MM2-eksamen

Gruppe 1.211

28/10/2020

We have the dataset ChestSim1000

```
data(chestSim1000, package="gRbase")
head(chestSim1000) # our data
```

```
##   asia tub smoke lung bronc either xray dysp
## 1   no  no    no   no   yes     no   no  yes
## 2   no  no   yes   no   yes     no   no  yes
## 3   no  no   yes   no    no     no   no   no
## 4   no  no    no   no    no     no   no   no
## 5   no  no   yes   no   yes     no   no  yes
## 6   no  no   yes  yes   yes    yes  yes  yes
```

```
length(chestSim1000[,1]) # our data consists of 1000 observations
```

```
## [1] 1000
```

This is a hyphotetical Chest Clinic problem, by Lauritzen and Spiegelhalter. (ref til https://arxiv.org/pdf/1301.7394.pdf)

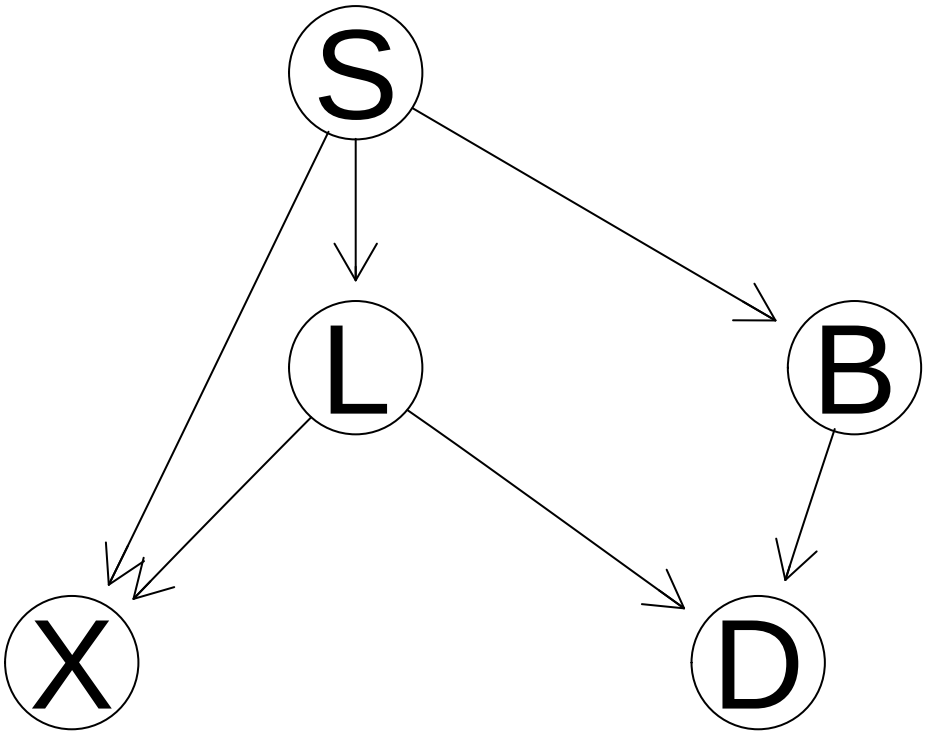
Here is a short explanation of the variables in the dataset.

- asia → subject has visited asia
- tub → subject has tuberculosis
- smoke → subject is a smoker
- lung → subject has lung cancer
- bronc → subject has bronchitis
- either → subject has either tuberculosis or lungcancer
- xray → subject has positive X-ray
- dysp → Subject has dyspnoea

Shortness-of-breath (dyspnoea) may be due to tuberculosis, lung cancer, bronchitis, 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 does neither the presence nor absence of dyspnoea. (citat direkte sat ind fra https://arxiv.org/pdf/1301.7394.pdf)

Now consider the following excerpt from the chest clinic examples:

```
dg1 <- dag(~ S + L|S + X|L:S + B|S + D|L:B)
plot(dg1) # example of a bayesian network
```



Exercise 1

Extract the necessary CPT's from data, and construct the Bayesian network.

Answer

Here let $V = \{Asia, Tub, Smoke, Lung, Either, Bronc, Xray, Dysp\} = \{a, t, s, l, e, x, d\}$ denote the total number of nodes. Each nodes represent a binary level $\{yes, no\}$.

For each node v and its parents $pa(v)$ there is a conditional distribution $P(v \mid pa(v))$. For more then one parent node, for example two, we write $P(v_1 \mid v_2, v_3)$

By inspection of the graph above, we observe that we need $P(S)$, $P(B|S)$, $P(D|B, L)$, $P(L|S)$ and $P(X|L, S)$ for the joint distribution.

```
n.cis <- xtabs(~smoke + lung + xray + bronc + dysp, data = chestSim1000) # An overview of the data
as.data.frame(n.cis) #Shows frequency of each combination
```

```
##    smoke lung xray bronc dysp Freq
## 1    yes  yes  yes   yes  yes   27
## 2     no  yes  yes   yes  yes    1
## 3    yes   no  yes   yes  yes    9
## 4     no   no  yes   yes  yes    3
## 5    yes  yes   no   yes  yes    0
## 6     no  yes   no   yes  yes    0
## 7    yes   no   no   yes  yes  191
## 8     no   no   no   yes  yes  129
## 9    yes  yes  yes    no  yes   13
## 10    no  yes  yes    no  yes    6
## 11   yes   no  yes    no  yes    2
## 12    no   no  yes    no  yes    3
```

```
## 13   yes  yes   no    no  yes    0
## 14    no  yes   no    no  yes    0
## 15   yes   no   no    no  yes    8
## 16    no   no   no    no  yes   36
## 17   yes  yes  yes   yes   no    3
## 18    no  yes  yes   yes   no    0
## 19   yes   no  yes   yes   no    4
## 20    no   no  yes   yes   no    0
## 21   yes  yes   no   yes   no    0
## 22    no  yes   no   yes   no    0
## 23   yes   no   no   yes   no   42
## 24    no   no   no   yes   no   27
## 25   yes  yes  yes    no   no    3
## 26    no  yes  yes    no   no    0
## 27   yes   no  yes    no   no    6
## 28    no   no  yes    no   no   15
## 29   yes  yes   no    no   no    0
## 30    no  yes   no    no   no    0
## 31   yes   no   no    no   no  157
## 32    no   no   no    no   no  315
```

Now the CPTs for the probabilities

```
p.s <- tabDist(n.cis, marg = ~ smoke);p.s #P(S)
```

```
## smoke
##   yes    no
## 0.465 0.535
```

```
cpt.bs <- tabDist(n.cis, marg = ~ bronc, cond = ~ smoke) #P(B|S)
cpt.bs %>% ftable(row.vars = "bronc")
```

```
##       smoke       yes        no
## bronc
## yes         0.5935484 0.2990654
## no          0.4064516 0.7009346
```

```
cpt.dbl <- tabDist(n.cis, marg = ~ dysp, cond = ~ bronc:lung) #P(D|B,L)
cpt.dbl %>% ftable(row.vars = "dysp")
```

```
##      bronc        yes                   no
##      lung         yes         no        yes         no
## dysp
## yes        0.90322581 0.81975309 0.86363636 0.09040590
## no         0.09677419 0.18024691 0.13636364 0.90959410
```

```
cpt.ls <- tabDist(n.cis, marg = ~ lung, cond = ~ smoke) #P(L|S)
cpt.ls %>% ftable(row.vars = "lung")
```

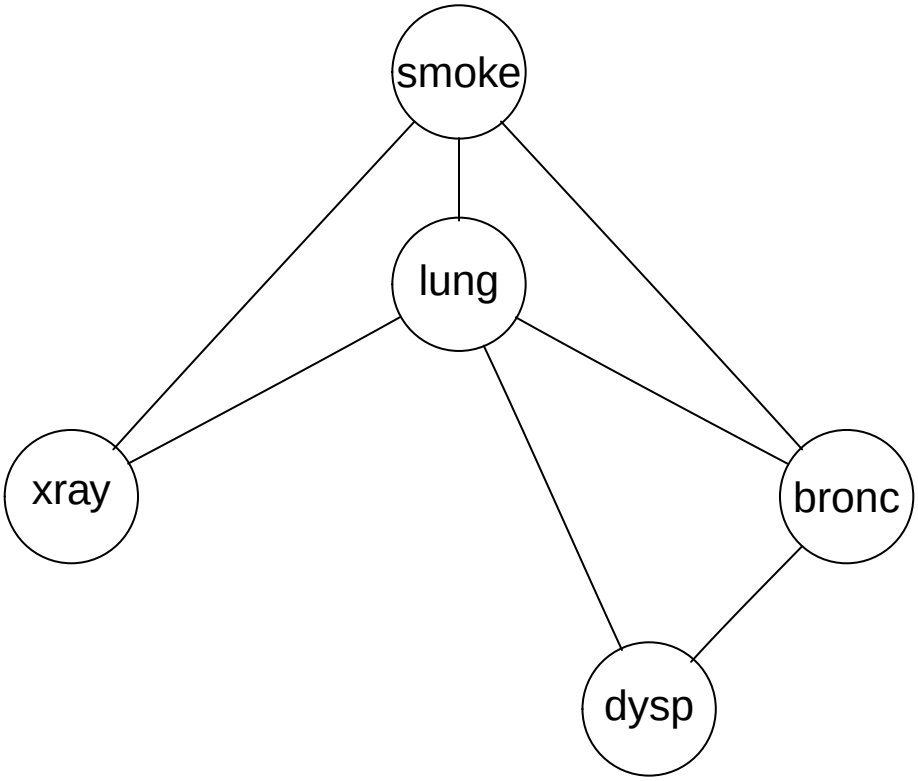
```
##      smoke        yes         no
## lung
## yes        0.09892473 0.01308411
## no         0.90107527 0.98691589
```

```
cpt.xls <- tabDist(n.cis, marg = ~ xray, cond = ~ lung:smoke) #P(X|L,S)
cpt.xls %>% ftable(row.vars = "xray")
```

```
##      lung         yes                   no
```

```
##      smoke        yes         no        yes         no
## xray
## yes        1.00000000 1.00000000 0.05011933 0.03977273
## no         0.00000000 0.00000000 0.94988067 0.96022727
```

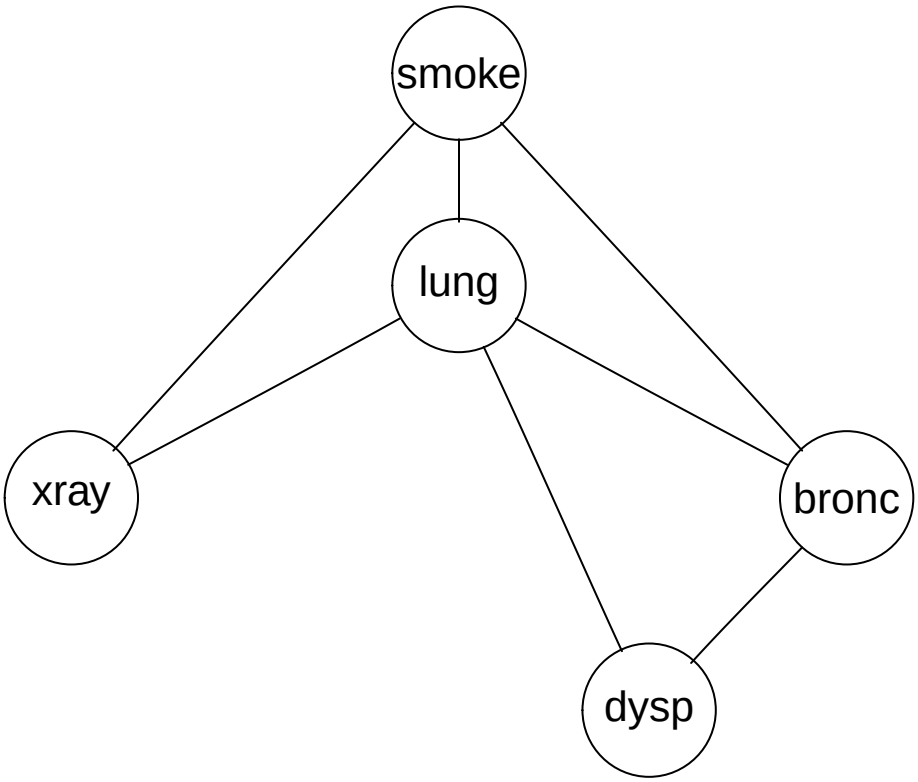
```
cptlist <- compileCPT(list(p.s, cpt.ls, cpt.xls, cpt.bs, cpt.dbl))
grn1 <- grain(cptlist)
plot(grn1)
```




```
grn1c <- propagate(grn1)
summary(grn1c)
```

```
## Independence network: Compiled: TRUE Propagated: TRUE
##  Nodes : chr [1:5] "smoke" "lung" "xray" "bronc" "dysp"
##  Number of cliques:                  3
##  Maximal clique size:                3
##  Maximal state space in cliques:     8
```

```
plot(grn1c)
```




```
library(gRain) #DET HER SKAL NOK SLETTES!
asia1 <- xtabs(~asia, chestSim1000); asia1 #putting observations together
```

```
## asia
## yes  no
##  14 986
```

```
tub.asia1 <- xtabs(~tub+asia, chestSim1000); tub.asia1 # counting tub given asia
```

```
##      asia
## tub   yes  no
##   yes   0   7
##   no   14 979
```

```
smoke1 <- xtabs(~smoke, chestSim1000); smoke1
```

```
## smoke
## yes  no
## 465 535
```

```
lung.smoke1 <- xtabs(~lung+smoke, chestSim1000); lung.smoke1
```

```
##      smoke
## lung  yes  no
##   yes  46   7
##   no  419 528
```

```
bronc.smoke1 <- xtabs(~bronc+smoke, chestSim1000); bronc.smoke1
```

```
##      smoke
## bronc yes  no
##   yes 276 160
```

```
##   no  189 375
```

```
either.lung.tub1 <- xtabs(~either+lung+tub, chestSim1000); either.lung.tub1
```

```
## , , tub = yes
##
##       lung
## either yes  no
##    yes   0   7
##    no    0   0
##
## , , tub = no
##
##       lung
## either yes  no
##    yes  53   0
##    no    0 940
```

```
xray.either1 <- xtabs(~xray+either, chestSim1000); xray.either1
```

```
##      either
## xray  yes  no
##   yes  60  35
##   no    0 905
```

```
dysp.bronc.either1 <- xtabs(~dysp+bronc+either, chestSim1000); dysp.bronc.either1
```

```
## , , either = yes
##
##      bronc
## dysp  yes  no
##   yes  29  21
##   no    4   6
##
## , , either = no
##
##      bronc
## dysp  yes  no
##   yes 331  47
##   no   72 490
```

```
# Constructing the conditional probability tables
asia1 <-  as.parray(asia1, normalize="first"); asia1
```

```
## asia
##   yes    no
## 0.014 0.986
```

```
tub.asia1 <-  as.parray(tub.asia1, normalize="first"); tub.asia1
```

```
##      asia
## tub           yes          no
##   yes 0.000000000 0.007099391
##   no  1.000000000 0.992900609
```

```
smoke1 <-  as.parray(smoke1, normalize="first"); smoke1
```

```
## smoke
```

```
##   yes    no
## 0.465 0.535
```

```
lung.smoke1 <-  as.parray(lung.smoke1, normalize="first"); lung.smoke1
```

```
##      smoke
## lung         yes         no
##   yes 0.09892473 0.01308411
##   no  0.90107527 0.98691589
```

```
bronc.smoke1 <-  as.parray(bronc.smoke1, normalize="first"); bronc.smoke1
```

```
##      smoke
## bronc       yes        no
##   yes 0.5935484 0.2990654
##   no  0.4064516 0.7009346
```

```
either.lung.tub1 <-  as.parray(either.lung.tub1, normalize="first"); either.lung.tub1
```

```
## , , tub = yes
##
##       lung
## either yes  no
##    yes       1
##    no        0
##
## , , tub = no
##
##       lung
## either yes  no
##    yes   1   0
##    no    0   1
```

```
  ftable(either.lung.tub1, row.vars = 1) #ftable helps us read the CPT when we have more the two variab
```

```
##        lung yes      no
##        tub  yes  no yes  no
## either
## yes         NaN   1   1   0
## no          NaN   0   0   1
```

```
xray.either1 <-  as.parray(xray.either1, normalize="first"); xray.either1
```

```
##      either
## xray         yes         no
##   yes 1.00000000 0.03723404
##   no  0.00000000 0.96276596
```

```
dysp.bronc.either1 <-  as.parray(dysp.bronc.either1, normalize="first"); dysp.bronc.either1
```

```
## , , either = yes
##
##      bronc
## dysp         yes         no
##   yes 0.87878788 0.77777778
##   no  0.12121212 0.22222222
##
## , , either = no
```

```
##
##      bronc
## dysp         yes         no
##   yes 0.82133995 0.08752328
##   no  0.17866005 0.91247672
```

```
  ftable(dysp.bronc.either1, row.vars = 1)
```

```
##      bronc         yes                    no
##      either        yes         no        yes         no
## dysp
## yes         0.87878788 0.82133995 0.77777778 0.08752328
## no          0.12121212 0.17866005 0.22222222 0.91247672
```

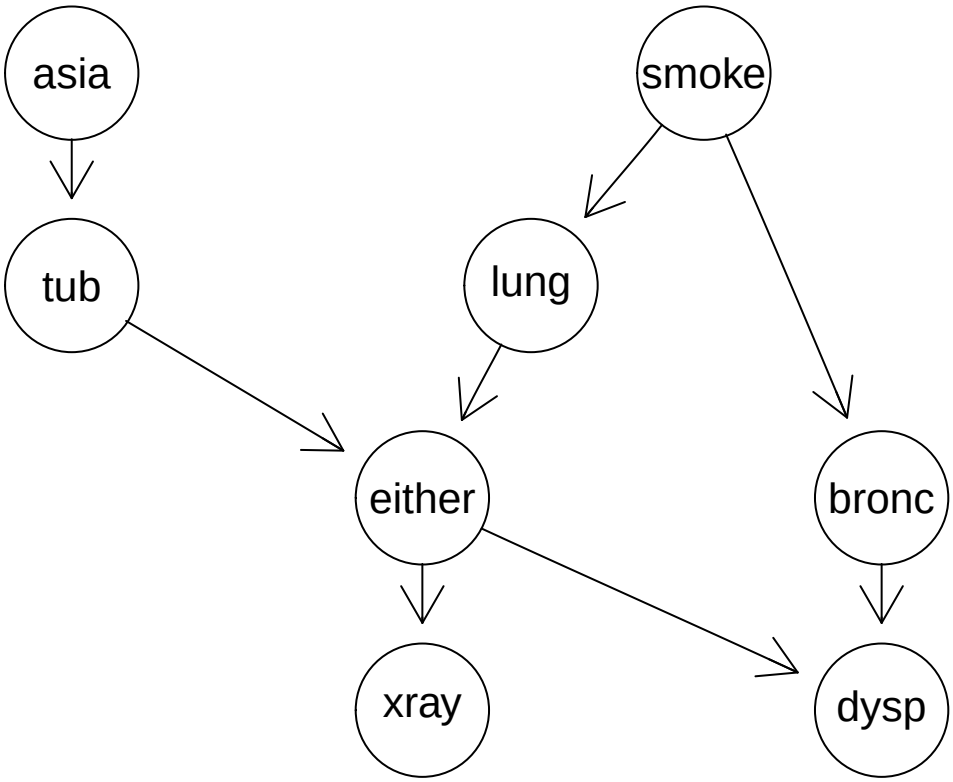
```
CPT.list1 <- compileCPT(list(asia1,tub.asia1,smoke1,lung.smoke1,bronc.smoke1,either.lung.tub1,xray.eith
CPT.list1 # overview of all CPT's
```

```
## cpt_spec with probabilities:
##  P( asia )
##  P( tub | asia )
##  P( smoke )
##  P( lung | smoke )
##  P( bronc | smoke )
##  P( either | lung tub )
##  P( xray | either )
##  P( dysp | bronc either )
```

```
# construct the Bayesian network
plot1=compile(grain(CPT.list1));plot1
```

```
## Independence network: Compiled: TRUE Propagated: FALSE
##   Nodes: chr [1:8] "asia" "tub" "smoke" "lung" "bronc" "either" "xray" "dysp"
```

```
plot(plot1$dag) # the Directed Acyclic Graph
```



The DAG shows the factorization of the joint probability function given by

$$P(V) = P(a)P(t|a)P(s)P(l|s)P(b|s)P(e|t,l)P(d|e,b)P(x|e)$$

More generally written we have that a DAG with V nodes allows us to construct a joint distribution by combining univariante condiional distribution by

$$P(V) = \prod_v P(v \mid pa(v))$$

The curse of dimensionality

We see here that for $P(V)$ representet by a table would be a table with $2^8 = 256$ entries

Exercise 2

What does information about "dysp" tell us about "smoke", i.e. what is the conditional distribution of "smoke" given "dysp"?

Answer

What we are looking for here is a simple conditional distribution of $p(v \mid pa(v))$ where v is a node and $pa(v)$ denotes the parents node

```
dysp1 <- xtabs(~dysp, chestSim1000); dysp1 #counting the number of observations for dysp
```

```
## dysp
## yes  no
## 428 572
```

```
smoke.dysp1 <- xtabs(~smoke+dysp, chestSim1000); smoke.dysp1  # counting smoke given dysp
```

```
##      dysp
## smoke yes  no
##   yes 250 215
##   no  178 357
```

```
# Constructing the conditional probability tables
dysp1 <-  as.parray(dysp1, normalize="first"); dysp1
```

```
## dysp
##   yes    no
## 0.428 0.572
```

```
smoke.dysp1 <-  as.parray(smoke.dysp1, normalize="first"); smoke.dysp1
```

```
##      dysp
## smoke       yes        no
##   yes 0.5841121 0.3758741
##   no  0.4158879 0.6241259
```

```
CPT.list2 <- compileCPT(list(dysp1,smoke.dysp1)) #creating our CPT list
CPT.list2 # overview of CPT's
```

```
## cpt_spec with probabilities:
##  P( dysp )
##  P( smoke | dysp )
```

Exercise 3

If we know “smoke”, what does additional infomration about “bronc” tell us about “lung”? That is, what is the conditional distribution of “lung” given smoke“, and what is the conditional distribution of”lung" given “smoke” and “bronc”?

Answer

The first distribution is the simple conditional distribution as we know it $p(v \mid pa(v))$ where as the second distribution is intreast is more challenging $p(v_1 \mid v_2, v_3)$ where $pa(v) = \{v_2, v_3\}$.

```
smoke1 <- xtabs(~smoke, chestSim1000); smoke1 # counting smoke
```

```
## smoke
## yes  no
## 465 535
```

```
bronc1 <- xtabs(~bronc, chestSim1000); bronc1 # counting bronc
```

```
## bronc
## yes  no
## 436 564
```

```
lung.smoke1 <- xtabs(~lung+smoke, chestSim1000); lung.smoke1 # counting lung given smoke
```

```
##      smoke
## lung  yes  no
##   yes  46   7
##   no  419 528
```

```
lung.smoke.bronc1 <- xtabs(~lung+smoke+bronc, chestSim1000); lung.smoke.bronc1 # counting smoke given s
```

```
## , , bronc = yes
##
##      smoke
## lung  yes  no
##   yes  30   1
##   no  246 159
##
## , , bronc = no
##
##      smoke
## lung  yes  no
##   yes  16   6
##   no  173 369
```

```
  ftable(lung.smoke.bronc1, row.vars = 1) #for CPT's of more then two variables
```

```
##      smoke yes      no
##      bronc yes  no yes  no
## lung
## yes         30  16   1   6
## no         246 173 159 369
```

```
# Constructing the conditional probability tables
smoke1 <-  as.parray(smoke1, normalize="first"); smoke1
```

```
## smoke
##   yes    no
## 0.465 0.535
```

```
bronc1 <-  as.parray(bronc1, normalize="first"); bronc1
```

```
## bronc
##   yes    no
## 0.436 0.564
```

```
lung.smoke1 <-  as.parray(lung.smoke1, normalize="first"); lung.smoke1
```

```
##      smoke
## lung         yes         no
##   yes 0.09892473 0.01308411
##   no  0.90107527 0.98691589
```

```
lung.smoke.bronc1 <-  as.parray(lung.smoke.bronc1, normalize="first"); lung.smoke.bronc1
```

```
## , , bronc = yes
##
##      smoke
## lung         yes         no
##   yes 0.10869565 0.00625000
##   no  0.89130435 0.99375000
##
## , , bronc = no
##
##      smoke
## lung         yes         no
```

```
##   yes 0.08465608 0.01600000
##   no  0.91534392 0.98400000
```

```
CPT.list3 <- compileCPT(list(smoke1,bronc1, lung.smoke.bronc1)) #creating our CPT's note lunge is inclu
CPT.list3 # overview
```

```
## cpt_spec with probabilities:
##  P( smoke )
##  P( bronc )
##  P( lung | smoke bronc )
```

Exercise 4

If we know "smoke" and "dysp", what does additional information about "bronc" tell us about "lung"?

Answer

Here we look at $P(v_1 \mid v_2, v_3)$ where $pa(v) = \{v_2, v_3\}$ for "smoke" and "dysp" and $P(v_1 \mid v_2, v_3, v_4)$ where $pa(v) = \{v_2, v_3, v_4\}$ for "smoke", "dysp" and "bronc".

```
smoke1 <- xtabs(~smoke, chestSim1000); smoke1 #counting smoke
```

```
## smoke
## yes  no
## 465 535
```

```
dysp1 <- xtabs(~dysp, chestSim1000); dysp1
```

```
## dysp
## yes  no
## 428 572
```

```
bronc1 <- xtabs(~bronc, chestSim1000); bronc1
```

```
## bronc
## yes  no
## 436 564
```

```
lung.smoke.dysp1 <- xtabs(~lung+smoke+dysp, chestSim1000); lung.smoke.dysp1
```

```
## , , dysp = yes
##
##      smoke
## lung  yes  no
##   yes  40   7
##   no  210 171
##
## , , dysp = no
##
##      smoke
## lung  yes  no
##   yes   6   0
##   no  209 357
```

```
lung.smoke.dysp.bronc1 <- xtabs(~lung+smoke+dysp+bronc, chestSim1000);lung.smoke.dysp.bronc1
```

```
## , , dysp = yes, bronc = yes
```

```
##
##      smoke
## lung  yes  no
##   yes  27   1
##   no  200 132
##
## , , dysp = no, bronc = yes
##
##      smoke
## lung  yes  no
##   yes   3   0
##   no   46  27
##
## , , dysp = yes, bronc = no
##
##      smoke
## lung  yes  no
##   yes  13   6
##   no   10  39
##
## , , dysp = no, bronc = no
##
##      smoke
## lung  yes  no
##   yes   3   0
##   no  163 330
```

```
# Constructing the conditional probability tables
smoke1 <-  as.parray(smoke1, normalize="first"); smoke1
```

```
## smoke
##   yes    no
## 0.465 0.535
```

```
dysp1 <-  as.parray(dysp1, normalize="first"); dysp1
```

```
## dysp
##   yes    no
## 0.428 0.572
```

```
bronc1 <-  as.parray(bronc1, normalize="first"); bronc1
```

```
## bronc
##   yes    no
## 0.436 0.564
```

```
lung.smoke.dysp1 <-  as.parray(lung.smoke.dysp1, normalize="first"); lung.smoke.dysp1
```

```
## , , dysp = yes
##
##      smoke
## lung         yes         no
##   yes 0.16000000 0.03932584
##   no  0.84000000 0.96067416
##
## , , dysp = no
##
```

```
##      smoke
## lung         yes         no
##   yes 0.02790698 0.00000000
##   no  0.97209302 1.00000000
```

```
  ftable(lung.smoke.dysp1, row.vars = 1) #pretty verison
```

```
##      smoke        yes                   no
##      dysp         yes         no        yes         no
## lung
## yes        0.16000000 0.02790698 0.03932584 0.00000000
## no         0.84000000 0.97209302 0.96067416 1.00000000
```

```
lung.smoke.dysp.bronc1 <-  as.parray(lung.smoke.dysp.bronc1, normalize="first"); lung.smoke.dysp.bronc1
```

```
## , , dysp = yes, bronc = yes
##
##      smoke
## lung          yes          no
##   yes 0.118942731 0.007518797
##   no  0.881057269 0.992481203
##
## , , dysp = no, bronc = yes
##
##      smoke
## lung          yes          no
##   yes 0.061224490 0.000000000
##   no  0.938775510 1.000000000
##
## , , dysp = yes, bronc = no
##
##      smoke
## lung          yes          no
##   yes 0.565217391 0.133333333
##   no  0.434782609 0.866666667
##
## , , dysp = no, bronc = no
##
##      smoke
## lung          yes          no
##   yes 0.018072289 0.000000000
##   no  0.981927711 1.000000000
```

```
  ftable(lung.smoke.dysp.bronc1, row.vars = 1) #pretty verison
```

```
##      smoke         yes                                           no
##      dysp          yes                     no                    yes                     no
##      bronc         yes          no         yes          no       yes          no         yes
## lung
## yes        0.118942731 0.565217391 0.061224490 0.018072289 0.007518797 0.133333333 0.000000000 0.000
## no         0.881057269 0.434782609 0.938775510 0.981927711 0.992481203 0.866666667 1.000000000 1.000
```

```
CPT.list4 <- compileCPT(list(smoke1,dysp1,bronc1,lung.smoke.dysp.bronc1)) #creating CPT's list again we
CPT.list4 # overview
```

```
## cpt_spec with probabilities:
##  P( smoke )
```

```
##  P( dysp )
##  P( bronc )
##  P( lung | smoke dysp bronc )
```

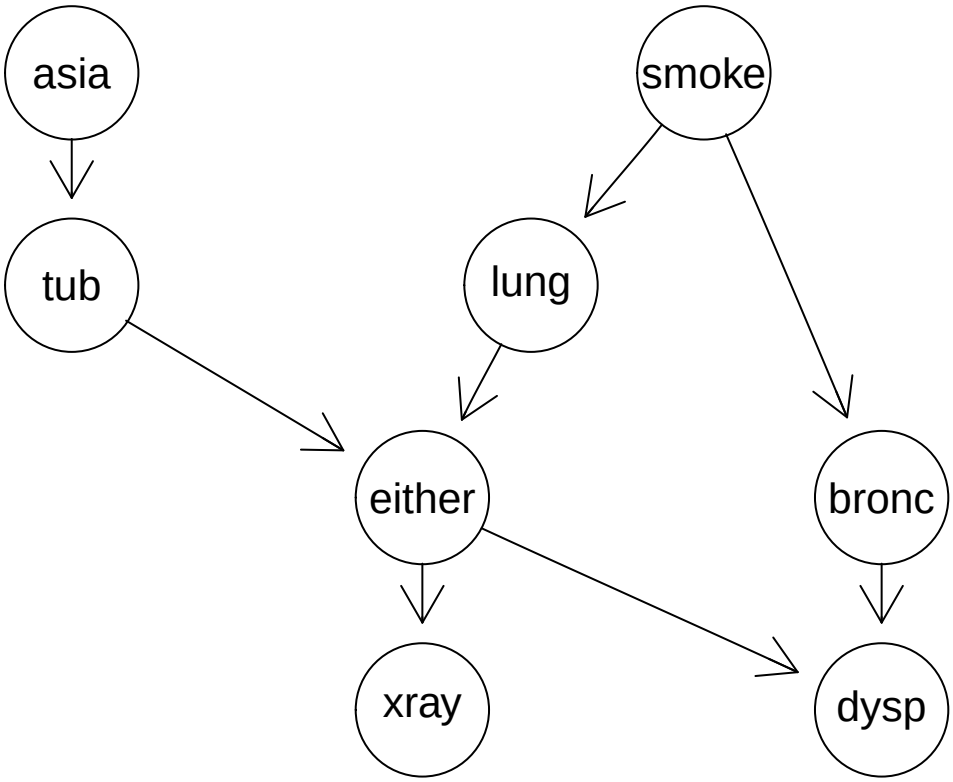
Exercise 5

Sketch the message passing algorithm for finding clique marginals for this specific example

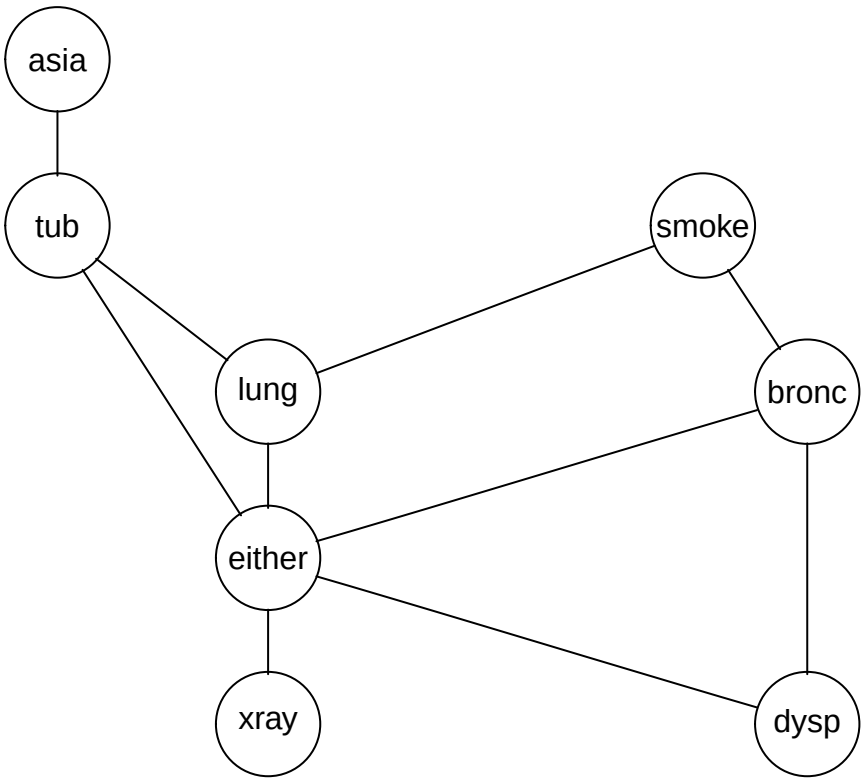
```
library(gRbase)
library(gRain)
```

Answer

```
# data from exercise 1
plot1=compile(grain(CPT.list1)) #turns CPT into a graphical independent network and the compiles them
plot(plot1$dag) # the plot containing the Directed Acyclic Graph
```




```
plot(moralize(plot1$dag)) # marrying parents and removing directions produces the Moral Graph
```



Note here that $P(V)$ has interactions only among neighbours of the undirected moral graph. To understand this let $q(v_2, v_1)$ denote a interaction function from point v_1 to v_2 without direction so for our data i exercise 1 we have

$$P(V) = P(a)p(t|a)P(s)P(l|s)P(b|s)P(e|t,l)P(d|e,b)P(x|e) = q(a)q(t,a)q(s)q(l,s)q(b,s)q(e,t,l)q(d,e,b)q(x,e)$$

Then merging the q-functions that are contained in large q-functions we get

$$P(V) = q(t,a)q(l,s)q(b,s)q(e,t,l)q(d,e,b)q(x,e)$$

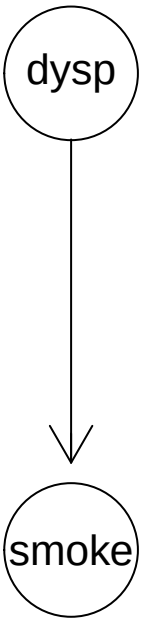
These are then the clique marginals in the sense that $P(l,s) = q(l,s)$ and so forth, these clique marginls can be extracted directly

```
lung.smoke1 <- xtabs(~lung+smoke, chestSim1000); lung.smoke1
```

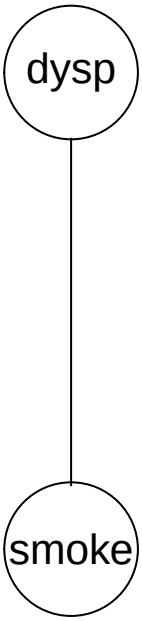
```
##      smoke
## lung  yes  no
##   yes  46   7
##   no  419 528
```

The q-function $q(e,t,l)$ is what creats the new interaction between t and l.

```
# data from exercise 2
plot2=compile(grain(CPT.list2))
plot(plot2$dag)
```




```
plot(moralize(plot2$dag))
```



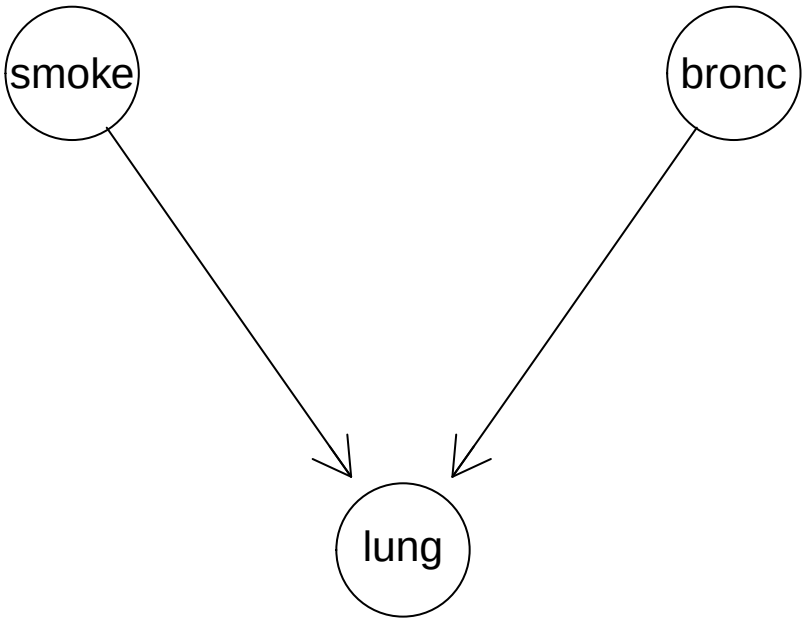
Here we have

$$P(V) = P(d)P(s \mid d) = q(d)q(s,d)$$

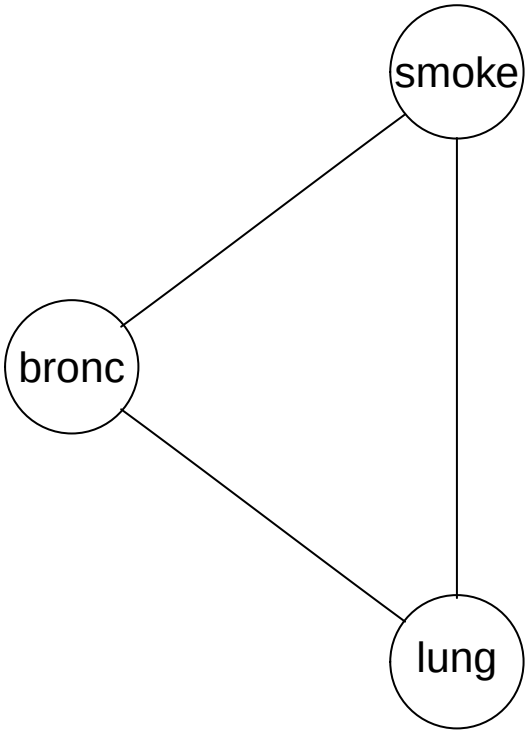
then merging gives

$$P(V) = q(s,d)$$

```
# data from exercise 3
plot3=compile(grain(CPT.list3))
plot(plot3$dag)
```




```
plot(moralize(plot3$dag))
```



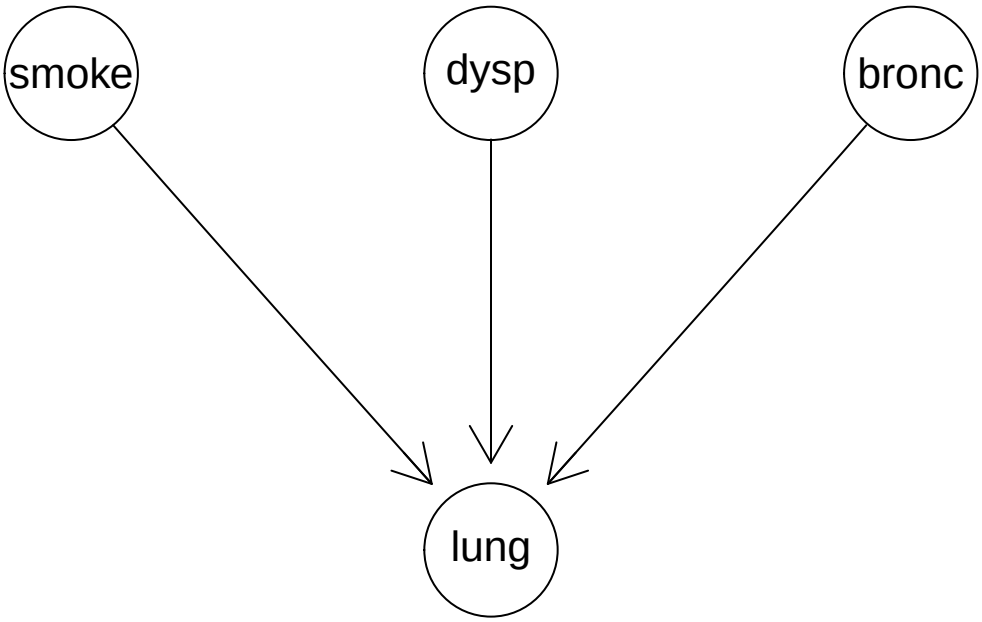
Here we have

$$P(V) = P(s)P(b)P(l \mid s, b) = q(s)q(b)q(l, s, b)$$

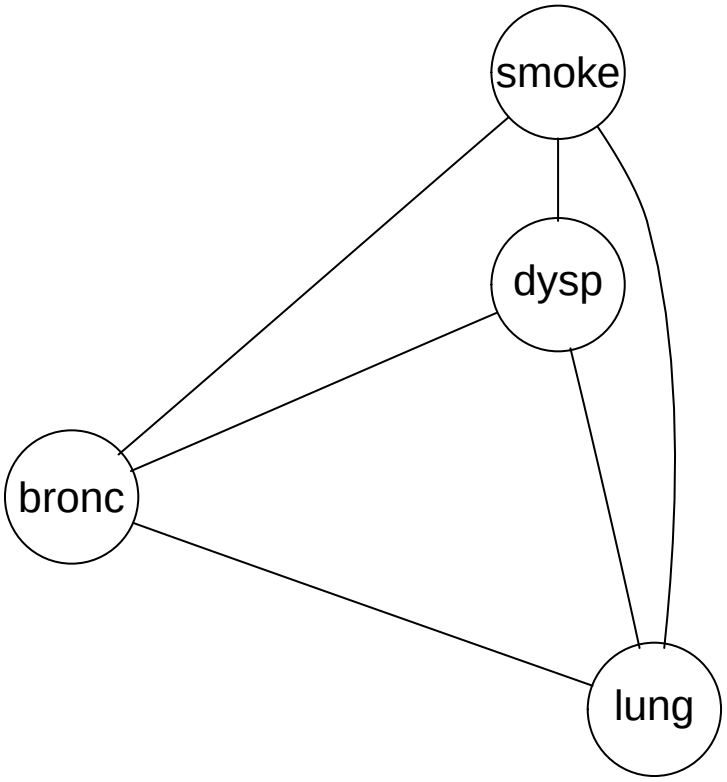
Merging gives

$$P(V) = q(l, s, b)$$

```
# data from exercise 4
plot4=compile(grain(CPT.list4))
plot(plot4$dag)
```




```
plot(moralize(plot4$dag))
```



Here we have

$$P(V) = P(s)P(d)P(b)P(l \mid s, d, b) = q(s)q(d)q(b)q(l, s, d, b)$$

Mergining gives

$$P(V) = q(l, s, d, b)$$

Part 2, initial work

Consider the "cad" data in "gRbase". There are two dataset: "cad1" which is complete and "cad2" which has missing values here and there.

```
library(graph)
```

```
## Loading required package: BiocGenerics
```

```
## Loading required package: parallel
```

```
##
## Attaching package: 'BiocGenerics'
```

```
## The following objects are masked from 'package:parallel':
##
##     clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##     clusterExport, clusterMap, parApply, parCapply, parLapply,
##     parLapplyLB, parRapply, parSapply, parSapplyLB
```

```
## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs
```

```
## The following objects are masked from 'package:base':
##
##     anyDuplicated, append, as.data.frame, basename, cbind, colnames,
##     dirname, do.call, duplicated, eval, evalq, Filter, Find, get, grep,
##     grepl, intersect, is.unsorted, lapply, Map, mapply, match, mget,
##     order, paste, pmax, pmax.int, pmin, pmin.int, Position, rank,
##     rbind, Reduce, rownames, sapply, setdiff, sort, table, tapply,
##     union, unique, unsplit, which, which.max, which.min
```

```
library(Rgraphviz)
```

```
## Loading required package: grid
```

```
library(RBGL)
library(gRbase)
library(gRain)
library(bnlearn)
```

```
##
## Attaching package: 'bnlearn'
```

```
## The following objects are masked from 'package:gRbase':
##
##     ancestors, children, parents
```

```
library(magrittr)
data(cad1, package="gRbase")
names(cad1)
```

```
##  [1] "Sex"         "AngPec"      "AMI"         "QWave"       "QWavecode"
##  [6] "STcode"      "STchange"    "SuffHeartF"  "Hypertrophi" "Hyperchol"
## [11] "Smoker"      "Inherit"     "Heartfail"   "CAD"
```

```
use <- c("Sex", "CAD", "Inherit", "Smoker", "Hyperchol", "Heartfail", "AMI")
dat1 <- cad1[, use] # Loader data
```

Exercise 6

Use the hill climbing algorithm function from the "bnlearn" package to estimate different Bayesian networks based on data "cad1". see R script provided elsewhere.

Answer

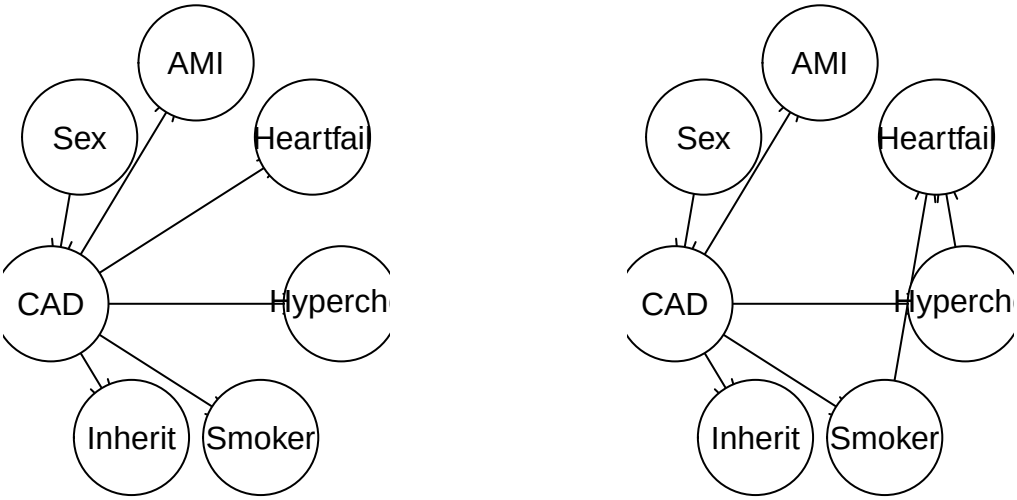
```
## Start search from empty graph
mm1 <- hc(dat1)
mm1
```

```
##
##   Bayesian network learned via Score-based methods
##
##   model:
##    [Sex][CAD|Sex][Inherit|CAD][Smoker|CAD][Hyperchol|CAD][Heartfail|CAD][AMI|CAD]
##   nodes:                                 7
##   arcs:                                  6
##     undirected arcs:                     0
##     directed arcs:                       6
##   average markov blanket size:           1.71
##   average neighbourhood size:            1.71
##   average branching factor:              0.86
##
##   learning algorithm:                    Hill-Climbing
##   score:                                 BIC (disc.)
##   penalization coefficient:              2.731916
##   tests used in the learning procedure:  57
##   optimized:                             TRUE
```

```
## Start search from complete graph
sat <-random.graph(use, prob = 1) # Generate empty or random directed acyclic graphs from a given set o
mm2 <- hc(dat1, start=sat)
mm2
```

```
##
##   Bayesian network learned via Score-based methods
##
##   model:
##    [Sex][CAD|Sex][Inherit|CAD][Smoker|CAD][Hyperchol|CAD][AMI|CAD]
##    [Heartfail|Smoker:Hyperchol]
##   nodes:                                 7
##   arcs:                                  7
##     undirected arcs:                     0
##     directed arcs:                       7
##   average markov blanket size:           2.29
##   average neighbourhood size:            2.00
##   average branching factor:              1.00
##
##   learning algorithm:                    Hill-Climbing
##   score:                                 BIC (disc.)
##   penalization coefficient:              2.731916
##   tests used in the learning procedure:  126
##   optimized:                             TRUE
```

```
# Now we have the two generated graphs
par(mfrow=c(1,2))
plot(mm1)
plot(mm2)
```




```
## Create Bayesian network
bn1 <- as.grain(bn.fit(mm1, dat1)) # This will be the focus point
bn2 <- as.grain(bn.fit(mm2, dat1))
```

Exercise 7

Predict the value of the CAD variable in the dataset "cad1" for each of the models you find. Predict the value of the CAD variable in the dataset "cad2" for each of the models you find. Is it most appropriate to evaluate the models based on "cad1" or "cad2".

Answer

```
## Predict data
## Sample 40 random rows
set.seed(1213)
userow <- sample(nrow(dat1), 40) # Take 40 observations out of dataset
wdat1 <- dat1[userow,] # Use them
pred1 <- predict(bn1, newdata=wdat1, response="CAD") # This will be the focus
pred2 <- predict(bn2, newdata=wdat1, response="CAD")
table(wdat1$CAD, pred1$pred$CAD) # The real dataset vs. the predicted for hc method
```

```
##
##       No Yes
##   No  16   7
##   Yes  4  13
```

```
table(wdat1$CAD, pred2$pred$CAD)
```

```
##
##       No Yes
##   No  15   8
##   Yes  4  13
```

```
## Procent
table(wdat1$CAD, pred1$pred$CAD)/40*100
```

```
##
##         No  Yes
##   No  40.0 17.5
##   Yes 10.0 32.5
```

```
table(wdat1$CAD, pred2$pred$CAD)/40*100
```

```
##
##         No  Yes
##   No  37.5 20.0
##   Yes 10.0 32.5
```

```
#' Notice:
#'
#' Prediction based on same data as we used for fitting / model search
#' is cheating. Use cad2 data instead.
#'
#' What are the misclassification errors under various models?
#'
#' Which misclassifications are the most serious ones?
#'
#' Using cad2:
data(cad2, package="gRbase")
names(cad2)
```

```
##  [1] "Sex"         "AngPec"      "AMI"         "QWave"       "QWavecode"
##  [6] "STcode"      "STchange"    "SuffHeartF"  "Hypertrophi" "Hyperchol"
## [11] "Smoker"      "Inherit"     "Heartfail"   "CAD"
```

```
use <- c("Sex", "CAD", "Inherit", "Smoker", "Hyperchol", "Heartfail", "AMI")
dat2 <- cad2[, use]
## Sample 40 random rows
set.seed(1213)
userow <- sample(nrow(dat2), 40) # Pick 40 observations
wdat2 <- dat2[userow,]
pred3 <- predict(bn1, newdata=wdat2, response="CAD")
pred4 <- predict(bn2, newdata=wdat2, response="CAD")
table(wdat2$CAD, pred3$pred$CAD)
```

```
##
##       No Yes
##   No  23   4
##   Yes  6   7
```

```
table(wdat2$CAD, pred4$pred$CAD)
```

```
##
##       No Yes
##   No  23   4
##   Yes  4   9
```

Exercise 8

Compute the misclassification probabilities for persons with CAD and persons without CAD for each model. Which misclassification is most severe?

Answer

```
## Procenter
table(wdat2$CAD, pred3$pred$CAD)/40*100
```

```
##
##         No  Yes
##   No  57.5 10.0
##   Yes 15.0 17.5
```

```
table(wdat2$CAD, pred4$pred$CAD)/40*100
```

```
##
##         No  Yes
##   No  57.5 10.0
##   Yes 10.0 22.5
```

Type 1 error: to reject, while the patient is positive

Type 2 error: to accept, while the patient is negative

It can be seen that a type 1 and type 2 error is the same for “pred4” for cad2"

For “pred3” for “cad2” there is 15% type 1 error and 10% type 2 error

If the treatment is hard on the patients, a type 2 error would be problematic, however introducing multiple test would avoid this

Type 1 errors are “the most servere” since the patient wouldn’t get the treatment

this can lead to large consequences for the patient.