Unit 6 Homework: Tests and and Confidence Intervals

w203: Statistics for Data Science

Low-Oxygen Statistics

The file `expeditions.csv` contains data about 10,000 climbing expeditions in the Himalayan Mountains of Nepal. The data was compiled by the Himalayan Database and published in csv format on Tidy Tuesday.

First, navigate to https://github.com/rfordatascience/tidytuesday/tree/master/data/2020/2020-09-22 to read some basic information about the data and examine the codebook.

The variable `highpoint_metres` represents the highest elevation reached by each expedition. Your task is to test whether the mean highest elevation is above 7400 meters.

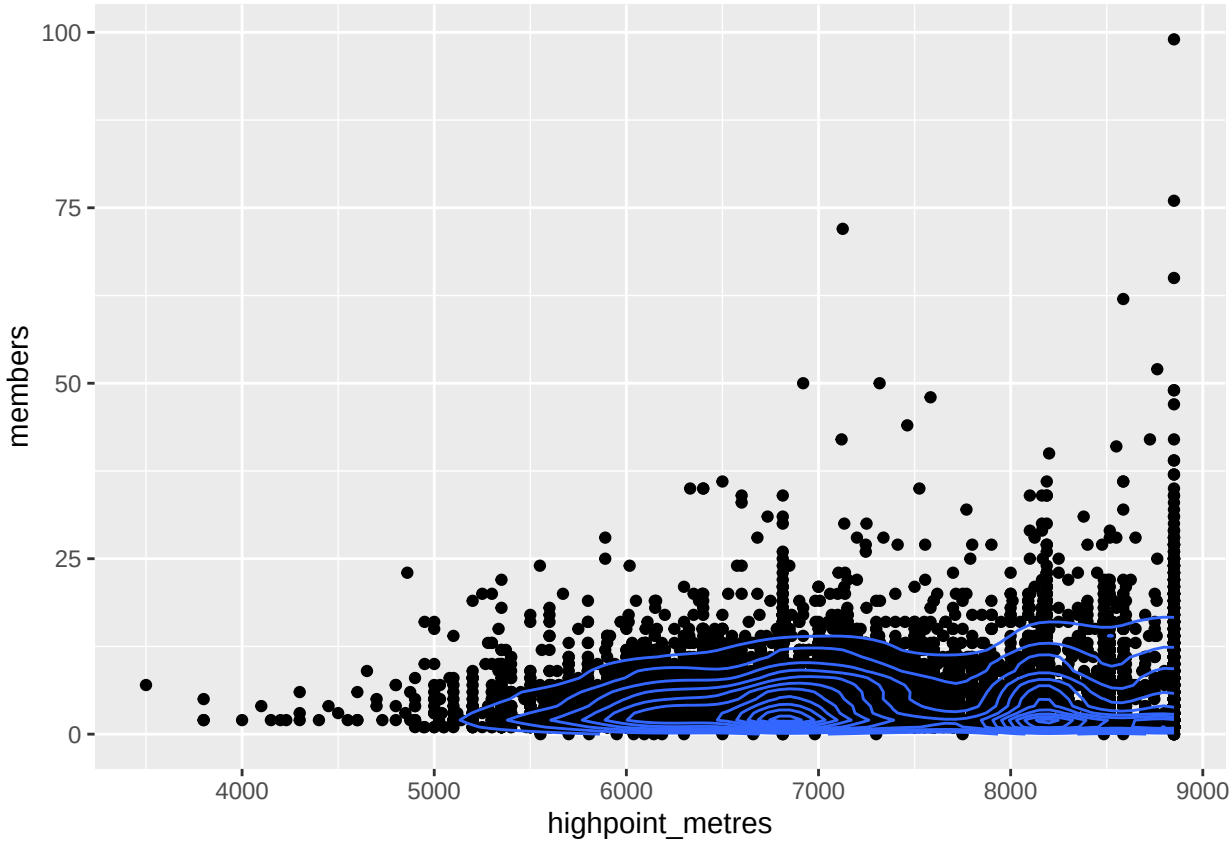
a. Using the documentation about the data, your background knowledge, and the data itself, assess whether the assumptions underlying a valid t-test are met. If plots are useful to make this argument, include them; if numeric statements are useful to make this argument, use them.

```
library(ggplot2)
e<-read.csv('e2.csv', header=TRUE)

mn=e$highpoint_metres
mn00<-mean(mn)
print(mn00)
```

```
## [1] 7408.924
```

```
ggplot(e, aes(x = highpoint_metres, y = members))+
  geom_point()+
  stat_density2d()
```



b. Provide an argument for why you should conduct a two-tailed test in this case, even though your personal interest is primarily in whether the mean is higher than 7400.

c. Compute the t-statistic by plugging in the values from the data manually into the formula. A *great* solution would write a function (perhaps called `t_statistic`) that takes arguments and returns a value. However, writing a function isn't necessary for a full solution. Feel free to use functions `mean()`, `sd()`, and `sqrt()`.

```
t_statistic <- function(highpoint_metres, mean_highest_elevation) {
   t <- (mean(highpoint_metres)-mean_highest_elevation)/
     (sd(highpoint_metres)/sqrt(length(highpoint_metres)))
   cat("t = ", t)
}

d<- e$highpoint_metre
h<-7400

t<-t_statistic(d, h)
```

```
## t =  0.8790473
```

d. Using `qt()`, compute the t-critical value for a two-tailed test.

```r
#df is degree of freedom

df<-length(data)-1

df
```

```
## [1] 0
```

```r
#compute the t-critical value for a left-tailed test
#t_critical_left_tailed<-qt(p=.05, df, lower.tail=TRUE)
#cat("t_critical_left_tailed=", t_critical_left_tailed)
#compute the t-critical value for a right-tailed test
#t_critical_right_tailed<-qt(p=.05, df, lower.tail=FALSE)
#cat("t_critical_right_tailed=", t_critical_right_tailed)

#compute the t-critical value for a two-tailed test
t_critical_two_tailed<-qt(p=.05/2, df, lower.tail=FALSE)
```

```
## Warning in qt(p = 0.05/2, df, lower.tail = FALSE): NaNs produced
```

```r
cat("t_critical_two_tailed=", t_critical_two_tailed)
```

```
## t_critical_two_tailed= NaN
```

When perform a two-tailed test, there will be two critical values. In this case, the T critical values are 1.960202 and -1.960202. Thus, if the test statistic is less than -1.960202 or greater than 1.960202, the results of the test are statistically significant.

e. Compute the p-value for your two-tailed test. You may use the `pt()` function.

```r
t<-0.879043

p_value<-2*pt(-abs(t),df=length(data)-1)
```

```
## Warning in pt(-abs(t), df = length(data) - 1): NaNs produced
```

```r
p_value
```

```
## [1] NaN
```

f. Explain what your rejection decision should be in two ways.

g. Confirm that your work is correct, by running the `t.test` command.

```r
t.test(e$highpoint_metres, mu=7400, alternative = "two.sided")
```

```
##
##  One Sample t-test
##
## data:  e$highpoint_metres
## t = 0.87905, df = 9949, p-value = 0.3794
## alternative hypothesis: true mean is not equal to 7400
## 95 percent confidence interval:
##  7389.024 7428.823
## sample estimates:
## mean of x
##  7408.924
```

g. Evaluate the practical significance of your result.