**Introduction:**

**Background:**

**Deep learning:**

**The coefficient of a linear regression can be simulated by using a deep learning approach. Before inputting the data set to the model, all data sets should be modified in a proper format to fit the deep learning model. In the deep learning model, multiple layers were set up to analyze and calculate the coefficient. In the processes of simulation, the proper activate function and proper looping echo should be significant. The analyze method “mean\_square\_ error” would be selected. After the model was finished training, the final “mean\_square\_ error” would be the final regression coefficient of the data set.**

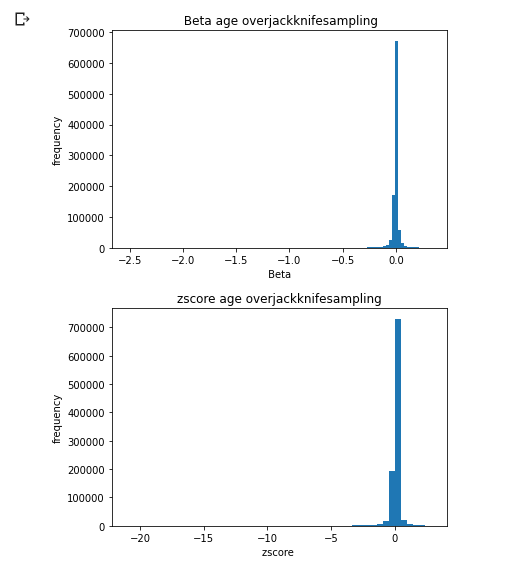
**Methodology:**

**Observation:**

**Dataset one:**

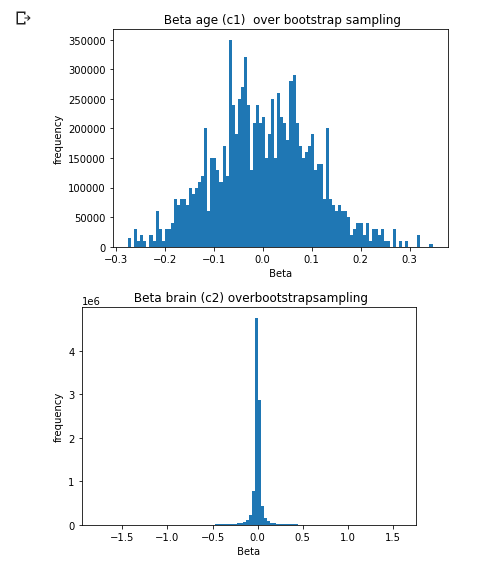
m,age,c = Generate\_X\_y\_c(100,10000)

resample\_time = 1000

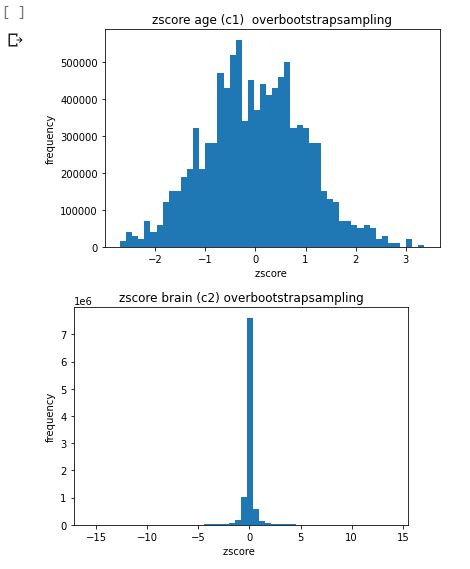


Running time: 14449.94848704338 Seconds

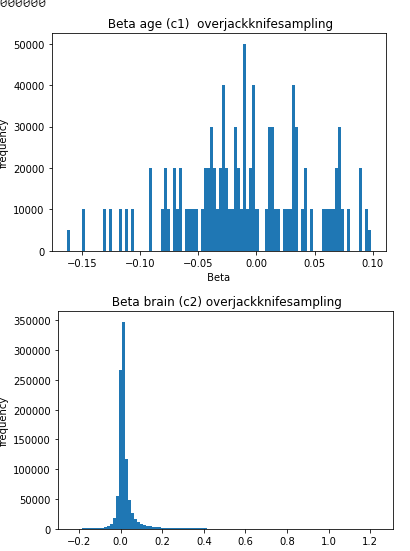
**Observation:**



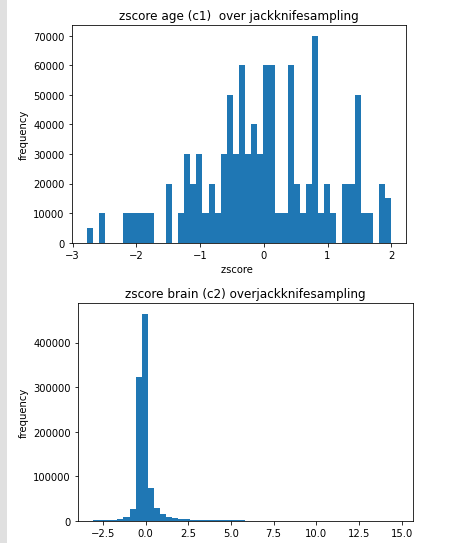
**Observation:**



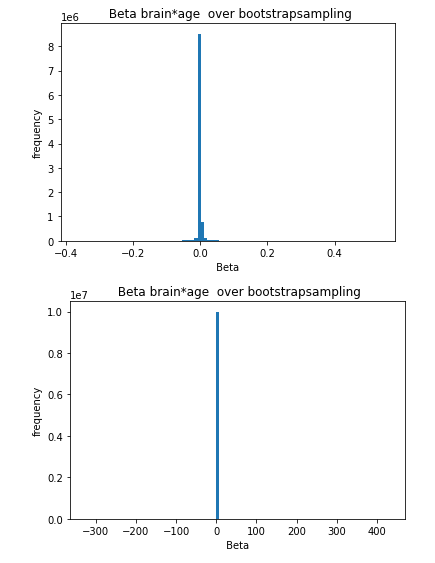
**Observation:**



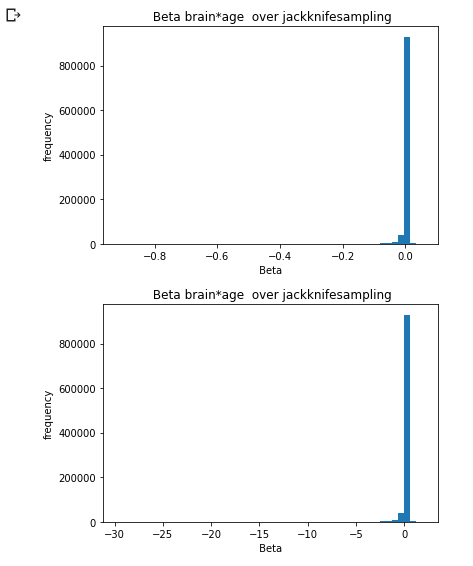
**Observation:**



**Observation:**



**Observation:**



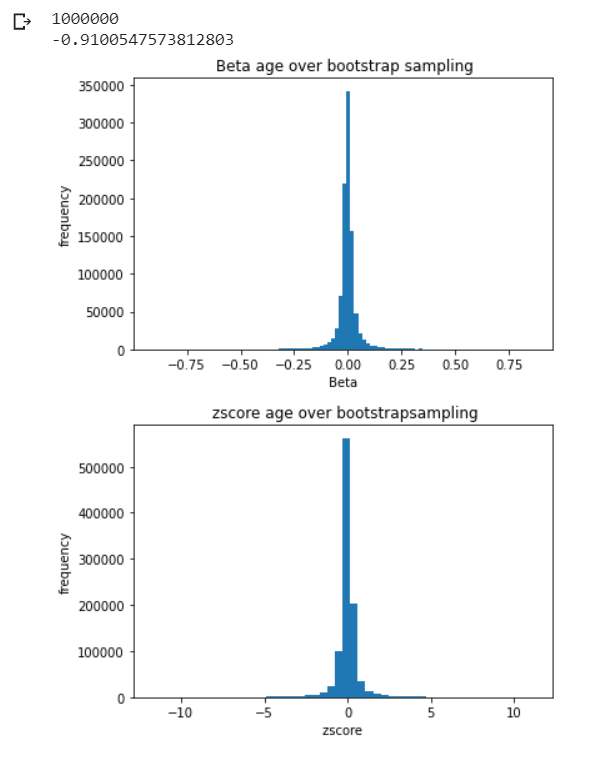
**Observation:**

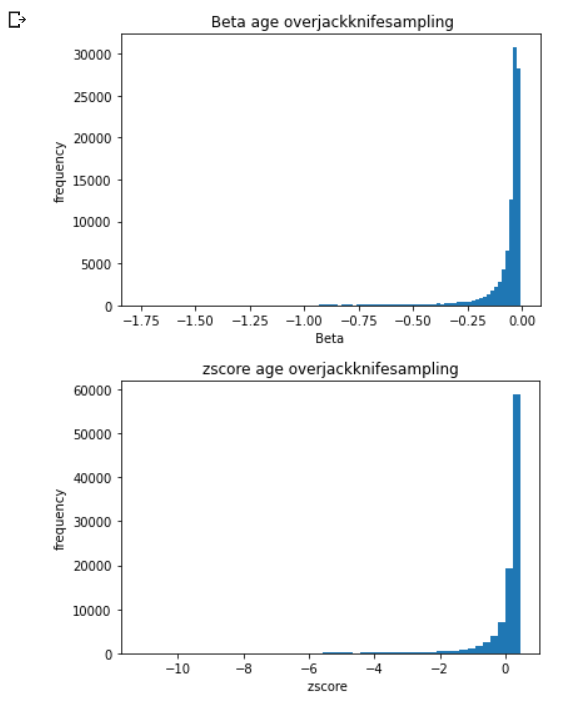
**Dataset two**

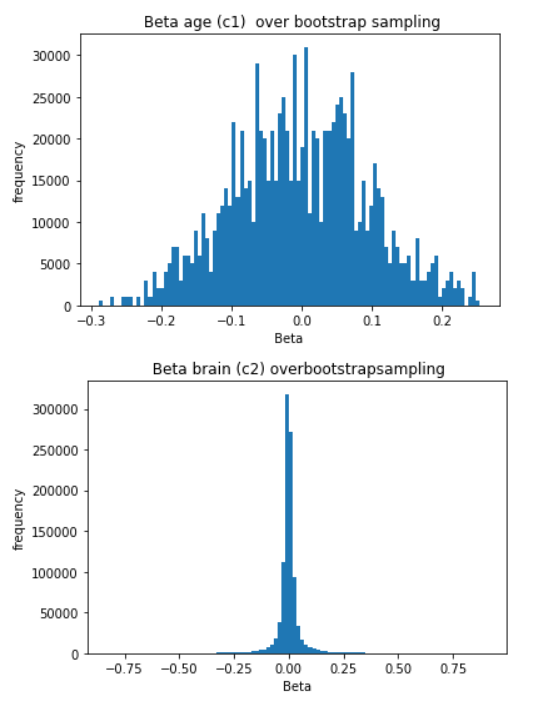
m,age,c = Generate\_X\_y\_c(100,1000)

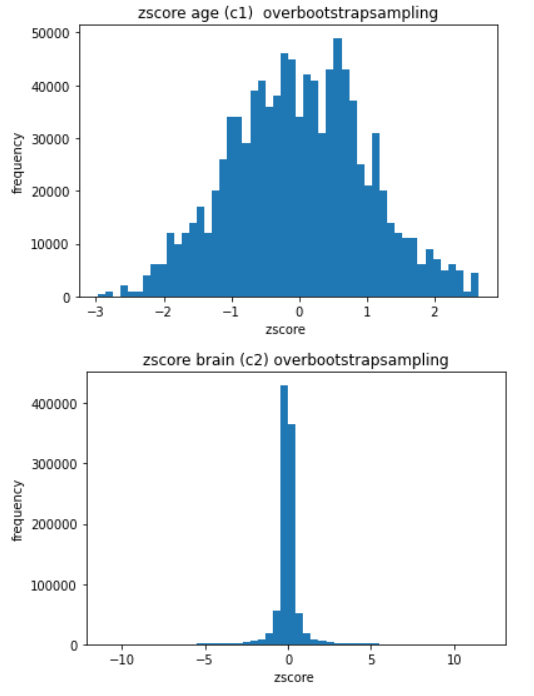
resample\_time = 1000

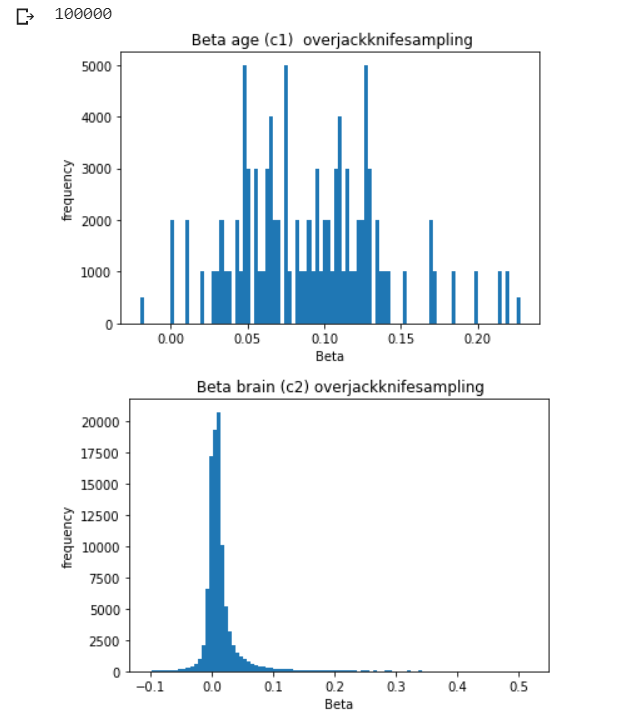
Running time: 1348.4067585468292 Seconds

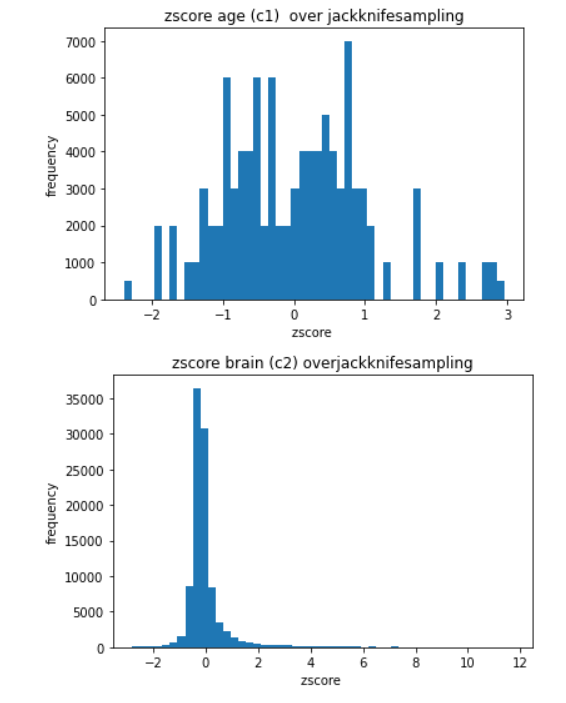


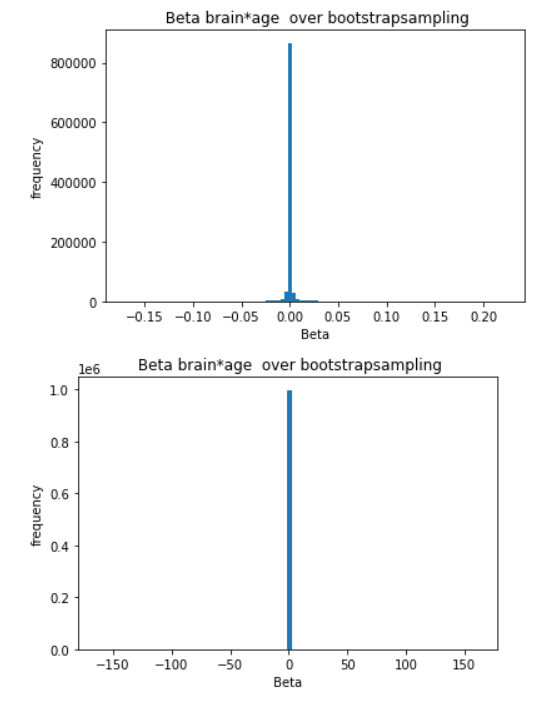


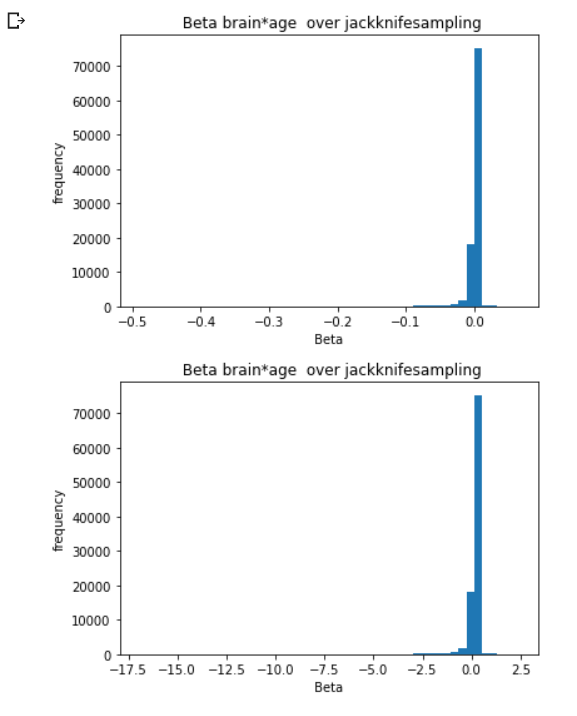


****





****

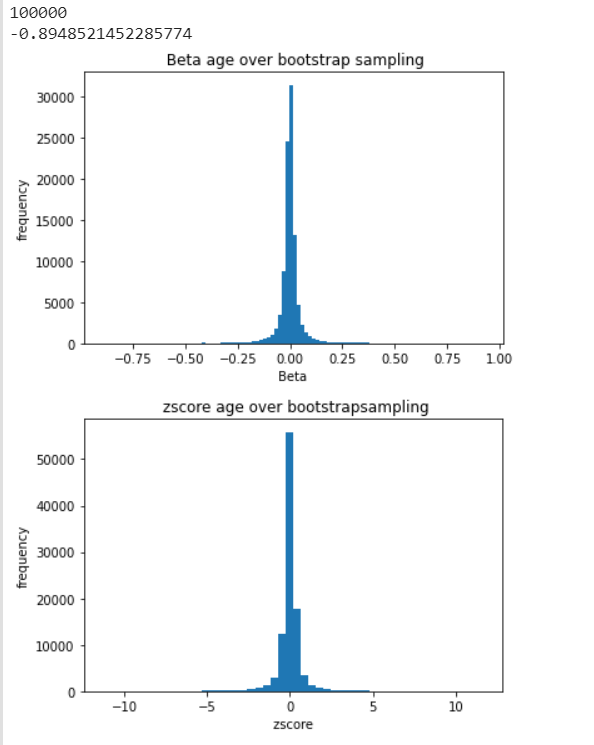
****

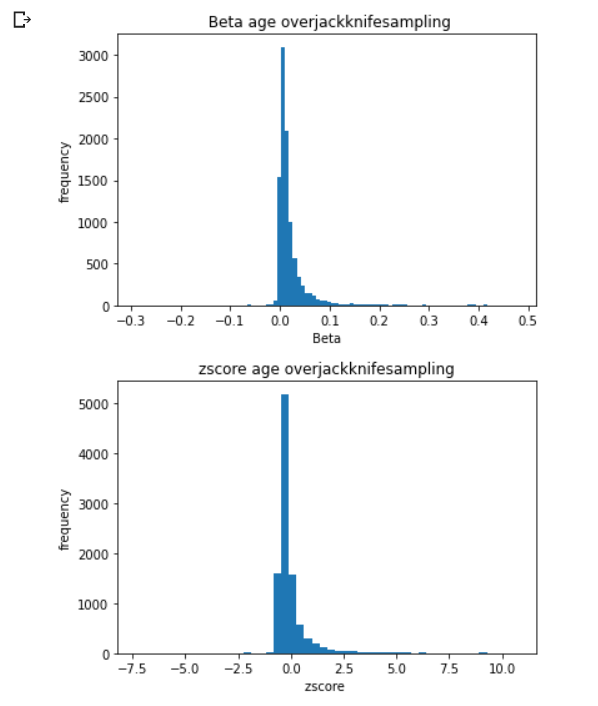
**Dataset three**

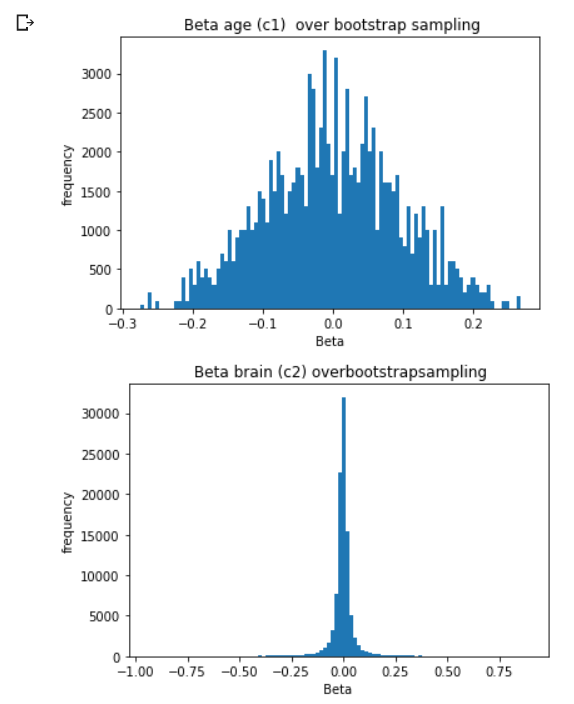
**m,age,c = Generate\_X\_y\_c(100,100)**

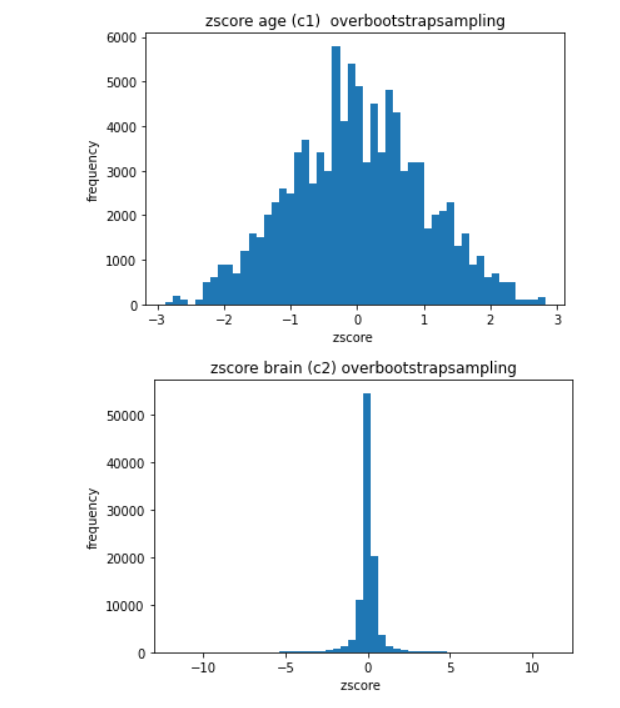
**resample\_time = 1000**

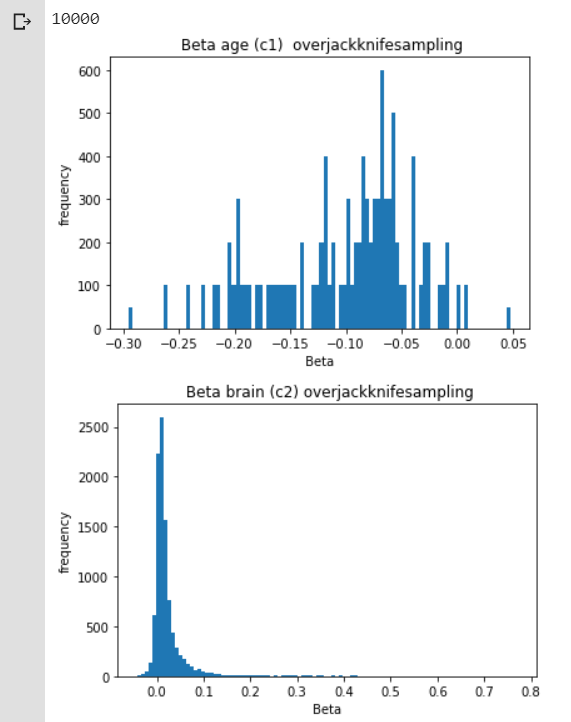
**Running time: 134.0968415737152 Seconds**

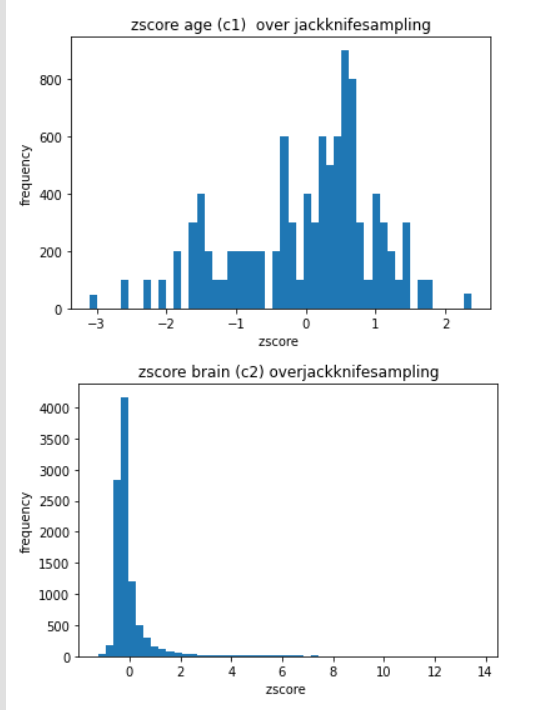


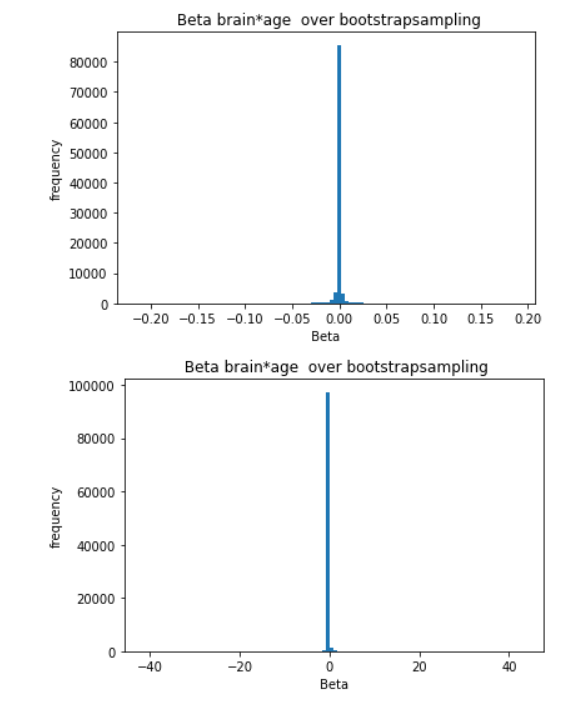


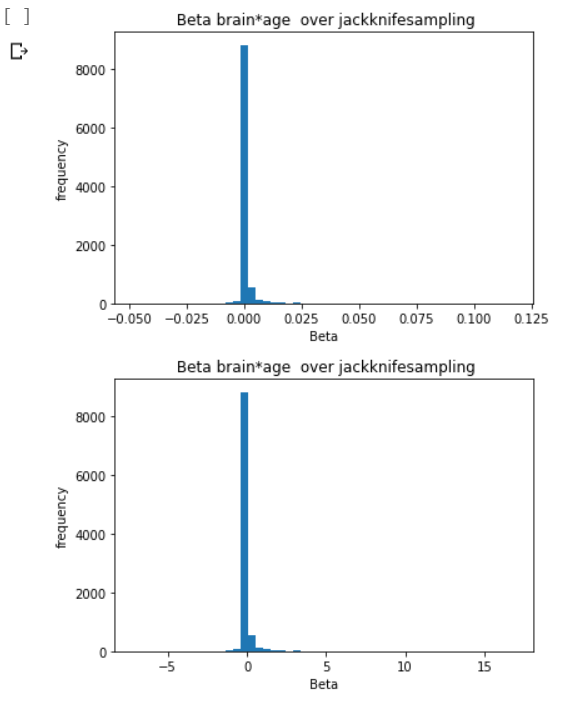












**Bootstrap:**

**Jackknife:**

**BCaCI(Biased Corrected Accelerated Confidence Interval):**

To find the confidence interval, we used the method from the Python library scipy.states. we need to give the percentage of the confidence interval we need, and the data we get from bootstrap and Jackknife. The calculation process is relatively fast, and the result is a tuple that contains two 16 digits numbers such as (a,b). The character a represents the lower bound of the interval and the character b represents the upper bound of the interval.

**Machine learning:**

**SKlearn:**

**numpy:**

**Analysis:**

**Bootstrap:**

**Jackknife:**

[3]The jackknife (or leave one out) method, invented by Quenouille (1949), is an alternative resampling method to the bootstrap.

• The method is based upon sequentially deleting one observation from the dataset, recomputing the estimator, here, , n times. That is, there are exactly n jackknife estimates obtained in a sample of size n.

• Like the bootstrap, the jackknife method provides a relatively easy way to estimate the precision of an estimator, θ.

• The jackknife is generally less computationally intensive than the bootstrap

The advantages of Jackknife is

Useful method for estimating and compensating for

bias in an estimator.

• Like the bootstrap, the methodology does not

require knowledge of the theoretical form of an

estimator’s standard error.

• Is generally less computationally intensive

compared to the bootstrap method.

**BCaCI(Biased Corrected Accelerated Confidence Interval):**

The main advantage to the BCa interval is that it corrects for bias and skewness in the distribution of bootstrap estimates. The BCa interval requires that you estimate two parameters. The bias-correction parameter(also is known as standard deviation), z, is related to the proportion of bootstrap estimates that are less than the observed statistic.

Assume that the data are independent and identically distributed. Suppose that you have already computed the original statistic and a large number of bootstrap estimates, as shown in the previous article. To compute a BCa confidence interval, you estimate z and use it to adjust the endpoints of the percentile confidence interval (CI). If the bootstrap distribution is positively skewed, the CI is adjusted to the right. If the bootstrap distribution is negatively skewed, the CI is adjusted to the left.

**Machine learning:**

**SKlearn:**

**numpy:**

[1]NumPy is the fundamental package for scientific computing with Python. It contains among other things:

· a powerful N-dimensional array object

· sophisticated (broadcasting) functions

· tools for integrating C/C++ and Fortran code

· useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

We use Numpy to calculate the linear regression, the two required functions we have to use are

* [2]numpy (obviously) to do all of the vectorized numerical computations on the dataset including the implementation of the algorithm.
* [2]matplotlib to plot graphs for better understanding the problem at hand with some visual aid.

The best way to install NumPy is going to install the SciPy Stack, and get more information on [scipy.org](https://www.scipy.org/).

The main reason we chose numpy as our linear regression calculation is that for the same numerical calculation task, using NumPy is much more convenient than writing Python code directly;The storage efficiency and input and output performance of arrays in NumPy are far superior to Python's equivalent basic data structures, and the performance that can be improved is proportional to the elements in the array;Most of NumPy's code is written in C, and its underlying algorithms have excellent performance in design, which makes NumPy much more efficient than pure Python code. In addition, the accuracy of numpy is not lost to sklearn of similar plugins.

**Discussion:**

**Conclusion:**

**Reference(IEEE Citation):**

**[1]“NumPy,” *NumPy*. [Online]. Available: https://numpy.org/. [Accessed: 03-Apr-2020].**

**[2]L. Baş “Linear Regression from Scratch with NumPy - Implementation (Finally!),” *Medium*, 05-Aug-2019. [Online]. Available: https://towardsdatascience.com/linear-regression-from-scratch-with-numpy-implementation-finally-8e617d8e274c. [Accessed: 03-Apr-2020].**

**[3]“The Bootstrap and Jackknife,” 2017. [Online]. Available: https://www.biostat.washington.edu/sites/default/files/modules/2017\_sisg\_1\_9\_v3.pdf.**