BACS-hw04-107070004

Question 1)

Some years ago, a Google engineer explained how they spot malicious apps (malware) on their Android mobile apps store. Some malware apps deliberately turn off a security feature on Android called Verify as soon as they are installed on an Android device. But there are also other, non-malicious, reasons why Verify might get turned off. So Google computes a "DOI score" for each app. The distribution of DOI scores is binomial, which Google approximates as a normal distribution (recall our reading on binomial distributions).

(a) Given the critical DOI score that Google uses to detect malicious apps (-3.7), what is the probability that a randomly chosen app from Google's app store will turn off the Verify security feature? (report a precise decimal fraction, not a percentage)

```r
pnorm(-3.7)
```

```
## [1] 0.0001077997
```

(b) Assuming there were ~2.2 million apps when the article was written, what number of apps on the Play Store did Google expect would maliciously turn off the Verify feature once installed?

```r
2200000*pnorm(-3.7)
```

```
## [1] 237.1594
```

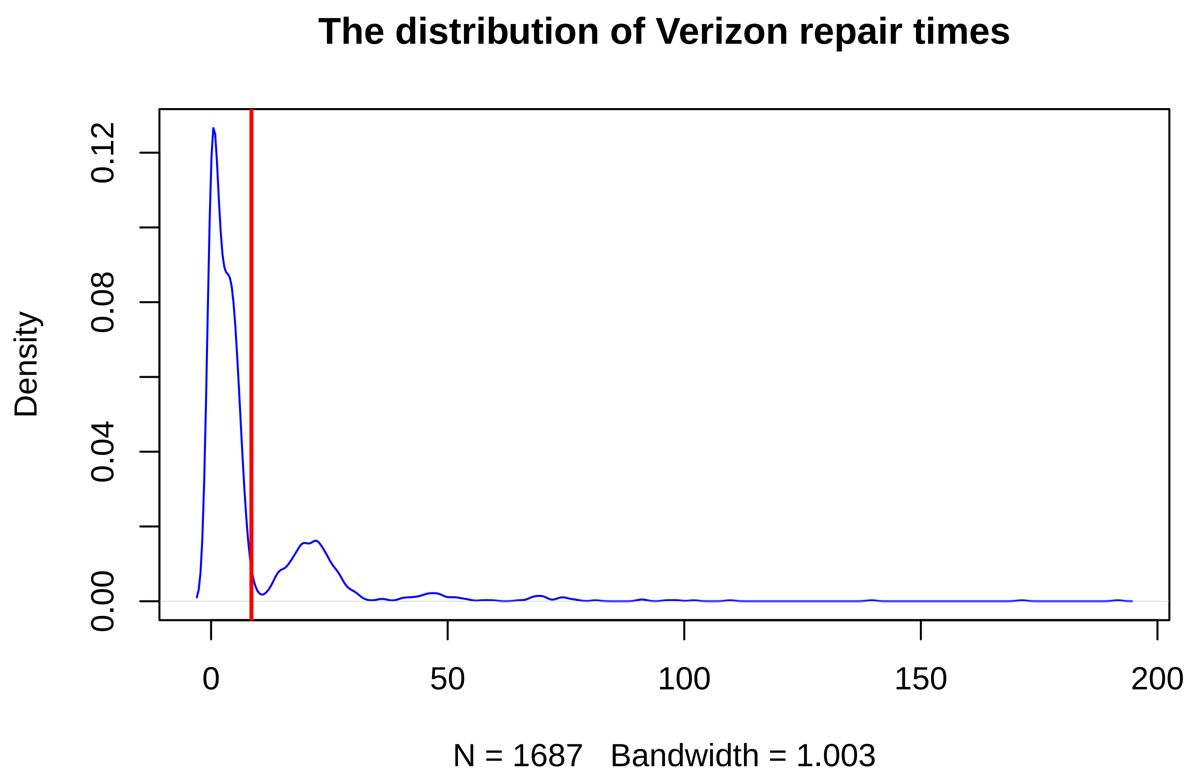
Question 2)

The following problem's data is real but the scenario is strictly imaginary The large American phone company Verizon had a monopoly on phone services in many areas of the US. The New York Public Utilities Commission (PUC) regularly monitors repair times with customers in New York to verify the quality of Verizon's services. Imagine that Verizon claims that they take 7.6 minutes to repair phone services for its customers on average. The file verizon.csv has a recent sample of repair times collected by PUC, who seeks to verify this claim at 99% confidence.

(a) The Null distribution of t-values:

- Visualize the distribution of Verizon's repair times, marking the mean with a vertical line

```
repairtimes <- read.csv("verizon.csv", header=TRUE)$Time
plot(density(repairtimes), col="blue", main="The distribution of Verizon repair times")
abline(v=mean(repairtimes), lwd=2, col="red")
```



- Given what PUC wishes to test, how would you write the hypothesis? (not graded)

$H_0 : \mu = 7.6$ $H_1 : \mu \neq 7.6$

- Estimate the population mean, and the 99% confidence interval (CI) of this estimate

```
population_mean <- mean(repairtimes)
population_mean
```

```
## [1] 8.522009
```

```
tra_ci_99 <- quantile(repairtimes, probs = c(0.005, 0.995))
tra_ci_99
```

```
##    0.5%   99.5%
##  0.0000 86.8103
```

- Using the traditional statistical testing methods we saw in class, find the t-statistic and p-value of the test

```
repairtimes_hyp <- 7.6
size <- length(repairtimes)
mean <- mean(repairtimes)
sd <- sd(repairtimes)
se <- (sd /sqrt(size))
t <- (mean(repairtimes)-repairtimes_hyp)/se
t
```

```
## [1] 2.560762
```

```
df <- size - 1
p <- 1 - pt(t, df)
p
```

```
## [1] 0.005265342
```

- Briefly describe how these values relate to the Null distribution of t (not graded)

If | t value | $<$ 2.58, the claim is included in the 99% null distribution. If the p value $>$ (1-99%)/2, the claim is included in the 99% null distribution.

- What is your conclusion about the advertising claim from this t-statistic, and why?

The t-value and p-value are both not in the rejected region. Therefore, we don't reject the null hypothesis, $\mu = 7.6$;that is, the advertising claim might be correct under 99% CI.

(b) Let's use bootstrapping on the sample data to examine this problem:

```
num_boots <- 2000
sample_statistic <- function(sample0) {
    resample <- sample(sample0, length(sample0), replace=TRUE)
    mean(resample)
    # stat_function(resample)
}
```

- Bootstrapped Percentile: Estimate the bootstrapped 99% CI of the mean

```
sample_means <- replicate(num_boots, sample_statistic(repairtimes))
mean_ci_99 <- quantile(sample_means, probs = c(0.005, 0.995))
mean_ci_99
```

```
##     0.5%    99.5%
## 7.614679 9.511959
```

- Bootstrapped Difference of Means: What is the 99% CI of the bootstrapped difference between the population mean and the hypothesized mean?

```
boot_mean_diffs <- function(sample0, mean_hyp) {
    resample <- sample(sample0, length(sample0), replace=TRUE)
    return( mean(resample) - mean_hyp )
}
set.seed(10000000)
mean_diffs <- replicate(num_boots,boot_mean_diffs(repairtimes, repairtimes_hyp))
diff_ci_99 <- quantile(mean_diffs, probs=c(0.005, 0.995))
diff_ci_99
```

```
##      0.5%     99.5%
## 0.0729364 1.8325855
```

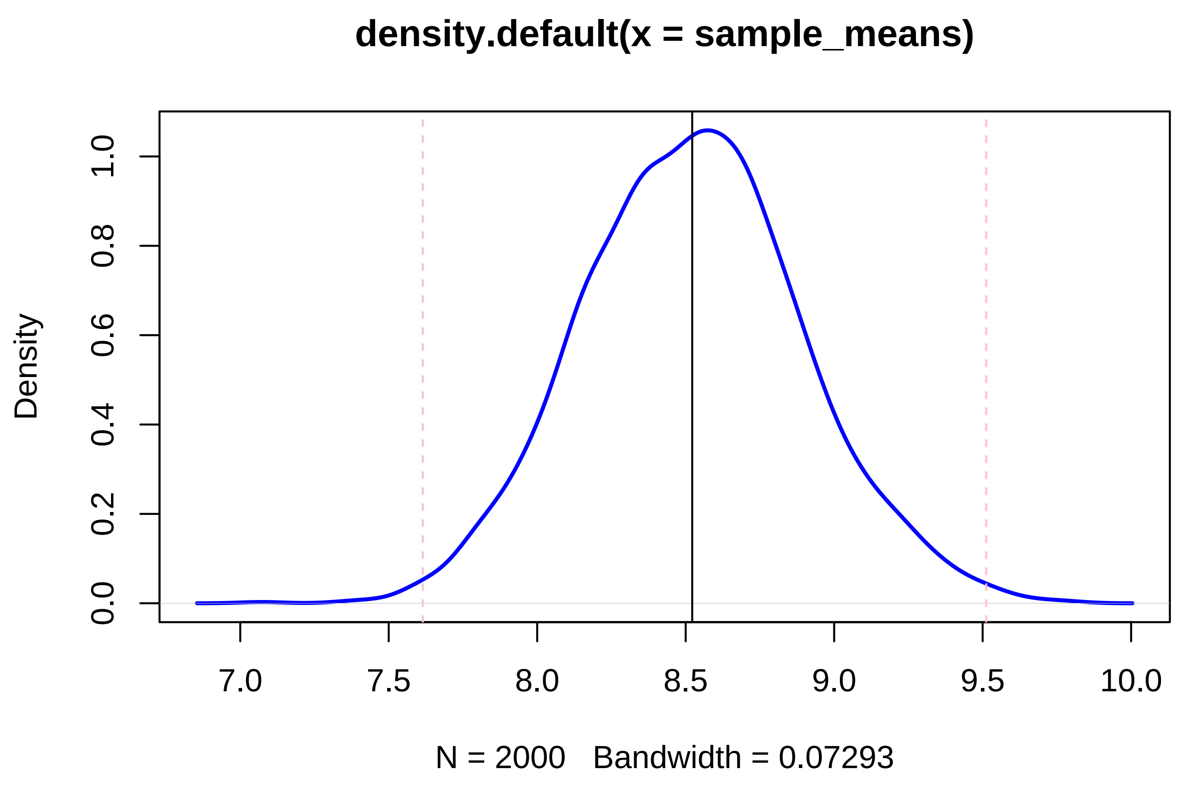
- Bootstrapped t-Interval: What is 99% CI of the bootstrapped t-statistic?

```
boot_t_stat <- function(sample0, mean_hyp) {
    resample <- sample(sample0, length(sample0), replace=TRUE)
    diff <- mean(resample) - mean_hyp
    se <- sd(resample)/sqrt(length(resample))
    return( diff / se )
}
set.seed(10000000)
t_boots <- replicate(num_boots, boot_t_stat(repairtimes, repairtimes_hyp))
t_boots  <- na.omit(t_boots)
t_ci_99 <- quantile(t_boots, probs=c(0.005, 0.995))
t_ci_99
```

```
##      0.5%     99.5%
## 0.2278367 4.5485614
```

- Plot separate distributions of all three bootstraps above (for ii and iii make sure to include zero on the x-axis)

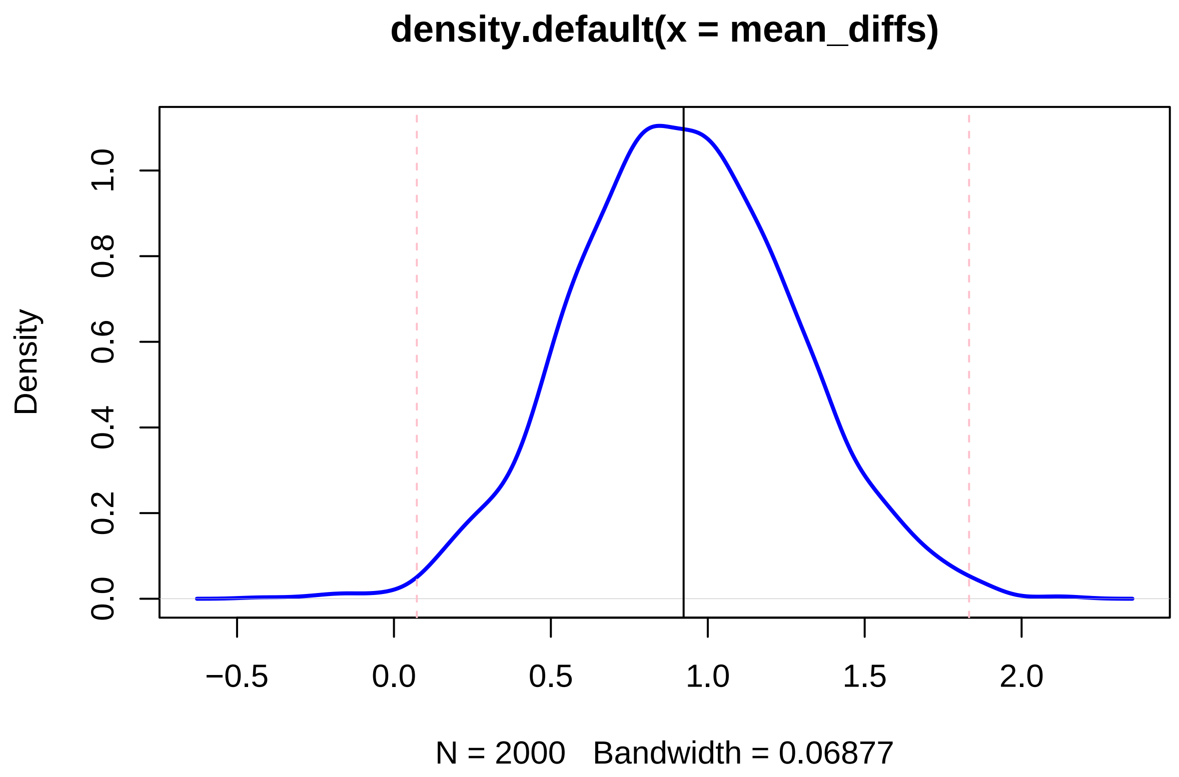
```
plot(density(sample_means), col="blue", lwd=2)
abline(v=mean(sample_means))
abline(v=mean_ci_99, lty="dashed", col="pink")
```



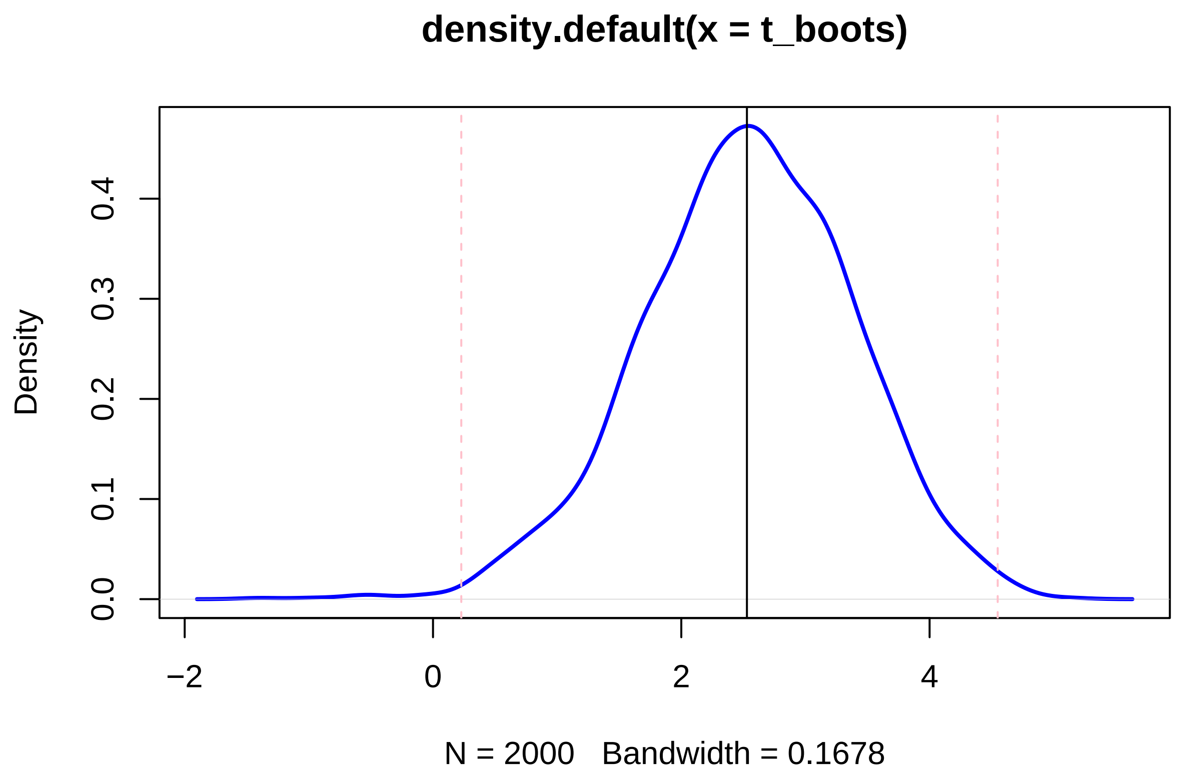
```
plot(density(mean_diffs), col="blue", lwd=2)
mean(mean_diffs)
```

```
## [1] 0.9230387
```

```
abline(v=mean(mean_diffs))
abline(v=diff_ci_99, lty="dashed", col="pink")
```



```r
plot(density(t_boots), col="blue", lwd=2)
abline(v=mean(t_boots))
abline(v=t_ci_99, lty="dashed", col="pink")
```



(c) Do the four methods (traditional test, bootstrapped percentile, bootstrapped difference of means, bootstrapped t-Interval) agree with each other on the test?

- traditional test : **Don't reject the H0**

As above, the t-value and p-value are both not in the rejected region, so the claim is reasonable under 99% CI.

- bootstrapped percentile : **Don't reject the H0**

```
mean_ci_99
```

```
##     0.5%    99.5% 
## 7.614679 9.511959
```

7.6 isn't in the 99% CI.

- bootstrapped difference of means : **Reject the H0**

```
diff_ci_99
```

```
##      0.5%     99.5% 
## 0.0729364 1.8325855
```

The 99% CIs of the difference do not contain zero.

- bootstrapped t-Interval : **Reject the H0**

```
t_ci_99
```

```
##      0.5%     99.5% 
## 0.2278367 4.5485614
```

The 99% CI of t-statistic does not contain zero.