Using Bayesian Networks and Gaussian Processes to Solve Medical Diagnosis

Stewart Charles Fisher II
School of Computer Science
University of Lincoln
Lincoln, United Kingdom
25020928@students.lincoln.ac.uk

***Abstract*—This report is a comparison of Bayesian networks and Gaussian process models. The Bayesian networks used include naive structures and structure learnt models, using the PC-Stable algorithm; Gaussian counterparts are also included for the original data formats. The Gaussian process model uses the original implementation by the delivery team with the addition of variable sample sizing.**

***Index Terms*—Bayesian Networks, Gaussian Processes, Medical Diagnosis, Comparative Analysis**

I. Introduction

The objective of this assignment is the analysis of two distinct datasets: Cardiovascular Disease and Diabetes. These datasets encompass a myriad of factors, including demographic information, lifestyle choices, and crucial health indicators. For the Cardiovascular Disease dataset, variables such as age, height, weight, blood pressure, cholesterol, and lifestyle habits are considered. On the other hand, the Diabetes dataset includes features like pregnancies, glucose levels, blood pressure, skin thickness, and genetic factors. Two versions of these datasets were provided, in discrete and continuous granularity.

For Task 1, the provided solution compares the usage of a Naive Bayesian structure against a structure learnt from an original implementation of the PC-Stable algorithm. The structures are compared on their performance using the inference methods provided by the delivery team. These methods were inference by enumeration and rejection sampling; a standardised number of samples is used for rejection sampling. All four datasets are used in this solution.

For Task 2, the provided solution is a comparison of performance for Gaussian Bayes networks and Gaussian processes, to determine the best choice for feature set classification and prediction. The model used for the Gaussian Processes was provided by the delivery team, however the Gaussian Bayes Networks will be examined using both Gaussian Naive Bayes and Gaussian Bayes networks, generated by the original structure generation implementations.

In this multifaceted exploration of medical diagnosis through computational means, the report aims to provide a comprehensive understanding of the chosen methodologies, their implementation, and the subsequent comparative analysis to shed light on the efficacy of Bayesian Networks and Gaussian processes in this domain.

II. Solution

A. Task 1: Bayesian Networks

1) Structure Learning: Two structure learning options are offered in this solution:

- Naive
- PC-Stable

The Naive option takes in the training set CSV file, initialises an `nx.DiGraph`, and add the nodes from the relevant column names. It then adds an edge from every column to the last column in the CSV file, which is the target variable.

The PC-Stable option [1] takes in the training set CSV file, but instead initialises an `nx.Graph` with the column names as nodes; this object has to be used as the orientation of the nodes is not yet known. Every node on the graph is given a node to every other node. Iterating through each node x on the graph, we take each remaining neighbour y of x and execute a Chi-squared independence test[1]. If the test returns positive, the corresponding edge between x and y is removed. Once we have gone through each node, we have generated the skeleton graph. The decision to use PC-Stable over Hill Climbing for Bayesian network structure learning was driven by the algorithm's strengths in handling latent variables, robustness to nonlinear relationships and noisy data, and the consistency of results across different samples. These considerations align with the complexities and characteristics observed in the given datasets.

The skeleton graph is then converted to an `nx.DiGraph` so that we can add oriented edges. We then go through each node x in the skeleton graph and create a list of the neighbour and parent nodes y. A conditional independence test is executed between the neighbours y_1 and y_2. If the two nodes are conditionally independent, we then check to see if adding an edge from y_1 to x and from y_2 to x would cause a cycle. If neither proposition causes a cycle, these edges are added to the graph. Before progressing to the next node x, there is a final check for any cycles that need to be removed. Once every node has been checked for possible V-structures, we have a complete directed acyclic graph. An example of a structure learnt DAG can be seen in Figure 1.

[1] There is also the option to use a G-squared conditional independence test.

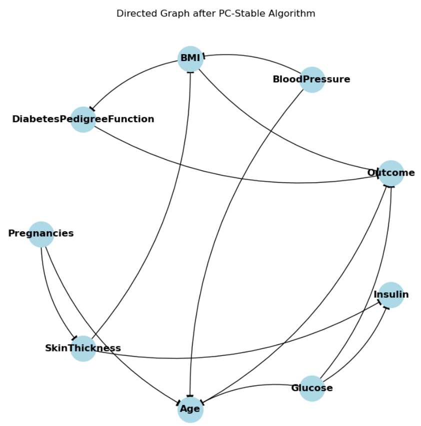

Fig. 1. A directed acyclic graph of the discretised Diabetes dataset.

2) Configuration File Generation: In order to use the `CPT_Generator.py` script provided by the delivery team, I had to implement an original method to parse a generated DAG and write it to the correct format. First, I took the generated DAG and sorted it into topological order. This is done because the formatting for the configuration file requires that the marginal probabilities be listed prior to the conditional probabilities. The dynamically selected model name is appended to the top of the file, then the random variables are listed below. Iterating through the marginal then conditional probabilities, the respective probability structures are appended to the structure line of the configuration file.

The structure that is learnt for the Cardiovascular dataset would require a very long amount of time to generate the CPTs/PDFs. To solve this, the variables that affect the `target` variable have been pruned. The pruned version used in the testing of this solution is `P(target|ap_hi,ap_lo, cholesterol,gluc)`.

3) CPT and PDF Generation: In the main driver script, `main.py`, the relevant CPT/PDF script is chosen based upon the flag used by the user to determine if the original or discretised dataset is to be used.

4) Miscellaneous:

- The training time is calculated using the time taken to execute the CPT/PDF script.
- The naive structures were all generated using the `networkx` library, to maintain testing standards across all models.
- The LL and BIC score calculations are implemented via an original script, and can be toggled to display with a flag.
- The graph of the generated DAG can be toggled to display with a flag.

B. Task 2: Gaussian Processes

The Gaussian processes implementation provided by the delivery team performs binary classification, to predict for two classes, either 0 or 1. Two variants of a baseline classifier are included, one utilising predicted means only and ignoring variance information, and the other using both predicted means and variance information.

Due to the need to calculate the determinant and the inverse of the covariance matrix, Gaussian processes have a cubic complexity. This makes computing difficult with larger datasets, such as the cardiovascular dataset. To counteract this, a variable sample size was implemented so that computation can still be run on both datasets. Another method that could have been approached would have been the implementation of a Sparse Gaussian process model, which have a smaller quadratic complexity [2].

III. Results and Analysis

A. Task 1: Bayesian Networks

Looking at the performance of the models on the Cardiovascular dataset in Table I, it shows that the models that used the PC-Stable structure learning had better balanced accuracy, F1 score, AUC, and KL divergence, in both of the data granularities. The BIC scores also indicate a better fit for the data. Despite this, training times are generally higher for the PC-Stable models, which is to be expected due to the higher complexity.

TABLE I
A table of collected performance metrics for the Cardiovascular dataset.

	Cardiovascular Dataset	
Metric	**Naive**	**PC-Stable**
Balanced Accuracy	0.7161176726676435	0.7210933227935516
F1 Score	0.6992016870010543	0.7050511124473843
Area Under Curve	0.7760630617700864	0.7366536784372323
Brier Score	0.1983908787644491	0.19791215876463394
KL Divergence	4301.195358060126	4180.73602646731
Training Time	0.21738171577453613	4.043051481246948
Inference Time	0.8016729354858398	1.1217472553253174
LL Score	-480619.90817693167	-447642.7232210715
BIC Score	-480931.46607828443	-753772.9579765666
	Cardiovascular Dataset	
Metric	**Gaussian Naive**	**Gaussian PC-Stable**
Balanced Accuracy	0.6461852699087784	0.7204406312788283
F1 Score	0.6301084856590876	0.6893119519709298
Area Under Curve	0.7014762577791323	0.7656669856660032
Brier Score	0.2203368373775878	0.19236423519600487
KL Divergence	4471.428884719917	4137.700485770201
Training Time	28.8204824924469	42.7391791343689
Inference Time	24.91458749771118	23.02874445915222
LL Score	-1284147.1441831004	-1166559.9517446097
BIC Score	-1376515.8630420486	-2540280726210.0396

Looking at the performance of the Diabetes dataset in Table II, we can see that the Naive models seems to perform the best across most metrics. Comparing this to the Cardiovascular dataset, it could imply that the smaller dataset doesn't lend itself well to structure learning.

TABLE II
A TABLE OF COLLECTED PERFORMANCE METRICS FOR THE DIABETES DATASET.

	Diabetes Dataset	
Metric	**Naive**	**PC-Stable**
Balanced Accuracy	0.7818181818181819	0.6545454545454545
F1 Score	0.7068965517241379	0.5499999999999999
Area Under Curve	0.8565289256198347	0.7505785123966943
Brier Score	0.14631909603825807	0.18721240104465883
KL Divergence	32.135337836548594	44.167263963668134
Training Time	0.002389669418334961	0.30907678604125977
Inference Time	0.0062716007232666016	0.026340007781982422
LL Score	-6309.191568353202	-5570.689373920111
BIC Score	-6530.057711640289	-11710.127965581483
	Diabetes Dataset	
Metric	**Gaussian Naive**	**Gaussian PC-Stable**
Balanced Accuracy	0.6863636363636363	0.7545454545454545
F1 Score	0.5794392523364486	0.6732673267326733
Area Under Curve	0.7872727272727272	0.8758677685950413
Brier Score	0.17582548854604121	0.1419338810830685
KL Divergence	44.41674747659786	31.98427400798863
Training Time	0.2919800281524658	0.1393887996673584
Inference Time	0.2088334560394287	0.09752941131591797
LL Score	-18509.58214282782	-11185.91132467099
BIC Score	-25740.547616531177	-28665940.01581507

B. Task 2: Gaussian Processes

Due to the size of the Cardiovascular dataset, it was decided that 5% of the training data would be used. From the performance metrics in Table III, we can see that despite the reduced training set, the model had similar performance to the Gaussian Bayes network in Table I, albeit with a relatively longer inference time. If more data were to be used, the performance of the model would likely increase, beyond the performance of the Gaussian Bayes network, but the inference times would also increase greatly.

TABLE III
A TABLE OF COLLECTED PERFORMANCE METRICS FOR THE GAUSSIAN PROCESS APPLIED TO 5% OF THE CARDIOVASCULAR DATASET.

	Cardiovascular Dataset
Metric	**Gaussian Process**
Balanced Accuracy	0.7214950611763749
F1 Score	0.7085201793721974
Area Under Curve	0.7792528758993537
Brier Score	0.21307602589455885
KL Divergence	249.9269927689685
Inference Time	51.58620548248291

Unlike the Cardiovascular dataset, the Gaussian process model used on the Diabetes dataset, in Table IV, did use the entirety of the dataset. Like the Cardiovascular dataset, the model showed similar performance to the Gaussian Bayes network with longer inference times. Given the fact that the Diabetes dataset is a much smaller dataset, it might suggest that Gaussian process models perform better with large quantities of data.

TABLE IV
A TABLE OF COLLECTED PERFORMANCE METRICS FOR THE GAUSSIAN PROCESS APPLIED TO THE DIABETES DATASET.

	Diabetes Dataset
Metric	**Gaussian Process**
Balanced Accuracy	0.7272727272727273
F1 Score	0.6336633663366337
Area Under Curve	0.8651239669421488
Brier Score	0.14251132135341318
KL Divergence	45.28698969384692
Inference Time	1.1336994171142578

IV. CONCLUSION

Upon final summary, the choice of model and structure learning approach depends on the characteristics of the dataset. Naive Bayesian structures, work best when there is a small amount of training data, while structure learnt Bayesian networks excel with larger datasets. Even though Gaussian processes show promise, they require careful consideration of dataset size and computational efficiency; they might be the best choice to use if the time taken for inference is not an issue. Otherwise, I would recommend using a structure learnt Bayesian network. Future considerations should involve looking at a comparison of the PC-Stable implementation against an implementation of a score-based function such as the Hill Climbing algorithm. It should also explore the usage of different kernels in the Gaussian process models, as well as different noise levels.

REFERENCES

[1] P. Spirtes and C. Glymour, "An algorithm for fast recovery of sparse causal graphs," *Social Science Computer Review*, vol. 9, pp. 62–72, Apr. 1991. DOI: 10.1177/089443939100900106. (visited on 11/13/2023).

[2] H. Liu, Y.-S. Ong, X. Shen, and J. Cai, "When gaussian process meets big data: A review of scalable gps," *arXiv (Cornell University)*, Jul. 2018. DOI: 10.48550/arxiv.1807.01065. (visited on 11/14/2023).