Advanced Machine Learning - Lab 1

Ali Etminan

9/13/2020

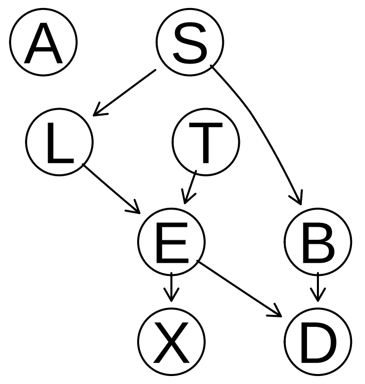
Question 1

As the initial step, we construct twoinitial graphs which will be used in a number of *Hill-Climbing* algorithm evaluations.

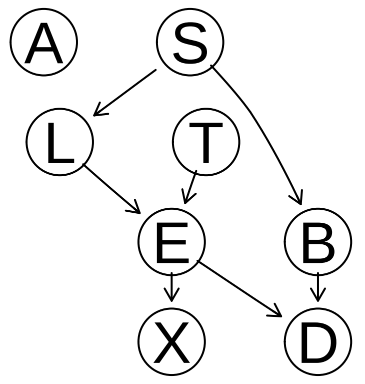
```r
data("asia")
init.graph <- empty.graph(nodes = c('A','S','T','L','B','E','X','D'))
init.arcs <- set.arc(init.graph, from = 'A', to = 'S')
init.arcs2 <- set.arc(init.graph, from = 'X', to = 'T')
```

Now we evaluate the Hill-Climbing algorithm on the Asia dataset with 6 different settings to see if the results are identical.

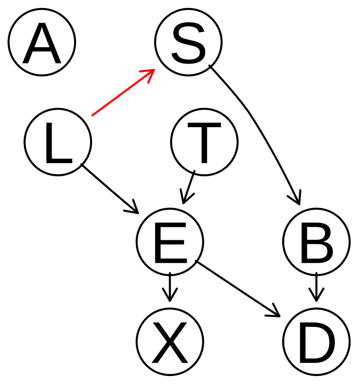
```r
asia.hc <- hc(asia, score = 'bic', start = init.arcs, restart = 10)
asia.hc2 <- hc(asia, score = 'bic', start = init.arcs2, restart = 10)
asia.hc3 <- hc(asia, score = 'bic', start = init.arcs2, restart = 50)
asia.hc4 <- hc(asia, score = 'bic', start = init.arcs2, restart = 50, perturb = 5)
asia.hc5 <- hc(asia, score = 'bic', restart = 10)
asia.hc6 <- hc(asia, score = 'bic', restart = 50, perturb = 5)
```



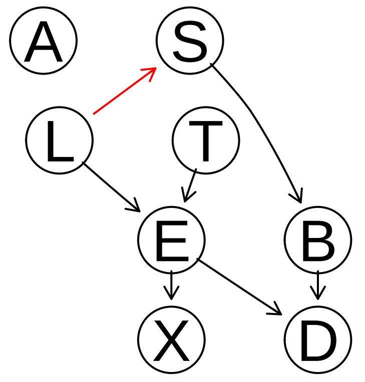
M1 − With init graph, from A to S, RR = 10



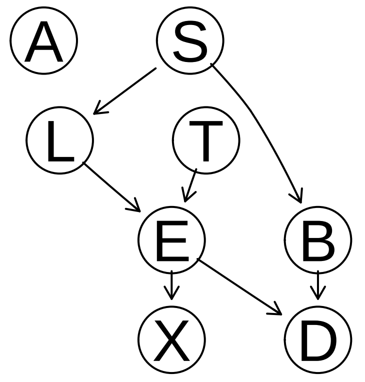
M2 − With init graph, from X to T, RR = 10



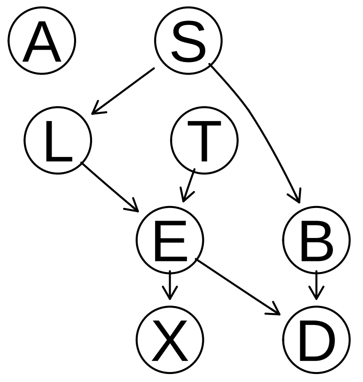
M3 − With init graph, from X to T, RR = 50



With init graph, from X to T, RR = 50, perturn = 5



M5 − Without init graph, RR = 10



M6 − Without init graph, RR = 50, perturb = 5

The resulting graphs indicate that the algorithm generates different results on different settings. This is due to the fact that Hill-Climbing performs heuristic search to find an optima and every time the algorithm is run, it picks a different location to search for the optima. Through an iterative process, it attempts to improve the score. However, since the algorithm performs local search, it tends to get stuck in a local optima.

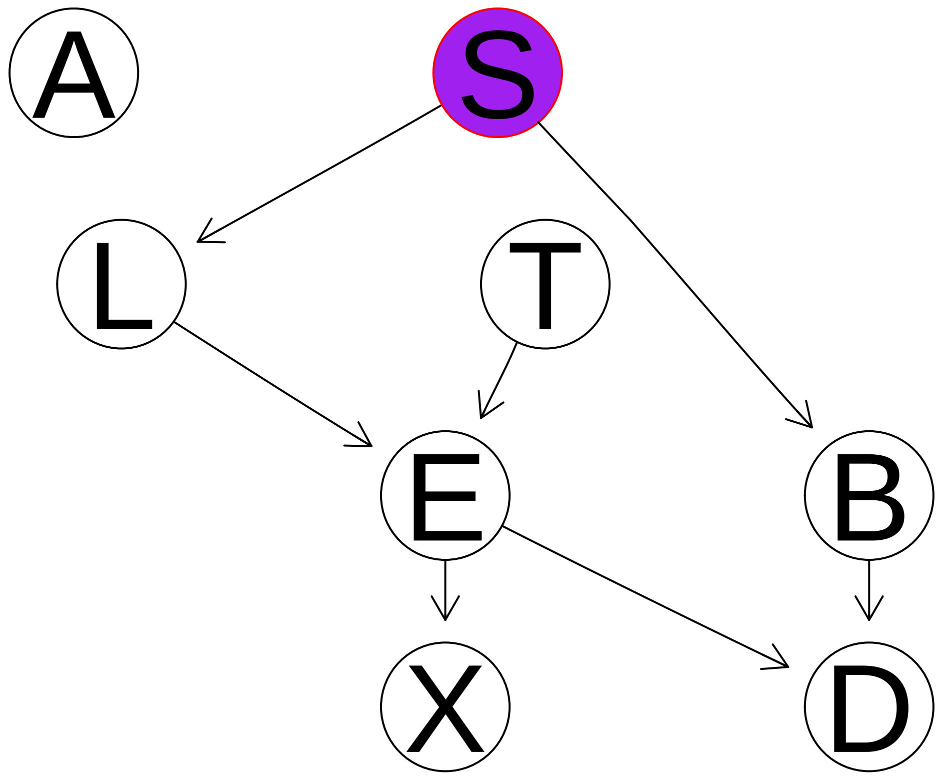
Hill-Climbing is a score-based algorithm. The direction of the edges do not the overall dependencies and results in a similar score. Therefore, the algorithm may pick an edge or the other if they yield a similar score. That is why we get different results on different runs. Below we can see that the score for all the models are similar but the models are not identical.

```
## BIC score for M1:  -11095.79

## BIC score for M2:  -11095.79

## BIC score for M3:  -11095.79

## BIC score for M4:  -11095.79

## BIC score for M5:  -11095.79

## BIC score for M6:  -11095.79

## [1] TRUE
```

Question 2

We divide the data randomly to train and validation sets with 80/20 percent proportions respectively. Then we fit the model using the HC algorithm on the train data and plot the fitted model.

Fitted model using HC algorithm

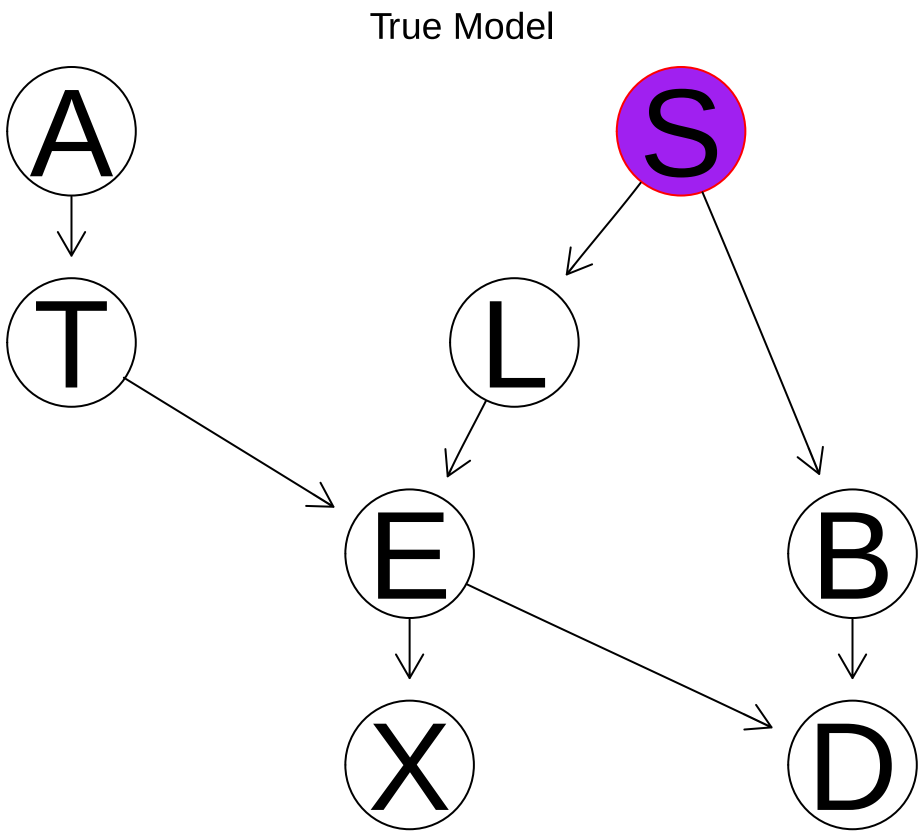


Next, we use our model and observations from features in the validation set to make predictions on the state of "S" given all the other features. The resulting confusion matrix and misclassification rate is reported below.

```
##      pred
##          no   yes
##   no  0.337 0.176
##   yes 0.121 0.366
```

```
## Misclassification Rate using model with HC algorithm:  0.297
```

Now, we fit a model using the true model for the Asia dataset instead of the one created by the HC algorithm.



The corresponding confusion matrix and misclassification rate are reported.

```
##      pred.true
##          no   yes
##   no  0.337 0.176
##   yes 0.121 0.366
```

```
## Misclassification Rate using the true model:  0.297
```

The Hill-Climbing and the true model generate the same result and accuracy.

Question 3

```
## Nodes return under the Markov Blanket:  L B
```

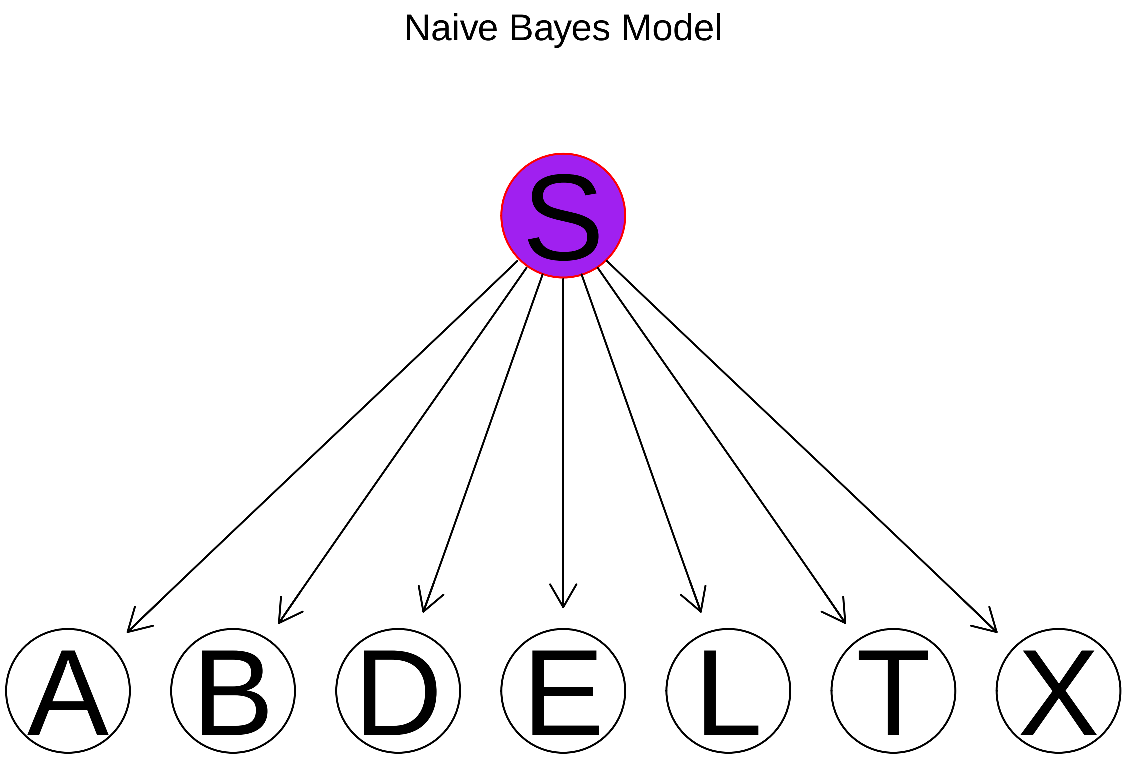
```
##      pred.mb
##          no   yes
##   no  0.337 0.176
##   yes 0.121 0.366
```

```
## Misclassification Rate using variables under Markov Blanket:  0.297
```

The Markov Blanket returns nodes “L” and “B” nodes given “S”. From the graph of the Hill-Climbing fitted model, it is observed that these three nodes construct a *fork* where “S” is a parent of the other two. In this sense “L” and “B” are independent given “S”. “S” is also independent from the rest of the network.

Question 4

Naive Bayes or Simple Bayes, assumes independence among all the predicting features. Therefore we define a model where all the features are independent from each other but are the child of the target node “S”.

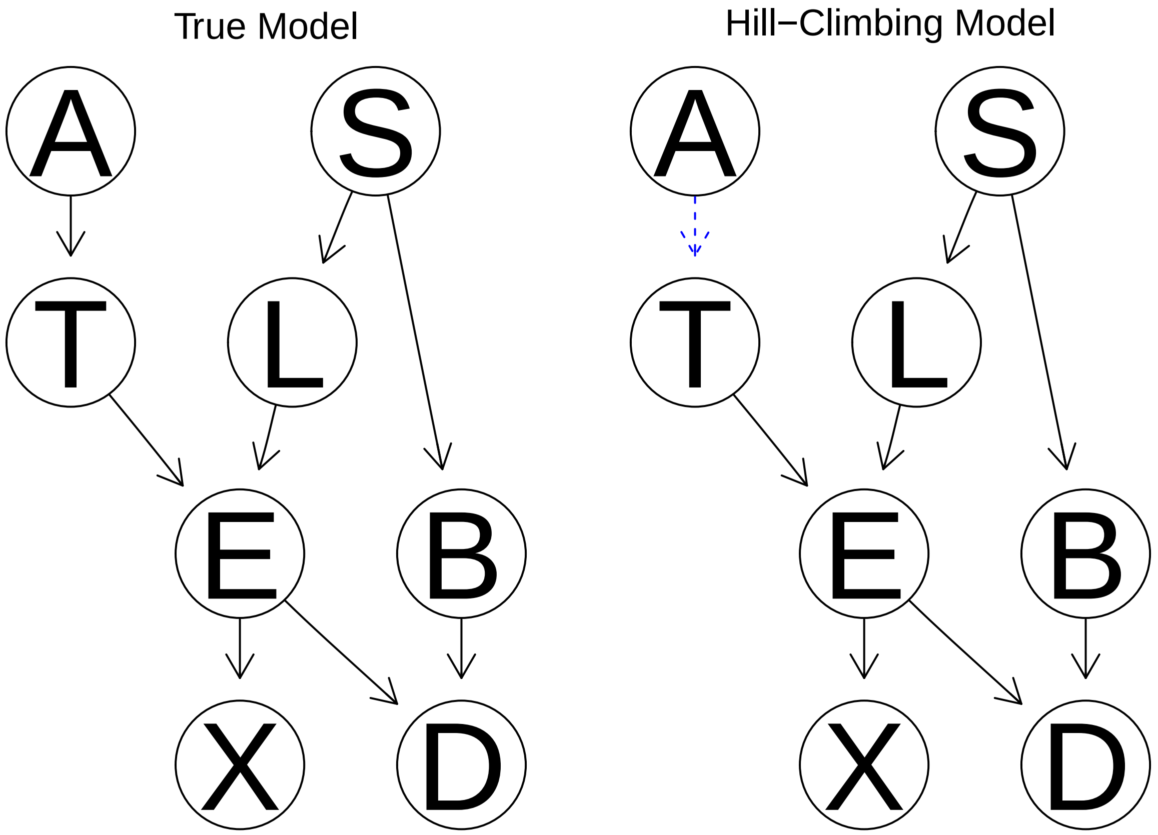


Here we use our Naive Bayes model to predict the state of “S”.

```
##      pred.mb
##          no   yes
##   no  0.337 0.176
##   yes 0.121 0.366
```

```
## Misclassification Rate using Naive Bayes classifier:  0.334
```

Question 5



We obtain similar results using the HC fit model and the true model. The above plots suggest that as opposed to the trained model, there is an arc from "A" to "T" in the true model. However, this does not affect the resulting score of the models as the path from "A" to "S" would be blocked by "E". Therefore, we can justify the identical results obtained in part 2.

As for the model using only the nodes chosen by the Markov Blanket, we should consider that "S" is inferred only from "L" and "B" (explained in section 3), and therefore the resulting confusion matrix and misclassification rate are identical to results from section 2 since the rest of the nodes are independent from "S" and would not have any impact on its state.

Naive Bayes classifier removes all the conditional dependencies among the nodes. In other words, it removes critical information from the underlying structure that is essential to making inference about "S". Thus, resulting in a higher mislassification rate.

Appendix

```r
library(bnlearn)
library(gRain)
#library(RBGL)
knitr::opts_chunk$set(echo = TRUE)

data("asia")
init.graph <- empty.graph(nodes = c('A','S','T','L','B','E','X','D'))
init.arcs <- set.arc(init.graph, from = 'A', to = 'S')
init.arcs2 <- set.arc(init.graph, from = 'X', to = 'T')


asia.hc <- hc(asia, score = 'bic', start = init.arcs, restart = 10)
asia.hc2 <- hc(asia, score = 'bic', start = init.arcs2, restart = 10)
asia.hc3 <- hc(asia, score = 'bic', start = init.arcs2, restart = 50)
asia.hc4 <- hc(asia, score = 'bic', start = init.arcs2, restart = 50, perturb = 5)
asia.hc5 <- hc(asia, score = 'bic', restart = 10)
asia.hc6 <- hc(asia, score = 'bic', restart = 50, perturb = 5)

par(mfrow=c(2,3))
graphviz.compare(asia.hc,asia.hc2,asia.hc3,
                 asia.hc4,asia.hc5,asia.hc6,
                 sub = c("M1 - With init graph, from A to S, RR = 10 ",
                         "M2 - With init graph, from X to T, RR = 10 ",
                         "M3 - With init graph, from X to T, RR = 50 ",
                         "M4 - With init graph, from X to T, RR = 50, perturn = 5 ",
                         "M5 - Without init graph, RR = 10 ",
                         "M6 - Without init graph, RR = 50, perturb = 5 "))


score1 <- score(asia.hc, asia, type = "bde")
score2 <- score(asia.hc2, asia, type = "bde")
score3 <- score(asia.hc3, asia, type = "bde")
score4 <- score(asia.hc4, asia, type = "bde")
score5 <- score(asia.hc5, asia, type = "bde")
score6 <- score(asia.hc6, asia, type = "bde")

cat('BIC score for M1: ', score1, '\n')
cat('BIC score for M2: ', score2, '\n')
cat('BIC score for M3: ', score3, '\n')
cat('BIC score for M4: ', score4, '\n')
cat('BIC score for M5: ', score5, '\n')
cat('BIC score for M6: ', score6, '\n')

comp <- all.equal(asia.hc, asia.hc2, asia.hc3, asia.hc4, asia.hc5, asia.hc6)
comp


n=dim(asia)[1]
set.seed(12345)
id=sample(1:n, floor(n*0.80))
train=asia[id,]
valid=asia[-id,]

hc.model <- hc(train)
hc.fit <- bn.fit(hc.model, train)
graphviz.plot(hc.fit, highlight = list(nodes = "S", fill = "purple"),
              main = "Fitted model using HC algorithm")


grain.fit.hc <- as.grain(hc.fit)
```

```r
new.valid <- valid[,-2]
valid.names <- colnames(new.valid)
pred <- c()

for (i in 1:nrow(new.valid)) {
  current.row <- sapply(new.valid[i,], toString)
  evid.hc <- setEvidence(grain.fit.hc, nodes = valid.names, states = current.row)
  current.pred <- querygrain(evid.hc, nodes = "S", type = "marginal")
  pred[i] <- ifelse(current.pred$S[1] > current.pred$S[2], "no", "yes")
}

confmat <- prop.table(table(valid$S, pred))
misclass <- 1 - sum(diag(confmat))/sum(confmat)
confmat

cat('Misclassification Rate using model with HC algorithm: ', misclass, "\n")


true.asia <- model2network("[A][S][T|A][L|S][B|S][D|B:E][E|T:L][X|E]")
graphviz.plot(true.asia, highlight = list(nodes = "S", fill = "purple"),
              main = "True Model")


true.fit <- bn.fit(true.asia, train)
grain.true <- as.grain(true.fit)

pred.true <- c()

for (i in 1:nrow(new.valid)) {
  current.row <- sapply(new.valid[i,], toString)
  evid.hc <- setEvidence(grain.true, nodes = valid.names, states = current.row)
  current.pred <- querygrain(evid.hc, nodes = "S", type = "marginal")
  pred.true[i] <- ifelse(current.pred$S[1] > current.pred$S[2], "no", "yes")
}


confmat.true <- prop.table(table(valid$S, pred.true))
misclass.true <- 1 - sum(diag(confmat.true))/sum(confmat.true)
confmat.true
cat('Misclassification Rate using the true model: ', misclass.true, "\n")


mb.nodes <- mb(hc.fit, node = "S")
cat("Nodes return under the Markov Blanket: ", mb.nodes)


mb.valid <- valid[,c(mb.nodes[1],mb.nodes[2])]
mb.valid.names <- colnames(mb.valid)
pred.mb <- c()

for (i in 1:nrow(mb.valid)) {
  current.row <- sapply(mb.valid[i,], toString)
  evid.hc <- setEvidence(grain.fit.hc, nodes = mb.valid.names, states = current.row)
  current.pred <- querygrain(evid.hc, nodes = "S", type = "marginal")
  pred.mb[i] <- ifelse(current.pred$S[1] > current.pred$S[2], "no", "yes")
}

confmat.mb <- prop.table(table(valid$S, pred.mb))
misclass.mb <- 1 - sum(diag(confmat.mb))/sum(confmat.mb)


confmat.mb
cat('Misclassification Rate using variables under Markov Blanket: ', misclass.mb)



naiveBayes <- model2network("[A|S][T|S][L|S][B|S][E|S][X|S][D|S][S]")

graphviz.plot(naiveBayes, highlight = list(nodes = "S", fill = "purple"),
              main = "Naive Bayes Model")



naivemodel <- bn.fit(naiveBayes, train)
naivegrain <- as.grain(naivemodel)

valid.nb <- valid[,-2]
column.names <- colnames(valid.nb)
pred.nb <- c()


for (i in 1:nrow(valid.nb)) {
  current.row <- sapply(valid.nb[i,], toString)
  evid.nb <- setEvidence(naivegrain, nodes = column.names, states = current.row)
  current.pred <- querygrain(evid.nb, nodes = "S", type = "marginal")
  pred.nb[i] <- ifelse(current.pred$S[1] > current.pred$S[2], "no", "yes")
}

confmat.nb <- prop.table(table(valid$S, pred.nb))

misclass.nb <- 1 - sum(diag(confmat.nb))/sum(confmat.nb)
confmat.mb
cat('Misclassification Rate using Naive Bayes classifier: ', misclass.nb)



par(mfrow = c(1,2))
graphviz.compare(true.asia, hc.model,
                 main = c("True Model",
                          "Hill-Climbing Model"))
```