Instrumental Variables and 2SLS

Today, we are running some code that will help us understand the basics of instrumental variables. We will analyze the relationship between eating chocolate and happiness. Clearly, we cannot run a simple regression: There may be many omitted variables (e.g., people with lactose intolerance are happier, but they also consume more chocolate) or even reverse causality (e.g., when your GSI is stressed and unhappy, they consume tons of chocolate).

Thankfully, we have two potential instrumental variables at hand: 1) We randomly assigned people a voucher that gives them free chocolate, and 2) we know how far they live away from a grocery store.

We will run through the mechanics of the IV estimation. To understand what exactly is going on, we also show you **how the data is generated**, i.e., what the actual **truth** is. This is a trick we can use when we want to check whether a method performs well: We simulate some data, and because we simulated it, we know the truth. Then we can just check whether running a regression with the method we want will give us the correct result.

Setting up the data

We first load the required packages and set the number of observations (3,000 individuals) and a "seed" - this allows us to use random numbers and get exactly the same numbers every time we run the code.

In [1]:
```
install.packages("ivreg")
install.packages("huxtable")
install.packages("jtools")

library('ivreg')
library('huxtable')
library('jtools')


set.seed(12345)
n=3000
```

```
Installing package into ‘/opt/r’
(as ‘lib’ is unspecified)

Installing package into ‘/opt/r’
(as ‘lib’ is unspecified)

Installing package into ‘/opt/r’
(as ‘lib’ is unspecified)
```

Next, we generate a data frame and fill it with some observations. The two instruments (voucher and distance) are random variables (one is a "binomial" random variable and will be a dummy, the other a uniform random variable).

In [6]:
```
data_iv = data.frame(seq(1, n))
colnames(data_iv)="n"

# The first instrument is a dummy variable: A lottery whether you received a vouche
data_iv$voucher = rbinom(n,1,0.5)
```

```
# The second instrument is a continuous variable: The distance to the closest super
data_iv$distance = runif(n,0,1)
```

Next, we generate some other variables: `unobserved_unhappiness` is how unhappy the respondent was *before* buying any chocolate. We do not observe this and this will generate omitted variable bias (strictly speaking, this is reverse causality). We also generate a truly random error that is unrelated to anything else in the data. And we also have data on whether or not a person is lactose intolerant.

In [7]:
```
# These are some other variables: Being unhappy on a given day, an unobserved error
data_iv$unobserved_unhappiness = rnorm(n,0,1)
data_iv$yerror = rnorm(n,0,1)
data_iv$lactose_intolerant = rbinom(n,1,0.5)
```

Finally, we know exactly what determines the consumption of chocolate, and what determines happiness. This is often called the "data-generating process".

In [8]:
```
# This is the "data-generating process" for chocolate consumption:
  # - People who got the voucher eat more chocolate
  # People who live further away from supermarket eat less chocolate, people who ar
data_iv$chocolate = 0.8*data_iv$voucher - data_iv$distance - data_iv$lactose_intole

# This is the DGP for happiness: Eating chocolate makes you happier, being lactose
data_iv$happiness = 1*data_iv$chocolate + data_iv$lactose_intolerant - data_iv$unob
```

Questions for you

- Can you see from the DGP: What is the true effect of chocolate on happiness? What would you want to see as regression result?
- Can you guess: If we run the OLS regression (pretending we do not know unobserved unhappiness), if there will be OVB?
- Are distance and voucher valid instruments in this framework (i.e., do they satisfy the relevance, independence, and exclusion restriction)?

Running OLS

In [11]:
```
# We immediately see that OLS is biased: unobserved_unhappiness is correlated with
summary(lm(happiness ~ chocolate + lactose_intolerant , data=data_iv))
```

```
Call:
lm(formula = happiness ~ chocolate + lactose_intolerant, data = data_iv)

Residuals:
    Min      1Q  Median      3Q     Max 
-3.5643 -0.7370 -0.0031  0.7222  4.0859 

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)    
(Intercept)        -0.04349    0.02788  -1.560  0.11885    
chocolate           0.18416    0.01744  10.558  < 2e-16 ***
lactose_intolerant  0.11486    0.04276   2.686  0.00727 ** 
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.073 on 2997 degrees of freedom
Multiple R-squared:  0.03673,	Adjusted R-squared:  0.03609 
F-statistic: 57.14 on 2 and 2997 DF,  p-value: < 2.2e-16
```

IV estimation

We can use the `ivreg` package to use the `voucher` as an instrument for `chocolate` consumption.

We can also verify that in this simple setup (where the instrument is a dummy variable), we can simply calculate four averages in the data and get **exactly** the same result - so we don't even need to run a regression!

Cheeky question: Can you come up with at least two reasons why we would still want to run a regression?

In [13]:
```
# Runnnig the IV regression
summary(ivreg(happiness ~ chocolate  | voucher , data=data_iv))



# Implementing the Wald estimator
a = mean(data_iv$happiness[data_iv$voucher==1])
  print(a)
b = mean(data_iv$happiness[data_iv$voucher==0])
  print(b)

c = mean(data_iv$chocolate[data_iv$voucher==1])
  print(c)
d = mean(data_iv$chocolate[data_iv$voucher==0])
  print(d)

wald_estimator = (a-b)/(c-d)
  print(wald_estimator)
```

```
Call:
ivreg(formula = happiness ~ chocolate | voucher, data = data_iv)

Residuals:
     Min       1Q   Median       3Q      Max 
-5.30086 -0.98411 -0.01969  0.99256  5.50574 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  0.52713    0.05003   10.54   <2e-16 ***
chocolate    1.02429    0.06850   14.95   <2e-16 ***

Diagnostic tests:
                  df1  df2 statistic p-value    
Weak instruments    1 2998     359.2  <2e-16 ***
Wu-Hausman          1 2997     390.0  <2e-16 ***
Sargan              0   NA        NA      NA    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.504 on 2998 degrees of freedom
Multiple R-Squared: -0.8934,	Adjusted R-squared: -0.894 
Wald test: 223.6 on 1 and 2998 DF,  p-value: < 2.2e-16
[1] 0.3186439
[1] -0.5027664
[1] -0.2035451
[1] -1.005476
[1] 1.024291
```

Two stage least squares

We have seen in class that we can also get the estimate from running two separate regressions and then getting the result as the ration between two OLS coefficients:

In [15]:
```
# Two-Stage least squares
reduced_form = summary(lm(happiness ~ voucher , data=data_iv))
print(reduced_form)
first_stage  = summary(lm(chocolate ~ voucher , data=data_iv))
print(first_stage)


tsls = reduced_form$coefficients[2,1] / first_stage$coefficients[2,1]
print(tsls)
```

```
Call:
lm(formula = happiness ~ voucher, data = data_iv)

Residuals:
    Min      1Q  Median      3Q     Max 
-3.6808 -0.6865  0.0049  0.6869  3.5050 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) -0.50277    0.02597  -19.36   <2e-16 ***
voucher      0.82141    0.03700   22.20   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.013 on 2998 degrees of freedom
Multiple R-squared:  0.1412,	Adjusted R-squared:  0.1409 
F-statistic: 492.9 on 1 and 2998 DF,  p-value: < 2.2e-16


Call:
lm(formula = chocolate ~ voucher, data = data_iv)

Residuals:
    Min      1Q  Median      3Q     Max 
-3.6845 -0.7832 -0.0153  0.8050  3.8856 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) -1.00548    0.02970  -33.86   <2e-16 ***
voucher      0.80193    0.04231   18.95   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.159 on 2998 degrees of freedom
Multiple R-squared:  0.107,	Adjusted R-squared:  0.1067 
F-statistic: 359.2 on 1 and 2998 DF,  p-value: < 2.2e-16

[1] 1.024291
```

Advantages of 2SLS

2SLS gives us several advantages:

- We can use **two instruments at the same time**: distance and voucher. This can help us get more precise estimates because we use more information on what determines chocolate consumption
- We can also **control for additional variables** that are important - such as, in our case, lactose intolerance
- We can directly **test whether instruments are relevant**. This is particularly useful if we have multiple instruments (how would we even do it otherwise?). The way we test this is by looking at the so-called "First stage F-statistic" or here, at the test for "Weak instruments".

In [16]:
```
summary(a <- ivreg(happiness ~ chocolate  | voucher , data=data_iv))
```

```r
# Including Distance as instrument
summary(b <- ivreg(happiness ~ chocolate  | distance , data=data_iv))
# Including both instrument
summary(c <- ivreg(happiness ~ chocolate  | voucher + distance , data=data_iv))
# Including control
summary(d <- ivreg(happiness ~ lactose_intolerant + chocolate   | voucher + dista

export_summs(a,b,c,d)
```

```
Call:
ivreg(formula = happiness ~ chocolate | voucher, data = data_iv)

Residuals:
     Min       1Q   Median       3Q      Max 
-5.30086 -0.98411 -0.01969  0.99256  5.50574 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  0.52713    0.05003   10.54   <2e-16 ***
chocolate    1.02429    0.06850   14.95   <2e-16 ***

Diagnostic tests:
                  df1  df2 statistic p-value    
Weak instruments    1 2998     359.2  <2e-16 ***
Wu-Hausman          1 2997     390.0  <2e-16 ***
Sargan              0   NA        NA      NA    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.504 on 2998 degrees of freedom
Multiple R-Squared: -0.8934,	Adjusted R-squared: -0.894 
Wald test: 223.6 on 1 and 2998 DF,  p-value: < 2.2e-16
Call:
ivreg(formula = happiness ~ chocolate | distance, data = data_iv)

Residuals:
     Min       1Q   Median       3Q      Max 
-5.43300 -1.01154 -0.02154  1.01202  5.56888 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  0.55089    0.07253   7.595 4.07e-14 ***
chocolate    1.06321    0.10956   9.705  < 2e-16 ***

Diagnostic tests:
                  df1  df2 statistic p-value    
Weak instruments    1 2998     137.1  <2e-16 ***
Wu-Hausman          1 2997     151.2  <2e-16 ***
Sargan              0   NA        NA      NA    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.538 on 2998 degrees of freedom
Multiple R-Squared: -0.9794,	Adjusted R-squared: -0.9801 
Wald test: 94.18 on 1 and 2998 DF,  p-value: < 2.2e-16
```

```
Call:
ivreg(formula = happiness ~ chocolate | voucher + distance, data = data_iv)

Residuals:
     Min       1Q   Median       3Q      Max 
-5.33925 -0.98975 -0.02054  0.99758  5.52409 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  0.53404    0.04493   11.89   <2e-16 ***
chocolate    1.03560    0.05803   17.85   <2e-16 ***

Diagnostic tests:
                  df1  df2 statistic p-value    
Weak instruments    2 2997   266.611  <2e-16 ***
Wu-Hausman          1 2997   637.897  <2e-16 ***
Sargan              1   NA     0.092   0.761    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.514 on 2998 degrees of freedom
Multiple R-Squared: -0.918,     Adjusted R-squared: -0.9186 
Wald test: 318.5 on 1 and 2998 DF,  p-value: < 2.2e-16
Call:
ivreg(formula = happiness ~ lactose_intolerant + chocolate | 
    voucher + distance + lactose_intolerant, data = data_iv)

Residuals:
     Min       1Q   Median       3Q      Max 
-5.08936 -0.96582 -0.03077  0.94779  5.04358 

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)    
(Intercept)         0.05495    0.03764    1.46    0.144    
lactose_intolerant  0.94259    0.07475   12.61   <2e-16 ***
chocolate           1.02798    0.05445   18.88   <2e-16 ***

Diagnostic tests:
                  df1  df2 statistic p-value    
Weak instruments    2 2996   334.962  <2e-16 ***
Wu-Hausman          1 2996   633.774  <2e-16 ***
Sargan              1   NA     0.056   0.813    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.432 on 2997 degrees of freedom
Multiple R-Squared: -0.7154,    Adjusted R-squared: -0.7166 
Wald test:   179 on 2 and 2997 DF,  p-value: < 2.2e-16
```

```
Registered S3 methods overwritten by 'broom':
  method            from   
  tidy.glht         jtools
  tidy.summary.glht jtools
```

A huxtable: 17 × 5

	names	Model 1	Model 2	Model 3	
	<chr>	<chr>	<chr>	<chr>	
		Model 1	Model 2	Model 3	
1	(Intercept)	0.527133275522715 ***	0.550892307613025 ***	0.53403668341017 ***	0.05495
2		(0.0500264760211853)	(0.0725296359271811)	(0.0449305453865909)	(0.037639
3	chocolate	1.02429063664533 ***	1.06321489358638 ***	1.03560044174847 ***	1.02798
4		(0.0685026617240189)	(0.109557616518632)	(0.0580305178439754)	(0.054452
5	lactose_intolerant				0.9425
6					(0.07475
1.1	nobs	3000	3000	3000	
2.1	r.squared	-0.89341100244053	-0.979414098772623	-0.91800718472674	-0.7154
3.1	adj.r.squared	-0.894042560480037	-0.980074343635456	-0.91864694696314	-0.7165
4.1	sigma	1.50427874523487	1.538063268442	1.51401781632436	1.432
5.1	statistic	223.579258814953	94.1796240401501	318.472224853309	178.9
6.1	p.value	8.24775765906999e-49	6.03885534040896e-22	8.86500481357345e-68	3.697497
7	df	2	2	2	
8	df.residual	2998	2998	2998	
9	nobs.1	3000	3000	3000	
.1	*** p < 0.001; ** p < 0.01; * p < 0.05.	*** p < 0.001; ** p < 0.01; * p < 0.05.	*** p < 0.001; ** p < 0.01; * p < 0.05.	*** p < 0.001; ** p < 0.01; * p < 0.05.	*** p < 0.

```
In [ ]:
```