A Summary Report on Bootstrapping Methods

Instructor: Dr. Feng George Yu

Rabin Thapa

The content is divided into two sections: Section A introduces the concepts of Bootstrapping, and Section B summarizes the paper on Bootlier-plot.

A. BOOTSTRAPPING

Introduction

Bootstrapping is one of the techniques that come under a class of non-parametric statistics called resampling methods. It was coined by Bradly Efron in 1979 in a paper he published in *Annals of Statistics*. In most situations, bootstrap is as good as or better than jackknife (which is an earlier resampling method and is usually applied on smalled samples as it gets computationally intensive on larger ones). Following his pioneering paper in the topic, Efron recognized the broad applicability of bootstrapping for estimating standard error, confidence intervals, hypothesis testing, and other complex problems.

Let us consider a sample population of size n denoted by $\mathbf{x} = (x_1, x_2, x_3, . . ., \ x_n)$. A sample (ideally of size n) drawn out of $\mathbf{x}$ with replacement is called a bootstrap sample and is denoted by $\mathbf{x}_i^*$ where $i \in [1, n]$. A statistic s calculated on bootstrap sample is called bootstrap replication and is denoted by $s(\mathbf{x}_i^*)$. In general, a large number (tens of thousands) of bootstrap samples are drawn from sample population, and the corresponding bootstrap replicates are calculated to estimate a statistic denoted by $s(\mathbf{x})$ for the entire population.

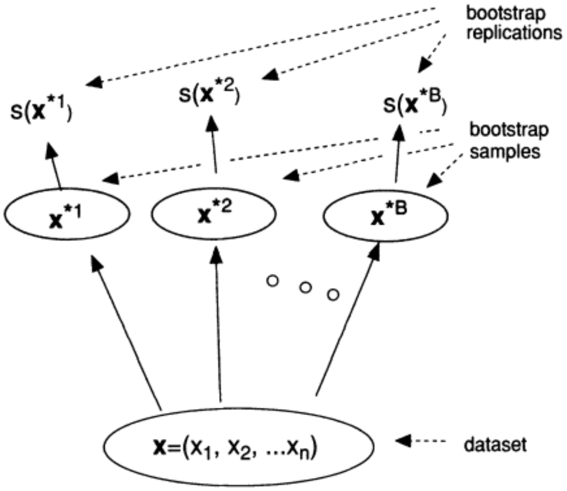


Figure 1: Bootstrap scheme for sample size n with B replications (Efron, 1994)

Estimations Using Bootstrap
Estimating the standard error of the mean: Let us consider a random sample $F \rightarrow x_1, x_2, x_3, \ldots, x_n$. We want to know $\bar{x}$ as an estimation of μ_F along with standard error of $\bar{x}$. Then, we can use plug-in principle: we substitute $\hat{F}$ in the formula $se(\bar{x}) = \sigma_F/\sqrt{n}$. Since $\mu_{\hat{F}} = \bar{x}$ and $E_{\hat{F}}g(x) = \frac{1}{n}\sum_{i=1}^{n} g(x_i)$ for any function g, the plug-in estimate of $\sigma_F = [E_F(x - \mu_F)^2]^{1/2}$ is

$$\hat{\sigma} = \sigma_{\hat{F}} = \Big[\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2\Big]^{1/2}.$$

This gives the *estimated standard error* $\hat{se}(\bar{x}) = se_{\hat{F}}(\bar{x})$,

$$\hat{se}(\bar{x}) = \sigma_{\hat{F}}/\sqrt{n} = \Big[\sum_{i=1}^{n}(x_i - \bar{x})^2/n^2\Big]^{1/2}.$$

Here, the statistic is the mean; however, this approach can be used in any statistic $s(\mathbf{x})$.

The bootstrap estimate of standard error: Let $\hat{F}$ be the empirical distribution with a probability of $1/n$ on each observed values x_i for $i = 1, 2, ..., n$. Then we draw B independent bootstrap samples $x^{*1}, x^{*2}, ..., x^{*n}$, each consisting of n data points sampled from x. Then, bootstrap replication is calculate corresponding to each sample,

$$\hat{\theta}^*(b) = s(x^{*b}) \text{ for } b = 1, 2, 3, ..., n.$$

Then the standard error $\hat{se}_F(\hat{\theta})$ by the sample standard deviation of the B replications.

$$\hat{se}_B = \Big[\sum_{b=1}^{B}[\hat{\theta}^*(b) - \hat{\theta}^*(\cdot)]^2/(B-1)\Big]^{1/2}.$$

where $\hat{\theta}^*(\cdot) = \sum_{b=1}^{B}\hat{\theta}^*(b)/B$. The limit of $\hat{se}_B$ as B goes to infinity is the ideal bootstrap estimate of $se_F(\hat{\theta})$ which means the empirical standard deviation approaches the population standard deviation as the number of bootstrap replicates increases.

$$\lim_{B\to\infty} \hat{se}_B = se_{\hat{F}} = se_{\hat{F}}(\hat{\theta}^*).$$

Estimating Confidence Intervals: Confidence intervals can be estimated for any statistic by bootstrapping. Let us consider a bootstrap statistic $\hat{\theta}^*(b) = s(x^{*b})$ for $b = 1, 2, 3, ..., n$. The $c\%$ confidence interval for the statistic is the interval that contains $c\%$ of all $\hat{\theta}^*(b)$.

However, there are multiple variants of bootstrap methods for confidence intervals (Chernick, 2011):

- Normal-t: A parametric Student's t confidence interval, with center point the sample variance and the standard error of variance estimated from that of the resampling distribution. This differs from a parametric normal bootstrap in that the percentile of the t - distribution with $n-1$ degrees of freedom is used instead of the standard normal percentile.
- EP: Efron percentile interval, with end points the plug - in quantiles of the resampling distribution.
- BC: The Efron BC interval. Simulations have shown that the BC method is, as expected, virtually identical in estimates to the BCa interval with the acceleration $a = 0$ (i.e., adjustment for median bias only).

- BCa: The Efron bias - corrected - and - accelerated interval, with median bias correction and skew correction via a jackknife estimate of the (biased) coefficient of skewness from the original sample.
- ABC: The Efron – DiCiccio approximate bootstrap confidence interval.

Hypothesis Testing: While testing hypothesis, two hypothesis statements are created: null hypothesis, and alternate hypothesis. Null hypothesis is formulated with the hope of rejecting it. To use bootstrap for hypothesis testing, firstly, a test statistic has to be determined. Then, the sampling distribution is done by bootstrapping the distribution ”under the null hypothesis”. We reject the null hypothesis whenever the p-value is less than or equal to α (type I error); if p-value is greater than α, we fail to reject the null hypothesis.

When is Bootstrap Inconsistent?

Too Small of a Sample Size: The bootstrap distribution is the distribution of all possible samples of size n that can be drawn from the original sample with replacement. The number of such samples is given by:

$$C_n^{2n-1} = (2n-1)![n!(n-1!)].$$

The number grows quite fast as n increases due to which it computationally inefficient to consider all possible samples. However, Hall showed that even for small number like $n = 20$, using $B = 2000$ for number of replications, there is 95% chance that none of the samples will be repeated. So, a sample size of 20 can be adequate for bootstrapping.

In addition, there are a number of reasons for bootstrap to be inconsistent which are as following:

- Distributions with infinite second moments: Bootstrap can fail for distribution with infinite variance, but when appropriately normalized, the mean converges to a stable distribution.
- Estimating extreme values: The m-out-of-n bootstrap can be used instead of naive bootstrap for this situation.
- Survey sampling: For a population size of N and sample size of n, if n/N is not very small, then the variance of sample averages is larger than that based on standard theory of infinite population.
- M-dependent sequence: Using naive bootstrap methods on m-dependent sequence can cause the methods to be inconsistent. A simple remedy to it is to correct the normalization.

B. OUTLIER DETECTION METHODS

An outlier is an observation that varies significantly from other (or most of) data points. It may be caused due to measurement error, data-entry error, or other various reasons. Presence of outlier(s) in a data can a significant impact on the statistic of interest and thus the inference we make from the data. So, it is crucial in data analysis to identify and remove those outliers. Some of the most popular methods for outlier detection are (Santoyo, 2018):

- Z-Score or Extreme Value Analysis (parametric)
- Probabilistic and Statistical Modeling (parametric)
- Linear Regression Models (PCA, LMS)
- Proximity Based Models (non-parametric)
- Information Theory Models
- High Dimensional Outlier Detection Methods (high dimensional sparse data)

Under Proximity based models, there is an approach called Bootlier-plot which uses bootstrapping to detect outliers.

Bootlier-plot - Bootstrap Based Outlier Detection Plot

When there is an outlier in a dataset (say, of size n) with an outlying observation (outlier) denoted by ξ, the probability that a bootstrap sample will not include the outlier is $\left(1 - \frac{1}{n}\right) = \frac{1}{e}$ (approx. 37%). So, in a repeated bootstrap sampling, 37% of the samples will not contain the outlier.

The presence of outlier in a sample will influence the sample mean; the more extreme the outlier is, the larger is the influence. So, the means of samples (67% of total) that contain the outlier one of more times can be quite different from the samples not containing the outlier. Due to this, one can expect to see multimodality in the histogram of means of the bootstrap samples. But it turns out that unless the outlier is extremely large (than other data points), the multimodality is not quite visible. To make the multimodality visible by fleshing out the difference between samples with and without outliers, the proposed method bootstraps the statistic "mean-trimmed mean". For a sample without outlier, there is not much difference between mean and trimmed mean. But for a sample with outlier, the difference can be significant (the difference is even more for the samples containing multiple outliers). Then, a histogram is plotted out of the statistic "mean-trimmed mean" of all the samples which shows clear multimodality when the dataset contains outlier.

To provide some mathematical explanation on the presence of multimodality, consider a random sample of size n from an unknown population and let's confine ourselves to the case when there are just a few (say, $p \leq 3$ or 4)potential outliers, all in the upper side. In that case, the probability that a bootstrap draw is free of potential outliers is $\left(1 - \frac{1}{n}\right) = e^{-p}$. Now, let's suppose that $100e^{-p}$% is somewhat significant (say, 10% or more). Consider the bootstrapped statistic "mean - trimmed mean" $T(Y^*)$, where k observations are trimmed from each side in defining the trimmed mean and k/n is a small fraction. Then, we get the following:

$$T(Y^*) = \frac{1}{n}\sum_{i=1}^{n} Y_i^* - \frac{1}{n-2k}\sum_{i=k+1}^{n-k} Y_{(i)}^*$$

$$= \frac{2k}{n}\Big[\frac{1}{2k}\Big\{\sum_{i=1}^{k} Y^*_{(i)} + \sum_{i=n-k+1}^{n} Y^*_{(i)}\Big\} - \frac{1}{n-2k}\sum_{i=k+1}^{n-k} Y^*_{(i)}\Big]$$

where Y_1^*, Y_2^*, . . ., Y_n^* denote the bootstrap samples and $Y^*_{(i)}$'s are the corresponding order statistics. In the case that the extreme values are well separated from the rest of the data, the histogram of $\frac{1}{2k}\sum_{i=n-k+1}^{n} Y_{(i)}$ will have at two well-separated modes: one corresponding to the case when the Y^* does not contain any extreme values, and the other(s) to the case(s) when Y^* contains one or more extreme values. The bootlier plot gets less bumpy as the value of k is increased; theoretical studies on the "optimal choice" of k are yet to be done.

Consider the case when the population contains extreme values in both the ends. In such a situation, there are three types of bootstrap samples: (1) those that are free of potential outliers, (2) those that contain potential outliers in only one end, either lower or upper; there are bumps in bootlier plot due to extremes on both sides, and (3) those that contain potential outliers in both the ends (upper and lower); it is best to plot the one-sided bootlier plots separately.

The boolier plot is a non-parametic approach to detecting outliers. It looks for uneven separation of the extremes from the nearby data points. In doing so, the plot accounts for the spread in the neighbouring tail part as well as the overall spread of data. If the gap between an extreme value and remaining sample is filled with additional data points, the bumpiness in the bootlier plot gradually disappears which is in agreement with our intuition of outliers.

Note: If the sample size is huge (say ≥ 200) and numerous outliers are suspected, one should look at several bootlier plots at different bootstrap sizes $m = [\alpha n]$ where $\alpha \in (0, 1]$. If any one is found bumpy, that would be indicative of the presence of outlier(s).

To point out precisely which data points are potential outliers, a bootlier plot is plotted for each data point. This method can be used for both univariate and multivariate data. Consider the sample $\{X_1, X_2, ..., X_n\}$ which could be either univariate or multivariate. For a fixed i, define the set

$$\mathcal{D}_i = \{d_{ij} : j = 1, 2, ..., n\},$$

where $d_{ij} = (X_i - X_j)^T(X_i - X_j)$. Since $d_{ii} = 0$, $\mathcal{D}_i$ will always contain 0. A bootlier plot is plotted on $\mathcal{D}_i$ with only left-side trimming in the statistic "mean - trimmed mean". The extremes in the upper side is allowed to cause multimodality. In the case that X_i is well separated from rest of the data points, the elements of $\mathcal{D}_i$ (except 0 itself) will be well separated from 0; this will cause multimodality in the bootlier plot.

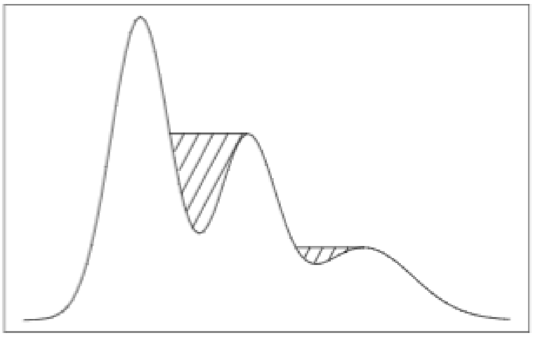


Figure 2: Bootlier Index is the value of the shaded area of a bootlier plot.

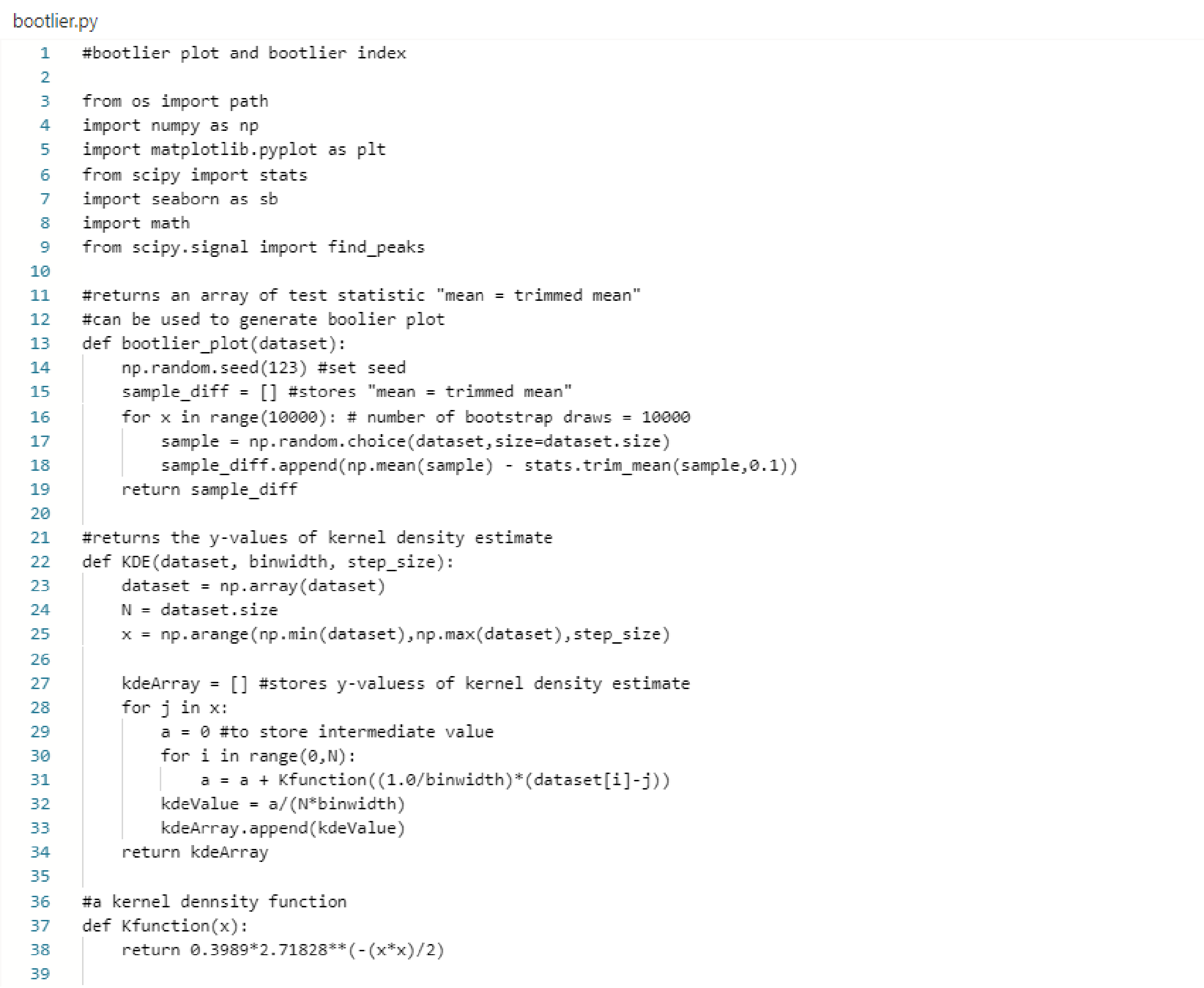
To detect the outlier(s) using this method, n bootlier plots are needed to be examined. However, the bumpy plots can easily be sorted out based on their bootlier indices. Bootlier index is an index defined to measure the bumpiness of the bootlier plot; it measured the total area of "vallies" in a bootlier plot. This index will be greater than zero when the bootlier plot is multimodal; otherwise it will be equal to zero.

Let M be the global mode of a density $f(x)$. The bootlier index (bi) is defined as

$$bi = \int_{x \geq M} \left\{ \underset{y \geq x}{sup} f(y) - f(x) \right\} dx + \int_{x \leq M} \left\{ \underset{y \leq x}{sup} f(y) - f(x) \right\} dx.$$

It can be seen from the above equation that: (1) for unimodal case, $bi = 0$ as *sup* value equals $f(x)$ for each x, and (2) for multimodal cases, $bi \geq 0$ as *sup* value is greater or equals to $f(x)$ for each x. Plots with index values between .01 and .10 are considered as border-line cases. Formal tests for outliers can be constructed with the bi as test statistic under a distributional assumption.

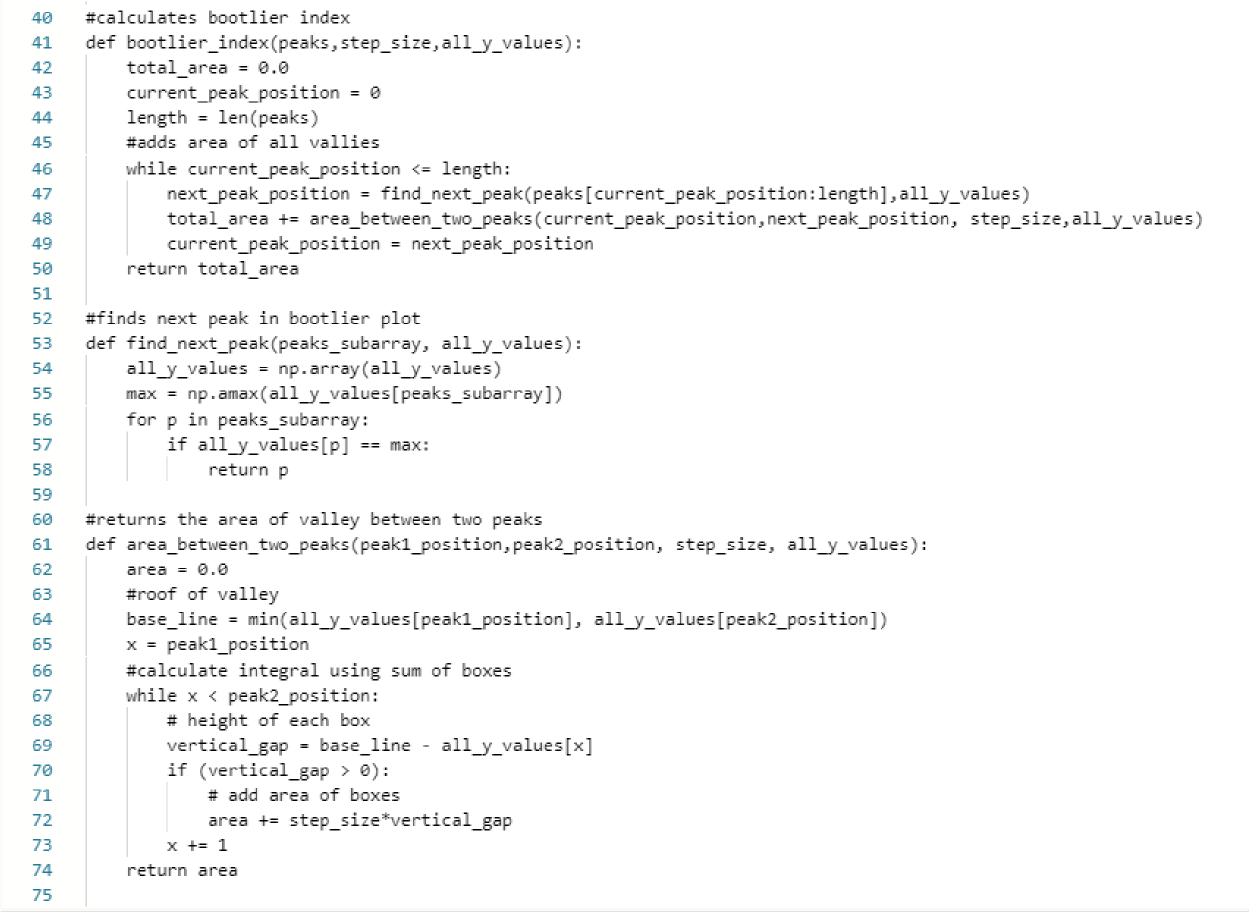
A concise version of the implementation of the concept in Python, *bootlier.py*, is as following: (*The code runs, but there are still some issues yet to be resolved.*)



bootlier.py

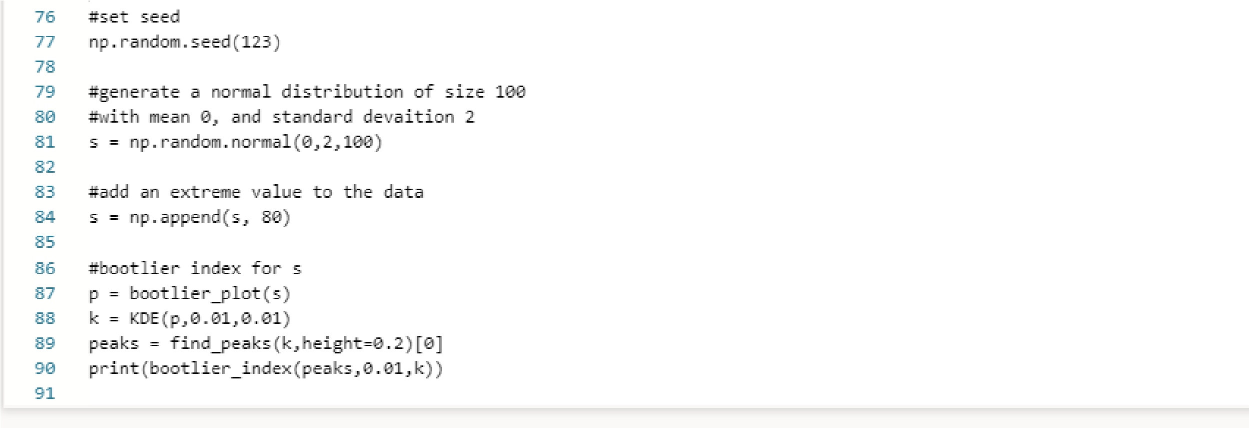
```
#bootlier plot and bootlier index

from os import path
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats
import seaborn as sb
import math
from scipy.signal import find_peaks

#returns an array of test statistic "mean = trimmed mean"
#can be used to generate boolier plot
def bootlier_plot(dataset):
    np.random.seed(123) #set seed
    sample_diff = [] #stores "mean = trimmed mean"
    for x in range(10000): # number of bootstrap draws = 10000
        sample = np.random.choice(dataset,size=dataset.size)
        sample_diff.append(np.mean(sample) - stats.trim_mean(sample,0.1))
    return sample_diff

#returns the y-values of kernel density estimate
def KDE(dataset, binwidth, step_size):
    dataset = np.array(dataset)
    N = dataset.size
    x = np.arange(np.min(dataset),np.max(dataset),step_size)

    kdeArray = [] #stores y-valuess of kernel density estimate
    for j in x:
        a = 0 #to store intermediate value
        for i in range(0,N):
            a = a + Kfunction((1.0/binwidth)*(dataset[i]-j))
        kdeValue = a/(N*binwidth)
        kdeArray.append(kdeValue)
    return kdeArray

#a kernel dennsity function
def Kfunction(x):
    return 0.3989*2.71828**(-(x*x)/2)

```

Figure 3: *bootlier.py* (part 1 of 3)



```
#calculates bootlier index
def bootlier_index(peaks,step_size,all_y_values):
    total_area = 0.0
    current_peak_position = 0
    length = len(peaks)
    #adds area of all vallies
    while current_peak_position <= length:
        next_peak_position = find_next_peak(peaks[current_peak_position:length],all_y_values)
        total_area += area_between_two_peaks(current_peak_position,next_peak_position, step_size,all_y_values)
        current_peak_position = next_peak_position
    return total_area

#finds next peak in bootlier plot
def find_next_peak(peaks_subarray, all_y_values):
    all_y_values = np.array(all_y_values)
    max = np.amax(all_y_values[peaks_subarray])
    for p in peaks_subarray:
        if all_y_values[p] == max:
            return p

#returns the area of valley between two peaks
def area_between_two_peaks(peak1_position,peak2_position, step_size, all_y_values):
    area = 0.0
    #roof of valley
    base_line = min(all_y_values[peak1_position], all_y_values[peak2_position])
    x = peak1_position
    #calculate integral using sum of boxes
    while x < peak2_position:
        # height of each box
        vertical_gap = base_line - all_y_values[x]
        if (vertical_gap > 0):
            # add area of boxes
            area += step_size*vertical_gap
        x += 1
    return area
```

Figure 4: *bootlier.py* (part 2 of 3)



```
#set seed
np.random.seed(123)

#generate a normal distribution of size 100
#with mean 0, and standard devaition 2
s = np.random.normal(0,2,100)

#add an extreme value to the data
s = np.append(s, 80)

#bootlier index for s
p = bootlier_plot(s)
k = KDE(p,0.01,0.01)
peaks = find_peaks(k,height=0.2)[0]
print(bootlier_index(peaks,0.01,k))
```

Figure 5: *bootlier.py* (part 3 of 3)

REFERENCES

Chernick, M. R., & LaBudde, R. A. (2011). *An Introduction to Bootstrap Methods with Applications to R* (1st ed.). Wiley.

Davison, A. C. (1997). *Bootstrap Methods and their Application (Cambridge Series in Statistical and Probabilistic Mathematics, Series Number 1)* (1st ed.). Cambridge University Press. https://doi.org/10.1017/CBO9780511802843

Efron, B., & Tibshirani, R. J. (1994). *An Introduction to the Bootstrap.* Taylor & Francis.

Santoyo, S. (2018, June 21). *A Brief Overview of Outlier Detection Techniques - Towards Data Science.* Medium. https://towardsdatascience.com/a-brief-overview-of-outlier-detection-techniques

Singh, Kesar & Xie, Minge. (2003). *Bootlier-Plot: Bootstrap Based Outlier Detection Plot.* Sankhyā: The Indian Journal of Statistics (2003-2007). 65. 532-559. 10.2307/25053287.