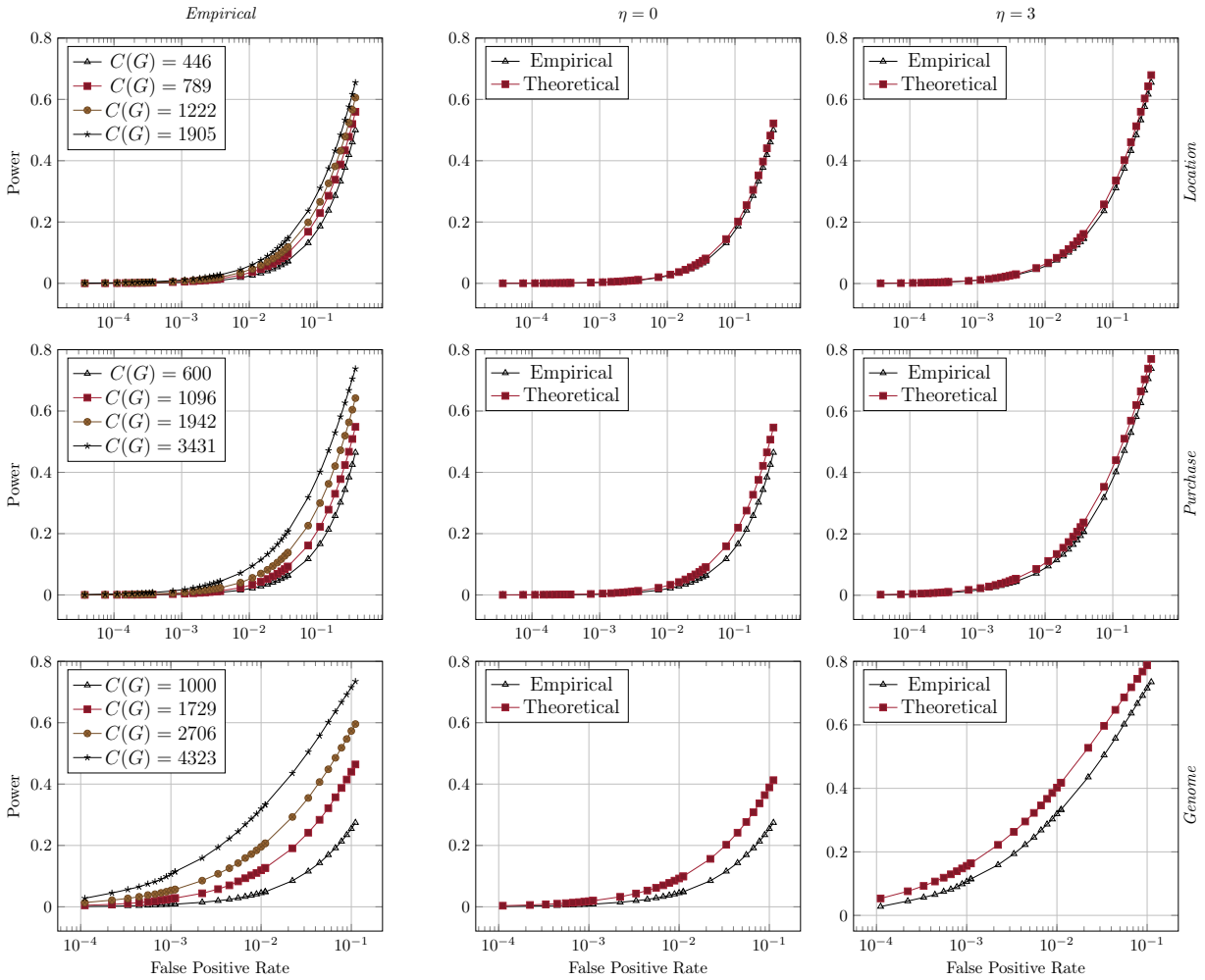


Figure 1: **Power of Attack on Real world data:** The effect of model complexity on the power of the attack is shown in the first column. When graphical models of increasing complexity are released, the power of attack increases, as we have more parameters that can leak information. In the second and third columns, we compare the observed powers with their corresponding theoretical bounds. We can clearly observe that the two curves (empirically observed power and theoretical bound) are very close to each other demonstrating the validity and usefulness of the bound.

which shows that existing techniques based on shadow models (Shokri et al., 2017) are approximations of this optimal attack. It is also shown that the power of optimal attack on black box models is the same as that on white box models, but no bound on this power is provided. We present a theoretical bound on the power of the attack, which is independent of the training data and the auxiliary knowledge of the adversary.

Differential Privacy (Dwork et al., 2006) has been accepted as the de facto standard notion of privacy. Zhang et al. (2017) learn a Bayesian network in a differentially private way and then use a noisy version of it to generate synthetic data. Bindschaedler et al. (2017) introduce the notion of plausible deniability and propose a mechanism that achieves it by generating synthetic data that is statistically similar to the given input data set. By definition, differential privacy decreases the power of tracing attack and its effect on our bound is discussed in Appendix H.

7 Summary

We provide a theoretical analysis of tracing attacks against probabilistic graphical models to address the existing gap between theoretical analysis for simple average statistics on data with independent attributes and empirical demonstrations for complex models on data with correlated attributes. Our bound quantifies the maximum attack performance measured with the error (false positive rate) and power (true positive rate) of a likelihood-ratio test. We experimentally validate and complement our results using sensitive datasets - location check-ins, purchase history, genomic data.

Acknowledgments

This work is partially supported by the Singapore Ministry of Education Academic Research Fund, R-252-000-660-133, the NUS Early Career Research Award (NUS ECRA), grant number NUS ECRA FY19 P16, and the National Research Foundation, Singapore under its Strategic Capability Research Centres Funding Initiative.

References

Rupesh Agrahari, Amir Foroushani, T Roderick Docking, Linda Chang, Gerben Duns, Monika Hudoba, Aly Karsan, and Habil Zare. Applications of bayesian network models in predicting types of hematological malignancies. *Scientific reports*, 8(1):6951, 2018.

Michael Backes, Pascal Berrang, Mathias Humbert, and Praveen Manoharan. Membership privacy in microrna-based studies. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 319–330. ACM, 2016.

Vincent Bindschaedler, Reza Shokri, and Carl A Gunter. Plausible deniability for privacy-preserving data synthesis. *Proceedings of the VLDB Endowment*, 10(5):481–492, 2017.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

Cynthia Dwork, Adam Smith, Thomas Steinke, Jonathan Ullman, and Salil Vadhan. Robust traceability from trace amounts. In *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on*, pages 650–669. IEEE, 2015.

Cynthia Dwork, Adam Smith, Thomas Steinke, and Jonathan Ullman. Exposed! a survey of attacks on private data. *Annual Review of Statistics and Its Application*, 4:61–84, 2017.

J Hayes, L Melis, G Danezis, and E De Cristofaro. Logan: Membership inference attacks against generative models. In *Proceedings on Privacy Enhancing Technologies (PoPETs)*, number 1. De Gruyter, 2018.

Nils Homer, Szabolcs Szelinger, Margot Redman, David Duggan, Waibhav Tembe, Jill Muehling, John V Pearson, Dietrich A Stephan, Stanley F Nelson, and David W Craig. Resolving individuals contributing trace amounts of dna to highly complex mixtures using high-density snp genotyping microarrays. *PLoS genetics*, 4(8):e1000167, 2008.

Hae Kyung Im, Eric R Gamazon, Dan L Nicolae, and Nancy J Cox. On sharing quantitative trait gwas results in an era of multiple-omics data and the limits of genomic privacy. *The American Journal of Human Genetics*, 90(4):591–598, 2012.

Edwin T Jaynes. Information theory and statistical mechanics. *Physical review*, 106(4):620, 1957a.

Edwin T Jaynes. Information theory and statistical mechanics. ii. *Physical review*, 108(2):171, 1957b.

Daphne Koller, Nir Friedman, and Francis Bach. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Exploiting unintended feature leakage in collaborative learning. *arXiv preprint arXiv:1805.04049*, 2018.

M. Nasr, R. Shokri, and A. Houmansadr. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *IEEE Symposium on Security and Privacy (SP)*, pages 1022–1036, 2019. doi: 10.1109/SP.2019.00065. URL `doi.ieeecomputersociety.org/10.1109/SP.2019.00065`.

Jerzy Neyman and Egon Sharpe Pearson. Ix. on the problem of the most efficient tests of statistical hypotheses. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 231(694-706): 289–337, 1933.

Apostolos Pyrgelis, Carmela Troncoso, and Emiliano De Cristofaro. Knock knock, who's there? membership inference on aggregate location data. *arXiv preprint arXiv:1708.06145*, 2017.

Alexandre Sablayrolles, Matthijs Douze, Yann Ollivier, Cordelia Schmid, and Hervé Jégou. White-box vs black-box: Bayes optimal strategies for membership inference. *arXiv preprint arXiv:1908.11229*, 2019.

Ahmed Salem, Yang Zhang, Mathias Humbert, Pascal Berrang, Mario Fritz, and Michael Backes. Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models. *arXiv preprint arXiv:1806.01246*, 2018.

Sriram Sankararaman, Guillaume Obozinski, Michael I Jordan, and Eran Halperin. Genomic privacy and limits of individual detection in a pool. *Nature genetics*, 41(9):965, 2009.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *Security and Privacy (SP), 2017 IEEE Symposium on*, pages 3–18. IEEE, 2017.

Suyash S Shringarpure and Carlos D Bustamante. Privacy risks from genomic data-sharing beacons. *The*

American Journal of Human Genetics, 97(5):631–646, 2015.

Chengwei Su, Angeline Andrew, Margaret R Karagas, and Mark E Borsuk. Using bayesian networks to discover relations between genes, environment, and disease. *BioData mining*, 6(1):6, 2013.

Rui Wang, Yong Fuga Li, XiaoFeng Wang, Haixu Tang, and Xiaoyong Zhou. Learning your identity and disease from research papers: information leaks in genome wide association study. In *Proceedings of the 16th ACM conference on Computer and communications security*, pages 534–544. ACM, 2009.

Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, pages 268–282. IEEE, 2018.

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): 25, 2017.

Appendix: Quantifying the Privacy Risks of Learning High-Dimensional Graphical Models

Contents

Symbol	Description
m	Number of attributes
n	Pool (training set) size
$\langle G, \hat{\theta} \rangle$	Released Model
$\langle G, \theta \rangle$	Population Model
X_i	Random variable for attribute i
x_i	A particular value for X_i
$V(X_i)$	Set of possible values of attribute i
$Pa^G_{X_i}$	Set of random variables that are parents of node X_i in G
p_i^v	$\Pr(x_i = 1 \mid Pa^G_{X_i} = v; \theta)$
$\hat{p}_i^v$	$\Pr(x_i = 1 \mid Pa^G_{X_i} = v; \hat{\theta})$
n_i^v	Number of samples used to compute $\hat{p}_i^v$
$C(G)$	Complexity of G (number of independent parameters)
η	Maximum number of parents per node in G
$L(x)$	Log-likelihood ratio statistic for data sample x
L_i	Contribution of X_i to the Log-likelihood ratio $L(x)$
F	CDF of $L(x)$ over the general population (under $\mathrm{H_{OUT}}$)
α	Error (False Positive Rate) of LR tracing attack
β	Power (True Positive Rate) of LR tracing attack
z_s	Quantile at level $1 - s$ of the Standard Normal distribution

Table 1: **Notations**

A Derivation of mean and variance of the likelihood ratio

We compute the mean and variance of $L(x)$ under the two hypotheses. We sketch the proof for the mean $\mathrm{E}(L)$ under the population hypothesis, followed by the variance $\mathrm{Var}(L)$. Similar calculations apply for the pool hypothesis.

Let the target x have the feature vector $(x_1, x_2, \ldots, x_m)$, and let us assume, for now, that all attributes are binary: $x_i \in \{0, 1\}, i = 1, \ldots, m$. In Appendix E we generalize to attributes that can take more than two values. We can take advantage of the Bayesian network decomposition to write the log-likelihood ratio for x as follows:

$$L(x) = \log\left[\frac{\Pr(x; \langle G, \theta\rangle)}{\Pr(x; \langle G, \hat{\theta}\rangle)}\right] = \sum_{i=1}^{m} L_i \tag{1}$$

where L_i is the contribution of X_i to the likelihood ratio, as defined as:

$$\begin{aligned} L_i &= \log\left(\frac{\Pr[X_i|Pa^G_{X_i}; \theta]}{\Pr[X_i|Pa^G_{X_i}; \hat{\theta}]}\right) \\ &= \sum_{v \in V(Pa^G_{X_i})} 1_{\{Pa^G_{X_i}=v\}} \underbrace{\left(x_i \log\frac{p_i^v}{\hat{p}_i^v} + (1 - x_i)\log\frac{1-p_i^v}{1-\hat{p}_i^v}\right)}_{L_i^v} \\ &= \sum_{v \in V(Pa^G_{X_i})} 1_{\{Pa^G_{X_i}=v\}} L_i^v \end{aligned} \tag{2}$$

where $p_i^v = \Pr\{X_i = 1|Pa^G_{X_i} = v; \theta\}$, and similarly $\hat{p}_i^v = \Pr\{X_i = 1|Pa^G_{X_i} = v; \hat{\theta}\}$. The notation $1_{\{Pa^G_{X_i}=v\}}$ is an indicator variable for a particular assignment of values to the parent nodes of X_i, i.e. $1_{\{Pa^G_{X_i}=v\}} = 1$ if $Pa^G_{X_i} = v$ and 0 otherwise. The sum ranges over $|V(Pa^G_{X_i})|$ terms L_i^v, one for each element of $V(Pa^G_{X_i})$.

The parameters θ and $\hat{\theta}$ are estimated from data (reference population and pool, respectively). By the central limit theorem, the distribution of such an estimate converges to a Gaussian around the mean value of the estimate as the number of data samples increases. In our derivations of the mean and variance, we use this approximation in (10) and (11). By the Berry-Esseen theorem [1, 3], the rate of convergence to the Gaussian is $O(\frac{1}{\sqrt{n}})$ if the third moment of the random variable being sampled is finite. In our case this condition is true, because each random variable can only take a finite number of possible finite values.

We compute the mean and variance of $L(x)$ as follows:

$$\mathrm{E}_{pop}(L) = \frac{C}{2n} + O(Cn^{-2}) \tag{3a}$$

$$\mathrm{E}_{pool}(L) = -\frac{C}{2n} + O(Cn^{-2}) \tag{3b}$$

$$\mathrm{Var}_{pop}(L) = \frac{C}{n} + O(C^2n^{-2}) \tag{3c}$$

$$\mathrm{Var}_{pool}(L) = \frac{C}{n} + O(C^2n^{-2}). \tag{3d}$$

Proof sketch - Mean under H_{OUT}. The mean $\mathrm{E}_{pop}(L)$ can be computed as follows:

$$\begin{aligned} \mathrm{E}_{pop}(L) &= \sum_{i=1}^{m} \mathrm{E}_{pop}(L_i) \\ &= \sum_{i=1}^{m} \sum_{v} \mathrm{E}_{pop}(1_{\{Pa^G_{X_i}=v\}} L_i^v) \end{aligned} \tag{4}$$

Using approximation (12) in Appendix Section C, we compute $\mathrm{E}_{pop}(1_{\{Pa^G_{X_i}=v\}}L_i^v) \approx \frac{1}{2n} + O(n^{-2})$. Since the total number of L_i^v parameters is $C = \sum_{i=1}^m |V(Pa^G_{X_i})|$, we conclude that

$$\mathrm{E}_{pop}(L) = \frac{C}{2n} + O(Cn^{-2}). \tag{5}$$

□

Proof sketch - Variance under H_{OUT}. By definition,

$$\mathrm{Var}_{pop}(L) = \mathrm{E}_{pop}[L^2] - (\mathrm{E}_{pop}[L])^2. \tag{6}$$

The latter term $(\mathrm{E}_{pop}[L])^2$ is the square of the mean, which we compute in (5). The former term $\mathrm{E}_{pop}[L^2]$ decomposes as follows:

$$\mathrm{E}_{pop}[L^2] = \sum_{i=1}^m \mathrm{E}_{pop}[L_i^2] + 2\sum_{1\le i<j\le m} \mathrm{E}_{pop}[L_iL_j] \tag{7}$$

We compute $\mathrm{E}_{pop}[L_i^2]$ by expanding $\mathrm{E}_{pop}[(\sum_v 1_{\{Pa^G_{X_i}=v\}}L_i^v)^2]$. Then, approximation (22) in Appendix D gives us that each square term $\mathrm{E}_{pop}[(1_{\{Pa^G_{X_i}=v\}}L_i^v)^2]$ is approximately equal to $\frac{1}{n}$. As for the product terms in the expansion, each term multiplies two different indicator variables $1_{\{Pa^G_{X_i}=v\}}$ and $1_{\{Pa^G_{X_i}=v'\}}$ with $v \neq v'$. Because at most one of the two is equal to 1, all product terms will be zero. Hence $\mathrm{E}_{pop}[L_i^2] = |V(Pa^G_{X_i})| \times \frac{1}{n}$.

The number of joint terms $\mathrm{E}_{pop}[L_iL_j]$ is $O(C^2)$. From the approximation in Appendix Section D.1 for $\mathrm{E}_{pop}[L_iL_j]$, each of these terms is equal to $\frac{1}{4n^2}$ with error term $O(n^{-2})$. Hence, the value of $\mathrm{E}_{pop}[L^2]$ is

$$E_{pop}[L^2] = \frac{C}{n} + \frac{C^2}{4n^2} + O(C^2n^{-2}). \tag{8}$$

We conclude that the variance is

$$\begin{aligned}\mathrm{Var}_{pop}(L) =& \mathrm{E}_{pop}[L^2] - (\mathrm{E}_{pop}[L])^2 \\ =& \frac{C}{n} + \frac{C^2}{4n^2} + O(C^2n^{-2}) - \left(\frac{C}{2n} + O(Cn^{-2})\right)^2 \\ =& \frac{C}{n} + O(C^2n^{-2})\end{aligned} \tag{9}$$

□

Although we haven't provided the calculation here, it is possible to calculate the exact value of the $O(C^2n^{-2})$ term from the released model. As a simple example, in appendix F, we calculate the exact value of this $O(C^2n^{-2})$ term, when the released model is a Naive Bayes model.

A.1 Distribution of the log-likelihood ratio

To compute the distribution of $L(x)$, the log-likelihood ratio of the parameter vector estimate with the actual value of the parameter vector in a graphical model, we need to understand what parameters contribute to the likelihood ratio given a data sample. As shown in equation (2), the parameter that contributes to the likelihood ratio for an attribute is determined by the value taken by its parent node. The contribution of attribute X_i to the likelihood function, denoted by L_i, is computed as:

$$L_i = \sum_{v\in V(Pa^G_{X_i})} 1_{\{Pa^G_{X_i}=v\}}L_i^v$$

Hence the distribution of L_i is a mixture of the distributions of L_i^v, where the mixing probabilities are determined by the distribution of the parent nodes (hence the dependence of $L(x)$'s distribution on the probability distribution

that generated the data). The distribution of L_i^v, the log-likelihood ratio for the estimate of a single parameter value, is asymptotically a chi-squared distribution with degree of freedom 1 (from Wilks' theorem). Hence, the exact distribution of log-likelihood ratio is a sum of mixture of chi-squared distributions, where the mixing distribution is dependent on the distribution that generated the data. In case of high-dimension models, this log-likelihood ratio distribution is very close to normal distribution (as it is a sum of large number of independent random variables (which are sum of mixtures themselves)). Hence, using the first two moments, that do not depend on the exact distribution of the sensitive data, is sufficient to produce a generic data-independent upper-bound on the privacy risk of learning the graphical model.

B Number of Samples for Estimating Conditional Probabilities

We use $\hat{p}_i^v$ to denote the estimated conditional probability that $X_i = 1$, given that the values of the activator variables are $Pa_{X_i}^G = v$. The number of samples n_i^v used to compute $\hat{p}_i^v$ are approximately Gaussian around np_v (n is the pool size, and p_v is the probability of $Pa_{X_i}^G = v$ in the general population):

$$n_i^v \approx np_v + \sqrt{np_v(1-p_v)}Z_1, \tag{10}$$

where Z_1 is a standard Gaussian random variable. In parallel, the value of $\hat{p}_i^v$ is also approximately Gaussian around the true value p_i^v:

$$\hat{p}_i^v \approx p_i^v + \sqrt{\frac{p_i^v(1-p_i^v)}{n_i^v}}Z_2, \tag{11}$$

where Z_2 is a standard Gaussian random variable.

Using these two approximations, we now prove the results required for derivation of LR statistic mean and variance.

C Approximation for mean derivation

As explained in section A, to compute the mean of the likelihood ratio we need the average contribution of each L_i^v i.e. value of $E_{pop}[1_{\{Pa_{X_i}^G=v\}}L_i^v]$. Here we prove that $E_{pop}[1_{\{Pa_{X_i}^G=v\}}L_i^v]$, when the expectation is over population is approximately equal to $\frac{1}{2n}$. When expectation is over pool, the derivation steps are similar and the value is $-\frac{1}{2n}$.

Lemma 1. *We will prove the following result:*

$$E_{pop}[1_{\{Pa_{X_i}^G=v\}}L_i^v] \approx \frac{1}{2n}\left(1 + \frac{1-p_v}{np_v}\right) \tag{12}$$

Proof. We first observe that

$$\begin{aligned} E_{pop}[1_{\{Pa_{X_i}^G=v\}}L_i^v] =& E_{\hat{p}_i^v}\left[E_x\left[1_{\{Pa_{X_i}^G=v\}}L_i^v \mid \hat{p}_i^v\right]\right] \\ =& p_v E_{\hat{p}_i^v}\left[p_i^v \log\frac{p_i^v}{\hat{p}_i^v} + (1-p_i^v)\log\frac{1-p_i^v}{1-\hat{p}_i^v}\right], \end{aligned}$$

and now all we need to show is that

$$E_{Z_1,Z_2}\left[p_i^v \log\frac{p_i^v}{\hat{p}_i^v} + (1-p_i^v)\log\frac{1-p_i^v}{1-\hat{p}_i^v}\right] \approx \frac{1}{2np_v}\left(1 + \frac{1-p_v}{np_v}\right) \tag{13}$$

We approximate $\hat{p}_i^v$ with (11) and we use the Taylor expansion of $\log(1+x) \approx x - \frac{1}{2}x^2$:

$$\begin{aligned} p_i^v \log \frac{p_i^v}{\hat{p}_i^v} &\approx - p_i^v \log \frac{p_i^v + \sqrt{\frac{p_i^v(1-p_i^v)}{n_i^v}} Z_2}{p_i^v} \\ &= - p_i^v \log \left(1 + \sqrt{\frac{1-p_i^v}{n_i^v p_i^v}} Z_2\right) \\ &\approx - p_i^v \left(\sqrt{\frac{1-p_i^v}{n_i^v p_i^v}} Z_2 - \frac{1-p_i^v}{2n_i^v p_i^v} Z_2^2\right) \\ &= - \sqrt{\frac{p_i^v(1-p_i^v)}{n_i^v}} Z_2 + \frac{1-p_i^v}{2n_i^v} Z_2^2 \end{aligned} \tag{14}$$

Similarly,

$$(1-p_i^v) \log \frac{1-p_i^v}{1-\hat{p}_i^v} \approx - \sqrt{\frac{p_i^v(1-p_i^v)}{n_i^v}} Z_2 + \frac{p_i^v}{2n_i^v} Z_2^2 \tag{15}$$

Adding (14) and (15), we have

$$p_i^v \log \frac{p_i^v}{\hat{p}_i^v} + (1-p_i^v) \log \frac{1-p_i^v}{1-\hat{p}_i^v} \approx -2\sqrt{\frac{p_i^v(1-p_i^v)}{n_i^v}} Z_2 + \frac{1}{2n_i^v} Z_2^2 \tag{16}$$

Taking the expectation $E_{Z_2}[.]$, and recalling that $E[Z_2] = 0$ and $E[Z_2^2] = 1$, we have

$$\begin{aligned} E_{Z_1,Z_2}\left[p_i^v \log \frac{p_i^v}{\hat{p}_i^v} + (1-p_i^v) \log \frac{1-p_i^v}{1-\hat{p}_i^v}\right] =& E_{Z_1}[E_{Z_2}[\dots|Z_1]] \\ \approx& E_{Z_1}\left[\frac{1}{2n_i^v}\right] \end{aligned} \tag{17}$$

We now approximate n_i^v with (10) and we use the Taylor expansion of $\frac{1}{1+x} \approx 1 - x + x^2$:

$$\begin{aligned} \frac{1}{2n_i^v} \approx& \frac{1}{2(np_v + \sqrt{np_v(1-p_v)}Z_1)} \\ =& \frac{1}{2np_v} \frac{1}{1 + \sqrt{\frac{1-p_v}{np_v}} Z_1} \\ \approx& \frac{1}{2np_v}\left(1 - \sqrt{\frac{1-p_v}{np_v}} Z_1 + \frac{1-p_v}{np_v} Z_1^2\right) \end{aligned} \tag{18}$$

Taking the expectation $E_{Z_1}[.]$, and recalling that $E[Z_1] = 0$ and $E[Z_1^2] = 1$, we have our final result:

$$E_{Z_1,Z_2}\left[p_i^v \log \frac{p_i^v}{\hat{p}_i^v} + (1-p_i^v) \log \frac{1-p_i^v}{1-\hat{p}_i^v}\right] \approx \frac{1}{2np_v}\left(1 + \frac{1-p_v}{np_v}\right)$$

□

D Approximation for variance derivation

For calculating the variance of likelihood ratio, we need the expected values of L_i^2 and $L_i L_j$. Here we first prove the below approximation and use it to calculate $E(L_i^2)$ and $E(L_i L_j)$. As explained in section A, using these values of $E(L_i^2)$ and $E(L_i L_j)$ in equation 7 we get the variance of LR statistic.

Lemma 2. *We will prove the following approximation:*

$$\mathrm{E}_{\hat{p}_i^v}\left[p_i^v \left(\log \frac{p_i^v}{\hat{p}_i^v}\right)^2 + (1-p_i^v)\left(\log \frac{1-p_i^v}{1-\hat{p}_i^v}\right)^2\right] \approx \frac{1}{np_v}\left(1 + \frac{1-p_v}{np_v}\right) \tag{19}$$

Proof. Using approximation (14)

$$\begin{aligned}
\mathrm{E}_{\hat{p}_i^v}\left[p_i^v\left(\log\frac{p_i^v}{\hat{p}_i^v}\right)^2\right] \approx& \mathrm{E}_{Z_1,Z_2}\left[\frac{1}{p_i^v}\left(-\sqrt{\frac{p_i^v(1-p_i^v)}{n_i^v}}Z_2+\frac{1-p_i^v}{2n_i^v}Z_2^2\right)^2\right]\\
=&\frac{1}{p_i^v}\,\mathrm{E}_{Z_1,Z_2}\left[\frac{p_i^v(1-p_i^v)}{n_i^v}Z_2^2+\left(\frac{1-p_i^v}{2n_i^v}\right)^2Z_2^4-2\sqrt{\frac{p_i^v(1-p_i^v)}{n_i^v}}\frac{1-p_i^v}{2n_i^v}Z_2^3\right]\\
=&\frac{1}{p_i^v}\,\mathrm{E}_{Z_1}\left[\frac{p_i^v(1-p_i^v)}{n_i^v}+3\left(\frac{1-p_i^v}{2n_i^v}\right)^2\right]\\
\approx&(1-p_i^v)\,\mathrm{E}_{Z_1}\left[\frac{1}{n_i^v}\right]\\
\approx&(1-p_i^v)\,\mathrm{E}_{Z_1}\left[\frac{1}{np_v}\left(1-\sqrt{\frac{1-p_v}{np_v}}Z_1+\frac{1-p_v}{np_v}Z_1^2\right)\right]\\
=&\frac{1-p_i^v}{np_v}\left(1+\frac{1-p_v}{np_v}\right)
\end{aligned}\tag{20}$$

Similar to (20), we have:

$$\mathrm{E}_{\hat{p}_i^v}\left[(1-p_i^v)\left(\log\frac{1-p_i^v}{1-\hat{p}_i^v}\right)^2\right]\approx\frac{p_i^v}{np_v}\left(1+\frac{1-p_v}{np_v}\right)\tag{21}$$

The desired result follows. □

D.1 Approximation of $\mathrm{E}_{pop}[L_i^2]$

We approximate $\mathrm{E}_{pop}[L_i^2]$ as:

$$\begin{aligned}
\mathrm{E}_{pop}[L_i^2] =& \mathrm{E}_{pop}\left[\left(\sum_v 1_{\{Pa_{X_i}^G=v\}}L_i^v\right)^2\right]\\
=&\mathrm{E}_{\hat{p}_i^v}\left[\mathrm{E}\left[\left(\sum_v 1_{\{Pa_{X_i}^G=v\}}L_i^v\right)^2\middle|\hat{p}_i^v\right]\right]\\
=&\sum_v p_v\,\mathrm{E}_{\hat{p}_i^v}[(L_i^v)^2]\\
=&\sum_v p_v\,\mathrm{E}_{\hat{p}_i^v}\left[p_i^v\left(\log\frac{p_i^v}{\hat{p}_i^v}\right)^2+(1-p_i^v)\left(\log\frac{1-p_i^v}{1-\hat{p}_i^v}\right)^2\right]\\
\approx&\sum_v\frac{1}{n}\left(1+\frac{1-p_v}{np_v}\right)\text{(from approximation (19))}\\
=&\frac{1}{n}|V(Pa_{X_i}^G)|+\frac{1}{n^2}\sum_v\frac{1-p_v}{p_v}
\end{aligned}\tag{22}$$

Combining the definition of complexity with equation (22), we have:

$$\sum_{i=1}^m\mathrm{E}_{pop}[L_i^2]\approx\frac{C}{n}+\frac{1}{n^2}\sum_{i=1}^m\sum_v\frac{1-p_v}{p_v}\tag{23}$$

D.2 Approximation of $\mathrm{E}_{pop}[L_iL_j]$

There are three possible cases while finding the value of $\mathrm{E}[L_iL_j]$. The random variables X_i and X_j might not have any common parents, might have some common parents or one is the parent of other. We start with the

case in which X_i and X_j have no common parents. Let $p(v_i, v_j)$ represent the joint probability of $Pa^G_{X_i} = v_i$ and $Pa^G_{X_j} = v_j$.

$$\begin{aligned}
\mathrm{E}_{pop}[L_i L_j] &= \mathrm{E}_{pop}\left[\left(\sum_v 1_{\{Pa^G_{X_i}=v_i\}} L_i^v\right)\left(\sum_{v_j} 1_{\{Pa^G_{X_j}=v_j\}} L_j^v\right)\right] \\
&= \mathrm{E}_{pop}\left[\sum_{v_i,v_j} 1_{\{Pa^G_{X_i}=v_i\}} 1_{\{Pa^G_{X_j}=v_j\}} L_i^v L_j^v\right] \\
&= \sum_{v_i,v_j} \mathrm{E}_{pop}\left[1_{\{Pa^G_{X_i}=v_i\}} 1_{\{Pa^G_{X_j}=v_j\}} L_i^v L_j^v\right] \\
&= \sum_{v_i,v_j} p(v_i, v_j)\, \mathrm{E}_{pop}\left[L_i^v L_j^v\right] \\
&\approx \sum_{v_i,v_j} p(v_i, v_j) \times \frac{1}{2np_{v_i}} \times \frac{1}{2np_{v_j}} (\text{from (13)}) \\
&= \sum_{v_i,v_j} \frac{1}{4n^2} \times \frac{p(v_i, v_j)}{p_{v_i} p_{v_j}}
\end{aligned} \tag{24}$$

In the case where X_i and X_j have common parents S_{ij}, let S_i represent the parents exclusive to X_i and S_j represent parents exclusive to X_j. Let $p(v_i, v_j, v_{ij})$ represent the joint probability of $Pa^G_{X_i} = v_i$ and $Pa^G_{X_j} = v_j$ and common parent of X_i and X_j, $Pa^G_{X_{i,j}} = v_{ij}$.

$$\begin{aligned}
\mathrm{E}_{pop}[L_i L_j] &= \mathrm{E}_{pop}\left[\left(\sum_{v_i,v_{ij}} 1_{\{Pa^G_{X_i}=v_i\}} 1_{\{Pa^G_{X_{ij}}=v_{ij}\}} L_i^v\right)\left(\sum_{v_j,v_{ij}} 1_{\{Pa^G_{X_j}=v_j\}} 1_{\{Pa^G_{X_{ij}}=v_{ij}\}} L_j^v\right)\right] \\
&= \mathrm{E}_{pop}\left[\sum_{v_i,v_j,v_{ij}} \left(1_{\{Pa^G_{X_i}=v_i\}} 1_{\{Pa^G_{X_j}=v_j\}} 1_{\{Pa^G_{X_{ij}}=v_{ij}\}} L_i^v L_j^v\right)\right] \\
&= \sum_{v_i,v_j,v_{ij}} p(v_i, v_j, v_{ij})\, \mathrm{E}_{pop}\left[L_i^v L_j^v\right] \\
&\approx \sum_{v_i,v_j,v_{ij}} p(v_i, v_j, v_{ij}) \times \frac{1}{2np(v_i, v_{ij})} \times \frac{1}{2np(v_j, v_{ij})} \qquad (\text{from (13)}) \\
&= \sum_{v_i,v_j,v_{ij}} \frac{1}{4n^2} \times \frac{p(v_i, v_j, v_{ij})}{p(v_i, v_{ij}) p(v_j, v_{ij})}
\end{aligned} \tag{25}$$

In the case where X_j is a parent of X_i,

$$\begin{aligned}
\mathrm{E}_{pop}[L_i L_j] =& \mathrm{E}_{pop}\left[\left(\sum_{v_i} 1_{\{Pa^G_{X_i}=v_i\}} x_j L_i^v\right)\left(\sum_{v_j} 1_{\{Pa^G_{X_j}=v_j\}} L_j^v\right)\right] \\
=& \mathrm{E}_{pop}\left[\sum_{v_i,v_j}\left(1_{\{Pa^G_{X_i}=v_i\}} 1_{\{Pa^G_{X_j}=v_j\}} x_j L_i^v \left(x_j \log\frac{p_j^v}{\hat{p}_j^v}\right)\right)\right] \\
=& \sum_{v_i,v_j} p(v_i,v_j,x_j)\,\mathrm{E}_{pop}\left[L_i^v \log\frac{p_j^v}{\hat{p}_j^v}\right] \\
\approx& \sum_{v_i,v_j} p(v_i,v_j,x_j) \times \frac{1}{2np(v_i,x_j)} \times \frac{1-p_j^v}{2np_j^v} (\text{from (13)}) \\
=& \sum_{v_i,v_j} \frac{1-p_j^v}{4n^2} \times \frac{p(v_i,v_j,x_j)}{p(v_i,x_j)p_j^v}
\end{aligned} \tag{26}$$

E Generic Categorical Variable

In this section, we generalize our results to any categorical variables (not just binary). The extension from binary to categorical is straightforward. We will have a similar expression for the likelihood ratio statistic:

$$\begin{aligned}
L(x) =& \log\left(\frac{\Pr(x;\langle G,\theta\rangle)}{\Pr(x;\langle G,\hat{\theta}\rangle)}\right) \\
=& \sum_{i=1}^{m} L_i,
\end{aligned}$$

where L_i is the contribution of X_i to $L(x)$:

$$L_i = \sum_v 1_{\{Pa^G_{X_i}=v\}} L_i^v$$

Instead of writing L_i^v as

$$L_i^v = x_i \log\frac{p_i^v}{\hat{p}_i^v} + (1-x_i)\log\frac{1-p_i^v}{1-\hat{p}_i^v}$$

we write

$$L_i^v = \sum_{o\in V(X_i)} 1_{\{x_i=o\}} \log\frac{p_{io}^v}{\hat{p}_{io}^v},$$

$$p_{io}^v = \Pr(x_i = o | Pa^G_{X_i} = v)$$

Now,

$$\begin{aligned}
\mathrm{E}_{pop}[L_i^v] =& \sum_{o\in V(X_i)} E\left[p_{io}^v \log\frac{p_{io}^v}{\hat{p}_{io}^v}\right] \\
=& \sum_{o\in V(X_i)} \frac{1-p_{io}^v}{2n_i^v} \quad (\text{from (14)}) \\
=& \frac{|V(X_i)|-1}{2n_i^v}
\end{aligned} \tag{27}$$

$$\begin{aligned}
\mathrm{E}_{pop}[L_i] =& \sum_v E[1_{\{Pa^G_{X_i}=v\}} L_i^v | Pa^G_{X_i} = v] \\
=& \sum_v E_{\hat{p}_i^v}\left[E_x\left[1_{\{Pa^G_{X_i}=v\}} L_i^v \mid \hat{p}_i^v\right]\right] \\
=& \sum_v p_v \frac{|V(X_i)|-1}{2n_i^v} \quad \text{(from (27))} \\
=& \sum_v \frac{|V(X_i)|-1}{2n} + O(n^{-2}) \quad \text{(from (18))} \\
=& |V(Pa_{X_i})| \times \frac{|V(X_i)|-1}{2n} + O(n^{-2})
\end{aligned}$$

Now we can calculate $\mathrm{E}_{pop}[L(x)]$ as:

$$\begin{aligned}
\mathrm{E}_{pop}[L(x)] =& \sum_{i=1}^m E_{pop}[L_i] \\
\approx& \sum_{i=1}^m |V(Pa_{X_i})| \times \frac{|V(X_i)|-1}{2n} + O(n^{-2}) \\
=& \frac{C}{2n} + O(Cn^{-2})
\end{aligned}$$

Hence,

$$\mathrm{E}_{pop}[L(x)] = \frac{C}{2n} + O(Cn^{-2}) \tag{28}$$

Similarly for deriving variance we have,

$$\begin{aligned}
\mathrm{E}_{pop}[(L_i^v)^2] =& \sum_{o \in V(X_i)} \mathrm{E}\left[p_{io}^v \left(\log \frac{p_{io}^v}{\hat{p}_{io}^v}\right)^2\right] \\
=& \sum_{o \in V(X_i)} \frac{1-p_{io}^v}{n_i^v} \quad \text{(from (20))} \\
=& \frac{|V(X_i)|-1}{n_i^v}
\end{aligned} \tag{29}$$

Using equation (29), we can calculate $\mathrm{E}_{pop}[L_i^2]$ as:

$$\begin{aligned}
\mathrm{E}_{pop}[L_i^2] =& \sum_v \mathrm{E}[1_{\{Pa^G_{X_i}=v\}} (L_i^v)^2 | Pa^G_{X_i} = v] \\
=& \sum_v \mathrm{E}_{\hat{p}_i^v}\left[E_x\left[1_{\{Pa^G_{X_i}=v\}} (L_i^v)^2 \mid \hat{p}_i^v\right]\right] \\
=& \sum_v p_v \frac{|V(X_i)|-1}{n_i^v} \quad \text{(from (29))} \\
=& \sum_v \frac{|V(X_i)|-1}{n} + O(n^{-2}) \quad \text{(from (18))} \\
=& |V(Pa_{X_i})| \times \frac{|V(X_i)|-1}{n} + O(n^{-2})
\end{aligned}$$

Hence,

$$\begin{aligned}\sum_{i=1}^{m} \mathrm{E}_{pop}[L_i^2] =& \sum_{i=1}^{m} |V(Pa_{X_i})| \times \frac{|V(X_i)|-1}{n} + O(n^{-2}) \\ =& \frac{C}{n} + O(Cn^{-2})\end{aligned}$$

$$\begin{aligned}\mathrm{Var}_{pop}(L) =& \mathrm{E}_{pop}[L^2] - (\mathrm{E}_{pop}[L])^2 \\ \mathrm{E}_{pop}[L^2] =& \sum_{i=1}^{m} \mathrm{E}[L_i^2] + 2 \sum_{1 \le i < j \le m} \mathrm{E}[L_i L_j]\end{aligned}$$

Similar to the derivations of $\sum \mathrm{E}_{pop}[L_i]$ and $\sum \mathrm{E}_{pop}[L_i^2]$, we will have

$$\sum_{i,j} \mathrm{E}_{pop}[L_i L_j] = \frac{C^2}{4n^2} + O(C^2 n^{-2})$$

Hence, for categorical variables:

$$Var_{pop}[L(x)] = \frac{C}{n} + O(C^2 n^{-2}) \tag{30}$$

F Naive Bayes

In section A, while deriving the variance, we haven't calculated the exact value of the $O(C^2n^{-2})$ term. From the released model, it is possible to calculate the exact value of this term. Here we derive the exact value of the $O(C^2n^{-2})$ term, when the released model is a Naive Bayes model. Let the number of attributes in the model be equal to m. Hence, the complexity of the model is $C = 2m - 1$. Let X_1 be the class variable and $p_i^1 = Pr(X_i = 1 | X_1 = 1)$. Then, using equation (13) we have:

$$\begin{aligned}\mathrm{E}_{pop}(L) =& \mathrm{E}_{pop}\left[x_1 \log \frac{p_1}{\hat{p}_1} + (1-x_1) \log \frac{1-p_1}{1-\hat{p}_1} + x_1 \sum_{i=2}^{m} \left(x_i \log \frac{p_i^1}{\hat{p}_i^1} + (1-x_i) \log \frac{1-p_i^1}{1-\hat{p}_i^1} \right) \right. \\ & \left. + (1-x_1) \sum_{i=2}^{m} \left(x_i \log \frac{p_i^0}{\hat{p}_i^0} + (1-x_i) \log \frac{1-p_i^0}{1-\hat{p}_i^0} \right) \right] \\ =& \frac{1}{2n} + \sum_{i=2}^{m} \left[p_1 \times \frac{1}{2np_1} + \frac{1}{2n^2} \left[\frac{1-p_1}{p_1} \right] \right] + \sum_{i=2}^{m} \left[(1-p_1) \times \frac{1}{2n(1-p_1)} + \frac{1}{2n^2} \left[\frac{p_1}{1-p_1} \right] \right] \\ =& \frac{2m-1}{2n} + O(mn^{-2}) \\ =& \frac{C}{2n} + O(Cn^{-2})\end{aligned} \tag{31}$$

We can calculate the exact value of $\mathrm{E}_{pop}(L^2)$ using the equations (19), (25) and (26) as below :

$$\begin{aligned}
\mathrm{E}_{pop}(L^2) = \mathrm{E}_{pop}\Bigg[\Bigg[x_1 \log\frac{p_1}{\hat{p}_1} + (1-x_1)\log\frac{1-p_1}{1-\hat{p}_1} + x_1\sum_{i=2}^{m}\left(x_i\log\frac{p_i^1}{\hat{p}_i^1} + (1-x_i)\log\frac{1-p_i^1}{1-\hat{p}_i^1}\right) \\
+ (1-x_1)\sum_{i=2}^{m}\left(x_i\log\frac{p_i^0}{\hat{p}_i^0} + (1-x_i)\log\frac{1-p_i^0}{1-\hat{p}_i^0}\right)\Bigg]^2\Bigg] \\
= \frac{1}{n} + \sum_{i=2}^{m}\left[p_1 \times \frac{1}{np_1} + \frac{1}{n^2}\left[\frac{1-p_1}{p_1}\right]\right] + \sum_{i=2}^{m}\left[(1-p_1)\times\frac{1}{n(1-p_1)} + \frac{1}{n^2}\left[\frac{p_1}{1-p_1}\right]\right] \\
+ 2\left[\binom{m-1}{2}\times\frac{p_1}{4n^2\hat{p}_1^2} + \binom{m-1}{2}\times\frac{1-p_1}{4n^2(1-\hat{p}_1)^2} + \frac{(m-1)(1-p_1)}{4n^2} + \frac{(m-1)(p_1)}{4n^2}\right] \\
= \frac{2m-1}{n} + \frac{(m-1)(m-2)}{4n^2}\left[\frac{p_1}{\hat{p}_1^2} + \frac{1-p_1}{(1-\hat{p}_1)^2}\right] + O(mn^{-2}) \\
\approx \frac{C}{n} + \frac{m^2}{4n^2}\left[\frac{1}{p_1(1-p_1)}\right] + O(mn^{-2}) \qquad (32)
\end{aligned}$$

Combining equations (31) and (32), we have the variance for Naive Bayes as:

$$\begin{aligned}
\mathrm{Var}_{pop}(L) &= \mathrm{E}_{pop}(L^2) - (\mathrm{E}_{pop}(L))^2 \\
&= \frac{C}{n} + \frac{m^2}{4n^2}\left[\frac{1}{p_1(1-p_1)} - 4\right] + O(mn^{-2}) \\
&\approx \frac{C}{n} + O(C^2n^{-2}) \qquad (33)
\end{aligned}$$

G Understanding the complexity metric - Parameter estimation errors

The complexity of a Bayesian network $\langle G, \theta\rangle$ with discrete random variables is the number of independent parameters used to define its probability distribution.

$$C(\langle G,\theta\rangle) = \sum_{i=1}^{m} |V(Pa_{X_i}^G)|(|V(X_i)| - 1)$$

The parameters θ are estimated from the pool data. To understand the privacy risk of this learning to members of the pool, we need to study the influence a member can have on the value of the parameters. Fisher information quantifies the amount of information a random variable carries about the parameter(s) θ of the probability distribution from which it is generated.

$$I(\theta) = -\mathrm{E}_\theta(\nabla^2 l(\theta)),$$

where $I(\theta)$ is Fisher information, and $l(\theta)$ is the log-likelihood function for θ.

If $\hat{\theta}$ is a Maximum Likelihood Estimate of θ, then it is known that

$$\hat{\theta} = Normal(\theta, I(\hat{\theta})^{-1}).$$

The log-likelihood functions of parameter θ from a PGM $\langle G, \theta\rangle$, given a sample x are typically of the form:

$$\begin{aligned}
l(\theta) &= \log\left[\Pr(x; \langle G,\theta\rangle)\right] \\
&= \sum_{i=1}^{m} l_i
\end{aligned}$$

where l_i is contribution of X_i to the likelihood function:

$$l_i = \sum_{v \in V(Pa^G_{X_i})} 1_{\{Pa^G_{X_i}=v\}} l_i^v$$

$$l_i^v = \sum_{o \in V(X_i)} 1_{\{x_i=o\}} \log p_{io}^v$$

$$l = \sum_{i,v,o} f_{i,v,o}(x_1, x_2, \ldots, x_m) \log(p_{io}^v),$$

where $f_{i,v,o}$ are activator functions (some combination of x_i's) for the parameter p_{io}^v.

$$I(p) = -E_p(\nabla^2 l(p))$$

All the non-diagonal elements of the information matrix are zero, because:

$$\frac{\partial}{\partial p_{io}^v} \frac{\partial}{\partial p_{jo}^v} [\sum_{i,v,o} f_{i,v,o}(x_1, x_2, \ldots, x_m) \log(p_{io}^v)] = 0, \forall i \neq j$$

This implies that all the standard normal variables used to represent frequencies in pool are pair-wise independent i.e. all the estimation errors are independent across parameters. The difference between an estimated parameter value (calculated from the pool) and the actual parameter value (calculated from the general population) is the estimation error that leaks information about the pool. Since these estimation errors are independent across parameters of a Bayesian network, each parameter makes a separate contribution to the power of the attacker. Hence, the complexity measure defined as the number of independent parameters, captures the potential privacy risk of the model.

H What about models trained with differential privacy?

The bound provided in Theorem 1 is computed assuming that the parameters are learned without any privacy defense. The parameters can also be learned with a privacy defense (like differential privacy) in place. The effect of a differentially private learning mechanism on our bound can be better reasoned under the recently introduced notion of "f-differential privacy" (f-DP) [2]. f-DP is a new relaxation of differential privacy based on a framework of hypothesis testing. It characterizes the trade-off between type I and type II errors in distinguishing any two neighboring datasets using a function f. When the function f is from a specific family that characterizes the trade-off between type I and type II errors in distinguishing the two normal distributions $\mathcal{N}(0,1)$ and $\mathcal{N}(\mu,1)$ based on one draw, it is said to be μ-GDP. If the learning mechanism satisfies μ-GDP, then the bound on power of membership inference in Theorem 1 will become:

$$z_\alpha + z_{1-\beta} \leq \mu \tag{34}$$

Corollary 2.13 in the paper [2] provides the relationship between μ-GDP and the standard $(\epsilon, \delta) - DP$.

Corollary 1 [2]: A mechanism is μ-GDP if and only if it is $\big(\epsilon, \delta(\epsilon)\big)$-DP for all $\epsilon \geq 0$, where

$$\delta(\epsilon) = \Phi\Big(-\frac{\epsilon}{\mu} + \frac{\mu}{2}\Big) - \mathrm{e}^{\epsilon}\Phi\Big(-\frac{\epsilon}{\mu} - \frac{\mu}{2}\Big)$$

, and Φ is the CDF of standard normal distribution.

Using Corollary 1 and equation 34, we can calculate how our bound in Theorem 1 changes when the parameters are learned with differential privacy guarantees.

I Evaluation details: Bayesian network learning and Data synthesis

In this section, we describe the methods used in the evaluation for learning structure and parameters of a Bayesian network and generating synthetic data. See [5] for a comprehensive overview of the methods to learn Bayesian networks.

I.1 Structure Learning

The objective is to learn the significant dependencies between random variables, and represent them as a graph. We used an existing algorithm based on maximizing a score function that measures how correlated different attributes are, according to the training data [4]. For each attribute we find a set of attributes which are highly correlated with it, yet are not significantly correlated among themselves.

$$score(Pa_{X_i}^G) = \frac{\sum_{X_j \in Pa_{X_i}^G} corr(X_i, X_j)}{\sqrt{|Pa_{X_i}^G| + \sum_{x_j, x_k \in Pa_{X_i}^G} corr(X_j, X_k)}}, \tag{35}$$

$$corr(X_i, X_j) = 2 - 2\frac{H(X_i, X_j)}{H(X_i) + H(X_j)},$$

where H is the entropy function.

While optimizing this score for each attribute, we need to make sure that the graph remains acyclic. Also, to control the complexity of the graph, we impose a condition on η, the maximum number of parents for each node. We use an iterative and greedy algorithm that adds parents to each node while maximizing the score for all nodes at each iteration, subject to the constraints.

I.2 Parameter Learning

We assume a prior distribution on all possible values of the parameters θ, and use the training data set to update this distribution, using a Bayesian approach.

Let X_i be the random variable for a categorical attribute. Let $\vec{\theta}_i$ be the parameters of the conditional probability $\Pr[X_i | Pa_{X_i}^G; \theta]$. For each assignment of values to $Pa_{X_i}^G; \theta$, we assume a prior distribution on all the possible k-dimensional multinomial distributions. The prior distribution for each assignment v comes from a Dirichlet family, i.e., $\vec{\theta}_i^v \sim Dirichlet(\vec{\alpha}_i^v)$, where $\vec{\alpha}_i^v$ is the hyper parameters of the distribution.

Let $\vec{c}_i^v = [c_{i1}^v, c_{i2}^v, \cdots, c_{ik}^v]$ include the frequency of the events $[X_i = j | Pa_{X_i}^G; \theta = v]$ in training data. We compute the posterior distribution for $\vec{\theta}_i^v$ as $Dirichlet(\vec{\alpha}_i^v + \vec{c}_i^v)$. Thus, the most likely estimation for set of parameters $\vec{\theta}_i^v$ is:

$$\theta_{ij}^v = \frac{\alpha_{ij}^v + c_{ij}^v}{\sum_{j=1}^k (\alpha_{ij}^v + c_{ij}^v)}. \tag{36}$$

In all our experiments, we use a uniform prior i.e., we set $\vec{\alpha}_i^v$ to 1 in all dimensions.

I.3 Data Synthesis

Given a data set D, we want to synthesize datasets that are close in distribution to D. Graphical models could be used for inference and prediction, as well as generating synthetic data (from the underlying distribution that they encode). We use the below process for generating synthetic datasets:

1. Learn a Bayesian network $\langle G, \theta \rangle$ from the data set D.
2. Create a Bayesian network $\langle G', \theta' \rangle$ with $G' = G$, and θ' drawn from the posterior Dirichlet distribution for θ, which was computed during parameter learning.
3. Draw independent samples from $\langle G', \theta' \rangle$.

In our experiments, while generating the synthetic data, we use $\eta = 3$ for learning the structure G of the Bayesian network $\langle G, \theta \rangle$ from the data set D.

J Additional evaluation

J.1 Effect of releasing statistically insignificant edges

We analyze the effect on the power of attack of releasing edges (conditional probabilities) that are statistically insignificant. To perform this evaluation, we consider the case where the structure of released model is not learned from data but generated in a random way.

Figure 1 compares the power of the attack when two different models of almost equal complexity are released. The structure of first model is generated by randomly adding edges and the structure of second model is learned from data. Adversary uses the released model as the population model to calculate the LR statistic and perform tracing attack. We can observe that the attack power is similar in both the cases.

The edges that are generated randomly might not be statistically significant, but they leak about membership. In the LR Test for tracing attack, we rely on the difference between probability distributions for pool and population. Adding a statistically insignificant edge gives similar probability distributions for all configurations of the parent. Although the conditional probabilities are similar, their values will be different for pool and population and hence they will leak about membership. **Statistically insignificant edges leak as much information about membership as significant edges.**

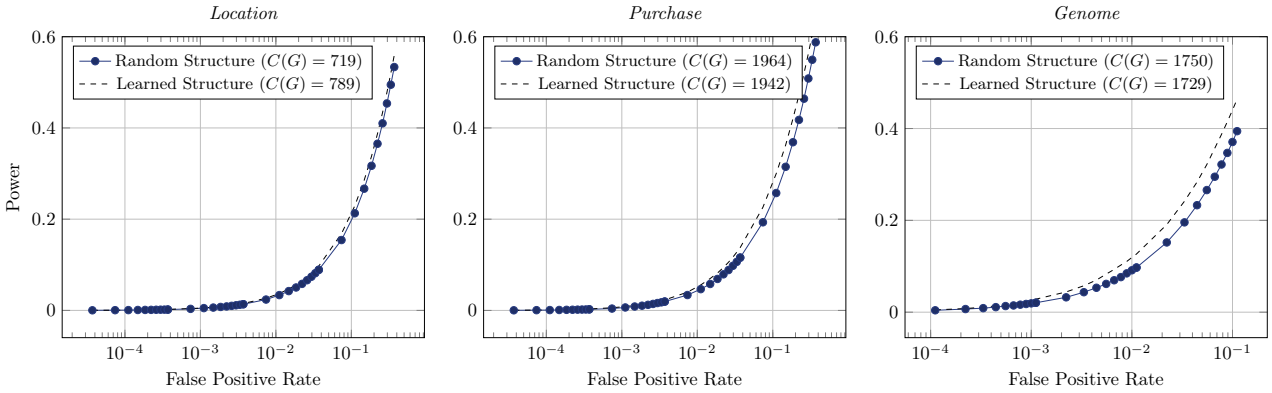


Figure 1: **Effect of Releasing Graphical Models with Random Edges:** Here we compare the power of attack, when two models of similar complexity learned on the dataset are released but structure of one model is learned from data and the structure of other is generated randomly. We can see that for close values of C, the power of attack is almost same for both the models. This shows that statistically insignificant edges leak as much information about membership, as that of significant edges.

J.2 Optimality of the theoretical threshold

In Figure 2, we compare theoretical thresholds for certain false positive rates with their corresponding values estimated using the reference population. The adversary has access to some reference population. For a given false positive rate, the adversary chooses the threshold based on the likelihood ratio on the reference population data. The attacker then runs the LR test tracing attack. When $\eta = 0$ (row 1), we observe that the theoretical threshold values are much higher than the estimated values. When $\eta = 3$ (row 2), the observed thresholds are closer to the estimated values.

When $\eta = 0$, the parameter estimation errors are correlated, which reduces the amount of information leakage. The adversary, when using the theoretical threshold, overestimates the amount of leakage (power) and hence chooses a higher threshold. When $\eta = 3$, the released model captures most of the dependencies among attributes in the data. Hence the observed threshold will be closer to the theoretical threshold values. **From the adversary's perspective, the theoretical threshold value is sub-optimal when the released model is underfitted (loss of utility).**

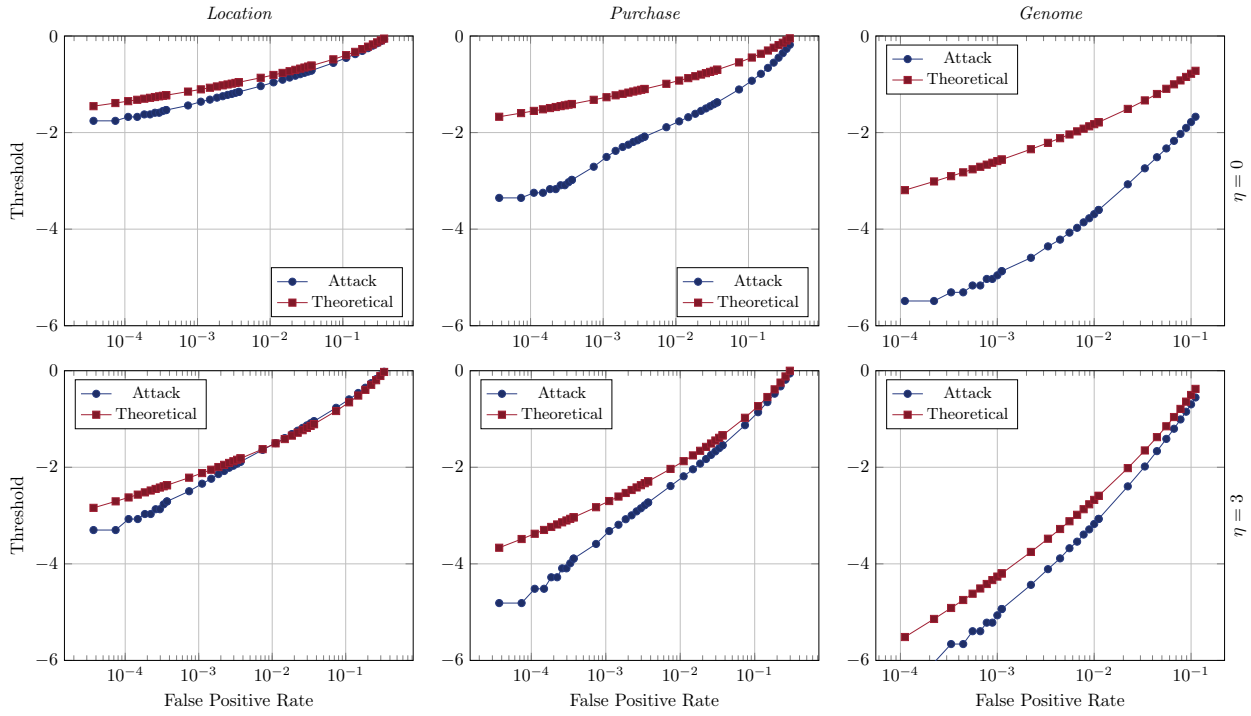


Figure 2: **Effect of releasing underfitted models on threshold selection:** This plot compares the threshold values estimated by the adversary using reference population at different false positive rates with their corresponding theoretical values. The label Attack indicates that the threshold is estimated by the adversary using reference population. We observe that for underfit models ($\eta = 0$) (first row), the threshold value estimated from the reference population is way less than the corresponding theoretical value. As the model gets closer to the generator distribution ($\eta = 3$) (second row), the estimated threshold values get closer to the theoretical values.

J.3 Effect of using a complex model for estimating likelihood of Null hypothesis

In this subsection, we present the effect of population model choice on the behavior of the likelihood ratio test and on the power of the tracing attack. Specifically, we study the effect of using models that are more complex than the released model as population model. The parameters of a graphical model $\langle G, \hat{\theta} \rangle$ with $\eta = 1$ are learned on the pool data and released. The adversary has access to a complex and better representative model $\langle G_{pop}, \theta \rangle$ that was learned with $\eta = 3$. The adversary can choose to use either the released model structure G or a complex model structure G_{pop} as population model structure.

Figure 3 compares the empirical distribution of test statistic (likelihood ratio) values computed on members of the pool and on non-members for both choices of population model on the genome dataset. On the right, we observe that the member distribution is indistinguishable from the non-member distribution when G_{pop} (learned with $\eta = 3$) is used as structure of population model. We also observe that the values of the likelihood ratio are much higher – from 20 to 70 – compared to the values we observe on the left (narrowly concentrated around 0) when the structure of population model is same as that of released model (learned with $\eta = 1$).

When a complex model is used as population model, the likelihood value of the null hypothesis increases for both members and non-members. Hence it cannot help in distinguishing members from non-members. *Also it changes the meaning of the hypothesis test. When a complex model is used to compute the likelihood of null hypothesis, the computed likelihood is no longer the likelihood of the target being a random sample from the population. The meaning of this new hypothesis test would be the following: which of the* ***models is more likely*** *given the target. Since complex models are more likely compared to simpler models, the test statistic (likelihood ratio) values will be very high and positive.* Figure 4 compares the power of the tracing attack for both choices of population model. The power of the attack is higher when the released model structure is used as the population model structure. **Knowledge of additional statistics about population other than the released statistics doesn't increase the power of adversary.**

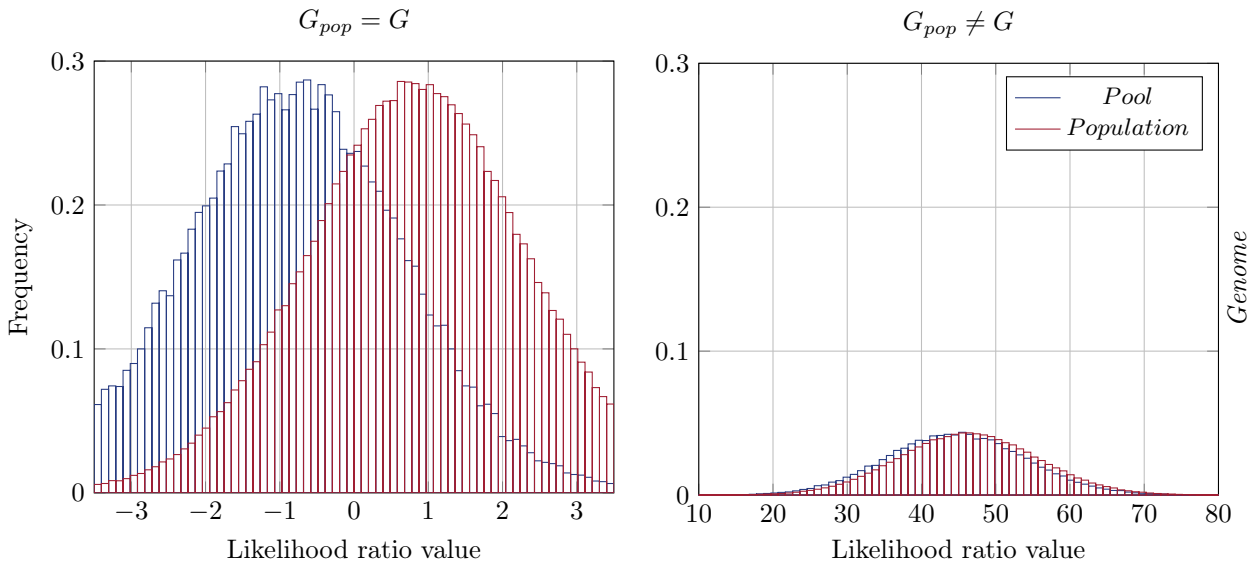
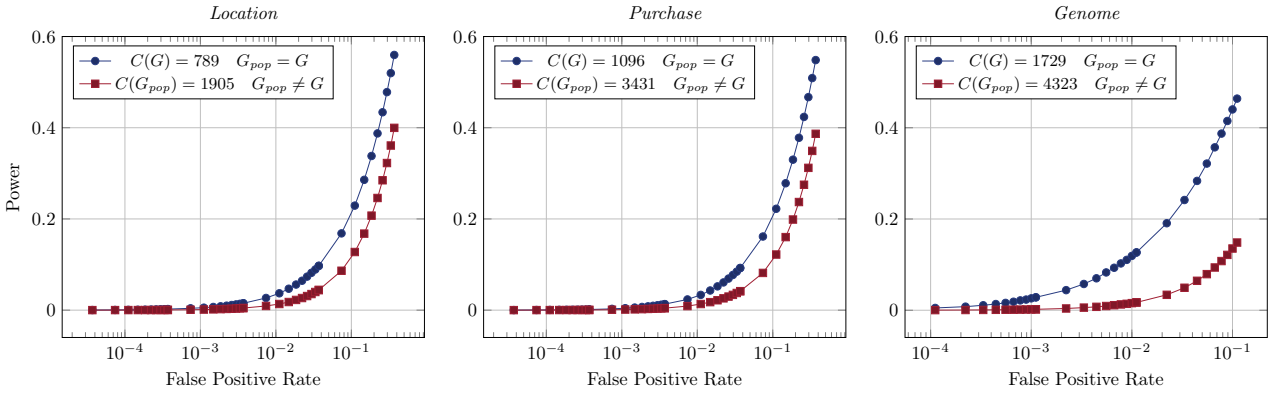


Figure 3: **Comparison of likelihood ratio distributions computed on members of the pool (blue histogram) and on non-members (red histogram): Left:** To calculate the likelihood of the null hypothesis $\mathrm{H}_{\mathrm{OUT}}$, we use the population model $\langle G, \theta \rangle$, whose structure is the same as that of the released model $\langle G, \hat{\theta} \rangle$ ($\eta = 1$). We observe that the member distribution is clearly distinguishable from the non-member distribution. **Right:** To calculate the likelihood of the null hypothesis $\mathrm{H}_{\mathrm{OUT}}$, we use the population model $\langle G_{pop}, \theta \rangle$, whose structure is different and more complex ($\eta = 3$) than the released model $\langle G, \hat{\theta} \rangle$ ($\eta = 1$). We observe that the member/non-member distributions are indistinguishable. Also, the values of the likelihood ratio are much higher compared to the left part of the figure. Using a complex population model might increase the likelihood of null hypothesis $\mathrm{H}_{\mathrm{OUT}}$, but it increases the value for both members and non-members (as a complex model can explain both members and non-members better than a simple model can), making them indistinguishable. Hence the optimal choice of population model for the adversary is the released model estimated over the reference population.

Figure 4: **Effect of using a complex model as population model:** The parameters of a graphical model $\langle G, \hat{\theta} \rangle$ with $\eta = 1$ are learned on the pool data and the model is released. The adversary has access to a better (more complex) generative model $\langle G_{pop}, \theta \rangle$ with $(\eta = 3)$. We observe how the power of the attack changes when calculating the likelihood of null hypothesis using this complex generative model structure instead of the released model structure. We can see that the power of the attack reduces when the population model structure is not the same as the released model structure. As shown in Figure 3, using a complex population model increases the likelihood for both members and non-members and hence cannot help in distinguishing them. This shows that it is not possible to increase the power of adversary using knowledge about additional statistics on the data that are not present in the released graphical model.



J.4 Effect of Biased Sampling

In this section, we empirically study the effect of sampling bias on the power of tracing attack. We model a case of sampling bias, where we discriminate against individuals with some attribute value (say 1). We add a bias in the sampling mechanism for pool, by making the probability of selecting an attribute with value 1 as $1 - bias$.

$$Pr(select|X_i = 0) = 1 \tag{37}$$

$$Pr(select|X_i = 1) = 1 - bias \tag{38}$$

Synthetic data for this experiment was generated from graphs learned on Genome data. Pool is sampled in a biased way as described above. The parameter *bias* can be used alter the amount of sampling bias. We generate a total of 10000 samples, of which we randomly select 2000 as pool and 4000 as reference population.

Figure 5 shows the effect of bias on power of attack. We can clearly observe that power of attack increases with increase in bias. When the pool is drawn from same distribution as population, we leveraged on finite sample estimation error for membership inference. If pool is drawn from distribution that is even slightly different from population distribution, power of attack increases and can be greater than provided bounds. **Biased sampling increases the power of tracing attack**

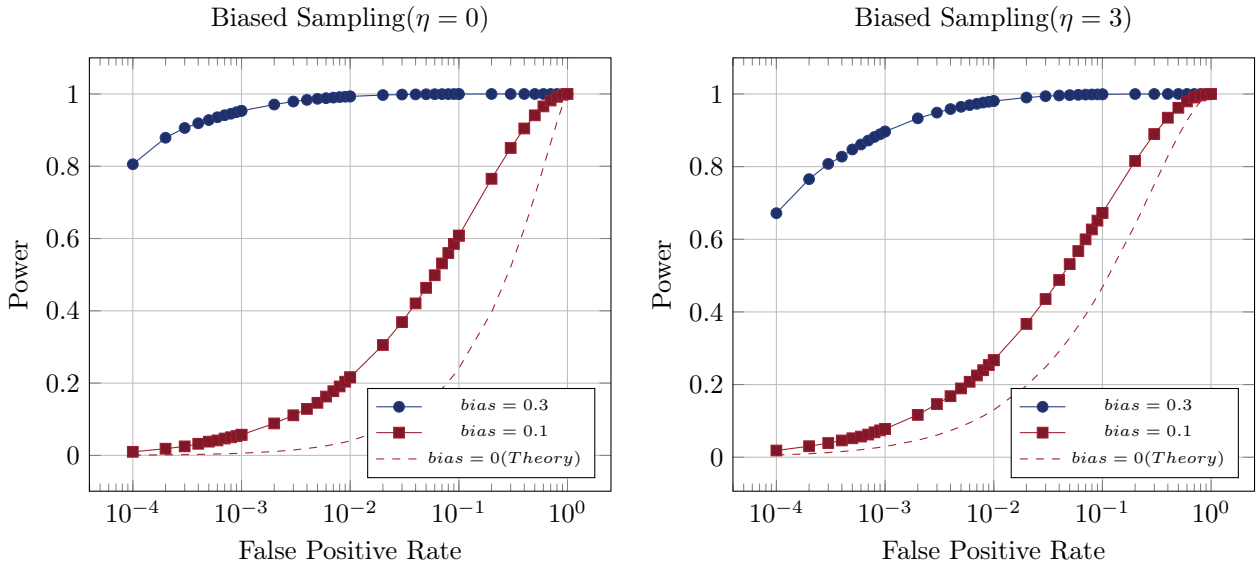


Figure 5: **Comparison of Power values in case of biased sampling:** Figure shows the effect of sampling bias on the power of tracing attack. We generate synthetic data using graph structures of different complexity that are learned on Genome data. The conditional probability values are generated from a Dirichlet distribution fitted to the conditional probabilities in corresponding graph of Genome data. The parameter *bias* is used to tune the bias in sampling of the pool. We can observe that power of attack in case of biased sampling is greater than the theoretical bound (with out considering bias). With increasing value of *bias*, the pool distribution deviates more from the population distribution, which increases the power of attack.

References

[1] A. C. Berry. The accuracy of the gaussian approximation to the sum of independent variates. *Transactions of the american mathematical society*, 49(1):122–136, 1941.

[2] J. Dong, A. Roth, and W. J. Su. Gaussian differential privacy. *arXiv preprint arXiv:1905.02383*, 2019.

[3] C.-G. Esseen. On the liapunoff limit of error in the theory of probability. *Arkiv för matematik, astronomi och fysik*, 1942.

[4] M. A. Hall. Correlation-based feature selection for machine learning. 1999.

[5] D. Koller, N. Friedman, and F. Bach. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.