# BACS HW - Week 6

Let's try out bootstrapping to test hypotheses. We will use the website loading-time data that we saw in class.

Recall the example from your earlier HW about Verizon's customer response times. You might have noted that each response time was labeled as ILEC or CLEC. Here is the full story. Verizon was an Incumbent Local Exchange Carrier (ILEC), responsible for maintaining land-line phone service in certain areas. Other competing providers, termed Competitive Local Exchange Carriers (CLEC), could also sell long-distance phone services in Verizon's areas. When something went wrong, Verizon would be responsible to respond and repair services as quickly for CLEC long-distance customers as for its own ILEC customers. The New York Public Utilities Commission (PUC) monitored fairness by comparing Verizon's response times for its ILEC customers versus CLEC customers. In each case, a hypothesis test was performed at the 1% significance level, to determine whether response times for CLEC customers were significantly slower than for Verizon's customers. If Verizon failed to provide fair treatment for CLEC customers, it would have to pay large penalties.

Verizon claims that mean response time for ILEC and CLEC customers are the same, but the PUC would like to test if CLEC customers were facing greater response times.

**Question 1)** The Verizon dataset this week is provided as a "wide" data frame. Let's practice reshaping it to a "long" data frame. You may use either shape (wide or long) for your analyses in later questions.

a. Pick a reshaping package (we discussed two in class) – research them online and tell us *why* you picked it over others (provide any helpful links that supported your decision).
b. Show the code to reshape the versizon_wide.csv data
c. Show us the "head" and "tail" of the data to show that the reshaping worked
d. Visualize Verizon's response times for ILEC vs. CLEC customers

**Question 2)** Let's test if the mean of response times for CLEC customers is greater than for ILEC customers

a. State the appropriate null and alternative hypotheses (one-tailed)
b. Use the appropriate form of the `t.test()` function to test the *difference between the mean of ILEC versus CLEC response times* at 1% significance. For each of the following tests, show us the results and tell us whether you would reject the null hypothesis.
    i. Conduct the test assuming variances of the two populations are equal
    ii. Conduct the test assuming variances of the two populations are not equal
c. Use a permutation test to compare the means of ILEC vs. CLEC response times
    i. Visualize the distribution of permuted differences, and indicate the observed difference as well.
    ii. What are the one-tailed and two-tailed p-values of the permutation test?
    iii. Would you reject the null hypothesis at 1% significance in a one-tailed test?

**Question 3)** Let's use the Wilcoxon test to see if the response times for CLEC are different than ILEC.

a. Compute the *W statistic* comparing the values. You may use either the permutation approach (with either for-loops or the vectorized form) or the rank sum approach.
b. Compute the one-tailed *p-value* for W.
c. Run the Wilcoxon Test again using the `wilcox.test()` function in R – make sure you get the same W as part [a]. Show the results.
d. At 1% significance, and one-tailed, would you reject the null hypothesis that the values of CLEC and ILEC are different from one another?

**Question 4)** One of the assumptions of some classical statistical tests is that our population data should be roughly normal. Let's explore one way of visualizing whether a sample of data is normally distributed.

a. Follow the following steps to create a function to see how a distribution of values compares to a perfectly normal distribution. *The ellipses (...) in the steps below indicate where you should write your own code*.

   Make a function called `norm_qq_plot()` that takes a set of values):
   `norm_qq_plot <- function(values) { … }`
   Within the function body, create the five lines of code as follows.

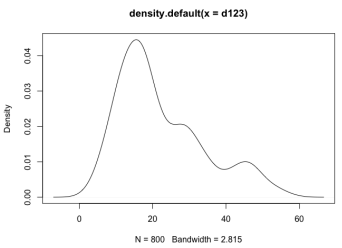
   i. Create a sequence of probability numbers from 0 to 1, with ~1000 probabilities in between
      `probs1000 <- seq(0, 1, 0.001)`
   ii. Calculate ~1000 quantiles of our `values` (you can use `probs=probs1000`), and name it `q_vals`
      `q_vals <- quantile(…)`
   iii. Calculate ~1000 quantiles of a perfectly normal distribution with the *same mean and standard deviation as our* `values`; name this vector of normal quantiles `q_norm`
      `q_norm <- qnorm(…)`
   iv. Create a *scatterplot* comparing the quantiles of a normal distribution versus quantiles of `values`
      `plot(q_norm, q_vals, xlab="normal quantiles", ylab="values quantiles")`
   v. Finally, draw a red line with intercept of 0 and slope of 1, comparing these two sets of quantiles
      `abline( … , col="red", lwd=2)`

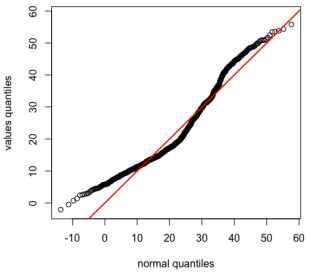
   You have now created a function that draws a "normal quantile-quantile plot" or Normal Q-Q plot (please show code for the whole function in your HW report)

b. Confirm that your function works by running it against the values of our `d123` distribution from week 3 and checking that it looks like the plot on the right:

```
set.seed(978234)
d1 <- rnorm(n=500, mean=15, sd=5)
d2 <- rnorm(n=200, mean=30, sd=5)
d3 <- rnorm(n=100, mean=45, sd=5)
d123 <- c(d1, d2, d3)

plot(density(d123))
norm_qq_plot(d123)
```




   Interpret the plot you produced (see this article on how to interpret normal Q-Q plots) and tell us if it suggests whether `d123` is normally distributed or not.

c. Use your normal Q-Q plot function to check if the values from each of the CLEC and ILEC samples we compared in question 2 could be normally distributed. What's your conclusion?