The Cost of Healthcare

Marco Scutari

Academic Year 2022–2023

The cost of healthcare is a recurring theme in most countries' public discourse due to the combination of an ageing population and the availability of more advanced (read, expensive) treatments. You will try to build a simple CGBN to model an individual's yearly medical expenditure. We will use the UK National Health Service (NHS) as an inspiration: it is free at the point of use for most things and it is financed through taxes (the National Insurance and general taxation).
For the sake of the example, we consider only seven variables:

- **Age** (`A`, discrete): the age, recorded as *young* (`"young"`), *adult* (`"adult"`) or *old* (`"old"`).
- **Pre-existing conditions** (`C`, discrete): whether the individual has *no pre-existing conditions* (`"none"`), *mild* (`"mild"`) or *severe pre-existing conditions* (`"severe"`).
- **Outpatient expenditure** (`O`, continuous): the cost of the individual's outpatient hospital visits, like specialist consultations.
- **Inpatient expenditure** (`I`, continuous): the cost of the individual's inpatient hospital visits, which include hospital admissions.
- **Any hospital say** (`H`, discrete): whether the individual spent any days at the hospital, recorded as `"none"` or `"any"`.
- **Days of hospital stay** (`D`, continuous): how many days the individual spent at the hospital.
- **Taxes** (`T`, continuous): the amount of money the individual should pay in taxes to cover his medical expenses.

Part 1: Construct the Network Structure

Firstly, you should try to get a feeling for the problem you are trying to model to have a term of comparison for the outputs of structure learning. Some considerations:

- Pre-existing conditions become more likely with age: older people are inherently more fragile than younger people.
- For the same reason, we expect older people to have both higher outpatient and inpatient expenditures, and to be admitted for longer periods of time during hospital stays.
- Both expenditures over the course of one year should be predictive of the overall expenditures for the following year, and therefore should be used to determine the amount of tax paid.
- It is illegal to discriminate individuals by either their age or pre-existing conditions in most settings (including health insurance).

What do the local distributions look like? What and how many parameters do they have?

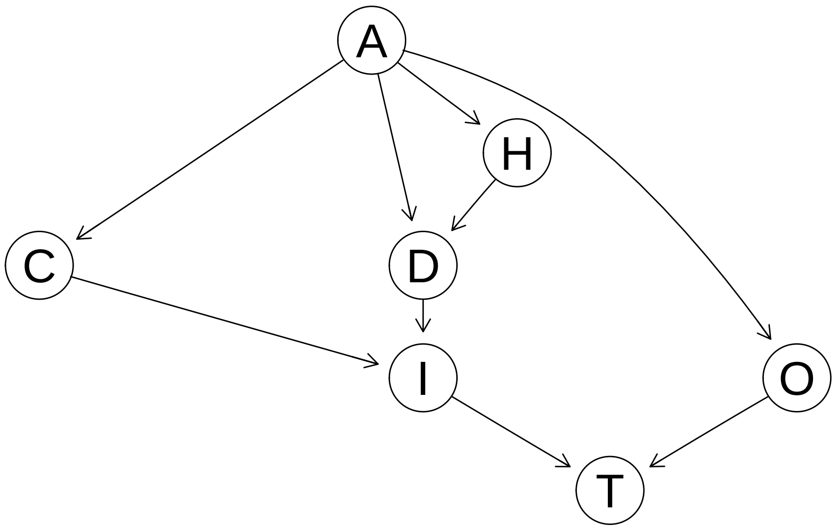
Commented Code for Part 1: Construct the Network Structure

The hints above will structure the BN using the following relationships:

$$\mathtt{A} \rightarrow \mathtt{O}, \quad \mathtt{A} \rightarrow \mathtt{C}, \quad \mathtt{A} \rightarrow \mathtt{H}, \quad \mathtt{A} \rightarrow \mathtt{D}, \quad \mathtt{H} \rightarrow \mathtt{D}, \quad \mathtt{C} \rightarrow \mathtt{I}, \quad \mathtt{D} \rightarrow \mathtt{I}, \quad \mathtt{O} \rightarrow \mathtt{T}, \quad \mathtt{I} \rightarrow \mathtt{T}.$$

Note that you should explicitly not include the arcs $\mathtt{A} \rightarrow \mathtt{T}$ and $\mathtt{C} \rightarrow \mathtt{T}$ because discriminating based on age or pre-existing conditions is illegal. The resulting DAG is as follows.

```
healthcare.dag = model2network("[A][C|A][H|A][D|A:H][I|C:D][O|A][T|O:I]")
graph.par(list(nodes = list(fontsize = 10)))
graphviz.plot(healthcare.dag)
```



The local distributions of the nodes are:

- `A`: a multinomial distribution, with a CPT with three probabilities.
- `C`: a multinomial distribution, with a CPT with three rows and three columns.
- `O`: a mixture of 3 simple regressions models, each with an intercept and a standard error.
- `I`: a mixture of 3 regressions models, each with an intercept, a regression coefficient for `D` and a standard error.
- `H`: a multinomial distribution, with a CPT with two rows and three columns.
- `D`: a mixture of 6 simple regressions models, each with an intercept and a standard error.
- `T`: a regression model with an intercept, regression coefficients for `I` and `O` and a standard error.

Part 2: Learn the Network Structure and the Parameters

The data to learn this model are available in a separate file named `healthcare.txt`, which you can read into R as follows:

```
costs = read.table("healthcare.txt", header = TRUE,
          colClasses = c("factor", "factor", "numeric", "factor", "numeric", "numeric", "numeric"))
```

1. Learn the structure of the BN using hill-climbing with the default BIC score.
2. Spoiler alert: the BN the data are generated from is

   ```
   modelstring(healthcare.dag)
   ```
   ```
   | [1] "[A][C|A][H|A][O|A][D|A:H][I|C:D][T|I:O]"
   ```

   Use `graphviz.compare()` to compare it with the DAG you learned: how does it differ?

3. Learn the parameters of both the `healthcare.dag` and othe DAG you learned from data. Can you guess why you cannot learn `healthcare.dag` from the data correctly? You can enable the debugging output in the structure learning algorithm you used and investigate which steps it went through.

4. Can you incrementally build a blacklist to learn the correct model? Use `shd()` and `compare()` as well as `graphviz.compare()`.

BONUS: you can do learn the BN with different structure learning algorithms and compare!

Commented Code for Part 2: Learn the Network Structure and the Parameters

1. The DAG learned from the data is the following:

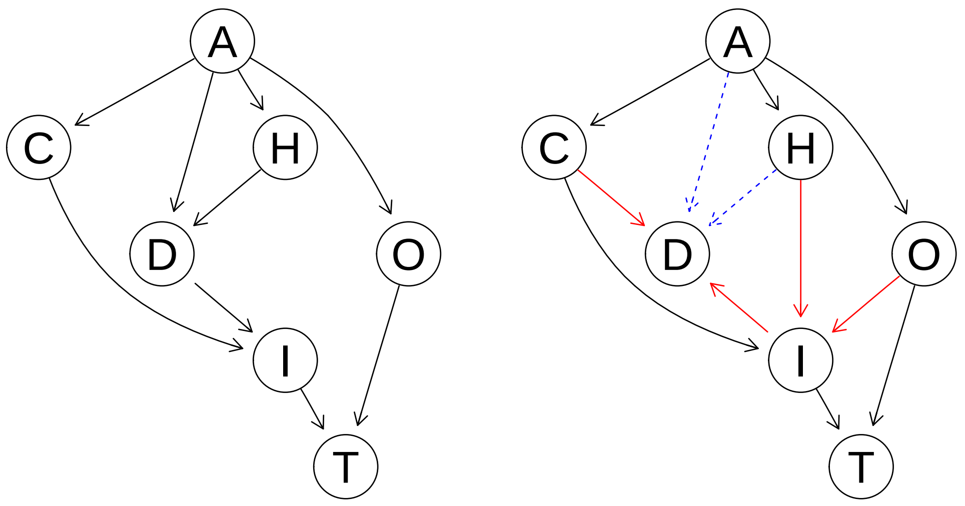
```
learned = hc(costs, score = "bic-cg")
modelstring(learned)
```

```
[1] "[A][C|A][H|A][O|A][I|C:H:O][D|C:I][T|I:O]"
```

   which differs from the original.

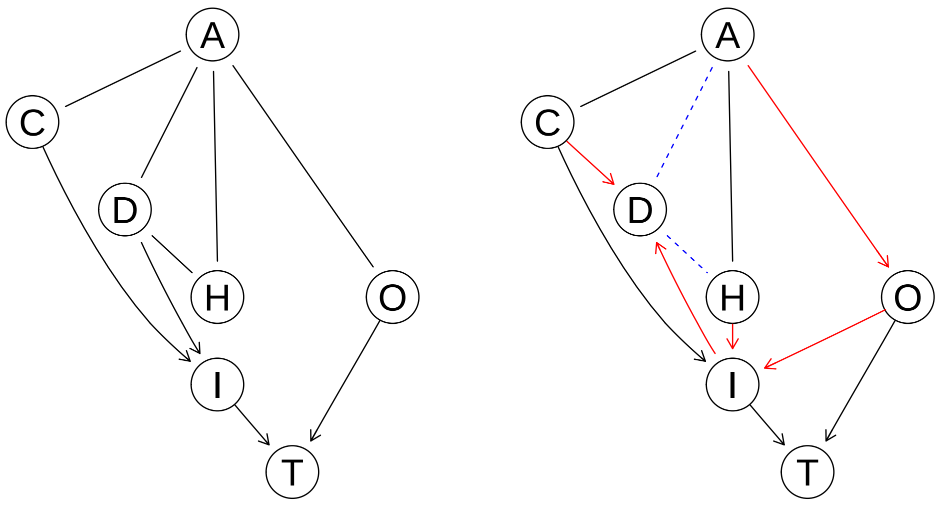
2. Visually:

```
par(mfrow = c(1, 2))
graph.par(list(nodes = list(fontsize = 10)))
graphviz.compare(healthcare.dag, learned)
```



Red arcs are false positives (should not be there), dashed blue arcs are false negatives (should be there, but are not). Comparing the CPDAGs does not improve things: the differences are not due to hill-climbing learning a different DAG in the same equivalence class as `healthcare.dag`.

```
par(mfrow = c(1, 2))
graph.par(list(nodes = list(fontsize = 10)))
graphviz.compare(cpdag(healthcare.dag), cpdag(learned))
```



Most of the arcs that you are unable to learn correctly are incident on the node `D`.

3. You can learn the parameters with `bn.fit` and deafult argument values, which gives you maximum likelihood estimates.

```
fitted = bn.fit(healthcare.dag, data = costs)
```

Printing the local distribution for node `D` reveals that it is singular. It stands to reason that BIC, which assumes that models are not singular, does not work well.

```
fitted$D

    Parameters of node D (conditional Gaussian distribution)

  Conditional density: D | A + H
  Coefficients:
                  0     1     2     3     4     5
  (Intercept)  4.02  7.13  1.00  0.00  0.00  0.00
  Standard deviation of the residuals:
      0      1      2      3      4      5
  0.953  1.554  0.445  0.000  0.000  0.000
  Discrete parents' configurations:
         A     H
  0  adult   any
  1    old   any
  2  young   any
  3  adult  none
  4    old  none
  5  young  none
```

More in general, any score that builds on the loglikelihood of the BN will display some kind of pathological behaviour because the loglikelihood of node `D` is `-Inf`. (The kernel of the loglikelihood is $-\log(\mathrm{VAR}(\varepsilon_{\mathrm{D}}))$ and $\mathrm{VAR}(\varepsilon_{\mathrm{D}}) = 0$ because the model is singular.)

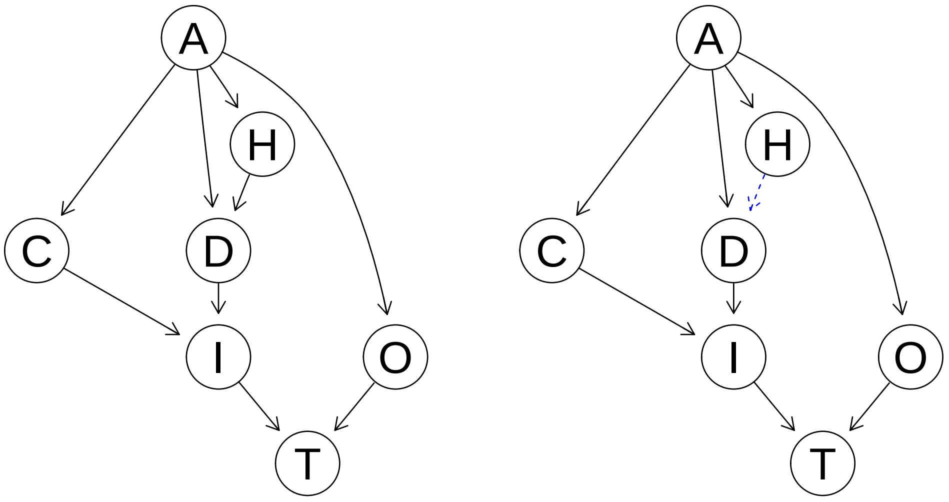
```
score(healthcare.dag, costs, type = "loglik-cg", by.node = TRUE)

      A     C     D     H     I     O     T
  -2092 -1234  -Inf -1008 -9983 -8651 -7459
```

4. The model you learned earlier has:

```
shd(healthcare.dag, learned)

  [1] 7

unlist(compare(healthcare.dag, learned))

  tp fp fn
   6  4  3
```

You can blacklist I, O and T from being parents of D: unless you discharge sick patients to save money, these arcs do not reflect real-world processes.

```r
bl = data.frame(
  from = c("I", "O", "T"),
  to = c("D", "D", "D")
)
par(mfrow = c(1, 2))
learned = hc(costs, blacklist = bl)
graph.par(list(nodes = list(fontsize = 10)))
graphviz.compare(healthcare.dag, learned)
```



```r
shd(healthcare.dag, learned)
```

```
[1] 3
```

```r
unlist(compare(healthcare.dag, learned))
```

```
tp fp fn
 8  0  1
```

The last arc that is missing, H → D, is essentially impossible to learn because **bnlearn** refuses to learn singular models by default. Adding it with a whitelist would work, though.

Part 3: Query and Validate the Network

You should query the true model to:

- check that the model agrees with publicly-available information on healthcare expenditures; and
- to reason about the amount of taxes required to finance healthcare in a sustainable and fair manner.

BONUS: you can do that with the BN you originally learned from data as well and compare!

In particular:

1. Check that the average per-day expenditure for an inpatient is about £400. In order to do this, supply to `cpdist()` the evidence that inpatients have spent at least 1 full day at a hospital (`D >= 1`) and thus have caused at least the fixed-costs expenditures (`I >= 100`).
2. Check the distribution of the outpatients expenditures `O`. Consider that on average a visit to a GP costs £37.5, a prescription costs £8, a referral costs £180, an ambulance trip costs £192–£252.
3. Check `D` and `C`. For the former, the average should be between 4.5 and 5. For the latter, around 14% of people under 40 should have a pre-existing condition: since `A == "young"` identifies people under 30 you would expect an even smaller number. Similarly, you would expect about 60% of old people to have some sort of condition.

4. Assess the distribution of the amount of taxes T. Are you running a surplus on average?
5. Incorporate the evidence that population is growing older into the BN.

```
new.A.prob = array(c(0.30, 0.40, 0.30), dim = 3, dimnames = list(A = A.lv))
healthcare.bn$A = new.A.prob
```

   How do expenditures change?

6. Incorporate the evidence that people develop on average more health conditions as they grow old.

```
new.C.prob = array(c(0.88, 0.10, 0.02, 0.70, 0.22, 0.08, 0.41,
                    0.51, 0.08), dim = c(3, 3),
               dimnames = list(C = C.lv, A = A.lv))
healthcare$C = new.C.prob
```

   How do expenditures change?

7. What would be the effect of shortening hospital stays by one day on average?

Commented Code for Part 3: Query and Validate the Network

1. The result is indeed close to £400:

```
part = cpdist(healthcare.bn, nodes = c("I", "D"),
         evidence = (D >= 1) & (I >= 100), n = 10^5)
per.day = (part$I - 100) / part$D
c(mean = mean(per.day), quantile(per.day, c(0.01, 0.99, 0.999)))

   mean    1%   99% 99.9%
    404   274   817   856
```

   and for the vast majority of people ranges from £274 to £856.

2. The range and the distribution of the simulated values are realistic.

```
part = cpdist(healthcare.bn, nodes = "O", evidence = (O >= 0), n = 10^5)
summary(part$O)

     Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
       17      66     171     174     205     509
```

   In the lower quartile, you have people who go their GP up to two times a year (which costs on average £37.5 per visit in 2018) and require up to two prescriptions (at £8 each). On the other hand, referrals cost on average £180 each; going to Accidents & Emergency (A&E) can cost anywhere between £45 and £400; ambulance trip costs on average £252 or £192, depending on whether they result in a trip to A&E or not. Hence it is easy to see how expenditures can increase all the way to £509 in the event of some serious accident or follow-up visits.

3.
```
part = cpdist(healthcare.bn, nodes = "D", evidence = (H == "any"), n = 10^5)
c(mean = mean(part$D), quantile(part$D, c(0.01, 0.99)))

   mean    1%   99%
  4.567 0.238 9.836

cpquery(healthcare.bn, event = (C %in% c("mild", "severe")),
        evidence = (A == "young"), n = 10^5)

  [1] 0.118

cpquery(healthcare.bn, event = (C %in% c("mild", "severe")),
        evidence = (A == "old"), n = 10^5)

  [1] 0.582
```

4.
```r
part = cpdist(healthcare.bn, nodes = c("I", "O", "T"),
          evidence = (I >= 0) & (O >= 0), n = 10^5)
summary(part$T)
```
```
     Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
      157     312     415     863     630   10609
```
```r
finances = c(mean.tax = mean(part$T),
             mean.expenditure = mean(part$I + part$O),
             surplus = mean(part$T) - mean(part$I + part$O))
finances
```
```
          mean.tax mean.expenditure          surplus
               863              724              140
```

5.
```r
new.A.prob = array(c(0.30, 0.40, 0.30), dim = 3, dimnames = list(A = A.lv))
healthcare.bn$A = new.A.prob
part = cpdist(healthcare.bn, nodes = c("I", "O"),
         evidence = (I >= 0) & (O >= 0), n = 10^5)
mean(part$I + part$O)
```
```
  [1] 850
```
```r
finances["mean.tax"] - mean(part$I + part$O)
```
```
  mean.tax
        13
```

6.
```r
new.C.prob = array(c(0.88, 0.10, 0.02, 0.70, 0.22, 0.08, 0.41,
                      0.51, 0.08), dim = c(3, 3),
                dimnames = list(C = C.lv, A = A.lv))
healthcare.bn$C = new.C.prob
part = cpdist(healthcare.bn, nodes = c("I", "O"),
          evidence = (I >= 0) & (O >= 0), n = 10^5)
mean(part$I + part$O)
```
```
  [1] 871
```
```r
finances["mean.tax"] - mean(part$I + part$O)
```
```
  mean.tax
     -7.32
```

7.
```r
new.D.coef = list(coef = array(c(0, 0, 0, 1, 4, 6), dim = c(1, 6),
                        dimnames = list("(Intercept)", NULL)),
                  sd = c(0, 0, 0, 0.5, 1, 1.5))
healthcare.bn$D = new.D.coef
part = cpdist(healthcare.bn, nodes = c("I", "O"),
         evidence = (I >= 0) & (O >= 0), n = 10^5)
finances["mean.tax"] - mean(part$I + part$O)
```
```
  mean.tax
      51.6
```