The Basics of Building DAGs in R

Leah Moubadder

2/15/2021

Contents

Setup

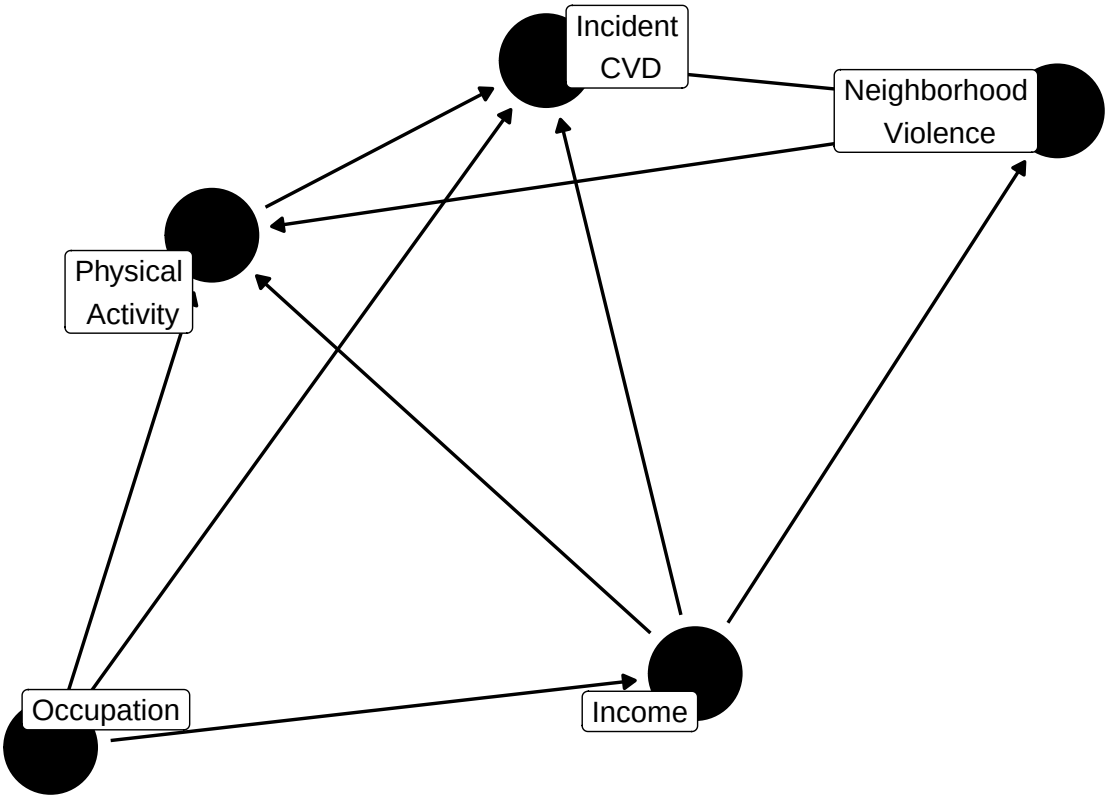
Loading relevant package. Here we will be using ggdag and setting a theme that is provided by the package.

```
library(ggdag)
theme_set(theme_dag())
```

Build DAG

Build and plot your DAG using the "dagify" and "ggdag" function, respectively. Note that the variable that is on the **left** of the tilde is what the arrow is going **into**. Additionally, the "+" symbol is used to graph a collider.

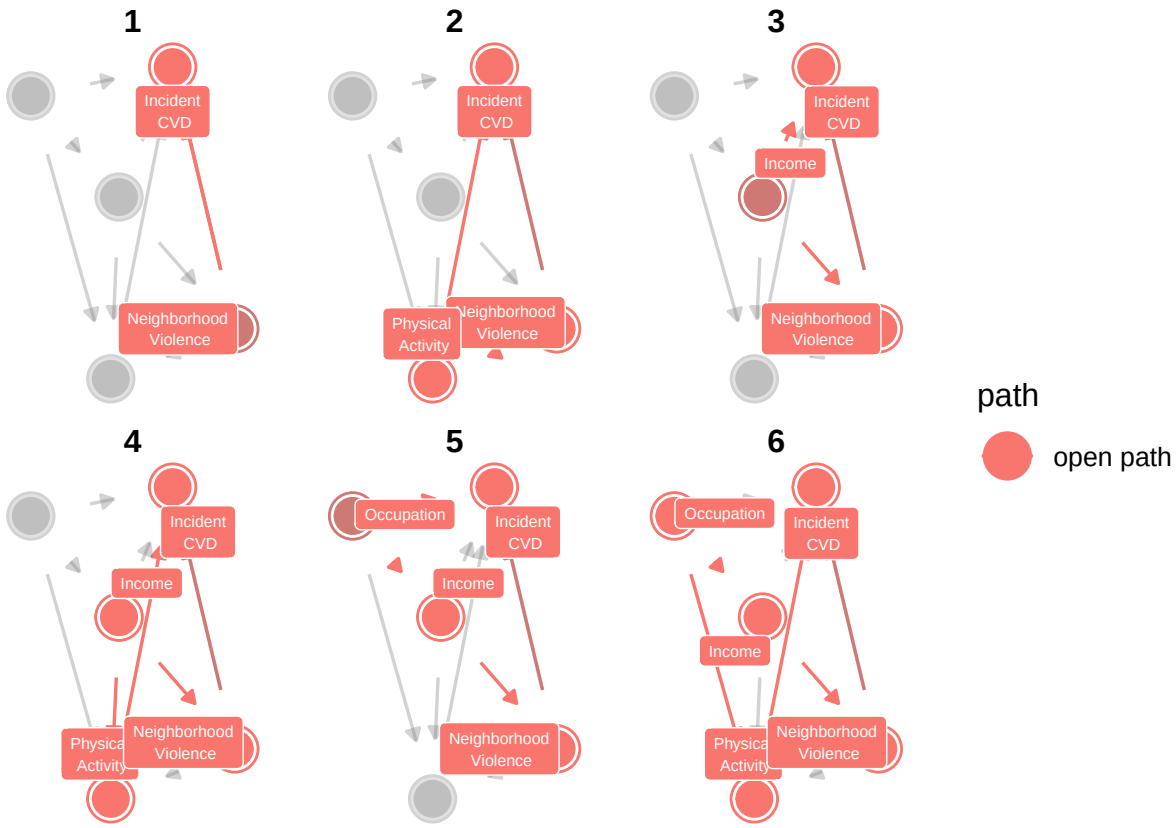
```
NV_CVD_dag<- dagify(CVD ~ NV,
                    NV ~ income,
                    income ~ occupation,
                    CVD ~ occupation,
                    PA ~ occupation,
                    PA ~ NV,
                    CVD ~ PA,
                    CVD ~ income,

#physical activity is a common child of income and occupation
                    PA ~ income + occupation,

#create labels for variables ("\n" denotes a space)
       labels = c("CVD" = "Incident\n CVD",
                  "NV" = "Neighborhood\n Violence",
                  "income" = "Income",
                  "occupation" = "Occupation",
                  "PA" = "Physical\n Activity"),

#identify exposure and outcome
                  exposure = "NV",
                  outcome = "CVD")

#plot DAG using the ggdag function
ggdag(NV_CVD_dag, text = FALSE, use_labels = "label",edge_type="link_arc")
```


Incident
CVD
Neighborhood
Violence
Physical
Activity
Occupation
Income

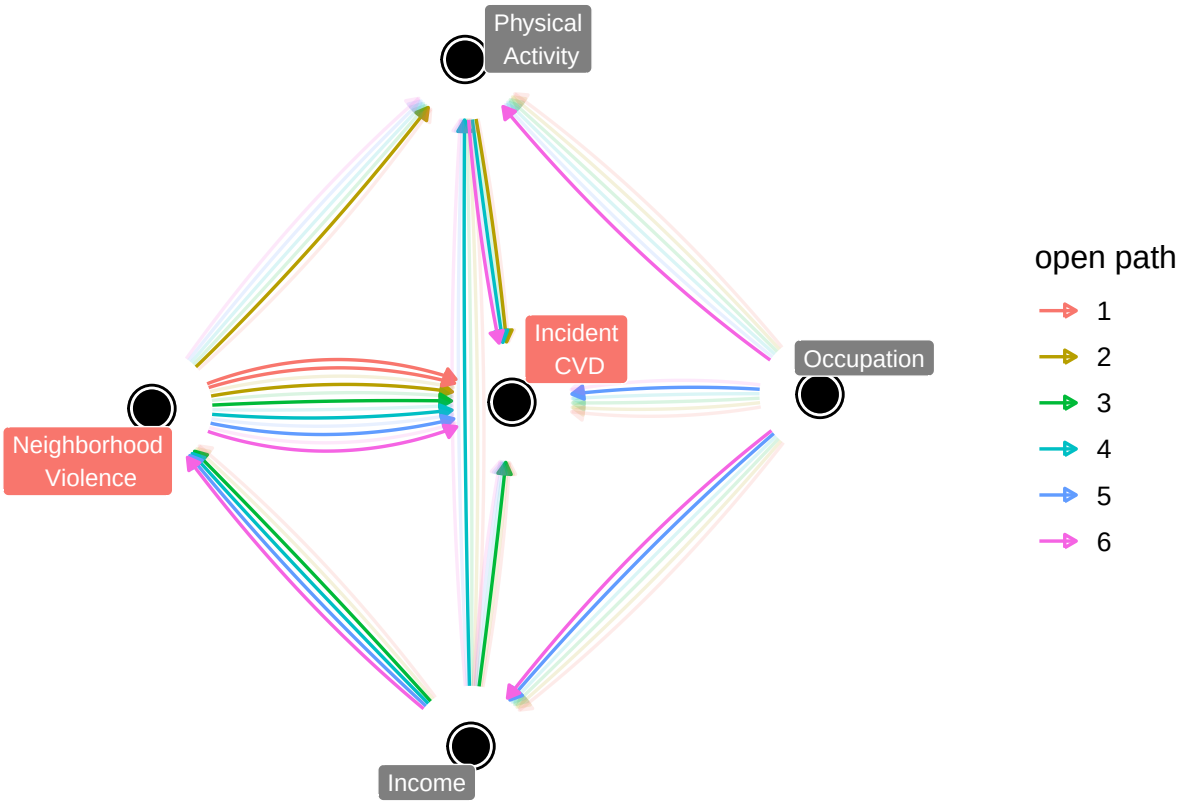
Identify Open Paths

Use the "ggdag_paths" function to identify **all open** paths (directed or non-directed). Consistent with our in-class example, we see six open paths. The first function might not be ideal, given that we have so many open paths. The "gg_paths_fan" (second example) is another option where you are able to overlay open paths.

```
ggdag_paths(NV_CVD_dag, text = FALSE, use_labels = "label", shadow = TRUE,

#directed=false indicates all paths, regardless of direction
            directed=FALSE,

#changing node/text sizes for aesthetics
            node_size = 8,   text_size = 2, stylized = TRUE)
```



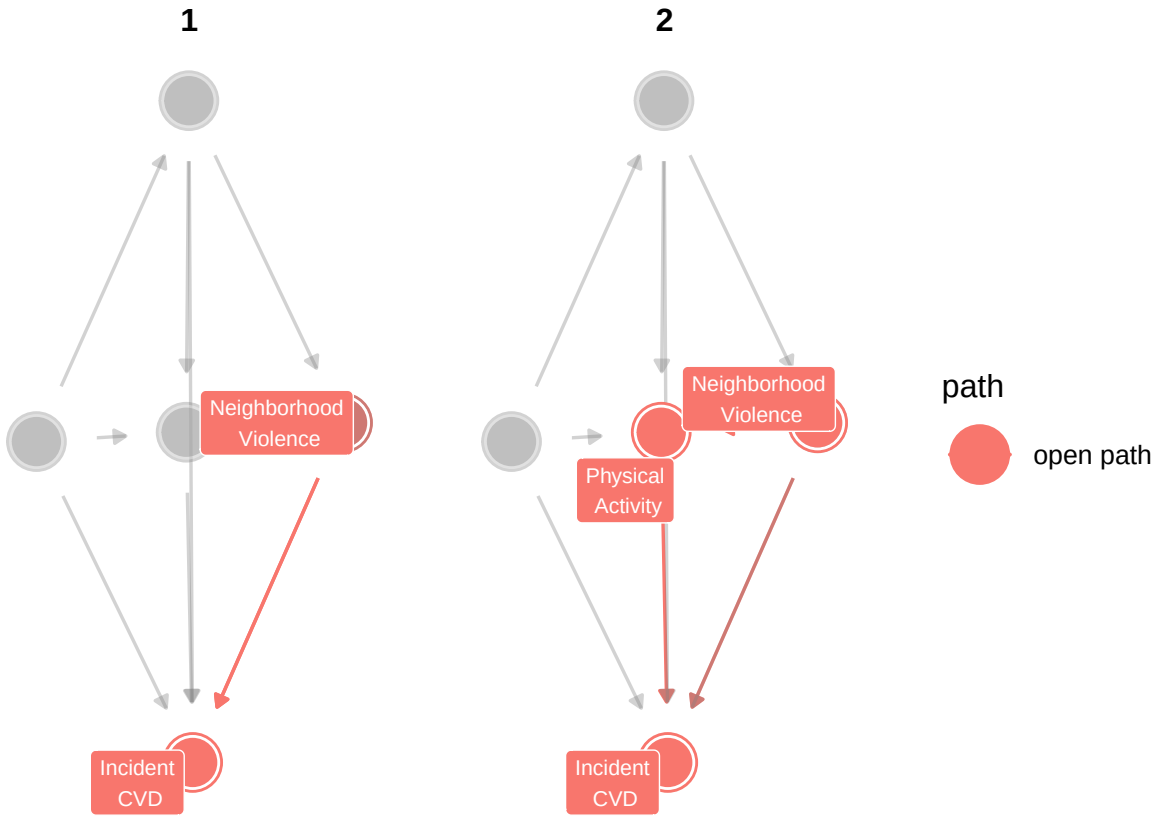
```r
#using "ggdag_paths_fan to overlap all paths
ggdag_paths_fan(NV_CVD_dag, text = FALSE, use_labels = "label", shadow = TRUE,
            directed=FALSE,
            node_size = 8,   text_size = 3, stylized = TRUE)
```



Identify Directed Paths

Use the "ggdag_paths" function to identify **directed** paths only.

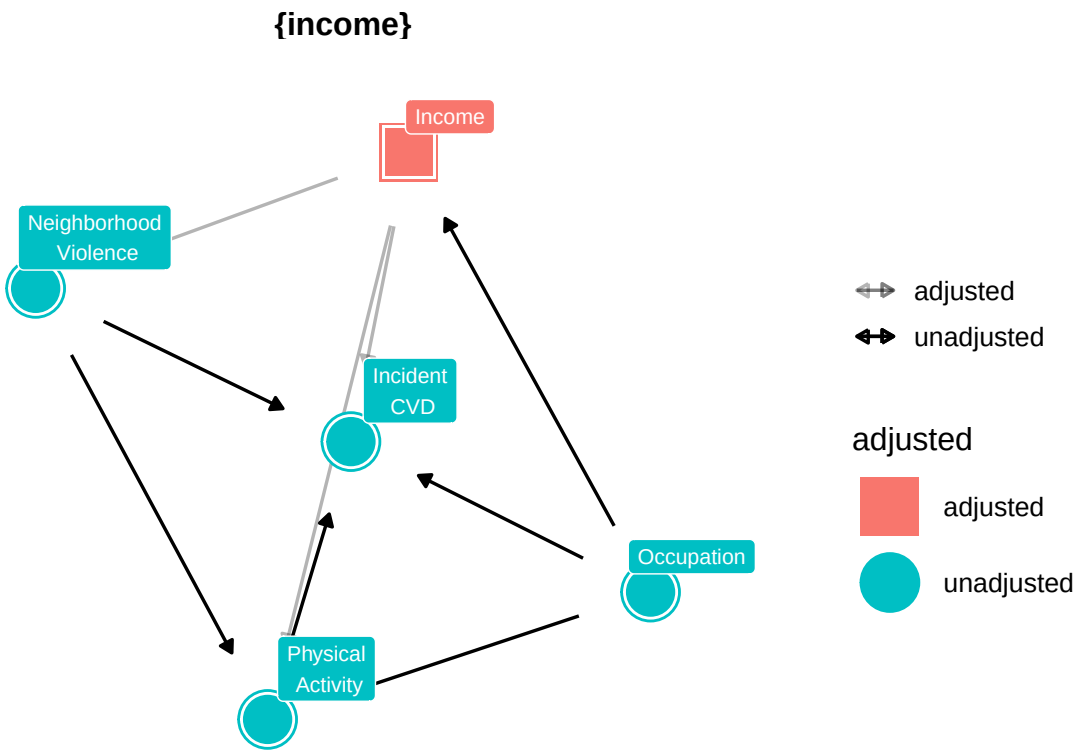
```
ggdag_paths(NV_CVD_dag, text = FALSE, use_labels = "label", shadow = TRUE,

#directed=true indicates directed paths
            directed=TRUE,
            node_size = 10,   text_size = 2.8, stylized = TRUE)
```



Identify Minimally Sufficient Set

Use the "ggdag_adjustment_set" function to identify the minimally sufficient set. As previously identified, {income} is the minimally sufficient set.

```
ggdag_adjustment_set(NV_CVD_dag, text = FALSE, use_labels = "label",
                     shadow = TRUE, stylized = TRUE, node_size = 10,   text_size = 2.8)
```



Identify d-separated & d-connected Paths

Use the "ggdag_drelationship" function to identify d-separated and d-connected paths after conditioning on the minimally sufficient set (income).

```
ggdag_drelationship(NV_CVD_dag, controlling_for  = "income", text = FALSE, use_labels = "label",
                    stylized = TRUE, node_size = 10,   text_size = 2.8, edge_type="link_arc")
```