Brunda Chouthoy

CSC 529 – Bayesian Networks

Case Study 3

Introduction

The main objective of this case study is to test and possibly expand my knowledge about learning Bayesian networks from data, by exploring various issues such as comparison of Structure learning algorithms and evaluation of the networks learnt. A Bayesian network can be used anytime one can model a system, meet the DAG (directed acyclic graph) requirements i.e. each node should have a directional relationship with at least one other node, there should not be any cycles or loops and d-separation.

I will be using the lung cancer synthetic data and explore the following steps to Bayesian network analysis:

- Network structure learning: algorithmically creating nodes and arcs, and selecting the 'best' network.
- Parameter learning/Training the network: creating conditional probability tables at each node.
- Model validation: validating that the model/network fits the data.
- Inference: estimating network outcomes, given a starting value(s).

k-fold cross validation will be used to validate and test the model to see which network fits the best. The model with the lower expected loss is to be selected as the optimal one.

Data description

'Asia' is a synthetic data set from Lauritzen and Spiegelhalter (1988) about lung diseases (tuberculosis, lung cancer or bronchitis) and visits to Asia. Lauritzen and Spiegelhalter (1988) motivate this example as follows: "Shortness-of-breath (Dyspnoea) may be due to tuberculosis, lung cancer or bronchitis, or none of them, or more than one of them. A recent visit to Asia increases the chances of tuberculosis, while smoking is known to be a risk factor for both lung cancer and bronchitis. The results of a single chest X-ray do not discriminate between lung cancer and tuberculosis, as neither does the presence or absence of dyspnoea."

Brunda Chouthoy
CSC 529 – Bayesian Networks
Case Study 3

Data Source: Lauritzen S, Spiegelhalter D (1988). "Local Computation with Probabilities on Graphical Structures and their Application to Expert Systems (with discussion)". Journal of the Royal Statistical Society: Series B (Statistical Methodology), 50(2), 157-224.

The synthetic data was generated using the R code provided in the bnlearn repository.

Number of instances: 5000

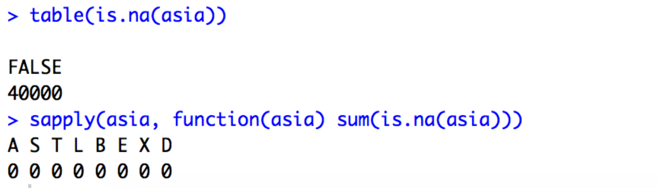
No of attributes: 8

Format: The data set contains the following variables:

- D (dyspnoea), a two-level factor with levels yes and no.
- T (tuberculosis), a two-level factor with levels yes and no.
- L (lung cancer), a two-level factor with levels yes and no.
- B (bronchitis), a two-level factor with levels yes and no.
- A (visit to Asia), a two-level factor with levels yes and no.
- S (smoking), a two-level factor with levels yes and no.
- X (chest X-ray), a two-level factor with levels yes and no.
- E (tuberculosis versus lung cancer/bronchitis), a two-level factor with levels yes and no.

Data Cleaning and analysis

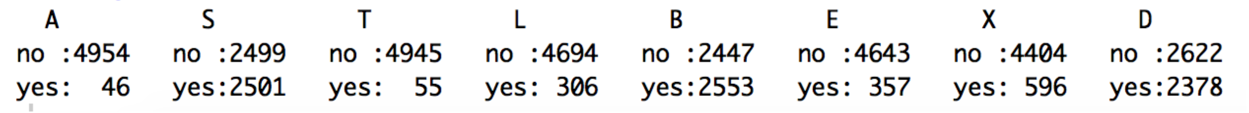
With initial analysis, it could be determined that the Asia-lung cancer data is well organized and has NO missing data.



```
> table(is.na(asia))

FALSE
40000
> sapply(asia, function(asia) sum(is.na(asia)))
A S T L B E X D
0 0 0 0 0 0 0 0
```

Table 1: Check for missing values

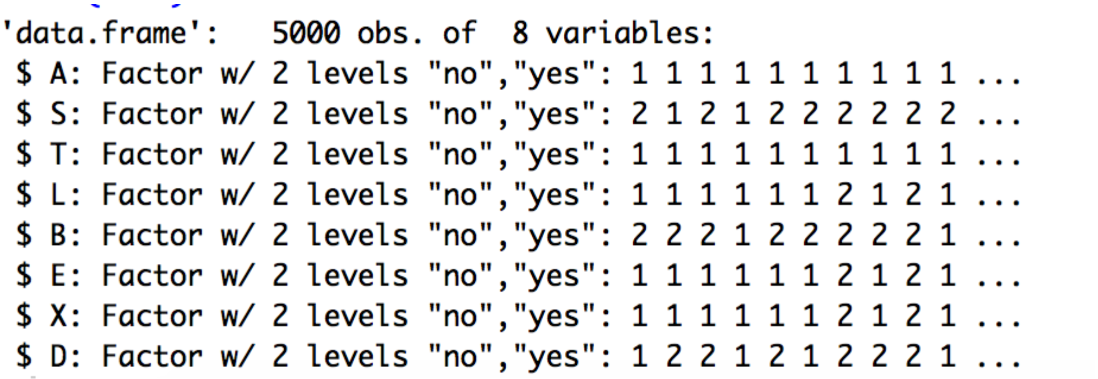
Data distribution and summary is shown in table 2.



```
   A          S          T          L          B          E          X          D
 no :4954   no :2499   no :4945   no :4694   no :2447   no :4643   no :4404   no :2622
 yes:  46   yes:2501   yes:  55   yes: 306   yes:2553   yes: 357   yes: 596   yes:2378
```

Table 2: Summary statistics

The dataset is appropriate for learning and doesn't require any cleaning. Table 3 depicts the structure of the dataset. There is a total of eight discrete variables, stored as factors, each with 1(for no) and 2 (for yes).



```
'data.frame':   5000 obs. of  8 variables:
 $ A: Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
 $ S: Factor w/ 2 levels "no","yes": 2 1 2 1 2 2 2 2 2 2 ...
 $ T: Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
 $ L: Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 2 1 2 1 ...
 $ B: Factor w/ 2 levels "no","yes": 2 2 2 1 2 2 2 2 2 1 ...
 $ E: Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 2 1 2 1 ...
 $ X: Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 2 1 2 1 ...
 $ D: Factor w/ 2 levels "no","yes": 1 2 2 1 2 1 2 2 2 1 ...
```

Table 3: Structure of the dataset

Experimental Results

There are three main types of structure learning algorithms: constraint-based, score-based, and hybrid. The user can specify either AIC (Akaike Information Criterion), BIC (Bayesian Information Criterion), or BDE (Bayesian Dirichlet) scoring to determine the best network structure. The algorithms use different techniques to cycle through various network structures, and then chooses as the 'best' network the structure with best score. The default scoring method for score-based and hybrid algorithms is BIC.

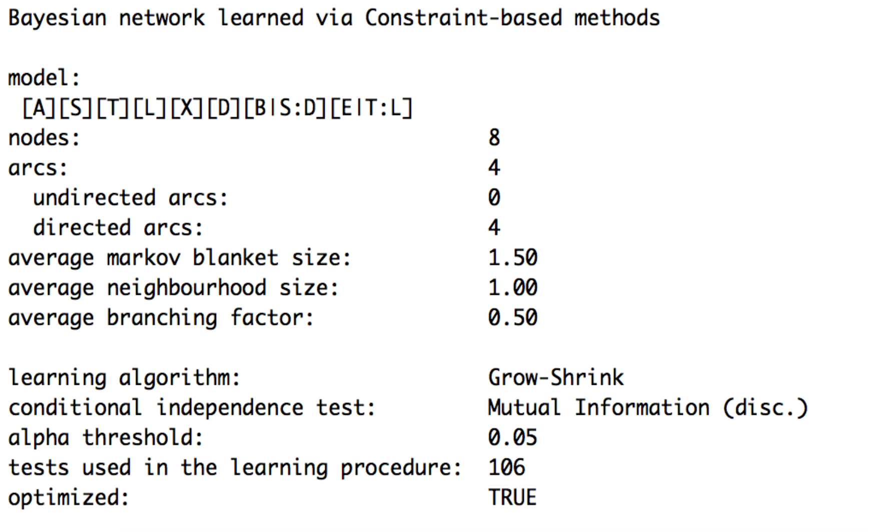
Constraint-based: The algorithms build the structure by searching for conditional dependencies between the variables. For instance, Grow-Shrink algorithm in the 'bnlearn' package.

Score based: User leverages their knowledge of the system to create a network, codes his/her confidence in the network, and inputs the data. The algorithm then estimates the most likely model structure. For instance, Hill Climbing algorithm.

Hybrid: Mixture of constraint-based and score-based methods like Max-Min Hill Climbing(MMHC) in the 'bnlearn' package.

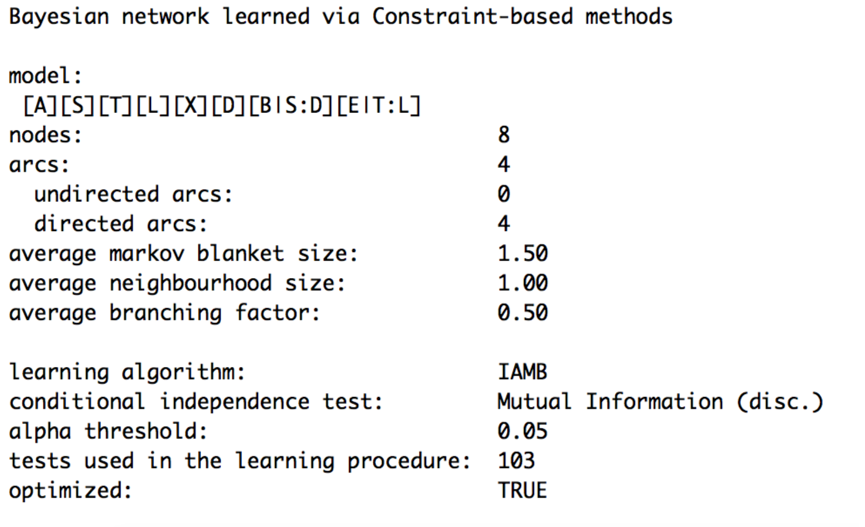
1. Network Structure Learning

Using constraint based algorithms to determine the structure of this dataset:



```
Bayesian network learned via Constraint-based methods

model:
 [A][S][T][L][X][D][B|S:D][E|T:L]
nodes:                                 8
arcs:                                  4
  undirected arcs:                     0
  directed arcs:                       4
average markov blanket size:           1.50
average neighbourhood size:            1.00
average branching factor:              0.50

learning algorithm:                    Grow-Shrink
conditional independence test:         Mutual Information (disc.)
alpha threshold:                       0.05
tests used in the learning procedure:  106
optimized:                             TRUE
```

Table 4: Applying the Grow-shrink algorithm

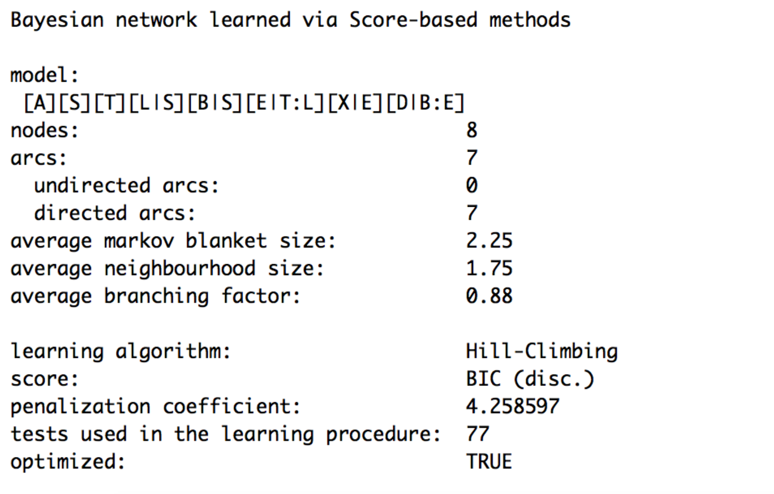


```
Bayesian network learned via Constraint-based methods

model:
 [A][S][T][L][X][D][B|S:D][E|T:L]
nodes:                                 8
arcs:                                  4
  undirected arcs:                     0
  directed arcs:                       4
average markov blanket size:           1.50
average neighbourhood size:            1.00
average branching factor:              0.50

learning algorithm:                    IAMB
conditional independence test:         Mutual Information (disc.)
alpha threshold:                       0.05
tests used in the learning procedure:  103
optimized:                             TRUE
```

Table 5: Applying the IAMB algorithm

Table 4 and Table 5 show the resulting network structure when using constraint based algorithms such as grow-shrink and IAMB respectively. It can be observed that both methods return the same partially directed network structure. There is a total of 8 nodes with 4 directed arcs.

Using the score-based **Hill Climbing** algorithm:



```
Bayesian network learned via Score-based methods

model:
 [A][S][T][L|S][B|S][E|T:L][X|E][D|B:E]
nodes:                                 8
arcs:                                  7
  undirected arcs:                     0
  directed arcs:                       7
average markov blanket size:           2.25
average neighbourhood size:            1.75
average branching factor:              0.88

learning algorithm:                    Hill-Climbing
score:                                 BIC (disc.)
penalization coefficient:              4.258597
tests used in the learning procedure:  77
optimized:                             TRUE
```

Table 6: Applying the Hill climb score based algorithm

Table 6 depicts the results of applying the Hill climb score based algorithm on the data, the structure of which is different from the previous results. The network structures can be compared by plotting the two results using the plot function.

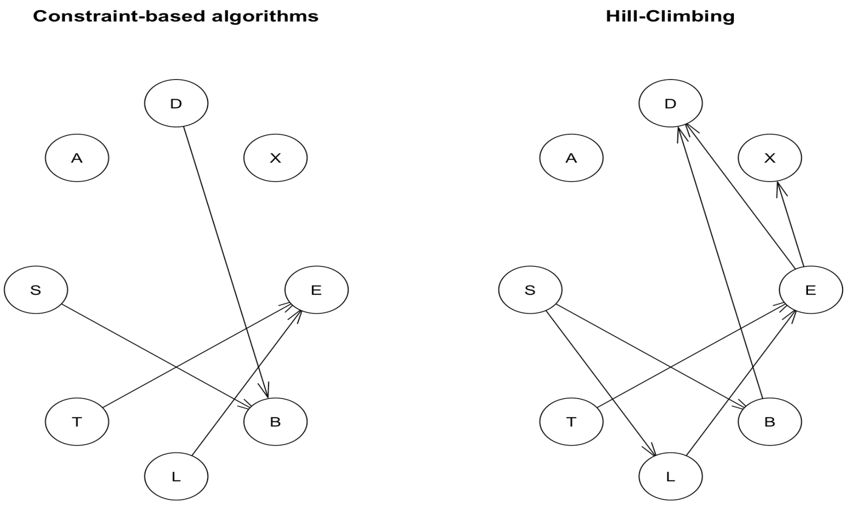
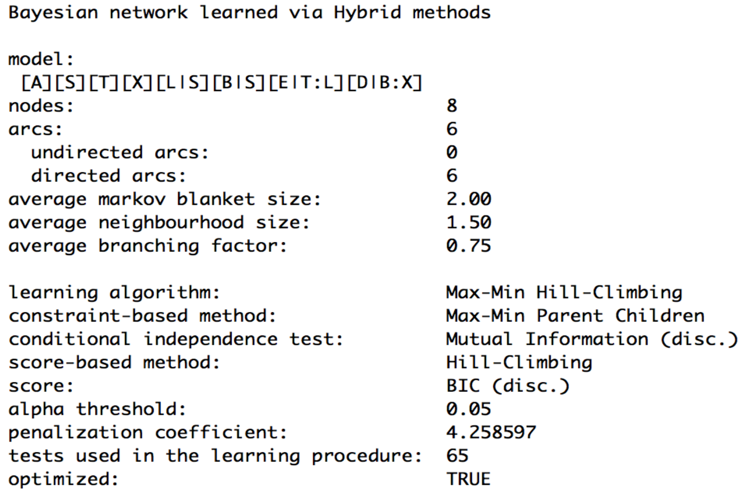


Figure 1: Comparing the network structures

Using the hybrid method with the **Max-Min Hill Climbing (MMHC)** algorthim:



```
 Bayesian network learned via Hybrid methods

 model:
  [A][S][T][X][L|S][B|S][E|T:L][D|B:X]
 nodes:                                 8
 arcs:                                  6
   undirected arcs:                     0
   directed arcs:                       6
 average markov blanket size:           2.00
 average neighbourhood size:            1.50
 average branching factor:              0.75

 learning algorithm:                    Max-Min Hill-Climbing
 constraint-based method:               Max-Min Parent Children
 conditional independence test:         Mutual Information (disc.)
 score-based method:                    Hill-Climbing
 score:                                 BIC (disc.)
 alpha threshold:                       0.05
 penalization coefficient:              4.258597
 tests used in the learning procedure:  65
 optimized:                             TRUE
```

Table 7: Applying the Hybrid algorithm

The results of using hybrid learning with the Max-Min Hill Climbing (MMHC) algorithm is depicted in Table 7. The network plot for the hybrid learning technique is shown in Figure2.

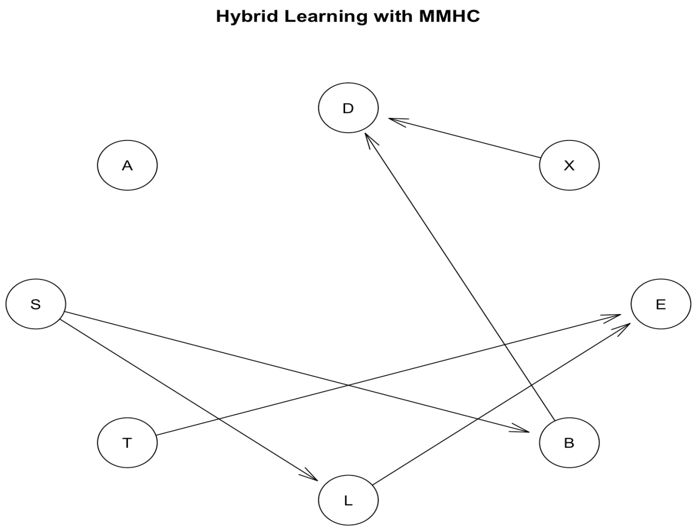


Figure 2: Network structure using the hybrid algorithm

The above algorithm results provide a good example of what happens when the 'best' network structure doesn't contain arcs for all the nodes. The network from the score-based algorithm is the closest to the 'true' network, but the A node is unconnected to the network.

Table 8: Scores for the Hybrid model

Score	HC	MMHC
AIC	-11103.43	-11998.09
BIC	-11196.91	-12086.26
BDE	-11158.83	-12050.23

Table 2: Scores for HC and MMHC algorithm

Table 8 depicts the AIC, BIC and BDE scores for the two networks Hill Climbing (HC) algorithm and hybrid MMHC algorithm.

The case of the unconnected node A can be investigated further. For example, from the true model we know that node A influences node T – i.e. Visits to Asia influences Tuberculosis. The score can be calculated from A to T, and then from T to A. Once the arc is set from A to T, or from T to A, we get the same network AIC score (-11051.09). Thus, the relationship between A and T is termed 'score equivalent', since either direction provides the same/equivalent network score - changing the node direction does not change the network score. Since the algorithms have not been able to determine the relationship of A to T (or other nodes), I have just relied on the true facts for the best relationship of node A to node T or the rest of the structure.

2. Training the Network

Prior to applying the bn.fit function for parameter learning process to get the conditional probabilities, undirected arcs must be set. The above estimated network structure that the 'A' node is not connected to the network structure. Hence, the first step is to connect the node A with a directional arc using the set.arc method in R.

Figure 3 shows the updated network structure of the Hill Climbing algorithm.

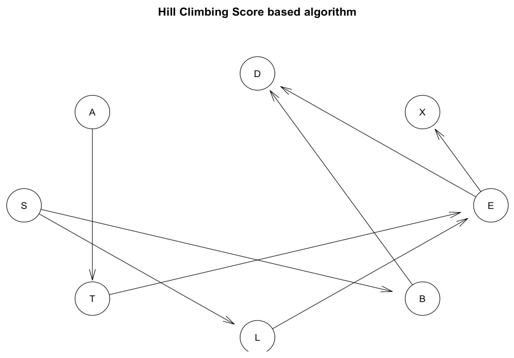
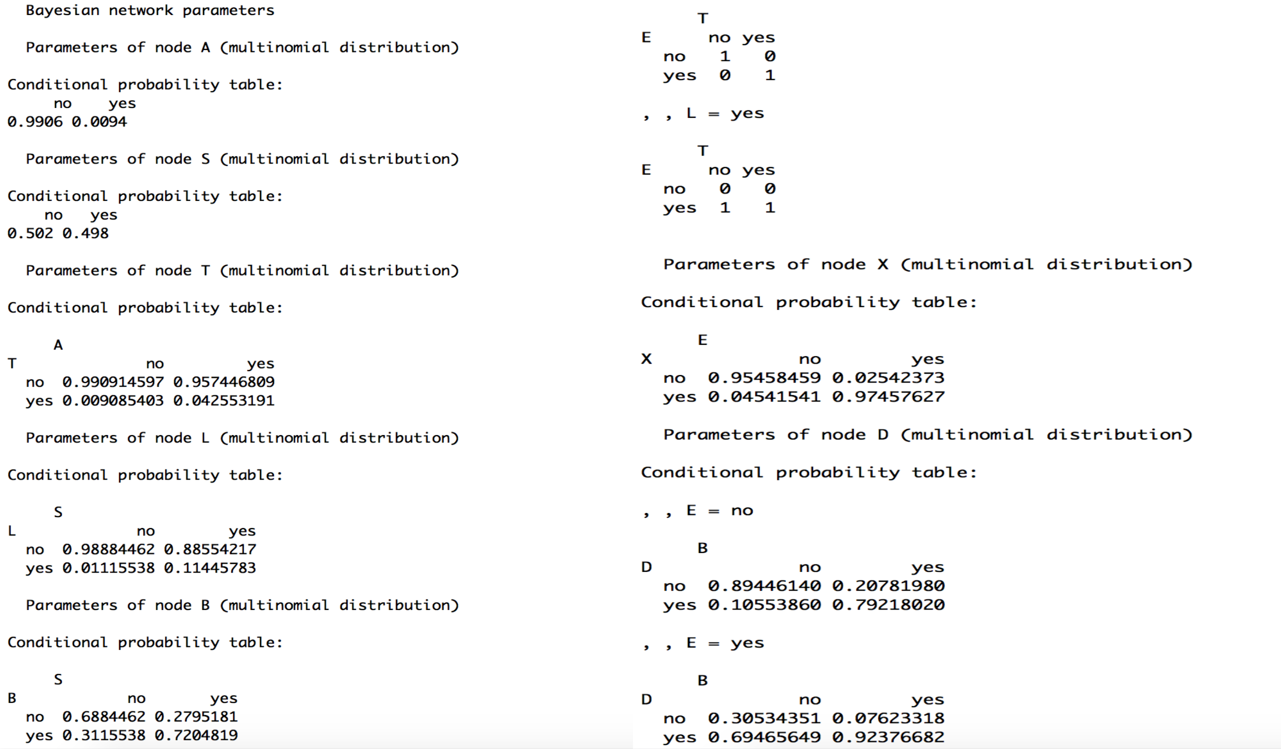


Figure 3: Hill Climb network structure after connecting node A

Brunda Chouthoy
CSC 529 – Bayesian Networks
Case Study 3

The network structure now represents a DAG and bn.fit function can be applied –



```
  Bayesian network parameters

  Parameters of node A (multinomial distribution)

Conditional probability table:
     no    yes 
0.9906 0.0094 

  Parameters of node S (multinomial distribution)

Conditional probability table:
    no   yes 
0.502 0.498 

  Parameters of node T (multinomial distribution)

Conditional probability table:

     A
T              no         yes
  no  0.990914597 0.957446809
  yes 0.009085403 0.042553191

  Parameters of node L (multinomial distribution)

Conditional probability table:

     S
L             no        yes
  no  0.98884462 0.88554217
  yes 0.01115538 0.11445783

  Parameters of node B (multinomial distribution)

Conditional probability table:

     S
B            no       yes
  no  0.6884462 0.2795181
  yes 0.3115538 0.7204819
```

```
      T
E      no yes
  no    1   0
  yes   0   1

, , L = yes

      T
E      no yes
  no    0   0
  yes   1   1


  Parameters of node X (multinomial distribution)

Conditional probability table:

      E
X             no        yes
  no  0.95458459 0.02542373
  yes 0.04541541 0.97457627

  Parameters of node D (multinomial distribution)

Conditional probability table:

, , E = no

      B
D             no        yes
  no  0.89446140 0.20781980
  yes 0.10553860 0.79218020

, , E = yes

      B
D             no        yes
  no  0.30534351 0.07623318
  yes 0.69465649 0.92376682
```

Table 9: Fitted model Conditional probability tables

The conditional probabilities for the score based model is shown in Table 9. The probability tables for individual attributes can also be visualized using a bar/dot plot.

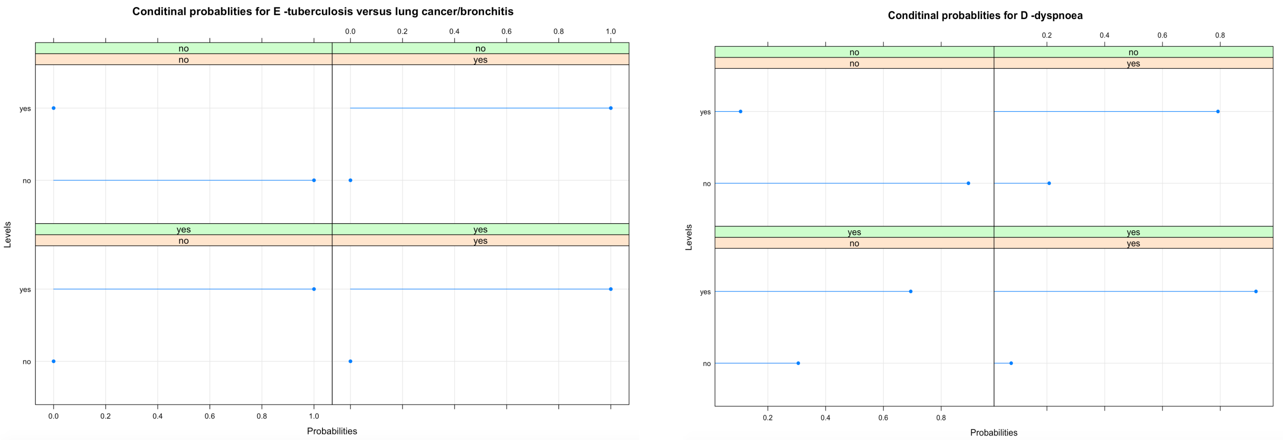


Figure 4: Conditional probability plots for the D and E nodes

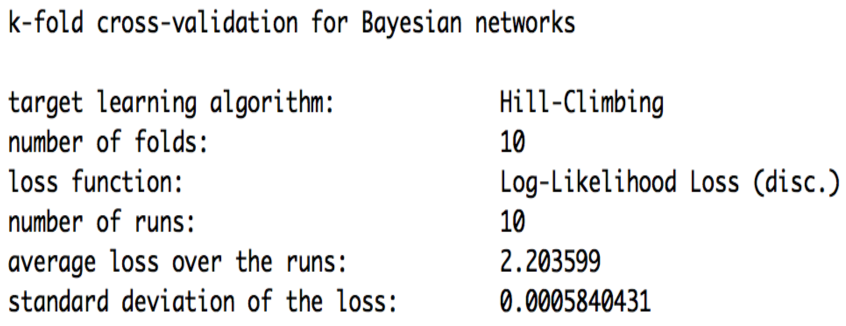
Figure 4 shows the conditional probability plots for nodes E (tuberculosis versus lung cancer/ bronchitis) and D (dyspnea) respectively. The default method for parameter estimation is the Maximum likelihood. Other methods such as Naïve- Bayes can also be used to generate conditional probability tables.

Experimental Analysis

3. Model Validation using k-fold Cross validation

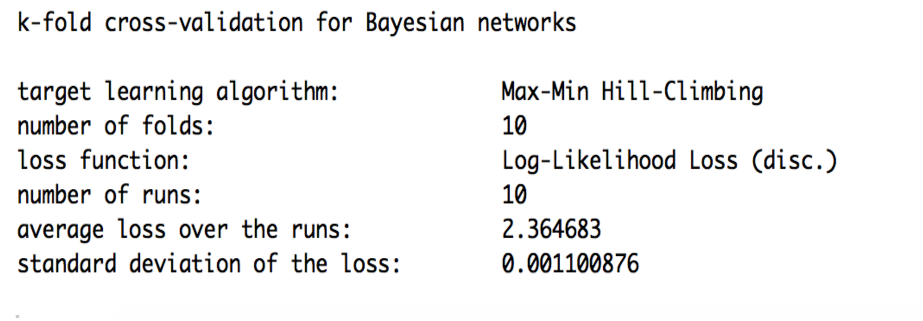
Once network structure and node conditional probability tables are determined, the next step is to validate the model, or assess model fit to the data. Cross-validation is a standard way to obtain unbiased estimates of a model's goodness of fit. By comparing such measures for different combinations of learning algorithms, fitting techniques and the respective parameters the optimal model can be chosen for the dataset.

With k-fold cross validation, the data are randomly partitioned into k subsets of equal size. Each subset is used in turn to validate the model fitted on the remaining k - 1 subsets. For the analysis, I will be cross-validating two learning algorithms - Max-Min Hill-Climb (mmhc) and Hill-Climb (hc). And the BDE scoring method is used, which requires an iss ('imaginery sample size' used for bde scores) term. The number of folds is specified as 10.



```
k-fold cross-validation for Bayesian networks

target learning algorithm:          Hill-Climbing
number of folds:                    10
loss function:                      Log-Likelihood Loss (disc.)
number of runs:                     10
average loss over the runs:         2.203599
standard deviation of the loss:     0.0005840431
```

Figure 5: 10-fold cross validation results for Hill Climb



```
 k-fold cross-validation for Bayesian networks

 target learning algorithm:            Max-Min Hill-Climbing
 number of folds:                      10
 loss function:                        Log-Likelihood Loss (disc.)
 number of runs:                       10
 average loss over the runs:           2.364683
 standard deviation of the loss:       0.001100876
```

Figure 6: 10-fold cross validation results with MMHC

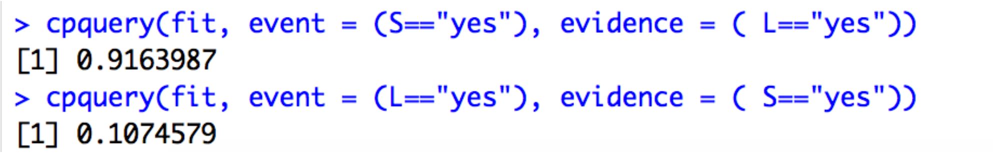
Based on the cross-validation results, it can be determined that the Hill-Climb algorithm produces a model/network structure that fits the data better - because its loss is approximately 2.203 compared to the Max-Min Hill-Climb loss at 2.364.

4. Inferences

Finally, once the network structure and parameter estimates are established, inferences can be made from the network using the 'cpquery' method in the bnlearn package. An advantage of Bayesian networks is that inferences can be Omni-directional, from the beginning to end, end to beginning, or middle to end or beginning of the process/system.

Example:

Table-10 shows the relationship between Lung cancer and smoking. It can be inferred that the Probability is about 91% that Smoking is 'Yes' when a person has Lung cancer and there's a 10.7% chance that the person has Lung cancer if smoking is 'yes'.



```
> cpquery(fit, event = (S=="yes"), evidence = ( L=="yes"))
[1] 0.9163987
> cpquery(fit, event = (L=="yes"), evidence = ( S=="yes"))
[1] 0.1074579
```

Table 10: Relation between smoking and Lung Cancer

Network score using AIC, BIC and BDE for the full network ('best' chosen network) is depicted in Table 11.

Type	Score
AIC	-11102.92
BIC	-11191.96

BDE	-11161.57

Conclusion:

- The main aim of this case study is to test and possibly expand my knowledge about learning Bayesian networks from data, by exploring various issues such as comparison of Structure learning algorithms and evaluation of the networks learnt.
- The lung cancer – Asia data was used from the bnlearn repository and is about lung diseases (tuberculosis, lung cancer or bronchitis) and visits to Asia.
- Four steps were followed for the Bayesian network analysis - Network structure learning, Parameter learning, Model validation and estimating network outcomes.
- It was determined that the score based learning algorithm – Hill climbing provides the best network structure to be analyzed.
- K-fold cross validation technique was used to calculate the loss function and the Hill-Climb algorithm produced a model/network structure that fits the data better - because its loss is approximately 2.203 compared to the Max-Min Hill-Climb loss at 2.364.
- Finally, Conditional probability tables generated by fitting the model to the data was used to make inferences on the data.
- With this study, I could estimate the unknown node relationships and entire network structure using the Bayesian networks.