Bayes assignment on regression lines

Sean van der Merwe

2022/05/05

# Currently marking student St2012345678

# Instructions

You are provided with a set of data from an anonymous survey where people with desktop computers were asked how many cookies were set on their favourite browser. Additionally, information was obtained on their internet privacy habits. Your goal is to determine the relationships between the number of cookies and the habits reported.

Specifically, the questions were:

1. What is the number of cookies reported by your favourite browser?
2. How many times a year do you clear your browser cookies and/or history?
3. Do you use a virtual private network (VPN) regularly when connecting to the internet?

## Part 1: Visualisation [15]

Read in and visualise your data. Give summary statistics overall and by VPN group, as well as a plot where the groups can be identified. Discuss any apparent patterns without making any firm conclusions yet.

## Part 2: First model fit [30]

A Negative Binomial GLM formulation (with log link function) is recommended as a starting point. Fit such a model and show the model fit on a plot along with the data to assess the fit. You must show uncertainty via prediction intervals at least.

In case it isn’t clear from the data itself: the dependent variable is the raw number of cookies; while the explanatory variables should be the square root of the number of clears, as well as a binary indicator of VPN use.

## Part 3: Changing variance [10]

There are good fundamental reasons to assume that VPN users would vary more in terms of number of cookies (after adjusting for the other factor measured). Calculate a reasonable posterior probability that this is the case (this may require adjusting your model to allow for different variances).

## Part 4: Model comparison [25]

Ensure that you have fitted a model that differentiates between VPN and non-VPN users, and one that does not (more than these two is fine, less is not). Compare all your models using at least one good criterion and say which model seems the most parsimonious.

## Part 5: Final step [20]

Pretend you answered the survey. What is the predicted number of cookies for you personally, according to the model you think is best? Where does the actual number (as reported by your favourite browser) lie on the predicted distribution?

# Generate samples that are different for each student  
# This is the code used to generate the data, for interest only.  
  
library(openxlsx)  
students <- c('Sipho','Nhlanhla','Malethena','Tebello','Sandisile','Mari','Palesa','Edrich','Sihle','Lwazilwenkosi','Katlego','Walena','Jan','Bereng','St2012345678','St2123456789','St9876543210')  
nn <- length(students)  
n <- 120:(119+nn)  
datasets <- vector('list',nn)  
Treatments <- c('noVPN', 'VPN')  
  
for (i in 1:nn) {  
 Tr <- sample(1:2, n[i], T)  
 Gr <- Treatments[Tr]  
 cept <- runif(1, 6.5, 7)  
 thetas <- Tr  
 slope <- runif(1, -1, -0.8)  
 numClears <- MASS::rnegbin(n[i], 6, 0.2)  
 (cept + slope\*sqrt(numClears)) |> exp() -> mu  
 y <- MASS::rnegbin(n[i], mu, thetas)  
 datasets[[i]] <- data.frame(ID=paste0('Resp', seq\_len(n[i])), VPNuse=Gr, TrNum=Tr, NumClears = numClears, Cookies = y)  
}  
names(datasets) <- students  
write.xlsx(datasets, file = "BayesAssignmentCookies.xlsx", overwrite = TRUE)

# Memorandum

library(tidyverse)  
library(parallel)  
library(rstan)  
mycores <- max(1,floor(detectCores(logical = FALSE)\*0.75))  
options(mc.cores = mycores)  
rstan\_options(auto\_write = TRUE)

## Part 1

'BayesAssignmentCookies.xlsx' |> openxlsx::read.xlsx(st) -> d  
names(d)

## [1] "ID" "VPNuse" "TrNum" "NumClears" "Cookies"

d |> ggplot(aes(x = NumClears, y = Cookies, colour = VPNuse)) + geom\_point()



d |> summarise(  
 n = length(Cookies),  
 Average = mean(Cookies),  
 StdDev = sd(Cookies),  
 Median = median(Cookies),  
 LQ = quantile(Cookies, 0.25),  
 UQ = quantile(Cookies, 0.75),  
 Min = min(Cookies),  
 Max = max(Cookies)  
) |> kable(digits = 3)

| n | Average | StdDev | Median | LQ | UQ | Min | Max |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 134 | 674.537 | 855.77 | 295.5 | 63.75 | 1124 | 0 | 4563 |

d |> group\_by(VPNuse) |> summarise(  
 n = length(Cookies),  
 Average = mean(Cookies),  
 StdDev = sd(Cookies),  
 Median = median(Cookies),  
 LQ = quantile(Cookies, 0.25),  
 UQ = quantile(Cookies, 0.75),  
 Min = min(Cookies),  
 Max = max(Cookies)  
) |> kable(digits = 3)

| VPNuse | n | Average | StdDev | Median | LQ | UQ | Min | Max |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| noVPN | 76 | 594.697 | 832.712 | 275.5 | 44.75 | 928.0 | 0 | 4563 |
| VPN | 58 | 779.155 | 881.354 | 344.0 | 87.75 | 1357.5 | 0 | 3876 |

**|| 2 Marks for reading in the data correctly, 5 for visualising it, 5 for summary statistics, and 3 for saying what is visible (not much besides a general negative relationship between clearing cookies and how many are left). ||**

## Part 2

d$sqrtClears <- sqrt(d$NumClears)  
fits <- vector('list', 4)

### Model definition

We define a flexible model that can accommodate all forms we wish to fit:

// This Stan block defines a Negative Binomial model with groups, by Sean van der Merwe, UFS  
data {  
 int<lower=1> n; // number of observations  
 int<lower=0> y[n]; // observations  
 int<lower=1> ng; // number of groups  
 int<lower=1, upper=ng> g[n]; // group membership  
 int<lower=1> nv; // number of variance groups  
 int<lower=1, upper=nv> v[n]; // variance group membership  
 real x[n]; // explanatory variable  
}  
// The parameters of the model  
parameters {  
 real alpha[ng]; // group intercepts  
 real beta[ng]; // group slopes  
 real<lower=0> phi[nv]; // group variances  
}  
transformed parameters {  
 vector[n] etas;  
 vector[n] phis;  
 for (i in 1:n) {  
 etas[i] = alpha[g[i]] + beta[g[i]]\*x[i];  
 phis[i] = phi[v[i]];  
 }  
}  
model {  
 y ~ neg\_binomial\_2\_log(etas, phis); // likelihood  
}  
generated quantities {  
 vector[n] log\_lik;  
 for (i in 1:n) {  
 log\_lik[i] = neg\_binomial\_2\_log\_lpmf(y[i] | etas[i], phis[i]);  
 }  
}

saveRDS(NegBinGrouped, file = 'NegBinGrouped.Rds')

### Fit 1: two lines, one variance

stan\_data <- list(n = nrow(d), y = d$Cookies, x = d$sqrtClears, ng=max(d$TrNum), g=d$TrNum, nv = 1, v = rep(1, nrow(d)))  
fits[[1]] <- sampling(NegBinGrouped, stan\_data, iter = 20000, chains = mycores)

draws <- extract(fits[[1]])  
kable(round(summary(fits[[1]])$summary[1:5,],3))

|  | mean | se\_mean | sd | 2.5% | 25% | 50% | 75% | 97.5% | n\_eff | Rhat |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| alpha[1] | 7.135 | 0.001 | 0.132 | 6.885 | 7.044 | 7.132 | 7.222 | 7.403 | 30302.60 | 1 |
| alpha[2] | 7.200 | 0.001 | 0.141 | 6.932 | 7.104 | 7.197 | 7.292 | 7.484 | 34241.12 | 1 |
| beta[1] | -1.101 | 0.000 | 0.067 | -1.233 | -1.145 | -1.101 | -1.056 | -0.971 | 30470.69 | 1 |
| beta[2] | -1.007 | 0.000 | 0.067 | -1.138 | -1.052 | -1.008 | -0.963 | -0.875 | 34247.44 | 1 |
| phi[1] | 1.441 | 0.001 | 0.171 | 1.131 | 1.323 | 1.433 | 1.551 | 1.797 | 38779.25 | 1 |

### Generate and plot predictions

It is usually better to generate new data for making predictions, like this

new\_d <- expand.grid(NumClears = seq(min(d$NumClears), max(d$NumClears), l = 100), VPNuse = unique(d$VPNuse))  
new\_d$TrNum <- (new\_d$VPNuse == "VPN") + 1  
new\_d$sqrtClears <- sqrt(new\_d$NumClears)  
new\_n <- nrow(new\_d)

Then we make a matrix of predictions of size (simulations by observations)

nsims <- nrow(draws$alpha)  
new\_n |> seq\_len() |> sapply(\(i) {  
 CookiesMu <- exp(draws$beta[,new\_d$TrNum[i]] \* new\_d$sqrtClears[i] + draws$alpha[,new\_d$TrNum[i]])  
 MASS::rnegbin(nsims, CookiesMu, draws$phi[,1])  
}) -> pred\_cookies

Then we calculate the statistics to plot

pred\_cookies |> apply(2, \(x) {  
 c(mean(x), quantile(x, c(0.025, 0.975)))  
}) |> t() -> pred\_stats  
colnames(pred\_stats) <- c('Average', 'Lower', 'Upper')  
data.frame(new\_d, pred\_stats) |>   
 pivot\_longer(Average:Upper, names\_to = 'Line', values\_to = 'Cookies') -> plot\_data

and plot them

d |> ggplot(aes(x = NumClears, y = Cookies, colour = VPNuse)) + geom\_point() + geom\_line(aes(lty = Line), data = plot\_data)



**|| 15 marks for the model and running it correctly, and 15 for plotting predictions with uncertainty. ||**

## Part 3

### Fit 2: two lines, two variances

stan\_data <- list(n = nrow(d), y = d$Cookies, x = d$sqrtClears, ng=max(d$TrNum), g=d$TrNum, nv = max(d$TrNum), v = d$TrNum)  
fits[[2]] <- sampling(NegBinGrouped, stan\_data, iter = 20000, chains = mycores)

draws <- extract(fits[[2]])  
kable(round(summary(fits[[2]])$summary[1:6,],3))

|  | mean | se\_mean | sd | 2.5% | 25% | 50% | 75% | 97.5% | n\_eff | Rhat |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| alpha[1] | 7.143 | 0.001 | 0.154 | 6.850 | 7.039 | 7.140 | 7.244 | 7.459 | 33265.01 | 1 |
| alpha[2] | 7.197 | 0.001 | 0.109 | 6.989 | 7.123 | 7.195 | 7.269 | 7.414 | 36120.17 | 1 |
| beta[1] | -1.108 | 0.000 | 0.076 | -1.260 | -1.158 | -1.108 | -1.057 | -0.960 | 33211.83 | 1 |
| beta[2] | -1.008 | 0.000 | 0.054 | -1.113 | -1.044 | -1.008 | -0.972 | -0.902 | 35618.75 | 1 |
| phi[1] | 1.102 | 0.001 | 0.174 | 0.791 | 0.981 | 1.091 | 1.212 | 1.473 | 48747.53 | 1 |
| phi[2] | 2.413 | 0.002 | 0.434 | 1.647 | 2.108 | 2.386 | 2.690 | 3.332 | 46255.95 | 1 |

The probability that is roughly

mean(draws$phi[,2] > draws$phi[,1])

## [1] 0.99945

which is very high, supporting our assumption.

**|| 8 marks for correctly bringing in the second variance and 2 marks for reporting the probability asked. ||**

## Part 4

### Fit 3: one line, one variance

stan\_data <- list(n = nrow(d), y = d$Cookies, x = d$sqrtClears, ng = 1, g = rep(1, nrow(d)), nv = 1, v = rep(1, nrow(d)))  
fits[[3]] <- sampling(NegBinGrouped, stan\_data, iter = 20000, chains = mycores)

draws <- extract(fits[[3]])  
kable(round(summary(fits[[3]])$summary[1:4,],3))

|  | mean | se\_mean | sd | 2.5% | 25% | 50% | 75% | 97.5% | n\_eff | Rhat |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| alpha[1] | 7.153 | 0.001 | 0.095 | 6.969 | 7.088 | 7.151 | 7.217 | 7.344 | 25727.37 | 1 |
| beta[1] | -1.053 | 0.000 | 0.046 | -1.144 | -1.084 | -1.053 | -1.022 | -0.963 | 25081.69 | 1 |
| phi[1] | 1.447 | 0.001 | 0.172 | 1.133 | 1.327 | 1.438 | 1.557 | 1.806 | 26637.25 | 1 |
| etas[1] | 7.153 | 0.001 | 0.095 | 6.969 | 7.088 | 7.151 | 7.217 | 7.344 | 25727.37 | 1 |

### Fit 4: one line, two variances

stan\_data <- list(n = nrow(d), y = d$Cookies, x = d$sqrtClears, ng=1, g=rep(1, nrow(d)), nv = max(d$TrNum), v = d$TrNum)  
fits[[4]] <- sampling(NegBinGrouped, stan\_data, iter = 20000, chains = mycores)

draws <- extract(fits[[4]])  
kable(round(summary(fits[[4]])$summary[1:5,],3))

|  | mean | se\_mean | sd | 2.5% | 25% | 50% | 75% | 97.5% | n\_eff | Rhat |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| alpha[1] | 7.170 | 0.000 | 0.088 | 7.002 | 7.111 | 7.168 | 7.228 | 7.344 | 32841.47 | 1 |
| beta[1] | -1.043 | 0.000 | 0.043 | -1.128 | -1.071 | -1.043 | -1.014 | -0.959 | 32853.65 | 1 |
| phi[1] | 1.107 | 0.001 | 0.173 | 0.799 | 0.986 | 1.097 | 1.217 | 1.475 | 38840.38 | 1 |
| phi[2] | 2.412 | 0.002 | 0.438 | 1.644 | 2.099 | 2.382 | 2.695 | 3.334 | 37907.47 | 1 |
| etas[1] | 7.170 | 0.000 | 0.088 | 7.002 | 7.111 | 7.168 | 7.228 | 7.344 | 32841.47 | 1 |

### Comparison

Here the models are compared using cross-validated information criteria.

library(loo)  
fits |> lapply(\(fit) {extract\_log\_lik(fit, merge\_chains = FALSE)}) -> log\_lik  
log\_lik |> lapply(\(ll) {relative\_eff(exp(ll), cores = mycores)}) -> r\_eff  
fits |> length() |> seq\_len() |> lapply(\(i) {loo(log\_lik[[i]], r\_eff = r\_eff[[i]], cores = mycores)}) -> loos  
comp <- loo\_compare(loos)  
print(comp, simplify=FALSE)

## elpd\_diff se\_diff elpd\_loo se\_elpd\_loo p\_loo se\_p\_loo looic se\_looic  
## model4 0.0 0.0 -873.9 25.3 4.6 1.2 1747.7 50.7   
## model2 -0.4 1.6 -874.3 25.7 6.1 1.3 1748.6 51.4   
## model3 -3.9 3.4 -877.8 25.4 3.5 1.1 1755.6 50.8   
## model1 -4.5 3.6 -878.3 25.7 5.0 1.1 1756.7 51.4

The model with one line and two variances should be the most parsimonious, but all the models appear to be valid within prediction uncertainty (the IC values differ by less than 2 standard deviations).

**|| 20 marks for correctly bringing in the log likelihood generation and export OR 20 marks for implementing DIC calculations for both models OR 20 marks for any other valid approach such as Bayes factors. 5 marks for doing and explaining the comparison to say which model is most parsimonious. ||**

## Part 5

MyClears <- 2  
MyCookies <- 815  
MyTr <- 1 # No VPN

For my personal values I obtain a vector of predictions from the most parsimonious model

CookiesMu <- exp(draws$beta[,MyTr] \* sqrt(MyClears) + draws$alpha[,MyTr])  
CookiesSims <- MASS::rnegbin(nsims, CookiesMu, draws$phi[,MyTr])

I plot the density and annotate the observed value

p\_value <- mean(CookiesSims > MyCookies)  
MyPred <- mean(CookiesSims)  
with(density(CookiesSims), data.frame(Cookies = x, Density = y, key = y \* (x > MyCookies))) |>  
 ggplot(aes(x = Cookies, y = Density)) + geom\_line(colour = 'purple', size = 1.5) + geom\_area(aes(x = Cookies, y = key), fill = 'red') + geom\_text(aes(x = MyCookies, y = max(key), label = paste('My cookies =', MyCookies, '\np-value =', round(p\_value,3)), hjust = 'left', vjust = 'bottom')) + geom\_vline(xintercept = MyPred, colour = 'blue') + geom\_label(aes(x = MyPred, y = max(Density)\*0.8), label = paste('Expected cookies\n=', round(MyPred,0)), hjust = 'left')



**|| 10 marks for correctly bringing in own data and making predictions with uncertainty. 10 marks for summarising and showing predictions with uncertainty. ||**