Transfer entropy for population data.

Conor Houghton

U Bristol

Edinburgh, June 2022

Bayes rule

$$p_{X,Y}(x, y) = p_{X|Y}(x|y)p_Y(Y)$$

and

$$p_{X,Y}(x, y) = p_{Y|X}(y|x)p_X(x)$$

so

$$p_{X|Y}(x|y) = \frac{p_{Y|X}(y|x)p_X(x)}{p_Y(y)}$$

Bayesian inference

Bayesian inference applies Bayes rule to a model

- We have a **model** parameterized with some **parameters** θ.
- We have some **data**.

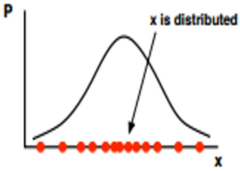
  Bayesian inference tells us what we know about the parameters given the data.

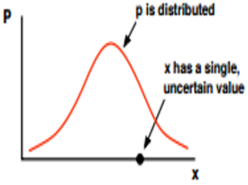
What does probability model?

The laws of probability are mathematics - the mathematics don't determine what probabilities model.

In Bayesian inference probabilities describe our knowledge.

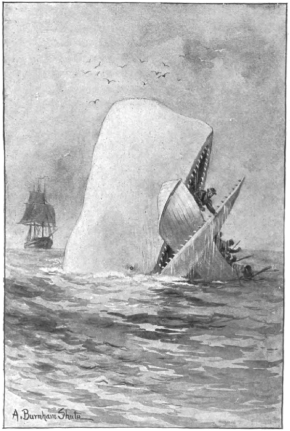
Frequentists versus Bayesians





Picture from Rosalyn Moran

Guess the missing letter

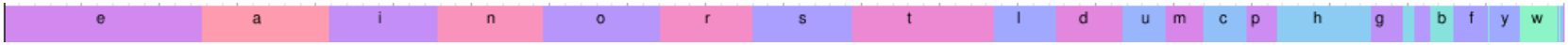


"Both jaws, like enormous shears, bit the craft completely in twain."

—Page 510.

CALL ME ISHMAE*

Guess the missing letter - letter frequencies



CALL ME ISHMAE*

with for example $p(E) = 0.13$, $p(L) = 0.04$ and $p(Q) = 0.001$.

Letter frequency table from Wikipedia

Guess the missing letter - 2-gram letter frequencies

Look at two letter pairs, *ER* and *EL* and so on, and work out conditional probabilities like

$$p(\text{second letter is L}|\text{first letter is E})$$

CALL ME ISHMAE*

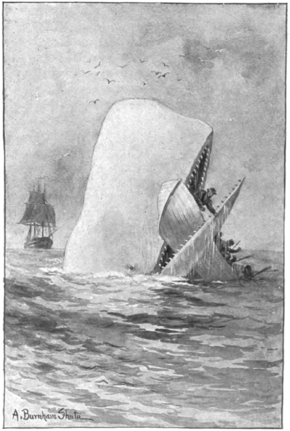
with for example $p(R) = 0.14$, $p(L) = 0.04$ and $p(Q) = 0.003$.

Guess the missing letter - 3-gram letter frequencies

CALL ME ISHMAE*

with for example $p(R) = 0.1$, $p(L) = 0.4$ and $p(P) = 0.001$.

Guess the missing letter



"Both jaws, like enormous shears, bit the craft completely in twain."

—*Page 510.*

CALL ME ISHMAEL

Bayesian inference

We have a **model** parameterized with some **parameters** say θ.

We have **priors**, some knowledge of θ described using probability distributions: $p(\theta)$.

We have some **data** points x_i.

We can calculate $p(\text{data}|\text{model})$, that is $p(\mathbf{x}_i|\theta)$.

Bayesian inference is about calculating $p(\text{model}|\text{data})$ or $p(\theta|\mathbf{x}_i)$ using Bayes rule.

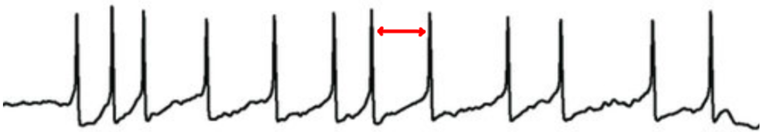
Bayesian inference

$$p(\text{model}|\text{data}) = \frac{p(\text{data}|\text{model})p(\text{model})}{p(\text{data})}$$

or

$$p(\theta|\mathbf{x}_i) = \frac{p(\mathbf{x}_i|\theta)p(\theta)}{p(\mathbf{x}_i)}$$

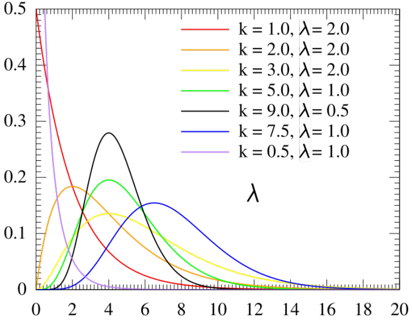
Example: interspike intervals



Interspike intervals: model

$$t \sim \text{Gamma}(k, \lambda)$$

where λ is a scale parameter and k is a shape parameter: $\theta = (\lambda, k)$.



Interspike intervals: data

For this example we will use the a spike train recorded from blowfly viewing a visual stimulus:

`www.gatsby.ucl.ac.uk/ dayan/book/exercises.html`

Interspike intervals: priors

Working out the mean and variance gives us some idea what the priors should be!

$$\mu = k\lambda$$

and

$$\sigma^2 = k\lambda^2$$

so, this gives us

$$\lambda \approx 0.027 \text{ s}$$

and

$$k \approx 0.30$$

We also know these parameters need to be positive so we will an exponential distribution.

Model and priors

Priors

$$\begin{aligned} \lambda &\sim \text{Exp}(0.027) \\ k &\sim \text{Exp}(0.3) \end{aligned}$$

and data

$$t_i \sim \text{Gamma}(k, \lambda)$$

Posteriors

Now all we need to do is work out the *posterior*: $p(\lambda, k|t_i)$.

In principle this is just Bayes rule

$$p(\lambda, k|t_i) = \frac{p(t_i|\lambda, k)p(\lambda, k)}{p(h_i)}$$

However, in practice we can't usually can't do this calculation, calculating $p(h_i)$ involves doing an integral we can't do!

Posteriors

We need to work out the *posterior*: $p(\lambda, k|t_i)$.

In practice we calculate the posterior using **magic**. The name of this magic is the *No-U Turn Sampler* or *NUTS*.

NUTS is one of the things that makes Bayesian inference possible and practical all of a sudden. It is important to understand it and to develop a feel for when it will work well and when small changes to an approach will make it work better. Here we will treat it as **magic**.

NUTS

A number of languages or libraries allow you to use NUTS; most obviously `STAN`, `turing.jl` and `PyMC3`.

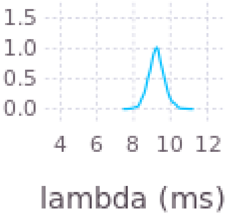
Here we will look at some `turing.jl` code as a sort of pseudocode

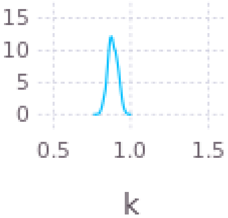
Some code

```
@model function gammaModel(isi)
      p=parameters(isi)
      meanL=p[1]
      meanK=p[2]

      l ~ Exponential(meanL)
      k ~ Exponential(meanK)

      for i in 1:length(isi)
          isi[i] ~ Gamma(k,l)
      end

end

chain = sample(gammaModel(isi), NUTS(), 1000)
```

Results

Using 10 s of data; that is 1221 ISIs we get:





with $\bar{\lambda} = 0.0093$ and $\bar{k} = 0.8861$.

What?

$\bar{\lambda} = 0.0093$ and $\bar{k} = 0.8861$.

but didn't we say

$$\lambda \approx 0.027 \text{ s}$$

and

$$k \approx 0.30$$

before, by calculating the empirical mean and variance?

Optimizing the log-pdf

$\bar{\lambda} = 0.0093$ and $\bar{k} = 0.8861$.

and by optimizing the log-pdf

$$L = \sum_i \log p(\delta t_i | \lambda, k)$$

over λ, k. This gives

$$\lambda \approx 0.092 \text{ s}$$

and

$$k \approx 0.89$$

which gives a similar answer!

Results - what does Bayes buy you?

Why not just optimize the log-pdf!

With Bayes we know what we know!

Results - what does Bayes buy you?

Using 1 s of data; that is 146 ISIs we get: