Big Data Essay: A Nice Theory About How To Sample Correctly

Daniele Di Grandi, student number: 7035616

April 2021

1 Introduction

"Data is the new oil", Clive Humby, 2006.
This quote explains concisely that data, like oil, is valuable but useless if unrefined. Nowadays, every computer process leaves a trail, which is recorded and collected as a single piece of data. If one thinks of this process as a global event, the phenomenon of Big Data will eventually occur: a massive amount of multiple types of data is streamed in very quickly, and not all the data streams are reliable as the others, meaning that if we don't have a way to refine these streams, we have only collected invaluable data.

However, even though having a lot of data might seem like a positive thing - and in a sense, it is - lots of problems also arise. For example, lots of data means more computational power to analyze them, more space to fit them in memory, spurious correlations may appear etc.

The aim of this essay would then be to show that the solution of taking a sample from a Dataset D that is representative of the population of the Dataset itself - of which we assume that data have been sampled from some distribution $\mathcal{D}$ - is a really good and clever methodology to actually handle those Big Data and use them to extract knowledge and learn that underlying distribution. To do that, two different methodologies of taking statistically representative samples (Toivonen 1996 [1], Riondato and Upfal 2014 [2]) will be analyzed.

Problem statement. The previous paragraph contains 4 hidden key problems that characterize Big Data (known as the 4Vs): Volume, Velocity, Variety and Veracity. These problems bring challenges of storing data, computational efficiency and significance. Although having a massive amount of data could make the predictions pretty accurate, there is also a drawback: by taking a sample of a large size, the small differences are emphasized resulting in a distorted statistical significance, that is, the tiny differences will be statistically significant, which doesn't give us any useful practical information.

Hence, the problem could be summarized in a very simple sentence: how can we be statistically sure that we have taken a representative sample of the *right* size which minimizes the probability of making errors?

2 Frequent item set mining

Frequent Itemset Mining is a method for market basket analysis and is aimed at finding regularities in the shopping behaviour of customers of supermarkets and shops in general. Hence, the objective is to find patterns in sets of products - Itemsets - that are frequently bought together. An Itemset is considered to be frequent when it has been bought more than θ times, where θ is defined as a threshold.

Problem definition. The problem is defined by Toivonen in his paper as follows. Let $\mathcal{I}$ be the set of all items in the store, t be a transaction (set of items that have been bought together), D a Database of transactions and I a possible set of items (that is a subset of $\mathcal{I}$ and could be a subset of t). Thus, the objective is to find all the set of items I that are frequent, hence, the number of times that they appear in total over the transactions t has to be greater or equal than the threshold θ. The number of times that an Itemset I appears over all the transactions in the Database is defined as the support of that Itemset I. Hence, if the support of I is greater or equal than the threshold θ, the Itemset I is considered to be frequent.

2.1 Apriori algorithm

The idea that inspired the so-called "Apriori" algorithm is the following: An Itemset I_1 that contains the items A, B and C, has, for sure, been bought less or equal number of times than an Itemset I_2 that contains only A and B. The statement is pretty obvious since the Itemset I_2 is a subset of the Itemset I_1, thus, if the Itemset I_1 has been bought k times, it is sure that the Itemset I_2 has been bought *at least* k times. However, since the Itemset I_2 is a combination of fewer items (it does not contain the item C), it is likely that it has been bought more than k times, considering that the probability of buying C does not play a role. This means that if the Itemset I_2 is not considered to be frequent, the Itemset I_1 wouldn't even be created in the first place, increasing efficiency.

Hence, this is an actual level-wise search algorithm, that will generate and check the Itemsets I_i only if in the previous level all the subsets of the items that compose I_i have a support that is greater or equal than the threshold θ. This algorithm could be applied to solve the Frequent Itemset Mining problem.

Since real Databases are sparse, the actual complexity of this algorithm is $O(|\mathcal{I}|^m)$.

2.2 Using a sample

As previously mentioned, sampling is a good idea to efficiently solve the Frequent Itemset Mining problem for several reasons.

However, sampling also has two major drawbacks which are two errors that are made: first, we could find frequent sets that are not frequent in the entire Database (first type error), and second, we could miss frequent sets that are frequent in the entire Database (second type error).

The probability of making these two errors directly depends on the sample size: intuitively, the bigger is the sample size and the lower is the probability of making these errors, since if the sample size would be equal to the Dataset size, the probability of making these errors is zero. While talking of error probabilities, it would be nice to find a bound such that we can say that the probability that we are making an error is at most the bound found.

The bound we will apply is called the *"Hoeffding's inequality"*, which is a bound that gives the quality of an estimation by saying that the probability that the average of a sequence of random *iid* variables differs of more than a value ϵ than the real expected value, is less or equal than $2e^{-\frac{2m\epsilon^2}{(b-a)^2}}$, where m is the sample size, a and b are respectively a lower and upper bound on the random variables.

In Section 4.1, this bound will be applied to find a proper sample size as Toivonen shown in his paper.

Among the two errors, the first type is not serious: with only one complete scan of the Database, we can check if each frequent Itemset discovered in the sample is actually frequent also in the Database, by simply counting how many times that Itemset is present in all the transactions, and if that number is less than the threshold, that Itemset were not frequent in the Database and we can consider it not frequent anymore. Unfortunately, for the second type of error, there is not such a simple fix, but only a way to mitigate it, that is, by lowering the threshold.

By using Hoeffding's inequality again, it is possible to calculate that the quantity we should lower the original threshold is $\sqrt{\frac{1}{2m}\log\frac{1}{\mu}}$, and if we do that, we know that the probability of making an error of the second type given a sample size of m is at most μ. Of course, lowering the threshold would lead to making more errors of the first type. However, by applying the following procedure, this should not be a problem: - 1: Draw a sample of sufficient size (as explained in Section 4.1) - 2: Compute the frequent Itemsets on the sample using the lowered threshold by running the Apriori algorithm - 3: Check the real support of these frequent Itemsets in the Database and remove the ones that have that support lower than the original threshold.
In this way, all the Itemsets found to be frequent in the sample will be removed if not frequent also in the Database, which is scanned only twice: the first time, when the sample is collected, and the second, during the step 3 of the procedure.

However, with probability μ we are still missing some frequent Itemsets. To check if this is true, the idea is to individuate, from the set of all frequent Itemsets, the complementary set formed by the Itemsets with minimal cardinality, which is called the *border* of the set of frequent Itemsets. For example, if the set of frequent Itemesets is:
$S = \{\{A\},\{B\},\{C\},\{A,B\},\{B,C\}\}$
Then, the border of this set is:
$Bd(S) = \{\{A,C\}\}$
Which is the complementary minimal set of S.

It is obvious to say that if in the Database there are frequent Itemsets that are not in S, logically, at least one set in the border of S is frequent. Now, in step 3 of the previous procedure, together with checking the Itemsets in S, in the same Database scan we can also check the Itemsets in the border of S.

From frequent item set to classification. Although this solution seems pretty good so far, there still are problems, such as that in the worst case scenario, this algorithm will compute all the combinations of Itemsets, thus performing really bad.

Furthermore, to have a tighter bound on the sample size and a direct control on the probability to miss frequent Itemsets (since lowering the threshold is an indirect control), the following observation is crucial: until now, we were using an Itemset associated with its support as an indicator function (or labelling function), which is 1 if that Itemset is a subset of a transaction and 0 otherwise. By doing such a mapping, we know that indicators are classifiers, that are functions that assign 1 or 0 (or every arbitrary dichotomy) to each element in the Dataset. Hence, we can conclude that Itemsets are also classifiers. This observation is crucial since now we are able to exploit all the theory behind the classification problems in order to solve our problem and find a probably approximately correct (PAC) result.

3 PAC-learning

Using the perspective of classification is the base of the so-called PAC learning method and Riondato and Upfal exploit its potential in their paper by trying to compute a tighter bound on the sample size for the frequent Itemset mining problem. PAC learning is a framework for analysis in which the learner receives samples and must select a generalization function (called the hypothesis or the classifier) among a class of possible labelling functions, called $\mathcal{H}$ (class of hypotheses). However, Riondato and Upfal used a different notation in their paper for this set, they talked about a *range space* (X, R) with X that denotes a set of points (finite or infinite) and R is a (finite or infinite) family of subsets of X which are called ranges (instead of classifiers or hypothesis, as before). For simplicity, we will use $\mathcal{H}$, but in section 4.2 we will come back to talking about the range space.

The objective of PAC learning is to select the hypothesis that with the highest probability will get a minimum prediction error on unseen data.

3.1 Classification, Quality and convergence

PAC learning is a methodology composed of other previous results: the best way to introduce PAC learning is by tackling step-by-step those previous results.

Loss of a function. The true loss of a function is defined as the probability that if we make a sample according to the distribution, we make an error, that is, the label assigned from that function is different from the true label. However, the problem is that the true distribution is unknown because its estimation is the main problem we are trying to solve. Hence, the loss function could be calculated using the learned-from-data distribution (loss on the Dataset), and in this case, it takes the name of empirical risk. The procedure of finding a function that minimizes this loss it's called Empirical Risk Minimization (ERM).

Realizability assumption. If we suppose that the true hypothesis is in $\mathcal{H}$ (which in this case is a finite set of hypotheses), meaning that the true loss function calculated using that hypothesis would have a value of 0, it means that using the same hypothesis for the computation of the loss function on the Dataset would yield to a loss value of 0. Hence, by using the true hypothesis on the Dataset, the loss function will be 0 and since the ERM learning will find the hypothesis which gives the minimum loss, the true hypothesis will be found among all the others in $\mathcal{H}$.

The learning algorithm. In the environment defined until now, there exists a simple learner that could be used in the ERM procedure: the Halving learner.

This algorithm is really simple: for each datapoint, predict its label using all the hypotheses in $\mathcal{H}$ and eliminate all the hypotheses that gave a wrong label. In the next iteration - for the next datapoint - do the same thing but only using the hypotheses that survived the iteration before. The algorithm stops when a single hypothesis remains, which is the optimal one (that gives loss equal to 0) for the realizability assumption, meaning that the algorithm is optimal.

The complexity of this algorithm is $O(|D|)$. The complexity is not bad - since it's linear - but neither optimal since sublinear would have been better. Even here, sampling could reduce the time needed to explore all the Dataset: we only need a guarantee that the probability of making a huge error by taking a bad sample is really small.

Define what a bad sample mean. A bad sample is a sample that gives a high true loss. If we now call δ the probability of having a bad sample and ϵ the accuracy, we can say that with a confidence of $(1-\delta)$ a sample is not bad if the true loss is less or equal than ϵ and is bad otherwise. It is also interesting to note that this probability and the sample size are tightly connected: intuitively, the bigger the size and the lower is the probability of having taken a bad sample. By reasoning on that, it is possible to find a bound on the probability to actually have a bad sample, simply by constructing the set that contains all the bad hypotheses, that is, all the hypotheses that give a loss greater than ϵ. Now, we can also define a bad samples' set, which is the set of samples for which exists a hypothesis from the set of bad hypotheses that would result in a loss function on the Dataset equal to 0. If for a sample we will find a hypothesis such that the loss function on the Dataset is 0, this means that the true loss function will be greater than ϵ, pointing out that our sample is bad, and this is true for all the samples in the bad samples' set.

It is obvious now that finding a bound on the probability of having a sample that belongs to the bad samples' set gives a bound on learning a bad hypothesis, which we were looking for. In the end, we will find that by choosing a sample size of at least $\lceil \frac{\log \frac{|\mathcal{H}|}{\delta}}{\epsilon} \rceil$, ensures that for any labelling function and for any distribution (where the realizability assumption holds) with a probability of at least $(1-\delta)$ we will have for any ERM hypotheses that the true loss would be less or equal than ϵ. The nice thing about this bound is that the term $|\mathcal{H}|$ is inside a logarithm, meaning that the sample size is not growing linearly with respect to $|\mathcal{H}|$, but is growing in a logarithmic way.

The procedure. The procedure of PAC learning is nothing less than the one just presented: we are able to define an optimal logarithmic sample size in order to probably approximately correct learning a classifier by finding the probably optimal hypothesis h (not optimal for sure because of the sampling procedure) with the Halving learner algorithm, under the realizability assumption and with two metrics that are ϵ and δ. Moreover, this procedure ensures that the true loss using h would be at most ϵ: $L_{\mathcal{D}}(h) \leq \epsilon$.

3.2 Realizable vs agnostic PAC-learning

If we lose the assumption that PAC learning is not meant only for binary classification - by saying that not all the errors weigh the same way, thus, the true loss will now be defined as how bad we expect to perform on a new unseen example - we have two assumptions that make PAC learning unrealistic left, and we are going to lose both of them.

First assumption. *Realizability assumption.* This assumption is not realistic. The correct assumption would be that for all hypothesis in $\mathcal{H}$, the true loss would be always greater than 0. In that case, we are interested in finding the entity of such an error we are making, by giving a bound between the difference of the true loss and the loss on the data. By applying Hoeffding's inequality, the new sample size should be at least: $\lceil\frac{\log\frac{2|\mathcal{H}|}{\delta}}{2\epsilon^2}\rceil$.

By comparing this result with the case where we still had the realizability assumption, the biggest difference is that ϵ is now squared, causing the sample size to be much bigger, and the algorithm to be much slower: to be realistic, there is always a price to pay.

Second assumption. *Labelling is governed by a labelling function.* This assumption would not allow having noisy data, hence no errors, which is unrealistic in a real Dataset. To lose this assumption, we will move to the so-called *agnostic* PAC learning: the setting is the same as before, but now we don't require that PAC learning holds "for any labelling function" since we don't have labelling functions anymore, and since we already lost the realizability assumption, we no longer require that the true hypothesis belongs to the set $\mathcal{H}$. In this setting, the labelling function would instead be a probabilistic function, that assigns the label 1 if the probability of the outcome to be 1 is greater or equal than 50%, since in this way the error is minimized. The algorithm would return a hypothesis that sits in $\mathcal{H}$ that is probably close to the best possible result.

We have obtained the most general definition of PAC learning, thus, we are able to use this framework to understand when we could expect reasonable results from a learning algorithm.

3.3 VC-dimension and the fundamental theorem

The question that immediately follows is: which classes of hypotheses are PAC learnable? To give an answer, other definitions have to be introduced.

Uniform Convergence property. We can define the Uniform Convergence Property (UC) using the same assumptions that hold for the agnostically PAC learnable case. The conclusion though is different: by taking a sample according to $\mathcal{D}$ of a size that is "big enough", then that sample is ϵ-representative (that is, for each hypothesis in the hypotheses class, the loss on the Dataset is no more than ϵ respect to the true loss) with a probability of $(1-\delta)$.

Now, we can extract a tool to understand whether a set of hypotheses class is PAC learnable: if the considered hypotheses class has the UC property, then it's sufficient and we can conclude that that class is PAC learnable. Furthermore, in this scenario, we can also conclude that the ERM rule is a successful learner.

In order to find the "big enough" sample size, we want to prove that over all samples of the searched size, the probability that for all hypotheses the difference between the estimated loss and the true loss is less than or equal to ϵ, is bigger or equal to $(1-\delta)$. By using the complement rule, finding this result is basically equal to looking at the probability that there exists a hypothesis that makes the difference between the estimated loss and the true loss bigger than ϵ, would be smaller than δ. By doing the same computation as always (union bound and Hoeffding bound) and by saying that the hypotheses class is agnostically PAC learnable using the ERM rule it is possible to obtain a bound on the sample size.

Hence, the conclusion is that the chosen *finite* hypotheses class has the UC Property (thus, is agnostically PAC learnable) if and only if we pick a sample with a complexity that is computed using the bound we found. From this, it follows that finite $\mathcal{H}$ are always PAC learnable (if the sample size is correct). Furthermore, another nice conclusion follows, that is that in order to PAC learn, it doesn't matter at all which loss function is used nor if the true classifier is among the chosen hypotheses class.

Shattering and VC-Dimension. Shattering is simply the notion that expresses whether or not, in a specific subset of elements, there exist all the possible combinations of classification. That is, if C is a subset of a domain X, we can count in how many ways we can classify the data points in C using $\mathcal{H}$, and we can say that C is shattered by $\mathcal{H}$ if there exist all the possible combinations of classification. For example, if C is composed of only 1 data point, and if that data point could be classified using $\mathcal{H}$ as 0 or 1 in some way, then C is shattered by $\mathcal{H}$ because the two combinations of classification are both possible for C.

Since in Section 3.4, we will understand that $\mathcal{H}$ is PAC learnable only if it's not too expressive, we have to find a mathematical way to define the notion of expressiveness.

The definition of VC-Dimension could be used exactly for that purpose: we can define the VC-Dimension of a set ($\mathcal{H}$, in our case) as the size of the largest set C such that C is shattered by $\mathcal{H}$. If $\mathcal{H}$ can shatter arbitrary sizes of C, then the VC-Dimension of

$\mathcal{H}$ is considered to be infinite. The consequence of this definition follows immediately: if the VC-Dimension of $\mathcal{H}$ is infinite, it means that $\mathcal{H}$ is a very rich set, thus, it is not PAC learnable, because of the "No Free Lunch" theorem, as explained in Section 3.4.

Unfortunately, there is no mechanical procedure in order to compute the VC-Dimension of a certain $\mathcal{H}$: the procedure is related to the structure of the problem itself.

Fundamental Theorem of PAC learning. The question that was left open is whether a certain infinite class of hypotheses is PAC learnable if its VC-Dimension is not infinite. That is, we want to understand if the scenario that sits in between the finite $\mathcal{H}$ (which is PAC learnable) and the infinite $\mathcal{H}$ with an infinite VC-Dimension (which is not PAC learnable) is PAC learnable. The Fundamental Theorem aims to prove just the following statement: if $\mathcal{H}$ has a finite VC-Dimension, then it's learnable.

The proof will consist of trying to show that classes with a finite VC-Dimension have the UC property, which is a sufficient condition to conclude that a class is PAC learnable. To prove the UC, we have to give a bound on the expectation of the difference between the supremum for each hypothesis of the measured loss and the true loss. By applying Jansen's inequality - that is, the value of a function in the average point of the x_i is smaller or equal than the average of the function already evaluated in that points - on this, we obtain the following:

$$\begin{aligned} \mathbb{E}_{D\sim\mathcal{D}^m}(sup_{h\in\mathcal{H}}|\mathcal{L}_D(h) - \mathcal{L}_\mathcal{D}(h)|) \leq \\ \mathbb{E}_{D,D'\sim\mathcal{D}^m}(sup_{h\in\mathcal{H}}|\mathcal{L}_D(h) - \mathcal{L}_{D'}(h)|) \end{aligned} \tag{1}$$

It is interesting to note that the expression $sup_{h\in\mathcal{H}}|\mathcal{L}_D(h) - \mathcal{L}_{D'}(h)|$ is just the cross-validation operation where D and D' are 2 iid samples taken from $\mathcal{D}$, which can be seen as the trained and test sets. By expanding more the losses considering also the single elements $z_i \in D$ and $z_i' \in D'$, it can be noticed that by switching z_i and z_i', the result would remain the same because the difference between the losses stays the same. Hence, the sign of the losses doesn't matter, thus, is possible to arbitrarily multiply them by ± 1, which are coded in a vector called σ, that is expected to follow a uniform distribution, hence, both the probabilities of +1 and -1 are 50%.

Since the expectation is a linear operator, it is possible to switch the expectation written before on the right-hand side, with the expectation on σ. Thus, now $\mathbb{E}_{\sigma\sim\ U_\pm^m}$ is the inner loop. In this inner loop, D and D' are constant: this means that we are in a finite context! We already managed to show that finite classes are PAC learnable, and by applying again Hoeffding's inequality, we are able to find a bound on the probability. Since we wanted a bound on the expectation and not on the probability, we can exploit a lemma that states that $\mathbb{E}(|X - x|) \leq a(4 + \sqrt{\log(b)})$, obtaining the searched bound on the expectation.

However, to prove the UC, we have to show that exists a sample size m that depends on ϵ and δ such that the right-hand side bound is less or equal than ϵ. By equal the obtained bound to ϵ, we obtain:

$$m \geq 4\frac{2d}{(\delta\epsilon)^2}\log\frac{2d}{(\delta\epsilon)^2} + \frac{4d\log\frac{2e}{d}}{(\delta\epsilon)^2} \tag{2}$$

With d that is the VC-Dimension of $\mathcal{H}$.

In conclusion, if the size is at least this m, we proved that $\mathcal{H}$ has the UC property, hence, it has a finite VC-Dimension, hence, it is PAC learnable.

3.4 Bias and no free lunch

When using PAC learning, there is no guarantee that a classifier is good, there is only the guarantee that given the set of classifiers you choose from, you choose one that is close to the optimal from *only* that set. This also means that we have a probability of making an error, which we can see as:

$$L_\mathcal{D}(h_d) = (min_{h\in\mathcal{H}}L_\mathcal{D}(h)) + \epsilon_{est} \tag{3}$$

Where the term $min_{h\in\mathcal{H}}L_\mathcal{D}(h)$ is called the *bias* term (how well $\mathcal{H}$ fit the distribution) and ϵ_{est} is called the *variance* term (how well the sample let us estimate the best classifier h). It can be seen that the bias term, unlike the variance term, doesn't depend on the sample. Hence, the question is: should we try to minimize the bias term? The answer is that there are cases in which it is possible and cases in which it is not possible. The method to perform such a minimization is simply to apply some background knowledge. However, most of the time we don't have this knowledge when solving a problem, and such a minimization would not be possible. Moreover, by the "No Free Lunch" theorem, it is not possible to manually constrain the bias term to 0 by making $\mathcal{H}$ rich enough, since there will be always cases where the algorithm will miserably fail. In fact, what the theorem says, is that if $\mathcal{H}$ is a very rich set, then a small sample is simply too small to let you pick out a good classifier, resulting in that with a probability of at least 1/7 on the choice of a sample, we have that the true loss using the algorithm on that sample will always be at least 1/8. Now, it simply follows that very rich infinite classes of hypotheses cannot be PAC learned. But is there something in between, since finite classes can be PAC learned? The trick is that when trying to PAC learn an infinite $\mathcal{H}$, it should not be too expressive.

4 Frequent item set mining on big data

This Section aims to give a conclusion on the results of both the paper considered and compare them.

4.1 Toivonen and frequent item sets

In Section 2.2 we were searching a bound on the probability of making an error during the sample size choice and we introduced Hoefdding's inequality. Hence, by applying Hoeffding's inequality on the probability that the difference between the true support of an Itemset in the Database and the support of the same Itemset but in the sample size is greater than ϵ, also using sampling with replacement, and by calling δ the bound of the probability of making an error, we will be ϵ in error with a confidence of $(1-\delta)$ by choosing a sample size m of at least:

$$m \geq \frac{\log \frac{2}{\delta}}{2\epsilon^2} \tag{4}$$

Although m seems to be not big for almost any values of ϵ and δ, the formula has a problem: this result is valid only for one single Itemset. What we really want to find is a bound on the probability of making an error considering all the Itemsets I, that finally is, using the union bound:

$$m \geq \frac{1}{\epsilon^2} \lceil (|\mathcal{I}| \log 2 + \log 2 + \log \frac{1}{\delta}) \rceil \tag{5}$$

Which can be pretty big since $|\mathcal{I}|$ could be a large number. In fact, a problem of this bound is that the sample size grows linearly with $|\mathcal{I}|$, hence, it's not tight as we wanted it to be.

4.2 PAC-learning and frequent item sets

Coming back to talking about range spaces (X, R), the same as is possible to restrict a function X to a subset of X, is also possible to restrict a subset of X to another subset of X, thus, for any subset $A \subset X$ the projection of the range space on that subset A is simply the intersection of the elements $r \in R$ with A. The idea is basically the shattering idea.

With the same methodology applied in the fundamental theorem proof, it is possible to derive an initial bound. To use that bound, we have to better define the VC-Dimension of the range space used. Given a Dataset D with associated a range space (X, R), then its VC-Dimension is at least v if there exists a subset of X with v elements such that for every subset $B \subset A$ there is an Itemset I_B such that all transactions in A that support this Itemset I_B, is exactly B. Thus, we want that for every subset of A we can find an Itemset such that that Itemset exactly occurs in all transactions in B, and there are no other transactions in A that also contain B, that is: B is closed in A. An immediate consequence of this is that the VC-Dimension is greater or equal than v and, by definition, the VC-Dimension is exactly the largest possible v such that there is a subset of v elements such that the previous condition holds. This gives us a way to compute the VC-Dimension for this problem.

However, the requirement that the statement holds for every $B \subset A$ makes the computation very costly, but fortunately, the consequence we have derived gives us an alternative: we need at least v transactions and these v transactions also have to be v long, since there is the need to be able to distinguish all the subsets. Thus, if the VC-Dimension is v, then we will find at least a set of v transactions that are v long. For that purpose, we introduce the d-bound, which is the largest integer such that D contains at least d different transactions of length at least d. This d-bound is a tight upper bound on the VC-Dimension and is possible to compute it with only one scan over the Dataset, simply maintaining the l longest different transactions that are at least l long.

Finally, this method gives us the searched bound for the sample size m:

$$m \geq \min\{|D|, \frac{4c}{\epsilon^2}(d + \log \frac{1}{\delta})\} \tag{6}$$

Where d is the d-bound of D and c is a constant (experimentally: ≤ 0.5).

Hence, with only 2 scans we can compute the frequent Itemsets of a Database with a probability of $(1-\delta)$ that we find them all. To be sure, we can always compute the border, but we can also compute another sample and see which Itemsets are considered to be frequent on the second sample, and compare them with the frequent Itemsets of the first sample.

4.3 Toivonen vs PAC-learning

With the same inputs $(d, |\mathcal{I}|, \delta, \epsilon)$, the result that returns a smaller sample size will be better. Hence, if the bound of Toivonen would result in a sample size greater than the bound of Riondato and Upfal, this last one is better. By applying this inequality on the general case (Equation 5 > Equation 6) considering $c = 0.5$ - and not considering the worst-case scenario of $|D|$ in Equation 6 - the result by Riondato and Upfal is tighter than the one by Toivonen only if:

$$\delta < 2^{-(|\mathcal{I}|+1)} e^{2d} \tag{7}$$

By doing some experiments with different values of $|\mathcal{I}|$ and d, we obtain that the bigger the d-bound and the better is the result of Toivonen, while the bigger is $|\mathcal{I}|$ and the better is the result of Riondato and Upfal. However, in a real Dataset, $|\mathcal{I}| >> d$-bound, resulting in that in almost all the cases the result of Riondato and Upfal is tighter. For example, $|\mathcal{I}| = 160$ will still result in $\delta = 1.08\%$ even for $d = 57$. In fact, the major problem that the result by Toivonen has is that the sample size grows linearly with $|\mathcal{I}|$, which in Riondato and Upfal solution this problem was solved.

Hence, Equation 7 could be used as a tool whether to choose which result is better, given δ, $|\mathcal{I}|$ and d.

References

[1] H. Toivonen, "Sampling large databases for association rules," in *Proceedings of the 22th International Conference on Very Large Data Bases*, VLDB '96, (San Francisco, CA, USA), p. 134–145, Morgan Kaufmann Publishers Inc., 1996.

[2] M. Riondato and E. Upfal, "Efficient discovery of association rules and frequent itemsets through sampling with tight performance guarantees," *ACM Trans. Knowl. Discov. Data*, vol. 8, Aug. 2014.