1.Statistical Inference Data

J.Hastings

2024-07-03

```
knitr::opts_chunk$set(echo = TRUE)
```

The "Results Score" column represents my observed data for (Bernoulli distribution) successfully completed the program through it's duration=1 vs. Unsuccessfully completing the program prior to expiration=0

The Goal of this project is to try to perdict which participats will successfully complete the program based on their entrance assesment scores.

```
results<-c(1,1,0,1,0,0,1,0,0,0,1,0,1,1,1,1,1,0,1,0,1,1)

### Calculate the number of trials
n <- length(results)


### Calculate the probability of success from observed data
p_hat <- sum(results) / n


cat("Sample Size:",n, "\n")
```

First we will Uncover the'Result_score' column from the observed data to get an understanding if this program actually benefits the clients.

```
## Sample Size: 22
```

```
cat("Result Score Column, 1=good, 0=bad:", results, "\n")
```

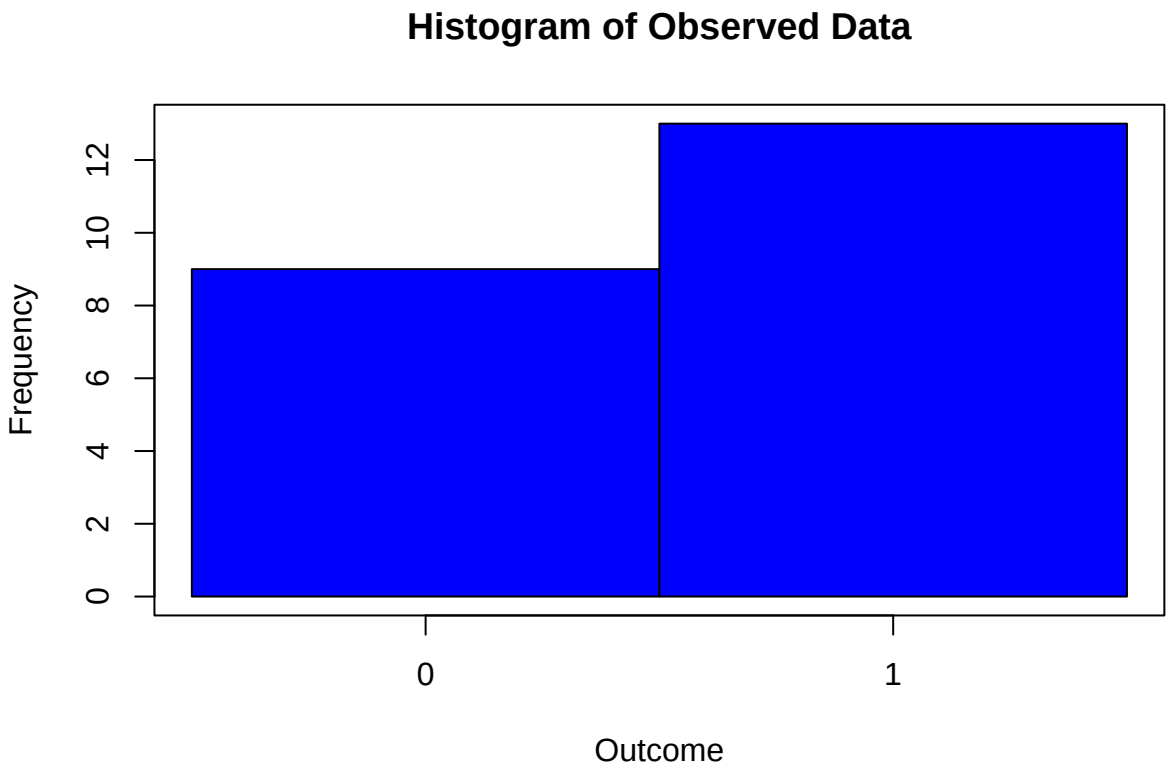
```
## Result Score Column, 1=good, 0=bad: 1 1 0 1 0 0 1 0 0 0 1 0 1 1 1 1 1 0 1 0 1 1
```

```
cat("**Probability of Success from observed Data:**",p_hat, "\n")
```

```
## **Probability of Success from observed Data:** 0.5909091
```

Create a histogram of the observed data

```r
hist(results, breaks=seq(-0.5, 1.5, by=1), col="blue", xlab="Outcome",
     ylab="Frequency", main="Histogram of Observed Data", xaxt='n', axes=FALSE)
axis(side=1, at=c(0, 1))  # Set x-axis labels to 0 and 1
axis(side=2)  # Add default y-axis
box()  # Add a box around the plot
```



Creat a theoretical Bernoulli distribution to simulate population parameters to compare our observations against.

Set the seed for reproducibility.

```r
set.seed(123)
```

Parameters for a single Bernoulli trial.

```r
size <- 1      # Each trial is a Bernoulli trial (size=1)
prob <- 0.5    # Probability of success in each trial
n.1 <- 10000          # Number of trials

### Generate  theoretical varience population data
pop_data <- rbinom(n.1, size, prob)
```

```r
pop_variance<- var(pop_data)
pop_variance
```

```
## [1] 0.2499925
```

Probability for the outcomes of a single Bernoulli trial based on 1000 trials

Calculate probabilities for 0 and 1 outcomes

Print the probabilities

Binomial test for success probability

Extract the confidence intervals

```r
size <- 1      # Each trial is a Bernoulli trial (size=1)
prob <- 0.5    # Probability of success in each trial
n.1 <- 10000   # Number of trials


y_0 <- dbinom(0, size, prob)  # Probability of 0 successes (failure)
y_1 <- dbinom(1, size, prob)  # Probability of 1 success (success)


cat("Probability of failure (0 successes):", y_0, "\n")
```

```
## Probability of failure (0 successes): 0.5
```

```r
cat("Probability of success (1 success):", y_1, "\n")
```

```
## Probability of success (1 success): 0.5
```

```r
test <- binom.test(n.1 * prob, n.1, prob)

low_interval <- round(test$conf.int[1], 3)
high_interval <- round(test$conf.int[2], 3)


cat("Confidence interval for the probability of success:\n")
```

```
## Confidence interval for the probability of success:
```

```r
cat("Low interval:", low_interval, "\n")
```

```
## Low interval: 0.49
```

```r
cat("High interval:", high_interval, "\n")
```

```
## High interval: 0.51
```

Print probabilities of 1 trial.

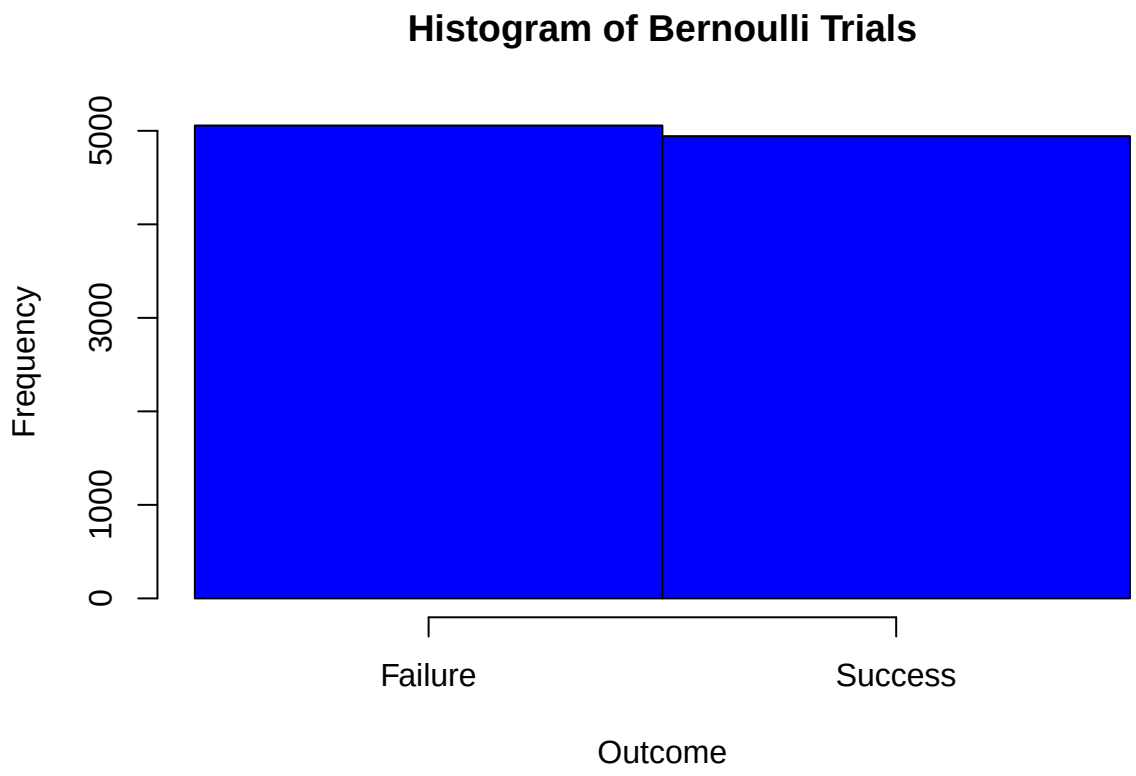
```r
print(paste("Probability of failure (0 successes):", y_0))
```

```
## [1] "Probability of failure (0 successes): 0.5"
```

```r
print(paste("Probability of success (1 success):", y_1))
```

```
## [1] "Probability of success (1 success): 0.5"
```

```r
### Display the histogram of the Bernoulli trials
hist(pop_data, breaks=seq(-0.5, 1.5, 1), main="Histogram of Bernoulli Trials", col="blue", xlab="Outcom
axis(1, at=c(0,1), labels=c("Failure", "Success"))
```



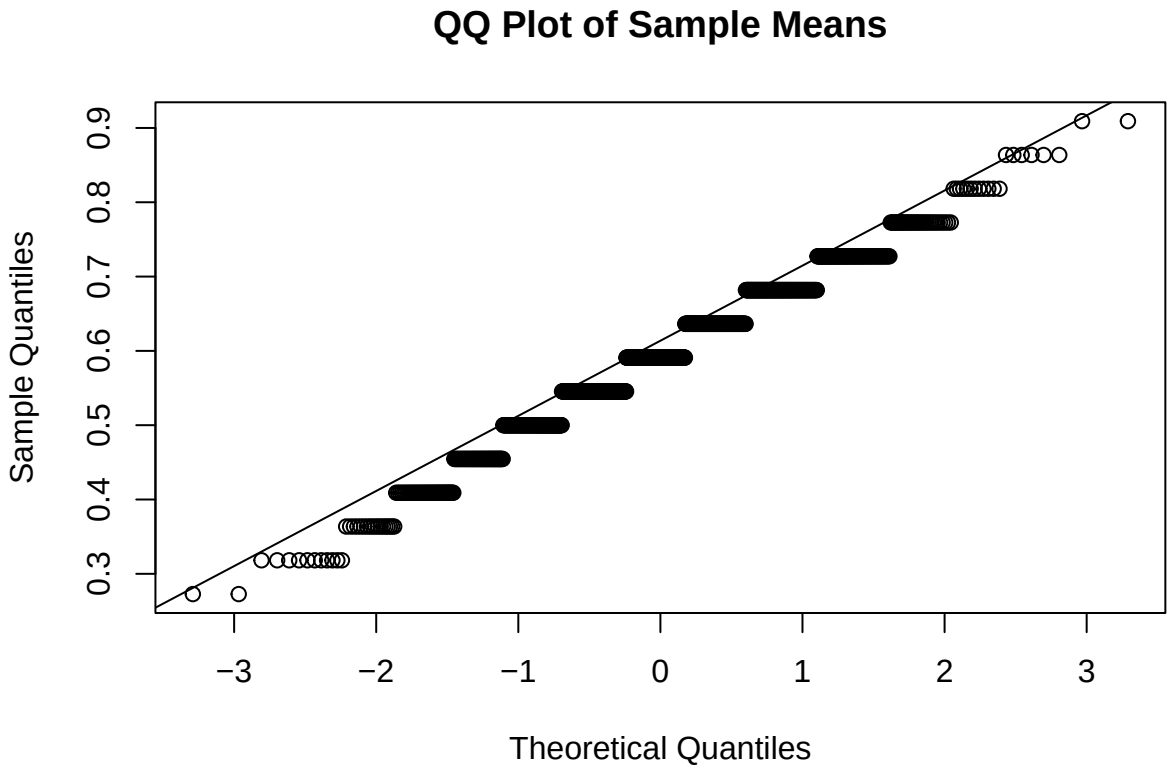
Checking for Normality to make sure we are working with a Bernoulli distribution. Normality also makes confidence intervals and hypothesis testing more percise.

```r
### Number of resamplesand observations in each sample
num_resamples <- 1000
n.2 <- length(results)

### Storage for sample means
sample_means <- numeric(num_resamples)

### Generating sample means for each individuale trial
set.seed(123)  # For reproducibility
for (i in 1:num_resamples) {
  sample_means[i] <- mean(sample(results, size = n.2, replace = TRUE))
}

### Generate QQ plot to get eye ball view of proper normality
qqnorm(sample_means, main = "QQ Plot of Sample Means")
qqline(sample_means)
```



Test weather the p_hat statistic is close to the true popluation mean.

```r
mu=0.50
omega_2<- mu*(1-mu)
```

```r
omega<-sqrt(omega_2)
omega_2<-round(omega_2,4)
omega<-round(omega,3)

### Calculate sample mean,varience and sd
p_hat <- sum(results) / n
sample_var=var(results)
samp_sd=sqrt(sample_var)

p_hat<-round(p_hat,3)
sample_var<-round(sample_var,3)
samp_sd=round(samp_sd,3)

### Print population parameters
cat("Population Parameters\n")
```

```
## Population Parameters
```

```r
cat("Population mean (mu):", mu, "\n")
```

```
## Population mean (mu): 0.5
```

```r
cat("Population variance (omega^2):", omega_2, "\n")
```

```
## Population variance (omega^2): 0.25
```

```r
cat("Population standard deviation (omega):", omega, "\n")
```

```
## Population standard deviation (omega): 0.5
```

```r
### Print sample statistics
cat("\nSample Statistics\n")
```

```
##
## Sample Statistics
```

```r
cat("Sample mean (p_hat):", p_hat, "\n")
```

```
## Sample mean (p_hat): 0.591
```

```r
cat("Sample variance:", sample_var, "\n")
```

```
## Sample variance: 0.253
```

```r
cat("Sample standard deviation:", samp_sd, "\n")
```

```
## Sample standard deviation: 0.503
```

```
### Since we have a small sample size we use a T_test to get a respectiable estimate for mean.
### we are using a 2-sampled test to get the confidence intervals for mean with a significance level of

df=n-1
### 95% confidence interval
alpha=0.05

cv=qt(alpha/2,df,lower.tail=FALSE)
se=samp_sd/sqrt(n)
moe=cv*se


low_interval=p_hat-moe
high_interval=p_hat+moe
low_interval=round(low_interval,4)
high_interval=round(high_interval,4)

### Print CI's
cat("Low CI :", low_interval, "\n")
```

```
## Low CI : 0.368
```

```
cat("High CI :", high_interval, "\n")
```

```
## High CI : 0.814
```

We are 95% sure that from several simulations the population mean will be between [0.368,0.814]

Calculate Hypothesis testing to check if our sample estimate of mean is better then out theoretical population mean.

Reject H0 in favor of H1 if p_hat > - c or p_hat < + c for two tailed test.C is our rejection region.

HO:mu=0.50; H1 does NOT=0.50

```
mu=0.5

### calculate for some c"(cut off):
c.up=mu+cv*(samp_sd/sqrt(n))
c.down=mu-cv*(samp_sd/sqrt(n))

### Round to 4 places
c.up=round(c.up,4)
c.down=round(c.down,4)

### Print C-levels
cat("Low-C :", c.down, "\n")
```

```
## Low-C : 0.277
```

```r
cat("High-C :", c.up, "\n")
```

```
## High-C : 0.723
```

Test Results: Fail to REJECT the Null hypothesis since p_hat(sample_mean) is between [0.277,0.723]

Calculate p-value at 0.05 significance level. H0:mu = 0.50 vs H1:mu > 0.50

```r
mu<-0.50
p_hat<-0.591
sd_sample<-0.53
n<-22
dsf=n-1

se=sd_sample/sqrt(n)

t=(p_hat-mu)/se

p_value=pt(t,df,lower.tail=FALSE)
p_value=round(p_value,3)

### Print C-levels
cat("P-Value :", p_value, "\n")
```

```
## P-Value : 0.215
```

Since p_value > alpha, we fail to reject the null(mu)!

Create a CI using a chi-square dist to find a varience statistic with 0.05 significance level.

```r
n=22
alpha<-0.05
df<-n-1
samp_var<- 0.253

low=qchisq(alpha/2,df,lower.tail=FALSE)
up=qchisq(1-alpha/2,df,lower.tail = FALSE)
chi_low=(df*sample_var)/low
chi_up=(df*sample_var)/up
chi_low=round(chi_low,4)
chi_up=round(chi_up,4)
### Print CI chi intervals
cat("lower Confidence Interval :", chi_low, "\n")
```

```
## lower Confidence Interval : 0.1498
```

```r
cat("Upper Confidence Interval :", chi_up, "\n")
```

```
## Upper Confidence Interval : 0.5167
```

We are 95% confident that from several simulations the true variance will be between [0.148, 0.5167].

Since we have a big spread we can use p-value to get us close to the correct estimate.

Since the sample varienace and population varience are pretty close we are going to create a 2-tailed hypothesis test:

H0=0.25 vs H1:NOT=0.25

```r
### Sample variance
samp_var<- 0.253

### Population variance
omega_2<- 0.25

n=22
alpha<-0.05
df<-n-1

# Test statistic
chi_sq <-(df * sample_var) /omega_2


# P-value
p_value <- 2 * pchisq(chi_sq, df,lower.tail=FALSE)

# Output the p-value:
p_value=round(p_value,3)
cat("P-Value :", p_value, "\n")
```

```
## P-Value : 0.887
```

Since the p-value $>$ 0.05(alpha), we fail to reject the null hypothesis. This means there is no significant difference between the observed(sample) variance and the true population variance.The results make sense since 0.25 is close to 0.253.

```r
mu<-0.50
omega_2<- 0.25

cat("True Mean :", mu, "\n")
```

```
## True Mean : 0.5
```

```r
cat("True Variance :", omega_2, "\n")
```

```
## True Variance : 0.25
```

As we gather more observations, we can expect our Bernoulli distribution to be normal with a mean and variance to be [0.50,0.25] respectively.

```r
sessionInfo()
```

```
## R version 4.2.1 (2022-06-23)
## Platform: aarch64-apple-darwin20 (64-bit)
## Running under: macOS 14.5
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.2-arm64/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.2-arm64/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## loaded via a namespace (and not attached):
##  [1] compiler_4.2.1  fastmap_1.1.1   cli_3.6.1       tools_4.2.1
##  [5] htmltools_0.5.5 rstudioapi_0.14 yaml_2.3.7      rmarkdown_2.22
##  [9] highr_0.10      knitr_1.43      xfun_0.39       digest_0.6.31
## [13] rlang_1.1.1     evaluate_0.21
```