Evaluation of Bayesian Networks for Synthetic Data Generation

Katon Minhas, Marc Johnson

Whiting School of Engineering
Johns Hopkins University
Reasoning Under Uncertainty – Fall '22

Abstract

While numerous methods for synthetic data generation exist, few are thought to be suitable for healthcare applications due to their lack of transparency. Fitting Bayesian networks with a defined directed acyclic graph structure to an existing dataset presents a possibility for synthetic data that is both accurate, transparent, and controllable. Two such networks were constructed – the first (using bnlearn in R) had high success with most variables, but struggled to capture certain between-variable relationships. The second (using pgmpy in Python), was unable to properly scale to the required level for practical use. Limitations and implications are discussed.

1. Introduction

1.1 – Problem Statement

As the era of Big Data progresses, data scientists are often challenged to handle large or excessive volumes of data. In the healthcare space, initiatives aimed at improving patient care, increasing profitability, and decreasing inefficiencies often make use of millions of rows of patient and claims data, allowing the same level of insights afforded by data scientists in other fields to be applied to healthcare. However, many challenges exist which make acquiring and accessing this data much harder to come by. Healthcare data tends to be stored in different systems and formats, making it challenging to aggregate and summarize. Data is often full of outdated or missing values, some of which are crucial and provide significant impediments to effective analysis. Finally, the issues of data privacy and transfer regulations mean that the process of receiving access to healthcare data is overly long and tedious, severely limiting the number of firms that are able to implement solutions.

Because of these challenges, synthetic data generation has become a necessity in the healthcare analytics space. Many solutions exist to generate synthetic data, but they typically suffer from either a lack of accuracy or scalability. In addition, many existing models suffer from project-specific issues such as inability to handle mixed datatypes and difficulty incorporating latent

variables (Gogoshin et al 2021). These defects mean that most solutions for synthetic data generation are suitable for certain cases and unsuitable for others. I propose a Bayesian Network-based solution for synthetic data generation that is accurate, broadly applicable, and practical for use in the healthcare analytics industry.

1.2 – Background

Synthetic data refers to any data that is generated artificially, as opposed to data that is measured and recorded. For many purposes, synthetic data can be generated very simply by randomly assigning feature values to a realistic range. However, for effective machine learning and insight generation, advanced methods are needed to generate synthetic data that is realistic enough to be useful. A commonly-used method of synthetic data generation is SMOTE (Synthetic Minority Oversampling Technique), which uses selective sampling to generate synthetic examples for the minority class. Methods for more robust synthetic data generation include Agent-Based Modeling and Generative Adversarial Networks.

Agent-Based Modeling works by producing a number of agents that simulate the actions of real individuals. This lends it very well to generation of synthetic behavioral data for consumer or financial modeling, but does not make it a practical method in the healthcare space.

Generative Adversarial Networks (GANs) have had more success in healthcare applications. These methods make use of two competing neural networks, each of which attempts to generate synthetic examples from a training set to "fool" the other. These networks are widely used in popular image and audio generation softwares. In the health data field, SynthEHR (Synthetic Electronic Health Records) is a commonly used library that improves upon the older MedGAN library (Baowaly). SynthEHR implements both a Wasserstein GAN and a Boundary-seeking GAN that have achieved adequate accuracy in generating synthetic data. However, a few limitations to GANs for synthetic healthcare data exist in practicality. First, the process of data generation with GANs takes substantially longer than other methods, as two large neural networks are being trained simultaneously. Second, the black-box nature of neural networks prevents the user from gaining insight into the nature of the relationships between variables that the network is able to deduce. Depending on the application, this may be an unacceptable hurdle. The medical field is rightfully very risk-averse, and has been reluctant to employ the use of neural networks in other medical applications because of this lack of explainability. If errors are made, they are highly consequential and can significantly affect the lives of patients. For this reason, some level of insight and explainability is necessary.

Bayesian Networks (BNs) offer this benefit over Generative Adversarial Networks. Using a Bayesian network, the structure and relationships of each variable is explicitly described, allowing a human-in-the-loop process that is more suitable for medical use. Bayesian Networks are a form of probabilistic graphical model in which the conditional relationships between different variables are represented using a graph structure. To generate synthetic data, a data analyst with subject-matter expertise is able to customize the synthetic dataset to reflect the important relationships between variables. This has the additional benefit of increasing efficiency as relationships that are irrelevant to the problem at hand are not learned (whereas a GAN would treat them as equally important features).

A few different Bayesian network systems have already been developed for healthcare data generation. Kaur et al (2020) was able to generate reasonably accurate synthetic data based off of the UCI heart disease and diabetes datasets. Bao et al (2021) and Gogoshin et al (2021) presented

similar results. Each of these solutions offered variations of the same general experimental setup: define variable relationships in a Bayesian network using existing libraries or software packages, then use these conditional dependencies to generate realistic new data points. While they achieved success, a significant limitation in each was relative lack of size of the networks. Real-world healthcare datasets may include a few hundred variables, dozens of which are significant for a given analysis. The networks listed above proved successful only for a handful of variables, which may not be sufficient for practical use.

2. Methods

2.1 – Data

Synthetic data was derived from an existing medical claims dataset. The dataset used (originally titled *fact_claim_line*, modified to simply *claims.csv*) was a random sample of 500,000 deidentified insurance claims, chosen from an original set of roughly 1.7 million claims. The original dataset contained 47 features, with various ID numbers and date values included. The final dataset was stripped of these features and only included the following 10 features for synthetic generation:

1) FINAL_STATUS_CD
2) LINE_UNIT_CNT
3) LINE_BILLED_AMT
4) LINE_ALLOWED_AMT
5) LINE_DENIED_AMT
6) LINE_PAID_AMT
7) LINE_COB_AMT
8) LINE_COPAY_AMT
9) LINE_COINSURANCE_AMT
10) LINE_DEDUCTIBLE_AMT

The feature 'FINAL_STATUS_CD' is a Boolean variable indicating whether the claim was accepted or denied. LINE_UNIT_CT is an integer variable representing the number of "units" billable in the given claim. Features 3-10 are all dollar values representing various amounts relevant to the claim.

2.2 - Models

For this analysis, Bayesian networks were constructed using two techniques:

bnlearn

bnlearn is "an R package for learning the graphical structure of Bayesian networks, estimating their parameters, and performing some useful inference" (Scutari). The package allows for Bayesian networks to be generated from discrete data, continuous data, or a hybrid of both.

Some preprocessing was applied to the data to make it suitable for Bayesian network learning. The variables FINAL_STATUS_CODE and LINE_UNIT_CNT were treated as noncontinuous, discrete variables. The other features, representing dollar amounts, were treated as continuous.

These continuous features were logarithmically scaled to reduce skew and to limit the effect of extreme outliers.

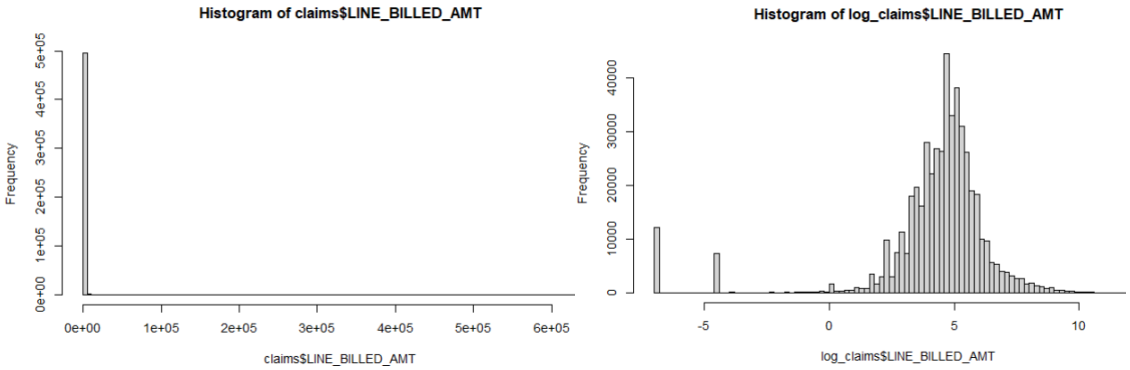

Figure 1. Comparing the distributions of LINE_BILLED_AMT before logarithmic transformation (left) and after (right)

The logarithmically-transformed dataset was fed into a hill-climbing greedy search algorithm using a hybrid Bayesian Information Criterion (BIC, also known as Schwarz information criterion) as the scoring metric. The formula for BIC is shown in Equation (1), where $\hat{L}$ is the maximized value of the likelihood function, k is the number of parameters, and n is the total number of observations.

$$BIC = k\ \ln(n) - 2\ln(\hat{L}) \tag{1}$$

The result of the hill-climbing algorithm was an R structure object containing all the conditional dependencies derived from the data. These dependencies can be visualized in the form of a directed acyclic graph, shown in figure 2 below.

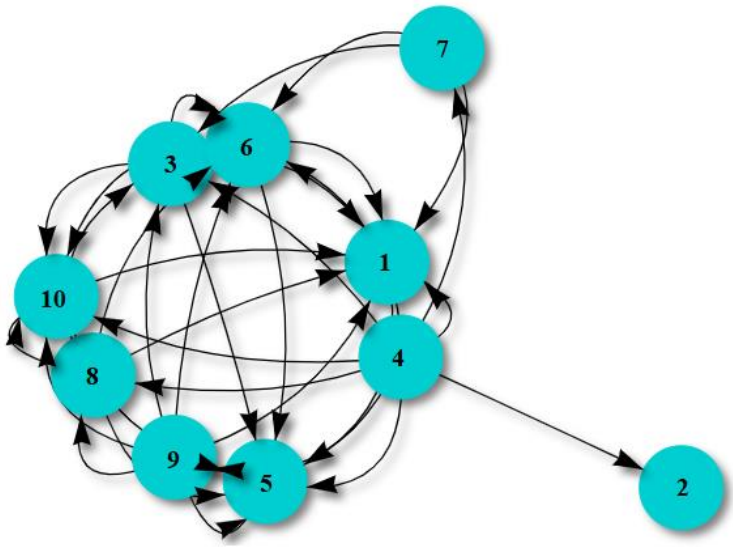

Figure 2. The DAG derived from the hill-climbing algorithm. Labels correspond to features listed in the data section

The graph seen above was used to fit the parameters of a Bayesian network on the log-transformed dataset. Parameter estimation was performed using the maximum likelihood estimator

(mle) for conditional probabilities (for the discrete variables) and least squares regression (for continuous variables).

After the network was generated, the rbn (random Bayesian network) function was used to generate 500,000 observations of synthetic data with the same properties as the logarithmically-transformed dataset. The synthetic data was z-score standardized, then exponentially-transformed to reverse the earlier logarithmic transformation and result in a finalized real-valued dataset.

pgmpy

pgmpy is a "Python implementation for Bayesian Networks with a focus on modularity and extensibility" (Ankur and Panda). Like bnlearn, the package supports operations for both discrete and continuous variables. However, it does not support hybrid Bayesian networks.

Generally, the modelling process using pgmpy as very similar to that of the bnlearn network. The input dataset was cleaned and log-transformed. Due to the large array sizes used to compute conditional probabilities of the network, the size of the dataset was severely limited; only 150 rows were included. Additionally, columns 7-10 were removed from the dataset to reduce computational strain.

An identical DAG as constructed for the bnlearn network was used to create the pgmpy BayesianNetwork model. The DAG effectively had several nodes and edges removed due to the exclusion of the four variables. Maximum likelihood estimation was again used as the estimator function. Forward sampling was used to generate new observations from the fit Bayesian network model.

After post-processing, a small set of synthetic data points was generated. While this data appeared to accurately reflect the relationships seen in the original data, size limitations imposed by the pgmpy BayesianNetwork model prevented a sufficient enough quantity for proper evaluation.

2.3 - Evaluation Methods

There are numerous methods available to measure "accuracy" of synthetically generated data. Accuracy here is a general term to describe data that truly represent some measured aspect of the original dataset. Two measures of accuracy used in this study are listed below:

1) Kullback-Leibler Divergence (KLD) – Measures the similarity between two probability mass functions (PMFs) or probability density functions (PDFs) for each variable. With roots in information theory, KLD can be thought of as a measure of the increase in entropy resulting from use of the synthetic approximation as opposed to the original data (Gogoshin et al). A KLD equal to 0 indicates no difference between the two probability functions. KLD is an asymmetrical measure, so $D_{KL}(P_v \parallel Q_v)$ is not necessarily equivalent to $D_{KL}(Q_v \parallel P_v)$, although the difference in values will typically not be large. For this reason, results reported here are the average of both directions of KLD. This measure also requires that each variable be treated independently, making KLD strictly a measure of within-variable accuracy. The formula for KLD of two PMFs/PDFs, P_v and Q_v is shown in equation 2.

$$D_{KL}(P_v \parallel Q_v) = \sum_{i=0}^{|v|} P_v(i) \log \frac{P_v(i)}{Q_v(i)} \quad (2)$$

2) Pairwise Correlation Difference (PCD) – Measures the difference in correlation between variables for the real data versus the synthetic data. If X_R represents columns *i* and *j* of the real dataset and X_S represents the same columns of the artificial dataset, then the PCD of X_R and X_S is the absolute difference of the Pearson correlations of *i* and *j*. Contrary to KLD, PCD is a between variable measure, computing how well the synthetic data captures the relationships between variables seen in the real data. A PCD equal to 0 indicates that, for that pair of variables, the correlations of the real and synthetic datasets were identical.

$$PCD(X_R, X_S) = |Corr(X_R) - Corr(X_S)| \quad (3)$$

3. Results

Resulting data from the bnlearn model was evaluated for accuracy using the metrics listed above. Evaluation of the pgmpy model was not conducted due to an insufficiently large input dataset and synthetic dataset.

Feature	bnlearn
1. FINAL_STATUS_CD	0.0398
2. LINE_UNIT_CNT	0.0103
3. LINE_BILLED_AMT	0.0074
4. LINE_ALLOWED_AMT	0.0350
5. LINE_DENIED_AMT	0.0295
6. LINE_PAID_AMT	0.0355
7. LINE_COB_AMT	0.0000
8. LINE_COPAY_AMT	0.0011
9. LINE_COINSURANCE_AMT	0.0009
10. LINE_DEDUCTIBLE_AMT	0.0007
Mean KLD	**0.0160**

Feature	bnlearn
1. FINAL_STATUS_CD	0.0347
2. LINE_UNIT_CNT	0.0306
3. LINE_BILLED_AMT	0.1967
4. LINE_ALLOWED_AMT	0.2536
5. LINE_DENIED_AMT	0.2536
6. LINE_PAID_AMT	0.2472
7. LINE_COB_AMT	0.0095
8. LINE_COPAY_AMT	0.0347
9. LINE_COINSURANCE_AMT	0.0453
10. LINE_DEDUCTIBLE_AMT	0.0516
Mean PCD	**0.1158**

Figure 3. Mean KLD of each variable (left) and mean PCD of each variable (right)

Overall, it was found that the synthetic data was able to accurately capture the significant properties seen within the variables of the real data. The mean KLD for each variable was 0.016, while several individual variable KLD values were near-0, indicating very good representation. The variables FINAL_STATUS_CD, LINE_ALLOWED_AMT, LINE_DENIED_AMT, and LINE_PAID_AMT saw KLD measurements significantly higher than the other variables. This higher divergence can be attributed to the highly skewed nature of these variables. In each of the three continuous variables, the real-dataset contained several thousand zero-values. Because the variables are continuous, the synthetic data represented these values with small nonzero values (values < 1.00). Additionally, each of these variables contained a few extreme outliers. For example, a few values >250,000 were seen in the real LINE_AMT column, despite the mean and median of this column being only 82.8 and 13.39, respectively. The extreme skew indicated by these mean and median values mean that when the synthetic dataset was transformed from z-scored and log-transformed values back to true values, even further extreme outliers were found in the synthetic data. Applying outlier removal techniques to the final synthetic dataset would likely result in significantly improved KLD scores for these skewed variables.

Similarly mixed results were seen in the between-variable PCD measures. The table on the right of Figure 3 shows the mean PCD values of each variable with every other variable. Figure 4 (below) shows the breakdown of each PCD score by variable, with each other variable. The highlighted values indicate the high PCD scores seen among LINE_BILLED_AMT, LINE_ALLOWED_AMT, LINE_DENIED_AMT, and LINE_PAID_AMT. Correlations between each of these four variables was much more poorly reflected in the data than that of the other variables. This lesser performance is a result of the same skewing effect seen in the KLD values for these same variables.

	1	2	3	4	5	6	7	8	9	10
1	0	0	0.1	0.1	0	0.1	0	0.1	0	0
2	0	0	0.1	0.1	0	0.1	0	0	0	0
3	0.1	0.1	0	0.4	0.7	0.4	0	0	0.1	0.1
4	0.1	0.1	0.4	0	0.7	0.1	0	0.1	0.1	0.1
5	0	0	0.7	0.7	0	0.7	0	0	0	0.1
6	0.1	0.1	0.4	0.1	0.7	0	0	0.1	0.1	0.1
7	0	0	0	0	0	0	0	0	0	0
8	0.1	0	0	0.1	0	0.1	0	0	0	0
9	0	0	0.1	0.1	0	0.1	0	0	0	0.1
10	0	0	0.1	0.1	0.1	0.1	0	0	0.1	0

Figure 4. PCD values for each variable, with high PCD scores highlighted

4. Discussion

As presented in the results section, the bnlearn model performed extremely well for some variables, and less well for others. The majority of variables were accurately represented in the synthetic data, both from a within and between variable perspective. The variables that were less successful, especially when measuring between-variable accuracy, were likely so due to properties inherent in the original dataset such as a large quantity of zeros and an extreme skew. Additional preprocessing of variables with these properties, in conjunction with applying outlier removal to the synthetic data, is recommended when using Bayesian networks for synthetic data generation.

In this case, the failure of certain aspects of the synthetic data was largely due to the claims dataset used to construct the network. The dollar value columns were extremely variable and highly skewed, presenting difficulties to accurately representing the relationships. The same process used on variables that more closely fit a normal distribution will have far greater success rates.

Furthermore, the additional network developed using pgmpy was less successful for practical applications. Bayesian networks are known to be relatively unscalable in their handling of large and high-dimensional datasets. When presented with the full dataset, the conditional probability arrays calculated by the pgmpy network quickly grew unwieldy and resulted in an extremely limited functionality. The function was only able to produce a few dozen rows of synthetic data at a time, using a DAG with less nodes and significantly less observations (120 versus 500,000) than the bnlearn network. It is concluded that, while the pgmpy implementation of Bayesian networks may prove quite effective for inference, it is unsuitable for practical synthetic data generation.

One major benefit of using a Bayesian network to create synthetic data as opposed to a Generative Adversarial network is the level of transparency and control provided. Explored in this paper was a DAG generated directly from the original data that derived all relevant relationships between variables. In a healthcare-specific setting, slightly inferior performance may be forgiven due to the added benefit of this transparency. Each of the values was generated according to a relationship that the practitioner is aware of. One aspect of this control not explored here is the use of subject matter expertise to inform additional relationships. While this paper utilized an algorithmically derived graph, users in the field may know of specific relationships that they wish to reflect in the synthetic data. These relationships can be easily defined in a DAG, or added/removed from the algorithmically-defined DAG, to create a Bayesian level with a very high level of customizability. The transparent and customizable nature of Bayesian networks should be

considered their primary selling point over GANs and other methods, especially in highly-consequential fields like healthcare.

5. Conclusion

In summary, Bayesian networks may be employed as an alternative to popular model-based approaches for synthetic data generations. However, due to their sensitive nature, these networks require additional care in data preprocessing and may not be suitable for all datasets. For datasets with high dimensionality, GANs and other methods are recommended, as BNs tend to not scale well to high number of features. Features with unusual characteristics such as skew or large quantities of zeros or missing values may not be well represented by Bayesian networks.

In accordance with these limitations, future research in this area should focus on the effect of additional preprocessing techniques, as well as the effect on different types of datasets. Another important element for further investigation is the effect of basing networks on user-defined rather than algorithmically defined DAGs. In some contexts, a subject-matter expert may have sufficient insight to better define variable dependencies than the hill-climbing algorithm would. If this is the case, the synthetic data would not be base purely on observed data, but rather a combination of observed data and assumed relationships. This potential presents interesting possibilities for the easy and advanced generation of custom datasets.

6. References

Ankur and Panda. “pgmpy: Probabilistic graphical models using python”. Python in Science. (2015). https://github.com/pgmpy/pgmpy.

Bao, Ergute, Xiaokui Xiao, Jun Zhao, Dongping Zhang, and Bolin Ding. “Synthetic Data Generation with Differential Privacy via Bayesian Networks.” *Journal of Privacy and Confidentiality* 11, no. 3 (December 24, 2021). https://doi.org/10.29012/jpc.776.

Baowaly, Mrinal Kanti, Chia-Ching Lin, Chao-Lin Liu, and Kuan-Ta Chen. “Synthesizing Electronic Health Records Using Improved Generative Adversarial Networks.” *Journal of the American Medical Informatics Association* 26, no. 3 (March 1, 2019): 228–41. https://doi.org/10.1093/jamia/ocy142.

Gogoshin, Grigoriy, Sergio Branciamore, Andrei S. Rodin, and Department of Computational and Quantitative Medicine, Beckman Research Institute, and Diabetes and Metabolism Research Institute, City of Hope National Medical Center, 1500 East Duarte Road, Duarte, CA 91010 USA. “Synthetic Data Generation with Probabilistic Bayesian Networks.” *Mathematical Biosciences and Engineering* 18, no. 6 (2021): 8603–21. https://doi.org/10.3934/mbe.2021426.

Goncalves, Andre, Priyadip Ray, Braden Soper, Jennifer Stevens, Linda Coyle, and Ana Paula Sales. “Generation and Evaluation of Synthetic Patient Data.” *BMC Medical Research Methodology* 20, no. 1 (December 2020): 108. https://doi.org/10.1186/s12874-020-00977-1.

Guillaudeux, Morgan, Olivia Rousseau, Julien Petot, Zineb Bennis, Charles-Axel Dein, Thomas Goronflot, Matilde Karakachoff, et al. “Patient-Centric Synthetic Data Generation, No

Reason to Risk Re-Identification in the Analysis of Biomedical Pseudonymised Data." Preprint. In Review, May 19, 2022. https://doi.org/10.21203/rs.3.rs-1674043/v1.

Jim Young, Patrick Graham, and Richard Penny. "Using Bayesian Networks to Create Synthetic Data." *Journal of Official Statistics* 25, no. 4 (2009): 549–67.

Kaur, Dhamanpreet, Matthew Sobiesk, Shubham Patil, Jin Liu, Puran Bhagat, Amar Gupta, and Natasha Markuzon. "Application of Bayesian Networks to Generate Synthetic Health Data." *Journal of the American Medical Informatics Association* 28, no. 4 (March 18, 2021): 801–11. https://doi.org/10.1093/jamia/ocaa303.

Scutari, Marco. "bnlearn". https://github.com/erdogant/bnlearn.