Statistical Methods in AI

PizzaPizzaPizza Report

Parameter Estimation

1) Introduction

1.1) Dataset

We have used two datasets for parameter estimation:

1. **Google Hangouts Chat Dataset**
2. **Covid World-wide Day-wise Cases Dataset**

1.2) Approaches

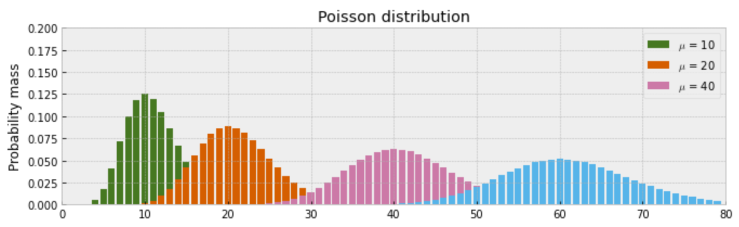
We have used two approaches for parameter estimation:

1. **Frequentist approach**
2. **Bayesian approach**

In both of these approaches we are assuming model as Poisson distribution as it is always > 0 and therefore trying to estimate the parameter (μ) which is both mean and variance of distribution.

$$P(x \mid \mu) = \frac{e^{-\lambda}\mu^x}{x!} \qquad \forall \qquad x \in \{1, 2, 3, \cdots\}$$

$$\lambda = E(x) = Var(x)$$



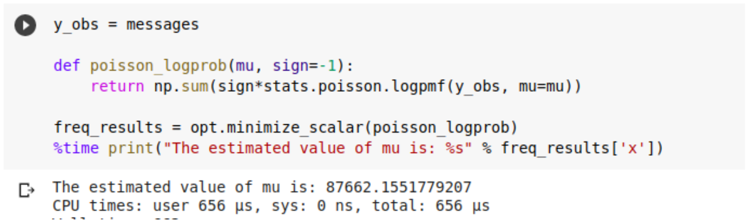
- In first dataset which is Google Hangout chats, we are trying to estimate the response time of sending message.
- In second dataset which is Covid day-wise cases dataset, we are trying to estimate the number of positive cases per day.

2) Frequentist method

Now the frequentist approach is slightly different than bayesian approach. In frequentist approach we try to maximise likelihood without taking prior into consideration i.e.

$$p(\text{data} \mid \mu)$$

So in frequentist method we will use an optimization technique that aims to maximize the likelihood of a function.



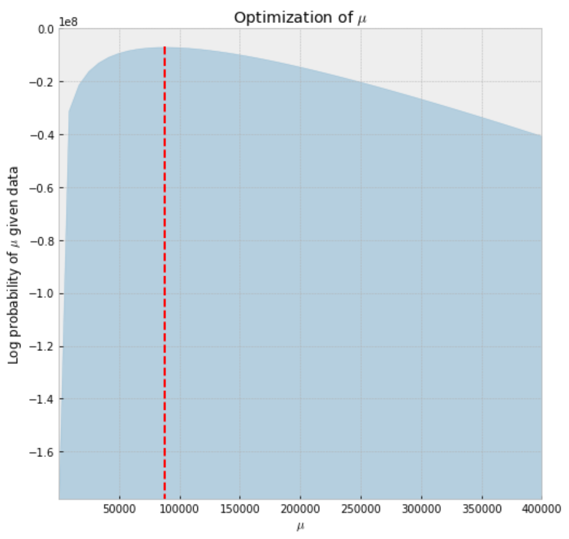
```
y_obs = messages

def poisson_logprob(mu, sign=-1):
    return np.sum(sign*stats.poisson.logpmf(y_obs, mu=mu))

freq_results = opt.minimize_scalar(poisson_logprob)
%time print("The estimated value of mu is: %s" % freq_results['x'])
```

```
The estimated value of mu is: 87662.1551779207
CPU times: user 656 µs, sys: 0 ns, total: 656 µs
```

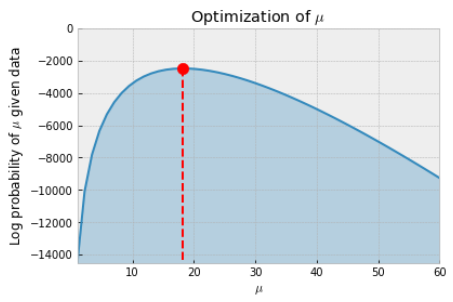
The above function `poisson_logprob()` returns the overall likelihood of the observed data given a Poisson model and parameter value. We use the method opt.minimize_scalar to find the value of μ that is most credible (maximizes the log likelihood) given the data observed. Under the hood, this optimization technique is intelligently iterating through possible values of mu until it finds a value with the highest likelihood.

The optimization technique doesn't provide any measure of uncertainty - it just returns a point value.

2.0.1) for covid dataset



2.0.2) for hangout dataset



3) Bayesian Method

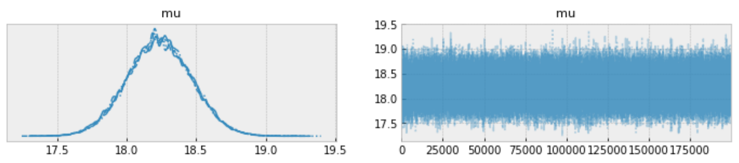
3.1) MCMC

The Markov Chain Monte Carlo method (MCMC) utilizes the Bayes Formula / Theorem to estimate μ.

The MCMC sampler picks up values from the prior distribution and computes the likelihood that the observed data came from a distribution with these parameter values.

$$\overbrace{p(\mu \mid Data)}^{posterior} \propto \overbrace{p(Data \mid \mu)}^{likelihood} \cdot \overbrace{p(\mu)}^{prior}$$

This formula is used to guide the sampler towards the higher probability areas.



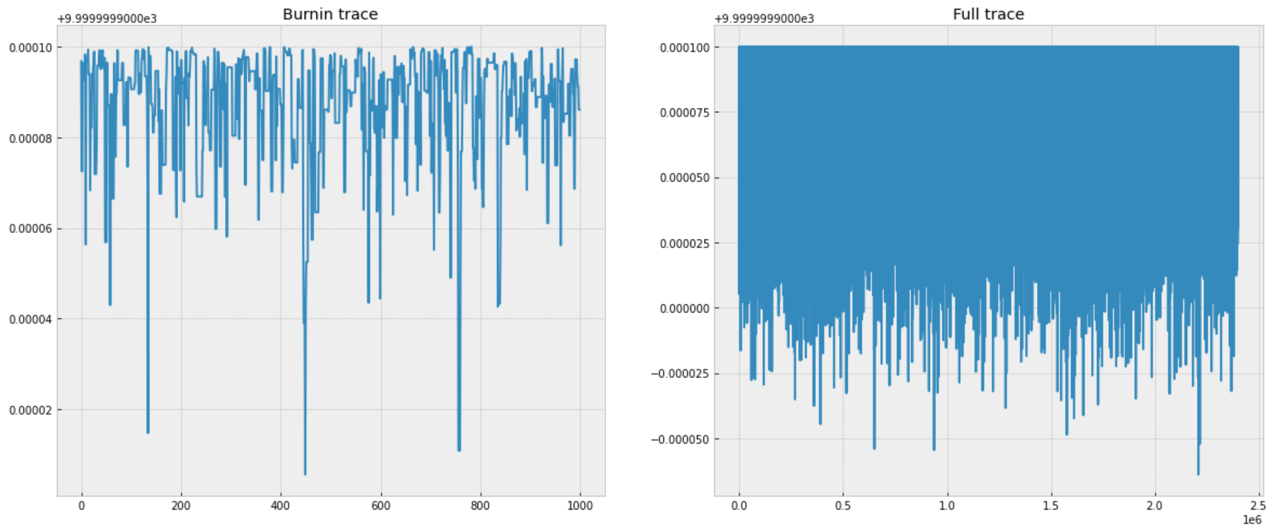
3.2) Burning

The parameters are initialized using the `pm.find_MAP()` function which stands for find maximum a posteriori estimation. It helps the MCMC sampler find a good place from which to start sampling. Ideally this will start the model off in an area of high likelihood - but sometimes that doesn't happen. As a result, the samples collected early in the trace (burnin samples) are often discarded.

3.3) Convergence

3.3.1) Trace

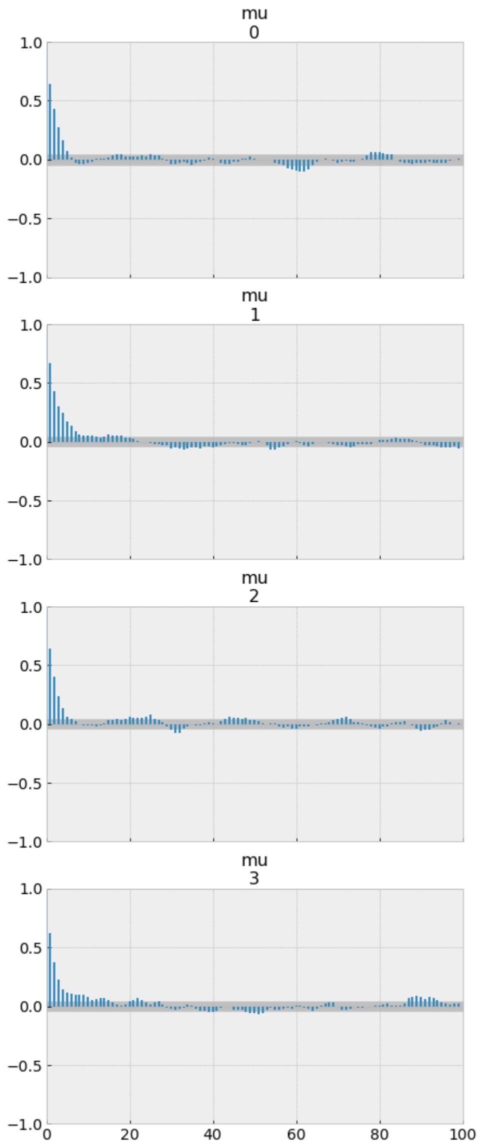
Just because the above model estimated a value for μ, doesn't mean the model estimated a good value given the data. There are some checks that we can make. Firstly, we can look at the trace output. We see the trace jumping around and generally looking like a hairy caterpillar. If we see the trace snake up and down or appear to be stuck in any one location - it is a sign that you have convergence issues and the estimations from the MCMC sampler cannot be trusted.



3.3.2) AutoCorrelation

The second test we can perform is the autocorrelation test. It is a measure of correlation between successive samples in the MCMC sampling chain. When samples have low correlation with each other, they are adding more "information" to the estimate of the parameter value than samples that are highly correlated.

Visually, we are looking for an autocorrelation plot that tapers off to zero relatively quickly and then oscillates above and below zero correlation. If the autocorrelation plot does not taper off - it is generally a sign of poor mixing and we should revisit our model selection (e.g. likelihood) and sampling methods (e.g. Metropolis).


mu
0
mu
1
mu
2
mu
3
1.0
0.5
0.0
−0.5
−1.0
0
20
40
60
80
100