732A96/TDDE15 ADVANCED MACHINE LEARNING

LAB 1: GRAPHICAL MODELS

JOSE M. PEÑA
IDA, LINKÖPING UNIVERSITY, SWEDEN

1. Instructions

- **Deadline for individual and group reports**
  See LISAM.
- **What and how to hand in**
  Each student must send a report to LISAM with his/her solutions to the lab. The file should be named FirstName_LastName.pdf. The report must be concise but complete. It should include (i) the code implemented or the calls made to existing functions, (ii) the results of such code or calls, and (iii) explanations for (i) and (ii).
  In addition, students must discuss their lab solutions in a group. Each group must compile a collaborative report that will be used for presentation at the seminar. The report should clearly state the names of the students that participated in its compilation and a short description of how each student contributed to the report. This report should be submitted via LISAM. The group reports are corrected and graded. The individual reports are also checked, but feedback on them will not be given. A student passes the lab if the group report passes the seminar and the individual report has reasonable quality, otherwise the student must complete his/her individual report by correcting the mistakes in it.
  Attendance to the seminar is obligatory. In the seminar, some groups will be responsible for presenting their group reports. Each student in these groups must be prepared to individually present an arbitrary part of the report. The selection of the speakers is done randomly during the seminar. In the seminar, some groups will act as opponents to the reports provided by the presenters. The opponent group should examine the group report of the presenter group before the seminar, and prepare a minimum of three questions, comments and/or improvements. The opponent group will ask these questions during the seminar. Check LISAM for the list of presenter and opponent groups.
- **Resources**
  The lab is designed to be solved with the R packages `bnlearn` and `gRain`. You may also want to use the RStudio development environment. To install the `gRain` package in R 4.0.2, do the following:

```
if (!requireNamespace("BiocManager", quietly = TRUE))
install.packages("BiocManager")

BiocManager::install("RBGL")
BiocManager::install("Rgraphviz")
BiocManager::install("gRain")
```

  - Literature:
    * Package documentation.
    * Højsgaard, S. Graphical Independence Networks with the `gRain` Package for R. *Journal of Statistical Software* 46, 2012.

    * Scutari, S. Learning Bayesian Networks with the `bnlearn` R Package. *Journal of Statistical Software* 35, 2010.

- Hint: Spend the first hour exploring the site `www.bnlearn.com`. Try the code in `www.bnlearn.com/examples`.

2. QUESTIONS

The purpose of the lab is to put in practice some of the concepts covered in the lectures.

(1) Show that multiple runs of the hill-climbing algorithm can return non-equivalent Bayesian network (BN) structures. Explain why this happens. Use the Asia dataset which is included in the `bnlearn` package. To load the data, run `data("asia")`.

Hint: Check the function `hc` in the `bnlearn` package. Note that you can specify the initial structure, the number of random restarts, the score, and the equivalent sample size (a.k.a imaginary sample size) in the BDeu score. You may want to use these options to answer the question. You may also want to use the functions `plot`, `arcs`, `vstructs`, `cpdag` and `all.equal`.

(2) Learn a BN from 80 % of the Asia dataset. The dataset is included in the `bnlearn` package. To load the data, run `data("asia")`. Learn both the structure and the parameters. Use any learning algorithm and settings that you consider appropriate. Use the BN learned to classify the remaining 20 % of the Asia dataset in two classes: $S = yes$ and $S = no$. In other words, compute the posterior probability distribution of S for each case and classify it in the most likely class. To do so, you **have** to use exact or approximate inference with the help of the `bnlearn` and `gRain` packages, i.e. you are not allowed to use functions such as `predict`. Report the confusion matrix, i.e. true/false positives/negatives. Compare your results with those of the true Asia BN, which can be obtained by running
`dag = model2network("[A][S][T|A][L|S][B|S][D|B:E][E|T:L][X|E]")`.

Hint: You already know the Lauritzen-Spiegelhalter algorithm for inference in BNs, which is an exact algorithm. There are also approximate algorithms for when the exact ones are too demanding computationally. For exact inference, you may need the functions `bn.fit` and `as.grain` from the `bnlearn` package, and the functions `compile`, `setFinding` and `querygrain` from the package `gRain`. For approximate inference, you may need the functions `prop.table`, `table` and `cpdist` from the `bnlearn` package.

(3) In the previous exercise, you classified the variable S given observations for all the rest of the variables. Now, you are asked to classify S given observations only for the so-called Markov blanket of S, i.e. its parents plus its children plus the parents of its children minus S itself. Report again the confusion matrix.

Hint: You may want to use the function `mb` from the `bnlearn` package.

(4) Repeat the exercise (2) using a naive Bayes classifier, i.e. the predictive variables are independent given the class variable. See p. 380 in Bishop's book or Wikipedia for more information on the naive Bayes classifier. Model the naive Bayes classifier as a BN. You **have** to create the BN by hand, i.e. you are not allowed to use the function `naive.bayes` from the `bnlearn` package.

Hint: Check `http://www.bnlearn.com/examples/dag/` to see how to create a BN by hand.

(5) Explain why you obtain the same or different results in the exercises (2-4).