Final Project

Giulio

Contents

Illustration of the Dataset

The dataset, available here downloading the comma-separated values file "data_640_validated.csv", aim to examine the relationship between game-playing, in-game behaviors, and environmental perceptions to fill in the gap of lacking resources for studying the effects of commercial video games.

The target of the survey are Nintendo's Animal Crossing: New Horizons (ACNH) game players. When playing ACNH, the players immerse into a deserted island with the responsibilities of building their paradisiac village by developing the ecosystem and community. Their daily activities are related to the environment, such as growing flowers, planting fruit, cut tree, catching fish, snaring bugs, or submitting the fish and bugs to the museum.

The data set includes six categories demonstrating different aspects of game players, here I will list only the two categories I will use for the project:

- Environmental perceptions: for examining the environmental perception of game players;
- In-game behavior: designed questions covering the most prominent activities associated with environmental values.

The survey was conducted from 15 to 30 May 2020 using Google Form in the communities of ACNH players on Discord, Reddit, and Facebook platforms.

Data and inferential goals of the analysis:

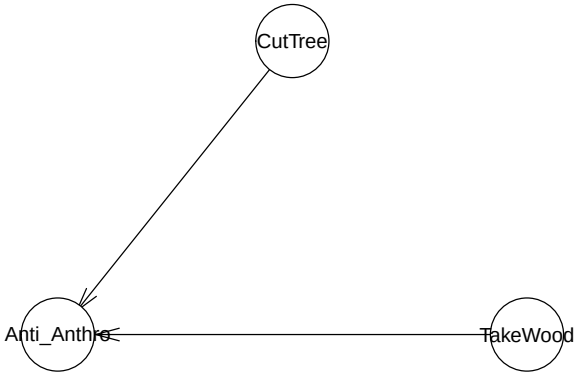
Following [1], we want to hypothesized that people holding an anti-anthropocentric (Anti_Anthro) perception would associate with the frequency of in-game behaviors that harm natural lifeforms. To test this hypothesis, they used three variables from the data set. The anti-anthropocentric perception is represented by the C12 variable, which measures the disagreement towards the statement "Humans were meant to rule over the rest of nature". Is a categorical variable that assume value in $\{1, 2, 3, 4, 5\}$, with higher value means a higher level of disagreement.

To explain the anti-anthropocentric, they use two variable E16 and E17, respectively the frequency action of taking wood (TakeWood) and cutting down a tree (CutTree). Each of them assuming value in $\{1, 2, 3, 4\}$, with higher value means a higher frequency on doing this action.

The two action is quite different since, whit the first we just obtain the natural resource without harming the natural lifeforms, while with the latter, we destroy the tree even after having received some wood from it.

We end up with the following regression method:

$Anti\_Anthro \sim \alpha + TakeWood + CutTre.$



Let's start import the dataset and keep only the useful variable.

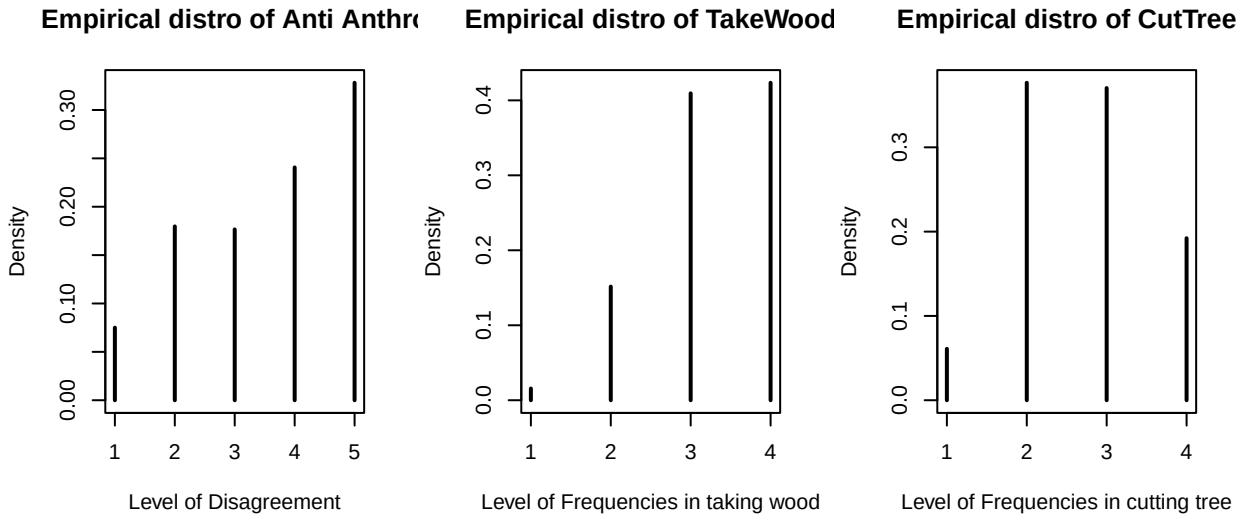
```
# -------- DATA  ---------- #
dat <- read.csv("data_640_validated.csv", header = TRUE)

# rename and keep this three
dat$Anti_Anthro <- dat$C12
dat$TakeWood    <- dat$E16
dat$CutTree     <- dat$E17

dat <- dat[, c("Anti_Anthro","TakeWood","CutTree")]
```

	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
Anti_Anthro	1	2	4	3.567188	5	5
TakeWood	1	3	3	3.240625	4	4
CutTree	1	2	3	2.693750	3	4

We can also look at the empirical distribution of our data:



and also inspect if there are some correlation:

```
corr_matrix <- rcorr(as.matrix(dat))$r
```

	Anti_Anthro	TakeWood	CutTree
Anti_Anthro	1.0000000	0.1171291	-0.0378046
TakeWood	0.1171291	1.0000000	0.3877392
CutTree	-0.0378046	0.3877392	1.0000000

We don't see significant correlation in the features.

The incorrect model: Guassian linear regression for categorical response

In the paper, they use a package called "bayesvl" [2] that use the software "STAN" for the MCMC simulation. What I discover is that basically they don't care about the prediction, but just do a Monte Carlo Markov Chain to obtain an estimate of the regression coefficient in a way to explain if a variable is positively or negatively associated to the response variable.

Their model (adding the variance prior to be able the running in Jags) was:

$$Y_i \sim N\left(\mu_i, \sigma^2\right)$$
$$\mu_i = \alpha + \beta_{TakeWood} \cdot TakeWood_i + \beta_{CutTree} \cdot CutTree_i$$

with prior distributions:

- $\alpha \sim N(0, 100)$
- $\beta_{TakeWood} \sim N(0, 10)$
- $\beta_{CutTree} \sim N(0, 10)$
- $\sigma^2 \sim IG(0.001, 0.001)$

As we can see they use a Gaussian distribution to simulate categorical value. Even if we can use STAN, I was able to translate the code to JAGS and obtain the exact results, adding a non informative Inverse Gamma prior for the variance of the Gaussian distribution. The model is show below.

```
  model
    {
        for( i in 1:N ) {
            Anti_Anthro[i] ~ dnorm(mu[i], precision)
            mu[i] <- a_Anti_Anthro + b_TakeWood * TakeWood[i] +
                          b_CutTree * CutTree[i]
        }

        # ---- PREDICTION ---- #
        # TakeWood = 1; CutTree = 4
    Ypred1 ~ dnorm(mu1, precision)  # random variable
    mu1 <- a_Anti_Anthro + b_TakeWood * 1 +
                          b_CutTree * 4

    # ---- PRIOR ---- #
        a_Anti_Anthro ~ dnorm(0, 0.01)
        b_TakeWood ~ dnorm(0.0, 0.1)
        b_CutTree ~ dnorm(0.0, 0.1)
        precision ~ dgamma(0.001, 0.001)
    }
```

We need to remark that in JAGS, the normal distribution takes in input the precision parameters, i.e the inverse of the variance. In our case we can directly input a Gamma distribution with the same hyperparameters.

Now we can run the simulation and reproduce their results:

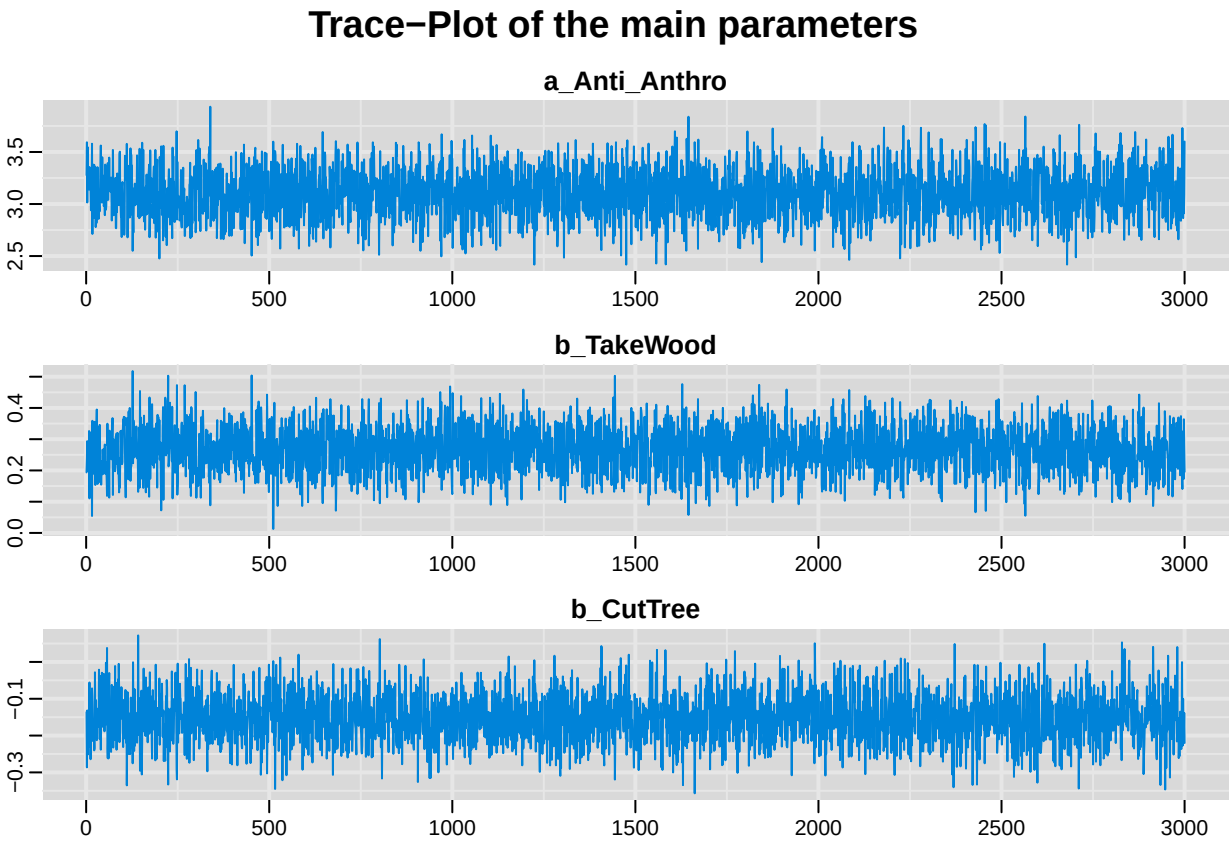
```r
# Building the data
data <- as.list(dat)
data$N <- nrow(dat)

# list of parameters name
parameters <- c("a_Anti_Anthro","b_TakeWood",
                "b_CutTree", "Ypred1")
# initial value
inits <- list(a_Anti_Anthro = 3, b_TakeWood = 0, b_CutTree = 0)
initial.values <- list(inits)

# MCMC with jags
set.seed(123) # set seed for reproducibility
model1 <- jags(data = data,
                   inits = initial.values,
                   parameters.to.save = parameters,
                   model.file = "model_jags.txt",
                   n.burnin = 2000,
```

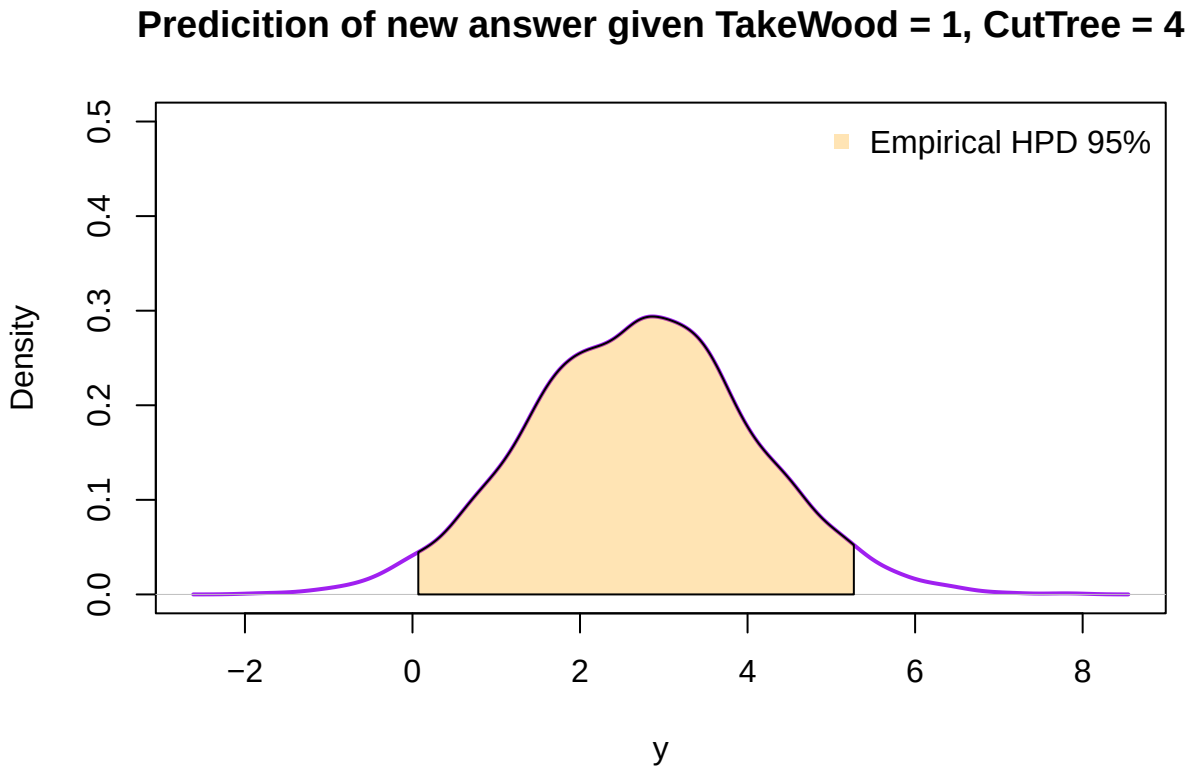
```
                    n.chains = 1, n.thin = 1,
                    n.iter = 5000)
```

	mean	sd	2.5%	25%	50%	75%	97.5%
a_Anti_Anthro	3.1065402	0.2372987	2.6465001	2.9386763	3.1075954	3.2698398	3.5622841
b_TakeWood	0.2682678	0.0733371	0.1243193	0.2190405	0.2697666	0.3180948	0.4085919
b_CutTree	-0.1520398	0.0661176	-0.2785039	-0.1959944	-0.1535395	-0.1084641	-0.0153052
Ypred1	2.7205422	1.3456423	0.0874592	1.8009502	2.7340773	3.6025123	5.3350224



As we can see, the trace-plot behave fine, the standard deviation of the main parameters is low, and also we obtain the same results as the authors of the article. We don't show here the other diagnostic, since we will explore it later for our model, but can be seen in the paper.

The problem is now, the prediction. Why we would want to use continuous prediction if the data are discrete? Also how we interpret the result and choose the right category?



We need to change the model.

The Multinomial Logistic Regression Model

Suppose we model a response variable $Y \in \{1, \dots, K\}$, where K is the number of level, and a set of p covariate $X = [X_1, \dots, X_p]^T \in \mathcal{X}$. Let $\pi_j(\boldsymbol{x}) = P(Y = j \mid \boldsymbol{x})$ with $\sum_{j=1}^{K} \pi_j(\boldsymbol{x}) = 1$ [3].

This type of model is also called "Baseline Logistic Regression Model" since we pair each response categories with a baseline category $k^\star$.

The Multinomial Logisitc Regression model in log-odd forms is:

$$\log\left(\frac{\pi_j(\boldsymbol{x})}{\pi_{k^\star}(\boldsymbol{x})}\right) = \alpha_j + \boldsymbol{\beta}_j \cdot \boldsymbol{x} \qquad \forall j \in \{1, \dots, K\} \setminus k^\star$$

where $\alpha \in \mathbb{R}^{K-1}$ and $\boldsymbol{\beta} \in \mathbb{R}^{(K-1)\times p}$ are the regression coefficients. The ratio of the probability of choosing one outcome category over the probability of choosing the baseline category is often referred as **relative risk** and it is also sometimes referred as **odds**. The **relative risk** is the linear model exponentiated, leading to the fact that the exponentiated regression coefficients are relative risk ratios for a unit change in the predictor variable.

The log is used to map the range $[0, 1] \mapsto \mathbb{R}_+$ given a meaning to the regression image space.

We end up with a system of $K - 1$ equations. We can also write the model in terms of the probabilities, for the notation I set $k^\star = K$:

$$\pi_j(\boldsymbol{x}) = \frac{\exp\{\alpha_j + \boldsymbol{\beta}_j \cdot \boldsymbol{x}\}}{1 + \sum_{r=1}^{K-1} \exp\{\alpha_r + \boldsymbol{\beta}_r \cdot \boldsymbol{x}\}} \qquad \forall j = 1, \ldots, K-1;$$

$$\pi_K(\boldsymbol{x}) = 1 - \pi_1(\boldsymbol{x}) - \ldots - \pi_{K-1}(\boldsymbol{x}).$$

Implementation in Jags

Using Jags I need a list with the observation of the response variable, the feature, and number of observation with the number of category.

```
# ---- Convert to list for Jags input ------- #
data <- as.list(dat)    # variable
data$N <- nrow(dat)     # n-row
data$J <- length(as.numeric(levels(as.factor(dat$Anti_Anthro))))  # n-categories
```

The formal multinomial logit regression model to implement is

$$Anti\_Anthro \sim \text{multinomial}(\pi, 1)$$

$$\pi_j = \frac{\phi_j}{\sum_{j=1}^{K} \phi_j}$$

$$\log(\phi_j) = \alpha_j + \beta_j^{TakeWood} \cdot TakeWood + \beta_j^{CutTree} \cdot CutTree$$

which in jags the code its the one below [4]. Here we use the *dcat* distribution which is the same as taken a multinomial distribution with a single sample draw, more detail can be found in the Jags documentation [5]. We will use as baseline category the one that occur more frequently in the dataset, which result in the last category 5.

```
model
  {
    # ------- Multinomial Logit Regression ------ #
    # baseline category 5
  for(i in 1:N){

      Anti_Anthro[i] ~ dcat(p[i, 1:J])

      for (j in 1:J){
        log(phi[i,j]) <-  intercept[j] +
                          b_TakeWood[j] * TakeWood[i] +
                          b_CutTree[j] * CutTree[i]

        p[i,j] <- phi[i,j]/sum(phi[i,1:J])
      }
    }
```

```
  # We need to fix the effects corresponding
  # to the >>last<< observation category to 0:
  intercept[J] <- 0
  b_TakeWood[J] <- 0
  b_CutTree[J] <- 0

  # ------  PRIOR --------- #
  for(j in 1:(J-1)){
      intercept[j] ~ dnorm(0, 0.01)
      b_TakeWood[j] ~ dnorm(0, 0.1)
      b_CutTree[j] ~ dnorm(0, 0.001)
  }

  # ------- PREDICTION  -------- #
  Anti_Anthro_new ~ dcat(pnew[1:J])

  for (j in 1:J){
    log(phinew[j]) <-  intercept[j] +
                    b_TakeWood[j] * 1 +
                    b_CutTree[j] * 4

    pnew[j] <- phinew[j]/sum(phinew[1:J])
    }
}
```

Now we need tu build up the next component of the simulations.

The *parameters* array used to store the variable names, not necessary only the prior distributions, that allow us to retrieve the simulation for the variable in interest.

The *initial.values* list in which we store a number of list corresponding to the number of chain we want to produce. Each list contains the starting value of each chain.

Other parameters:

- n.chain = 2: because we want to do different simulation with different starting value to see how the chain is effected by the starting point;
- n.thin = 10: because we want to achieve less autocorrelation during the Markov Process, otherwise will be very high;
- n.burnin = 7000: large number of iteration to be discarded before the actual chain is considered. We use this high value because the chain of each parameters are very slow to converge to a stationary value;
- n.iter = 40000: large number of iteration to compensate the n.thin and n.burnin that reduce the number of sample we generate, also to have a consistent number of effective (usable) sample size.

To decide the number put above and below, I look at some diagnostic like Trace Plot, ACF plot, Cumulative Mean, that allow me to decide the burnin time or the thinning number, along with the starting value.

```r
# List of parameters name
parameters <- c("intercept", "b_TakeWood", "b_CutTree", "Anti_Anthro_new")

# Starting value initialization
inits1 <- list( intercept = c(0,0,0,0,NA),
                b_TakeWood = c(0,0,0,0,NA),
                b_CutTree = c(0,0,0,0,NA))

inits2 <- list( intercept = c(0.5,0.5,0.5,0.5,NA),
                b_TakeWood = c(-0.5,-0.5,-0.5,-0.5,NA),
                b_CutTree = c(0.5,0.5,0.5,0.5,NA))

initial.values <- list(inits1 = inits1, inits2 = inits2)

# MCMC with jags
set.seed(123)
MNL <- jags(data = data,
            inits = initial.values,
            parameters.to.save = parameters,
            model.file = "model_multicategory.txt",
            n.chains = 2, n.thin = 10,
            n.burnin = 7000,
            n.iter = 40000)
```

We can look briefly some point estimate like posterior mean, standard deviation and the effective sample size we manage to achieve for the regression coefficient. Along with the quantile and the median, the 50% quantile.

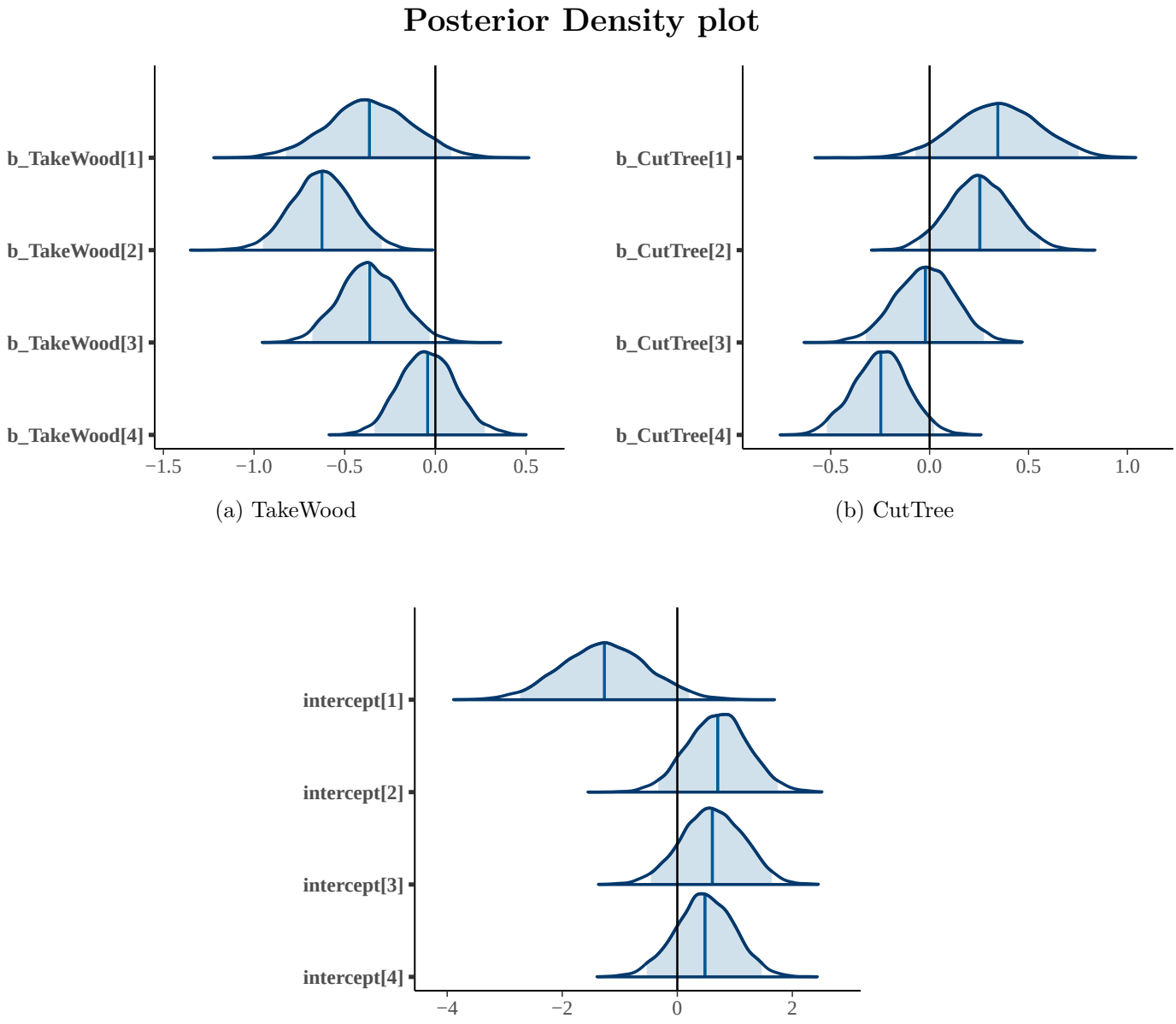
	mean	sd	2.5%	25%	50%	75%	97.5%	Rhat	n.eff
b_TakeWood[1]	-0.3634910	0.2314050	-0.8240214	-0.5168557	-0.3653721	-0.2066157	0.0867703	1.001473	2400
b_TakeWood[2]	-0.6254905	0.1679993	-0.9511199	-0.7377799	-0.6256846	-0.5134844	-0.2961969	1.001910	1400
b_TakeWood[3]	-0.3616129	0.1673039	-0.6787718	-0.4744787	-0.3658801	-0.2472740	-0.0314093	1.002183	1100
b_TakeWood[4]	-0.0424661	0.1543957	-0.3355127	-0.1497099	-0.0451959	0.0619991	0.2735366	1.002157	1100
b_CutTree[1]	0.3446162	0.2117321	-0.0711096	0.2002174	0.3445963	0.4912036	0.7536144	1.000997	6600
b_CutTree[2]	0.2537373	0.1558718	-0.0503122	0.1470681	0.2519781	0.3595380	0.5585593	1.007552	230
b_CutTree[3]	-0.0214741	0.1523527	-0.3227725	-0.1240064	-0.0181938	0.0812711	0.2746912	1.002299	1000
b_CutTree[4]	-0.2468131	0.1389514	-0.5200658	-0.3390727	-0.2442340	-0.1540127	0.0221728	1.001280	3500
intercept[1]	-1.2678249	0.7465066	-2.7302549	-1.7819604	-1.2680930	-0.7681111	0.2034422	1.002145	1200
intercept[2]	0.7033872	0.5302338	-0.3317949	0.3461893	0.7103842	1.0543060	1.7475303	1.003603	1200
intercept[3]	0.6098319	0.5413578	-0.4596075	0.2405106	0.6101575	0.9900124	1.6430897	1.000861	6600
intercept[4]	0.4816724	0.5115879	-0.5328926	0.1453303	0.4836918	0.8288441	1.4690768	1.003663	530

Looking at the posterior estimate of the standard deviation we can see how variable is the posterior distribution around the mean, and this values are not so low.

Also the effective sample size are relatively large for the simulation we have done, although some parameters are not.

We can extract the chain and begin our analysis.

```r
# ------ MCMC Posterior parameters ----- #
MNL.mcmc = as.mcmc(MNL)
b_TakeWood <- MNL.mcmc[, c(paste("b_TakeWood[", seq(1,4), "]", sep = ""))]
b_CutTree  <- MNL.mcmc[, c(paste("b_CutTree[", seq(1,4), "]", sep = ""))]
intercept  <- MNL.mcmc[, c(paste("intercept[", seq(1,4), "]", sep = ""))]
```



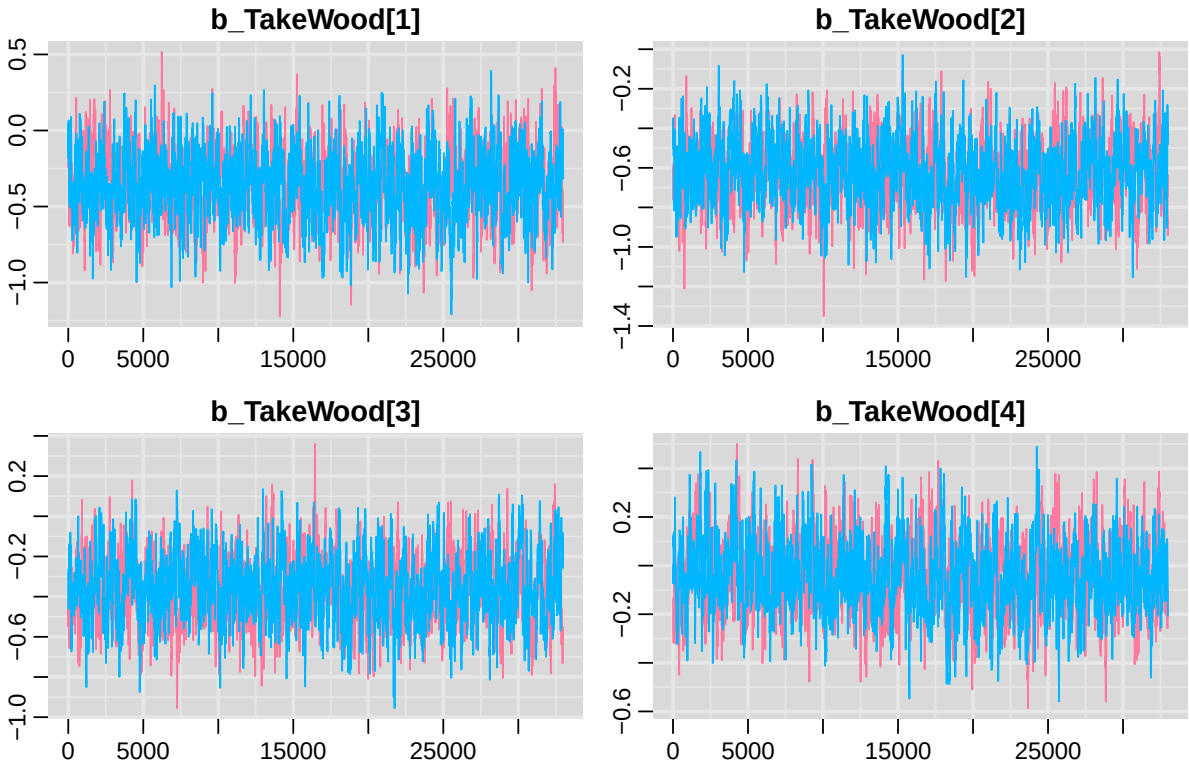
(a) TakeWood

(b) CutTree

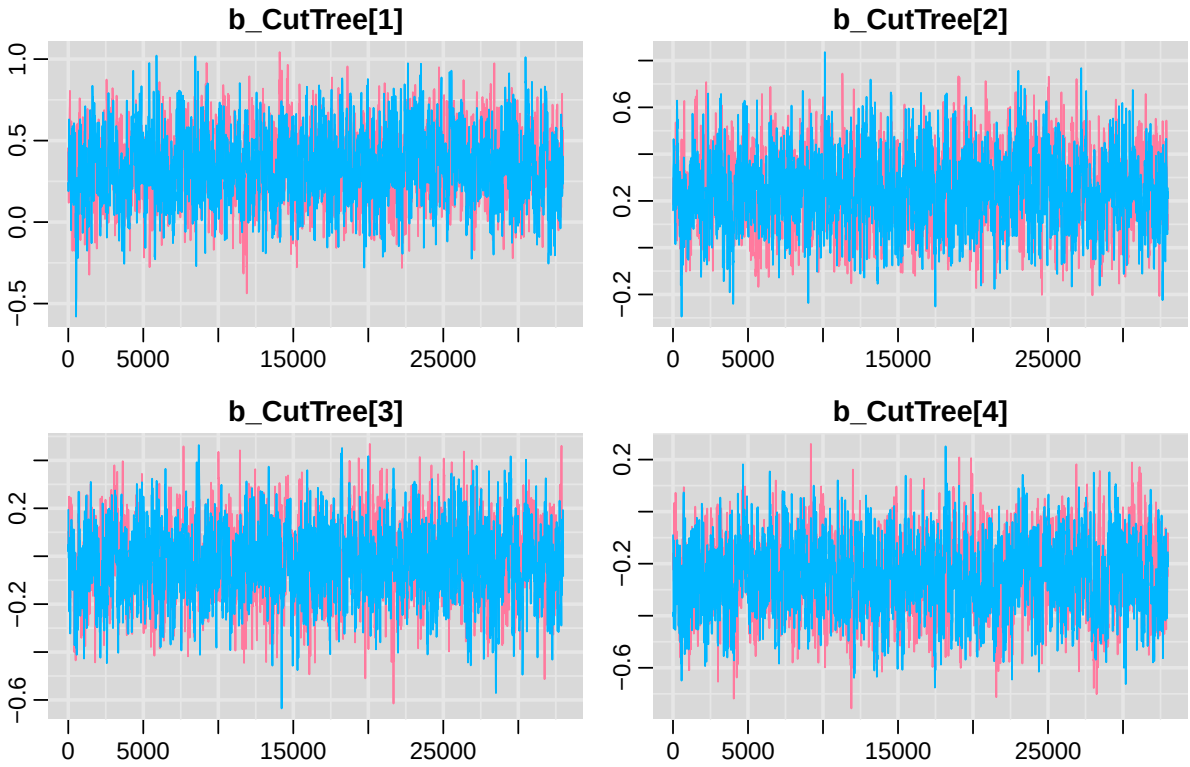
(c) Intercept

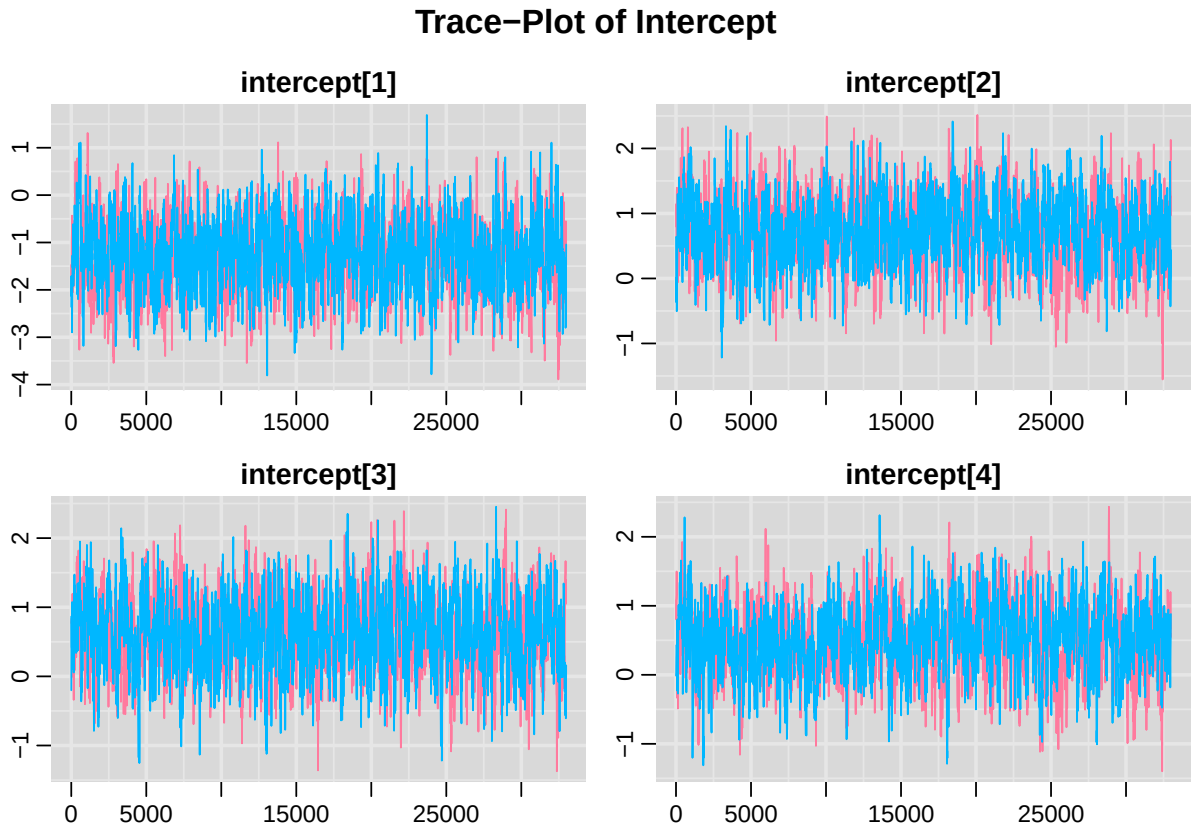
Here we plot the posterior densities of each regression coefficients, along with the 95% credible interval. We can see that the uncertainty around the mean is very large as we notice before with the standard deviation. We can also say that not all the parameters have a Credible interval that allow us to define a constant sign.

Trace−Plot of TakeWood



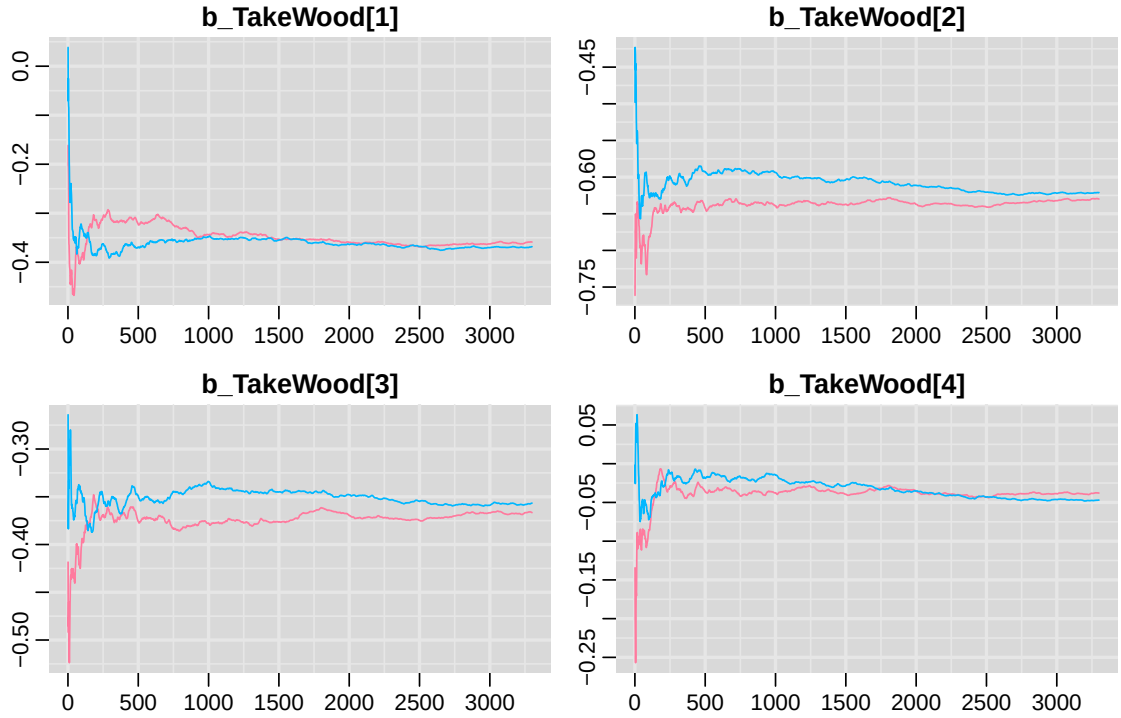
Trace−Plot of CutTree



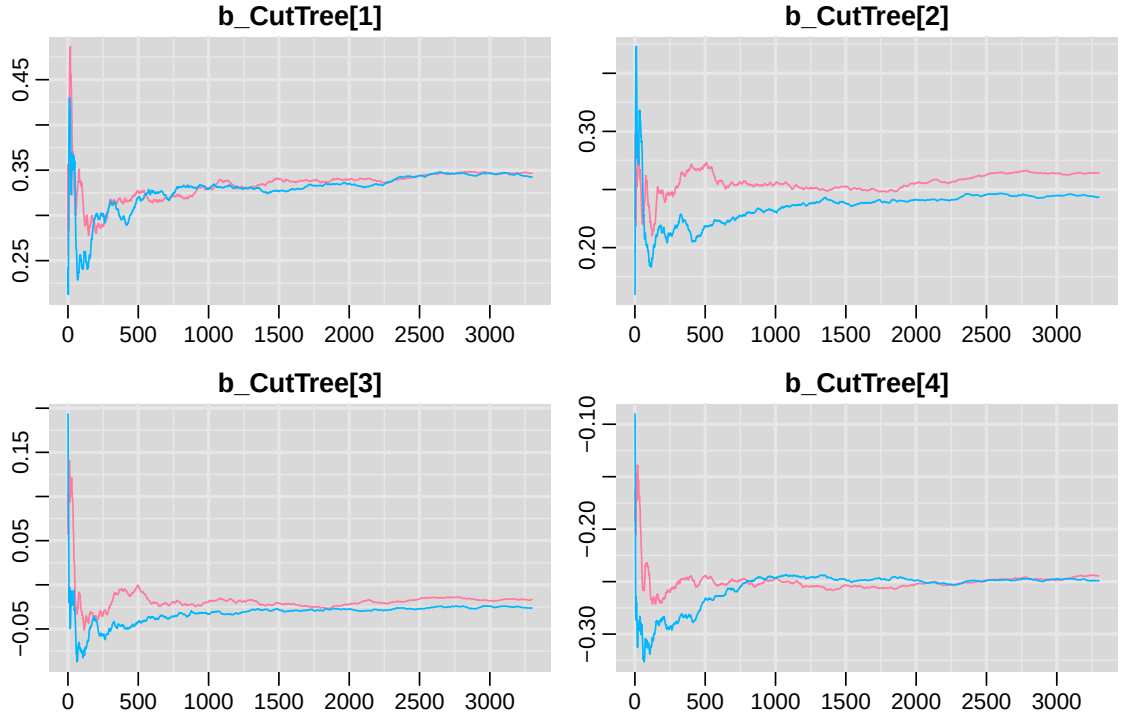


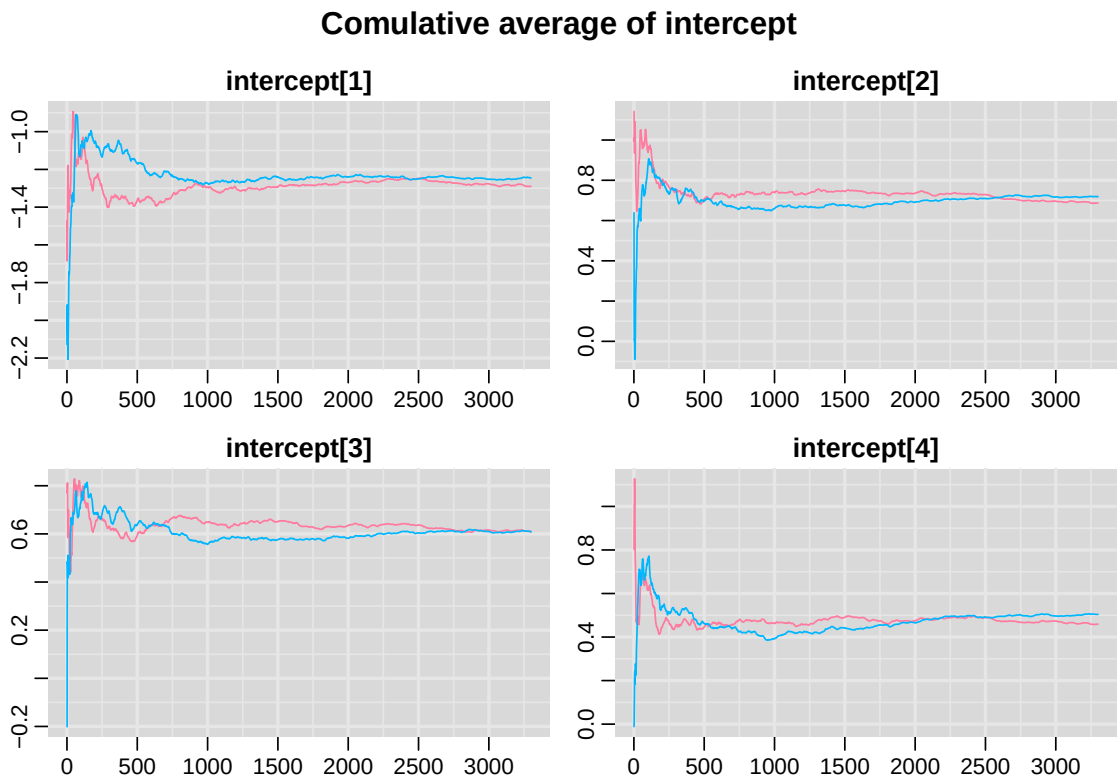
We can see that the traceplot are very dense, meaning that we achieve a good amount of effective sample size. Other insight is that the chain with different starting value overlap meaning that the Markov Process behave well, along with the fact that swing in a bounded range.

Comulative average of Takewood



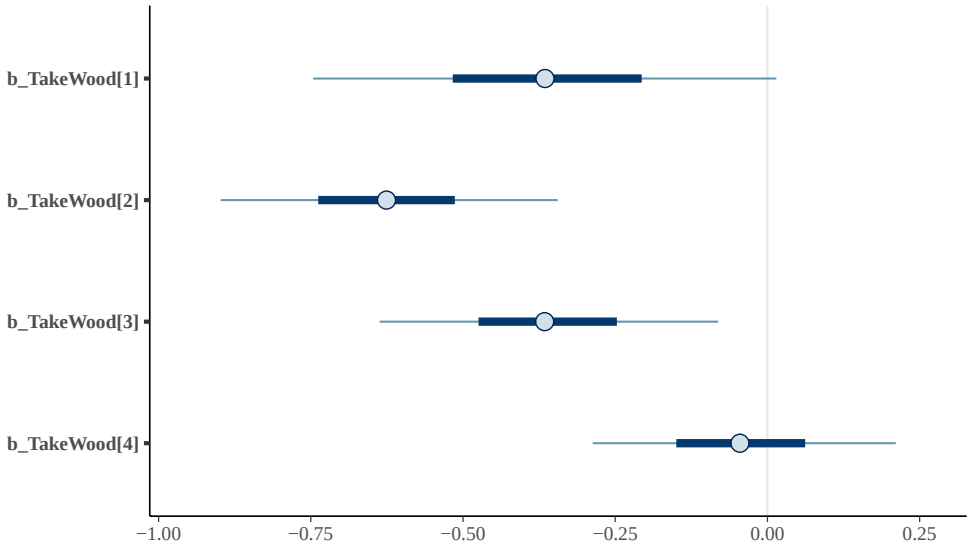
Comulative average of CutTree



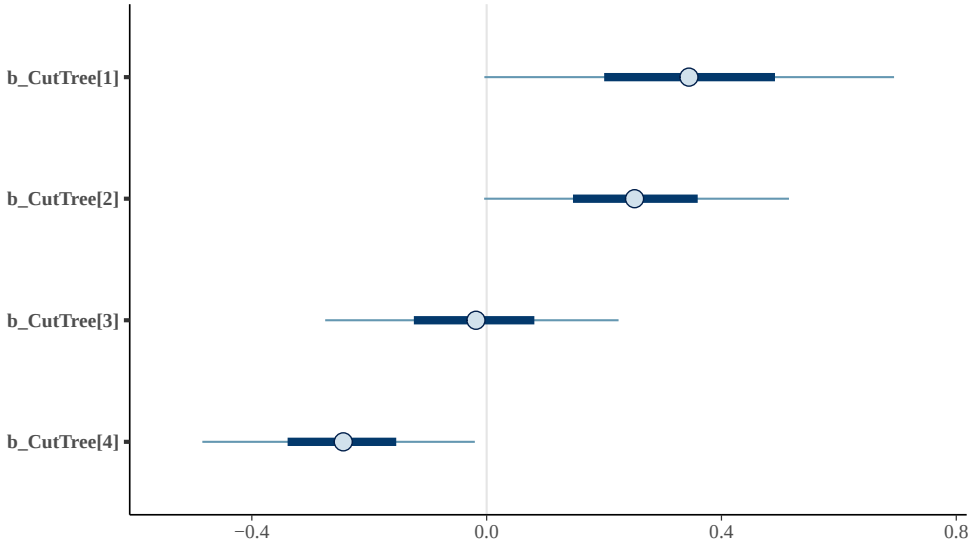


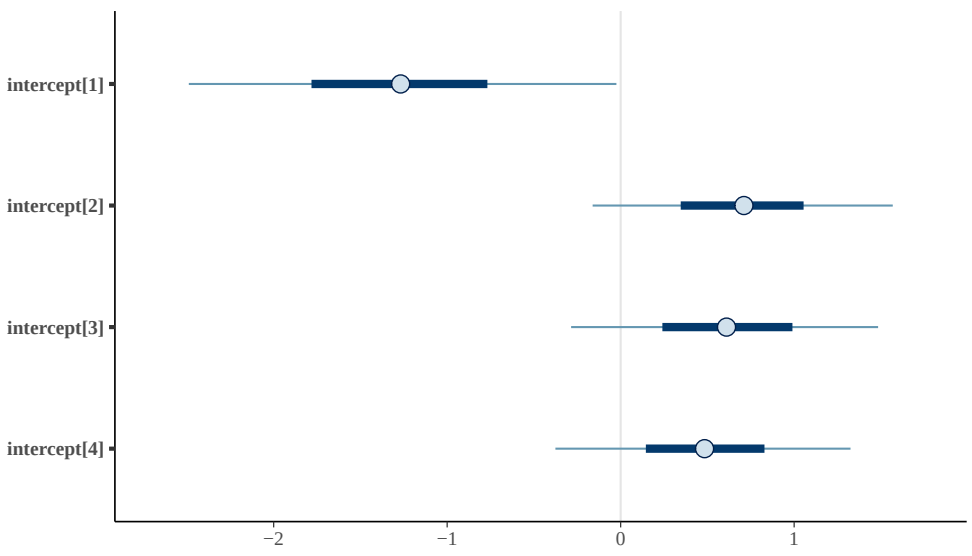
Most of the parameters seems to be reach a stationary value and also with different value they reach the same conclusion. Other parameters instead may need some extra time to reach a convergence.

TakeWood Intervals



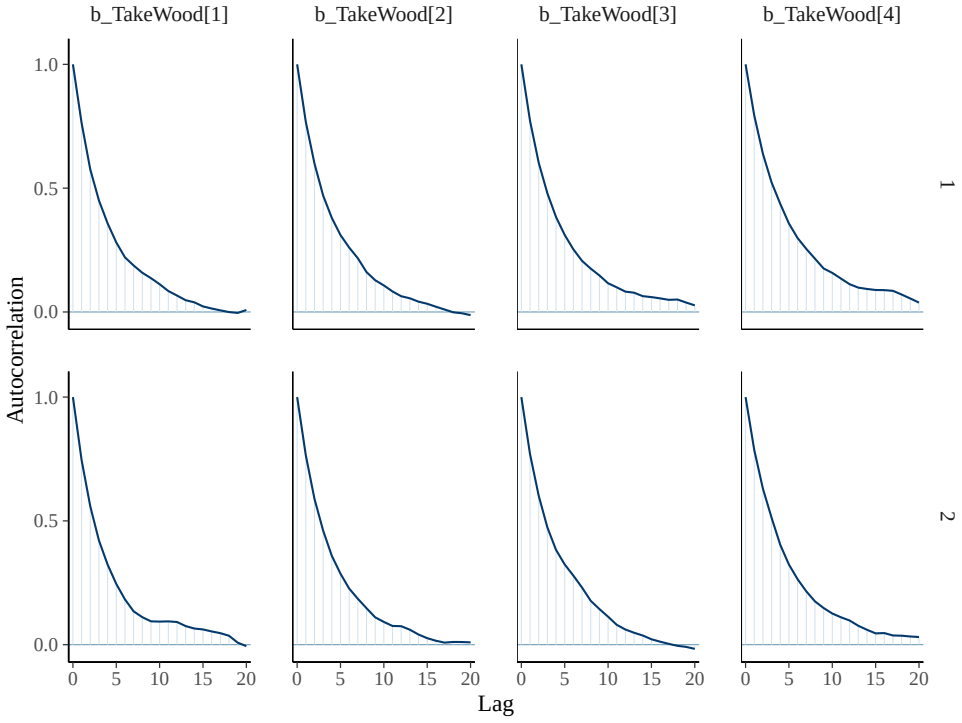
CutTree Intervals



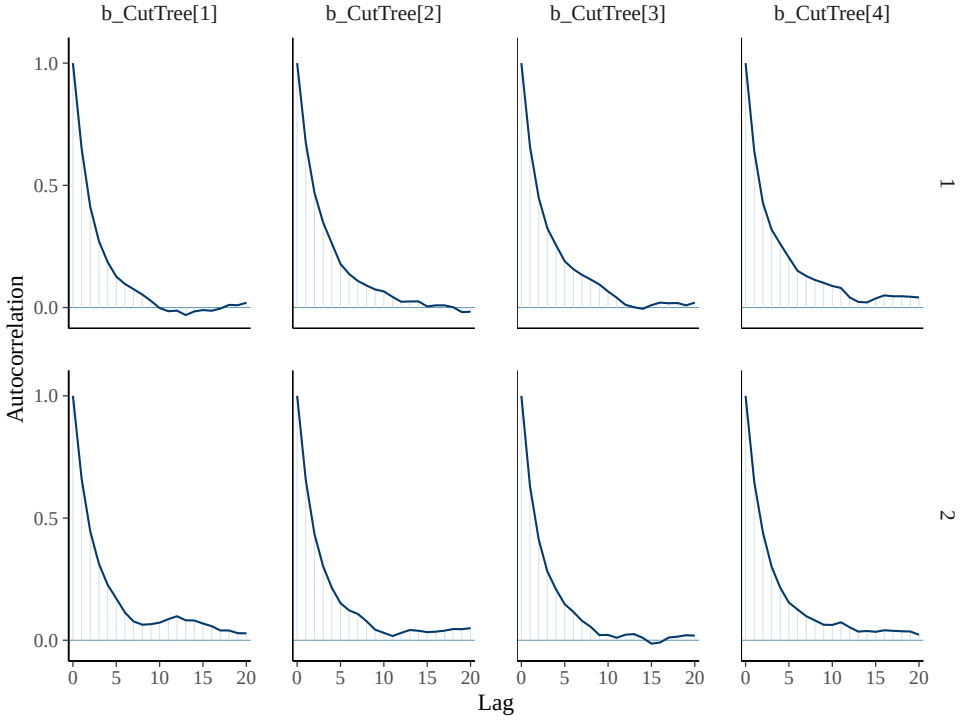


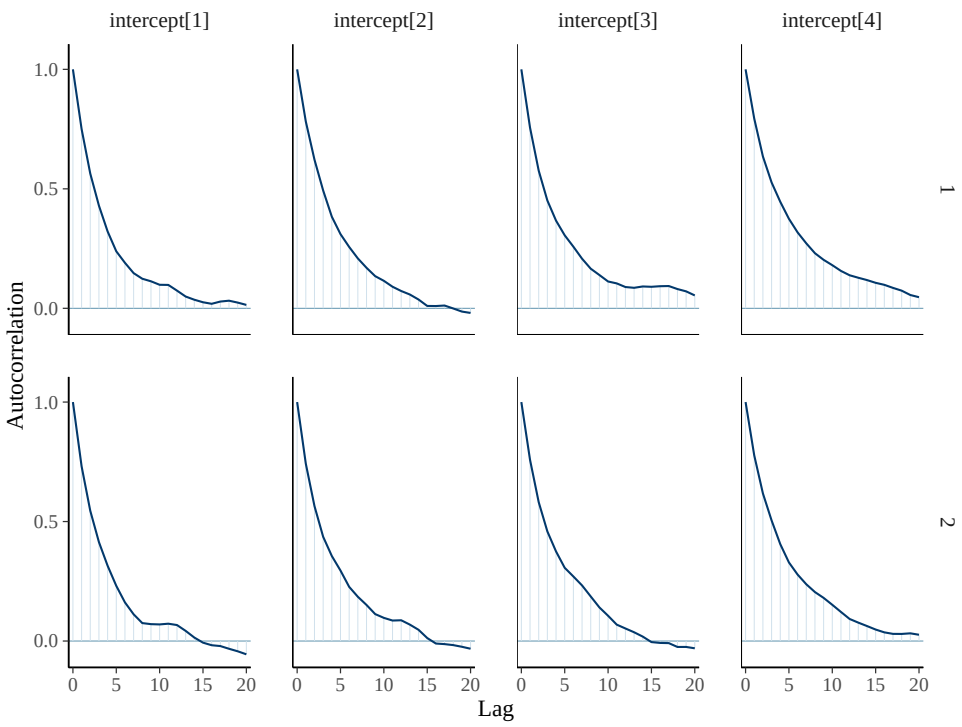
Here we can visualize better the coefficient sign, in base of a 50% and 90% CI. We can see that not all the parameters agree grouped in sign, or remain with high probability with the same sign.

TakeWood ACF



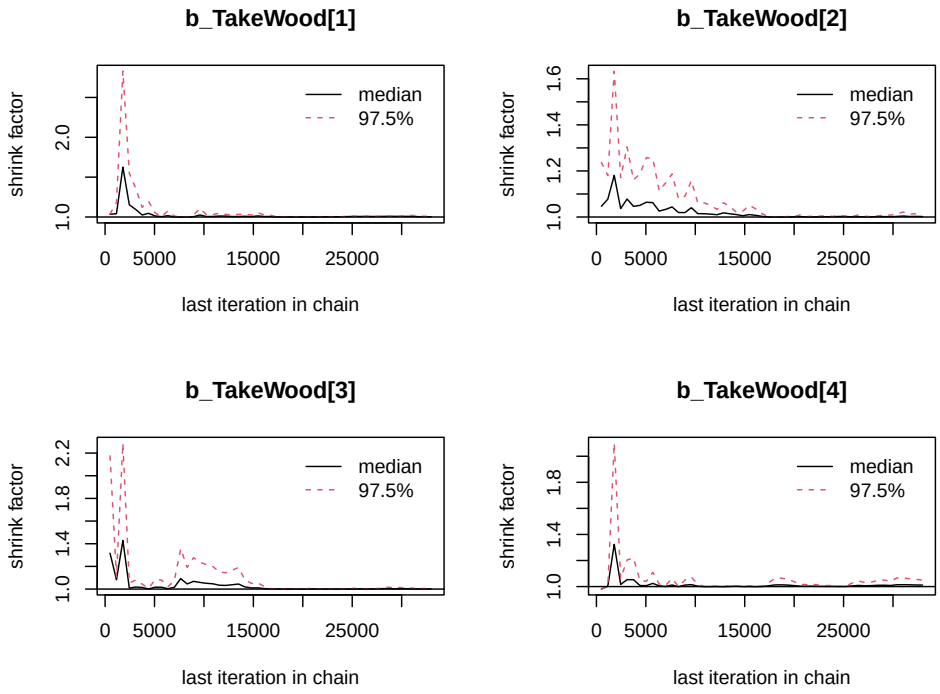
CutTree ACF



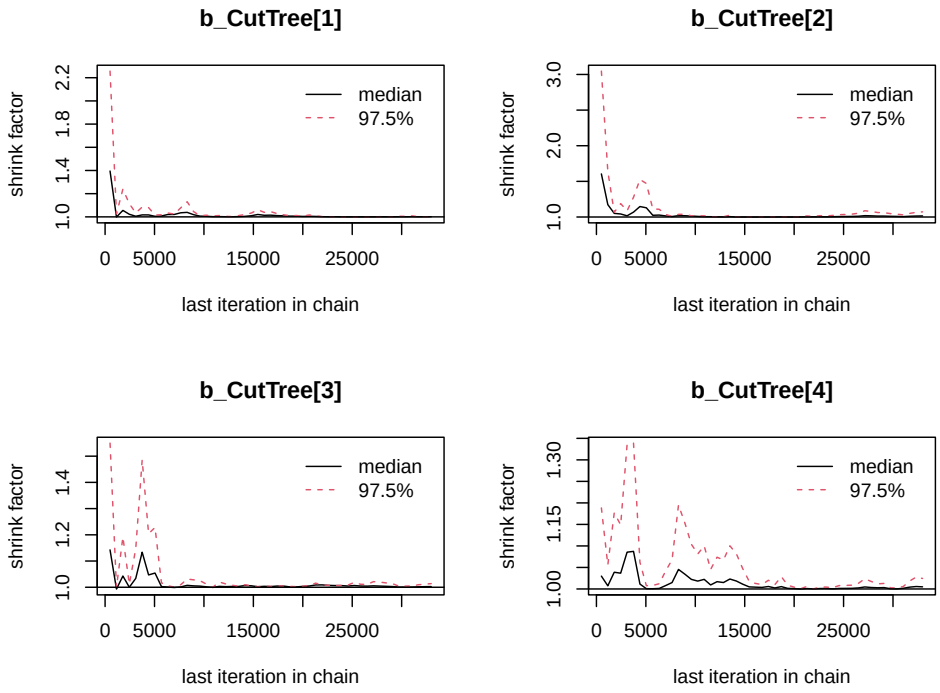


From this Autocorrelation Plot we can figure out that Markov Chain doesn't have low autocorrelation during its whole process, but more we increase the lag, more the autocorrelation decrease faster. We can increase the number of thinning in our simulation but this will provide less number of effective sample size unless we increase the already high number of iteration.

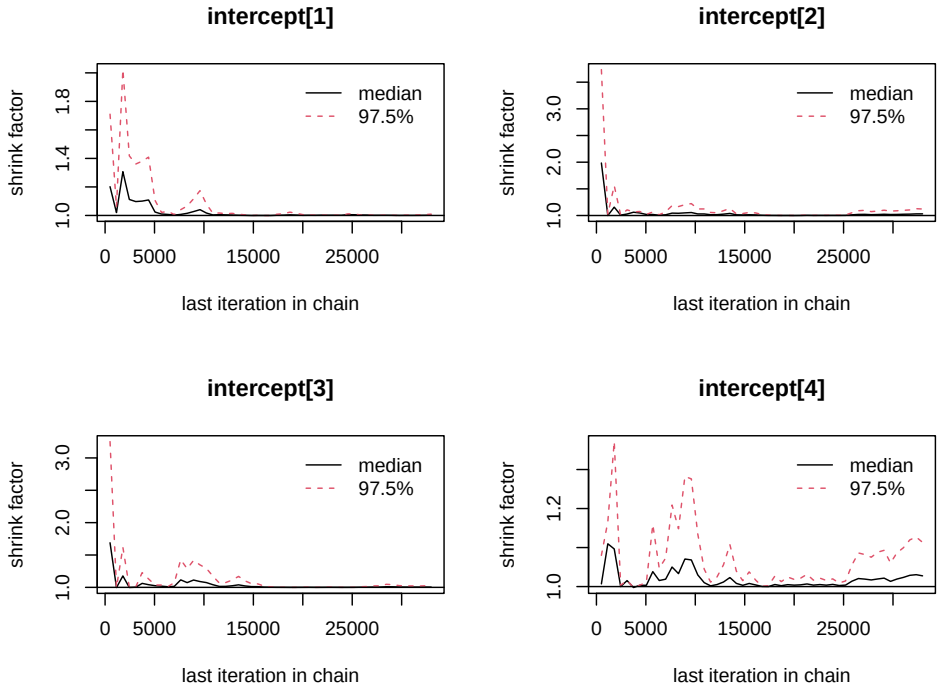
TakeWood Gelman Plot



CutTree Gelman Plot



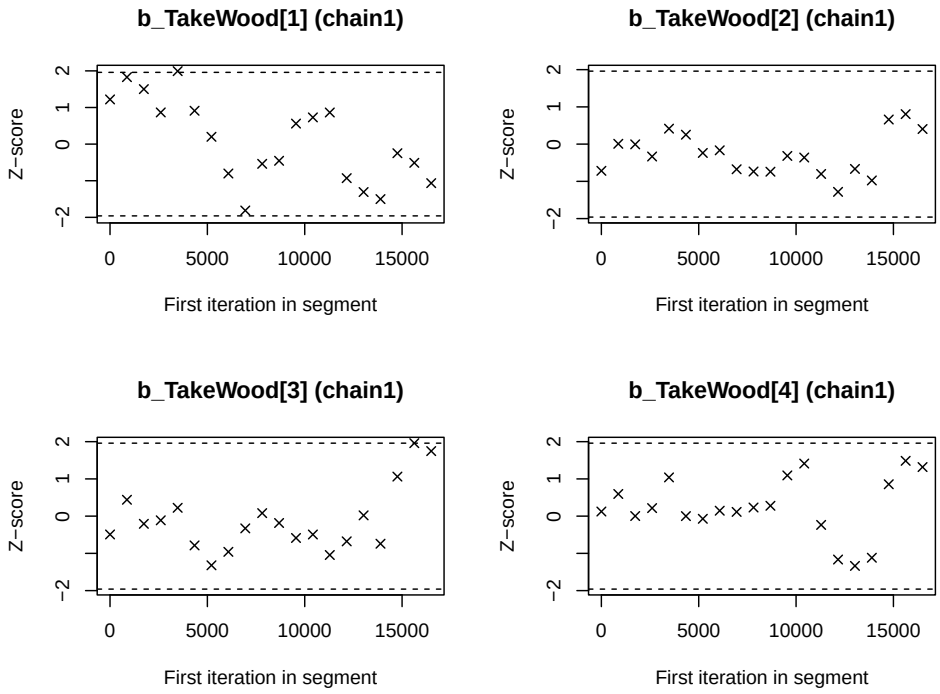
Intercept Gelman Plot



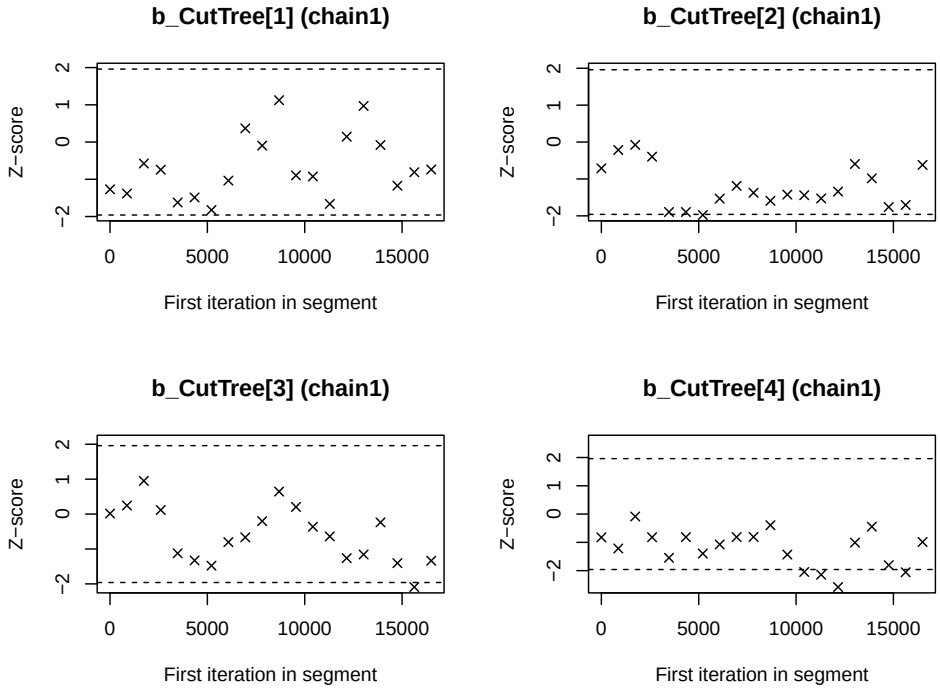
The Gelman and Rubin test compares the within and between chain variances. It produces a test statistic called "R.hat". If R.hat is larger than 1 is a sign of non convergence.

In our case we can see that in a long run the chain behave quite well for all the parameters except the coefficient $Intercept[4]$ which in a long run tends to increase the R.hat score.

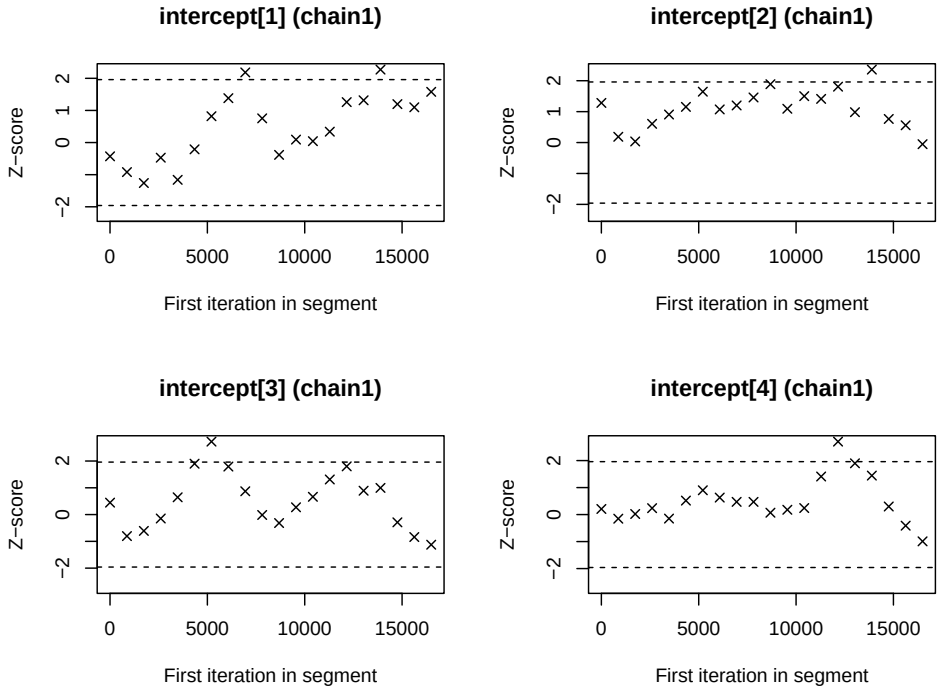
TakeWood Geweke Plot



CutTree Geweke Plot



Intercept Geweke Plot



The Geweke convergence diagnostic for Markov chains is based on a test for equality of the means of the first and last part of a Markov chain (by default the first 10% and the last 50%). If the samples are drawn from the stationary distribution of the chain, the two means are equal and Geweke's statistic has an asymptotically standard normal distribution. Since for some parameters we can see that the score exceed the bands, its a sign of non convergence yet and may require some extra iteration.

Heidelberger e Welch Diagnostic

The Heidelberger-Welch test directly tests whether or not the draws for each parameter come from a stationary distribution. A "failure" indicates that a longer MCMC run is needed, while if "passed" the number of iterations to keep and the number to discard (burn-in) are reported.

```
## [1] "TakeWood coefficients"

## [[1]]
##
##                Stationarity start     p-value
##                test         iteration
## b_TakeWood[1] passed       1         0.320
## b_TakeWood[2] passed       1         0.401
## b_TakeWood[3] passed       1         0.524
## b_TakeWood[4] passed       1         0.969
##
##                Halfwidth Mean    Halfwidth
##                test
## b_TakeWood[1] passed    -0.3588 0.0220
## b_TakeWood[2] passed    -0.6300 0.0159
## b_TakeWood[3] passed    -0.3666 0.0158
## b_TakeWood[4] failed    -0.0379 0.0156
##
## [[2]]
##
##                Stationarity start     p-value
##                test         iteration
## b_TakeWood[1] passed         1       0.5794
## b_TakeWood[2] passed         1       0.0773
## b_TakeWood[3] passed         1       0.4359
## b_TakeWood[4] passed       331       0.0767
##
##                Halfwidth Mean   Halfwidth
##                test
## b_TakeWood[1] passed    -0.368 0.0207
## b_TakeWood[2] passed    -0.621 0.0156
## b_TakeWood[3] passed    -0.357 0.0175
## b_TakeWood[4] failed    -0.050 0.0165

## [1] "CutTree coefficients"

## [[1]]
##
##                Stationarity start     p-value
##                test         iteration
## b_CutTree[1] passed       1         0.1489
## b_CutTree[2] passed       1         0.0836
## b_CutTree[3] passed       1         0.4992
```

```
## b_CutTree[4] passed       1         0.1024
##
##              Halfwidth Mean    Halfwidth
##              test
## b_CutTree[1] passed     0.3467 0.0155
## b_CutTree[2] passed     0.2641 0.0131
## b_CutTree[3] failed    -0.0167 0.0128
## b_CutTree[4] passed    -0.2445 0.0118
##
## [[2]]
##
##              Stationarity start     p-value
##              test         iteration
## b_CutTree[1] passed       1         0.188
## b_CutTree[2] passed       1         0.409
## b_CutTree[3] passed       1         0.786
## b_CutTree[4] passed       1         0.646
##
##              Halfwidth Mean    Halfwidth
##              test
## b_CutTree[1] passed     0.3425 0.0162
## b_CutTree[2] passed     0.2433 0.0115
## b_CutTree[3] failed    -0.0262 0.0116
## b_CutTree[4] passed    -0.2492 0.0105

## [1] "Intercept coefficients"

## [[1]]
##
##              Stationarity start     p-value
##              test         iteration
## intercept[1] passed       1         0.4434
## intercept[2] passed       1         0.0578
## intercept[3] passed       1         0.3011
## intercept[4] passed       1         0.5459
##
##              Halfwidth Mean   Halfwidth
##              test
## intercept[1] passed    -1.290 0.0676
## intercept[2] passed     0.688 0.0550
## intercept[3] passed     0.612 0.0528
## intercept[4] failed     0.460 0.0571
##
## [[2]]
##
##              Stationarity start     p-value
##              test         iteration
## intercept[1] passed         1       0.920
## intercept[2] passed         1       0.151
```

```
## intercept[3] passed         1       0.618
## intercept[4] passed       661       0.091
##
##              Halfwidth Mean   Halfwidth
##              test
## intercept[1] passed    -1.246 0.0644
## intercept[2] passed     0.719 0.0476
## intercept[3] passed     0.608 0.0575
## intercept[4] failed     0.518 0.0599
```

While we can see that we pass the stationary test for all the parameters, we can also view that some don't succeed the half-width test which calculates a 95% confidence interval for the mean, using the portion of the chain which passed the stationary test. Is an hint that we may have to increase the sample size.

Coefficient interpretation

Before going to the interpretation in the Multinomial Logistic model sense, we can inspect if there is some linear correlation among the main parameters:

Correlation Matrix:

```
corr_matrix <- rcorr(cbind(b_TakeWood[[1]], b_CutTree[[1]]))$r
```

	b_TakeWood[1]	b_TakeWood[2]	b_TakeWood[3]	b_TakeWood[4]	b_CutTree[1]	b_CutTree[2]	b_CutTree[3]	b_CutTree[4]
b_TakeWood[1]	1.0000000	0.3134961	0.3019714	0.2867066	-0.4354609	-0.1022401	-0.1142858	-0.0990691
b_TakeWood[2]	0.3134961	1.0000000	0.4446417	0.4108316	-0.1062845	-0.4125277	-0.1818006	-0.1559717
b_TakeWood[3]	0.3019714	0.4446417	1.0000000	0.4238713	-0.1332975	-0.1436757	-0.4167591	-0.1619835
b_TakeWood[4]	0.2867066	0.4108316	0.4238713	1.0000000	-0.1149051	-0.1290111	-0.1694094	-0.3899213
b_CutTree[1]	-0.4354609	-0.1062845	-0.1332975	-0.1149051	1.0000000	0.2149386	0.2354748	0.2448369
b_CutTree[2]	-0.1022401	-0.4125277	-0.1436757	-0.1290111	0.2149386	1.0000000	0.3521039	0.3740327
b_CutTree[3]	-0.1142858	-0.1818006	-0.4167591	-0.1694094	0.2354748	0.3521039	1.0000000	0.3774650
b_CutTree[4]	-0.0990691	-0.1559717	-0.1619835	-0.3899213	0.2448369	0.3740327	0.3774650	1.0000000

As we can see, looking at the grouped coefficient TakeWood and CutTree, there is no evidence of linear correlation (we can see that the absolute value is less than ~ 15). We can see perhaps that there is a negative relationship (all value are negative) and we reach a moderate value for regression coefficient of the same category (~ 45).

Instead viewing the coefficients for the proper block, the parameters present a positive relationship but we don't reach value greater than ~ 45.

Log Odd base interpretation:

Here we view the mean of the coefficients in log-odd probabilities along with the 90% Equal tails.

	lower	mean	upper
b_TakeWood[1]	-0.7459979	-0.3634910	0.0145990
b_TakeWood[2]	-0.8980720	-0.6254905	-0.3444104
b_TakeWood[3]	-0.6367926	-0.3616129	-0.0808827
b_TakeWood[4]	-0.2868338	-0.0424661	0.2108080
b_CutTree[1]	-0.0038927	0.3446162	0.6940087
b_CutTree[2]	-0.0042657	0.2537373	0.5150233
b_CutTree[3]	-0.2750519	-0.0214741	0.2247202
b_CutTree[4]	-0.4843196	-0.2468131	-0.0201463

Focusing on the *Takewood features*:

- b_TakeWood[1]: the odds of choose "(1) Strongly agree" over "(5) Strongly disagree" decrease as the frequency of $x = TakeWood$ increase, meaning that more often we do this action, less we agree in violating the nature. Is important to remark that this value is not often negative, but 90% ET shows us that there, also if small, a probability to have a positive value, and also we include the zero.

- b_TakeWood[2], b_TakeWood[3]: the odds of choosing "(2) Agree", "(3) Unsure" instead of "(5) Strongly disagree" decrease as the frequency of $x = TakeWood$ increase, this time, they are always negative, and as before more often we do this action, less we agree in violating the nature.
- b_TakeWood[4]: choosing "(4) Disagree" instead of (5) neither increase or decrease in a constant way as the frequency of Taken wood increase. What we can say is that in average, remain negative, coherent with the theoretical meaning of the covariate, and result smaller than the previous ones.

In the end we can say that overall, increasing how often we Take Wood in the game, leads to disagree (have higher category) in "humans were meant to rule over nature". Citing the Authors: *"TakeWood is positively associated with Anti_Anthro. [..] A game player that takes wood from the tree is more likely to disagree that "humans were meant to rule over nature".*

Instead for the *CutTree*:

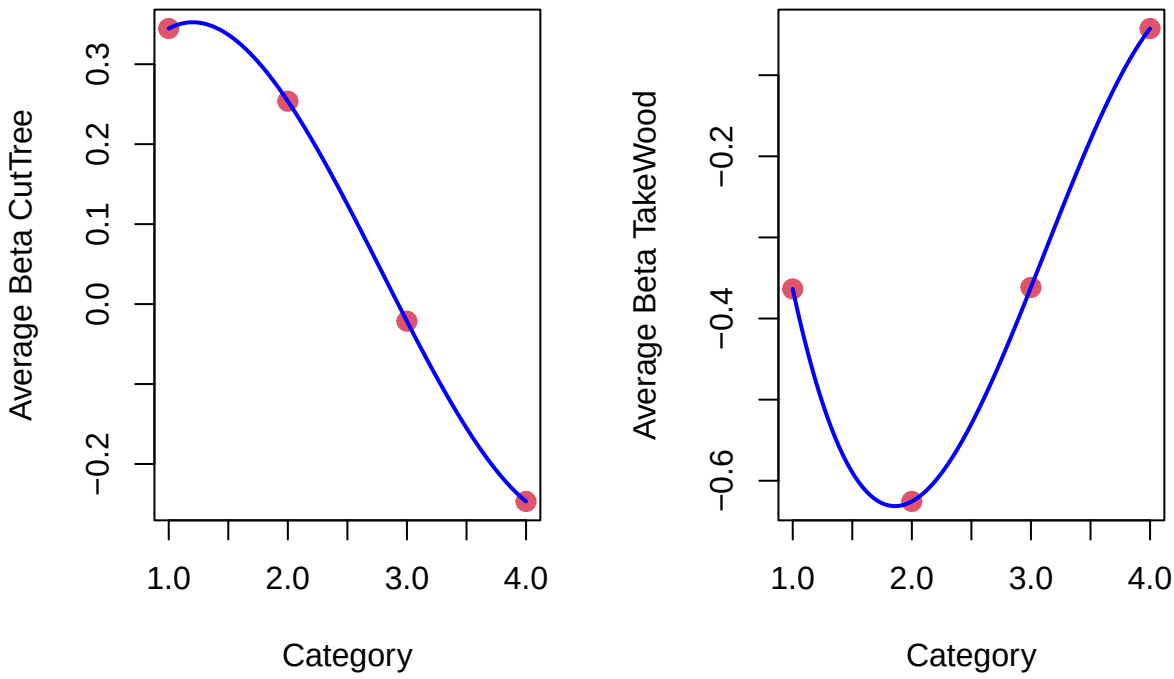
- b_CutTree[1], b_CutTree[2]: specular as b_TakeWood[1], the estimate odds of choose "(1) Strongly agree" or "(2) Agree", instead of "(5) Strongly disagree" increase as the frequency of $x = CutTree$ increase, meaning that more often we do this action, more we are likely to agree with the statement. But the ET let us see that the sign is not constant, but we have low probability that we can have negative value.
- b_CutTree[3]: the odds of choosing the neutral option over "(5) Strongly disagree", doesn't let us a precise interpretation of this coefficients. What I can say is that as neutral option, we can't stick with the interpretation of increase or decrease. In average is negative, and decrease respect the above two. We can say that more we Cut the Tree less we are likely to be Neutral.
- b_CutTree[4]: this parameters is always negative. The estimate odds of choose "(4) Disagree" instead of (5) decrease as the frequency of $x = CutTree$ increase, meaning that if we Cut Tree more often, we want to avoid as much a neutral or weak option, and be more rigid choosing a strong disagree.

In the end we can say that CutTree has an influence in choosing to agree with violet the nature. Influence that decrease more we consider only positive option. Looking instead on the paper we have a precise statement: *"... CutTree has the opposite association (negative) with Anti_Anthro. [..] In contrast, if a player intentionally cuts down the tree even when he/she has taken wood from the tree (chopping the tree more than three times), he/she is more likely to agree with the anthropocentric worldview".*

Relative Risk:

	1	2	3	4
b_Takewood	0.695245	0.534999	0.6965520	0.9584230
b_CutTree	1.411448	1.288833	0.9787548	0.7812867

Other possible insight are that looking at the plot below, we can see that the TakeWood coefficients (and in a slighter way also CutTree ones) has a sort of non linear behavior in the average value. We want now using the simple model seen in the first part of the report add a quadratic term in this feature and see how the new regression coefficient behave (also for CutTree).



Also, CutTree seems to behave well in the decreasing of the "Agreement", since more we Disagree with the statement less we want the influence of this specific covariate.

And TakeWood instead increase as we reach a "Disagreement".

Quadratic model in TakeWood

$$Y_i \sim N\left(\mu_i, \sigma^2\right)$$

$$\mu_i = \alpha + \beta_{TakeWood} \cdot TakeWood_i + \gamma_{TakeWood} \cdot TakeWood_i^2 + \beta_{CutTree} \cdot CutTree_i$$

here: $\gamma_{TakeWood} \sim N(0, 10)$.

```
# ------- MODEL JAGS (QUADRATIC IN TAKEWOOD) ---------- #

# list of parameters name
parameters <- c("a_Anti_Anthro",
                "b_TakeWood", "gamma_TakeWood",
                "b_CutTree")
# initial value
inits <- list(a_Anti_Anthro = 3, b_TakeWood = 0,
              gamma_TakeWood = 0, b_CutTree = 0)
initial.values <- list(inits)
```

```
# MCMC with jags
set.seed(123) # set seed for reproducibility
model2 <- jags(data = data,
               inits = initial.values,
               parameters.to.save = parameters,
               model.file = "model_jags_quadratic1.txt",
               n.burnin = 2000,
               n.chains = 1, n.thin = 1,
               n.iter = 5000)
```

	mean	sd	2.5%	25%	50%	75%	97.5%
a_Anti_Anthro	4.0432444	0.6884834	2.7079587	3.5578431	4.0465093	4.5149072	5.3618649
b_CutTree	-0.1548686	0.0650598	-0.2840622	-0.1994378	-0.1542896	-0.1102847	-0.0265720
b_TakeWood	-0.3993474	0.4696160	-1.2905432	-0.7166663	-0.3973364	-0.0738278	0.5206827
gamma_TakeWood	0.1113163	0.0780961	-0.0409401	0.0567528	0.1112861	0.1649830	0.2636823

As we can see the γ parameter include the zero only with the 95% credible interval, while in average or looking at a lower credible interval the value zero isn't include. What we can say is that there is a sort of non linear dependence with the TakeWood covariate. Also the variability around the estimation of the linear regression coefficient of TakeWood increase by a lot. ' The result doesn't change much, in fact we can see that the parabola (using the average estimation) is convex (increase as $TakeWood$ increase).

Quadratic model in CutTree

$$Y_i \sim N\left(\mu_i, \sigma^2\right)$$

$$\mu_i = \alpha + \beta_{TakeWood} \cdot TakeWood_i + \beta_{CutTree} \cdot CutTree_i + \gamma_{CutTree} \cdot CutTree^2$$

here: $\gamma_{TakeWood} \sim N(0, 10)$.

```
# ------- MODEL JAGS (QUADRATIC IN CutTree) ---------- #

# list of parameters name
parameters <- c("a_Anti_Anthro","b_TakeWood",
                "b_CutTree", "gamma_CutTree")
# initial value
inits <- list(a_Anti_Anthro = 3, b_TakeWood = 0,
              gamma_CutTree = 0, b_CutTree = 0)
initial.values <- list(inits)


# MCMC with jags
set.seed(123) # set seed for reproducibility
model3 <- jags(data = data,
               inits = initial.values,
```

```
                parameters.to.save = parameters,
                model.file = "model_jags_quadratic2.txt",
                n.burnin = 2000,
                n.chains = 1, n.thin = 1,
                n.iter = 5000)
```

	mean	sd	2.5%	25%	50%	75%	97.5%
a_Anti_Anthro	2.9708393	0.4881139	2.0243033	2.6264635	2.9735334	3.3069402	3.9042720
b_CutTree	-0.0450864	0.3467397	-0.6970248	-0.2857489	-0.0512616	0.1889748	0.6284000
b_TakeWood	0.2694664	0.0742277	0.1224544	0.2188306	0.2702580	0.3189072	0.4120892
gamma_CutTree	-0.0197564	0.0628399	-0.1439626	-0.0631485	-0.0200856	0.0245242	0.1003502

On the other end, here the quadratic term is well centered around the zero, and also has a very smaller value, meaning that the effect of the quadratic term cannot be negligible, but doesn't have a strong impact on the model choice. Also Here the variability around the linear term increase.

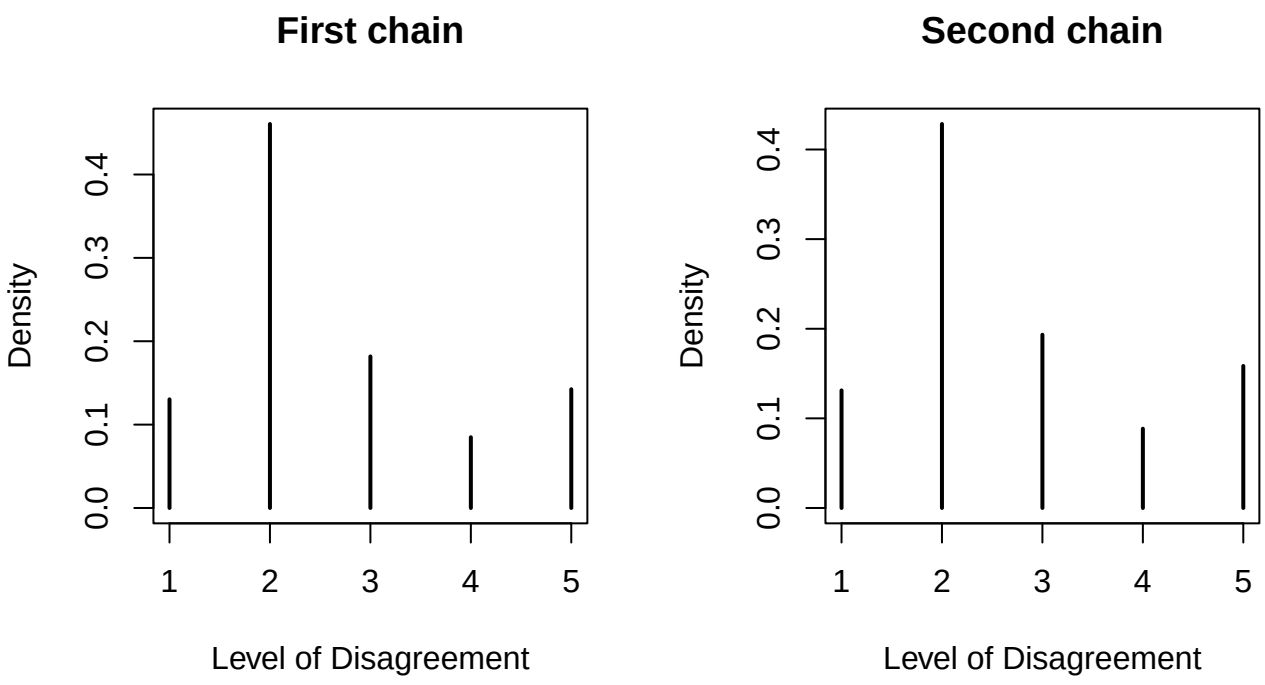
The result doesn't change much, in fact we can see that the parabola (using the average estimation) is concave (decrease as $CutTree$ increase).

Linear model	QuadraticTakeWood	QuadraticCutTree
2153.721	2155.868	2153.752

The DIC score are slightly the same. We don't see an improvement.

Prediction

Lets predict the a possible answer to the survey of a gamer whose rarely TakeWood, but very often CutTree.



We can see that as we can image, the user will Agree (category 2) with the statement: "Humans were meant to rule over the rest of nature".

The frequentist approach

From a frequentistic perspective we have an in-built machine learning model in the *nnet* package which implement the multinomial logistic regression.

First of all we need to train the model, and possibly tuning it. So we need to split the already small dataset in train and test data, instead of using all the data as in the Bayesian case. We will use package *caret* for this task.

```
# take a copy of the data
data <- dat

# factorize the response for the model
data$Anti_Anthro <- as.factor(data$Anti_Anthro)
# impose as reference level the category 5
data$out <- relevel(data$Anti_Anthro,ref = "5")

# Train-Test split
set.seed(1234) # set seed for reproducibility
idx.tr = createDataPartition(y = data$out,
                             p = .70, list = FALSE)

dat.tr = data[ idx.tr, ]
dat.te = data[-idx.tr, ]
```

Let's look now our data:

```
table(data$Anti_Anthro)
```

```
##
##   1   2   3   4   5
##  48 115 113 154 210
```

```
table(dat.tr$Anti_Anthro)
```

```
##
##   1   2   3   4   5
##  34  81  80 108 147
```

```
table(dat.te$Anti_Anthro)
```

```
##
##  1  2  3  4  5
## 14 34 33 46 63
```

As we can see the class are not balanced itself, this result in having lower number of value in each split.

Now lets tune the model and predict on the test set. We can extract the accuracy of the model with the corresponding confusion matrix.

```r
# Tuning parameter
trctrl <- trainControl(method = "repeatedcv", number = 10, repeats = 3)

set.seed(1234) # For reproducibility
multinom_fit <- train(out ~ TakeWood + CutTree,
                 data = dat.tr, method = "multinom",
                 trControl  = trctrl,
                 tuneLength = 15
)

# Prediction
test_pred <- predict(multinom_fit, newdata = dat.te[, -1])
```

```
## Accuracy
##      0.3
```

```
##            Reference
## Prediction  5  1  2  3  4
##          5 47 13 25 26 41
##          1  0  0  0  0  0
##          2  5  0  6  2  1
##          3  0  0  0  0  0
##          4 11  1  3  5  4
```

We can see that we mismatch a lot of categories due to have an unbalanced dataset.

In the end we can do the same prediction as before and hope we land on the same results.

```r
# prediction on new test point
TakeWood <- 1
CutTree <- 4
new.data <- data.frame(TakeWood, CutTree)

test_pred <- predict(multinom_fit, newdata = new.data, type="prob")
```

```
##           5         1         2         3         4
## 1 0.0939744 0.1048438 0.4875468 0.1968004 0.1168347
```

We can see that the highest probability is obtained by the class 2.

Also looking at the regression coefficients in log-odds and exponential form:

```
## [1] "Log-odd coeff"
```

```
##                      1          2          3          4
## (Intercept) -0.7917426  1.0785364  0.5279243  1.0043505
## TakeWood    -0.4937004 -0.8137731 -0.4901348 -0.3011661
## CutTree      0.3487230  0.3454001  0.1753445 -0.1213618
```

```
## [1] "Exp coeff risk"
```

```
##                     1         2         3         4
## (Intercept) 0.4530546 2.9403728 1.6954095 2.7301335
## TakeWood    0.6103636 0.4431828 0.6125438 0.7399548
## CutTree     1.4172566 1.4125550 1.1916567 0.8857135
```

The result as quite the same as the Bayesian analysis.

References

1. Vuong Q-H et al (2021) A multinational data set of game players' behaviors in a virtual world and environmental perceptions.
2. Vuong Q-H, La V-P (2019) The bayesvl r package. User guide v0.8.
3. Agresti A (1990) Categorical data analysis, Wiley.
4. Robert C, Ntzoufras I Bayesian modeling using WinBUGS.
5. Plummer M (2017) JAGS version 4.3.0 user manual.