SPuManTE: Significant Pattern Mining with Unconditional Testing

Leonardo Pellegrina
Dept. of Information Engineering
Università di Padova
Padova, Italy
pellegri@dei.unipd.it

Matteo Riondato
Dept. of Computer Science
Amherst College
Amherst, MA, USA
mriondato@amherst.edu

Fabio Vandin
Dept. of Information Engineering
Università di Padova
Padova, Italy
fabio.vandin@unipd.it

vsqprd: *Vino Spumante di Qualità Prodotto in Regione Determinata (Italian: Quality Sparkling Wine Produced in Determined Region)*

ABSTRACT

We present SPuManTE, an efficient algorithm for mining significant patterns from a transactional dataset. SPuManTE controls the Family-wise Error Rate: it ensures that the probability of reporting one or more false discoveries is less than an user-specified threshold. A key ingredient of SPuManTE is ut, our novel unconditional statistical test for evaluating the significance of a pattern, that requires fewer assumptions on the data generation process and is more appropriate for a knowledge discovery setting than classical conditional tests, such as the widely used Fisher's exact test. Computational requirements have limited the use of unconditional tests in significant pattern discovery, but ut overcomes this issue by obtaining the required probabilities in a novel efficient way. SPuManTE combines ut with recent results on the supremum of the deviations of pattern frequencies from their expectations, founded in statistical learning theory. This combination allows SPuManTE to be very efficient, while also enjoying high statistical power. The results of our experimental evaluation show that SPuManTE allows the discovery of statistically significant patterns while properly accounting for uncertainties in patterns' frequencies due to the data generation process.

CCS CONCEPTS

- **Mathematics of computing → Contingency table analysis**;
- **Information systems → Data mining**.

KEYWORDS

Family-wise Error Rate, Hypothesis Testing, Itemset Mining

ACM Reference Format:
Leonardo Pellegrina, Matteo Riondato, and Fabio Vandin. 2019. SPuManTE: Significant Pattern Mining with Unconditional Testing. In *The 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '19), August 4–8, 2019, Anchorage, AK, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3292500.3330978

1 INTRODUCTION

Significant Pattern Mining [10] is a variant of *Frequent Pattern Mining* [1], in which transactions have binary class labels and the objective is to identify patterns with a *statistically significant association* with one of the two labels. This task finds applications in a wide range of domains where finding *reliable* associations is crucial, such as market basket analysis, social networks, and molecular medicine. Significant patterns supply different information than frequent ones (i.e., those appearing in a fraction at least θ of all the transactions of a dataset), providing important insights in many applications: for example, in molecular medicine, they underscore the molecular features that distinguish two groups of patients (e.g., responsive to therapy vs. unresponsive).

The significance of a pattern is commonly assessed through *statistical hypothesis testing*: a statistical test is used to obtain a p-value that quantifies the probability that the association observed in the data is due to chance. The most commonly used test to assess the association of a pattern with class labels is Fisher's exact test [8]. Fisher's test is a *conditional test*: it assumes that the data generating process only produces datasets in which both the number of transactions in each class *and* the number of transactions in which the pattern appears are the same as in the observed dataset, i.e., it *conditions* on the observed variables of interest.

In contrast, *unconditional tests* such as Barnard's exact test [2] assume that the frequency of the pattern observed in the real dataset is (the realization of) a *random* variable. Unconditional tests therefore assess the association between a pattern and class labels considering also scenarios (i.e., datasets) where the frequency of the pattern is different from what is observed in the real data. The computation of the p-value for unconditional tests is usually more expensive than for conditional tests, since one needs to explore the space of the possible values for *nuisance parameters* describing the (unknown) values of the underlying process that generated the data. In significant pattern mining, the nuisance parameter of a pattern is the probability that it appears in a transaction generated by the underlying process.

Conditional tests and unconditional tests are based on different assumptions regarding how data is generated and collected, namely, whether the variables of interests (e.g., patterns frequencies) would be the same in a different repetition of the experiment (*conditional tests*) or not (*unconditional tests*). To understand when the two situations arise, consider for example the study of the appearance of

KDD '19, August 4–8, 2019, Anchorage, AK, USA

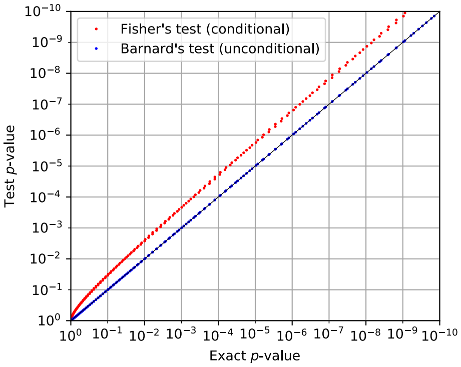

Figure 1: p-values from Fisher's test (conditional) and Barnard's tests (unconditional) vs. exact p-values under the unconditional null hypothesis. p-values of both tests are displayed for all contingency tables with $n = 10^4$, $n_1/n = 0.25$, $f(S) = 0.1$, and $\sigma_1(S) \in [0, \sigma(S)]$ (see Sect. 3 for parameters' definitions) and exact p-value $\geq 10^{-10}$. The diagonal black line corresponds to a test p-value equal to the exact p-value. The p-values from Fisher's test are *smaller* than the exact p-values for many table (the values on the axes *decrease* toward the right and upwards) and may lead to different conclusions (i.e., declaring a pattern as significant when it is not) and to a FWER *higher* than expected.

(behavioral) patterns (e.g., posting information regarding a specific topic) in members of two online communities (defining the two classes). When deciding to collect the data (e.g., whether a user posted information on the topic or not), you may decide to stop once enough members (overall) have posted about the topic. In this case the assumption of conditional tests are met, since every repetition of the experiment would result in the exact same number of appearances of the pattern. In a different situation one may instead decide to collect data for a fixed amount of time: the number of times the pattern appears is, thus, not fixed, and would change among repetitions of the experiment. In this scenario, unconditional tests better reflect the process with which data is generated and collected.

In data mining, the latter scenario is far more common and natural than the former: data is collected from two different groups or conditions for some amount of time, and then the data is analyzed. However, conditional tests such as Fisher's are commonly employed in such scenarios. The difference between using conditional and unconditional tests is usually small when testing the significance of a *single* pattern S: in this case one can flag S as significant if its p-value is below a fixed threshold α with the guarantee that this corresponds to a *false discovery* (i.e., reporting S as significant when it is not) is bounded by α.

The situation is dramatically different in significant pattern mining, where a huge number of patterns appearing in the datasets are tested, resulting in a *multiple hypothesis testing* problem. In this case, if one tests m patterns, the expected number of patterns with p-values below α is $m\alpha$ even when no pattern is significant.

Several methods have been proposed to deal with multiple hypothesis testing [3, 5, 30]. These tests provide various guarantees, among which the most commonly considered are guarantees on the probability of one or more false discoveries, called Family-Wise Error Rate (FWER), and they all require a pattern to have very small p-value in order to be flagged as significant. For patterns with very small p-values, conditional tests and unconditional tests display strikingly different p-values (Fig. 1). In particular, Fisher's exact test produces p-values much lower than the exact p-value when there is no association between patterns and class labels, that may results in a much higher FWER than expected.

To the best of our knowledge, no practical method to identify significant patterns with unconditional testing exists.

Contributions. We present SPuManTE, the first efficient algorithm for mining significant patterns without conditioning on the observed values of the pattern frequencies and while controlling the FWER. In detail, our contributions are the following.

- At the core of SPuManTE is UT, our novel unconditional statistical test for the significance of a single pattern. UT, being unconditional, is more appropriate for significant pattern mining. UT is, to our knowledge, the first computationally efficient unconditional test. To achieve this efficiency, it combines confidence intervals for the *expected* frequency of the pattern, with deep insights on the computation of bounds on the p-value, and a smart strategy to explore the space of contingency tables. UT's usefulness extends beyond its employment in SPuManTE, and may find applications in other significant pattern mining problems.
- SPuManTE controls the FWER at level α, for an user-specified $\alpha \in (0, 1)$. To achieve this goal, we develop an efficient way to compute a lower bound to the p-value for UT, and use it by adapting the strategy used in LAMP [25]. SPuManTE uses UT in combination with recently developed bounds on the maximum deviation of the observed frequency of a pattern from its expectation that hold simultaneously over all patterns [20], rather than having to expensively compute a different confidence interval for each pattern. To the best of our knowledge SPuManTE is the first algorithm in which such uniform bounds have been used, and we believe that this approach could be applied to other methods for significant pattern mining.
- We evaluate SPuManTE on real datasets and compare its performance with the state-of-the-art method LAMP [25], based on Fisher's exact test. The results show that SPuManTE has high statistical power and is faster than LAMP in particular for large datasets, due to the high number of patterns that do not require the explicit computation of the p-value but can be flagged as significant based on the confidence intervals alone.

2 RELATED WORK

Our work focuses on *significant* pattern mining [10]. We develop a significant pattern mining algorithm (called SPuManTE) that controls the FWER while maintaining high statistical power, and uses a novel unconditional test.

Controlling the FWER can be done with classical statistical tools such as the Bonferroni's method, or its refinements [12]. This approach has been proposed in many previous works on significant pattern mining [27–29]. These methods exhibit limited statistical power, because of the very large number of associations being tested. For this reason, SPuMAnTE does not use classical statistical approaches for the control of the FWER. Instead, SPuMAnTE uses a strategy similar to the one proposed in LAMP [17, 25], a recently-introduced method for achieving higher statistical power while controlling the FWER. LAMP is designed for the Fisher's exact test, which is a *conditional.* SPuMAnTE uses a significance-level-search strategy similar to the one in LAMP, modified to be appropriate for our novel test UT, which is *unconditional* and, as argued in Sect. 1, more appropriate for the significant pattern mining setting.

Other works [15, 19, 24] use the Westfall-Young permutation test [30], that is more powerful than LAMP but also more computationally expensive. Our novel unconditional test UT can be used in combination with the Westfall-Young permutation testing as well.

Barnard's test [2] is an unconditional exact test to assess the statistical significance of associations from a 2×2 contingency table. Compared to Fisher's test [8], it offers more statistical power [4, 6, 16], and it only requires conditioning on one set of marginals of the contingency tables, rather than two as Fisher's. The popularity of Barnard's test was hindered partially by Fisher's criticism, and partially by the excessive computational cost required by naïve implementations of the test. We do not enter the debate on which of the two tests to use [7], but our results suggest that unconditional tests like Barnard's exact test may be more appropriate for significant pattern mining. Rather, our work focuses on the computational aspects, and specifically in how to speed up the execution of the test. Our work is similar, in spirit, to that of Hämäläinen [9], who studied computationally efficient upper bounds to the p-value of Fisher's test. Differences between the two statistical tests may become significant for small p-values (see Fig. 1), and thus become important in the multiple hypothesis testing scenario where they may lead to a lack of control on the FWER. Vandin et al. [26] observed such lack of control for the log-rank test. To the best of our knowledge our work is the first time that the observation is reported for Fisher's exact test (see Fig. 1).

We study the fundamental task of significant pattern mining, but extensions to this settings are possible and have been studied. Komiyama et al. [13] look at emerging patterns that significantly differ between two databases. Papaxanthos et al. [18] and Terada et al. [23] present methods to mine significant patterns in the presence of confounding variables and covariates, while Sugiyama et al. [21] focuses on mining significant subgraphs from networks, and He et al. [11] look at the extraction of significant sequential patterns. We believe our method can be extended to these other settings and we will study these extensions in the future.

3 PRELIMINARIES

Let $\mathcal{I}$ be an alphabet of ordered *items*, and let $\{\ell^0, \ell^1\}$ be two *(class) labels*, i.e., items that do not belong to $\mathcal{I}$. A *dataset* $\mathcal{D} = \{(t_1, \ell_1), (t_2, \ell_2), \ldots, (t_n, \ell_n)\}$ is a multiset of $|\mathcal{D}| = n$ pairs (t_i, ℓ_i) where $t_i \subseteq \mathcal{I}$ is a *transaction*, and $\ell_i \in \{\ell^0, \ell^1\}$ is a label, for $1 \le i \le n$. The multiset of the first elements of the pairs in $\mathcal{D}$ is naturally partitioned into two multisets $\mathcal{Z}^0$ and $\mathcal{Z}^1$, where $\mathcal{Z}^i$ contains all and only the first elements of the pairs in $\mathcal{D}$ with second element ℓ^i, for $i \in \{0, 1\}$. We define $n_i = |\mathcal{Z}^i|$, with $n_1 + n_0 = n$.

Variables	$S \subseteq t$	$S \not\subseteq t$	Row totals
$t \in \mathcal{Z}^1$	$\sigma_1(S)$	$n_1 - \sigma_1(S)$	n_1
$t \in \mathcal{Z}^0$	$\sigma_0(S)$	$n_0 - \sigma_0(S)$	n_0
Column totals	$\sigma(S)$	$n - \sigma(S)$	n

Table 1: 2×2 contingency table summarizing the appearance of the pattern S in $\mathcal{Z}^0$ and $\mathcal{Z}^1$.

A *pattern* (or itemset) S is a set of items, $S \subseteq \mathcal{I}$. We say that S *appears* in a transaction t if $S \subseteq t$, and say that t *contains* S. The *support* $\sigma_i(S)$ (resp. *frequency* $f_i(S)$) *of* S *in* $\mathcal{Z}^i$ is the *number* (resp. *fraction*) of transactions in $\mathcal{Z}^i$ that contain S, for $i \in \{0, 1\}$. With a slight abuse of notation, we denote as $\sigma(S)$ (resp. $f(S)$) the number (resp. fraction) of transactions in the pairs of $\mathcal{D}$ that contain S. Thus, $f_i(S) = \sigma_i(S)/n_i$, $i \in \{0, 1\}$, and $f(S) = \sigma(S)/n$. For any pattern S, these quantities (and their complements) are summarized in a 2×2 contingency table such as the one in Table 1.[1] We conveniently define the quantities $\breve{\sigma}_{i,x} = \max\{0, n_i - (n - x)\}$ and $\widehat{\sigma}_{i,x} = \min\{n_i, x\}$, that define the range $[\breve{\sigma}_{i,x}, \widehat{\sigma}_{i,x}]$ of the admissible values of $\sigma_i(S)$ for any S with $\sigma(S) = x$.

In significant pattern mining, the dataset $\mathcal{D}$ is assumed to be the outcome of a stochastic process that generates sets of pairs (t, λ). Different assumptions can be made on this process (details in Sect. 3.1 and 3.2). Independently on the assumptions, for any pattern $S \subseteq \mathcal{I}$, we let $\pi_{S,i}$ be the probability that a pair (t, λ) generated by the process is such that $S \subseteq t$ and $\lambda = \ell^i$, for $i \in \{0, 1\}$.

The key task in significant pattern mining is to identify the patterns that exhibit a *significant association* with one of the two labels, i.e., for which $\pi_{S,0} \neq \pi_{S,1}$. Given a single pattern S, assessing the statistical significance of an association corresponds to using the *observed* contingency table for S to evaluate whether it *supports* the *null hypothesis* $H_S : \pi_{S,0} = \pi_{S,1}$, i.e., whether the observed data $\mathcal{D}$ is likely to have been generated from a process satisfying H_S.

All available information about S is contained in the contingency table, thus we cannot be *deterministically certain* in our assessment of the significance of the association of S with one label: due to the randomness involved in the data generation process, there is the possibility of flagging the association as significant when it is not, i.e., of making a *false discovery.*

The assessment of whether the null hypothesis for S is supported by the observed contingency table for S involves computing a *p-value* p_S. This quantity is defined as the probability, w.r.t. the data generating distribution and under the assumption that the null hypothesis is true, of observing a contingency table for S that is *as or more extreme* (i.e., has equal or lower probability of being observed) than the one that is actually observed. The set of more extreme contingency tables and the probability associated with each of them depend on the *conditions* imposed on the data generation process (details in Sect. 3.1 and 3.2).

No matter what the conditions are, once the value of p_S or an *upper bound to it* is known, *rejecting* the null hypothesis H_S, i.e., flagging the pattern S as having a significant association with one of the two labels *iff* $p_S \le \delta$ ensures that the probability (w.r.t. the

[1] The "2×2" attribute refers to the four central cells of the table, which contain the actual information about the pattern S.

randomness in the generating process) that S is a false discovery is not greater than $\delta \in (0,1)$.

3.1 Conditional testing

Let's focus for now on a single pattern $S \subseteq \mathcal{I}$. A set of conditions commonly imposed on the data generating process, made for example by the widely-used Fisher's exact test [8], is to *condition on all values in the marginals* (i.e., in the bottom row and rightmost column) of the observed contingency table for S: the space of possible contingency tables for S contains all and only those with the same values for n, n_1 (thus n_0), and $\sigma(S)$, as in the observed one.

Under these conditions and assuming that the null hypothesis H_S is true, the quantity $\sigma_1(S)$ follows a *hypergeometric distribution*: given $x = \sigma(S)$ and $a \in [\breve{\sigma}_{1,x}, \widehat{\sigma}_{1,x}]$, the probability $P^{\mathsf{F}}(a)$ (the F stands for "Fisher") of observing a contingency table for S where $\sigma_1(S) = a$ is

$$P^{\mathsf{F}}(a) = \frac{\binom{n_1}{a}\binom{n_0}{\sigma(S)-a}}{\binom{n}{\sigma(S)}} \enspace .$$

Let b be the value for $\sigma_1(S)$ in the observed contingency table for S, the p-value p_S^{F} is then

$$p_S^{\mathsf{F}}(b) = \sum_{a \,:\, P^{\mathsf{F}}(a) \le P^{\mathsf{F}}(b)} P^{\mathsf{F}}(a) \enspace . \tag{1}$$

Drawbacks. Imposing on the generative process the conditions we just described may be reasonable when only considering a single pattern S. In the significant pattern mining setting, the space of possible patterns is the powerset of $\mathcal{I}$. Since there is a single generative process, one would have to impose on it that, for *each* S of the $2^{|\mathcal{I}|}$ possible patterns, the generative process only generates contingency tables for S with the observed value of $\sigma(S)$. Imposing such a large number of conditions seems excessively restrictive.

3.2 Unconditional testing

A more reasonable set of conditions is to consider only n and n_1 (and thus n_0) fixed as in the observed dataset. These quantities are characteristics of the observed dataset, and much more readily accessible than the observed frequencies (in $\mathcal{D}$) of any of the patterns. With a somewhat unfortunate name choice, tests that impose such conditions, e.g., Barnard's (exact) test [2], are known as *unconditional tests.* For a pattern S, the space of possible contingency tables contains all and only those with the same values for n and n_1 (thus n_0), as in the observed one. The value for $\sigma(S)$ is not fixed, hence not known a priori. This set of assumptions is much more reasonable for the significant pattern mining setting, where the value of $\sigma(S)$ for each pattern S is unknown and must be obtained by running a pattern mining algorithm on the dataset.

To define the p-value for S we need to first introduce the concept of the *nuisance parameter* $\pi \in [0,1]$. The nuisance parameter is the *assumed value* for $\pi_{S,0}$ and $\pi_{S,1}$ under the null hypothesis that these quantities are equal.

Under the above conditions, and assuming that the null hypothesis H_S is true, and a fixed, known value for the nuisance parameter π, the probability of observing a contingency table for S with $\sigma(S) = x$ and $\sigma_1(S) = a$, for $x \in [0,n]$, and $a \in [\breve{\sigma}_{1,x}, \widehat{\sigma}_{1,x}]$, is

$$P(x, a \mid \pi) = \binom{n_0}{x-a}\binom{n_1}{a}\pi^x(1-\pi)^{(n-x)} \enspace .$$

For $y \in [0,n]$, $b \in [\breve{\sigma}_{1,y}, \widehat{\sigma}_{1,y}]$, and $\pi \in (0,1)$, define $T(y,b,\pi)$ as the set of pairs (x,a) such that $P(x,a \mid \pi) \le P(y,b \mid \pi)$ and define the function

$$\phi(y,b,\pi) = \sum_{(x,a) \in T(y,b,\pi)} P(x,a \mid \pi) \enspace .$$

The value $\phi(y,b,\pi)$ is the probability, under the null hypothesis $\pi_{S,0} = \pi_{S,1}$ and for a fixed value π of the nuisance parameter, to observe a contingency table as or more extreme than the one with $\sigma(S) = y$, $\sigma_1(S) = b$, and $\sigma_0(S) = y - b$ (see Table 1).

To obtain the actual p-value for a pattern S or an upper bound to it (sufficient to perform the test), it is necessary to eliminate the dependency on nuisance parameter π. For example, Barnard's test [2] uses as upper bound p_S^{B} the maximum of $\phi(\sigma(S), \sigma_1(S), \pi)$ over all values of $\pi \in (0,1)$:

$$p_S^{\mathsf{B}} = \max\left\{\phi(\sigma(S), \sigma_1(S), \pi), 0 \le \pi \le 1\right\} \enspace . \tag{2}$$

Finding this maximum is computationally expensive. One of our goals is designing an unconditional test with an upper bound to the p-value that is efficient to compute (see Sect. 4.1).

3.3 Multiple hypothesis testing and LAMP

When a single pattern S is tested, flagging it as *significant* when its p-value is smaller than a *significance threshold* $\delta \in [0,1]$, fixed *a priori*, guarantees that the probability of a false discovery (i.e., reporting S as significant when it is not) is bounded by δ. If such approach is followed when testing $h > 1$ patterns (i.e., *multiple hypotheses*), the expected number of false discoveries could be as large as δh. As h grows large, as is typically the case for significant pattern mining, the probability of having at least one (and possibly many) false discovery also grows. An appropriate *multiple hypothesis testing correction* of the significance threshold used for testing each individual pattern is needed to ensure a low probability of false discoveries.

One common approach is to perform such a correction to bound the *Family-Wise Error Rate* FWER, i.e., the probability of reporting *at least one* false discovery, where the sample space is the set of possible datasets generated by the underlying process.

For a given $\delta \in [0,1]$, using δ as the significance threshold for each test, i.e., rejecting all null hypotheses with p-value $\le \delta$ would lead to a FWER $\mathsf{FWER}(\delta)$. In most cases it is not possible to derive a closed formula for FWER; a more tractable and desirable inverse approach allows to obtain, given an acceptable-FWER threshold $\alpha \in (0,1]$, a so-called *corrected significance threshold* δ to use in each test in order to have $\mathsf{FWER}(\delta) \le \alpha$. A simple, but conservative, way to set δ is the Bonferroni's correction [5]: given α, use $\delta = \alpha/h$ to guarantee that $\mathsf{FWER}(\delta) \le h\delta = \alpha$. The problem with Bonferroni's correction is that, for large h, δ is close to 0, resulting in low *statistical power*, i.e., many actually significant patterns are not flagged as such [27–29]. More sophisticated techniques have been devised to obtain higher statistical power while keeping the FWER at most α. Tarone [22] introduces the notion of *minimum attainable (min. att.) p-value*: if we use a corrected significance threshold δ, patterns whose min. att. p-value is greater than δ, called *untestable*, do not need to be counted among the h to be tested, thus should not be considered in the computation of δ with the Bonferroni's correction. Defining $\mathrm{k}(\delta)$ as the number of patterns with min. att. p-value $\leq \delta$, the corrected significance threshold can be obtained as $\delta = \alpha/\mathrm{k}(\delta)$.

Terada et al. [25] present LAMP, a significant pattern mining algorithm that uses the concept of min. att. p-value for Fisher's exact test: for a given pattern S, its p-value p_S can be expressed as a function of $\sigma_1(S)$ only (see (1)). Since the set of possible values of $\sigma_1(S)$ for $\sigma(S) = x$ is $[\check{\sigma}_{1,x}, \widehat{\sigma}_{1,x}]$, the min. att. p-value $\psi(S)$ is

$$\psi^{\mathsf{L}}(S) = \min\left\{p^{\mathsf{F}}_S(a), a \in [\check{\sigma}_{1,x}, \widehat{\sigma}_{1,x}]\right\} \quad .$$

For a candidate corrected significance threshold δ, if $\psi^{\mathsf{L}}(S) > \delta$ the pattern S will never be flagged as significant and is therefore *untestable*. In order to use the min. att p-value in a significant pattern mining setting, Terada et al. [25] introduce an easy-to-compute lower bound $\check{\psi}^{\mathsf{L}}(S)$ to $\psi^{\mathsf{L}}(S)$ (assuming w.l.o.g. that $n_1 \leq n_0$):

$$\check{\psi}^{\mathsf{L}}(S) = \begin{cases} \psi^{\mathsf{L}}(S) & \text{if } 0 \leq \sigma(S) \leq n_1 \\ 1/\binom{n}{n_1} & \text{if } n_1 < \sigma(S) \leq n \end{cases}$$

which is *monotonically non-increasing* in the support of S. Since n and n_1 are fixed by the dataset, $\check{\psi}^{\mathsf{L}}(S)$ depends only on the support $\sigma(S)$ of S, that is, $\check{\psi}^{\mathsf{L}}(S)$ is a function $g(\sigma_S)$ of σ_S. Therefore the number $\mathrm{k}(\delta)$ of patterns with min. att. p-value $\leq \delta$ is equal to the number of patterns $k(\sigma_\delta)$ with support at least σ_δ. LAMP employs the lower bound $\check{\psi}^{\mathsf{L}}(S)$ to efficiently explore the lattice of *closed* patterns in order to identify the maximum value of the support $\sigma_{\max}$ such that $g(\sigma_{\max} - 1) > \alpha/k(\sigma_{\max} - 1)$ while $g(\sigma_{\max}) \leq \alpha/k(\sigma_{\max})$.

To identify a suitable significance thresold δ^*, SPuMaNTE employs a strategy similar to the one in LAMP, but uses a new unconditional test (see Sect. 4.1), thus requiring the development of an efficiently computable lower bound to the min. att. p-value for a pattern S using such test (Sect. 4.2.2).

4 SIGNIFICANT PATTERN MINING WITH UNCONDITIONAL TESTING

We now describe SPuMaNTE, our algorithm for significant pattern mining with unconditional testing. We start by introducing UT, a novel Unconditional Test that is based on confidence intervals for the expected frequencies of the patterns. Due to space constraints, some of our proofs are presented in the Appendix.

4.1 The UT test

Let S be a pattern of which we are assessing the association with the labels. Our novel Unconditional Test UT assumes to know two confidence intervals $C_0(S)$ and $C_1(S)$, for $\pi_{S,0}$ and $\pi_{S,1}$, respectively, s.t. the event E_S="$\pi_{S,0} \in C_0(S)$ and $\pi_{S,1} \in C_1(S)$" holds with probability $1 - \gamma$ (over the randomness in the data generation process). We discuss in Sect. 4.2.1 how such confidence intervals can be obtained *simultaneously* for all patterns S. Let the interval $C(S)$ be $C(S) = C_0(S) \cap C_1(S)$. We define the p-value p_S conditioned on the event E_S as

$$p_S = \begin{cases} 0 & \text{if } C(S) = \emptyset \\ \max\{\phi(\sigma(S), \sigma_1(S), \pi), \pi \in C(S)\} & \text{othw.} \end{cases} \quad .$$

This p-value should be compared with the one from (2). The p-value p_S is a *conditional probability*: it is the probability of observing contingency tables at least as extreme as the one seen in the dataset *conditioning* on the event E_S. This conditioning is entirely different than the conditioning made by conditional tests such as Fisher's [8], which conditions on the *observed* supports of the patterns.

Given a fixed threshold α, UT flags S as significant *iff* $p_S \leq \alpha - \gamma$. The following property holds.

THEOREM 4.1. *Let S be a fixed pattern. The probability that UT flags S as significant when it is not is at most α.*

PROOF. Let F be the event "UT flags S as significant when it is not", which corresponds to a false discovery. It holds

$$\Pr(F) = \Pr(F \mid E_S)\Pr(E_S) + \Pr(F \mid \bar{E}_S)\Pr(\bar{E}_S) \leq \Pr(F \mid E_S) + \Pr(\bar{E}_S),$$

where $\bar{E}_S$ denotes the event complementary to E_S. By the definition of confidence interval, $\Pr(\bar{E}_S) \leq \gamma$, while $\Pr(F \mid E_S) \leq \alpha - \gamma$ since when E_S holds we are using the standard hypothesis testing framework with significance threshold $\alpha - \gamma$ and the p-value p_S. □

4.1.1 *An upper bound to the p-value.*

The exact computation of p_S when $C(S) \neq \emptyset$ requires an expensive search over the values of $\pi \in C(S)$. For the purpose of testing the significance of a pattern and ensuring that Thm. 4.1 still holds, only an efficient-to-compute *upper bound* to the p-value is needed. We prove the following.

THEOREM 4.2. $p_S \leq P(\sigma(S), \sigma_1(S) \mid f(S))(n_0 + 1)(n_1 + 1)$.

The upper bound $\widehat{p}_S = P(\sigma(S), \sigma_1(S) \mid f(S))(n_0 + 1)(n_1 + 1)$ can be computed in $O(1)$ time.

4.1.2 *A lower bound to the p-value.*

We now show a lower bound to p_S. More than being just of theoretical interest, this lower bound is the starting point to derive, in Sect. 4.2.2, an efficiently-computable, monotonically-non-increasing lower bound to the minimum attainable p-value for a pattern S (required to compute the corrected significance threshold in a way similar to what is done by LAMP [25]).

Computing our lower bound does not require an expensive search over the values of π, thanks to the following result.

LEMMA 4.3. *If $C(S) \neq \emptyset$, then $p_S \geq \phi(y, b, \pi)$ for any $\pi \in C(S)$.*

PROOF. Follows from the definition of p_S, that is the maximum over $\pi \in C(S)$ of the r.h.s. □

Lemma 4.3 states that any $\pi \in C(S)$ allows to obtain a lower bound to p_S, so we choose $\pi = f(S)$, and define the lower bound $\check{p}_S$ to p_S as

$$\check{p}_S = \phi(\sigma(S), \sigma_1(S), f(S)) \quad . \tag{3}$$

Our choice for π is driven by the objective of maximizing the number of contingency tables in $T(\sigma(S), \sigma_1(S), \pi)$, which we try heuristically to achieve by maximizing $P(\sigma(S), \sigma_1(S) \mid \pi)$, which is straightforward as shown in the following result.

PROPOSITION 4.4. *It holds* $\arg\max_\pi\{P(x, a \mid \pi)\} = x/n$.

We show in Sect. 5.1 that $\check{p}_S$ provides a tight lower bound to p_S.

4.2 SPuMaNTE: mining significant patterns

We now describe SPuMaNTE, our algorithm for mining significant patterns with UT while controlling the FWER. First, we need to discuss some important technical facts.

4.2.1 Simultaneous confidence intervals. For each tested pattern S, UT requires confidence intervals for $\pi_{S,0}$ and $\pi_{S,1}$. Rather than computing these confidence intervals separately for each pattern, SPuManTE uses recently developed probabilistic bounds to the maximum deviation of the frequency of a pattern from its expectation [20] to derive confidence intervals for the quantities $\pi_{S,0}$ and $\pi_{S,1}$ of *every* pattern that hold *simultaneously* with high probability. Specifically, given $\mathcal{D}$ and a confidence parameter $\gamma \in (0,1)$, we use a modified version of the work by Riondato and Upfal [20], called AMIRA, to obtain a value $\varepsilon \in (0,1)$ with the following property. Given this ε, define, for each pattern S, the intervals $C_0(S)$ and $C_1(S)$ as

$$C_0(S) := \left[f_0(S) - \varepsilon \frac{n}{n_0}, f_0(S) + \varepsilon \frac{n}{n_0} \right],$$

$$C_1(S) := \left[f_1(S) - \varepsilon \frac{n}{n_1}, f_1(S) + \varepsilon \frac{n}{n_1} \right],$$

and define the event $E_{S,\varepsilon}$ = "$\pi_{S,0} \in C_0(S)$ *and* $\pi_{S,1} \in C_1(S)$". Consider the event $E_\varepsilon = \bigcap_{S \subseteq \mathcal{I}} E_{S,\varepsilon}$. The ε returned by AMIRA when run on $\mathcal{D}$ with confidence parameter γ is such that E_ε holds with probability at least $1-\gamma$ (w.r.t. the randomness in the data generative process).

4.2.2 Lower bound to the min. att. p-value. In order to use UT in our algorithm SPuManTE to efficiently mine significant patterns while rigorously controlling the FWER, we need to define a *monotone lower bound* $\breve{\psi}(x)$ to the minimum attainable p-value of any S for which $\sigma(S) \le x$, $x \in [0,n]$. Having this lower bound is crucial to prune the search space of testable patterns.

Our bounds crucially hinges on the following results, under the ongoing assumption $n_1 \le n_0$.

Theorem 4.5. *Let $C_0(S) \cap C_1(S) = C(S) \neq \emptyset$. Then $f(S) \in C(S)$.*

Theorem 4.6. $\arg\min_a\{\phi(x,a,\pi)\} = \min\{x, n_1\}$.

For $x \in [0,n]$ let

$$Q(x) = \Bigg\{ (r,w), r \in [\breve{\sigma}_{0,y}, \widehat{\sigma}_{0,y}], w \in [\breve{\sigma}_{1,y}, \widehat{\sigma}_{1,y}] \text{ s.t. } y = r + w \le x$$
$$\wedge \left(\frac{r}{n_0} + \varepsilon \frac{n}{n_0} < \frac{w}{n_1} - \varepsilon \frac{n}{n_1} \vee \frac{w}{n_1} + \varepsilon \frac{n}{n_1} < \frac{r}{n_0} - \varepsilon \frac{n}{n_0} \right) \Bigg\}.$$

Intuitively, $Q(x)$ contains all the pairs (r,w) such that the confidence intervals around r/n_0 and w/n_1 do not intersect. Checking whether $Q(x)$ is non-empty (i.e., there exists *at least* one contingency table with marginal $\le x$ where the confidence intervals do not intersect) can be done by checking if $Q(x)$ contains either $(\breve{\sigma}_{0,x}, \widehat{\sigma}_{1,x})$ or $(\widehat{\sigma}_{0,x}, \breve{\sigma}_{1,x})$: if $Q(x)$ does not contain either, then it must be empty because the frequencies in each class of less *biased* contingency tables have smaller absolute difference w.r.t. those two cases, therefore it is not possible that the intersection of their confidence intervals is empty. Thus, we can define

$$\psi(x) = \begin{cases} 0 & \text{if } Q(x) \neq \emptyset \\ \phi(x,x,x/n) & \text{otherwise} \end{cases}$$

as a lower bound to the minimum attainable p_S for all patterns S with $\sigma(S) = x$. For our purposes, we need a *monotonically non-increasing* lower bound to the min. att. p-value, so we define

$$\breve{\psi}(x) = \begin{cases} \psi(x) & \text{if } x = 0 \\ \min\{\psi(x), \breve{\psi}(x-1)\} & \text{if } x \in [1,n] \end{cases}.$$

SPuManTE uses $\breve{\psi}$ to check whether to mark a pattern S with $\sigma(S) = x$ as untestable when looking for the corrected significance threshold δ (see Sect. 4.2.4). We observe in practice that $\min\{\psi(x), \breve{\psi}(x-1)\} = \psi(x)$ for all $x \le n_1$, i.e., $\psi(x)$ appears to be monotone w.r.t. x. The computation of $\breve{\psi}(x)$ can be done efficiently starting from $x = 0$ and increasing x, keeping the values of $\breve{\psi}(x)$ in memory. We describe an even simpler approach later in Sect. 4.2.4.

4.2.3 Efficient computation of ϕ. After having defined $\breve{p}_S$ and $\breve{\psi}(x)$, we still have to address how to compute them efficiently in the case $C(S) \neq \emptyset$, i.e., how to compute the value $\phi(y,b,\pi)$ efficiently. A naïve approach is to enumerate *all* $(x, a_1) \in T(y,b,\pi)$; since this set is not known a priori (i.e., there is no simple algorithm to generate *only* pairs (x,a_1) that are in $T(y,b,\pi)$), this approach requires computing $P(a_0 + a_1, a_1 \mid \pi)$ for all possible pairs $(a_0, a_1) \in [0,n_0] \times [0,n_1]$, leading to the computation of $\Theta(n_0 n_1)$ probabilities. As we show in Sect. 5.1, even for samples of moderate size this approach is not feasible in reasonable time.

Enumerating only pairs $(x,a_1) \in T(y,b,\pi)$ or only pairs $(x,a_1) \notin T(y,b,\pi)$ would still require to evaluate $P(x, a_1 \mid \pi)$ a corresponding number of times, i.e., in the order of $\Theta(\min\{|T(y,b,\pi)|, n_0 n_1 - |T(y,b,\pi)|\})$, which is impractical for most cases. We address this issue with an efficient algorithm to compute $\phi(y,b,\pi)$ while avoiding the enumeration of many contingency tables, thanks to the novel formulation of $\phi(y,b,\pi)$ provided by the following result.

Proposition 4.7. *Let $y \in [0,n]$, $b \in [\breve{\sigma}_{1,y}, \widehat{\sigma}_{1,y}]$, and $\pi \in (0,1)$. Let $A_1 = \{a_1 : P(a_1 + \lfloor (n_0+1)\pi \rceil, a_1 \mid \pi) > P(y,b \mid \pi)\}$, and define the set $A_{0,a_1} = \{a_0 : P(a_1 + a_0, a_1 \mid \pi) \le P(y,b \mid \pi)\}$. Then*

$$\sum_{(x,a) \in T(y,b,\pi)} P(x,a \mid \pi)$$
$$= \sum_{a_1 \notin A_1} B(a_1, n_1, \pi) + \sum_{a_1 \in A_1} \left(B(a_1,n_1,\pi) \sum_{a_0 \in A_{0,a_1}} B(a_0,n_0,\pi) \right), \tag{4}$$

where $B(z,h,\pi) = \binom{h}{z} \pi^z (1-\pi)^{h-z}$ is the probability of obtaining z successes on h independent trials with success probability π.

This formulation leads to the efficient algorithm to compute $\phi(y,b,\pi)$ shown in Alg. 1, where we use the *incomplete beta function* to compute the cumulative distribution function (CDF) for Binomial distributions.

In fact, if we let $F(a,n,\pi) = \sum_{a'=0}^{a} B(a',n,\pi)$ be the CDF for value a of the Binomial distribution of parameters n, π, we can compute the terms of (4) with $O(1)$ computations of the incomplete beta function $\beta_{1-\pi}(n+1-a, a+1) = F(a,n,\pi)$, that is efficiently computable using Lentz's algorithm [14], a fast and precise method to evaluate continued fractions. The algorithm makes two such calls in total: the first is performed at the *first* iteration of the **forall** loop on line 9, while the second on line 12. The computation of

Algorithm 1: Efficient computation of $\phi(y, b, \pi)$

```
   Input: $y \in [0, n]$, $b \in [\check{\sigma}_{1,y}, \widehat{\sigma}_{1,y}]$, $\pi \in [0, 1]$.
   Output: $\phi(y, b, \pi)$.
1  $v \leftarrow 0$
2  $z \leftarrow P(y, b \mid \pi)$
3  $a'_0 \leftarrow \lfloor (n_0 + 1)\pi \rceil$
4  $A_1 \leftarrow \left\{a_1 : P\left(a_1 + a'_0, a_1 \mid \pi\right) > z\right\}$
5  forall $a_1 \in A_1$ do
6      $a' \leftarrow \min_{a_0} \left\{a_0 \leq a'_0 \mid P\left(a_0 + a_1, a_1 \mid \pi\right) > z\right\}$
7      $a'' \leftarrow \min_{a_0} \left\{a_0 > a'_0 \mid P\left(a_0 + a_1, a_1 \mid \pi\right) \leq z\right\}$
8      $p' \leftarrow \binom{n_1}{a_1}(\pi)^{a_1}(1 - \pi)^{(n_1 + a_1)}$
9      $v \leftarrow v + \beta_\pi(a', n_0 - a')p' + \beta_{1-\pi}(n_0 + 1 - a'', a'')p'$
10 $a' \leftarrow \min_{a_1} \{a_1 > \max\{A_1\}\}$
11 $a'' \leftarrow \max_{a_1} \{a_1 < \min\{A_1\}\}$
12 $v \leftarrow v + \beta_\pi(a' + 1, n_0 + 1 - a') + \beta_{1-\pi}(n_0 + 1 - a'', a'')$
13 return $v$
```

values a' and a'' (lines 6–7) across *all* iterations of the **forall** loop requires two binary searches for the first iteration, while their values and the updates of v in subsequent iterations can be computed with $O(1)$ operations by updating their previously computed values. Therefore, lines 6–7 require $O(\log(n_0))$ operations across *all* iterations of the **forall** loop. As we prove in the Appendix, the time complexity of Alg. 1 is thus $O\left(\log(n_0) + |A_1| + n_0 - |A_{0,a_1}| + La\right)$, where $O(La)$ is the time complexity of Lentz's algorithm.

4.2.4 SPuManTE. Our algorithm SPuManTE outputs a set of significant patterns in $\mathcal{D}$ with FWER $\leq \alpha$. Its pseudocode is presented in Alg. 2. SPuManTE first obtains (line 1) the maximum deviation ε from AMIRA with parameter γ, so that the event E_ε holds with probability $\geq 1 - \gamma$. Then (line 2), SPuManTE uses the lower bound $\check{\psi}(x)$ derived in Sect. 4.1.2 (and computed using Alg. 1) together with a strategy similar to the one in LAMP [25] to efficiently derive a corrected significance threshold δ to use in each test while ensuring that the FWER is at most $\alpha - \gamma$. In particular, such strategy [17] initializes the support threshold of *testable patterns* σ_T to 1, and increases it while exploring the closed patterns, reducing the set of testable patterns until the final value of δ is found. Hence, we can incrementally compute the values of $\check{\psi}(x)$ after increasing σ_T by simply comparing $\check{\psi}(\sigma_T - 1)$ to $\psi(\sigma_T)$, therefore only keeping in memory $\check{\psi}(\sigma_T - 1)$ and not the entire function $\check{\psi}(x)$. SPuManTE then loops over the testable patterns to test them, to decide whether to flag them as significant or not. It does so by first generating the set of closed patterns children($\emptyset$) that are extensions of the empty pattern $\emptyset$. For every pattern S of those, it only processes S if it is testable (therefore if the support $\sigma(S)$ of S is $\sigma(S) \geq \sigma_T$) using the processPattern(S) procedure. This procedure first computes the interval $C(S)$ (lines 7–9), and then computes the upper bound $\widehat{p}_S$ to the p-value (lines 10–12). If $C(S) = \emptyset$, $\widehat{p}_S$ is set to 0 (line 10); otherwise SPuManTE computes $\widehat{p}_S$ using the bound from Thm. 4.2. SPuManTE uses the upper bound $\widehat{p}_S$ to decide whether S is significant, returning S in output if $\widehat{p}_S < \delta$ (line 13). Then (lines 14–15), the current pattern S is "grown" generating the set of closed patterns that are extension of S using children(S), enumerating the space of the testable patterns exhaustively in a depth-first order.

We can show the following property of SPuManTE.

Theorem 4.8. *The output of SPuManTE has FWER at most α.*

Proof (Sketch). Consider the event F="the number of false discoveries reported by SPuManTE is > 0". The FWER of the output of SPuManTE is $\Pr(F)$. Recalling the event E_ε defined in Sect. 4.2.1, let $\bar{E}_\varepsilon$ be the complementary event. It holds:

$$\Pr(F) = \Pr(F \mid E_\varepsilon)\Pr(E_\varepsilon) + \Pr(F \mid \bar{E}_\varepsilon)\Pr(\bar{E}_\varepsilon) \leq \Pr(F \mid E_\varepsilon) + \Pr(\bar{E}_\varepsilon) \ .$$

Using AMIRA with parameter γ guarantees that $\Pr(\bar{E}_\varepsilon) \leq \gamma$. By employing the LAMP strategy with parameter $\alpha - \gamma$ and using the upper bound $\widehat{p}_S$ to decide if S is significant, it holds that $\Pr(F \mid E_\varepsilon) \leq \alpha - \gamma$. Therefore $\Pr(F) \leq \alpha - \gamma + \gamma = \alpha$. □

4.2.5 Increasing the power of UT. SPuManTE provides an efficient method to identify all significant patterns with bounded FWER. However, while extremely fast to compute, the upper bound of Thm. 4.2 does not always provide a tight approximation to the p-value p_S of a pattern S, resulting in a potential reduction in power, even if as we show in Sect. 5.1 the most significant patterns are still reported. In the scenarios where one is interested in reporting a larger number of patterns, at the expense of weakening the guarantees of the FWER, one can use the lower bound $\check{p}_S$ of Sect. 4.1.2 in place of the upper bound of Thm. 4.2 in lines 11–12. While in this case there is no guarantee on the FWER of the reported patterns, we show in Sect. 5.1 that $\check{p}_S$ is very close to the actual p-value p_S, leading to a relatively low risk of reporting false discoveries.

Algorithm 2: SPuManTE

```
   Input: Dataset $\mathcal{D}$, acceptable FWER $\alpha \in (0, 1)$, confidence
          parameter $\gamma \in (0, \alpha)$.
   Output: Set of significant patterns with FWER $\leq \alpha$.
1  $\varepsilon \leftarrow$ AMIRA $(\mathcal{D}, \gamma)$
2  $\delta \leftarrow$ correctedSignificanceThreshold$(\alpha - \gamma)$
3  $\sigma_T \leftarrow \min\{\sigma : \check{\psi}(\sigma) \leq \delta, 1 \leq \sigma \leq n\}$
4  forall $S \in$ children$(\emptyset)$ do
5      if $\sigma(S) \geq \sigma_T$ then processPattern$(S)$
6  Function processPattern(S)
7      $C_0(S) \leftarrow [f_0(S) - \varepsilon\frac{n}{n_0}, f_0(S) + \varepsilon\frac{n}{n_0}]$
8      $C_1(S) \leftarrow [f_1(S) - \varepsilon\frac{n}{n_1}, f_1(S) + \varepsilon\frac{n}{n_1}]$
9      $C_S \leftarrow C_0(S) \cap C_1(S)$
10     if $C_S = \emptyset$ then $\widehat{p}_S \leftarrow 0$
11     else
12         $\widehat{p}_S \leftarrow P\left(\sigma(S), \sigma_1(S) \mid f(S)\right)(n_0 + 1)(n_1 + 1)$
13     if $\widehat{p}_S \leq \delta$ then output $S$
14     forall $S' \in$ children$(S)$ do
15         if $\sigma(S') \geq \sigma_T$ then processPattern$(S')$
```

5 EXPERIMENTAL EVALUATION

We implemented SPuManTE and tested it on several datasets. Our experimental evaluation has the following goals:

- assess the tightness of the lower bound $\breve{p}_S$ from (3) w.r.t. the exact p-value p_S.
- evaluate the computational performance of UT: since no other method for performing efficiently an unconditional test for significant patterns exists, we compare UT with Fisher's exact test, the de-facto standard *conditional* test employed for significant pattern mining algorithms.
- assess the effectiveness and the impact of the upper bound $\widehat{p}_S$ and of the AMIRA confidence intervals on reporting significant patterns.

Implementation and environment. We implemented SPuManTE[2] and UT by modifying a C implementation of LAMP[3]. For computing the incomplete beta function in lines 9 and 12 of Alg. 1, we use a publicly available implementation[4] based on Lentz's algorithm [14]. All the code was compiled with GCC 8 and run on a machine with a 2.30 GHz Intel Xeon CPU, 512 GB of RAM, on Ubuntu 14.04.

Datasets. We tested SPuManTE on eight datasets commonly used for the benchmark of significant pattern mining algorithms, gathered from FIMI'04[5] and libSVM[6]. Due to space constraints we only report results for three datasets (the results for other datasets are analogous and will be shown in the full version). Descriptive statistics and preprocessing for these datasets are in the Appendix.

Parameters and experiments. In all our experiments, we set $\alpha = 0.05$ and $\gamma = 0.01$. In order to study the impact of the dataset size on SPuManTE's performance, for all datasets we generate a random sample of size s by taking s transactions uniformly at random with replacement, varying s in the interval $[10^3, 10^6]$.

We compare SPuManTE to three different variants: the first, that we denote SPuManTE*, is the one described in Sect. 4.2.5, that provides increased power at the expense of relaxed guarantees on FWER; the third version, SPuManTE$^{\mathrm{C}}$, flags an itemset S as significant only if its confidence interval $C(S)$ is $C(S) = \emptyset$; the last one, SPuManTE$^{\mathrm{n}}$, uses a naïve implementation of Alg. 1, that enumerates all the contingency tables for every pattern S, fixing π to $f(S)$. However, we do not include the results for SPuManTE$^{\mathrm{n}}$, since its naïve enumeration strategy results in impractical running times: for $s = 10^3$, the running time of SPuManTE$^{\mathrm{n}}$ is always at least one order of magnitude higher than all other approaches, and could not complete in one day for $s \geq 10^4$. SPuManTE$^{\mathrm{n}}$ would require even more time if an expensive search over the values of π is performed to compute p_S exactly.

5.1 Results

Quality of the lower bound. Our first experiment aims at evaluating the quality of the lower bound $\breve{p}_S$. For fixed n, n_1 and S (described in Fig. 2), we compute $\phi(\sigma(S), \sigma_1(S), \pi)$ varying π over 10^3 equally spaced values in a fixed interval. The value $\phi(\sigma(S), \sigma_1(S), f(S))$ coincides with our lower bound $\breve{p}_S$. Fig. 2 shows the resulting curve: even if $\breve{p}_S \neq p_S$ (therefore $\pi = f(S)$ does not maximize $\phi(\sigma(S), \sigma_1(S), \pi)$), using $\breve{p}_S$ provides a principled choice to obtain a very tight lower bound to p_S. Similar results, not shown for space constraints, hold for different choices of n, n_1, and S.

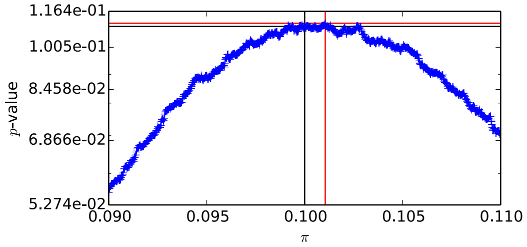

Figure 2: Values of $\phi(\sigma(S), \sigma_1(S), \pi)$ for 10^3 equally spaced values of $\pi \in I_{f(S)} = [0.9 \cdot f(S), 1.1 \cdot f(S)]$ with $n = 10^3$, $n_1 = 500$, $\sigma(S) = 100$, $\sigma_1(S) = 40$. The red vertical line corresponds to $\pi^* = \arg\max_{\pi \in I_{f(S)}} \{\phi(\sigma(S), \sigma_1(S), \pi)\}$, the black vertical line to $f(S)$, the red horizontal line to $\phi(\sigma(S), \sigma_1(S), \pi^*)$, the black horizontal line to $\phi(\sigma(S), \sigma_1(S), f(S))$.

Running time of SPuManTE. In Fig. 3.(a), we compare the running times of SPuManTE and SPuManTE* w.r.t. the state-of-the-art by LAMP with Fisher's exact test, denoted with LAMPF. Contrary to the common belief that unconditional tests are computationally expensive, SPuManTE is, in almost all cases, faster than LAMPF. These results stress the efficiency of the upper bound from Thm. 4.2. The only cases where SPuManTE is slower is for small sample sizes ($s \leq 10^4$), for running times ≤ 10 seconds, and for `retail` dataset. When s is small, the time to compute ε in SPuManTE dominates on the total execution time. For larger sample sizes, SPuManTE is faster than LAMPF by up to almost one order of magnitude. SPuManTE*, despite computing $\breve{p}_S$ for every S, generally requires comparable running time w.r.t. LAMPF, thanks to our efficient strategy for computing $\breve{p}_S$ (Sect. 4.1.2). These results show that SPuManTE provides an efficient strategy for significant pattern mining, even more efficient than the state-of-the-art even if it (correctly) employs an unconditional test.

Statistical power of SPuManTE. We evaluate the effectiveness of the upper bound from Thm. 4.2 in reporting significant patterns. Fig. 3.(b) displays the number of patterns in the output of SPuManTE, SPuManTE*, and LAMPF. The set of results reported by SPuManTE* is always a super-set of the set of results with guarantees on the FWER, since SPuManTE* uses a lower bound to the exact p-value. In all cases SPuManTE reports a large set of results, comparable in size with the output of SPuManTE*, therefore UT retains most of the statistical power that would be achieved with the expensive computing of the exact value of p_S. We can observe that, in almost all cases, LAMPF reports more (in some cases, twice as many) patterns than SPuManTE*. Since the p-value computed by Fisher's exact test can provide an *underestimate* of the true p-value of a pattern (due to the conditional assumptions not being met, see Fig. 1), the additional patterns reported by LAMPF may represent spurious associations.

Lastly, we investigate the impact of using the confidence intervals in SPuManTE. Fig. 3.(c) shows the number of output patterns by SPuManTE and the ones reported by SPuManTE$^{\mathrm{C}}$ (the variant of SPuManTE that only checks whether $C(S) = \emptyset$ to flag a pattern S

[2] The code of SPuManTE and the scripts to replicate all experiments are available at https://github.com/VandinLab/SPuManTE. See also the Appendix.
[3] https://github.com/fllinares/wylight
[4] https://github.com/codeplea/incbeta
[5] http://fimi.ua.ac.be
[6] https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/

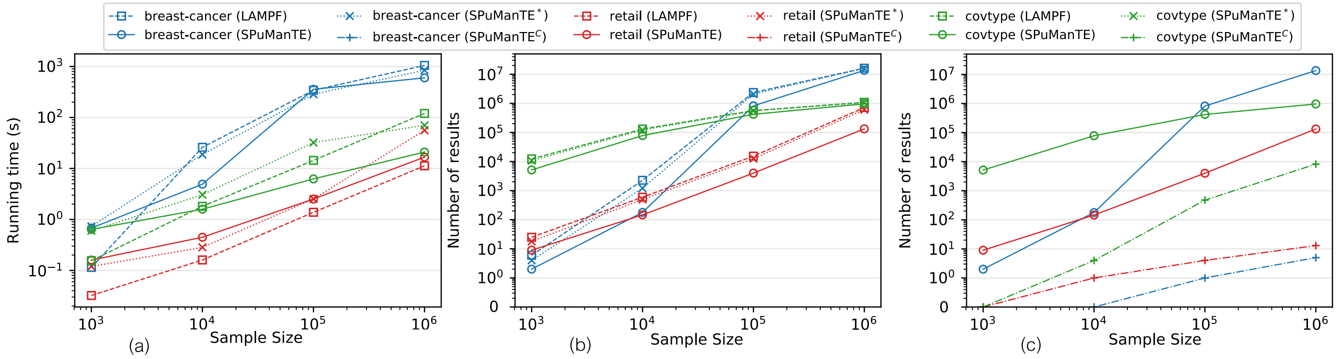

Figure 3: Comparison between LAMPF, SPuManTE and SPuManTE* in terms of: (a) running time; (b) number of patterns in output. In (c) we compare the number of patterns in output found with SPuManTE and SPuManTE^C.

as significant): the larger the sample, the higher is the number of patterns flagged as significant by SPuManTE^C, since ε decreases as the sample size grows, so the confidence intervals are more narrow. For the majority of the datasets we considered, a large number of patterns are marked as significant just by checking whether $C(S) = \emptyset$, proving that the use of confidence intervals is a crucial component of SPuManTE.

6 CONCLUSION

We introduce SPuManTE, an efficient algorithm to mine significant patterns using our novel unconditional test UT. UT requires fewer assumptions on the data generation process than commonly used conditional tests, such as Fisher's exact test. We prove that SPuManTE controls the FWER (i.e., the probability of reporting one or more false discoveries) at the desired level α set by the user. Our extensive experimental evaluation shows that SPuManTE efficiently identifies significant patterns while properly accounting for the stochastic variations in patterns frequencies due to the probabilistic nature of the data generation process. There are several future directions of research, including adapting our results to the mining of other significant patterns (e.g., subgraphs) and the use of UT in combination with different procedures for multiple hypothesis correction (e.g., Westfall-Young permutation testing).

ACKNOWLEDGMENTS

This work is supported, in part by the University of Padova grants SID2017 and STARS: Algorithms for Inferential Data Mining. Part of this work was conducted while L.P. was visiting the Department of Computer Science of Brown University, supported by a "Fondazione Ing. Aldo Gini" fellowship.

REFERENCES

[1] R. Agrawal, T. Imieliński, and A. Swami. 1993. Mining association rules between sets of items in large databases. *SIGMOD'93*.
[2] G. A. Barnard. 1945. A new test for 2×2 tables. *Nature* 156 (1945).
[3] Y. Benjamini and Y. Hochberg. 1995. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *J. Royal Stat. Soc.* (1995).
[4] R. Berger. 1994. Power comparison of exact unconditional tests for comparing two binomial proportions. *Institute of Statistics Mimeo Series* (1994).
[5] C. E. Bonferroni. 1936. Teoria statistica delle classi e calcolo delle probabilità. *Pubb. del Regio Istituto Superiore di Scienze Econ. e Comm. di Firenze* 8 (1936).
[6] R. D. Boschloo. 1970. Raised conditional level of significance for the 2× 2-table when testing the equality of two probabilities. *Statistica Neerlandica* 24 (1970).
[7] Leena Choi, Jeffrey D. Blume, and William D. Dupont. 2015. Elucidating the foundations of statistical inference with 2×2 tables. *PloS one* 10, 4 (2015), e0121263.
[8] Ronald A. Fisher. 1922. On the interpretation of χ^2 from contingency tables, and the calculation of P. *Journal of the Royal Statistical Society* 85, 1 (1922), 87–94.
[9] W. Hämäläinen. 2016. New upper bounds for tight and fast approximation of Fisher's exact test in dependency rule mining. *Comp. Stat. & Data Anal.* 93 (2016).
[10] W. Hämäläinen and G. I. Webb. 2018. A Tutorial on Statistically Sound Pattern Discovery. *Data Mining and Knowledge Discovery* (2018).
[11] Zengyou He, Simeng Zhang, and Jun Wu. 2018. Significance-based Discriminative Sequential Pattern Mining. *Expert Systems with Applications* (2018).
[12] Sture Holm. 1979. A simple sequentially rejective multiple test procedure. *Scandinavian journal of statistics* (1979).
[13] J. Komiyama, M. Ishihata, H. Arimura, T. Nishibayashi, and S. Minato. 2017. Statistical Emerging Pattern Mining with Multiple Testing Correction. *KDD'17*.
[14] W. J. Lentz. 1976. Generating Bessel functions in Mie scattering calculations using continued fractions. *Applied Optics* 15 (1976).
[15] F. Llinares-López, M. Sugiyama, L. Papaxanthos, and K. Borgwardt. 2015. Fast and memory-efficient significant pattern mining via permutation testing. *KDD'15*.
[16] C. R. Mehta and P. Senchaudhuri. 2003. Conditional versus unconditional exact tests for comparing two binomials. *Cytel Software Corporation* 675 (2003).
[17] S. Minato, T. Uno, K. Tsuda, A. Terada, and J. Sese. 2014. A fast method of statistical assessment for combinatorial hypotheses based on frequent itemset enumeration. In *ECML-PKDD'14*.
[18] Laetitia Papaxanthos, F. Llinares-López, D. Bodenham, and K. Borgwardt. 2016. Finding significant combinations of features in the presence of categorical covariates. *NIPS'16*.
[19] L. Pellegrina and F. Vandin. 2018. Efficient Mining of the Most Significant Patterns with Permutation Testing. *KDD'18*.
[20] M. Riondato and E. Upfal. 2015. Mining frequent itemsets through progressive sampling with Rademacher averages. *KDD'15*.
[21] M. Sugiyama, F. Llinares-López, N. Kasenburg, and K. M. Borgwardt. 2015. Significant subgraph mining with multiple testing correction. *SDM'15*.
[22] R. E. Tarone. 1990. A modified Bonferroni method for discrete data. *Biometrics* (1990).
[23] A. Terada, D. duVerle, and K. Tsuda. 2016. Significant Pattern Mining with Confounding Variables. *PAKDD'16*.
[24] A. Terada, H. Kim, and J. Sese. 2015. High-speed Westfall-Young permutation procedure for genome-wide association studies. *ACM-BCB'15*.
[25] A. Terada, M. Okada-Hatakeyama, K. Tsuda, and J. Sese. 2013. Statistical significance of combinatorial regulations. *Proc. of the Nat. Acad. of Scien.* 110 (2013).
[26] F. Vandin, A. Papoutsaki, B. J. Raphael, and E. Upfal. 2015. Accurate computation of survival statistics in genome-wide studies. *PLoS Comp. Bio.* 11 (2015).
[27] G. I. Webb. 2006. Discovering significant rules. In *KDD'06*.
[28] G. I. Webb. 2007. Discovering significant patterns. *Machine learning* 68 (2007).
[29] G. I. Webb. 2008. Layered critical values: a powerful direct-adjustment approach to discovering significant patterns. *Machine Learning* 71 (2008).
[30] P. H. Westfall and S. S. Young. 1993. Resampling-Based Multiple Testing: Examples and Methods for p-Value Adjustment. *Wiley Series in Prob. and Stat.* (1993).

A APPENDIX

In this appendix we present the proofs that we could not include in the main text and describe how to reproduce our experimental results.

Missing proofs

The proofs of our results are provided here.

Theorem A.1 (Thm. 4.2 in the main text). *It holds*

$$p_S \le P\left(\sigma(S), \sigma_1(S) \mid f(S)\right)(n_0+1)(n_1+1) \ .$$

Proof. It holds

$$\begin{aligned} p_S &= \max_{\pi \in C_S} \left\{ \sum_{(x,a) \in T(\sigma(S), \sigma_1(S), \pi)} P(x, a \mid \pi) \right\} \\ &\le \max_{\pi \in C_S} \left\{ P\left(\sigma(S), \sigma_1(S) \mid \pi\right) |T(\sigma(S), \sigma_1(S), \pi)| \right\} \\ &\le \max_{\pi \in C_S} \left\{ P\left(\sigma(S), \sigma_1(S) \mid \pi\right) \right\} \max_{\pi \in C_S} \left\{ |T(\sigma(S), \sigma_1(S), \pi)| \right\} \\ &= P\left(\sigma(S), \sigma_1(S) \mid f(S)\right)(n_0+1)(n_1+1). \end{aligned}$$

where in the last step we use Prop. 4.4 and the fact that the total number of contingency tables is $(n_0+1)(n_1+1)$, that is an upper bound to the size of $T(a, b, \pi)$ for any a, b, π. □

Proposition A.2 (Prop. 4.4 in the main text). *It holds*

$$\arg\max_{\pi} \{P(x, a \mid \pi)\} = x/n.$$

Proof. Define the function $g(\pi) = a'(\pi)^b (1-\pi)^{(c-b)}$ for some constants $a' > 0$, $b = x$, $c = n$. Then

$$\frac{\partial g(\pi)}{\partial \pi} = \frac{a'(1-\pi)^{(c-b)} \pi^{(b-1)} (c\pi - b)}{\pi - 1} \ .$$

The only root of $\frac{\partial g(\pi)}{\partial \pi}$ for $\pi \in (0,1)$ is given by $\pi = \frac{b}{c} = \frac{x}{n}$. It is trivial to check that the sign of the second order derivative is always < 0, and this fact completes the proof. □

Theorem A.3 (Thm. 4.5 in the main text). *Let $C_0(S) \cap C_1(S) = C_S \ne \emptyset$. Then $f(S) \in C_S$.*

Proof. We prove the result assuming $\sigma_0(S)/n_0 > \sigma_1(S)/n_1$ (the proof for the other case is analogous) and assuming that the confidence intervals have the form provided by amira. Recall that $f(S) = \sigma(S)/n$. $C_0(S) \cap C_1(S) \ne \emptyset$ is equivalent to

$$\frac{\sigma_1(S)}{n_1} + \varepsilon \frac{n}{n_1} \ge \frac{\sigma_0(S)}{n_0} - \varepsilon \frac{n}{n_0}. \tag{5}$$

and that proving $\sigma(S)/n \in C_0(S) \cap C_1(S)$ corresponds to prove that

$$\frac{\sigma_1(S)}{n_1} + \varepsilon \frac{n}{n_1} \ge \frac{\sigma(S)}{n} \tag{6}$$

and

$$\frac{\sigma_0(S)}{n_0} - \varepsilon \frac{n}{n_0} \le \frac{\sigma(S)}{n} \tag{7}$$

both hold. Equation (5) above is equivalent to

$$\frac{\sigma_0(S)}{n_0} - \frac{\sigma_1(S)}{n_1} \le \varepsilon n \left(\frac{1}{n_0} + \frac{1}{n_1} \right) \ . \tag{8}$$

It holds

$$\frac{\sigma(S)}{n} = \frac{\sigma_1(S)}{n_1} + \frac{n_0}{n} \left(\frac{\sigma_0(S)}{n_0} - \frac{\sigma_1(S)}{n_1} \right)$$

and from (8) we derive

$$\begin{aligned} \frac{\sigma(S)}{n} &= \frac{\sigma_1(S)}{n_1} + \frac{n_0}{n} \left(\frac{\sigma_0(S)}{n_0} - \frac{\sigma_1(S)}{n_1} \right) \le \frac{\sigma_1(S)}{n_1} + \frac{n_0}{n} \varepsilon n \left(\frac{1}{n_0} + \frac{1}{n_1} \right) \\ &= \frac{\sigma_1(S)}{n_1} + n_0 \varepsilon \left(\frac{1}{n_0} + \frac{1}{n_1} \right) = \frac{\sigma_1(S)}{n_1} + \varepsilon \left(1 + \frac{n_0}{n_1} \right) = \frac{\sigma_1(S)}{n_1} + \varepsilon \frac{n}{n_1}, \end{aligned}$$

that proves (6).

For (7), it holds

$$\frac{\sigma(S)}{n} = \frac{\sigma_0(S)}{n_0} - \frac{n_1}{n} \left(\frac{\sigma_0(S)}{n_0} - \frac{\sigma_1(S)}{n_1} \right)$$

and from (8) we derive

$$\begin{aligned} \frac{\sigma(S)}{n} &= \frac{\sigma_0(S)}{n_0} - \frac{n_1}{n} \left(\frac{\sigma_0(S)}{n_0} - \frac{\sigma_1(S)}{n_1} \right) \ge \frac{\sigma_0(S)}{n_0} - \frac{n_1}{n} \varepsilon n \left(\frac{1}{n_0} + \frac{1}{n_1} \right) \\ &= \frac{\sigma_0(S)}{n_0} - n_1 \varepsilon \left(\frac{1}{n_0} + \frac{1}{n_1} \right) = \frac{\sigma_0(S)}{n_0} - \varepsilon \left(1 + \frac{n_1}{n_0} \right) = \frac{\sigma_0(S)}{n_0} - \varepsilon \frac{n}{n_0}, \end{aligned}$$

that proves (7). □

Theorem A.4 (Thm. 4.6 in the main text).

$$\arg\min_{a} \{\phi(x, a, \pi)\} = \min\{x, n_1\}.$$

Proof. We assume $n_1 \le n_0$. Let $\breve{a} = \breve{\sigma}_{1,x}$ and $\widehat{a} = \widehat{\sigma}_{1,x}$. We first prove that

$$\min_{a} \{P(x, a, \pi)\} = P(x, \widehat{a}, \pi) \tag{9}$$

by observing that, $\forall a \in [\breve{a}, \widehat{a}]$,

$$\frac{P(x, a, \pi)}{P(x, \widehat{a}, \pi)} = \frac{B(x-a, n-n_1, \pi) B(a, n_1, \pi)}{B(x-\widehat{a}, n-n_1, \pi) B(\widehat{a}, n_1, \pi)} = \frac{\binom{n-n_1}{x-a}\binom{n_1}{a}}{\binom{n-n_1}{x-\widehat{a}}\binom{n_1}{\widehat{a}}} \ge 1.$$

A direct consequence of (9) and the definition of $T(x, a, \pi)$ is

$$T(x, \widehat{a}, \pi) \subseteq T(x, a, \pi), \forall a \in [\breve{a}, \widehat{a}]. \tag{10}$$

Then (10) leads to

$$\frac{\phi(x, a, \pi)}{\phi(x, \widehat{a}, \pi)} = \frac{\sum_{(y,b) \in T(x,a,\pi)} P(y, b \mid \pi)}{\sum_{(y,b) \in T(x,\widehat{a},\pi)} P(y, b \mid \pi)} \ge 1, \forall a \in [\breve{a}, \widehat{a}],$$

that proves the statement. □

Proposition A.5 (Prop. 4.7 in the main text). *Let $y \in [0, n]$, $b \in [\breve{\sigma}_{1,y}, \widehat{\sigma}_{1,y}]$, and $\pi \in (0,1)$. Let*

$$A_1 = \left\{ a_1 : P\left(a_1 + \lfloor (n_0+1)\pi \rceil, a_1 \mid \pi\right) > P(y, b \mid \pi) \right\},$$

and define the set $A_{0,a_1} = \{a_0 : P\left(a_1 + a_0, a_1 \mid \pi\right) \le P(y, b \mid \pi)\}$. Then

$$\begin{aligned} &\sum_{(x,a) \in T(y,b,\pi)} P(x, a \mid \pi) \\ &= \sum_{a_1 \notin A_1} B(a_1, n_1, \pi) + \sum_{a_1 \in A_1} \left(B(a_1, n_1, \pi) \sum_{a_0 \in A_{0,a_1}} B(a_0, n_0, \pi) \right), \end{aligned}$$

where $B(z, h, \pi) = \binom{h}{z} \pi^z (1-\pi)^{h-z}$ is the probability of obtaining z successes on h independent trials with success probability π.

dataset	$\|D\|$	$\|\mathcal{I}\|$	avg	n_1/n
breast cancer	12,773	1,129	6.7	0.09
retail(U)	88,162	16,470	10.3	0.47
covtype	581,012	64	11.9	0.49

Table 2: Datasets statistics. For each dataset we report: name (see Sect. 5 for the meaning of *U*), number $|\mathcal{D}|$ of transactions; the number $|\mathcal{I}|$ of items; average transaction length avg; **fraction n_1/n of transactions in $\mathcal{Z}^1$.**

Proof. We formulate $\sum_{(x,a)\in T(y,b,\pi)} P(x,a \mid \pi)$ as

$$\begin{aligned}
&\sum_{(x,a)\in T(y,b,\pi)} P(x,a \mid \pi)\\
&= \sum_{(a_0+a_1,a_1)\in T(y,b,\pi)} \binom{n_0}{a_0}\binom{n_1}{a_1}\pi^{a_0+a_1}(1-\pi)^{(n_0+n_1-a_0+a_1)}\\
&= \sum_{(a_0+a_1,a_1)\in T(y,b,\pi)} B(a_0,n_0,\pi)B(a_1,n_1,\pi)
\end{aligned}$$

Let $\lfloor a \rceil$ denote the closest integer to a. It holds

$$P\left(a_1 + \lfloor (n_0+1)\pi \rceil, a_1 \mid \pi\right) \geq P\left(a_1+a_0, a_1 \mid \pi\right), \forall a_0 \in [0,n_0] \ .$$

Given the above and the definition of A_1 and A_{0,a_1}, we obtain

$$\begin{aligned}
&\sum_{(x,a)\in T(y,b,\pi)} P(x,a \mid \pi)\\
&= \sum_{(a_0+a_1,a_1)\in T(y,b,\pi)} B(a_0,n_0,\pi)B(a_1,n_1,\pi)\\
&= \sum_{a_1\notin A_1} B(a_1,n_1,\pi) + \sum_{a_1\in A_1}\left(B(a_1,n_1,\pi)\sum_{a_0\in A_{0,a_1}} B(a_0,n_0,\pi)\right)
\end{aligned}$$

□

Proposition A.6. *The time complexity of Alg. 1 is*

$$O\left(\log(n_0) + |A_1| + n_0 - |A_{0,a_1}| + La\right) \ ,$$

where $O(La)$ is the time complexity of Lentz's algorithm.

Proof. The $\log(n_0)$ term follows from a binary search over n_0 sorted elements, that is performed on the first iteration of the **forall** loop. Every iteration of the **forall** loop requires $O(1)$ work, resulting in the $O(|A_1|)$ term. The sum of all the work done during the **forall** loop does not exceed $O(n_0 - |A_{0,a_1}|)$, that is the time to increase a' and to decrease a'' in its every iteration and to update p' and v. The $O(La)$ term follows since the total number of calls to Lentz's algorithm is constant. □

Experimental reproducibility

We now describe how to reproduce our experimental results. Code and data are available at https://github.com/VandinLab/SPuManTE.

Datasets and preprocessing. The statistics of the datasets we analysed are described in Table 2. For datasets whose transactions are not naturally divided in two groups (marked with U), we selected the single item whose frequency is closer from below to 0.5, removed the corresponding item from every transaction, and use its appearance to divide the dataset in two groups. The reported ratio n_1/n refers to the output of this process. For real-valued features we obtained two bins by thresholding at the mean value and using one item for each bin.

Reproducing our simulations. The plot of Fig. 1 can be created with the Python script `fisher_simulations.py` in the `scripts/` folder. The results of Fig. 2 can be obtained using the `find_max_pi.py` script. All the parameters of the experiments can be modified with appropriate input parameters, or by directly modifying the scripts.

Reproducing our experiments. The code of SPuManTE and all variants of ut are in the sub-folder `unconditional/`, while the code for lampf is in the sub-folder `fisher/`. Inside each folder, the `correct/` directory contains the code for computing the corrected significance threshold δ, while the `enumerate/` directory contains the code to actually compute the significant patterns.

To compile all the software, use the `make` command inside all `correct/` and `enumerate/` sub-folders. Then, also compile amira by running `make` inside the `amira/` folder. A recent version of GCC (e.g., GCC 8.0) is needed to compile amira.

Once everything has been compiled, convenient scripts can be used to run the experiments. In particular, `run_amira.py`, `run_unconditional.py` and `run_fisher.py` automatically execute amira, SPuManTE, and lampf, respectively. These scripts accept a variety of input parameters. In particular, you need to specify a particular dataset and the size of a random sample to create using the flags `-db` and `-sz`. As an example, the command line to process with SPuManTE a random sample of 10^3 transactions from the dataset `mushroom` is

```
run_unconditional.py -db mushroom -sz 1000
```

and it automatically executes amira and SPuManTE.

The `run_all_datasets.py` script runs all the instances of SPuManTE ad lampf in parallel, and can be used to reproduce all the experiments described in Sect. 5.