DU Crime: Evaluating Liquor Law Counts in September vs May

Daniel Parada & Marnie Biando

March 16, 2018

Overview

Our final project for COMP 4441 (Probability and Statistics for Data Science) investigates the mean counts for crime categories as reported to the Department of Crime at the University of Denver.

Our dataset includes 11 years of data (2007-2017), which at first glance seems like ample amounts of data. However, once we looked into the data in greater detail, we found that, for the purposes of predicting the number of crimes for a given category and a given month, we would need to summarize data by crime category, by month, leaving us with one datapoint per year, or a total of 11 data points in each of our samples.

Plotting the trends of crime counts by category demonstrated that for certain categories, the samples did not follow a Normal distribution, hence our choice of Bias-Corrected and Accelerated (BCa) to calculate predicted means with confidence intervals.

Data and Research Question
University of Denver's Campus Security Department publishes crime reports regularly and keeps in its records, no more than 11 years worth of data. We obtained reports from Campus Security and found that crimes were categorized into over 26 categories of crime.

Data cleanup included loading of data and code to parse data by category and subcategory. We looked into several categories before identifying a few datasets to further investigate.

We settled on the Liquor Laws category. Our research question: is the average count of liquor laws broken higher at the beginning of the school year or at the end of the school year? We hypothesized that the count would be higher at the beginning of the year, when students are still new to the school (and perhaps naive to liquor laws on campus).

Statistic: BCa Confidence Intervals
Before jumping head first into the Bias-Corrected and Accelerated confidence intervals, let's take a step back and cover our bases.

Bootstrapping 101 : we have a sample with n numeric type elements $x_1, x_2, ..., x_n$ from which we have calculated a statictic of interest noted θ. From this sample we are going to resample it n times randomly with replacement and we do this N times, with N $>$ 5000. Voila, now we have 5000 bootstrap distributions with n number of elements in each for each of which we also calculate our statistic of interest noted $\hat{\theta}_1^*, \hat{\theta}_2^*, ..., \hat{\theta}_N^*$.

The most intuitive way of calculating the confidence intervals we would use the percentile method where we use $100 * \alpha^{th}$, as lower bound, and $100 * (1 - \alpha)^{th}$, as the upper bound. The Bias-Corrected Accelerated confidence interval is based on the percentile method but instead of using the percentile directly we will first take into account the skewness of our data, which would result in skewed bootstrap distributions.

To take this into account we need to :

- calculate the bias-correction parameter $\hat{z}_0$. With $\hat{z}_0$ is the proportion of the statistic of interest calculated on each the bootstrap samples, $\hat{\theta}_1^*, ..., \hat{\theta}_N^*$, that is less than our sample data statistic, noted $\hat{\theta}^*$.

$\hat{z}_0 = \Phi^{-1}(\frac{\#(\hat{\theta}_b^* < \hat{\theta})}{B})$

with Φ^{-1} the inverse of the CDF (cummulative distribution function), # the number of times $\hat{\theta}_b^*$ (statistic of interest calculated on a bootstrap sample) is less than $\hat{\theta}$ (statistic of interest calculated on original sample).

- calculate the acceleration parameter a, which corresponds to how far we are from the true value of the statistic of interest.

$\hat{a} = \frac{\sum_{i=1}^{n}(\hat{\theta}_{(.)} - \hat{\theta}_{(i)})^3}{6(\sum_{i=1}^{n}(\hat{\theta}_{(.)} - \hat{\theta}_{(i)})^2)^{\frac{3}{2}}}$

with $\sum_{i=1}^{n}(\hat{\theta}_{(.)} - \hat{\theta}_{(i)})^3$ the expected value and $\sum_{i=1}^{n}(\hat{\theta}_{(.)} - \hat{\theta}_{(i)})^2)^{\frac{3}{2}}$ being the variance.

This results in an adjusted upper and lower bound for the confidence interval :

$\alpha_1 = \Phi(\hat{z}_0 + \frac{\hat{z}_0 + z_{(\alpha)}}{1 - \hat{a}(\hat{z}_0 + z_{(\alpha)})})$

$\alpha_2 = \Phi(\hat{z}_0 + \frac{\hat{z}_0 + z_{(1-\alpha)}}{1 - \hat{a}(\hat{z}_0 + z_{(1-\alpha)})})$

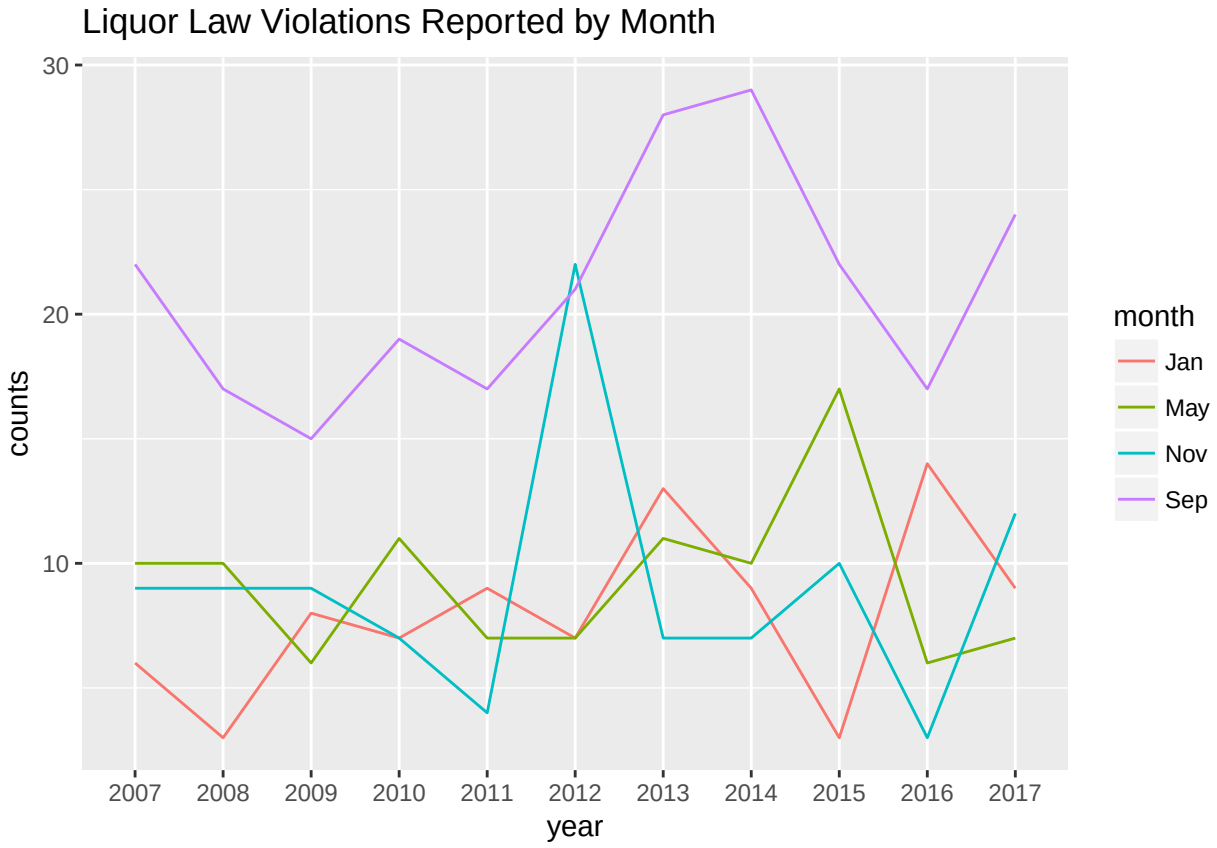
Loading of Dataset & Exploratory Data Analysis

For the sake of brevity, the data for loading 10 years of campus crime reports has been left out of this report. The R code below loads a cleaned up CSV file that was generated from filtering for Liquor Law Crime Counts from 2007 to 2017.

A simple plot of the total liquor law counts reported by year shows definite skew in the data, making it perfect for applying BCa after 10,000 samples are generated from the sample data.

```
## Parsed with column specification:
## cols(
##   `2007` = col_integer(),
##   `2008` = col_integer(),
##   `2009` = col_integer(),
##   `2010` = col_integer(),
##   `2011` = col_integer(),
##   `2012` = col_integer(),
##   `2013` = col_integer(),
##   `2014` = col_integer(),
##   `2015` = col_integer(),
##   `2016` = col_integer(),
##   `2017` = col_integer()
## )
```

```
## Warning: package 'bindrcpp' was built under R version 3.4.4
```



Boot Package in R: boot() and boot.ci() functions

R contains a package "boot" which generates bootstrap replicates of your statistic of choice.

The boot() function has several arguments, but the required ones are:
- data (the data you will be resampling from) - statistic (you must provide a function to calculate one or more statistics) - R (number of replicates)

boot() produces a 'bootobject' which is a collection of bootstrapped samples and the bootstrap statistic calculated form the bootstrapped samples.

boot.ci() generates different types of confidence intervals, using different formulae for each: Normal, Basic, Studentized, Percentile, and BCa.

Function to Calculate Statistic

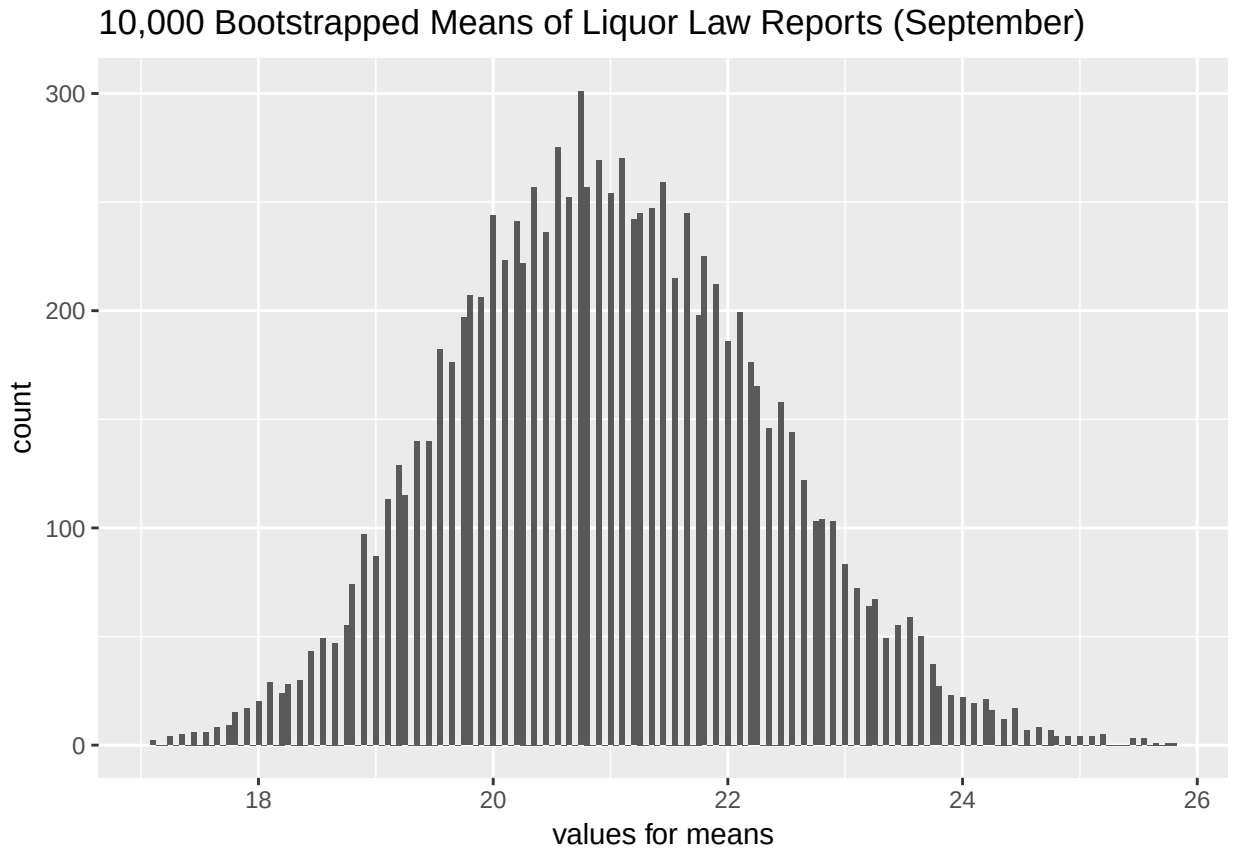
In our case, we needed our function to calculate the mean of the crime counts. After we figured out how indices work with the boot() function (indices tells the boot function how to apply the statistics to your dataset, by row or by column), we wrote two different functions to calculate means in one of two ways: (1) calculate bootstrapped means by the year: df[,indices] (2) calculate bootstrapped means by the month: df[indices,]

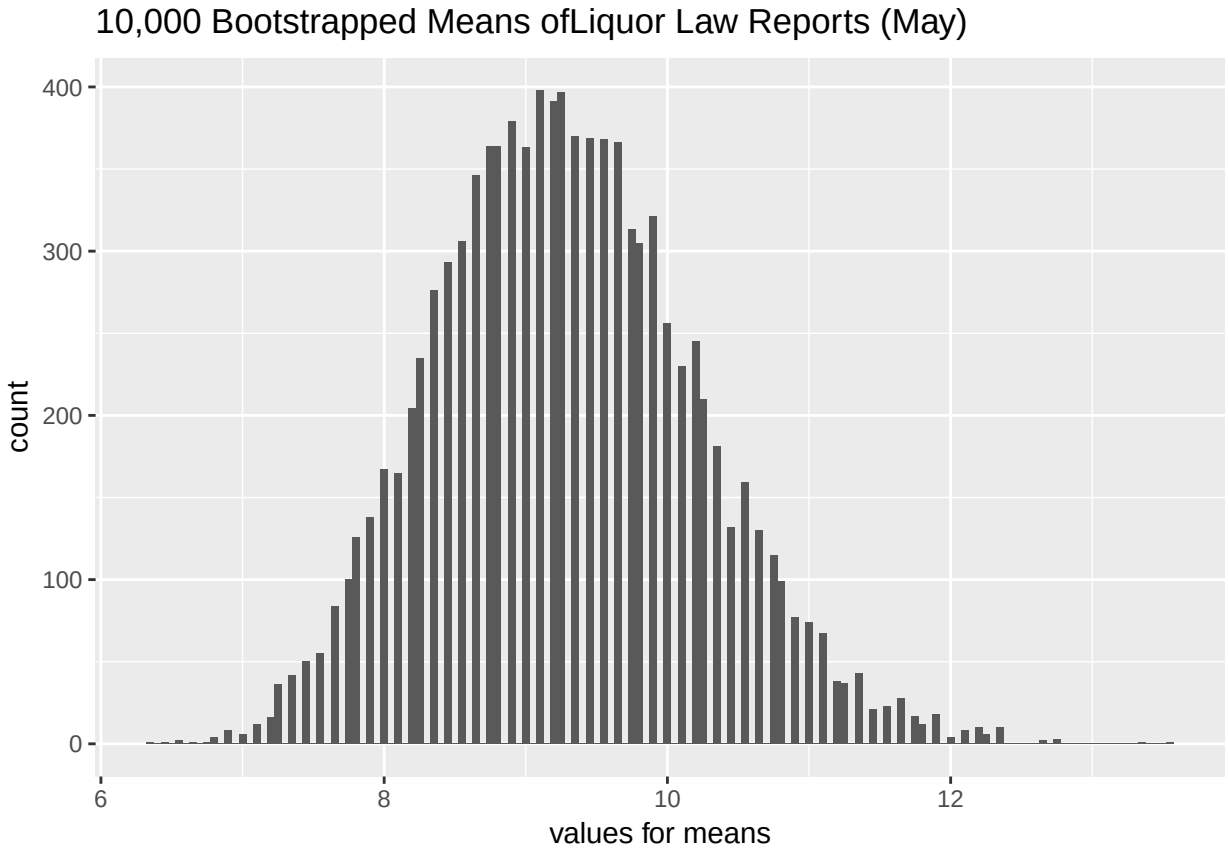
The boot function generates the following output values:

```
##  [1] "t0"        "t"         "R"         "data"      "seed"
##  [6] "statistic" "sim"       "call"      "stype"     "strata"
## [11] "weights"
```

Visual of Bootstrapped Means

Use bootobject$t to get the bootstrapped statistics for your bootobject:





Compute P-value for Bootstrapped Means

To calculate the p-value, we apply the definition of p-value and count the number of bootstrapped means that are equal to or greater than our observed mean, the mean from our samples for September and May.

$$p = \frac{1 + count(bootstrappedMeans_{month} >= sampleMean_{month})}{N}$$

The p-values for our bootstrapped means for September and May are:

```
## [1] 0.4964
```

```
## [1] 0.5067
```

Using Boot.ci to Compute Confidence Intervals

Once you have used boot to produce a boot object, you can now generate five different types of confidence intervals.

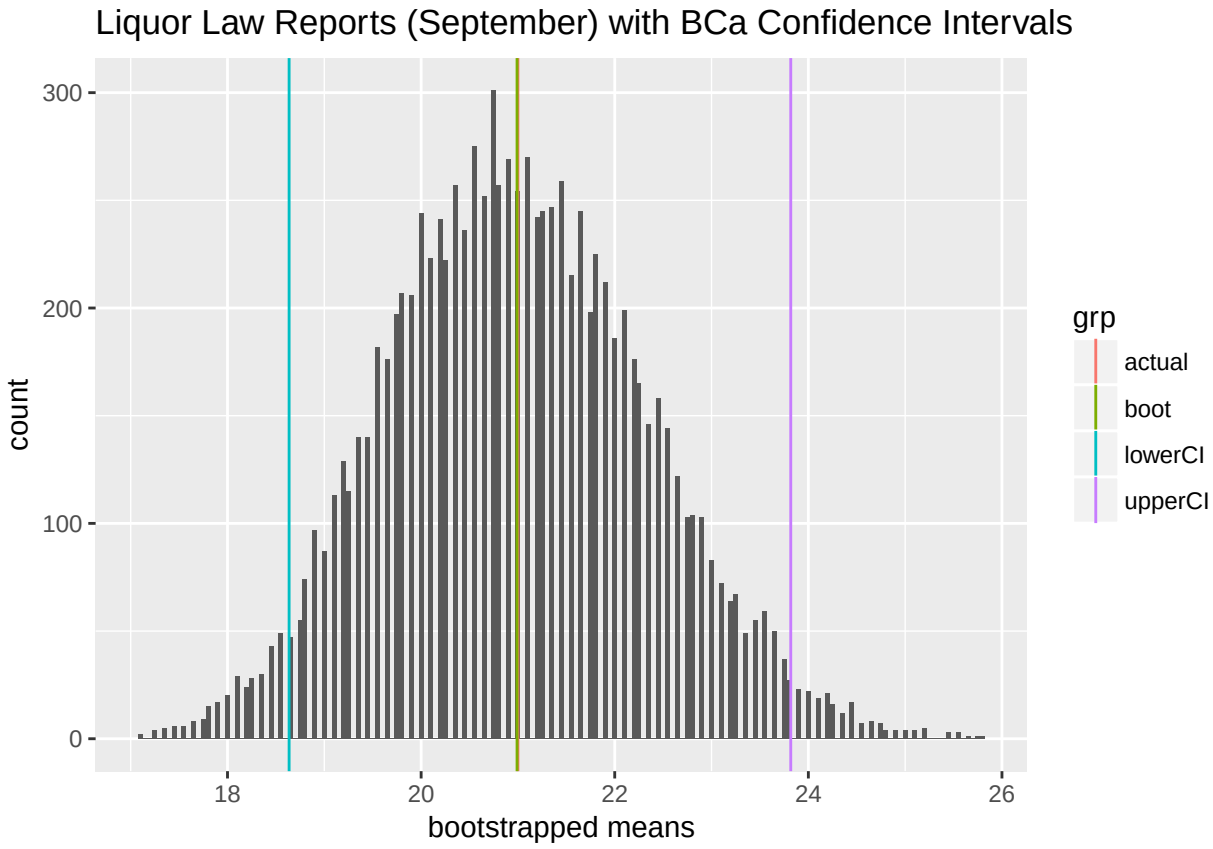
```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 10000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = bootMonth, conf = 0.95, type = "all")
##
## Intervals :
```

```
## Level      Normal              Basic             Studentized
## 95%   ( 6.071,  9.935 )   ( 6.091,  9.909 )   ( 6.122, 10.017 )
##
## Level     Percentile            BCa
## 95%   ( 6.091,  9.909 )   ( 6.091,  9.909 )
## Calculations and Intervals on Original Scale
```

September: bootstrapped values versus actual values

Here we call the boot.ci function and pass it the bootObject generated by the boot function. The output of this function is the bootstrapped confidence intervals, which we apply to our previous graph of the 10,000 botstrapped means. We also plotted the bootstrapped mean (mean of the means as well as the actual mean of our original sample:

```
## Warning in boot.ci(bootMonth, conf = 0.95, type = "all", index = 9):
## bootstrap variances needed for studentized intervals
```




```
## [1] 21
```

```
## [1] 20.99085
```

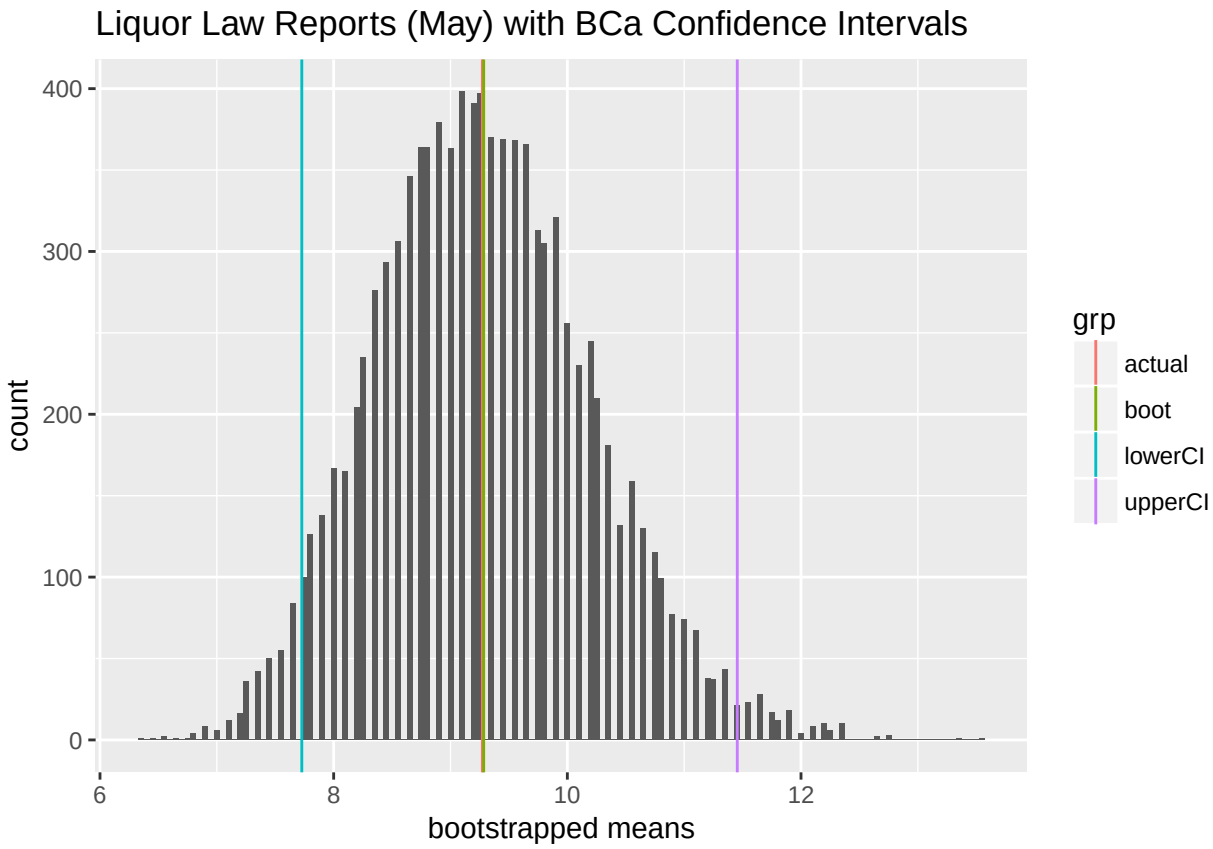
```
## [1] 18.63636
```

```
## [1] 23.81818
```

May: bootstrapped values versus actual values

We repeat the same steps for the moth of May.

```
## Warning in boot.ci(bootMonth, conf = 0.95, type = "all", index = 5):
## bootstrap variances needed for studentized intervals
```




```
## [1] 9.272727
## [1] 9.284073
## [1] 7.727273
## [1] 11.45455
```

View of bootstrapped means across all 12 months

Using the same methods we used to calculate a single bootstrapped mean (a mean of the means) for September and May, we can get a single value for all 12 months and compare the bootstrapped means to the actual means of our original dataset (11 year of data for all 12 months in each year).

```
## [1] 88
## [1] 90
## [1] 74
## [1] 79
## [1] 102
## [1] 11
## [1] 2
## [1] 9
## [1] 231
## [1] 221
## [1] 99
```

```
## [1] 3
```

```
##    actual means boot means
## 1     8.0000000  7.9968727
## 2     8.1818182  8.1714091
## 3     6.7272727  6.7153000
## 4     7.1818182  7.1748364
## 5     9.2727273  9.2840727
## 6     1.0000000  0.9931818
## 7     0.1818182  0.1828000
## 8     0.8181818  0.8120727
## 9    21.0000000 20.9908455
## 10   20.0909091 20.0848000
## 11    9.0000000  8.9923091
## 12    0.2727273  0.2725636
```

Conclusion

Despite have a small sample (n=11), we were able to find a mean value for the number of liquor laws reported for any given month, based on 11 years of Liquor Law crime reports.

Using our BCa intervals, we can conclude that 95% of the time the number of liquor laws reported in the months of September and May will be: September: 18.54545 - 23.72727
May: 7.727273 - 11.63636

There is no overlap between these ranges, so we can say definitively that the number of liquor laws in September will be greater than the number of liquor laws broken in May.