Title
Population health thinking with Bayesian networks

Author
Aragon, Tomas J.

Publication Date
2018-11-23

Population health thinking with Bayesian networks

Tomás J. Aragón, MD, DrPH[1–4,*]

Working DRAFT, Version December 8, 2019

Contents

[1] Health Officer, City and County of San Francisco
[2] Director, Population Health Division, San Francisco Department of Public Health
[3] Affiliate faculty, UC San Francisco, Department of Epidemiology and Biostatistics
[4] Adjunct faculty, University of California, Berkeley School of Public Health
* Contact: tomas.aragon@sfdph.org PDF: bit.ly/ph-thinking

> *Our comforting conviction that the world makes sense rests on a secure foundation: our almost unlimited ability to ignore our ignorance.*
>
> — Daniel Kahneman [1]

1 Introduction: data science and decision quality

Population health is a systems framework for studying and improving the health of populations through collective action and learning [2]. *Population health data science* (PHDS) is the art and science of transforming data and information into actionable knowledge to improve health [2]. *Actionable knowledge* is information that informs, influences, or optimizes decision-making. A *decision* is a choice between two or more alternatives that involves an irrevocable allocation of time or resources [3,4]. Every decision has an *opportunity cost*—the lost net benefit of the better option not chosen or not considered. Hence: "The roads we take are more important than the goals we announce. Decisions determine destiny."[1]

Decision-making is the most important daily activity. Decisions drive vision, strategy, execution, evaluation, problem-solving, performance, and continuous improvement. Every decision has a *causal assumption* and a *prediction* ("choosing and doing action A (over say, B) will achieve net effect Y with probability p."). This prediction is a "prior probability". Based on our evaluation, we adjust our causal assumption and/or prediction—this is *learning*. Learning leads to new decisions and new actions (adaptation). *Improvements* are adaptations that make processes and/or results better. Continuous improvement is a foundational pillar of a *learning organization*.[2] Continuous decision improvement ensures continuous performance improvement.

The human brain specializes in prediction (also called concepts, schema, memory, etc.) [6]. In 2002, psychologist Daniel Kahneman won the Nobel Prize in Economics for the pioneering studies that cataloged human cognitive biases and pitfalls that formed the foundation of the new field of behavioral economics [1]. It turns out that humans are not good at estimating probabilities (probabilistic reasoning), especially for novel circumstances. We also have nonconscious cognitive biases that affect our ability to draw valid causal inferences and to change course when we are wrong. We are prone to defensiveness to protect our ego and to avoid our fears [7]. Population health thinking requires *intellectual humility* to acknowledge our innate cognitive limitations and *curiosity* to experiment with a new way of computational and inferential thinking [8].

The purpose of this paper is to introduce a new *reasoning framework* which we call **population health thinking** (PHT); that is, the conceptual and/or computational use of Bayesian networks (BNs) (and it's variant) for

1. **probabilistic reasoning** (PR) with *BNs*,
2. **causal inference** (CI) with *causal BNs* (i.e., directed acyclic graphs), and
3. **decision quality** (DQ) with *decision BNs*.

BNs are probabilistic graphical models that can be drawn using one's expert knowledge and wisdom. (Figure 1 depicts the wide applicability of BNs.) First, to get

[1] — Frederick Speakman

[2] A *learning organization* requires other components. For details, see Aragón, et al. [5]

the most out of PHT, master the BN concepts with pencil and paper. Second, explore deploying computational tools to work your intuition and build your confidence. In this article I illustrate the concepts using R—an open source language and environment for statistical computing and graphics [9]. Advanced PHT usually requires turning to computers or to colleagues for computational support.

PHT is the foundational core of population health data science. What is population health data science? In epidemiology, analyses are generally classified as descriptive or analytic. In PHDS we extend this to five analytic domains (Figure 1 and Table 1), all of which should produce actionable knowledge in service of a strategic, tactical or operational decision-making. PHT is important for daily problem-solving, and for analysis, is critical for PHDS levels two through five.

Table 1. Population health data science levels of analysis

Analysis	Description
1 Description	(a) surveillance and early detection of events
	(b) prevalence and incidence of risks and outcomes
2 Prediction	(a) early prediction and targeting of interventions
3 Explanation[a]	(a) discovery and testing of new causal pathways
	(b) estimation of intervention efficacy/effectiveness
4 Simulation	(a) modeling for epidemiologic or decision insights
5 Optimization[b]	(a) optimizing a decision, effectiveness, or efficiency metric

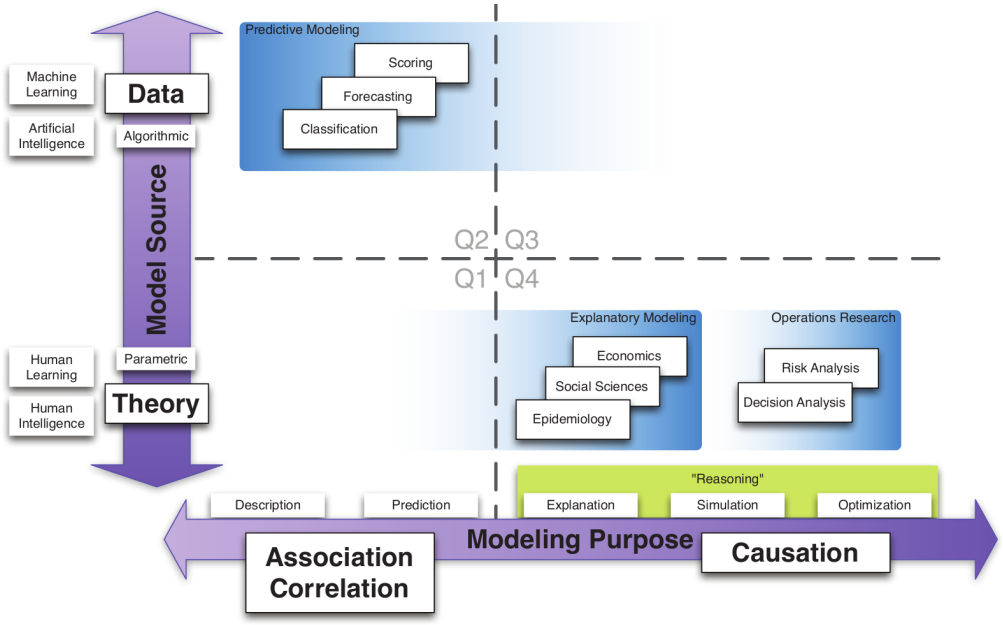


Figure 1. Population health data science landscape (source: http://www.bayesia.com/)

2 Program theory is for the DAGs (motivation)

Basded on causal and prediction assumptions (often implicit), we make decisions and takes actions towards our goals. Our programmatic activities are built upon these causal assumptions which collectively we call **program theory**. Every public health intervention has a program theory; however, many practitioners cannot describe the program theory supporting their primary programmatic activity or research. I too could not describe the program theories supporting my own work until I read Funnell Rogers' book *Purposeful Program Theory* [10].

It turns out that program evaluators not only live and breathe program theory, but they call it by different names: logic model, program logic, theory of change, causal model, results chain, intervention logic, etc. Hence the confusion! From BetterEvaluation.org:[3] "A program theory explains how an intervention (a project, a program, a policy, a strategy) is understood to contribute to a chain of results that produce the intended or actual impacts."[4]

In public health, the logic model is very popular. For me, a logic model is a good high-level summary for non-technical purposes (summary, communication, etc.); however, I do not like them (or variants) as a place to start. For me, *program theory is for the DAGs—directed acyclic graphs—and must include the theories of causation, change, and action.*

Program theory has three components and answers why? what? and how?:

- theory of *causation* (*Why?* primary *roots causes* before an intervention),
- theory of *change* (*What?* key *strategies* to affect the root causes), and
- theory of *action* (*How?* key specific *interventions* to activate theory of change).

In public health we have two common *DAG archetypes* [11]: a *risk (adverse) event* and a *benefit (opportunity) event* (Figure 2). For both, a trigger is an exposure, condition, activity, or incident that increases the probability of a risk or benefit event. A trigger can be a cumulative process. *Before an intervention*, these DAGs represent the **theory of causation** component of program theory.

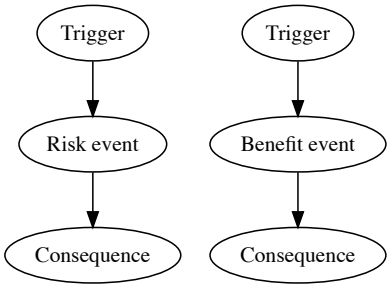


Figure 2. Causal taxonomy for risk event (left) vs. benefit event (right)

Figure 3 depicts the program theory for a public health intervention to reduce automobile crash injuries (a risk event). The **theory of change** has three *strategies* (prevention, control, and mitigation), and the **theory of action** has three *interventions* (speed bumps, automatic breaking, and seat belts).

[3] See http://www.betterevaluation.org/en/plan/define/develop_logic_model.>
[4] See PDF: http://www.betterevaluation.org/sites/default/files/Define%20-%20Compact.pdf.

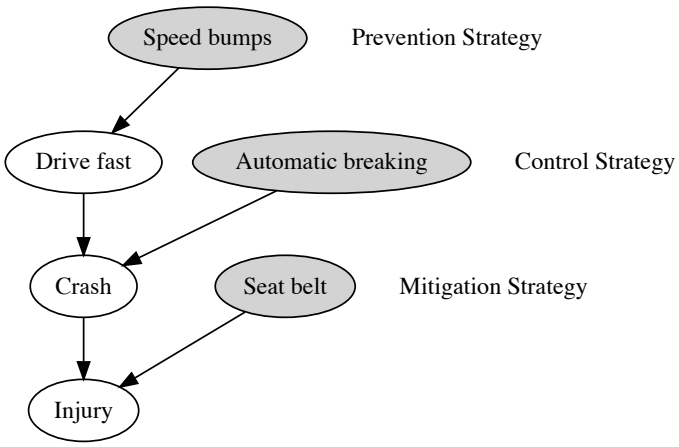


Figure 3. Risk-reduction program theory: theory of causation, theory of change (strategy), and theory of action (intervention)

In a risk-event outcome (consequence), the **5 whys** of root-cause analysis move backwards: Why was there an injury? Because of a crash. Why was there a crash? Because of fast driving? Why was there fast driving? We cannot answer this question (yet).

The program theory is not complete. We must also understand why people drive fast. We have not included the theory of causation from drivers' perspectives. Suppose, for instructional purposes, Figure 4 represents the most common DAG that explains why drivers speed. Therefore, why was he or she driving fast? To make a meeting. Why was this meeting important? To win a contract? Why was this contract important? (unemployment?)

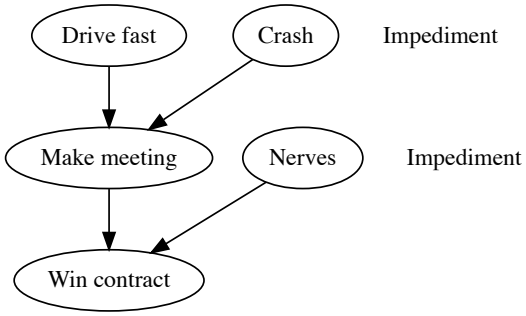


Figure 4. Benefit-event model from the driver's perspective

We can now really appreciate the importance of evaluating multiple perspectives (other causal drivers—not to be confused with vehicle driver in the example). For example, the motivation to drive fast might cancel out the effect of any traditional public health intervention (Figure 3). We must be able to integrate multiple causal pathways reflecting multiple perspectives.

Figure 5 depicts the unified DAG that integrates driver motivation into a holistic, improved public health program theory. We cannot emphasize enough the importance of building causal graphs from multiple perspectives that include risks and benefits,

and different strategy levels. This DAG is a big improvement.

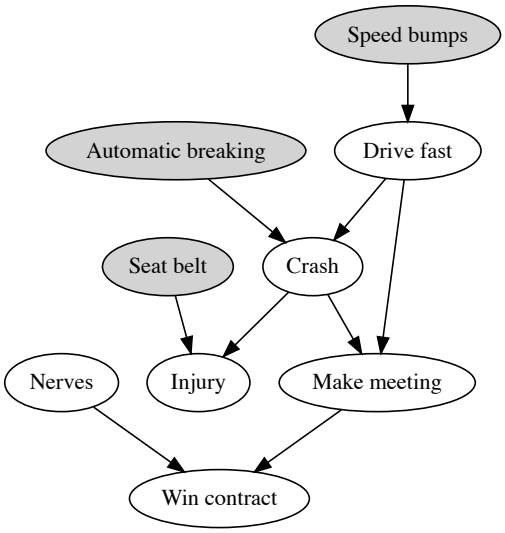


Figure 5. Unified causal model that includes driver's perspective (benefit-seeking) and program theory (risk-reduction)

However, when you review it with subject matter experts they suggest adding "gender" and "age" nodes because both are causally associated with driving fast and wearing seat belts (Figure 6). This will enable you to evaluate the effectiveness of the public health intervention while controlling for the confounding effects of gender and age. For example, if drivers are predominately young males (who drive fast and do not wear seat belts) then the seat belt intervention may appear falsely ineffective. These DAGs encode expert and community knowledge and wisdom, and are used for causal, evidential, and decision reasoning.

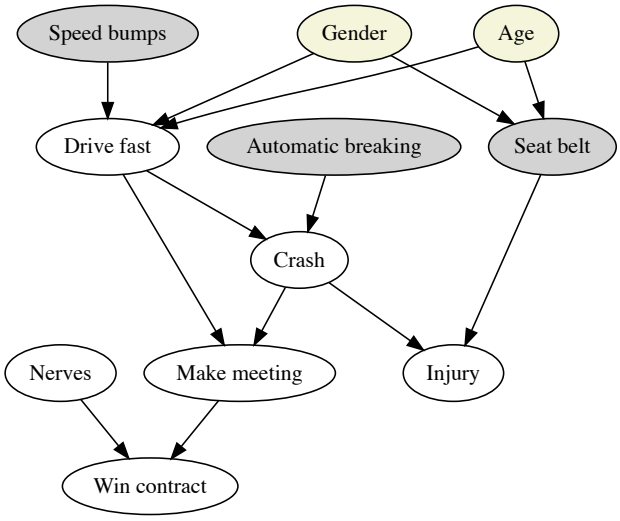


Figure 6. Expanded unified causal model with age and gender

If we would like to know if one of our interventions (e.g., seat belt use) is working

we need to design a study or analysis to test our hypothesis. Unfortunately, without a causal model (Figure 6) to guide us, we cannot know what variables are required to test our intervention, and what variables to control for ("confounders") that threaten the validity of our conclusions.

Here is a summary of program theory:

1. *Every* intervention has a *program theory* (whether expressed or not).
2. Program theory includes *theories of causation, change, and action.*
3. DAGs have *archetypes*: *risk (adverse) event* and *benefit (opportunity) event.*
4. Always include *multiple causal perspectives* (other causal drivers).
5. Use DAGs for *root cause analyses* and *program theory design.*
6. Don't forget to consider *confounders* (that threaten validity).
7. Use DAGs to *test interventions* and to *control confounding.*

Getting this right is important because future decisions and resource allocations depends on these causal inferences. Understanding the basics of program theory is foundational. The sections ahead cover population health thinking concepts and computational tools that support program theory and population health data science.

3 Probabilistic reasoning (with Bayesian networks)

A Bayesian network (BN) is a graphical model for representing probabilistic, but not necessarily causal, relationships between variables called *nodes* [12,13]. The nodes are connected by lines called *edges* which, for our purposes, are always *directed* with an arrow. Consider this *noncausal* BN:

Smell smoke ⟶ Fire nearby

Smelling smoke increases the probability of a fire burning nearby, but obviously smoke alone does not cause a fire. In other words, does knowing X (smell smoke) change the *credibility* of Y (fire nearby)? In contrast, now consider this *causal* BN:[5]

Fire ⟶ Smoke

This causal BN depicts fire causing smoke. Notice that both noncausal and causal BNs have probabilistic dependence which we will use for probabilistic reasoning. Noncausal BNs are commonly used in *influence diagrams*[6] for decision analysis which we cover later.

A two-node causal BN which has two types of proabilistic reasoning (Table 2).

Cause ⟶ Effect

Table 2. Types of probabilistic reasoning for two-node causal Bayesian network

Probabilistic reasoning	Conditional probabilities
Causal (predictive) reasoning	$P(\text{Effect} \mid \text{Cause})$
Evidential (diagnostic) reasoning	$P(\text{Cause} \mid \text{Effect})$

When a causal effect is not firmly established, the BN asserts this concept:

[5] also called a *causal graph* or a *directed acyclic graph* (DAG)
[6] also called *decision networks* or *relevance diagrams*

$$\text{Hypothesis} \longrightarrow \text{Evidence}$$

Table 3. Bayesian network involving a hypothesis and evidence

Conditional probabilities	Probabilistic reasoning
$P(\text{Evidence} \mid \text{Hypothesis})$	Causal reasoning
$P(\text{Hypothesis} \mid \text{Evidence})$	Evidential reasoning

Evidential reasoning require Bayes Theorem.

$$P(H \mid E) = \frac{P(H)P(E \mid H)}{P(E)}$$

$P(H)$ is the *prior (old) belief*, $P(H \mid E)$ is the *posterior (new) belief*, and $P(E \mid H)$ is called the *likelihood*. The likelihood is critical because it is usually measurable, allowing us to update our update our belief.

The marginal probabilty $P(E)$ can be "marginalized over H" which is a sum of conditional probabilities.

$$P(H \mid E) = \frac{P(H)P(E \mid H)}{P(H)P(E \mid H) + P(\overline{H})P(E \mid \overline{H})}$$

We need Bayes Theorem for two important reasons:

1. to use evidence and theory to update our belief from $P(H)$ to $P(H \mid E)$, and
2. to avoid the *fallacy of the transposed conditional*; i.e., confusing $P(E \mid H)$ with $P(H \mid E)$.[7]

3.1 How to remember Baye's Theorem

Again, consider the causal BN

$$\text{Hypothesis} \bullet \rightarrow \bullet \text{ Evidence}$$

Starting from left to right, and then from right to left we can tabulate the marginal and conditional probabilites.

Table 4. Using a causal Bayesian Network ($H \rightarrow E$) to derive expressions for causal reasoning and evidencial reasoning, and to derive Bayes' Formula (see Figure 7)

Reasoning	N	Probability	Description
Causal reasoning	1	$P(H)$	Prior probability (margin probability of H)
... leads to	2	$P(E \mid H)$	Likelihood (TP [sensitivity], FP [1-specificity])
Evidential reasoning	3	$P(E)$	Marginal probability of E
... leads to	4	$P(H \mid E)$	Posterior probability (conditional probability)

For **Bayes' theorem** just substitute probability expressions from Table 4:

[7] For example, because African Americans make up a high proportion in our criminal justice system, some people believe, mistakenly, that a high proportion African Americans are involved in crime. To understand this phenomenon we would need a full causal model.

$$(4) = \frac{(1)(2)}{(3)}$$

Figure 7 is an infographical depiction of Table 4.

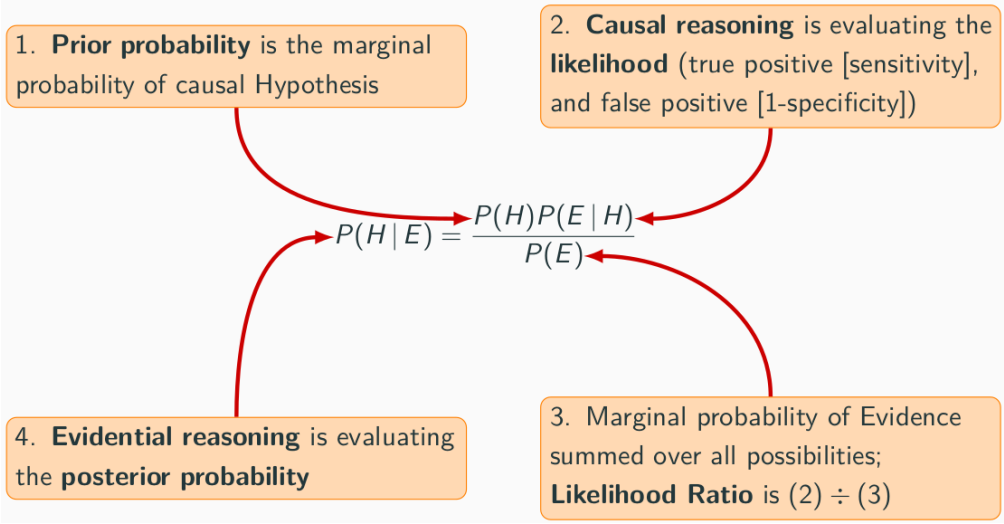


Figure 7. Bayes' theorem for causal BN (Hypothesis -> Evidence).

3.2 Example 1: HIV testing

For example, from Neapolitan (p. 491, [14]), suppose Sam takes a test (evidence) to determine whether he has HIV infection (hypothesis). Here is the BN:

$$\text{HIV} \longrightarrow \text{Test}$$

In diagnostic testing we use Bayes Theorem to calculate the post-test probability from the test results, pre-test probability (prevalence of infection), and test characteristics (sensitivity, specificity). Table 5 displays the data we need for applying Bayes Theorem.

Table 5. Probabilities for using Bayes Theorem in diagnostic testing

Name	Probabilities	Value
Pre-test (prior) probability of HIV+	$P(\text{HIV}+)$	0.00001
Sensitivity	$P(\text{Test}+ \mid \text{HIV}+)$	0.999
Specificity	$P(\text{Test}- \mid \text{HIV}-)$	0.998
Post-test (posterior) probability	$P(\text{HIV} \mid \text{Test})$	TBD

To illustrate, we can calculate the "positive predictive value" (PPV)—what is the probability of being HIV-positive given a positve diagnostic test?

$$P(H+ \mid T+) = \frac{P(T+ \mid H+)P(H+)}{P(T+ \mid H+)P(H+) + P(T+ \mid H-)P(H-)}$$

This is easy to calculate in R.

```
prior <- 0.00001
sens <- 0.999
spec <- 0.998
(sens*prior)/(sens*prior+(1-spec)*(1-prior)) # PPV
```

```
#> [1] 0.004970223
```

Calculating the PPV (or NPV) is evidential reasoning and it requires applying Bayes Theorem. Our brains are not capable of this calculation; we need computational tools. In other words, our brains are not able to "flip the arrow" from $P(\text{Evidence} \mid Hypothesis)$ to $P(\text{Hypothesis} \mid \text{Evidence})$ and make valid Bayesian calculations.

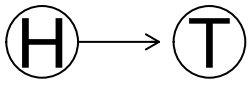
3.2.1 Bayesian network packages in R

Since we need computational tools, why not just use BN tools that can scale to the complexity of any problem and always provide us with valid Bayesian calculations. We will be using the `bnlearn` and `gRain` packages in R.

1. Install `bnlearn` by following the instructions at http://www.bnlearn.com/. From CRAN you will be installing `bnlearn`. From Bioconductor you will be installing `graph`, `Rgraphviz`, and `RBGL`.
2. Install `gRain`.

Here is the same calculation using the `bnlearn` package in R [12].

```
library(bnlearn)
dag <- empty.graph(nodes = c("H", "T"))   # create nodes
dag <- set.arc(dag, from = "H", to = "T") # link nodes
graphviz.plot(dag, layout = "circo")
```




```
H.lv <- c("Pos", "Neg") ## create levels
T.lv <- c("Pos", "Neg")
#### create conditional probability tables
H.prob <- array(c(prior, 1-prior), dim = 2, dimnames = list(H = H.lv))
T.prob <- array(c(sens, 1-sens, 1-spec, spec), dim = c(2, 2),
  dimnames = list(T.lv, H.lv))
cpt <- list(H = H.prob, T = T.prob)
bn <- custom.fit(dag, cpt)
```

We have captured this BN as an "expert knowledge system" in the R object `bn` which we can now query.

```
names(bn)
```

```
#> [1] "H" "T"
```

```
bn$T
```

```
#>
#>   Parameters of node T (multinomial distribution)
#>
```

```
#> Conditional probability table:
#>
#>      H
#> T       Pos   Neg
#>   Pos 0.999 0.002
#>   Neg 0.001 0.998
```

With BNs we can can ask "What if?" questions. For example, we can ask what is the probability of HIV infection given a positive test? Again, $P(H+ \mid T+)$ is the positive predictive value. We have the choice between two R packages:

1. `bnlearn` for approximate inference
2. `gRain` for exact inference

3.2.2 Approximate inference with `bnlearn` (Monte Carlo simulation)

We can answer what is the positive predictive value (PPV); that is, what is $P(HIV = \text{pos} \mid Test = \text{pos})$?

```
(ppv1 <- cpquery(bn, event = (H == "Pos"), evidence = (T == "Pos"),
                 n = 10^6))
```

```
#> [1] 0.004882812
```

Notice that the `cpquery` function has a very intuitive syntax. In `bnlearn` approximate inference is accomplished using Monte Carlo simulation. This scales efficiently to very large, complex BNs.

3.2.2.1 Calculating PPV with new prior probability: Remember, for the query above, the prior probability was 0.00001. What if the prior probability of HIV infection was 0.1. How does the PPV change.

In the `bn` object we need to change the probabilities for node H.

```
bn$H$prob # view prior probabilities that we need to change
```

```
#> H
#>     Pos     Neg
#> 0.00001 0.99999
```

```
bn_newprior <- bn # make copy of BN model object
bn_newprior$H <- array(c(.1,.9), dim = dim(bn$H$prob),
                       dimnames = dimnames(bn$H$prob)) # assign new probs
bn_newprior$H$prob # confirm change to new probabilities
```

```
#> H
#> Pos Neg
#> 0.1 0.9
```

```
#### Calculate PPV w/ new prior of 0.1 and 'bn_newprior'
(ppv2 <- cpquery(bn_newprior, event = (H == "Pos"),
                 evidence = (T == "Pos"), n = 10^6))
```

```
#> [1] 0.9823476
```

For prior of 0.00001, the $PPV = 0.0048828$. In constrast, for prior of 0.1, the $PPV = 0.9823476$.

3.2.3 Exact inference with gRain

The gRain package calculates exact probabilities but is more computationally expensive for large BNs. Also the syntax is less intuitive compared to bnlearn. Notice that in our example below gRain is using the bn object we compiled using bnlearn above.

```
library(gRain) # for exact inference
junction <- compile(as.grain(bn)) # `bn` object is from `bnlearn` output
jtest <- setEvidence(junction, nodes = "T", states = "Pos")
querygrain(jtest, nodes = "H")$H # Positive Predictive Value (PPV)
```

```
#> H
#>         Pos         Neg
#> 0.004970223 0.995029777
```

3.3 Example 2: Evaluating respiratory diseases

For example, Figure 8 is from Neapolitan (p. 491, [14]) and depicts a BN representing relationships among respiratory disease variables.

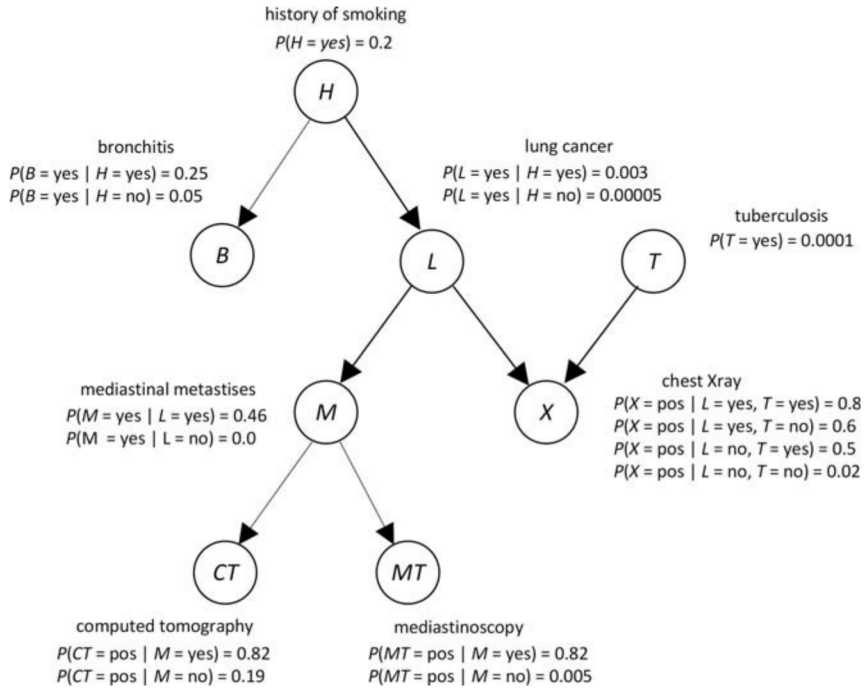
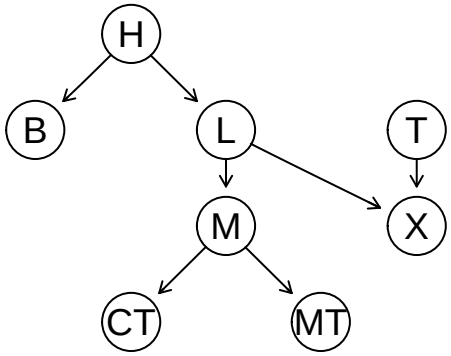


Figure 8. A Bayesian network of relationships among respiratory disease variables

```
dag2 <- empty.graph(nodes=c("H", "B", "L", "T", "M", "X", "CT", "MT"))
dag2 <- set.arc(dag2, from = "H", to = "B")
dag2 <- set.arc(dag2, from = "H", to = "L")
dag2 <- set.arc(dag2, from = "L", to = "M")
dag2 <- set.arc(dag2, from = "L", to = "X")
dag2 <- set.arc(dag2, from = "T", to = "X")
```

```
dag2 <- set.arc(dag2, from = "M", to = "CT")
dag2 <- set.arc(dag2, from = "M", to = "MT")
graphviz.plot(dag2)
```




```
#### create levels
yn <- c("Yes", "No"); pn <- c("Pos", "Neg")
H.lv <- yn; B.lv <- yn; L.lv <- yn; T.lv <- yn; M.lv <- yn
X.lv <- pn; CT.lv <- pn; MT.lv <- pn
#### create conditional probability tables
H.prob <- array(c(0.2,1-0.2), dim = 2, dimnames = list(H = H.lv))
B.prob <- array(c(0.25,1-0.25,0.05,1-0.05), dim=c(2,2),
  dimnames = list(B=B.lv, H=H.lv))
L.prob <- array(c(0.003,1-0.003,0.00005,1-0.00005), dim=c(2,2),
  dimnames = list(L=L.lv, H=H.lv))
T.prob <- array(c(0.0001,1-0.0001), dim=2, dimnames=list(T = T.lv))
M.prob <- array(c(0.46,1-0.46,0,1 - 0), dim=c(2,2),
  dimnames = list(M=M.lv, L=L.lv))
X.prob <- array(c(0.8,1-0.8,0.6,1-0.6,0.5,1-0.5,0.02,1-0.02),
  dim=c(2,2,2), dimnames = list(X=X.lv, T=T.lv, L=L.lv))
CT.prob <- array(c(0.82,1-0.82,0.19,1-0.19), dim=c(2,2),
  dimnames = list(CT=CT.lv, M=M.lv))
MT.prob <- array(c(0.82,1-0.82,0.005,1-0.005), dim=c(2,2),
  dimnames = list(MT=MT.lv, M=M.lv))
cpt <- list(H=H.prob, B=B.prob, L=L.prob, T=T.prob,
            M=M.prob, X=X.prob, CT=CT.prob, MT=MT.prob)
bn2 <- custom.fit(dag2, cpt)
```

Creating the conditional probability tables was straight forward. In the Neapolitan paper the authors ask: "if a patient has a smoking history (H = yes), a positive chest X-ray (X = pos), and a positive computer tomogram (CT = pos), we can determine the probability of the patient having lung cancer (L = yes)." That is, what is $P(L = \text{Yes} \mid H = \text{Yes}, X = \text{Pos}, CT = \text{Pos})$?

3.3.1 Approximate inference with `bnlearn` (Monte Carlo simulation)

```
cpquery(bn2, event = (L == "Yes"), evidence = (H == "Yes") & (X=="Pos")
  & (CT=="Pos"), n = 10^6)
```

```
#> [1] 0.1926007
```

3.3.2 Exact inference with gRain

```
junction2 <- compile(as.grain(bn2))
jriskfactors <- setEvidence(junction2, nodes = c("H","X","CT"),
  states = c("Yes","Pos","Pos"))
(p <- querygrain(jriskfactors, nodes = "L")$L) # Pos. Predictive Value
```

```
#> L
#>       Yes        No
#> 0.1852825 0.8147175
```

The $P(L = \text{Yes} \mid H = \text{Yes}, X = \text{Pos}, CT = \text{Pos}) = 0.1852825$. The prior probability of lung cancer in this model is 0.0064. So, the evidence has increased the probability of lung cancer substantially.

For practice, how did we determine the prior probability of lung cancer? For this query I do not need to set evidence, only query the marginal probability of L (lung cancer) from the fitted BN.

```
querygrain(junction2, nodes = c("L"), type = "marginal")$L
```

```
#> L
#>     Yes      No
#> 0.00064 0.99936
```

3.4 Example 3: Fitting Bayesian network to data

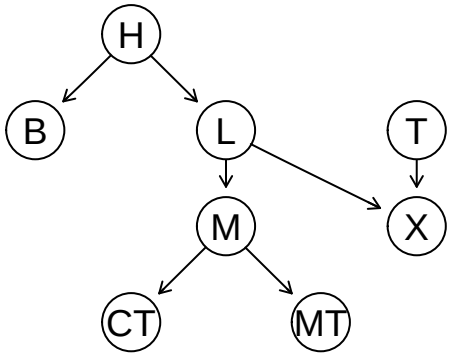
An alternative method to build a BN is by fitting the model to data. We will read the respiratory disease data frame with 20,000 observations. The key tip to remember is that the BN node names must match the data frame variable names exactly.

```
#### Read respiratory disease data
rurl <- 'https://raw.githubusercontent.com/taragonmd/data/master/resp_dis.csv'
rd <- read.csv(rurl, header = TRUE)
str(rd)
```

```
#> 'data.frame':    20000 obs. of  8 variables:
#>  $ H : Factor w/ 2 levels "No","Yes": 2 1 2 1 1 1 1 1 1 2 ...
#>  $ B : Factor w/ 2 levels "No","Yes": 2 1 1 1 1 1 1 1 1 1 ...
#>  $ L : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
#>  $ T : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
#>  $ M : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
#>  $ X : Factor w/ 2 levels "Neg","Pos": 1 1 1 1 1 1 1 1 1 1 ...
#>  $ CT: Factor w/ 2 levels "Neg","Pos": 1 1 1 1 1 1 1 2 1 1 ...
#>  $ MT: Factor w/ 2 levels "Neg","Pos": 1 1 1 1 1 1 1 1 1 1 ...
```

```
#### Build the Bayesian network
dag3 <- empty.graph(nodes=c("H", "B", "L", "T", "M", "X", "CT", "MT"))
dag3 <- set.arc(dag3, from = "H", to = "B")
dag3 <- set.arc(dag3, from = "H", to = "L")
dag3 <- set.arc(dag3, from = "L", to = "M")
```

```
dag3 <- set.arc(dag3, from = "L", to = "X")
dag3 <- set.arc(dag3, from = "T", to = "X")
dag3 <- set.arc(dag3, from = "M", to = "CT")
dag3 <- set.arc(dag3, from = "M", to = "MT")
graphviz.plot(dag3)              # Plot the BN
```




```
bn3 <- bn.fit(dag3, data = rd) # Fit the BN to the data
```

We can query the fitted BN, what is $P(L = \text{Yes} \mid H = \text{Yes}, X = \text{Pos}, CT = \text{Pos})$?

```
cpquery(bn3, event = (L == "Yes"), evidence = (H == "Yes") & (X=="Pos")
  & (CT=="Pos"), n = 10^6)
```

```
#> [1] 0.1559724
```

4 Causal inference (with causal Bayesian networks)

4.1 Directed acyclic graph (DAG)

Causal BNs are directed acyclic graphs (DAGs) (also called causal graphs) [15–17]. All of the probabalistic reasoning concepts we learned with BNs continue to apply. With BNs the causal links are

1. *assumptions* based on expert knowledge,
2. *evidenced-based* from scientific research, or
3. *conjecture* to gain insights of new assumptions.

Causal inference is drawing valid, unbiased conclusions about cause-effect relationships. In causal inference we set out to

1. *discover* new causal pathways *or* models,
2. *test* causal hypotheses,
3. *estimate* causal effects.

Consider two variables (nodes), X and Y. What can explain an association (correlation) between X and Y?

1. random chance,
2. X caused Y ($X \rightarrow Y$),
3. Y caused X ($X \leftarrow Y$),
4. X and Y share a common cause ($X \leftarrow Z \rightarrow Y$), or
5. X and Y share a common effect, Z, which is conditioned on ($X \rightarrow \boxed{Z} \leftarrow Y$; which is called collider bias).

By design, DAGs are not causal loops. Causal loops have an important role in systems thinking and modeling but will not be discussed further. Statistical inference supports causal inference with quantitative methods for estimation, chance, and bias. A common cause involves three or more nodes and is discussed next.

Figure 9 displays the core DAG patterns of three nodes with two causal links. In a *chain* ($X \rightarrow Y \rightarrow Z$), also called a *sequential cause*, X and Z are *unconditionally dependent*. This means that X and Z are dependent without conditioning on any variable. Likewise, in a *fork* ($Y \leftarrow X \rightarrow Z$), also called a *common cause*, X and Z are *unconditionally dependent*. In epidemiology, forks are the principle cause of **confounding**.

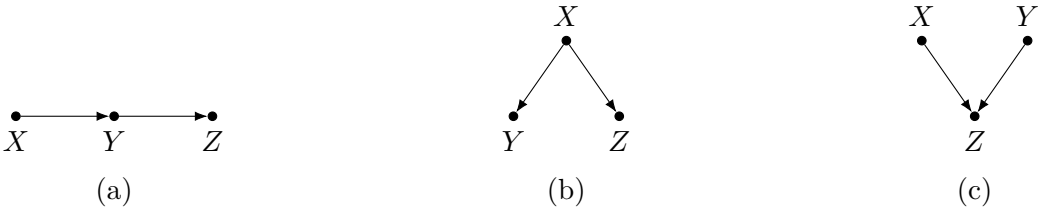


Figure 9. Core DAG patterns for three nodes and two edges: (a) chain (sequential cause), (b) fork (common cause), and (c) collider (common effect).

Chains and forks make intuitive sense, colliders do not. In a *collider* ($X \rightarrow Z \leftarrow Y$), also called a *common effect*, X and Y are *unconditionally independent*. However, when we condition on Z (or any descendent of Z), X and Y become *conditionally dependent*. Epidemiologists specialize in "adjusting for potential confounders." When we condition ("adjust") on a collider we introduce a spurious association—and worse—might conclude that the association is causal—which is impossible because X and Y are unconditionally independent! In other words, we *introduce confounding* where none existed! This is considered *epidemiologic malpractice!*[8]

Here is the classic example of *collider bias*. We flip a fair coin twice $\{0 = \text{tail}, 1 = \text{head}\}$. T_1 is the outcome of the first coin flip $\{0, 1\}$; T_2 is the outcome of the second coin flip $\{0, 1\}$; and S is sum of T_1 and T_2 $\{0, 1, 2\}$. Knowing the value of T_1 tells us absolutely nothing about the value of T_2, and vice versa. They are completely independent (represented by no solid edge in the DAG).

However, if we are told the value of S (say, 1) (this is "conditioning"), then T_1 and T_2 are now dependent (Figure 10). If $T_1 = 0$, then we know the value of T_2 must be 1. If $T_1 = 1$, then we know the value of T_2 must be 0. The reverse is true: knowing T_2 informs us of the value of T_1. For an epidemiologic example see Cole [18].

Our motivation for introducing DAGs is to emphasize that our causal and probabilistic reasoning is very vulnerable—even when we have 'lots of data! Do your analysts understand colliders and their perils? If not, why not?

[8] Or *statistical malpractice* if you are a statistician.

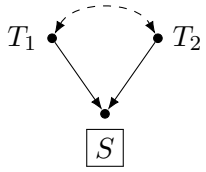


Figure 10. Collider bias when flipping two fair coins. Conditioning on collider S introduces a dependency (dashed edge) between T_1 and T_2.

4.2 Deconfounding (controlling for confounding)

Remember, while causal arrows are unidirectional, probabilistic dependence propagates in both directions. By conditioning on a variable we block this propagation unless it is a collider. In an observation study A is either a treatment or exposure, and Y is the outcome or effect. If we are interested in measuring the causal effect of A on Y we want to block all pathways that have arrows pointing into A (backdoor) and that have arrows pointing into Y, but that are not descendents of A (frontdoor).

Figure 11 depicts four DAGs and the possible combinations of nodes that can be chosen to block the backdoor pathway. The backdoor pathways are pathways that point into X *and* connect to Y *and* do not have a collider. Colliders block propagation unless we condition on them. If you condition on a collider you must also condition on another variable to block the new pathway the conditioned collider just opened.

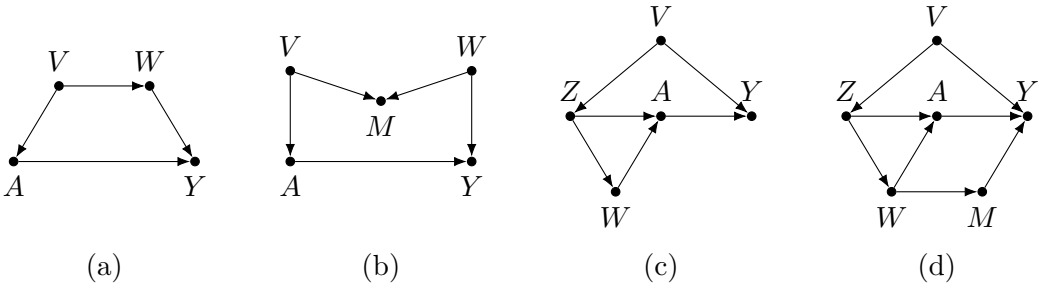


Figure 11. In an observational study, A is either a treatment or exposure, and Y is the outcome or effect. To measure the causal effect of A on Y we must block backdoor pathways from A to Y by conditioning on a set of correct variables. We do not want to condition on descedents of A, including intermediate nodes between A and Y. We do not want to condition on colliders alone otherwize we introduce spurious associations. For (a) we can condition on $\{V\}$, $\{W\}$, or $\{V, W\}$. For (b) we can condition on $\{\}$, $\{V\}$, $\{W\}$, $\{M, W\}$, $\{M, V\}$, $\{M, V, W\}$, but not on $\{M\}$ (collider bias). For (c) we can condition on $\{V\}$, $\{V, Z\}$, $\{Z, W\}$, $\{V, Z, W\}$, but not on $\{Z\}$ alone (collider bias), and not on $\{W\}$ (leaves other path open). Finally for (d) we can condition on $\{W, Z\}$, $\{W, V\}$, $\{M, Z\}$, $\{M, V\}$, $\{W, Z, V\}$, $\{M, Z, V\}$, $\{W, M, Z\}$, $\{W, M, V\}$, or $\{W, M, Z, V\}$.

In Figure 11(b), conditioning on M (a collider) opens the path from A to Y by creating a dependency between V and W. Figure 12 illustrates this new path which is called M-bias. To the untrained analyst M behaves like a confounder because V

and W are common causes (forks) that create an association between M and A, and between M and Y. Without DAGs to guide us, controlling for confounders is like flying blind: you are bound to get into trouble (i.e., introduce bias).

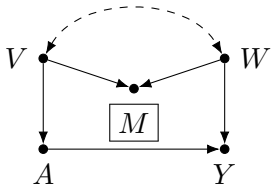


Figure 12. Conditioning on M, a collider, introduces M-bias.

By deconfounding we mean estimating a specific causal effect while controlling for confounding, and without introducing bias (e.g., collifer bias). In general, we have three methodologic approaches:

1. Backdoor criterion (our primary focus)
2. Frontdoor criterion
3. Instrumental variable

For all of these we need to have a full DAG for the process we are studying. This requires reviewing the scientific literature and collaborating with subject matter experts (SMEs). Having a complete DAG does not mean we have data on every variable. We may not need it depending on which variables we select for adjustment. Hence, DAGs are also important for study and analysis design.

4.3 Backdoor criterion

DAGs are great because they are valid for whatever functional forms that connect the variables (e.g., linear vs. nonlinear). So far, we have selected a set of variables that will block the backdoor path. This approach is called the **backdoor criterion**:

1. block all spurious paths between A and Y,
2. leave all directed paths from A to Y unperturbed, and
3. create no new spurious paths.

Now we need a calculation formula. Pearl developed *do*-calculus as a method to derive this formula. The thinking goes like this: if I disconnect all backdoor arrows into A ("graph surgery") and set $X = x$ for *everyone* in the population, then I get the modified DAG in Figure 13(b). This $do(A = a)$ is a hypothetical intervention. The magic of *do*-calculus is to derive a formula that only has variable terms representing the observation data from Figure 13(a).

Using *do*-calculus Pearl derived this **backdoor adjustment formula** [16]:

$$\begin{aligned} P(y \mid do(x)) &= \sum_x P(Y = y \mid A = a, X = x) P(X = x) \\ &= \sum_x \frac{P(Y = y, A = a, X = x)}{P(A = a \mid X = x)} \end{aligned}$$

In this formula, the joint probability, $P(y, a, x)$ is weighted by the term $1/P(a \mid x)$, also called *inverse probability weighting* When the denominator term is used to

Figure 13. Backdoor criterion: (a) unmodified causal graph where A affects Y, and X represents the covariate set of variables selected for adjustment; (b) modified causal graph where for everyone in the population X is set to x. This is *do*-calculus, and is used to derive an adjustment formula.

calculate the probability of treatment ($A = 1$) given covariate set X it is called the *propensity score* or $P(A = 1 \mid X)$

Now we can calculate the **average causal effect** (ACE) or *causal risk difference*:

$$\text{causal RD} = P(Y = 1 \mid do(A = 1)) - P(Y = 1 \mid do(A = 0))$$

And the *causal risk ratio* is

$$\text{causal RR} = \frac{P(Y = 1 \mid do(A = 1))}{P(Y = 1 \mid do(A = 0))}$$

And the *causal odds ratio* is

$$\text{causal OR} = \frac{P(Y = 1 \mid do(A = 1))/\{1 - P(Y = 1 \mid do(A = 1))\}}{P(Y = 1 \mid do(A = 0))/\{1 - P(Y = 1 \mid do(A = 0))\}}$$

Our goal is to measure the causal effect of A on Y. Our DAG guided the selection of a set of variables (X) that will block the backdoor path. We see that the backdoor adjustment formula is the weighted sum $P(y, a, x)$ weighted by $1/P(a \mid x)$, the inverse probability weigtht.

Let's briefly review the general methods for deconfounding [17,19]:

1. **Randomization**
2. **Generalized (G) methods**
   - Inverse probability weighting
   - Standardization (g-formula)
   - G-estimation
3. **Stratification methods**
   - Restriction
   - Matching (on covariates or propensity score)
   - Weighted average (e.g., Mantel-Haenszel methods)
   - Regression
4. **Quasi-experimental** methods
   - Natural experiments
   - Instrumental variables
   - Differences in differences
   - Regression discontinuity

Our focus is obervational studies, so we will not review randomization. The G-methods estimate the **average causal effect** on the entire **synthetic study population** that has been deconfounded. The advantage of a synthetic study population is that it can be analyzed as if it were a randomized controlled trial. However, special attention is required for calculating valid confidence intervals.

In contrast, stratification methods estimate the **average causal effect of treatment on the treated** (ATT) (or the average causal effect of exposure on the exposed). These traditional methods are familiar to most epidemiologists not trained in G-methods. Next, we cover a selection of methods.

4.3.1 Inverse probability weighting

4.3.1.1 IPW with stratified contingency table: Consider an observational study where 700 patients were given access to a new drug for an ailment (Table 6) [16]. A total of 350 patients chose to take the drug and 350 patients did not. From previous studies we know that estrogen has a negative effect on recovery, and women are more likely to take the drug compared to men.

Table 6. Recovery outcomes of 700 patients given access to a new drug, Stratified by sex

	Men		**Women**		**Combined**	
	Drug	No drug	Drug	No drug	Drug	No drug
Recovered	81	234	192	55	273	289
No recovery	6	36	71	25	77	61
Total at risk	87	270	263	80	350	350

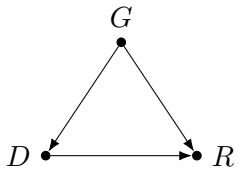


Figure 14. Directed acyclic graph for Table 6. D = drug; R = recovery; and G = gender

Using the backdoor adjustment formula for this DAG (Figure 14), let's estimate the probability of recovery (R), if the entire study population receives the drug ($D = 1$), deconfounded for gender (G); and if the entire study population does not receive the drug ($D = 0$), deconfounded for gender (G).

$$P(R = 1 \mid do(D = 1)) = \sum_x \frac{P(R = 1, D = 1, X = x)}{P(D = 1 \mid G = g)} = r_0$$

$$P(R = 1 \mid do(D = 0)) = \sum_x \frac{P(R = 1, D = 0, X = x)}{P(D = 0 \mid G = g)} = r_1$$

We can then calculate average causal effect as either the

- causal risk difference,

- causal risk ratio, or
- causal odds ratio.

Using the equations above, we use R code to calculate a vector of average causal risks:

$$[r_0, r_1]$$

```
durl <- 'https://raw.githubusercontent.com/taragonmd/data/master/drugrx-pearl2.csv'
jd <- read.csv(durl, header = TRUE)
names(jd) <- c('id', 'R', 'D', 'G')
tab_rdg <- xtabs(~ R + D + G, data = jd)
tab_dg <- apply(tab_rdg, c(2, 3), sum)
(jprob_r1dg <- prop.table(tab_rdg)['Yes',,]) # P(R = yes, D = d, G = g)
```

```
#>      G
#> D            Men      Women
#>   No  0.33428571 0.07857143
#>   Yes 0.11571429 0.27428571
```

```
(cprob_dg <- prop.table(tab_dg, 2))              # P(d | g)
```

```
#>      G
#> D           Men     Women
#>   No  0.7563025 0.2332362
#>   Yes 0.2436975 0.7667638
```

```
(jprob_r1dg_ipw <- jprob_r1dg/cprob_dg)          # IPW matrix
```

```
#>      G
#> D           Men     Women
#>   No  0.4420000 0.3368750
#>   Yes 0.4748276 0.3577186
```

```
(cprob_r1d_ipw <- apply(jprob_r1dg_ipw, 1, sum)) # summation over G
```

```
#>        No       Yes
#> 0.7788750 0.8325462
```

```
r0 <- unname(cprob_r1d_ipw['No'])              # P(R = yes | D = no)
r1 <- unname(cprob_r1d_ipw['Yes'])             # P(R = yes | D = yes)
#### Average causal effects (ACEs)
c(causal_RD=r1-r0, causal_RR=r1/r0, causal_OR=(r1/(1-r1))/(r0/(1-r0)))
```

```
#>  causal_RD  causal_RR  causal_OR
#> 0.05367122 1.06890864 1.41150841
```

Here is a slightly different approach that calculates the same results but creates the `IPW array` that allows us to generate a **IPW synthetic population** that is deconfounded by removing the arrow into D (treatment).

```
jprob_rdg <- prop.table(tab_rdg)  # P(r, d, g)
cprob_dg <- prop.table(tab_dg, 2) # P(d | g)
jprob_rdg_ipw <- sweep(jprob_rdg, c(2,3), cprob_dg, '/') # IPW array
cprob_rd_ipw <- apply(jprob_rdg_ipw, c(1,2), sum)        # sum over G
```

```
r0 <- cprob_rd_ipw['Yes','No']  # P(R = yes | D = no)
r1 <- cprob_rd_ipw['Yes','Yes'] # P(R = yes | D = yes)
#### Average causal effects (ACEs)
c(causal_RD=r1-r0, causal_RR=r1/r0, causal_OR=(r1/(1-r1))/(r0/(1-r0)))
```

```
#>  causal_RD  causal_RR  causal_OR
#> 0.05367122 1.06890864 1.41150841
```

To retrieve the original population we multiple the joint probability by 700 (the total study population). To view the synthetic population we multiple the IPW joint probability by 700 (the total study population). In the R code below we create a data frame to display both the original and IPW synthetic study populations.

```
df_rdg_both <- as.data.frame(jprob_rdg*700)      # temp; original
df_rdg_ipw2  <- as.data.frame(jprob_rdg_ipw*700) # IPW synthetic pop
df_rdg_both$Freq_IPW <- df_rdg_ipw2$Freq # final # combine both
df_rdg_both                                      # display
```

```
#>     R   D     G Freq  Freq_IPW
#> 1  No  No   Men   36  47.60000
#> 2 Yes  No   Men  234 309.40000
#> 3  No Yes   Men    6  24.62069
#> 4 Yes Yes   Men   81 332.37931
#> 5  No  No Women   25 107.18750
#> 6 Yes  No Women   55 235.81250
#> 7  No Yes Women   71  92.59696
#> 8 Yes Yes Women  192 250.40304
```

In fact, to prove that the synthetic population is deconfounded we can analyze the new data and show that the back-door is blocked; that is, there is no association between Drug (treatment or exposure) and Gender (confounder).

```
ipwpop_rdg <- jprob_rdg_ipw*700 # create stratified synthetic table
(ipwpop_dg <- apply(ipwpop_rdg, c(2,3), sum)) # Drug vs Gender (no assoc)
```

```
#>      G
#> D     Men Women
#>   No  357   343
#>   Yes 357   343
```

```
ft <- fisher.test(ipwpop_dg) # no assoc betw D and G (i.e., deconfounded)
c(ft$estimate, p.value = ft$p.value)
```

```
#> odds ratio    p.value
#>          1          1
```

```
(ipwpop_rd <- apply(ipwpop_rdg, c(1,2), sum)) # R vs D (crude table)
```

```
#>      D
#> R           No      Yes
#>   No  154.7875 117.2176
#>   Yes 545.2125 582.7824
```

```
(risk <- prop.table(ipwpop_rd, 2))
```

```
#>      D
#> R           No       Yes
#>   No  0.221125 0.1674538
#>   Yes 0.778875 0.8325462
```

```
r0 <- risk['Yes', 'No']  # P(R = yes | D = no)
r1 <- risk['Yes', 'Yes'] # P(R = yes | D = yes)
#### Average causal effects (ACEs)
c(causal_RD=r1-r0, causal_RR=r1/r0, causal_OR=(r1/(1-r1))/(r0/(1-r0)))
```

```
#>  causal_RD  causal_RR  causal_OR
#> 0.05367122 1.06890864 1.41150841
```

The is a *marginal* (crude table) analysis of the IPW sythetic population. In the next section we repeat IPW with a generalized linear model (GLM) which is called a *marginal structural model* (MSM). By "marginal" we measure the average causal effect in the population without conditioning on any variables (e.g., confounders). By "structural" we mean modeling potential outcomes, not observed outcomes.

4.3.1.2 IPW with marginal structural model (MSM):

The general MSM is represented by

$$g\{E(Y^a \mid V)\} = h(a, V; \psi),$$

where h is some parametric function, usually linear and additive; and g is a link function (log, identity, logit) in a generalized linear model for estimating

- causal risk difference,
- causal risk ratio, or
- causal odds ratio.

V represents additional variables that might be used, for example, for evaluating effect modification. For our purposes, we will ignore V.

Recall that the IPW is related to the propensity score. The IPW for the i-th treated (or exposed) subject is

$$W_i = \frac{1}{P(A = 1 \mid X_i)} = \frac{1}{\text{propensity score}},$$

and the IPW for the i-th untreated (or non-exposed) subject is

$$W_i = \frac{1}{P(A = 0 \mid X_i)} = \frac{1}{1 - \text{propensity score}},$$

where A = treatment (or exposure) and X_i is the set of variables ("covariates") that will be used for deconfounding. In our example, A is drug (D) and X is gender (G).

We have everything we need to estimate ACEs with IPW/MSM. Here are the general steps:

1. Estimate propensity scores
2. Create inverse probability weights
3. Specify the MSM of interest (causal RR, OR, or RD)

4. Fit IPW generalized linear model
5. Test for deconfounding; i.e., confounder(s) independent of treatment (exposure)
6. Use asymptotic (sandwich) variance estimator (or bootstrapping) to account for artificial "sample size" of synthetic population.

We will analyze the same data from Table 6. A key feature of the analyses that follow is the display of "Table 1" using the `tableone` R package. Once we have deconfounded the back-door path, by any method (IPW, matching, etc.), then the confounding variables will not be associated with the treatment (exposure) variable. In other words, their distribution will be balanced across treatment (exposure) levels. To assess balance we use the **standardized mean distance** (SMD).

$$SMD = (\bar{x}_{A=1} - \bar{x}_{A=0}) \,/\, \left(\sqrt{\frac{s^2_{A=1} + s^2_{A=0}}{2}} \right)$$

Deconfounding means achieving balance; that is, the covariates are independent from treatment (or exposure) level. The $SMD < 0.1$ indicates balance. With respect to causal inference, achieving this conditional independence is called achieving *exchangeability* (or conditional exchangeability) or achieving *ignorability* [19].

Finally, because we are generating an IPW synthetic population we will need special attention in calculating appropiate confidence intervals. We will use R's `sandwich` package for robust variance estimation.

```
library(tableone)
library(sandwich) #for robust variance estimation
library(survey)

jd2 <-jd # copy original data set with factors
jd2$R <- as.numeric(jd2$R=='Yes')   # recode yes = 1, no = 0
jd2$D <- as.numeric(jd2$D=='Yes')   # recode yes = 1, no = 0
jd2$G <- as.numeric(jd2$G=='Women') # recode women = 1, men = 0

#### look at a table 1
table1 <- CreateTableOne(vars = "G", strata = "D",
                         data = jd2, test = FALSE)
#### include standardized mean difference (SMD)
print(table1,smd=TRUE)
```

```
#>                Stratified by D
#>                 0           1           SMD
#>   n               350         350
#>   G (mean (SD)) 0.23 (0.42) 0.75 (0.43)  1.225
```

```
#### propensity score model
psmodel <- glm(D ~ G, family  = binomial(link ="logit"), data = jd2)

#### value of propensity score for each subject
## ps <- psmodel$fitted.values # alternative
ps <- predict(psmodel, type = "response")
```

```
#### create IP weights
jd2$wt <- ifelse(jd2$D==1, 1/(ps), 1/(1-ps))

#### apply weights to data
weighteddata <- svydesign(ids = ~ 1, data =jd2, weights = ~ wt)

#### weighted table 1
weightedtable <- svyCreateTableOne(vars = "G", strata = "D",
                                   data = weighteddata, test = FALSE)
#### Show table with SMD
print(weightedtable, smd = TRUE)
```

```
#>                Stratified by D
#>                 0              1              SMD
#>   n             700.00         700.00
#>   G (mean (SD))   0.49 (0.50)    0.49 (0.50) <0.001
```

```
#### Causal Risk Difference with weigthed GLM
glm_obj <- glm(R ~ D, weights = wt, family =
               quasibinomial(link="identity"), data = jd2)
beta_ipw <- unname(coef(glm_obj))
SE <- unname(sqrt(diag(vcovHC(glm_obj, type="HC0"))))

#### get point estimate and CI for risk difference
cRD <- beta_ipw[2]
conf_level = 0.95
Z <- qnorm((1 + conf_level)/2)
cint <- beta_ipw[2] + c(-1,1) *  Z * SE[2]
list(causal_RD = cRD, conf_int = cint, conf_level = conf_level )
```

```
#> $causal_RD
#> [1] 0.05367122
#>
#> $conf_int
#> [1] -0.01419789  0.12154033
#>
#> $conf_level
#> [1] 0.95
```

```
#### Causal Risk Ratio with weighted GLM
glm_obj <- glm(R ~ D, weights = wt, family =
               quasibinomial(link = log), data = jd2)
beta_ipw <- unname(coef(glm_obj))

#### To account for weighting, use asymptotic (sandwich) variance
SE <- unname(sqrt(diag(vcovHC(glm_obj, type="HC0"))))
```

```
#### get point estimate and CI for risk ratio
cRR <- exp(beta_ipw[2])
conf_level = 0.95
Z <- qnorm((1 + conf_level)/2)
cint <- exp(beta_ipw[2] + c(-1,1) *  Z * SE[2])
list(causal_RR = cRR, conf_int = cint, conf_level = conf_level)
```

```
#> $causal_RR
#> [1] 1.068909
#>
#> $conf_int
#> [1] 0.9815544 1.1640371
#>
#> $conf_level
#> [1] 0.95
```

```
#### Causal Odds Ratio with weighted GLM
glm_obj <- glm(R ~ D, weights = wt, family =
               quasibinomial(link = logit), data = jd2)
beta_ipw <- unname(coef(glm_obj))

#### To account for weighting, use asymptotic (sandwich) variance
SE <- unname(sqrt(diag(vcovHC(glm_obj, type="HC0"))))

#### get point estimate and CI for odds ratio
cOR <- exp(beta_ipw[2])
conf_level = 0.95
Z <- qnorm((1 + conf_level)/2)
cint <- exp(beta_ipw[2] + c(-1,1) *  Z * SE[2])
list(causal_OR = cOR, conf_int = cint, conf_level = conf_level)
```

```
#> $causal_OR
#> [1] 1.411508
#>
#> $conf_int
#> [1] 0.9199587 2.1657016
#>
#> $conf_level
#> [1] 0.95
```

4.3.1.3 IPW/MSM with multiple confounders ("covariates"): The following analysis is with data from a hospital-based observational study of right heart catherization (RHC). A description of the data set is available from http://biostat.mc.vanderbilt.edu/wiki/pub/Main/DataSets/rhc.html.

```
#### load packages
library(tableone)
library(sandwich) #for robust variance estimation
library(survey)
```

```r
#read in data
load(url("http://biostat.mc.vanderbilt.edu/wiki/pub/Main/DataSets/rhc.sav"))

#### create smaller data set and binary (0, 1) X variables
ARF <- as.numeric(rhc$cat1=='ARF')
CHF <- as.numeric(rhc$cat1=='CHF')
Cirr <- as.numeric(rhc$cat1=='Cirrhosis')
colcan <- as.numeric(rhc$cat1=='Colon Cancer')
Coma <- as.numeric(rhc$cat1=='Coma')
COPD <- as.numeric(rhc$cat1=='COPD')
lungcan <- as.numeric(rhc$cat1=='Lung Cancer')
MOSF <- as.numeric(rhc$cat1=='MOSF w/Malignancy')
sepsis <- as.numeric(rhc$cat1=='MOSF w/Sepsis')
female <- as.numeric(rhc$sex=='Female')
died <- as.numeric(rhc$death=='Yes')
age <- rhc$age
treatment <- as.numeric(rhc$swang1=='RHC')
meanbp1 <- rhc$meanbp1

#### new dataset
mydata <- cbind(ARF, CHF, Cirr, colcan, Coma, lungcan, MOSF,
                sepsis, age, female, meanbp1, treatment, died)
mydata <- data.frame(mydata)

#### covariate set
xvars <- c("ARF", "CHF", "Cirr", "colcan", "Coma", "lungcan", "MOSF",
           "sepsis", "age","female","meanbp1")

#### look at a table 1
table1 <- CreateTableOne(vars = xvars, strata = "treatment",
                         data = mydata, test = FALSE)

#### include standardized mean difference (SMD)
print(table1,smd=TRUE)
```

```
#>                       Stratified by treatment
#>                        0              1              SMD
#>   n                     3551           2184
#>   ARF (mean (SD))       0.45 (0.50)    0.42 (0.49)    0.059
#>   CHF (mean (SD))       0.07 (0.25)    0.10 (0.29)    0.095
#>   Cirr (mean (SD))      0.05 (0.22)    0.02 (0.15)    0.145
#>   colcan (mean (SD))    0.00 (0.04)    0.00 (0.02)    0.038
#>   Coma (mean (SD))      0.10 (0.29)    0.04 (0.20)    0.207
#>   lungcan (mean (SD))   0.01 (0.10)    0.00 (0.05)    0.095
#>   MOSF (mean (SD))      0.07 (0.25)    0.07 (0.26)    0.018
#>   sepsis (mean (SD))    0.15 (0.36)    0.32 (0.47)    0.415
```

```
#>   age (mean (SD))     61.76 (17.29) 60.75 (15.63)  0.061
#>   female (mean (SD))   0.46 (0.50)   0.41 (0.49)   0.093
#>   meanbp1 (mean (SD)) 84.87 (38.87) 68.20 (34.24)  0.455
```

```
#### propensity score model
psmodel <- glm(treatment ~ age + female + meanbp1+ARF+CHF+Cirr+colcan+
         Coma+lungcan+MOSF+sepsis, family  = binomial(link ="logit"))

#### value of propensity score for each subject
## ps <- psmodel$fitted.values # alternative
ps <- predict(psmodel, type = "response")

#### create IP weights
weight <- ifelse(treatment==1,1/(ps),1/(1-ps))

#### apply weights to data
weighteddata<-svydesign(ids = ~ 1, data =mydata, weights = ~ weight)

#### weighted table 1
weightedtable <-svyCreateTableOne(vars = xvars, strata = "treatment",
                                  data = weighteddata, test = FALSE)
#### Show table with SMD
print(weightedtable, smd = TRUE)
```

```
#>                       Stratified by treatment
#>                        0               1               SMD
#>   n                    5732.49         5744.88
#>   ARF (mean (SD))        0.44 (0.50)     0.44 (0.50)   0.010
#>   CHF (mean (SD))        0.08 (0.27)     0.08 (0.27)   0.005
#>   Cirr (mean (SD))       0.04 (0.19)     0.04 (0.19)   0.001
#>   colcan (mean (SD))     0.00 (0.04)     0.00 (0.06)   0.042
#>   Coma (mean (SD))       0.08 (0.26)     0.07 (0.25)   0.023
#>   lungcan (mean (SD))    0.01 (0.08)     0.01 (0.09)   0.014
#>   MOSF (mean (SD))       0.07 (0.26)     0.07 (0.26)   0.004
#>   sepsis (mean (SD))     0.21 (0.41)     0.22 (0.41)   0.002
#>   age (mean (SD))       61.36 (17.56)   61.43 (15.33)  0.004
#>   female (mean (SD))     0.45 (0.50)     0.45 (0.50)   0.001
#>   meanbp1 (mean (SD))   78.60 (37.58)   79.26 (40.31)  0.017
```

```
#### Causal Risk Difference with weigthed GLM
glm.obj <- glm(died ~ treatment, weights = weight, family =
               quasibinomial(link="identity"))
beta_ipw <- unname(coef(glm.obj))
SE <- unname(sqrt(diag(vcovHC(glm.obj, type="HC0"))))

#### get point estimate and CI for risk difference
cRD <- beta_ipw[2]
conf_level = 0.95
```

```
Z <- qnorm((1 + conf_level)/2)
cint <- beta_ipw[2] + c(-1,1) * Z * SE[2]
list(causal_RD = cRD, conf_int = cint, conf_level = conf_level)
```

```
#> $causal_RD
#> [1] 0.05154951
#>
#> $conf_int
#> [1] 0.02333275 0.07976627
#>
#> $conf_level
#> [1] 0.95
```

```
#### Causal Risk Ratio with weighted GLM
glm.obj <- glm(died ~ treatment, weights = weight, family =
               quasibinomial(link=log))
beta_ipw <- unname(coef(glm.obj))

#### To account for weighting, use asymptotic (sandwich) variance
SE <- unname(sqrt(diag(vcovHC(glm.obj, type="HC0"))))

#### get point estimate and CI for risk ratio
cRR <- exp(beta_ipw[2])
conf_level = 0.95
Z <- qnorm((1 + conf_level)/2)
cint <- exp(beta_ipw[2] + c(-1,1) * Z * SE[2])
list(causal_RR = cRR, conf_int = cint, conf_level = conf_level)
```

```
#> $causal_RR
#> [1] 1.081764
#>
#> $conf_int
#> [1] 1.036698 1.128789
#>
#> $conf_level
#> [1] 0.95
```

```
#### Causal Odds Ratio with weighted GLM
glm.obj <- glm(died ~ treatment, weights = weight, family =
               quasibinomial(link=logit))
beta_ipw <- unname(coef(glm.obj))

#### To account for weighting, use asymptotic (sandwich) variance
SE <- unname(sqrt(diag(vcovHC(glm.obj, type="HC0"))))

#### get point estimate and CI for odds ratio
cOR <- exp(beta_ipw[2])
conf_level = 0.95
Z <- qnorm((1 + conf_level)/2)
```

```
cint <- exp(beta_ipw[2] + c(-1,1) * Z * SE[2])
list(causal_OR = cOR, conf_int = cint, conf_level = conf_level)
```

```
#> $causal_OR
#> [1] 1.257132
#>
#> $conf_int
#> [1] 1.107062 1.427545
#>
#> $conf_level
#> [1] 0.95
```

4.3.2 Matching on covariate set of confounders

In this section I briefly review some of the analyses from Professor Jason Roy's Coursera course [20]. This is an outstanding course and I highly recommend that everyone view or take it. My review is no substitute for his course. Here are the general steps:

1. Prepare data
2. Match on covariate set X (using Mahalanobis distance)
3. Check for balanced covariate means (using standardized mean differencee)
4. Estimate average causal effect (as if randomized controlled trial)
5. Sensitivity analysis for hidden bias (not shown; instead see [20])

We will be pairwise matching control subjects ($A = 0$) to treated subjects ($A = 1$). Remember that "treated" is a general term for being exposed and "control" for not being exposed.[9] By matching on an appropriate set of confounders we are actually

- simulating a randomized controlled study, and
- measuring the **average causal effect of treatment on the treated** (ATT).

There are several R packages for matching: Matching, optmatch, MatchIt

Table 7. R packages for matching

Method or task	`tableone`	`Matching`	`rcbalance`
Mahalanobis distance		✓	✓
Robust M-distance			✓
Greedy matching		✓	
Optimal matching			✓
Assessing balance	✓		

The Mahalanobis distance is a multi-dimensional measure of mean distance between two vectors of covariates comparing treated to control. The smaller the M-distance the better the match. Because mean values can be influencd by outliers, an alternative is the robust M-distance that uses a rank statistic.

Greedy (nearest neighbor) matching is computationally fast but not globally optimized. Optimal matching finds the best global match but is computationally

[9] In a case-control study this would be matching controls (non-disease) to cases (disease).

demanding. We will use greedy matching.

We will analyze data from a health care observational study on right health catheterization (treatment) and death (outcome). We will match on a set of confounders, including age, sex, and mean blood pressure.

```
#load packages
library(tableone)
library(Matching)

#read in data
load(url("http://biostat.mc.vanderbilt.edu/wiki/pub/Main/DataSets/rhc.sav"))

#### create smaller data set and binary (0, 1) X variables
ARF <- as.numeric(rhc$cat1=='ARF')
CHF <- as.numeric(rhc$cat1=='CHF')
Cirr <- as.numeric(rhc$cat1=='Cirrhosis')
colcan <- as.numeric(rhc$cat1=='Colon Cancer')
Coma <- as.numeric(rhc$cat1=='Coma')
COPD <- as.numeric(rhc$cat1=='COPD')
lungcan <- as.numeric(rhc$cat1=='Lung Cancer')
MOSF <- as.numeric(rhc$cat1=='MOSF w/Malignancy')
sepsis <- as.numeric(rhc$cat1=='MOSF w/Sepsis')
female <- as.numeric(rhc$sex=='Female')
died <- as.numeric(rhc$death=='Yes')
age <- rhc$age
treatment <- as.numeric(rhc$swang1=='RHC')
meanbp1 <- rhc$meanbp1

#### new dataset
mydata <- cbind(ARF, CHF, Cirr, colcan, Coma, lungcan, MOSF,
                sepsis, age, female, meanbp1, treatment, died)
mydata <- data.frame(mydata)

#### covariate set
xvars <- c("ARF","CHF","Cirr","colcan","Coma","lungcan","MOSF","sepsis",
           "age","female","meanbp1")

#### look at a table 1
table1 <- CreateTableOne(vars=xvars,strata="treatment", data=mydata,
                         test=FALSE)
#### include standardized mean difference (SMD)
print(table1, smd = TRUE)
```

```
#>                       Stratified by treatment
#>                        0             1             SMD
#>   n                      3551          2184
#>   ARF (mean (SD))        0.45 (0.50)   0.42 (0.49)   0.059
#>   CHF (mean (SD))        0.07 (0.25)   0.10 (0.29)   0.095
```

```
#>   Cirr (mean (SD))     0.05 (0.22)   0.02 (0.15)   0.145
#>   colcan (mean (SD))   0.00 (0.04)   0.00 (0.02)   0.038
#>   Coma (mean (SD))     0.10 (0.29)   0.04 (0.20)   0.207
#>   lungcan (mean (SD))  0.01 (0.10)   0.00 (0.05)   0.095
#>   MOSF (mean (SD))     0.07 (0.25)   0.07 (0.26)   0.018
#>   sepsis (mean (SD))   0.15 (0.36)   0.32 (0.47)   0.415
#>   age (mean (SD))     61.76 (17.29) 60.75 (15.63)  0.061
#>   female (mean (SD))   0.46 (0.50)   0.41 (0.49)   0.093
#>   meanbp1 (mean (SD)) 84.87 (38.87) 68.20 (34.24)  0.455
```

In the table we want the standardized mean difference (SMD) to be less than 0.1. We see that several variables are not balanced (cirrhosis, coma, sepsis, and mean BP). Now let's perform the greedy match.

```
greedymatch <- Match(Tr=treatment,M=1,X=mydata[xvars],replace=FALSE)
#### create new data set with treated and matched controls
matched <- mydata[unlist(greedymatch[c("index.treated","index.control")]),]

#### get table 1 for matched data with standardized differences
matchedtab1 <- CreateTableOne(vars=xvars, strata ="treatment",
                              data=matched, test = FALSE)
print(matchedtab1, smd = TRUE)
```

```
#>                       Stratified by treatment
#>                        0             1             SMD
#>   n                     2184          2184
#>   ARF (mean (SD))       0.42 (0.49)   0.42 (0.49)   0.006
#>   CHF (mean (SD))       0.10 (0.29)   0.10 (0.29)  <0.001
#>   Cirr (mean (SD))      0.02 (0.15)   0.02 (0.15)  <0.001
#>   colcan (mean (SD))    0.00 (0.02)   0.00 (0.02)  <0.001
#>   Coma (mean (SD))      0.04 (0.20)   0.04 (0.20)  <0.001
#>   lungcan (mean (SD))   0.00 (0.05)   0.00 (0.05)  <0.001
#>   MOSF (mean (SD))      0.07 (0.26)   0.07 (0.26)  <0.001
#>   sepsis (mean (SD))    0.24 (0.43)   0.32 (0.47)   0.177
#>   age (mean (SD))      61.53 (16.15) 60.75 (15.63)  0.049
#>   female (mean (SD))    0.44 (0.50)   0.41 (0.49)   0.042
#>   meanbp1 (mean (SD)) 73.12 (34.28) 68.20 (34.24)  0.144
```

Once we have matched we can precede with the analysis as if we had data from a randomized controlled trial.

```
#### outcome analysis
y_trt <- matched$died[matched$treatment == 1]
y_con <- matched$died[matched$treatment == 0]

#### pairwise difference
diffy <- y_trt - y_con

#### paired t-test
t.test(diffy)
```

```
#>
#>  One Sample t-test
#>
#> data:  diffy
#> t = 3.9289, df = 2183, p-value = 8.799e-05
#> alternative hypothesis: true mean is not equal to 0
#> 95 percent confidence interval:
#>  0.02706131 0.08099730
#> sample estimates:
#> mean of x
#> 0.0540293
```

```
#### McNemar test is an alternative
(mctab <- table(y_trt, y_con))
```

```
#>      y_con
#> y_trt   0   1
#>     0 303 395
#>     1 513 973
```

```
mcnemar.test(mctab)
```

```
#>
#>  McNemar's Chi-squared test with continuity correction
#>
#> data:  mctab
#> McNemar's chi-squared = 15.076, df = 1, p-value = 0.0001033
```

4.3.3 Matching on propensity score

A propensity score (PS), $P(A = 1 \mid X)$ is the probability of treatment given the selected covariate set X. The PS is a *balancing score*, meaning that if we match controls to treated using PS, we also balance the distribution of covariates between treated and controls. *That's really cool!*[10] We will use logistic regression to calculate the PS. Once we have successfully matched the analysis proceeds as before.

```
psmodel <- glm(treatment~ARF+CHF+Cirr+colcan+Coma+lungcan+MOSF+
               sepsis+age+female+meanbp1, family=binomial(),
               data=mydata)
#### create propensity score
## pscore <- predict(psmodel, type = "response") # alternative
pscore <- psmodel$fitted.values

#### do greedy matching on logit(PS) using Match with a caliper
logit <- function(p) {log(p)-log(1-p)}
psmatch <- Match(Tr = mydata$treatment, M = 1, X = logit(pscore),
                 replace = FALSE, caliper = .2)
#### create new data set with treated and matched controls
matched <- mydata[unlist(psmatch[c("index.treated","index.control")]),]
```

[10] To understand the theory see [20].

```
xvars <- c("ARF", "CHF", "Cirr", "colcan", "Coma", "lungcan", "MOSF",
           "sepsis", "age", "female", "meanbp1")

#### get standardized differences
matchedtab1<-CreateTableOne(vars = xvars, strata ="treatment",
                            data = matched, test = FALSE)
print(matchedtab1, smd = TRUE)
```

```
#>                       Stratified by treatment
#>                        0              1              SMD
#>   n                     1932           1932
#>   ARF (mean (SD))       0.47 (0.50)    0.47 (0.50)    0.002
#>   CHF (mean (SD))       0.10 (0.30)    0.09 (0.29)    0.009
#>   Cirr (mean (SD))      0.02 (0.15)    0.03 (0.16)    0.010
#>   colcan (mean (SD))    0.00 (0.03)    0.00 (0.02)    0.019
#>   Coma (mean (SD))      0.05 (0.21)    0.05 (0.22)    0.015
#>   lungcan (mean (SD))   0.00 (0.06)    0.00 (0.05)    0.010
#>   MOSF (mean (SD))      0.09 (0.28)    0.08 (0.27)    0.019
#>   sepsis (mean (SD))    0.25 (0.43)    0.25 (0.43)    0.004
#>   age (mean (SD))      60.99 (17.69) 60.91 (15.51)   0.005
#>   female (mean (SD))    0.44 (0.50)    0.43 (0.49)    0.027
#>   meanbp1 (mean (SD)) 71.39 (34.13) 71.00 (35.01)   0.011
```

```
#### outcome analysis
y_trt <- matched$died[matched$treatment == 1]
y_con <- matched$died[matched$treatment == 0]
#### pairwise difference
diffy <- y_trt-y_con
#### paired t-test
t.test(diffy)
```

```
#>
#>  One Sample t-test
#>
#> data:  diffy
#> t = 2.6578, df = 1931, p-value = 0.007931
#> alternative hypothesis: true mean is not equal to 0
#> 95 percent confidence interval:
#>  0.01058136 0.07016399
#> sample estimates:
#>  mean of x
#> 0.04037267
```

4.4 Front-door criterion

For public health epidemiologists, the backdoor criterion is our go-to approach. However, we have occasions where unmeasured or unknown potential confounders complicate our health promotion strategy. For example, because studies linking

smoking to lung cancer were observational (i.e., not randomized controlled trials) the tobacco industry argued that investigators were unable to control for unknown confounders such as a "carcinogenic genotype that also induces an inborn craving for nicotine" [16].

Figure 15(a) depicts this challenge faced by investigators: The DAG does not satisfy the backdoor criterion because the variable U (carcinogenic genetype) is unobserved and cannot be used to block the backdoor path from A (smoking) to Y (lung cancer) ($A \leftarrow U \rightarrow Y$). The causal effect of smoking on lung cancer is not identifiable in this DAG.

Figure 15. Frontdoor criterion: (a) backdoor criterion will not work; we cannot block backdoor path ($A \leftarrow U \rightarrow Y$); (b) in the presence of a mediator (M) we can apply the backdoor criterion twice and measure the causal effect of A on Y. This is called the frontdoor criterion.

However, if a measurable mediator exist between A (smoking) and Y (lung cancer), for example, M (tar deposits), then we can measure the causal effect $P(Y = y \mid do(X = x))$ through two consecutive application of the backdoor criterion (see Figure 15(b)). The effect of A on M is identifiable because there is no backdoor path from A to M. Also, the effect of M on Y is identifiable because there is no backdoor path from M to Y.

For the frontdoor criterion, the backdoor criterion is applied twice: A to M, then M to Y. Here is the full expression:

$$P(Y = y \mid do(A = a)) = \sum_m P(Y = y \mid do(M = m))P(M = m \mid do(A = a))$$

Solving for each expression on the right side yields:

$$P(M = m \mid do(A = a)) = P(M = m \mid A = a)$$

$$P(Y = y \mid do(M = m)) = \sum_a P(Y = y \mid M = m, A = a)P(A = a)$$

And by substituting back into to full expression we get the **front-door adjustment formula**, where $P(Y = y \mid do(A = a)) =$

$$\sum_m \sum_{a'} P(Y = y \mid M = m, A = a')P(A = a')P(M = m \mid A = a)$$

And the front-door adjustment formula can be written in shorter form:

$$P(y \mid do(a)) = \sum_m P(m \mid a) \sum_{a'} P(y \mid m, a')P(a')$$

4.4.1 Example: lung cancer, smoking, tar and carcinogenic genotype

Pearl presents a hypothetical population of 800,000 subjects at high risk for lung cancer due to smoking and other exposures [16]. Data was available on smoking history (A), lung cancer (Y), and lung tar deposits (M). Figure 15(b) represents the DAG for this data. The data numbers are in the thousands.

```
A.lv <- c("Smokers", "Non.smokers")
M.lv <- c("Tar", "No.tar")
Y.lv <- c("Cancer", "No.cancer")
tab.yma <- array(c(323, 57, 18, 2, 1, 19, 38, 342), dim = c(2, 2, 2),
                 dimnames = list(Y = Y.lv, M = M.lv, A = A.lv))
tab.may <- aperm(tab.yma, perm=c(2,3,1))
tab.amy <- aperm(tab.yma, perm=c(3,2,1))
ftable(tab.may)
```

```
#>                     Y Cancer No.cancer
#> M      A
#> Tar    Smokers           323        57
#>        Non.smokers         1        19
#> No.tar Smokers            18         2
#>        Non.smokers        38       342
```

```
ftable(tab.amy)
```

```
#>                     Y Cancer No.cancer
#> A           M
#> Smokers     Tar           323        57
#>             No.tar         18         2
#> Non.smokers Tar             1        19
#>             No.tar         38       342
```

This hypothetical data set could be analyzed in such a way to make smoking appear healthy (smokers have less lung cancer), or that tar deposits are unhealthy (tar deposits associated with lung cancer).[11] However, we recognize this as an opportunity to use the front-door criterion to assess the causal effect of smoking on lung cancer while controlling for unknown confounders.

Here is the front-door adjustment formula again where $P(y \mid do(a)) =$

$$\sum_m P(m \mid a) \sum_{a'} P(y \mid m, a')P(a')$$

We need the sum of these products over all combinations of m and a'. Because there are two summations we will use nested `for` loops twice: once for nonsmokers ($A = 0$) and once for smokers ($A = 1$).

[11] See Pearl [16], p. 67, for these arguments.

```
(P.m_a <- prop.table(apply(tab.yma, c(2,3), sum),2))
```

```
#>          A
#> M         Smokers Non.smokers
#>   Tar        0.95        0.05
#>   No.tar     0.05        0.95
```

```
(P.y_m.a <- prop.table(tab.yma,c(2,3)))
```

```
#> , , A = Smokers
#>
#>             M
#> Y             Tar No.tar
#>   Cancer    0.85    0.9
#>   No.cancer 0.15    0.1
#>
#> , , A = Non.smokers
#>
#>             M
#> Y             Tar No.tar
#>   Cancer    0.05    0.1
#>   No.cancer 0.95    0.9
```

```
(P.a <- prop.table(apply(tab.yma, 3, sum)))
```

```
#>     Smokers Non.smokers
#>         0.5         0.5
```

```
####
#### Calculate front-door adjustment
#### Y = cancer among nonsmokers
pY1_do.A0.vec <- rep(NA, length(A.lv)*length(M.lv))
y1 <- Y.lv[1] # cancer
a0 <- A.lv[2] # nonsmokers
step <- 1
for(ap in A.lv){
  for(m in M.lv){
    pY1_do.A0.vec[step] <- P.y_m.a[y1, m, ap] * P.a[ap] * P.m_a[m, a0]
    step <- step + 1
} }
(pY1_do.A0 <- sum(pY1_do.A0.vec))
```

```
#> [1] 0.4975
```

```
####
#### Y = cancer among smokers
pY1_do.A1.vec <- rep(NA, length(A.lv)*length(M.lv))
y1 <- Y.lv[1] # cancer
a1 <- A.lv[1] # smoker
step <- 1
for(ap in A.lv){
```

```
  for(m in M.lv){
    pY1_do.A1.vec[step] <- P.y_m.a[y1, m, ap] * P.a[ap] * P.m_a[m, a1]
    step <- step + 1
} }
(pY1_do.A1 <- sum(pY1_do.A1.vec))
```

```
#> [1] 0.4525
```

This problem is fully worked out here: http://bayes.cs.ucla.edu/BOOK-2K/ch3-3.pdf. Because this is a hypothetical data set the final answer is contrary to known facts about smoking and lung cancer. This exercise was designed to illustrate the front-door adjustment formula.

4.5 Instrumental variable

TBD

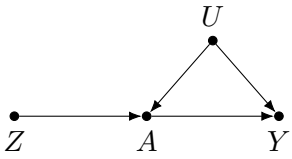


Figure 16. Instrumental variable

```
#### read dataset
data(card.data)

#IV is nearc4 (near 4 year college)
#outcome is lwage (log of wage)
#'treatment' is educ (number of years of education)

#summary stats
mean(card.data$nearc4)
par(mfrow=c(1,2))
hist(card.data$lwage)
hist(card.data$educ)

#is the IV associated with the treatment? strenght of IV
mean(card.data$educ[card.data$nearc4==1])
mean(card.data$educ[card.data$nearc4==0])


#make education binary
educ12<-card.data$educ>12
#estimate proportion of 'compliers'
propcomp<-mean(educ12[card.data$nearc4==1])-
  mean(educ12[card.data$nearc4==0])
propcomp
```

```
#intention to treat effect
itt<-mean(card.data$lwage[card.data$nearc4==1])-
  mean(card.data$lwage[card.data$nearc4==0])
itt

#complier average causal effect
itt/propcomp


#two stage least squares

#stage 1: regress A on Z
s1<-lm(educ12~card.data$nearc4)
## get predicted value of A given Z for each subject
predtx <-predict(s1, type = "response")
table(predtx)

#stage 2: regress Y on predicted value of A
lm(card.data$lwage~predtx)


#2SLS using ivpack
ivmodel=ivreg(lwage ~ educ12, ~ nearc4, x=TRUE, data=card.data)
robust.se(ivmodel)


ivmodel=ivreg(lwage ~ educ12 + exper + reg661 + reg662 +
                reg663 + reg664 + reg665+ reg666 + reg667 + reg668,
                ~ nearc4 + exper +
                  reg661+ reg662 + reg663 + reg664 + reg665 + reg666 +
                reg667 + reg668, x=TRUE, data=card.data)
```

5 Decision quality (with decision Bayesian networks)

Public health practice is replete with claims of "data-driven decision-making." This usually means reading scientific articles, looking at local data and information, and implementing evidence-based strategies. With few exceptions, there is generally no formal, deliberative, structured process for translating data into actionable knowledge, and for *knowledge integration*—the management, synthesis, and translation of knowledge into decision support systems to improve policy, practice, and—ultimately—population health.

Because we make intuitive decisions daily, most of us are not formerly trained in decision making. Decisions are oftens based on power, politics, advocacy, organizational history and culure, and personal interests. Yet, decisions determine how we spend our time and allocate scarce resources, so they should be driven by strategic goals, stakeholder needs, evidence, and decision quality (DQ).

DQ (Table TODO) is a checklist for continuous decision improvement (decision competence), and a road map for advanced methods such as decision analysis (DA). DA is a formal method for tackling decisions in the face of uncertainty and trade-offs. DA is usually conducted with decision trees (Figure 17) where squares represent decision nodes and ellipses represent uncertainty (chance) nodes. On the right side of the decision tree are the utilities (weighted outcomes [usually weighted by preference]). The general approach is to calculate the expected utility (value) for each decision option, and to choose the option that maximizes this utility.

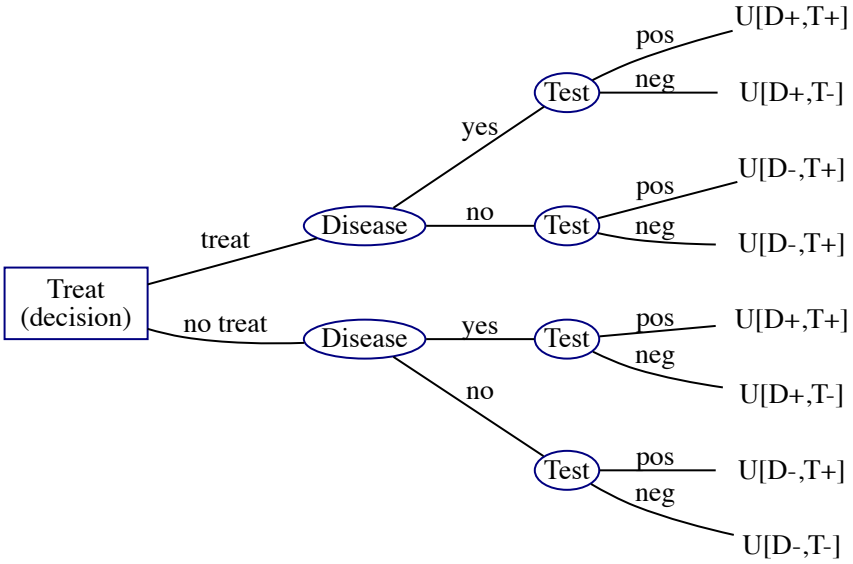


Figure 17. A decision tree for decision analysis

An alternative way to structure a decision is with a decision Bayesian network (BN) (also called an influence diagram). An **influence diagram** is a BN that includes *decision*, *uncertainty* (chance), *calculation* (deterministic), and *value* (utility) nodes (Figure 18).

Figure 19 (p. 41) is the influence diagram version of Figure 17 (p. 40). Compared to decision trees, influence diagrams can more efficiently represent complex decision

problems. Figure 19 has some key features:

1. Decision nodes alway connect to the final utility (value) node.
2. A node connecting into a decision node (e.g., "Test") represents information that is available to "influence" the decision node.
3. The disease node has a causal effect on the diagnostic test results.
4. The disease node has a causal effect on the final outcome (utility node).

Notice that this influence diagram is able to integrate disease prevalence (prior probability), diagnostic testing, test characteristics (sensitivity, specificity), and outcomes.

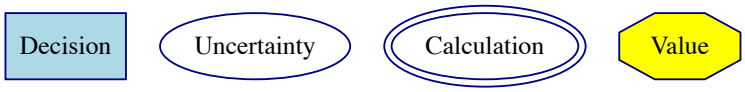


Figure 18. Node definitions for influence diagrams

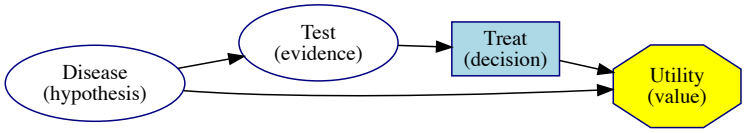


Figure 19. A Bayesian decision network (influence diagram) for decision analysis

5.1 Example 1: Decision to buy stock

A basic decision that involves uncertainty is to choose between (a) an option with an uncertain outcome and payout, or (b) continue with the status quo. Figure 20 from Neapolitan (p. 492, [14]) is the decision tree that represents the decision ("d1") to invest \$1000 and buy stock X with a 0.6 chance to grow in value to \$1100 and 0.4 chance to shrink in value to \$900, or the decision ("d2") to do nothing and keep the \$1000.

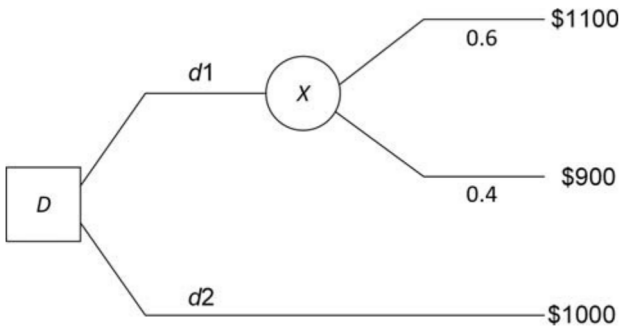


Figure 20. Simple decision tree for decision whether to buy stock X.

Figure 21 (p. 42) from Neapolitan (p. 492, [14]) is the influence diagram version of the Figure 20 decision tree. Now, here are the expected value calculations for each decision option:

$$E(\text{d1}) = 0.6 \times \$1100 + 0.4 \times \$900 = \$1020$$

$$E(\text{d2}) = \$1000$$

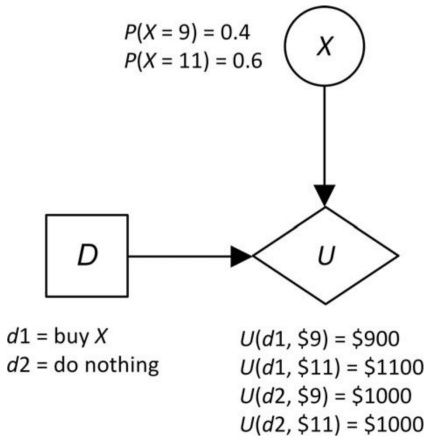


Figure 21. Simple influence diagram for decision whether to buy stock X.

Using R, we create the decision BN. Actually, it is a regular BN with minor tweaks. For the decision node, we set the probabilities for each decision option (d1, d2) to 0.5. To get the expected value (EV) we set the decision node evidence to "d1" and calculate EV, and then to "d2" and calculate EV again.

For the utility (value) node we set the possible dollar outcome levels as 900, 1100, and 1000. After setting decision node evidence to "d1", we derive the new distribution of dollar outcomes, and we calculate the EV. We repeat this for "d2". See the R code below with the details.

```
#### reload R packages if necessary
#### library(bnlearn)
#### library(gRain)
dag3 <- empty.graph(nodes=c("D", "U", "X"))
dag3 <- set.arc(dag3, from = "X", to = "U")
dag3 <- set.arc(dag3, from = "D", to = "U")
## graphviz.plot(dag3)
#### create levels
X.lv <- c("9", "11")
D.lv <- c("d1", "d2")
U.lv <- c("900", "1100", "1000") ## possible dollars outcomes
#### create conditional probability tables
X.prob <- array(c(0.4,1-0.4), dim = 2, dimnames = list(X = X.lv))
D.prob <- array(c(0.5,0.5), dim = 2, dimnames = list(D = D.lv))
U.prob <- array(c(1,0,0,0,1,0,0,0,1,0,0,1), dim = c(3,2,2),
 dimnames = list(U = U.lv, X=X.lv, D=D.lv))
cpt3 <- list(X=X.prob, D=D.prob, U=U.prob)
bn3 <- custom.fit(dag3, cpt3)
```

```
junction3 <- compile(as.grain(bn3))
querygrain(junction3) ## display BN
```

```
#> $D
#> D
#>  d1  d2
#> 0.5 0.5
#>
#> $U
#> U
#>  900 1100 1000
#>  0.2  0.3  0.5
#>
#> $X
#> X
#>   9  11
#> 0.4 0.6
```

```
#### Expected value for D = "d1" (buy stock X)
jbuy <- setEvidence(junction3, nodes = "D", states = "d1")
U.buy <- querygrain(jbuy, nodes = "U")$U
(EV.buy <- sum(U.buy*as.numeric(names(U.buy))))
```

```
#> [1] 1020
```

```
#### Expected value for D = "d2" (do nothing)
jdont <- setEvidence(junction3, nodes = "D", states = "d2")
U.dont <- querygrain(jdont, nodes = "U")$U
(EV.dont <- sum(U.dont*as.numeric(names(U.dont))))
```

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

5.2 Example 2: Decision to buy Spiffycar

From Neapolitan (p. 492, [14]) we now tackle a decision problem that includes a diagnostic test (Figure 22, p. 44):

> "... Suppose Sam has the opportunity to buy a 1996 Spiffycar automobile for $10,000, and he had a prospect that would be willing to pay $11,000 for the auto if it were in excellent mechanical shape. Suppose further that if the transmission is bad, Sam will have to spend $3000 to repair it before he could sell the vehicle. So he would end up with only $8,000 if he bought the vehicle and its transmission was bad. Finally, suppose Sam has a friend who could run a test on the transmission, and we have the following:"

$$P(Test = \text{positive} \mid Tran = \text{good}) = 0.3$$

$$P(Test = \text{positive} \mid Tran = \text{bad}) = 0.9$$

$$P(Tran = \text{good}) = 0.8$$

The Bayesian network (Figure 22) "in this influence diagram contains 2 nodes, $Tran$ and $Test$. There is an edge from $Tran$ to $Test$ because the value of the test is probabilistically dependent on the state of the transmission. There is an edge from $Test$ to D because the outcome of the test will be known at the time the decision is made. That is, D follows $Test$ in sequence. Finally, the utility U depends only on the value of $Tran$ and the decision D. It does not depend on the outcome of the $Test$. So there are arrows from $Tran$ and D to U. If Sam makes decision `d1` and $Tran$ is good, the utility of the outcome will be \$11 000. On the other hand, if Sam makes decision `d1` and $Tran$ is bad, the utility of the outcome will be \$8,000. However, if Sam makes decision `d2`, the utility of the outcome is \$10,000 regardless of whether $Tran$ is good or bad because he has decided not to buy the car."

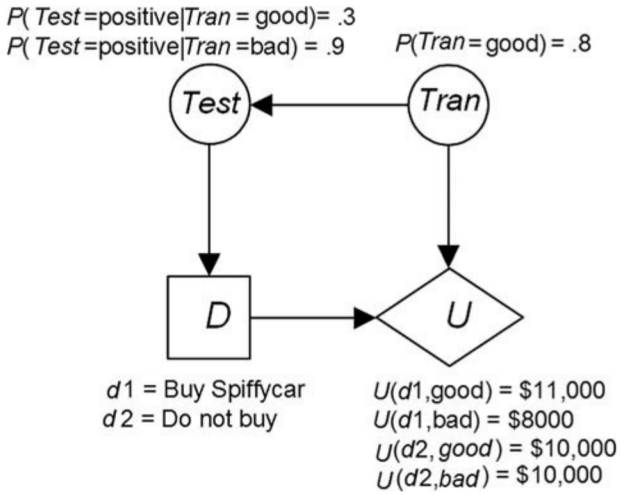


Figure 22. Influence diagram representing decision to buy Spiffycar.

Using R, we create the "decision BN." Actually, it is a regular BN with minor tweaks. For the decision node, we set the probabilities for each decision option (d1, d2) to 0.5. To get the expected utility (EU) we set the decision node evidence to "d1" and calculate EU, and then to "d2" and calculate EU again.

For the utility (value) node we set the possible dollar outcome levels as 8000, 10000, and 11000. After setting decision node evidence to "d1", we derive the new distribution of dollar outcomes, and we calculate the EU. We repeat this for "d2". See the R code below with the details.

```
dag4 <- empty.graph(nodes=c("Test", "Tran", "D", "U"))
dag4 <- set.arc(dag4, from = "Tran", to = "Test")
dag4 <- set.arc(dag4, from = "Tran", to = "U")
dag4 <- set.arc(dag4, from = "Test", to = "D")
dag4 <- set.arc(dag4, from = "D", to = "U")
#### create levels
Tran.lv <- c("Good", "Bad")
Test.lv <- c("Pos", "Neg")
```

```
D.lv <- c("d1", "d2") ## d1 = buy spiffycar; d2 = do not buy
U.lv <- c("11000", "8000", "10000")
#### create conditional probability tables
Tran.prob <- array(c(0.8,1-0.8), dim = 2, dimnames = list(Tran = Tran.lv))
Test.prob <- array(c(0.3,1-0.3,0.9,1-0.9), dim = c(2,2),
  dimnames = list(Test = Test.lv, Tran = Tran.lv))
D.prob <- array(c(.5, .5, .5, .5), dim = c(2,2),
  dimnames = list(D = D.lv, Test = Test.lv))
U.prob <- array(c(1,0,0,0,1,0,0,0,1,0,0,1), dim = c(3,2,2),
 dimnames = list(U = U.lv, Tran=Tran.lv, D=D.lv))
cpt4 <- list(Tran=Tran.prob, Test=Test.prob, D=D.prob, U=U.prob)
bn4 <- custom.fit(dag4, cpt4)
junction4 <- compile(as.grain(bn4))
querygrain(junction4) ## display BN
```

```
#> $Test
#> Test
#>  Pos  Neg
#> 0.42 0.58
#>
#> $Tran
#> Tran
#> Good  Bad
#>  0.8  0.2
#>
#> $D
#> D
#>  d1  d2
#> 0.5 0.5
#>
#> $U
#> U
#> 11000  8000 10000
#>   0.4   0.1   0.5
```

What is the EU of choosing "d1" even if the transmision test is positive? That is, what is $EU(D = \text{d1}, Test = \text{positive})$?

```
#### What is EU(D = d1, Test = positive)?
jTest.pos_D.d1 <- setEvidence(junction4, nodes = c("Test", "D"),
                              states = c("Pos", "d1"))
U_T.p_D.d1 <- querygrain(jTest.pos_D.d1, nodes = "U")$U
(EU_T.p_D.d1 <- sum(U_T.p_D.d1 * as.numeric(names(U_T.p_D.d1))))
```

```
#> [1] 9714.286
```

What is the EU of choosing "d1" even if the transmision test is negative? That is, what is $EU(D = \text{d1}, Test = \text{negative})$?

```
#### What is EU(D = d1, Test = negative)?
jTest.neg_D.d1 <- setEvidence(junction4, nodes = c("Test", "D"),
                              states = c("Neg", "d1"))
U_T.n_D.d1 <- querygrain(jTest.neg_D.d1, nodes = "U")$U
(EU_T.n_D.d1 <- sum(U_T.n_D.d1 * as.numeric(names(U_T.n_D.d1))))
```

```
#> [1] 10896.55
```

What is the EU of choosing "d2" regardless of transmission test results? That is, what is $EU(D = \text{d2})$?

```
#### What is EU(D = d2)?
jD.d2 <- setEvidence(junction4, nodes = "D", states = "d2")
U_D.d2 <- querygrain(jD.d2, nodes = "U")$U
(EU_D.d2 <- sum(U_D.d2 * as.numeric(names(U_D.d2))))
```

```
#> [1] 10000
```

5.3 Example 3: Comparing complex policy options

TBD

5.4 Example 4: Discrete time Markov chains

TBD

5.5 Example 5: Cost-effective analysis

TBD

References

1. Kahneman D. Thinking, fast and slow. New York: Farrar, Straus; Giroux; 2011.

2. Aragón TJ, Colfax G. We will be the best at getting better! A playbook for population health improvement. UC Berkeley eScholarship [Internet]. 2019; Available from: https://escholarship.org/uc/item/9xg5t30s

3. Howard RA, Abbas AE. Foundations of decision analysis. 1st ed. Pearson; 2015.

4. Spetzler C, Winter H, Meyer J. Decision quality: Value creation from better business decisions. 1st ed. Wiley; 2016.

5. Aragón TJ. PDSA problem-solving: With a gentle introduction to double-loop learning, program theory, and causal graphs. University of California, eScholarship [Internet]. 2017; Available from: http://www.escholarship.org/uc/item/8wp451vd

6. Barrett L. How emotions are made: The secret life of the brain. Mariner Books; 2018.

7. Hess E. Humility is the new smart: Rethinking human excellence in the smart machine age. Oakland, CA: Berrett-Koehler Publishers, a BK Business Book; 2017.

8. Adhikari A, DeNero J. Computational and inferential thinking: The foundations of data science [Internet]. Self-published; 2018. Available from: https://www.inferentialthinking.com/

9. R Core Team. R: A Language and Environment for Statistical Computing [Internet]. Vienna, Austria: R Foundation for Statistical Computing; Available from: https://www.r-project.org/

10. Funnell SC, Rogers PJ. Purposeful program theory: Effective use of theories of change and logic models. San Francisco, CA: Jossey-Bass; 2011.

11. Fenton N, Neil M. Risk assessment and decision analysis with bayesian networks. CRC Press; 2012.

12. Scutari M, Denis J-B. Bayesian networks: With examples in r. Boca Raton: Chapman; Hall; 2014.

13. Fenton N, Neil M. Risk assessment and decision analysis with bayesian networks. Chapman; Hall/CRC; 2 edition; 2019.

14. Neapolitan R, Jiang X, Ladner DP, Kaplan B. A primer on Bayesian decision analysis with an application to a kidney transplant decision. Transplantation. 2016 Mar;100(3):489–96.

15. Pearl J. The Book of Why: The new science of cause and effect. Basic Books; 2018.

16. Pearl J, Glymour M, Jewell NP. Causal inference in statistics: A primer. 1st ed. Wiley; 2016.

17. Hernán MA, Robins JM. Causal inference: What if [Internet]. Boca Raton: Chapman & Hall/CRC. 2020. Available from: https://www.hsph.harvard.edu/miguel-hernan/causal-inference-book/

18. Cole SR, Platt RW, Schisterman EF, Chu H, Westreich D, Richardson D, et al. Illustrating bias due to conditioning on a collider. International journal of epidemiology. 2010 Apr;39(2):417–20.

19. Westreich D. Epidemiology by design: A causal approach to the health sciences. Oxford University Press; 2019.

20. Roy J. A Crash Course in Causality: Inferring causal effects from observational data [Internet]. 2018. Available from: https://www.coursera.org/learn/crash-course-in-causality