Bayesian Hierarchical Analysis of Esophageal Cancer Risk Factors: Evaluating the Role of Age, Tobacco, and Alcohol Consumption

2023-11-29

Introduction

Esophageal cancer is cancer in the esophagus – the food pipe connecting the throat and the stomach. In recent years, esophageal cancer has been among the most common types of cancer: in 2020, it ranked 8th in the most diagnosed type of cancer globally, with 604,000 new cases. It is also the 6th most deadly cancer killing over 544,000 people in 2020.

Smoking and alcohol use have long been known as two major risk factors associated with esophageal cancer, and they are among the few risk factors that can be controlled or changed by the subject. The goal of this project is to utilize Bayesian modeling to the data from a case-control study of esophageal cancer in hopes of drawing a relation between a person's level of tobacco and alcohol intake and their chance of developing esophageal cancer. Our modelling will also account for the patients' age as that is an important factor in the development of cancers.

Data set

Our data set was created by Tuyns AJ, Péquignot G, Jensen OM. in a paper *Esophageal cancer in Ille-et-Vilaine about levels of alcohol and tobacco consumption. Risks are multiplying. (1977).*

It was also used in a statistical book by *Breslow, N. E. and Day, N. E. (1980) Statistical Methods in Cancer Research. Volume 1: The Analysis of Case-Control Studies. IARC Lyon / Oxford University Press.* In the book, the data set was used to showcase statistical methods and techniques. There was also logistic regression performed on the data set. However, no Bayesian statistics was used as the book focuses on multiple regression models.

```
##   agegp     alcgp    tobgp ncases ncontrols total prob
## 1 25-34 0-39g/day 0-9g/day      0        40    40    0
## 2 25-34 0-39g/day    10-19      0        10    10    0
## 3 25-34 0-39g/day    20-29      0         6     6    0
## 4 25-34 0-39g/day      30+      0         5     5    0
## 5 25-34     40-79 0-9g/day      0        27    27    0
## 6 25-34     40-79    10-19      0         7     7    0
```

The data set contains the following columns:

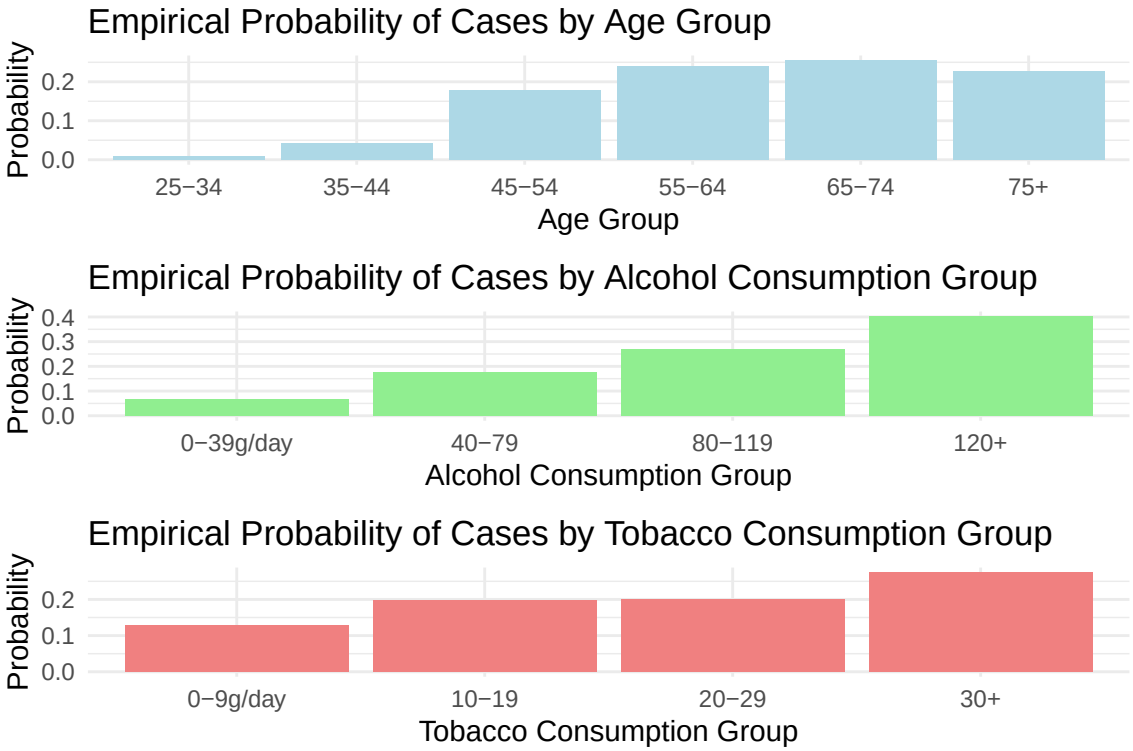
1. **`agegp`**: Age group
2. **`alcgp`**: Alcohol consumption group
3. **`tobgp`**: Tobacco consumption group
4. **`ncases`**: Number of cases
5. **`ncontrols`**: Number of controls
6. **`total`**: Total number of subjects
7. **`prob`**: Empirical probability

Visualization

Empirical Probabilities by Age Group: This chart shows the empirical probabilities across different age groups.

Empirical Probabilities by Alcohol Consumption Group: This chart displays the empirical probabilities in relation to different levels of alcohol consumption.

Empirical Probabilities by Tobacco Consumption Group: Similar to the alcohol consumption chart, this one shows the empirical probabilities across various levels of tobacco consumption.



Data pre-processing

Transform the original data set into a new data set suitable for Bernoulli fit:

1. `ID`: ID of the patient
2. `agegp`: Age group
3. `alcgp`: Alcohol consumption group
4. `tobgp`: Tobacco consumption group
5. `cancer`: Binary indicator whether there is presence in the patient

```
##   ID   age       tob      alc cancer
## 1  1 25-34 0-39g/day 0-9g/day      1
## 2  2 25-34 0-39g/day 0-9g/day      1
## 3  3 25-34 0-39g/day 0-9g/day      0
## 4  4 25-34 0-39g/day 0-9g/day      0
## 5  5 25-34 0-39g/day 0-9g/day      0
## 6  6 25-34 0-39g/day 0-9g/day      0
```

Split the Data

We will split the Data into Training and Test Sets using the function split(). The `SplitRatio = 0.7` argument of the function indicates that approximately 70% of the data should be allocated to the training set, and the remaining 30% in the test set.

Motivation

The motivation of the project is to model the probability of having cancer based on several explanatory variables: age, tobacco usage, and alcohol consumption. The response variable (cancer occurrence) is binary, where '1' indicates the presence of cancer and '0' indicates its absence.

Mathematical Notation:

The logistic regression model can be expressed as follows:

$$\text{logit}(P(Y=1)) = \log\left(\frac{P(Y=1)}{1-P(Y=1)}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k$$

Where: - $P(Y=1)$ is the probability of having cancer.

- β_0 is the intercept. - $\beta_1, \beta_2, \cdots, \beta_k$ are the coefficients for the explanatory variables $X_1, X_2, \cdots, X_k$ (age, tobacco, and alcohol consumption in your case). These parameter estimate will show the change in the log-odds of the outcome when the variable changes from its reference level (usually 0) to the other level. In practical terms, this parameter tells us the difference in the log-odds of the probability of having cancer.

Pooled ordinal logistic regression model

Priors and Likelihood:

1. **Priors**:

- For the β coefficients, a common choice is to use normal distributions as priors due to their mathematical convenience and the central limit theorem. For example, $\beta_i \sim \mathcal{N}(0, \sigma^2)$, where σ^2 is the variance. However, we can also use the flat priors suggested by BRMS and then later test the prior sensitivity of the model.
- According to American Cancer Society, Age, Tobacco, and Alcohol are ones of the risk factors [2]. A risk factor is 'anything that increases your chance of getting the esophageal cancer'. But having a risk factor, or even many, does not mean that you will get the cancer. Therefore, one possible choice for the priors of the coefficients is $\beta_i \sim \mathcal{N}(0.02, 10)$, with a close-to-zero positive mean and a very big standard deviation for it to be a flat priors, resulting in a posterior mainly influenced by the data.

```
#Examine the default priors suggested
get_prior(cancer ~ 1 + tob + alc + age,
             data = train_data,
             family = bernoulli("logit"))
```

```
##                 prior     class  coef group resp dpar nlpar lb ub       source
##                (flat)         b                                       default
##                (flat)         b age.C                           (vectorized)
##                (flat)         b age.L                           (vectorized)
##                (flat)         b age.Q                           (vectorized)
##                (flat)         b ageE4                           (vectorized)
##                (flat)         b ageE5                           (vectorized)
```

```
##                (flat)         b alc.C                                (vectorized)
##                (flat)         b alc.L                                (vectorized)
##                (flat)         b alc.Q                                (vectorized)
##                (flat)         b tob.C                                (vectorized)
##                (flat)         b tob.L                                (vectorized)
##                (flat)         b tob.Q                                (vectorized)
##  student_t(3, 0, 2.5) Intercept                                            default
```

2. **Likelihood**: The likelihood of observing the data given the parameters is modeled by a binomial distribution (for binary outcomes), which in the case of logistic regression simplifies to a product of Bernoulli trials.

BRMS code

```
# Defining the ordinal logistic model
model <- brm(cancer ~ 1 + tob + alc + age,
             data = train_data,
             family = bernoulli("logit"),
             prior = c(prior(normal(0.02, 10), class = "b"),
                       prior(normal(0, 10), class = "Intercept")),
             chains=4,
             file="model1",
             backend="cmdstanr",
             cores=4)
```

The model uses 4 chains, each with 2000 iterations. The warm-up length is 1000. This are mostly default settings but, as we show below, the MCMC chains converge well.

Convergence diagnostic

```
##  Family: bernoulli
##   Links: mu = logit
## Formula: cancer ~ 1 + tob + alc + age
##    Data: train_data (Number of observations: 863)
##   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup draws = 4000
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept    -0.98      0.11    -1.20    -0.76 1.00     7575     3399
## tob.L         1.16      0.20     0.76     1.55 1.00     7357     3212
## tob.Q        -0.19      0.19    -0.55     0.17 1.00     6915     3091
## tob.C        -0.13      0.17    -0.47     0.22 1.00     6225     3451
## alc.L         0.93      0.19     0.56     1.30 1.00     6425     3025
## alc.Q         0.05      0.19    -0.32     0.42 1.00     6666     3286
## alc.C         0.10      0.19    -0.26     0.47 1.00     7974     3097
## age.L         0.46      0.30    -0.14     1.05 1.00     6076     3049
## age.Q         0.14      0.28    -0.43     0.68 1.00     6818     3279
## age.C        -0.31      0.26    -0.81     0.20 1.00     5890     3478
## ageE4         0.15      0.22    -0.30     0.59 1.00     7354     3116
## ageE5         0.08      0.19    -0.28     0.45 1.00     7099     2876
##
## Draws were sampled using sample(hmc). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

From the summary we can learn that our model estimation converged well: $\hat{R} \sim 1.0$ and effective sample size greater than 100 times the number of chains (=4) for each parameter (Vehtari et al. 2021) [1].

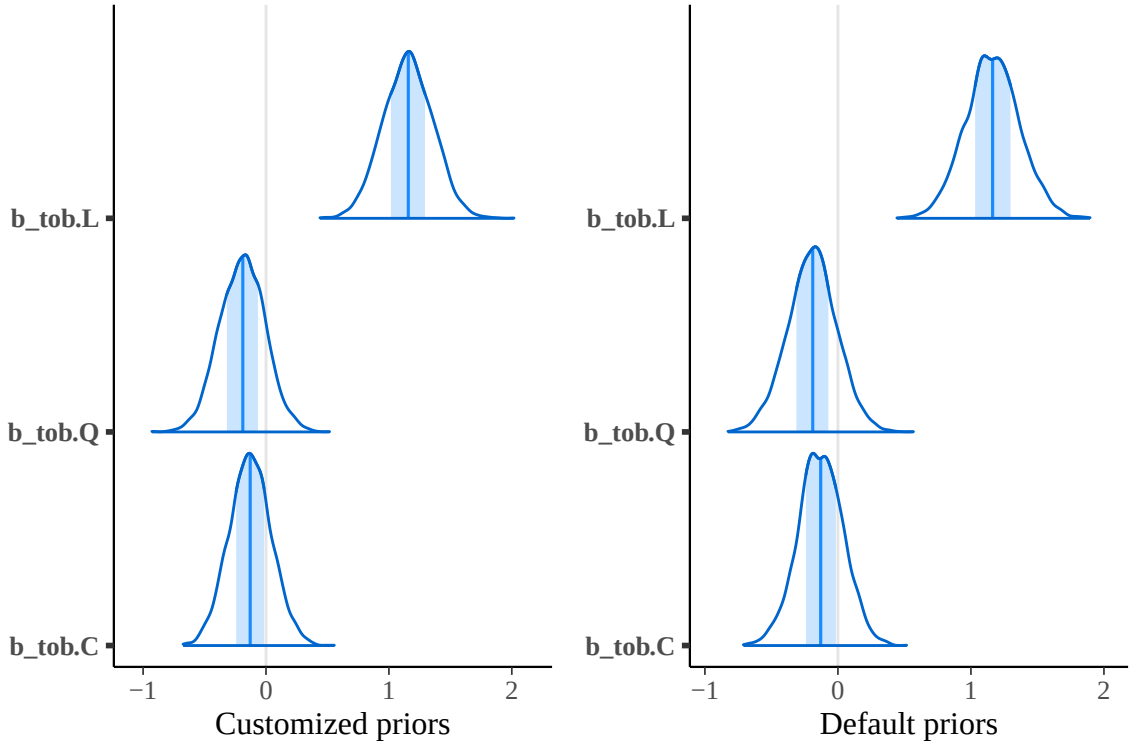
```
## Number of divergence of the transitions: 0
```

Zero divergences indicates that the posterior distribution has been explored by the HMC algorithm.

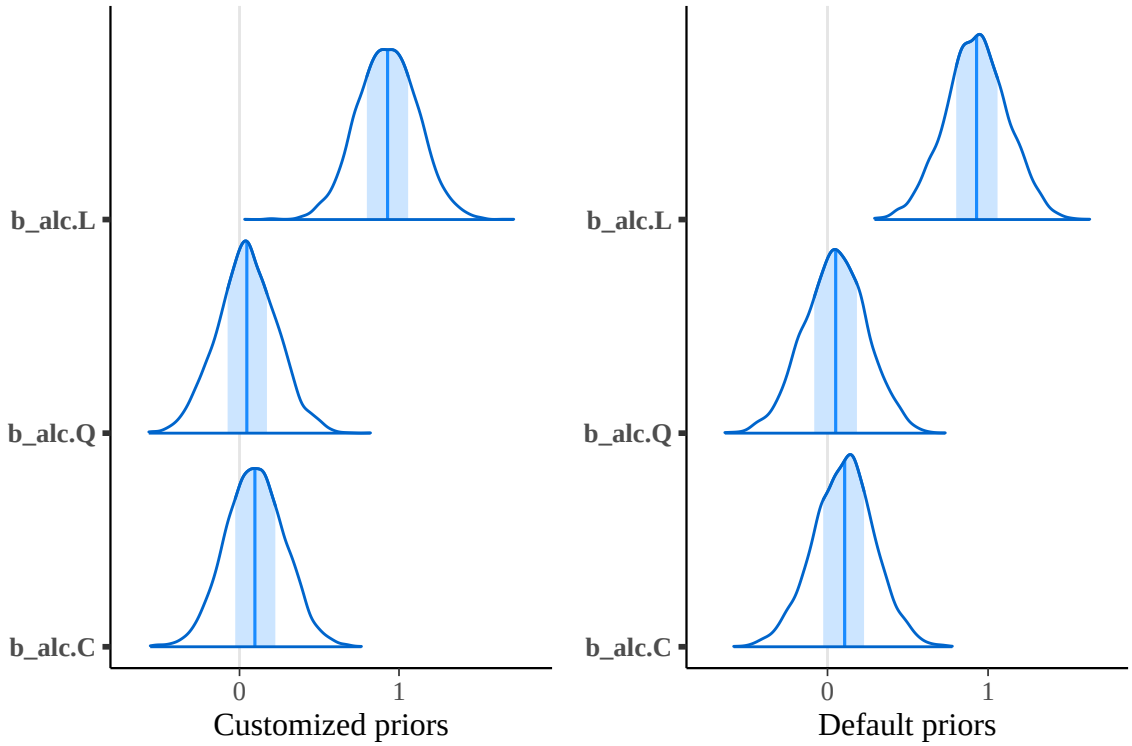
Sensitivity Analysis with respect to Priors

This time let use the default priors suggested by BRMS model.

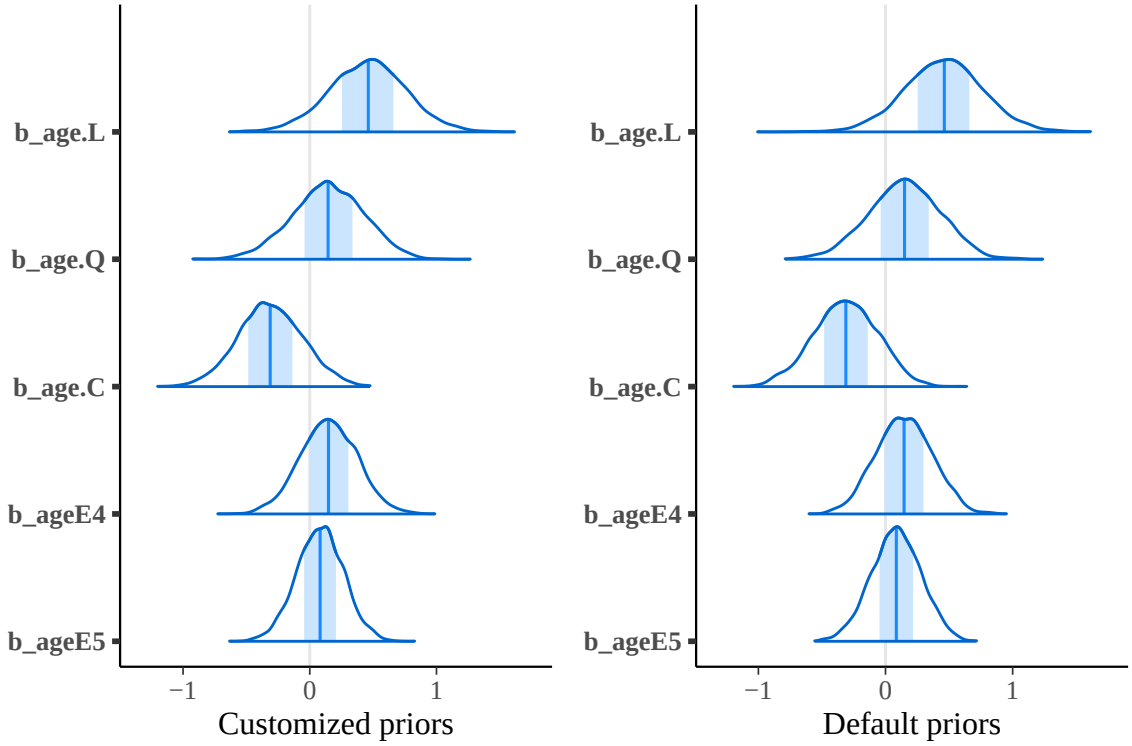
Posterior of Tobacco's coefficient parameters:



Posterior of Alcohol's coefficient parameters:



Posterior of Age's coefficient parameters:



In conclusion, we can say that the model is rather indifferent to prior choices as there are no considerable differences between the posteriors of the parameters.

Hierarchical logistic regression model

In the hierarchical model, we will assume that each patient will inherently have a distinct tendency to have cancer, accounting for patient-level variation. For example, depending on the patient's genes, the patient is either vulnerable or more resistant to cancer.

Priors and Likelihood:

1. **Priors**: We can reuse the priors in the Pooled Model as well as priors suggested by BRMS, then test the Model's sensitivity to priors.

```
#Examine the default priors suggested
get_prior(cancer ~ 1 + tob + alc + age + (1 | ID),
    data = train_data,
          family = bernoulli("logit"))
```

```
##                 prior     class      coef group resp dpar nlpar lb ub
##                (flat)         b
##                (flat)         b     age.C
##                (flat)         b     age.L
##                (flat)         b     age.Q
##                (flat)         b     ageE4
##                (flat)         b     ageE5
##                (flat)         b     alc.C
##                (flat)         b     alc.L
##                (flat)         b     alc.Q
##                (flat)         b     tob.C
##                (flat)         b     tob.L
##                (flat)         b     tob.Q
##  student_t(3, 0, 2.5) Intercept
##  student_t(3, 0, 2.5)        sd                                  0
##  student_t(3, 0, 2.5)        sd              ID                  0
##  student_t(3, 0, 2.5)        sd Intercept    ID                  0
##        source
##       default
##  (vectorized)
##  (vectorized)
##  (vectorized)
##  (vectorized)
##  (vectorized)
##  (vectorized)
##  (vectorized)
##  (vectorized)
##  (vectorized)
##  (vectorized)
##  (vectorized)
##  (vectorized)
##       default
##       default
##  (vectorized)
##  (vectorized)
```

2. **Likelihood**: We will reuse the Bernoulli family as in the Pooled Model.

BRMS code

```
# Defining the ordinal logistic model
model2 <- brm(cancer ~ 1 + tob + alc + age + (1 | ID),
              data = train_data,
              family = bernoulli("logit"),
              prior = c(prior(normal(0.02, 10), class = "b"),
                        prior(normal(0, 10), class = "Intercept")),
              chains=4,
              file="model2",
              backend="cmdstanr",
              cores=4)
```

The model uses 4 chains, each with 2000 iterations. The warm-up length is 1000. This are mostly default settings but, as we show below, the MCMC chains converge well.

Convergence diagnostic

```
##  Family: bernoulli
##   Links: mu = logit
## Formula: cancer ~ 1 + tob + alc + age + (1 | ID)
##    Data: train_data (Number of observations: 863)
##   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup draws = 4000
##
## Group-Level Effects:
## ~ID (Number of levels: 863)
##               Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(Intercept)    22.70      5.70    12.56    34.52 1.00     1604     2003
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept   -13.23      3.62   -21.21    -7.09 1.00     1629     2531
## tob.L        14.27      4.00     7.39    23.15 1.00     1875     2366
## tob.Q        -2.20      2.46    -7.52     2.31 1.00     1844     2450
## tob.C        -1.79      2.32    -6.65     2.57 1.00     2098     2513
## alc.L        11.52      3.57     5.58    19.36 1.00     1738     2347
## alc.Q         0.37      2.43    -4.57     5.32 1.00     2048     2398
## alc.C         1.22      2.42    -3.49     6.30 1.00     1918     2273
## age.L         4.97      3.67    -1.93    12.76 1.00     2302     2708
## age.Q         1.47      3.46    -5.51     8.35 1.00     2147     2653
## age.C        -4.11      3.23   -10.97     1.75 1.00     2302     2341
## ageE4         1.63      2.76    -3.88     7.10 1.00     2041     2612
## ageE5         0.95      2.39    -3.83     5.84 1.00     2225     2545
##
## Draws were sampled using sample(hmc). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

From the summary we can learn that our model estimation converged well: $\hat{R} \sim 1.00$ and effective sample size greater than 100 times the number of chains (=4) for each parameter (Vehtari et al. 2021) [1].

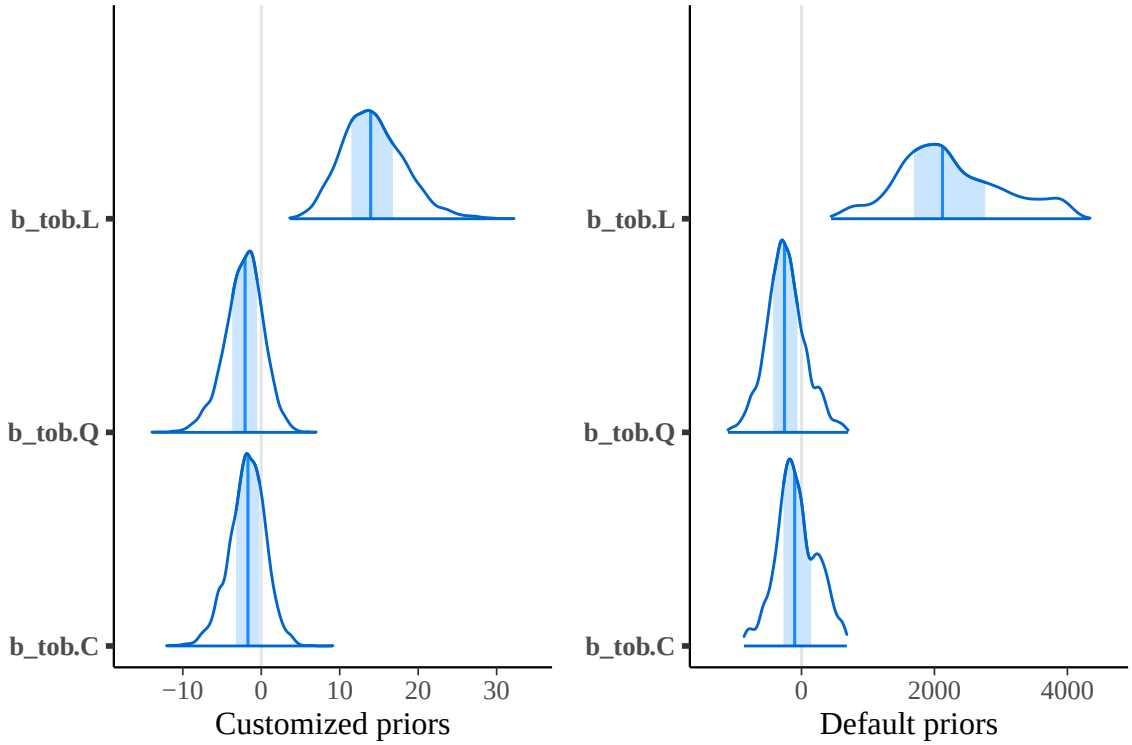
```
## Number of divergence of the transitions: 0
```

Zero divergences indicates that the posterior distribution has been explored by the HMC algorithm.

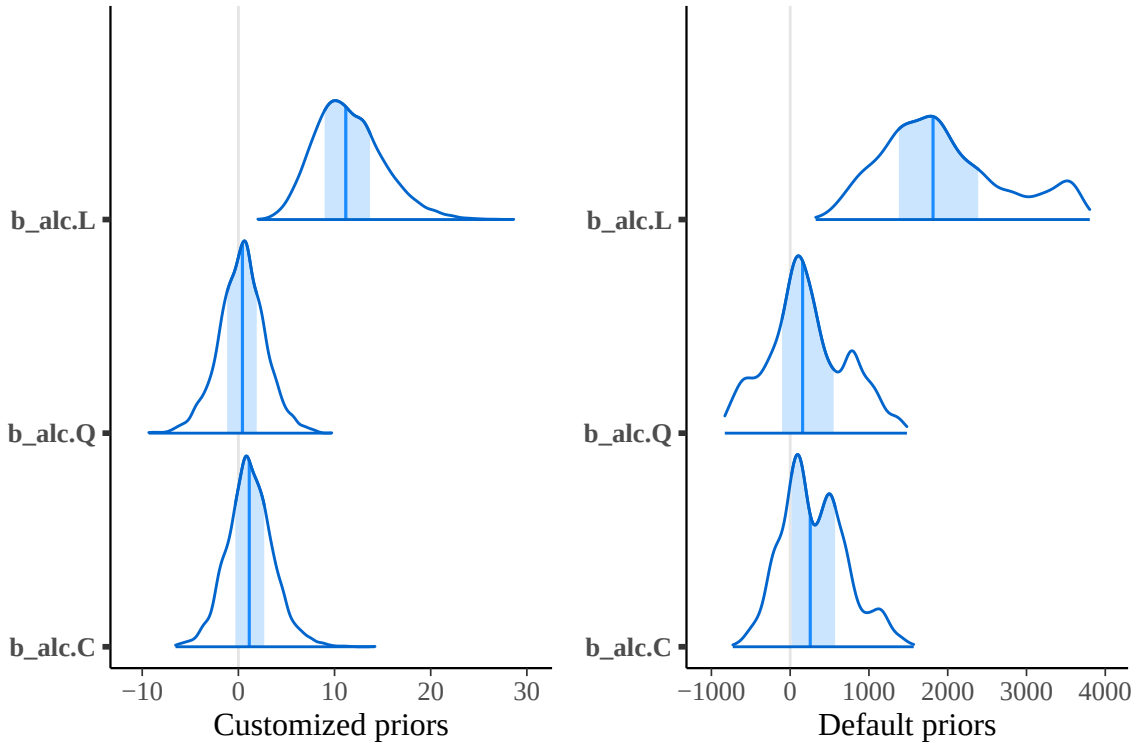
Sensitivity Analysis with respect to Priors

This time let use the default priors suggested by BRMS model.

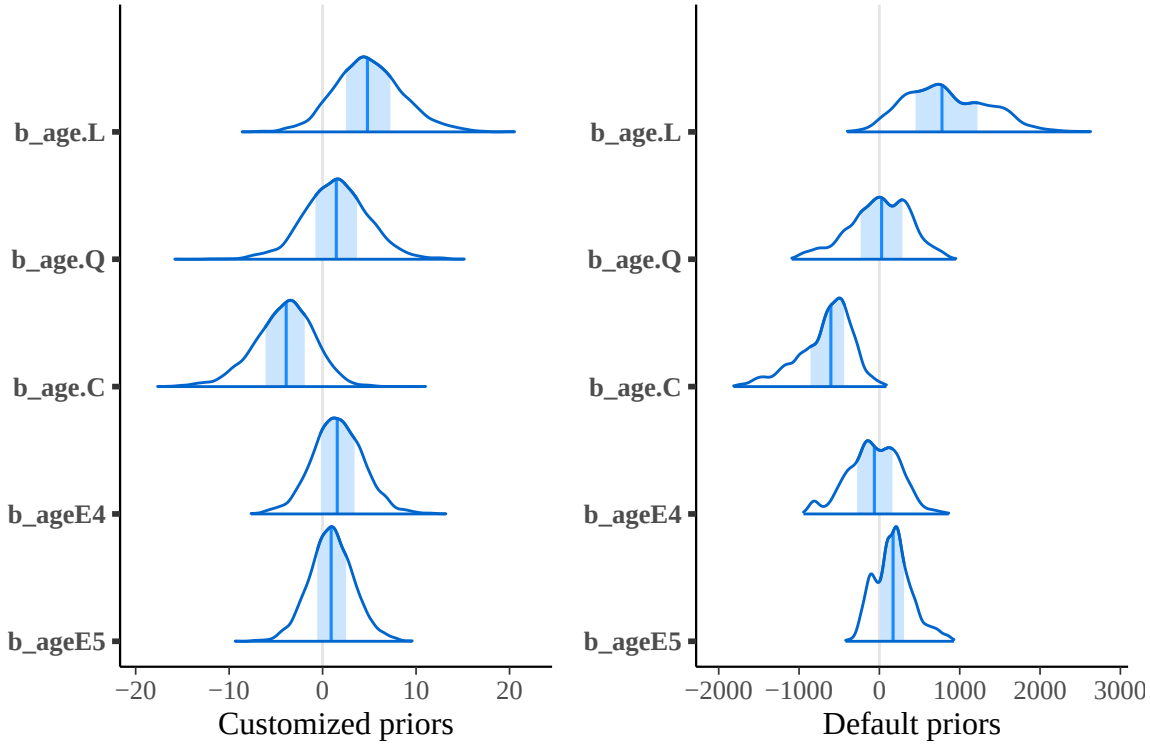
Posterior of Tobacco's coefficient parameters:



Posterior of Alcohol's coefficient parameters:



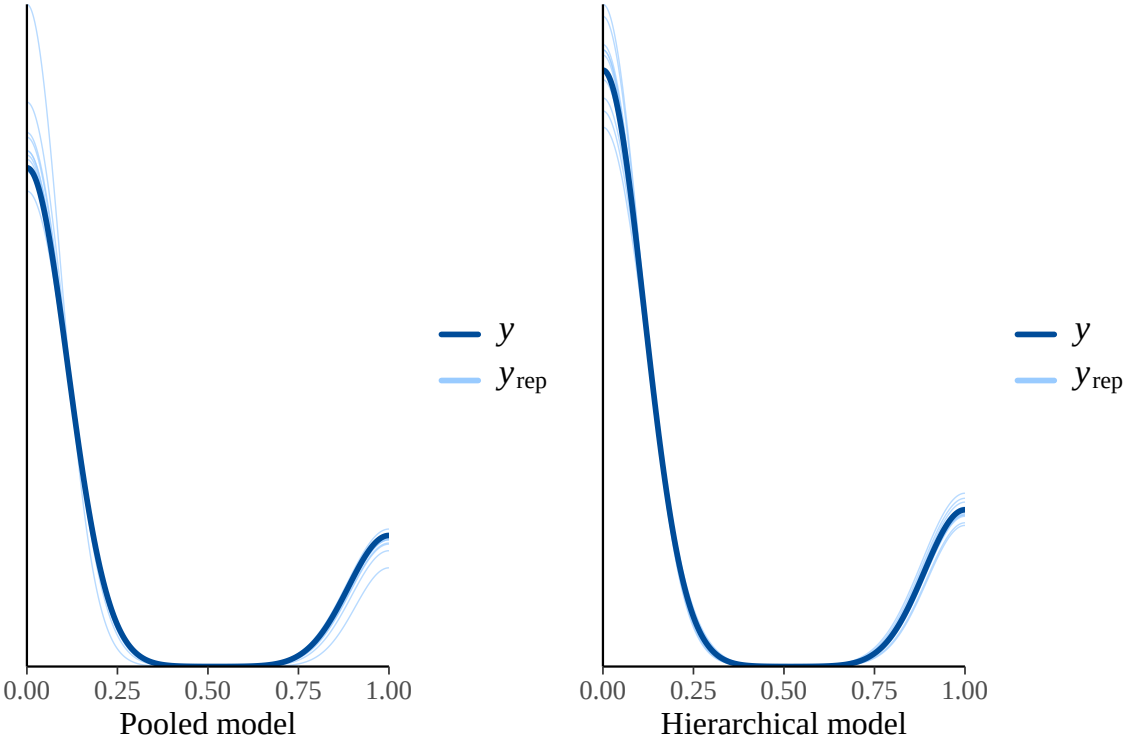
Posterior of Age's coefficient parameters:

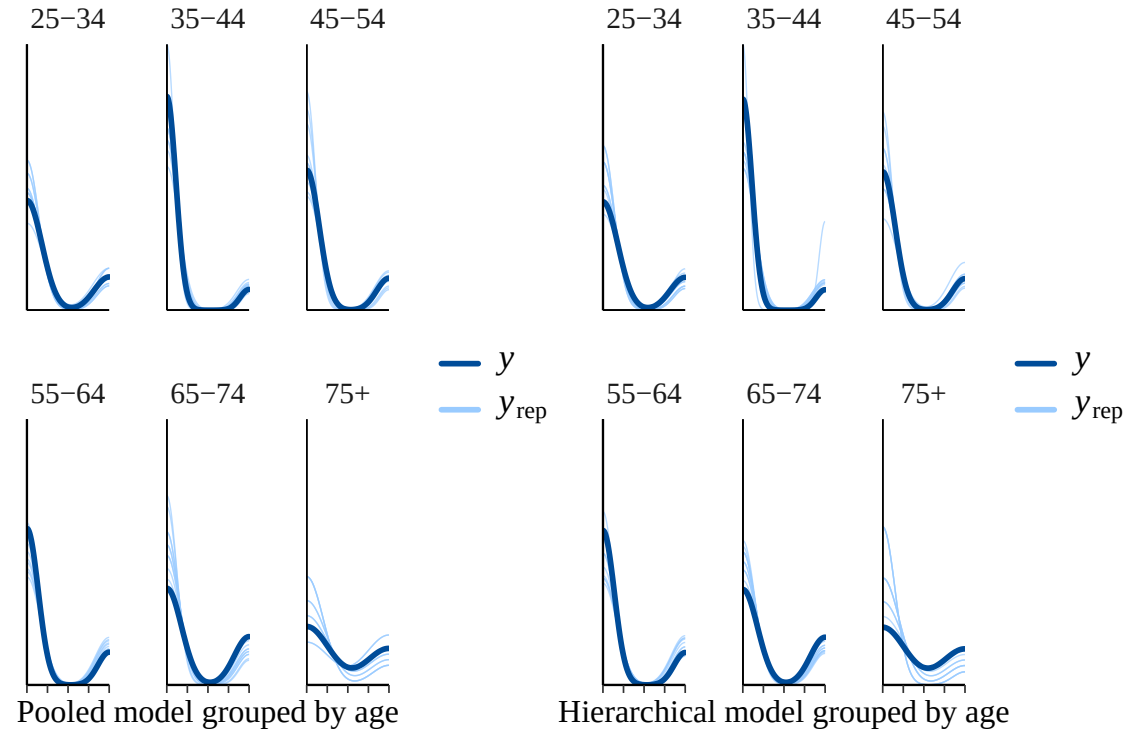


In conclusion, we can say that the model is rather sensitive to prior choices as there are some considerable differences between the posteriors of the parameters.

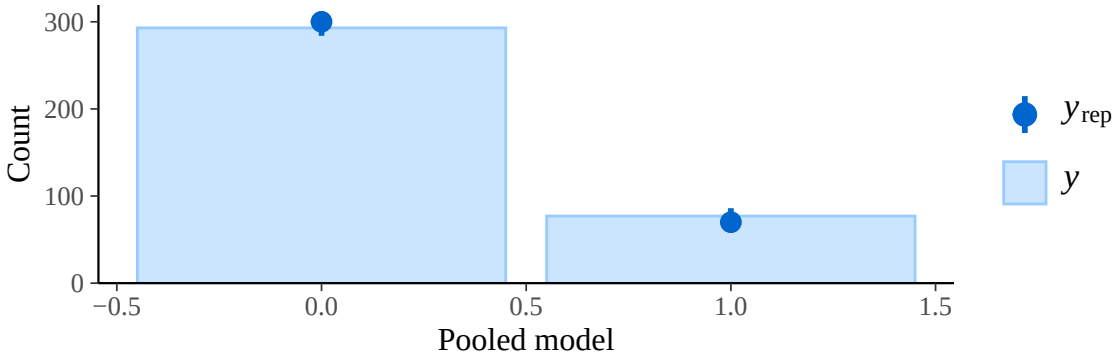
Posterior predictive checks

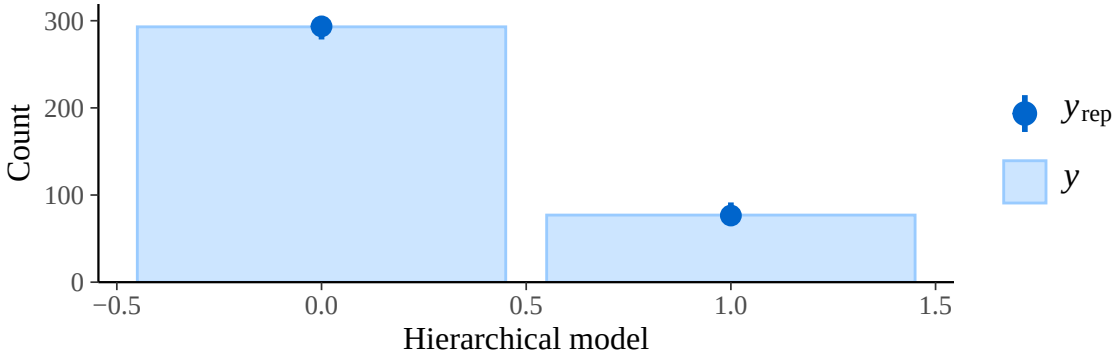
These first two plots are used for visualizing the overlay of posterior predictive densities on top of the observed data. We placed both models next to each other to allow for easier model comparison.

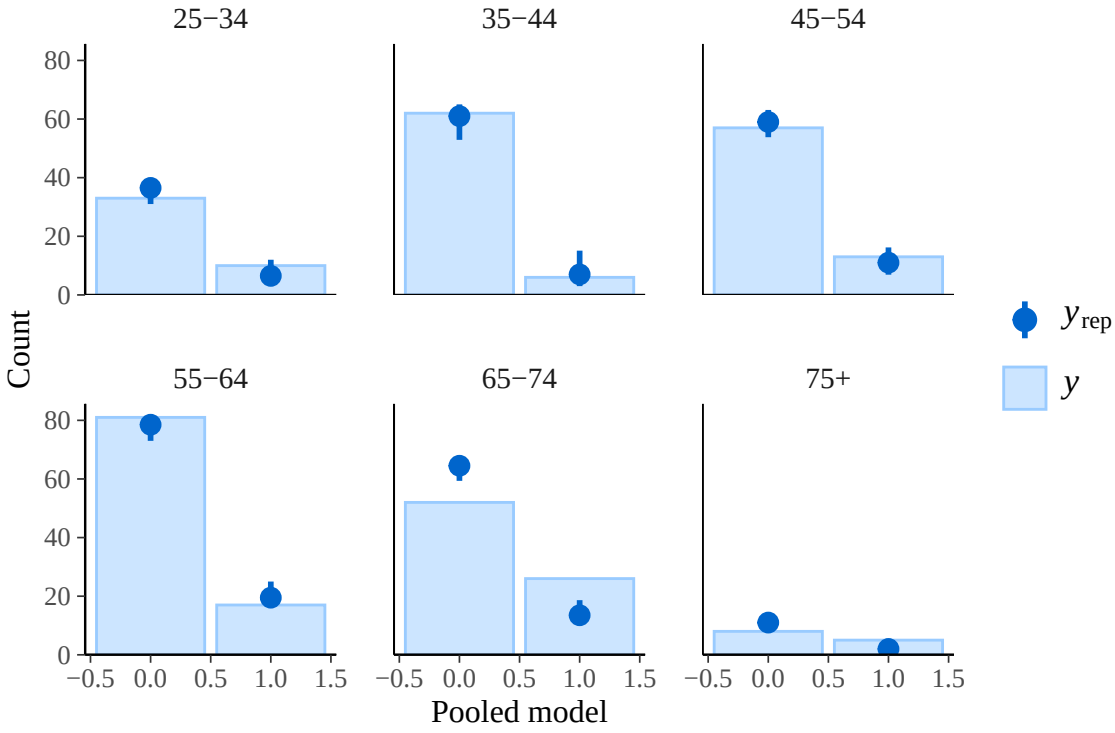


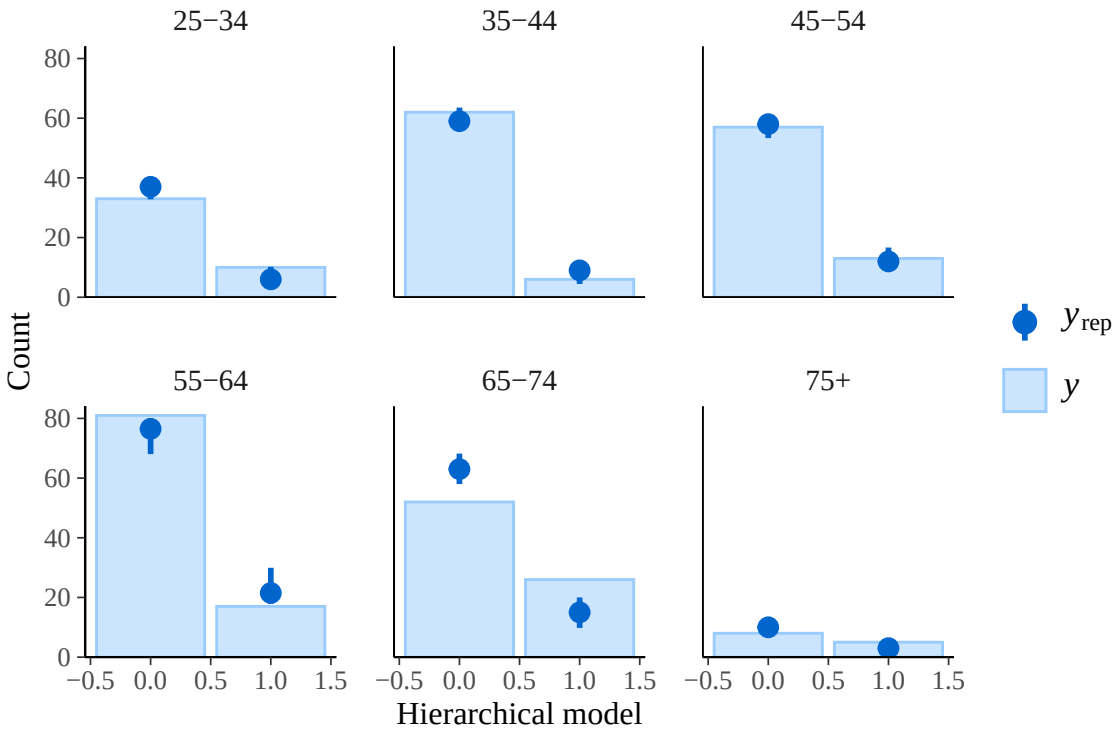


The next three plots displays two bars representing the observed counts of the two outcomes, with overlaid bars from the simulated data.



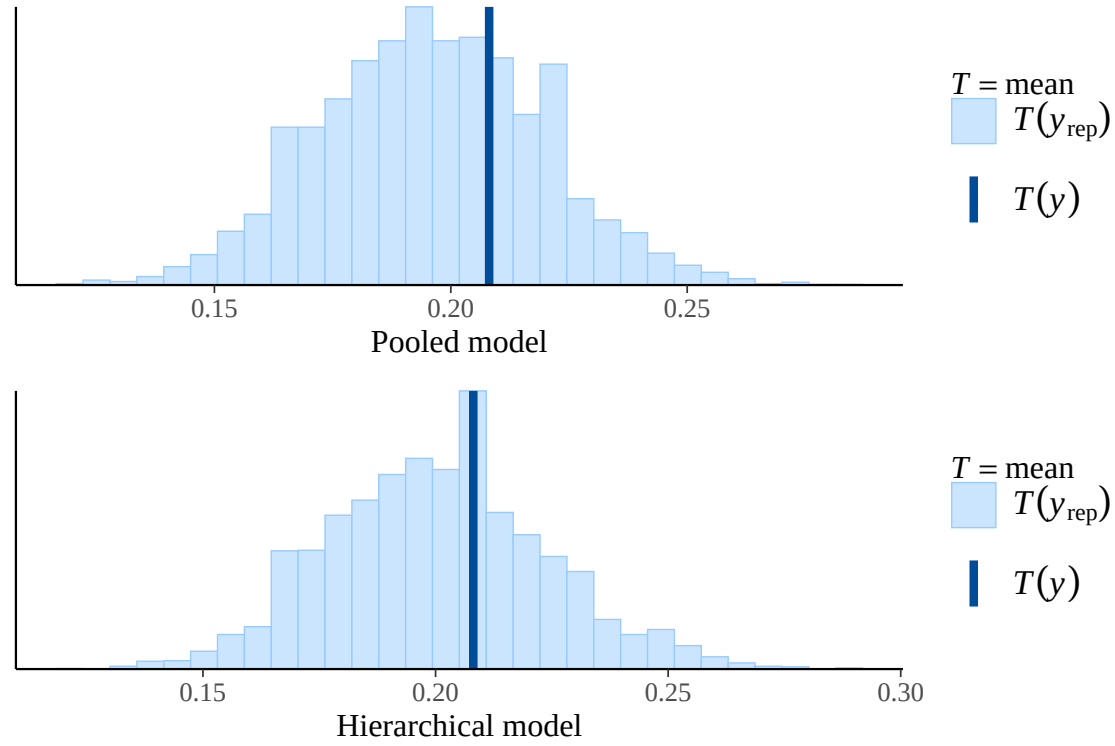




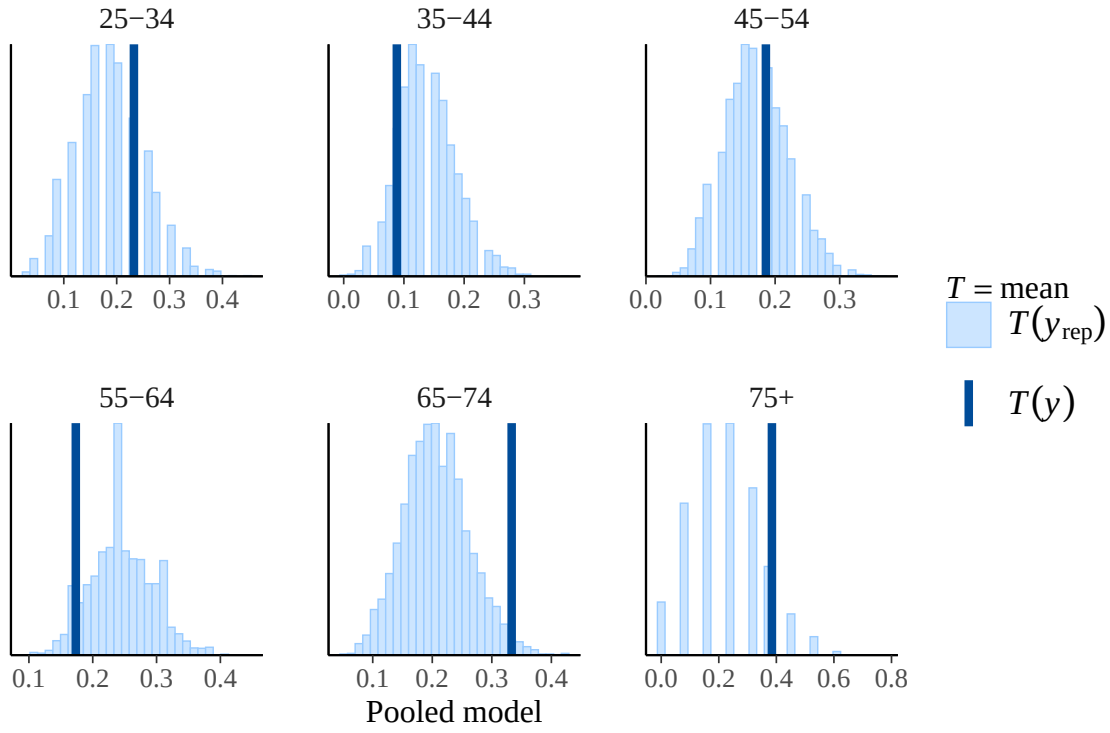


The final three plots show the observed mean statistic and a distribution of the statistic computed from the simulated data set.

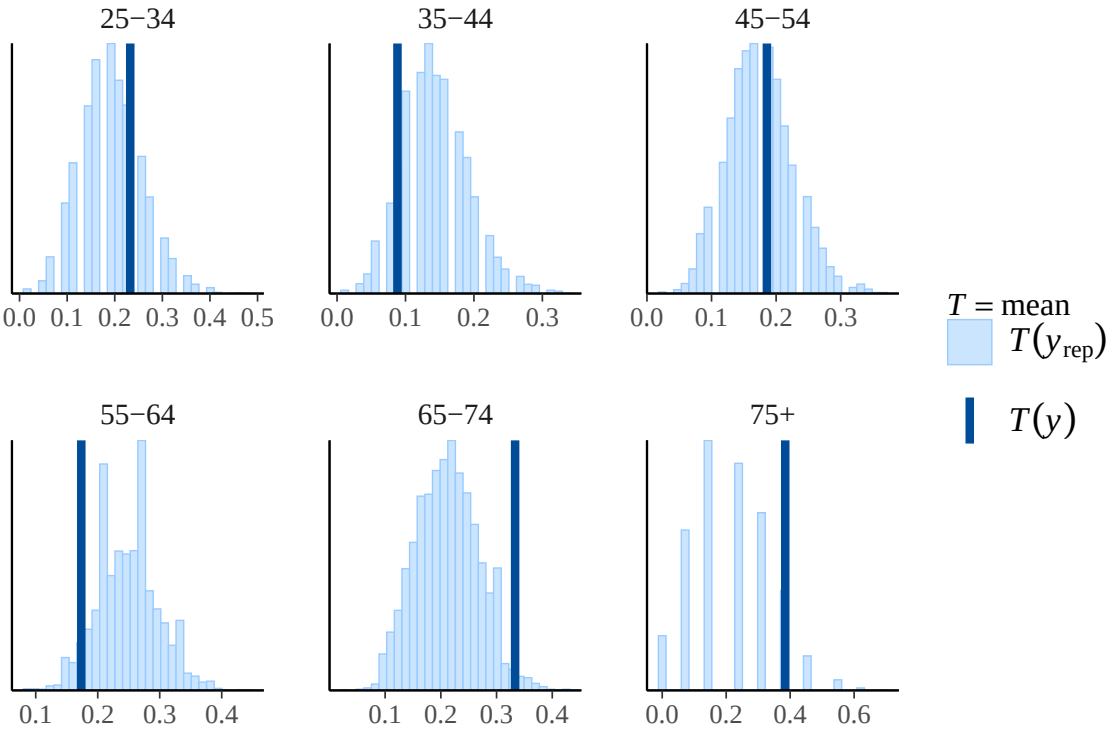
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
## Using all posterior draws for ppc type 'stat_grouped' by default.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Conclusion: We can see that the predicted data points following log-Bernoulli distribution has a similar distribution to the observed data, as both are bi-modal. However, from the grouped posterior, we can see that the predictions are not really good for some age group.

Predictive performance assessments

Now we will make predictions on the Test Set and then construct a confusion matrix by comparing predictions to the actual outcomes in the test set.

The Confusion Matrix:

```
## Pooled model:
```

```
##           Actual
## Predicted   0   1
##         0 287  73
##         1   6   4
```

```
## Hierarchical model:
```

```
##           Actual
## Predicted   0   1
##         0 287  74
##         1   6   3
```

Then we will evaluate the classification accuracy of both models using some of the common metrics: Accuracy, Precision, Recall, and F1 Score.

- **Accuracy**: Accuracy measures how often a classifier makes the correct prediction. It is the ratio of the number of correct predictions to the total number of predictions.
- **Precision**: Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. It's a measure of a classifier's exactness. High precision relates to a low rate of false positives, and it is particularly important in cases where the cost of a false positive is high.
- **Recall (or Sensitivity)**: Recall is the ratio of correctly predicted positive observations to all observations in the actual class. It's a measure of a classifier's completeness. High recall relates to a low rate of false negatives, and it is important in cases where the cost of a false negative is high.
- **F1 Score**: The F1 Score is the weighted average of Precision and Recall. It's a measure of a test's accuracy and considers both the precision and the recall. This is useful when seeking a balance between Precision and Recall, especially if there is an uneven class distribution.

In medical diagnostics, these metrics play crucial roles in evaluating the performance of diagnostic tests or predictive models. However, for life-threatening diseases such as cancer, **recall (sensitivity)** is often prioritized to ensure that as many true cases as possible are identified. Missing a diagnosis in such cases could be fatal.

```
## Pooled model:
```

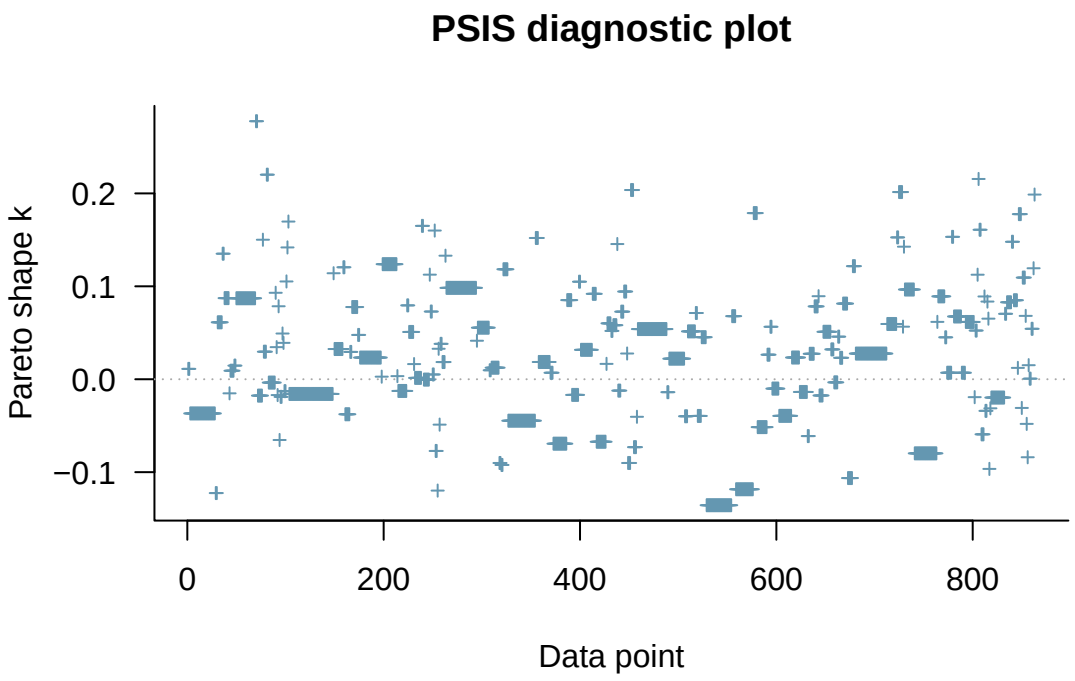
```
## Accuracy:    0.8
## Precision:   0.4
## Recall:      0.1
## F1 Score:    0.1
```

```
## Hierarchical model:
```

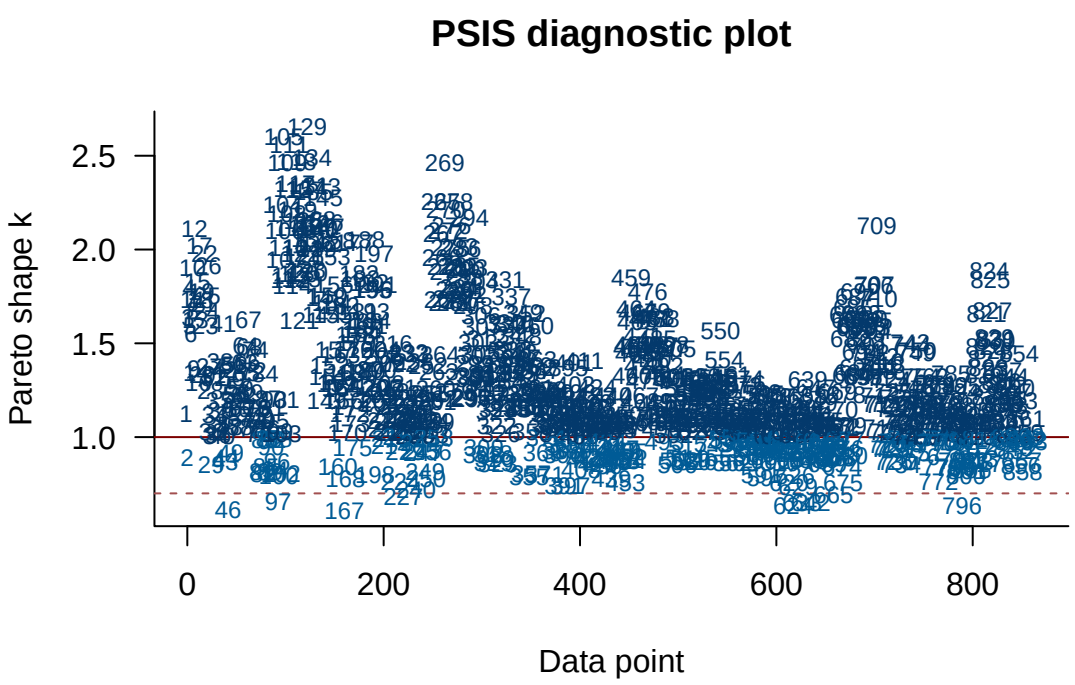
```
## Accuracy:    0.8
## Precision:   0.3
## Recall:      0.0
## F1 Score:    0.1
```

- As we can see the performances of both models are practically the same in terms of classification accuracy.
- Recall (Sensitivity) scores of both models are pretty bad. Hence, the model should be improved before being deployed.

Model comparisons using ELPD



All Pareto $\hat{k}$ are smaller than 0.5. The distribution of raw importance rations has finite variance.



We can see that almost all data points have Pareto $\hat{k}$ bigger than 0.7. It is due to our model having only one ID observation for each patient meaning that LOO cross validation does not provide a big insight.

```
##        elpd_diff se_diff
## model2    0.0       0.0
## model  -285.9      12.9
```

The loo comparison indicates that the hierarchical model is better than the pooled logistic regression. However, looking at posterior predictive checks, we can clearly see that is not the case. Leave-one-out is faulty in this case, as the hierarchical model has 1 observation per ID. To solve the problem with loo of the hierarchical model we would need to redo the ELPD 800-900 times or acquire more data about each patient.

Therefore, based on the posterior predictive checks and classification accuracy, pooled logistic regression is a better, more suitable model.

Possible improvements

- Both of the models have quite few false negatives when we try to predict the cancer on test data. This is not desired in cancer diagnosis. Therefore, the model should be improved to try and limit the number of false negatives. One possible solution to that is gathering more data, as our data set is quite small and the rate of cancer also is not very high.
- Gather new data, our data set is quite limited right now and if we could gather more information about each patient it would help our analysis a lot. For example getting the tobacco, alcohol consumption and age of each patient would be helpful. This would allow as to introduce more complicated models that could fit and explain the data better.

Conclusion

- Probability of getting esophageal cancer in each age group. How significant is the impact of the alcohol and tobacco as risk factors in being diagnosed with the cancer.
- The tobacco and alcohol are risk factors and not direct causes of the cancer. This with the low probability of getting the cancer, makes it difficult to accurately predict that somebody will develop the disease.

Reflections

Throughout the project we managed to deepen our understanding of Bayesian data analysis, as well as improve in many technical aspects of the analysis. Our skills in Stan, R or many different libraries that are associated with them excelled by a lot. However, everything did not go smoothly. We definitely learned how to work under pressure as we needed to change our models quite a bit in the final week of the project work due to some wrong assumptions. Moreover, our pick for data set was also quite flawed and did not allow us to express all of the ideas we had for the project.

References

[1]. Vehtari, Aki, Andrew Gelman, Daniel Simpson, Bob Carpenter, and Paul-Christian Bürkner. 2021. "Rank-Normalization, Folding, and Localization: An Improved $\widehat{R}$ for Assessing Convergence of Mcmc." Bayesian Analysis. https://doi.org/10.1214/20-BA1221.

[2]. Esophagus cancer: Esophageal cancer (no date) Esophageal Cancer | American Cancer Society. Available at: https://www.cancer.org/cancer/types/esophagus-cancer.html (Accessed: 30 November 2023).

Appendix

Here we attach the whole code that was used in the project.

```
#Import libraries

library(medicaldata)
library(bayesplot)
library(ggplot2)
library(dplyr)
library(brms)
library(caret)
library(caTools)
library(reshape2)
library(gridExtra)
library(rstan)

#For reproducibility
set.seed(932)

# Globally specfiy cmdstan backend for brms
options(brms.backend="cmdstanr")
options(brms.file_refit="on_change")

# Toggle 'bayesplot' color options
color_scheme_set("brightblue")
```

```r
# Fetch the dataset
espoh <- medicaldata::esoph_ca
espoh$total <- espoh$ncases + espoh$ncontrols
espoh$prob <- espoh$ncases / espoh$total
head(espoh)
```

```r
# Calculate empirical probabilities
data_age <- espoh %>%
            group_by(agegp) %>%
            summarise(TotalCases = sum(ncases),
                      Total = sum(ncases + ncontrols)) %>%
            mutate(Probability = TotalCases / Total)
```

```r
# Calculate empirical probabilities
data_alc <- espoh %>%
            group_by(alcgp) %>%
            summarise(TotalCases = sum(ncases),
                      Total = sum(ncases + ncontrols)) %>%
            mutate(Probability = TotalCases / Total)
```

```r
# Calculate empirical probabilities
data_tob <- espoh %>%
            group_by(tobgp) %>%
            summarise(TotalCases = sum(ncases),
                      Total = sum(ncases + ncontrols)) %>%
            mutate(Probability = TotalCases / Total)
```

```r
# Plot for Age Group
p1 <- ggplot(data_age, aes(x = agegp, y = Probability)) +
      geom_bar(stat = "identity", fill = "lightblue") +
      labs(title = "Empirical Probability of Cases by Age Group",
           x = "Age Group",
           y = "Probability") +
      theme_minimal()

# Plot for Alcohol Consumption Group
p2 <- ggplot(data_alc, aes(x = alcgp, y = Probability)) +
      geom_bar(stat = "identity", fill = "lightgreen") +
      labs(title = "Empirical Probability of Cases by Alcohol Consumption Group",
           x = "Alcohol Consumption Group",
           y = "Probability") +
      theme_minimal()

# Plot for Tobacco Consumption Group
p3 <- ggplot(data_tob, aes(x = tobgp, y = Probability)) +
      geom_bar(stat = "identity", fill = "lightcoral") +
      labs(title = "Empirical Probability of Cases by Tobacco Consumption Group",
           x = "Tobacco Consumption Group",
           y = "Probability") +
      theme_minimal()

# Combine the plots
grid.arrange(p1, p2, p3, ncol = 1)
```

```r
column_names <- c("Age", "Tobac", "Alco", "Has_cancer")
new_df <- data.frame(matrix(ncol = length(column_names), nrow = 0))
colnames(new_df) <- column_names

for (i in 1:dim(espoh)[1]) {
  for (j in 1:espoh[i, "ncases"]) {
    new_row = data.frame(
      Age = espoh[i, "agegp"],
      Tobac = espoh[i, "alcgp"],
      Alco = espoh[i, "tobgp"],
      Has_cancer = 1
    )
    new_df <- rbind(new_df, new_row)
  }
  for (k in 1:espoh[i, "ncontrols"]) {
    new_row = data.frame(
      Age = espoh[i, "agegp"],
      Tobac = espoh[i, "alcgp"],
      Alco = espoh[i, "tobgp"],
      Has_cancer = 0
    )
    new_df <- rbind(new_df, new_row)
  }
}
colnames(new_df) <- c("age", "tob", "alc", "cancer")
new_df$ID <- 1:dim(new_df)[1]
new_df <- new_df[, c(5, 1, 2, 3, 4)]
head(new_df)
```

```r
#Factorize the covariates into ordinary variables
new_df$age <- factor(new_df$age, ordered=TRUE)
new_df$tob <- factor(new_df$tob, ordered=TRUE)
new_df$alc <- factor(new_df$alc, ordered=TRUE)
```

```r
data_split <- sample.split(new_df$cancer, SplitRatio = 0.7)
train_data <- subset(new_df, data_split == TRUE)
test_data <- subset(new_df, data_split == FALSE)
```

```r
#Examine the default priors suggested
get_prior(cancer ~ 1 + tob + alc + age,
             data = train_data,
             family = bernoulli("logit"))
```

```r
# Defining the ordinal logistic model
model <- brm(cancer ~ 1 + tob + alc + age,
             data = train_data,
             family = bernoulli("logit"),
             prior = c(prior(normal(0.02, 10), class = "b"),
                       prior(normal(0, 10), class = "Intercept")),
             chains=4,
             file="model1",
             backend="cmdstanr",
             cores=4)
```

```r
summary(model)
```

```r
np1 <- nuts_params(model)
# extract the number of divergence transitions
cat(sprintf("Number of divergence of the transitions: %i", sum(subset(np1,
    Parameter == "divergent__")$Value)))
```

```r
model1_alt <- brm(cancer ~ 1 + tob + alc + age,
              data = train_data,
              family = bernoulli("logit"),
              chains=4,
              file="model1_alt",
              backend="cmdstanr",
              cores=4)
```

```r
# Extracting posterior samples
posterior_samples_model1 <- as_draws(model)
posterior_samples_model1_alt <- as_draws(model1_alt)
```

```r
posterior_tob_model1 <- mcmc_areas(posterior_samples_model1,
  pars = c("b_tob.L", "b_tob.Q", "b_tob.C")) +
  xaxis_title() +
  ggplot2::xlab("Customized priors")

posterior_tob_model1_alt <- mcmc_areas(posterior_samples_model1_alt,
  pars = c("b_tob.L", "b_tob.Q", "b_tob.C")) +
  xaxis_title() +
  ggplot2::xlab("Default priors")
```

```r
# Combine the plots
grid.arrange(posterior_tob_model1, posterior_tob_model1_alt, ncol = 2)
```

```r
posterior_alc_model1 <- mcmc_areas(posterior_samples_model1,
  pars = c("b_alc.L", "b_alc.Q", "b_alc.C")) +
  xaxis_title() +
  ggplot2::xlab("Customized priors")

posterior_alc_model1_alt <- mcmc_areas(posterior_samples_model1_alt,
  pars = c("b_alc.L", "b_alc.Q", "b_alc.C")) +
  xaxis_title() +
  ggplot2::xlab("Default priors")
```

```r
# Combine the plots
grid.arrange(posterior_alc_model1, posterior_alc_model1_alt, ncol = 2)
```

```r
posterior_age_model1 <- mcmc_areas(posterior_samples_model1,
  pars = c("b_age.L", "b_age.Q", "b_age.C", "b_ageE4", "b_ageE5")) +
  xaxis_title() +
  ggplot2::xlab("Customized priors")

posterior_age_model1_alt <- mcmc_areas(posterior_samples_model1_alt,
  pars = c("b_age.L", "b_age.Q", "b_age.C", "b_ageE4", "b_ageE5")) +
  xaxis_title() +
  ggplot2::xlab("Default priors")
```

```r
# Combine the plots
grid.arrange(posterior_age_model1, posterior_age_model1_alt, ncol = 2)
```

```r
#Examine the default priors suggested
get_prior(cancer ~ 1 + tob + alc + age + (1 | ID),
    data = train_data,
          family = bernoulli("logit"))
```

```r
# Defining the ordinal logistic model
model2 <- brm(cancer ~ 1 + tob + alc + age + (1 | ID),
              data = train_data,
              family = bernoulli("logit"),
              prior = c(prior(normal(0.02, 10), class = "b"),
                        prior(normal(0, 10), class = "Intercept")),
              chains=4,
              file="model2",
              backend="cmdstanr",
              cores=4)
```

```r
summary(model2)
```

```r
np2 <- nuts_params(model2)
# extract the number of divergence transitions
cat(sprintf("Number of divergence of the transitions: %i", sum(subset(np2,
    Parameter == "divergent__")$Value)))
```

```r
model2_alt <- brm(cancer ~ 1 + tob + alc + age + (1 | ID),
              data = train_data,
              family = bernoulli("logit"),
              chains=4,
              file="model2_alt",
              backend="cmdstanr",
              cores=4)
```

```r
# Extracting posterior samples
posterior_samples_model2 <- as_draws(model2)
posterior_samples_model2_alt <- as_draws(model2_alt)
```

```r
posterior_tob_model2 <- mcmc_areas(posterior_samples_model2,
  pars = c("b_tob.L", "b_tob.Q", "b_tob.C")) +
  xaxis_title() +
  ggplot2::xlab("Customized priors")

posterior_tob_model2_alt <- mcmc_areas(posterior_samples_model2_alt,
  pars = c("b_tob.L", "b_tob.Q", "b_tob.C")) +
  xaxis_title() +
  ggplot2::xlab("Default priors")
```

```r
# Combine the plots
grid.arrange(posterior_tob_model2, posterior_tob_model2_alt, ncol = 2)
```

```r
posterior_alc_model2 <- mcmc_areas(posterior_samples_model2,
  pars = c("b_alc.L", "b_alc.Q", "b_alc.C")) +
  xaxis_title() +
  ggplot2::xlab("Customized priors")
```

```r
posterior_alc_model2_alt <- mcmc_areas(posterior_samples_model2_alt,
  pars = c("b_alc.L", "b_alc.Q", "b_alc.C")) +
  xaxis_title() +
  ggplot2::xlab("Default priors")
```

```r
# Combine the plots
grid.arrange(posterior_alc_model2, posterior_alc_model2_alt, ncol = 2)
```

```r
posterior_age_model2 <- mcmc_areas(posterior_samples_model2,
  pars = c("b_age.L", "b_age.Q", "b_age.C", "b_ageE4", "b_ageE5")) +
  xaxis_title() +
  ggplot2::xlab("Customized priors")

posterior_age_model2_alt <- mcmc_areas(posterior_samples_model2_alt,
  pars = c("b_age.L", "b_age.Q", "b_age.C", "b_ageE4", "b_ageE5")) +
  xaxis_title() +
  ggplot2::xlab("Default priors")
```

```r
# Combine the plots
grid.arrange(posterior_age_model2, posterior_age_model2_alt, ncol = 2)
```

```r
# Posterior predictive checks for the pooled model
pp_dens_1_model_1 <- pp_check(model, newdata = test_data) +
  xaxis_title() +
  ggplot2::xlab("Pooled model")
pp_dens_2_model_1 <- pp_check(model, type="dens_overlay_grouped", group="age", newdata = test_data) +
  xaxis_title() +
  xaxis_ticks(on = TRUE) +
  xaxis_text(on = FALSE) +
  ggplot2::xlab("Pooled model grouped by age")
# Posterior predictive checks for the hierarchical model
pp_dens_1_model_2 <- pp_check(model2, newdata = test_data,
                              allow_new_levels=TRUE,
                              sample_new_levels="gaussian") +
  xaxis_title() +
  ggplot2::xlab("Hierarchical model")
pp_dens_2_model_2 <- pp_check(model2, type="dens_overlay_grouped", group="age",
                              newdata = test_data, allow_new_levels=TRUE,
                              sample_new_levels="gaussian") +
  xaxis_title() +
  xaxis_ticks(on = TRUE) +
  xaxis_text(on = FALSE) +
  ggplot2::xlab("Hierarchical model grouped by age")
```

```r
# Combine the plots
grid.arrange(pp_dens_1_model_1, pp_dens_1_model_2, ncol = 2)
```

```r
grid.arrange(pp_dens_2_model_1, pp_dens_2_model_2, ncol = 2)
```

```r
pp_bar_model_1 <- pp_check(model, type="bars", newdata = test_data) +
  xaxis_title() +
  ggplot2::xlab("Pooled model")
pp_bar_model_2 <- pp_check(model2, type="bars", newdata = test_data,
                           allow_new_levels=TRUE,
                           sample_new_levels="gaussian") +
```

```
  xaxis_title() +
  ggplot2::xlab("Hierarchical model")
```

```
grid.arrange(pp_bar_model_1, pp_bar_model_2)
```

```
pp_bar_grouped_model_1 <- pp_check(model, type="bars_grouped", group="age", newdata = test_data) +
  xaxis_title() +
  ggplot2::xlab("Pooled model")
```

```
pp_bar_grouped_model_1
```

```
pp_bar_grouped_model_2 <- pp_check(model2, type="bars_grouped", group="age",
                                   newdata = test_data,
                                   allow_new_levels=TRUE,
                                   sample_new_levels="gaussian") +
  xaxis_title() +
  ggplot2::xlab("Hierarchical model")
```

```
pp_bar_grouped_model_2
```

```
p_stat1 <- pp_check(model, type="stat", newdata = test_data) +
  xaxis_title() +
  ggplot2::xlab("Pooled model")
p_stat2 <- pp_check(model2, type="stat",
                       newdata = test_data,
                       allow_new_levels=TRUE,
                       sample_new_levels="gaussian") +
  xaxis_title() +
  ggplot2::xlab("Hierarchical model")
```

```
grid.arrange(p_stat1, p_stat2)
```

```
pp_stat_grouped_model_1 <- pp_check(model, type="stat_grouped", group="age", newdata = test_data) +
  xaxis_title() +
  ggplot2::xlab("Pooled model")
```

```
pp_stat_grouped_model_1
```

```
pp_stat_grouped_model_2 <- pp_check(model2, type="stat_grouped", group="age", newdata = test_data,
  allow_new_levels=TRUE, sample_new_levels="gaussian")
```

```
pp_stat_grouped_model_2
```

```
# For pooled model:
#Get posterior predictions for the test set
posterior_predict_model1 <- posterior_predict(model, newdata=test_data)

#Calculate the mean of the posterior predictions
predicted_probabilities_model1 <- colMeans(posterior_predict_model1)

predicted_classes_model1 <- ifelse(predicted_probabilities_model1 > 0.5, 1, 0)

# For hierarchical model:
#Get posterior predictions for the test set
posterior_predict_model2 <- posterior_predict(model2, newdata=test_data,
        allow_new_levels=TRUE, sample_new_levels = "gaussian", cores=4)
```

```r
#Calculate the mean of the posterior predictions
predicted_probabilities_model2 <- colMeans(posterior_predict_model2)

predicted_classes_model2 <- ifelse(predicted_probabilities_model2 > 0.5, 1, 0)
```

```r
#Actual classes in the test set
actual_classes <- test_data$cancer

#Create a confusion matrix
conf_matrix_model1 <- table(Predicted = predicted_classes_model1, Actual = actual_classes)

conf_matrix_model2 <- table(Predicted = predicted_classes_model2, Actual = actual_classes)
conf_matrix_df_model2 <- as.data.frame(as.table(conf_matrix_model2))

cat("Pooled model: \n")
conf_matrix_model1
cat("\n")
cat("Hierarchical model: \n")
conf_matrix_model2
```

```r
true_positives_model1 <- conf_matrix_model1[2, 2]
true_negatives_model1 <- conf_matrix_model1[1, 1]
false_positives_model1 <- conf_matrix_model1[2, 1]
false_negatives_model1 <- conf_matrix_model1[1, 2]

accuracy_model1 <- (true_positives_model1 + true_negatives_model1) / sum(conf_matrix_model1)
precision_model1 <- true_positives_model1 / (true_positives_model1 + false_positives_model1)
recall_model1 <- true_positives_model1 / (true_positives_model1 + false_negatives_model1)
f1_score_model1 <- 2 * (precision_model1 * recall_model1) / (precision_model1 + recall_model1)

cat("Pooled model: \n")
cat(sprintf("Accuracy:    %.1f\nPrecision:   %.1f\nRecall:      %.1f\nF1 Score:    %.1f",
    accuracy_model1, precision_model1, recall_model1, f1_score_model1))
```

```r
true_positives_model2 <- conf_matrix_model2[2, 2]
true_negatives_model2 <- conf_matrix_model2[1, 1]
false_positives_model2 <- conf_matrix_model2[2, 1]
false_negatives_model2 <- conf_matrix_model2[1, 2]

accuracy_model2 <- (true_positives_model2 + true_negatives_model2) / sum(conf_matrix_model2)
precision_model2 <- true_positives_model2 / (true_positives_model2 + false_positives_model2)
recall_model2 <- true_positives_model2 / (true_positives_model2 + false_negatives_model2)
f1_score_model2 <- 2 * (precision_model2 * recall_model2) / (precision_model2 + recall_model2)

cat("Hierarchical model: \n")
cat(sprintf("Accuracy:    %.1f\nPrecision:   %.1f\nRecall:      %.1f\nF1 Score:    %.1f",
    accuracy_model2, precision_model2, recall_model2, f1_score_model2))
```

```r
loo1 <- loo::loo(model)
loo2 <- loo::loo(model2)
```

```r
plot(loo1, label_points=TRUE)
```

```r
plot(loo2, label_points=TRUE)
```

```
loo::loo_compare(loo1, loo2)
```