UNSW
SYDNEY

COMP9418 ASSIGNMENT 1 REPORT

Bayesian Network Analysis of Breast Cancer Symptoms and Diagnosis

Authors
Zhan Wai Kam (z5076319@ad.unsw.edu.au)
Xingyu Chen (z5187472@ad.unsw.edu.au)

Task 1

Implementation Summary

The implementation d-separation test is based on d-separation algorithm. The algorithm first deletes any leaf node in G which does not belong to **X** U **Y** U **Z**. Then, the algorithm will delete all edges outgoing from nodes in Z by simply make the corresponding list to empty. To eventually check whether X and Y are disconnected the algorithm need to traverse the entire graph via DFS. If X and Y is connected then they are dependent otherwise they are independent.

Discussion

The algorithm first deletes any leaf node in G which not belong to **X** U **Y** U **Z**. This process will take at most O(n) times by look through graph representation once. After that, the algorithm will delete all edges outgoing from nodes in Z which will take at most O(1) times which is constant by simply checking corresponding list. Finally, the DFS will take at most O(mn) times in order to traverse the new pruning graph. Therefore, the time complexity is O(mn).

Task 2

Implementation Summary

The conditional probability tables of each node were generated by identifying all combinations of the parameters of the node and its immediate parents, summing all occurrences of each combination, and dividing by the total number. This produces sets of probabilities which sum to 1 within the conditional probability table. The categorical values of each parameter were mapped to integers ranging from 0 – 4. It can be noted that some combinations have zero occurrences, resulting in a zero probability such as P(Mass = Malign, BreastDensity = high, BC = No).

Discussion

The size of the joint distribution of all 16 variables would be $2^9 \times 3^4 \times 4^3$ as it is the product of the outcome space of each variable. In spite of the graph being a parameter of the learn_bayes_net function, it was found that the graph was not required for this function.

Task 3

Implementation Summary

Forward sampling was achieved by identifying all parents and ancestors of a given node and arranging them in topological order. A uniform prior probability distribution, or beta(1,1) distribution, was used to generate random probabilities at each node because we have no knowledge of the priors.

Laplacian smoothing was applied to prevent zero probability occurrences which can jeopardise the calculation of the joint probability distribution later.

The output of the forward sampling function is a list the randomly selected variable and the states of itself and its immediate parents.

Discussion

As we add more observed variables in the query, the number of occurrences of that particular query in a finite sample set would be reduced. This would reduce the probability output of the query, affecting its accuracy. This can be counteracted by increasing the sample size proportionally so that the accuracy is less affected by adding more observed variables into the query.

Task 4

Implementation Summary

To train the network, the data had to be segmented into a training and test set for cross-validation. A 80:20 split was chosen to sufficiently train the network to predict the test set. For the Bayesian network, the labels were not separated from the training set as they were required to generate the conditional probability table for BC (Breast Cancer) which would be used to produce the maximum a posteriori hypothesis (h_{MAP}). The labels of the test set, however, had to be separated to be used for verifying the predictions.

To classify the Bayesian Network, we first calculate the joint probability distribution of all 16 variables given the input parameters from the test set. The joint probability distribution can be simplified using Bayes' Rule to equal the product of the conditional probability of all nodes given the state of their immediate parents. This was then easily applied to our model as we already have the conditional probability distributions of each variable.

Discussion

The Bayesian Network classification was able to achieve a 91.90% accuracy score with a 80:20 training and test set split of the dataset. We compared this to an open-source Python library, pgmpy, which provided a Bayesian model where conditional probability distributions can be added. Using the same training set, the pgmpy Bayesian model achieved the same accuracy score of 91.90%, reaffirming that our model is valid.

We then compared the accuracies of other machine learning classifiers such as RandomForestClassifier, KNearestNeighborsClassifier, DecisionTreeClassifer and MLPClassifier (neural network). All of which scored lower than the Bayesian Network classifiers however they had smaller time complexities, except MLPClassifer as it requires building a neural network.

Model	Accuracy Score
Bayesian Network	91.90%
Pgmpy Bayesian Network	91.90%
RandomForestClassifier	90.70%
KNearestNeighborsClassifier	88.85%
DecisionTreeClassifier	88.05%
MLPClassifier	91.65%