Analyzing Radioactive Decay

Phsx 815- Project 2

Rakshak Adhikari

March 2021

1 Introduction

This project involves analyzing counts over a fixed time from a radioactive substance, which will be distributed like a Poisson distribution, with the caveat that the parameters themselves are distributed like a Poisson distribution. We then test whether or not it can be distinguished from a Poisson distribution with a different parameter (mu) from which the rate parameters are sampled.

While the code is testing to identify Radioactive substance, the code can be used to test any Poisson Distribution for given lambda values.

2 Algorithm

Simulating the Decay: Simulator.py simulates the radioactive decay using the python standard poisson distribution generator. It takes μ values and internally generates the respecitve λ value using the standard poission distributon and then used the value of λ to generate the observed distribution. The number of measurements is also supplied. For the run, the following values were used:

$$\mu_0 = 50$$
$$\mu_1 = 70$$
$$\lambda_0(\mu_0)$$
$$\lambda_1(\mu_1)$$

no of experiments $=10^6$
no of measurements per experiment= 1

Poisson distribution is characterized by the following equation.

$$P(x|\lambda) = \frac{exp(-\lambda)\lambda^x}{x!} \tag{1}$$

In this project, λ itself is a function of μ so the probability has to be integrated over all values of lambda parameter.

$$P(x|\lambda|\mu) = \frac{exp(-\lambda)\lambda^x}{x!} \int \frac{exp(-\mu)\mu^\lambda}{\lambda!} d\lambda \tag{2}$$

This can be tricky to do as most integrals are. Even then, the closed form of the integral may not exist which is true for this case. Furthermore, it requires that lambda values be known for each data point. But lambda is a hidden parameter in our simulation,so we proceed with a numerical approach. If we run the simulation to get a large number of x values, the number of counts for each value of In the DecaySimulator, μ_0 and $\mu_1$are used for null and alternative hypothesis respectively. The code outputs the results in textfile.

Analyzing the Output: analysis.py reads the output textfile produced by simulator.py and calculates the Likelihood for each of the parameter values and also plots the histogram.

For numerical calculation of the likelihood we simply make bins for each count and plot histogram of frequency of each count for each of the two mu values. Once we normalize the data, it is a measure of the probability and we get our likelihood. This is simplified by the fact that for each experiment, we have one measurement and there is a one-to-one mapping between the likelihood and the counts.

Λ_{crit} is calculated by sorting the array of the distribution and finding the x value for which (1-α) percent of the values lied to the left of the x value. Beta value can be calculated similarly.
For the sorting needs, python's built in sorting algorithm was utilized.

3 Result

The distribution of the results from the Simulator is plotted in figure 1.

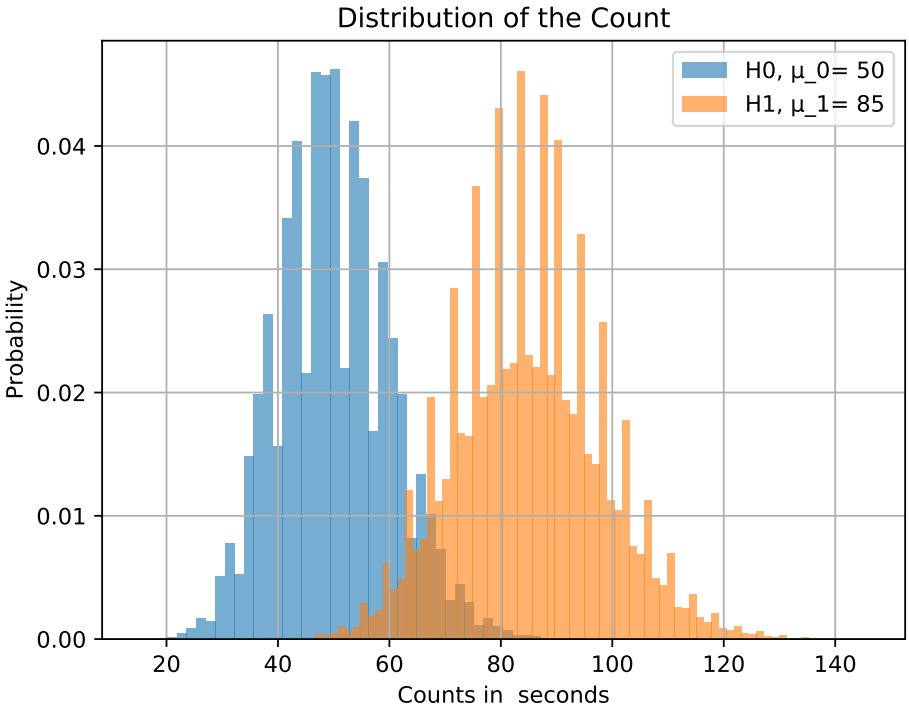

Figure 1: Normalized Distribution

Since we are only taking one measurement per experiment, we can use Likelihood curves for each of the lambda values as shown below instead of the Log-Likelihood plot. We chose our alpha to be 0.05.

As seen from the plot, for alpha of 0.05, we get Lambda critical of 67 and beta of 0.084.

4 Conclusion

We can conclude from our analysis that if $x < 62 = \Lambda_{critical}$, we can reject our Alternative Hypothesis in favour of the Null Hypothesis with a confidence level of 95%.
We have sucessfully identified the radioactive substance.
Similar tests can be done to determine whether a set of values follow a Poisson

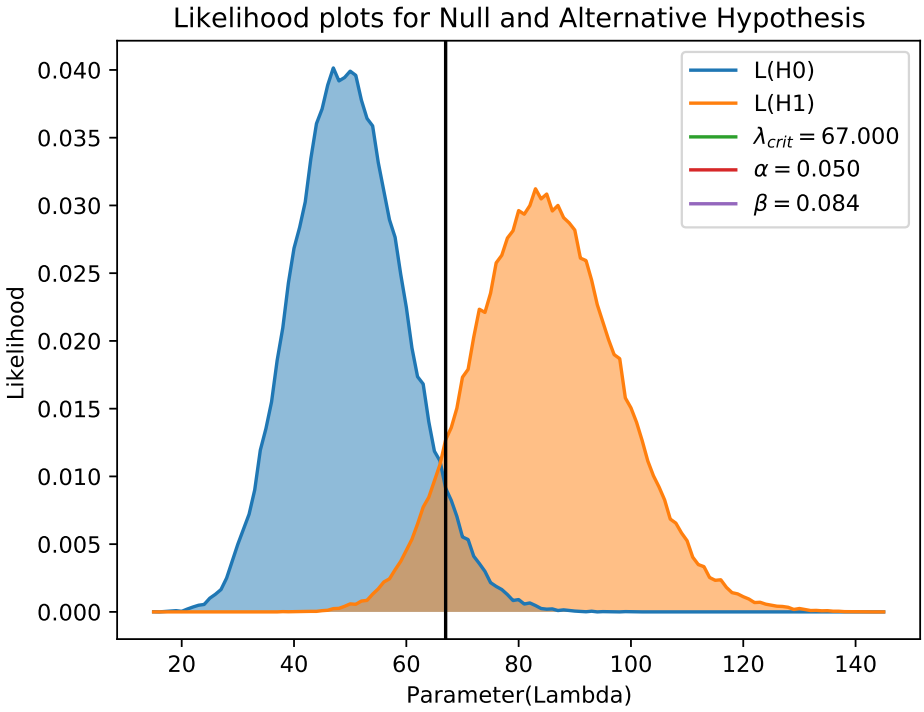

Figure 2: Likelihood Plot

distribution where the parameter itself is a poisson distribution with a known rate paramter.

References

[1] `https://github.com/Raxxak/PHSX815_Project2/blob/main/simulator.py`

[2] `https://github.com/Raxxak/PHSX815_Project2/blob/main/analysis.py`