Abstract—Software prediction unveils itself as a difficult but important task which can aid the manager on decision making, possibly allowing for time and resources sparing, achieving higher software quality among other benefits. Bayesian Networks are one of the machine learning techniques proposed to perform this task. However, the data pre-processing procedures related to their application remain scarcely investigated in this field. In this context, this study extends a previously published paper, benchmarking data-driven Bayesian Networks against mean and median baseline models and also against ordinary least squares regression with a logarithmic transformation across three public datasets. The results were obtained through a 10-fold cross validation procedure and measured by five accuracy metrics. Some current limitations of Bayesian Networks are highlighted and possible improvements are discussed. Furthermore, we assess the effectiveness of some pre-processing procedures and bring forward some guidelines on the exploration of data prior to Bayesian Networks’ model learning. These guidelines can be useful to any Bayesian Networks that use data for model learning. Finally, this study also confirms the potential benefits of feature selection in software effort prediction.

I. INTRODUCTION

Accurate software predictions can provide significant advantages in project planning and are essential for effective project management being strongly linked to the success of software projects. Underestimating the effort can cause delays, degrade software quality and bring about increased costs and dissatisfied customers. On the other hand, overestimating the project’s effort can lose a contract bid or waste resources that could be allocated elsewhere. Although the primary objective of software effort prediction is budgeting, there are also other important objectives. Boehm et al. [1] mention tradeoff and risk analysis, project planning and control and software improvement investment analysis.

Since the nineties, researchers began applying machine learning techniques for software effort prediction [2] [3] [4]. Ever since, studies on machine learning techniques for software prediction have grown more and more common. Currently this is visibly a thriving trend with many empirical studies being published regularly and comprising a very active research field. In a systematic review, Wen et al. [5] identified eight machine learning techniques employed in software prediction including CART (a type of decision tree) [3], Case-based Reasoning (CBR) [4], Artificial Neural Networks, Genetic algorithms, Support Vector Regression among others. CBR, Artificial Neural Networks and Decision Trees were considered by Wen et al. [5] the most popular machine learning techniques in software development effort prediction research.

One of these machine learning techniques are Bayesian Networks (henceforth BNs), which is the technique we assess in this study. BNs were initially proposed and are generally more common in software quality prediction. Since then there has been a steady increase of efforts towards BNs in software effort prediction and in software projects management in general. Wen et al. ranked BNs fourth in popularity in software development effort prediction among the machine learning techniques. This technique has some distinguishing features that make it look suitable to deal with the uncertainties prevalent in this field. BNs will be discussed briefly in the next section.

This research field has suffered with contradictions and few clear conclusions. In spite of the large number of empirical studies there are conflicting results and conclusions instability [6] [7] [8]. Shepperd and Macdonell [9] state that ‘empirical evaluation has not led to consistent or easy to interpret results’. This matters because it is hard to know what advice to offer to practitioners. There are many examples of contradictions in comparisons among different machine learning and statistical techniques, as described for instance in, e.g., [7], [9]. Part of these inconsistencies stem from differences in the experiments and sometimes from errors in the procedures like discussed in, e.g., [9] and [10]. The latter study points out mistakes in the application of regression models. Myrteit, Stensrud and Shepperd [11] discuss reasons for the unreliability of conclusions in detail, chiefly focusing on validation and measuring, and concluded that more reliable research procedures are necessary. Several other researchers have made suggestions about the validation of results in comparative studies, e.g., [12], [13] and [9].

With regard to BNs, details on their employment and the preparation and pre-processing prior to model learning remain scarcely investigated. There is some uncertainty about its effectiveness and about the pre-processing procedures applied prior to model learning. Given the relevancy of BNs in software prediction research, investigations on its employment and effectiveness are necessary.

In this context, this study strives to assess the employment of data-driven BNs in software effort prediction through extensive validation procedures, including analyses on data pre-processing, providing guidelines on how to best explore data, and discussing BNs’ current limitations and possibilities of improvements. The investigation of data-driven BNs matters because even if this might not become the best way to apply
them, the optimization of data exploration is an important direction of development for this technique. By finding ways to optimize the exploration of data there can be benefits to any BNs that use data.

This paper extends a preliminary work [14] by assessing other pre-processing steps, and extending significantly the validation by including other metrics and another dataset, and also by refining the observations on the results.

This paper is organized as follows. In section II we present a brief overview on BNs. In section III we mention some closely related studies. In section IV we bring forward the empirical procedures, datasets used, and how we compared the prediction systems. In section V we analyze and discuss the results and finally put forth the conclusions in the last section.

II. BAYESIAN NETWORKS

BNs [15] [16] are a modeling technique which boasts some distinguishing characteristics. A striking feature of this modelling approach is the possibility, through application of probability theory, to model uncertainty or subjectivity. The probability distributions allow for the integration of objective evaluations, learned from data, with subjective evaluations defined by experts. Furthermore, this allows the model to output several possible outcomes with varying degrees of certainty, unlike deterministic models like linear regression which simply output a single possible outcome, i.e., a numeric value.

BNs comprise a qualitative part, i.e., the graph structure that models the dependencies among a set of variables, and a quantitative part made up of node probability tables (NPT’s) which contain the probability distributions for each node. The graph structure is a directed acyclic graph (DAG) encoding the dependencies among the variables. The nodes represent the relevant variables or factors in the domain being modeled, and each directed arc depicts the dependencies among these factors which can be causality relationships. The NPT’s contain the prior probabilities (in case the variables has no parents) or conditional probabilities (in case the variable has one or more parents). The conditional probabilities define the state of a variable given the combination of states of its parents. With the definition of these probabilities during the training phase a test record can later be classified. These components are illustrated on a simple example in Fig. 1.

Fig. 1. A simple Bayesian Network.

BNs can be modeled fully based on data, through a hybrid approach, i.e., integrating data modeling and experts knowledge or fully expert-based. When the BNs are learnt from data, the learning algorithm strives to identify the dependencies among the variables and thus making up the network structure, i.e., the DAG. The algorithm will identify a model that best fits the relationship between the attribute set and the response variable on the input data (training data). Thereafter, the probability distributions are learned for every combination of variables. This happens during the so called training or learning phase.

The BNs found in this research field most frequently consist of discrete variables. The tool used in this study currently does not support continuous variables. Although some tools offer support to continuous variables, this support has limitations, e.g., imposing restrictions in the relationships among the variables or making assumptions about the distributions of the continuous variables. There are progresses concerning continuous variables in machine learning research and there are also constant developments in the BNs tools, so these limitations could be overcome in the future. For a more detailed review on BNs we refer the reader to other works in the field, e.g., [17], [18], [19] and to data mining literature [15], [16].

III. RELATED WORK

In this section we describe some closely related studies. Radlinski and Hoffman [20] carried out a comprehensive benchmarking study comparing 23 classifiers in WEKA over four public datasets. The authors state their main research question is: “Is it possible to easily predict software development effort from local data?”. So, they establish two specific constraints: easy predictions and using local data, i.e., data from a single company. This paper focused more on the practitioners viewpoint, trying to avoid complex and time-consuming procedures. So, the authors do not address specific details of the techniques but provide a wide-ranging assessment of easy-to-use machine learning classifiers. By comparing so many classifiers this study illustrates very well the lack of stability of the ranking of the techniques across different datasets. They mentioned that due to the ranking instability it is difficult to recommend practitioners with a particular model even though they did conclude that K* technique with feature selection was the most accurate overall. BNs were among the most accurate predictors in two of the four datasets but did not particularly stand out. They also demonstrate that most techniques achieve higher accuracy by performing feature selection.

Mendes and Mosley [13] outline thorough experiments comparing BNs, CBR, manual stepwise regression and simple mean and median based models for web effort prediction using Tukutuku, a proprietary cross-company dataset. The study compares four automatic and four hybrid BN models. The results were unfavourable to BNs, with most of the models being more inaccurate than the median model and two of them barely matching it. The authors conclude that manual stepwise regression may be the only effective technique for web effort estimation. Furthermore, they recommend that researchers benchmark proposed models against mean and median based models as they show these can be more effective than more complex models.

One of the last investigations can be found in [21] wherein comprehensive experiments are laid out yielding a benchmark of some statistical and data mining techniques, not including however, BNs. This study benchmarks numeric predictors, as
opposed to [20] which assesses classifiers, i.e., discrete class predictors. This study included thirteen techniques over eight public and private datasets. Their results “indicate that ordinary least squares regression with a logarithmic transformation performs best”. They also investigate feature subset selection with a wrapper approach confirming the improvements brought by this technique. The authors also discuss appropriate procedures and address efforts towards statistically rigorous analyses.

A survey covering BNs for software development effort prediction can be found in [19].

IV. EXPERIMENTS SETUP

We assess data-driven BNs by comparing them to ordinary least squares regression with a logarithmic transformation, which was found in [21] to be invariably among the most accurate predictors. We remind the reader once again that we are comparing a classifier, i.e., a discrete class predictor, to a regression technique, i.e., a numerical predictor. We do this by converting the BN’s class predictions to numeric ones by means of a variant of the method originally proposed in [18] which will be explained in subsection C.

We decided to experiment performing a logarithmic transformation on the data prior to BNs’ building. So, this variant is included in the comparison amounting so far to three prediction systems. Furthermore, we also assess the effectiveness of feature subset selection [22] [15] as a pre-processing step. This technique has been employed with good results in this field, e.g., [23], [21], [20]. So, for each of the aforementioned models there is a variant with the application of feature selection prior to model building which multiplies by two the number of prediction systems. So, there are four variants of BNs and two variants of OLS regression amounting so far to six prediction systems.

Finally, we include in the comparison mean and median based models like proposed in [13]. These models simply use the mean and median of all projects effort as a constant prediction. These are very simple benchmark models and an effective model should be able to be more accurate than them. The comparison with such models allows us to better assess the effectiveness of the other techniques by establishing a minimum benchmark of accuracy. The inclusion of such benchmark models is another recent trend proposed in several studies like [13] and [9], with the goal of verifying whether the models are effectively predicting and therefore bringing clarity to the results. So, with these two benchmark models we have in total eight prediction systems.

An abstract outline of the experiments we carried out is shown in Fig. 2. We omitted the different versions of the dataset and the two models of BNs on log-transformed data to avoid cluttering up the figure, for intuitiveness’ sake. So, prepared data is an abstract entity which represents any of the datasets versions (log-transformed or not, and discretized or not) and besides the six prediction systems depicted in this figure there are two BNs on log-transformed data (with and without FSS) which are not shown. We will explain these procedures in the next sections.

These experiments were carried out in the WEKA data mining tool [24]. The next subsections describe briefly the datasets, the conversion method necessary to compare the techniques and the metrics used to assess accuracy.

A. Datasets

A significant barrier for analysis of findings and replication of experiments has been the lack of publicly available datasets since the employment of proprietary datasets inhibits the replication of experiments and confirmation of results. The PROMISE repository [25] is an initiative that attempts to counter to some extent the lack of transparency that pervades this research field. Datasets are made available allowing for replication and scrutiny of findings with the intent of improving research efforts and to stir up analyses and discussion.

In this work, we used three widely studied datasets available in the PROMISE repository [25], these are the Desharnais, Maxwell and Cocomo81 datasets. These datasets are relatively clean in comparison to other datasets we have checked. They are local datasets, i.e., data was collected within a single company. Table I describes basic information on the datasets.

<table>
<thead>
<tr>
<th>Data set</th>
<th>Local data</th>
<th>Domain</th>
<th>Effort unit</th>
<th>Range years</th>
</tr>
</thead>
<tbody>
<tr>
<td>Desharnais</td>
<td>Yes</td>
<td>Unknown</td>
<td>Person-Hours</td>
<td>1981-1988</td>
</tr>
<tr>
<td>Maxwell</td>
<td>Yes</td>
<td>Finnish bank</td>
<td>Person-Hours</td>
<td>1985-1993</td>
</tr>
<tr>
<td>Cocomo81</td>
<td>Yes</td>
<td>Various domains</td>
<td>Person-Months</td>
<td>1970-1981</td>
</tr>
</tbody>
</table>

The histograms in Fig. 3, Fig. 4 and Fig. 5 illustrate the distribution of data over effort, the dependent variable. Effort is measured in person-hours on Desharnais and Maxwell and in person-months of 152 hours in Cocomo81. In all three cases the variables are positively skewed, i.e., variables with most records situated towards lower values and a few very high outlying values. Desharnais is the least skewed of the three at 2.00. Maxwell is significantly more skewed at 3.35 and Cocomo81 is the most skewed one at 4.48. Skewness is a very common characteristic in software project datasets.

This characteristic poses some hindrances for modeling. In order to carry out linear regression these variables must be transformed as to approximate a Gaussian distribution. With regard to BNs, this is also a problem since the discretization could yield very uneven classes intervals. In such a scenario, the qual-widths discretization technique [26] [15] can produce empty classes and dispose most of the dataset population within just a couple of classes, thus turning the validation highly dubious. If almost all of the data is within just a
couple of classes the model can hardly predict wrong or find meaningful patterns. A very high hit-rate would not be surprising, but the predictions would be meaningless.

When software managers carry out effort predictions they do not know, for instance, how long a project will last, even though they can have an estimate. Therefore, variables whose values are unknown at the time the prediction is to be performed must be removed, e.g., ‘Duration’, ‘Defects’. This is standard practice in the software prediction field. On the other hand, when a sizing variable is quantified in Functions points it is usually included since it can be obtained in the specification phase, depending on the process model.

On Desharnais dataset three variables were removed: the ID variable, ‘YearEnd’ and ‘Length’. On Maxwell dataset three variables were removed: ‘Duration’, ‘Time’ and ‘Syear’. Finally, no variables were removed from Cocomo81 dataset.

In order to carry out OLS regression we removed records with missing values. This amounts to four records on Desharnais dataset and two records on Maxwell dataset. There were no missing values on Cocomo81 dataset. For the BNs models all the records were kept. We also experimented not removing the missing values for the OLS regression model by performing median imputation and the difference on Desharnais dataset, which is the one with more missing values, was minimal. So, we decided to show the results on the dataset without records with missing values because these are the same we used in our previous paper [14]. On Maxwell dataset there were two records with missing values and on Cocomo dataset there were none.

The categorical variables were coded to dummy variables for the linear regression model following good statistical practices [10]. This study also suggests the removal of outliers. Although this is the standard practice for statistical procedures, we decided to keep the outliers for both models for two reasons: To keep the same conditions for both models; and chiefly because these outliers are actual projects which are rare but can happen. They are not noisy or irrelevant entries. Other studies in software prediction also keep the outliers, e.g., [21], [20].

For more detailed information on these datasets we refer the reader to [20] and to the original works referenced in the PROMISE repository [25].

### B. Comparing the Predictions

The prediction systems are compared through numerical metrics. This has been another controversial topic. There is no consensus on what is the most reliable metric [11], the standard metric some years ago used to be MMRE [27], but due to some flaws it lost popularity. MMRE, like other numerical metrics used in this study, is based on the magnitude of relative error. MRE is a measure of the relative error of the actual effort $e_i$ against the predicted effort $\hat{e}_i$.

$$\text{MRE}_i = \frac{|e_i - \hat{e}_i|}{e_i}$$ (1)

MMRE measures the mean of all the predictions’ MREs. This metric has not passed without criticism [27] [6], for it is highly affected by outliers and it favours models that underestimate. MMRE is biased towards underestimates because the magnitude of the error is unbounded when overestimating and limited to at most 1 (or 100%) when underestimating. This is well explained by means of a didactic example in [9]. This bias entails that models that tend to underestimate will be likely to have smaller MREs overall, therefore performing better according to MRE based metrics. Even though this bias is present in all MRE based metrics it is specially so in MMRE.

MdMRE is the median of the MRE’s. It smoothes out MMRE’s bias, for it is more robust to outliers. Aply in-accurate predictions do not bear on MdMRE like on MMRE. So, on the one hand it shows which models are generally more accurate, but on the other hand it conceals which models can be occasionally very inaccurate. This effect is even more pronounced on Pred metric because it ignores completely the predictions with large errors. Pred measures how frequently predictions fall within a specified percentage of the actual effort, e.g., $\text{Pred}_{25}$ tells us how often the predicted effort is within 25% of the project’s actual effort (25 is a common parameter value for this metric). Therefore, this metric ignores the predictions whose errors are in excess of 25% magnitude, i.e., it does not matter for this metric if the error is 30% or 200% (assuming $\text{Pred}_{25}$). This is a limitation which we criticize about these metrics. Obtaining a model whose predictions rarely lie too far from the actual value is certainly
advantageous. This is a desirable quality in a model and these metrics overlook this aspect.

Several studies proposed new metrics discussing their characteristics. But none of these metrics was widely adopted in the research field. MdMRE and Pred appear to be still the most popular. Foss et al. [27] concluded that every metric studied has flaws or limitations and that it is unlikely that a single entirely reliable metric will be found. So, the use of complementary metrics is recommended.

Miyazaki et al. [28], being the first to observe MRE’s bias towards underestimates, proposed MBRE (Mean Balanced Relative Error). This metric addresses this flaw because it makes the relative error unbounded towards both underestimates and overestimates. By making the ratio relative to the lowest value (between actual and predicted values) the bias of MRE based metrics is eliminated, therefore avoiding favouring models that underestimate.

\[
BRE_i = \frac{|e_i - \hat{e}_i|}{\min(\{e_i, \hat{e}_i\})} \quad (2)
\]

However, it has a flaw in that it does not account for negative predictions. Linear regression models can at times predict a negative number and therefore distort a bit the results under MBRE. Kitchenham et al. [12] propose the use of the absolute residuals as another alternative to bypass these problems of MRE based metrics. MAR (Mean Absolute Residuals) being an absolute measure also avoids this bias of ratio metrics like MRE. MAR has the disadvantage of not being comparable across different datasets.

\[
MAR_i = \frac{\sum_{i} |e_i - \hat{e}_i|}{n} \quad (3)
\]

We consider our selection of metrics to be robust with MAR and MBRE being complementary to the MRE based metrics and making the evaluation more reliable. Higher accuracy in MMRE, MdMRE, MAR and MBRE is inferred from lower values, whereas for Pred metric, the higher the value the more accurate the model. In our result tables, the results under MMRE, MdMRE, Pred and MBRE are multiplied by 100 to keep them in a percentage perspective, e.g., 0.253 turns into 25.3.

C. Comparing the Bayesian Classifier to regression techniques

In order to compare BNs’ results to linear regression we used a variant of the conversion method first proposed in [18], and also used in [13], in which the numerical prediction is the sum of the multiplication of each class’ mean by its respective class probability after the probabilities are normalized so that their sum equals one. Instead of using the mean however, we used the median. Each class’ median value \( Md \) is multiplied by its respective normalized class probability \( \rho \), output in the probability distributions of the BN’s predictions. See formula below.

\[
Effort = \rho_{class1} \cdot Md_{class1} + \ldots + \rho_{classN} \cdot Md_{classN}. \quad (4)
\]

Like the aforementioned studies, we used the mean in a preliminary study [14]. We report here accuracy improvements under MdMRE and Pred metric and significant and consistent improvements in MMRE and MAR results when using the median for the numerical conversion. This modification increased accuracy and lessened the amount of outliers, i.e., wildly inaccurate predictions. This happens because the mean of each class is more affected by outliers than the median. These datasets are positively skewed, therefore each class’s mean value (and specially the highest effort class) will be closer to where the outliers are and farther from the majority of the data, pushing the numerical conversion of the output towards higher values. Therefore, when skewness is present the median is a more faithful and accurate representative of the data which makes up each class. An evidence supporting this reasoning is that the larger improvements were achieved on Maxwell dataset which is the more skewed one.

### TABLE II
**Numerical conversion for BNs on Desharnais data set.**

<table>
<thead>
<tr>
<th>Prediction System</th>
<th>MMRE</th>
<th>MdMRE</th>
<th>Pred</th>
<th>MAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bayesian Networks (mean)</td>
<td>70.29</td>
<td>35.69</td>
<td>38.27</td>
<td>2556.98</td>
</tr>
<tr>
<td>Bayesian Networks (median)</td>
<td>57.23</td>
<td>32.66</td>
<td>33.33</td>
<td>2153.52</td>
</tr>
</tbody>
</table>

### TABLE III
**Numerical conversion for BNs with FSS on Desharnais data set.**

<table>
<thead>
<tr>
<th>Prediction System</th>
<th>MMRE</th>
<th>MdMRE</th>
<th>Pred</th>
<th>MAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bayesian Networks + FSS (mean)</td>
<td>68.94</td>
<td>35.49</td>
<td>39.5</td>
<td>2509.52</td>
</tr>
<tr>
<td>Bayesian Networks + FSS (median)</td>
<td>56.18</td>
<td>34.16</td>
<td>39.5</td>
<td>2133.84</td>
</tr>
</tbody>
</table>

The effectiveness of this modification can be seen in the tables. Table II shows the results for BNs on Desharnais dataset. Table III shows the results for BNs with the employment of feature subset selection on the same dataset. Tables II and III show the results on Maxwell dataset. These tables show the results comparing the conversion with the mean against the conversion with the median. BNs with and without feature subset selection are different prediction systems. So, the effect of the conversion method can be assessed by comparing the results on the same prediction system. Comparisons between the two prediction systems do not belong in this section and will be discussed in the results section. Here we are discussing only the improvements provided by this adaptation to the method proposed in [18].

### TABLE IV
**Numerical conversion for BNs on Maxwell data set.**

<table>
<thead>
<tr>
<th>Prediction System</th>
<th>MMRE</th>
<th>MdMRE</th>
<th>Pred</th>
<th>MAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bayesian Networks (mean)</td>
<td>132.69</td>
<td>64.44</td>
<td>22.58</td>
<td>6726.23</td>
</tr>
<tr>
<td>Bayesian Networks (median)</td>
<td>86.18</td>
<td>58.77</td>
<td>24.19</td>
<td>4655.29</td>
</tr>
</tbody>
</table>

### TABLE V
**Numerical conversion for BNs with FSS on Maxwell data set.**

<table>
<thead>
<tr>
<th>Prediction System</th>
<th>MMRE</th>
<th>MdMRE</th>
<th>Pred</th>
<th>MAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bayesian Networks + FSS (mean)</td>
<td>163.53</td>
<td>67.74</td>
<td>19.35</td>
<td>6281.83</td>
</tr>
<tr>
<td>Bayesian Networks + FSS (median)</td>
<td>97.50</td>
<td>55.99</td>
<td>27.42</td>
<td>4854.74</td>
</tr>
</tbody>
</table>
The effect of using the median in the conversion is quite clear for both prediction systems and in both datasets. However, on Desharnais dataset under Pred metric there is no improvement. This can be ascribed to the limitation about Pred discussed in the previous section. This metric ignores predictions whose errors are larger than the parameter used, i.e., 25. All errors over this threshold are ignored. So, an error that is reduced from 100% MRE to 50% MRE will not affect this metric despite being a valuable improvement. We can infer from this that the improvements happened in the predictions that lie outside the 25% error range since all the other metrics clearly show there were improvements.

We can see the impact of this adaptation is quite significant on the Maxwell dataset, which is the more skewed one.

This result can probably be more easily grasped in all detail by the reader after reading the analysis and discussion of results in the next section.

V. RESULTS AND ANALYSIS

Table VI reports on the results for the Desharnais dataset according to the continuous metrics previously exposed. On the Desharnais dataset there is an obvious improvement in the BNs’ hit-rates when applying feature selection. The hit-rates are simply the percentage of times the classifier predicted the right class (therefore, the higher the value the more precise the model). However, when we consider the continuous metrics there were generally no improvements except under Pred. Pred metric resembles the hit-rates in its characteristic of only considering the accurate predictions and ignoring predictions lying far from the actual value. This shows there were more accurate predictions but that there were also more wrong predictions since the other metrics do not show improvements. This illustrates the limitation of Pred metric that we highlighted in subsection B of the previous section. For OLS regression, there is a small improvement under MMRE, MdMRE, MAR and MBRE and a marginal degradation under Pred metric. The improvements were relatively small because the number of variables dwindles in such cases and the feature selection technique cannot find much improvements by further decreasing the number of variables.

The accuracy of the BNs on log-transformed data was about the same as on the non transformed data. The log transformation did not bring improvements to BNs’ predictions. BNs’ performance was very constant regardless of data preprocessing. So, on this dataset, BNs performed relatively well but were more prone to large inaccuracies.

Finally, BNs clearly overcame the baseline models.

Table VII reports on the results for the Maxwell dataset. For being the dataset with the largest amount of variables in this study, it is likely to contain irrelevant variables and benefit the most by undergoing feature selection. This expectation is fulfilled for OLS regression. Feature selection reduced by half the mean of residuals and all the other metrics show large improvements as well.

But again, like on Desharnais dataset, BNs’ performance did not improve convincingly with the application of feature selection. There is a clear improvement on the hit-rates and an improvement under Pred metric, but the other metrics show that the increase of good predictions (i.e., predictions close to the actual value) was offset by larger errors.

It is interesting to observe that when data did not undergo feature selection, the performance of BNs is comparable to the performance of OLS regression. But with the application of feature selection OLS regression has a large improvement in accuracy as opposed to BNs which do not collect any improvement. This highlights that the BNs models are missing very significant improvements in accuracy which are expected with the application of feature selection.

With regard to the logarithmic transformation, the results show small improvements for BNs under all metrics but Pred as opposed to the Desharnais dataset in which there was no effect.

Like on Desharnais dataset, BNs clearly overcame the baseline models. In our view, an important observation on this dataset is the improvement with feature selection that is being missed by BNs. We will discuss the reasons for this after exposing all results.

Table VIII reports on the results for Cocomo81. On this dataset, the logarithmic transformation did yield an observable improvement on the BNs’ predictions, specially under MMRE. This suggests a decrease of large overestimates. We can observe the difference in performance compared to OLS regression grew in comparison to the previous datasets, even though this effect can be slightly reduced by the application of the logarithmic transformation.

Feature selection brought an improvement for OLS regression though not as pronounced as on Maxwell. For BNs, the same pattern of improved hit-rates and no improvements under other metrics which was observed in the other dataset stands on this dataset. This appears to be related to the skewness of the datasets and the loss of precision brought about by the discretization process. Skewness increases this imprecision because it makes the classes more uneven. The logarithmic transformation is only to some extent able to reduce this effect.

Nevertheless, even in this very skewed dataset they were able to overcome both baseline models.

Table IX shows the frequency of underestimates and overestimates for each model over the three datasets. OLS models
have a tendency to underestimate, which is considered less desirable than a tendency to overestimate.

The variables most frequently identified by the feature selection algorithm were related to ‘Size’. In all datasets studies here, a size variable was selected. This variable appears to be frequently the one with the highest predictive value for effort estimation.

### Table VIII

<table>
<thead>
<tr>
<th>Predictor</th>
<th>Hit-rate</th>
<th>MMRE</th>
<th>MdMRE</th>
<th>Pred</th>
<th>MAR</th>
<th>MBRE</th>
</tr>
</thead>
<tbody>
<tr>
<td>BNs</td>
<td>50.79%</td>
<td>134.85</td>
<td>58.64</td>
<td>25.81</td>
<td>551.95</td>
<td>197.82</td>
</tr>
<tr>
<td>BNs+FSS</td>
<td>55.56%</td>
<td>270.64</td>
<td>130.37</td>
<td>9.68</td>
<td>606.22</td>
<td>336.39</td>
</tr>
<tr>
<td>BNs+log</td>
<td>52.38%</td>
<td>91.19</td>
<td>53.64</td>
<td>19.35</td>
<td>536.54</td>
<td>233.15</td>
</tr>
<tr>
<td>BNs+log+FSS</td>
<td>55.56%</td>
<td>76.94</td>
<td>64.93</td>
<td>28.81</td>
<td>530.61</td>
<td>212.73</td>
</tr>
<tr>
<td>OLS+log</td>
<td>-</td>
<td>46.6</td>
<td>30.49</td>
<td>44.44</td>
<td>278</td>
<td>61.83</td>
</tr>
<tr>
<td>OLS+log+FSS</td>
<td>-</td>
<td>44.28</td>
<td>22.98</td>
<td>53.96</td>
<td>297.47</td>
<td>55.97</td>
</tr>
<tr>
<td>Mean model</td>
<td>-</td>
<td>1775.35</td>
<td>571.16</td>
<td>4.76</td>
<td>891.64</td>
<td>1905.81</td>
</tr>
<tr>
<td>Median model</td>
<td>-</td>
<td>235.42</td>
<td>86.25</td>
<td>15.87</td>
<td>642.63</td>
<td>842.24</td>
</tr>
</tbody>
</table>

We can observe in all of these results that feature selection improved clearly and consistently the hit-rates of BNs and the accuracy of linear regression over all datasets. This effect is very pronounced on the Maxwell dataset which is the one with the highest number of variables. Such improvements are expected because the larger the amount of variables in a dataset, the more likely it is for the dataset to contain irrelevant or redundant variables. This emphasizes the importance of applying the feature selection especially on datasets with many variables. It also highlights the fact that many variables in software projects datasets have a small predictive value and can actually make the models less accurate. Therefore, collecting a smaller amount of variables focusing on high data quality may be more interesting for data-based predictions.

This finding is a confirmation of the findings of previous studies, e.g., [23], [21] and [20].

In spite of these clear improvements however, we can see that the improvements of BNs predictions when measured by the continuous metrics was small or at times the accuracy even worsened. Specially on the Maxwell and Cocomo81 datasets, on which the predictions were significantly less accurate than without feature selection as opposed to what one would expect. This contradiction is illustrated in Fig. 6, where we can see improvements in hit-rates and a degradation according to MBRE. According to data mining literature, wrapper approaches like the one applied here use the algorithm’s own accuracy measure to assess the feature subset [22] [15]. And it is obvious the BNs algorithm is not using this numerical conversion to measure accuracy. The model selection is clearly favouring the hit-rates. This brings into question the validity of hit-rates as an accuracy measure or at least highlights its limitation. Improved hit-rates were offset by larger magnitude errors, i.e., less wrong predictions but when the predictions were wrong they were wrong by a larger margin. This could also be seen in the confusion matrices, but they were omitted due to lack of room. So, does the improved hit-rate really reflect a more accurate model? In all these experiments, BNs ended up missing the improvements expected from feature selection. This could make a significant difference in Maxwell and Cocomo81 datasets which are the ones with larger amounts of variables.

It follows from this observation that an interesting development for BNs would be to investigate the feasibility of incorporating this numerical conversion into the BNs algorithms and tools, using it as a measure of accuracy instead of the hit-rates or error-rates. This modification could bring in some improvements in the predictions and also in the effect of the feature selection technique. The application of feature selection would find improvements in overall accuracy even if with lower hit-rates. As it is, the potential improvements expected from feature selection are being wasted in the strive for higher hit-rates. Alternatively, a suggestion for future research is to experiment with other BNs search algorithms, score types and CPT estimators and check out whether these bypass this focus on hit-rates. In this study we restricted ourselves to the K2 search algorithm [29] with Bayes method for scoring of the networks and Simple estimator to estimate the NPTs.

### Table IX

<table>
<thead>
<tr>
<th>Prediction System</th>
<th>Overestimates (count)</th>
<th>Underestimates (count)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BNs</td>
<td>110</td>
<td>96</td>
</tr>
<tr>
<td>BNs + FSS</td>
<td>127</td>
<td>79</td>
</tr>
<tr>
<td>BNs + log</td>
<td>99</td>
<td>107</td>
</tr>
<tr>
<td>BNs + log + FSS</td>
<td>104</td>
<td>102</td>
</tr>
<tr>
<td>OLS + log</td>
<td>86</td>
<td>114</td>
</tr>
<tr>
<td>OLS + log + FSS</td>
<td>91</td>
<td>109</td>
</tr>
</tbody>
</table>

We can also observe a trend in these results. BNs accuracy degrades according to the datasets’ skewness. With increases in skewness BNs struggle to predict accurately. BNs best performance in these experiments was achieved in the least skewed dataset, i.e., Desharnais. When the data is too skewed the discretized classes become too uneven and there is an increased loss of precision with the largest discretized intervals. The highest effort classes tend to be very sparse. An example is the highest effort class defined for the Maxwell dataset which spans a wider interval than all others put together (ranges from 10000 to 64000 person-hours), thus being very imprecise. Besides the effect on the discretization, there is also an effect on the numerical conversion because even a small probability of the highest effort class (Very High) affects the conversion quite significantly. In Fig. 7 we illustrate this degradation by dividing the error margin of BNs by the error of OLS, for each dataset and according to two metrics. We can see that BNs’ error margin increases significantly in comparison to OLS.
as the skewness of the dataset increases under both metrics (datasets are sorted from left to right according to skewness).

Much of the imprecision of the BNs can be ascribed to the discretization process. This subject has been neglected to some extent in this research field and the establishment of guidelines on this could benefit research initiatives. The imprecision brought about by the discretization process is directly related to the skewness of the datasets. In this scenario of highly skewed datasets, the equal-frequencies discretization generates classes' intervals of too different widths and the numerical conversion will show larger error margins. The alternative of equal-widths discretization causes meaningless results, for there will be empty or near empty classes and the model learning will simply state the obvious, predicting nearly always the same class which is the lowest effort class since it contains most of the records. High hit-rates are not only unsurprising but very likely when using equal-widths in very skewed datasets. Unless a log transformation is applied to the data, predictions based on skewed data discretized with the equal-widths method bring in deceitful results. Related to these findings are the results of [30], which compared equal-widths, equal-frequencies and k-means discretization on a subset of a well known dataset and concluded that equal-frequencies with a log transformation can improve the accuracy results according to most evaluation criteria. Further investigations on discretization methods are necessary.

An interesting undertaking was to investigate the effect of the log transformation on the Bayesian classifier. Even though a couple of studies used this transformation, we are not aware of studies assessing its effects. The log transformation was able to provide only slight improvements of accuracy. The results show that in very skewed datasets, transforming the data can be beneficial. Fig. 8 illustrates this improvement according to MdMRE metric. As another suggestion for future research, we observe that it would be interesting to try out this data transformation with BNs that support continuous variables since in these experiments much of the benefit of performing this transformation appears to have been lost with the discretization.

These experiments on data-driven BNs are relevant because the way data is explored can have a significant impact on the model’s performance. Much of the excitement over BNs revolves around their capability to integrate objective and subjective knowledge. Therefore, learning how to optimize the use of data (i.e., the objective part) can improve the performance of not only data-driven BNs, but also hybrid BNs which appear to be the most promising for this research field. Even though BNs solely based on data may not become the most accurate approach in software effort prediction, improvements on the use of data for BNs benefit this technique as a whole and given its relevancy in software engineering, these investigations are necessary. Optimizing the performance of the data mining capabilities of BNs is an essential part in the development of this modelling technique.

Our results on these datasets are more optimistic for BNs than the ones reported in [13], which were obtained on another dataset. Our experiments show the BNs models struggle in very skewed datasets but are still capable of achieving a minimum standard of accuracy. In [13], most BNs, including hybrid BNs, performed worse than the baseline models. Fig. 9 compares the BNs prediction systems to the baseline models according to MBRE metric.

From our studies on the literature and our own experiments, we observe that it appears to be hard to overcome OLS regression when it is properly applied. Our results on OLS regression confirm the results of [21] and the results of [13]. While OLS regression does perform better with regard to accuracy, one must observe that OLS regression as a well established statistical technique is optimized to its best. On the other hand, we have shown in this study that techniques like BNs have room for improvements and are under constant development. As BNs theory evolves and the tools catch up with
the developments, more accurate predictions will be possible. Ideally, if data-driven BNs catch up with OLS regression, they will be very advantageous due to their flexibility and powerful experimenting features. When such a standard is achieved BNs users will be able to trust this technique is exploring data as well as the most accurate data-based models.

Specifically, we have observed room for improvements for BNs with regard to discretization techniques and experimenting with different model selection methods which could provide improvements in accuracy under other metrics than the hit-rates and also optimizing the effects of feature selection. This appears to be a fundamental problem. Furthermore, there are developments in data mining research concerning support for ordinal and continuous variables. These could also bring further improvements in accuracy. And besides these improvements on BNs’ data mining capabilities, there are also improvements concerning support for experts’ model building.

The BNs tools are currently a limitation [19]. The latest developments are not available for most of the tools. In these experiments we did not have the opportunity to experiment with continuous variables nor with dynamic discretization. It would be interesting to verify the improvements techniques like dynamic discretization proposed in [31] could bring in. Although WEKA offers validation advantages over other tools, it does not have other developments from BNs theory. As we already mentioned, an interesting development would be the incorporation of the numerical conversion method. This conversion is not automated in the tools and it can be somewhat cumbersome to perform which may hinder its employment. Having this conversion automated into the tools could be interesting.

Some studies on BNs indicate that BNs’ main strength for the software prediction area lies in their possibility to incorporate domain knowledge and qualitative factors, therefore favouring hybrid or expert-driven approaches. Currently, an advantage of data-driven models like this, as pointed out in [20], is that by owning a projects dataset it is possible to obtain quick predictions as supporting evidence for the expert’s prediction, as opposed to expert based networks which take much more effort to build and to have the NPT’s elicited. The employment of data-based models to support expert estimates has been indicated to practitioners as a means to increase safety and reliability on experts’ estimates, since the situation with expert-based estimations has not been easier than the situation seen in this research field.

Finally, an observation obtained with this study and the difficulties in the field is that it is important to show faithful and realistic results even if they are not positive towards a particular technique. This research field has suffered in the last twenty years due to over-optimism towards some techniques. In recent years, efforts towards correcting inconsistencies and addressing reasons for conflicting results are on the rise even if these show a less than flattering state of affairs in the field. To move forward it is important to recognize the actual situation paving the way for improvements and solutions.

VI. CONCLUSION

This study provided a sound assessment of automatic BNs by means of a comparison with a well established statistical technique and with benchmark models, thereby illustrating its current limitations and possibilities of improvements. BNs’ limitations are discussed and some guidelines on its employment are provided. Specifically, the skewness of datasets prevalent in this research field and the discretization are shown to bring about inaccuracies that limit BNs’ effectiveness.

One suggestion arising from these observations and set forth to the research community is to investigate the feasibility of incorporating the numerical conversion into BNs model building as we consider it portrays accuracy more faithfully than the basic hit-rates. This could make BNs models generally more accurate even if achieving lower hit-rates. Also, the inclusion of this conversion in the tools would be interesting for research undertakings.

We consider this study discusses important matters that are scarcely discussed in software prediction studies and that can be a source of confusion. Most studies have not addressed much attention to dataset properties and implications on model’s functioning. Shedding light on these somewhat neglected topics is an important step to address some of the current difficulties in the field. This study showed some of the problems arising from the datasets in the field and the constraints they impose specially on classifiers. Much of this is related to the discretization process and the uneven classes that it generates. We brought forward some points concerning the exploration of data which we believe to be important for the development of BNs.

There is a limit on how accurate data-driven prediction techniques can be depending on the data used. Therefore, more efforts should be addressed in studying software prediction datasets properties and data pre-processing in order to increase prediction accuracy. The performance of these models is highly dependent on data quality, which is a subject that has not received sufficient attention. Significant improvements could come from investigations on this.

Our observations indicate that BNs have a potential for data-based predictions but still need improvements to catch up with the most accurate data based models. In spite of the apparent advantage of linear models in this scenario, i.e., data-driven modeling, it must be observed that this is only part of the potentiality of BNs. BNs offer experimenting possibilities beyond that of linear regression. The linear regression method can only provide a point estimate, whereas BNs meet other requirements expected from a prediction model.

Furthermore, due to the human factors and inherent uncertainties in software projects, the capability to incorporate expert’s subjective knowledge can provide an advantage over models solely based on data. Bayesian Networks appear to be one of the most suitable techniques for future progresses in this aspect. BNs theory and tools are under constant development and some technical breakthroughs regarding discretization and NPT’s elicitation appear to herald progresses for BNs in software prediction and software projects management in general.

A. Future Work

A topic that could provide some improvements for the software prediction field and that warrants investigations is data pre-processing. Carrying out this work we observed the impact discretization, data transformations and feature selection can have on the models’ performance. Moreover, we observed the implications of and hindrances posed by the characteristics of software projects datasets. In our view, discretization is a
topic that needs thorough investigations as there are currently no guidelines on this.

In this work we applied a specific feature subset selection technique (a Wrapper approach with BestFirst algorithm). It would be interesting to assess whether other feature selection techniques can bypass this focus on hit-rates that this wrapper approach demonstrated. Good improvements could be obtained if BNs could better extract the accuracy improvements expected from feature selection.

Another suggestion is to experiment with other learning and selection algorithms, as in this work we restricted ourselves to the K2 search algorithm with Bayes method for scoring of the networks and Simple estimator to estimate the NPTs. We have the expectation that other algorithms could assess accuracy in a different way, as in this study the algorithms were clearly favouring the hit-rates, which we questioned as an accuracy measure.

Furthermore, investigating BNs with continuous variables and the related pre-processing procedures could yield interesting results.

Also, statistical significance tests could be performed to enhance the validation of the results.

REFERENCES


