BDA - Project

Hamza Hanchi, Rui Qu

Introduction

In this report we look at the data of the passengers on the Titanic. The cruiseship tragically hit an iceberg which led to the death of the majority of the passengers. This dataset is popular to use to showcase statistical knowhow and will give it a try and analyze the data the bayesian way and will be doing so in this manner:

- Overview of the dataset and the analysis problem
- Data interpreation and prior choice discussion
- Modeling the reduced model and the full model
- Model analysis and results
- Discussion and conclusion

Dataset

In dataset has information about 891 passenger, we don't have all the data for some of the passengers. We have conclusive data of 714 passengers which will focus on. The information we have about the passengers:

- Name
- Age
- Sex
- Passenger Class - (1st, 2nd, 3rd)

- SibSp - Number of siblings and spouses on the boat
- Parch - Number of parents and children on the boat
- Ticket - Ticket number
- Survived
- Fare - Price of the fare
- Cabin - Which Cabins belong to the passenger
- Embarked - Port of embarkation of passenger (Southampton, Cherbourg, Queenstown)

Problem analysis

We want to create a model that has a high prediction accuracy on the survival rate of the passengers on Titanic in terms of the previously mentioned variables. We will create 3 models which we will compare on prediction accuracy and then present the best model. The 1st reduced model will consists of fewer varibles which of which we will select later based on their correlation to survival. The 2nd model will consist of all varibles except for Cabin, Fare and that is because these variables do not provide a lot of useful information as they are highly dependant on other varibles such as fare or cabin and passenger class. We also exclude Ticket because it's just a code that has no relationship to the other varibles and thus will not help the prediction in anyway. The 3rd model we created is a hierarchical model where we treat passenger class in a group with respect to age and sex. we exclude those variavles like what's done in the 1st model.

project setup

```
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
library(brms)
```

```
## Loading required package: Rcpp
```

```
## Registered S3 method overwritten by 'xts':
##   method     from
##   as.zoo.xts zoo
```

```
## Loading 'brms' package (version 2.10.0). Useful instructions
## can be found by typing help('brms'). A more detailed introduction
## to the package is available through vignette('brms_overview').
```

```
library(stringr)
library(bayesplot)
```

```
## This is bayesplot version 1.7.1
```

```
## - Online documentation and vignettes at mc-stan.org/bayesplot
```

```
## - bayesplot theme set to bayesplot::theme_default()
```

```
##    * Does _not_ affect other ggplot2 plots
```

```
##    * See ?bayesplot_theme_set for details on theme setting
```

```
library(projpred)
```

```
## This is projpred version 1.1.4.
```

```
library(rstan)
```

```
## Loading required package: StanHeaders
```

```
## Loading required package: ggplot2
```

```
## rstan (Version 2.19.2, GitRev: 2e1f913d3ca3)
```

```
## For execution on a local, multicore CPU with excess RAM we recommend calling
## options(mc.cores = parallel::detectCores()).
## To avoid recompilation of unchanged Stan programs, we recommend calling
## rstan_options(auto_write = TRUE)
```

```
## For improved execution time, we recommend calling
## Sys.setenv(LOCAL_CPPFLAGS = '-march=native')
## although this causes Stan to throw an error on a few processors.
```

```
rstan_options(auto_write = TRUE)
options(mc.cores = parallel::detectCores())
library(loo)
```

```
## This is loo version 2.1.0.
## **NOTE: As of version 2.0.0 loo defaults to 1 core but we recommend using as many as possible. Use tl
```

```
## **NOTE for Windows 10 users: loo may be very slow if 'mc.cores' is set in your .Rprofile file (see ht
```

```
##
## Attaching package: 'loo'
```

```
## The following object is masked from 'package:rstan':
##
##     loo
```

```
require(ggplot2)
library(titanic)

fulldata <- titanic_train
testdata <- titanic_test
```

Data preparation

We filter our the passengers with inconclusive data and we also convert the sex string value to numeric values. We do the same for the embarked values so we can create stan models based on the data. Male = 1, Female = 0. C = 0, Q = 1, S = 2.

```
data <- fulldata
data$Sex <- ifelse(data$Sex=="male", 1, 0)
testdata$Sex <- ifelse(testdata$Sex=="male", 1, 0)
data$Embarked <- ifelse(data$Embarked=="C", 0, ifelse(data$Embarked=="Q", 1, 2))
testdata$Embarked <- ifelse(testdata$Embarked=="C", 0, ifelse(testdata$Embarked=="Q", 1, 2))
data = na.omit(data)
testdata = na.omit(testdata)
```

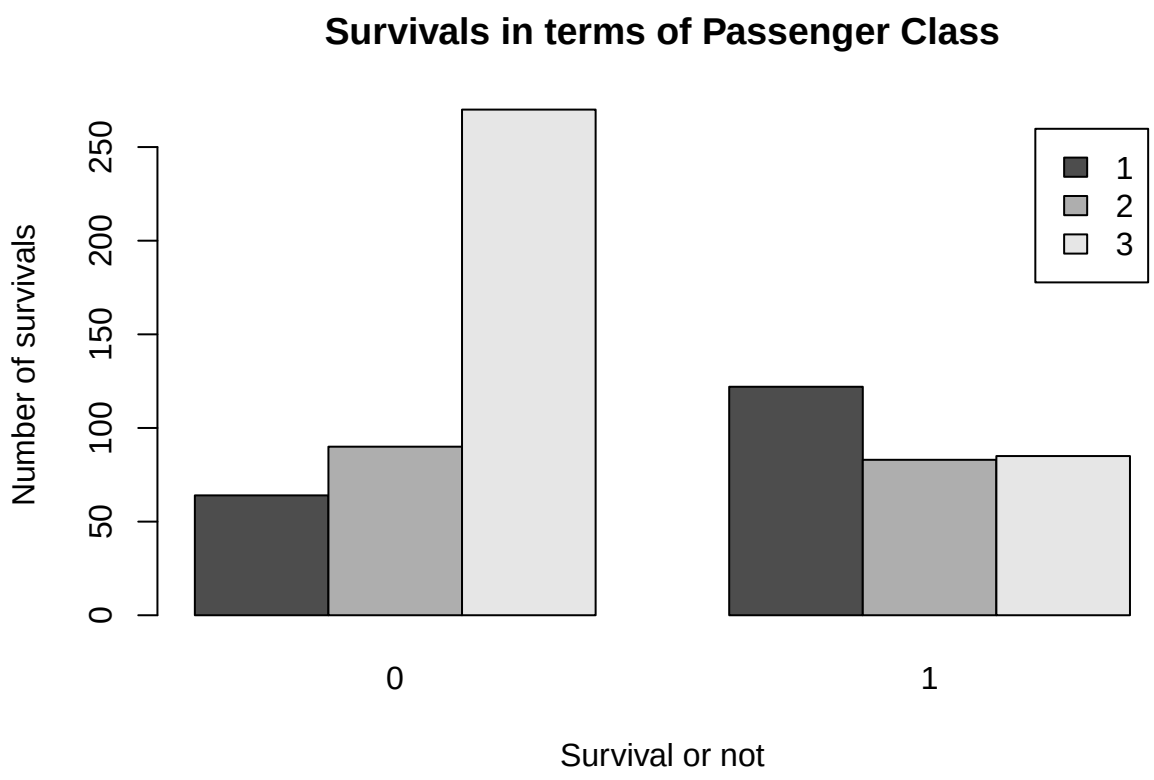
Data visualization

The ages of the passengers are varying but from the graph we can see that it closely resembles a normal distrubtion with the majority being between the ages 18-40.

```
counts <- table(data$Survived,data$Age)
barplot(table(data$Survived,data$Age),,main="Survivals in terms of age",
  xlab="Age", ylab="Number of People",legend=rownames(table(fulldata$Survived,fulldata$Age)))
```

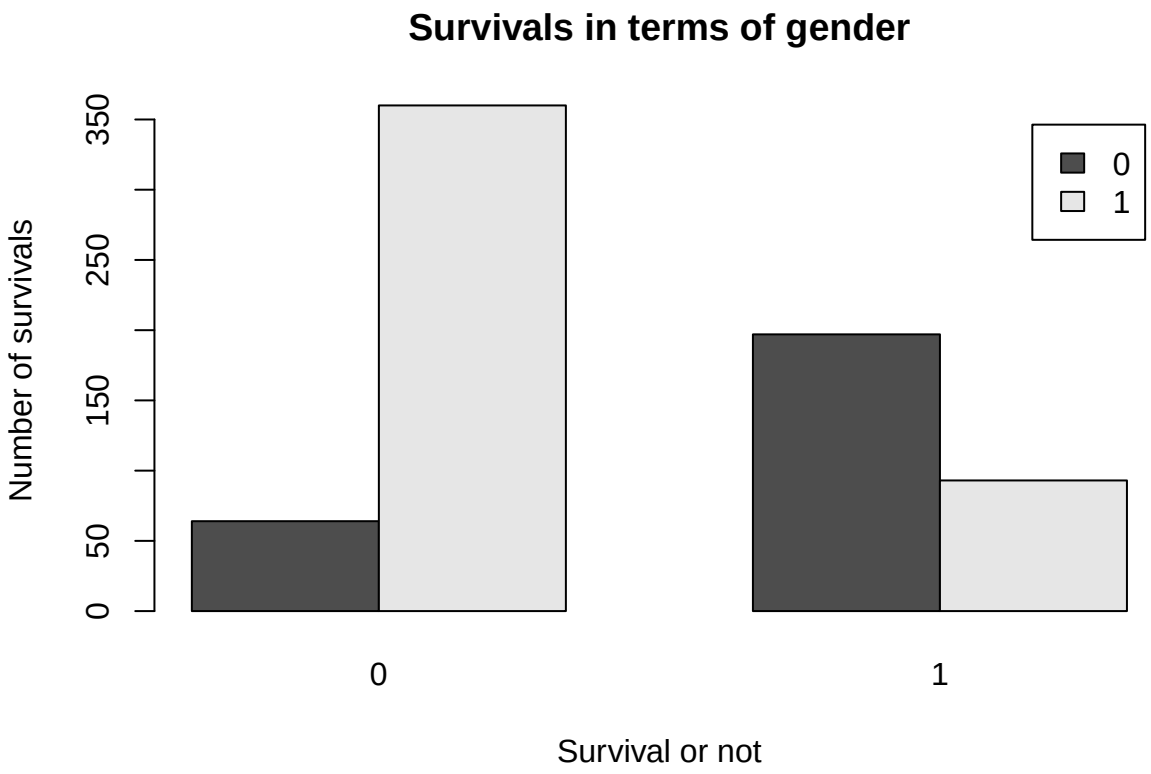
A slight majority of the people on the boat were 3rd class and the first class had a slight majority over the second class. It can be usefult to mention that the unfiltered data has a larger majority for the third passenger class. From the plot below we can see that survivals in term of passenger class varied, the 3rd class passenger had a higher rate of death while the 1st class had the highest rate of survival.

```r
counts <- table(data$Pclass,data$Survived)
barplot(counts,main="Survivals in terms of Passenger Class",
  xlab="Survival or not", ylab="Number of survivals",
  legend = rownames(counts), beside=TRUE)
```



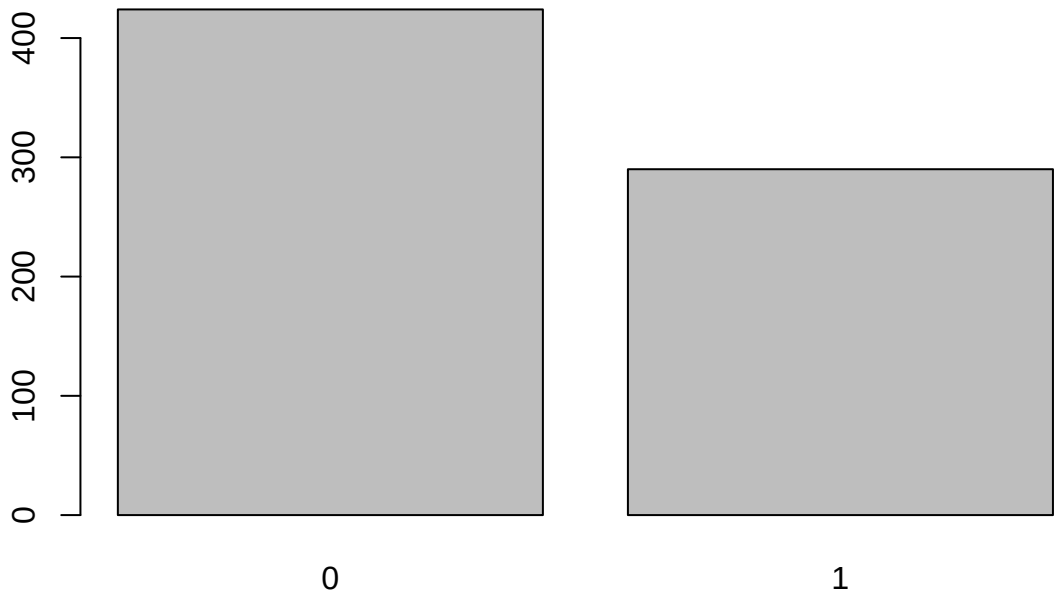
There were 261 women and 453 men on the dataset and from the plot below we can see that there is a higher rate of death among men.

```r
counts <- table(data$Sex,data$Survived)
barplot(counts,main="Survivals in terms of gender",
  xlab="Survival or not", ylab="Number of survivals",
  legend = rownames(counts), beside=TRUE)
```



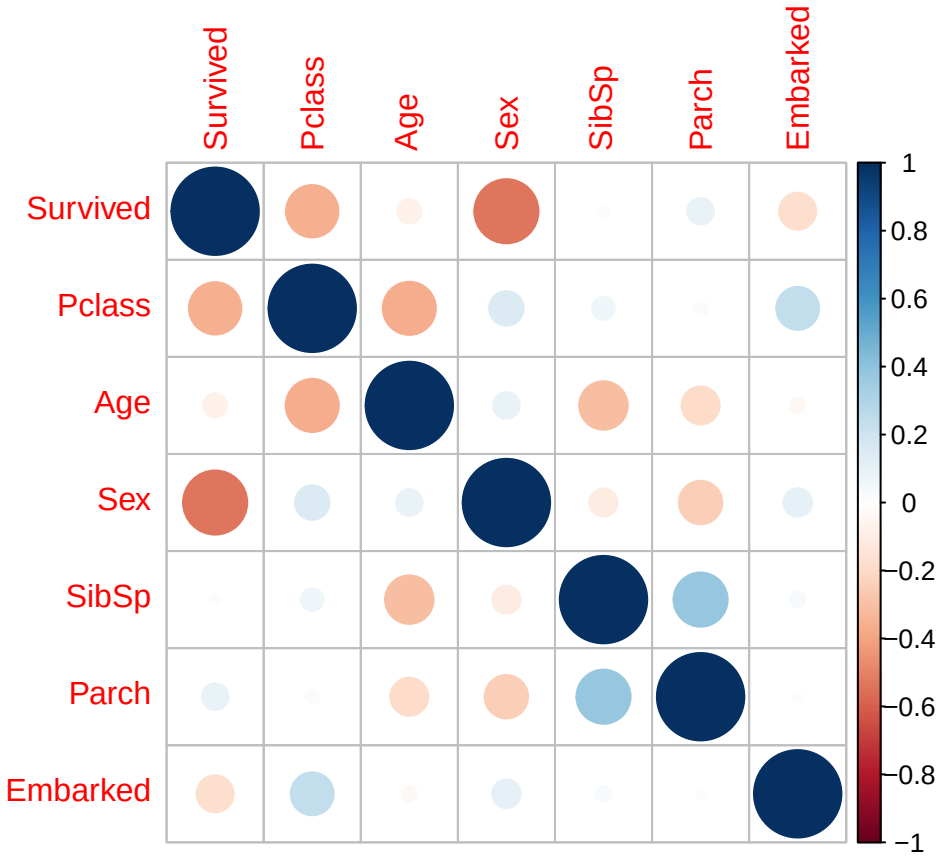
290 passengers survived, 424 did not survive (714 in total). From the data we can see that the majority of the passengers did not make it out of the boat alive but a lot of people did survive.

```
barplot(table(data$Survived))
```



Below we show the correlation plot of the variables in the dataset and we can clearly see that the varibles Passenger class and sex are closely corelated to survival while the other ones played a smaller role. There are also other correlation visible such as the correlation between age and passenger class For the correlation between Sibling/Spouse and Parent/child. By looking at the correlation plot we can select which variables to select for our first reduced model and we can clearly see that passenger class and sex are closely correlated to the survival rate and therefore we will use these varibles in the reduced model. We will also use age because we believe that is a natural way of describing a passenger.

```
pred <- c("Pclass", "Age", "Sex", "SibSp", "Parch", "Embarked")
target <- c("Survived")
formula <- paste("Survived ~", paste(pred, collapse = "+"))
p <- length(pred)
n <- nrow(data)
x = cor(data[, c(target,pred)])
corrplot(x)
```



Data interpretion and prior choice

The relationships of the data seem to be linear and therefore will not explore non-linear solutions for the varibles. From this dataset we can quickly see that a majority of people did not survive the trip, we will use the data to make predictions about the survival of someone if he/she was on the boat. The types of priors we were considering are uniformative priors, weakly informative/regularizing priors and informative priors. Titanic was of course not the first boat to ever have such a tragic fate but it was also the first of it class, the largest and most advanced passengerboat. Therefore we don't believe it would be suitable to use previous knowledge from other tragic events as informative priors for this dataset. Using a weakly informative prior means that we deliberatly use a less informative prior than our actual knowledge which will affect the posterior less than an informative prior but we believe this option isn't the right option with the same reasoning as earlier. An uninformative prior is preferred as we don't believe the prior knowledge is relevant and because we want the data to speak for itself. By following the ecommendations of good deafult priors made by Stan prior choice recommendations. Therefore we will use the super-vague but proper prior: normal(0, 1e6) for our model.

Later on we will test the models prior sensitivity with the default prior of brms. Student's t distribution with 3 degrees of freedom, location 0 and scale 10.

Modeling

We will do a bayesian linear regression where we predict the survivability using a linear model. We will be working with two bernoulli linear models, we will be using all the variables except for the fare, cabin and ticker and in the full model and in the reduced model we will only use age, sex and passenger class to predict on that. We use BRMS to generate our stan models. The reason we choose to use the bernoulli family is because we are predicting survivability which has only two possible outcomes.

To run and generate the stan code:

```
TitanicSelection<- brm(Survived ~ Age + Sex + Pclass, family = bernoulli(),
    data = data, prior = set_prior('normal(0, 1000000)'),save_all_pars = T)
```

```
## Compiling the C++ model
```

```
## Start sampling
```

```
TitanicFull <- brm(Survived ~ Age + Sex + Pclass + SibSp + Parch + Embarked,  family = bernoulli(),
    data = data, prior = set_prior('normal(0, 1000000)'), save_all_pars = T )
```

```
## Compiling the C++ model
```

```
## recompiling to avoid crashing R session
```

```
## Start sampling
```

```
TitanicHier <- brm(Survived ~ Age + Sex + (Age + Sex |Pclass), family = bernoulli(),
    data = data, prior = set_prior('normal(0, 1000000)'), save_all_pars = T )
```

```
## Compiling the C++ model
## Start sampling
```

```
## Warning: There were 84 divergent transitions after warmup. Increasing adapt_delta above 0.8 may help
## http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
```

```
## Warning: Examine the pairs() plot to diagnose sampling problems
```

```
## Warning: Tail Effective Samples Size (ESS) is too low, indicating posterior variances and tail quant
## Running the chains for more iterations may help. See
## http://mc-stan.org/misc/warnings.html#tail-ess
```

Stan models

Selection Model

With three varibles: Age, Sex and passenger class.

```
// generated with brms 2.10.0
functions {
}
data {
  int<lower=1> N;  // number of observations
  int Y[N];  // response variable
  int<lower=1> K;  // number of population-level effects
  matrix[N, K] X;  // population-level design matrix
  int prior_only;  // should the likelihood be ignored?
}
transformed data {
  int Kc = K - 1;
  matrix[N, Kc] Xc;  // centered version of X without an intercept
  vector[Kc] means_X;  // column means of X before centering
  for (i in 2:K) {
    means_X[i - 1] = mean(X[, i]);
    Xc[, i - 1] = X[, i] - means_X[i - 1];
  }
}
parameters {
  vector[Kc] b;  // population-level effects
```

```
  // temporary intercept for centered predictors
  real Intercept;
}
transformed parameters {
}
model {
  // priors including all constants
  target += normal_lpdf(b | 0, 1000000);
  target += student_t_lpdf(Intercept | 3, 0, 10);
  // likelihood including all constants
  if (!prior_only) {
    target += bernoulli_logit_glm_lpmf(Y | Xc, Intercept, b);
  }
}
generated quantities {
  // actual population-level intercept
  real b_Intercept = Intercept - dot_product(means_X, b);
}
```

Full model

With all relevant variables such as: Age, Sex, Passenger Class, Siblings/Spouse, Parent/childeren, and City of embarktion

```
// generated with brms 2.10.0
functions {
}
data {
  int<lower=1> N;  // number of observations
  int Y[N];  // response variable
  int<lower=1> K;  // number of population-level effects
  matrix[N, K] X;  // population-level design matrix
  int prior_only;  // should the likelihood be ignored?
}
transformed data {
  int Kc = K - 1;
  matrix[N, Kc] Xc;  // centered version of X without an intercept
  vector[Kc] means_X;  // column means of X before centering
  for (i in 2:K) {
    means_X[i - 1] = mean(X[, i]);
    Xc[, i - 1] = X[, i] - means_X[i - 1];
  }
}
parameters {
  vector[Kc] b;  // population-level effects
  // temporary intercept for centered predictors
  real Intercept;
}
transformed parameters {
}
model {
  // priors including all constants
  target += normal_lpdf(b | 0, 1000000);
  target += student_t_lpdf(Intercept | 3, 0, 10);
  // likelihood including all constants
```

```
  if (!prior_only) {
    target += bernoulli_logit_glm_lpmf(Y | Xc, Intercept, b);
  }
}
generated quantities {
  // actual population-level intercept
  real b_Intercept = Intercept - dot_product(means_X, b);
}
```

With three varibles: Age, Sex and passenger class.

Hierarchical model

```
// generated with brms 2.10.0
functions {
}
data {
  int<lower=1> N;  // number of observations
  int Y[N];  // response variable
  int<lower=1> K;  // number of population-level effects
  matrix[N, K] X;  // population-level design matrix
  // data for group-level effects of ID 1
  int<lower=1> N_1;  // number of grouping levels
  int<lower=1> M_1;  // number of coefficients per level
  int<lower=1> J_1[N];  // grouping indicator per observation
  // group-level predictor values
  vector[N] Z_1_1;
  vector[N] Z_1_2;
  vector[N] Z_1_3;
  int<lower=1> NC_1;  // number of group-level correlations
  int prior_only;  // should the likelihood be ignored?
}
transformed data {
  int Kc = K - 1;
  matrix[N, Kc] Xc;  // centered version of X without an intercept
  vector[Kc] means_X;  // column means of X before centering
  for (i in 2:K) {
    means_X[i - 1] = mean(X[, i]);
    Xc[, i - 1] = X[, i] - means_X[i - 1];
  }
}
parameters {
  vector[Kc] b;  // population-level effects
  // temporary intercept for centered predictors
  real Intercept;
  vector<lower=0>[M_1] sd_1;  // group-level standard deviations
  matrix[M_1, N_1] z_1;  // standardized group-level effects
  // cholesky factor of correlation matrix
  cholesky_factor_corr[M_1] L_1;
}
transformed parameters {
  // actual group-level effects
  matrix[N_1, M_1] r_1 = (diag_pre_multiply(sd_1, L_1) * z_1)';
  // using vectors speeds up indexing in loops
  vector[N_1] r_1_1 = r_1[, 1];
```

```
  vector[N_1] r_1_2 = r_1[, 2];
  vector[N_1] r_1_3 = r_1[, 3];
}
model {
  // initialize linear predictor term
  vector[N] mu = Intercept + Xc * b;
  for (n in 1:N) {
    // add more terms to the linear predictor
    mu[n] += r_1_1[J_1[n]] * Z_1_1[n] + r_1_2[J_1[n]] * Z_1_2[n] + r_1_3[J_1[n]] * Z_1_3[n];
  }
  // priors including all constants
  target += normal_lpdf(b | 0, 1000000);
  target += student_t_lpdf(Intercept | 3, 0, 10);
  target += student_t_lpdf(sd_1 | 3, 0, 10)
    - 3 * student_t_lccdf(0 | 3, 0, 10);
  target += normal_lpdf(to_vector(z_1) | 0, 1);
  target += lkj_corr_cholesky_lpdf(L_1 | 1);
  // likelihood including all constants
  if (!prior_only) {
    target += bernoulli_logit_lpmf(Y | mu);
  }
}
generated quantities {
  // actual population-level intercept
  real b_Intercept = Intercept - dot_product(means_X, b);
  // group-level correlations
  corr_matrix[M_1] Cor_1 = multiply_lower_tri_self_transpose(L_1);
  vector<lower=-1,upper=1>[NC_1] cor_1;
  // extract upper diagonal of correlation matrix
  for (k in 1:M_1) {
    for (j in 1:(k - 1)) {
      cor_1[choose(k - 1, 2) + j] = Cor_1[j, k];
    }
  }
}
```

Before comparing the models performances we need to check that they have conveged to make sure the comparision is fair. We do this by looking at the summary of both models.

Model analysis and results

```
summary(TitanicSelection)
```

```
##  Family: bernoulli
##   Links: mu = logit
## Formula: Survived ~ Age + Sex + Pclass
##    Data: data (Number of observations: 714)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     5.09      0.52     4.10     6.10 1.00     3354     2488
## Age          -0.04      0.01    -0.05    -0.02 1.00     3557     2728
```

```
## Sex          -2.54      0.21    -2.96    -2.14 1.00     3863     3117
## Pclass       -1.30      0.14    -1.57    -1.02 1.00     3615     3181
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
check_divergences(TitanicSelection$fit)
```

```
## 0 of 4000 iterations ended with a divergence.
```

```
check_treedepth(TitanicSelection$fit)
```

```
## 0 of 4000 iterations saturated the maximum tree depth of 10.
```

```
summary(TitanicFull)
```

```
##  Family: bernoulli
##   Links: mu = logit
## Formula: Survived ~ Age + Sex + Pclass + SibSp + Parch + Embarked
##    Data: data (Number of observations: 714)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     5.87      0.57     4.76     7.00 1.00     4538     3015
## Age          -0.04      0.01    -0.06    -0.03 1.00     5061     3367
## Sex          -2.66      0.22    -3.10    -2.23 1.00     4887     3403
## Pclass       -1.29      0.15    -1.58    -1.00 1.00     4914     3162
## SibSp        -0.36      0.12    -0.61    -0.12 1.00     4599     3155
## Parch        -0.05      0.12    -0.29     0.19 1.00     5969     3149
## Embarked     -0.19      0.13    -0.45     0.08 1.00     5253     2987
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
check_divergences(TitanicFull$fit)
```

```
## 0 of 4000 iterations ended with a divergence.
```

```
check_treedepth(TitanicFull$fit)
```

```
## 0 of 4000 iterations saturated the maximum tree depth of 10.
```

```
summary(TitanicHier)
```

```
## Warning: There were 84 divergent transitions after warmup. Increasing adapt_delta above 0.8 may help
## See http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
```

```
##  Family: bernoulli
##   Links: mu = logit
## Formula: Survived ~ Age + Sex + (Age + Sex | Pclass)
##    Data: data (Number of observations: 714)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Group-Level Effects:
```

```
## ~Pclass (Number of levels: 3)
##                     Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
## sd(Intercept)           4.82      3.21     1.56    12.97 1.01      932
## sd(Age)                 0.08      0.08     0.00     0.33 1.01      701
## sd(Sex)                 3.69      2.73     1.01    10.75 1.00      713
## cor(Intercept,Age)     -0.10      0.48    -0.88     0.80 1.00     1669
## cor(Intercept,Sex)     -0.32      0.45    -0.96     0.65 1.01     1411
## cor(Age,Sex)            0.12      0.49    -0.80     0.91 1.01      934
##                     Tail_ESS
## sd(Intercept)            890
## sd(Age)                  897
## sd(Sex)                  497
## cor(Intercept,Age)       774
## cor(Intercept,Sex)      1526
## cor(Age,Sex)             728
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     3.31      2.70    -1.71     9.16 1.01      708      420
## Age          -0.05      0.06    -0.21     0.05 1.01      591      304
## Sex          -3.13      2.28    -7.55     2.01 1.00      796      284
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
check_divergences(TitanicHier$fit)
```

```
## 84 of 4000 iterations ended with a divergence (2.1%).
## Try increasing 'adapt_delta' to remove the divergences.
```

```
check_treedepth(TitanicHier$fit)
```

```
## 0 of 4000 iterations saturated the maximum tree depth of 10.
```

No divergences were observed in the non-hierarchical models and we believe that is because of the linear nature of the data. And since the summaries of The stan fits are not complaining about the tree depth being exceeding we can conclude that our chains converged well.

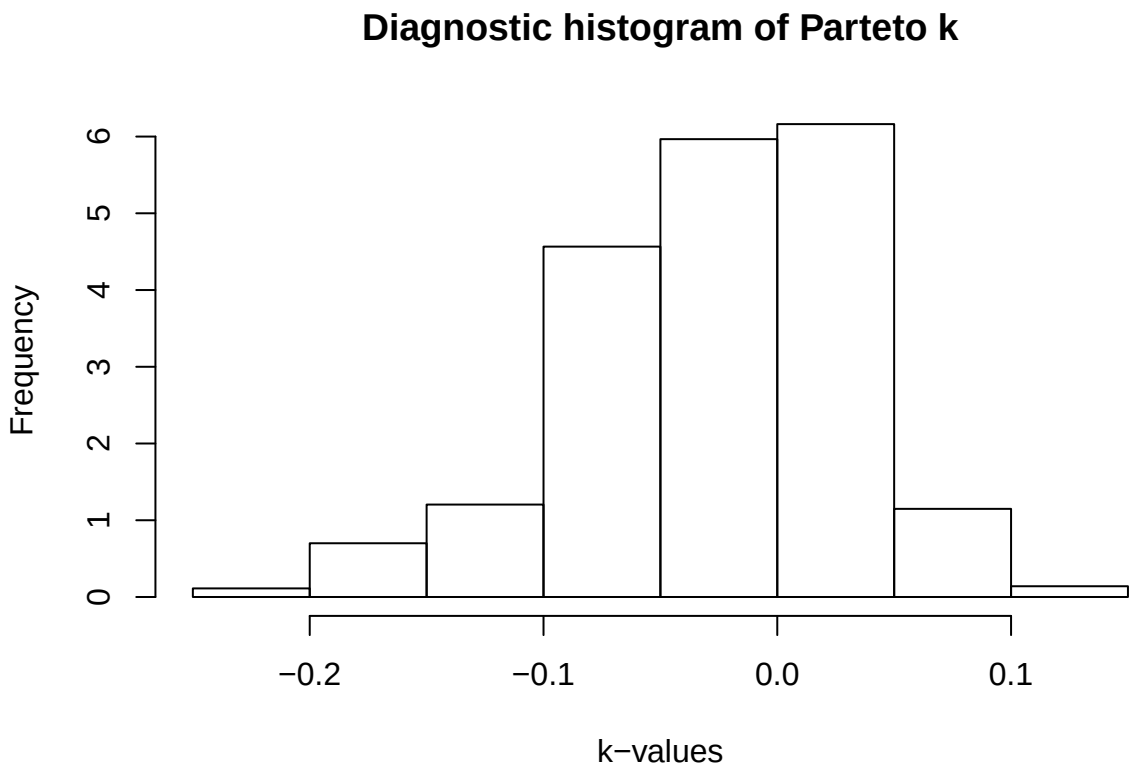
We can also see that the Rhat values for all models are all less than 1.05 which is an indication of well converged chain which means that we can consider the model to be reliable to further analyize. We can also observe very high values for both non-hierarchical models for ESS. The ESS corresponds to the number of independant samples and measures how much independence there is in autocorrelated chains. We can see that our non-hierarchical models have values well over 1000 which is the number that indicates stable estimations for most applications.

The hierarchical model has a small number of diverging chains and the ESS values are lower than we would like them to be which is can indicate unstable estimations but since the Rhat values look good we will continue analyzing this model.

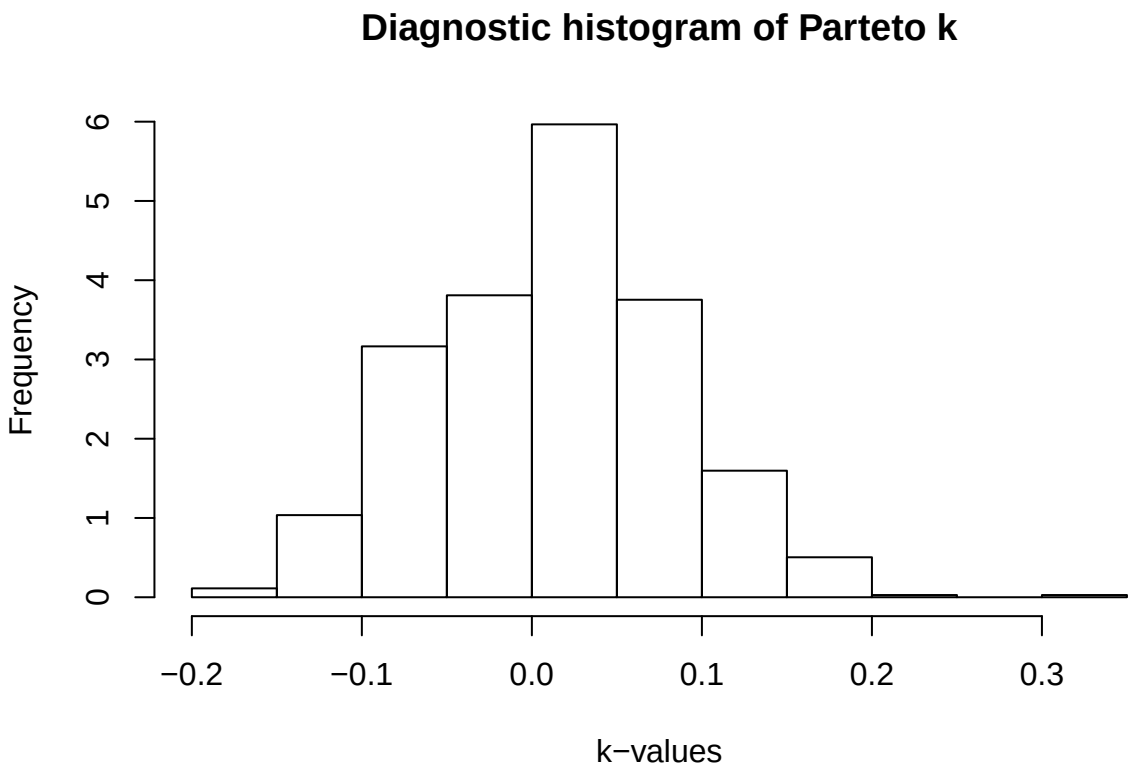
We also need to make sure that the k-values and loo-cv are low enough for us to be able to trust the data. We will make use off Loo-library for the diagnostics.

```
looSelection <- loo(TitanicSelection)
looFull <- loo(TitanicFull)
looHier <- loo(TitanicHier)
hist(looSelection$diagnostics$pareto_k,
     main = "Diagnostic histogram of Parteto k",
```

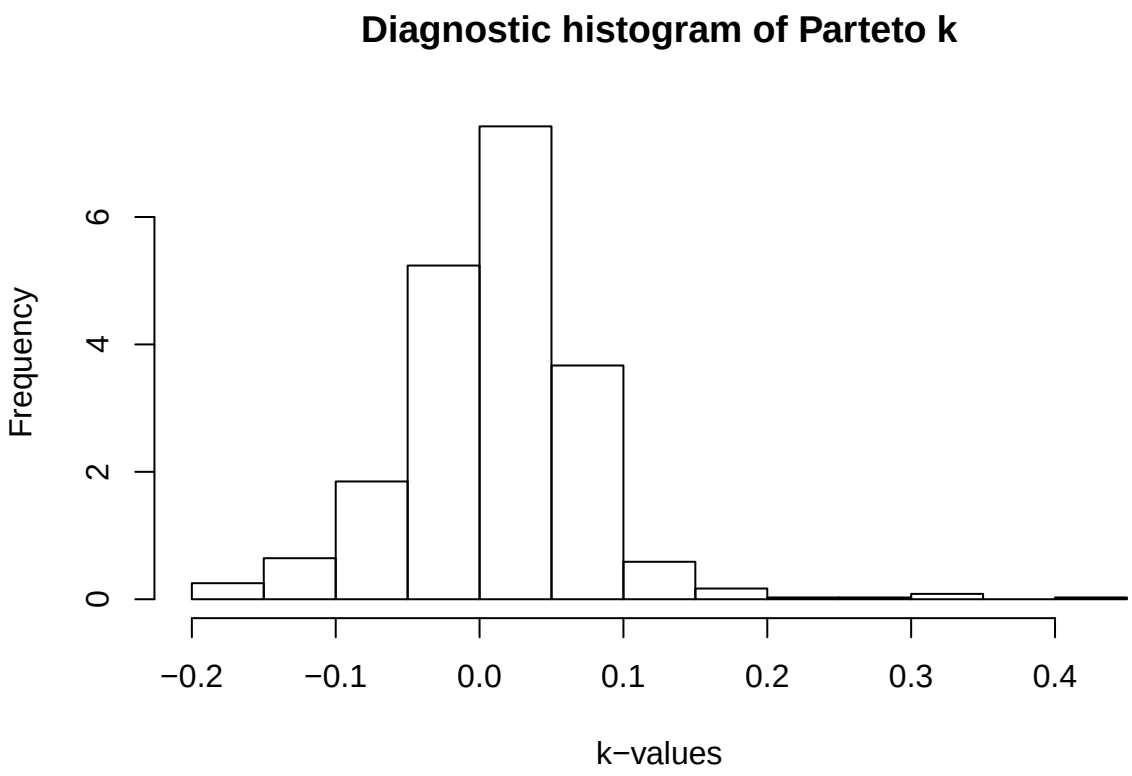
```
     xlab = "k-values",
     ylab = "Frequency",
     freq = FALSE)
```




```
hist(looFull$diagnostics$pareto_k,
     main = "Diagnostic histogram of Parteto k",
     xlab = "k-values",
     ylab = "Frequency",
     freq = FALSE)
```




```
hist(looHier$diagnostics$pareto_k,
     main = "Diagnostic histogram of Parteto k",
     xlab = "k-values",
     ylab = "Frequency",
     freq = FALSE)
```



We can see that the k-values of all model are less than < 0.5 which means that the raw importance ratios have finite variance and that the central limit holds which means that the models did converge well and with low bias.

Now we can compare both models the find out which one has the better performance.

Model analysis and results

Model comparison

We will make use of LOO-CV (leave-one-out cross validation) to compare the models.

```
cat("\nFull Model PSIS_LOO:" ,looFull$estimates[2], "\n");
```

```
##
## Full Model PSIS_LOO: 7.145094
```

```
cat("\nSelection Model PSIS_LOO:" , looSelection$estimates[2], "\n");
```

```
##
## Selection Model PSIS_LOO: 4.268774
```

```
cat("\nHierarchical Model PSIS_LOO:" ,looHier$estimates[2], "\n\n");
```

```
##
## Hierarchical Model PSIS_LOO: 10.08085
```

```
loo_compare(list(looSelection, looFull, looHier))
```

```
##                 elpd_diff se_diff
## TitanicHier        0.0       0.0
```

```
## TitanicFull      -10.9       7.0
## TitanicSelection -14.3       6.3
```

From the PSIS_LOO we can see that the hierarchical model has the highest value compared to the other two models. This value is an indicator of model performance and the highest has better performance. With loo we can estimate the difference in the models expected predictive accuracy and loo_compare will place the model with the largest ELPD (smallest LOOIC) first with zero values because there is no difference between it and the best model (itself). We can see that the hierarchical model performs the best and that there is a big difference between the hierarchical model and the other two non-hierarchical models.

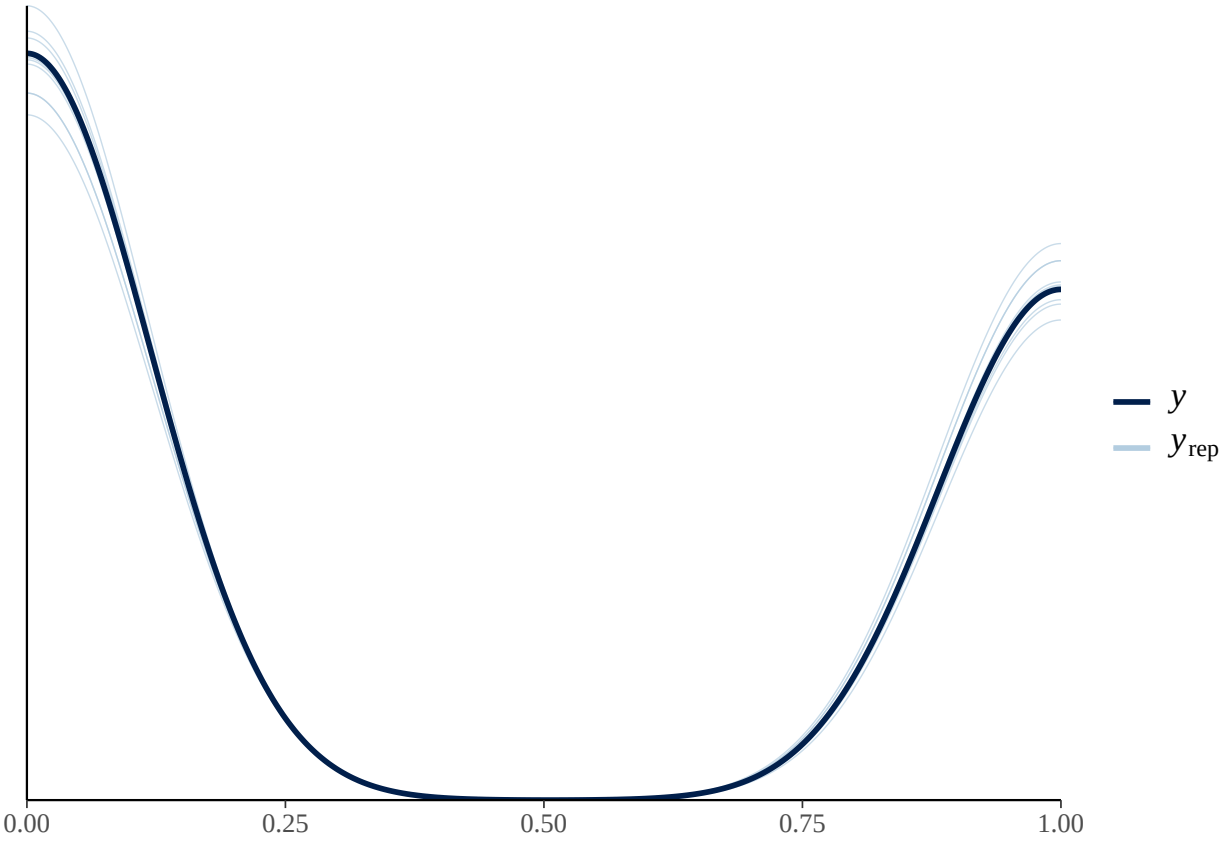
We will from now on go further with the hierarchical model and check the posterior predicitive performance.

Graphical posterior predictive analysis

We will use pp_check() to test the posterior predictive checking which will plot the y values from the posterior distribution compared to the true values of y.

```
pp_check(TitanicHier, newdata = data)
```

```
## Using 10 posterior samples for ppc type 'dens_overlay' by default.
```



From this we can see that the prediction ability of the full model is fairly good, it has better accuracy than the selection model. Some more feature engineering could probably help make the prediction better.

Sensitivity analysis

We believe that our choice of priors are fair but just to make sure we will try another set of priors for both models, We will use the the alternative priors that we mentioned before. The default priors of brms which is Student's t distribution with 3 degrees of freedom, location 0 and scale 10.

```
## Compiling the C++ model

## Start sampling

## Compiling the C++ model

## recompiling to avoid crashing R session

## Start sampling

## Compiling the C++ model

## Start sampling

## Warning: There were 74 divergent transitions after warmup. Increasing adapt_delta above 0.8 may help
## http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup

## Warning: There were 63 transitions after warmup that exceeded the maximum treedepth. Increase max_tr
## http://mc-stan.org/misc/warnings.html#maximum-treedepth-exceeded

## Warning: Examine the pairs() plot to diagnose sampling problems

## Warning: Tail Effective Samples Size (ESS) is too low, indicating posterior variances and tail quanti
## Running the chains for more iterations may help. See
## http://mc-stan.org/misc/warnings.html#tail-ess

##  Family: bernoulli
##   Links: mu = logit
## Formula: Survived ~ Age + Sex + Pclass
##    Data: data (Number of observations: 714)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     5.10      0.50     4.13     6.13 1.00     3219     2508
## Age          -0.04      0.01    -0.05    -0.02 1.00     3723     3212
## Sex          -2.54      0.21    -2.97    -2.14 1.00     3813     3165
## Pclass       -1.30      0.14    -1.58    -1.03 1.00     3402     2973
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).

## 0 of 4000 iterations ended with a divergence.

## 0 of 4000 iterations saturated the maximum tree depth of 10.
```

```
summary(TitanicFullDP)
```

```
##  Family: bernoulli
##   Links: mu = logit
## Formula: Survived ~ Age + Sex + Pclass + SibSp + Parch + Embarked
##    Data: data (Number of observations: 714)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     5.88      0.59     4.78     7.06 1.00     4905     3027
## Age          -0.04      0.01    -0.06    -0.03 1.00     4683     3157
```

```
## Sex          -2.66      0.22    -3.09    -2.23 1.00     6384     3448
## Pclass       -1.30      0.15    -1.59    -1.01 1.00     4505     2961
## SibSp        -0.36      0.13    -0.62    -0.12 1.00     4805     2946
## Parch        -0.05      0.12    -0.29     0.19 1.00     5739     3111
## Embarked     -0.19      0.13    -0.45     0.08 1.00     5921     2786
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
check_divergences(TitanicFullDP$fit)
```

```
## 0 of 4000 iterations ended with a divergence.
```

```
check_treedepth(TitanicFullDP$fit)
```

```
## 0 of 4000 iterations saturated the maximum tree depth of 10.
```

```
summary(TitanicHierDP)
```

```
## Warning: There were 74 divergent transitions after warmup. Increasing adapt_delta above 0.8 may help
## See http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
```

```
##  Family: bernoulli
##   Links: mu = logit
## Formula: Survived ~ Age + Sex + (Age + Sex | Pclass)
##    Data: data (Number of observations: 714)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Group-Level Effects:
## ~Pclass (Number of levels: 3)
##                     Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
## sd(Intercept)           4.85      3.08     1.47    13.20 1.00     1170
## sd(Age)                 0.10      0.13     0.00     0.45 1.00      596
## sd(Sex)                 3.80      2.90     1.04    11.78 1.00     1164
## cor(Intercept,Age)     -0.10      0.47    -0.88     0.79 1.00     1675
## cor(Intercept,Sex)     -0.33      0.46    -0.96     0.63 1.00     1446
## cor(Age,Sex)            0.12      0.50    -0.82     0.92 1.00     1726
##                     Tail_ESS
## sd(Intercept)           1464
## sd(Age)                  507
## sd(Sex)                  987
## cor(Intercept,Age)      1515
## cor(Intercept,Sex)      1863
## cor(Age,Sex)            2187
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     3.38      2.93    -2.75     9.52 1.00     1232     1284
## Age          -0.05      0.07    -0.23     0.07 1.01      477      286
## Sex          -3.27      2.40    -8.30     1.89 1.00      946     1395
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
check_divergences(TitanicHierDP$fit)
```

```
## 74 of 4000 iterations ended with a divergence (1.85%).
## Try increasing 'adapt_delta' to remove the divergences.
```

```
check_treedepth(TitanicHierDP$fit)
```

```
## 63 of 4000 iterations saturated the maximum tree depth of 10 (1.575%).
## Try increasing 'max_treedepth' to avoid saturation.
```

```
loo_compare(looSelection, looFull, loo(TitanicHier), loo(TitanicSelectionDP),loo(TitanicFullDP), loo(Ti
```

```
##                     elpd_diff se_diff
## TitanicHierDP         0.0       0.0
## TitanicHier          -0.2       0.1
## TitanicFullDP       -11.1       7.0
## TitanicFull         -11.1       7.0
## TitanicSelectionDP  -14.4       6.4
## TitanicSelection    -14.5       6.3
```

We get similar results from the summaries as we did for the previous priors so the same diagnostics regarding convergence, tree depth, and ESS apply. From the loo results we can that the posterior is not very sensitive to the prior as the changes are very small, although here is a slightly bigger change in regards to the hierarchical model. We notice that the Student t prior performs slightly better in the case of the full model and hierarchical model.

Conclusion and discussion

We were interested in how the different characteristics of the passenger could affect the survival rate of each passenger and if there was a pattern between the passenger that could be observed. We wanted to explore how two different linear approches to predicting survival on the Titanic boat. There were varibles which had a closer correlation to the survivability while other had less and we looked at this relationship when selecting the varibles for our reduced model. The second full model had most varibles except for three varibles which we discussed and excluded. The third model is a hierarchical model used 3 variables like in the first model, while it consider passenger calss with respect to age and genda in a group. We compared the accuracy of 3 models and we can conclude that the hierarchical model performed the best. With the posterior performance assessment we can conclude that the model fairly good and we would recommended this model as an accurate predictior of survivability for the Titanic.

Future improvement could be looking at more advanced future engineering or creating a hierarchical model by grouping the passengers to the varibles with more high correlation with survivability. We believe that these changes would create models with higher prediction accuracy.

Appendix

Sensitivity analysis

The Stan models used created to test the sensetivity of our priors:

Selection Model with brms steudent t(2,0,10) default priors

```
// generated with brms 2.10.0
functions {
}
data {
  int<lower=1> N;  // number of observations
```

```
  int Y[N];  // response variable
  int<lower=1> K;  // number of population-level effects
  matrix[N, K] X;  // population-level design matrix
  int prior_only;  // should the likelihood be ignored?
}
transformed data {
  int Kc = K - 1;
  matrix[N, Kc] Xc;  // centered version of X without an intercept
  vector[Kc] means_X;  // column means of X before centering
  for (i in 2:K) {
    means_X[i - 1] = mean(X[, i]);
    Xc[, i - 1] = X[, i] - means_X[i - 1];
  }
}
parameters {
  vector[Kc] b;  // population-level effects
  // temporary intercept for centered predictors
  real Intercept;
}
transformed parameters {
}
model {
  // priors including all constants
  target += student_t_lpdf(Intercept | 3, 0, 10);
  // likelihood including all constants
  if (!prior_only) {
    target += bernoulli_logit_glm_lpmf(Y | Xc, Intercept, b);
  }
}
generated quantities {
  // actual population-level intercept
  real b_Intercept = Intercept - dot_product(means_X, b);
}
```

Full model with brms student t(2,0,10) default priors

```
// generated with brms 2.10.0
functions {
}
data {
  int<lower=1> N;  // number of observations
  int Y[N];  // response variable
  int<lower=1> K;  // number of population-level effects
  matrix[N, K] X;  // population-level design matrix
  int prior_only;  // should the likelihood be ignored?
}
transformed data {
  int Kc = K - 1;
  matrix[N, Kc] Xc;  // centered version of X without an intercept
  vector[Kc] means_X;  // column means of X before centering
  for (i in 2:K) {
    means_X[i - 1] = mean(X[, i]);
    Xc[, i - 1] = X[, i] - means_X[i - 1];
```

```
  }
}
parameters {
  vector[Kc] b;  // population-level effects
  // temporary intercept for centered predictors
  real Intercept;
}
transformed parameters {
}
model {
  // priors including all constants
  target += student_t_lpdf(Intercept | 3, 0, 10);
  // likelihood including all constants
  if (!prior_only) {
    target += bernoulli_logit_glm_lpmf(Y | Xc, Intercept, b);
  }
}
generated quantities {
  // actual population-level intercept
  real b_Intercept = Intercept - dot_product(means_X, b);
}
```

Hierarchical model with brms studen t(2,0,10) default priors

```
// generated with brms 2.10.0
functions {
}
data {
  int<lower=1> N;  // number of observations
  int Y[N];  // response variable
  int<lower=1> K;  // number of population-level effects
  matrix[N, K] X;  // population-level design matrix
  // data for group-level effects of ID 1
  int<lower=1> N_1;  // number of grouping levels
  int<lower=1> M_1;  // number of coefficients per level
  int<lower=1> J_1[N];  // grouping indicator per observation
  // group-level predictor values
  vector[N] Z_1_1;
  vector[N] Z_1_2;
  vector[N] Z_1_3;
  int<lower=1> NC_1;  // number of group-level correlations
  int prior_only;  // should the likelihood be ignored?
}
transformed data {
  int Kc = K - 1;
  matrix[N, Kc] Xc;  // centered version of X without an intercept
  vector[Kc] means_X;  // column means of X before centering
  for (i in 2:K) {
    means_X[i - 1] = mean(X[, i]);
    Xc[, i - 1] = X[, i] - means_X[i - 1];
  }
}
parameters {
```

```
  vector[Kc] b;  // population-level effects
  // temporary intercept for centered predictors
  real Intercept;
  vector<lower=0>[M_1] sd_1;  // group-level standard deviations
  matrix[M_1, N_1] z_1;  // standardized group-level effects
  // cholesky factor of correlation matrix
  cholesky_factor_corr[M_1] L_1;
}
transformed parameters {
  // actual group-level effects
  matrix[N_1, M_1] r_1 = (diag_pre_multiply(sd_1, L_1) * z_1)';
  // using vectors speeds up indexing in loops
  vector[N_1] r_1_1 = r_1[, 1];
  vector[N_1] r_1_2 = r_1[, 2];
  vector[N_1] r_1_3 = r_1[, 3];
}
model {
  // initialize linear predictor term
  vector[N] mu = Intercept + Xc * b;
  for (n in 1:N) {
    // add more terms to the linear predictor
    mu[n] += r_1_1[J_1[n]] * Z_1_1[n] + r_1_2[J_1[n]] * Z_1_2[n] + r_1_3[J_1[n]] * Z_1_3[n];
  }
  // priors including all constants
  target += student_t_lpdf(Intercept | 3, 0, 10);
  target += student_t_lpdf(sd_1 | 3, 0, 10)
    - 3 * student_t_lccdf(0 | 3, 0, 10);
  target += normal_lpdf(to_vector(z_1) | 0, 1);
  target += lkj_corr_cholesky_lpdf(L_1 | 1);
  // likelihood including all constants
  if (!prior_only) {
    target += bernoulli_logit_lpmf(Y | mu);
  }
}
generated quantities {
  // actual population-level intercept
  real b_Intercept = Intercept - dot_product(means_X, b);
  // group-level correlations
  corr_matrix[M_1] Cor_1 = multiply_lower_tri_self_transpose(L_1);
  vector<lower=-1,upper=1>[NC_1] cor_1;
  // extract upper diagonal of correlation matrix
  for (k in 1:M_1) {
    for (j in 1:(k - 1)) {
      cor_1[choose(k - 1, 2) + j] = Cor_1[j, k];
    }
  }
}
```