Comparing Bootstrap-t and Parametric Confidence Intervals for the Estimated Mean of Emergency Department Visits

Bingtang Huang
Pete Pham

Portland State University
Stat 573 Final Project
Dr. Jong Kim
December 4th, 2024

I. Introduction

Background

Diabetes rates among adults has been steadily increasing for the past decade. It is estimated that in the next 10 years, the number of diabetic adults in the world will increase by approximately 20%[1]. According to the Centers for Disease Control (CDC), 9% of the US adult population have been diagnosed with diabetes while nearly 30% of adults have an official diagnosis of pre-diabetes[2]. The same CDC report lists diabetes as the eighth leading cause of death in 2022 and points to the disease as the primary cause of 103,294 deaths.

An economic study examining the cost of diabetes estimates that nearly 1 in 4 health care dollars spent is spent on diabetes[3]. Additionally, Parker et al. conclude that people diagnosed with diabetes have medical expenditures 2.6 times higher than those without the condition.

If not managed in a timely manner, diabetes can lead to long-term chronic illnesses and comorbidities, increasing both morbidity and healthcare costs. The long-term nature of the disease have drawn attention from government agencies and public health organizations aiming to reduce the strain on overburdened healthcare systems. One strategy is to reduce emergency department utilization by increasing outpatient engagement with primary care and specialty physicians. Higher engagement with outpatient settings is thought to provide better avenues for preventative care and reduction in preventable ED visits.

Purpose

Type II diabetes can be prevented or managed through lifestyle change[4]. However, diabetes is generally estimated to be underdiagnosed. In resource-limited public health settings, interventionalists often relay on established parametric methods to estimate population parameters (e.g. mean ED visits, prescriptions, cost, etc). Bootstrapping offers a nonparametric alternative, providing robust estimates for population estimators without requiring assumptions or prior knowledge of the underlying population distribution.

This paper seeks to compare 95% CIs for the mean number of ER visits using normal, Poisson, and bootstrap-t approaches for both high and low outpatient (OP) visit groups. We look to evaluate the merits of each approach.

[1] *IDF Diabetes Atlas, 10th Edition* (2021)
[2] *National Diabetes Statistics Report* (2024)
[3] Bannuru (2023)
[4] Stephanie M Gruss (2019)

II. Data

Population of Interest

In this case, our data comes from a well-defined natural population: Medicare-enrolled members with a diabetes diagnosis. Though there is some shift in year-to-year enrollment, Medicare eligibility has been fairly consistent over many decades. Approaching this data from an experiment sense, we are interested in the two prospective populations. Specifically:

- Medicare Diabetic Population → Treatment A: Higher Outpatient (OP) Visits → Prospective Population A
- Medicare Diabetic Population → Treatment B: Low Outpatient (OP) Visits → Prospective Population B

We are interested to know if more outpatient visits are an influencing factor on the count of ER visits.

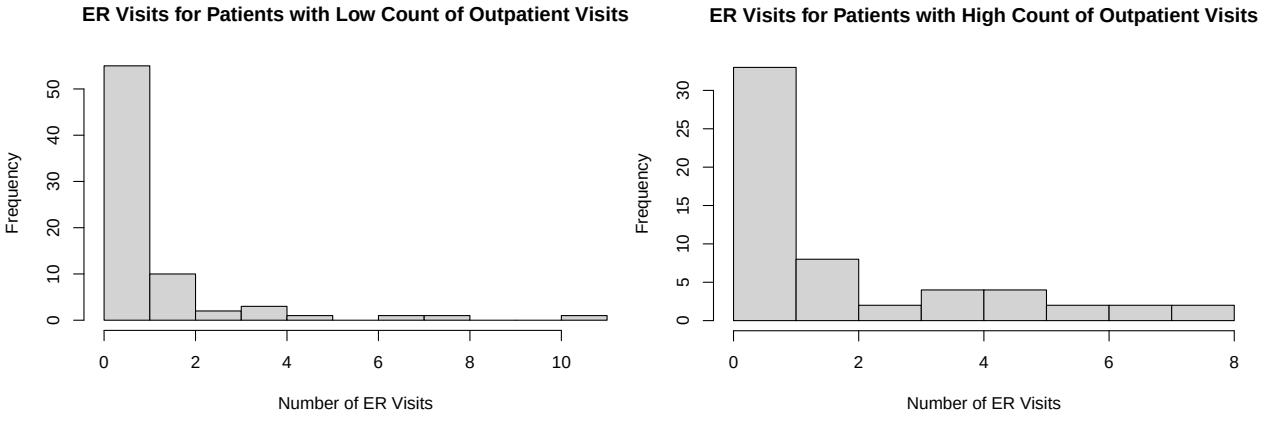
Sample

Our dataset contains a random sample of 131 diabetes patients who received medical treatment for diabetes at a hospital system in 2005. Of those patients, 57 patients were categorized as having high engagement (>=14 visits) and 74 patients were categorized as low engagement(<14 visits).

The table below provides a descriptive summary of the sample set:

Engagement Group	# of Patients	# ER Visits	Mean ER Visits	Variance ER Visits
High OP Engagement	57	110	1.930	5.531
Low OP Engagmeent	74	86	1.162	3.864

In the table above, patients with high OP engagement, though fewer in number, have a higher total (110 vs. 86) and mean (1.929 vs. 1.162) of ER visits. The high OP group also has a higher variance in ER visits. Additionally, the low OP group shows more extreme values in ER visits, with a greater range of 11, compared to 8 in the high OP group.



The histograms, above, show that the low PCP visit group is more clustered around lower counts of ER visits with a few extreme higher values. In comparison, the high OP group is slightly more distributed along its range. At face value, the highly engaged group has a higher mean number of ER visits. This goes

against conventional understanding of primary care engagement and has a myriad of explanations. This could potentially be skewed due to the needs of complex patients inherently requiring more attention. This will be further discussed later in this paper.

III. Methods

Count data is commonly recorded in the healthcare settings. These data are typically skewed right with high frequencies of zero counts. Our ER Visit attribute is an excellent example of this (see histogram above). In fast paced public health settings, many administators commonly default to parametric methods without consideration for distribution specific assumptions. For this reason, we'll compare the normal, Poisson, and bootstrap-t 95% confidence intervals for an estimated population mean.

Normal distribution based approaches are exceptionally common when it comes to evaluating health care data (e.g. lean six sigma). It's application for our data set is not appropriate as our data is asymmetrical, discrete, and heavily skewed. However, it is still often used as a first step given its ubiquity in many healthcare programs. Here, we can define a normal-based, 95% confidence interval for a population mean estimate:

Normal Distribution Confidence Interval

- $\bar{x} \pm 1.96 * \sqrt{\sigma^2/n}$

Our data does have a shape resembling a Poisson distribution cdf at $\lambda = 0.5$. Since many hospital-recorded attributes are discrete count data (e.g. visit count, number of medications, etc), it's common to see Poisson or binomial-based methods to estimate parameters. For our example here, we'll use the Poisson distribution where the parameter $\hat{\lambda} = \bar{x} = \sigma$. This is not the case in our data as our mean and variance are not equal for both OP groups. Although our data may resemble a Poisson distribution, it is clear that all of the assumptions are not met and would serve as an approximation, instead.

Poisson Distribution Confidence Interval

- Assumption: $\lambda = \mu = \sigma^2$
- $\hat{\lambda} \pm 1.96 * \sqrt{\hat{\lambda/n}}$

Finally, we will use a non-parametric boostrap-t approach to estimate a 95% confidence interval. Examining the histograms from above, it appears that our data would be an excellent fit for a trimmed mean. Especially since we are concerned with the most central values of ER visits as it relates to OP visits. We will be finding our interval around a 20% trimmed mean. CDC figures estimate that around 38.1 million adults had diabetes in 2021(*National Diabetes Statistics Report* 2024). From this, it can be determined that our population is sufficiently large as:

- $C = (N/n) = (38.1 million/131) > 20$

This also means that we will be resampling with replacement in our bootstrap-t process. We will be finding the lower and upper bounds for our bootstrap-t method as follows:

Bootstrap-t 95% Confidence Interval

- Lower Bound: $t - F^{-1}_{t(t*)}(0.975) * \hat{SE}(t)$
- Upper Boundc: $t - F^{-1}_{t(t*)}(0.025) * \hat{SE}(t)$

Where t is our trimmed sample mean, $F^{-1}_{t(t*)}$ is the inverse CDF, and $\hat{SE}(t)$ is the explicit standard error of our trimmed mean. We set $\alpha = 0.05$.

Here, we will resample with B=1000 replications and will be obtaining our confidence interval for a trimmed mean ($\gamma = 0.20$). In our approach, we want to capture the most central number of ED visits. Patients who are complex are likely to overutilize while many patients with no ER visits may be under engaged in the healthcare system anyways (e.g. younger, healthier patients). We are primarily concerned with the central values. Our trimmed mean will take the form:

- $\bar{x}_{i,\gamma} = (\frac{1}{n-2\gamma}) \sum_{i=g+1}^{(n-g)} x[i]$

Since we are choosing to trim the data, we will also employ winsorization to replace the values that we are removing on either end of our data set:

1. For either end of our ascending-sorted data, we remove $(n * \gamma)$ values.
2. For the lower end, we take the $x_{[n*\gamma+1]}$ value and replicate this $n * \gamma$ times.
3. For the upper end, we take the $x_{[n-(n*\gamma)]}$ value and replicate this $n * \gamma$ times.

We carry out these procedures for both the high and low OP groups. Due to the winsorization, the range of new data set will still contain n values, but will be much more compact in comparison as we have removed some of the extreme values. However, since our data contains high 0-counts, the left ends remain unchange – the zeroes are just replaced when we conduct our lower end winsorization. Additionally, since we are changing the data, we lose any advantages in population inference as we lose the randomness of our original sample.

IV. Results

The results from our comparison can be seen in the table below:

```
# Non-trimmed mean
tmuhat.high.nt = mean(high)
var.t.high.nt = var(high)
den.high.nt = length(high)
```

Method	Group	Sample Mean	95% CI	Interval Range
Normal	High OP Visits	1.930	[1.314, 2.546]	1.232
Poisson	High OP Visits	1.930	[1.566, 2.294]	0.728
Bootstrap-t	High OP Visits	1.930	[1.316, 2.509]	1.193
Bootstrap-t, 20% Trimmed Mean	High OP Visits	1.257	[0.718, 1.802]	1.084
Normal	Low OP Visits	1.162	[0.711, 1.613]	0.902
Poisson	Low OP Visits	1.162	[0.915, 1.409]	0.494
Bootstrap-t	Low OP Visits	1.162	[0.662, 1.568]	0.906
Bootstrap-t, 20% Trimmed Mean	Low OP Visits	0.587	[0.4, 0.814]	0.414

```
##### p-values #####
```

```r
# normal/t-tests
t.test(high, low, var.equal = FALSE)
```

```
##
##  Welch Two Sample t-test
##
## data:  high and low
## t = 1.9871, df = 108.4, p-value = 0.04943
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.001942487 1.533382312
## sample estimates:
## mean of x mean of y
##  1.929825  1.162162
```

```r
# poisson
poisson.test(
    c(sum(high > 0), sum(high == 0)),
    c(sum(low > 0), sum(low == 0)),
    alternative = c('two.sided')
)
```

```
##
##  Comparison of Poisson rates
##
## data:  c(sum(high > 0), sum(high == 0)) time base: c(sum(low > 0), sum(low == 0))
## count1 = 34, expected count1 = 27.73, p-value = 0.1117
## alternative hypothesis: true rate ratio is not equal to 1
## 95 percent confidence interval:
##  0.892731 2.774295
## sample estimates:
## rate ratio
##   1.560386
```

```r
# bootstrap t
t.stat = mean(high)-mean(low)
se.t = sqrt(
    (var(high)/length(high))+
    (var(low)/length(low))
)
to.t = (t.stat - 0)/se.t
c(t.stat, se.t, to.t)
```

```
## [1] 0.7676624 0.3863195 1.9871180
```

```r
bootstrap.t = function(x, y, B) {
    df.outputs = data.frame(matrix(ncol = 2, nrow = B))
    names(df.outputs) = c('t.stars', 'to.tstars')

    # bootstrap
    for(i in 1:B) {
        sample1 = sample(x, length(x), replace = TRUE)
        sample2 = sample(y, length(y), replace = TRUE)

        # find to(t*)
        t.star = mean(sample1) - mean(sample2)
        se.tstar = sqrt(
            (var(sample1)/length(sample1)) +
            (var(sample2)/length(sample2))
        )
        to.tstar = (t.star - t.stat)/se.tstar

        # store
        df.outputs$t.stars[i] = t.star
        df.outputs$to.tstars[i] = to.tstar
    }

    # /Shoiw Results
    return(df.outputs)
}

output = bootstrap.t(high, low, 2000)
sum(output$to.tstars > to.t)
```

```
## [1] 56
```

```r
out.tail = sum(output$to.tstars > to.t)
out.pval = out.tail/length(output$to.tstars)
c(out.tail, out.pval)
```

```
## [1] 56.000  0.028
```

```r
# library boot
library(boot)

s = mean(high) - mean(low)

scores = c(high, low)
group = c(
    rep('high',57),
```

```
    rep('low',74)
)

df = data.frame(group, scores)
df
```

```
##     group scores
## 1    high      0
## 2    high      0
## 3    high      0
## 4    high      0
## 5    high      0
## 6    high      0
## 7    high      0
## 8    high      0
## 9    high      0
## 10   high      0
## 11   high      0
## 12   high      0
## 13   high      0
## 14   high      0
## 15   high      0
## 16   high      0
## 17   high      0
## 18   high      0
## 19   high      0
## 20   high      0
## 21   high      0
## 22   high      0
## 23   high      0
## 24   high      1
## 25   high      1
## 26   high      1
## 27   high      1
## 28   high      1
## 29   high      1
## 30   high      1
## 31   high      1
## 32   high      1
## 33   high      1
## 34   high      2
## 35   high      2
## 36   high      2
```

```
## 37   high      2
## 38   high      2
## 39   high      2
## 40   high      2
## 41   high      2
## 42   high      3
## 43   high      3
## 44   high      4
## 45   high      4
## 46   high      4
## 47   high      4
## 48   high      5
## 49   high      5
## 50   high      5
## 51   high      5
## 52   high      6
## 53   high      6
## 54   high      7
## 55   high      7
## 56   high      8
## 57   high      8
## 58    low      0
## 59    low      0
## 60    low      0
## 61    low      0
## 62    low      0
## 63    low      0
## 64    low      0
## 65    low      0
## 66    low      0
## 67    low      0
## 68    low      0
## 69    low      0
## 70    low      0
## 71    low      0
## 72    low      0
## 73    low      0
## 74    low      0
## 75    low      0
## 76    low      0
## 77    low      0
## 78    low      0
## 79    low      0
```

```
## 80    low      0
## 81    low      0
## 82    low      0
## 83    low      0
## 84    low      0
## 85    low      0
## 86    low      0
## 87    low      0
## 88    low      0
## 89    low      0
## 90    low      0
## 91    low      0
## 92    low      0
## 93    low      0
## 94    low      0
## 95    low      0
## 96    low      1
## 97    low      1
## 98    low      1
## 99    low      1
## 100   low      1
## 101   low      1
## 102   low      1
## 103   low      1
## 104   low      1
## 105   low      1
## 106   low      1
## 107   low      1
## 108   low      1
## 109   low      1
## 110   low      1
## 111   low      1
## 112   low      1
## 113   low      2
## 114   low      2
## 115   low      2
## 116   low      2
## 117   low      2
## 118   low      2
## 119   low      2
## 120   low      2
## 121   low      2
## 122   low      2
```

```
## 123   low      3
## 124   low      3
## 125   low      4
## 126   low      4
## 127   low      4
## 128   low      5
## 129   low      7
## 130   low      8
## 131   low     11
```

```
colnames(df)
```

```
## [1] "group"  "scores"
```

```
mean.diff = function(x, i) {
    return(
        mean(x$scores[group == 'high'][i], na.rm = TRUE) - mean(x$scores[group == 'low'][i], na.rm = TRU
    )
}
just.mean = function(x, i) {
    return(mean(x[i]))
}

boot.high = boot(high, just.mean, R = 1000)
boot.ci(
    boot.high,
    c(0.95, .96, .97, .98, .99),
    'bca'
)
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = boot.high, conf = c(0.95, 0.96, 0.97, 0.98,
##     0.99), type = "bca")
##
## Intervals :
## Level       BCa
## 95%   ( 1.355,  2.579 )
## 96%   ( 1.351,  2.632 )
## 97%   ( 1.316,  2.667 )
## 98%   ( 1.281,  2.725 )
## 99%   ( 1.244,  2.915 )
## Calculations and Intervals on Original Scale
```

```
## Some BCa intervals may be unstable
```

```
boot.low = boot(low, just.mean, R=1000)
boot.ci(
    boot.low,
    c(0.95, .96, .97, .98, .99),
    'bca'
)
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = boot.low, conf = c(0.95, 0.96, 0.97, 0.98,
##     0.99), type = "bca")
##
## Intervals :
## Level       BCa
## 95%   ( 0.824,  1.716 )
## 96%   ( 0.797,  1.788 )
## 97%   ( 0.779,  1.900 )
## 98%   ( 0.743,  1.920 )
## 99%   ( 0.716,  2.012 )
## Calculations and Intervals on Original Scale
## Some BCa intervals may be unstable
```

```
boot.out = boot(df, mean.diff, R=1000)

boot.ci(
    boot.out,
    c(0.95, .96, .97, .98, .99),
    'bca'
)
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = boot.out, conf = c(0.95, 0.96, 0.97, 0.98,
##     0.99), type = "bca")
##
## Intervals :
## Level       BCa
## 95%   ( 0.0581,  1.4344 )
## 96%   ( 0.0452,  1.4578 )
```

```
## 97%   (-0.0626,  1.4926 )
## 98%   (-0.1242,  1.5421 )
## 99%   (-0.2333,  1.6620 )
## Calculations and Intervals on Original Scale
## Some BCa intervals may be unstable
```

For the normal-based 95% confidence interval:

- High OP group: [1.313, 2.546], range = 1.233
- Low OP group: [0.711, 1.613], range = 0.902

For the Poisson-based 95% confidence interval:

- High OP group: [1.566, 2.294], range = 0.728
- Low OP group: [0.915, 1.409], range = 0.494

For our bootstrap-t 95% confidence interval of the trimmed mean:

- High OP group: [0.718, 1.802], range = 1.084
- Low OP group: [0.4, 0.814], range = 0.414

For the high OP group, the normal-based method produces the widest interval (range = 1.233) while the Poisson-based interval produces the most narrow (0.728). Our boostrap-t interval for the trimmed mean is right in-between with a difference of 1.084.

For our low OP group, again, the normal-based interval have the widest difference in its interval bounds (range = 0.902). Our Poisson-based interval is between our normal and bootstrap-t results (range = 0.494). Our bootstrap-t interval had the most narrow range 0.414.

V. Discussion

Though we were expecting that the highly engaged group would have the same or lower number of ED visits, the results generated in our analysis highlights the differences between our approaches. The normal distribution and bootstrap-t based intervals for the two OP groups overlap, hinting that some differences are in effect, while the Poisson intervals are clearly separat that suggests a higher ER visit rate for the high OP group. However, each method's suitability needs further evaluation.

The normal distribution is not ideal for our skewed, discrete count data. Poisson, while attractive due to its suitability for count data, is also not a perfect fit as our sample does not meet all assumptions as our $\hat{\lambda} \neq \bar{x} \neq s^2$. At best Poisson would be an approximate fit that would heavily rely on larger and more reliable sampling.

Thus, bootstrapping emerges as the most practical and flexible method. It does not rely on assumptions like our normal or Poisson distribution approaches, making it especially useful for estimating parameters for new or complex populations – especially with today's available compute power. It should be noted that our estimation of a 20% trimmed mean would naturally lead to a narrower interval. However, we are interested in knowing the central values of ER visits. Zero counts are commmon as there are multiple reasons why a

patient may not decide to visit the ED. For extreme high counts, there are cases of patients with multiple complex conditions who are not indicative of the typical diabetes patient.

Resampling through bootstrapping provides a robust evaluation tool for public health interventionalists looking to free their analysis from parametric constraints. Other data may not benefit some looking as regular as ours and bootstrapping would be the ideal tool to deploy in those scenarios. In conclusion, Bootstrapping offers a robust tool for public health researchers looking to be free of parametric constraints and improve estimation accuracy, particularly when data distributions are irregular.

Appendix 1: R Code

```
##### Library #####
library(tidyverse)
library(ggplot2)
library(boot)
library(kableExtra)
library(hrbrthemes)
library(viridis)

##### Working Directory #####
setwd('/home/petepham/Documents/School/STAT 573/Final Project/data')

###### Functions #####
# icdf
icdf = function(x, alpha){
    # x = data or vector
    # alpha = percentile of interest

    # sort & define length
    list.data = sort(x)
    n = length(x)
    # Apply logic for alpha < 0.50 && alpha >= 50
    i = ifelse(
        alpha < 0.50,
        floor(alpha * (n+1)),
        (n+1) - floor((1-alpha)*(n+1))
    )
    # Return results
    return(list.data[i])
}

# Bootstrap-T Functions
tmean = function(x, gama = 0.2) {
    x = sort(x)
    n = length(x)
    a = mean(x, trim = gama)
    den = n * (1-2*gama)^2
    t = floor(n*gama)
    l = x[t+1]
    u = x[n-t]
    wv = c(rep(l, times=t),x[(t+1):(n-1)], rep(u, times=t))
    b = (a - tmuhat)/sqrt(var(wv)/den)
```

```
}

boot.t.mean = function(x, B) {
    n = length(x)
    boot.t = rep(0, B)

    for(i in 1:B) {
        xstar = sample(x, n, replace = TRUE)
        boot.t[i] = tmean(xstar)
    }

    return(boot.t)
}

##### Data Import & Transformation #####
data = read.csv('quality.csv')

df = data %>%
    select(
        MemberID,
        OfficeVisits,
        ERVisits,
        MedicalClaims,
        InpatientDays
    ) %>%
    mutate(
        pcp.group = ifelse(
            OfficeVisits >= 14,
            'high.engagement',
            'low.engagement'
        )
    )

df.group = df %>%
    group_by(pcp.group) %>%
    summarise(
        patient_count = n(),
        total_er = sum(ERVisits),
        mean_er = mean(ERVisits),
        var_er = var(ERVisits)
    ) %>%
    mutate(
        pcp.group = recode(
```

```
            pcp.group,
            "high.engagement" = "High OP Engagement",
            "low.engagement" = "Low OP Engagmeent"
        ),
        total_er = round(total_er, 3),
        mean_er = round(mean_er, 3),
        var_er = round(var_er, 3)
    )

high = sort(df$ERVisits[df$pcp.group == 'high.engagement'])
low = sort(df$ERVisits[df$pcp.group == 'low.engagement'])

colnames(df.group) = c(
    'Engagement Group', '# of Patients', '# ER Visits', 'Mean ER Visits', 'Variance ER Visits'
    )

kable(df.group, align = "lcccc")

p1 = hist(
    low,
    main = "ER Visits for Patients with Low Count of Outpatient Visits",
    xlab = "Number of ER Visits"
    )
p2 = hist(
    high,
    main = "ER Visits for Patients with High Count of Outpatient Visits",
    xlab = "Number of ER Visits")


# Non-trimmed mean
tmuhat.high.nt = mean(high)
var.t.high.nt = var(high)
den.high.nt = length(high)

# reframe
n.high = length(high)
mean.high = mean(high)
var.high = var(high)

n.low = length(low)
mean.low = mean(low)
var.low = var(low)
```

```
###### Normal Results #####
norm.low.lower = mean.low - (1.96*(sqrt(var.low/(n.low-1))))
norm.low.upper = mean.low + (1.96*(sqrt(var.low/(n.low-1))))

norm.high.lower = mean.high - (1.96*(sqrt(var.high/(n.high-1))))
norm.high.upper = mean.high + (1.96*(sqrt(var.high/(n.high-1))))

##### Poisson Results #####
pois.high.lower = mean.high - (1.96*(sqrt(mean.high/(n.high-1))))
pois.high.upper = mean.high + (1.96*(sqrt(mean.high/(n.high-1))))

pois.low.lower = mean.low - (1.96*(sqrt(mean.low/(n.low-1))))
pois.low.upper = mean.low + (1.96*(sqrt(mean.low/(n.low-1))))


##### BootStrap Results Untrimmed #####
# high
t.star.high = rep(NA, 2000)
for(i in 1:length(t.star.high)) {
    sample = sample(high, length(high), replace = TRUE)

    t.star.high[i] = mean(sample)
}

bs.mean.high = mean(t.star.high)
bs.sdev.high = sqrt(var(t.star.high))

bs.high.lower = floor((length(t.star.high)+1)*(0.025))
bs.high.upper = (length(t.star.high)+1)-floor((length(t.star.high)+1)*0.025)

t.star.high.sorted = sort(t.star.high)
t.star.high.sorted[bs.high.lower]
t.star.high.sorted[bs.high.upper]

bs.high.ci.lower = 2*mean(high) - t.star.high.sorted[bs.high.upper]
bs.high.ci.upper = 2*mean(high) - t.star.high.sorted[bs.high.lower]

# low
t.star.low = rep(NA, 2000)
for(i in 1:length(t.star.low)) {
    sample = sample(low, length(low), replace = TRUE)

    t.star.low[i] = mean(sample)
```

```
}

bs.mean.low = mean(t.star.low)
bs.sdev.low = sqrt(var(t.star.low))

bs.low.lower = floor((length(t.star.low)+1)*(0.025))
bs.low.upper = (length(t.star.low)+1)-floor((length(t.star.low)+1)*0.025)

t.star.low.sorted = sort(t.star.low)
t.star.low.sorted[bs.low.lower]
t.star.low.sorted[bs.low.upper]

bs.low.ci.lower = 2*mean(low) - t.star.low.sorted[bs.low.upper]
bs.low.ci.upper = 2*mean(low) - t.star.low.sorted[bs.low.lower]

##### BootStrap Results 20% Trimmed #####
# high
tmuhat.high = mean(high, trim = 0.2)
floor.high = floor(length(high) * 0.2)
var.t.high = var(
    c(
        rep(high[floor.high+1],floor.high),
        high[(floor.high+1):(length(high)-floor.high)],
        rep(high[length(high)-floor.high],floor.high)
    )
)
den.high = length(high)*(1-2*0.2)^2
sqrt(var.t.high/den.high)

tmuhat = tmuhat.high
out.high = boot.t.mean(high, 1000)

bt.high.upper = tmuhat.high - icdf(out.high, 0.025)*sqrt(var.t.high/den.high) # upper
bt.high.lower = tmuhat.high - icdf(out.high, 0.975)*sqrt(var.t.high/den.high) # lower

# low
tmuhat.low = mean(low, trim = 0.2)
floor.low = floor(length(low) * 0.2)
floor.low
var.t.low = var(
    c(
        rep(low[floor.low+1],floor.low),
        low[(floor.low+1):(length(low)-floor.low)],
```

```
        rep(low[length(low)-floor.low],floor.low)
    )
)
den.low = length(high)*(1-2*0.2)^2
sqrt(var.t.low/den.low)

tmuhat = tmuhat.low
out.low = boot.t.mean(low, 1000)

bt.low.upper = tmuhat.low - icdf(out.low, 0.025)*sqrt(var.t.low/den.low) # upper
bt.low.lower = tmuhat.low - icdf(out.low, 0.975)*sqrt(var.t.low/den.low) # lower


##### Tabling #####
method = c("Normal","Normal","Poisson","Poisson","Bootstrap-t", "Bootstrap-t","Bootstrap-t, 20% Trimmed
group = rep(c("High OP Visits", "Low OP Visits"), 4)
mean = round(c(
    mean.high, mean.low,
    mean.high, mean.low,
    mean.high, mean.low,
    tmuhat.high, tmuhat.low
    ),3)
lower = round(c(
    norm.high.lower, norm.low.lower,
    pois.high.lower, pois.low.lower,
    bs.high.ci.lower, bs.low.ci.lower,
    bt.high.lower, bt.low.lower
    ),3)
upper = round(c(
    norm.high.upper, norm.low.upper,
    pois.high.upper, pois.low.upper,
    bs.high.ci.upper, bs.low.ci.upper,
    bt.high.upper, bt.low.upper
    ),3)

ci = paste0(
    "[",
    lower,
    ", ",
    upper,
    "]"
)
df.results = data.frame(method, group, mean, ci, lower, upper) %>%
```

```
    mutate(
        mean = round(mean, 3),
        lower = round(lower, 3),
        upper = round(upper, 3),
        range = upper - lower
    ) %>%
    select(
        method,
        group,
        mean,
        ci,
        range
    )

colnames(df.results) = c("Method", "Group", "Sample Mean", "95% CI","Interval Range")

df.results2 = df.results %>%
    arrange(group)

kable(df.results2, align ="lccccc") %>%
    row_spec(4, hline_after = TRUE)

##### p-values #####

# normal/t-tests
t.test(high, low, var.equal = FALSE)

# poisson
poisson.test(
    c(sum(high > 0), sum(high == 0)),
    c(sum(low > 0), sum(low == 0)),
    alternative = c('two.sided')
)

# bootstrap t
t.stat = mean(high)-mean(low)
se.t = sqrt(
    (var(high)/length(high))+
    (var(low)/length(low))
)
to.t = (t.stat - 0)/se.t
c(t.stat, se.t, to.t)
```

```r
bootstrap.t = function(x, y, B) {
    df.outputs = data.frame(matrix(ncol = 2, nrow = B))
    names(df.outputs) = c('t.stars', 'to.tstars')

    # bootstrap
    for(i in 1:B) {
        sample1 = sample(x, length(x), replace = TRUE)
        sample2 = sample(y, length(y), replace = TRUE)

        # find to(t*)
        t.star = mean(sample1) - mean(sample2)
        se.tstar = sqrt(
            (var(sample1)/length(sample1)) +
            (var(sample2)/length(sample2))
        )
        to.tstar = (t.star - t.stat)/se.tstar

        # store
        df.outputs$t.stars[i] = t.star
        df.outputs$to.tstars[i] = to.tstar
    }

    # /Shoiw Results
    return(df.outputs)
}

output = bootstrap.t(high, low, 2000)
sum(output$to.tstars > to.t)
out.tail = sum(output$to.tstars > to.t)
out.pval = out.tail/length(output$to.tstars)
c(out.tail, out.pval)

# library boot
library(boot)

s = mean(high) - mean(low)

scores = c(high, low)
group = c(
    rep('high',57),
    rep('low',74)
)
```

```r
df = data.frame(group, scores)
df
colnames(df)

mean.diff = function(x, i) {
    return(
        mean(x$scores[group == 'high'][i], na.rm = TRUE) - mean(x$scores[group == 'low'][i], na.rm = TRU
    )
}
just.mean = function(x, i) {
    return(mean(x[i]))
}

boot.high = boot(high, just.mean, R = 1000)
boot.ci(
    boot.high,
    c(0.95, .96, .97, .98, .99),
    'bca'
)
boot.low = boot(low, just.mean, R=1000)
boot.ci(
    boot.low,
    c(0.95, .96, .97, .98, .99),
    'bca'
)
boot.out = boot(df, mean.diff, R=1000)

boot.ci(
    boot.out,
    c(0.95, .96, .97, .98, .99),
    'bca'
)
```

Appendix 2: References

Bannuru, Emily D. Parker; Janice Lin; Troy Mahoney; Nwanneamaka Ume; Grace Yang; Robert A. Gabbay; Nuha A. ElSayed; Raveendhara R. 2023. *Economic Costs of Diabetes in the u.s. In 2022.* American Diabetes Association. https://diabetesjournals.org/care/article/47/1/26/153797/Economic-Costs-of-Diabetes-in-the-U-S-in-2022.

Donald K Cherry, Elizabeth A Rechtsteiner, David A Woodwell. 2007. *Optimization of Multigrain Premix for High Protein and Dietary Fibre Biscuits Using Response Surface Methodology (RSM).* National Institute of Health. https://pubmed.ncbi.nlm.nih.gov/17703793/.

IDF Diabetes Atlas, 10th Edition. 2021. International Diabetes Federation. https://diabetesatlas.org/idfawp/resource-files/2021/07/IDF_Atlas_10th_Edition_2021.pdf.

Jingyao Hong, Natalie Daya, Aditya Surapaneni. 2021. *Retinopathy and Risk of Kidney Disease in Persons with Diabetes.* National Institute of Health. https://pmc.ncbi.nlm.nih.gov/articles/PMC8515075/.

National Diabetes Statistics Report. 2024. Centers for Disease Control. https://www.cdc.gov/diabetes/php/data-research/?CDC_AAref_Val=https://www.cdc.gov/diabetes/data/statistics-report/index.html.

Racial and Ethnic Disparities in Diabetes Prevalence, Self-Management, and Health Outcomes Among Medicare Beneficiaries. 2017. Centers for Medicare & Medicaid Services. https://www.cms.gov/About-CMS/Agency-Information/OMH/Downloads/March-2017-Data-Highlight.pdf#:~:text=Diabetes%20prevalence%20was%20higher%20among%20Black%20(30.0,beneficiaries%20who%20were%20male%20(22.3%20percent%20vs.

Stephanie M Gruss, Edward Gregg, Kunthea Nhim. 2019. *Public Health Approaches to Type 2 Diabetes Prevention: The US National Diabetes Prevention Program and Beyond.* National Institute of Health. https://pmc.ncbi.nlm.nih.gov/articles/PMC6682852/.