A7.1 Hypothesis Testing on Different Types of Data

Jeffrey An

December 2, 2016

Introduction

In this assignment, we will calculate sample sizes, explore several data sets, create confidence intervals, and test samples. The goal of this assignment is to compare samples with statistical significance.

Taste Testing Data Sample Size Determination

In this section, I will calculate two sample sizes. For the first calculation, it was assumed that the population mean is 85% and for the second calculation, the population mean is unknown. The provided stats are as follows:

```
1. 95% confidence interval
2. Overall width, margin of error, just less than 2%
```

Since the confidence interval is two tailed, I divided the alpha by two and subtracted it from 1 to get .975. Next, I applied qnorm to calculate the z score. Next, the population mean and the error was stored into variables. Using the sample size calculation formula, the sample size assuming a 85% population was calculated to be 4898. The interpretation is that we need to choose a sample size of at least 4898 people to create a confidence interval of 95% with an overall width of less than 2%.

```
za2_pref<- qnorm(.975) #calculating z a/2
    p_pref<-.85 #Assuming the preference rate at 85%
    e_pref<-.01 #Error term - half of the width

nsize_pref<- za2_pref^2*p_pref*(1-p_pref)/e_pref^2 #calculating the sample size
round(nsize_pref) #rounding up since we're calculating for people
```

```
## [1] 4898
```

Next, another sample size was calculated with the same values except the population mean was assumed to be unknown. Since we're assuming that the population mean is unknown, I'll use .5 as the preference rate to be conservative. The sample size is calculated to be 9604.

```
za2_unk<- qnorm(.975)
    p_unk<-.5 #Assuming the preference rate is unknown
    e_unk<-.01 #Error term - half of the width

nsize_unk<-za2_unk^2*p_unk*(1-p_unk)/e_unk^2
round(nsize_unk)
```

```
## [1] 9604
```

The sample size obtained from using an assumed population mean is much smaller than the one calculated without any assumptions because there is risk that the resulting confidence interval may be wider than desired when assuming the population mean. Moreover, if you knew the population mean, you would have more information on what the sample will look like and a smaller sample size will be able to product the same quality of predictability.

Confidence Intervals for Hot Dog Data

In this section, a dataset on three types of hot dogs will be analyzed. Our goal is to:

```
1. Read in the data set to a data frame
2. Note the summary statistics for the overall data set
3. Create individual data frames for each hot dog type (beef, meat & poultry)
4. Look at the summary statistics for each subsetted data frame
5. Create boxplots for each hot dog type
6. Create confidence intervals for mean # of calories for each hot dog type at 95% & 99% confidence leve
7. Run t tests to see if the means are different
8. Determine which type of hot dog has an average Sodium level different from 425 milligrams.
```

The data was read into a variable and the summary statistics was explored. There were 3 columns, one of which was used to identify the type of hot dog. Using the deplyr, the types of hot dogs were isolated by filtering then the type column was deleted and the remaining data was stored into a new dataset. After creating subsets by type of hot dog, box plots were created in one row for comparison.

```
hotdogs<-read.csv("hot_dogs.csv") #reading in the dataset
summary(hotdogs) #summarizing the data set
```

```
##       Type       Calories         Sodium
##  Beef   :20   Min.   : 86.0   Min.   :144.0
##  Meat   :17   1st Qu.:132.0   1st Qu.:362.5
##  Poultry:17   Median :145.0   Median :405.0
##               Mean   :145.4   Mean   :424.8
##               3rd Qu.:172.8   3rd Qu.:503.5
##               Max.   :195.0   Max.   :645.0
```

```
library(dplyr) #using dplyr
```

```
## Warning: package 'dplyr' was built under R version 3.3.2
```

```
##
## 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
```

```
beef<-filter(hotdogs, Type == "Beef") #subsetting beef hotdogs
    beef<-beef[2:3] #removing the type column
    summary(beef) #checking summary statistics for new data frame
```

```
##     Calories         Sodium
##  Min.   :111.0   Min.   :253.0
```

```
##  1st Qu.:140.5   1st Qu.:321.2
##  Median :152.5   Median :380.5
##  Mean   :156.8   Mean   :401.1
##  3rd Qu.:177.2   3rd Qu.:477.5
##  Max.   :190.0   Max.   :645.0
```

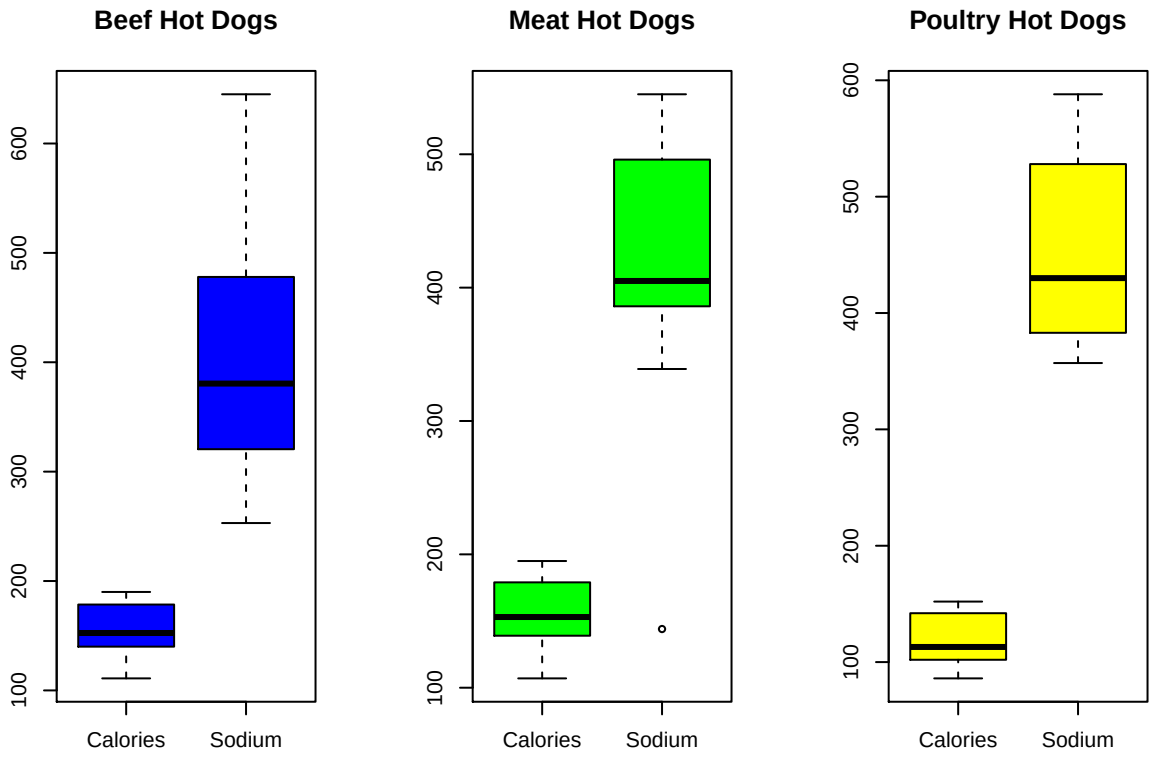
```
meat<-filter(hotdogs, Type == "Meat") #subsetting meat hotdogs
    meat<-meat[2:3] #removing the type column
    summary(meat) #checking summary statistics for new data frame
```

```
##     Calories         Sodium
##  Min.   :107.0   Min.   :144.0
##  1st Qu.:139.0   1st Qu.:386.0
##  Median :153.0   Median :405.0
##  Mean   :158.7   Mean   :418.5
##  3rd Qu.:179.0   3rd Qu.:496.0
##  Max.   :195.0   Max.   :545.0
```

```
poultry<-filter(hotdogs, Type == "Poultry") #subsetting poultry hotdogs
    poultry<-poultry[2:3] #removing the type column
    summary(poultry) #checking summary statistics for new data frame
```

```
##     Calories         Sodium
##  Min.   : 86.0   Min.   :357
##  1st Qu.:102.0   1st Qu.:383
##  Median :113.0   Median :430
##  Mean   :118.8   Mean   :459
##  3rd Qu.:142.0   3rd Qu.:528
##  Max.   :152.0   Max.   :588
```

```
par(mfrow=c(1,3)) #setting the format of boxplots
    boxplot(beef, main ="Beef Hot Dogs", col = "blue") #creating boxplots
    boxplot(meat, main ="Meat Hot Dogs", col = "green")
    boxplot(poultry, main ="Poultry Hot Dogs", col = "yellow")
```



6. Create confidence intervals for mean # of calories for each hot dog type at 95% & 99% confidence levels

Next, confidence levels were calculated for each hot dog type's calories. To do so, the number of observations was stored by calculating the number of rows. then the mean and standard deviation was calculated into new variables. Margin of error was calculated by finding the z score. For 95% level, a two tail test was used and for the 99% confidence level the higher point was calculated. To do this, the two tail test was calculated by dividing the alpha by 2 and the one tail test was calculated by keeping the alpha value whole. Finally, the confidence interval was calculated by subtracting and adding from the mean.

```
#Beef Hot Dogs Confidence Interval at 95% level

n_bha <- nrow(beef) #number of sample observations
    meancal_bha <- mean(beef$Calories) #storing the mean
    sdcal_bha <- sd(beef$Calories) #storing the standard deviation
    ecal_bha <- qnorm(1-(0.05/2))* sdcal_bha/sqrt(n_bha) #calculating two tail z values
    conf.int_bha <- c(meancal_bha - ecal_bha, meancal_bha + ecal_bha) #storing confidence intervals
    conf.int_bha #printing confidence intervals
```

```
## [1] 146.9269 166.7731
```

```
#Beef Hot Dogs Confidence Interval at 99% level
n_bhb <- nrow(beef) #number of sample observations
    meancal_bhb <- mean(beef$Calories) #storing the mean
    sdcal_bhb <- sd(beef$Calories) #storing the standard deviation
    ecal_bhb <- qnorm(.99, lower.tail = FALSE)* sdcal_bhb/sqrt(n_bhb) #calculating one tail z values
```

```
    conf.int_bhb <- meancal_bhb - ecal_bhb #storing confidence interval at higher point
    conf.int_bhb #printing confidence interval
```

```
## [1] 168.6281
```

```
#Meat Hot Dogs Confidence Interval at 95% level

n_mha <- nrow(meat)
    meancal_mha <- mean(meat$Calories)
    sdcal_mha <- sd(meat$Calories)
    ecal_mha <- qnorm(1-(0.05/2))* sdcal_mha/sqrt(n_mha)
    conf.int_mha <- c(meancal_mha - ecal_mha, meancal_mha + ecal_mha)
    conf.int_mha
```

```
## [1] 146.7098 170.7020
```

```
#Meat Hot Dogs Confidence Interval at 99% level
n_mhb <- nrow(meat)
    meancal_mhb <- mean(meat$Calories)
    sdcal_mhb <- sd(meat$Calories)
    ecal_mhb <- qnorm(.99, lower.tail = FALSE)* sdcal_mhb/sqrt(n_mhb)
    conf.int_mhb <- meancal_mhb - ecal_mhb
    conf.int_mhb
```

```
## [1] 172.9445
```

```
#Poultry Hot Dogs Confidence Interval at 95% level
n_pha <- nrow(poultry)
    meancal_pha <- mean(poultry$Calories)
    sdcal_pha <- sd(poultry$Calories)
    ecal_pha <- qnorm(1-(0.05/2))* sdcal_pha/sqrt(n_pha)
    conf.int_pha <- c(meancal_pha - ecal_pha, meancal_pha + ecal_pha)
    conf.int_pha
```

```
## [1] 108.0446 129.4848
```

```
#Poultry Hot Dogs Confidence Interval at 99% level
n_phb <- nrow(poultry)
    meancal_phb <- mean(poultry$Calories)
    sdcal_phb <- sd(poultry$Calories)
    ecal_phb <- qnorm(.99, lower.tail = FALSE)* sdcal_phb/sqrt(n_phb)
    conf.int_phb <- meancal_phb - ecal_phb
    conf.int_phb
```

```
## [1] 131.4887
```

7. Run T tests to see if the means are different

Next t test were ran to see if there was a difference in the mean number of calories for each hot dog type. The results are below:

1. Each result was ran at a 95% confidence level alpha = .05
2. Null hypothesis = The difference in means is 0
3. Alternative hypothesis = The difference in means is not 0
4. Beef vs Meat hot dog - 0.8167 > .05 Null hypothesis can't be rejected, means are likely the same
5. Beef vs Meat hot dog - 0.00001229 < 0.5 Null hypothesis is rejected, means are likely to be differen
6. Meat vs Poultry hot dog - 0.00003017 < 0.5 Null hypothesis is rejected, means are likely to be diffe

```
t.test(x = beef$Calories, y = meat$Calories) #testing beef and meat hot dogs
```

```
##
##  Welch Two Sample t-test
##
## data:  beef$Calories and meat$Calories
## t = -0.23364, df = 32.553, p-value = 0.8167
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -18.02490  14.31314
## sample estimates:
## mean of x mean of y
##  156.8500  158.7059
```

```
t.test(x = beef$Calories, y = poultry$Calories) #testing beef and poultry hot dogs
```

```
##
##  Welch Two Sample t-test
##
## data:  beef$Calories and poultry$Calories
## t = 5.11, df = 34.09, p-value = 1.229e-05
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  22.94024 53.23035
## sample estimates:
## mean of x mean of y
##  156.8500  118.7647
```

```
t.test(x = meat$Calories, y = poultry$Calories) #testing meat and poultry
```

```
##
##  Welch Two Sample t-test
##
## data:  meat$Calories and poultry$Calories
## t = 4.8659, df = 31.604, p-value = 3.017e-05
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  23.21307 56.66929
## sample estimates:
## mean of x mean of y
##  158.7059  118.7647
```

8. Determine which type of hot dog has an average Sodium level different from 425 milligrams.

T tests of each hot dog's sodium level was analyzed to see if they were different from a level of 425 milligrams. The results are below:

1. Each result was ran at a 95% confidence level alpha = .05
2. Null hypothesis = The difference in means is 0
3. Alternative hypothesis = The difference in means is not 0
4. Beef vs mu = 425 - 0.1175 > .05 Null hypothesis can't be rejected, means are likely the same
5. Meat vs mu = 425 - 0.7799 > .05 Null hypothesis can't be rejected, means are likely the same
6. Poultry vs mu = 425 - 0.1175 > .05 Null hypothesis can't be rejected, means are likely the same

```
t.test(x = beef$Sodium, mu = 425) #testing beef hot dogs
```

```
##
##  One Sample t-test
##
## data:  beef$Sodium
## t = -1.0413, df = 19, p-value = 0.3108
## alternative hypothesis: true mean is not equal to 425
## 95 percent confidence interval:
##  353.2091 449.0909
## sample estimates:
## mean of x
##    401.15
```

```
t.test(x = meat$Sodium, mu = 425) #testing meat hot dogs
```

```
##
##  One Sample t-test
##
## data:  meat$Sodium
## t = -0.2842, df = 16, p-value = 0.7799
## alternative hypothesis: true mean is not equal to 425
## 95 percent confidence interval:
##  370.2647 466.7941
## sample estimates:
## mean of x
##  418.5294
```

```
t.test(x = poultry$Sodium, mu = 425) #testing poultry hot dogs
```

```
##
##  One Sample t-test
##
## data:  poultry$Sodium
## t = 1.6543, df = 16, p-value = 0.1175
## alternative hypothesis: true mean is not equal to 425
## 95 percent confidence interval:
##  415.4311 502.5689
## sample estimates:
## mean of x
##       459
```

Hypothesis Testing on Forbes' CEO Salary Data

In this section, CEO salary data from Forbes will be analyzed to:

1. Read in the data to a data frame
2. Check the structure and summary stats for the data
3. Test the hypothesis at 95% confidence that at least 50% of the CEOs are 45 years old or older.
4. Test the hypothesis at 95% confidence that at least 50% of the CEOs earn less than $500,000 per year

```
CEOsalary <- read.csv("salaries.csv") #reading data into R

str(CEOsalary) #checking the structure of the data
```

```
## 'data.frame':    60 obs. of  2 variables:
##  $ AGE: int  53 43 33 45 46 55 41 55 36 45 ...
##  $ SAL: int  145 621 262 208 362 424 339 736 291 58 ...
```

```
summary(CEOsalary) #checking summary stats for the data
```

```
##       AGE             SAL
##  Min.   :32.00   Min.   :  21.0
##  1st Qu.:45.75   1st Qu.: 250.0
##  Median :50.00   Median : 350.0
##  Mean   :51.47   Mean   : 404.6
##  3rd Qu.:57.00   3rd Qu.: 537.8
##  Max.   :74.00   Max.   :1103.0
```

Next, the age and the salaries were analyzed. Proportion test was used because the goal was to see if at least 50% of the CEOS were older than 45 and then the salary data was tested to see if atleast 50% of the CEOS were paid less than $500,000. First, the salaries that fit the tested hypothesis was isolated and the count of the group was calculated. Then the total number of CEOs in the data set was stored in a variable and the proportion test was conducted. The results are below:

1. Each result was ran at a 95% confidence level alpha = .05
2. Null hypothesis = true proportion is not greater than .5
3. Age: 0.7166667 > .05 Null can't be rejected, likely that proportion of CEOs greater than or equal to
4. Salary: 0.8166667 > .05 Null can't be rejected, likely that proportion of CEOs who get paid less tha

```
ceoage45<- CEOsalary$AGE >= 45
countage<- sum(ceoage45)
totalage<-length(CEOsalary$AGE)
prop.test(x = countage, n = totalage, alternative = "greater")
```

```
##
##  1-sample proportions test with continuity correction
##
## data:  countage out of totalage, null probability 0.5
## X-squared = 22.817, df = 1, p-value = 8.911e-07
## alternative hypothesis: true p is greater than 0.5
## 95 percent confidence interval:
##  0.7121941 1.0000000
## sample estimates:
##         p
## 0.8166667
```

```r
ceopay<- CEOsalary$SAL <= 500
countpay<- sum(ceopay)
totalpay<-length(CEOsalary$SAL)
prop.test(x = countpay, n = totalpay, alternative = "greater")
```

```
##
##  1-sample proportions test with continuity correction
##
## data:  countpay out of totalpay, null probability 0.5
## X-squared = 10.417, df = 1, p-value = 0.0006244
## alternative hypothesis: true p is greater than 0.5
## 95 percent confidence interval:
##  0.6045034 1.0000000
## sample estimates:
##         p
## 0.7166667
```