Educational Resources / Recursos Educacionais

Causal Discovery: An Introduction

Luís Cavique
DCeT, Univ. Aberta
Luis.Cavique@uab.pt

Lisboa, April 2024

Abstract

This document aims to complement the bibliography of the Data Knowledge Extraction (Data Mining) curricular unit offered in the MEIW (Mestrado em Engenharia Informática e Tecnologia Web) and MBB (Mestrado em Bioestatística e Biometria) masters courses.

Causal discovery aims to identify causal relationships within a given data set. There are various approaches to achieve this, including the PC algorithm. PC algorithm is widely implemented in causal discovery tasks and can be coded to analyze different datasets to verify causal connections.

Use cases of the PC algorithm include its application to datasets such as house prices, where it might uncover factors that significantly influence property values, and the Titanic dataset, which can help identify the key factors contributing to survival rates.

Contents

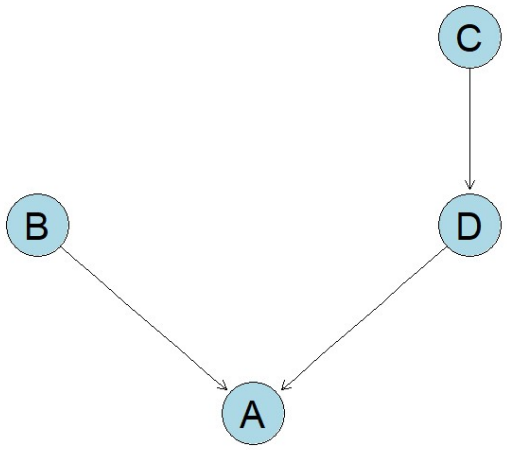
1. Causal discovery objective

This work aims to find the corresponding Directed Acyclic Graph (DAG) for any given data set. This task involves identifying the structure within the data that can represent directional dependencies, which is crucial for understanding the underlying relationships and influences among variables. The input and output of causal discovery are the following:

a) the input is a set of data:

A	B	C	D
0.8	1.2	2.2	2.0
2.0	2.0	1.1	3.2
3.9	1.2	2.0	2.1
3.1	1.3	2.1	2.0
1.0	2.0	1.1	3.2
3.9	3.6	1.9	2.1
3.1	1.3	5.1	2.1
1.4	1.1	1.1	3.2
3.3	3.8	2.3	2.1
1.0	1.5	3.1	3.1

b) the output corresponds to a DAG:



Given a data set with four attributes {A, B, C, D} and after calculating the relationships (or correlations) between the attributes, the aim is to find a direct acyclic graph, DAG, with causality relationships. The example shows that A = f1(B, D) and D = f2 (C).

2. Different approaches

For DAG generation, there are three approaches: (i) functional causal models, (ii) score-based methods, and (iii) constraint-based methods [Zanga et al. 2022], [Molak 2023].

(i) Functional Causal Models
Functional Causal Models (FCMs) assume that the relationships between variables can be represented by structural equations involving the direct causes of each variable. LiNGAM (Linear Non-Gaussian Acyclic Models) focuses explicitly on the data's linear relationships between variables and non-Gaussianity. LiNGAM utilizes non-Gaussianity properties, such as the Independent Component Analysis (ICA), to identify the causal ordering of variables. By iteratively estimating the structural coefficients, LiNGAM aims to discover the underlying causal structure of the linear FCM.

(ii) Score-based methods
Score-based methods have also been used to discover causal structures. Score-based methods include the Greedy Equivalence Search (GES), Fast GES, and K2 algorithms. The Greedy Equivalence Search (GES) algorithm starts with an empty graph and iteratively adds and removes directed edges to maximize the improvement of the scoring function. The Bayesian Information Criterion (BIC) or the Akaike Information Criterion (AIC) are examples of scoring functions. Fast GES is an improved and parallelized version of GES. K2 performs a constructive heuristic search for the parents of each node. Max-Min Hill Climbing finds the skeleton of the Bayesian network followed by a heuristic to orient the direction of the edges.

(iii) Constraint-based methods
The Inductive Cause and PC algorithms are among the most common constraint-based algorithms. IC (Inductive Causation) returns the equivalent class of the DAG based on the estimated probability distribution of random variables and an underlying DAG structure. PC (Peter-Clark, named in honor of the authors Peter Spirtes and Clark Glymour) is produced by iteratively checking the conditional independence conditions of adjacent nodes given a partially directed acyclic graph. In causal discovery, the PC algorithm has more bibliographic references.

3. PC algorithm

The PC algorithm [Spirtes et al. 2000] is a well-known algorithm used in causal discovery. It is used to learn the structure of a causal graph from observational data.

The PC algorithm is a constraint-based approach that combines statistical independence tests with graph-based algorithms. The algorithm follows a two-step process:

Step 1: Determine the DAG skeleton
- Given a complete undirected graph, the algorithm identifies the skeleton of the causal graph by testing for statistical independence between variables. The algorithm checks the independence of pairs (X, Y) of variables conditioned by a third variable, Z, $X \perp\!\!\!\perp Y \mid Z$. If variable Z explains the correlation, the relation is called d-separated, and the edge is removed.
- The result of this step is an undirected graph that does not specify the directionality of the causal links.

Step 2: Determine the orientation of the arcs
- The algorithm applies a set of orientation rules based on the case of collider/immorality in the subgraph (X, Y, Z). There are different possibilities to draw three variables: the chain $X \rightarrow Y \rightarrow Z$ or $X \leftarrow Y \leftarrow Z$; the fork $X \leftarrow Y \rightarrow Z$, and the collider/immorality $X \rightarrow Y \leftarrow Z$.
- Using the conditional independence tests again, if $X \perp\!\!\!\perp Y \mid Z$, we are in a fork chain. Otherwise, if $\sim X \perp\!\!\!\perp Y \mid Z$, it is the case of a collider/immorality. The algorithm adjusts the orientations of colliders to satisfy the global Markov property, a fundamental assumption in causal graphical models.

The first step of the PC algorithm removes edges based on conditional independence given a third variable. The second step determines the directionality of the edges to establish the causal order between variables. The PC algorithm is widely used in causal discovery due to its efficiency and ability to handle large datasets. It provides a valuable tool for understanding causal relationships and inferring causal structures from observational data.

4. PC code

```
# PC algorithm
library("Rgraphviz")
library(ggpubr); library(ppcor) # computes partial correlation
df=read.table("diabetes.csv", header=TRUE, sep=";") #<-- please, update dataset
# df=df[-c(1,6)];
minR=0.2       #<-- please, update cols, minR
Target=0       #<-- please,  Target(0/1)

N=ncol(df); matrixR=cor(df); matrixAdj=matrix(0,N,N)

matrixAdj_init <-function() # upper triangular matrix
{ for (i in 1:N)
   for (j in i:N)
      if (i!=j & abs(matrixR[i,j])>minR)
         matrixAdj[i,j]<<-(1)
}

dif3 <- function (a,b,c) {return (a!=b & b!=c & a!=c)}

print3 <- function (txt,a,b,c)
{ if (matrixAdj[a,b]>=1) str1 = paste(txt, names(df[a]), "->", names(df[b]))
  else str1 = paste(txt, names(df[a]), "<-", names(df[b]))
  if (matrixAdj[b,c]>=1) string= paste (str1,"->",names(df[c]))
  else string= paste (str1,"<-",names(df[c]))
  print(string)
}

independent <- function (r, X, Y, Z) # (X_|_Y) | Z,  correlation r
{ pr = pcor.test(X,Y,Z);
  ind_bool = (abs(r) > 2*abs(pr$estimate)) #pr$estimate=partial correlation
  return (ind_bool)
}

All_independent <- function() #step 1, skeleton x-y-z
{ for (x in 1:N)
   for (y in 1:N)
    for (z in 1:N)
      if (dif3(x,y,z))
      if (matrixAdj[x,y]+matrixAdj[y,x]>=1 & matrixAdj[y,z]+matrixAdj[z,y]>=1)
      if (independent(matrixR[x,y], df[x], df[y], df[z]))
         {print(paste("ind (",names(df[x]),"--", names(df[y]),") |", names(df[z])))
          matrixAdj[x,y]<<-0; matrixAdj[y,x]<<-0
         }
}
```

```
All_direct <- function() # step 2, orientation of the arcs
{ for (x in 1:(N-Target))
   for (y in 1:(N-Target))
     for (z in 1:(N-Target))
       if (dif3(x,y,z))
       if (matrixAdj[y,x]+matrixAdj[x,y]>=1 & matrixAdj[y,z]+matrixAdj[z,y]>=1)
       if (!(independent(matrixR[x,z], df[x], df[z], df[y])))
       if (!(matrixAdj[x,y]>=1 & matrixAdj[z,y]>=1))
       {    print3('before:',x,y,z)
            matrixAdj[y,x]<<-0; matrixAdj[x,y]<<-0
            matrixAdj[y,z]<<-0; matrixAdj[z,y]<<-0
            matrixAdj[x,y]<<-1; matrixAdj[z,y]<<-1
            print3(' after:',x,y,z); print("")
       }
}

print_GraphViz <- function(X)
{  f<-file(paste("ZGraphViz",X,".txt"),"w")
   cat("Digraph G {rankdir=LR","\n", file=f)
   for (i in 1:N)
      for (j in 1:N)
         if (matrixAdj[i,j] >= 1)
            cat (names(df[i]),"->",names(df[j]),
                                  "[label=",round(matrixR[i,j],2) ,"]", "\n", file=f)
                                   cat("}","\n", file=f)
   close(f)
   my.matrix<-matrixAdj
   colnames(my.matrix)<-names(df)
   my.graph<-new("graphAM", adjMat=my.matrix, edgemode="directed")
   plot(my.graph, attrs = list(node = list(fillcolor = "lightblue",
                                  lty="dotted", lwd=2,fontsize=22),
                                  edge = list(arrowsize = 0.5)))
}

matrixAdj_init()
matrixAdj; print_GraphViz("01")
All_independent()

matrixAdj; print_GraphViz("02")
All_direct()

matrixAdj; print_GraphViz("03")
```

5. Use cases

In this section, we apply the PC implementation to three different datasets. For all the datasets, the default parameters are set with minR=0.20 (minimum correlation) and Target=1, meaning that the column corresponds to the target.

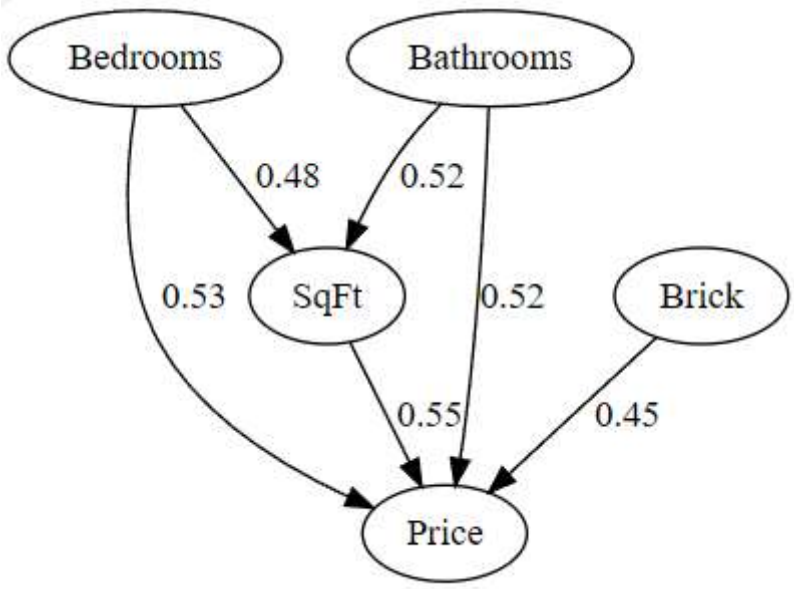
The DAGs are drawn using Graphviz Online https://dreampuf.github.io/GraphvizOnline/.

5.1. House prices dataset

Housing Prices Dataset, Prediction - Regression Problem can be downloaded from Kaggle, https://www.kaggle.com/datasets/yasserh/housing-prices-dataset, and has as target variable Price.

Using minR=0.35 and the following code in Graphviz Online corresponds to the next figure:

```
Digraph G {rankdir=TD
SqFt -> Price [label= 0.55 ]
Bedrooms -> SqFt [label= 0.48 ]
Bedrooms -> Price [label= 0.53 ]
Bathrooms -> SqFt [label= 0.52 ]
Bathrooms -> Price [label= 0.52 ]
Brick -> Price [label= 0.45 ]
}
```



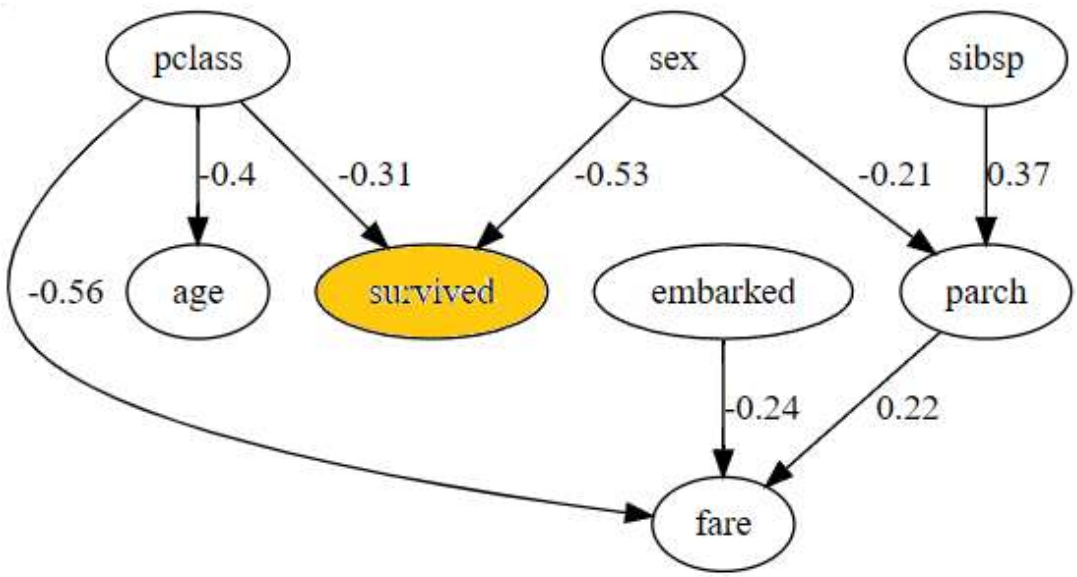
The Price is strongly dependent on Brick construction and area in SqFt.

5.2 Titanic dataset

Titanic - Machine Learning from Disaster can be downloaded from Kaggle, https://www.kaggle.com/c/titanic/data, and has as target variable Survived.

Using minR=0.20 and the following code in Graphviz Online corresponds to the next figure:

```
Digraph G {rankdir=TD
pclass -> survived [label= -0.31 ]
sex -> survived [label= -0.53 ]
sex -> parch [label= -0.21 ]
pclass -> age [label= -0.4 ]
pclass -> fare [label= -0.56 ]
sibsp -> parch [label= 0.37 ]
parch -> fare [label= 0.22 ]
embarked -> fare [label= -0.24 ]
}
```



The people who survived are strongly dependent on sex and class, in particular, women of the first and second classes.

REFERENCES

Molnar C. (2020), Interpretable Machine Learning: a guide for making black box interpretable, lulu.com Editor, ISBN-13: 978-0244768522.

Spirtes P., Glymour C. N., Scheines, R. (2000), Causation, prediction, and search, MIT Press, ISBN-13: 978-1461276500.

Zanga A., E. Ozkirimli, F. Stella (2022), A Survey on Causal Discovery: Theory and Practice, International Journal of Approximate Reasoning, vol. 151, pp. 101-129, ISSN 0888-613X, https:// doi.org/ 10.1016/ j.ijar.2022.09.004.