Lecture_4_MACSS

Auffhammer

2025-09-16

Now let's get ambitious. We are going to generate some data on health status of some made up folks and play random sampling and random assignment. This forst exercise lets you simulate what happens if insurance is randomly assigned, versus if it is not.... It will also show us how to do the t-test to compare group means and how to use a canned routine to calculate these automatically, which is generally better (since the right test depends on the nature of the random variable you are looking at.)

```
rm(list = ls()) # clear memory
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(MASS)
```

```
##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select
```

```
library(crosstable)
library(flextable)
library(ggplot2)
library(plyr)
```

```
## ------------------------------------------------------------------------------

## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)

## ------------------------------------------------------------------------------

##
## Attaching package: 'plyr'

## The following object is masked from 'package:crosstable':
##
##     compact
```

```
## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```r
set.seed(22092008) # set random number generator seed
n <- 1000 # Sample Size
mu <- c(0, 0,0)
# a <- 0.5 #Gender Income Covariance
# b <- 0.1 #Gender Insurance Covariance
# c <- 0.8 #Income Insurance

# If insurance were randomized
a <- 0.5 # Set to 0.5 as default
b <- 0.0 # Set to 0.1 as default
c <- 0.0 # Set to 0.8 as default

# Some betas for later
b1 <-1 #Gender Beta
b2 <-5 # Income Beta
b3 <-3 #Insurance
shifter <- 30
Sigma <- matrix(c(1, a, b, a, 1, c,b, c, 1), nrow=3)
data = mvrnorm(n, mu, Sigma, empirical=FALSE)
Gender = data[, 1]  # standard normal (mu=0, sd=1)
Income = data[, 2]  # standard normal (mu=0, sd=1)
Insurance= data[, 3]  # standard normal (mu=0, sd=1)

# Gender and income should be binary
Ins <- Insurance>0
Gend <- Gender>0
cor(Ins,Gend)
```

```
## [1] 0.04729256
```

```r
cor(Ins,Income)
```

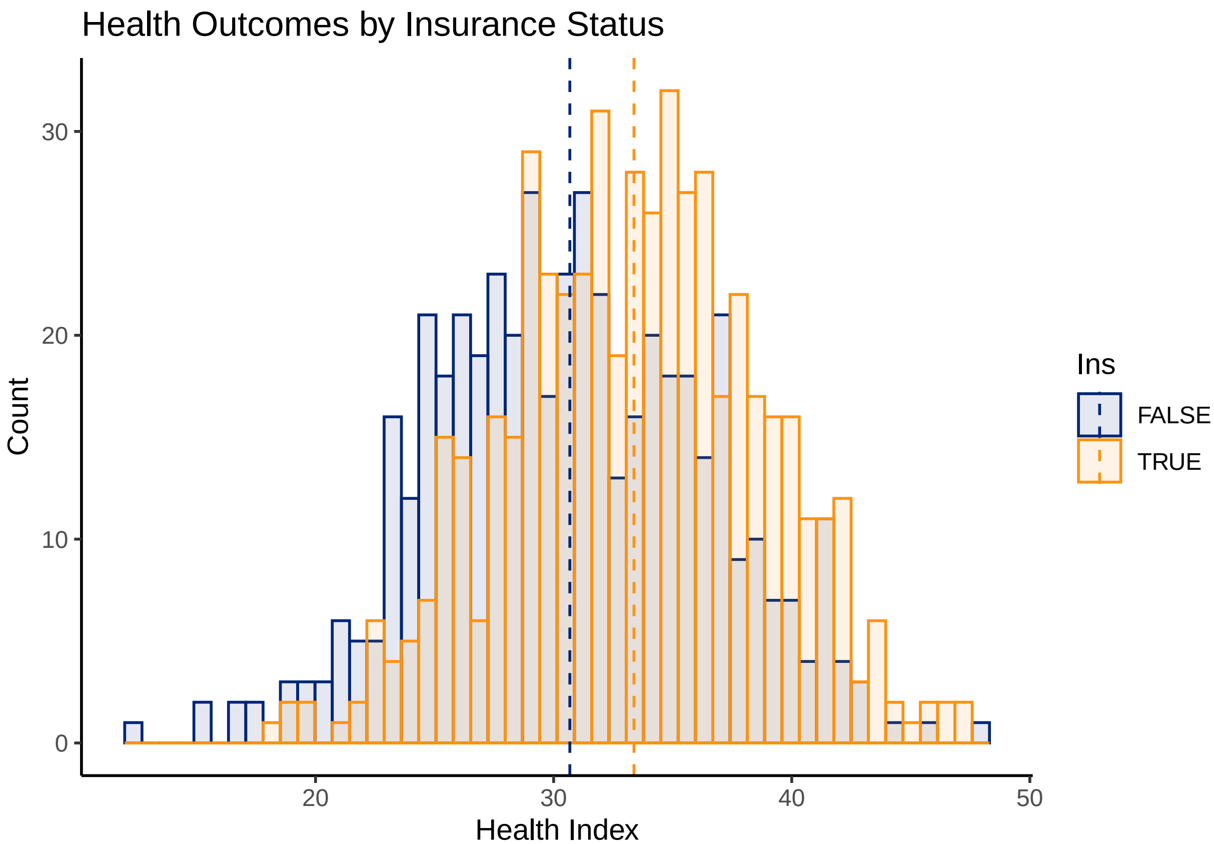
```
## [1] -0.04325857
```

```r
cor(Gend,Income)
```

```
## [1] 0.3974529
```

```r
# We are going to generate some arbitrary Health Index
Health <- shifter +  rnorm(n,mean=0,sd=1) + b1*Gend + b2*Income + b3* Ins

# Let's do a manual comparison of Health Across the Insured and Uninsured.
mydata <- data.frame(Income, Gend, Health, Ins)
# Calculate Means by Group (using ddply).
mu <- ddply(mydata, "Ins", summarise, grp.mean=mean(Health))

# Plot my health outcome by Insurance Status in Pretty Graph
ggplot(mydata, aes(x=Health, color=Ins, fill=Ins)) +
  scale_color_manual(values=c("#002676", "#FC9313")) +
  scale_fill_manual(values=c("#002676", "#FC9313")) +
  geom_histogram(alpha=0.1, position="identity", bins=50)+
  geom_vline(data=mu, aes(xintercept=grp.mean, color=Ins),
```

```
             linetype="dashed") +
  labs(title="Health Outcomes by Insurance Status",x="Health Index", y = "Count")+
  theme_classic()
```




```
# Let's compare across treatment
# First - do this by hand. Difference in means. Unknown and uneuqal variances.

M1 <- mean(mydata[Ins == 'TRUE', 'Health'])
M2 <- mean(mydata[Ins == 'FALSE', 'Health'])
n1 <- sum(Ins)
n2 <- n-n1
V1 <- var(mydata[Ins == 'TRUE', 'Health'])
V2 <- var(mydata[Ins == 'FALSE', 'Health'])
S <- sqrt((V1 / n1) + (V2 / n2))
statistic <- (M1 - M2 - 0) / S
print(statistic)
```

```
## [1] 7.652668
```

```
t_health <- t.test(Health ~ Ins)
print(t_health)
```

```
##
##  Welch Two Sample t-test
##
## data:  Health by Ins
## t = -7.6527, df = 969.8, p-value = 4.742e-14
## alternative hypothesis: true difference in means between group FALSE and group TRUE is not equal to 0
```

```
## 95 percent confidence interval:
##  -3.384072 -2.002712
## sample estimates:
## mean in group FALSE  mean in group TRUE
##            30.68058            33.37397
```

```
t_inc <- t.test(Income ~ Ins)
print(t_inc)
```

```
##
##  Welch Two Sample t-test
##
## data:  Income by Ins
## t = 1.3635, df = 972.55, p-value = 0.173
## alternative hypothesis: true difference in means between group FALSE and group TRUE is not equal to (
## 95 percent confidence interval:
##  -0.03967316  0.22033754
## sample estimates:
## mean in group FALSE  mean in group TRUE
##          0.04924931         -0.04108288
```

```
my_test_args=crosstable_test_args(show_method=FALSE)
ft1 <- crosstable(mydata,by="Ins", test=TRUE, funs=c(mean=mean),test_args=my_test_args) %>%
  as_flextable()
```

```
## Warning in crosstable(mydata, by = "Ins", test = TRUE, funs = c(mean = mean), : Be aware that automa
## context, as it would cause extensive alpha inflation otherwise.
## This warning is displayed once every 8 hours.
```

```
 print (ft1)
```

```
## a flextable object.
## col_keys: `label`, `variable`, `FALSE`, `TRUE`, `test`
## header has 2 row(s)
## body has 4 row(s)
## original dataset sample:
## 'data.frame':    4 obs. of  6 variables:
##  $ .id     : chr  "Income" "Gend" "Gend" "Health"
##  $ label   : chr  "Income" "Gend" "Gend" "Health"
##  $ variable: chr  "mean" "FALSE" "TRUE" "mean"
##  $ FALSE   : chr  "0.05" "246 (50.00%)" "230 (45.28%)" "30.7"
##  $ TRUE    : chr  "-0.04" "246 (50.00%)" "278 (54.72%)" "33.4"
##  $ test    : chr  "0.1717" "0.1348" "0.1348" "<0.0001"
##  - attr(*, "debug")=List of 3
##   ..$ interface: chr "quosure"
##   ..$ x_class  : Named chr [1:3] "numeric" "character" "numeric"
##   .. ..- attr(*, "names")= chr [1:3] "Income" "Gend" "Health"
##   ..$ y_class  : Named chr "character"
##   .. ..- attr(*, "names")= chr "Ins"
##  - attr(*, "N")= int 1000
##  - attr(*, "showNA")= chr "ifany"
##  - attr(*, "variables")= chr [1:3] "Income" "Gend" "Health"
##  - attr(*, "has_test")= logi TRUE
##  - attr(*, "has_effect")= logi FALSE
##  - attr(*, "has_total")= num 0
##  - attr(*, "has_label")= logi TRUE
```

```
##  - attr(*, "by")= chr "Ins"
##  - attr(*, "by_label")= Named chr "Ins"
##   ..- attr(*, "names")= chr "Ins"
##  - attr(*, "by_table")= 'table' int [1:2(1d)] 476 524
##   ..- attr(*, "dimnames")=List of 1
##   .. ..$ Ins: chr [1:2] "FALSE" "TRUE"
##  - attr(*, "by_levels")=List of 1
##   ..$ Ins: chr [1:2] "FALSE" "TRUE"
```

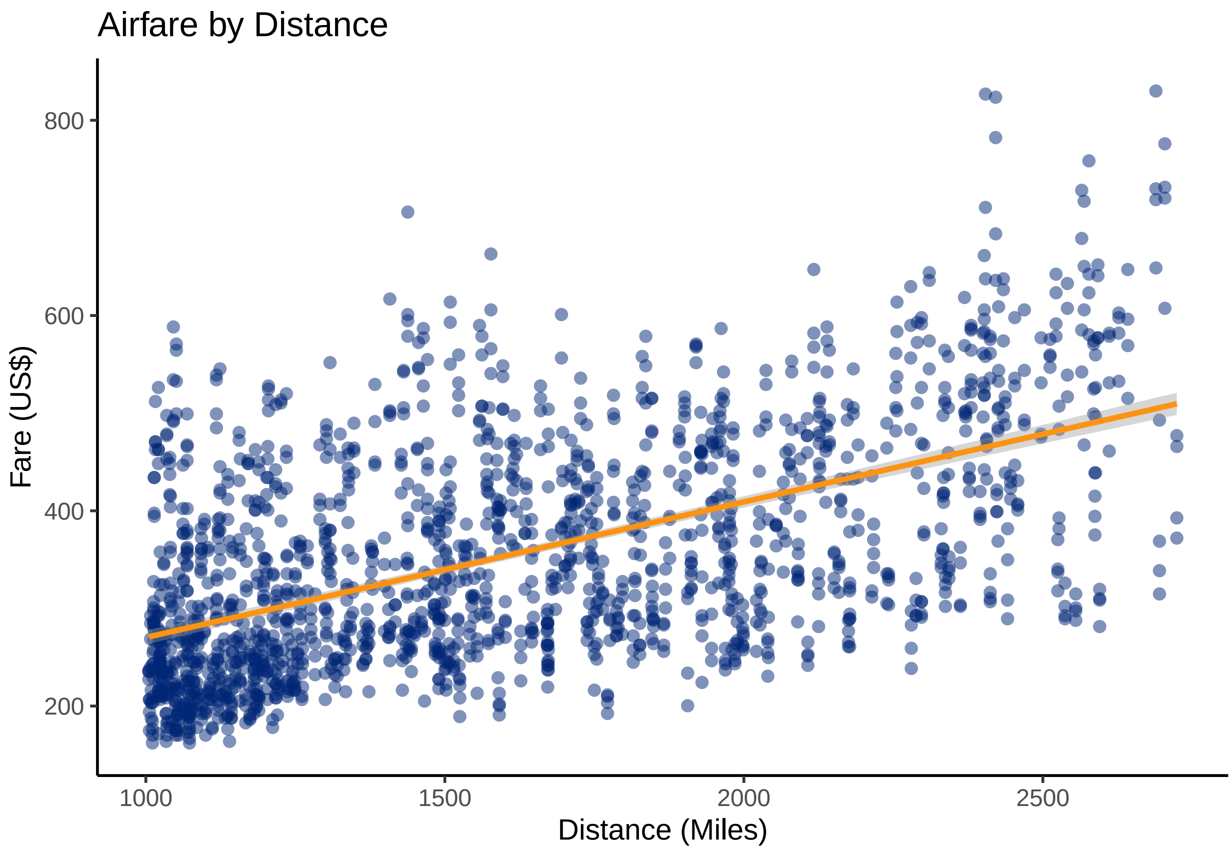
Now let's turn to some simple regression analysis. It is so simple, my teenager can do it. In fact, I checked and he can. That said, interpreting what it tells you is going to be the art form.

```r
rm(list = ls()) # clear memory
library(dplyr)
library(MASS)
library(stargazer) # For pretty Regression Tables
```

```
##
## Please cite as:

##  Hlavac, Marek (2022). stargazer: Well-Formatted Regression and Summary Statistics Tables.

##  R package version 5.2.3. https://CRAN.R-project.org/package=stargazer
```

```r
setwd("/Users/auffhammer/Library/CloudStorage/Dropbox/06_Teaching/MACSS/2024/code/public-repository-1/we

airfares <- read.csv("airfares.csv")
#Let's plot some data.
ggplot(airfares, aes(x=dist, y=fare)) +
  geom_point(alpha=0.5, shape=16, fill="#002676", color="#002676", size=2)+
  geom_smooth(method=lm, color="#FC9313")+
  labs(title="Airfare by Distance",
       x="Distance (Miles)", y = "Fare (US$)")+
  theme_classic()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



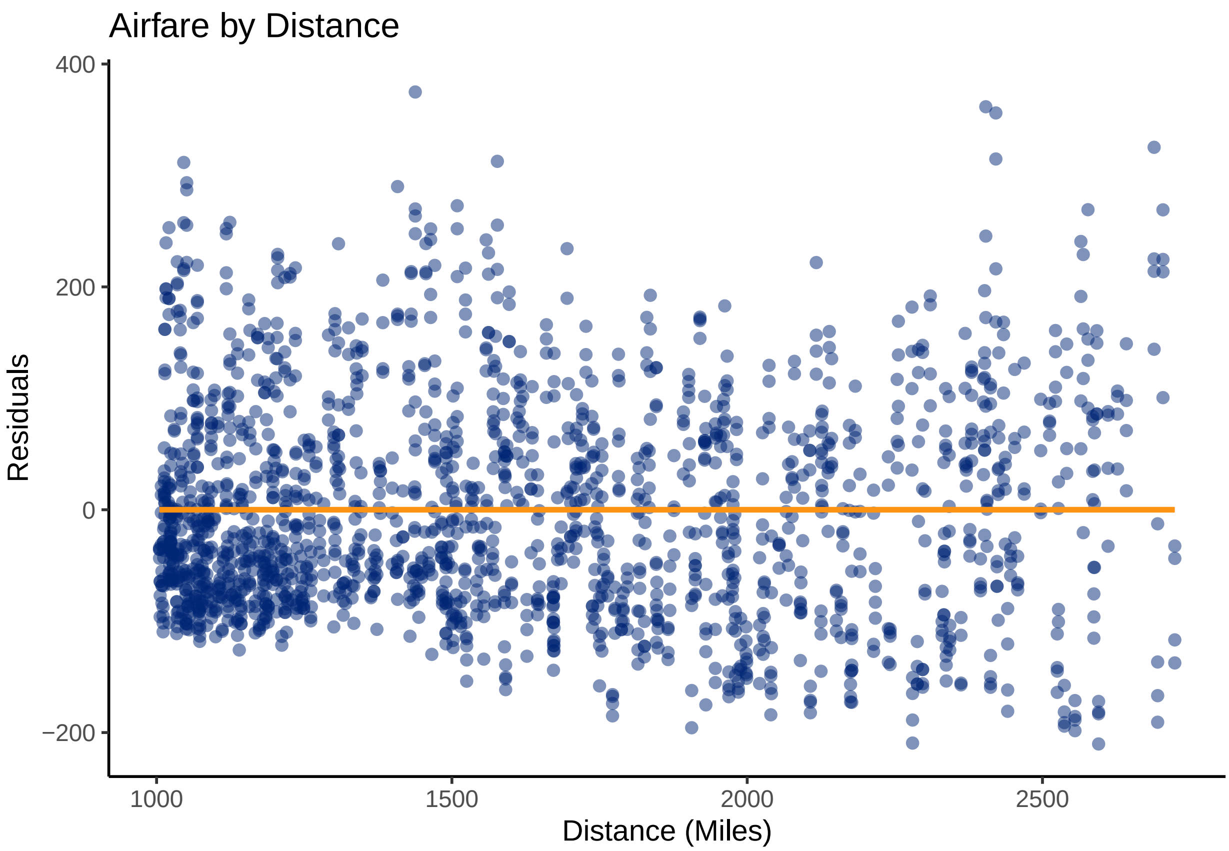

```r
#Let's run a regression.
planes <- lm(airfares$fare ~ airfares$dist)
# Let's record some residuals and join them to our data frame.
airfares$res <- planes$resid

#Let's make some nice looking regression output.

stargazer(planes, type='text', digits = 3, title = 'Linear Airfare Distance Regression', style = 'qje')
```

```
##
## Linear Airfare Distance Regression
## ==========================================================
##                                       fare
## ----------------------------------------------------------
## dist                                0.139***
##                                     (0.005)
##
## Constant                           131.434***
##                                     (7.993)
##
## N                                    1,820
## R2                                   0.318
## Adjusted R2                          0.318
## Residual Std. Error           97.587 (df = 1818)
## F Statistic               847.869*** (df = 1; 1818)
## ==========================================================
## Notes:                ***Significant at the 1 percent level.
```

```
##                           **Significant at the 5 percent level.
##                           *Significant at the 10 percent level.
```

```
# Plot Residuals - Playing with colors (HEX Colors - official Cal!
# Also meesing with background and Axis Labels. )
ggplot(airfares, aes(x=dist, y=res)) +
  geom_point(alpha=0.5, shape=16, fill="#002676", color="#002676", size=2)+
  geom_smooth(method=lm, se=FALSE, color="#FC9313")+
  labs(title="Airfare by Distance",
       x="Distance (Miles)", y = "Residuals")+
  theme_classic()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

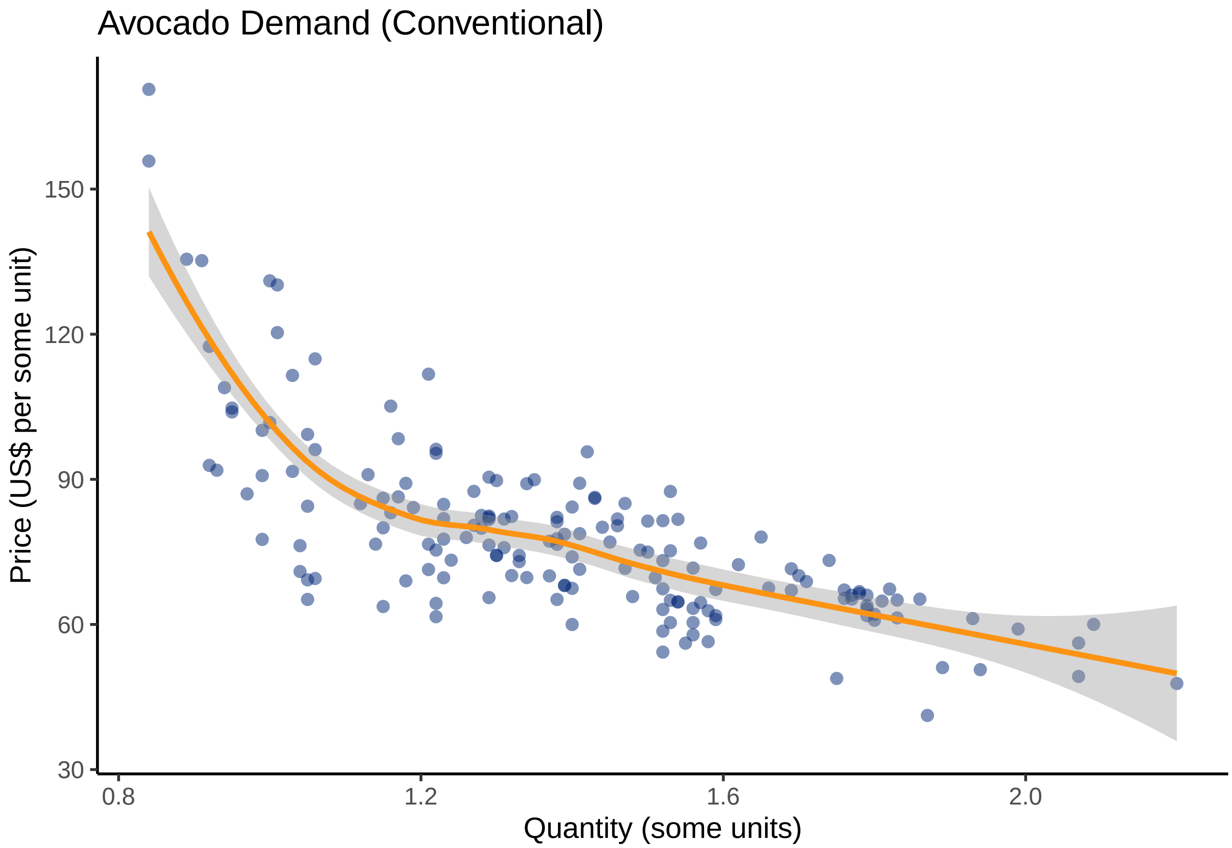



```
# Much nicer than the junk I showed in lecture. Apologies.
```

```
library(dplyr)
library(MASS)
library(stargazer) # For pretty Regression Tables
setwd("/Users/auffhammer/Library/CloudStorage/Dropbox/06_Teaching/MACSS/2024/code/public-repository-1/w

avocado <- read.csv("avocado.csv")

#Let's plot some data. I am fitting a smoother (loess) to the data to see what the functional # form lo

ggplot(avocado, aes(x=price_reg, y=quantity_reg)) +
  geom_point(alpha=0.5, shape=16, fill="#002676", color="#002676", size=2)+
  geom_smooth(method=loess, color="#FC9313")+
```

```r
  labs(title="Avocado Demand (Conventional)",
       x="Quantity (some units)", y = "Price (US$ per some unit)")+
  theme_classic()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



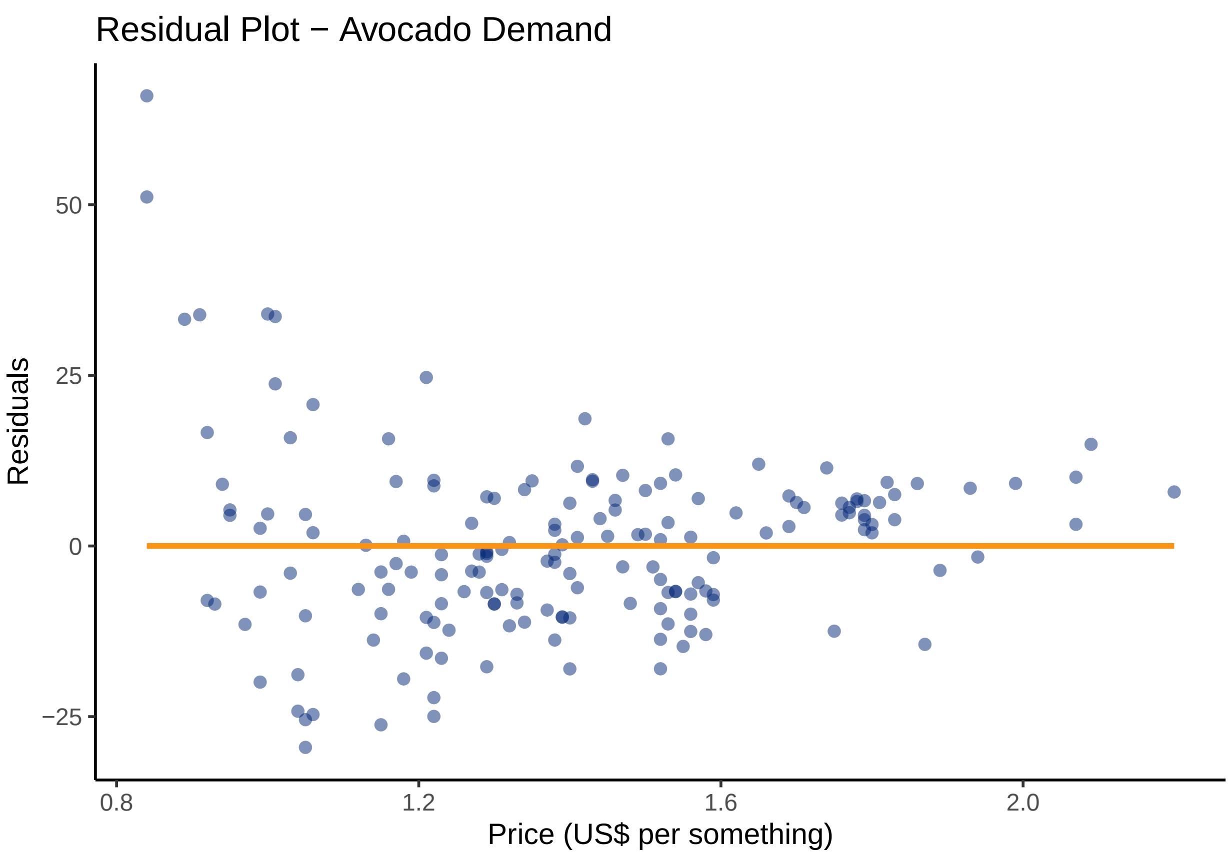

```r
# Now run a regression of the linear model No transformation.
avo_lin <- lm(avocado$quantity_reg ~ avocado$price_reg)

# Let's record some residuals and join them to our data frame.
avocado$res <- avo_lin$resid

#Let's make some nice looking regression output.
stargazer(avo_lin, type='text', digits = 3, title = 'Linear Price Regression', style = 'qje')
```

```
##
## Linear Price Regression
## ==========================================================
##                                  quantity_reg
## ----------------------------------------------------------
## price_reg                         -47.641***
##                                    (3.544)
##
## Constant                          144.695***
##                                    (5.055)
##
## N                                    169
```

```
## R2                                  0.520
## Adjusted R2                         0.517
## Residual Std. Error           13.263 (df = 167)
## F Statistic                180.722*** (df = 1; 167)
## ==========================================================
## Notes:                 ***Significant at the 1 percent level.
##                         **Significant at the 5 percent level.
##                          *Significant at the 10 percent level.
```

```
# Plot Residuals - same pretty graph as before.
ggplot(avocado, aes(x=price_reg, y=res)) +
  geom_point(alpha=0.5, shape=16, fill="#002676", color="#002676", size=2)+
  geom_smooth(method=lm, se=FALSE, color="#FC9313")+
  labs(title="Residual Plot - Avocado Demand",
       x="Price (US$ per something)", y = "Residuals")+
  theme_classic()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```




```
# Transform our variables using natural logs.
avocado$l_price <- log(avocado$price_reg)
avocado$l_q <- log(avocado$quantity_reg)

# Run the log log regression.
avo_log <- lm(avocado$l_q ~ avocado$l_price)

# Let's record some residuals and join them to our data frame.
avocado$resl <- avo_log$resid
```

```r
# Plot Residuals - same pretty graph as before.
ggplot(avocado, aes(x=l_price, y=resl)) +
  geom_point(alpha=0.5, shape=16, fill="#002676", color="#002676", size=2)+
  geom_smooth(method=lm, se=FALSE, color="#FC9313")+
  labs(title="Residual Plot - Log-Log Avocado Demand Residuals",
       x="Log(Price) (US$ per something)", y = "Residuals")+
  theme_classic()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```