

Applying Decision Support Models in the Presence of Incomplete Evidence

Alexander Statnikov, Eva Kasparova, Constantin F. Aliferis

Discovery Systems Laboratory, Department of Biomedical Informatics,
Vanderbilt University, 2209 Garland Avenue, Nashville, TN 37232, USA

Corresponding author: Constantin F. Aliferis

Email address: constantin.aliferis@vanderbilt.edu

Fax: 1-615-936-1427

Telephone: 1-615-936-1425

Summary

Objectives: *Classification models underlying most machine-learning-based medical decision support systems are typically built on complete (or imputed) training data. However, when it comes to applying these models to new patients, many of the predictors can be unavailable. In medical practice, laboratory tests and other diagnostic evidence are collected progressively and not in one round. Therefore, in most practical situations such models have to be used when only a small fraction of the complete set of predictors is known. In the present paper we focus on methodological techniques to address this difficulty.*

Methods: *We evaluated three strategies for dealing with incomplete evidence in application data: support vector machines (SVM) classification with imputation, lazy SVM learning, and Bayesian network learning with exact inference.*

Results: *We found that Bayesian network learning with exact inference provides the most accurate probability estimates regardless of the number of missing predictors. Lazy learning is the second best performing method, significantly outperforming classification with imputation. When there are only a few missing predictors, classification with imputation can be the method of choice due to its computational advantages. Finally, while the exact Bayesian network inference does not scale to domains with very large number of predictors, the lazy learning and classification with imputation methods are applicable in such cases.*

Keywords

Decision support systems, Missing data, Bayesian belief networks, Support vector machines.

Applying Decision Support Models in the Presence of Incomplete Evidence

Alexander Statnikov, Eva Kasparova, Constantin F. Aliferis*

Discovery Systems Laboratory, Department of Biomedical Informatics,
Vanderbilt University, 2209 Garland Avenue, Nashville, TN 37232, USA

1. Introduction

In recent years, the majority of medical decision support systems are constructed using machine learning methods (see breakdown of related citations in PubMed in [1]). Typically, these techniques induce a classification model from a training set of patients that have all findings (predictors) observed. In case there is a typically small number of missing values in predictors in the training data, a variety of imputation methods can be used [2,3]. However, when the system is applied in real-world settings to a new patient, many values of predictors can be unavailable. This is a typical situation in medicine, where predictors are almost always collected sequentially and not in one step. For example, if a diagnostic decision support system incorporates N findings, some patients may have data only for $m < N$ findings since it may be very expensive, time consuming, or dangerous for the patients' health to collect the remaining predictors. It is currently unknown how to optimally deal with incomplete data during application of a decision support system.

In this paper, we investigate three reasonable solutions to this problem¹: In the *first method*, a single classification model M is induced on the training data. When it comes to applying this

* Corresponding author. E-mail address: constantin.aliferis@vanderbilt.edu (C.F. Aliferis).

¹ We assume that both training data and application data are representative of the general population.

model to a new patient case, all missing predictors are imputed based on the training data and the classification model M is applied to the imputed case. The *second method* employs the paradigm of lazy learning; that is, the induction of the classifier is delayed until application time [4]. For example, if the patient has only m out of N predictors observed, we build a patient-specific classification model M_s using only m predictors in the training data and apply this model to the patient. Finally, the *third method* involves learning of the joint probability distribution that generated the training data and performing inference using specialized algorithms (in medicine, it is common to model the complete distribution using Bayesian networks).

We perform a systematic comparison among the above three methods for dealing with incomplete application data. Because in medicine it is not only important to identify the most likely diagnosis or outcome given evidence for a specific patient, but also to estimate a probability of this class, we focused on classification models and error metrics suitable for estimation of conditional probabilities. In our evaluation, we also consider the effect of different datasets, training sample sizes, and proportion of observed predictors.

2. Methods

2.1. Datasets

In order to properly evaluate the classification methods, we needed data for which we know the underlying (i.e., population) joint probability distribution. Thus, we decided to use data sampled from known Bayesian networks (BNs) obtained from established machine learning repositories and previously published in the literature. These datasets are frequently used as benchmarks in biomedically-oriented machine learning experiments. Since most available Bayesian networks in the public domain are limited to a few dozen or a few hundred variables, we “tiled” existing Bayesian networks to produce networks and data with large number of variables, and we study

scalability of methods. We used the algorithm TileBN [5] implemented in the Causal Explorer toolkit [6] that preserves structural and probabilistic properties of the tiles so that the distribution of the resulting tiled Bayesian network resembles the real-world distribution of the original Bayesian networks. Table 1 describes Bayesian networks used in our study.

For each of the above Bayesian networks, we generated 5 random samples of sizes 300, 500, and 1,000 cases each. These datasets were used for training, and they have all predictors present. Next, for each Bayesian network we generated 1,000 test cases (i.e. “application data”) for different proportions of observed predictors: 10%, 20%, ..., 90%, 100%. These test cases correspond to queries involving different variables in the Bayesian network. One can think of each query as a patient case for which we want to estimate conditional probability with respect to some response variable.

2.2. Gold standard

Since the joint probability distribution is provided by the original Bayesian network, we can use exact probabilistic inference algorithms to compute probability of the outcome for test cases. We used the probability propagation in trees of clusters (PPTC) method [7] implemented in the SMILE library for reasoning in graphical models [8].

2.3. Classification methods

The following three methods were used for probabilistic classification of unseen cases. We note that each of the methods receives on input a training data and an unseen test case to be classified. No method has access to the original data generating Bayesian network.

a. Classification with imputation. We used support vector machines (SVM) classifiers since these methods perform very well in the domain of biomedicine, they are relatively insensitive to the “curse of dimensionality”, and they can capture arbitrary relationships among variables

[9,10]. For multicategory outcome variables, we executed SVM in one-versus-one fashion [11]. Even though this is sometimes mildly suboptimal compared to one-versus-rest multicategory classification [10], the one-versus-one method is much faster [11] which is crucial in our experimental setting. We used the libSVM implementation of SVMs with RBF kernel [12]. In order to convert SVM outputs to probabilities, we used the method [13] implemented in [12]. The parameters of this method were optimized by cross-validation.

Given a probabilistic SVM classifier model M obtained on the training data, we applied it to the unseen test case with missing values imputed by the nearest neighbour imputation method [2].

b. Lazy learning. For each previously unseen case with missing predictors, we build a probabilistic SVM classification model M_s using training data only for predictors observed in the test case. Then we apply the model M_s to this case.

c. Bayesian network learning and inference. In this method, we first learned a graph of the Bayesian network from the training data using the Max-Min Hill-Climbing (MMHC) algorithm [14] implemented in the Causal Explorer toolkit [6]. The choice of the MMHC was motivated by a recent comprehensive comparison of BN learning methods which revealed that MMHC outperforms other algorithms in terms of both reconstructing the original graph of the Bayesian network and capturing the joint probability distribution [14]. Given a graph produced by MMHC, we obtained a Bayesian network by computing conditional probability distribution of each node given its parents from counts in the training data. Finally, for each unseen test case with incomplete predictors, we estimated class probabilities using PPTC exact inference algorithm discussed above.

2.4. Evaluation metric

We used the mean squared error metric to compare probabilities output by each classification algorithm with the true probabilities produced by exact inference from the data generating Bayesian network. This metric is commonly used in classification and regression problems to compare continuous outputs such as probabilities [15]. The smaller the value of mean squared error, the better the classification algorithm performance.

3. Results

3.1. Analysis of mean squared error

In Child10 dataset (Figure 1), the Bayesian network learning method with exact inference yields the most accurate probabilities; lazy learning does slightly worse; and classification with imputation is significantly outperformed by the above two methods. Another observation is that the results are not significantly affected by the training sample size.

Notice that the ability of the BN learning and lazy learning methods to accurately estimate conditional probabilities decreases with more observed predictors. At first glance this seems counterintuitive since the less predictors are missing the better one would expect a model to predict the outcome. To understand why one should not *a priori* expect this dependency between the number of observed predictors and mean squared error in estimating probabilities, consider the following example. Assume, we have a decision support system that diagnoses myocardial infarction (MI) on the basis of 30 predictors: presence of chest pain at rest, “squeezing” chest pain, irradiation of chest pain, age, diabetes, fatigue, and 24 ECG features. An elderly diabetic patient shows up at the hospital who feels fatigued but does not feel chest pain. The decision support system predicts MI with probability equal to 0.02%, while the true conditional probability is equal to 0.01%. Thus a decision is made to send the patient home. Suppose further that if the

ECG data were available for this patient, the decision support system would diagnose MI with probability equal to 70%, while the true conditional probability is equal to 90%. Thus the patient would be examined further and hospitalized as necessary. Notice that the squared error in the former case is much smaller than in the latter case. However, diagnosis in the presence of more predictors, despite being less accurate, has more clinical value. The larger prediction error when there are many predictors observed is natural due to learning from a small training sample and the “curse of dimensionality” [16].

Results in the Alarm5 dataset (see Figure 2) are analogous to the Child10 data: the BN learning with exact inference method produces the most accurate probabilities and lazy learning and classification with imputation perform worse. Also, the results are not significantly affected by the sample size. However, all values of mean squared errors here are much larger than in the Child10 data. Likewise, the performance of BN learning and lazy learning methods does not improve significantly with decreased number of predictors. We attribute these differences to the underlying probability distributions. The conditional probabilities (of a variable given its parents) in the Child10 data generating BN cover the interval $[0, 1]$ more uniformly than in Alarm5 BN where most probabilities fall into the ends of $[0, 1]$ interval (Figure 3). When there are many conditional probabilities close to 0 (or 1), one would generally need much more sample to estimate the relationships due to the quasi-determinism in the training data. Another interesting observation for Alarm5 data is that classification with imputation does reasonably well (in terms of mean squared error) compared to lazy learning when $<20\%$ of predictors are missing.

The results for Pigs data are presented in Figure 4. They are very similar to results obtained in Child10 data. The best performing algorithm here is BN learning with inference. Also, the classification with imputation is outperformed by lazy learning. When the sample size increases, per-

formance of all methods but imputation becomes slightly better. Another observation is that BN learning with inference method achieves extremely small errors. This can be attributed to the underlying distribution; it is known from prior literature that learning the Pigs BN from data is relatively easier than other BNs given a powerful BN learning algorithm [14]. This implies that one can obtain an accurate approximation of the data generating BN and thus answer the queries accurately.

3.1. Analysis of running time

The time results are summarized in Table 2. The experiments were executed on workstations with Intel Xeon 2.8-3.2 GHz CPUs. The variance in results is primarily due to differences in sample size. Since SVMs are quadratic to the sample size in the training data [9], it takes ~10 times faster to learn a model on 300 samples than on 1,000 samples. The time results reveal that classification with imputation is the fastest method for small samples. This is mostly due to the choice of SVM classifier for our experiments. In general we expect that classification with imputation would be the fastest method overall since it builds only 1 model (for each outcome variable) and avoids costly BN inference. Also, it can be seen from Table 2 that lazy learning is a very computationally expensive method. In order to answer 1,000 queries for 10 different sizes of predictor sets, it requires building of 10,000 classification models which can be very time consuming for many powerful learners. Finally, learning of a single classifier model took <1 min. and answering a single patient query took <10 sec. for any method.

3.2. Studying scalability of the methods

The results in the previous sections illustrate that BN learning with exact inference is the most accurate approach for dealing with incomplete application data. Even though it has been shown that BN learning using MMHC scales well to domains with thousands of variables [14], exact

inference is usually infeasible in such domains [17]. On the other hand, methods such as lazy learning and classification with imputation are not affected by this problem and scale well. We applied the latter two techniques to Alarm50 and Child200 datasets (containing up to 4,000 variables). The running times for experiments are shown in Figure 5. In the region where the exact Bayesian network inference is not applicable, both lazy learning and classification with imputation work, although lazy learning is more computationally expensive.

4. Discussion

In order to increase the scalability of the BN approach, one can resort to approximate inference techniques which unfortunately can still be computationally expensive and are not always accurate [18]. The comparisons presented in our work are based on 5 discrete Bayesian networks from 3 domains. We plan to extend our analysis to include more BNs with different number of predictors. One can also consider applicability of these methods when the distribution is continuous. We also point out that the imputation method used in our study is quite simple and it will be interesting to consider more sophisticated (and computationally costlier) techniques [2,3]. Another extension of this research is to guide decisions for collecting predictors for a patient to be classified by the decision support system. Fortunately, there exist a large body of literature on this topic and methods exist that are readily applicable to our task [19].

Instance-specific modelling is an area of active research in different subfields of propositional and relational machine learning [20,21]. However, the methods originating from this research do not readily translate to biomedical domains. A recent exception is the work in [22] that introduced a method for patient-specific Bayesian model averaging for classification. Contrary to our assumption that application data is representative of the general population, [22] is founded on the opposite assumption: “*A population-wide model is optimized such that it predicts well on av-*

erage when applied to future patients. In contrast, a patient-specific model is specifically constructed for a particular patient. Such a model is optimized to predict especially well for the single patient case for which it is intended.” Furthermore, unlike [22] that imputes all missing testing data in their biomedical evaluation, we considered a variety of situations when different proportions of predictors are missing, including extreme cases of missing the totality of values for the majority of predictors. Finally, [22] uses only a Simple Bayesian classifier for lazy modeling. It is known that Simple Bayes is not able to capture the majority of distributions well, especially so with increased number of predictors. To this end, in our work we based lazy learning approach on SVMs which can represent arbitrary complex functions and are among the best performing methods in biomedicine [9].

5. Conclusion

In this paper we evaluated three strategies for dealing with incomplete application data when applying a machine learning-based decision support system: Bayesian network learning with exact inference, lazy SVM learning, and SVM classification with imputation. We found that the Bayesian network learning with exact inference method provides the most accurate estimates of probabilities regardless of the number of missing predictors. The lazy learning method, although inferior to BN learning with exact inference, performs reasonably well compared to classification with imputation. The main drawback of the lazy learning technique is its computational time which is significantly larger than that of the remaining methods. The classification with imputation is extremely computationally efficient, although its performance decreases with more missing predictors. Finally, while the exact Bayesian network inference does not scale to domains with very large number of predictors, which are frequently encountered in bioinformatics, the lazy SVM learning and classification with imputation methods are applicable to these tasks.

Acknowledgment

The last author of the study is supported in part by grant LM007948-01.

References

1. Aliferis CF, Tsamardinos I. *Machine Learning for Biomedical Decision Support and Discovery*. MedInfo 2004 Tutorial, Available from: http://www.dsl-lab.org/ml_tutorial
2. Little RJA, Rubin DB. *Statistical Analysis with Missing Data*. 2nd ed. New York: John Wiley & Sons; 2002.
3. Tabachnik BG, Fidell LS. *Using Multivariate Statistics*. Needham Heights: Allyn & Bacon; 2001.
4. Aha DW. *Lazy Learning: Special Issue Editorial*. *Artificial Intelligence Review*. 1997; 11:7–10.
5. Tsamardinos I, Statnikov A, Brown LE, Aliferis CF. *Generating Realistic Large Bayesian Networks by Tiling*. In proceedings of the 19th International Florida Artificial Intelligence Research Society (FLAIRS) Conference, 2006.
6. Aliferis CF, Tsamardinos I, Statnikov A, Brown LE. *Causal Explorer: A Probabilistic Network Learning Toolkit for Biomedical Discovery*. In proceedings of the 2003 International Conference on Mathematics and Engineering Techniques in Medicine and Biological Sciences (METMBS), 2003.
7. Huang C, Darwiche A. *Inference in Belief Networks: A Procedural Guide*. *International Journal of Approximate Reasoning*. 1996; 15:225-263.

8. Druzdzel MJ. *GeNIe: A Development Environment for Graphical Decision-Analytic Models*. In proceedings of the 1999 American Medical Informatics Association (AMIA) Annual Symposium, 1999.
9. Cristianini N, Shawe-Taylor J. *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. Cambridge: Cambridge University Press; 2000.
10. Statnikov A, Aliferis CF, Tsamardinos I, Hardin D, Levy S. *A Comprehensive Evaluation of Multicategory Classification Methods for Microarray Gene Expression Cancer Diagnosis*. *Bioinformatics*. 2005; 21(5):631-643.
11. Scholkopf B, et al. *Advances in Kernel Methods: Support Vector Learning*. Cambridge: MIT Press; 1999.
12. Chang CC, Lin CJ. *LIBSVM: A Library for Support Vector Machines*. 2006. Available from: <http://www.csie.ntu.edu.tw/~cjlin/libsvm/>
13. Wu TF, Lin CJ, Weng RC. *Probability estimates for multi-class classification by pairwise coupling*. *Journal of Machine Learning Research*, 5:975–1005, 2004.
14. Tsamardinos I, Brown LE, Aliferis CF. *The Max-Min Hill-Climbing Bayesian Network Structure Learning Algorithm*. *Machine Learning*, 2006.
15. Hastie T, Tibshirani R, Friedman J. *The Elements of Statistical Learning*. New York: Springer; 2001.
16. Mitchell TM. *Machine Learning*. New York: McGraw-Hill; 1997.
17. Cooper GF. *The Computational Complexity of Probabilistic Inference Using Bayesian Belief networks*. *Artificial Intelligence*. 1990; 42: 393–405.

18. Murphy KP, Weiss Y, Jordan MI. *Loopy Belief Propagation for Approximate Inference: An Empirical Study*. In proceedings of the 15th Conference on Uncertainty in Artificial Intelligence (UAI), 1999.
19. Heckerman DE, Horvitz EJ, Nathwani BN. *Toward Normative Expert Systems, Part I: The Pathfinder Project*. *Methods Inf Med*. 1992; 31(2): 90-105.
20. Neville J, Rattigan M, Jensen D. *Statistical Relational Learning: Four Claims and a Survey*. In proceedings of the Workshop on Learning Statistical Models from Relational Data, 18th International Joint Conference on Artificial Intelligence (IJCAI), 2003.
21. Li J, Topor R, Shen H. *Construct Robust Rule Sets for Classification*. In proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2002.
22. Visweswaran S, Cooper GF. *Patient-Specific Models for Predicting the Outcomes of Patients with Community Acquired Pneumonia*. In proceedings of the 2005 American Medical Informatics Association (AMIA) Annual Symposium, 2005.

Table 1. Bayesian networks used in this study. The first 3 BNs were used for main experiments of the paper, and last 2 BNs were used to demonstrate scalability of the methods.

Dataset name	Original BN	Application domain	# of tiles	# of variables
Alarm5	ALARM – BN for monitoring of emergency care patients.	Emergency medicine	5	185
Child10	Child – BN for the diagnosis of infants born with a congenital heart defect.	Pediatrics	10	200
Pigs	Pigs – BN for pedigree of breeding pigs. The pedigree is used for diagnosing the PSE disease.	Veterinary medicine	1	441
Alarm50	ALARM	Emergency medicine	50	1,850
Child200	Child	Pediatrics	200	4,000

Table 2. Single CPU time in minutes required to learn the classifier model(s) and answer 1,000 queries for 10 different sizes of predictor sets.

		Child10	Alarm5	Pigs
BN learning with inference	Learn BN	1-2	1-3	10-25
	Inference	15-16	15-16	29-31
	Total	16-18	16-19	39-56
Lazy learning		90-1000	45-450	250-2800

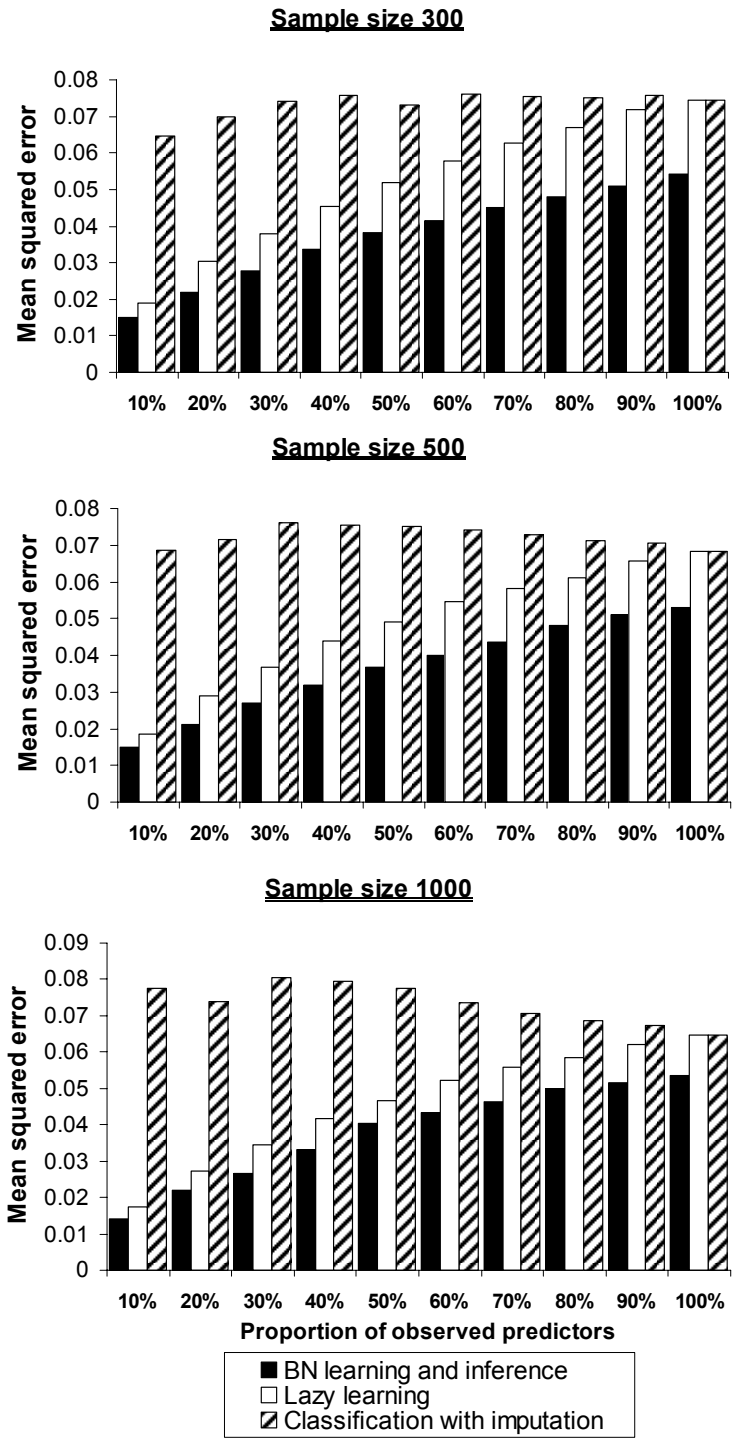


Figure 1. Mean squared error results for Child10 data.

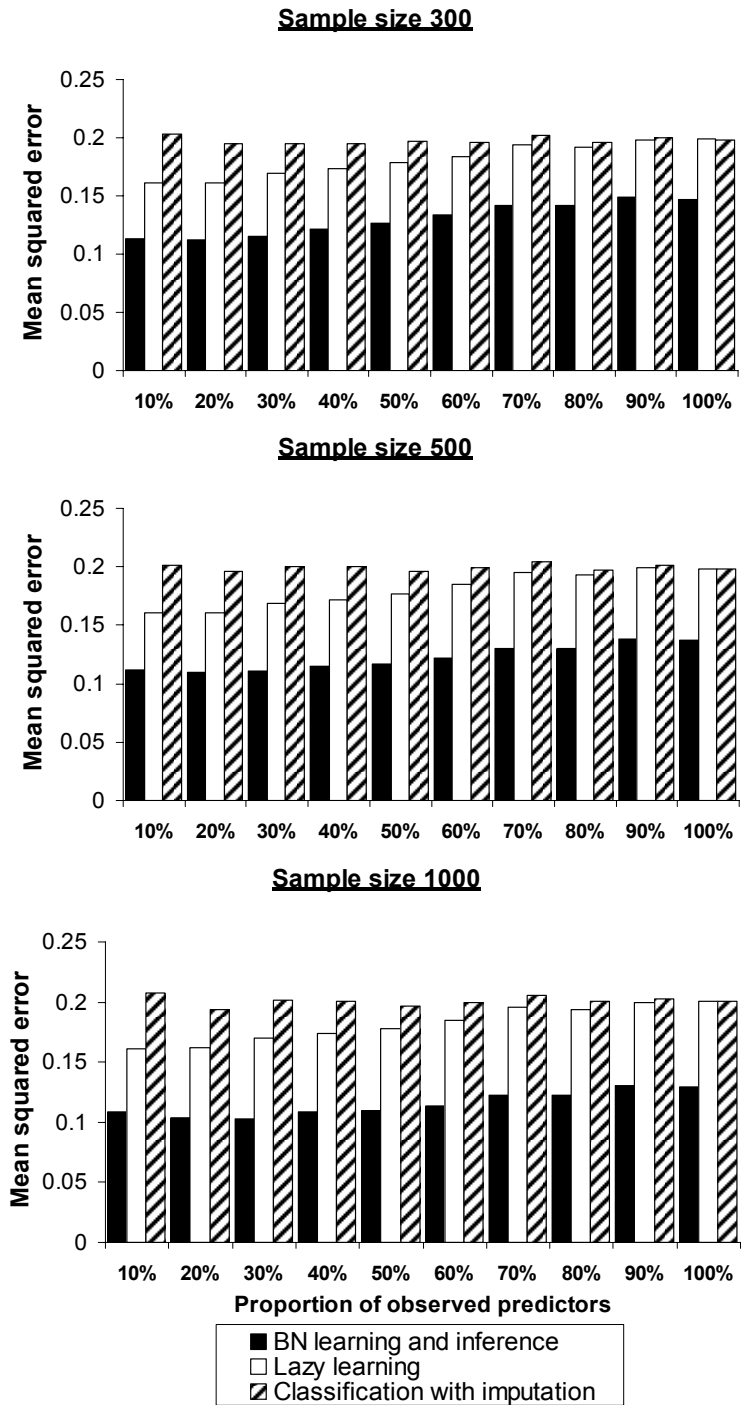


Figure 2. Mean squared error results for Alarm5 data.

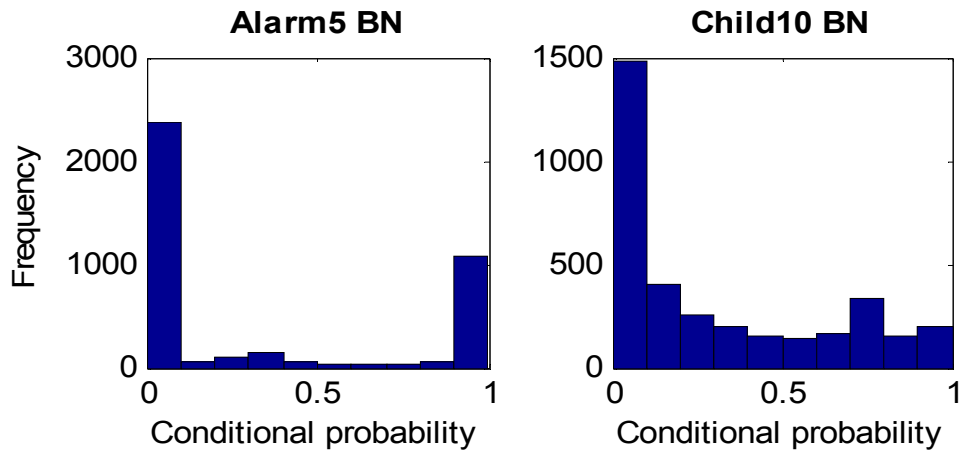


Figure 3. Histograms of conditional probability distributions (of a node given its parents) for Alarm5 and Child10 Bayesian networks.

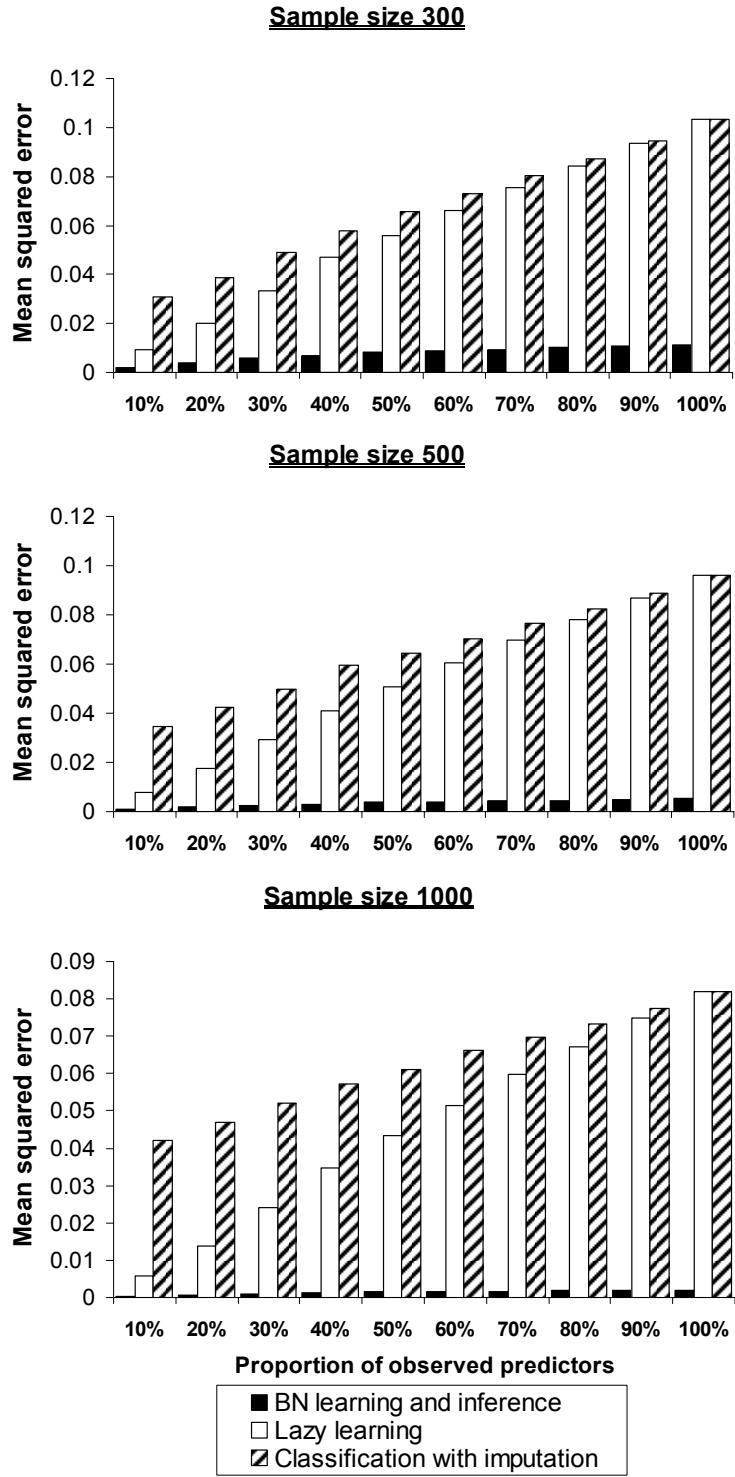


Figure 4. Mean squared error results for Pigs data.

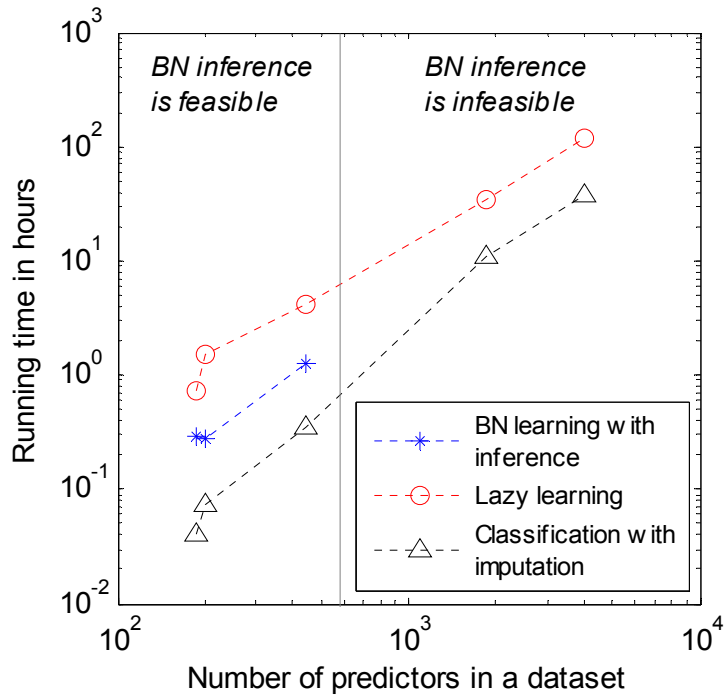


Figure 5. Single CPU time in hours required to learn the classifier model(s) and answer 1,000 queries for 10 different sizes of predictor set for training sample size 300.