Problem Set 2-Final

Olivia Bogiages

2025-10-23

```
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("ggplot2")
library(tidyr)
#set seed for reproducibility
set.seed(123)

#Establish Population and Characteristics
#population size
population_size<-10000

#create population and traits and distribution of traits
Eye_Color <- c("Blue", "Green","Brown","Hazel","Purple")
proportion_population <- c(.2 , .1 , .3 , .3, .1)

#Proportions of Eye Color Combined
names(proportion_population)<-Eye_Color

population<-tibble(
id=1:population_size,
trait= sample(Eye_Color,
size=population_size, replace=TRUE,
prob=
  proportion_population)
)
#randomly assign either treatment or control with equal probability,
#randomly sample n from population
#repeat the process of random assignment many times
random_assignment<-function(n)
  {sampled_rows<-population[sample(1:nrow(population), size=n, replace=TRUE),]
 sampled_rows%>%
mutate(group=sample(c("Treatment", "Control"), size= n, replace=TRUE))
       }
```

```r
#show as n increases, distribution of traits in sample has similar
#proportions to population
#increasing population size
n_size<-c(100, 300, 500,1000,5000)
iterations<-100

#store results for n_sizes, loop for differnt sample sizes
results_list<-list()
for (n in n_size) {
  all_iterations<-data.frame()

for (i in 1:iterations) {
  diff_n<-random_assignment(n)%>%
    mutate(sample_size=n, iteration=i)
  all_iterations<-bind_rows(all_iterations, diff_n)
}


#Population traits distribution
pop_traits<-all_iterations %>%
count(trait) %>% mutate(group="Sample", prop=n/sum(n)) %>%
select(group, trait, prop)
#Treatment Group proportions
treat_traits <-all_iterations%>%
filter(group=="Treatment")%>%
  count(trait)%>%
  mutate(group="Treatment",prop=n/sum(n))%>%
  select(group, trait, prop)
#Control Group Proportions
control_traits<-all_iterations%>%
  filter(group=="Control")%>%
  count(trait)%>%
  mutate(group="Control", prop=n/sum(n))%>%
  select(group, trait, prop)
#Combine results
summary_data<-bind_rows(pop_traits, treat_traits, control_traits)%>%
  mutate(sample_size=n)

results_list[[as.character(n)]]<-summary_data
}

final_results<-bind_rows(results_list)

# show as n increases similar proportion between sample and population
#proportions for population for frequency table
pop_dist<-tibble(
  trait=names(proportion_population),
   pop_prop=proportion_population)
#sample population proportions for table
avg_sample_prop<-final_results%>%
  filter(group=="Sample")%>%
  group_by(sample_size, trait)%>%
  summarise(avg_prop=mean(prop), .groups="drop")
```

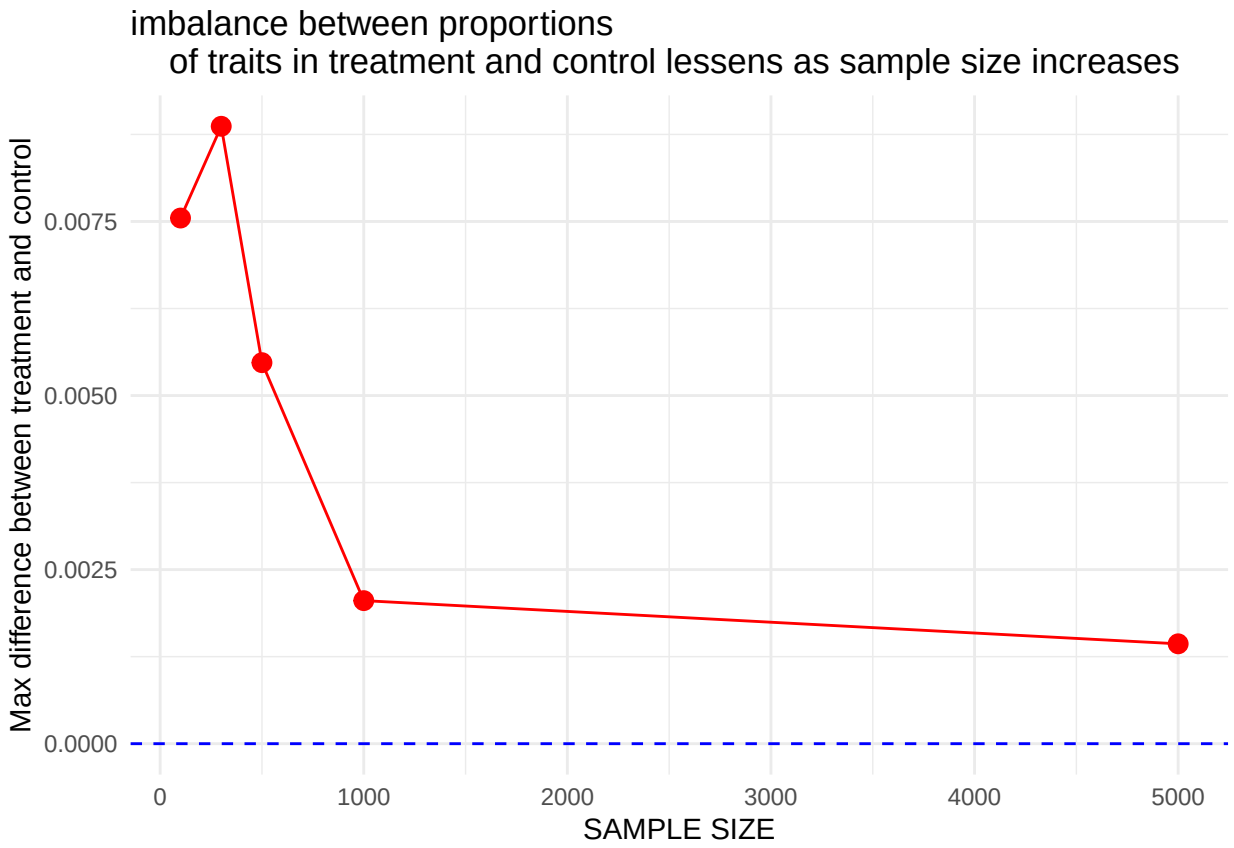
```r
#both sets of proportions combined
both_prop_freq_table<-avg_sample_prop%>%
  left_join(pop_dist, by="trait")%>%
  arrange(sample_size, trait)

#include all sample sizes intable
freq_table_wide<-both_prop_freq_table%>%
  select(sample_size, trait, avg_prop, pop_prop)%>%
  pivot_wider(
    names_from=sample_size,
    values_from=avg_prop,
  )%>%
  arrange(trait)
print(freq_table_wide)
```

```
## # A tibble: 5 x 7
##   trait  pop_prop  `100`  `300`  `500` `1000` `5000`
##   <chr>     <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
## 1 Blue        0.2 0.200  0.199  0.202  0.199  0.201
## 2 Brown       0.3 0.304  0.303  0.302  0.304  0.304
## 3 Green       0.1 0.0933 0.0929 0.0939 0.0946 0.0942
## 4 Hazel       0.3 0.304  0.304  0.303  0.301  0.300
## 5 Purple      0.1 0.0999 0.101  0.0994 0.101  0.101
```

```r
##as n increases, similar proportions between treatment and control groups
imbalance<-final_results %>%
  filter(group %in% c("Treatment","Control")) %>%
  rename(cat=trait, n=sample_size)%>%
  select(n, cat, group, prop)%>%
  pivot_wider(names_from=group, values_from=prop)%>%
  mutate(abs_diff=abs(Treatment-Control))%>%
  group_by(n)%>%
  summarise(
    max_abs_diff=max(abs_diff),
    sum_diff=sum(abs_diff),
    .groups="drop"
  )

#Plot showing
ggplot(imbalance, aes(x=n, y=max_abs_diff))+
  geom_hline(yintercept=0, linetype=2, color="blue")+
  geom_point(size=3, color="red")+
  geom_line(color="red")+
  labs(
    x="SAMPLE SIZE",
    y=("Max difference between treatment and control"),
    title="imbalance between proportions
    of traits in treatment and control lessens as sample size increases"
  )+
    theme_minimal()
```




```
#################
```

```
##Data Analysis
voting<-read.csv("https://raw.githubusercontent.com/MLBurnham/pols_602/refs/heads/main/data/voting.csv")
#1. What is the Treatment variable? Is it discrete or continuous?
#What is the variable's data type?
#The treatment variable is if the individual recieved the message communicating
#social pressure about voting. Specifically, they had four treatment groups,
#with four different mailers sent, all communicating about civic duty, but with
#different degrees of social pressure emphasized. This variable is discrete and
#the data type is categorical.


##Create a new treatment variable in the data frame that is a binary version
#of the existing treatment variable. (1 if treated, 0 if not)
voting$treated<-ifelse(voting$message=="yes",1,0)
##Compute average outcome for treatment group and average outcome for control.
#Interpret the results (1-2sentences)
mean(voting$voted)
```

```
## [1] 0.3101759
```

```
mean(voting$control)
```

```
## Warning in mean.default(voting$control): argument is not numeric or logical:
## returning NA
```

```
## [1] NA
```

```
#Use brackets to subset the data frame, create 2 new data frames.
#one for treatment and one for control
treatment<-voting[voting$treated==1,]
control<-voting[voting$treated==0,]

#what is the average birth year for treatment and control?

c("average birth year for treatment group"=mean(treatment$birth),
"average birth year for control group"=mean(control$birth))
```

```
## average birth year for treatment group   average birth year for control group
##                                1956.147                                1956.186
```

```
##What is the estimated causal effect for this experiment?provided calculated
#average and a substantive interpretation use the difference in means estimator
mean(treatment$voted)-mean(control$voted)
```

```
## [1] 0.08130991
```

```
#The average causal effect for this experiment is .081, found using
#the difference in means estimator. This tells us that, on average, recieving
#a mailer communicating about civic duty and some form of social pressure
#increased a person's likelihood of voting by 8.1 percentage points.

#Suppose we want to say the average causal effect is an estimand effect for the
#entire US population. What assumption would need to hold to make this claim?

#The assumption that would need to hold is that the population sampled is
#comparable to the US population in order to generalize the average causal
#effect in the experiment to an estimand effect for all of the US.
#In this case, the experiment was only run in MI and it was conducted
#solely among households. Other states and different living arrangements,
#like apartment complexes, are therefore not accounted for making it the case,
#that this assumption, that the sample is comparable to the US population,
#does not hold.
```

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

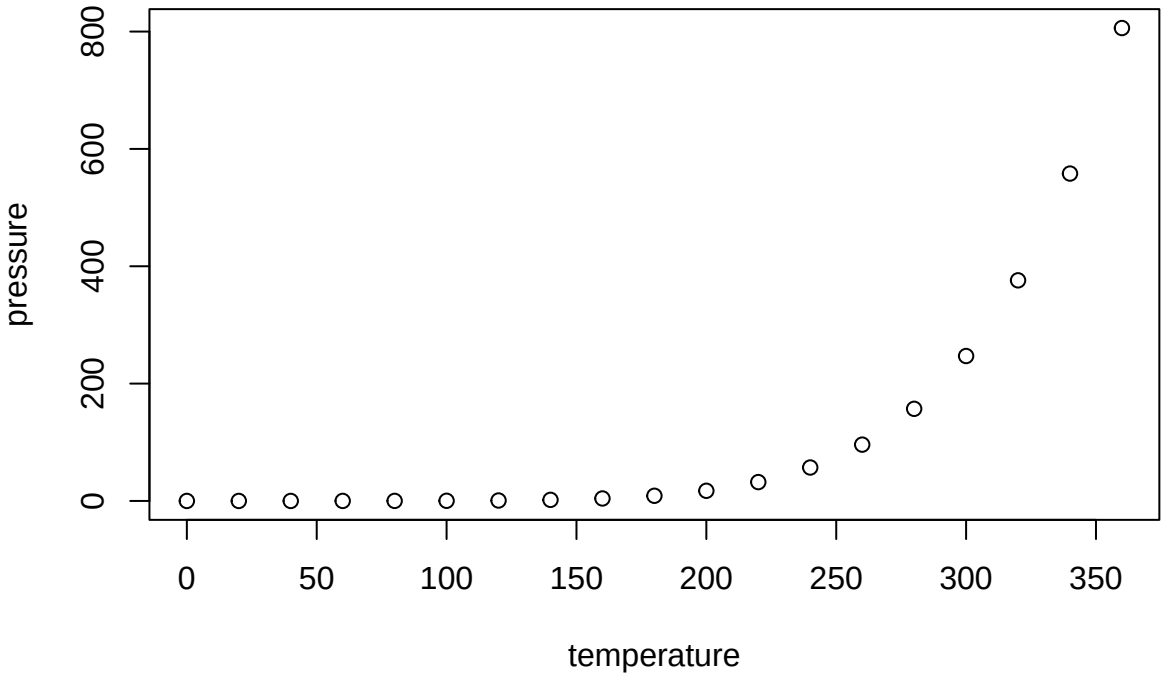
When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
summary(cars)
```

```
##      speed           dist
##  Min.   : 4.0   Min.   :  2.00
##  1st Qu.:12.0   1st Qu.: 26.00
##  Median :15.0   Median : 36.00
##  Mean   :15.4   Mean   : 42.98
##  3rd Qu.:19.0   3rd Qu.: 56.00
##  Max.   :25.0   Max.   :120.00
```

Including Plots

You can also embed plots, for example:



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.